Contents

About SQL Server Analysis Services

Overview

- What's new in SQL Server Analysis Services 2016
- What's new in SQL Server Analysis Services 2017
- Tools and applications
- Client libraries (data providers)
- Comparing Tabular and Multidimensional solutions
- Analysis Services Features and Tasks
- Analysis Services features supported by SQL Server editions
- Backward compatibility (SQL Server 2016 Analysis Services)
- Backward compatibility (SQL Server 2017 Analysis Services)
- Globalization and translation
  - Languages and collations
  - Translations
  - Currency conversions
  - Tips and best practices

Tutorials

- Tabular modeling (1400 compatibility level)
  - 1 - Create a tabular model project
  - 2 - Get data
  - 3 - Mark as Date Table
  - 4 - Create relationships
  - 5 - Create calculated columns
  - 6 - Create measures
  - 7 - Create Key Performance Indicators
  - 8 - Create perspectives
  - 9 - Create hierarchies
  - 10 - Create partitions
  - 11 - Create roles

12 - Analyze in Excel
13 - Deploy
Supplemental lesson - Detail Rows
Supplemental lesson - Dynamic security
Supplemental lesson - Ragged hierarchies
Tabular modeling (1200 compatibility level)
1 - Create a New Tabular Model Project
2 - Add Data
3 - Mark as Date Table
4 - Create relationships
5 - Create calculated columns
6 - Create measures
7 - Create Key Performance Indicators
8 - Create perspectives
9 - Create hierarchies
10 - Create partitions
11 - Create roles
12 - Analyze in Excel
13 - Deploy
Supplemental Lesson - Implement dynamic security by using row filters
Supplemental Lesson - Configure reporting properties for Power View reports
Multidimensional modeling
Tutorial scenario
Install sample data and projects for the Analysis Services Multidimensional modeling tutorial
Lesson 1: Defining a data source view within an Analysis Services project
Lesson 1-1 - Creating an Analysis Services project
Lesson 1-2 - Defining a data source
Lesson 1-3 - Defining a data source view
Lesson 1-4 - Modifying default table names
Lesson 2: Defining and deploying a cube
Lesson 2-1 - Defining a dimension
Lesson 2-2 - Defining a cube

Lesson 2-3 - Adding attributes to dimensions
Lesson 2-4 - Reviewing cube and dimension properties
Lesson 2-5 - Deploying an Analysis Services project
Lesson 2-6 - Browsing the cube
Lesson 3: Modifying measures, attributes and hierarchies
Lesson 3-1 - Modifying measures
Lesson 3-2 - Modifying the customer dimension
Lesson 3-3 - Modifying the product dimension
Lesson 3-4 - Modifying the date dimension
Lesson 3-5 - Browsing the deployed cube
Lesson 4: Defining advanced attribute and dimension properties
Lesson 4-1 - Using a modified version of the Analysis Services tutorial project
Lesson 4-2 - Defining parent attribute properties in a parent-child hierarchy
Lesson 4-3 - Automatically grouping attribute members
Lesson 4-4 - Hiding and disabling attribute hierarchies
Lesson 4-5 - Sorting attribute members based on a secondary attribute
Lesson 4-6 - Specifying attribute relationships between attributes in a user-defined hierarchy
Lesson 4-7 - Defining the unknown member and null processing properties
Lesson 5: Defining relationships between dimensions and measure groups
Lesson 5-1 - Defining a referenced relationship
Lesson 5-2 - Defining a fact relationship
Lesson 5-3 - Defining a many-to-many relationship
Lesson 5-4 - Defining dimension granularity within a measure group
Lesson 6: Defining calculations
Lesson 6-1 - Defining calculated members
Lesson 6-2 - Defining named sets
Lesson 7: Defining Key Performance Indicators (KPIs)
Lesson 7-1 - Defining and browsing KPIs
Lesson 8: Defining actions
Lesson 8-1 - Defining and using a drillthrough action
Lesson 9: Defining perspectives and translations
Lesson 9-1 - Defining and browsing perspectives

Lesson 9-2 - Defining and browsing translations
Lesson 10: Defining administrative roles
Lesson 10 - Granting process database permissions
Data Mining
How-to
Install and configure
Install in tabular or multidimensional mode
Install in Power Pivot for Sharepoint mode
Verify a Power Pivot for SharePoint Installation
Example of a Minimum-Privilege Configuration for Power Pivot for SharePoint 2016
Example of a Minimum-Privilege Configuration for Power Pivot for SharePoint 2013
Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2016)
Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2013)
Configure Power Pivot and Deploy Solutions (SharePoint 2016)
Configure Power Pivot and Deploy Solutions (SharePoint 2013)
Migrate Power Pivot to SharePoint 2013
Upgrade Workbooks and Scheduled Data Refresh (SharePoint 2013)
Configure Analysis Services and Kerberos Constrained Delegation (KCD)
CheckList Use PowerShell to Verify Power Pivot for SharePoint
Deploying SQL Server 2016 PowerPivot and Power View in SharePoint 2016
Deploying SQL Server 2016 PowerPivot and Power View in a Multi-Tier SharePoint 2016 Farm
Install client libraries (AMO, ADOMD.NET, MSOLAP)
Verify cumulative update build version
Post-install configuration
Post install and confiuration overview
Configure the Windows Firewall
Configure the service account
Add administrators
Features off by default
Server groups in SSMS

- Determine the Server Mode
- Rename an Analysis Services Instance
- Tabular modeling
  - Tabular modeling overview
  - Compatibility level
  - Data sources supported in tabular 1400 models
  - Data sources supported in tabular 1200 models
  - Create Tabular Models
    - Tabular model designer
    - Workspace database
    - Tabular Model Projects
      - Create a New Tabular Model Project
      - Import from Analysis Services
      - Import from Power Pivot
    - Properties
      - Power View - Reporting Properties
      - Power View - Configure Default Field Set
      - Power View - Configure Table Behavior Properties
      - Project properties
      - Model properties
      - Table Properties
      - Column Properties
      - Configure default data modeling and deployment properties
    - Import Data
      - Data sources supported in tabular 1400 models
      - Data sources supported in tabular 1200 models
      - Data types supported
      - Impersonation
      - Import data by using a native query
    - DirectQuery Mode
      - Enable DirectQuery mode in SSDT
      - Enable DirectQuery mode in SSMS

Add sample data to DirectQuery models
Define partitions in DirectQuery models
Test a model in DirectQuery mode
DAX formula compatibility in DirectQuery mode (SSAS 2016)
Tables and Columns
Add columns to a table
Delete a Column
Change table, column, or row filter mappings
Specify Mark as Date Table for use with Time Intelligence
Add a table
Delete a Table
Create a Calculated Table
Rename a Table or Column
Set the Data Type of a Column
Hide or freeze columns
Calculated Columns
Create a Calculated Column
Sort Data in a Table
Filter Data in a Table
Relationships
Create a Relationship Between Two Tables
Delete Relationships
Bi-directional cross filters
Calculations
Understanding DAX in Tabular Models
Measures
Create and Manage Measures
KPIs
Create and Manage KPIs
Hierarchies
Create and Manage Hierarchies
Partitions

- Create and Manage Partitions in the Workspace Database
- Process Partitions in the Workspace Database
- Perspectives
  - Create and Manage Perspectives
- Translations
- Roles
  - Create and Manage Roles
  - Object-level security
- Analyze in Excel
  - Analyze a tabular model in Excel
- String storage and collation in tabular models
- Deploy tabular model solutions
  - Deploy from SQL Server Data Tools
- Manage Tabular Model Databases
  - Process Database, Table, or Partition
  - Tabular Model Roles
    - Manage Roles by using SSMS
  - Tabular Model Partitions
    - Create and Manage Tabular Model Partitions
    - Process Tabular Model Partitions
- Connect to a tabular model database
- Restore from Power Pivot
- Multidimensional modeling
  - Multidimensionl modeling overview
  - Multidimensional Solutions
    - Creating Multidimensional Models Using SQL Server Data Tools (SSDT)
      - Create an Analysis Services Project (SSDT)
      - Configure Analysis Services Project Properties (SSDT)
      - Build Analysis Services Projects (SSDT)
      - Deploy Analysis Services Projects (SSDT)
      - Create and Run an MDX Script (SSDT)
      - View the XML for an Analysis Services Project (SSDT)

Connect in Online Mode to an Analysis Services Database
Working with Analysis Services Projects and Databases During the Development Phase
Working with Analysis Services Projects and Databases in a Production Environment
Data Sources
Supported Data Sources
Create a Data Source
Delete a Data Source
Set Data Source Properties
Set Impersonation Options for a Data Source Connection
Data Source Views
Data Sources and Bindings
Define a Data Source View
Add or Remove Tables or Views in a Data Source View
Explore Data in a Data Source View
Work with Diagrams in Data Source View Designer
Refresh the Schema in a Data Source View
Replace a Table or a Named Query in a Data Source View
Define Logical Relationships in a Data Source View
Define Logical Primary Keys in a Data Source View
Define Named Calculations in a Data Source View
Define Named Queries in a Data Source View
Delete a Data Source View
Change Properties in a Data Source View
Schema Generation Wizard
Use the Schema Generation Wizard
Understanding the Database Schemas
Understanding Incremental Generation
Manage Changes to Data Source Views and Data Sources
Dimensions
Create a Dimension Using the Dimension Wizard
Create a Dimension by Using an Existing Table

Create a Time Dimension by Generating a Time Table
Create a Dimension by Generating a Non-Time Table in the Data Source
Database Dimensions
Database Dimensions - Configure the (All) Level for Attribute Hierarchies
Database Dimensions - Modify or Delete in Solution Explorer
Database Dimensions - Browse Data in Designer
Database Dimensions - BI Wizard in Designer
Database Dimensions - Create a Date type
Database Dimensions - Finance Account of parent-child
Database Dimensions - Currency type
Dimension Attribute Properties Reference
  Attribute Properties - Rename an Attribute
  Attribute Properties - Add an Attribute to a Dimension
  Attribute Properties - Remove an Attribute from a Dimension
  Attribute Properties - Configure Attribute Types
  Attribute Properties - Group Attribute Members
  Attribute Properties - Define Custom Member Formulas
  Attribute Properties - Define a Default Member
  Attribute Properties - View Attributes in Dimension Designer
  Attribute Properties - Define a New Attribute Automatically
  Attribute Properties - View Attributes in Dimension Designer
  Attribute Properties - Modify the KeyColumn Property
  Attribute Properties - Bind an Attribute to a Key Column
  Attribute Properties - Bind an Attribute to a Name Column
  Attribute Properties - Set Usage Property
  Attribute Properties - Define Member Groups
Parent-Child Dimension
Parent-Child Dimension Attributes
Parent-Child Dimension Attributes - Custom Rollup Operators
Parent-Child Dimension Attributes - Unary Operators
User-Defined Hierarchies - Create
User-Defined Hierarchies - Ragged Hierarchies

User-Defined Hierarchies - Add or Delete a User-Defined Hierarchy
Attribute Relationships - Define
Attribute Relationships - Configure Attribute Properties
Attribute Relationships - Create, Modify, or Delete Relationship
Attribute Relationships - Define the Relationship Type
Attribute Relationships - Arrange Shapes in the Diagram
BI Wizard - Add Account Intelligence to a Dimension
BI Wizard - Add Dimension Intelligence to a Dimension
BI Wizard - Add a Custom Aggregation to a Dimension
BI Wizard - Custom Member Formulas for Attributes in a Dimension
BI Wizard - Define the Ordering for a Dimension
BI Wizard - Enable Dimension Writeback
Configure String Storage for Dimensions and Partitions
Define Linked Dimensions
Cubes
Create a Cube Using the Cube Wizard
Create a Cube from a template without using a Data Source View
Create a Cube using a Data Source View
Browse data and metadata in Cube
View the Cube Schema
Define Cube Dimension Properties
Define Cube Hierarchy Properties
Define Cube Attribute Properties
Define a Regular Relationship and Regular Relationship Properties
Define a Referenced Relationship and Referenced Relationship Properties
Define a Fact Relationship and Fact Relationship Properties
Define a Many-to-Many Relationship and Many-to-Many Relationship Properties
Measures and Measure Groups
Create Measures and Measure Groups
Configure Measure Group Properties
Configure Measure Properties
Use Aggregate Functions

- Define Semiadditive Behavior
- Defined Linked Measure Groups
- Partitions
  - Create Local Partitions
  - Create Remote Partitions
  - Change a Partition Source
  - Designing Aggregations
  - Edit or Delete Partitions
  - Merge Partitions
  - Set Partition Storage
  - Set Partition Slice
  - Set Partition Writeback
- Calculations
  - Create Calculated Members
  - Create Named Sets
  - Define Assignments and Other Script Commands
  - Define Time Intelligence Calculations using the Business Intelligence Wizard
- KPIs
- Actions overview
  - Actions
  - Add a Standard Action
  - Test an Action
  - Use a Template to Create an Action
- Perspectives
- Translations
- Solution Deployment
  - Requirements and Considerations for Analysis Services Deployment
  - Deploy Model Solutions Using XMLA
  - Deploy Model Solutions Using the Deployment Wizard
    - Running the Analysis Services Deployment Wizard
    - Deployment Script Files - Input to Create Script
    - Deployment Script Files - Specify the Install Target

Deployment Script Files - Partition and Role Deployment Options
Deployment Script Files - Solution Deployment Config Settings
Deployment Script Files - Specify Processing Options
Understanding the Analysis Services Deployment Script
Deploy Model Solutions with the Deployment Utility
Multidimensional Databases
Attach and Detach Analysis Services Databases
Backup and Restore of Analysis Services Databases
Backup Options
Restore Options
Document and Script an Analysis Services Database
Modify or Delete an Analysis Services Database
Move an Analysis Services Database
Rename a Multidimensional Database
Compatibility Level
Set Multidimensional Database Properties
Database Storage Location
Database ReadWriteModes
Synchronize Analysis Services Databases
Switch ReadOnly and ReadWrite modes
Processing a multidimensional model
Processing Options and Settings
Processing Databases, Dimensions and Partitions
Tools and Approaches for Processing
Batch Processing
Remote Processing
Error Configuration
Roles and Permissions
Authorizing access to objects and operations
Database permissions
Cube or model permissions
Process permissions

- Read Definition permissions
- Dimension permissions
- Custom access to dimension data
- Custom access to cell data
- Data Source object permissions
- Data Mining permissions
- Power View for Multidimensional Models
  - Understanding Power View for Multidimensional Models
  - Create a Report Data Source
  - Create a Power View Report
- Assemblies Management
- Data Mining
  - Data Mining overview
  - Data Mining concepts
  - Algorithms
    - Microsoft Association
      - Microsoft Association Algorithm Technical Reference
      - Mining Model Content for Association Models
      - Association Model Query Examples
    - Microsoft Clustering
      - Microsoft Clustering Algorithm Technical Reference
      - Mining Model Content for Clustering Models
      - Clustering Model Query Examples
    - Microsoft Decision Trees
      - Microsoft Decision Trees Algorithm Technical Reference
      - Mining Model Content for Decision Tree Models
      - Decision Trees Model Query Examples
    - Microsoft Linear Regression
      - Microsoft Linear Regression Algorithm Technical Reference
      - Mining Model Content for Linear Regression Models
      - Linear Regression Model Query Examples
    - Microsoft Logistic Regression

- Microsoft Logistic Regression Algorithm Technical Reference
- Mining Model Content for Logistic Regression Models
- Logistic Regression Model Query Examples
- Microsoft Naive Bayes
  - Microsoft Naive Bayes Algorithm Technical Reference
  - Mining Model Content for Naive Bayes Models
  - Naive Bayes Model Query Examples
- Microsoft Neural Network
  - Microsoft Neural Network Algorithm Technical Reference
  - Mining Model Content for Neural Network Models
  - Neural Network Model Query Examples
- Microsoft Sequence Clustering
  - Microsoft Sequence Clustering Algorithm Technical Reference
  - Mining Model Content for Sequence Clustering Models
  - Sequence Clustering Model Query Examples
- Microsoft Time Series
  - Microsoft Time Series Algorithm Technical Reference
  - Mining Model Content for Time Series Models
  - Time Series Model Query Examples
- Plugin Algorithms

Structures
- Mining Structure Columns
  - Data Types (Data Mining)
  - Nested Tables
  - Column Distributions
  - Discretization Methods
  - Classified Columns
- Drillthrough on Mining Structures
- Properties for Mining Structure and Structure Columns
- Mining Structure Tasks and How-tos
  - Create a New Relational Mining Structure
  - Create a New OLAP Mining Structure

- Add Columns to a Mining Structure
- Remove Columns from a Mining Structure
- Add a Nested Table to a Mining Structure
- Change the Properties of a Mining Structure
- Edit the Data Source View used for a Mining Structure
- Process a Mining Structure

Models

- Mining Model Columns
- Content Types (Data Mining)
- Mining Model Properties
- Missing Values
- Feature Selection (Data Mining)
- Modeling Flags (Data Mining)
- Mining Model Content
- Drillthrough on Mining Models
- Filters for Mining Models
  - Model Filter Syntax and Examples
- Mining Model Tasks and How-tos
  - Add a Mining Model to an Existing Mining Structure
  - Delete a Mining Model from a Mining Structure
  - Exclude a Column from a Mining Model
  - Create an Alias for a Model Column
  - Change the Discretization of a Column in a Mining Model
  - View or Change Modeling Flags
  - Specify a Column to Use as Regressor in a Model
  - Change the Properties of a Mining Model
  - Apply a Filter to a Mining Model
  - Delete a Filter from a Mining Model
  - Enable Drillthrough for a Mining Model
  - View or Change Algorithm Parameters
  - Process a Mining Model
  - Make a Copy of a Mining Model

- Create a Data Mining Dimension
- Testing and Validation
  - Training and Testing Data Sets
  - Lift Chart
  - Profit Chart
  - Classification Matrix
  - Scatter Plot
  - Cross-Validation
    - Measures in the Cross-Validation Report
    - Cross-Validation Formulas
  - Testing and Validation Tasks and How-tos
    - Create a Lift Chart, Profit Chart, or Classification Matrix
    - Create a Cross-Validation Report
    - Choose and Map Model Testing Data
    - Apply Filters to Model Testing Data
    - Choose the Column to Use for Testing a Mining Model
    - Choose an Accuracy Chart Type and Set Chart Options
    - Using Nested Table Data as an Input for an Accuracy Chart
- Queries
  - Prediction Queries
  - Content Queries
  - Drillthrough Queries
    - Drill Through to Case Data from a Mining Model
    - Create Drillthrough Queries using DMX
  - Data Definition Queries
  - Data Mining Schema Rowsets
  - Data Mining Query Tools
  - Data Mining Query Tasks and How-tos
    - Create a Prediction Query Using the Prediction Query Builder
    - Create a DMX Query in SQL Server Management Studio
    - Create a Singleton Query in the Data Mining Designer
    - Apply Prediction Functions to a Model

- Choose and Map Input Data for a Prediction Query
- Manually Edit a Prediction Query
- View and Save the Results of a Prediction Query
- Create a Content Query on a Mining Model
- Query the Parameters Used to Create a Mining Model
- Create a Singleton Prediction Query from a Template
- Create a Data Mining Query by Using XMLA
- Change the Time-out Value for Data Mining Queries

Solutions

- Data Mining Projects
  - Import a Data Mining Project using the Analysis Services Import Wizard
- Processing Data Mining Objects
- Related Projects for Data Mining Solutions
- Deployment of Data Mining Solutions
  - Deploy a Data Mining Solution to Previous Versions of SQL Server
  - Export and Import Data Mining Objects

Architecture

- Logical Architecture
- Physical Architecture
- Data Mining Services and Data Sources
- Management of Data Mining Solutions and Objects
  - Moving Data Mining Objects
  - Processing Requirements and Considerations (Data Mining)
  - Using SQL Server Profiler to Monitor Data Mining
- Security Overview (Data Mining)

Tools

- Data Mining Wizard
  - Create a Relational Mining Structure
  - Create an OLAP Mining Structure
  - Add Mining Models to a Structure
  - Customize Mining Models and Structure
- Data Mining Designer

- Data Mining Model Viewers
  - Browse a Model Using the Microsoft Tree Viewer
  - Browse a Model Using the Microsoft Cluster Viewer
  - Browse a Model Using the Microsoft Time Series Viewer
  - Browse a Model Using the Microsoft Naive Bayes Viewer
  - Browse a Model Using the Microsoft Sequence Cluster Viewer
  - Browse a Model Using the Microsoft Association Rules Viewer
  - Browse a Model Using the Microsoft Neural Network Viewer
  - Browse a Model Using the Microsoft Generic Content Tree Viewer
  - Mining Model Viewer Tasks and How-tos
    - Select a Mining Model and a Data Mining Viewer
    - Copy a View of a Mining Model
    - Find a Specific Node in a Dependency Network
    - Filter a Rule in an Association Rules Model
    - Filter an Itemset in an Association Rules Model
    - Use Drillthrough from the Model Viewers
    - View the Formula for a Time Series Model
    - Change the Colors Used in the Data Mining Viewer
  - Data Mining Add-Ins for Office
- Stored Procedures
  - SystemGetCrossValidationResults
  - SystemGetClusterCrossValidationResults
  - SystemGetAccuracyResults
  - SystemGetClusterAccuracyResults
- Data Mining Extensions (DMX)
- Power Pivot for Sharepoint
- Power Pivot for SharePoint overview
- Power Pivot Server Administration and Configuration in Central Administration
  - Deploy Power Pivot Solutions to SharePoint
  - Create and Configure a Power Pivot Service Application
  - Connect a Power Pivot Service Application to a SharePoint Web Application
  - Activate Power Pivot Feature Integration for Site Collections

Add MSOLAP.5 as a Trusted Data Provider in Excel Services
Create a Trusted Location for Power Pivot Sites
Configure Power Pivot Service Accounts
Configure and View SharePoint Log Files
Configure Disk Space Usage
Configure Maximum File Upload Size
Configuration Setting Reference
Start or Stop a Power Pivot for SharePoint Server
Power Pivot Configuration using Windows PowerShell
Power Pivot Configuration Tools
Configure or Repair Power Pivot for SharePoint 2013 (Power Pivot Configuration Tool)
Power Pivot Availability and Disaster Recovery
Power Pivot Authentication and Authorization
Configure Power Pivot Health Rules
Health Rules Reference
Power Pivot Management Dashboard and Usage Data
Power Pivot Usage Data Collection
Configure Usage Data Collection for (Power Pivot for SharePoint
Create and Customize Power Pivot Gallery
Use Power Pivot Gallery
Delete Power Pivot Gallery
Power Pivot Data Access
Power Pivot Data Refresh
View Power Pivot Data Refresh History
Power Pivot Data Refresh with SharePoint 2013
Schedule Data Refresh and Data Sources That Do Not Support Windows Authentication (Power Pivot for SharePoint)
Power Pivot Data Feeds
Create or Customize a Data Feed Library
Use Power Pivot Data Feeds
Delete a Power Pivot Data Feed Library
Share Power Pivot Data Feeds in a Data Feed Library

Power Pivot BI Semantic Model Connection (.bism)
Add a BI Semantic Model Connection Content Type to a Library
Create a BI Semantic Model Connection to a Power Pivot Workbook
Create a BI Semantic Model Connection to a Tabular Model Database
Use a BI Semantic Model Connection in Excel or Reporting Services
Errors and Events Reference (Power Pivot for SharePoint)
An error occurred during an attempt to establish a connection to the external data source. The following connections failed to refresh: Power Pivot Data
Could not load file or assembly Microsoft.Data.Services
Could not load file or assembly 'Microsoft.AnalysisServices.SharePoint.Integration'
The data connection path in the workbook points to a file on the local drive or is an invalid URI
The data connection uses Windows Authentication and user credentials could not be delegated. The following connections failed to refresh: Power Pivot Data
The following features are not supported by Excel Services and may not display or may display only partially: Comments, Shapes, or other objects
The trusted location where the workbook is stored does not allow external data connections
Workbook contains one or more queries that refresh external data
Unable to refresh data for a data connection in the workbook
Manage servers
SQL Server Analysis Services server management overview
Server properties
Server properties overview
Data Mining Properties
DAX Properties
Feature Properties
Filestore Properties
General Properties
Lock Manager Properties
Log Properties
Memory Properties
Network Properties
OLAP Properties

- Security Properties
- Thread Pool Properties
- Monitor Analysis Services
  - Monitoring tools
  - SQL Server Profiler
    - Monitoring Analysis Services with SQL Server Profiler
    - Create Profiler Traces for Replay
  - Extended Events
  - Dynamic Management Views (DMVs)
  - Performance counters
  - Trace-events
  - Log operations
- Clear server caches
- Database Consistency Checker (DBCC)
- Connect
  - Connect to servers overview
  - Connect from client applications
  - Connection String Properties
  - Authentication methodologies
  - Kerberos authentication
  - SPN registration
  - HTTP Access
  - Client libraries (data providers)
  - Disconnect users and sessions
- Developer
  - Analysis Services Developer Documentation
  - Tabular models
    - Tabular 1200 and higher
      - Tabular model programming for compatibility level 1200 and higher
      - Tabular Model Scripting Language (TMSL)
      - Tabular Object Model (TOM)
      - Analysis Management Objects (AMO)

- Tabular 1103 and lower
  - Tabular model programming for compatibility level 1103 and lower
  - Conceptual Schema Definition Language (CSDL)
  - Analysis Services Scripting Language (ASSL)
  - IMDEmbeddedData Interface
  - Representation
    - Connection Representation (Tabular)
    - Database Representation(Tabular)
    - Perspective Representation (Tabular)
    - Querying a Tabular Model
    - Relationship Representation (Tabular)
    - Tables - Calculated Column Representation
    - Tables - Calculated Measure Representation
    - Tables - Hierarchy Representation
    - Tables - Key Performance Indicator Representation
    - Tables - Partition Representation
    - Tables Representation (Tabular)
    - Understanding Tabular Object Model at Levels 1050 through 1103
- Multidimensional models
  - Multidimensional model programming
  - Analysis Management Objects (AMO)
  - Analysis Services Scripting Language (ASSL)
  - XML for Analysis (XMLA)
  - ADOMD.NET
  - Analysis Services Scripting Language (ASSL for XMLA)
  - OLAP logical
  - OLAP physical
  - Extending OLAP
- Data Mining
  - Data Mining programming
  - Analysis Management Objects (AMO)
  - ADOMD.NET

- Data Mining Extensions (DMX)
- XML for Analysis (XMLA)
- OLE DB for Data Minining

Samples

- Analysis Services samples

Reference

- PowerShell for Analysis Services
- PowerShell for Power Pivot for SharePoint
- Data Analysis Expressions (DAX)
- Power Query M
- Data Mining Extensions (DMX)
- Multidimensional Data Expressions (MDX)
- Tabular Model Scripting Language (TMSL)
- Tabular Object Model (TOM)
- Analysis Management Objects (AMO)
- Conceptual Schema Definition Language (CSDL)
- XML for Analysis (XMLA)
- ADOMD.NET
- Analysis Services Scripting Language (ASSL for XMLA)
- Schema rowsets
- Trace events

About SQL Server Analysis Services

12/18/2018 • 2 minutes to read • Edit Online

Analysis Services is an analytical data engine used in decision support and business analytics. It provides enterprise-grade semantic data models for business reports and client applications such as Power BI, Excel, Reporting Services reports, and other data visualization tools.

A typical workflow includes creating a tabular or multidimensional data model project in Visual Studio, deploying the model as a database to a server instance, setting up recurring data processing, and assigning permissions to allow data access by end-users. When it's ready to go, your semantic data model can be accessed by client applications supporting Analysis Services as a data source.

Analysis Services is available in two different platforms:

Azure Analysis Services - Supports tabular models at the 1200 and higher compatibility levels. DirectQuery, partitions, row-level security, bi-directional relationships, and translations are all supported. To learn more, see Azure Analysis Services.

SQL Server Analysis Services - Supports tabular models at all compatibility levels, multidimensional models, data mining, and Power Pivot for SharePoint.

Documentation by area

In general, Azure Analysis Services documentation is included with Azure documentation. If you're interested in having your tabular models in the cloud, it's best to start there. This article and documentation in this section is mostly for SQL Server Analysis Services. However, at least for tabular models, how you create and deploy your tabular model projects is much the same, regardless of the platform you're using. Check out these sections to learn more:

- Comparing Tabular and Multidimensional Solutions
- Install SQL Server Analysis Services
- Tabular models
- Multidimensional models
- Data Mining
- Power Pivot for SharePoint
- Tutorials
- Server management
- Developer documentation
- Technical reference

See also

Azure Analysis Services documentation SQL Server Documentation

What's New in Analysis Services

12/10/2018 • 13 minutes to read • Edit Online

APPLIES TO: SQL Server 2016 Analysis Services Azure Analysis Services

SQL Server 2016 Analysis Services includes many new enhancements providing improved performance, easier solution authoring, automated database management, enhanced relationships with bi-directional cross filtering, parallel partition processing, and much more. At the heart of most enhancements for this release is the new 1200 compatibility level for tabular model databases.

Azure Analysis Services

Announced at the 2016 SQL PASS Conference, Analysis Services is now available in the cloud as an Azure service. **Azure Analysis Services** supports tabular models at the 1200 and higher compatibility levels. DirectQuery, partitions, row-level security, bi-directional relationships, and translations are all supported. To learn more and give it a try for free, see Azure Analysis Services.

What's new in SQL Server 2016 Service Pack 1 (SP1) Analysis Services

Download SQL Server 2016 SP1

SQL Server 2016 Service SP1 Analysis Services provides improved performance and scalability through Non-Uniform Memory Access (NUMA) awareness and optimized memory allocation based on **Intel Threading Building Blocks** (Intel TBB). This new functionality helps lower Total Cost of Ownership (TCO) by supporting more users on fewer, more powerful enterprise servers.

In particular, SQL Server 2016 SP1 Analysis Services features improvements in these key areas:

- **NUMA awareness** - For better NUMA support, the in-memory (VertiPaq) engine inside Analysis Services now maintains a separate job queue on each NUMA node. This guarantees the segment scan jobs run on the same node where the memory is allocated for the segment data. Note, NUMA awareness is only enabled by default on systems with at least four NUMA nodes. On two-node systems, the costs of accessing remote allocated memory generally doesn't warrant the overhead of managing NUMA specifics.
- **Memory allocation** - Analysis Services has been accelerated with Intel Threading Building Blocks, a scalable allocator that provides separate memory pools for every core. As the number of cores increases, the system can scale almost linearly.
- **Heap fragmentation** - The Intel TBB-based scalable allocator also helps to mitigate performance problems due to heap fragmentation that have been shown to occur with the Windows Heap.

Performance and scalability testing showed significant gains in query throughput when running SQL Server 2016 SP1 Analysis Services on large multi-node enterprise servers.

What's new in SQL Server 2016 Analysis Services

While most enhancements in this release are specific to tabular models, a number of enhancements have been made to multidimensional models; for example, distinct count ROLAP optimization for data sources like DB2 and Oracle, drill-through multi-selection support with Excel 2016, and Excel query optimizations.

Get the latest tools

In order to take full advantage of all the enhancements in this release, be sure to install the latest versions of SSDT and SSMS.

- Download SQL Server Data Tools (SSDT)
- Download SQL Server Management Studio (SSMS)

If you have a custom AMO-dependent application, you might need to install an updated version of AMO. For instructions, see Install Analysis Services data providers (AMO, ADOMD.NET, MSOLAP).

Modeling

Improved modeling performance for tabular 1200 models

For tabular 1200 models, metadata operations in SSDT are much faster than tabular 1100 or 1103 models. By comparison, on the same hardware, creating a relationship on a model set to the SQL Server 2014 compatibility level (1103) with 23 tables takes 3 seconds, whereas the same relationship on a model created set to compatibility level 1200 takes just under a second.

Project templates added for tabular 1200 models in SSDT

With this release, you no longer need two versions of SSDT for building relational and BI projects. SQL Server Data Tools for Visual Studio 2015 adds project templates for Analysis Services solutions, including **Analysis Services Tabular Projects** used for building models at the 1200 compatibility level. Other Analysis Services project templates for multidimensional and data mining solutions are also included, but at the same functional level (1100 or 1103) as in previous releases.

Display folders

Display folders are now available for tabular 1200 models. Defined in SQL Server Data Tools and rendered in client applications like Excel or Power BI Desktop, display folders help you organize large numbers of measures into individual folders, adding a visual hierarchy for easier navigation in field lists.

Bi-directional cross filtering

New in this release is a built-in approach for enabling bi-directional cross filters in tabular models, eliminating the need for hand-crafted DAX workarounds for propagating filter context across table relationships. Filters are only auto-generated when the direction can be established with a high degree of certainty. If there is ambiguity in the form of multiple query paths across table relationships, a filter won't be created automatically. See Bi-directional cross filters for tabular models in SQL Server 2016 Analysis Services for details.

Translations

You can now store translated metadata in a tabular 1200 model. Metadata in the model includes fields for **Culture**, translated captions, and translated descriptions. To add translations, use the **Model** > **Translations** command in SQL Server Data Tools. See Translations in tabular models (Analysis Services) for details.

Pasted tables

You can now upgrade an 1100 or 1103 tabular model to 1200 when the model contains pasted tables. We recommend using SQL Server Data Tools. In SSDT, set **CompatibilityLevel** to 1200 and then deploy to a SQL Server 2017 instance of Analysis Services. See Compatibility Level for Tabular models in Analysis Services for details.

Calculated tables in SSDT

A *calculated table* is a model-only construction based on a DAX expression or query in SSDT. When deployed in a database, a calculated table is indistinguishable from regular tables.

There are several uses for calculated tables, including the creation of new tables to expose an existing table in a specific role. The classic example is a Date table that operates in multiple contexts (order date, ship date, and so forth). By creating a calculated table for a given role, you can now activate a table relationship to facilitate queries or data interaction using the calculated table. Another use for calculated tables is to combine parts of existing tables into an entirely new table that exists only in the model. See Create a Calculated Table to learn more.

Formula fixup

With formula fixup on a tabular 1200 model, SSDT will automatically update any measures that is referencing a column or table that was renamed.

Support for Visual Studio configuration manager

To support multiple environments, like Test and Pre-production environments, Visual Studio allows developers to create multiple project configurations using the configuration manager. Multidimensional models already leverage this but tabular models did not. With this release, you can now use configuration manager to deploy to different servers.

Instance management

Administer Tabular 1200 models in SSMS

In this release, an Analysis Services instance in Tabular server mode can run tabular models at any compatibility level (1100, 1103, 1200). The latest SQL Server Management Studio is updated to display properties and provide database model administration for tabular models at the 1200 compatibility level.

Parallel processing for multiple table partitions in tabular models

This release includes new parallel processing functionality for tables with two or more partitions, increasing processing performance. There are no configuration settings for this feature. For more information about configuring partitions and processing tables, see Tabular Model Partitions.

Add computer accounts as Administrators in SSMS

Analysis Services administrators can now use SQL Server Management Studio to configure computer accounts to be members of the Analysis Services administrators group. In the **Select Users or Groups** dialog, set the **Locations** for the computers domain and then add the **Computers** object type. For more information, see Grant server admin rights to an Analysis Services instance.

DBCC for Analysis Services

Database Consistency Checker (DBCC) runs internally to detect potential data corruption issues on database load, but can also be run on demand if you suspect problems in your data or model. DBCC runs different checks depending on whether the model is tabular or multidimensional. See Database Consistency Checker (DBCC) for Analysis Services tabular and multidimensional databases for details.

Extended Events updates

This release adds a graphical user interface to SQL Server Management Studio to configure and manage Analysis Services Extended Events. You can set up live data streams to monitor server activity in real time, keep session data loaded in memory for faster analysis, or save data streams to a file for offline analysis. For more information, see Monitor Analysis Services with SQL Server Extended Events and Using extended events with Analysis Services (Guy in a Cube blog post and video).

Scripting

PowerShell for Tabular models

This release includes PowerShell enhancements for tabular models at compatibility level 1200. You can use all of the applicable cmdlets, plus cmdlets specific to Tabular mode: Invoke-ProcessASDatabase and Invoke-ProcessTable cmdlet.

SSMS scripting database operations

In the latest SQL Server Management Studio (SSMS), script is now enabled for database commands, including Create, Alter, Delete, Backup, Restore, Attach, Detach. Output is Tabular Model Scripting Language (TMSL) in JSON. See Tabular Model Scripting Language (TMSL) Reference for more information.

Analysis Services Execute DDL Task

Analysis Services Execute DDL Task now also accepts Tabular Model Scripting Language (TMSL) commands.

SSAS PowerShell cmdlet

SSAS PowerShell cmdlet **Invoke-ASCmd** now accepts Tabular Model Scripting Language (TMSL) commands. Other SSAS PowerShell cmdlets may be updated in a future release to use the new tabular metadata (exceptions will be called out in the release notes).
See Analysis Services PowerShell Reference for details.

Tabular Model Scripting Language (TMSL) supported in SSMS

Using the latest version of SSMS, you can now create scripts to automate most administrative tasks for tabular 1200 models. Currently, the following tasks can be scripted: Process at any level, plus CREATE, ALTER, DELETE at the database level.

Functionally, TMSL is equivalent to the XMLA ASSL extension that provides multidimensional object definitions, except that TMSL uses native descriptors like **model**, **table**, and **relationship** to describe tabular metadata. See Tabular Model Scripting Language (TMSL) Reference for details about the schema.

A generated JSON-based script for a tabular model might look like the following:

```
{
  "create": {
    "database": {
      "name": "AdventureWorksTabular1200",
      "id": "AdventureWorksTabular1200",
      "compatibilityLevel": 1200,
      "readWriteMode": "readWrite",
      "model": {}
    }
  }
}
```

The payload is a JSON document that can be as minimal as the example shown above, or highly embellished with the full set of object definitions. Tabular Model Scripting Language (TMSL) Reference describes the syntax.

At the database level, CREATE, ALTER, and DELETE commands will output TMSL script in the familiar XMLA window. Other commands, such as Process, can also be scripted in this release. Script support for many other actions may be added in a future release.

SCRIPTABLE COMMANDS	DESCRIPTION
create	Adds a database, connection, or partition. The ASSL equivalent is CREATE.
createOrReplace	Updates an existing object definition (database, connection, or partition) by overwriting a previous version. The ASSL equivalent is ALTER with AllowOverwrite set to true and ObjectDefinition to ExpandFull.
delete	Removes an object definition. ASSL equivalent is DELETE.
refresh	Processes the object. ASSL equivalent is PROCESS.

DAX

Improved DAX formula editing

Updates to the formula bar help you write formulas with more ease by differentiating functions, fields and

measures using syntax coloring, it provides intelligent function and field suggestions and tells you if parts of your DAX expression are wrong using error *squiggles*. It also allows you to use multiple lines (Alt + Enter) and indentation (Tab). The formula bar now also allows you to write comments as part of your measures, just type "//" and everything after these characters on the same line will be considered a comment.

DAX variables

This release now includes support for variables in DAX. Variables can now store the result of an expression as a named variable, which can then be passed as an argument to other measure expressions. Once resultant values have been calculated for a variable expression, those values do not change, even if the variable is referenced in another expression. For more information, see VAR Function.

New DAX functions

With this release, DAX introduces over fifty new functions to support faster calculations and enhanced visualizations in Power BI. To learn more, see New DAX Functions.

Save incomplete measures

You can now save incomplete DAX measures directly in a tabular 1200 model project and pick it up again when you are ready to continue.

Additional DAX enhancements

- Non empty calculation - Reduces the number of scans needed for non empty.
- Measure Fusion - Multiple measures from the same table will be combined into a single storage engine - query.
- Grouping sets - When a query asks for measures at multiple granularities (Total/Year/Month), a single - query is sent at the lowest level and the rest of the granularities are derived from the lowest level.
- Redundant join elimination - A single query to the storage engine returns both the dimension columns and the measure values.
- Strict evaluation of IF/SWITCH - A branch whose condition is false will no longer result in storage engine queries. Previously, branches were eagerly evaluated but results discarded later on.

Developer

Microsoft.AnalysisServices.Tabular namespace for Tabular 1200 programmability in AMO

Analysis Services Management Objects (AMO) is updated to include a new tabular namespace for managing a Tabular Mode instance of SQL Server 2016 Analysis Services, as well as provide the data definition language for creating or modifying tabular 1200 models programmatically. Visit Microsoft.AnalysisServices.Tabular to read up on the API.

Analysis Services Management Objects (AMO) updates

Analysis Services Management Objects (AMO) has been re-factored to include a second assembly, Microsoft.AnalysisServices.Core.dll. The new assembly separates out common classes like Server, Database, and Role that have broad application in Analysis Services, irrespective of server mode.

Previously, these classes were part of the original Microsoft.AnalysisServices assembly. Moving them to a new assembly paves the way for future extensions to AMO, with clear division between generic and context-specific APIs.

Existing applications are unaffected by the new assemblies. However, should you choose to rebuild applications using the new AMO assembly for any reason, be sure to add a reference to Microsoft.AnalysisServices.Core.

Similarly, PowerShell scripts that load and call into AMO must now load Microsoft.AnalysisServices.Core.dll. Be sure to update any scripts.

JSON editor for BIM files

Code View in Visual Studio 2015 now renders the BIM file in JSON format for tabular 1200 models. The version

of Visual Studio determines whether the BIM file is rendered in JSON via the built-in JSON Editor, or as simple text.

To use the JSON editor, with the ability to expand and collapse sections of the model, you will need the latest version of SQL Server Data Tools plus Visual Studio 2015 (any edition, including the free Community edition). For all other versions of SSDT or Visual Studio, the BIM file is rendered in JSON as simple text. At a minimum, an empty model will contain the following JSON:

```
{
  "name": "SemanticModel",
  "id": "SemanticModel",
  "compatibilityLevel": 1200,
  "readWriteMode": "readWrite",
  "model": {}
}
```

WARNING

Avoid editing the JSON directly. Doing so can corrupt the model.

New elements in MS-CSDLBI 2.0 schema

The following elements have been added to the **TProperty** complex type defined in the [MS-CSDLBI] 2.0 schema:

ELEMENT	DEFINITION
DefaultValue	A property that specifies the value used when evaluating the query. The DefaultValue property is optional, but it is automatically selected if the values from the member cannot be aggregated.
Statistics	A set of statistics from the underlying data that is associated with the column. These statistics are defined by the TPropertyStatistics complex type and are provided only if they are not computationally expensive to generate, as described in section 2.1.13.5 of the Conceptual Schema Definition File Format with Business Intelligence Annotations document.

DirectQuery

New DirectQuery implementation

This release sees significant enhancements in DirectQuery for tabular 1200 models. Here's a summary:

- DirectQuery now generates simpler queries that provide better performance.
- Extra control over defining sample datasets used for model design and testing.
- Row level security (RLS) is now supported for tabular 1200 models in DirectQuery mode. Previously, the presence of RLS prevented deploying a tabular model in DirectQuery mode.
- Calculated columns are now supported for tabular 1200 models in DirectQuery mode. Previously, the presence of calculated columns prevented deploying a tabular model in DirectQuery mode.
- Performance optimizations include redundant join elimination for VertiPaq and DirectQuery.

New data sources for DirectQuery mode

Data sources supported for tabular 1200 models in DirectQuery mode now include Oracle, Teradata and Microsoft Analytics Platform (formerly known as Parallel Data Warehouse).

To learn more, see DirectQuery Mode.

See Also

Analysis Services team blog
What's New in SQL Server 2016

What's new in SQL Server 2017 Analysis Services

11/28/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server 2017 Analysis Services Azure Analysis Services

SQL Server 2017 Analysis Services see some of the most important enhancements since SQL Server 2012. Building on the success of Tabular mode (first introduced in SQL Server 2012 Analysis Services), this release makes tabular models more powerful than ever.

Multidimensional mode and Power Pivot for SharePoint mode are a staple for many Analysis Services deployments. In the Analysis Services product lifecycle, these modes are mature. There are no new features for either of these modes in this release. However, bug fixes and performance improvements are included.

The features described here are included in SQL Server 2017 Analysis Services. But in order to take advantage of them, you must also use the latest versions of SQL Server Data Tools (SSDT) and SQL Server Management Studio (SSMS). SSDT and SSMS are updated monthly with new and improved features that typically coincide with new functionality in SQL Server.

While it's important to learn about all the new features, it's also important to know what is being deprecated and discontinued in this release and future releases. Be sure to check out Backward compatibility (SQL Server 2017 Analysis Services).

Let's take a look at some of the key new features in this release.

1400 Compatibility level for tabular models

To take advantage of many of the new features and functionality described here, new or existing tabular models must be set or upgraded to the 1400 compatibility level. Models at the 1400 compatibility level cannot be deployed to SQL Server 2016 SP1 or earlier, or downgraded to lower compatibility levels. To learn more, see Compatibility level for Analysis Services tabular models.

In SSDT, you can select the new 1400 compatibility level when creating new tabular model projects.

To upgrade an existing tabular model in SSDT, in Solution Explorer, right-click **Model.bim**, and then in **Properties**, set the **Compatibility Level** property to **SQL Server 2017 (1400)**.

It's important to keep in mind, once you upgrade an existing model to 1400, you can't downgrade. Be sure to keep a backup of your 1200 model database.

Modern Get Data experience

When it comes to importing data from data sources into your tabular models, SQL Server Data Tools (SSDT) introduces the modern **Get Data** experience for models at the 1400 compatibility level. This new feature is based on similar functionality in Power BI Desktop and Microsoft Excel 2016. The modern Get Data experience provides immense data transformation and data mashup capabilities by using the Get Data query builder and M expressions.

The modern Get Data experience provides support for a wide range of data sources. Going forward, updates will include support for even more.

A powerful and intuitive user interface makes selecting your data and data transformation/mashup capabilities easier than ever.

The modern Get Data experience and M mashup capabilities do not apply to existing tabular models upraded from

the 1200 compatibility level to 1400. The new experience only applies to new models created at the 1400 compatibility level.

Encoding hints

This release introduces encoding hints, an advanced feature used to optimize processing (data refresh) of large in-memory tabular models. To better understand encoding, see Performance Tuning of Tabular Models in SQL Server 2012 Analysis Services whitepaper to better understand encoding.

- Value encoding provides better query performance for columns that are typically only used for aggregations.
- Hash encoding is preferred for group-by columns (often dimension-table values) and foreign keys. String columns are always hash encoded.

Numeric columns can use either of these encoding methods. When Analysis Services starts processing a table, if either the table is empty (with or without partitions) or a full-table processing operation is being performed, samples values are taken for each numeric column to determine whether to apply value or hash encoding. By default, value encoding is chosen when the sample of distinct values in the column is large enough - otherwise hash encoding usually provides better compression. It is possible for Analysis Services to change the encoding method after the column is partially processed based on further information about the data distribution, and restart the encoding process; however, this increases processing time and is inefficient. The performance-tuning whitepaper discusses re-encoding in more detail and describes how to detect it using SQL Server Profiler.

Encoding hints allow the modeler to specify a preference for the encoding method given prior knowledge from data profiling and/or in response to re-encoding trace events. Since aggregation over hash-encoded columns is slower than over value-encoded columns, value encoding may be specified as a hint for such columns. It is not guaranteed that the preference is applied. It is a hint as opposed to a setting. To specify an encoding hint, set the EncodingHint property on the column. Possible values are "Default", "Value" and "Hash". The following snippet of JSON-based metadata from the Model.bim file specifies value encoding for the Sales Amount column.

```
{
    "name": "Sales Amount",
    "dataType": "decimal",
    "sourceColumn": "SalesAmount",
    "formatString": "\\$#,0.00;(\\$#,0.00);\\$#,0.00",
    "sourceProviderType": "Currency",
    "encodingHint": "Value"
}
```

Ragged hierarchies

In tabular models, you can model parent-child hierarchies. Hierarchies with a differing number of levels are often referred to as ragged hierarchies. By default, ragged hierarchies are displayed with blanks for levels below the lowest child. Here's an example of a ragged hierarchy in an organizational chart:

	A	B
1	Row Labels	Reseller Total Sales
2	Ken Sánchez	$80,450,596.98
3	Brian Welcker	$80,450,596.98
4	Amy Alberts	$15,535,946.26
5		$732,078.44
6		$732,078.44
7		$732,078.44
8	Jae Pak	$8,503,338.65
9	Rachel Valdez	$1,790,640.23
10	Ranjit Varkey Chudukatil	$4,509,888.93
11	Stephen Jiang	$63,320,315.35
12		$1,092,123.86
13	David Campbell	$3,729,945.35
14		$3,729,945.35

This release introduces the **Hide Members** property. You can set the **Hide Members** property for a hierarchy to **Hide blank members**.

> **NOTE**
>
> Blank members in the model are represented by a DAX blank value, not an empty string.

When set to **Hide blank members**, and the model deployed, an easier to read version of the hierarchy is shown in reporting clients like Excel.

	A	B
1	Row Labels	Reseller Total Units
2	Ken Sánchez	209,274
3	Brian Welcker	209,274
4	Amy Alberts	47,214
5	Jae Pak	26,231
6	Rachel Valdez	6,898
7	Ranjit Varkey Chudukatil	14,085
8	Stephen Jiang	157,112
9	David Campbell	8,172
10	Garrett Vargas	11,544

Detail Rows

You can now define a custom row set contributing to a measure value. Detail Rows is similar to the default drillthrough action in multidimensional models. This allows end-users to view information in more detail than the aggregated level.

The following PivotTable shows Internet Total Sales by year from the Adventure Works sample tabular model. You

can right-click a cell with an aggregated value from the measure and then click **Show Details** to view the detail rows.

By default, the associated data in the Internet Sales table is displayed. This limited behavior is often not meaningful for the user because the table may not have the necessary columns to show useful information such as customer name and order information. With Detail Rows, you can specify a **Detail Rows Expression** property for measures.

Detail Rows Expression property for measures

The **Detail Rows Expression** property for measures allows model authors to customize the columns and rows returned to the end-user.

The SELECTCOLUMNS DAX function is commonly used in a Detail Rows Expression. The following example defines the columns to be returned for rows in the Internet Sales table in the sample Adventure Works tabular model:

```
SELECTCOLUMNS(
    'Internet Sales',
    "Customer First Name", RELATED( Customer[Last Name]),
    "Customer Last Name", RELATED( Customer[First Name]),
    "Order Date", 'Internet Sales'[Order Date],
    "Internet Total Sales", [Internet Total Sales]
)
```

With the property defined and the model deployed, a custom row set is returned when the user selects **Show Details**. It automatically honors the filter context of the cell that was selected. In this example, only the rows for 2010 value are displayed:

	A	B	C	D
1	Data returned for Internet Total Sales, 2010 (First 1000 rows).			
2				
3	[Customer First Name]	[Customer Last Name]	[Order Date]	[Internet Total Sales]
4	Colin	Anand	12/30/2010	3578.27
5	Edward	Brown	12/31/2010	3578.27
6	Emma	Brown	12/31/2010	3578.27
7	Ruben	Prasad	12/29/2010	699.0982
8	Sydney	Wright	12/29/2010	3399.99

Default Detail Rows Expression property for tables

In addition to measures, tables also have a property to define a detail rows expression. The **Default Detail Rows Expression** property acts as the default for all measures within the table. Measures that do not have their own expression defined inherits the expression from the table and show the row set defined for the table. This allows reuse of expressions, and new measures added to the table later automatically inherits the expression.

DETAILROWS DAX Function

Included in this release is a new `DETAILROWS` DAX function that returns the row set defined by the detail rows expression. It works similarly to the `DRILLTHROUGH` statement in MDX, which is also compatible with detail rows expressions defined in tabular models.

The following DAX query returns the row set defined by the detail rows expression for the measure or its table. If no expression is defined, the data for the Internet Sales table is returned because it's the table containing the measure.

```
EVALUATE DETAILROWS([Internet Total Sales])
```

Object-level security

This release introduces object-level security for tables and columns. In addition to restricting access to table and column data, sensitive table and column names can be secured. This helps prevent a malicious user from discovering such a table exists.

Object-level security must be set using the JSON-based metadata, Tabular Model Scripting Language (TMSL), or Tabular Object Model (TOM).

For example, the following code helps secure the Product table in the sample Adventure Works tabular model by setting the **MetadataPermission** property of the **TablePermission** class to **None**.

```
//Find the Users role in Adventure Works and secure the Product table
ModelRole role = db.Model.Roles.Find("Users");
Table productTable = db.Model.Tables.Find("Product");
if (role != null && productTable != null)
{
    TablePermission tablePermission;
    if (role.TablePermissions.Contains(productTable.Name))
    {
        tablePermission = role.TablePermissions[productTable.Name];
    }
    else
    {
        tablePermission = new TablePermission();
        role.TablePermissions.Add(tablePermission);
        tablePermission.Table = productTable;
    }
    tablePermission.MetadataPermission = MetadataPermission.None;
}
db.Update(UpdateOptions.ExpandFull);
```

Dynamic Management Views (DMVs)

DMVs are queries in SQL Server Profiler that return information about local server operations and server health. This release includes improvements to Dynamic Management Views (DMV) for tabular models at the 1200 and 1400 compatibility levels.

DISCOVER_CALC_DEPENDENCY Now works with tabular 1200 and 1400 models. Tabular 1400 models show dependencies between M partitions, M expressions and structured data sources. To learn more, see the Analysis Services blog.

MDSCHEMA_MEASUREGROUP_DIMENSIONS Improvements are included for this DMV, which is used by various client tools to show measure dimensionality. For example, the Explore feature in Excel Pivot Tables allows the user to cross-drill to dimensions related to the selected measures. This release corrects the cardinality columns, which were previously showing incorrect values.

DAX enhancements

This release includes support for new DAX functions and functionality. In order to take advantage, you need to use the latest version of SSDT. To learn more, see New DAX functions.

One of the most important pieces of new DAX functionality is the new IN Operator / CONTAINSROW Function for DAX expressions. This is similar to the `TSQL IN` operator commonly used to specify multiple values in a `WHERE` clause.

Previously, it was common to specify multi-value filters using the logical `OR` operator, like in the following measure expression:

```
Filtered Sales:=CALCULATE (
        [Internet Total Sales],
                 'Product'[Color] = "Red"
            || 'Product'[Color] = "Blue"
            || 'Product'[Color] = "Black"
    )
```

This is simplified using the `IN` operator:

```
Filtered Sales:=CALCULATE (
        [Internet Total Sales], 'Product'[Color] IN { "Red", "Blue", "Black" }
    )
```

In this case, the `IN` operator refers to a single-column table with 3 rows; one for each of the specified colors. Note the table constructor syntax uses curly braces.

The `IN` operator is functionally equivalent to the `CONTAINSROW` function:

```
Filtered Sales:=CALCULATE (
        [Internet Total Sales], CONTAINSROW({ "Red", "Blue", "Black" }, 'Product'[Color])
    )
```

The `IN` operator can also be used effectively with table constructors. For example, the following measure filters by combinations of product color and category:

```
Filtered Sales:=CALCULATE (
        [Internet Total Sales],
        FILTER( ALL('Product'),
              ( 'Product'[Color] = "Red"   && Product[Product Category Name] = "Accessories" )
         || ( 'Product'[Color] = "Blue"  && Product[Product Category Name] = "Bikes" )
         || ( 'Product'[Color] = "Black" && Product[Product Category Name] = "Clothing" )
        )
    )
```

By using the new `IN` operator, the measure expression above is now equivalent to the one below:

```
Filtered Sales:=CALCULATE (
        [Internet Total Sales],
        FILTER( ALL('Product'),
            ('Product'[Color], Product[Product Category Name]) IN
            { ( "Red", "Accessories" ), ( "Blue", "Bikes" ), ( "Black", "Clothing" ) }
        )
    )
```

Additional improvements

In addition to all the new features, Analysis Services, SSDT, and SSMS also include the following improvements:

- Hierarchy and column reuse surfaced in more helpful locations in the Power BI field list.
- Date relationships to easily create relationships to date dimensions based on date fields.
- Default installation option for Analysis Services is now for tabular mode.
- New Get Data (Power Query) data sources.
- DAX Editor for SSDT.
- Existing DirectQuery data sources support for M queries.
- SSMS improvements, such as viewing, editing, and scripting support for structured data sources.

See also

SQL Server 2017 Release Notes
What's new in SQL Server 2017

Tools and applications used in Analysis Services

12/10/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Find the tools and applications you'll need for building Analysis Services models and managing deployed databases.

Analysis Services Model Designers

Models are authored by using project templates in SQL Server Data Tools (SSDT), a Visual Studio shell. Project templates provide model designers for creating the data model objects that comprise an Analysis Services solution. SSDT is a free web download updated monthly.

Download SQL Server Data Tools (SSDT)

Models have a compatibility level setting that determines feature availability and which release of Analysis Services run the model. Whether you can specify a given compatibility level is determined in-part by the model designer.

Tabular models using the latest functionality, such as BIM files in tabular JSON format and bi-directional cross filtering, must be created or upgraded at compatibility level 1200 or higher.

If you require a lower compatibility level, perhaps because you want to deploy a model on an earlier version of Analysis Services, you can still use the model designer in SSDT. Newer versions of the tool support creating any model type (tabular or multidimensional), at any compatibility level.

Administrative tools

SQL Server Management Studio (SSMS) is the primary administration tool for all SQL Server features, including Analysis Services. SSMS is a free web download updated monthly.

Download SQL Server Management Studio

SSMS includes extended events (xEvents), providing a lightweight alternative to SQL Server Profiler traces used for monitoring activity and diagnosing problems on SQL Server 2016 and Azure Analysis Services servers. See Monitor Analysis Services with SQL Server Extended Events to learn more.

SQL Server Profiler

Although it's officially deprecated in favor of xEvents, SQL Server Profiler provides a familiar approach for monitoring connections, MDX query execution, and other server operations. SQL Server Profiler is installed by default. You can find it with SQL Server applications on Apps in Windows.

PowerShell

You can use PowerShell commands to perform many administrative tasks. See PowerShell reference for more information.

Community and Third-party tools

Check the Analysis Services codeplex page for community code samples. Forums can be helpful when seeking recommendations for third-party tools that support Analysis Services.

See Also

Compatibility Level of a Multidimensional Database

Client libraries (data providers) used for Analysis Services connections

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services provides three client libraries, also known as **data providers**, for server and data access from tools and client applications. Tools like SSMS and SSDT, and applications like Power BI Desktop and Excel connect to Analysis Services by using these libraries. Two of the client libraries, ADOMD.NET and Analysis Services Management Objects (AMO), are managed client libraries. The Analysis Services OLE DB provider (MSOLAP DLL) is a native client library. Client libraries are the same for both SQL Server Analysis Services and Azure Analysis Services.

Where to get newer versions

The version installed on a client computer should match the major version of the server providing the data. If the server installation is newer than the data providers installed on the workstations in your network, you might need to install newer libraries.

Client libraries included in earlier SQL Server Feature Packs correspond to that SQL version; however, they may not be the latest. Connecting to Azure Analysis Services may require later versions. All versions are backward compatible.

To get the latest, see Client libraries for connecting to Azure Analysis Services.

See also

Connect to Analysis Services

Comparing tabular and multidimensional solutions

12/7/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

SQL Server Analysis Services provides several approaches for creating a business intelligence semantic model: Tabular, Multidimensional, and Power Pivot for SharePoint.

Having more than one approach enables a modeling experience tailored to different business and user requirements. Multidimensional is a mature technology built on open standards, embraced by numerous vendors of BI software, but can be hard to master. Tabular offers a relational modeling approach that many developers find more intuitive. Power Pivot is even simpler, offering visual data modeling in Excel, with server support provided via SharePoint.

All models are deployed as databases that run on an Analysis Services instance, accessed by client tools using a single set of data providers, and visualized in interactive and static reports via Excel, Reporting Services, Power BI, and BI tools from other vendors.

Tabular and multidimensional solutions are built using SSDT and are intended for corporate BI projects that run on a standalone Analysis Services instance on-premises, and for tabular models, an Azure Analysis Services server in the cloud. Both solutions yield high performance analytical databases that integrate easily with BI clients. Yet each solution differs in how they are created, used, and deployed. The bulk of this topic compares these two types so that you can identify the right approach for you.

For new projects, we generally recommend tabular models. Tabular models are faster to design, test, and deploy; and will work better with the latest self-service BI applications and cloud services from Microsoft.

Overview of modeling types

New to Analysis Services? The following table enumerates the different models, summarizes the approach, and identifies the initial release vehicle.

> **NOTE**
>
> **Azure Analysis Services** supports tabular models at the 1200 and higher compatibility levels. However, not all tabular modeling functionality described in this topic is supported in Azure Analysis Services. While creating and deploying tabular models to Azure Analysis Services is much the same as it is for on-premises, it's important to understand the differences. To learn more , see What is Azure Analysis Services?

Type	Modeling description	Released
Tabular	Relational modeling constructs (model, tables, columns). Internally, metadata is inherited from OLAP modeling constructs (cubes, dimensions, measures). Code and script use OLAP metadata.	SQL Server 2012 and later (compatibility levels 1050 - 1103) [1]

Tabular in SQL Server 2016	Relational modeling constructs (model, tables, columns), articulated in tabular metadata object definitions in Tabular Model Scripting Language (TMSL) and Tabular Object Model (TOM) code.	SQL Server 2016 (compatibility level 1200)
Tabular in SQL Server 2017	Relational modeling constructs (model, tables, columns), articulated in tabular metadata object definitions in Tabular Model Scripting Language (TMSL) and Tabular Object Model (TOM) code.	SQL Server 2017 (compatibility level 1400)
Multidimensional	OLAP modeling constructs (cubes, dimensions, measures).	SQL Server 2000 and later
Power Pivot	Originally an add-in, but now fully integrated into Excel. Visual modeling only, over an internal Tabular infrastructure. You can import a Power Pivot model into SSDT to create a new Tabular model that runs on an Analysis Services instance.	via Excel and Power Pivot BI Desktop

[1] Compatibility levels are significant in the current release due to tabular metadata engine and support for scenario-enabling features available only at the higher level. Later versions support earlier compatibility levels, but it is recommended you create new models or upgrade existing models to the highest compatibility level supported by the server version.

Model Features

The following table summarizes feature availability at the model level. Review this list to ensure that the feature you want to use is available in the type of model you plan to build.

	Multidimensional	Tabular
Actions	Yes	No
Aggregations	Yes	No
Calculated Column	No	Yes
Calculated Measures	Yes	Yes
Calculated Tables	No	Yes[1]
Custom Assemblies	Yes	No
Custom Rollups	Yes	No
Default Member	Yes	No
Display folders	Yes	Yes[1]

Distinct Count	Yes	Yes (via DAX)
Drillthrough	Yes	Yes (depends on client application)
Hierarchies	Yes	Yes
KPIs	Yes	Yes
Linked objects	Yes	Yes (linked tables)
M expressions	No	Yes[1]
Many-to-many relationships	Yes	No (but there is bi-directional cross filters at 1200 and higher compatibility levels)
Named sets	Yes	No
Ragged Hierarchies	Yes	Yes[1]
Parent-child Hierarchies	Yes	Yes (via DAX)
Partitions	Yes	Yes
Perspectives	Yes	Yes
Row-level Security	Yes	Yes
Object-level Security	Yes	Yes[1]
Semi-additive Measures	Yes	Yes
Translations	Yes	Yes
User-defined Hierarchies	Yes	Yes
Writeback	Yes	No

[1] See Compatibility Level for Tabular models in Analysis Services for information about functional differences between compatibility levels.

Data Considerations

Tabular and multidimensional models use imported data from external sources. The amount and type of data you need to import can be a primary consideration when deciding which model type best fits your data.

Compression

Both tabular and multidimensional solutions use data compression that reduces the size of the Analysis Services database relative to the data warehouse from which you are importing data. Because actual compression will vary based on the characteristics of the underlying data, there is no way to know precisely how much disk and memory will be required by a solution after data is processed and used in queries.

An estimate used by many Analysis Services developers is that primary storage of a multidimensional database will be about one third size of the original data. Tabular databases can sometimes get greater amounts of compression, about one tenth the size, especially if most of the data is imported from fact tables.

Size of the model and resource bias (in-memory or disk)

The size of an Analysis Services database is constrained only by the resources available to run it. The model type and storage mode will also play a role in how large the database can grow.

Tabular databases run either in-memory or in DirectQuery mode that offloads query execution to an external database. For tabular in-memory analytics, the database is stored entirely in memory, which means you must have sufficient memory to not only load all the data, but also additional data structures created to support queries.

DirectQuery, revamped in SQL Server 2016, has fewer restrictions than before, and better performance. Leveraging the backend relational database for storage and query execution makes building a large scale Tabular model more feasible than was previously possible.

Historically, the largest databases in production are multidimensional, with processing and query workloads running independently on dedicated hardware, each one optimized for its respective use. Tabular databases are catching up quickly, and new advancements in DirectQuery will help close the gap even further.

For multidimensional offloading data storage and query execution is available via ROLAP. On a query server, rowsets can be cached, and stale ones paged out. Efficient and balanced use of memory and disk resources often guide customers to multidimensional solutions.

Under load, both disk and memory requirements for either solution type can be expected to increase as Analysis Services caches, stores, scans, and queries data. For more information about memory paging options, see Memory Properties. To learn more about scale, see High availability and Scalability in Analysis Services.

Power Pivot for Excel has an artificial file size limit of 2 gigabytes, which is imposed so that workbooks created in Power Pivot for Excel can be uploaded to SharePoint, which sets maximum limits on file upload size. One of the main reasons for migrating a Power Pivot workbook to a tabular solution on a standalone Analysis Services instance is to get around the file size limitation. For more information about configuring maximum file upload size, see Configure Maximum File Upload Size (Power Pivot for SharePoint).

Supported Data Sources

Tabular models can import data from relational data sources, data feeds, and some document formats. You can also use OLE DB for ODBC providers with tabular models. Tabular models at the 1400 compatibility level offer a significant increase in the variety of data sources from which you can import from. This is due to the introduction of the modern Get Data data query and import features in SSDT utilizing the M formula query language.

Multidimensional solutions can import data from relational data sources using OLE DB native and managed providers.

To view the list of external data sources that you can import to each model, see the following topics:

- Data Sources Supported
- Supported Data Sources (SSAS - Multidimensional)

Query and Scripting Language Support

Analysis Services includes MDX, DMX, DAX, XML/A, ASSL, and TMSL. Support for these languages can vary by model type. If query and scripting language requirements are a consideration, review the following list.

- Tabular model databases support DAX calculations, DAX queries, and MDX queries. This is true at all compatibilities levels. Script languages are ASSL (over XMLA) for compatibility levels 1050-1103, and TMSL (over XMLA) for compatibility level 1200 and higher.

- Power Pivot workbooks use DAX for calculations, and DAX or MDX for queries.
- Multidimensional model databases support MDX calculations, MDX queries, DAX queries, and ASSL.
- Data mining models support DMX and ASSL.
- Analysis Services PowerShell is supported for Tabular and Multidimensional models and databases.

All databases support XML/A. See Query and Expression Language Reference (Analysis Services) and Analysis Services Developer Documentation for more information.

Security Features

All Analysis Services solutions can be secured at the database level. More granular security options vary by mode. If granular security settings are requirement for your solution, review the following list to ensure the level of security you want is supported in the type of solution you want to build:

- Tabular model databases can use row-level security, using role-based permissions.
- Multidimensional model databases can use dimension and cell-level security, using role-based permissions.
- Power Pivot workbooks are secured at the file level, using SharePoint permissions.

Power Pivot workbooks can be restored to a Tabular mode server. Once the file is restored, it is decoupled from SharePoint, allowing you to use all tabular modeling features, including row-level security.

Design Tools

Data modeling skills and technical expertise can vary widely among users who are tasked with building analytical models. If tool familiarity or user expertise is a consideration for your solution, compare the following experiences for model creation.

MODELING TOOL	HOW USED
SQL Server Data Tools (SSDT)	Use to create tabular, multidimensional, and data mining solutions. This authoring environment uses the Visual Studio shell to provide workspaces, property panes, and object navigation. Technical users who already use Visual Studio will most likely prefer this tool for building business intelligence applications.
Power Pivot for Excel	Use to create a Power Pivot workbook that you later deploy to a SharePoint farm that has an installation of Power Pivot for SharePoint. Power Pivot for Excel has a separate application workspace that opens over Excel. It uses the same visual metaphors (tabbed pages, grid layout, and formula bar) as Excel. Users who are proficient in Excel will prefer this tool over SQL Server Data Tools (SSDT).

Client Application Support

In-general, tabular and multidimensional solutions support client applications using one or more of the Analysis Services client libraries (MSOLAP, AMOMD, ADOMD). For example, Excel, Power BI Desktop, and custom applications.

If you are using Reporting Services, report feature availability varies across editions and server modes. For this reason, the type of report that you want to build might influence which server mode you choose to install.

Power View, a Reporting Services authoring tool that runs in SharePoint, is available on a report server that is

deployed in a SharePoint 2010 farm. The only type of data source that can be used with this report is an Analysis Services tabular model database or a Power Pivot workbook. This means that you must have a tabular mode server or a Power Pivot for SharePoint server to host the data source used by this type of report. You cannot use a multidimensional model as a data source for a Power View report. You must create a Power Pivot BI Semantic Model connection or a Reporting Services shared data source to use as the data source for a Power View report.

Report Builder and Report Designer can use any Analysis Services database, including Power Pivot workbooks that are hosted on Power Pivot for SharePoint.

Excel PivotTable reports are supported by all Analysis Services databases. Excel functionality is the same whether you use a tabular .database, multidimensional database, or Power Pivot workbook, although Writeback is only supported for multidimensional databases.

See Also

Analysis Services Instance Management
What's New in Analysis Services

Analysis Services Features and Tasks

12/10/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services foundational documentation is organized by model type and server mode so that you can focus on just the tools, tasks, and features that are available in the mode you installed. Server administration tasks that span multiple modes can be found in Instance Management.

In This Section

Comparing Tabular and Multidimensional Solutions

Tools and applications used in Analysis Services

Analysis Services Instance Management

Tabular Models

Multidimensional Models

Data Mining

Power Pivot for SharePoint

See Also

Analysis Services
What's New in Analysis Services

Analysis Services features supported by SQL Server editions

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

This topic provides details of features supported by the different editions of SQL Server 2016 Analysis Services. For features supported by Evaluation and Developer editions, see Enterprise edition.

Analysis Services (servers)

FEATURE	ENTERPRISE	STANDARD	WEB	EXPRESS WITH ADVANCED SERVICES	EXPRESS WITH TOOLS	EXPRESS	DEVELOPER
Scalable shared databases	Yes						Yes
Backup/Restore & Attach/Detach databases	Yes	Yes					Yes
Synchronize databases	Yes						Yes
Always On failover cluster instances	Yes Number of nodes is the operating system maximum	Yes Support for 2 nodes					Yes Number of nodes is the operating system maximum
Programmability (AMO, ADOMD.Net, OLEDB, XML/A, ASSL, TMSL)	Yes	Yes					Yes

Tabular models

FEATURE	ENTERPRISE	STANDARD	WEB	EXPRESS WITH ADVANCED SERVICES	EXPRESS WITH TOOLS	EXPRESS	DEVELOPER
Hierarchies	Yes	Yes					Yes

FEATURE	ENTERPRISE	STANDARD	WEB	EXPRESS WITH ADVANCED SERVICES	EXPRESS WITH TOOLS	EXPRESS	DEVELOPER
KPIs	Yes	Yes					Yes
Perspectives	Yes						Yes
Translations	Yes	Yes					Yes
DAX calculations, DAX queries, MDX queries	Yes	Yes					Yes
Row-level security	Yes	Yes					Yes
Multiple partitions	Yes						Yes
In-memory storage mode	Yes	Yes					Yes
DirectQuery storage mode	Yes						Yes

Multidimensional models

FEATURE	ENTERPRISE	STANDARD	WEB	EXPRESS WITH ADVANCED SERVICES	EXPRESS WITH TOOLS	EXPRESS	DEVELOPER
Semi-additive measures	Yes	No [1]					Yes
Hierarchies	Yes	Yes					Yes
KPIs	Yes	Yes					Yes
Perspectives	Yes						Yes
Actions	Yes	Yes					Yes
Account intelligence	Yes	Yes					Yes

FEATURE	ENTERPRISE	STANDARD	WEB	EXPRESS WITH ADVANCED SERVICES	EXPRESS WITH TOOLS	EXPRESS	DEVELOPER
Time intelligence	Yes	Yes					Yes
Custom rollups	Yes	Yes					Yes
Writeback cube	Yes	Yes					Yes
Writeback dimensions	Yes						Yes
Writeback cells	Yes	Yes					Yes
Drillthrough	Yes	Yes					Yes
Advanced hierarchy types (parent-child and ragged hierarchies)	Yes	Yes					Yes
Advanced dimensions (reference dimensions, many-to-many dimensions)	Yes	Yes					Yes
Linked measures and dimensions	Yes	Yes [2]					Yes
Translations	Yes	Yes					Yes
Aggregations	Yes	Yes					Yes
Multiple partitions	Yes	Yes, up to 3					Yes
Proactive caching	Yes						Yes
Custom assemblies (stored procedures)	Yes	Yes					Yes

FEATURE	ENTERPRISE	STANDARD	WEB	EXPRESS WITH ADVANCED SERVICES	EXPRESS WITH TOOLS	EXPRESS	DEVELOPER
MDX queries and scripts	Yes	Yes					Yes
DAX queries	Yes	Yes					Yes
Role-based security model	Yes	Yes					Yes
Dimension and cell-level security	Yes	Yes					Yes
Scalable string storage	Yes	Yes					Yes
MOLAP, ROLAP, and HOLAP storage models	Yes	Yes					Yes
Binary and compressed XML transport	Yes	Yes					Yes
Push-mode processing	Yes						Yes
Direct writeback	Yes						Yes
Measure expressions	Yes						Yes

[1] The LastChild semi-additive measure is supported in Standard edition, but other semi-additive measures, such as None, FirstChild, FirstNonEmpty, LastNonEmpty, AverageOfChildren, and ByAccount, are not. Additive measures, such as Sum, Count, Min, Max, and non-additive measures (DistinctCount) are supported on all editions.
[2] Standard edition supports linking measures and dimensions within the same database, but not from other databases or instances.

Power Pivot for SharePoint

FEATURE	ENTERPRISE	STANDARD	WEB	EXPRESS WITH ADVANCED SERVICES	EXPRESS WITH TOOLS	EXPRESS	DEVELOPER
SharePoint farm integration based on shared service architecture	Yes						Yes
Usage reporting	Yes						Yes
Health monitoring rules	Yes						Yes
Power Pivot gallery	Yes						Yes
Power Pivot data refresh	Yes						Yes
Power Pivot data feeds	Yes						Yes

Data Mining

FEATURE NAME	ENTERPRISE	STANDARD	WEB	EXPRESS WITH ADVANCED SERVICES	EXPRESS WITH TOOLS	EXPRESS	DEVELOPER
Standard algorithms	Yes	Yes					Yes
Data mining tools (Wizards, Editors, Query Builders)	Yes	Yes					Yes
Cross validation	Yes						Yes
Models on filtered subsets of mining structure data	Yes						Yes

FEATURE NAME	ENTERPRISE	STANDARD	WEB	EXPRESS WITH ADVANCED SERVICES	EXPRESS WITH TOOLS	EXPRESS	DEVELOPER
Time series: Custom blending between ARTXP and ARIMA methods	Yes						Yes
Time series: Prediction with new data	Yes						Yes
Unlimited concurrent DM queries	Yes						Yes
Advanced configuratio n & tuning options for data mining algorithms	Yes						Yes
Support for plug-in algorithms	Yes						Yes
Parallel model processing	Yes						Yes
Time series: cross-series prediction	Yes						Yes
Unlimited attributes for association rules	Yes						Yes
Sequence prediction	Yes						Yes
Multiple prediction targets for naïve Bayes, neural network and logistic regression	Yes						Yes

See also

Product Specifications for SQL Server 2016
Installation for SQL Server

Analysis Services backward compatibility (SQL Server 2016)

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server 2016 Analysis Services Azure Analysis Services

This article describes changes in feature availability and behavior between the current version and the previous version.

Deprecated features

A *deprecated feature* will be discontinued from the product in a future release, but is still supported and included in the current release to maintain backward compatibility. It's recommended you discontinue using deprecated features in new and existing projects to maintain compatibility with future releases.

The following features are deprecated in this release:

Mode/Category	Feature
Multidimensional	Remote partitions
Multidimensional	Remote linked measure groups
Multidimensional	Dimensional writeback
Multidimensional	Linked dimensions
Multidimensional	SQL Server table notifications for proactive caching. The replacement is to use polling for proactive caching. See Proactive Caching (Dimensions) and Proactive Caching (Partitions).
Multidimensional	Session cubes. There is no replacement.
Multidimensional	Local cubes. There is no replacement.
Tabular	Tabular model 1100 and 1103 compatibility levels will not be supported in a future release. The replacement is to set models at compatibility level 1200 or higher, converting model definitions to tabular metadata. See Compatibility Level for Tabular models in Analysis Services.
Tools	SQL Server Profiler for Trace Capture The replacement is to use Extended Events Profiler embedded in SQL Server Management Studio. See Monitor Analysis Services with SQL Server Extended Events.

Tools	Server Profiler for Trace Replay Replacement. There is no replacement.
Trace Management Objects and Trace APIs	Microsoft.AnalysisServices.Trace objects (contains the APIs for Analysis Services Trace and Replay objects). The replacement is multi-part: - Trace Configuration: Microsoft.SqlServer.Management.XEvent - Trace Reading: Microsoft.SqlServer.XEvent.Linq - Trace Replay: None

> **NOTE**
> Previously deprecated feature announcements from SQL Server 2014 (12.x) remain in effect. Because the code supporting those features has not yet been cut from the product, many of these features are still present in this release. While previously deprecated features might be accessible, they are still considered deprecated and could be physically removed from the product at any time.

Discontinued features

A *discontinued feature* was deprecated in an earlier release. It may continue to be included in the current release, but is no longer supported. Discontinued features may be removed entirely in a future release or update.

The following features were deprecated in an earlier release and are no longer supported in this release.

Feature	Replacment or workaround
CalculationPassValue (MDX)	None. This feature was deprecated in SQL Server 2005.
CalculationCurrentPass (MDX)	None. This feature was deprecated in SQL Server 2005.
NON_EMPTY_BEHAVIOR query optimizer hint	None. This feature was deprecated in SQL Server 2008.
COM assemblies	None. This feature was deprecated in SQL Server 2008.
CELL_EVALUATION_LIST intrinsic cell property	None. This feature was deprecated in SQL Server 2005.

> **NOTE**
> Previously deprecated feature announcements from SQL Server 2014 (12.x) remain in effect. Because the code supporting those features has not yet been cut from the product, many of these features are still present in this release. While previously deprecated features might be accessible, they are still considered deprecated and could be physically removed from the product at any time.

Breaking changes

A *breaking change* causes a data model, application code, or script to no longer function after upgrading either the model or the server.

.NET 4.0 version upgrade

Analysis Services Management Objects (AMO), ADOMD.NET, and Tabular Object Model (TOM) client libraries now target the .NET 4.0 runtime. This can be a breaking change for applications that target .NET 3.5. Applications using newer versions of these assemblies must now target .NET 4.0 or later.

AMO version upgrade

This release is a version upgrade for Analysis Services Management Objects (AMO) and is a breaking change under certain circumstances. Existing code and scripts that call into AMO will continue to run as before if you upgrade from a previous version. However, if you need to *recompile* your application and you are targeting a SQL Server 2016 Analysis Services instance, you must add the following namespace to make your code or script operational:

```
using Microsoft.AnalysisServices;
using Microsoft.AnalysisServices.Core;
```

The Microsoft.AnalysisServices.Core namespace is now required whenever you reference the Microsoft.AnalysisServices assembly in your code. Objects that were previously only in the **Microsoft.AnalysisServices** namespace are moved to the Core namespace in this release if the object is used the same way in both tabular and multidimensional scenarios. For example, server-related APIs are relocated to the Core namespace.

Although there are now multiple namespaces, both exist in the same assembly (Microsoft.AnalysisServices.dll).

XEvent DISCOVER changes

To better support XEvent DISCOVER streaming in SSMS for SQL Server 2016 Analysis Services, `DISCOVER_XEVENT_TRACE_DEFINITION` is replaced with the following XEvent traces:

- DISCOVER_XEVENT_PACKAGES
- DISCOVER_XEVENT_OBJECT
- DISCOVER_XEVENT_OBJECT_COLUMNS
- DISCOVER_XEVENT_SESSION_TARGETS

Behavior changes

A *behavior change* affects how features work or interact in the current version as compared to earlier versions of SQL Server.

Revisions to default values, manual configuration required to complete an upgrade or restore functionality, or a new implementation of an existing feature are all examples of a behavior change in the product.

Feature behaviors that changed in this release, yet do not break an existing model or code post-upgrade, are listed here.

Analysis Services in SharePoint mode

Running the Power Pivot Configuration wizard is no longer required as a post-installation task. This is true for all supported versions of SharePoint that load models from the current SQL Server 2016 Analysis Services.

DirectQuery mode for Tabular models

DirectQuery is a data access mode for tabular models, where query execution is performed on a backend relational database, retrieving a result set in real time. It's often used for very large datasets that cannot fit in memory or when data is volatile and you want the most recent data returned in queries against a tabular model.

DirectQuery has existed as a data access mode for the last several releases. In SQL Server 2016 Analysis Services,

the implementation has been slightly revised, assuming the tabular model is at compatibility level 1200 or higher. DirectQuery has fewer restrictions than before. It also has different database properties.

If you are using DirectQuery in an existing tabular model, you can keep the model at its currently compatibility level of 1100 or 1103 and continue to use DirectQuery as its implemented for those levels. Alternatively, you can upgrade to 1200 or higher to benefit from enhancements made to DirectQuery.

There is no in-place upgrade of a DirectQuery model because the settings from older compatibility levels do not have exact counterparts in the newer 1200 and higher compatibility levels. If you have an existing tabular model that runs in DirectQuery mode, you should open the model in SQL Server Data Tools, turn DirectQuery off, set the **Compatibility Level** property to 1200 or higher, and then reconfigure the DirectQuery properties. See DirectQuery Mode for details.

See also

Analysis Services backward compatibility (SQL Server 2017)

Analysis Services backward compatibility (SQL 2017)

1/9/2019 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server 2017 Analysis Services Azure Analysis Services

This article describes changes in feature availability and behavior between the current release and the previous release.

Deprecated features

A *deprecated feature* will be discontinued from the product in a future release, but is still supported and included in the current release to maintain backward compatibility. It's recommended you discontinue using deprecated features in new and existing projects to maintain compatibility with future releases. Documentation is not updated for deprecated features.

The following features are deprecated in this release:

Mode/Category	Feature
Multidimensional	Data Mining
Multidimensional	Remote linked measure groups
Tabular	Models at the 1100 and 1103 compatibility level
Tabular	Tabular Object Model properties: Column.TableDetailPosition, Column.IsDefaultLabel, Column.IsDefaultImage
Tools	SQL Server Profiler for Trace Capture The replacement is to use Extended Events Profiler embedded in SQL Server Management Studio. See Monitor Analysis Services with SQL Server Extended Events.
Tools	Server Profiler for Trace Replay Replacement. There is no replacement.
Trace Management Objects and Trace APIs	Microsoft.AnalysisServices.Trace objects (contains the APIs for Analysis Services Trace and Replay objects). The replacement is multi-part: - Trace Configuration: Microsoft.SqlServer.Management.XEvent - Trace Reading: Microsoft.SqlServer.XEvent.Linq - Trace Replay: None

Discontinued features

A *discontinued feature* was deprecated in an earlier release. It may continue to be included in the current release, but is no longer supported. Discontinued features may be removed entirely in a future release or update.

The following features were deprecated in an earlier release and are no longer supported in this release.

Mode/Category	Feature
Tabular	VertipaqPagingPolicy memory property value (2), enable paging to disk using memory mapped files.
Multidimensional	Remote partitions
Multidimensional	Remote linked measure groups
Multidimensional	Dimensional writeback
Multidimensional	Linked dimensions

Breaking changes

A *breaking change* causes a feature, data model, application code, or script to no longer function after upgrading to the current release.

There are no breaking changes in this release.

Behavior changes

A *behavior change* affects how the same feature works in the current release as compared to the previous release. Only significant behavior changes are described. Changes in user interface are not included.

Changes to MDSCHEMA_MEASUREGROUP_DIMENSIONS and DISCOVER_CALC_DEPENDENCY, detailed in the What's new in SQL Server 2017 CTP 2.1 for Analysis Services announcement.

See also

Analysis Services backward compatibility (SQL Server 2016)

Globalization scenarios for Analysis Services

12/10/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services stores and manipulates multilingual data and metadata for both tabular and multidimensional data models. Data storage is Unicode (UTF-16), in character sets that use Unicode encoding. If you load ANSI data into a data model, characters are stored using Unicode equivalent code points.

The implications of Unicode support means that Analysis Services can store data in any of the languages supported by the Windows client and server operating systems, allowing read, write, sort, and comparison of data in any character set used on a Windows computer. BI client applications consuming Analysis Services data can represent data in the user's language of choice, assuming the data exists in that language in the model.

Language support can mean different things to different people. The following list addresses a few common questions related to how Analysis Services supports languages.

- Data, as already noted, is stored in any Unicode-encoded character set found on a Windows client operating system.
- Metadata, such as object names, can be translated. Although support varies by model type, both multidimensional and tabular models support the addition of translated strings inside the model. You can define multiple translations, and then use a locale identifier to determine which translation is returned to the client. See Features below for more details
- Error, warning, and informational messages returned by the Analysis Services engine (msmdsrv) are localized into the 43 languages supported by Office and Office 365. No configuration is required to get messages in a specific language. The locale of the client application determines which strings are returned.
- Configuration file (msmdsrv.ini) and AMO PowerShell are in English only.
- Log files will contain a mix of English and localized messages, assuming you have installed a language pack on the Windows server that Analysis Services runs on.
- Documentation and tools, such as Management Studio and SQL Server Data Tools, are translated into these languages: Chinese Simplified, Chinese Traditional, French, German, Italian, Japanese, Korean, Portuguese (Brazil), Russian, and Spanish. Culture is specified during installation.

For multidimensional models, Analysis Services lets you set language, collation, and translations independently throughout the object hierarchy. For tabular models, you can only add translations: language and collation are inherited by the host operating system.

Scenarios enabled through Analysis Services globalization features include:

- One data model provides multiple translated captions so that field names and values appear in the user's language of choice. For companies operating in bi-lingual countries such as Canada, Belgium, or Switzerland, supporting multiple languages across client and server applications is a standard coding requirement. This scenario is enabled through translations and currency conversions. See Features below for details and links.
- Development and production environments are geo-located in different countries. It's increasingly common to develop a solution in one country and then deploy it another. Knowing how to set language and collation properties is essential if you are tasked with preparing a solution developed in one language, for deployment on a server that uses a different language pack. Setting these properties allows you to

override the inherited defaults that you get from the original host system. See Languages and Collations (Analysis Services) for details about setting properties.

Features for building a globalized multi-lingual solution

At the client level, globalized applications that consume or manipulate Analysis Services multidimensional data can use the multilingual and multicultural features in Analysis Services.

You can retrieve data and metadata from Analysis Services objects on which translations have been defined automatically by providing a locale identifier when connecting to an Analysis Services instance.

See Globalization Tips and Best Practices (Analysis Services) for design and coding practices that can help you avoid problems related to multi-language data.

Capability	Tabular	Multidimensional
Languages and Collations (Analysis Services)	Inherited from the operating system.	Inherited, but with the ability to override both language and collation for major objects in the model hierarchy.
Scope of translation support	Captions and descriptions.	Translations can be created for object names, captions, identifiers, and descriptions, can also be in any Unicode language and script. This is true even when the tools and environment are in another language. For example, in a development environment that uses English language and a Latin collation throughout the stack, you can include in your model an object that uses Cyrillic characters in its name.
Implementing translation support	Create using SQL Server Data Tools (SSDT) to generate translation files that you fill in and then import back into the model. See Translations in Tabular models (Analysis Services) for details.	Create usingSQL Server Data Tools (SSDT) to define the translations for the caption, description, and account types for cubes and measures, dimension and attributes. See Translations in Multidimensional Models (Analysis Services) for more information. A lesson on how to use this feature can be found in Lesson 9: Defining Perspectives and Translations of the Analysis Services tutorial.
Currency conversion	Not available.	Currency conversion is through specialized MDX scripts that convert measures containing currency data. You can use the Business Intelligence Wizard in SQL Server Data Tools - Business Intelligence to generate an MDX script that uses a combination of data and metadata from dimensions, attributes, and measure groups to convert measures containing currency data. See Currency Conversions (Analysis Services).

See Also

Translation support in Analysis Services
Internationalization for Windows Applications
Go Global Developer Center
Writing Windows Store apps with locale-based adaptive design
Developing Universal Windows Apps with C# and XAML

Languages and Collations (Analysis Services)

12/10/2018 • 13 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services supports the languages and collations provided by Microsoft Windows operating systems. **Language** and **Collation** properties are initially set at the instance level during installation, but can be changed afterwards at different levels of the object hierarchy.

In a multidimensional model (only), you can set these properties on a database or cube - you can also set them on translations that you create for objects within a cube. In a tabular model, language and collation are inherited from the host operating system.

When setting **Language** and **Collation** in a multidimensional model, you are either specifying settings used by the data model during processing and query execution, or you are equipping a model with multiple translations so that foreign language speakers can work with the model in their native language. Explicitly setting **Language** and **Collation** properties on an object (database, model, or cube) is for situations where the development environment and production server are configured for different locales, and you want to be sure that the language and collation matches those of the intended target environment.

Objects that support Language and Collation properties

Language and **Collation** properties are often exposed together - where you can set **Language**, you can also set **Collation**.

You can set **Language** and **Collation** on these objects:

- **Instance**. All projects deployed to the instance will adopt the language and collation of the instance, assuming the language and collation are undefined. By default, a multidimensional model leaves language and collation empty. When the project is deployed, the resulting database and cubes get the language and collation of the instance.

  Initially, language and collation properties are established during setup but an administrator can override them in Management Studio. See Change the default language or collation on the instance for details.

- **Database**. To break inheritance, you can explicitly set language and collation at the project level that is used by all cubes contained in the database. Unless you indicate otherwise, all cubes in the database will get the language and collation you specify at this level. If you routinely code and deploy to different locales (for example, developing the solution on a Chinese computer, but deploying it to a server owned by a French subsidiary), setting language and collation at the database level is the first and most important step to ensuring the solution works in the target environment. The best place to set these properties is inside the project (via the **Edit Database** command on the project).

- **Database dimension**. Although the designer exposes **Language** and **Collation** properties on a database dimension, setting properties on this object is not useful. Database dimensions are not used as standalone objects, so it could be difficult if not impossible to make use of the properties you define. When in a cube, a dimension always inherits **Language** and **Collation** from its cube parent. Any values you might have set on the standalone database dimension object are ignored.

- **Cube**. As the primary query structure, you can set language and collation at the cube level. For example, you might want to create multiple language versions of a cube, such as English and Chinese versions, within the same project, where each cube has its own language and collation.

Whatever language and collation you set on the cube is used by all measures and dimensions contained in the cube. The only way to set collation properties at a finer grain is if you are creating translations on a dimension attribute. Otherwise, assuming no translations at the attribute level, there is one collation per cube.

Additionally, you can set **Language**, by itself, on a **Translation** object.

A translation object is created when you add translations to a cube or dimension. **Language** is part of the translation definition. **Collation**, on the other hand, is set on the cube or higher, and shared by all translations. This is evident in the XMLA of a cube containing translations, where you will see multiple language properties (one for each translation), but only one collation. Note that there is one exception for dimension attribute translations, where you can override cube collation to specify an attribute collation that matches the source column (the database engine supports setting collation on individual columns, and it's common to configure individual translations to obtain member data from different source columns). But otherwise, for all other translations, **Language** is used by itself, without a **Collation** corollary. See Translation support in Analysis Services for details.

Language support in Analysis Services

The **Language** property sets the locale of an object, used during processing, queries, and with **Captions** and **Translations** to support multi-lingual scenarios. Locales are based on a language identifier, such as English, and a territory, such as United States or Australia, that further refines date and time representations.

At the instance level, the property is set during installation and is based on the language of the Windows server operating system (one of 37 languages, assuming a language pack is installed). You cannot change the language in Setup.

Post-installation, you can override **Language** using the server properties page in Management Studio, or in the msmdsrv.ini configuration file. You can choose from many more languages, including all those supported by the Windows client. When set at the instance level, on the server, **Language** determines the locale of all databases that are subsequently deployed. For example, if you set **Language** to German, all databases that are deployed to the instance will have a Language property of 1031, the LCID for German.

Value of the Language property is a Locale Identifier (LCID)

Valid values include any LCID that appears in the drop-down list. In Management Studio and SQL Server Data Tools, LCIDs are represented in string equivalents. The same languages show up wherever the **Language** property is exposed, regardless of the tool. Having an identical list of languages ensures that you can implement and test translations consistently throughout the model.

Although Analysis Services lists the languages by name, the actual value stored for the property is an LCID. When setting a language property programmatically or through the msmdsrv.ini file, use the locale identifier (LCID) as the value. An LCID is a 32-bit value consisting of a language ID, sort ID, and reserved bits that identify a particular language. Analysis Services uses LCIDs to specify the selected language for Analysis Services instances and objects.

You can set the LCID using either hexadecimal or decimal formats. A few examples of valid values for the **Language** property include:

- 0x0409 or 1033 for **English (United States)**
- 0x0411 or 1041 for **Japanese**
- 0x0407 or 1031 for **Germany (German)**
- 0x0416 or 1046 for **Portuguese (Brazil)**.

To view a longer list, see Locale IDs Assigned by Microsoft. For more background, see Encoding and Code Pages.

> **NOTE**
> The **Language** property does not determine the language for returning system messages, or which strings appear in the user interface. Errors, warnings, and messages are localized into all languages supported in Office and Office 365 and are used automatically when the client connection specifies one of the supported locales.

Collation support in Analysis Services

Analysis Services uses Windows (versions _90 and _100) and binary collations exclusively. It does not use legacy SQL Server collations. Within a cube, a single collation is used throughout, with the exception of translations at the attribute level. For more information about defining attribute translations, see Translation support in Analysis Services.

Collations control the case-sensitivity of all strings in a bicameral language script, with the exception of object identifiers. If you use upper and lower case characters in an object identifier, be forewarned that the case-sensitivity of object identifiers is determined not by the collation, but by Analysis Services. For object identifiers composed in English script, object identifiers are always case-insensitive, regardless of collation. Cyrillic and other bicameral languages do the opposite (always case-sensitive). See Globalization Tips and Best Practices (Analysis Services) for details.

Collation in Analysis Services is compatible with that of the SQL Server relational database engine, assuming you maintain parity in the sort options you select for each service. For example, if the relational database is accent sensitive, you should configure the cube the same way. Problems can occur when collations settings diverge. For an example and workarounds, see Blanks in a Unicode string have different processing outcomes based on collation. To learn more about collation and the database engine, see Collation and Unicode Support.

Collation Types

Analysis Services supports two collation types:

- **Windows collations (versions _90 and _100)**

  Windows collation versions are _90 (an unmarked older version) and the newer _100 version. Only the _100 version shows the version number in the collation name:

  - latin1_general
  - latin1_general_100

  A Windows collation sorts characters based on the linguistic and cultural characteristics of the language. In Windows, collations outnumber the locales (or languages) used with them, due to many languages sharing common alphabets and rules for sorting and comparing characters. For example, 33 Windows locales, including all the Portuguese and English Windows locales, use the Latin1 code page (1252) and follow a common set of rules for sorting and comparing characters.

  > **NOTE**
  > When deciding on a collation, you should go with the same collation used by the underlying database. However, if you can choose, the _100 version is more up-to-date and offers a more linguistically accurate cultural sorting convention.

- **Binary collations (either BIN or BIN2)**

  Binary collations sort on Unicode code points, not on linguistic values. For example, Latin1_General_BIN and Japanese_BIN yield identical sorting results when used on Unicode data. Whereas a linguistic sort might yield results like aAbBcCdD, a binary sort would be ABCDabcd because the code point of all

uppercase characters is collectively higher than the code points of the lowercase characters.

Sort Order Options

Sort options are used to refine sorting and comparison rules based on case, accent, kana, and width sensitivity. For example, the default value of the **Collation** configuration property for Analysis Services is Latin1_General_AS_CS, specifying that the Latin1_General collation is used, with an accent-sensitive, case-sensitive sort order.

Note that BIN and BIN2 are mutually exclusive of other sort options, if you want to use BIN or BIN2, clear the sort option for Accent Sensitive. Similarly, when BIN2 is selected, the case-sensitive, case-insensitive, accent-sensitive, accent-insensitive, kana-sensitive, and width-sensitive options are not available.

The following table describes Windows collation sort order options and associated suffixes for Analysis Services.

SORT ORDER (SUFFIX)	SORT ORDER DESCRIPTION
Binary (_BIN) or BIN2 (_BIN2)	There are two types of binary collations in SQL Server; the older BIN collations and the newer BIN2 collations. In a BIN2 collation all characters are sorted according to their code points. In a BIN collation only the first character is sorted according to the code point, and remaining characters are sorted according to their byte values. (Because the Intel platform is a little endian architecture, Unicode code characters are always stored byte-swapped.) For binary collations on Unicode data types, the locale is not considered in data sorts. For example, Latin_1_General_BIN and Japanese_BIN yield identical sorting results when they are used on Unicode data. Binary sort order is case sensitive and accent sensitive. Binary is also the fastest sorting order.
Case-sensitive (_CS)	Distinguishes between uppercase and lowercase letters. If selected, lowercase letters sort ahead of their uppercase versions. You can explicitly set case-insensitivity by specifying _CI. Collation-specific case settings do not apply to object identifiers, such as the ID of a dimension, cube, and other objects. See Globalization Tips and Best Practices (Analysis Services) for details.
Accent-sensitive (_AS)	Distinguishes between accented and unaccented characters. For example, 'a' is not equal to 'á'. If this option is not selected, Analysis Services considers the accented and unaccented versions of letters to be identical for sorting purposes. You can explicitly set accent-insensitivity by specifying _AI.
Kana-sensitive (_KS)	Distinguishes between the two types of Japanese kana characters: hiragana and katakana. If this option is not selected, Analysis Services considers hiragana and katakana characters to be equal for sorting purposes. There is no sort order suffix for kana-insensitive sorting.
Width-sensitive (_WS)	Distinguishes between a single-byte character and the same character when represented as a double-byte character. If this option is not selected, Analysis Services considers the single-byte and double-byte representation of the same character to be identical for sorting purposes. There is no sort order suffix for width-insensitive sorting.

Change the default language or collation on the instance

Default language and collation is established during setup, but can be changed as part of post-installation configuration. Changing the collation at the instance level is non-trivial and comes with these requirements:

- A service restart.
- Update the collation settings of existing objects. Collation settings are inherited once when the object is created. Subsequent changes to collation must be done manually. See Change language and collation within a data model using XMLA for tips on how to propagate collation changes throughout the model.
- Reprocess partitions and dimensions after the collation is updated.

You can use SQL Server Management Studio or AMO PowerShell to change the default language or collation at the server level. Alternatively, you can modify the **<Language>** and **<CollationName>** settings in the msmdsrv.ini file, specifying the LCID of the language.

1. In Management Studio, right-click server name | **Properties** | **Language/Collation**.
2. Choose the sort options. To select either **Binary** or **Binary 2**, first clear the checkbox for **Accent Sensitive**.

   Note that collation and language are fully independent settings. If you change one, the values of the other are not filtered to show common combinations.
3. Update the data model to use the new collation (see the following section).
4. Restart the service.

Change the language or collation on a cube

1. In Solution Explorer, double-click the cube to open it in cube designer.
2. In either the Measures or Dimensions pane, select the top node. The top-level object for either pane is the cube.
3. In Properties, set **Language** and **Collation**. The values you choose will be used by all cube objects, including cube dimensions and measures, and affects processing and query operations.

   The only way to embed alternate language and collation properties on objects within the cube is through translations. See Translation support in Analysis Services for details.

Change language and collation within a data model using XMLA

Language and collation settings are inherited once when the object is created. Subsequent changes to those properties must be done manually. One approach for changing collation multiple objects quickly is to use an ALTER command on an XMLA script.

By default, collation is set once, at the database level. Inheritance is implied throughout the rest of the object hierarchy. If you explicitly set **Collation** on objects within the cube, which is allowed on individual dimension attributes, it will appear in the XMLA definition. Otherwise, only the top-level collation property exists.

Before using XMLA to modify an existing database, make sure that you're not introducing discrepancies between the database and the source files used to build it. For example, you might want to use XMLA to quickly change language or collation for proof-of-concept testing, but then follow-up with changes to the source file (see Change the language or collation on a cube), redeploying the solution using the existing operating procedures already in place.

1. In Management Studio, right-click the database | **Script Database as** | **ALTER To** | **New Query Editor Window**.

2. Search and replace the existing language or collation with an alternative value.
3. Press F5 to execute the script.
4. Reprocess the cube.

Boost performance for English locales through EnableFast1033Locale

If you use the English (United States) language identifier (0x0409, or 1033) as the default language for the Analysis Services instance, you can get additional performance benefits by setting the **EnableFast1033Locale** configuration property, an advanced configuration property available only for that language identifier. Setting the value of this property to **true** enables Analysis Services to use a faster algorithm for string hashing and comparison. For more information about setting configuration properties, see Server Properties in Analysis Services.

GB18030 Support in Analysis Services

GB18030 is a separate standard used in the People's Republic of China for encoding Chinese characters. In GB18030, characters can be 1, 2, or 4 bytes in length. In Analysis Services, there is no data conversion when processing data from external sources. The data is simply stored as Unicode. At query time, a GB18030 conversion is performed through the Analysis Services client libraries (specifically, the MSOLAP.dll OLE DB provider) when text data is returned in query results, based on the client OS settings. The database engine also supports GB18030. For details, see Collation and Unicode Support.

See Also

Globalization scenarios for Analysis Services
Globalization Tips and Best Practices (Analysis Services)
Collation and Unicode Support

Translation support in Analysis Services

1/9/2019 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Analysis Services data models, you can embed multiple translations of a caption or description to provide culture-specific strings based on the locale identifier (LCID). For multidimensional models, translations can be added for the database name, cube objects, and database dimension objects. For tabular models, you can translate table and column captions and descriptions.

Defining a translation creates the metadata and translated caption inside the model, but to render localized strings in a client application, you must either set the **Language** property on the object, or pass a **Culture** or **Locale Identifier** parameter on the connection string (for example, by setting `LocaleIdentifier=1036` to return French strings).

Plan on using **Locale Identifier** if you want to support multiple, simultaneous translations of the same object in different languages. Setting the **Language** property works, but it also impacts processing and queries, which could have unintended consequences. Setting **Locale Identifier** is the better choice because it's only used to return translated strings.

A translation consists of a locale identifier (LCID), a translated caption for the object (for example, the dimension or attribute name), and optionally, a binding to a column that provides data values in the target language. You can have multiple translations, but you can only use one for any given connection. There is no theoretical limit on the number of translations you can embed in model, but each translation adds complexity to testing, and all translations must share the same collation, so when designing your solution keep these natural constraints in mind.

> **TIP**
> You can use client applications such as Excel, SQL Server Management Studio, and SQL Server Profiler to return translated strings. See Globalization Tips and Best Practices (Analysis Services) for details.

How to add translated metadata to model in Analysis Services

Visit these links for step-by-step instructions:

- Translations in Tabular models
- Translations in Multidimensional models

See also

Globalization scenarios for Analysis Services
Languages and Collations (Analysis Services)
Set or Change the Column Collation
Globalization Tips and Best Practices (Analysis Services)

Currency Conversions (Analysis Services)

12/10/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Applies to: Multidimensional only

Analysis Services uses a combination of features, guided by Multidimensional Expressions (MDX) scripts, to provide currency conversion support in cubes supporting multiple currencies.

Currency Conversion Terminology

The following terminology is used in Analysis Services to describe currency conversion functionality:

Pivot currency
The currency against which exchange rates are entered in the rate measure group.

Local currency
The currency used to store transactions on which measures to be converted are based.

The local currency can be identified by either:

- A currency identifier in the fact table stored with the transaction, as is commonly the case with banking applications where the transaction itself identifies the currency used for that transaction.
- A currency identifier associated with an attribute in a dimension table that is then associated with a transaction in the fact table, as is commonly the case in financial applications where a location or other identifier, such as a subsidiary, identifies the currency used for an associated transaction.

Reporting currency
The currency to which transactions are converted from the pivot currency.

> **NOTE**
> For many-to-one currency conversions, the pivot currency and reporting currency are the same.

Currency dimension
A database dimension defined with the following settings:

- The **Type** property of the dimension is set to Currency.
- The **Type** property of one attribute for the dimension is set to CurrencyName.

> **IMPORTANT**
> The values of this attribute must be used in all columns that should contain a currency identifier.

Rate measure group
A measure group in a cube, defined with the following settings:

- A regular dimension relationship exists between a currency dimension and the rate measure group.
- A regular dimension relationship exists between a time dimension and the rate measure group.

- Optionally, the **Type** property is set to ExchangeRate. While the Business Intelligence Wizard uses the relationships with the currency and time dimensions to identify likely rate measure groups, setting the **Type** property to ExchangeRate allows client applications to more easily identify rate measure groups.
- One or more measures, representing the exchange rates contained by the rate measure group.

Reporting currency dimension
The dimension, defined by the Business Intelligence Wizard after a currency conversion is defined, that contains the reporting currencies for that currency conversion. The reporting currency dimension is based on a named query, defined in the data source view on which the currency dimension associated with the rate measure group is based, from the dimension main table of the currency dimension. The dimension is defined with the following settings:

- The **Type** property of the dimension is set to Currency.
- The **Type** property of the key attribute for the dimension is set to CurrencyName.
- The **Type** property of one attribute within the dimension is set to CurrencyDestination, and the column bound to the attribute contains the currency identifiers that represent the reporting currencies for the currency conversion.

Defining Currency Conversions

You can use the Business Intelligence Wizard to define currency conversion functionality for a cube, or you can manually define currency conversions using MDX scripts.

Prerequisites

Before you can define a currency conversion in a cube using the Business Intelligence Wizard, you must first define at least one currency dimension, at least one time dimension, and at least one rate measure group. From these objects, the Business Intelligence Wizard can retrieve the data and metadata used to construct the reporting currency dimension and MDX script needed to provide currency conversion functionality.

Decisions

You need to make the following decisions before the Business Intelligence Wizard can construct the reporting currency dimension and MDX script needed to provide currency conversion functionality:

- Exchange rate direction
- Converted members
- Conversion type
- Local currencies
- Reporting currencies

Exchange Rate Directions

The rate measure group contains measures representing exchange rates between local currencies and the pivot currency (commonly referred to as the corporate currency). The combination of exchange rate direction and conversion type determines the operation performed on measures to be converted by the MDX script generated using the Business Intelligence Wizard. The following table describes the operations performed depending on the exchange rate direction and conversion type, based on the exchange rate direction options and conversion directions available in the Business Intelligence Wizard.

Exchange rate direction	**Many-to-one**	**One-to-many**	**Many-to-many**

n pivot currency to 1 sample currency	Multiply the measure to be converted by the exchange rate measure for the local currency in order to convert the measure into the pivot currency.	Divide the measure to be converted by the exchange rate measure for the reporting currency in order to convert the measure into the reporting currency.	Multiply the measure to be converted by the exchange rate measure for the local currency in order to convert the measure into the pivot currency, then divide the converted measure by the exchange rate measure for the reporting currency in order to convert the measure into the reporting currency.
n sample currency to 1 pivot currency	Divide the measure to be converted by the exchange rate measure for the local currency in order to convert the measure into the pivot currency.	Multiply the measure to be converted by the exchange rate measure for the reporting currency in order to convert the measure into the reporting currency.	Divide the measure to be converted by the exchange rate measure for the local currency in order to convert the measure into the pivot currency, then multiply the converted measure by the exchange rate measure for the reporting currency in order to convert the measure into the reporting currency.

You choose the exchange rate direction on the **Set currency conversion options** page of the Business Intelligence Wizard. For more information about setting conversion direction, see Set Currency Conversion Options (Business Intelligence Wizard).

Converted Members

You can use the Business Intelligence Wizard to specify which measures from the rate measure group are used to convert values for:

- Measures in other measure groups.
- Members of an attribute hierarchy for an account attribute in a database dimension.
- Account types, used by members of an attribute hierarchy for an account attribute in a database dimension.

The Business Intelligence Wizard uses this information within the MDX script generated by the wizard to determine the scope of the currency conversion calculation. For more information about specifying members for currency conversion, see Select Members (Business Intelligence Wizard).

Conversion Types

The Business Intelligence Wizard supports three different types of currency conversion:

- **One-to-many**

  Transactions are stored in the fact table in the pivot currency, and then converted to one or more other reporting currencies.

  For example, the pivot currency can be set to United States dollars (USD), and the fact table stores transactions in USD. This conversion type converts these transactions from the pivot currency to the specified reporting currencies. The result is that transactions can be stored in the specified pivot currency and viewed either in the specified pivot currency or in any of the reporting currencies specified in the reporting currency dimension defined for the currency conversion.

- **Many-to-one**

Transactions are stored in the fact table in local currencies, and then converted into the pivot currency. The pivot currency serves as the only specified reporting currency in the reporting currency dimension.

For example, the pivot currency can be set to United States dollars (USD), and the fact table stores transactions in euros (EUR), Australian dollars (AUD), and Mexican pesos (MXN). This conversion type converts these transactions from their specified local currencies to the pivot currency. The result is that transactions can be stored in the specified local currencies and viewed in the pivot currency, which is specified in the reporting currency dimension defined for the currency conversion.

- **Many-to-many**

  Transactions are stored in the fact table in local currencies. The currency conversion functionality converts such transactions into the pivot currency, and then to one or more other reporting currencies.

  For example, the pivot currency can be set to United States dollars (USD), and the fact table stores transactions in euros (EUR), Australian dollars (AUD), and Mexican pesos (MXN). This conversion type converts these transactions from their specified local currencies to the pivot currency, and then the converted transactions are converted again from the pivot currency to the specified reporting currencies. The result is that transactions can be stored in the specified local currencies and viewed either in the specified pivot currency or in any of the reporting currencies that are specified in the reporting currency dimension defined for the currency conversion.

Specifying the conversion type allows the Business Intelligence Wizard to define the named query and dimension structure of the reporting currency dimension, as well as the structure of the MDX script defined for the currency conversion.

Local Currencies

If you choose a many-to-many or many-to-one conversion type for your currency conversion, you need to specify how to identify the local currencies from which the MDX script generated by the Business Intelligence Wizard performs the currency conversion calculations. The local currency for a transaction in a fact table can be identified in one of two ways:

- The measure group contains a regular dimension relationship to the currency dimension. For example, in the Adventure Works DW Multidimensional 2012 sample Analysis Services database, the Internet Sales measure group has a regular dimension relationship to the Currency dimension. The fact table for that measure group contains a foreign key column that references the currency identifiers in the dimension table for that dimension. In this case, you can select the attribute from the currency dimension that is referenced by the measure group to identify the local currency for transactions in the fact table for that measure group. This situation most often occurs in banking applications, where the transaction itself determines the currency used within the transaction.

- The measure group contains a referenced dimension relationship to the currency dimension, through another dimension that directly references the currency dimension. For example, in the Adventure Works DW Multidimensional 2012 sample Analysis Services database, the Financial Reporting measure group has a referenced dimension relationship to the Currency dimension through the Organization dimension. The fact table for that measure group contains a foreign key column that references members in the dimension table for the Organization dimension. The dimension table for the Organization dimension, in turn, contains a foreign key column that references the currency identifiers in the dimension table for the Currency dimension. This situation most often occurs in financial reporting applications, where the location or subsidiary for a transaction determines the currency for the transaction. In this case, you can select the attribute that references the currency dimension from the dimension for the business entity.

Reporting Currencies

If you choose a many-to-many or one-to-many conversion type for your currency conversion, you need to specify the reporting currencies for which the MDX script generated by the Business Intelligence Wizard performs the

currency conversion calculations. You can either specify all the members of the currency dimension related to the rate measure group, or select individual members from the dimension.

The Business Intelligence Wizard creates a reporting currency dimension, based on a named query constructed from the dimension table for the currency dimension using the selected reporting currencies.

> **NOTE**
> If you select the one-to-many conversion type, a reporting currency dimension is also created. The dimension contains only one member representing the pivot currency, because the pivot currency is also used as the reporting currency for a one-to-many currency conversion.

A separate reporting currency dimension is defined for each currency conversion defined in a cube. You can change the name of the reporting currency dimensions after creation, but if you do so you must also update the MDX script generated for that currency conversion to ensure that the correct name is used by the script command when referencing the reporting currency dimension.

Defining Multiple Currency Conversions

Using the Business Intelligence Wizard, you can define as many currency conversions as needed for your business intelligence solution. You can either overwrite an existing currency conversion or append a new currency conversion to the MDX script for a cube. Multiple currency conversions defined in a single cube provide flexibility in business intelligence applications that have complex reporting requirements, such as financial reporting applications that support multiple, separate conversion requirements for international reporting.

Identifying Currency Conversions

The Business Intelligence Wizard identifies each currency conversion by framing the script commands for the currency conversion in the following comments:

```
//<Currency conversion>
...
[MDX statements for the currency conversion]
...
//</Currency conversion>
```

If you change or remove these comments, the Business Intelligence Wizard is unable to detect the currency conversion, so you should not change these comments.

The wizard also stores metadata in comments within these comments, including the creation date and time, the user, and the conversion type. These comments should also not be changed because the Business Intelligence Wizard uses this metadata when displaying existing currency conversions.

You can change the script commands contained in a currency conversion as needed. If you overwrite the currency conversion, however, your changes will be lost.

See Also

Globalization scenarios for Analysis Services

Globalization Tips and Best Practices (Analysis Services)

12/10/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Applies to: Multidimensional only

These tips and guidelines can help increase the portability of your business intelligence solutions and avoid errors that are directly related to language and collation settings.

Use similar collations throughout the stack

If possible, try to use the same collation settings in Analysis Services that you use for the database engine, striving for correspondence in width-sensitivity and case-sensitivity, and access-sensitivity.

Each service has its own collation settings, with the database engine default set to SQL_Latin1_General_CP1_CI_AS and Analysis Services set to Latin1_General_AS. The defaults are compatible in term of case, width, and accent sensitivity. Be forewarned that if you vary the settings of either collation, you can run into problems when the collation properties diverge in fundamental ways.

Even when collation settings are functionally equivalent, you can run into a special case where an empty space anywhere within a string is interpreted differently by each service.

The space character is a 'special case' because it can be represented as a single-byte (SBCS) or double-byte character set (DBCS) in Unicode. In the relational engine, two compound strings separated by a space -- one using SBCS, the other in DBCS -- are considered identical. In Analysis Services, during processing, the same two compound strings are not identical, and the second instance will be flagged as a duplicate.

For more details and suggested workarounds, see Blanks in a Unicode string have different processing outcomes based on collation.

Common collation recommendations

Analysis Services always presents the full list of all available languages and collations; it does not filter the collations based on the language you selected. Be sure to choose a workable combination.

Some of the more commonly used collations include those in the following list.

You should consider this list to be a starting point for further investigation, rather than a definitive recommendation that excludes other options. You might find that a collation not specifically recommended is the one that works best for your data. Thorough testing is the only way to verify whether data values are sorted and compared appropriately. As always, be sure to run both processing and query workloads when testing collation.

- Latin1_General_100_AS is often used for applications that use the 26 characters of the ISO basic Latin alphabet.
- North European languages that include Scandinavian letters (such as ø) can use Finnish_Swedish_100.
- East European languages, such as Russian, often use Cyrillic_General_100.
- Chinese language and collations vary by region, but are generally either Chinese Simplified or Chinese Traditional.

In PRC and Singapore, Microsoft Support tends to see Simplified Chinese, with Pinyin as the preferred sort order. The recommended collations are Chinese_PRC (for SQL Server 2000), Chinese_PRC_90 (for SQL Server 2005) or Chinese_Simplified_Pinyin_100 (for SQL Server 2008 and later).

In Taiwan, it is more common to see Traditional Chinese with the recommended sort order is based on Stroke Count: Chinese_Taiwan_Stroke (for SQL Server 2000), Chinese_Taiwan_Stroke_90 (for SQL Server 2005) or Chinese_Traditional_Stroke_Count_100 (for SQL Server 2008 and later).

Other regions (such as Hong Kong and Macau) also use Traditional Chinese. For collations, in Hong Kong, it's not unusual to see Chinese_Hong_Kong_Stroke_90 (on SQL Server 2005). In Macau, Chinese_Traditional_Stroke_Count_100 (on SQL Server 2008 and later) is used fairly often.

- For Japanese, the most commonly used collation is Japanese_CI_AS. Japanese_XJIS_100 is used in installations supporting JIS2004. Japanese_BIN2 is typically seen in data migration projects, with data originating from non-Windows platforms, or from data sources other than the SQL Server relational database engine.

  Japanese_Bushu_Kakusu_100 is rarely seen in servers that run Analysis Services workloads.

- Korean_100 is recommended for Korean. Although Korean_Wansung_Unicode is still available in the list, it has been deprecated.

Case sensitivity of object identifiers

Starting in SQL Server 2012 SP2, case sensitivity of object IDs is enforced independently of the collation, but the behavior varies by language:

LANGUAGE SCRIPT	CASE SENSITIVITY
Basic Latin alphabet	Object identifiers expressed in Latin script (any of the 26 English upper or lower case letters) are treated as case-insensitive, regardless of collation. For example, the following object IDs are considered identical: 54321**abcdef**, 54321**ABCDEF**, 54321**AbCdEf**. Internally, Analysis Services treats the characters in the string as if all are uppercase, and then performs a simple byte comparison that is independent of language. Note that only the 26 characters are affected. If the language is West European, but uses Scandinavian characters, the additional character will not be upper-cased.
Cyrillic, Greek, Coptic, Armenian	Object identifiers in non-Latin bicameral script, such as Cyrillic, are always case-sensitive. For example, Измерение and измерение are considered two distinct values, even though the only difference is the case of the first letter.

Implications of case-sensitivity for object identifiers

Only object identifiers, and not object names, are subject to the casing behaviors described in the table. If you see a change in how your solution works (a before and after comparison -- after installing SQL Server 2012 SP2 or later), it will most likely be a processing issue. Queries are not impacted by object identifiers. For both query languages (DAX and MDX), the formula engine uses the object name (not the identifier).

> **NOTE**
>
> Code changes related to case-sensitivity have been a breaking change for some applications. See Breaking Changes to Analysis Services Features in SQL Server 2016 for more information.

Locale testing using Excel, SQL Server Profiler and SQL Server Management Studio

When testing translations, the connection must specify the LCID of the translation. As documented in Get Different Language from SSAS into Excel, you can use Excel to test your translations.

You can do this by hand editing the .odc file to include the locale identifier connection string property. Try this out with the Adventure Works sample multidimensional database.

- Search for existing .odc files. When you find the one for Adventure Works multidimensional, right-click the file to open it in Notepad.
- Add `Locale Identifier=1036` to the connection string. Save and close the file.
- Open Excel | **Data** | **Existing Connections**. Filter the list to just connections files on this computer. Find the connection for Adventure Works (look at the name carefully; you might have more than one). Open the connection.

  You should see the French translations from the Adventure Works sample database.

Montant Étendu	Column Labels					
Row Labels	CY 2010	CY 2011	CY 2012	CY 2013	CY 2014	Grand Total
Accessoire	$1,695.67	$46,448.30	$149,738.89	$1,050,491.71	$30,371.35	$1,278,745.91
Vélo	$496,652.84	$23,598,927.04	$28,470,924.71	$42,548,662.26		$95,115,166.84
Vêtements	$2,875.15	$137,791.42	$767,121.56	$1,215,466.43	$15,323.37	$2,138,577.94
Composant	$31,525.96	$1,643,653.12	$4,752,196.11	$5,376,916.21		$11,804,291.40
Grand Total	**$532,749.62**	**$25,426,819.87**	**$34,139,981.27**	**$50,191,536.61**	**$45,694.72**	**$110,336,782.09**

As a follow up, you can use SQL Server Profiler to confirm the locale. Click a `Session Initialize` event and then look at the property list in the text area below to find `<localeidentifier>1036</localeidentifier>`.

In Management Studio, you can specify Locale Identifier on a server connection.

- In Object Explorer | **Connect** | **Analysis Services** | **Options**, click the **Additional Connection Parameters** tab.
- Enter `Local Identifier=1036` and then click **Connect**.
- Execute an MDX query against the Adventure Works database. The query results should be the French translations.

```
select non empty [Date].[Calendar].[Calendar Year] on 0,
[Product].[Product Categories].members on 1
FROM [Adventure Works]
```

100 %

Messages | Results

	CY 2010	CY 2011	CY 2012	CY 2013
Tous les Produits	$489,328.58	$18,192,802.71	$28,193,631.53	$33,574,834.16
Accessoire	$1,695.67	$45,596.79	$145,107.49	$378,897.98
Porte-vélo	(null)	(null)	$14,905.27	$182,830.88
	(null)	(null)	$14,905.27	$182,830.88
Range-vélo	(null)	(null)	(null)	(null)
Range-vélo pour tous modèles	(null)	(null)	(null)	(null)
Bidon et porte-bidon	(null)	(null)	$645.77	$6,830.84
Porte-bidon de VTT	(null)	(null)	(null)	(null)
Porte-bidon de vélo de route	(null)	(null)	(null)	(null)

Writing MDX queries in a solution containing Translations

Translations provide display information for the names of Analysis Services objects, but the identifiers for the same objects are not translated. Whenever possible, use the identifiers and keys for Analysis Services objects instead of the translated captions and names. For example, use member keys instead of member names for Multidimensional Expressions (MDX) statements and scripts to ensure portability across multiple languages.

> **NOTE**
>
> Recall that tabular object names are always case-insensitive, regardless of collation. Multidimensional object names, on the other hand, follow the case sensitivity of the collation. Since only multidimensional object names are case-sensitive make sure that all MDX queries referencing multidimensional objects are cased correctly.

Writing MDX queries containing Date and Time Values

The following suggestions to make your date and time-based MDX queries more portable across different languages:

1. **Use numeric parts for comparisons and operations**

   When you perform month and day-of-week comparisons and operations, use the numeric date and time parts instead of the string equivalents (for example, use MonthNumberofYear instead of MonthName). Numeric values are least affected by differences in language translations.

2. **Use string equivalents in a result set**

   When building result sets seen by end-users, consider using the string (such as MonthName) so that your multi-lingual audience can benefit from the translations you've provided.

3. **Use ISO date formats for universal date and time information**

   One Analysis Services expert has this recommendation: "I always use the ISO date format yyyy-mm-dd for any date strings that I pass in to queries in SQL or MDX because it's unambiguous and will work regardless of the client or server's regional settings. I would agree that the server should defer to its regional settings when parsing an ambiguous date format, but I also think that if you've got an option that is not open to interpretation that you are better choosing that anyway".

4. **Use the Format function to enforce a specific format, regardless of regional language settings**

   The following MDX query, borrowed from a forum post, illustrates how to use Format to return dates in a specific format, irrespective of the underlying regional settings.

```
WITH MEMBER [LinkTimeAdd11Date_Manual] as Format(dateadd("d",15,"2014-12-11"), "mm/dd/yyyy")
member [LinkTimeAdd15Date_Manual] as Format(dateadd("d",11,"2014-12-13"), "mm/dd/yyyy")
SELECT
{ [LinkTimeAdd11Date_Manual]
,[LinkTimeAdd15Date_Manual]
}
ON COLUMNS
FROM [Adventure Works]
```

See Also

Globalization scenarios for Analysis Services
Write International Transact-SQL Statements

Tabular modeling (1400 compatibility level)

8/27/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

This tutorial provides lessons on how to create and deploy a tabular model at the 1400 compatibility level. If you're new to Analysis Services and tabular modeling, completing this tutorial is the quickest way to learn how to create and deploy a basic tabular model by using Visual Studio. Once you have the prerequisites in-place, it should take two to three hours to complete.

What you learn

- How to create a new tabular model project at the **1400 compatibility level** in Visual Studio with SSDT.
- How to import data from a relational database into a tabular model project workspace database.
- How to create and manage relationships between tables in the model.
- How to create calculated columns, measures, and Key Performance Indicators that help users analyze critical business metrics.
- How to create and manage perspectives and hierarchies that help users more easily browse model data by providing business and application-specific viewpoints.
- How to create partitions that divide table data into smaller logical parts that can be processed independent from other partitions.
- How to secure model objects and data by creating roles with user members.
- How to deploy a tabular model to an **Azure Analysis Services** server or **SQL Server 2017 Analysis Services** server by using SSDT.

Prerequisites

To complete this tutorial, you need:

- An Azure Analysis Services server or a SQL Server 2017 Analysis Services server in Tabular mode. Sign up for a free Azure Analysis Services trial and create a server or download a free SQL Server 2017 Developer Edition.
- An Azure SQL Data Warehouse with the **sample AdventureWorksDW database**, or an on-premises SQL Server Data Warehouse with an AdventureWorksDW sample database. When installing an AdventureWorksDW database to an on-premises SQL Server Data Warehouse, use the sample datbase version that corresponds with your server version.

  Important: If you install the sample database to an on-premises SQL Server Data Warehouse, and deploy your model to an Azure Analysis Services server, an On-premises data gateway is required.
- The latest version of SQL Server Data Tools (SSDT). Or, if you already have Visual Studio 2017, you can download and install Microsoft Analysis Services Projects (VSIX) package. For this tutorial, references to SSDT and Visual Studio are synonymous.
- The latest version of SQL Server Management Studio (SSMS).
- A client application such as Power BI Desktop or Excel.

Scenario

This tutorial is based on Adventure Works Cycles, a fictitious company. Adventure Works is a large, multinational manufacturing company that produces and distributes bicycles, parts, and accessories for commercial markets in North America, Europe, and Asia. The company employs 500 workers. Additionally, Adventure Works employs several regional sales teams throughout its market base. Your project is to create a tabular model for sales and marketing users to analyze Internet sales data in the AdventureWorksDW database.

To complete the tutorial, you must complete various lessons. In each lesson, there are tasks. Completing each task in order is necessary for completing the lesson. While in a particular lesson there may be several tasks that accomplish a similar outcome, but how you complete each task is slightly different. This method shows there is often more than one way to complete a task, and to challenge you by using skills you've learned in previous lessons and tasks.

The purpose of the lessons is to guide you through authoring a basic tabular model by using many of the features included in SSDT. Because each lesson builds upon the previous lesson, you should complete the lessons in order.

This tutorial does not provide lessons about managing a server in Azure portal, managing a server or database by using SSMS, or using a client application to browse model data.

Lessons

This tutorial includes the following lessons:

LESSON	ESTIMATED TIME TO COMPLETE
1 - Create a new tabular model project	10 minutes
2 - Get data	10 minutes
3 - Mark as Date Table	3 minutes
4 - Create relationships	10 minutes
5 - Create calculated columns	15 minutes
6 - Create measures	30 minutes
7 - Create Key Performance Indicators (KPI)	15 minutes
8 - Create perspectives	5 minutes
9 - Create hierarchies	20 minutes
10 - Create partitions	15 minutes
11 - Create roles	15 minutes
12 - Analyze in Excel	5 minutes
13 - Deploy	5 minutes

Supplemental lessons

These lessons are not required to complete the tutorial, but can be helpful in better understanding advanced tabular model authoring features.

LESSON	ESTIMATED TIME TO COMPLETE
Detail Rows	10 minutes
Dynamic security	30 minutes
Ragged hierarchies	20 minutes

Next steps

To get started, see Lesson 1: Create a new tabular model project.

Create a tabular model project

11/27/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this lesson, you use Visual Studio with SQL Server Data Tools (SSDT) or Visual Studio 2017 with Microsoft Analysis Services Projects VSIX to create a new tabular model project at the 1400 compatibility level. Once your new project is created, you can begin adding data and authoring your model. This lesson also gives you a brief introduction to the tabular model authoring environment in Visual Studio.

Estimated time to complete this lesson: **10 minutes**

Prerequisites

This article is the first lesson in a tabular model authoring tutorial. To complete this lesson, there are several prerequisites you need to have in-place. To learn more, see Analysis Services - Adventure Works tutorial.

Create a new tabular model project

To create a new tabular model project

1. In Visual Studio, on the **File** menu, click **New** > **Project**.
2. In the **New Project** dialog box, expand **Installed** > **Business Intelligence** > **Analysis Services**, and then click **Analysis Services Tabular Project**.
3. In **Name**, type **AW Internet Sales**, and then specify a location for the project files.

   By default, **Solution Name** is the same as the project name; however, you can type a different solution name.
4. Click **OK**.
5. In the **Tabular model designer** dialog box, select **Integrated workspace**.

   The workspace hosts a tabular model database with the same name as the project during model authoring. Integrated workspace means Visual Studio uses a built-in instance, eliminating the need to install a separate Analysis Services server instance just for model authoring.
6. In **Compatibility level**, select **SQL Server 2017 / Azure Analysis Services (1400)**.

If you don't see SQL Server 2017 / Azure Analysis Services (1400) in the Compatibility level listbox, you're not using the latest version of SQL Server Data Tools. To get the latest version, see Install SQL Server Data tools.

Understanding the SSDT tabular model authoring environment

Now that you've created a new tabular model project, let's take a moment to explore the tabular model authoring environment in Visual Studio.

After your project is created, it opens in Visual Studio. On the right side, in **Tabular Model Explorer**, you see a tree view of the objects in your model. Since you haven't yet imported data, the folders are empty. You can right-click an object folder to perform actions, similar to the menu bar. As you step through this tutorial, you use the Tabular Model Explorer to navigate different objects in your model project.

Click the **Solution Explorer** tab. Here, you see your **Model.bim** file. If you don't see the designer window to the left (the empty window with the Model.bim tab), in **Solution Explorer**, under **AW Internet Sales Project**, double-click the **Model.bim** file. The Model.bim file contains the metadata for your model project.

Click **Model.bim**. In the **Properties** window, you see the model properties, most important of which is the **DirectQuery Mode** property. This property specifies if the model is deployed in In-Memory mode (Off) or DirectQuery mode (On). For this tutorial, you author and deploy your model in In-Memory mode.

When you create a model project, certain model properties are set automatically according to the Data Modeling settings that can be specified in the **Tools** menu > **Options** dialog box. Data Backup, Workspace Retention, and Workspace Server properties specify how and where the workspace database (your model authoring database) is backed up, retained in-memory, and built. You can change these settings later if necessary, but for now, leave these properties as they are.

In **Solution Explorer**, right-click **AW Internet Sales** (project), and then click **Properties**. The **AW Internet Sales Property Pages** dialog box appears. You set some of these properties later when you deploy your model.

When you installed SSDT, several new menu items were added to the Visual Studio environment. Click the **Model** menu. From here, you can import data, refresh workspace data, browse your model in Excel, create perspectives and roles, select the model view, and set calculation options. Click the **Table** menu. From here, you can create and manage relationships, specify date table settings, create partitions, and edit table properties. If you click the **Column** menu, you can add and delete columns in a table, freeze columns, and specify sort order. SSDT also adds some buttons to the bar. Most useful is the AutoSum feature to create a standard aggregation measure for a selected column. Other toolbar buttons provide quick access to frequently used features and commands.

Explore some of the dialogs and locations for various features specific to authoring tabular models. While some items are not yet active, you can get a good idea of the tabular model authoring environment.

What's next?

Lesson 2: Get data.

Get data

8/27/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this lesson, you use **Get Data** to connect to the AdventureWorksDW sample database, select data, preview and filter, and then import into your model workspace.

Under the hood, Get Data is Power Query, which provides a vast array of tools for connecting to and reshaping data for modeling and analysis. To learn more, see Power Query Documentation.

> **NOTE**
> Tasks and images in this tutorial show connecting to an AdventureWorksDW2014 database on an on-premises server. In some cases, an AdventureWorksDW database on Azure SQL Data Warehouse may show different objects; however, they are fundamentally the same.

Estimated time to complete this lesson: **10 minutes**

Prerequisites

This article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 1: Create a new tabular model project.

Create a connection

To create a connection to the AdventureWorksDW database

1. In **Tabular Model Explorer**, right-click **Data Sources** > **Import from Data Source**.

   This launches **Get Data**, which guides you through connecting to a data source. If you don't see Tabular Model Explorer, in **Solution Explorer**, double-click **Model.bim** to open the model in the designer.

2. In Get Data, click **Database** > **SQL Server Database** > **Connect**.

3. In the **SQL Server Database** dialog, in **Server**, type the name of the server where you installed the AdventureWorksDW database, and then click **Connect**.

4. When prompted to enter credentials, you need to specify the credentials Analysis Services uses to connect to the data source when importing and processing data. In **Impersonation Mode**, select **Impersonate Account**, then enter credentials, and then click **Connect**. It's recommended you use an account where the password doesn't expire.

> **NOTE**
> Using a Windows user account and password provides the most secure method of connecting to a data source.

5. In Navigator, select the **AdventureWorksDW** database, and then click **OK**.This creates the connection to the database.

6. In Navigator, select the check box for the following tables: **DimCustomer**, **DimDate**, **DimGeography**, **DimProduct**, **DimProductCategory**, **DimProductSubcategory**, and **FactInternetSales**.

After you click OK, Query Editor opens. In the next section, you select only the data you want to import.

Filter the table data

Tables in the AdventureWorksDW sample database have data that isn't necessary to include in your model. When possible, you want to filter out unnecessary data to save in-memory space used by the model. You filter out some of the columns from tables so they're not imported into the workspace database, or the model database after it has been deployed.

To filter the table data before importing

1. In Query Editor, select the **DimCustomer** table. A view of the DimCustomer table at the datasource (your AdventureWorksDW sample database) appears.

2. Multi-select (Ctrl + click) **SpanishEducation**, **FrenchEducation**, **SpanishOccupation**, **FrenchOccupation**, then right-click, and then click **Remove Columns**.

Since the values for these columns are not relevant to Internet sales analysis, there is no need to import these columns. Eliminating unnecessary columns makes your model smaller and more efficient.

> **TIP**
>
> If you make a mistake, you can backup by deleting a step in **APPLIED STEPS**.

3. Filter the remaining tables by removing the following columns in each table:

DimDate

COLUMN
DateKey
SpanishDayNameOfWeek
FrenchDayNameOfWeek
SpanishMonthName
FrenchMonthName

DimGeography

COLUMN
SpanishCountryRegionName
FrenchCountryRegionName
IpAddressLocator

DimProduct

COLUMN
SpanishProductName
FrenchProductName
FrenchDescription
ChineseDescription
ArabicDescription
HebrewDescription
ThaiDescription
GermanDescription
JapaneseDescription
TurkishDescription

DimProductCategory

COLUMN
SpanishProductCategoryName
FrenchProductCategoryName

DimProductSubcategory

COLUMN
SpanishProductSubcategoryName
FrenchProductSubcategoryName

FactInternetSales

No columns removed.

Import the selected tables and column data

Now that you've previewed and filtered out unnecessary data, you can import the rest of the data you do want. The wizard imports the table data along with any relationships between tables. New tables and columns are created in the model and data that you filtered out is not be imported.

To import the selected tables and column data

1. Review your selections. If everything looks okay, click **Import**. The Data Processing dialog shows the status of data being imported from your datasource into your workspace database.

2. Click **Close**.

Save your model project

It's important to frequently save your model project.

To save the model project

- Click **File** > **Save All**.

What's next?

Lesson 3: Mark as Date Table.

Mark as Date Table

8/27/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In Lesson 2: Get data, you imported a dimension table named **DimDate**. While in your model this table is named DimDate, it can also be known as a *Date table*, in that it contains date and time data.

Whenever you use DAX time-intelligence functions, like when you create measures later, you must specify properties which include a *Date table* and a unique identifier *Date column* in that table.

In this lesson, you mark the **DimDate** table as the *Date table* and the **Date** column (in the Date table) as the *Date column* (unique identifier).

Before you mark the date table and date column, it's a good time to do a little housekeeping to make your model easier to understand. Notice in the DimDate table a column named **FullDateAlternateKey**. This column contains one row for every day in each calendar year included in the table. You use this column a lot in measure formulas and in reports. But, FullDateAlternateKey isn't really a good identifier for this column. You rename it to **Date**, making it easier to identify and include in formulas. Whenever possible, it's a good idea to rename objects like tables and columns to make them easier to identify in SSDT and client reporting applications.

Estimated time to complete this lesson: **Three minutes**

Prerequisites

This article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 2: Get data.

To rename the FullDateAlternateKey column

1. In the model designer, click the **DimDate** table.
2. Double-click the header for the **FullDateAlternateKey** column, and then rename it to **Date**.

To set Mark as Date Table

1. Select the **Date** column, and then in the **Properties** window, under **Data Type**, make sure **Date** is selected.
2. Click the **Table** menu, then click **Date**, and then click **Mark as Date Table**.
3. In the **Mark as Date Table** dialog box, in the **Date** listbox, select the **Date** column as the unique identifier. It's usually selected by default. Click **OK**.

What's next?

Lesson 4: Create relationships.

Create relationships

11/27/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this lesson, you verify the relationships that were created automatically when you imported data and add new relationships between different tables. A relationship is a connection between two tables that establishes how the data in those tables should be correlated. For example, the DimProduct table and the DimProductSubcategory table have a relationship based on the fact that each product belongs to a subcategory. To learn more, see Relationships.

Estimated time to complete this lesson: **10 minutes**

Prerequisites

This article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 3: Mark as Date Table.

Review existing relationships and add new relationships

When you imported data by using Get Data, you got seven tables from the AdventureWorksDW database. Generally, when you import data from a relational source, existing relationships are automatically imported together with the data. In order for Get Data to automatically create relationships in the data model, there must be relationships between tables at the data source.

Before you proceed with authoring your model, you should verify those relationships between tables were created properly. For this tutorial, you also add three new relationships.

To review existing relationships

1. Click the **Model** menu > **Model View** > **Diagram View**.

   The model designer now appears in Diagram View, a graphical format displaying all the tables you imported with lines between them. The lines between tables indicate the relationships that were automatically created when you imported the data.

> **NOTE**
> If you don't see any relationships between tables, it likely means there are no relationships between those tables at the datasource.

Include as many of the tables as possible by using minimap controls in the lower-right corner of the model designer. You can also click and drag tables to different locations, bringing tables closer together, or putting them in a particular order. Moving tables does not affect the relationships between the tables. To view all the columns in a particular table, click and drag on a table edge to expand or make it smaller.

2. Click the solid line between the **DimCustomer** table and the **DimGeography** table. The solid line between these two tables shows this relationship is active, that is, it is used by default when calculating DAX formulas.

   Notice the **GeographyKey** column in the **DimCustomer** table and the **GeographyKey** column in the **DimGeography** table now both each appear within a box. These columns are used in the relationship. The relationship's properties now also appear in the **Properties** window.

> **TIP**
> You can also use the Manage Relationships dialog box to show the relationships between all tables in a table format. In Tabular Model Explorer, right-click **Relationships** > **Manage Relationships**.

3. Verify the following relationships were created when each of the tables were imported from the AdventureWorksDW database:

ACTIVE	TABLE	RELATED LOOKUP TABLE
Yes	**DimCustomer [GeographyKey]**	**DimGeography [GeographyKey]**
Yes	**DimProduct [ProductSubcategoryKey]**	**DimProductSubcategory [ProductSubcategoryKey]**
Yes	**DimProductSubcategory [ProductCategoryKey]**	**DimProductCategory [ProductCategoryKey]**
Yes	**FactInternetSales [CustomerKey]**	**DimCustomer [CustomerKey]**
Yes	**FactInternetSales [ProductKey]**	**DimProduct [ProductKey]**

   If any of the relationships are missing, verify your model includes the following tables: DimCustomer, DimDate, DimGeography, DimProduct, DimProductCategory, DimProductSubcategory, and FactInternetSales. If tables from the same datasource connection are imported at separate times, any relationships between those tables are not be created and must be created manually. If no relationships appear, it means there are no relationships at the datasource. You can create them manually in the data model.

Take a closer look

In Diagram View, notice an arrow, an asterisk, and a number on the lines that show the relationship between tables.

The arrow shows the filter direction. The asterisk shows this table is the *many* side in the relationship's cardinality, and the one shows this table is the *one* side of the relationship. If you need to edit a relationship; for example, change the relationship's filter direction or cardinality, double-click the relationship line to open the Edit Relationship dialog.

These features are meant for advanced data modeling and are outside the scope of this tutorial. To learn more, see Bi-directional cross filters for tabular models in Analysis Services.

In some cases, you may need to create additional relationships between tables in your model to support certain business logic. For this tutorial, you need to create three additional relationships between the FactInternetSales table and the DimDate table.

To add new relationships between tables

1. In the model designer, in the **FactInternetSales** table, click, and hold on the **OrderDate** column, then drag the cursor to the **Date** column in the **DimDate** table, and then release.

   A solid line appears showing you have created an active relationship between the **OrderDate** column in the **Internet Sales** table, and the **Date** column in the **Date** table.

> **NOTE**
>
> When creating relationships, the cardinality and filter direction between the primary table and the related lookup table is automatically selected.

2. In the **FactInternetSales** table, click and hold on the **DueDate** column, then drag the cursor to the **Date** column in the **DimDate** table, and then release.

   A dotted line appears showing you have created an inactive relationship between the **DueDate** column in the **FactInternetSales** table, and the **Date** column in the **DimDate** table. You can have multiple relationships between tables, but only one relationship can be active at a time. Inactive relationships can be made active to perform special aggregations in custom DAX expressions.

3. Finally, create one more relationship. In the **FactInternetSales** table, click and hold on the **ShipDate** column, then drag the cursor to the **Date** column in the **DimDate** table, and then release.

What's next?

Lesson 5: Create calculated columns.

Create calculated columns

8/27/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this lesson, you create data in your model by adding calculated columns. You can create calculated columns (as custom columns) when using Get Data, by using the Query Editor, or later in the model designer like you do here. To learn more, see Calculated columns.

You create five new calculated columns in three different tables. The steps are slightly different for each task showing there are several ways to create columns, rename them, and place them in various locations in a table.

This lesson is also where you first use Data Analysis Expressions (DAX). DAX is a special language for creating highly customizable formula expressions for tabular models. In this tutorial, you use DAX to create calculated columns, measures, and role filters. To learn more, see DAX in tabular models.

Estimated time to complete this lesson: **15 minutes**

Prerequisites

This article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 4: Create relationships.

Create calculated columns

Create a MonthCalendar calculated column in the DimDate table

1. Click the **Model** menu > **Model View** > **Data View**.

   Calculated columns can only be created by using the model designer in Data View.

2. In the model designer, click the **DimDate** table (tab).

3. Right-click the **CalendarQuarter** column header, and then click **Insert Column**.

   A new column named **Calculated Column 1** is inserted to the left of the **Calendar Quarter** column.

4. In the formula bar above the table, type the following DAX formula: AutoComplete helps you type the fully qualified names of columns and tables, and lists the functions that are available.

   ```
   =RIGHT(" " & FORMAT([MonthNumberOfYear],"#0"), 2) & " - " & [EnglishMonthName]
   ```

   Values are then populated for all the rows in the calculated column. If you scroll down through the table, you see rows can have different values for this column, based on the data in each row.

5. Rename this column to **MonthCalendar**.

The MonthCalendar calculated column provides a sortable name for Month.

Create a DayOfWeek calculated column in the DimDate table

1. With the **DimDate** table still active, click the **Column** menu, and then click **Add Column**.
2. In the formula bar, type the following formula:

   ```
   =RIGHT(" " & FORMAT([DayNumberOfWeek],"#0"), 2) & " - " & [EnglishDayNameOfWeek]
   ```

   When you've finished building the formula, press ENTER. The new column is added to the far right of the table.
3. Rename the column to **DayOfWeek**.
4. Click the column heading, and then drag the column between the **EnglishDayNameOfWeek** column and the **DayNumberOfMonth** column.

   > **TIP**
   > Moving columns in your table makes it easier to navigate.

The DayOfWeek calculated column provides a sortable name for the day of week.

Create a ProductSubcategoryName calculated column in the DimProduct table

1. In the **DimProduct** table, scroll to the far right of the table. Notice the right-most column is named **Add Column** (italicized), click the column heading.
2. In the formula bar, type the following formula:

   ```
   =RELATED('DimProductSubcategory'[EnglishProductSubcategoryName])
   ```
3. Rename the column to **ProductSubcategoryName**.

The ProductSubcategoryName calculated column is used to create a hierarchy in the DimProduct table, which includes data from the EnglishProductSubcategoryName column in the DimProductSubcategory table. Hierarchies cannot span more than one table. You create hierarchies later in Lesson 9.

Create a ProductCategoryName calculated column in the DimProduct table

1. With the **DimProduct** table still active, click the **Column** menu, and then click **Add Column**.
2. In the formula bar, type the following formula:

   ```
   =RELATED('DimProductCategory'[EnglishProductCategoryName])
   ```
3. Rename the column to **ProductCategoryName**.

The ProductCategoryName calculated column is used to create a hierarchy in the DimProduct table, which includes data from the EnglishProductCategoryName column in the DimProductCategory table. Hierarchies cannot span more than one table.

Create a Margin calculated column in the FactInternetSales table

1. In the model designer, select the **FactInternetSales** table.
2. Create a new calculated column between the **SalesAmount** column and the **TaxAmt** column.
3. In the formula bar, type the following formula:

```
=[SalesAmount]-[TotalProductCost]
```

4. Rename the column to **Margin**.

	ProductCost	SalesAmount	Margin	TaxAmt	Freight	CarrierTrackingNumber	CustomerPONum
25553	$41.57	$53.99	$12.42	$4.32	$1.35		
25554	$2.97	$7.95	$4.98	$0.64	$0.20		
25555	$20.57	$54.99	$34.42	$4.40	$1.37		
25556	$9.16	$24.49	$15.33	$1.96	$0.61		
25557	$44.88	$120.00	$75.12	$9.60	$3.00		
25558	$1.87	$4.99	$3.12	$0.40	$0.12		
25559	$1.49	$3.99	$2.50	$0.32	$0.10		

The Margin calculated column is used to analyze profit margins for each sale.

What's next?

Lesson 6: Create measures.

Create measures

8/27/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this lesson, you create measures to be included in your model. Similar to the calculated columns you created, a measure is a calculation created by using a DAX formula. However, unlike calculated columns, measures are evaluated based on a user selected *filter*. For example, a particular column or slicer added to the Row Labels field in a PivotTable. A value for each cell in the filter is then calculated by the applied measure. Measures are powerful, flexible calculations that you want to include in almost all tabular models to perform dynamic calculations on numerical data. To learn more, see Measures.

To create measures, you use the *Measure Grid*. By default, each table has an empty measure grid; however, you typically do not create measures for every table. The measure grid appears below a table in the model designer when in Data View. To hide or show the measure grid for a table, click the **Table** menu, and then click **Show Measure Grid**.

You can create a measure by clicking an empty cell in the measure grid, and then typing a DAX formula in the formula bar. When you click ENTER to complete the formula, the measure then appears in the cell. You can also create measures using a standard aggregation function by clicking a column, and then clicking the AutoSum button (∑) on the toolbar. Measures created using the AutoSum feature appear in the measure grid cell directly beneath the column, but can be moved.

In this lesson, you create measures by both entering a DAX formula in the formula bar, and by using the AutoSum feature.

Estimated time to complete this lesson: **30 minutes**

Prerequisites

This article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 5: Create calculated columns.

Create measures

To create a DaysCurrentQuarterToDate measure in the DimDate table

1. In the model designer, click the **DimDate** table.
2. In the measure grid, click the top-left empty cell.
3. In the formula bar, type the following formula:

```
DaysCurrentQuarterToDate:=COUNTROWS( DATESQTD( 'DimDate'[Date]))
```

   Notice the top-left cell now contains a measure name, **DaysCurrentQuarterToDate**, followed by the result, **92**. The result is not relevant at this point because no user filter has been applied.

Unlike calculated columns, with measure formulas you can type the measure name, followed by a colon, followed by the formula expression.

To create a DaysInCurrentQuarter measure in the DimDate table

1. With the **DimDate** table still active in the model designer, in the measure grid, click the empty cell below the measure you created.

2. In the formula bar, type the following formula:

```
DaysInCurrentQuarter:=COUNTROWS( DATESBETWEEN( 'DimDate'[Date], STARTOFQUARTER(
LASTDATE('DimDate'[Date])), ENDOFQUARTER('DimDate'[Date])))
```

When creating a comparison ratio between one incomplete period and the previous period. The formula must calculate the proportion of the period that has elapsed and compare it to the same proportion in the previous period. In this case, [DaysCurrentQuarterToDate]/[DaysInCurrentQuarter] gives the proportion elapsed in the current period.

To create an InternetDistinctCountSalesOrder measure in the FactInternetSales table

1. Click the **FactInternetSales** table.

2. Click the **SalesOrderNumber** column heading.

3. On the toolbar, click the down-arrow next to the AutoSum (∑) button, and then select **DistinctCount**.

The AutoSum feature automatically creates a measure for the selected column using the DistinctCount standard aggregation formula.

4. In the measure grid, click the new measure, and then in the **Properties** window, in **Measure Name**, rename the measure to **InternetDistinctCountSalesOrder**.

To create additional measures in the FactInternetSales table

1. By using the AutoSum feature, create and name the following measures:

COLUMN	MEASURE NAME	AUTOSUM (∑)	FORMULA
SalesOrderLineNumber	InternetOrderLinesCount	Count	=COUNTA([SalesOrderLine Number])

COLUMN	MEASURE NAME	AUTOSUM (∑)	FORMULA
OrderQuantity	InternetTotalUnits	Sum	=SUM([OrderQuantity])
DiscountAmount	InternetTotalDiscountAmount	Sum	=SUM([DiscountAmount])
TotalProductCost	InternetTotalProductCost	Sum	=SUM([TotalProductCost])
SalesAmount	InternetTotalSales	Sum	=SUM([SalesAmount])
Margin	InternetTotalMargin	Sum	=SUM([Margin])
TaxAmt	InternetTotalTaxAmt	Sum	=SUM([TaxAmt])
Freight	InternetTotalFreight	Sum	=SUM([Freight])

2. By clicking an empty cell in the measure grid, and by using the formula bar, create, the following custom measures in order:

```
InternetPreviousQuarterMargin:=CALCULATE([InternetTotalMargin],PREVIOUSQUARTER('DimDate'[Date]))
```

```
InternetCurrentQuarterMargin:=TOTALQTD([InternetTotalMargin],'DimDate'[Date])
```

```
InternetPreviousQuarterMarginProportionToQTD:=[InternetPreviousQuarterMargin]*
([DaysCurrentQuarterToDate]/[DaysInCurrentQuarter])
```

```
InternetPreviousQuarterSales:=CALCULATE([InternetTotalSales],PREVIOUSQUARTER('DimDate'[Date]))
```

```
InternetCurrentQuarterSales:=TOTALQTD([InternetTotalSales],'DimDate'[Date])
```

```
InternetPreviousQuarterSalesProportionToQTD:=[InternetPreviousQuarterSales]*
([DaysCurrentQuarterToDate]/[DaysInCurrentQuarter])
```

Measures created for the FactInternetSales table can be used to analyze critical financial data such as sales, costs, and profit margin for items defined by the user selected filter.

What's next?

Lesson 7: Create Key Performance Indicators.

Create Key Performance Indicators

8/27/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this lesson, you create Key Performance Indicators (KPIs). KPIs are used to gauge performance of a value defined by a *Base* measure, against a *Target* value also defined by a measure, or by an absolute value. In reporting client applications, KPIs can provide business professionals a quick and easy way to understand a summary of business success or to identify trends. To learn more, see KPIs

Estimated time to complete this lesson: **15 minutes**

Prerequisites

This article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 6: Create measures.

Create Key Performance Indicators

To create an InternetCurrentQuarterSalesPerformance KPI

1. In the model designer, click the **FactInternetSales** table.
2. In the measure grid, click an empty cell.
3. In the formula bar, above the table, type the following formula:

   ```
   InternetCurrentQuarterSalesPerformance
   :=DIVIDE([InternetCurrentQuarterSales]/[InternetPreviousQuarterSalesProportionToQTD],BLANK())
   ```

   This measure serves as the Base measure for the KPI.
4. In the measure grid, right-click **InternetCurrentQuarterSalesPerformance** > **Create KPI**.
5. In the Key Performance Indicator (KPI) dialog box, in **Target** select **Absolute Value**, and then type **1.1**.
6. In the left (low) slider field, type **1**, and then in the right (high) slider field, type **1.07**.
7. In **Select Icon Style**, select the diamond (red), triangle (yellow), circle (green) icon type.

> **TIP**
>
> Notice the expandable **Descriptions** label below the available icon styles. Use descriptions for the various KPI elements to make them more identifiable in client applications.

8. Click **OK** to complete the KPI.

   In the measure grid, notice the icon next to the **InternetCurrentQuarterSalesPerformance** measure. This icon indicates that this measure serves as a Base value for a KPI.

To create an InternetCurrentQuarterMarginPerformance KPI

1. In the measure grid for the **FactInternetSales** table, click an empty cell.
2. In the formula bar, above the table, type the following formula:

```
InternetCurrentQuarterMarginPerformance :=IF([InternetPreviousQuarterMarginProportionToQTD]<>0,
([InternetCurrentQuarterMargin]-
[InternetPreviousQuarterMarginProportionToQTD])/[InternetPreviousQuarterMarginProportionToQTD],BLANK())
```

3. Right-click **InternetCurrentQuarterMarginPerformance** > **Create KPI**.
4. In the Key Performance Indicator (KPI) dialog box, in **Target** select **Absolute Value**, and then type **1.25**.
5. In the left (low) slider field, slide until the field displays **0.8**, and then slide the right (high) slider field, until the field displays **1.03**.
6. In **Select Icon Style**, select the diamond (red), triangle (yellow), circle (green) icon type, and then click **OK**.

What's next?

Lesson 8: Create perspectives.

Create perspectives

8/27/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this lesson, you create an Internet Sales perspective. A perspective defines a viewable subset of a model that provides focused, business-specific, or application-specific viewpoints. When a user connects to a model by using a perspective, they see only those model objects (tables, columns, measures, hierarchies, and KPIs) as fields defined in that perspective. To learn more, see Perspectives.

The Internet Sales perspective you create in this lesson excludes the DimCustomer table object. When you create a perspective that excludes certain objects from view, that object still exists in the model. However, it is not visible in a reporting client field list. Calculated columns and measures either included in a perspective or not can still calculate from object data that is excluded.

The purpose of this lesson is to describe how to create perspectives and become familiar with the tabular model authoring tools. If you later expand this model to include additional tables, you can create additional perspectives to define different viewpoints of the model, for example, Inventory and Sales.

Estimated time to complete this lesson: **Five minutes**

Prerequisites

This article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 7: Create Key Performance Indicators.

Create perspectives

To create an Internet Sales perspective

1. Click the **Model** menu > **Perspectives** > **Create and Manage**.
2. In the **Perspectives** dialog box, click **New Perspective**.
3. Double-click the **New Perspective** column heading, and then rename **Internet Sales**.
4. Select the all the tables *except* **DimCustomer**.

In a later lesson, you use the Analyze in Excel feature to test this perspective. The Excel PivotTable Fields List includes each table except the DimCustomer table.

What's next?

Lesson 9: Create hierarchies.

Create hierarchies

8/27/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this lesson, you create hierarchies. Hierarchies are groups of columns arranged in levels. For example, a Geography hierarchy might have sublevels for Country, State, County, and City. Hierarchies can appear separate from other columns in a reporting client application field list, making them easier for client users to navigate and include in a report. To learn more, see Hierarchies

To create hierarchies, use the model designer in *Diagram View*. Creating and managing hierarchies is not supported in Data View.

Estimated time to complete this lesson: **20 minutes**

Prerequisites

This article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 8: Create perspectives.

Create hierarchies

To create a Category hierarchy in the DimProduct table

1. In the model designer (diagram view), right-click the **DimProduct** table > **Create Hierarchy**. A new hierarchy appears at the bottom of the table window. Rename the hierarchy **Category**.

2. Click and drag the **ProductCategoryName** column to the new **Category** hierarchy.

3. In the **Category** hierarchy, right-click **ProductCategoryName** > **Rename**, and then type **Category**.

> **NOTE**
>
> Renaming a column in a hierarchy does not rename that column in the table. A column in a hierarchy is just a representation of the column in the table.

4. Click and drag the **ProductSubcategoryName** column to the **Category** hierarchy. Rename it **Subcategory**.

5. Right-click the **ModelName** column > **Add to hierarchy**, and then select **Category**. Rename it **Model**.

6. Finally, add **EnglishProductName** to the Category hierarchy. Rename it **Product**.

To create hierarchies in the DimDate table

1. In the **DimDate** table, create a hierarchy named **Calendar**.
2. Add the following columns in-order:
   - CalendarYear
   - CalendarSemester
   - CalendarQuarter
   - MonthCalendar
   - DayNumberOfMonth
3. In the **DimDate** table, create a **Fiscal** hierarchy. Include the following columns in-order:
   - FiscalYear
   - FiscalSemester
   - FiscalQuarter
   - MonthCalendar
   - DayNumberOfMonth
4. Finally, in the **DimDate** table, create a **ProductionCalendar** hierarchy. Include the following columns in-order:
   - CalendarYear
   - WeekNumberOfYear
   - DayNumberOfWeek

What's next?

Lesson 10: Create partitions.

Create partitions

11/27/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this lesson, you create partitions to divide the FactInternetSales table into smaller logical parts that can be processed (refreshed) independent of other partitions. By default, every table you include in your model has one partition, which includes all the table's columns and rows. For the FactInternetSales table, we want to divide the data by year; one partition for each of the table's five years. Each partition can then be processed independently. To learn more, see Partitions.

Estimated time to complete this lesson: **15 minutes**

Prerequisites

This article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 9: Create Hierarchies.

Create partitions

To create partitions in the FactInternetSales table

1. In Tabular Model Explorer, expand **Tables**, and then right-click **FactInternetSales** > **Partitions**.
2. In Partition Manager, click **Copy**, and then change the name to **FactInternetSales2010**.

   Because you want the partition to include only those rows within a certain period, for the year 2010, you must modify the query expression.
3. Click **Design** to open Query Editor, and then click the **FactInternetSales2010** query.
4. In preview, click the down arrow in the **OrderDate** column heading, and then click **Date/Time Filters** > **Between**.

5. In the Filter Rows dialog box, in **Show rows where: OrderDate**, leave **is after or equal to**, and then in the

date field, enter **1/1/2010**. Leave the **And** operator selected, then select **is before**, then in the date field, enter **1/1/2011**, and then click **OK**.

Notice in Query Editor, in APPLIED STEPS, you see another step named Filtered Rows. This filter is to select only order dates from 2010.

6. Click **Import**.

   In Partition Manager, notice the query expression now has an additional Filtered Rows clause.

```
let
    Source = #"SQL/owend_dt1;AdventureWorksDW2014",
    dbo_FactInternetSales = Source{[Schema="dbo",Item="FactInternetSales"]}[Data],
    #"Filtered Rows" = Table.SelectRows(dbo_FactInternetSales, each [OrderDate] >= #datetime(2010, 1, 1, 0, 0, 0) and
[OrderDate] < #datetime(2011, 1, 1, 0, 0, 0))
in
    #"Filtered Rows"
```

   This statement specifies this partition should include only the data in those rows where the OrderDate is in the 2010 calendar year as specified in the filtered rows clause.

To create a partition for the 2011 year

1. In the partitions list, click the **FactInternetSales2010** partition you created, and then click **Copy**. Change the partition name to **FactInternetSales2011**.

   You do not need to use Query Editor to create a new filtered rows clause. Because you created a copy of the query for 2010, all you need to do is make a slight change in the query for 2011.

2. In **Query Expression**, in-order for this partition to include only those rows for the 2011 year, replace the years in the Filtered Rows clause with **2011** and **2012**, respectively, like:

```
let
    Source = #"SQL/localhost;AdventureWorksDW2014",
    dbo_FactInternetSales = Source{[Schema="dbo",Item="FactInternetSales"]}[Data],
    #"Removed Columns" = Table.RemoveColumns(dbo_FactInternetSales,{"OrderDateKey", "DueDateKey",
"ShipDateKey"}),
    #"Filtered Rows" = Table.SelectRows(#"Removed Columns", each [OrderDate] >= #datetime(2011, 1, 1,
0, 0, 0) and [OrderDate] < #datetime(2012, 1, 1, 0, 0, 0))
in
    #"Filtered Rows"
```

To create partitions for 2012, 2013, and 2014.

- Follow the previous steps, creating partitions for 2012, 2013, and 2014, changing the years in the Filtered Rows clause to include only rows for that year.

Delete the FactInternetSales partition

Now that you have partitions for each year, you can delete the FactInternetSales partition; preventing overlap when choosing Process all when processing partitions.

To delete the FactInternetSales partition

- Click the **FactInternetSales** partition, and then click **Delete**.

Process partitions

In Partition Manager, notice the **Last Processed** column for each of the new partitions you created shows these partitions have never been processed. When you create partitions, you should run a Process Partitions or Process Table operation to refresh the data in those partitions.

To process the FactInternetSales partitions

1. Click **OK** to close Partition Manager.
2. Click the **FactInternetSales** table, then click the **Model** menu > **Process** > **Process Partitions**.
3. In the Process Partitions dialog box, verify **Mode** is set to **Process Default**.
4. Select the checkbox in the **Process** column for each of the five partitions you created, and then click **OK**.

If you're prompted for Impersonation credentials, enter the Windows user name and password you specified in Lesson 2.

The **Data Processing** dialog box appears and displays process details for each partition. Notice that a different number of rows for each partition are transferred. Each partition includes only those rows for the year specified in the WHERE clause in the SQL Statement. When processing is finished, go ahead and close the Data Processing dialog box.

What's next?

Go to the next lesson: Lesson 11: Create Roles.

Create roles

11/28/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this lesson, you create roles. Roles provide model database object and data security by limiting access to only those users that are role members. Each role is defined with a single permission: None, Read, Read and Process, Process, or Administrator. Roles can be defined during model authoring by using Role Manager. After a model has been deployed, you can manage roles by using SQL Server Management Studio (SSMS). To learn more, see Roles.

> **NOTE**
> Creating roles is not necessary to complete this tutorial. By default, the account you are currently logged in with has Administrator privileges on the model. However, for other users in your organization to browse by using a reporting client, you must create at least one role with Read permissions and add those users as members.

You create three roles:

- **Sales Manager** - This role can include users in your organization for which you want to have Read permission to all model objects and data.
- **Sales Analyst US** - This role can include users in your organization for which you want only to be able to browse data related to sales in the United States. For this role, you use a DAX formula to define a *Row Filter*, which restricts members to browse data only for the United States.
- **Administrator** - This role can include users for which you want to have Administrator permission, which allows unlimited access and permissions to perform administrative tasks on the model database.

Because Windows user and group accounts in your organization are unique, you can add accounts from your particular organization to members. However, for this tutorial, you can also leave the members blank. You test the effect of each role later in Lesson 12: Analyze in Excel.

Estimated time to complete this lesson: **15 minutes**

Prerequisites

This article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 10: Create partitions.

Create roles

To create a Sales Manager user role

1. In Tabular Model Explorer, right-click **Roles** > **Roles**.
2. In Role Manager, click **New**.
3. Click the new role, and then in the **Name** column, rename the role to **Sales Manager**.
4. In the **Permissions** column, click the dropdown list, and then select the **Read** permission.

5. Optional: Click the **Members** tab, and then click **Add**. In the **Select Users or Groups** dialog box, enter the Windows users or groups from your organization you want to include in the role.

To create a Sales Analyst US user role

1. In Role Manager, click **New**.
2. Rename the role to **Sales Analyst US**.
3. Give this role **Read** permission.
4. Click the Row Filters tab, and then for the **DimGeography** table only, in the DAX Filter column, type the following formula:

```
=DimGeography[CountryRegionCode] = "US"
```

A Row Filter formula must resolve to a Boolean (TRUE/FALSE) value. With this formula, you are specifying that only rows with the Country Region Code value of "US" are visible to the user.

5. Optional: Click the **Members** tab, and then click **Add**. In the **Select Users or Groups** dialog box, enter the Windows users or groups from your organization you want to include in the role.

To create an Administrator user role

1. Click **New**.
2. Rename the role to **Administrator**.
3. Give this role **Administrator** permission.
4. Optional: Click the **Members** tab, and then click **Add**. In the **Select Users or Groups** dialog box, enter the Windows users or groups from your organization you want to include in the role.

What's next?

Lesson 12: Analyze in Excel.

Analyze in Excel

8/27/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this lesson, you use the Analyze in Excel feature to open Microsoft Excel, automatically create a connection to the model workspace, and automatically add a PivotTable to the worksheet. The Analyze in Excel feature is meant to provide a quick and easy way to test the efficacy of your model design prior to deploying your model. You do not perform any data analysis in this lesson. The purpose of this lesson is to familiarize you, the model author, with the tools you can use to test your model design.

To complete this lesson, Excel must be installed on the same computer as Visual Studio.

Estimated time to complete this lesson: **5 minutes**

Prerequisites

This article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 11: Create roles.

Browse using the Default and Internet Sales perspectives

In these first tasks, you browse your model by using both the default perspective, which includes all model objects, and also by using the Internet Sales perspective you earlier. The Internet Sales perspective excludes the Customer table object.

To browse by using the Default perspective

1. Click the **Model** menu > **Analyze in Excel**.
2. In the **Analyze in Excel** dialog box, click **OK**.

   Excel opens with a new workbook. A data source connection is created using the current user account and the Default perspective is used to define viewable fields. A PivotTable is automatically added to the worksheet.
3. In Excel, in the **PivotTable Field List**, notice the **DimDate** and **FactInternetSales** measure groups appear. The **DimCustomer**, **DimDate**, **DimGeography**, **DimProduct**, **DimProductCategory**, **DimProductSubcategory**, and **FactInternetSales** tables with their respective columns also appear.
4. Close Excel without saving the workbook.

To browse by using the Internet Sales perspective

1. Click the **Model** menu, and then click **Analyze in Excel**.
2. In the **Analyze in Excel** dialog box, leave **Current Windows User** selected, then in the **Perspective** drop-down listbox, select **Internet Sales**, and then click **OK**.

3. In Excel, in **PivotTable Fields**, notice the DimCustomer table is excluded from the field list.

4. Close Excel without saving the workbook.

Browse by using roles

Roles are an important part of any tabular model. Without at least one role to which users are added as members, users cannot access and analyze data using your model. The Analyze in Excel feature provides a way for you to test the roles you have defined.

To browse by using the Sales Manager user role

1. In SSDT, click the **Model** menu, and then click **Analyze in Excel**.

2. In **Specify the user name or role to use to connect to the model**, select **Role**, and then in the drop-down listbox, select **Sales Manager**, and then click **OK**.

   Excel opens with a new workbook. A PivotTable is automatically created. The Pivot Table Field List includes all the data fields available in your new model.

3. Close Excel without saving the workbook.

What's next?

Go to the next lesson: Lesson 13: Deploy.

Deploy

8/27/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this lesson, you configure deployment properties; specifying a server to deploy to and a name for the model. You then deploy the model to the server. After your model is deployed, users can connect to it by using a reporting client application. To learn more, see Deploy to Azure Analysis Services and Tabular model solution deployment.

Estimated time to complete this lesson: **5 minutes**

Prerequisites

This article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 12: Analyze in Excel.

> **IMPORTANT**
>
> If deploying to Azure Analysis Services, you must have Administrator permissions on the serever.

> **IMPORTANT**
>
> If you installed the AdventureWorksDW sample database on an on-premises SQL Server, and you're deploying your model to an Azure Analysis Services server, an On-premises data gateway is required.

Deploy the model

To configure deployment properties

1. In **Solution Explorer**, right-click the **AW Internet Sales** project, and then click **Properties**.
2. In the **AW Internet Sales Property Pages** dialog box, under **Deployment Server**, in the **Server** property, enter the full server name. If connecting to Azure Analysis Services, server name must inlcude the full URL.

3. In the **Database** property, type **Adventure Works Internet Sales**.

4. In the **Model Name** property, type **Adventure Works Internet Sales Model**.

5. Verify your selections and then click **OK**.

To deploy the Adventure Works Internet Sales

1. In **Solution Explorer**, right-click the **AW Internet Sales** project > **Build**.

2. Right-click the **AW Internet Sales** project > **Deploy**.

   When deploying to Azure Analysis Services, you may be prompted to enter your account. Enter your organizational account and password, for example nancy@adventureworks.com. This account must be in Admins on the server.

   The Deploy dialog box appears and displays the deployment status of the metadata and each table included in the model.

3. When deployment successfully completes, go ahead and click **Close**.

This lesson describes the most common and easiest method to deploy a tabular model from SSDT. Advanced deployment options such as the Deployment Wizard or automating with XMLA and AMO provide greater flexibility, consistency, and scheduled deployments. To learn more, see Tabular model solution deployment.

Conclusion

Congratulations! You're finished authoring and deploying your first Analysis Services Tabular model. This tutorial has helped guide you through completing the most common tasks in creating a tabular model. Now that your Adventure Works Internet Sales model is deployed, you can use SQL Server Management Studio to manage the model; create process scripts and a backup plan. Users can also now connect to the model using a reporting client application such as Microsoft Excel or Power BI.

What's next?

Connect with Power BI Desktop
Supplemental Lesson - Dynamic security
Supplemental Lesson - Detail rows
Supplemental Lesson - Ragged hierarchies

Supplemental lesson - Detail Rows

8/27/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this supplemental lesson, you use the DAX Editor to define a custom Detail Rows Expression. A Detail Rows Expression is a property on a measure, providing end-users more information about the aggregated results of a measure.

Estimated time to complete this lesson: **10 minutes**

Prerequisites

This supplemental lesson article is part of a tabular modeling tutorial. Before performing the tasks in this supplemental lesson, you should have completed all previous lessons or have a completed Adventure Works Internet Sales sample model project.

What's the issue?

Let's look at the details of the InternetTotalSales measure, before adding a Detail Rows Expression.

1. In SSDT, click the **Model** menu > **Analyze in Excel** to open Excel and create a blank PivotTable.
2. In **PivotTable Fields**, add the **InternetTotalSales** measure from the FactInternetSales table to **Values**, **CalendarYear** from the DimDate table to **Columns**, and **EnglishCountryRegionName** to **Rows**. The PivotTable now gives an aggregated results from the InternetTotalSales measure by regions and year.

	A	B	C	D	E	F	G
1	**InternetTotalSales**	**Column Labels**					
2	**Row Labels**	**2010**	**2011**	**2012**	**2013**	**2014**	**Grand Total**
3	Australia	$20,909.78	$2,563,732.25	$2,128,407.46	$4,339,443.38	$8,507.72	$9,061,000.58
4	Canada	$3,578.27	$571,571.80	$307,604.52	$1,085,632.65	$9,457.62	$1,977,844.86
5	France	$3,399.99	$410,845.33	$648,065.54	$1,578,511.80	$3,195.06	$2,644,017.71
6	Germany		$520,500.16	$608,657.98	$1,761,876.36	$3,277.83	$2,894,312.34
7	United Kingdom	$699.10	$550,591.22	$712,700.96	$2,124,007.29	$3,713.64	$3,391,712.21
8	United States	$14,833.90	$2,458,285.17	$1,437,048.73	$5,462,078.86	$17,542.85	$9,389,789.51
9	**Grand Total**	**$43,421.04**	**$7,075,525.93**	**$5,842,485.20**	**$16,351,550.34**	**$45,694.72**	**$29,358,677.22**

3. In the PivotTable, double-click an aggregated value for a year and a region name. Here we double-clicked the value for Australia and the year 2014. A new sheet opens containing data, but not useful data.

A1 | Data returned for InternetTotalSales, Australia, 2014 (First 1000 rows).

	A	B	C	D
1	Data returned for InternetTotalSales, Australia, 2014 (First 1000 rows).			
2				
3	FactInternetSales[PromotionKey]	FactInternetSales[CurrencyKey]	FactInternetSales[SalesTerritoryKey]	FactInternetSales[SalesOrderNumber]
4	1	100	9	SO74417
5	1	100	9	SO74506
6	1	100	9	SO74653
7	1	100	9	SO74721
8	1	100	9	SO74757
9	1	100	9	SO74795
10	1	100	9	SO74812
11	1	100	9	SO74845
12	1	100	9	SO74874
13	1	100	9	SO74876
14	1	100	9	SO74938
15	1	100	9	SO74969
16	1	100	9	SO75032
17	1	100	9	SO75090

What we want to see here is a table containing columns and rows of data that contribute to the aggregated result of the InternetTotalSales measure. To do that, we can add a Detail Rows Expression as a property of the measure.

Add a Detail Rows Expression

To create a Detail Rows Expression

1. In the FactInternetSales table's measure grid, click the **InternetTotalSales** measure.

2. In **Properties** > **Detail Rows Expression**, click the editor button to open the DAX Editor.

3. In DAX Editor, enter the following expression:

```
SELECTCOLUMNS(
 FactInternetSales,
 "Sales Order Number", FactInternetSales[SalesOrderNumber],
 "Customer First Name", RELATED(DimCustomer[FirstName]),
 "Customer Last Name", RELATED(DimCustomer[LastName]),
 "City", RELATED(DimGeography[City]),
 "Order Date", FactInternetSales[OrderDate],
 "Internet Total Sales", [InternetTotalSales]
)
```

   This expression specifies names, columns, and measure results from the FactInternetSales table and related tables are returned when a user double-clicks an aggregated result in a PivotTable or report.

4. Back in Excel, delete the sheet created in Step 3, then double-click an aggregated value. This time, with a Detail Rows Expression property defined for the measure, a new sheet opens containing a lot more useful data.

J5

	A	B	C	D	E	F	G
1	Data returned for InternetTotalSales, Australia, 2014 (First 1000 rows).						
2							
3	[Sales Order Number]	[Customer First Name]	[Customer Last Name]	[City]	[Order Date]	[Internet Total Sales]	
4	SO74417	Roger	Zhang	Sydney	1/6/2014	4.99	
5	SO74506	Joy	Gutierrez	Perth	1/9/2014	4.99	
6	SO74653	Chad	Raje	Melbourne	1/14/2014	4.99	
7	SO74721	Candace	Srini	North Ryde	1/16/2014	4.99	
8	SO74757	Roger	Sun	Rockhampton	1/17/2014	4.99	
9	SO74795	Bianca	Liu	Springwood	1/18/2014	4.99	
10	SO74812	Angel	Evans	Sunbury	1/19/2014	4.99	
11	SO74845	Jamie	Zhu	Findon	1/20/2014	4.99	
12	SO74874	Erick	Suri	Coffs Harbour	1/21/2014	4.99	
13	SO74876	Nina	Raje	Townsville	1/21/2014	4.99	
14	SO74938	Kellie	Blanco	East Brisbane	1/23/2014	4.99	
15	SO74969	Byron	Navarro	Newcastle	1/24/2014	4.99	
16	SO75032	Isaac	Phillips	Sydney	1/26/2014	4.99	
17	SO75090	Vincent	Zhang	Lane Cove	1/28/2014	4.99	

5. Redeploy your model.

See also

SELECTCOLUMNS Function (DAX)

Supplemental lesson - Dynamic security

Supplemental lesson - Ragged hierarchies

Supplemental lesson - Dynamic security

11/27/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this supplemental lesson, you create an additional role that implements dynamic security. Dynamic security provides row-level security based on the user name or login id of the user currently logged on.

To implement dynamic security, you add a table to your model containing the user names of those users that can connect to the model and browse model objects and data. The model you create using this tutorial is in the context of Adventure Works; however, to complete this lesson, you must add a table containing users from your own domain. You do not need the passwords for the user names that are added. To create an EmployeeSecurity table, with a small sample of users from your own domain, you use the Paste feature, pasting employee data from an Excel spreadsheet. In a real-world scenario, the table containing user names would typically be a table from an actual database as a data source; for example, a real DimEmployee table.

To implement dynamic security, you use two DAX functions: USERNAME Function (DAX) and LOOKUPVALUE Function (DAX). These functions, applied in a row filter formula, are defined in a new role. By using the LOOKUPVALUE function, the formula specifies a value from the EmployeeSecurity table. The formula then passes that value to the USERNAME function, which specifies the user name of the user logged on belongs to this role. The user can then browse only data specified by the role's row filters. In this scenario, you specify that sales employees can only browse Internet sales data for the sales territories in which they are a member.

Those tasks that are unique to this Adventure Works tabular model scenario, but would not necessarily apply to a real-world scenario are identified as such. Each task includes additional information describing the purpose of the task.

Estimated time to complete this lesson: **30 minutes**

Prerequisites

This supplemental lesson article is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this supplemental lesson, you should have completed all previous lessons.

Add the DimSalesTerritory table to the AW Internet Sales Tabular Model Project

To implement dynamic security for this Adventure Works scenario, you must add two additional tables to your model. The first table you add is DimSalesTerritory (as Sales Territory) from the same AdventureWorksDW database. You later apply a row filter to the SalesTerritory table that defines the particular data the logged on user can browse.

To add the DimSalesTerritory table

1. In Tabular Model Explorer > **Data Sources**, right-click your connection, and then click **Import New Tables**.

   If the Impersonation Credentials dialog box appears, type the impersonation credentials you used in Lesson 2: Add Data.

2. In Navigator, select the **DimSalesTerritory** table, and then click **OK**.

3. In Query Editor, click the **DimSalesTerritory** query, and then remove **SalesTerritoryAlternateKey** column.

4. Click **Import**.

   The new table is added to the model workspace. Objects and data from the source DimSalesTerritory table are then imported into your AW Internet Sales Tabular Model.

5. After the table has been imported successfully, click **Close**.

Add a table with user name data

The DimEmployee table in the AdventureWorksDW sample database contains users from the AdventureWorks domain. Those user names do not exist in your own environment. You must create a table in your model that contains a small sample (at least three) of actual users from your organization. You then add these users as members to the new role. You do not need the passwords for the sample user names, but you do need actual Windows user names from your own domain.

To add an EmployeeSecurity table

1. Open Microsoft Excel, creating a worksheet.

2. Copy the following table, including the header row, and then paste it into the worksheet.

```
|EmployeeId|SalesTerritoryId|FirstName|LastName|LoginId|
|--------------|---------------------|-------------|------------|-----------|
|1|2|<user first name>|<user last name>|\<domain\username>|
|1|3|<user first name>|<user last name>|\<domain\username>|
|2|4|<user first name>|<user last name>|\<domain\username>|
|3|5|<user first name>|<user last name>|\<domain\username>|
```

3. Replace the first name, last name, and domain\username with the names and login ids of three users in your organization. Put the same user on the first two rows, for EmployeeId 1, showing this user belongs to more than one sales territory. Leave the EmployeeId and SalesTerritoryId fields as they are.

4. Save the worksheet as **SampleEmployee**.

5. In the worksheet, select all the cells with employee data, including the headers, then right-click the selected data, and then click **Copy**.

6. In SSDT, click the **Edit** menu, and then click **Paste**.

   If Paste is grayed out, click any column in any table in the model designer window, and try again.

7. In the **Paste Preview** dialog box, in **Table Name**, type **EmployeeSecurity**.

8. In **Data to be pasted**, verify the data includes all the user data and headers from the SampleEmployee worksheet.

9. Verify **Use first row as column headers** is checked, and then click **Ok**.

   A new table named EmployeeSecurity with employee data copied from the SampleEmployee worksheet is created.

Create relationships between FactInternetSales, DimGeography, and DimSalesTerritory table

The FactInternetSales, DimGeography, and DimSalesTerritory table all contain a common column, SalesTerritoryId. The SalesTerritoryId column in the DimSalesTerritory table contains values with a different Id for each sales territory.

To create relationships between the FactInternetSales, DimGeography, and the DimSalesTerritory table

1. In Diagram View, in the **DimGeography** table, click, and hold on the **SalesTerritoryId** column, then drag the cursor to the **SalesTerritoryId** column in the **DimSalesTerritory** table, and then release.

2. In the **FactInternetSales** table, click, and hold on the **SalesTerritoryId** column, then drag the cursor to the **SalesTerritoryId** column in the **DimSalesTerritory** table, and then release.

   Notice the Active property for this relationship is False, meaning it's inactive. The FactInternetSales table already has another active relationship.

Hide the EmployeeSecurity Table from client applications

In this task, you hide the EmployeeSecurity table, keeping it from appearing in a client application's field list. Keep in mind that hiding a table does not secure it. Users can still query EmployeeSecurity table data if they know how. To secure the EmployeeSecurity table data, preventing users from being able to query any of its data, you apply a filter in a later task.

To hide the EmployeeSecurity table from client applications

- In the model designer, in Diagram View, right-click the **Employee** table heading, and then click **Hide from Client Tools**.

Create a Sales Employees by Territory user role

In this task, you create a user role. This role includes a row filter defining which rows of the DimSalesTerritory table are visible to users. The filter is then applied in the one-to-many relationship direction to all other tables related to DimSalesTerritory. You also apply a filter that secures the entire EmployeeSecurity table from being queryable by any user that is a member of the role.

> **NOTE**
>
> The Sales Employees by Territory role you create in this lesson restricts members to browse (or query) only sales data for the sales territory to which they belong. If you add a user as a member to the Sales Employees by Territory role that also exists as a member in a role created in Lesson 11: Create Roles, you get a combination of permissions. When a user is a member of multiple roles, the permissions, and row filters defined for each role are cumulative. That is, the user has the greater permissions determined by the combination of roles.

To create a Sales Employees by Territory user role

1. In SSDT, click the **Model** menu, and then click **Roles**.

2. In **Role Manager**, click **New**.

   A new role with the None permission is added to the list.

3. Click the new role, and then in the **Name** column, rename the role to **Sales Employees by Territory**.

4. In the **Permissions** column, click the dropdown list, and then select the **Read** permission.

5. Click the **Members** tab, and then click **Add**.

6. In the **Select User or Group** dialog box, in **Enter the object named to select**, type the first sample user name you used when creating the EmployeeSecurity table. Click **Check Names** to verify the user name is valid, and then click **Ok**.

   Repeat this step, adding the other sample user names you used when creating the EmployeeSecurity table.

7. Click the **Row Filters** tab.

8. For the **EmployeeSecurity** table, in the **DAX Filter** column, type the following formula:

```
=FALSE()
```

This formula specifies that all columns resolve to the false Boolean condition. No columns for the EmployeeSecurity table can be queried by a member of the Sales Employees by Territory user role.

9. For the **DimSalesTerritory** table, type the following formula:

```
='Sales Territory'[Sales Territory Id]=LOOKUPVALUE('Employee Security'[Sales Territory Id],
  'Employee Security'[Login Id], USERNAME(),
  'Employee Security'[Sales Territory Id],
  'Sales Territory'[Sales Territory Id])
```

In this formula, the LOOKUPVALUE function returns all values for the DimEmployeeSecurity[SalesTerritoryId] column, where the EmployeeSecurity[LoginId] is the same as the current logged on Windows user name, and EmployeeSecurity[SalesTerritoryId] is the same as the DimSalesTerritory[SalesTerritoryId].

The set of sales territory IDs returned by LOOKUPVALUE is then used to restrict the rows shown in the DimSalesTerritory table. Only rows where the SalesTerritoryID for the row is in the set of IDs returned by the LOOKUPVALUE function are displayed.

10. In Role Manager, click **Ok**.

Test the Sales Employees by Territory User Role

In this task, you use the Analyze in Excel feature in SSDT to test the efficacy of the Sales Employees by Territory user role. You specify one of the user names you added to the EmployeeSecurity table and as a member of the role. This user name is then used as the effective user name in the connection created between Excel and the model.

To test the Sales Employees by Territory user role

1. In SSDT, click the **Model** menu, and then click **Analyze in Excel**.

2. In the **Analyze in Excel** dialog box, in **Specify the user name or role to use to connect to the model**, select **Other Windows User**, and then click **Browse**.

3. In the **Select User or Group** dialog box, in **Enter the object name to select**, type a user name you included in the EmployeeSecurity table, and then click **Check Names**.

4. Click **Ok** to close the **Select User or Group** dialog box, and then click **Ok** to close the **Analyze in Excel** dialog box.

   Excel opens with a new workbook. A PivotTable is automatically created. The PivotTable Fields list includes most of the data fields available in your new model.

   Notice the EmployeeSecurity table is not visible in the PivotTable Fields list. You hid this table from client tools in a previous task.

5. In the **Fields** list, in **∑ Internet Sales** (measures), select the **InternetTotalSales** measure. The measure is entered into the **Values** fields.

6. Select the **SalesTerritoryId** column from the **DimSalesTerritory** table. The column is entered into the **Row Labels** fields.

   Notice Internet sales figures appear only for the one region to which the effective user name you used belongs. If you select another column, like City from the DimGeography table as Row Label field, only cities in the sales territory to which the effective user belongs are displayed.

This user cannot browse or query any Internet sales data for territories other than the one they belong to. This restriction is because the row filter defined for the DimSalesTerritory table, in the Sales Employees by Territory user role, secures data for all data related to other sales territories.

See Also

USERNAME Function (DAX)
LOOKUPVALUE Function (DAX)
CUSTOMDATA Function (DAX)

Supplemental lesson - Ragged hierarchies

8/27/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

In this supplemental lesson, you resolve a common problem when pivoting on hierarchies that contain blank values (members) at different levels. For example, an organization where a high-level manager has both departmental managers and non-managers as direct reports. Or, geographic hierarchies composed of Country-Region-City, where some cities lack a parent State or Province, such as Washington D.C., Vatican City. When a hierarchy has blank members, it often descends to different, or ragged, levels.

Employee	ResellerTotalSales
Ken J Sánchez	**$80,450,596.98**
Brian S Welcker	**$80,450,596.98**
Amy E Alberts	$15,535,946.26
	$732,078.44
	$732,078.44
Jae B Pak	**$8,503,338.65**
	$8,503,338.65
Rachel B Valdez	**$1,790,640.23**
	$1,790,640.23
Ranjit R Varkey Chudukatil	**$4,509,888.93**
	$4,509,888.93
Stephen Y Jiang	$63,320,315.35

Tabular models at the 1400 compatibility level have an additional **Hide Members** property for hierarchies. The **Default** setting assumes there are no blank members at any level. The **Hide blank members** setting excludes blank members from the hierarchy when added to a PivotTable or report.

Estimated time to complete this lesson: **20 minutes**

Prerequisites

This supplemental lesson article is part of a tabular modeling tutorial. Before performing the tasks in this supplemental lesson, you should have completed all previous lessons or have a completed Adventure Works Internet Sales sample model project.

If you've created the AW Internet Sales project as part of the tutorial, your model does not yet contain any data or hierarchies that are ragged. To complete this supplemental lesson, you first have to create the problem by adding some additional tables, create relationships, calculated columns, a measure, and a new Organization hierarchy. That part takes about 15 minutes. Then, you get to solve it in just a few minutes.

Add tables and objects

To add new tables to your model

1. In Tabular Model Explorer, expand **Data Sources**, then right-click your connection > **Import New Tables**.
2. In Navigator, select **DimEmployee** and **FactResellerSales**, and then click **OK**.
3. In Query Editor, click **Import**
4. Create the following relationships:

TABLE 1	COLUMN	FILTER DIRECTION	TABLE 2	COLUMN	ACTIVE
FactResellerSales	OrderDateKey	Default	DimDate	Date	Yes
FactResellerSales	DueDate	Default	DimDate	Date	No
FactResellerSales	ShipDateKey	Default	DimDate	Date	No
FactResellerSales	ProductKey	Default	DimProduct	ProductKey	Yes
FactResellerSales	EmployeeKey	To Both Tables	DimEmployee	EmployeeKey	Yes

5. In the **DimEmployee** table, create the following calculated columns:

   Path

   ```
   =PATH([EmployeeKey],[ParentEmployeeKey])
   ```

   FullName

   ```
   =[FirstName] & " " & [MiddleName] & " " & [LastName]
   ```

   Level1

   ```
   =LOOKUPVALUE(DimEmployee[FullName],DimEmployee[EmployeeKey],PATHITEM([Path],1,1))
   ```

   Level2

   ```
   =LOOKUPVALUE(DimEmployee[FullName],DimEmployee[EmployeeKey],PATHITEM([Path],2,1))
   ```

   Level3

   ```
   =LOOKUPVALUE(DimEmployee[FullName],DimEmployee[EmployeeKey],PATHITEM([Path],3,1))
   ```

   Level4

   ```
   =LOOKUPVALUE(DimEmployee[FullName],DimEmployee[EmployeeKey],PATHITEM([Path],4,1))
   ```

   Level5

   ```
   =LOOKUPVALUE(DimEmployee[FullName],DimEmployee[EmployeeKey],PATHITEM([Path],5,1))
   ```

6. In the **DimEmployee** table, create a hierarchy named **Organization**. Add the following columns in-order: **Level1**, **Level2**, **Level3**, **Level4**, **Level5**.

7. In the **FactResellerSales** table, create the following measure:

```
ResellerTotalSales:=SUM([SalesAmount])
```

8. Use Analyze in Excel to open Excel and automatically create a PivotTable.

9. In **PivotTable Fields**, add the **Organization** hierarchy from the **DimEmployee** table to **Rows**, and the **ResellerTotalSales** measure from the **FactResellerSales** table to **Values**.

As you can see in the PivotTable, the hierarchy displays rows that are ragged. There are many rows where blank members are shown.

To fix the ragged hierarchy by setting the Hide members property

1. In **Tabular Model Explorer**, expand **Tables** > **DimEmployee** > **Hierarchies** > **Organization**.

2. In **Properties** > **Hide Members**, select **Hide blank members**.

3. Back in Excel, refresh the PivotTable.

Now that looks a whole lot better!

See Also

Lesson 9: Create hierarchies

Supplemental Lesson - Dynamic security

Supplemental Lesson - Detail rows

Tabular Modeling (1200 compatibility level)

12/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

This tutorial provides lessons on how to create an Analysis Services tabular model at the 1200 compatibility level by using SQL Server Data Tools (SSDT), and deploy your model to an Analysis Services server on-premises or in Azure.

If you're using SQL Server 2017 or Azure Analysis Services, and you want to create your model at the 1400 compatibility level, use the Tabular modeling (1400 compatibility level). This updated version uses the modern Get Data feature to connect and import source data, uses the M language to configure partitions, and includes additional supplemental lessons.

> **IMPORTANT**
> You should create your tabular models at the latest compatibility level supported by your server. Later compatibility level models provide improved performance, additional features, and will upgrade to future compatibility levels more seamlessly.

What you learn

- How to create a new tabular model project in SSDT.
- How to import data from a SQL Server relational database into a tabular model project.
- How to create and manage relationships between tables in the model.
- How to create and manage calculations, measures, and Key Performance Indicators that help users analyze model data.
- How to create and manage perspectives and hierarchies that help users more easily browse model data by providing business and application-specific viewpoints.
- How to create partitions dividing table data into smaller logical parts, that can be processed independent from other partitions.
- How to secure model objects and data by creating roles with user members.
- How to deploy a tabular model to an Analysis Services server on-premises or in Azure.

Scenario

This tutorial is based on Adventure Works Cycles, a fictitious company. Adventure Works is a large, multinational manufacturing company that produces bicycles, parts, and accessories for commercial markets in North America, Europe, and Asia. With headquarters in Bothell, Washington, the company employs 500 workers. Additionally, Adventure Works employs several regional sales teams throughout its market base.

To better support the data analysis needs of sales and marketing teams and of senior management, you are tasked with creating a tabular model for users to analyze Internet sales data in the AdventureWorksDW sample database.

In order to complete the tutorial, and the Adventure Works Internet Sales tabular model, you must complete a number of lessons. In each lesson is a number of tasks; completing each task in order is necessary for completing the lesson. While in a particular lesson there may be several tasks that accomplish a similar outcome, but how you

complete each task is slightly different. This is to show that there is often more than one way to complete a particular task, and to challenge you by using skills you've learned in previous tasks.

The purpose of the lessons is to guide you through authoring a basic tabular model running in In-Memory mode by using many of the features included in SSDT. Because each lesson builds upon the previous lesson, you should complete the lessons in order. Once you've completed all of the lessons, you have authored and deployed the Adventure Works Internet Sales sample tabular model on an Analysis Services server.

This tutorial does not provide lessons or information about managing a deployed tabular model database by using SQL Server Management Studio, or using a reporting client application to connect to a deployed model to browse model data.

Prerequisites

In order to complete this tutorial, you need the following prerequisites:

- The latest version of SSDT.
- The latest version of SQL Server Management Studio. Get the latest version.
- A client application such as Power BI Desktop or Excel.
- A SQL Server instance with the Adventure Works DW sample database. This sample database includes the data necessary to complete this tutorial. Get the latest version.
- An Azure Analysis Services or SQL Server 2016 or later Analysis Services instance to deploy your model to. Sign up for a free Azure Analysis Services trial.

Lessons

This tutorial includes the following lessons:

LESSON	ESTIMATED TIME TO COMPLETE
Lesson 1: Create a New Tabular Model Project	10 minutes
Lesson 2: Add Data	20 minutes
Lesson 3: Mark as Date Table	3 minutes
Lesson 4: Create Relationships	10 minutes
Lesson 5: Create Calculated Columns	15 minutes
Lesson 6: Create Measures	30 minutes
Lesson 7: Create Key Performance Indicators	15 minutes
Lesson 8: Create Perspectives	5 minutes
Lesson 9: Create Hierarchies	20 minutes
Lesson 10: Create Partitions	15 minutes
Lesson 11: Create Roles	15 minutes

LESSON	ESTIMATED TIME TO COMPLETE
Lesson 12: Analyze in Excel	20 minutes
Lesson 13: Deploy	5 minutes

Supplemental lessons

This tutorial also includes Supplemental Lessons. Topics in this section are not required to complete the tutorial, but can be helpful in better understanding advanced tabular model authoring features.

LESSON	ESTIMATED TIME TO COMPLETE
Implement Dynamic Security by Using Row Filters	30 minutes

Next step

To begin the tutorial, continue to the first lesson: Lesson 1: Create a New Tabular Model Project.

Lesson 1: Create a New Tabular Model Project

11/27/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this lesson, you will create a new, blank tabular model project in SQL Server Data Tools (SSDT). Once your new project is created, you can begin adding data by using the Table Import Wizard. This lesson also gives you a brief introduction to the tabular model authoring environment in SSDT.

Estimated time to complete this lesson: **10 minutes**

Prerequisites

This topic is the first lesson in a tabular model authoring tutorial. To complete this lesson, you must have the AdventureWorksDW sample database installed on a SQL Server instance. To learn more, see Tabular Modeling (Adventure Works Tutorial).

Create a new tabular model project

To create a new tabular model project

1. In SSDT, on the **File** menu, click **New** > **Project**.
2. In the **New Project** dialog box, expand **Installed** > **Business Intelligence** > **Analysis Services**, and then click **Analysis Services Tabular Project**.
3. In **Name**, type **AW Internet Sales**, and then specify a location for the project files.

   By default, **Solution Name** will be the same as the project name; however, you can type a different solution name.
4. Click **OK**.
5. In the **Tabular model designer** dialog box, select **Integrated workspace**.

   The workspace will host a tabular model database with the same name as the project during model authoring. Integrated workspace means SSDT will use a built-in instance, eliminating the need to install a separate Analysis Services server instance just for model authoring. To learn more, see Workspace Database.
6. In **Compatibility level**, verify **SQL Server 2016 (1200)** is selected, and then click **OK**.

If you don't see SQL Server 2016 RTM (1200) in the Compatibility level listbox, you're not using the latest version of SQL Server Data Tools. To get the latest version, see Install SQL Server Data tools.

If you're using the latest version of SSDT, you can also choose SQL Server 2017 (1400). However, to complete lesson 13: Deploy, you'll need a SQL Server 2017 or Azure server to deploy to.

Selecting an earlier compatibility level is only recommended if you intend on deploying your completed tabular model to a different Analysis Services instance running an earlier version of SQL Server. Integrated workspace is not supported for earlier compatibility levels. To learn more see, Compatibility level.

Understanding the SSDT tabular model authoring environment

Now that you've created a new tabular model project, let's take a moment to explore the tabular model authoring environment in SSDT.

After your project is created, it opens in SSDT. On the right side, in **Tabular Model Explorer**, you'll see a tree view of the objects in your model. Since you haven't yet imported data, the folders will be empty. You can right-click an object folder to perform actions, similar to the menu bar. As you step through this tutorial, you'll use the Tabular Model Explorer to navigate different objects in your model project.

Click the **Solution Explorer** tab. Here, you'll see your **Model.bim** file. If you don't see the designer window to

the left (the empty window with the Model.bim tab), in **Solution Explorer**, under **AW Internet Sales Project**, double-click the **Model.bim** file. The Model.bim file contains all of the metadata for your model project.

Let's look at the model properties. Click **Model.bim**. In the **Properties** window, you'll see the model properties, most important of which is the **DirectQuery Mode** property. This property specifies whether or not the model is deployed in In-Memory mode (Off) or DirectQuery mode (On). For this tutorial, you will author and deploy your model in In-Memory mode.

When you create a new model, certain model properties are set automatically according to the Data Modeling settings that can be specified in the **Tools** > **Options** dialog box. Data Backup, Workspace Retention, and Workspace Server properties specify how and where the workspace database (your model authoring database) is backed up, retained in-memory, and built. You can change these settings later if necessary, but for now, just leave these properties as they are.

In **Solution Explorer**, right-click **AW Internet Sales** (project), and then click **Properties**. The **AW Internet Sales Property Pages** dialog box appears. These are the advanced project properties. You will set some of these properties later when you are ready to deploy your model.

When you installed SSDT, several new menu items were added to the Visual Studio environment. Let's look at those specific to authoring tabular models. Click on the **Model** menu. From here, you can launch the Table Import Wizard, view and edit existing connections, refresh workspace data, browse your model in Excel with the Analyze in Excel feature, create perspectives and roles, select the model view, and set calculation options.

Click on the **Table** menu. Here, you can create and manage relationships between tables, create and manage, specify date table settings, create partitions, and edit table properties.

Click on the **Column** menu. Here, you can add and delete columns in a table, freeze columns, and specify sort order. You can also use the AutoSum feature to create a standard aggregation measure for a selected column. Other toolbar buttons provide quick access to frequently used features and commands.

Explore some of the dialogs and locations for various features specific to authoring tabular models. While some items will not yet be active, you can get a good idea of the tabular model authoring environment.

Additional resources

To learn more about the different types of tabular model projects, see Tabular Model Projects. To learn more about the tabular model authoring environment, see Tabular Model Designer .

What's next?

Go to the next lesson: Lesson 2: Add data.

Lesson 2: Add Data

11/27/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this lesson, you'll use the Table Import Wizard in SSDT to connect to the AdventureWorksDW SQL sample database, select data, preview and filter the data, and then import the data into your model workspace.

By using the Table Import Wizard, you can import data from a variety of relational sources: Access, SQL, Oracle, Sybase, Informix, DB2, Teradata, and more. The steps for importing data from each of these relational sources are very similar to what is described below. Data can also be selected using a stored procedure. To learn more about importing data and the different types of data sources you can import from, see Data Sources.

Estimated time to complete this lesson: **20 minutes**

Prerequisites

This topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 1: Create a New Tabular Model Project.

Create a connection

To create a connection to a the AdventureWorksDW2014 database

1. In Tabular Model Explorer, right-click **Data Sources** > **Import from Data Source**.

   This launches the Table Import Wizard, which guides you through setting up a connection to a data source. If you don't see Tabular Model Explorer, double click **Model.bim** in **Solution Explorer** to open the model in the designer.

   Note: If you're creating your model at the 1400 compatibility level, you'll see the new Get Data experience instead of the Table Import Wizard. The dialogs will appear a little different from the steps below, but you'll still be able to follow along.

2. In the Table Import Wizard, under **Relational Databases**, click **Microsoft SQL Server** > **Next**.

3. In the **Connect to a Microsoft SQL Server Database** page, in **Friendly Connection Name**, type **Adventure Works DB from SQL**.

4. In **Server name**, type the name of the server where you installed the AdventureWorksDW database.

5. In the **Database name** field, select **AdventureWorksDW**, and then click **Next**.

6. In the **Impersonation Information** page, you need to specify the credentials Analysis Services will use to connect to the data source when importing and processing data. Verify **Specific Windows user name and password** is selected, and then in **User Name** and **Password**, enter your Windows logon credentials, and then click **Next**.

> **NOTE**
>
> Using a Windows user account and password provides the most secure method of connecting to a data source. For more information, see Impersonation.

7. In the **Choose How to Import the Data** page, verify **Select from a list of tables and views to choose the data to import** is selected. You want to select from a list of tables and views, so click **Next** to display a list of all the source tables in the source database.

8. In the **Select Tables and Views** page, select the check box for the following tables: **DimCustomer**, **DimDate**, **DimGeography**, **DimProduct**, **DimProductCategory**, **DimProductSubcategory**, and **FactInternetSales**.

   DO NOT click **Finish**.

Filter the table data

The DimCustomer table that you're importing from the sample database contains a subset of the data from the original SQL Server Adventure Works database. You will filter out some more of the columns from the DimCustomer table that aren't necessary when imported into your model. When possible, you'll want to filter out data that won't be used in order to save in-memory space used by the model.

To filter the table data prior to importing

1. Select the row for the **DimCustomer** table, and then click **Preview & Filter**. The **Preview Selected Table** window opens with all the columns in the DimCustomer source table displayed.

2. Clear the checkbox at the top of the following columns: **SpanishEducation**, **FrenchEducation**, **SpanishOccupation**, **FrenchOccupation**.

	childrenAt...	EnglishEdu...	SpanishEdu...	FrenchEdu...	EnglishOccu...	SpanishOccu...	FrenchOcc
1	0	Bachelors	Licenciatura	Bac + 4	Professional	Profesional	Cadre
2	3	Bachelors	Licenciatura	Bac + 4	Professional	Profesional	Cadre
3	3	Bachelors	Licenciatura	Bac + 4	Professional	Profesional	Cadre
4	0	Bachelors	Licenciatura	Bac + 4	Professional	Profesional	Cadre
5	5	Bachelors	Licenciatura	Bac + 4	Professional	Profesional	Cadre
6	0	Bachelors	Licenciatura	Bac + 4	Professional	Profesional	Cadre
7	0	Bachelors	Licenciatura	Bac + 4	Professional	Profesional	Cadre
8	3	Bachelors	Licenciatura	Bac + 4	Professional	Profesional	Cadre
9	4	Bachelors	Licenciatura	Bac + 4	Professional	Profesional	Cadre
10	0	Bachelors	Licenciatura	Bac + 4	Professional	Profesional	Cadre
11	0	Bachelors	Licenciatura	Bac + 4	Professional	Profesional	Cadre
12	4	Bachelors	Licenciatura	Bac + 4	Professional	Profesional	Cadre
13	0	Bachelors	Licenciatura	Bac + 4	Management	Gestión	Direction
14	0	Bachelors	Licenciatura	Bac + 4	Management	Gestión	Direction
15	0	Bachelors	Licenciatura	Bac + 4	Management	Gestión	Direction

Since the values for these columns are not relevant to Internet sales analysis, there is no need to import these columns. Eliminating unnecessary columns will make your model smaller and more efficient.

3. Verify that all other columns are checked, and then click **OK**.

Notice the words **Applied filters** are now displayed in the **Filter Details** column in the **DimCustomer** row; if you click on that link you'll see a text description of the filters you just applied.

4. Filter the remaining tables by clearing the checkboxes for the following columns in each table:

DimDate

COLUMN
DateKey
SpanishDayNameOfWeek
FrenchDayNameOfWeek
SpanishMonthName
FrenchMonthName

DimGeography

COLUMN
SpanishCountryRegionName
FrenchCountryRegionName
IpAddressLocator

DimProduct

COLUMN
SpanishProductName
FrenchProductName
FrenchDescription
ChineseDescription
ArabicDescription
HebrewDescription
ThaiDescription
GermanDescription
JapaneseDescription
TurkishDescription

DimProductCategory

COLUMN
SpanishProductCategoryName
FrenchProductCategoryName

DimProductSubcategory

COLUMN
SpanishProductSubcategoryName
FrenchProductSubcategoryName

FactInternetSales

COLUMN
OrderDateKey
DueDateKey
ShipDateKey

Import the selected tables and column data

Now that you've previewed and filtered out unnecessary data, you can import the rest of the data you do want. The wizard imports the table data along with any relationships between tables. New tables and columns are created in the model and data that you filtered out will not be imported.

To import the selected tables and column data

1. Review your selections. If everything looks okay, click **Finish**.

   While importing the data, the wizard displays how many rows have been fetched. When all the data has been imported, a message indicating success is displayed.

> **TIP**
>
> To see the relationships that were automatically created between the imported tables, on the **Data preparation** row, click **Details**.

2. Click **Close**.

   The wizard closes and the model designer now shows your imported tables.

Save your model project

It's important to frequently save your model project.

To save the model project

- Click **File** > **Save All**.

What's next?

Go to the next lesson: Lesson 3: Mark as Date Table.

Lesson 3: Mark as Date Table

8/23/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In Lesson 2: Add Data, you imported a dimension table named DimDate. While in your model this table is named DimDate, it can also be known as a *Date table*, in that it contains date and time data.

Whenever you use DAX time-intelligence functions in calculations, as you'll do when you create measures a little later, you must specify date table properties, which include a *Date table* and a unique identifier *Date column* in that table.

In this lesson, you'll mark the DimDate table as the *Date table* and the Date column (in the Date table) as the *Date column* (unique identifier).

Before we mark the date table and date column, we need to do a little housekeeping to make our model easier to understand. You'll notice in the DimDate table a column named **FullDateAlternateKey**. It contains one row for every day in each calendar year included in the table. We'll be using this column a lot in measure formulas and in reports. But, FullDateAlternateKey is not really a good identifier for this column. We'll rename it to **Date**, making it easier to identify and include in formulas. Whenever possible, it's a good idea to rename objects like tables and columns to make them easier to identify in client reporting applications like Power BI and Excel.

Estimated time to complete this lesson: **3 minutes**

Prerequisites

This topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 2: Add data.

To rename the FullDateAlternateKey column

1. In the model designer, click the **DimDate** table.
2. Double click the header for the **FullDateAlternateKey** column, and then rename it to **Date**.

To set Mark as Date Table

1. Select the **Date** column, and then in the **Properties** window, under **Data Type**, make sure **Date** is selected.
2. Click the **Table** menu, then click **Date**, and then click **Mark as Date Table**.
3. In the **Mark as Date Table** dialog box, in the **Date** listbox, select the **Date** column as the unique identifier. It will usually be selected by default. Click **OK**.

What's next?

Go to the next lesson: Lesson 4: Create Relationships.

Lesson 4: Create Relationships

11/27/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this lesson, you will verify the relationships that were created automatically when you imported data and add new relationships between different tables. A relationship is a connection between two tables that establishes how the data in those tables should be correlated. For example, the DimProduct table and the DimProductSubcategory table have a relationship based on the fact that each product belongs to a subcategory. To learn more, see Relationships.

Estimated time to complete this lesson: **10 minutes**

Prerequisites

This topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 3: Mark as Date Table.

Review existing relationships and add new relationships

When you imported data by using the Table Import Wizard, you got seven tables from the AdventureWorksDW database. Generally, when you import data from a relational source, existing relationships are automatically imported together with the data. However, before you proceed with authoring your model you should verify those relationships between tables were created properly. For this tutorial, you will also add three new relationships.

To review existing relationships

1. Click the **Model** menu > **Model View** > **Diagram View**.

   The model designer now appears in Diagram View, a graphical format displaying all of the tables you imported with lines between them. The lines between tables indicate the relationships that were automatically created when you imported the data.

   Use the minimap controls in the lower-right corner of the model designer to adjust the view to include as many of the tables as possible. You can also click, and drag tables to different locations, bringing tables closer together, or putting them in a particular order. Moving tables does not affect the relationships already

between the tables. To view all of the columns in a particular table, click, and drag on a table edge to expand or make it smaller.

2. Click the solid line between the **DimCustomer** table and the **DimGeography** table. The solid line between these two tables show this relationship is active, that is, it is used by default when calculating DAX formulas.

   Notice the **GeographyKey** column in the **DimCustomer** table and the **GeographyKey** column in the **DimGeography** table now both each appear within a box. This show these are the columns used in the relationship. The relationship's properties now also appear in the **Properties** window.

   > **TIP**
   > In addition to using the model designer in diagram view, you can also use the Manage Relationships dialog box to show the relationships between all tables in a table format. Right-click **Relationships** in Tabular Model Explorer, and then click **Manage Relationships**. The Manage Relationships dialog box show the relationships that were automatically created when you imported data.

3. Use the model designer in diagram view, or the Manage Relationships dialog box, to verify the following relationships were created when each of the tables were imported from the AdventureWorksDW database:

ACTIVE	TABLE	RELATED LOOKUP TABLE
Yes	**DimCustomer [GeographyKey]**	**DimGeography [GeographyKey]**
Yes	**DimProduct [ProductSubcategoryKey]**	**DimProductSubcategory [ProductSubcategoryKey]**
Yes	**DimProductSubcategory [ProductCategoryKey]**	**DimProductCategory [ProductCategoryKey]**
Yes	**FactInternetSales [CustomerKey]**	**DimCustomer [CustomerKey]**
Yes	**FactInternetSales [ProductKey]**	**DimProduct [ProductKey]**

   If any of the relationships in the table above are missing, verify that your model includes the following tables: DimCustomer, DimDate, DimGeography, DimProduct, DimProductCategory, DimProductSubcategory, and FactInternetSales. If tables from the same data source connection are imported at separate times, any relationships between those tables will not be created and must be created manually.

Take a closer look

In Diagram View, you'll notice an arrow, an asterisk, and a number on the lines that show the relationship between tables.

The arrow shows the filter direction, the asterisk shows this table is the many side in the relationship's cardinality, and the 1 shows this table is the one side of the relationship. If you need to edit a relationship; for example, change the relationship's filter direction or cardinality, double-click the relationship line in Diagram View to open the Edit Relationship dialog.

Most likely, you will never need to edit a relationship. These features are meant for advanced data modeling and are outside the scope of this tutorial. To learn more, see Bi-directional cross filters for tabular models in SQL Server 2016 Analysis Services.

In some cases, you may need to create additional relationships between tables in your model to support certain business logic. For this tutorial, you need to create three additional relationships between the FactInternetSales table and the DimDate table.

To add new relationships between tables

1. In the model designer, in the **FactInternetSales** table, click, and hold on the **OrderDate** column, then drag the cursor to the **Date** column in the **DimDate** table, and then release.

   A solid line appears showing you have created an active relationship between the **OrderDate** column in the **Internet Sales** table and the **Date** column in the **Date** table.

> **NOTE**
>
> When creating relationships, the cardinality and filter direction between the primary table and the related lookup table is automatically selected.

2. In the **FactInternetSales** table, click, and hold on the **DueDate** column, then drag the cursor to the **Date** column in the **DimDate** table, and then release.

   A dotted line appears showing you have created an inactive relationship between the **DueDate** column in the **FactInternetSales** table and the **Date** column in the **DimDate** table. You can have multiple relationships between tables, but only one relationship can be active at a time.

3. Finally, create one more relationship; in the **FactInternetSales** table, click, and hold on the **ShipDate** column, then drag the cursor to the **Date** column in the **DimDate** table, and then release.

What's next?

Go to the next lesson: Lesson 5: Create Calculated Columns.

Lesson 5: Create Calculated Columns

8/23/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this lesson, you will create new data in your model by adding calculated columns. A calculated column is based on data that already exists in the model. To learn more, see Calculated Columns.

You will create five new calculated columns in three different tables. The steps are slightly different for each task. This is to show you there are several ways to create new columns, rename them, and place them in various locations in a table.

Estimated time to complete this lesson: **15 minutes**

Prerequisites

This topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 4: Create Relationships.

Create calculated columns

Create a MonthCalendar calculated column in the DimDate table

1. Click the **Model** menu > **Model View** > **Data View**.

   Calculated columns can only be created by using the model designer in Data View.

2. In the model designer, click the **DimDate** table (tab).

3. Right-click the **CalendarQuarter** column header, and then click **Insert Column**.

   A new column named **Calculated Column 1** is inserted to the left of the **Calendar Quarter** column.

4. In the formula bar above the table, type the following formula. AutoComplete helps you type the fully qualified names of columns and tables, and lists the functions that are available.

   ```
   =RIGHT(" " & FORMAT([MonthNumberOfYear],"#0"), 2) & " - " & [EnglishMonthName]
   ```

   Values are then populated for all the rows in the calculated column. If you scroll down through the table, you will see that rows can have different values for this column, based on the data that is in each row.

5. Rename this column to **MonthCalendar**.

The MonthCalendar calculated column provides a sortable name for Month.

Create a DayOfWeek calculated column in the DimDate table

1. With the **DimDate** table still active, click on the **Column** menu, and then click **Add Column**.
2. In the formula bar, type the following formula:

```
=RIGHT(" " & FORMAT([DayNumberOfWeek],"#0"), 2) & " - " & [EnglishDayNameOfWeek]
```

   When you've finished building the formula, press ENTER. The new column is added to the far right of the table.

3. Rename the column to **DayOfWeek**.
4. Click on the column heading, and then drag the column between the **EnglishDayNameOfWeek** column and the **DayNumberOfMonth** column.

> **TIP**
> Moving columns in your table makes it easier to navigate.

The DayOfWeek calculated column provides a sortable name for the day of week.

Create a ProductSubcategoryName calculated column in the DimProduct table

1. In the **DimProduct** table, scroll to the far right of the table. Notice the right-most column is named **Add Column** (italicized), click the column heading.
2. In the formula bar, type the following formula.

```
=RELATED('DimProductSubcategory'[EnglishProductSubcategoryName])
```

3. Rename the column to **ProductSubcategoryName**.

The ProductSubcategoryName calculated column is used to create a hierarchy in the DimProduct table which includes data from the EnglishProductSubcategoryName column in the DimProductSubcategory table. Hierarchies cannot span more than one table. You will create hierarchies later in Lesson 9.

Create a ProductCategoryName calculated column in the DimProduct table

1. With the **DimProduct** table still active, click the **Column** menu, and then click **Add Column**.
2. In the formula bar, type the following formula:

```
=RELATED('DimProductCategory'[EnglishProductCategoryName])
```

3. Rename the column to **ProductCategoryName**.

The ProductCategoryName calculated column is used to create a hierarchy in the DimProduct table which includes data from the EnglishProductCategoryName column in the DimProductCategory table. Hierarchies cannot span more than one table.

Create a Margin calculated column in the FactInternetSales table

1. In the model designer, select the **FactInternetSales** table.
2. Add a new column.
3. In the formula bar, type the following formula:

```
=[SalesAmount]-[TotalProductCost]
```

4. Rename the column to **Margin**.

5. Drag the column between the **SalesAmount** column and the **TaxAmt** column.

	ProductCost	SalesAmount	Margin	TaxAmt	Freight	CarrierTrackingNumber	CustomerPONum
25553	$41.57	$53.99	$12.42	$4.32	$1.35		
25554	$2.97	$7.95	$4.98	$0.64	$0.20		
25555	$20.57	$54.99	$34.42	$4.40	$1.37		
25556	$9.16	$24.49	$15.33	$1.96	$0.61		
25557	$44.88	$120.00	$75.12	$9.60	$3.00		
25558	$1.87	$4.99	$3.12	$0.40	$0.12		
25559	$1.49	$3.99	$2.50	$0.32	$0.10		

The Margin calculated column is used to analyze profit margins for each sale.

What's next?

Go to the next lesson: Lesson 6: Create Measures.

Lesson 6: Create Measures

8/23/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this lesson, you will create measures to be included in your model. Similar to the calculated columns you created in the previous lesson, a measure is a calculation created by using a DAX formula. However, unlike calculated columns, measures are evaluated based on a user selected *filter*; for example, a particular column or slicer added to the Row Labels field in a PivotTable. A value for each cell in the filter is then calculated by the applied measure. Measures are powerful, flexible calculations that you will want to include in almost all tabular models to perform dynamic calculations on numerical data. To learn more, see Measures.

To create measures, you will use the *Measure Grid*. By default, each table has an empty measure grid; however, you typically will not create measures for every table. The measure grid appears below a table in the model designer when in Data View. To hide or show the measure grid for a table, click the **Table** menu, and then click **Show Measure Grid**.

You can create a measure by clicking on an empty cell in the measure grid, and then typing a DAX formula in the formula bar. When you click ENTER to complete the formula, the measure will then appear in the cell. You can also create measures using a standard aggregation function by clicking on a column, and then clicking on the AutoSum button (**∑**) on the toolbar. Measures created using the AutoSum feature will appear in the measure grid cell directly beneath the column, but can be moved.

In this lesson, you will create measures by both entering a DAX formula in the formula bar and by using the AutoSum feature.

Estimated time to complete this lesson: **30 minutes**

Prerequisites

This topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 5: Create Calculated Columns.

Create measures

To create a DaysCurrentQuarterToDate measure in the DimDate table

1. In the model designer, click the **DimDate** table.
2. In the measure grid, click the top-left empty cell.
3. In the formula bar, type the following formula:

   ```
   DaysCurrentQuarterToDate:=COUNTROWS( DATESQTD( 'DimDate'[Date]))
   ```

   Notice the top-left cell now contains a measure name, **DaysCurrentQuarterToDate**, followed by the result, **92**.

Unlike calculated columns, with measure formulas you can type the measure name, followed by a comma, followed by the formula expression.

To create a DaysInCurrentQuarter measure in the DimDate table

1. With the **DimDate** table still active in the model designer, in the measure grid, click the empty cell below the measure you just created.

2. In the formula bar, type the following formula:

```
DaysInCurrentQuarter:=COUNTROWS( DATESBETWEEN( 'DimDate'[Date], STARTOFQUARTER(
LASTDATE('DimDate'[Date])), ENDOFQUARTER('DimDate'[Date])))
```

When creating a comparison ratio between one incomplete period and the previous period; the formula must take into account the proportion of the period that has elapsed, and compare it to the same proportion in the previous period. In this case, [DaysCurrentQuarterToDate]/[DaysInCurrentQuarter] gives the proportion elapsed in the current period.

To create an InternetDistinctCountSalesOrder measure in the FactInternetSales table

1. Click the **FactInternetSales** table.

2. Click on the **SalesOrderNumber** column heading.

3. On the toolbar, click the down-arrow next to the AutoSum (∑) button, and then select **DistinctCount**.

The AutoSum feature automatically creates a measure for the selected column using the DistinctCount standard aggregation formula.

4. In the measure grid, click the new measure, and then in the **Properties** window, in **Measure Name**, rename the measure to **InternetDistinctCountSalesOrder**.

To create additional measures in the FactInternetSales table

1. By using the AutoSum feature, create and name the following measures:

MEASURE NAME	COLUMN	AUTOSUM (∑)	FORMULA
InternetOrderLinesCount	SalesOrderLineNumber	Count	=COUNTA([SalesOrderLine Number])

MEASURE NAME	COLUMN	AUTOSUM (∑)	FORMULA
InternetTotalUnits	OrderQuantity	Sum	=SUM([OrderQuantity])
InternetTotalDiscountAmount	DiscountAmount	Sum	=SUM([DiscountAmount])
InternetTotalProductCost	TotalProductCost	Sum	=SUM([TotalProductCost])
InternetTotalSales	SalesAmount	Sum	=SUM([SalesAmount])
InternetTotalMargin	Margin	Sum	=SUM([Margin])
InternetTotalTaxAmt	TaxAmt	Sum	=SUM([TaxAmt])
InternetTotalFreight	Freight	Sum	=SUM([Freight])

2. By clicking on an empty cell in the measure grid, and by using the formula bar, create and name the following measures in order:

```
InternetPreviousQuarterMargin:=CALCULATE([InternetTotalMargin],PREVIOUSQUARTER('DimDate'[Date]))
```

```
InternetCurrentQuarterMargin:=TOTALQTD([InternetTotalMargin],'DimDate'[Date])
```

```
InternetPreviousQuarterMarginProportionToQTD:=[InternetPreviousQuarterMargin]*
([DaysCurrentQuarterToDate]/[DaysInCurrentQuarter])
```

```
InternetPreviousQuarterSales:=CALCULATE([InternetTotalSales],PREVIOUSQUARTER('DimDate'[Date]))
```

```
InternetCurrentQuarterSales:=TOTALQTD([InternetTotalSales],'DimDate'[Date])
```

```
InternetPreviousQuarterSalesProportionToQTD:=[InternetPreviousQuarterSales]*
([DaysCurrentQuarterToDate]/[DaysInCurrentQuarter])
```

Measures created for the FactInternetSales table can be used to analyze critical financial data such as sales, costs, and profit margin for items defined by the user selected filter.

What's next?

Go to the next lesson: Lesson 7: Create Key Performance Indicators.

Lesson 7: Create Key Performance Indicators

8/23/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this lesson, you will create Key Performance Indicators (KPIs). KPIs are used to gauge performance of a value, defined by a *Base* measure, against a *Target* value, also defined by a measure or by an absolute value. In reporting client applications, KPIs can provide business professionals a quick and easy way to understand a summary of business success or to identify trends. To learn more, see KPIs.

Estimated time to complete this lesson: **15 minutes**

Prerequisites

This topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 6: Create Measures.

Create Key Performance Indicators

To create an InternetCurrentQuarterSalesPerformance KPI

1. In the model designer, click the **FactInternetSales** table (tab).
2. In the measure grid, click an empty cell.
3. In the formula bar, above the table, type the following formula:

   ```
   InternetCurrentQuarterSalesPerformance
   :=IFERROR([InternetCurrentQuarterSales]/[InternetPreviousQuarterSalesProportionToQTD],BLANK())
   ```

   This measure will serve as the Base measure for the KPI.
4. Right-click **InternetCurrentQuarterSalesPerformance** > **Create KPI**.
5. In the Key Performance Indicator (KPI) dialog box, in **Target** select **Absolute Value**, and then type **1.1**.
6. In the left (low) slider field, type **1**, and then in the right (high) slider field, type **1.07**.
7. In **Select Icon Style**, select the diamond (red), triangle (yellow), circle (green) icon type.

> **TIP**
>
> Notice the expandable **Descriptions** label below the available icon styles. Use this to enter descriptions for the various KPI elements to make them more identifiable in client applications.

8. Click **OK** to complete the KPI.

   In the measure grid, notice the icon next to the **InternetCurrentQuarterSalesPerformance** measure. This icon indicates that this measure serves as a Base value for a KPI.

To create an InternetCurrentQuarterMarginPerformance KPI

1. In the measure grid for the **FactInternetSales** table, click an empty cell.
2. In the formula bar, above the table, type the following formula:

```
InternetCurrentQuarterMarginPerformance :=IF([InternetPreviousQuarterMarginProportionToQTD]<>0,
([InternetCurrentQuarterMargin]-
[InternetPreviousQuarterMarginProportionToQTD])/[InternetPreviousQuarterMarginProportionToQTD],BLANK())
```

3. Right-click **InternetCurrentQuarterMarginPerformance** > **Create KPI**.
4. In the Key Performance Indicator (KPI) dialog box, in **Target** select **Absolute Value**, and then type **1.25**.
5. In **Define Status Thresholds**, slide the left (low) slider field until the field displays **0.8**, and then slide the right (high) slider field, until the field displays **1.03**.
6. In **Select Icon Style**, select the diamond (red), triangle (yellow), circle (green) icon type, and then click **OK**.

What's next?

Go to the next lesson: Lesson 8: Create Perspectives.

Lesson 8: Create Perspectives

8/23/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this lesson, you will create an Internet Sales perspective. A perspective defines a viewable subset of a model that provides focused, business-specific, or application-specific viewpoints. When a user connects to a model by using a perspective, they see only those model objects (tables, columns, measures, hierarchies, and KPIs) as fields defined in that perspective.

The Internet Sales perspective you create in this lesson will exclude the DimCustomer table object. When you create a perspective that excludes certain objects from view, that object still exists in the model; however, it is not visible in a reporting client field list. Calculated columns and measures either included in a perspective or not can still calculate from object data that is excluded.

The purpose of this lesson is to describe how to create perspectives and become familiar with the tabular model authoring tools. If you later expand this model to include additional tables, you can create additional perspectives to define different viewpoints of the model, for example, Inventory and Sales. To learn more, see Perspectives.

Estimated time to complete this lesson: **5 minutes**

Prerequisites

This topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 7: Create Key Performance Indicators.

Create perspectives

To create an Internet Sales perspective

1. Click the **Model** menu > **Perspectives** > **Create and Manage**.
2. In the **Perspectives** dialog box, click **New Perspective**.
3. Double-click the **New Perspective** column heading, and then rename **Internet Sales**.
4. Select the all of the tables *except* **DimCustomer**.

In a later lesson, you will use the Analyze in Excel feature to test this perspective. The Excel PivotTable Fields List will include each table except the DimCustomer table.

What's next?

Go to the next lesson: Lesson 9: Create Hierarchies.

Lesson 9: Create Hierarchies

8/23/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this lesson, you will create hierarchies. Hierarchies are groups of columns arranged in levels; for example, a Geography hierarchy might have sub-levels for Country, State, County, and City. Hierarchies can appear separate from other columns in a reporting client application field list, making them easier for client users to navigate and include in a report. To learn more, see Hierarchies.

To create hierarchies, you'll use the model designer in *Diagram View*. Creating and managing hierarchies is not supported in Data View.

Estimated time to complete this lesson: **20 minutes**

Prerequisites

This topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 8: Create Perspectives.

Create hierarchies

To create a Category hierarchy in the DimProduct table

1. In the model designer (diagram view), right-click the **DimProduct** table > **Create Hierarchy**. A new hierarchy appears at the bottom of the table window. Rename the hierarchy **Category**.

2. Click and drag the **ProductCategoryName** column to the new **Category** hierarchy.

3. In the **Category** hierarchy, right-click the **ProductCategoryName** > **Rename**, and then type **Category**.

> **NOTE**
>
> Renaming a column in a hierarchy does not rename that column in the table. A column in a hierarchy is just a representation of the column in the table.

4. Click and drag the **ProductSubcategoryName** column to the **Category** hierarchy. Rename it **Subcategory**.

5. Right-click the **ModelName** column > **Add to hierarchy**, and then select **Category**. Do the same for **EnglishProductName**. Rename these columns in the hierarchy **Model** and **Product**.

To create hierarchies in the DimDate table

1. In the **DimDate** table, create a new hierarchy named **Calendar**.
2. Add the following columns in-order:
   - CalendarYear
   - CalendarSemester
   - CalendarQuarter
   - MonthCalendar
   - DayNumberOfMonth
3. In the **DimDate** table, create a **Fiscal** hierarchy. Include the following columns:
   - FiscalYear
   - FiscalSemester
   - FiscalQuarter
   - MonthCalendar
   - DayNumberOfMonth
4. Finally, in the **DimDate** table, create a **ProductionCalendar** hierarchy. Include the following columns:
   - CalendarYear
   - WeekNumberOfYear
   - DayNumberOfWeek

What's next?

Go to the next lesson: Lesson 10: Create Partitions.

Lesson 10: Create Partitions

11/27/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this lesson, you will create partitions to divide the FactInternetSales table into smaller logical parts that can be processed (refreshed) independent of other partitions. By default, every table you include in your model has one partition which includes all of the table's columns and rows. For the FactInternetSales table, we want to divide the data by year; one partition for each of the table's five years. Each partition can then be processed independently. To learn more, see Partitions.

Estimated time to complete this lesson: **15 minutes**

Prerequisites

This topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 9: Create Hierarchies.

Create partitions

To create partitions in the FactInternetSales table

1. In Tabular Model Explorer, expand **Tables**, right-click **FactInternetSales** > **Partitions**.
2. In the Partition Manager dialog box, click **Copy**.
3. In **Partition Name**, change the name to **FactInternetSales2010**.

   > **TIP**
   > Notice the column names in the Table Preview window display those columns included in the model table (checked) with the column names from the source. This is because the Table Preview window displays columns from the source table, not from the model table.

4. Select the **SQL** button just above the right side of the preview window to open the SQL Statement editor.

   Because you want the partition to include only those rows within a certain period, you must include a WHERE clause. You can only create a WHERE clause by using a SQL Statement.

5. In the **SQL Statement** field, replace the existing statement by copying and pasting the following statement:

```
SELECT
[dbo].[FactInternetSales].[ProductKey],
[dbo].[FactInternetSales].[CustomerKey],
[dbo].[FactInternetSales].[PromotionKey],
[dbo].[FactInternetSales].[CurrencyKey],
[dbo].[FactInternetSales].[SalesTerritoryKey],
[dbo].[FactInternetSales].[SalesOrderNumber],
[dbo].[FactInternetSales].[SalesOrderLineNumber],
[dbo].[FactInternetSales].[RevisionNumber],
[dbo].[FactInternetSales].[OrderQuantity],
[dbo].[FactInternetSales].[UnitPrice],
[dbo].[FactInternetSales].[ExtendedAmount],
[dbo].[FactInternetSales].[UnitPriceDiscountPct],
[dbo].[FactInternetSales].[DiscountAmount],
[dbo].[FactInternetSales].[ProductStandardCost],
[dbo].[FactInternetSales].[TotalProductCost],
[dbo].[FactInternetSales].[SalesAmount],
[dbo].[FactInternetSales].[TaxAmt],
[dbo].[FactInternetSales].[Freight],
[dbo].[FactInternetSales].[CarrierTrackingNumber],
[dbo].[FactInternetSales].[CustomerPONumber],
[dbo].[FactInternetSales].[OrderDate],
[dbo].[FactInternetSales].[DueDate],
[dbo].[FactInternetSales].[ShipDate]
FROM [dbo].[FactInternetSales]
WHERE (([OrderDate] >= N'2010-01-01 00:00:00') AND ([OrderDate] < N'2011-01-01 00:00:00'))
```

This statement specifies the partition should include all of the data in those rows where the OrderDate is for the 2010 calendar year as specified in the WHERE clause.

6. Click **Validate**.

To create a partition for the 2011 year

1. In the partitions list, click the **FactInternetSales2010** partition you just created, and then click **Copy**.
2. In **Partition Name**, type **FactInternetSales2011**.
3. In the SQL Statement, in-order for the partition to include only those rows for the 2011 year, replace the WHERE clause with the following:

```
WHERE (([OrderDate] >= N'2011-01-01 00:00:00') AND ([OrderDate] < N'2012-01-01 00:00:00'))
```

To create a partition for the 2012 year

- Follow the steps above, using the following WHERE clause.

```
WHERE (([OrderDate] >= N'2012-01-01 00:00:00') AND ([OrderDate] < N'2013-01-01 00:00:00'))
```

To create a partition for the 2013 year

- Follow the steps above, using the following WHERE clause.

```
WHERE (([OrderDate] >= N'2013-01-01 00:00:00') AND ([OrderDate] < N'2014-01-01 00:00:00'))
```

To create a partition for the 2014 year

- Follow the steps above, using the following WHERE clause.

```
WHERE (([OrderDate] >= N'2014-01-01 00:00:00') AND ([OrderDate] < N'2015-01-01 00:00:00'))
```

Delete the FactInternetSales partition

Now that you have partitions for each year, you can delete the FactInternetSales partition. This prevents overlap when choosing Process all when processing partitions.

To delete the FactInternetSales partition

- Click the FactInternetSales partition, and then click **Delete**.

Process partitions

In Partition Manager, notice the **Last Processed** column for each of the new partitions you just created shows these partitions have never been processed. When you create new partitions, you should run a Process Partitions or Process Table operation to refresh the data in those partitions.

To process the FactInternetSales partitions

1. Click **OK** to close the Partition Manager dialog box.
2. Click the **FactInternetSales** table, then click the **Model** menu > **Process** > **Process Partitions**.
3. In the Process Partitions dialog box, verify **Mode** is set to **Process Default**.
4. Select the checkbox in the **Process** column for each of the five partitions you created, and then click **OK**.

If you're prompted for Impersonation credentials, enter the Windows user name and password you specified in Lesson 2.

The **Data Processing** dialog box appears and displays process details for each partition. Notice that a different number of rows for each partition are transferred. This is because each partition includes only those rows for the year specified in the WHERE clause in the SQL Statement. When processing is finished, go ahead and close the Data Processing dialog box.

What's next?

Go to the next lesson: Lesson 11: Create Roles.

Lesson 11: Create Roles

11/28/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this lesson, you will create roles. Roles provide model database object and data security by limiting access to only those Windows users which are role members. Each role is defined with a single permission: None, Read, Read and Process, Process, or Administrator. Roles can be defined during model authoring by using Role Manager. After a model has been deployed, you can manage roles by using SQL Server Management Studio. To learn more, see Roles.

> **NOTE**
> Creating roles is not necessary to complete this tutorial. By default, the account you are currently logged in with will have Administrator privileges on the model. However, to allow other users in your organization to browse the model by using a reporting client, you must create at least one role with Read permissions and add those users as members.

You will create three roles:

- **Sales Manager** - This role can include users in your organization for which you want to have Read permission to all model objects and data.
- **Sales Analyst US** - This role can include users in your organization for which you want only to be able to browse data related to sales in the United States. For this role, you will use a DAX formula to define a *Row Filter*, which restricts members to browse data only for the United States.
- **Administrator** - This role can include users for which you want to have Administrator permission, which allows unlimited access and permissions to perform administrative tasks on the model database.

Because Windows user and group accounts in your organization are unique, you can add accounts from your particular organization to members. However, for this tutorial, you can also leave the members blank. You will still be able to test the effect of each role later in Lesson 12: Analyze in Excel.

Estimated time to complete this lesson: **15 minutes**

Prerequisites

This topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 10: Create Partitions.

Create roles

To create a Sales Manager user role

1. In Tabular Model Explorer, right-click **Roles** > **Roles**.
2. In Role Manager, click **New**.
3. Click on the new role, and then in the **Name** column, rename the role to **Sales Manager**.
4. In the **Permissions** column, click the dropdown list, and then select the **Read** permission.

5. Optional: Click the **Members** tab, and then click **Add**. In the **Select Users or Groups** dialog box, enter the Windows users or groups from your organization you want to include in the role.

To create a Sales Analyst US user role

1. In Role Manager, click **New**.
2. Rename the role to **Sales Analyst US**.
3. Give this role **Read** permission.
4. Click on the Row Filters tab, and then for the **DimGeography** table only, in the DAX Filter column, type the following formula:

```
=DimGeography[CountryRegionCode] = "US"
```

A Row Filter formula must resolve to a Boolean (TRUE/FALSE) value. With this formula, you are specifying that only rows with the Country Region Code value of "US" be visible to the user.

5. Optional: Click on the **Members** tab, and then click **Add**. In the **Select Users or Groups** dialog box, enter the Windows users or groups from your organization you want to include in the role.

To create an Administrator user role

1. Click **New**.
2. Rename the role to **Administrator**.
3. Give this role **Administrator** permission.
4. Optional: Click on the **Members** tab, and then click **Add**. In the **Select Users or Groups** dialog box, enter the Windows users or groups from your organization you want to include in the role.

What's next?

Go to the next lesson: Lesson 12: Analyze in Excel.

Lesson 12: Analyze in Excel

8/23/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this lesson, you will use the Analyze in Excel feature in SSDT to open Microsoft Excel, automatically create a data source connection to the model workspace, and automatically add a PivotTable to the worksheet. The Analyze in Excel feature is meant to provide a quick and easy way to test the efficacy of your model design prior to deploying your model. You will not perform any data analysis in this lesson. The purpose of this lesson is to familiarize you, the model author, with the tools you can use to test your model design. Unlike using the Analyze in Excel feature, which is meant for model authors, end-users will use client reporting applications like Excel or Power BI to connect to and browse deployed model data.

In order to complete this lesson, Excel must be installed on the same computer as SSDT. To learn more, see Analyze in Excel.

Estimated time to complete this lesson: **20 minutes**

Prerequisites

This topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 11: Create Roles.

Browse using the Default and Internet Sales perspectives

In these first tasks, you will browse your model by using both the default perspective, which includes all model objects, and also by using the Internet Sales perspective you earlier. The Internet Sales perspective excludes the Customer table object.

To browse by using the Default perspective

1. Click the **Model** menu > **Analyze in Excel**.
2. In the **Analyze in Excel** dialog box, click **OK**.

   Excel will open with a new workbook. A data source connection is created using the current user account and the Default perspective is used to define viewable fields. A PivotTable is automatically added to the worksheet.
3. In Excel, in the **PivotTable Field List**, notice the **DimDate** and **FactInternetSales** measure groups appear, as well as the **DimCustomer**, **DimDate**, **DimGeography**, **DimProduct**, **DimProductCategory**, **DimProductSubcategory**, and **FactInternetSales** tables with all of their respective columns appear.
4. Close Excel without saving the workbook.

To browse by using the Internet Sales perspective

1. Click the **Model** menu, and then click **Analyze in Excel**.
2. In the **Analyze in Excel** dialog box, leave **Current Windows User** selected, then in the **Perspective** drop-down listbox, select **Internet Sales**, and then click **OK**.

3. In Excel, in **PivotTable Fields**, notice the DimCustomer table is excluded from the field list.

4. Close Excel without saving the workbook.

Browse by using roles

Roles are an integral part of any tabular model. Without at least one role to which users are added as members, users will not be able to access and analyze data using your model. The Analyze in Excel feature provides a way for you to test the roles you have defined.

To browse by using the Sales Manager user role

1. In SSDT, click the **Model** menu, and then click **Analyze in Excel**.

2. In the **Analyze in Excel** dialog box, in **Specify the user name or role to use to connect to the model**, select **Role**, and then in the drop-down listbox, select **Sales Manager**, and then click **OK**.

   Excel will open with a new workbook. A PivotTable is automatically created. The Pivot Table Field List includes all of the data fields available in your new model.

3. Close Excel without saving the workbook.

What's next?

Go to the next lesson: Lesson 13: Deploy.

Lesson 13: Deploy

8/23/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this lesson, you will configure deployment properties; specifying an on-premises or Azure server instance, and a name for the model. You'll then deploy the model to that instance. After your model is deployed, users can connect to it by using a reporting client application. To learn more about deploying, see Tabular model solution deployment and Deploy to Azure Analysis Services.

Estimated time to complete this lesson: **5 minutes**

Prerequisites

This topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this lesson, you should have completed the previous lesson: Lesson 12: Analyze in Excel.

Deploy the model

To configure deployment properties

1. In **Solution Explorer**, right-click the **AW Internet Sales** project, and then click **Properties**.
2. In the **AW Internet Sales Property Pages** dialog box, under **Deployment Server**, in the **Server** property, type the name of an Azure Analysis Services server or an on-premises server instance running in Tabular mode. This will be the server instance your model will be deployed to.

> **IMPORTANT**
>
> You must have Administrator permissions on the remote Analysis Services instance in-order to deploy to it.

3. In the **Database** property, type **Adventure Works Internet Sales**.
4. In the **Model Name** property, type **Adventure Works Internet Sales Model**.

5. Verify your selections and then click **OK**.

To deploy the Adventure Works Internet Sales tabular model

1. In **Solution Explorer**, right-click the **AW Internet Sales** project > **Build**.

2. Right-click the **AW Internet Sales** project > **Deploy**.

   When deploying to Azure Analysis Services, you'll likely be prompted to enter your account. Enter your organizational account and passsword, for example nancy@adventureworks.com. This account must be in Admins on the server instance.

   The Deploy dialog box appears and displays the deployment status of the metadata as well as each table included in the model.

3. When deployment successfully completes, go ahead and click **Close**.

Conclusion

Congratulations! You're finished authoring and deploying your first Analysis Services Tabular model. This tutorial has helped guide you through completing the most common tasks in creating a tabular model. Now that your Adventure Works Internet Sales model is deployed, you can use SQL Server Management Studio to manage the model; create process scripts and a backup plan. Users can also now connect to the model using a reporting client application such as Microsoft Excel or Power BI.

See also

DirectQuery Mode
Configure default data modeling and deployment properties
Tabular Model Databases

What's next?

- Supplemental Lesson - Implement Dynamic Security by Using Row Filters.
- Supplemental Lesson - Configure Reporting Properties for Power View Reports.

Supplemental Lesson - Implement Dynamic Security by Using Row Filters

11/27/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this supplemental lesson, you will create an additional role that implements dynamic security. Dynamic security provides row-level security based on the user name or login id of the user currently logged on. To learn more, see Roles.

To implement dynamic security, you must add a table to your model containing the Windows user names of those users that can create a connection to the model as a data source and browse model objects and data. The model you create using this tutorial is in the context of Adventure Works Corp.; however, in order to complete this lesson, you must add a table containing users from your own domain. You will not need the passwords for the user names that will be added. To create an EmployeeSecurity table, with a small sample of users from your own domain, you will use the Paste feature, pasting employee data from an Excel spreadsheet. In a real-world scenario, the table containing user names you add to a model would typically use a table from an actual database as a data source; for example, a real DimEmployee table.

In order to implement dynamic security, you will use two new DAX functions: USERNAME Function (DAX) and LOOKUPVALUE Function (DAX). These functions, applied in a row filter formula, are defined in a new role. Using the LOOKUPVALUE function, the formula specifies a value from the EmployeeSecurity table and then passes that value to the USERNAME function, which specifies the user name of the user logged on belongs to this role. The user can then browse only data specified by the role's row filters. In this scenario, you will specify that sales employees can only browse Internet sales data for the sales territories in which they are a member.

In order to complete this supplemental lesson, you will complete a series of tasks. Those tasks that are unique to this Adventure Works tabular model scenario, but would not necessarily apply to a real-world scenario are identified as such. Each task includes additional information describing the purpose of the task.

Estimated time to complete this lesson: **30 minutes**

Prerequisites

This supplemental lesson topic is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this supplemental lesson, you should have completed all previous lessons.

Add the DimSalesTerritory table to the AW Internet Sales Tabular Model Project

In order to implement dynamic security for this Adventure Works scenario, you must add two additional tables to your model. The first table you will add is DimSalesTerritory (as Sales Territory) from the same AdventureWorksDW database. You will later apply a row filter to the SalesTerritory table that defines the particular data the logged on user can browse.

To add the DimSalesTerritory table

1. In SSDT, click the **Model** menu, and then click **Existing Connections**.
2. In the **Existing Connections** dialog box, verify the **Adventure Works DB from SQL** data source connection is selected, and then click **Open**.

If the Impersonation Credentials dialog box appears, type the impersonation credentials you used in Lesson 2: Add Data.

3. On the **Choose How to Import the Data** page, leave **Select from a list of tables and views to choose the data to import** selected, and then click **Next**.

4. On the **Select Tables and Views** page, select the **DimSalesTerritory** table.

5. Click **Preview and Filter**.

6. Deselect the **SalesTerritoryAlternateKey** column, and then click **Ok**.

7. On the **Select Tables and Views** page, click **Finish**.

   The new table will be added to the model workspace. Objects and data from the source DimSalesTerritory table are then imported into your AW Internet Sales Tabular Model.

8. After the table has been imported, click **Close**.

Add a table with user name data

Because the DimEmployee table in the AdventureWorksDW sample database contains users from the AdventureWorks domain, and those user names do not exist in your own environment, you must create a table in your model that contains a small sample (three) of actual users from your organization. You will then add these users as members to the new role. You do not need the passwords for the sample user names, but you will need actual Windows user names from your own domain.

To add an EmployeeSecurity table

1. Open Microsoft Excel, creating a new worksheet.

2. Copy the following table, including the header row, and then paste it into the worksheet.

```
|EmployeeId|SalesTerritoryId|FirstName|LastName|LoginId|
|---------------|----------------------|--------------|-------------|------------|
|1|2|<user first name>|<user last name>|\<domain\username>|
|1|3|<user first name>|<user last name>|\<domain\username>|
|2|4|<user first name>|<user last name>|\<domain\username>|
|3|5|<user first name>|<user last name>|\<domain\username>|
```

3. Replace the first name, last name, and domain\username with the names and login ids of three users in your organization. Put the same user on the first two rows, for EmployeeId 1. This will show this user belongs to more than one sales territory. Leave the EmployeeId and SalesTerritoryId fields as they are.

4. Save the worksheet as **SampleEmployee**.

5. In the worksheet, select all of the cells with employee data, including the headers, then right-click the selected data, and then click **Copy**.

6. In SSDT, click the **Edit** menu, and then click **Paste**.

   If Paste is grayed out, click any column in any table in the model designer window, and try again.

7. In the **Paste Preview** dialog box, in **Table Name**, type **EmployeeSecurity**.

8. In **Data to be pasted**, verify the data includes all of the user data and headers from the SampleEmployee worksheet.

9. Verify **Use first row as column headers** is checked, and then click **Ok**.

   A new table named EmployeeSecurity with employee data copied from the SampleEmployee worksheet is created.

Create relationships between FactInternetSales, DimGeography, and DimSalesTerritory table

The FactInternetSales, DimGeography, and DimSalesTerritory table all contain a common column, SalesTerritoryId. The SalesTerritoryId column in the DimSalesTerritory table contains values with a different Id for each sales territory.

To create relationships between the FactInternetSales, DimGeography, and the DimSalesTerritory table

1. In the model designer, in Diagram View, in the **DimGeography** table, click and hold on the **SalesTerritoryId** column, then drag the cursor to the **SalesTerritoryId** column in the **DimSalesTerritory** table, and then release.

2. In the **FactInternetSales** table, click and hold on the **SalesTerritoryId** column, then drag the cursor to the **SalesTerritoryId** column in the **DimSalesTerritory** table, and then release.

   Notice the Active property for this relationship is False, meaning it's inactive. This is because the FactInternetSales table already has another active relationship that is used in measures.

Hide the EmployeeSecurity Table from client applications

In this task, you will hide the EmployeeSecurity table, keeping it from appearing in a client application's field list. Keep in-mind that hiding a table does not secure it. Users can still query EmployeeSecurity table data if they know how. In order to secure the EmployeeSecurity table data, preventing users from being able to query any of its data, you will apply a filter in a later task.

To hide the EmployeeSecurity table from client applications

- In the model designer, in Diagram View, right-click the **Employee** table heading, and then click **Hide from Client Tools**.

Create a Sales Employees by Territory user role

In this task, you will create a new user role. This role will include a row filter defining which rows of the DimSalesTerritory table are visible to users. The filter is then applied in the one-to-many relationship direction to all other tables related to DimSalesTerritory. You will also apply a simple filter that secures the entire EmployeeSecurity table from being queryable by any user that is a member of the role.

> **NOTE**
> The Sales Employees by Territory role you create in this lesson restricts members to browse (or query) only sales data for the sales territory to which they belong. If you add a user as a member to the Sales Employees by Territory role that also exists as a member in a role created in Lesson 11: Create Roles, you will get a combination of permissions. When a user is a member of multiple roles, the permissions, and row filters defined for each role are cumulative. That is, the user will have the greater permissions determined by the combination of roles.

To create a Sales Employees by Territory user role

1. In SSDT, click the **Model** menu, and then click **Roles**.

2. In **Role Manager**, click **New**.

   A new role with the None permission is added to the list.

3. Click on the new role, and then in the **Name** column, rename the role to **Sales Employees by Territory**.

4. In the **Permissions** column, click the dropdown list, and then select the **Read** permission.

5. Click on the **Members** tab, and then click **Add**.

6. In the **Select User or Group** dialog box, in **Enter the object named to select**, type the first sample user name you used when creating the EmployeeSecurity table. Click **Check Names** to verify the user name is valid, and then click **Ok**.

   Repeat this step, adding the other sample user names you used when creating the EmployeeSecurity table.

7. Click on the **Row Filters** tab.

8. For the **EmployeeSecurity** table, in the **DAX Filter** column, type the following formula.

   ```
   =FALSE()
   ```

   This formula specifies that all columns resolve to the false Boolean condition; therefore, no columns for the EmployeeSecurity table can be queried by a member of the Sales Employees by Territory user role.

9. For the **DimSalesTerritory** table, type the following formula.

   ```
   ='Sales Territory'[Sales Territory Id]=LOOKUPVALUE('Employee Security'[Sales Territory Id],
     'Employee Security'[Login Id], USERNAME(),
     'Employee Security'[Sales Territory Id],
     'Sales Territory'[Sales Territory Id])
   ```

   In this formula, the LOOKUPVALUE function returns all values for the DimEmployeeSecurity[SalesTerritoryId] column, where the EmployeeSecurity[LoginId] is the same as the current logged on Windows user name, and EmployeeSecurity[SalesTerritoryId] is the same as the DimSalesTerritory[SalesTerritoryId].

   The set of sales territory IDs returned by LOOKUPVALUE is then used to restrict the rows shown in the DimSalesTerritory table. Only rows where the SalesTerritoryID for the row is in the set of IDs returned by the LOOKUPVALUE function are displayed.

10. In Role Manager, click **Ok**.

Test the Sales Employees by Territory User Role

In this task, you will use the Analyze in Excel feature in SSDT to test the efficacy of the Sales Employees by Territory user role. You will specify one of the user names you added to the EmployeeSecurity table and as a member of the role. This user name will then be used as the effective user name in the connection created between Excel and the model.

To test the Sales Employees by Territory user role

1. In SSDT, click the **Model** menu, and then click **Analyze in Excel**.

2. In the **Analyze in Excel** dialog box, in **Specify the user name or role to use to connect to the model**, select **Other Windows User**, and then click **Browse**.

3. In the **Select User or Group** dialog box, in **Enter the object name to select**, type one of the user names you included in the EmployeeSecurity table, and then click **Check Names**.

4. Click **Ok** to close the **Select User or Group** dialog box, and then click **Ok** to close the **Analyze in Excel** dialog box.

   Excel will open with a new workbook. A PivotTable is automatically created. The PivotTable Fields list includes most of the data fields available in your new model.

   Notice the EmployeeSecurity table is not visible in the PivotTable Fields list. This is because you chose to hide this table from client tools in a previous task.

5. In the **Fields** list, in **∑ Internet Sales** (measures), select the **InternetTotalSales** measure. The measure will be entered into the **Values** fields.

6. Select the **SalesTerritoryId** column from the **DimSalesTerritory** table. The column will be entered into the **Row Labels** fields.

   Notice Internet sales figures appear only for the one region to which the effective user name you used belongs. If you select another column; for example, City, from the DimGeography table as Row Label field, only cities in the sales territory to which the effective user belongs are displayed.

   This user cannot browse or query any Internet sales data for territories other than the one they belong because the row filter defined for the Sales Territory table in the Sales Employees by Territory user role effectively secures data for all data related to other sales territories.

See Also

USERNAME Function (DAX)
LOOKUPVALUE Function (DAX)
CUSTOMDATA Function (DAX)

Supplemental Lesson - Configure Reporting Properties for Power View Reports

11/28/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2016) Azure Analysis Services

In this supplemental lesson, you will set reporting properties for the AW Internet Sales project. Reporting properties make it easier for end-users to select and display model data in Power View. You will also set properties to hide certain columns and tables, and create new data for use in charts.

Estimated time to complete this lesson: **30 minutes**

Prerequisites

This supplemental lesson is part of a tabular modeling tutorial, which should be completed in order. Before performing the tasks in this supplemental lesson, you should have completed all previous lessons.
In order to complete this particular supplemental lesson, you must also have the following:

- The AW Internet Sales project (completed through this tutorial) ready to be deployed or already deployed to an Analysis Services server.

Model properties that affect reporting

When authoring a tabular model, there are certain properties that you can set on individual columns and tables to enhance the end-user reporting experience in Power View. In addition, you can create additional model data to support data visualization and other features specific to the reporting client. For the sample Adventure Works Internet Sales Model, here are some of the changes you will make:

- **Add new data** - Adding new data in a calculated column by using a DAX formula creates date information in a format that is easier to display in charts.
- **Hide tables and columns that are not useful to the end user** - The **Hidden** property controls whether tables and table columns are displayed in the reporting client. Items with that are hidden are still part of the model and remain available for queries and calculations.
- **Enable one-click tables** - By default, no action occurs if an end-user clicks a table in the field list. To change this behavior so that a click on the table adds the table to the report, you will set Default Field Set on each column that you want to include in the table. This property is set on the table columns that end users will most likely want to use.
- **Set grouping where needed** - The **Keep Unique Rows** property determines if the values in the column should be grouped by values in a different field, such as an identifier field. For columns that contain duplicate values such as Customer Name (for example, multiple customers named John Smith), it is important to group (keep unique rows) on the **Row Identifier** field in order to provide your end users with the correct results.
- **Set data types and data formats** - By default, Power View applies rules based on column data type to determine whether the field can be used as a measure. Because each data visualization in Power View also has rules about where measures and non-measures can be placed, it is important to set the data type in the model, or override the default, to achieve the behavior you want for your end-user.
- **Set the Sort by Column** property - The **Sort By Column** property specifies if the values in the column

should be sorted by values in a different field. For example, on the Month Calendar column that contains the month name, sort by the column Month Number.

Hide tables from client Tools

Because there is already a Product Category calculated column and Product Subcategory calculated column in the Product table, it is not necessary to have the Product Category and Product Subcategory tables visible to client applications.

To hide the Product Category and Product Subcategory tables

1. In the model designer, right-click on the **Product Category** table (tab), and then click **Hide from Client Tools**.
2. Right-click on the **Product Subcategory** table (tab), and then click **Hide from Client Tools**.

Create new data for charts

Sometimes it may be necessary to create new data in your model by using DAX formulas. In this task, you will add two new calculated columns to the Date table. These new columns will provide date fields in a format convenient for use in charts.

To create new data for charts

1. In the **Date** table, scroll to the far right, and then click on **Add Column**.
2. Add two new calculated columns using the following formulas in the formula bar:

COLUMN NAME	FORMULA
Year Quarter	=[Calendar Year] & " Q" & [Calendar Quarter]
Year Month	=[Calendar Year] & FORMAT([Month],"#00")

Default field set

The Default Field Set is a predefined list of columns and measures for a table that are automatically added to a report canvas when the table is clicked on in the report field list. Essentially, you can specify the default columns, measures, and field ordering users will want to see when this table is visualized in Power View reports. For the Internet Sales model, you will define a default field set and order for the Customer, Geography, and Product tables. Included are only those most common columns that users will want to see when analyzing Adventure Works Internet Sales data by using Power View reports.

For detailed information about Default Field Set, see Configure Default Field Set for Power View Reports in SQL Server Books Online.

To set default field set for tables

1. In the model designer, click the **Customer** table (tab).
2. In the **Properties** window, under **Reporting Properties**, in the **Default Field Set** property, click **Click to edit** to open the **Default Field Set** dialog box.
3. In the **Default Field Set** dialog box, in the **Fields in the table** list box, press Ctrl, and select the following fields, and then click **Add**.

   Birth Date, **Customer Alternate Id**, **First Name**, **Last Name**.
4. In the **Default fields, in order** window, use the Move Up and Move Down buttons to put the following order:

Customer Alternate Id

First Name

Last Name

Birth Date.

5. Click **Ok** to close the **Default Field Set** dialog box for the **Customer** table.

6. Perform these same steps for the **Geography** table, selecting the following fields and putting them in this order.

   City, **State Province Code**, **Country Region Code**.

7. Finally, perform these same steps for the **Product** table, selecting the following fields and putting them in this order.

   Product Alternate Id, **Product Name**.

Table behavior

By using Table Behavior properties, you can change the default behavior for different visualization types and grouping behavior for tables used in Power View reports. This allows better default placement of identifying information such as names, images, or titles in tile, card, and chart layouts.

For detailed information about Table Behavior properties, see Configure Table Behavior Properties for Power View Reports in SQL Server Books Online.

To set table behavior

1. In the model designer, click the **Customer** table (tab).

2. In the **Properties** window, in the **Table Behavior** property, click **Click to edit**, to open the **Table Behavior** dialog box.

3. In the **Table Behavior** dialog box, in the **Row Identifier** dropdown list box, select the **Customer Id** column.

4. In the **Keep Unique Rows** list box, select **First Name** and **Last Name**.

   This property setting specifies these columns provide values that should be treated as unique even if they are duplicates, for example, when two or more employees share the same name.

5. In the **Default Label** dropdown list box, select the **Last Name** column.

   This property setting specifies this column provides a display name to represent row data.

6. Repeat these steps for the **Geography** table, selecting the **Geography Id** column as the Row Identifier, and the **City** column in the **Keep Unique Rows** list box. You do not need to set a Default Label for this table.

7. Repeat these steps, for the **Product** table, selecting the **Product Id** column as the Row Identifier, and the **Product Name** column in the **Keep Unique Rows** list box. For **Default Label**, select **Product Alternate Id**.

Reporting properties for columns

There are a number of basic column properties and specific reporting properties on columns you can set to improve the model reporting experience. For example, it may not be necessary for users to see every column in every table. Just as you hid the Product Category and Product Subcategory tables earlier, by using a column's Hidden property, you can hide particular columns from a table that is otherwise shown. Other properties, such as

Data Format and Sort by Column, can also affect how column data can appear in reports. You will set some of those on particular columns now. Other columns require no action, and are not shown below.

You will only set a few different column properties here, but there are many others. For more detailed information about column reporting properties, see Column Properties in SQL Server Books Online.

To set properties for columns

1. In the model designer, click the **Customer** table (tab).
2. Click on the **Customer Id** column to display the column properties in the **Properties** window.
3. In the **Properties** window, set the **Hidden** property to True. The **Customer Id** column then becomes greyed out in the model designer.
4. Repeat these steps, setting the following column and reporting properties for each table specified. Leave all other properties at their default settings.

   Note: For all date columns, make sure **Data Type** is **Date**.

   Customer

COLUMN	PROPERTY	VALUE
Geography Id	Hidden	True
Birth Date	Data Format	Short Date

   Date

> **NOTE**
> Because the Date table was selected as the models date table by using the Mark as Date Table setting, in Lesson 7: Mark as Date Table, and the Date column in the Date table as the column to be used as the unique identifier, the Row Identifier property for the Date column will automatically be set to True, and cannot be changed. When using time-intelligence functions in DAX formulas, you must specify a date table. In this model, you created a number of measures using time-intelligence functions to calculate sales data for various periods such as previous and current quarters, and also for use in KPIs. For more information about specifying a date table, see Specify Mark as Date Table for use with Time Intelligence in SQL Server Books Online.

COLUMN	PROPERTY	VALUE
Date	Data Format	Short Date
Day Number of Week	Hidden	True
Day Name	Sort By Column	Day Number of Week
Day of Week	Hidden	True
Day of Month	Hidden	True
Day of Year	Hidden	True
Month Name	Sort By Column	Month

COLUMN	PROPERTY	VALUE
Month	Hidden	True
Month Calendar	Hidden	True
Fiscal Quarter	Hidden	True
Fiscal Year	Hidden	True
Fiscal Semester	Hidden	True

Geography

COLUMN	PROPERTY	VALUE
Geography Id	Hidden	True
Sales Territory Id	Hidden	True

Product

COLUMN	PROPERTY	VALUE
Product Id	Hidden	True
Product Alternate Id	Default Label	True
Product Subcategory Id	Hidden	True
Product Start Date	Data Format	Short Date
Product End Date	Data Format	Short Date

Internet Sales

COLUMN	PROPERTY	VALUE
Product Id	Hidden	True
Customer Id	Hidden	True
Promotion Id	Hidden	True
Currency Id	Hidden	True
Sales Territory Id	Hidden	True
Order Quantity	Data Type	Decimal Number
	Data Format	Decimal Number
	Decimal Places	0

COLUMN	PROPERTY	VALUE
Order Date	Data Format	Short Date
Due Date	Data Format	Short Date
Ship Date	Data Format	Short Date

Redeploy the Adventure Works Internet Sales tabular model

Because you have changed the model, you must re-deploy it.

To redeploy the Adventure Works Internet Sales tabular model

- In SSDT, click the **Build** menu, and then click **Deploy Adventure Works Internet Sales Model**.

  The **Deploy** dialog box appears and displays the deployment status of the metadata as well as each table included in the model.

Next steps

You can now use Power View to visualize data from the model. Ensure the Analysis Services and Reporting Services accounts on the SharePoint site have read permissions to the Analysis Services instance where you deployed your model.

To create a Reporting Services report data source that points to your model, see Table Model Connection Type (SSRS).

Multidimensional Modeling (Adventure Works Tutorial)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Welcome to the Analysis Services Tutorial. This tutorial describes how to use SQL Server Data Tools to develop and deploy an Analysis Services project, using the fictitious company Adventure Works Cycles for all examples.

What you learn

In this tutorial, you will learn the following:

- How to define data sources, data source views, dimensions, attributes, attribute relationships, hierarchies, and cubes in an Analysis Services project within SQL Server Data Tools.
- How to view cube and dimension data by deploying the Analysis Services project to an instance of Analysis Services, and how to then process the deployed objects to populate them with data from the underlying data source.
- How to modify the measures, dimensions, hierarchies, attributes, and measure groups in the Analysis Services project, and how to then deploy the incremental changes to the deployed cube on the development server.
- How to define calculations, Key Performance Indicators (KPIs), actions, perspectives, translations, and security roles within a cube.

A scenario description accompanies this tutorial so that you can better understand the context for these lessons. For more information, see Analysis Services Tutorial Scenario.

Prerequisites

You will need sample data, sample project files, and software to complete all of the lessons in this tutorial. For instructions on how to find and install the prerequisites for this tutorial, see Install Sample Data and Projects for the Analysis Services Multidimensional Modeling Tutorial.

Additionally, the following permissions must be in place to successfully complete this tutorial:

- You must be a member of the Administrators local group on the Analysis Services computer or be a member of the server administration role in the instance of Analysis Services.
- You must have Read permissions in the **AdventureWorksDW** sample database. This sample database is valid for the SQL Server 2017 release.

Lessons

This tutorial includes the following lessons.

LESSON	ESTIMATED TIME TO COMPLETE
Lesson 1: Defining a Data Source View within an Analysis Services Project	15 minutes

LESSON	ESTIMATED TIME TO COMPLETE
Lesson 2: Defining and Deploying a Cube	30 minutes
Lesson 3: Modifying Measures, Attributes and Hierarchies	45 minutes
Lesson 4: Defining Advanced Attribute and Dimension Properties	120 minutes
Lesson 5: Defining Relationships Between Dimensions and Measure Groups	45 minutes
Lesson 6: Defining Calculations	45 minutes
Lesson 7: Defining Key Performance Indicators (KPIs)	30 minutes
Lesson 8: Defining Actions	30 minutes
Lesson 9: Defining Perspectives and Translations	30 minutes
Lesson 10: Defining Administrative Roles	15 minutes

NOTE

The cube database that you will create in this tutorial is a simplified version of the Analysis Services multidimensional model project that is part of the Adventure Works sample databases available for download on GitHub. The tutorial version of the Adventure Works multidimensional database is simplified to bring greater focus to the specific skills that you will want to master right away. After you complete the tutorial, consider exploring the multidimensional model project on your own to further your understanding of Analysis Services multidimensional modeling.

Next Step

To begin the tutorial, continue to the first lesson: Lesson 1: Defining a Data Source View within an Analysis Services Project.

Analysis Services Tutorial Scenario

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This tutorial is based on Adventure Works Cycles, a fictitious company. Adventure Works Cycles is a large, multinational manufacturing company that produces and distributes metal and composite bicycles to commercial markets in North America, Europe, and Asia. The headquarters for Adventure Works Cycles is Bothell, Washington, where the company employs 500 workers. Additionally, Adventure Works Cycles employs several regional sales teams throughout its market base.

In recent years, Adventure Works Cycles bought a small manufacturing plant, Importadores Neptuno, which is located in Mexico. Importadores Neptuno manufactures several critical subcomponents for the Adventure Works Cycles product line. These subcomponents are shipped to the Bothell location for final product assembly. In 2005, Importadores Neptuno became the sole manufacturer and distributor of the touring bicycle product group.

Following a successful fiscal year, Adventure Works Cycles now wants to broaden its market share by targeting advertising to its best customers, extending product availability through an external Web site, and reducing the cost of sales by reducing production costs.

Current Analysis Environment

To support the data analysis needs of the sales and marketing teams and of senior management, the company currently takes transactional data from the AdventureWorks2012 database, and non-transactional information such as sales quotas from spreadsheets, and consolidates this information into the **AdventureWorksDW2012** relational data warehouse. However, the relational data warehouse presents the following challenges:

- Reports are static. Users have no way to interactively explore the data in the reports to obtain more detailed information, such as they could do with a Microsoft Office Excel pivot table. Although the existing set of predefined reports is sufficient for many users, more advanced users need direct query access to the database for interactive queries and specialized reports. However, because of the complexity of the **AdventureWorksDW2012** database, too much time is needed for such users to master how to create effective queries.
- Query performance is widely variable. For example, some queries return results very quickly, in only a few seconds, while other queries take several minutes to return.
- Aggregate tables are difficult to manage. In an attempt to improve query response times, the data warehouse team at Adventure Works built several aggregate tables in the **AdventureWorksDW2012** database. For example, they built a table that summarizes sales by month. However, while these aggregate tables greatly improve query performance, the infrastructure that they built to maintain the tables over time is fragile and prone to errors.
- Complex calculation logic is buried in report definitions and is difficult to share between reports. Because this business logic is generated separately for each report, summary information sometimes is different between reports. Therefore, management has limited confidence in the data warehouse reports.
- Users in different business units are interested in different views of the data. Each group is distracted and confused by data elements that are irrelevant to them.
- Calculation logic is particularly challenging for users who need specialized reports. Because such users must define the calculation logic separately for each report, there is no centralized control over how the calculation logic is defined. For example, some users know that they should use basic statistical techniques

such as moving averages, but they do not know how to construct such calculations and so do not use these techniques.

- It is difficult to combine related sets of information. Specialized queries that combine two sets of related information, such as sales and sales quotas, are difficult for business users to construct. Such queries overwhelmed the database, so the company requires that users request cross-subject-area sets of data from the data warehouse team. As a result, only a handful of predefined reports have been defined that combine data from multiple subject areas. Additionally, users are reluctant to try to modify these reports because of their complexity.
- Reports are focused primarily on business information in the United States. Users in the non-U.S. subsidiaries are very dissatisfied with this focus, and want to be able to view reports in different currencies and different languages.
- Information is difficult to audit. The Finance department currently uses the **AdventureWorksDW2012** database only as a source of data from which to query in bulk. They then download the data into individual spreadsheets, and spend significant time preparing the data and manipulating the spreadsheets. The corporate financial reports are therefore difficult to prepare, audit, and manage across the company.

The Solution

The data warehouse team recently performed a design review of the current analysis system. The review included a gap analysis of current issues and future demands. The data warehouse team determined that the **AdventureWorksDW2012** database is a well-designed dimensional database with conformed dimensions and surrogate keys. Conformed dimensions enable a dimension to be used in multiple data marts, such as a time dimension or a product dimension. Surrogate keys are artificial keys that link dimension and fact tables and that are used to ensure uniqueness and to improve performance. Moreover, the data warehouse team determined that there currently are no significant problems with the loading and management of the base tables in the **AdventureWorksDW2012** database. The team has therefore decided to use Microsoft Analysis Services to accomplish the following:

- Provide unified data access through a common metadata layer for analytical analysis and reporting.
- Simplify users' view of data, speeding the development of both interactive and predefined queries and predefined reports.
- Correctly construct queries that combine data from multiple subject areas.
- Manage aggregates.
- Store and reuse complex calculations.
- Present a localized experience to business users outside the United States.

The lessons in the Analysis Services tutorial provide guidance in building a cube database that meets all of these goals. To get started, continue to the first lesson: Lesson 1: Create a New Tabular Model Project.

See Also

Multidimensional Modeling (Adventure Works Tutorial)

Install sample data and multidimensional projects

12/7/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use the instructions and links provided in this article to install the data and project files used in the Analysis Services Tutorials.

Step 1: Install prerequisites

The lessons in this tutorial assume that you have the following software installed. You can install all of the features on a single computer. To install these features, run SQL Server Setup and select them from the Feature Selection page.

- SQL Server Database Engine
- SQL Server Analysis Services (SSAS)

  Analysis Services is available in these editions only: Evaluation, Enterprise, Business Intelligence, Standard. Multidimensional models are not supported in Azure Analysis Services.

  By default, Analysis Services 2016 and later is installed as a tabular instance, which you can override by choosing Multidimensional Server Mode in the server configuration page of the Installation Wizard.

Step 2: Download and install developer and management tools

SQL Server Data Tools (SSDT) for Visual Studio is downloaded and installed separately from other SQL Server features. The designers and project templates used to create BI models and reports are included in SSDT for Visual Studio 2015 or as Nuget packages for Visual Studio 2017.

Download SQL Server Data Tools.

SQL Server Management Studio (SSMS) is downloaded and installed separately from other SQL Server features.

Download SQL Server Management Studio

Optionally, consider installing Excel to browse your multidimensional data as you proceed through the tutorial. Installing Excel enables the **Analyze in Excel** feature that starts Excel using a PivotTable field list that is connected to the cube you are building. Using Excel to browse data is recommended because you can quickly build a pivot report that lets you interact with the data.

Alternatively, you can browse data using the built-in MDX query designer that is built into SQL Server Data Tools (SSDT). The query designer returns the same data, except the data is presented as a flat rowset.

Step 3: Install databases

An Analysis Services multidimensional model uses transactional data that you import from a relational database management system. For the purposes of this tutorial, you use the following relational database as your data source.

- **AdventureWorksDW2012 or later** - This is a relational data warehouse that runs on a Database Engine instance. It provides the original data used by the Analysis Services databases and projects that you build and deploy throughout the tutorial. The tutorial assumes you are using AdventureWorksDW2012, however, later versions do work.

You can use this sample database with SQL Server 2012 (11.x) and later. In-general, you should use the sample database version matching your database engine version.

To install the database, do the following:

1. Download an AdventureWorkDW database backup from GitHub.
2. Copy the backup file to the data directory of the local SQL Server Database Engine instance.
3. Start SQL Server Management Studio and connect to the Database Engine instance.
4. Restore the database.

Step 4: Grant database permissions

The sample projects use data source impersonation settings that specify the security context under which data is imported or processed. By default, the impersonation settings specify the Analysis Services service account for accessing the data. To use this default setting, you must ensure that the service account under which Analysis Services runs has data reader permissions on the **AdventureWorksDW** database.

> **NOTE**
> For learning purposes, it is recommended that you use the default service account impersonation option and grant data reader permissions to the service account in SQL Server. Although other impersonation options are available, not all of them are suitable for processing operations. Specifically, the option for using the credentials of the current user is not supported for processing.

1. Determine the service account. You can use SQL Server Configuration Manager or the Services console application to view account information. If you installed Analysis Services as the default instance, using the default account, the service is running as **NT Service\MSSQLServerOLAPService**.
2. In Management Studio, connect to the database engine instance.
3. Expand the Security folder, right-click Logins and select **New Login**.
4. On the General page, in Login name, type **NT Service\MSSQLServerOLAPService** (or whatever account the service is running as).
5. Click **User Mapping**.
6. Select the checkbox next to the **AdventureWorksDW** database. Role membership should automatically include **db_datareader** and **public**. Click **OK** to accept the defaults.

Step 5: Install projects

The tutorial includes sample projects so that you can compare your results against a finished project, or start a lesson that is further on in the sequence.

1. Download the adventure-works-multidimensional-tutorial-projects.zip from the Adventure Works for Analysis Services samples page on GitHub.

   The tutorial projects work for SQL Server 2012 (11.x) and later.

2. Move the .zip file to a folder just below the root drive (for example, C:\Tutorial). This step mitigates the "Path too long" error that sometimes occurs if you attempt to unzip the files in the Downloads folder.
3. Unzip the sample projects: right-click on the file and select **Extract All**. After extracting the files, you should have folders Lesson 1, 2, 3, 5, 6, 7, 8, 9, 10 Complete and Lesson 4 Start.

4. Remove the read-only permissions on these files. Right-click the parent folder, select **Properties**, and clear the checkbox for **Read-only**. Click **OK**. Apply the changes to this folder, subfolders, and files.

5. Open the solution (.sln) file that corresponds to the lesson you are in. For example, in the folder named "Lesson 1 Complete", you would open the Analysis Services Tutorial.sln file.

6. Deploy the solution to verify that database permissions and server location information are setup correctly.

   If Analysis Services and the Database Engine are installed as the default instance (MSSQLServer) and all software is running on the same computer, you can click **Deploy Solution** on the Build menu to build and deploy the sample project to the local Analysis Services instance. During deployment, data is processed (or imported) from the **AdventureWorksDW** database on the local Database Engine instance. A new Analysis Services database is created on the Analysis Services instance that contains the data retrieved from the Database Engine.

   If you encounter errors, review the previous steps on setting up database permissions. Additionally, you might also need to change server names. The default server name is localhost. If your servers are installed on remote computers or as named instances, you must override the default to use a server name that is valid for your installation. Furthermore, if the servers are on remote computers, you might need to configure Windows Firewall to allow access to the servers.

   The server name for connecting to the database engine is specified in the Data Source object of the multidimensional solution (Adventure Works Tutorial), visible in Solution Explorer.

   The server name for connecting to Analysis Services is specified in the Deployment tab of the Property Pages of the project, also visible in Solution Explorer.

7. In SQL Server Management Studio, connect to Analysis Services. Verify the database named **Analysis Services Tutorial** is running on the server.

Next step

You are now ready to use the tutorial. For more information about how to get started, see Multidimensional Modeling (Adventure Works Tutorial).

See also

Configure the Windows Firewall to Allow Analysis Services Access
Configure the Windows Firewall to Allow SQL Server Access

Lesson 1: Defining a Data Source View within an Analysis Services Project

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Designing a business intelligence application in SQL Server starts with creating an Analysis Services project in SQL Server Data Tools (SSDT). Within this project, you define all the elements of your solution, starting with a data source view.

This lesson contains the following tasks:

Creating an Analysis Services Project
In this task, you create the Analysis Services Tutorial project, based on an Analysis Services multidimensional model template.

Defining a Data Source
In this task, you specify the **AdventureWorksDW** database as the data source for the Analysis Services dimensions and cubes that you define in subsequent lessons.

Defining a Data Source View
In this task, you define a single unified view of the metadata from selected tables in the **AdventureWorksDW** database.

Modifying Default Table Names
In this task, you modify table names in the data source view, so that the names of subsequent Analysis Services objects that you define are more user-friendly.

Next Lesson

Lesson 2: Defining and Deploying a Cube

See Also

Data Source Views in Multidimensional Models

Multidimensional Modeling (Adventure Works Tutorial)

Lesson 1-1 - Creating an Analysis Services Project

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In the following task, you use SQL Server Data Tools (SSDT) to create a new Analysis Services project named **Analysis Services Tutorial**, based on the Analysis Services Project template. A *project* is a collection of related objects. Projects exist within a solution, which includes one or more projects. For more information, see Create an Analysis Services Project (SSDT).

To create a new Analysis Services project

1. Open SQL Server Data Tools.
2. Create a new Analysis Services Multidimensional project. Choose the **Analysis Services Multidimensional and Data Mining Project** template.

   Notice the default project name, location, and the default solution name are generated in the bottom of the dialog box. By default, a new directory is created for the solution.
3. Change the project Name to **Analysis Services Tutorial**, which also changes the **Solution name** box, and then click **OK**.

You have successfully created the **Analysis Services Tutorial** project, based on the **Analysis Services Multidimensional and Data Mining Project** template, within a new solution that is also named **Analysis Services Tutorial**.

Next Task in Lesson

Defining a Data Source

See Also

Creating Multidimensional Models Using SQL Server Data Tools (SSDT)
Create an Analysis Services Project (SSDT)

Lesson 1-2 - Defining a Data Source

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you create an Analysis Services project, you generally start working with the project by defining one or more data sources that the project will use. When you define a data source, you are defining the connection string information that will be used to connect to the data source. For more information, see Create a Data Source (SSAS Multidimensional).

In the following task, you define the AdventureWorksDWSQLServer2012 sample database as the data source for the Analysis Services Tutorial project. While this database is located on your local computer for the purposes of this tutorial, source databases are frequently hosted on one or more remote computers.

To define a new data source

1. In Solution Explorer (on the right of the Microsoft Visual Studio window), right-click **Data Sources**, and then click **New Data Source**.

2. On the **Welcome to the Data Source Wizard** page of the **Data Source Wizard**, click **Next** to open the **Select how to define the connection** page.

3. On the **Select how to define the connection** page, you can define a data source based on a new connection, based on an existing connection, or based on a previously defined data source object. In this tutorial, you define a data source based on a new connection. Verify that **Create a data source based on an existing or new connection** is selected, and then click **New**.

4. In the **Connection Manager** dialog box, you define connection properties for the data source. In the **Provider** list box, verify that **Native OLE DB\SQL Server Native Client 11.0** is selected.

   Analysis Services also supports other providers, which are displayed in the **Provider** list.

5. In the **Server name** text box, type **localhost**.

   To connect to a named instance on your local computer, type **localhost**. To connect to the specific computer instead of the local computer, type the computer name or IP address.

6. Verify that **Use Windows Authentication** is selected. In the **Select or enter a database name** list, select **AdventureWorksDW2012**.

7. Click **Test Connection** to test the connection to the database.

8. Click **OK**, and then click **Next**.

9. On the **Impersonation Information** page of the wizard, you define the security credentials for Analysis Services to use to connect to the data source. Impersonation affects the Windows account used to connect to the data source when Windows Authentication is selected. Analysis Services does not support impersonation for processing OLAP objects. Select **Use the service account**, and then click **Next**.

10. On the **Completing the Wizard** page, accept the default name, **Adventure Works DW 2012**, and then click **Finish** to create the new data source.

> **NOTE**
> To modify the properties of the data source after it has been created, double-click the data source in the **Data Sources** folder to display the data source properties in **Data Source Designer**.

Next Task in Lesson

Defining a Data Source View

See Also

Create a Data Source (SSAS Multidimensional)

Lesson 1-3 - Defining a Data Source View

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you define the data sources that you will use in an Analysis Services project, the next step is generally to define a data source view for the project. A data source view is a single, unified view of the metadata from the specified tables and views that the data source defines in the project. Storing the metadata in the data source view enables you to work with the metadata during development without an open connection to any underlying data source. For more information, see Data Source Views in Multidimensional Models.

In the following task, you define a data source view that includes five tables from the **AdventureWorksDW2012** data source.

To define a new data source view

1. In Solution Explorer (on the right of the Microsoft Visual Studio window), right-click **Data Source Views**, and then click **New Data Source View**.

2. On the **Welcome to the Data Source View Wizard** page, click **Next**. The **Select a Data Source** page appears.

3. Under **Relational data sources**, the **Adventure Works DW 2012** data source is selected. Click **Next**.

   > **NOTE**
   > To create a data source view that is based on multiple data sources, first define a data source view that is based on a single data source. This data source is then called the primary data source. You can then add tables and views from a secondary data source. When designing dimensions that contain attributes based on related tables in multiple data sources, you might need to define a MicrosoftSQL Server data source as the primary data source to use its distributed query engine capabilities.

4. On the **Select Tables and Views** page, select tables and views from the list of objects that are available from the selected data source. You can filter this list to help you select tables and views.

   > **NOTE**
   > Click the maximize button in the upper-right corner so that the window covers the full screen. This makes it easier to see the complete list of available objects.

   In the **Available objects** list, select the following objects. You can select multiple tables by clicking each while holding down the CTRL key:

   - **DimCustomer (dbo)**
   - **DimDate (dbo)**
   - **DimGeography (dbo)**
   - **DimProduct (dbo)**
   - **FactInternetSales (dbo)**

5. Click **>** to add the selected tables to the **Included objects** list.

6. Click **Next.**

7. In the Name field, make sure **Adventure Works DW 2012** displays, and then click **Finish**.

   The **Adventure Works DW 2012** data source view appears in the **Data Source Views** folder in Solution Explorer. The content of the data source view is also displayed in Data Source View Designer in SQL Server Data Tools (SSDT). This designer contains the following elements:

   - A **Diagram** pane in which the tables and their relationships are represented graphically.
   - A **Tables** pane in which the tables and their schema elements are displayed in a tree view.
   - A **Diagram Organizer** pane in which you can create subdiagrams so that you can view subsets of the data source view.
   - A toolbar that is specific to Data Source View Designer.

8. To maximize the Microsoft Visual Studio development environment, click the **Maximize** button.

9. To view the tables in the **Diagram** pane at 50 percent, click the **Zoom** icon on the Data Source View Designer toolbar. This will hide the column details of each table.

10. To hide Solution Explorer, click the **Auto Hide** button, which is the pushpin icon on the title bar. To view Solution Explorer again, position your pointer over the Solution Explorer tab along the right side of the development environment. To unhide Solution Explorer, click the **Auto Hide** button again.

11. If the windows are not hidden by default, click **Auto Hide** on the title bar of the Properties and Solution Explorer windows.

    You can now view all the tables and their relationships in the **Diagram** pane. Notice that there are three relationships between the FactInternetSales table and the DimDate table. Each sale has three dates associated with the sale: an order date, a due date, and a ship date. To view the details of any relationship, double-click the relationship arrow in the **Diagram** pane.

Next Task in Lesson

Modifying Default Table Names

See Also

Data Source Views in Multidimensional Models

Lesson 1-4 - Modifying Default Table Names

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can change the value of the **FriendlyName** property for objects in the data source view to make them easier to notice and use.

In the following task, you will change the friendly name of each table in the data source view by removing the "**Dim**" and "**Fact**" prefixes from these tables. This will make the cube and dimension objects (that you will define in the next lesson) easier to notice and use.

> **NOTE**
>
> You can also change the friendly names of columns, define calculated columns, and join tables or views in the data source view to make them easier to use.

To modify the default name of a table

1. In the **Tables** pane of **Data Source View Designer**, right-click the **FactInternetSales** table, and then click **Properties**.

2. If the Properties window on the right side of the Microsoft Visual Studio window is not displayed, click the **Auto Hide** button on the title bar of the Properties window so that this window remains visible.

   It is easier to change the properties for each table in the data source view when the Properties window remains open. If you do not pin the window open by using the **Auto Hide** button, the window will close when you click a different object in the **Diagram** pane.

3. Change the **FriendlyName** property for the **FactInternetSales** object to ***InternetSales***.

   When you click away from the cell for the **FriendlyName** property, the change is applied. In the next lesson, you will define a measure group that is based on this fact table. The name of the fact table will be InternetSales instead of FactInternetSales because of the change you made in this lesson.

4. Click **DimProduct** in the **Tables** pane. In the Properties window, change the **FriendlyName** property to ***Product***.

5. Change the **FriendlyName** property of each remaining table in the data source view in the same way, to remove the "**Dim**" prefix.

6. When you have finished, click the **Auto Hide** button to hide the Properties window again.

7. On the **File** menu, or on the toolbar of SQL Server Data Tools, click **Save All** to save the changes you have made to this point in the Analysis Services Tutorial project. You can stop the tutorial here if you want and resume it later.

Next Lesson

Lesson 2: Defining and Deploying a Cube

See Also

Data Source Views in Multidimensional Models

Change Properties in a Data Source View (Analysis Services)

Lesson 2: Defining and Deploying a Cube

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you define a data source view in your Microsoft Analysis Services project, you are ready to define an initial Analysis Services cube.

You can define a cube and its dimensions in a single pass using the Cube Wizard. Alternatively, you can define one or more dimensions and then use the Cube Wizard to define a cube that uses those dimensions. If you are designing a complex solution, you generally start by defining the dimensions. For more information, see Dimensions in Multidimensional Models or Cubes in Multidimensional Models.

> **NOTE**
> Completed projects for all of the lessons in this tutorial are available online. You can jump ahead to any lesson by using the completed project from the previous lesson as a starting point. Click here to download the sample projects that go with this tutorial.

This lesson contains the following tasks:

Defining a Dimension
In this task, you use the Dimension Wizard to define a dimension.

Defining a Cube
In this task, you use the Cube Wizard to define an initial Analysis Services cube.

Adding Attributes to Dimensions
In this task, you add attributes to the dimensions that you created.

Reviewing Cube and Dimension Properties
In this task, you review the structure of the cube that you defined by using the Cube Wizard.

Deploying an Analysis Services Project
In this task, you deploy the Analysis Services project to your local instance of Analysis Services, and learn about certain deployment properties.

Browsing the Cube
In this task, you browse the cube and dimension data by using Excel or the MDX query designer.

Next Lesson

Lesson 3: Modifying Measures, Attributes and Hierarchies

See Also

Analysis Services Tutorial Scenario
Multidimensional Modeling (Adventure Works Tutorial)
Dimensions in Multidimensional Models
Cubes in Multidimensional Models
Configure Analysis Services Project Properties (SSDT)
Build Analysis Services Projects (SSDT)

Deploy Analysis Services Projects (SSDT)

Lesson 2-1 - Defining a Dimension

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In the following task, you will use the Dimension Wizard to build a Date dimension.

> **NOTE**
>
> This lesson requires that you have completed all the procedures in Lesson 1.

To define a dimension

1. In Solution Explorer (on the right side of Microsoft Visual Studio), right-click **Dimensions**, and then click **New Dimension**. The Dimension Wizard appears.
2. On the **Welcome to the Dimension Wizard** page, click **Next**.
3. On the **Select Creation Method** page, verify that the **Use an existing table** option is selected, and then click **Next**.
4. On the **Specify Source Information** page, verify that the **Adventure Works DW 2012** data source view is selected.
5. In the **Main table** list, select **Date**.
6. Click **Next**.
7. On the **Select Dimension Attributes** page, select the check boxes next to the following attributes:
   - **Date Key**
   - **Full Date Alternate Key**
   - **English Month Name**
   - **Calendar Quarter**
   - **Calendar Year**
   - **Calendar Semester**
8. Change the setting of the **Full Date Alternate Key** attribute's **Attribute Type** column from **Regular** to **Date**. To do this, click **Regular** in the **Attribute Type** column. Then click the arrow to expand the options. Next, click **Date** > **Calendar** > **Date**. Click **OK**. Repeat these steps to change the attribute type of the attributes as follows:
   - **English Month Name** to **Month**
   - **Calendar Quarter** to **Quarter**
   - **Calendar Year** to **Year**
   - **Calendar Semester** to **Half Year**
9. Click **Next**.
10. On the **Completing the Wizard** page, in the Preview pane, you can see the **Date** dimension and its

attributes.

11. Click **Finish** to complete the wizard.

    In Solution Explorer, in the Analysis Services Tutorial project, the Date dimension appears in the **Dimensions** folder. In the center of the development environment, Dimension Designer displays the Date dimension.

12. On the **File** menu, click **Save All**.

Next Task in Lesson

Defining a Cube

See Also

Dimensions in Multidimensional Models
Create a Dimension by Using an Existing Table
Create a Dimension Using the Dimension Wizard

Lesson 2-2 - Defining a Cube

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Cube Wizard helps you define the measure groups and dimensions for a cube. In the following task, you will use the Cube Wizard to build a cube.

To define a cube and its properties

1. In Solution Explorer, right-click **Cubes**, and then click **New Cube**. The Cube Wizard appears.
2. On the **Welcome to the Cube Wizard** page, click **Next**.
3. On the **Select Creation Method** page, verify that the **Use existing tables** option is selected, and then click **Next**.
4. On the **Select Measure Group Tables** page, verify that the **Adventure Works DW 2012** data source view is selected.
5. Click **Suggest** to have the cube wizard suggest tables to use to create measure groups.

   The wizard examines the tables and suggests **InternetSales** as a measure group table. Measure group tables, also called fact tables, contain the measures you are interested in, such as the number of units sold.
6. Click **Next**.
7. On the **Select Measures** page, review the selected measures in the **Internet Sales** measure group, and then clear the check boxes for the following measures:

   - **Promotion Key**
   - **Currency Key**
   - **Sales Territory Key**
   - **Revision Number**

   By default, the wizard selects as measures all numeric columns in the fact table that are not linked to dimensions. However, these four columns are not actual measures. The first three are key values that link the fact table with dimension tables that are not used in the initial version of this cube.
8. Click **Next**.
9. On the **Select Existing Dimensions** page, make sure the **Date** dimension that you created earlier is selected, and then click **Next**.
10. On the **Select New Dimensions** page, select the new dimensions to be created. To do this, verify that the **Customer**, **Geography**, and **Product** check boxes are selected, and then clear the **InternetSales** check box.
11. Click **Next**.
12. On the **Completing the Wizard** page, change the name of the cube to **Analysis Services Tutorial**. In the Preview pane, you can see the **InternetSales** measure group and its measures. You can also see the **Date**, **Customer,** and **Product** dimensions.
13. Click **Finish** to complete the wizard.

In Solution Explorer, in the Analysis Services Tutorial project, the Analysis Services Tutorial cube appears in the **Cubes** folder, and the Customer and Product database dimensions appear in the **Dimensions** folder. Additionally, in the center of the development environment, the Cube Structure tab displays the Analysis Services Tutorial cube.

14. On the toolbar of the Cube Structure tab, change the **Zoom** level to 50 percent, so that you can more easily see the dimensions and fact tables in the cube. Notice that the fact table is yellow and the dimension tables are blue.

15. On the **File** menu, click **Save All**.

Next Task in Lesson

Adding Attributes to Dimensions

See Also

Cubes in Multidimensional Models
Dimensions in Multidimensional Models

Lesson 2-3 - Adding Attributes to Dimensions

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Now that you have defined dimensions, you can populate them with attributes that represent each data element in the dimension. Attributes are commonly based on fields from a data source view. When adding attributes to a dimension, you can include fields from any table in the data source view.

In this task, you will use Dimension Designer to add attributes to the Customer and Product dimensions. The Customer dimension will include attributes based on fields from both the Customer and Geography tables.

Adding Attributes to the Customer Dimension

To add attributes

1. Open Dimension Designer for the Customer dimension. To do this, double-click the **Customer** dimension in the **Dimensions** node of Solution Explorer.
2. In the **Attributes** pane, notice the Customer Key and Geography Key attributes that were created by the Cube Wizard.
3. On the toolbar of the **Dimension Structure** tab, make sure the Zoom icon to view the tables in the **Data Source View** pane is set at 100 percent.
4. Drag the following columns from the **Customer** table in the **Data Source View** pane to the **Attributes** pane:
   - **BirthDate**
   - **MaritalStatus**
   - **Gender**
   - **EmailAddress**
   - **YearlyIncome**
   - **TotalChildren**
   - **NumberChildrenAtHome**
   - **EnglishEducation**
   - **EnglishOccupation**
   - **HouseOwnerFlag**
   - **NumberCarsOwned**
   - **Phone**
   - **DateFirstPurchase**
   - **CommuteDistance**
5. Drag the following columns from the **Geography** table in the **Data Source View** pane to the **Attributes** pane:

- **City**
- **StateProvinceName**
- **EnglishCountryRegionName**
- **PostalCode**

6. On the File menu, click **Save All**.

Adding Attributes to the Product Dimension

To add attributes

1. Open Dimension Designer for the Product dimension. Double-click the **Product** dimension in Solution Explorer.
2. In the **Attributes** pane, notice the Product Key attribute that was created by the Cube Wizard.
3. On the toolbar of the **Dimension Structure** tab, make sure the Zoom icon to view the tables in the **Data Source View** pane is set at 100 percent.
4. Drag the following columns from the **Product** table in the **Data Source View** pane to the **Attributes** pane:
   - **StandardCost**
   - **Color**
   - **SafetyStockLevel**
   - **ReorderPoint**
   - **ListPrice**
   - **Size**
   - **SizeRange**
   - **Weight**
   - **DaysToManufacture**
   - **ProductLine**
   - **DealerPrice**
   - **Class**
   - **Style**
   - **ModelName**
   - **StartDate**
   - **EndDate**
   - **Status**
5. On the File menu, click **Save All**.

Next Task in Lesson

Reviewing Cube and Dimension Properties

See Also

Dimension Attribute Properties Reference

Lesson 2-4 - Reviewing Cube and Dimension Properties

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you have defined a cube, you can review the results by using Cube Designer. In the following task, you review the structure of the cube in the Analysis Services Tutorial project.

To review cube and dimension properties in Cube Designer

1. To open the Cube Designer, double-click the **Analysis Services Tutorial** cube in the **Cubes** node of Solution Explorer.

2. In the **Measures** pane of the **Cube Structure** tab in Cube Designer, expand the **Internet Sales** measure group to reveal the defined measures.

   You can change the order by dragging the measures into the order that you want. The order you create affects how certain client applications order these measures. The measure group and each measure that it contains have properties that you can edit in the Properties window.

3. In the **Dimensions** pane of the **Cube Structure** tab in Cube Designer, review the cube dimensions that are in the Analysis Services Tutorial cube.

   Notice that although only three dimensions were created at the database level, as displayed in Solution Explorer, there are five cube dimensions in the Analysis Services Tutorial cube. The cube contains more dimensions than the database because the Date database dimension is used as the basis for three separate date-related cube dimensions, based on different date-related facts in the fact table. These date-related dimensions are also called *role playing dimensions*. The three date-related cube dimensions let users dimension the cube by three separate facts that are related to each product sale: the product order date, the due date for fulfillment of the order, and the ship date for the order. By reusing a single database dimension for multiple cube dimensions, Analysis Services simplifies dimension management, uses less disk space, and reduces overall processing time.

4. In the **Dimensions** pane of the **Cube Structure** tab, expand **Customer**, and then click **Edit Customer** to open the dimension in Dimension Designer.

   Dimension Designer contains these tabs: **Dimension Structure**, **Attribute Relationships**, **Translations**, and **Browser**. Notice that the **Dimension Structure** tab includes three panes: **Attributes**, **Hierarchies**, and **Data Source View**. The attributes that the dimension contains appear in the **Attributes** pane. For more information, see Dimension Attribute Properties Reference, Create User-Defined Hierarchies, and Define Attribute Relationships.

5. To switch to Cube Designer, right-click the **Analysis Services Tutorial** cube in the **Cubes** node in Solution Explorer, and then click **View Designer**.

6. In Cube Designer, click the **Dimension Usage** tab.

   In this view of the Analysis Services Tutorial cube, you can see the cube dimensions that are used by the Internet Sales measure group. Also, you can define the type of relationship between each dimension and each measure group in which it is used.

7. Click the **Partitions** tab.

The Cube Wizard defines a single partition for the cube, by using the multidimensional online analytical processing (MOLAP) storage mode without aggregations. With MOLAP, all leaf-level data and all aggregations are stored within the cube for maximum performance. Aggregations are precalculated summaries of data that improve query response time by having answers ready before questions are asked. You can define additional partitions, storage settings, and writeback settings on the **Partitions** tab. For more information, see Partitions (Analysis Services - Multidimensional Data), Aggregations and Aggregation Designs.

8. Click the **Browser** tab.

   Notice that the cube cannot be browsed because it has not yet been deployed to an instance of Analysis Services. At this point, the cube in the Analysis Services Tutorial project is just a definition of a cube, which you can deploy to any instance of Analysis Services. When you deploy and process a cube, you create the defined objects in an instance of Analysis Services and populate the objects with data from the underlying data sources.

9. In Solution Explorer, right-click **Analysis Services Tutorial** in the **Cubes** node, and then click **View Code**. You might need to wait.

   The XML code for the Analysis Services Tutorial cube is displayed on the **Analysis Services Tutorial.cube [XML]** tab. This is the actual code that is used to create the cube in an instance of Analysis Services during deployment. For more information, see View the XML for an Analysis Services Project (SSDT).

10. Close the XML code tab.

Next Task in Lesson

Deploying an Analysis Services Project

See Also

Browse Dimension Data in Dimension Designer

Lesson 2-5 - Deploying an Analysis Services Project

11/28/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

To view the cube and dimension data for the objects in the Analysis Services Tutorial cube in the Analysis Services Tutorial project, you must deploy the project to a specified instance of Analysis Services and then process the cube and its dimensions. *Deploying* an Analysis Services project creates the defined objects in an instance of Analysis Services. *Processing* the objects in an instance of Analysis Services copies the data from the underlying data sources into the cube objects. For more information, see Deploy Analysis Services Projects (SSDT) and Configure Analysis Services Project Properties (SSDT).

At this point in the development process, you generally deploy the cube to an instance of Analysis Services on a development server. Once you have finished developing your business intelligence project, you will generally use the Analysis Services Deployment Wizard to deploy your project from the development server to a production server. For more information, see Multidimensional Model Solution Deployment and Deploy Model Solutions Using the Deployment Wizard.

In the following task, you review the deployment properties of the Analysis Services Tutorial project and then deploy the project to your local instance of Analysis Services.

To deploy the Analysis Services project

1. In Solution Explorer, right-click the **Analysis Services Tutorial** project, and then click **Properties**.

   The **Analysis Services Tutorial Property Pages** dialog box appears and displays the properties of the Active(Development) configuration. You can define multiple configurations, each with different properties. For example, a developer might want to configure the same project to deploy to different development computers and with different deployment properties, such as database names or processing properties. Notice the value for the **Output Path** property. This property specifies the location in which the XMLA deployment scripts for the project are saved when a project is built. These are the scripts that are used to deploy the objects in the project to an instance of Analysis Services.

2. In the **Configuration Properties** node in the left pane, click **Deployment**.

   Review the deployment properties for the project. By default, the Analysis Services Project template configures an Analysis Services project to incrementally deploy all projects to the default instance of Analysis Services on the local computer, to create an Analysis Services database with the same name as the project, and to process the objects after deployment by using the default processing option. For more information, see Configure Analysis Services Project Properties (SSDT).

   > **NOTE**
   >
   > If you want to deploy the project to a named instance of Analysis Services on the local computer, or to an instance on a remote server, change the **Server** property to the appropriate instance name, such as *<ServerName*>\ <**InstanceName**>*.

3. Click **OK**.

4. In Solution Explorer, right-click the **Analysis Services Tutorial** project, and then click **Deploy**. You might need to wait.

> **NOTE**
> If you get errors during deployment, use SQL Server Management Studio to check the database permissions. The account you specified for the data source connection must have a login on the SQL Server instance. Double-click the login to view User Mapping properties. The account must have db_datareader permissions on the **AdventureWorksDW2012** database.

SQL Server Data Tools (SSDT) builds and then deploys the Analysis Services Tutorial project to the specified instance of Analysis Services by using a deployment script. The progress of the deployment is displayed in two windows: the **Output** window and the **Deployment Progress - Analysis Services Tutorial** window.

Open the Output window, if necessary, by clicking **Output** on the **View** menu. The **Output** window displays the overall progress of the deployment. The **Deployment Progress - Analysis Services Tutorial** window displays the detail about each step taken during deployment. For more information, see Build Analysis Services Projects (SSDT) and Deploy Analysis Services Projects (SSDT).

5. Review the contents of the **Output** window and the **Deployment Progress - Analysis Services Tutorial** window to verify that the cube was built, deployed, and processed without errors.
6. To hide the **Deployment Progress - Analysis Services Tutorial** window, click the **Auto Hide** icon (it looks like a pushpin) on the toolbar of the window.
7. To hide the **Output** window, click the **Auto Hide** icon on the toolbar of the window.

You have successfully deployed the Analysis Services Tutorial cube to your local instance of Analysis Services, and then processed the deployed cube.

Next Task in Lesson

Browsing the Cube

See Also

Deploy Analysis Services Projects (SSDT)
Configure Analysis Services Project Properties (SSDT)

Lesson 2-6 - Browsing the Cube

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you deploy a cube, the cube data is viewable on the **Browser** tab in Cube Designer, and the dimension data is viewable on the **Browser** tab in Dimension Designer. Browsing cube and dimension data is way to check your work incrementally. You can verify that small changes to properties, relationships, and other objects have the desired effect once the object is processed. While the Browser tab is used to view both cube and dimension data, the tab provides different capabilities based on the object you are browsing.

For dimensions, the Browser tab provides a way to view members or navigate a hierarchy all the way down to the leaf node. You can browse dimension data in different languages, assuming you have added the translations to your model.

For cubes, the Browser tab provides two approaches for exploring data. You can use the built-in MDX Query Designer to build queries that return a flattened rowset from a multidimensional database. Alternatively, you can use an Excel shortcut. When you start Excel from within SQL Server Data Tools, Excel opens with a PivotTable already in the worksheet and a predefined connection to the model workspace database.

Excel generally offers a better browsing experience because you can explore cube data interactively, using horizontal and vertical axes to analyze the relationships in your data. In contrast, the MDX Query Designer is limited to a single axis. Moreover, because the rowset is flattened, you do not get the drilldown that an Excel PivotTable provides. As you add more dimensions and hierarchies to your cube, which you will do in subsequent lessons, Excel will be the preferred solution for browsing data.

To browse the deployed cube

1. Switch to **Dimension Designer** for the Product dimension in SQL Server Data Tools (SSDT). To do this, double-click the **Product** dimension in the **Dimensions** node of Solution Explorer.

2. Click the **Browser** tab to display the **All** member of the **Product Key** attribute hierarchy. In lesson three, you will define a user hierarchy for the Product dimension that will let you browse the dimension.

3. Switch to **Cube Designer** in SQL Server Data Tools (SSDT). To do this, double-click the **Analysis Services Tutorial** cube in the **Cubes** node of Solution Explorer.

4. Select the **Browser** tab, and then click the **Reconnect** icon on the toolbar of the designer.

   The left pane of the designer shows the objects in the Analysis Services Tutorial cube. On the right side of the **Browser** tab, there are two panes: the upper pane is the **Filter** pane, and the lower pane is the **Data** pane. In an upcoming lesson, you will use the cube browser to do analysis.

Next Lesson

Lesson 3: Modifying Measures, Attributes and Hierarchies

See Also

MDX Query Editor (Analysis Services - Multidimensional Data)

Lesson 3: Modifying Measures, Attributes and Hierarchies

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After defining your initial cube, you are ready to improve the usefulness and friendliness of the cube. You can do this by adding hierarchies that support navigation and aggregation at various levels, by applying formats to specific measure, and by defining calculations and relationships.

> **NOTE**
> Completed projects for all of the lessons in this tutorial are available online. You can jump ahead to any lesson by using the completed project from the previous lesson as a starting point. Click here to download the sample projects that go with this tutorial.

This lesson contains the following tasks:

Modifying Measures
In this task, you specify formatting properties for the currency and percentage measures in the Analysis Services Tutorial cube.

Modifying the Customer Dimension
In this task, you define a user hierarchy, create named calculations, modify attributes to use named calculations, and group attributes and user hierarchies into display folders.

Modifying the Product Dimension
In this task, you define a user hierarchy, create named calculations, define the All member name, and define display folders.

Modifying the Date Dimension
In this task, you define a user hierarchy, modify attribute member names, and use composite keys to specify unique attribute members.

Browsing the Deployed Cube
In this task, you browse cube data by using the browser in Cube Designer.

See Also

Analysis Services Tutorial Scenario
Multidimensional Modeling (Adventure Works Tutorial)

Lesson 3-1 - Modifying Measures

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can use the **FormatString** property to define formatting settings that control how measures are displayed to users. In this task, you specify formatting properties for the currency and percentage measures in the Analysis Services Tutorial cube.

To modify the measures of the cube

1. Switch to the **Cube Structure** tab of Cube Designer for the Analysis Services Tutorial cube, expand the **Internet Sales** measure group in the **Measures** pane, right-click **Order Quantity**, and then click **Properties**.

2. In the Properties window, click the **Auto Hide** pushpin icon to pin the Properties window open.

   It is easier to change properties for several items in the cube when the Properties window remains open.

3. In the Properties window, click the **FormatString** list, and then type **#,#**.

4. On the toolbar of the **Cube Structure** tab, click the **Show Measures Grid** icon on the left.

   The grid view lets you select multiple measures at the same time.

5. Select the following measures. You can select multiple measures by clicking each while holding down the CTRL key:

   - **Unit Price**
   - **Extended Amount**
   - **Discount Amount**
   - **Product Standard Cost**
   - **Total Product Cost**
   - **Sales Amount**
   - **Tax Amt**
   - **Freight**

6. In the Properties window, in the **FormatString** list, select **Currency**.

7. In the drop-down list at the top of the Properties window (right below the title bar), select the measure **Unit Price Discount Pct**, and then select **Percent** in the **FormatString** list.

8. In the Properties window, change the **Name** property for the **Unit Price Discount Pct** measure to **Unit Price Discount Percentage**.

9. In the **Measures** pane, click **Tax Amt** and change the name of this measure to **Tax Amount**.

10. In the Properties window, click the **Auto Hide** icon to hide the Properties window, and then click **Show Measures Tree** on the toolbar of the **Cube Structure** tab.

11. On the **File** menu, click **Save All**.

Next Task in Lesson

Modifying the Customer Dimension

See Also

Define Database Dimensions
Configure Measure Properties

Lesson 3-2 - Modifying the Customer Dimension

11/28/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

There are many different ways that you can increase the usability and functionality of the dimensions in a cube. In the tasks in this topic, you modify the Customer dimension.

Renaming Attributes

You can change attribute names with the **Dimension Structure** tab of Dimension Designer.

To rename an attribute

1. Switch to **Dimension Designer** for the Customer dimension in SQL Server Data Tools (SSDT). To do this, double-click the **Customer** dimension in the **Dimensions** node of Solution Explorer.
2. In the **Attributes** pane, right-click **English Country Region Name**, and then click **Rename**. Change the name of the attribute to **Country-Region**.
3. Change the names of the following attributes in the same manner:
   - **English Education** attribute - change to **Education**
   - **English Occupation** attribute - change to **Occupation**
   - **State Province Name** attribute - change to **State-Province**
4. On the **File** menu, click **Save All**.

Creating a Hierarchy

You can create a new hierarchy by dragging an attribute from the **Attributes** pane to the **Hierarchies** pane.

To create a hierarchy

1. Drag the **Country-Region** attribute from the **Attributes** pane into the **Hierarchies** pane.
2. Drag the **State-Province** attribute from the **Attributes** pane into the cell in the **Hierarchies** pane, underneath the **Country-Region** level.
3. Drag the **City** attribute from the **Attributes** pane into the cell in the **Hierarchies** pane, underneath the **State-Province** level.
4. In the **Hierarchies** pane of the **Dimension Structure** tab, right-click the title bar of the **Hierarchy** hierarchy, select **Rename**, and then type **Customer Geography**.

   The name of the hierarchy is now **Customer Geography**.
5. On the **File** menu, click **Save All**.

Adding a Named Calculation

You can add a named calculation, which is a SQL expression that is represented as a calculated column, to a table in a data source view. The expression appears and behaves as a column in the table. Named calculations let you extend the relational schema of existing tables in a data source view without modifying the table in the underlying data source. For more information, see Define Named Calculations in a Data Source View (Analysis Services)

To add a named calculation

1. Open the **Adventure Works DW 2012** data source view by double-clicking it in the **Data Source Views** folder in Solution Explorer.

2. In the **Tables** pane on the left, right-click **Customer**, and then click **New Named Calculation**.

3. In the **Create Named Calculation** dialog box, type **FullName** in the **Column name** box, and then type or copy and paste the following **CASE** statement in the **Expression** box:

   ```
   CASE
      WHEN MiddleName IS NULL THEN
      FirstName + ' ' + LastName
      ELSE
      FirstName + ' ' + MiddleName + ' ' + LastName
   END
   ```

   The **CASE** statement concatenates the **FirstName**, **MiddleName**, and **LastName** columns into a single column that you will use in the Customer dimension as the displayed name for the **Customer** attribute.

4. Click **OK**, and then expand **Customer** in the **Tables** pane.

   The **FullName** named calculation appears in the list of columns in the Customer table, with an icon that indicates that it is a named calculation.

5. On the **File** menu, click **Save All**.

6. In the **Tables** pane, right-click **Customer**, and then click **Explore Data**.

7. Review the last column in the **Explore Customer Table** view.

   Notice that the **FullName** column appears in the data source view, correctly concatenating data from several columns from the underlying data source and without modifying the original data source.

8. Close the **Explore Customer Table** tab.

Using the Named Calculation for Member Names

After you have created a named calculation in the data source view, you can use the named calculation as a property of an attribute.

To use the named calculation for member names

1. Switch to Dimension Designer for the Customer dimension.

2. In the **Attributes** pane of the **Dimension Structure** tab, click the **Customer Key** attribute.

3. Open the Properties window and click the **Auto Hide** button on the title bar so that it stays open.

4. In the **Name** property field, type **Full Name**.

5. Click in the **NameColumn** property field at the bottom, and then click the browse (**...**) button to open the **Name Column** dialog box.

6. Select **FullName** at the bottom of the **Source column** list, and then click **OK**.

7. In the Dimensions Structure tab, drag the **Full Name** attribute from the **Attributes** pane into the cell in the **Hierarchies** pane, underneath the **City** level.

8. On the **File** menu, click **Save All**.

Defining Display Folders

You can use display folders to group user and attribute hierarchies into folder structures to increase usability.

To define display folders

1. Open the **Dimension Structure** tab for the Customer dimension.
2. In the **Attributes** pane, select the following attributes by holding down the CTRL key while clicking each of them:
   - **City**
   - **Country-Region**
   - **Postal Code**
   - **State-Province**
3. In the Properties window, click the **AttributeHierarchyDisplayFolder** property field at the top (you might need to point to it to see the full name), and then type **Location**.
4. In the **Hierarchies** pane, click **Customer Geography**, and then in the Properties window on the right, select **Location** as the value of the **DisplayFolder** property.
5. In the **Attributes** pane, select the following attributes by holding down the CTRL key while clicking each of them:
   - **Commute Distance**
   - **Education**
   - **Gender**
   - **House Owner Flag**
   - **Marital Status**
   - **Number Cars Owned**
   - **Number Children At Home**
   - **Occupation**
   - **Total Children**
   - **Yearly Income**
6. In the Properties window, click the **AttributeHierarchyDisplayFolder** property field at the top, and then type **Demographic**.
7. In the **Attributes** pane, select the following attributes by holding down the CTRL key while clicking each of them:
   - **Email Address**
   - **Phone**
8. In the Properties window, click the **AttributeHierarchyDisplayFolder** property field and type **Contacts**.
9. On the **File** menu, click **Save All**.

Defining Composite KeyColumns

The **KeyColumns** property contains the column or columns that represent the key for the attribute. In this lesson, you create a composite key for the **City** and **State-Province** attributes. Composite keys can be helpful when you

need to uniquely identify an attribute. For example, when you define attribute relationships later in this tutorial, a **City** attribute must uniquely identify a **State-Province** attribute. However, there could be several cities with the same name in different states. For this reason, you will create a composite key that is composed of the **StateProvinceName** and **City** columns for the **City** attribute. For more information, see Modify the KeyColumn Property of an Attribute.

To define composite KeyColumns for the City attribute

1. Open the **Dimension Structure** tab for the Customer dimension.
2. In the **Attributes** pane, click the **City** attribute.
3. In the **Properties** window, click in the **KeyColumns** field near the bottom, and then click the browse (**...**) button.
4. In the **Key Columns** dialog box, in the **Available Columns** list, select the column **StateProvinceName**, and then click the **>** button.

   The **City** and **StateProvinceName** columns are now displayed in the **Key Columns** list.
5. Click **OK**.
6. To set the **NameColumn** property of the **City** attribute, click the **NameColumn** field in the Properties window, and then click the browse (**...**) button.
7. In the **Name Column** dialog box, in the **Source column** list, select **City**, and then click **OK**.
8. On the **File** menu, click **Save All**.

To define composite KeyColumns for the State-Province attribute

1. Make sure the **Dimension Structure** tab for the Customer dimension is open.
2. In the **Attributes** pane, click the **State-Province** attribute.
3. In the **Properties** window, click in the **KeyColumns** field, and then click the browse (**...**) button.
4. In the **Key Columns** dialog box, in the **Available Columns** list, select the column **EnglishCountryRegionName**, and then click the **>** button.

   The **EnglishCountryRegionName** and **StateProvinceName** columns are now displayed in the **Key Columns** list.
5. Click **OK**.
6. To set the **NameColumn** property of the **State-Province** attribute, click the **NameColumn** field in the Properties window, and then click the browse (**...**) button.
7. In the **Name Column** dialog box, in the **Source column** list, select **StateProvinceName**, and then click **OK**.
8. On the **File** menu, click **Save All**.

Defining Attribute Relationships

If the underlying data supports it, you should define attribute relationships between attributes. Defining attribute relationships speeds up dimension, partition, and query processing. For more information, see Define Attribute Relationships and Attribute Relationships.

To define attribute relationships

1. In the **Dimension Designer** for the Customer dimension, click the **Attribute Relationships** tab. You might need to wait.

2. In the diagram, right-click the **City** attribute, and then click **New Attribute Relationship**.
3. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **City**. Set the **Related Attribute** to **State-Province**.
4. In the **Relationship type** list, set the relationship type to **Rigid**.

   The relationship type is **Rigid** because relationships between the members will not change over time. For example, it would be unusual for a city to become part of a different state or province.
5. Click **OK**.
6. In the diagram, right-click the **State-Province** attribute and then select **New Attribute Relationship**.
7. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **State-Province**. Set the **Related Attribute** to **Country-Region**.
8. In the **Relationship type** list, set the relationship type to **Rigid**.
9. Click **OK**.
10. On the **File** menu, click **Save All**.

Deploying Changes, Processing the Objects, and Viewing the Changes

After you have changed attributes and hierarchies, you must deploy the changes and reprocess the related objects before you can view the changes.

To deploy the changes, process the objects, and view the changes

1. On the **Build** menu of SQL Server Data Tools, click **Deploy Analysis Services Tutorial**.
2. After you receive the **Deployment Completed Successfully** message, click the **Browser** tab of Dimension Designer for the Customer dimension, and then click the Reconnect button on the left side of the toolbar of the designer.
3. Verify that **Customer Geography** is selected in the **Hierarchy** list, and then in the browser pane, expand **All**, expand **Australia**, expand **New South Wales**, and then expand **Coffs Harbour**.

   The browser displays the customers in the city.
4. Switch to **Cube Designer** for the Analysis Services Tutorial cube. To do this, double-click the **Analysis Services Tutorial** cube in the **Cubes** node of **Solution Explorer**.
5. Click the **Browser** tab, and then click the Reconnect button on the toolbar of the designer.
6. In the **Measure Group** pane, expand **Customer**.

   Notice that instead of a long list of attributes, only the display folders and the attributes that do not have display folder values appear underneath Customer.
7. On the **File** menu, click **Save All**.

Next Task in Lesson

Modifying the Product Dimension

See Also

Dimension Attribute Properties Reference
Remove an Attribute from a Dimension
Rename an Attribute

Define Named Calculations in a Data Source View (Analysis Services)

Lesson 3-3 - Modifying the Product Dimension

11/28/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In the tasks in this topic, you use a named calculation to provide more descriptive names for the product lines, define a hierarchy in the Product dimension, and specify the (All) member name for the hierarchy. You also group attributes into display folders.

Adding a Named Calculation

You can add a named calculation to a table in a data source view. In the following task, you create a named calculation that displays the full product line name.

To add a named calculation

1. To open the **Adventure Works DW 2012** data source view, double-click **Adventure Works DW 2012** in the **Data Source Views** folder in Solution Explorer.
2. In the bottom of the diagram pane, right-click the **Product** table header, and then click **New Named Calculation**.
3. In the **Create Named Calculation** dialog box, type **ProductLineName** in the **Column name** box.
4. In the **Expression** box, type or copy and paste the following **CASE** statement:

   ```
   CASE ProductLine
      WHEN 'M' THEN 'Mountain'
      WHEN 'R' THEN 'Road'
      WHEN 'S' THEN 'Accessory'
      WHEN 'T' THEN 'Touring'
      ELSE 'Components'
   END
   ```

   This **CASE** statement creates user-friendly names for each product line in the cube.
5. Click **OK** to create the **ProductLineName** named calculation. You might need to wait.
6. On the **File** menu, click **Save All**.

Modifying the NameColumn Property of an Attribute

To modify the NameColumn property value of an attribute

1. Switch to Dimension Designer for the Product dimension. To do this, double-click the **Product** dimension in the **Dimensions** node of Solution Explorer.
2. In the **Attributes** pane of the **Dimension Structure** tab, select **Product Line**.
3. In the Properties window on the right side of the screen, click the **NameColumn** property field at the bottom of the window, and then click the browse (**...**) button to open the **Name Column** dialog box. (You might need to click the **Properties** tab on the right side of the screen to open the Properties window.
4. Select **ProductLineName** at the bottom of the **Source column** list, and then click **OK**.

   The NameColumn field now contains the text, **Product.ProductLineName (WChar)**. The members of the **Product Line** attribute hierarchy now display the full name of the product line instead of an abbreviated

product line name.

5. In the **Attributes** pane of the **Dimension Structure** tab, select **Product Key**.
6. In the Properties window, click the **NameColumn** property field, and then click the ellipsis browse (**...**) button to open the **Name Column** dialog box.
7. Select **EnglishProductName** in the **Source column** list, and then click **OK**.

   The NameColumn field now contains the text, **Product.EnglishProductName (WChar)**.
8. In the Properties window, scroll up, click the **Name** property field, and then type **Product Name**.

Creating a Hierarchy

To create a hierarchy

1. Drag the **Product Line** attribute from the **Attributes** pane into the **Hierarchies** pane.
2. Drag the **Model Name** attribute from the **Attributes** pane into the cell in the **Hierarchies** pane, underneath the **Product Line** level.
3. Drag the **Product Name** attribute from the **Attributes** pane into the cell in the **Hierarchies** pane, underneath the **Model Name** level. (You renamed Product Key to Product Name in the previous section.)
4. In the **Hierarchies** pane of the **Dimension Structure** tab, right-click the title bar of the **Hierarchy** hierarchy, click **Rename**, and then type **Product Model Lines**.

   The name of the hierarchy is now **Product Model Lines**.
5. On the **File** menu, click **Save All**.

Specifying Folder Names and All Member Names

To specify the folder and member names

1. In the **Attributes** pane, select the following attributes by holding down the CTRL key while clicking each of them:
   - **Class**
   - **Color**
   - **Days To Manufacture**
   - **Reorder Point**
   - **Safety Stock Level**
   - **Size**
   - **Size Range**
   - **Style**
   - **Weight**
2. In the **AttributeHierarchyDisplayFolder** property field in the Properties window, type **Stocking**.

   You have now grouped these attributes into a single display folder.
3. In the **Attributes** pane, select the following attributes:
   - **Dealer Price**

- **List Price**
- **Standard Cost**

4. In the **AttributeHierarchyDisplayFolder** property cell in the Properties window, type **Financial**.

   You have now grouped these attributes into a second display folder.

5. In the **Attributes** pane, select the following attributes:

   - **End Date**
   - **Start Date**
   - **Status**

6. In the **AttributeHierarchyDisplayFolder** property cell in the Properties window, type **History**.

   You have now grouped these attributes into a third display folder.

7. Select the **Product Model Lines** hierarchy in the **Hierarchies** pane, and then change the **AllMemberName** property in the Properties window to **All Products**.

8. Click an open area of the **Hierarchies** pane, and then change the **AttributeAllMemberName** property at the top of the Properties window to **All Products**.

   Clicking an open area lets you modify properties of the Product dimension itself. You could also click **Product** at the top of the attributes list in the **Attributes** pane.

9. On the **File** menu, click **Save All**.

Defining Attribute Relationships

If the underlying data supports it, you should define attribute relationships between attributes. Defining attribute relationships speeds up dimension, partition, and query processing. For more information, see Define Attribute Relationships and Attribute Relationships.

To define attribute relationships

1. In the **Dimension Designer** for the Product dimension, click the **Attribute Relationships** tab.

2. In the diagram, right-click the **Model Name** attribute, and then click **New Attribute Relationship**.

3. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Model Name**. Set the **Related Attribute** to **Product Line**.

   In the **Relationship type** list, leave the relationship type set to **Flexible** because relationships between the members might change over time. For example, a product model might eventually be moved to a different product line.

4. Click **OK**.

5. On the **File** menu, click **Save All**.

Reviewing Product Dimension Changes

To review the Product dimension changes

1. On the **Build** menu of SQL Server Data Tools (SSDT), click **Deploy Analysis Services Tutorial**.

2. After you have received the **Deployment Completed Successfully** message, click the **Browser** tab of **Dimension Designer** for the **Product** dimension, and then click the Reconnect button on the toolbar of the designer.

3. Verify that **Product Model Lines** is selected in the **Hierarchy** list, and then expand **All Products**.

   Notice that the name of the **All** member appears as **All Products**. This is because you changed the **AllMemberName** property for the hierarchy to **All Products** earlier in the lesson. Also, the members of the **Product Line** level now have user-friendly names, instead of single-letter abbreviations.

Next Task in Lesson

Modifying the Date Dimension

See Also

Define Named Calculations in a Data Source View (Analysis Services)
Create User-Defined Hierarchies
Configure the (All) Level for Attribute Hierarchies

Lesson 3-4 - Modifying the Date Dimension

11/28/2018 • 11 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In the tasks in this topic, you create a user-defined hierarchy and change the member names that are displayed for the Date, Month, Calendar Quarter, and Calendar Semester attributes. You also define composite keys for attributes, control the sort order of dimension members, and define attribute relationships.

Adding a Named Calculation

You can add a named calculation, which is a SQL expression that is represented as a calculated column, to a table in a data source view. The expression appears and behaves as a column in the table. Named calculations enable you to extend the relational schema of existing tables in a data source view without modifying the table in the underlying data source. For more information, see Define Named Calculations in a Data Source View (Analysis Services)

To add a named calculation

1. To open the **Adventure Works DW 2012** data source view, double-click it in the **Data Source Views** folder in Solution Explorer.

2. Near the bottom of the **Tables** pane, right-click **Date**, and then click **New Named Calculation**.

3. In the **Create Named Calculation** dialog box, type **SimpleDate** in the **Column name** box, and then type or copy and paste the following **DATENAME** statement in the **Expression** box:

   ```
   DATENAME(mm, FullDateAlternateKey) + ' ' +
   DATENAME(dd, FullDateAlternateKey) + ', ' +
   DATENAME(yy, FullDateAlternateKey)
   ```

   The **DATENAME** statement extracts the year, month, and day values from the FullDateAlternateKey column. You will use this new column as the displayed name for the FullDateAlternateKey attribute.

4. Click **OK**, and then expand **Date** in the **Tables** pane.

   The **SimpleDate** named calculation appears in the list of columns in the Date table, with an icon that indicates that it is a named calculation.

5. On the **File** menu, click **Save All**.

6. In the **Tables** pane, right-click **Date**, and then click **Explore Data**.

7. Scroll to the right to review the last column in the **Explore Date Table** view.

   Notice that the **SimpleDate** column appears in the data source view, correctly concatenating data from several columns from the underlying data source, without modifying the original data source.

8. Close the **Explore Date Table** view.

Using the Named Calculation for Member Names

After you create a named calculation in the data source view, you can use the named calculation as a property of an attribute.

To use the named calculation for member names

1. Open **Dimension Designer** for the Date dimension in SQL Server Data Tools (SSDT). To do this, double-click the **Date** dimension in the **Dimensions** node of **Solution Explorer**.
2. In the **Attributes** pane of the **Dimension Structure** tab, click the **Date Key** attribute.
3. If the Properties window is not open, open the Properties window, and then click the **Auto Hide** button on the title bar so that it stays open.
4. Click the **NameColumn** property field near the bottom of the window, and then click the ellipsis browse (**...**) button to open the **Name Column** dialog box.
5. Select **SimpleDate** at the bottom of the **Source column** list, and then click **OK**.
6. On the **File** menu, click **Save All**.

Creating a Hierarchy

You can create a new hierarchy by dragging an attribute from the **Attributes** pane to the **Hierarchies** pane.

To create a hierarchy

1. In **Dimension Structure** tab of the Dimension Designer for the **Date** dimension, drag the **Calendar Year** attribute from the **Attributes** pane into the **Hierarchies** pane.
2. Drag the **Calendar Semester** attribute from the **Attributes** pane into the cell in the **Hierarchies** pane, underneath the **Calendar Year** level.
3. Drag the **Calendar Quarter** attribute from the **Attributes** pane into the cell in the **Hierarchies** pane, underneath the **Calendar Semester** level.
4. Drag the **English Month Name** attribute from the **Attributes** pane into the cell in the **Hierarchies** pane, underneath the **Calendar Quarter** level.
5. Drag the **Date Key** attribute from the **Attributes** pane into the cell in the **Hierarchies** pane, underneath the **English Month Name** level.
6. In the **Hierarchies** pane, right-click the title bar of the **Hierarchy** hierarchy, click **Rename**, and then type **Calendar Date**.
7. By using the right-click context menu, in the **Calendar Date** hierarchy, rename the **English Month Name** level to **Calendar Month**, and then rename the **Date Key** level to **Date**.
8. Delete the **Full Date Alternate Key** attribute from the **Attributes** pane because you will not be using it. Click **OK** in the **Delete Objects** confirmation window.
9. On the **File** menu, click **Save All**.

Defining Attribute Relationships

If the underlying data supports it, you should define attribute relationships between attributes. Defining attribute relationships speeds up dimension, partition, and query processing.

To define attribute relationships

1. In the **Dimension Designer** for the **Date** dimension, click the **Attribute Relationships** tab.
2. In the diagram, right-click the **English Month Name** attribute, and then click **New Attribute Relationship**.
3. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **English Month Name**. Set the **Related Attribute** to **Calendar Quarter**.

4. In the **Relationship type** list, set the relationship type to **Rigid**.

   The relationship type is **Rigid** because relationships between the members will not change over time.

5. Click **OK**.

6. In the diagram, right-click the **Calendar Quarter** attribute, and then click **New Attribute Relationship**.

7. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Calendar Quarter**. Set the **Related Attribute** to **Calendar Semester**.

8. In the **Relationship type** list, set the relationship type to **Rigid**.

9. Click **OK**.

10. In the diagram, right-click the **Calendar Semester** attribute, and then click **New Attribute Relationship**.

11. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Calendar Semester**. Set the **Related Attribute** to **Calendar Year**.

12. In the **Relationship type** list, set the relationship type to **Rigid**.

13. Click **OK**.

14. On the **File** menu, click **Save All**.

Providing Unique Dimension Member Names

In this task, you will create user-friendly name columns that will be used by the **EnglishMonthName**, **CalendarQuarter**, and **CalendarSemester** attributes.

To provide unique dimension member names

1. To switch to the **Adventure Works DW 2012** data source view, double-click it in the **Data Source Views** folder in Solution Explorer.

2. In the **Tables** pane, right-click **Date**, and then click **New Named Calculation**.

3. In the **Create Named Calculation** dialog box, type **MonthName** in the **Column name** box, and then type or copy and paste the following statement in the **Expression** box:

   ```
   EnglishMonthName+' '+ CONVERT(CHAR (4), CalendarYear)
   ```

   The statement concatenates the month and year for each month in the table into a new column.

4. Click **OK**.

5. In the **Tables** pane, right-click **Date**, and then click **New Named Calculation**.

6. In the **Create Named Calculation** dialog box, type **CalendarQuarterDesc** in the **Column name** box, and then type or copy and paste the following SQL script in the **Expression** box:

   ```
   'Q' + CONVERT(CHAR (1), CalendarQuarter) +' '+ 'CY ' +
   CONVERT(CHAR (4), CalendarYear)
   ```

   This SQL script concatenates the calendar quarter and year for each quarter in the table into a new column.

7. Click **OK**.

8. In the **Tables** pane, right-click **Date**, and then click **New Named Calculation**.

9. In the **Create Named Calculation** dialog box, type **CalendarSemesterDesc** in the **Column name** box, and then type or copy and paste the following SQL script in the **Expression** box:

```
CASE
WHEN CalendarSemester = 1 THEN 'H1' + ' ' + 'CY' + ' '
       + CONVERT(CHAR(4), CalendarYear)
ELSE
'H2' + ' ' + 'CY' + ' ' + CONVERT(CHAR(4), CalendarYear)
END
```

   This SQL script concatenates the calendar semester and year for each semester in the table into a new column.

10. Click **OK.**

11. On the **File** menu, click **Save All**.

Defining Composite KeyColumns and Setting the Name Column

The **KeyColumns** property contains the column or columns that represent the key for the attribute. In this task, you will define composite **KeyColumns**.

To define composite KeyColumns for the English Month Name attribute

1. Open the **Dimension Structure** tab for the Date dimension.

2. In the **Attributes** pane, click the **English Month Name** attribute.

3. In the **Properties** window, click the **KeyColumns** field, and then click the browse (**...**) button.

4. In the **Key Columns** dialog box, in the **Available Columns** list, select the column **CalendarYear**, and then click the **>** button.

5. The **EnglishMonthName** and **CalendarYear** columns are now displayed in the **Key Columns** list.

6. Click **OK**.

7. To set the **NameColumn** property of the **EnglishMonthName** attribute, click the **NameColumn** field in the Properties window, and then click the browse (**...**) button.

8. In the **Name Column** dialog box, in the **Source Column** list, select **MonthName**, and then click **OK**.

9. On the **File** menu, click **Save All**.

To define composite KeyColumns for the Calendar Quarter attribute

1. In the **Attributes** pane, click the **Calendar Quarter** attribute.

2. In the **Properties** window, click the **KeyColumns** field, and then click the browse (**...**) button.

3. In the **Key Columns** dialog box, in the **Available Columns** list, select the column **CalendarYear**, and then click the **>** button.

   The **CalendarQuarter** and **CalendarYear** columns are now displayed in the **Key Columns** list.

4. Click **OK**.

5. To set the **NameColumn** property of the **Calendar Quarter** attribute, click the **NameColumn** field in the Properties window, and then click the browse (**...**) button.

6. In the **Name Column** dialog box, in the **Source Column** list, select **CalendarQuarterDesc**, and then click **OK**.

7. On the **File** menu, click **Save All**.

To define composite KeyColumns for the Calendar Semester attribute

1. In the **Attributes** pane, click the **Calendar Semester** attribute.
2. In the **Properties** window, click the **KeyColumns** field, and then click the browse (**...**) button.
3. In the **Key Columns** dialog box, in the **Available Columns** list, select the column, **CalendarYear**, and then click the **>** button.

   The **CalendarSemester** and **CalendarYear** columns are now displayed in the **Key Columns** list.
4. Click **OK**.
5. To set the **NameColumn** property of the **Calendar Semester** attribute, click the **NameColumn** field in the property window, and then click the browse (**...**) button.
6. In the **Name Column** dialog box, in the **Source Column** list, select **CalendarSemesterDesc**, and then click **OK**.
7. On the **File** menu, click **Save All**.

Deploying and Viewing the Changes

After you have changed attributes and hierarchies, you must deploy the changes and reprocess the related objects before you can view the changes.

To deploy and view the changes

1. On the **Build** menu of SQL Server Data Tools, click **Deploy Analysis Services Tutorial**.
2. After you have received the **Deployment Completed Successfully** message, click the **Browser** tab of **Dimension Designer** for the **Date** dimension, and then click the Reconnect button on the toolbar of the designer.
3. Select **Calendar Quarter** from the **Hierarchy** list. Review the members in the **Calendar Quarter** attribute hierarchy.

   Notice that the names of the members of the **Calendar Quarter** attribute hierarchy are clearer and easier to use because you created a named calculation to use as the name. Members now exist in the **Calendar Quarter** attribute hierarchy for each quarter in each year. The members are not sorted in chronological order. Instead they are sorted by quarter and then by year. In the next task in this topic, you will modify this behavior to sort the members of this attribute hierarchy by year and then by quarter.
4. Review the members of the **English Month Name** and **Calendar Semester** attribute hierarchies.

   Notice that the members of these hierarchies are also not sorted in chronological order. Instead, they are sorted by month or semester, respectively, and then by year. In the next task in this topic, you will modify this behavior to change this sort order.

Changing the Sort Order by Modifying Composite Key Member Order

In this task, you will change the sort order by changing the order of the keys that make up the composite key.

To modify the composite key member order

1. Open the **Dimension Structure** tab of Dimension Designer for the **Date** dimension, and then select **Calendar Semester** in the **Attributes** pane.
2. In the Properties window, review the value for the **OrderBy** property. It is set to **Key**.

   The members of the **Calendar Semester** attribute hierarchy are sorted by their key value. With a

composite key, the ordering of the member keys is based first on the value of the first member key, and then on the value of the second member key. In other words, the members of the **Calendar Semester** attribute hierarchy are sorted first by semester and then by year.

3. In the Properties window, click the ellipsis browse button (**...**) to change the **KeyColumns** property value.

4. In the **Key Columns** list of the **Key Columns** dialog box, verify that **CalendarSemester** is selected, and then click the down arrow to reverse the order of the members of this composite key. Click **OK**.

   The members of the attribute hierarchy are now sorted first by year and then by semester.

5. Select **Calendar Quarter** in the **Attributes** pane, and then click the ellipsis browse button (**...**) for the **KeyColumns** property in the Properties window.

6. In the **Key Columns** list of the **Key Columns** dialog box, verify that **CalendarQuarter** is selected, and then click the down arrow to reverse the order of the members of this composite key. Click **OK**.

   The members of the attribute hierarchy are now sorted first by year and then by quarter.

7. Select **English Month Name** in the **Attributes** pane, and then click the ellipsis button (**...**) for the **KeyColumns** property in the Properties window.

8. In the **Key Columns** list of the **Key Columns** dialog box, verify that **EnglishMonthName** is selected, and then click the down arrow to reverse the order of the members of this composite key. Click **OK**.

   The members of the attribute hierarchy are now sorted first by year and then by month.

9. On the **Build** menu of SQL Server Data Tools, click **Deploy Analysis Services Tutorial**. When deployment has successfully completed, click the **Browser** tab in Dimension Designer for the **Date** dimension.

10. On the toolbar of the **Browser** tab, click the Reconnect button.

11. Review the members of the **Calendar Quarter** and **Calendar Semester** attribute hierarchies.

    Notice that the members of these hierarchies are now sorted in chronological order, by year and then by quarter or semester, respectively.

12. Review the members of the **English Month Name** attribute hierarchy.

    Notice that the members of the hierarchy are now sorted first by year and then alphabetically by month. This is because the data type for the EnglishCalendarMonth column in the data source view is a string column, based on the nvarchar data type in the underlying relational database. For information about how to enable the months to be sorted chronologically within each year, see Sorting Attribute Members Based on a Secondary Attribute.

Next Task in Lesson

Browsing the Deployed Cube

See Also

Dimensions in Multidimensional Models

Lesson 3-5 - Browsing the Deployed Cube

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In the following task, you browse the Analysis Services Tutorial cube. Because our analysis compares measure across multiple dimensions, you will use an Excel PivotTable to browse your data. Using a PivotTable lets you place customer, date, and product information on different axes so that you can see how Internet Sales change when viewed across specific time periods, customer demographics, and product lines.

To browse the deployed cube

1. To switch to Cube Designer in SQL Server Data Tools, double-click the **Analysis Services Tutorial** cube in the **Cubes** folder of the Solution Explorer.
2. Open the **Browser** tab, and then click the **Reconnect** button on the toolbar of the designer.
3. Click the Excel icon to launch Excel using the workspace database as the data source. When prompted to enable connections, click **Enable**.
4. In the PivotTable Field List, expand **Internet Sales**, and then drag the **Sales Amount** measure to the **Values** area.
5. In the PivotTable Field List, expand **Product**.
6. Drag the **Product Model Lines** user hierarchy to the **Columns** area.
7. In the PivotTable Field List, expand **Customer**, expand **Location**, and then drag the **Customer Geography** hierarchy from the Location display folder in the Customer dimension to the **Rows** area.
8. In the PivotTable Field List, expand **Order Date**, and then drag the **Order Date.Calendar Date** hierarchy to the **Report Filter** area.
9. Click the arrow to the right of the **Order Date.Calendar Date** filter in the data pane, clear the check box for the **(All)** level, expand **2006**, expand **H1 CY 2006**, expand **Q1 CY 2006**, select the check box for **February 2006**, and then click **OK**.

   Internet sales by region and product line for the month of February, 2006 appear as shown in the following image.

Next Lesson

Lesson 4: Defining Advanced Attribute and Dimension Properties

Lesson 4: Defining Advanced Attribute and Dimension Properties

12/5/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In this lesson, you will learn how to use some of the advanced properties of attributes, attribute hierarchies, and dimension properties.

> **NOTE**
> This lesson is based on an enhanced version of the Analysis Services Tutorial project that you completed in the first three lessons of this tutorial. The first task in this lesson describes where to locate the appropriate sample project to use for the lesson, and the difference between this project and the project that you created in the first three lessons.

This lesson contains the following tasks:

Using a Modified Version of the Analysis Services Tutorial Project
In this task, you open, review, and deploy a modified version of the Analysis Services Tutorial project, which has multiple measure groups and additional dimensions.

Defining Parent Attribute Properties in a Parent-Child Hierarchy
In this task, you define level names in a parent-child dimension and specify whether data related to parent members is displayed. For more information, see Parent-Child Dimensions and Attributes in Parent-Child Hierarchies.

Automatically Grouping Attribute Members
In this task, you automatically create groupings of attribute members based on the distribution of the members within the attribute hierarchy. For more information, see Group Attribute Members (Discretization).

Hiding and Disabling Attribute Hierarchies
In this task, you learn how and when to disable or hide attribute hierarchies.

Sorting Attribute Members Based on a Secondary Attribute
In this task, you learn how to sort dimension members based on a secondary attribute, to achieve the sort order that you want.

Specifying Attribute Relationships Between Attributes in a User-Defined Hierarchy
In this task, you learn how to define member properties for attributes and to specify aggregation relationships between them. For more information, see Define Attribute Relationships and User Hierarchy Properties.

Defining the Unknown Member and Null Processing Properties
In this task, you configure the UnknownMember and UnknownMemberName properties to handle error conditions caused by null dimension members.

Next Lesson

Lesson 5: Defining Relationships Between Dimensions and Measure Groups

See Also

Lesson 4-1 - Using a Modified Version of the Analysis Services Tutorial Project

11/27/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The remaining lessons in this tutorial are based on an enhanced version of the Analysis Services Tutorial project that you completed in the first three lessons. Additional tables and named calculations have been added to the **Adventure Works DW 2012** data source view, additional dimensions have been added to the project, and these new dimensions have been added to the Analysis Services Tutorial cube. In addition, a second measure group has been added, which contains measures from a second fact table. This enhanced project will enable you to continue learning how to add functionality to your business intelligence application without having to repeat the skills you have already learned.

Before you can continue with the tutorial, you must download, extract, load and process the enhanced version of the Analysis Services Tutorial project. Use the instructions in this lesson to ensure you have performed all the steps.

Downloading and Extracting the Project File

1. Click here to go to the download page that provides the sample projects that go with this tutorial. The tutorial projects are included in the **adventure-works-multidimensional-tutorial-projects.zip** download.

2. Click **adventure-works-multidimensional-tutorial-projects.zip** to download the package that contains the projects for this tutorial.

   By default, a .zip file is saved to the Downloads folder. You must move the .zip file to a location that has a shorter path (for example, create a C:\Tutorials folder to store the files). You can then extract the files contained in the .zip file. If you attempt to unzip the files from the Downloads folder, which has a longer path, you will only get Lesson 1.

3. Create a subfolder at or near the root drive, for example, C:\Tutorial.

4. Move the **adventure-works-multidimensional-tutorial-projects.zip** file to the subfolder.

5. Right-click the file and select **Extract All**.

6. Browse to the **Lesson 4 Start** folder to find the **Analysis Services Tutorial.sln** file.

Loading and Processing the Enhanced Project

1. In SQL Server Data Tools, on the **File** menu, click **Close Solution** to close files you won't be using.

2. On the **File** menu, point to **Open**, and then click **Project/Solution**.

3. Browse to the location where you extracted the tutorial project files.

   Find the folder named **Lesson 4 Start**, and then double-click Analysis Services Tutorial.sln.

4. Deploy the enhanced version of the Analysis Services Tutorial project to the local instance of Analysis Services, or to another instance, and verify that processing completes successfully.

Understanding the Enhancements to the Project

The enhanced version of the project is different from the version of the Analysis Services Tutorial project that you completed in the first three lessons. The differences are described in the following sections. Review this information before continuing with the remaining lessons in the tutorial.

Data Source View

The data source view in the enhanced project contains one additional fact table and four additional dimension tables from the **AdventureWorksDW2012** database.

Notice that with ten tables in the data source view, the diagram is becoming crowded. This makes it difficult to easily understand the relationships between the tables and to locate specific tables. To solve this problem, the tables are organized into two logical diagrams, the **Internet Sales** diagram and the **Reseller Sales** diagram. These diagrams are each organized around a single fact table. Creating logical diagrams lets you view and work with a specific subset of the tables in a data source view instead of always viewing all the tables and their relationships in a single diagram.

Internet Sales Diagram

The **Internet Sales** diagram contains the tables that are related to the sale of Adventure Works products directly to customers through the Internet. The tables in the diagram are the four dimension tables and one fact table that you added to the **Adventure Works DW 2012** data source view in Lesson 1. These tables are as follows:

- **Geography**
- **Customer**
- **Date**
- **Product**
- **InternetSales**

Reseller Sales Diagram

The **Reseller Sales** diagram contains the tables that are related to the sale of Adventure Works products by resellers. This diagram contains the following seven dimension tables and one fact table from the **AdventureWorksDW2012** database:

- **Reseller**
- **Promotion**
- **SalesTerritory**
- **Geography**
- **Date**
- **Product**
- **Employee**
- **ResellerSales**

Notice that the **DimGeography**, **DimDate**, and **DimProduct** tables are used in both the **Internet Sales** diagram and the **Reseller Sales** diagram. Dimension tables can be linked to multiple fact tables.

Database and Cube Dimensions

The Analysis Services Tutorial project contains five new database dimensions, and the Analysis Services Tutorial cube contains these same five dimensions as cube dimensions. These dimensions have been defined to have user hierarchies and attributes that were modified by using named calculations, composition member keys, and display

folders. The new dimensions are described in the following list.

Reseller Dimension
The Reseller dimension is based on the **Reseller** table in the **Adventure Works DW 2012** data source view.

Promotion Dimension
The Promotion dimension is based on the **Promotion** table in the **Adventure Works DW 2012** data source view.

Sales Territory Dimension
The Sales Territory dimension is based on the **SalesTerritory** table in the **Adventure Works DW 2012** data source view.

Employee Dimension
The Employee dimension is based on the **Employee** table in the **Adventure Works DW 2012** data source view.

Geography Dimension
The Geography dimension is based on the **Geography** table in the **Adventure Works DW 2012** data source view.

Analysis Services Cube

The **Analysis Services Tutorial** cube now contains two measure groups, the original measure group based on the **InternetSales** table and a second measure group based on the **ResellerSales** table in the **Adventure Works DW 2012** data source view.

Next Task in Lesson

Defining Parent Attribute Properties in a Parent-Child Hierarchy

See Also

Deploying an Analysis Services Project

Lesson 4-2 - Defining Parent Attribute Properties in a Parent-Child Hierarchy

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A parent-child hierarchy is a hierarchy in a dimension that is based on two table columns. Together, these columns define the hierarchical relationships among the members of the dimension. The first column, called the *member key column*, identifies each dimension member. The other column, called the *parent column*, identifies the parent of each dimension member. The **NamingTemplate** property of a parent attribute determines the name of each level in the parent-child hierarchy, and the **MembersWithData** property determines whether data for parent members should be displayed.

For more information, see Parent-Child Dimensions, Attributes in Parent-Child Hierarchies

> **NOTE**
> When you use the Dimension Wizard to create a dimension, the wizard recognizes the tables that have parent-child relationships and automatically defines the parent-child hierarchy for you.

In the tasks in this topic, you will create a naming template that defines the name for each level in the parent-child hierarchy in the **Employee** dimension. You will then configure the parent attribute to hide all parent data, so that only the sales for leaf-level members are displayed.

Browsing the Employee Dimension

1. In Solution Explorer, double-click **Employee.dim** in the **Dimensions** folder to open Dimension Designer for the Employee dimension.
2. Click the **Browser** tab, verify that **Employees** is selected in the **Hierarchy** list, and then expand the **All Employees** member.

   Notice that **Ken J. Sánchez** is the top-level manager in this parent-child hierarchy.
3. Select the **Ken J. Sánchez** member.

   Notice that the level name for this member is **Level 02**. (The level name appears after **Current level:** immediately above the **All Employees** member.) In the next task, you will define more descriptive names for each level.
4. Expand **Ken J. Sánchez** to view the names of the employees who report to this manager, and then select **Brian S. Welcker** to view the name of this level.

   Notice that the level name for this member is **Level 03**.
5. In Solution Explorer, double-click **Analysis Services Tutorial.cube** in the **Cubes** folder to open Cube Designer for the Analysis Services Tutorial cube.
6. Click the **Browser** tab.
7. Click the Excel icon, and then click **Enable** when prompted to enable connections.
8. In the PivotTable Field List, expand **Reseller Sales**. Drag **Reseller Sales-Sales Amount** to the Values area.

9. In the PivotTable Field List, expand **Employee**, and then drag the **Employees** hierarchy to the **Rows** area.

   All the members of the Employees hierarchy are added to column A of the PivotTable report.

   The following image shows the Employees hierarchy expanded.

10.

   Notice that the sales made by each manager in Level 03 are also displayed in Level 04. This is because each manager is also an employee of another manager. In the next task, you will hide these sale amounts.

Modifying Parent Attribute Properties in the Employee Dimension

1. Switch to Dimension Designer for the **Employee** dimension.

2. Click the **Dimension Structure** tab, and then select the **Employees** attribute hierarchy in the **Attributes** pane.

   Notice the unique icon for this attribute. This icon signifies that the attribute is the parent key in a parent-child hierarchy. Notice also, in the Properties window, that the **Usage** property for the attribute is defined as **Parent**. This property was set by the Dimension Wizard when the dimension was designed. The wizard automatically detected the parent-child relationship.

3. In the Properties window, click the ellipsis button (**...**) in the **NamingTemplate** property cell.

   In the **Level Naming Template** dialog box, you define the level naming template that determines the level names in the parent-child hierarchy that are displayed to users as they browse cubes.

4. In the second row, the ***** row, type **Employee Level *** in the **Name** column, and then click the third row.

   Notice under **Result** that each level will now be named "Employee Level" followed by a sequentially increasing number.

The following image shows the changes in the **Level Naming Template** dialog box.

5. Click **OK**.

6. In the Properties window for the **Employees** attribute, in the **MembersWithData** property cell, select **NonLeafDataHidden** to change this value for the **Employees** attribute.

   This will cause data that is related to non-leaf level members in the parent-child hierarchy to be hidden.

Browsing the Employee Dimension with the Modified Attributes

1. On the **Build** menu of SQL Server Data Tools (SSDT), click **Deploy Analysis Services Tutorial**.

2. When deployment has successfully completed, switch to Cube Designer for the Analysis Services Tutorial cube, and then click **Reconnect** on the toolbar of the **Browser** tab.

3. Click the Excel icon, and then click **Enable**.

4. Drag **Reseller Sales-Sales Amount** to the Values area.

5. Drag the **Employees** hierarchy to the Row Labels area.

   The following image shows the changes that you made to the Employees hierarchy. Notice that Stephen Y. Jiang no longer appears as an employee of himself.

Next Task in Lesson

Automatically Grouping Attribute Members

See Also

Parent-Child Dimensions
Attributes in Parent-Child Hierarchies

Lesson 4-3 - Automatically Grouping Attribute Members

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you browse a cube, you typically dimension the members of one attribute hierarchy by the members of another attribute hierarchy. For example, you might group customer sales by city, by product purchased, or by gender. However, with certain types of attributes, it is useful to have Microsoft Analysis Services automatically create groupings of attribute members based on the distribution of the members within an attribute hierarchy. For example, you can have Analysis Services create groups of yearly income values for customers. When you do this, users who browse the attribute hierarchy will see the names and values of the groups instead of the members themselves. This limits the number of levels that are presented to users, which can be more useful for analysis.

The **DiscretizationMethod** property determines whether Analysis Services creates groupings, and determines the type of grouping that is performed. By default, Analysis Services does not perform any groupings. When you enable automatic groupings, you can allow Analysis Services to automatically determine the best grouping method based on the structure of the attribute, or you can choose one of the grouping algorithms in the following list to specify the grouping method:

EqualAreas
Analysis Services creates group ranges so that the total population of dimension members is distributed equally across the groups.

Clusters
Analysis Services creates groups by performing single-dimensional clustering on the input values by using the K-Means clustering method with Gaussian distributions. This option is valid only for numeric columns.

After you specify a grouping method, you must specify the number of groups, by using the **DiscretizationBucketCount** property. For more information, see Group Attribute Members (Discretization)

In the tasks in this topic, you will enable different types of groupings for the following: the yearly income values in the **Customer** dimension; the number of employee sick leave hours in the **Employees** dimension; and the number of employee vacation hours in the **Employees** dimension. You will then process and browse the Analysis Services Tutorial cube to view the effect of the member groups. Finally, you will modify the member group properties to see the effect of the change in grouping type.

Grouping Attribute Hierarchy Members in the Customer Dimension

1. In Solution Explorer, double-click **Customer** in the **Dimensions** folder to open Dimension Designer for the Customer dimension.

2. In the **Data Source View** pane, right-click the **Customer** table, and then click **Explore Data**.

   Notice the range of values for the **YearlyIncome** column. These values become the members of the **Yearly Income** attribute hierarchy, unless you enable member grouping.

3. Close the **Explore Customer Table** tab.

4. In the **Attributes** pane, select **Yearly Income**.

5. In the Properties window, change the value for the **DiscretizationMethod** property to **Automatic** and change the value for the **DiscretizationBucketCount** property to **5**.

The following image shows the modified properties for **Yearly Income**.

Grouping Attribute Hierarchy Members in the Employee Dimension

1. Switch to Dimension Designer for the Employee dimension.
2. In the **Data Source View** pane, right-click the **Employee** table, and then click **Explore Data**.

   Notice the values for the **SickLeaveHours** column and the **VacationHours** column.
3. Close the **Explore Employee Table** tab.
4. In the **Attributes** pane, select **Sick Leave Hours**.
5. In the Properties window, change the value for the **DiscretizationMethod** property to **Clusters** and change the value for the **DiscretizationBucketCount** property to **5**.
6. In the **Attributes** pane, select **Vacation Hours**.
7. In the Properties window, change the value for the **DiscretizationMethod** property to **Equal Areas** and change the value for the **DiscretizationBucketCount** property to **5**.

Browsing the Modified Attribute Hierarchies

1. On the **Build** menu of SQL Server Data Tools (SSDT), click **Deploy Analysis Services Tutorial**.
2. When deployment has successfully completed, switch to Cube Designer for the Analysis Services Tutorial

cube, and then click **Reconnect** on the **Browser** tab.

3. Click the Excel icon, and then click **Enable**.

4. Drag the **Internet Sales-Sales Amount** measure to the Values area of the PivotTable Field List.

5. In the field list, expand the **Product** dimension, and then drag the **Product Model Lines** user hierarchy to the **Row Labels** area of the field list.

6. Expand the **Customer** dimension in the field list, expand the **Demographic** display folder, and then drag the **Yearly Income** attribute hierarchy to the **Column Labels** area.

   The members of the **Yearly Income** attribute hierarchy are now grouped into six buckets, including a bucket for sales to customers whose yearly income is unknown. Not all buckets are displayed.

7. Remove the **Yearly Income** attribute hierarchy from the columns area and remove the **Internet Sales-Sales Amount** measure from the **Values** area.

8. Add the **Reseller Sales-Sales Amount** measure to the data area.

9. In the field list, expand the **Employee** dimension, expand **Organization**, then drag **Sick Leave Hours** to **Column Labels**.

   Notice that all sales are made by employees within one of two groups. Notice also that the employees with 32 - 42 sick leave hours made significantly more sales than employees with 20 - 31 sick leave hours.

   The following image shows sales dimensioned by employee sick leave hours.

10. Remove the **Sick Leave Hours** attribute hierarchy from the column area of the **Data** pane.

11. Add **Vacation Hours** to the column area of the **Data** pane.

    Notice that two groups appear, based on the equal areas grouping method. Three other groups are hidden because they contain no data values.

Modifying Grouping Properties and Reviewing the Effect of the Changes

1. Switch to Dimension Designer for the **Employee** dimension, and then select **Vacation Hours** in the **Attributes** pane.
2. In the Properties window, change the value of the **DiscretizationBucketCount** property to **10.**
3. On the **Build** menu of SQL Server Data Tools, click **Deploy Analysis Services Tutorial**.
4. When deployment has successfully completed, switch back to Cube Designer for the Analysis Services Tutorial cube.
5. Click **Reconnect** on the **Browser** tab, click the Excel icon, and then reconstruct the PivotTable so that you can view the effect of the change to the grouping method:

   a. Drag Reseller Sales-Sales Amount to Values

   b. Drag Vacation Hours (in the Employees Organization folder) to Columns

   c. Drag Product Model Lines to Rows

   Notice that there are now three groups of members of the **Vacation Hours** attribute that have sales values for products. (The other seven groups contain members with no sales data.)

Next Task in Lesson

Hiding and Disabling Attribute Hierarchies

See Also

Group Attribute Members (Discretization)

Lesson 4-4 - Hiding and Disabling Attribute Hierarchies

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

By default, an attribute hierarchy is created for every attribute in a dimension, and each hierarchy is available for dimensioning fact data. This hierarchy consists of an "All" level and a detail level containing all members of the hierarchy. As you have already learned, you can organize attributes into user-defined hierarchies to provide navigation paths in a cube. Under certain circumstances, you may want to disable or hide some attributes and their hierarchies. For example, certain attributes such as social security numbers or national identification numbers, pay rates, birth dates, and login information are not attributes by which users will dimension cube information. Instead, this information is generally only viewed as details of a particular attribute member. You may want to hide these attribute hierarchies, leaving the attributes visible only as member properties of a specific attribute. You may also want to make members of other attributes, such as customer names or postal codes, visible only when they are viewed through a user hierarchy instead of independently through an attribute hierarchy. One reason to do so may be the sheer number of distinct members in the attribute hierarchy. Finally, to improve processing performance, you should disable attribute hierarchies that users will not use for browsing.

The value of the **AttributeHierarchyEnabled** property determines whether an attribute hierarchy is created. If this property is set to **False**, the attribute hierarchy is not created and the attribute cannot be used as a level in a user hierarchy; the attribute hierarchy exists as a member property only. However, a disabled attribute hierarchy can still be used to order the members of another attribute. If the value of the **AttributeHierarchyEnabled** property is set to **True**, the value of the **AttributeHierarchyVisible** property determines whether the attribute hierarchy is visible independent of its use in a user-defined hierarchy.

When an attribute hierarchy is enabled, you may want to specify values for the following three additional properties:

- **IsAggregatable**

  By default, an (All) level is defined for all attribute hierarchies. To disable the (All) level for an enabled attribute hierarchy, set the value for this property to **False**.

  > **NOTE**
  > An attribute that has its **IsAggregatable** property set to false can only be used as the root of a user-defined hierarchy and must have a default member specified (otherwise, one will be chosen for you by the Analysis Services engine).

- **AttributeHierarchyOrdered**

  By default, Analysis Services orders the members of enabled attribute hierarchies during processing, and then stores the members by the value of the **OrderBy** property, such as by Name or Key. If you do not care about ordering, you can increase processing performance by setting the value of this property to **False**.

- **AttributeHierarchyOptimizedState**

  By default, Analysis Services creates an index for each enabled attribute hierarchy during processing, to improve query performance. If you do not plan to use an attribute hierarchy for browsing, you can increase processing performance by setting the value of this property to **NotOptimized**. However, if you use a

hidden hierarchy as the key attribute for the dimension, creating an index of the attribute members will still improve performance.

These properties do not apply if an attribute hierarchy is disabled.

In the tasks in this topic, you will disable social security numbers and other attributes in the Employee dimension that will not be used for browsing. You will then hide the customer name and postal code attribute hierarchies in the Customer dimension. The large number of attribute members in these hierarchies will make browsing these hierarchies very slow independent of a user hierarchy.

Setting Attribute Hierarchy Properties in the Employee Dimension

1. Switch to Dimension Designer for the Employee dimension, and then click the **Browser** tab.
2. Verify that the following attribute hierarchies appear in the **Hierarchy** list:
   - **Base Rate**
   - **Birth Date**
   - **Login ID**
   - **Manager SSN**
   - **SSN**
3. Switch to the **Dimension Structure** tab, and then select the following attributes in the **Attributes** pane. You can select multiple measures by clicking each while holding down the CTRL key:
   - **Base Rate**
   - **Birth Date**
   - **Login ID**
   - **Manager SSN**
   - **SSN**
4. In the Properties window, set the value of the **AttributeHierarchyEnabled** property to **False** for the selected attributes.

   Notice in the **Attributes** pane that the icon for each attribute has changed to indicate that the attribute is not enabled.

   The following image shows the **AttributeHierarchyEnabled** property set to False for the selected attributes.

5. On the **Build** menu, click **Deploy Analysis Services Tutorial**.

6. When processing has successfully completed, switch to the **Browser** tab, click **Reconnect**, and then try to browse the modified attribute hierarchies.

   Notice that the members of the modified attributes are not available for browsing as attribute hierarchies in the **Hierarchy** list. If you try to add one of the disabled attribute hierarchies as a level in a user hierarchy, you will receive an error notifying you that the attribute hierarchy must be enabled to participate in a user-defined hierarchy.

Setting Attribute Hierarchy Properties in the Customer Dimension

1. Switch to Dimension Designer for the Customer dimension, and then click the **Browser** tab.

2. Verify that the following attribute hierarchies appear in the **Hierarchy** list:

   - **Full Name**
   - **Postal Code**

3. Switch to the **Dimension Structure** tab, and then select the following attributes in the **Attributes** pane by using the CTRL key to select multiple attributes at the same time:

   - **Full Name**
   - **Postal Code**

4. In the Properties window, set the value of the **AttributeHierarchyVisible** property to **False** for the selected attributes.

Because the members of these attribute hierarchies will be used for dimensioning fact data, ordering and optimizing the members of these attribute hierarchies will improve performance. Therefore, the properties of these attributes should not be changed.

The following image shows the **AttributeHierarchyVisible** property set to False.

5. Drag the **Postal Code** attribute from the **Attributes** pane into the **Customer Geography** user hierarchy in the **Hierarchies and Levels** pane, immediately under the **City** level.

   Notice that a hidden attribute can still become a level in a user hierarchy.

6. On the **Build** menu, click **Deploy Analysis Services Tutorial**.

7. When deployment has successfully completed, switch to the **Browser** tab for the Customer dimension, and then click **Reconnect**.

8. Try to select either of the modified attribute hierarchies from the **Hierarchy** list.

   Notice that neither of the modified attribute hierarchies appears in the **Hierarchy** list.

9. In the **Hierarchy** list, select **Customer Geography**, and then browse each level in the browser pane.

   Notice that the hidden levels, **Postal Code** and **Full Name**, are visible in the user-defined hierarchy.

Next Task in Lesson

Sorting Attribute Members Based on a Secondary Attribute

Lesson 4-5 - Sorting Attribute Members Based on a Secondary Attribute

12/5/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Lesson 3, you learned how to sort attribute members based on either their name or key value. You also learned how to use a composite member key to affect attribute members and sort order. For more information, see Modifying the Date Dimension. However, if neither the name nor the key of the attribute provide the sort order that you want, you can use a secondary attribute to achieve the desired sort order. By defining a relationship between the attributes, you can use the second attribute to sort the members of the first attribute.

Attribute relationships define the relationships or dependencies between attributes. In a dimension that is based on a single relational table, all attributes are typically related to each other through the key attribute. This is because all the attributes for a dimension provide information about the members linked by the key attribute of the dimension to the facts in the fact table for each related measure group. In a dimension that is based on multiple tables, attributes are typically linked based on the join key between the tables. If the underlying data supports it, related attributes can be used to specify a sort order. For example, you might create a new attribute that provides the sort logic for a related attribute.

Dimension Designer lets you define additional relationships between attributes or change the default relationships to increase performance. The main constraint when you create an attribute relationship is to make sure that the attribute referred to has no more than one value for any member in the attribute to which it is related. When you define a relationship between two attributes, you can define the relationship as rigid or flexible, based on whether the relationships between members will change over time. For example, an employee might move to a different sales region, but a city will not move to a different state. If a relationship is defined as rigid, attribute aggregations are not recalculated every time the dimension is incrementally processed. However, if the relationship between members does change, the dimension must be fully processed. For more information, see Attribute Relationships, Define Attribute Relationships, Configure Attribute Relationship Properties, and Specifying Attribute Relationships Between Attributes in a User-Defined Hierarchy.

In the tasks in this topic, you will define a new attribute in the **Date** dimension based on an existing column in the underlying dimension table. You will use this new attribute to sort calendar month members chronologically instead of alphabetically. You will also define a new attribute in the **Customer** dimension based on the named calculation that you will use to sort the **Commute Distance** attribute members. In the tasks in the next topic, you will learn to use attribute relationships to increase query performance.

Defining an Attribute Relationship and Sort Order in the Date Dimension

1. Open Dimension Designer for the **Date** dimension, and then review the **OrderBy** property for the **Month Name** attribute in the Properties window.

   Notice that the **Month Name** attribute members are ordered by their key values.

2. Switch to the **Browser** tab, verify that **Calendar Date** is selected in the **Hierarchy** list, and then expand the levels in the user-defined hierarchy to review the sort order for the calendar months.

   Notice that the members of the attribute hierarchy are sorted based on the ASCII values of their member keys, which are month and year. In this case, sorting by the attribute name or key does not sort calendar months chronologically. To solve this, you will sort the members of the attribute hierarchy based on a new

attribute, the **MonthNumberOfYear** attribute. You will create this attribute based on a column that conveniently exists in the **Date** dimension table.

3. Switch to the **Dimension Structure** tab for the Date dimension, right-click **MonthNumberOfYear** in the **Data Source View** pane, and then click **New Attribute from Column**.

4. In the **Attributes** pane, select **Month Number Of Year**, and then set the **AttributeHierarchyEnabled** property to **False** in the Properties window, set the **AttributeHierarchyOptimizedState** property to **NotOptimized**, and set the **AttributeHierarchyOrdered** property to **False**.

   These settings will hide the attribute from users and will improve processing time. This attribute will not be used for browsing. It will only be used for ordering the members of another attribute.

   > **NOTE**
   > Sorting properties in the Properties window alphabetically will simplify this task as these three properties will be sorted adjacent to each other.

5. Click the **Attribute Relationships** tab.

   Notice that all the attributes in the **Date** dimension are related directly to the **Date** attribute, which is the member key that relates the dimension members to the facts in the related measure groups. There is no relationship defined between the **Month Name** attribute and the **Month Number Of Year** attribute.

6. In the diagram, right-click the **Month Name** attribute and then select **New Attribute Relationship**.

7. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Month Name**. Set the **Related Attribute** to **Month Number Of Year**.

8. In the **Relationship type** list, set the relationship type to **Rigid**.

   The relationships between the members of the **Month Name** attribute and the **Month Number Of Year** attribute will not change over time. As a result, Analysis Services will not drop aggregations for this relationship during incremental processing. If a change does occur, a processing error will occur during incremental processing and you will need to perform a full process of the dimension. You are now ready to set the sort order for the members of **Month Name**.

9. Click **OK**.

10. Click the **Dimension Structure** tab.

11. Select **Month Name** in the **Attributes** pane, and then change the value of the **OrderBy** property in the Properties window to **AttributeKey** and change the value of the **OrderByAttribute** property to **Month Number Of Year**.

12. On the **Build** menu, click **Deploy Analysis Services Tutorial**.

13. When deployment has successfully completed, switch to the **Browser** tab for the Date dimension, click **Reconnect**, and then browse the **Calendar Date** and **Fiscal Date** user hierarchies to verify that months now sort in chronological order.

    Notice that the months are now sorted in chronological order, as shown in the following image.

Defining Attribute Relationships and Sort Order in the Customer Dimension

1. Switch to the **Browser** tab in Dimension Designer for the Customer dimension, and then browse the members of the **Commute Distance** attribute hierarchy.

   Notice that the members of this attribute hierarchy are sorted based on the ASCII values of the member key. In this case, sorting by the attribute name or key does not sort the commute distances from least to most. In this task, you sort the members of the attribute hierarchy based on the **CommuteDistanceSort** named calculation that ascribes the appropriate sort number to each distinct value in the column. To save time, this named calculation has already been added to the **Customer** table in the Adventure Works DW data source view. You can switch to this data source view to view the SQL script that is used in this named calculation. For more information, see Define Named Calculations in a Data Source View (Analysis Services).

   The following image shows the members of the **Commute Distance** attribute hierarchy, sorted by the ASCII values of the member key.

2. Switch to the **Dimension Structure** tab in Dimension Designer for the Customer dimension, right-click **CommuteDistanceSort** in the **Customer** table in the **Data Source View** pane, and then click **New Attribute from Column**.

3. In the **Attributes** pane, select **Commute Distance Sort**, and then set the **AttributeHierarchyEnabled** property for this attribute to **False** in the Properties window, set the **AttributeHierarchyOptimizedState** property to **NotOptimized**, and set the **AttributeHierarchyOrdered** property to **False**.

   These settings will hide the attribute from users and will improve processing time. This attribute will not be used for browsing. It will only be used for ordering the members of another attribute.

4. Select **Geography**, and then set its **AttributeHierarchyVisible** property to **False** in the Properties window, set its **AttributeHierarchyOptimizedState** property to **NotOptimized**, and set its **AttributeHierarchyOrdered** property to **False**.

   These settings will hide the attribute from users and will improve processing time. This attribute will not be

used for browsing. It will be only be used for ordering the members of another attribute. Because **Geography** has member properties, its **AttributeHierarchyEnabled** property must be set to **True**. Therefore, to hide the attribute, you set the **AttributeHierarchyVisible** property to **False**.

5. Click the **Attribute Relationships** tab.

6. In the attributes list, right-click the **Commute Distance** attribute and then select **New Attribute Relationship**.

7. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Commute Distance**. Set the **Related Attribute** to **Commute Distance Sort**.

8. In the **Relationship type** list, set the relationship type to **Rigid**.

   The relationship between the members of the **Commute Distance** attribute and the **Commute Distance Sort** attribute will not change over time.

9. Click **OK**.

   You are now ready to set the sort order for the **Commute Distance** attribute.

10. Click the **Dimension Structure** tab.

11. In the **Attributes** pane, select **Commute Distance**, and then change the value of the **OrderBy** property in the Properties window to **AttributeKey**, and change the value of the **OrderByAttribute** property to **Commute Distance Sort**.

12. On the **Build** menu, click **Deploy Analysis Services Tutorial**.

13. When deployment has successfully completed, switch to the **Browser** tab of Dimension Designer for the Customer dimension, click **Reconnect**, and then browse the **Commute Distance** attribute hierarchy.

    Notice that the attribute hierarchy members are now sorted in a logical order based on increasing distance, as shown in the following image.

Next Task in Lesson

Specifying Attribute Relationships Between Attributes in a User-Defined Hierarchy

4-6-Specifying Attribute Relationships in User-Defined Hierarchy

12/5/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

As you have already learned in this tutorial, you can organize attribute hierarchies into levels within user hierarchies to provide navigation paths for users in a cube. A user hierarchy can represent a natural hierarchy, such as city, state, and country, or can just represent a navigation path, such as employee name, title, and department name. To the user navigating a hierarchy, these two types of user hierarchies are the same.

With a natural hierarchy, if you define attribute relationships between the attributes that make up the levels, Analysis Services can use an aggregation from one attribute to obtain the results from a related attribute. If there are no defined relationships between attributes, Analysis Services will aggregate all non-key attributes from the key attribute. Therefore, if the underlying data supports it, you should define attribute relationships between attributes. Defining attribute relationships improves dimension, partition, and query processing performance. For more information, see Define Attribute Relationships and Attribute Relationships.

When you define attribute relationships, you can specify that the relationship is either flexible or rigid. If you define a relationship as rigid, Analysis Services retains aggregations when the dimension is updated. If a relationship that is defined as rigid actually changes, Analysis Services generates an error during processing unless the dimension is fully processed. Specifying the appropriate relationships and relationship properties increases query and processing performance. For more information, see Define Attribute Relationships, and User Hierarchy Properties.

In the tasks in this topic, you define attribute relationships for the attributes in the natural user hierarchies in the Analysis Services Tutorial project. These include the **Customer Geography** hierarchy in the **Customer** dimension, the **Sales Territory** hierarchy in the **Sales Territory** dimension, the **Product Model** Lines hierarchy in the **Product** dimension, and the **Fiscal Date** and **Calendar Date** hierarchies in the **Date** dimension. These user hierarchies are all natural hierarchies.

Defining Attribute Relationships for Attributes in the Customer Geography Hierarchy

1. Switch to Dimension Designer for the Customer dimension, and then click the **Dimension Structure** tab.

   In the **Hierarchies** pane, notice the levels in the **Customer Geography** user-defined hierarchy. This hierarchy is currently just a drill-down path for users, as no relationship between levels or attributes have been defined.

2. Click the **Attribute Relationships** tab.

   Notice the four attribute relationships that link the non-key attributes from the **Geography** table to the key attribute from the **Geography** table. The **Geography** attribute is related to the **Full Name** attribute. The **Postal Code** attribute is indirectly linked to the **Full Name** attribute through the **Geography** attribute, because the **Postal Code** is linked to the **Geography** attribute and the **Geography** attribute is linked to the **Full Name** attribute. Next, we will change the attribute relationships so that they do not use the **Geography** attribute.

3. In the diagram, right-click the **Full Name** attribute and then select **New Attribute Relationship**.

4. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Full Name**. Set the **Related Attribute** to **Postal Code**. In the **Relationship type** list, leave the relationship type set to **Flexible** because relationships between the members might change over time.

5. Click **OK**.

   A warning icon appears in the diagram because the relationship is redundant. The relationship **Full Name** -> **Geography**-> **Postal Code** already existed, and you just created the relationship **Full Name** -> **Postal Code**. The relationship **Geography**-> **Postal Code** is now redundant, so we will remove it.

6. In the **Attribute Relationships** pane, right-click **Geography**-> **Postal Code** and then click **Delete**.

7. When the **Delete Objects** dialog box appears, click **OK**.

8. In the diagram, right-click the **Postal Code** attribute and then select **New Attribute Relationship**.

9. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Postal Code**. Set the **Related Attribute** to **City**. In the **Relationship type** list, leave the relationship type set to **Flexible**.

10. Click **OK**.

    The relationship **Geography**-> **City** is now redundant so we will delete it.

11. In the Attribute Relationships pane, right-click **Geography**-> **City** and then click **Delete**.

12. When the **Delete Objects** dialog box appears, click **OK**.

13. In the diagram, right-click the **City** attribute and then select **New Attribute Relationship**.

14. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **City**. Set the **Related Attribute** to **State-Province**. In the **Relationship type** list, set the relationship type to **Rigid** because the relationship between a city and a state will not change over time.

15. Click **OK**.

16. Right-click the arrow between **Geography** and **State-Province** and then click **Delete**.

17. When the **Delete Objects** dialog box appears, click **OK**.

18. In the diagram, right-click the **State-Province** attribute and then select **New Attribute Relationship**.

19. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **State-Province**. Set the **Related Attribute** to **Country-Region**. In the **Relationship type** list, set the relationship type to **Rigid** because the relationship between a state-province and a country-region will not change over time.

20. Click **OK**.

21. In the Attribute Relationships pane, right-click **Geography**-> **Country-Region** and then click **Delete**.

22. When the **Delete Objects** dialog box appears, click **OK**.

23. Click the **Dimension Structure** tab.

    Notice that when you delete the last attribute relationship between **Geography** and other attributes, that **Geography** itself is deleted. This is because the attribute is no longer used.

24. On the File menu, click **Save All**.

Defining Attribute Relationships for Attributes in the Sales Territory Hierarchy

1. Open Dimension Designer for the **Sales Territory** dimension, and then click the **Attribute Relationships**

tab.

2. In the diagram, right-click the **Sales Territory Country** attribute and then select **New Attribute Relationship**.

3. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Sales Territory Country**. Set the **Related Attribute** to **Sales Territory Group**. In the **Relationship type** list, leave the relationship type set to **Flexible**.

4. Click **OK**.

   Sales Territory Group is now linked to **Sales Territory Country**, and **Sales Territory Country** is now linked to **Sales Territory Region**. The **RelationshipType** property for each of these relationships is set to **Flexible** because the groupings of regions within a country might change over time and because the groupings of countries into groups might change over time.

Defining Attribute Relationships for Attributes in the Product Model Lines Hierarchy

1. Open Dimension Designer for the **Product** dimension, and then click the **Attribute Relationships** tab.

2. In the diagram, right-click the **Model Name** attribute and then select **New Attribute Relationship**.

3. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Model Name**. Set the **Related Attribute** to **Product Line**. In the **Relationship type** list, leave the relationship type set to **Flexible**.

4. Click **OK**.

Defining Attribute Relationships for Attributes in the Fiscal Date Hierarchy

1. Switch to Dimension Designer for the **Date** dimension, and then click the **Attribute Relationships** tab.

2. In the diagram, right-click the **Month Name** attribute and then select **New Attribute Relationship**.

3. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Month Name**. Set the **Related Attribute** to **Fiscal Quarter**. In the **Relationship type** list, set the relationship type to **Rigid**.

4. Click **OK**.

5. In the diagram, right-click the **Fiscal Quarter** attribute and then select **New Attribute Relationship**.

6. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Fiscal Quarter**. Set the **Related Attribute** to **Fiscal Semester**. In the **Relationship type** list, set the relationship type to **Rigid**.

7. Click **OK**.

8. In the diagram, right-click the **Fiscal Semester** attribute and then select **New Attribute Relationship**.

9. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Fiscal Semester**. Set the **Related Attribute** to **Fiscal Year**. In the **Relationship type** list, set the relationship type to **Rigid**.

10. Click **OK**.

Defining Attribute Relationships for Attributes in the Calendar Date Hierarchy

1. In the diagram, right-click the **Month Name** attribute and then select **New Attribute Relationship**.

2. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Month Name**. Set the **Related Attribute** to **Calendar Quarter**. In the **Relationship type** list, set the relationship type to **Rigid**.
3. Click **OK**.
4. In the diagram, right-click the **Calendar Quarter** attribute and then select **New Attribute Relationship**.
5. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Calendar Quarter**. Set the **Related Attribute** to **Calendar Semester**. In the **Relationship type** list, set the relationship type to **Rigid**.
6. Click **OK**.
7. In the diagram, right-click the **Calendar Semester** attribute and then select **New Attribute Relationship**.
8. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Calendar Semester**. Set the **Related Attribute** to **Calendar Year**. In the **Relationship type** list, set the relationship type to **Rigid**.
9. Click **OK**.

Defining Attribute Relationships for Attributes in the Geography Hierarchy

1. Open Dimension Designer for the Geography dimension, and then click the **Attribute Relationships** tab.
2. In the diagram, right-click the **Postal Code** attribute and then select **New Attribute Relationship**.
3. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Postal Code**. Set the **Related Attribute** to **City**. In the **Relationship type** list, set the relationship type to **Flexible**.
4. Click **OK**.
5. In the diagram, right-click the **City** attribute and then select **New Attribute Relationship**.
6. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **City**. Set the **Related Attribute** to **State-Province**. In the **Relationship type** list, set the relationship type to **Rigid**.
7. Click **OK**.
8. In the diagram, right-click the **State-Province** attribute and then select **New Attribute Relationship**.
9. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **State-Province**. Set the **Related Attribute** to **Country-Region**. In the **Relationship type** list, set the relationship type to **Rigid**.
10. Click **OK**.
11. In the diagram, right-click the **Geography Key** attribute and then select **Properties**.
12. Set the **AttributeHierarchyOptimizedState** property to **NotOptimized**, set the **AttributeHierarchyOrdered** property to **False**, and set the **AttributeHierarchyVisible** property to **False**.
13. On the **File** menu, click **Save All**.
14. On the **Build** menu of SQL Server Data Tools (SSDT), click **Deploy Analysis Services Tutorial**.

Next Task in Lesson

Defining the Unknown Member and Null Processing Properties

See Also

Define Attribute Relationships

User Hierarchy Properties

Lesson 4-7 - Defining the Unknown Member and Null Processing Properties

11/28/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When Analysis Services processes a dimension, all the distinct values from the underlying columns in the tables, or views in the data source view, populate the attributes in the dimension. If Analysis Services encounters a null value during processing, by default, it converts this null to a zero for numeric columns or to an empty string for string columns. You can modify the default settings or convert null values in your extract, transform, and load process (if any) of the underlying relational data warehouse. Additionally, you can have Analysis Services convert the null value to a designated value by configuring three properties: the **UnknownMember** and **UnknownMemberName** properties for the dimension, and the **NullProcessing** property for the dimension's key attribute.

The Dimension Wizard and the Cube Wizard will enable these properties for you based on whether the key attribute of a dimension is nullable or the root attribute of a snowflake dimension is based on a nullable column. In these cases, the **NullProcessing** property of the key attribute will be set to **UnknownMember** and the **UnknownMember** property will be set to **Visible**.

However, when you build snowflaked dimensions incrementally, as we are doing with the Product dimension in this tutorial, or when you define dimensions using Dimension Designer and then incorporate these existing dimensions into a cube, the **UnknownMember** and **NullProcessing** properties might need to be set manually.

In the tasks in this topic, you will add the product category and product subcategory attributes to the Product dimension from snowflaked tables that you will add to the Adventure Works DW data source view. You will then enable the **UnknownMember** property for the Product dimension, specify **Assembly Components** as the value for the **UnknownMemberName** property, relate the **Subcategory** and **Category** attributes to the product name attribute, and then define custom error handling for the member key attribute that links the snowflaked tables.

> **NOTE**
>
> If you have added the Subcategory and Category attributes when you originally defined the Analysis Services Tutorial cube using the Cube Wizard, these steps would have been performed for you automatically.

Reviewing Error Handling and Unknown Member Properties in the Product Dimension

1. Switch to Dimension Designer for the **Product** dimension, click the **Dimension Structure** tab, and then select **Product** in the **Attributes** pane.

   This enables you to view and modify the properties of the dimension itself.

2. In the Properties window, review the **UnknownMember** and **UnknownMemberName** properties.

   Notice that the **UnknownMember** property is not enabled, because its value is set to **None** instead of **Visible** or **Hidden**, and that no name is specified for the **UnknownMemberName** property.

3. In the Properties window, select **(custom)** in the **ErrorConfiguration** property cell, and then expand the **ErrorConfiguration** properties collection.

Setting the **ErrorConfiguration** property to **(custom)** allows you to view the default error configuration settings - it does not change any settings.

4. Review the key and null key error configuration properties, but do not make any changes.

   Notice that, by default, when null keys are converted to the unknown member and the processing error associated with this conversion is ignored.

   The following image shows the property settings for the **ErrorConfiguration** properties collection.

ErrorConfiguration	(custom)
KeyDuplicate	IgnoreError
KeyErrorAction	ConvertToUnknown
KeyErrorLimit	0
KeyErrorLimitAction	StopProcessing
KeyErrorLogFile	
KeyNotFound	ReportAndContinue
NullKeyConvertedToUnknown	IgnoreError
NullKeyNotAllowed	ReportAndContinue

5. Click the **Browser** tab, verify that **Product Model Lines** is selected in the **Hierarchy** list, and then expand **All Products**.

   Notice the five members of the Product Line level.

6. Expand **Components**, and then expand the unlabeled member of the **Model Name** level.

   This level contains the assembly components that are used when building other components, starting with the **Adjustable Race** product, as shown in the following image.

Defining Attributes from Snowflaked Tables and a Product Category User-Defined Hierarchy

1. Open Data Source View Designer for the Adventure Works DW data source view, select **Reseller Sales** in the **Diagram Organizer** pane, and then click **Add/Remove Objects** on the **Data Source View** menu of SQL Server Data Tools (SSDT).

   The **Add/Remove Tables** dialog box opens.

2. In the **Included objects** list, select **DimProduct (dbo)**, and then click **Add Related Tables**.

Both **DimProductSubcategory (dbo)** and **FactProductInventory (dbo)** are added. Remove **FactProductInventory (dbo)** so that just the **DimProductSubcategory (dbo)** table is added to the **Included objects** list.

3. With the **DimProductSubcategory (dbo)** table selected by default as the table most recently added, click **Add Related Tables** again.

   The **DimProductCategory (dbo)** table is added to the **Included objects** list.

4. Click **OK**.

5. On the **Format** menu of SQL Server Data Tools, point to **Auto Layout**, and then click **Diagram**.

   Notice that the **DimProductSubcategory (dbo)** table and **DimProductCategory (dbo)** table are linked to each other, and also to the **ResellerSales** table through the **Product** table.

6. Switch to Dimension Designer for the **Product** dimension, and then click the **Dimension Structure** tab.

7. Right-click anywhere in the **Data Source View** pane, and then click **Show All Tables**.

8. In the **Data Source View** pane, locate the **DimProductCategory** table, right-click **ProductCategoryKey** in that table, and then click **New Attribute from Column**.

9. In the **Attributes** pane, change the name of this new attribute to **Category**.

10. In the Properties window, click in the **NameColumn** property field and then click the browse (**...**) button to open the **Name Column** dialog box.

11. Select **EnglishProductCategoryName** in the **Source column** list and then click **OK**.

12. In the **Data Source View** pane, locate the **DimProductSubcategory** table, right-click **ProductSubcategoryKey** in that table, and then click **New Attribute from Column**.

13. In the **Attributes** pane, change the name of this new attribute to **Subcategory**.

14. In the Properties window, click in the **NameColumn** property field and then click the browse **(...)** button to open the **Name Column** dialog box.

15. Select **EnglishProductSubcategoryName** in the **Source column** list and then click **OK**.

16. Create a new user-defined hierarchy called **Product Categories** with the following levels, in order from top to bottom: **Category**, **Subcategory**, and **Product Name**.

17. Specify **All Products** as the value for the **AllMemberName** property of the Product Categories user-defined hierarchy.

Browsing the User-Defined Hierarchies in the Product Dimension

1. On the toolbar of the **Dimension Structure** tab of **Dimension Designer** for the **Product** dimension, click **Process**.

2. Click **Yes** to build and deploy the project, and then click **Run** to process the **Product** dimension.

3. When processing has succeeded, expand **Processing Dimension 'Product' completed successfully** in the **Process Progress** dialog box, expand **Processing Dimension Attribute 'Product Name' completed**, and then expand **SQL queries 1**.

4. Click the SELECT DISTINCT query and then click **View Details**.

   Notice that a WHERE clause has been added to the SELECT DISTINCT clause that removes those products that have no value in the ProductSubcategoryKey column, as shown in the following image.

5. Click **Close** three times to close all processing dialog boxes.

6. Click the **Browser** tab in Dimension Designer for the **Product** dimension, and then click **Reconnect**.

7. Verify that **Product Model Lines** appears in the **Hierarchy** list, expand **All Products**, and then expand **Components**.

8. Select **Product Categories** in the **Hierarchy** list, expand **All Products**, and then expand **Components**.

   Notice that none of the assembly components appear.

To modify the behavior mentioned in the previous task, you will enable the **UnknownMember** property of the Products dimension, set a value for the **UnknownMemberName** property, set the **NullProcessing** property for the **Subcategory** and **Model Name** attributes to **UnknownMember**, define the **Category** attribute as a related attribute of the **Subcategory** attribute, and then define the **Product Line** attribute as a related attribute of the **Model Name** attribute. These steps will cause Analysis Services to use the unknown member name value for each product that does not have a value for the **SubcategoryKey** column, as you will see in the following task.

Enabling the Unknown Member, Defining Attribute Relationships, and Specifying Custom Processing Properties for Nulls

1. Click the **Dimension Structure** tab in Dimension Designer for the **Product** dimension, and then select **Product** in the **Attributes** pane.

2. In the **Properties** window, change the **UnknownMember** property to **Visible**, and then change the value for the **UnknownMemberName** property to **Assembly Components**.

   Changing the **UnknownMember** property to either **Visible** or **Hidden** enables the **UnknownMember** property for the dimension.

3. Click the **Attribute Relationships** tab.

4. In the diagram, right-click the **Subcategory** attribute and then select **New Attribute Relationship**.
5. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Subcategory**. Set the **Related Attribute** to **Category**. Leave the relationship type set to **Flexible**.
6. Click **OK**.
7. In the **Attributes** pane, select **Subcategory.**
8. In the Properties window, expand the **KeyColumns** property and then expand the **DimProductSubcategory.ProductSubcategoryKey (Integer)** property.
9. Change the **NullProcessing** property to **UnknownMember**.
10. In the **Attributes** pane, select **Model Name**.
11. In the Properties window, expand the **KeyColumns** property and then expand the **Product.ModelName (WChar)** property.
12. Change the **NullProcessing** property to **UnknownMember**.

    Because of these changes, when Analysis Services encounters a null value for the **Subcategory** attribute or the **Model Name** attribute during processing, the unknown member value will be substituted as the key value, and the user-defined hierarchies will be constructed correctly.

Browsing the Product Dimension Again

1. On the **Build** menu, click **Deploy Analysis Services Tutorial**.
2. When deployment has successfully completed, click the **Browser** tab in Dimension Designer for the **Product** dimension, and then click **Reconnect**.
3. Verify that **Product Categories** is selected in the **Hierarchy** list, and then expand **All Products**.

   Notice that Assembly Components appears as a new member of the Category level.
4. Expand the **Assembly Components** member of the **Category** level and then expand the **Assembly Components** member of the **Subcategory** level.

   Notice that all the assembly components now appear at the **Product Name** level, as shown in the following image.

Next Lesson

Lesson 5: Defining Relationships Between Dimensions and Measure Groups

Lesson 5: Defining Relationships Between Dimensions and Measure Groups

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In the previous lessons in this tutorial, you learned that database dimensions added to a cube can be used as the basis for one or more cube dimensions. In this lesson, you learn to define different types of relationships between cube dimensions and measure groups, and to specify the properties of these relationships.

For more information, see Dimension Relationships.

> **NOTE**
> Completed projects for all of the lessons in this tutorial are available online. You can jump ahead to any lesson by using the completed project from the previous lesson as a starting point. Click here to download the sample projects that go with this tutorial.

This lesson contains the following tasks:

Defining a Referenced Relationship
In this task, you learn to link a dimension to a fact table indirectly through a dimension that is linked directly through a primary key-foreign key relationship.

Defining a Fact Relationship
In this task, you learn to define a dimension based on data in the fact table, and to define the dimension relationship as a fact relationship.

Defining a Many-to-Many Relationship
In this task, you learn to relate a fact to multiple dimension members through the definition of a many-to-many relationship between dimension tables and fact tables.

Defining Dimension Granularity within a Measure Group
In this task, you learn to modify the granularity of a dimension for a specific measure group.

Next Lesson

Lesson 6: Defining Calculations

See Also

Analysis Services Tutorial Scenario
Multidimensional Modeling (Adventure Works Tutorial)
Dimension Relationships

Lesson 5-1 - Defining a Referenced Relationship

11/28/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Up to this point in the tutorial, each cube dimension that you defined was based on a table that was directly linked to the fact table for a measure group by a primary key to foreign key relationship. In the tasks in this topic, you link the **Geography** dimension to the fact table for reseller sales through the **Reseller** dimension, which is called a *reference dimension*. This enables users to dimension reseller sales by geography. For more information, see Define a Referenced Relationship and Referenced Relationship Properties.

Dimensioning Reseller Sales by Geography

1. In Solution Explorer, right-click **Analysis Services Tutorial** in the **Cubes** folder, and then click **Browse**.

2. Remove all hierarchies from the data pane, and then verify that the **Reseller Sales-Sales Amount** measure appears in the data area of the data pane. Add it to the data pane if it is not already there.

3. From the **Geography** dimension in the metadata pane, drag the **Geographies** user-defined hierarchy to the **Drop Row Fields Here** area of the data pane.

   Notice that the **Reseller Sales-Sales Amount** measure is not correctly dimensioned by the **Country-Region** attribute members in the **Regions** hierarchy. The value for **Reseller Sales-Sales Amount** repeats for each **Country-Region** attribute member.

4. Open Data Source View Designer for the **Adventure Works DW 2012** data source view.

5. In the **Diagram Organizer** pane, view the relationship between the **Geography** table and the **ResellerSales** table.

   Notice that there is no direct link between these tables. However, there is an indirect link between these tables through either the **Reseller** table or the **SalesTerritory** table.

6. Double-click the arrow that represents the relationship between the **Geography** table and the **Reseller** table.

   In the **Edit Relationship** dialog box, notice that the **GeographyKey** column is the primary key in the **Geography** table and the foreign key in the **Reseller** table.

7. Click **Cancel**, switch to Cube Designer for the Analysis Services Tutorial cube, and then click the **Dimension Usage** tab.

Notice that the **Geography** cube dimension does not currently have a relationship with either the **Internet Sales** measure group or the **Reseller Sales** measure group.

8. Click the ellipsis button (**...**) in the **Full Name** cell at the intersection of the **Customer** dimension and the **Internet Sales** measure group.

   In the **Define Relationship** dialog box, notice that a **Regular** relationship is defined between the **DimCustomer** dimension table and the **FactInternetSales** measure group table based on the **CustomerKey** column in each of these tables. All the relationships that you have defined within this tutorial up to this point have been regular relationships.

   The following image shows the **Define Relationship** dialog box with a regular relationship between the **DimCustomer** dimension table and the **FactInternetSales** measure group table.

9. Click **Cancel**.

10. Click the ellipsis button (**...**) in the unnamed cell at the intersection of the **Geography** dimension and the **Reseller Sales** measure group.

    In the **Define Relationship** dialog box, notice that no relationship is currently defined between the Geography cube dimension and the Reseller Sales measure group. You cannot define a regular relationship because there is no direct relationship between the dimension table for the Geography dimension and the fact table for the Reseller Sales measure group.

11. In the **Select relationship type** list, select **Referenced**.

    You define a referenced relationship by specifying a dimension that is directly connected to the measure group table, called an *intermediate dimension*, that Analysis Services can use to link the reference dimension to the fact table. You then specify the attribute that links the reference dimension to the intermediate dimension.

12. In the **Intermediate dimension** list, select **Reseller**.

    The underlying table for the Geography dimension is linked to the fact table through the underlying table for the Reseller dimension.

13. In the **Reference dimension attribute** list, select **Geography Key**, and then try to select **Geography Key** in the **Intermediate dimension attribute** list.

Notice that **Geography Key** does not appear in the **Intermediate dimension attribute** list. This is because the **GeographyKey** column is not defined as an attribute in the **Reseller** dimension.

14. Click **Cancel**.

In the next task, you will solve this problem by defining an attribute that is based on the GeographyKey column in the Reseller dimension.

Defining the Intermediate Dimension Attribute and the Referenced Dimension Relationship

1. Open Dimension Designer for the **Reseller** dimension, and view the columns in the **Reseller** table in the **Data Source View** pane, and view the defined attributes in the **Reseller** dimension in the **Attributes** pane.

   Notice that although GeographyKey is defined as a column in the Reseller table, no dimension attribute is defined in the Reseller dimension based on this column. Geography is defined as a dimension attribute in the Geography dimension because it is the key column that links the underlying table for that dimension to the fact table.

2. To add a **Geography Key** attribute to the **Reseller** dimension, right-click **GeographyKey** in the **Data Source View** pane, and then click **New Attribute from Column**.

3. In the **Attributes** pane, select **Geography Key**, and then, in the Properties window, set the **AttributeHierarchyOptimizedState** property to **NotOptimized**, the **AttributeHierarchyOrdered** property to **False**, and the **AttributeHierarchyVisible** property to **False.**

   The Geography Key attribute in the Reseller dimension will only be used to link the Geography dimension to the Reseller Sales fact table. Because it will not be used for browsing, there is no value in defining this attribute hierarchy as visible. Additionally, ordering and optimizing the attribute hierarchy will only negatively affect processing performance. However, the attribute must be enabled to serve as the link between the two dimensions.

4. Switch to Cube Designer for the Analysis Services Tutorial cube, click the **Dimension Usage** tab, and then click the ellipsis button (**...**) at the intersection of the **Reseller Sales** measure group and the **Geography** cube dimension.

5. In the **Select relationship type** list, select **Referenced**.

6. In the **Intermediate dimension** list, select **Reseller**.

7. In the **Reference dimension attribute** list, select **Geography Key**, and then select **Geography Key** in the **Intermediate dimension attribute** list.

   Notice that the **Materialize** check box is selected. This is the default setting for MOLAP dimensions. Materializing the dimension attribute link causes the value of the link between the fact table and the reference dimension for each row to be materialized, or stored, in the dimension's MOLAP structure during processing. This will have a minor effect on processing performance and storage requirements, but will increase query performance (sometimes significantly).

8. Click **OK**.

   Notice that the **Geography** cube dimension is now linked to the **Reseller Sales** measure group. The icon indicates that the relationship is a referenced dimension relationship.

9. In the **Dimensions** list on the **Dimension Usage** tab, right-click **Geography**, and then click **Rename**.

10. Change the name of this cube dimension to **Reseller Geography**.

    Because this cube dimension is now linked to the **Reseller Sales** measure group, users will benefit from

explicitly defining its use in the cube, to avoid possible user confusion.

Successfully Dimensioning Reseller Sales by Geography

1. On the **Build** menu, click **Deploy Analysis Services Tutorial**.

2. When deployment has successfully completed, click the **Browser** tab in Cube Designer for the Analysis Services Tutorial cube, and then click the **Reconnect** button.

3. In the metadata pane, expand **Reseller Geography**, right-click **Geographies**, and then click **Add to Row Area**.

   Notice that the **Reseller Sales-Sales Amount** measure is now correctly dimensioned by the **Country-Region** attribute of the **Geographies** user-defined hierarchy, as shown in the following image.

Dimension	Hierarchy
<Select dimension>	

Country-Region	Reseller Sales-Sales Am...
Australia	1594335.37669999
Canada	14662231.2266002
France	4607537.93499998
Germany	1983988.0373
United Kingdom	4271961.23019997
United States	53330543.1764993

Next Task in Lesson

Defining a Fact Relationship

See Also

Attribute Relationships
Define a Referenced Relationship and Referenced Relationship Properties

Lesson 5-2 - Defining a Fact Relationship

11/28/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Users sometimes want to be able to dimension measures by data items that are in the fact table or to query the fact table for specific additional related information, such as invoice numbers or purchase order numbers related to specific sales facts. When you define a dimension based on such a fact table item, the dimension is called a *fact dimension*. Fact dimensions are also known as degenerate dimensions. Fact dimensions are useful for grouping together related fact table rows, such as all the rows that are related to a particular invoice number. Although you can put this information in a separate dimension table in the relational database, creating a separate dimension table for the information provides no benefit because the dimension table would grow at the same rate as the fact table, and would just create duplicate data and unnecessary complexity.

Within Analysis Services, you can determine whether to duplicate the fact dimension data in a MOLAP dimension structure for increased query performance, or whether to define the fact dimension as a ROLAP dimension to save storage space at the expense of query performance. When you store a dimension with the MOLAP storage mode, all the dimension members are stored in the instance of Analysis Services in a highly compressed MOLAP structure, in addition to being stored in the measure group's partitions. When you store a dimension with the ROLAP storage mode, only the dimension definition is stored in the MOLAP structure-the dimension members themselves are queried from the underlying relational fact table at query time. You decide the appropriate storage mode based on how frequently the fact dimension is queried, the number of rows returned by a typical query, the performance of the query, and the processing cost. Defining a dimension as ROLAP does not require that all cubes that use the dimension also be stored with the ROLAP storage mode. The storage mode for each dimension can be configured independently.

When you define a fact dimension, you can define the relationship between the fact dimension and the measure group as a fact relationship. The following constraints apply to fact relationships:

- The granularity attribute must be the key column for the dimension, which creates a one-to-one relationship between the dimension and the facts in the fact table.
- A dimension can have a fact relationship with only a single measure group.

> **NOTE**
> Fact dimensions must be incrementally updated after every update to the measure group that the fact relationship references.

For more information, see Dimension Relationships, and Define a Fact Relationship and Fact Relationship Properties.

In the tasks in this topic, you add a new cube dimension based on the **CustomerPONumber** column in the **FactInternetSales** fact table. You then define the relationship between this new cube dimension and the **Internet Sales** measure group as a fact relationship.

Defining the Internet Sales Orders Fact Dimension

1. In Solution Explorer, right-click **Dimensions**, and then click **New Dimension**.
2. On the **Welcome to the Dimension Wizard** page, click **Next**.

3. On the **Select Creation Method** page, verify that the **Use an existing table** option is selected, and then click **Next**.
4. On the **Specify Source Information** page, verify that the **Adventure Works DW 2012** data source view is selected.
5. In the **Main table** list, select **InternetSales**.
6. In the **Key columns** list, verify that **SalesOrderNumber** and **SalesOrderLineNumber** are listed.
7. In the **Name column** list, select **SalesOrderLineNumber**.
8. Click **Next**.
9. On the **Select Related Tables** page, clear the check boxes beside all of the tables, and then click **Next**.
10. On the **Select Dimension Attributes** page, click the check box in the header twice to clear all of the check boxes. The **Sales Order Number** attribute will remain selected because it is the key attribute.
11. Select the **Customer PO Number** attribute, and then click **Next**.
12. On the **Completing the Wizard** page, change the name to **Internet Sales Order Details** and then click **Finish** to complete the wizard.
13. On the **File** menu, click **Save All**.
14. In the **Attributes** pane of the Dimension Designer for the **Internet Sales Order Details** dimension, select **Sales Order Number**, and then change the **Name** property in the Properties window to **Item Description.**
15. In the **NameColumn** property cell, click the browse button **(...)**. In the **Name Column** dialog box, select **Product** from the **Source table** list, select **EnglishProductName** for the **Source column**, and then click **OK**.
16. Add the **Sales Order Number** attribute to the dimension by dragging the **SalesOrderNumber** column from the **InternetSales** table in the **Data Source View** pane to the **Attributes** pane.
17. Change the **Name** property of the new **Sales Order Number** attribute to **Order Number**, and change the **OrderBy** property to **Key**.
18. In the **Hierarchies** pane, create an **Internet Sales Orders** user hierarchy that contains the **Order Number** and **Item Description** levels, in that order.
19. In the **Attributes** pane, select **Internet Sales Order Details**, and then review the value for the **StorageMode** property in the Properties window.

    Notice that, by default, this dimension is stored as a MOLAP dimension. Although changing the storage mode to ROLAP will save processing time and storage space, it occurs at the expense of query performance. For the purposes of this tutorial, you will use MOLAP as the storage mode.
20. To add the newly created dimension to the Analysis Services Tutorial cube as a cube dimension, switch to **Cube Designer**. On the **Cube Structure** tab, right-click in the **Dimensions** pane and select **Add Cube Dimension**.
21. In the **Add Cube Dimension**.dialog box, select **Internet Sales Order Details** and then click **OK**.

Defining a Fact Relationship for the Fact Dimension

1. In the Cube Designer for the Analysis Services Tutorial cube, click the **Dimension Usage** tab.

   Notice that the **Internet Sales Order Details** cube dimension is automatically configured as having a fact

relationship, as shown by the unique icon.

2. Click the browse button (**...**) in the **Item Description** cell, at the intersection of the **Internet Sales** measure group and the **Internet Sales Order Details** dimension, to review the fact relationship properties.

   The **Define Relationship** dialog box opens. Notice that you cannot configure any of the properties.

   The following image shows the fact relationship properties in the **Define Relationship** dialog box.

3. Click **Cancel**.

Browsing the Cube by Using the Fact Dimension

1. On the **Build** menu, click **Deploy Analysis Services Tutorial** to deploy the changes to the instance of Analysis Services and process the database.

2. After deployment has successfully completed, click the **Browser** tab in Cube Designer for the Analysis Services Tutorial cube, and then click the **Reconnect** button.

3. Clear all measures and hierarchies from the data pane, and then add the **Internet Sales-Sales Amount** measure to the data area of the data pane.

4. In the metadata pane, expand **Customer**, expand **Location**, expand **Customer Geography**, expand **Members**, expand **All Customers**, expand **Australia**, expand **Queensland**, expand **Brisbane**, expand **4000**, right-click **Adam Powell**, and then click **Add to Filter**.

   Filtering to limit the sales orders returned to a single customer lets the user drill down to the underlying detail in a large fact table without suffering a significant loss in query performance.

5. Add the **Internet Sales Orders** user-defined hierarchy from the **Internet Sales Order Details** dimension to the row area of the data pane.

   Notice that the sales order numbers and the corresponding Internet sales amounts for Adam Powell appear in the data pane.

   The following image shows the result of the previous steps.

Dimension	Hierarchy	Operator	Filter Expression
Customer	Customer Geography	Equal	{ Adam Powell }
<Select dimension>			

Order Number	Item Description	Internet Sales-Sales ...
SO49206	Road-250 Black, 48	2181.5625
SO61522	Road-350-W Yellow, 48	1700.99
SO61522	Short-Sleeve Classic Jersey, XL	53.99

Next Task in Lesson

Defining a Many-to-Many Relationship

See Also

Dimension Relationships

Define a Fact Relationship and Fact Relationship Properties

Lesson 5-3 - Defining a Many-to-Many Relationship

11/28/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you define a dimension, typically each fact joins to one and only one dimension member, whereas a single dimension member can be associated with many different facts. For example, each customer can have many orders but each order belongs to a single customer. In relational database terminology, this is referred to as a *one-to-many relationship*. However, sometimes a single fact can join to multiple dimension members. In relational database terminology, this is referred to as a *many-to-many relationship*. For example, a customer may have multiple reasons for making a purchase, and a purchase reason can be associated with multiple purchases. A join table is used to define the sales reasons that relate to each purchase. A Sales Reason dimension constructed from such relationships would then have multiple members that relate to a single sales transaction. Many-to-many dimensions expand the dimensional model beyond the classic star schema and support complex analytics when dimensions are not directly related to a fact table.

In Analysis Services, you define a many-to-many relationship between a dimension and a measure group by specifying an intermediate fact table that is joined to the dimension table. An intermediate fact table is joined, in turn, to an intermediate dimension table to which the fact table is joined. The many-to-many relationships between the intermediate fact table and both the dimension tables in the relationship and the intermediate dimension creates the many-to-many relationships between members of the primary dimension and measures in the measure group that is specified by the relationship. In order to define a many-to-many relationship between a dimension and a measure group through an intermediate measure group, the intermediate measure group must share one or more dimensions with the original measure group.

With a many-to-many dimension, values are distinct summed, which means that they do not aggregate more than once to the All member.

> **NOTE**
>
> In order to support a many-to-many dimension relationship, a primary key-foreign key relationship must be defined in the data source view between all the tables that are involved. Otherwise, you will not be able to select the correct intermediate measure group when you establish the relationship in the **Dimension Usage** tab of Cube Designer.

For more information, see Dimension Relationships, and Define a Many-to-Many Relationship and Many-to-Many Relationship Properties.

In the tasks in this topic, you define the Sales Reasons dimension and the Sales Reasons measure group, and you define a many-to-many relationship between the Sales Reasons dimension and the Internet Sales measure group through the Sales Reasons measure group.

Adding Required Tables to the Data Source View

1. Open Data Source View Designer for the **Adventure Works DW 2012** data source view.
2. Right-click anywhere in the **Diagram Organizer** pane, click **New Diagram**, and specify **Internet Sales Order Reasons** as the name for this new diagram.
3. Drag the **InternetSales** table to the **Diagram** pane from the **Tables** pane.
4. Right-click anywhere in the **Diagram** pane, and then click **Add/Remove Tables**.

5. In the **Add/Remove Tables** dialog box, add the **DimSalesReason** table and the **FactInternetSalesReason** table to the **Included objects** list, and then click **OK**.

   Notice that the primary key-foreign key relationships between the tables that are involved are established automatically because those relationships are defined in the underlying relational database. If these relationships were not defined in the underlying relational database, you would have to define them in the data source view.

6. On the **Format** menu, point to **Auto Layout**, and then click **Diagram**.

7. In the Properties window, change the **FriendlyName** property of the **DimSalesReason** table to **SalesReason**, and then change the **FriendlyName** property of the **FactInternetSalesReason** table to **InternetSalesReason**.

8. In the **Tables** pane, expand **InternetSalesReason (dbo.FactInternetSalesReason)**, click **SalesOrderNumber**, and then review the **DataType** property for this data column in the Properties window.

   Notice that the data type for the **SalesOrderNumber** column is a string data type.

9. Review the data types for the other columns in the **InternetSalesReason** table.

   Notice that the data types for the other two columns in this table are numeric data types.

10. In the **Tables** pane, right-click **InternetSalesReason (dbo.FactInternetSalesReason)**, and then click **Explore Data**.

    Notice that, for each line number within each order, a key value identifies the sales reason for the purchase of that line item, as shown in the following image.

SalesOrderNu	SalesOrderLin	SalesReasonK
SO43697	1	5
SO43697	1	9
SO43702	1	5
SO43702	1	9
SO43703	1	5
SO43703	1	9
SO43706	1	5
SO43706	1	9
SO43707	1	5
SO43707	1	9
SO43709	1	5
SO43709	1	9
SO43710	1	5
SO43710	1	9
SO43711	1	5
SO43711	1	9
SO43712	1	5

Defining the Intermediate Measure Group

1. Switch to Cube Designer for the Analysis Services Tutorial cube, and then click the **Cube Structure** tab.

2. Right-click anywhere in the **Measures** pane, and then click **New Measure Group**. For more information, see Create Measures and Measure Groups in Multidimensional Models.

3. In the **New Measure Group** dialog box, select **InternetSalesReason** in the **Select a table from the data source view** list, and then click **OK**.

   Notice that the **Internet Sales Reason** measure group now appears in the **Measures** pane.

4. Expand the **Internet Sales Reason** measure group.

Notice that only a single measure is defined for this new measure group, the **Internet Sales Reason Count** measure.

5. Select **Internet Sales Reason Count** and review the properties of this measure in the Properties window.

   Notice that the **AggregateFunction** property for this measure is defined as **Count** instead of **Sum**. Analysis Services chose **Count** because the underlying data type is a string data type. The other two columns in the underlying fact table were not selected as measures because Analysis Services detected them as numeric keys instead of as actual measures. For more information, see Define Semiadditive Behavior.

6. In the Properties window, change the **Visible** property of the **Internet Sales Reason Count** measure to **False**.

   This measure will only be used to join the Sales Reason dimension that you will define next to the Internet Sales measure group. Users will not browse this measure directly.

   The following image shows the properties for the **Internet Sales Reason Count** measure.

Defining the Many-to-Many Dimension

1. In Solution Explorer, right-click **Dimensions**, and then click **New Dimension**.

2. On the **Welcome to the Dimension Wizard** page, click **Next**.

3. On the **Select Creation Method** page, verify that the **Use an existing table** option is selected, and then click **Next**.

4. On the **Specify Source Information** page, verify that the Adventure Works DW 2012 data source view is selected.

5. In the **Main table** list, select **SalesReason**.

6. In the **Key columns** list, verify that **SalesReasonKey** is listed.

7. In the **Name column** list, select **SalesReasonName**.

8. Click **Next**.
9. On the **Select Dimension Attributes** page, the **Sales Reason Key** attribute is automatically selected because it is the key attribute. Select the check box beside the **Sales Reason Reason Type** attribute, change its name to **Sales Reason Type**, and then click **Next**.
10. On the **Completing the Wizard** page, click **Finish** to create the Sales Reason dimension.
11. On the **File** menu, click **Save All**.
12. In the **Attributes** pane of the Dimension Designer for the **Sales Reason** dimension, select **Sales Reason Key**, and then change the **Name** property in the Properties window to **Sales Reason.**
13. In the **Hierarchies** pane of the Dimension Designer, create a **Sales Reasons** user hierarchy that contains the **Sales Reason Type** level and the **Sales Reason** level, in that order.
14. In the Properties window, define **All Sales Reasons** as the value for the **AllMemberName** property of the Sales Reasons hierarchy.
15. Define **All Sales Reasons** as the value for **AttributeAllMemberName** property of the Sales Reason dimension.
16. To add the newly created dimension to the Analysis Services Tutorial cube as a cube dimension, switch to **Cube Designer**. On the **Cube Structure** tab, right-click in the **Dimensions** pane and select **Add Cube Dimension**.
17. In the **Add Cube Dimension** dialog box, select **Sales Reason** and then click **OK**.
18. On the **File** menu, click **Save All**.

Defining the Many to Many Relationship

1. Switch to Cube Designer for the Analysis Services Tutorial cube, and then click the **Dimension Usage** tab.

   Notice that the **Sales Reason** dimension has a regular relationship defined with the **Internet Sales Reason** measure group, but has no relationship defined with the **Internet Sales** or **Reseller Sales** measure groups. Notice also that the **Internet Sales Order Details** dimension has a regular relationship defined with the **Internet Sales Reason** dimension, which in turn has a **Fact Relationship** with the **Internet Sales** measure group. If this dimension was not present (or another dimension with a relationship with both the **Internet Sales Reason** and the **Internet Sales** measure group were not present), you would not be able to define the many-to-many relationship.

2. Click the cell at the intersection of the **Internet Sales** measure group and the **Sales Reason** dimension and then click the browse button (**...**).
3. In the **Define Relationship** dialog box, select **Many-to-Many** in the **Select relationship type** list.

   You have to define the intermediate measure group that connects the Sales Reason dimension to the Internet Sales measure group.

4. In the **Intermediate measure group** list, select **Internet Sales Reason**.

   The following image shows the changes in the **Define Relationship** dialog box.

5. Click **OK**.

   Notice the many-to-many icon that represents the relationship between the Sales Reason dimension and the Internet Sales measure group.

Browsing the Cube and the Many-to-Many Dimension

1. On the **Build** menu, click **Deploy Analysis Services Tutorial**.
2. When deployment has successfully completed, switch to the **Browser** tab in Cube Designer for the Analysis Services Tutorial cube, and then click **Reconnect**.
3. Add the **Internet Sales-Sales Amount** measure to the data area of the data pane.
4. Add the **Sales Reasons** user-defined hierarchy from the **Sales Reason** dimension to the row area of the data pane.
5. In the metadata pane, expand **Customer**, expand **Location**, expand **Customer Geography**, expand **Members**, expand **All Customers**, expand **Australia**, right-click **Queensland**, and then click **Add to Filter**.
6. Expand each member of the **Sales Reason Type** level to review the dollar values that are associated with each reason a customer in Queensland gave for their purchase of an Adventure Works product over the Internet.

   Notice that the totals that are associated with each sales reason add up to more than the total sales. This is because some customers cited multiple reasons for their purchase.

   The following image shows the **Filter** pane and **Data** pane of Cube Designer.

Dimension	Hierarchy	Operator	Filter Expression
Customer	Customer Geography	Equal	{ Queensland }
<Select dimension>			

Sales Reaso...	Sales Reason	Internet Sal...
Marketing	Television ...	1203.54
Other	Manufacturer	424760.16
Other	Other	11041.43
Other	Price	569067.13...
Other	Quality	375718.35
Other	Review	157451.962
Promotion	On Promotion	454888.43

Next Task in Lesson

Defining Dimension Granularity within a Measure Group

See Also

Work with Diagrams in Data Source View Designer (Analysis Services)
Dimension Relationships
Define a Many-to-Many Relationship and Many-to-Many Relationship Properties

Lesson 5-4 - Defining Dimension Granularity within a Measure Group

11/28/2018 • 11 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Users will want to dimension fact data at different granularity or specificity for different purposes. For example, sales data for reseller or internet sales may be recorded for each day, whereas sales quota information may only exist at the month or quarter level. In these scenarios, users will want a time dimension with a different grain or level of detail for each of these different fact tables. While you could define a new database dimension as a time dimension with this different grain, there is an easier way with Analysis Services.

By default in Analysis Services, when a dimension is used within a measure group, the grain of the data within that dimension is based on the key attribute of the dimension. For example, when a time dimension is included within a measure group and the default grain of the time dimension is daily, the default grain of that dimension within the measure group is daily. Many times this is appropriate, such as for the **Internet Sales** and **Reseller Sales** measure groups in this tutorial. However, when such a dimension is included in other types of measure groups, such as in a sales quota or budget measure group, a monthly or quarterly grain is generally more appropriate.

To specify a grain for a cube dimension other than the default grain, you modify the granularity attribute for a cube dimension as used within a particular measure group on the **Dimension Usage** tab of Cube Designer. When you change the grain of a dimension within a specific measure group to an attribute other than the key attribute for that dimension, you must guarantee that all other attributes in the measure group are directly or indirectly related to new granularity attribute. You do this by specifying attribute relationships between all other attributes and the attribute that is specified as the granularity attribute in the measure group. In this case, you define additional attribute relationships rather than move attribute relationships. The attribute that is specified as the granularity attribute effectively becomes the key attribute within the measure group for the remaining attributes in the dimension. If you do not specify attribute relationships appropriately, Analysis Services will not be able to aggregate values correctly, as you will see in the tasks in this topic.

For more information, see Dimension Relationships, Define a Regular Relationship and Regular Relationship Properties.

In the tasks in this topic, you add a Sales Quotas measure group and define the granularity of the Date dimension in this measure group to be monthly. You then define attribute relationships between the month attribute and other dimension attributes to ensure that Analysis Services aggregates values correctly.

Adding Tables and Defining the Sales Quotas Measure Group

1. Switch to the **Adventure Works DW 2012** data source view.
2. Right-click anywhere in the **Diagram Organizer** pane, click **New Diagram**, and then name the diagram **Sales Quotas**.
3. Drag the **Employee**, **Sales Territory**, and **Date** tables from the **Tables** pane to the **Diagram** pane.
4. Add the **FactSalesQuota** table to the **Diagram** pane by right-clicking anywhere in the **Diagram** pane and selecting **Add/Remove Tables**.

   Notice that the **SalesTerritory** table is linked to the **FactSalesQuota** table through the **Employee** table.
5. Review the columns in the **FactSalesQuota** table and then explore the data in this table.

Notice that the grain of the data within this table is the calendar quarter, which is the lowest level of detail in the FactSalesQuota table.

6. In Data Source View Designer, change the **FriendlyName** property of the **FactSalesQuota** table to **SalesQuotas**.

7. Switch to the Analysis Services Tutorial cube, and then click the **Cube Structure** tab.

8. Right-click anywhere in the **Measures** pane, click **New Measure Group**, click **SalesQuotas** in the **New Measure Group** dialog box, and then click **OK**.

   The **Sales Quotas** measure group appears in the **Measures** pane. In the **Dimensions** pane, notice that a new **Date** cube dimension is also defined, based on the **Date** database dimension. A new time-related cube dimension is defined because Analysis Services does not know which of the existing time-related cube dimensions to relate to the **DateKey** column in the **FactSalesQuota** fact table that underlies the Sales Quotas measure group. You will change this later in another task in this topic.

9. Expand the **Sales Quotas** measure group.

10. In the **Measures** pane, select **Sales Amount Quota**, and then set the value for the **FormatString** property to **Currency** in the Properties window.

11. Select the **Sales Quotas Count** measure, and then type **#,#** as the value for the **FormatString** property in the Properties window.

12. Delete the **Calendar Quarter** measure from the **Sales Quotas** measure group.

    Analysis Services detected the column that underlies the Calendar Quarter measure as a column that contains measures. However, this column and the CalendarYear column contain the values that you will use to link the Sales Quotas measure group to the Date dimension later in this topic.

13. In the **Measures** pane, right-click the **Sales Quotas** measure group, and then click **New Measure**.

    The **New Measure** dialog box opens, containing the available source columns for a measure with a usage type of **Sum**.

14. In the **New Measure** dialog box, select **Distinct count** in the **Usage** list, verify that **SalesQuotas** is selected in the **Source table** list, select **EmployeeKey** in the **Source column** list, and then click **OK**.

    Notice that the measure is created in a new measure group named **Sales Quotas 1**. Distinct count measures in SQL Server are created in their own measure groups to maximize processing performance.

15. Change the value for the **Name** property for the **Employee Key Distinct Count** measure to **Sales Person Count**, and then type **#,#** as the value for the **FormatString** property.

Browsing the Measures in the Sales Quota Measure Group by Date

1. On the **Build** menu, click **Deploy Analysis Services Tutorial**.

2. When deployment has successfully completed, click the **Browser** tab in Cube Designer for the Analysis Services Tutorial cube, and then click the **Reconnect**button.

3. Click the Excel shortcut, and then click **Enable**.

4. In the PivotTable Field List, expand the **Sales Quotas** measure group, and then drag the **Sales Amount Quota** measure to the Values area.

5. Expand the **Sales Territory** dimension, and then drag the **Sales Territories** user-defined hierarchy to Row Labels.

   Notice that the Sales Territory cube dimension is not related, directly or indirectly, to the Fact Sales Quota

table, as shown in the following image.

In the next series of steps in this topic you will define a reference dimension relationship between this dimension and this fact table.

6. Move the **Sales Territories** user hierarchy from the Rows Labels area to the Column Labels area.

7. In the PivotTable Field list, select the **Sales Territories** user-defined hierarchy, and then click the down arrow to the right.

8. In the filter, click the Select All checkbox to clear all the selections, and then choose just **North America**.

9. In the PivotTable Field List, expand **Date**.

10. Drag the **Date.Fiscal Date** user hierarchy to Row Labels

11. On the PivotTable, click the down arrow next to Row Labels. Clear all of the years except for **FY 2008**.

    Notice that only the **July 2007** member of the **Month** level appears, instead of the **July, 2007**, **August, 2007**, and **September, 2007** members of **Month** level, and that only the **July 1, 2007** member of the **Date** level appears, instead of all 31 days. This behavior occurs because the grain of the data in the fact table is at the quarter level and the grain of the **Date** dimension is the daily level. You will change this behavior in the next task in this topic.

    Notice also that the **Sales Amount Quota** value for the month and day levels is the same value as for the quarter level, $13,733,000.00. This is because the lowest level of data in the Sales Quotas measure group is at the quarter level. You will change this behavior in Lesson 6.

    The following image shows the values for **Sales Amount Quota**.

Defining Dimension Usage Properties for the Sales Quotas Measure Group

1. Open Dimension Designer for the **Employee** dimension, right-click **SalesTerritoryKey** in the **Data Source View** pane, and then click **New Attribute from Column**.

2. In the **Attributes** pane, select **SalesTerritoryKey**, and then set the **AttributeHierarchyVisible** property to **False** in the Properties window, set the **AttributeHierarchyOptimizedState** property to **NotOptimized**, and set the **AttributeHierarchyOrdered** property to **False**.

   This attribute is required to link the **Sales Territory** dimension to the **Sales Quotas** and **Sales Quotas 1** measure groups as a referenced dimension.

3. In Cube Designer for the Analysis Services Tutorial cube, click the **Dimension Usage** tab, and then review the dimension usage within the **Sales Quotas** and **Sales Quotas 1** measure groups.

   Notice that the **Employee** and **Date** cube dimensions are linked to the **Sales Quotasand Sales Quotas 1** measure groups through regular relationships. Notice also that the **Sales Territory** cube dimension is not linked to either of these measure groups.

4. Click the cell at the intersection of the **Sales Territory** dimension and the **Sales Quotas** measure group and then click the browse button (**...**). The **Define Relationship** dialog box opens.

5. In the **Select relationship type** list, select **Referenced**.

6. In the **Intermediate dimension** list, select **Employee**.

7. In the **Reference dimension attribute** list, select **Sales Territory Region.**

8. In the **Intermediate dimension attribute** list, select **Sales Territory Key**. (The key column for the Sales Territory Region attribute is the SalesTerritoryKey column.)

9. Verify that the **Materialize** check box is selected.

10. Click **OK**.

11. Click the cell at the intersection of the **Sales Territory** dimension and the **Sales Quotas 1** measure group and then click the browse button (**...**). The **Define Relationship** dialog box opens.

12. In the **Select relationship type** list, select **Referenced**.

13. In the **Intermediate dimension** list, select **Employee**.

14. In the **Reference dimension attribute** list, select **Sales Territory Region.**

15. In the **Intermediate dimension attribute** list, select **Sales Territory Key**. (The key column for the Sales Territory Region attribute is the SalesTerritoryKey column.)

16. Verify that the **Materialize** check box is selected.

17. Click **OK**.

18. Delete the **Date** cube dimension.

    Instead of having four time-related cube dimensions, you will use the **Order Date** cube dimension in the **Sales Quotas** measure group as the date against which sales quotas will be dimensioned. You will also use this cube dimension as the primary date dimension in the cube.

19. In the **Dimensions** list, rename the **Order Date** cube dimension to **Date**.

    Renaming the **Order Date** cube dimension to **Date** makes it easier for users to understand its role as the primary date dimension in this cube.

20. Click the browse button (**...**) in the cell at the intersection of the **Sales Quotas** measure group and the **Date** dimension.

21. In the **Define Relationship** dialog box, select **Regular** in the **Select relationship type** list.

22. In the **Granularity attribute** list, select **Calendar Quarter**.

    Notice that a warning appears to notify you that because you have selected a non-key attribute as the granularity attribute, you must make sure that all other attributes are directly or indirectly related to the granularity attribute by specifying them as member properties.

23. In the **Relationship** area of the **Define Relationship** dialog box, link the **CalendarYear** and **CalendarQuarter** dimension columns from the table that underlies the Date cube dimension to the **CalendarYear** and **CalendarQuarter** columns in the table that underlies the Sales Quota measure group, and then click **OK**.

    > **NOTE**
    > The Calendar Quarter is defined as the granularity attribute for the Date cube dimension in the Sales Quotas measure group, but the Date attribute continues to be the granularity attribute for the Internet Sales and Reseller Sales measure groups.

24. Repeat the previous four steps for the **Sales Quotas 1** measure group.

Defining Attribute Relationships Between the Calendar Quarter Attribute and the Other Dimension Attributes in the Date Dimension

1. Switch to **Dimension Designer** for the **Date** dimension, and then click the **Attribute Relationships** tab.

   Notice that although **Calendar Year** is linked to **Calendar Quarter** through the **Calendar Semester** attribute, the fiscal calendar attributes are linked only to one another; they are not linked to the **Calendar Quarter** attribute and therefore will not aggregate correctly in the **Sales Quotas** measure group.

2. In the diagram, right-click the **Calendar Quarter** attribute and then select **New Attribute Relationship**.
3. In the **Create Attribute Relationship** dialog box, the **Source Attribute** is **Calendar Quarter**. Set the **Related Attribute** to **Fiscal Quarter**.
4. Click **OK**.

   Notice that a warning message appears stating that the **Date** dimension contains one or more redundant attribute relationships that may prevent data from being aggregated when a non-key attribute is used as a granularity attribute.
5. Delete the attribute relationship between the **Month Name** attribute and the **Fiscal Quarter** attribute.
6. On the **File** menu, click **Save All**.

Browsing the Measures in the Sales Quota Measure Group by Date

1. On the **Build** menu, click **Deploy Analysis Services Tutorial**.
2. When deployment has successfully completed, click the **Browser** tab in Cube Designer for the Analysis Services Tutorial cube, and then click **Reconnect**.
3. Click the Excel shortcut, and then click **Enable**.
4. Drag the **Sales Amount Quota** measure to the Values area.
5. Drag the **Sales Territories** user hierarchy to the Column Labels, and then filter on **North America**.
6. Drag the **Date.FiscalDate** user hierarchy to the Row Labels, and then click the down arrow next to **Row Labels** on the PivotTable, and clear all check boxes other than **FY 2008**, to display only fiscal year 2008.
7. Click OK.
8. Expand **FY 2008**, expand **H1 FY 2008**, and then expand **Q1 FY 2008**.

   The following image shows a PivotTable for the Analysis Services Tutorial cube, with the Sales Quota measure group dimensioned correctly.

   Notice that each member of the fiscal quarter level has the same value as the quarter level. Using **Q1 FY 2008** as an example, the quota of $9,180,000.00 for **Q1 FY 2008** is also the value for each of its members. This behavior occurs because the grain of the data in the fact table is at the quarter level and the grain of the Date dimension is also at the quarter level. In Lesson 6, you will learn how to allocate the quarterly amount proportionally to each month.

Next Lesson

Lesson 6: Defining Calculations

See Also

Dimension Relationships
Define a Regular Relationship and Regular Relationship Properties
Work with Diagrams in Data Source View Designer (Analysis Services)

Lesson 6: Defining Calculations

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In this lesson, you learn to define calculations, which are Multidimensional Expressions (MDX) expressions or scripts. Calculations enable you to define calculated members, named sets, and execute other script commands to extend the capabilities of an Analysis Services cube. For example, you can run a script command to define a subcube and then assign a calculation to the cells in the subcube.

When you define a new calculation in Cube Designer, the calculation is added to the **Script Organizer** pane of the **Calculations** tab of Cube Designer, and the fields for the particular calculation type are displayed in a calculations form in the **Calculation Expressions** pane. Calculations are executed in the order in which they are listed in the **Script Organizer** pane. You can reorder the calculations by right-clicking on a particular calculation and then selecting **Move Up** or **Move Down**, or by clicking a particular calculation and then using the **Move Up** or **Move Down** icons on the toolbar of the **Calculations** tab.

On the **Calculations** tab, you can add new calculations and view or edit existing calculations in the following views in the **Calculation Expressions** pane:

- Form view. This view shows the expressions and properties for a single command in a graphical format. When you edit an MDX script, an expression box fills the Form view.
- Script view. This view displays all calculation scripts in a code editor, which lets you easily change the calculation scripts. When the **Calculation Expressions** pane is in Script view, the **Script Organizer** is hidden. The Script view provides color coding, parenthesis matching, auto-complete, and MDX code regions. You can expand or collapse the MDX code regions to make editing easier.

To switch between these views in the **Calculation Expressions** pane, click **Form View** or **Script View** on the toolbar of the **Calculations** tab.

> **NOTE**
>
> If Analysis Services detects a syntax error in any calculation, the Form view will not display until the error is corrected in the Script view.

You can also use the Business Intelligence Wizard to add certain calculations to a cube. For example, you can use this wizard to add time intelligence to a cube, which means defining calculated members for time-related calculations such as period-to-date, moving averages, or period over period growth. For more information, see Define Time Intelligence Calculations using the Business Intelligence Wizard.

> **IMPORTANT**
>
> On the **Calculations** tab, the calculation script starts with the CALCULATE command. The CALCULATE command controls the aggregation of the cells in the cube and you should edit this command only if you intend to manually specify how the cube cells should be aggregated.

For more information, see Calculations, and Calculations in Multidimensional Models.

> **NOTE**
> Completed projects for all of the lessons in this tutorial are available online. You can jump ahead to any lesson by using the completed project from the previous lesson as a starting point. Click here to download the sample projects that go with this tutorial.

This lesson contains the following tasks:

Defining Calculated Members
In this task, you learn to define calculated members.

Defining Named Sets
In this task, you learn to define named sets.

Next Lesson

Lesson 7: Defining Key Performance Indicators (KPIs)

See Also

Analysis Services Tutorial Scenario
Multidimensional Modeling (Adventure Works Tutorial)
Create Named Sets
Create Calculated Members

Lesson 6-1 - Defining Calculated Members

5/16/2018 • 11 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Calculated members are members of a dimension or a measure group that are defined based on a combination of cube data, arithmetic operators, numbers, and functions. For example, you can create a calculated member that calculates the sum of two physical measures in the cube. Calculated member definitions are stored in cubes, but their values are calculated at query time.

To create a calculated member, use the **New Calculated Member** command on the **Calculations** tab of Cube Designer. You can create a calculated member within any dimension, including the measures dimension. You can also place a calculated member within a display folder in the **Calculation Properties** dialog box. For more information, see Calculations, Calculations in Multidimensional Models, and Create Calculated Members.

In the tasks in this topic, you define calculated measures to let users view the gross profit margin percentage and sales ratios for Internet sales, reseller sales, and for all sales.

Defining Calculations to Aggregate Physical Measures

1. Open Cube Designer for the Analysis Services Tutorial cube, and then click the **Calculations** tab.

   Notice the default CALCULATE command in the **Calculation Expressions** pane and in the **Script Organizer** pane. This command specifies that the measures in the cube should be aggregated according to the value that is specified by their AggregateFunction properties. Measure values are generally summed, but may also be counted or aggregated in some other manner.

   The following image shows the **Calculations** tab of Cube Designer.

2. On the toolbar of the **Calculations** tab, click **New Calculated Member**.

   A new form appears in the **Calculation Expressions** pane within which you define the properties of this new calculated member. The new member also appears in the **Script Organizer** pane.

   The following image shows the form that appears in the **Calculation Expressions** pane when you click **New Calculated Member**.

3. In the **Name** box, change the name of the calculated measure to **[Total Sales Amount]**.

   If the name of a calculated member contains a space, the calculated member name must be enclosed in square brackets.

   Notice in the **Parent hierarchy** list that, by default, a new calculated member is created in the **Measures** dimension. A calculated member in the Measures dimension is also frequently called a calculated measure.

4. On the **Metadata** tab in the **Calculation Tools** pane of the **Calculations** tab, expand **Measures** and then expand **Internet Sales** to view the metadata for the **Internet Sales** measure group.

   You can drag metadata elements from the **Calculation Tools** pane into the **Expression** box and then add operators and other elements to create Multidimensional Expressions (MDX) expressions. Alternatively, you can type the MDX expression directly into the **Expression** box.

   > **NOTE**
   >
   > If you cannot view any metadata in the **Calculation Tools** pane, click **Reconnect** on the toolbar. If this does not work, you may have to process the cube or start the instance of Analysis Services.

5. Drag **Internet Sales-Sales Amount** from the **Metadata** tab in the **Calculation Tools** pane into the **Expression** box in the **Calculation Expressions** pane.

6. In the **Expression** box, type a plus sign (+) after **[Measures].[Internet Sales-Sales Amount]**.

7. On the **Metadata** tab in the **Calculation Tools** pane, expand **Reseller Sales**, and then drag **Reseller Sales-Sales Amount** into the **Expression** box in the **Calculation Expressions** pane after the plus sign (+).

8. In the **Format string** list, select **"Currency".**

9. In the **Non-empty behavior** list, select the check boxes for **Internet Sales-Sales Amount** and **Reseller Sales-Sales Amount**, and then click **OK**.

   The measures you specify in the **Non-empty behavior** list are used to resolve NON EMPTY queries in MDX. When you specify one or more measures in the **Non-empty behavior** list, Analysis Services treats the calculated member as empty if all the specified measures are empty. If the **Non-empty behavior** property is blank, Analysis Services must evaluate the calculated member itself to determine whether the member is empty.

   The following image shows the **Calculation Expressions** pane populated with the settings that you specified in the previous steps.

10. On the toolbar of the **Calculations** tab, click **Script View**, and then review the calculation script in the **Calculation Expressions** pane.

    Notice that the new calculation is added to the initial CALCULATE expression; each individual calculation is separated by a semicolon. Notice also that a comment appears at the beginning of the calculation script. Adding comments within the calculation script for groups of calculations is a good practice, to help you and other developers understand complex calculation scripts.

11. Add a new line in the calculation script after the **Calculate;** command and before the newly added calculation script, and then add the following text to the script on its own line:

```
/* Calculations to aggregate Internet Sales and Reseller Sales measures */
```

    The following image shows the calculation scripts as they should appear in the **Calculation Expressions** pane at this point in the tutorial.

```
/*
The CALCULATE command controls the aggregation of leaf cells in the cube.
If the CALCULATE command is deleted or modified, the data within the cube is affected.
You should edit this command only if you manually specify how the cube is aggregated.
*/
CALCULATE;
/* Calculations to aggregate Internet Sales and Reseller Sales measures */
CREATE MEMBER CURRENTCUBE.[MEASURES].[Total Sales Amount]
 AS [Measures].[Internet Sales-Sales Amount]+[Measures].[Reseller Sales-Sales Amount],
FORMAT_STRING = "Currency",
NON_EMPTY_BEHAVIOR = { [Internet Sales-Sales Amount], [Reseller Sales-Sales Amount] },
VISIBLE = 1  ;
```

12. On the toolbar of the **Calculations** tab, click **Form View**, verify that **[Total Sales Amount]** is selected in the **Script Organizer** pane, and then click **New Calculated Member**.

13. Change the name of this new calculated member to **[Total Product Cost]**, and then create the following expression in the **Expression** box:

```
[Measures].[Internet Sales-Total Product Cost] + [Measures].[Reseller Sales-Total Product Cost]
```

14. In the **Format string** list, select **"Currency"**.

15. In the **Non-empty behavior** list, select the check boxes for **Internet Sales-Total Product Cost** and **Reseller Sales-Total Product Cost**, and then click **OK**.

    You have now defined two calculated members, both of which are visible in the **Script Organizer** pane. These calculated members can be used by other calculations that you define later in the calculation script. You can view the definition of any calculated member by selecting the calculated member in the **Script Organizer** pane; the definition of the calculated member will appear in the **Calculation Expressions** pane in the Form view. Newly defined calculated members will not appear in the **Calculation Tools** pane until these objects have been deployed. Calculations do not require processing.

Defining Gross Profit Margin Calculations

1. Verify that **[Total Product Cost]** is selected in the **Script Organizer** pane, and then click **New Calculated Member** on the toolbar of the **Calculations** tab.

2. In the **Name** box, change the name of this new calculated measure to **[Internet GPM]**.

3. In the **Expression** box, create the following MDX expression:

```
([Measures].[Internet Sales-Sales Amount] -
[Measures].[Internet Sales-Total Product Cost]) /
[Measures].[Internet Sales-Sales Amount]
```

4. In the **Format string** list, select **"Percent"**.

5. In the **Non-empty behavior** list, select the check box for **Internet Sales-Sales Amount**, and then click **OK**.

6. On the toolbar of the **Calculations** tab, click **New Calculated Member**.

7. In the **Name** box, change the name of this new calculated measure to **[Reseller GPM]**.

8. In the **Expression** box, create the following MDX expression:

```
([Measures].[Reseller Sales-Sales Amount] -
[Measures].[Reseller Sales-Total Product Cost]) /
[Measures].[Reseller Sales-Sales Amount]
```

9. In the **Format string** list, select **"Percent"**.
10. In the **Non-empty behavior** list, select the check box for **Reseller Sales-Sales Amount**, and then click **OK**.
11. On the toolbar of the **Calculations** tab, click **New Calculated Member**.
12. In the **Name** box, change the name of this calculated measure to **[Total GPM]**.
13. In the **Expression** box, create the following MDX expression:

```
([Measures].[Total Sales Amount] -
[Measures].[Total Product Cost]) /
[Measures].[Total Sales Amount]
```

Notice that this calculated member is referencing other calculated members. Because this calculated member will be calculated after the calculated members that it references, this is a valid calculated member.

14. In the **Format string** list, select **"Percent"**.
15. In the **Non-empty behavior** list, select the check boxes for **Internet Sales-Sales Amount** and **Reseller Sales-Sales Amount**, and then click **OK**.
16. On the toolbar of the **Calculations** tab, click **Script View** and review the three calculations you just added to the calculation script.
17. Add a new line in the calculation script immediately before the **[Internet GPM]** calculation, and then add the following text to the script on its own line:

```
/* Calculations to calculate gross profit margin */
```

The following image shows the **Expressions** pane with the three new calculations.

```
/* Calculations to calculate gross profit margin */
CREATE MEMBER CURRENTCUBE.[MEASURES].[Internet GPM]
 AS ([Measures].[Internet Sales-Sales Amount] -
    [Measures].[Internet Sales-Total Product Cost]) /
    [Measures].[Internet Sales-Sales Amount],
FORMAT_STRING = "Percent",
NON_EMPTY_BEHAVIOR = { [Internet Sales-Sales Amount] },
VISIBLE = 1  ;
CREATE MEMBER CURRENTCUBE.[MEASURES].[Reseller GPM]
 AS ([Measures].[Reseller Sales-Sales Amount] -
    [Measures].[Reseller Sales-Total Product Cost]) /
    [Measures].[Reseller Sales-Sales Amount],
FORMAT_STRING = "Percent",
NON_EMPTY_BEHAVIOR = { [Reseller Sales-Sales Amount] },
VISIBLE = 1  ;
CREATE MEMBER CURRENTCUBE.[MEASURES].[Total GPM]
 AS ([Measures].[Total Sales Amount] -
    [Measures].[Total Product Cost]) /
    [Measures].[Total Sales Amount],
FORMAT_STRING = "Percent",
NON_EMPTY_BEHAVIOR = { [Internet Sales-Sales Amount], [Reseller Sales-Sales Amount] },
VISIBLE = 1  ;
```

Defining the Percent of Total Calculations

1. On the toolbar of the **Calculations** tab, click **Form View**.
2. In the **Script Organizer** pane, select **[Total GPM]**, and then click **New Calculated Member** on the toolbar of the **Calculations** tab.

   Clicking the final calculated member in the **Script Organizer** pane before you click **New Calculated Member** guarantees that the new calculated member will be entered at the end of the script. Scripts execute in the order that they appear in the **Script Organizer** pane.
3. Change the name of this new calculated member to **[Internet Sales Ratio to All Products]**.
4. Type the following expression in the **Expression** box:

   ```
   Case
       When IsEmpty( [Measures].[Internet Sales-Sales Amount] )
       Then 0
       Else ( [Product].[Product Categories].CurrentMember,
              [Measures].[Internet Sales-Sales Amount]) /
            ( [Product].[Product Categories].[(All)].[All],
              [Measures].[Internet Sales-Sales Amount] )
       End
   ```

   This MDX expression calculates the contribution to total Internet sales of each product. The Case statement together with the IS EMPTY function ensures that a divide by zero error does not occur when a product has no sales.
5. In the **Format string** list, select **"Percent"**.
6. In the **Non-empty behavior** list, select the check box for **Internet Sales-Sales Amount**, and then click **OK**.
7. On the toolbar of the **Calculations** tab, click **New Calculated Member**.
8. Change the name of this calculated member to **[Reseller Sales Ratio to All Products]**.
9. Type the following expression in the **Expression** box:

   ```
   Case
       When IsEmpty( [Measures].[Reseller Sales-Sales Amount] )
       Then 0
       Else ( [Product].[Product Categories].CurrentMember,
              [Measures].[Reseller Sales-Sales Amount]) /
            ( [Product].[Product Categories].[(All)].[All],
              [Measures].[Reseller Sales-Sales Amount] )
       End
   ```
10. In the **Format string** list, select **"Percent"**.
11. In the **Non-empty behavior** list, select the check box for **Reseller Sales-Sales Amount**, and then click **OK**.
12. On the toolbar of the **Calculations** tab, click **New Calculated Member**.
13. Change the name of this calculated member to **[Total Sales Ratio to All Products]**.
14. Type the following expression in the **Expression** box:

```
Case
    When IsEmpty( [Measures].[Total Sales Amount] )
    Then 0
    Else ( [Product].[Product Categories].CurrentMember,
           [Measures].[Total Sales Amount]) /
         ( [Product].[Product Categories].[(All)].[All],
           [Measures].[Total Sales Amount] )
    End
```

15. In the **Format string** list, select **"Percent"**.
16. In the **Non-empty behavior** list, select the check boxes for **Internet Sales-Sales Amount** and **Reseller Sales-Sales Amount**, and then click **OK**.
17. On the toolbar of the **Calculations** tab, click **Script View**, and then review the three calculations that you just added to the calculation script.
18. Add a new line in the calculation script immediately before the **[Internet Sales Ratio to All Products]** calculation, and then add the following text to the script on its own line:

    ```
    /* Calculations to calculate percentage of product to total product sales */
    ```

    You have now defined a total of eight calculated members, which are visible in the **Script Organizer** pane when you are in Form view.

Browsing the New Calculated Members

1. On the **Build** menu of SQL Server Data Tools (SSDT), click **Deploy Analysis Services Tutorial**.
2. When deployment has successfully completed, switch to the **Browser** tab, click **Reconnect**.
3. Click the Excel icon, and then click **Enable**.
4. In the **PivotTable Field List** pane, expand **Values** folder to view the new calculated members in the Measures dimension.
5. Drag the **Total Sales Amount** to the Values area, and then review the results.

   Drag **Internet Sales-Sales Amount** and **Reseller Sales-Sales Amount** measures from the **Internet Sales** and **Reseller Sales** measure groups to the Values area.

   Notice that the **Total Sales Amount** measure is the sum of the **Internet Sales-Sales Amount** measure and the **Reseller Sales-Sales Amount** measure.
6. Add the **Product Categories** user-defined hierarchy to the filter area of the **Report Filter** area, and then filter the data by **Mountain Bikes**.

   Notice that the **Total Sales Amount** measure is calculated for the **Mountain Bikes** category of product sales based on the **Internet Sales-Sales Amount** and the **Reseller Sales-Sales Amount** measures for **Mountain Bikes**.
7. Add the **Date.Calendar Date** user-defined hierarchy to the Row labels area, and then review the results.

   Notice that the **Total Sales Amount** measure for each calendar year is calculated for the **Mountain Bikes** category of product sales based on the **Internet Sales-Sales Amount** and the **Reseller Sales-Sales Amount** measures for **Mountain Bikes**.
8. Add the **Total GPM**, **Internet GPM**, and **Reseller GPM** measures to the Values area, and then review the results.

Notice that the gross profit margin for reseller sales is significantly lower than for sales over the Internet, as shown in the following image.

9. Add the **Total Sales Ratio to All Products**, **Internet Sales Ratio to All Products**, and **Reseller Sales Ratio to All Products** measures to the Values area.

   Notice that the ratio of the sales of mountain bikes to all products has increased over time for Internet sales, but is decreasing over time for reseller sales. Notice also that the ratio of the sale of mountain bikes to all products is lower from sales through resellers than it is for sales over the Internet.

10. Change the filter from **Mountain Bikes** to **Bikes**, and review the results.

    Notice that the gross profit margin for all bikes sold through resellers is negative, because touring bikes and road bikes are being sold at a loss.

11. Change the filter to **Accessories**, and then review the results.

    Notice that the sale of accessories is increasing over time, but that these sales make up only a small fraction of total sales. Notice also that the gross profit margin for sales of accessories is higher than for bikes.

Next Task in Lesson

Defining Named Sets

See Also

Calculations
Calculations in Multidimensional Models
Create Calculated Members

Lesson 6-2 - Defining Named Sets

11/28/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A named set is a Multidimensional Expressions (MDX) expression that returns a set of dimension members. You can define named sets and save them as part of the cube definition; you can also create named sets in client applications. You create named sets by combining cube data, arithmetic operators, numbers, and functions. Named sets can be used by users in MDX queries in client applications and can also be used to define sets in subcubes. A subcube is a collection of crossjoined sets that restricts the cube space to the defined subspace for subsequent statements. Defining a restricted cube space is a fundamental concept to MDX scripting.

Named sets simplify MDX queries and provide useful aliases for complex, typically used, set expressions. For example, you can define a named set called Large Resellers that contains the set of members in the Reseller dimension that have the most employees. End users could then use the Large Resellers named set in queries, or you could use the named set to define a set in a subcube. Named set definitions are stored in cubes, but their values exist only in memory. To create a named set, use the **New Named Set** command on the **Calculations** tab of Cube Designer. For more information, see Calculations, Create Named Sets.

In the tasks in this topic, you will define two named sets: a Core Products named set and a Large Resellers named set.

Defining a Core Products Named Set

1. Switch to the **Calculations** tab of Cube Designer for the Analysis Services Tutorial cube, and then click **Form View** on the toolbar.

2. Click **[Total Sales Ratio to All Products]** in the **Script Organizer** pane, and then click **New Named Set** on the toolbar of the **Calculations** tab.

   When you define a new calculation on the **Calculations** tab, remember that calculations are resolved in the order in which they appear in the **Script Organizer** pane. Your focus within that pane when you create a new calculation determines the order of the execution of the calculation; a new calculation is defined immediately after the calculation on which you are focused.

3. In the **Name** box, change the name of the new named set to **[Core Products]**.

   In the **Script Organizer** pane, notice the unique icon that differentiates a named set from a script command or a calculated member.

4. On the **Metadata** tab in the **Calculation Tools** pane, expand **Product**, expand **Category**, expand **Members**, and then expand **All Products**.

   > **NOTE**
   >
   > If you cannot view any metadata in the **Calculation Tools** pane, click **Reconnect** on the toolbar. If this does not work, you may have to process the cube or start the instance of Analysis Services.

5. Drag **Bikes** into the **Expression** box.

   You now have created a set expression that will return the set of members that are in the Bike category in the Product dimension.

Defining a Large Resellers Named Set

1. Right-click **[Core Products]** in the **Script Organizer** pane, and then click **New Named Set**.

2. In the **Name** box, change the name of this named set to **[Large Resellers]**.

3. In the **Expression** box, type **Exists()**.

   You will use the Exists function to return the set of members from the Reseller Name attribute hierarchy that intersects with the set of members in the Number of Employees attribute hierarchy that has the largest number of employees.

4. On the **Metadata** tab in the **Calculation Tools** pane, expand the **Reseller** dimension, and then expand the **Reseller Name** attribute hierarchy.

5. Drag the **Reseller Name** level into the parenthesis for the Exists set expression.

   You will use the Members function to return all members of this set. For more information, see Members (Set) (MDX).

6. After the partial set expression, type a period, and then add the Members function. Your expression should look like the following:

   ```
   Exists([Reseller].[Reseller Name].[Reseller Name].Members)
   ```

   Now that you have defined the first set for the Exists set expression, you are ready to add the second set-the set of members of the Reseller dimension that contains the largest number of employees.

7. On the **Metadata** tab in the **Calculation Tools** pane, expand **Number of Employees** in the Reseller dimension, expand **Members**, and then expand **All Resellers**.

   Notice that the members of this attribute hierarchy are not grouped.

8. Open Dimension Designer for the **Reseller** dimension, and then click **Number of Employees** in the **Attributes** pane.

9. In the Properties window, change the **DiscretizationMethod** property to **Automatic**, and then change the **DiscretizationBucketCount** property to **5**. For more information, see Group Attribute Members (Discretization).

10. On the **Build** menu of SQL Server Data Tools (SSDT), click **Deploy Analysis Services Tutorial**.

11. When deployment has successfully completed, switch to Cube Designer for the Analysis Services Tutorial cube, and then click **Reconnect** on the toolbar of the **Calculations** tab.

12. On the **Metadata** tab in the **Calculation Tools** pane, expand **Number of Employees** in the **Reseller** dimension, expand **Members**, and then expand **All Resellers**.

    Notice that the members of this attribute hierarchy are now contained in five groups, numbered 0 through 4. To view the number of a group, pause the pointer over that group to view an InfoTip. For the range `2 -17`, the InfoTip should contain `[Reseller].[Number of Employees].&[0]`.

    The members of this attribute hierarchy are grouped because the DiscretizationBucketCount property is set to **5** and the DiscretizationMethod property is set to **Automatic**.

13. In the **Expression** box, add a comma in the Exists set expression after the Members function and before the closing parenthesis, and then drag **83 - 100** from the **Metadata** pane and position it after the comma.

    You have now completed the Exists set expression that will return the set of members that intersects with these two specified sets, the set of all resellers and the set of resellers who have 83 to 100 employees, when

the Large Resellers named set is put on an axis.

The following image shows the **Calculation Expressions** pane for the **[Large Resellers]** named set.

14. On the toolbar of the **Calculations** tab, click **Script View**, and then review the two named sets that you have just added to the calculation script.

15. Add a new line in the calculation script immediately before the first CREATE SET command, and then add the following text to the script on its own line:

```
/* named sets */
```

You have now defined two named sets, which are visible in the **Script Organizer** pane. You are now ready to deploy these named sets, and then to browse these measures in the Analysis Services Tutorial cube.

Browsing the Cube by Using the New Named Sets

1. On the **Build** menu of SQL Server Data Tools, click **Deploy Analysis Services Tutorial**.

2. When deployment has successfully completed, click the **Browser** tab, and then click **Reconnect**.

3. Clear the grid in the data pane.

4. Add the **Reseller Sales-Sales Amount** measure to the data area.

5. Expand the Product dimension, and then add Category and Subcategory to the row area, as shown in the following image.

Dimension	Hierarchy	Operator	Filter Expression
<Select dimension>			

Category	Subcategory	Reseller Sales-..
Accessories	Bike Racks	197736.156
Accessories	Bottles and Cages	7476.603599...
Accessories	Cleaners	11188.3725
Accessories	Helmets	258712.9323...
Accessories	Hydration Packs	65518.74850...
Accessories	Locks	16225.22
Accessories	Pumps	13514.6873
Acessories	Tires and Tubes	925.2076
Bikes	Mountain Bikes	26492684.37...
Bikes	Road Bikes	29358206.96...
Bikes	Touring Bikes	10451490.21...
Clothing	Bib-Shorts	166739.7086...
Clothing	Caps	31541.34610...
Clothing	Gloves	207775.1742...
Clothing	Jerseys	579308.7083...

6. In the **Metadata** pane, in the **Product** dimension, drag **Core Products** to the filter area.

Notice that only the **Bike** member of the **Category** attribute and members of the **Bike** subcategories remain in the cube. This is because the **Core Products** named set is used to define a subcube. This subcube limits the members of the **Category** attribute in the **Product** dimension within the subcube to those members of the **Core Product** named set, as shown in the following image.

Dimension	Hierarchy	Operator	Filter Expression
Product	Category	In	Core Products
<Select dimension>			

Category	Subcategory	Reseller Sales-...
Bikes	Mountain Bikes	26492684.37...
Bikes	Road Bikes	29358206.96...
Bikes	Touring Bikes	10451490.21...

7. In the **Metadata** pane, expand **Reseller**, add **Large Resellers** to the filter area.

   Notice that the Reseller Sales Amount measure in the Data pane only displays sales amounts for large resellers of bikes. Notice also that the Filter pane now displays the two named sets that are used to define this particular subcube, as shown in the following image.

Dimension	Hierarchy	Operator	Filter Expression
Product	Category	In	Core Products
Reseller	Reseller Name	In	Large Resellers
<Select dimension>			

Category	Subcategory	Reseller Sales-...
Bikes	Mountain Bikes	5748640.5721
Bikes	Road Bikes	6985271.3659
Bikes	Touring Bikes	3357480.2444

Next Lesson

Lesson 7: Defining Key Performance Indicators (KPIs)

See Also

Calculations
Create Named Sets

Lesson 7: Defining Key Performance Indicators (KPIs)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In this lesson, you learn to define Key Performance Indicators (KPIs) in your Analysis Services project. KPIs provide a framework for defining server-side calculations that measure your business, and they standardize how the resulting information is displayed. KPIs can be displayed in reports, portals, and dashboards, through data access APIs, and through Microsoft tools and third-party tools. KPIs are metadata wrappers around regular measures and other Multidimensional Expressions (MDX) expressions. For more information, see Key Performance Indicators (KPIs) in Multidimensional Models.

> **NOTE**
> Completed projects for all of the lessons in this tutorial are available online. You can jump ahead to any lesson by using the completed project from the previous lesson as a starting point. Click here to download the sample projects that go with this tutorial.

This lesson contains the following task:

Defining and Browsing KPIs
In this task, you define KPIs in the Form view and then switch to the Browser view to browse the cube data by using the KPIs.

Next Lesson

Lesson 8: Defining Actions

See Also

Analysis Services Tutorial Scenario
Multidimensional Modeling (Adventure Works Tutorial)
Key Performance Indicators (KPIs) in Multidimensional Models

Lesson 7-1 - Defining and Browsing KPIs

5/16/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

To define key performance indicators (KPIs), you first define a KPI name and the measure group to which the KPI is associated. A KPI can be associated with all measure groups or with a single measure group. You then define the following elements of the KPI:

- The value expression

  A value expression is a physical measure such as Sales, a calculated measure such as Profit, or a calculation that is defined within the KPI by using a Multidimensional Expressions (MDX) expression.

- The goal expression

  A goal expression is a value, or an MDX expression that resolves to a value, that defines the target for the measure that the value expression defines. For example, a goal expression could be the amount by which the business managers of a company want to increase sales or profit.

- The status expression

  A status expression is an MDX expression that Analysis Services uses to evaluate the current status of the value expression compared to the goal expression. A goal expression is a normalized value in the range of -1 to +1, where -1 is very bad, and +1 is very good. The status expression displays a graphic to help you easily determine the status of the value expression compared to the goal expression.

- The trend expression

  A trend expression is an MDX expression that Analysis Services uses to evaluate the current trend of the value expression compared to the goal expression. The trend expression helps the business user to quickly determine whether the value expression is becoming better or worse relative to the goal expression. You can associate one of several graphics with the trend expression to help business users be able to quickly understand the trend.

In addition to these elements that you define for a KPI, you also define several properties of a KPI. These properties include a display folder, a parent KPI if the KPI is computed from other KPIs, the current time member if there is one, the weight of the KPI if it has one, and a description of the KPI.

> **NOTE**
>
> For more examples of KPIs, see the KPI examples on the Templates tab in the Calculation Tools pane or in the examples in the **Adventure Works DW 2012** sample data warehouse. For more information about how to install this database, see Install Sample Data and Projects for the Analysis Services Multidimensional Modeling Tutorial.

In the task in this lesson, you define KPIs in the Analysis Services Tutorial project, and you then browse the Analysis Services Tutorial cube by using these KPIs. You will define the following KPIs:

- Reseller Revenue

  This KPI is used to measure how actual reseller sales compare to sales quotas for reseller sales, how close the sales are to the goal, and what the trend is toward reaching the goal.

- Product Gross Profit Margin

This KPI is used to determine how close the gross profit margin is for each product category to a specified goal for each product category, and also to determine the trend toward reaching this goal.

Defining the Reseller Revenue KPI

1. Open Cube Designer for the Analysis Services Tutorial cube, and then click the **KPIs** tab.

   The **KPIs** tab includes several panes. On the left side of the tab are the **KPI Organizer** pane and the **Calculation Tools** pane. The display pane in the middle of the tab contains the details of the KPI that is selected in the **KPI Organizer** pane.

   The following image shows the **KPIs** tab of Cube Designer.

2. On the toolbar of the **KPIs** tab, click the **New KPI** button.

   A blank KPI template appears in the display pane, as shown in the following image.

3. In the **Name** box, type **Reseller Revenue**, and then select **Reseller Sales** in the **Associated measure group** list.

4. On the **Metadata** tab in the **Calculation Tools** pane, expand **Measures**, expand **Reseller Sales**, and then drag the **Reseller Sales-Sales Amount** measure to the **Value Expression** box.

5. On the **Metadata** tab in the **Calculation Tools** pane, expand **Measures**, expand **Sales Quotas**, and then drag the **Sales Amount Quota** measure to the **Goal Expression** box.

6. Verify that **Gauge** is selected in the **Status indicator** list, and then type the following MDX expression in the **Status expression** box:

```
Case
 When
  KpiValue("Reseller Revenue")/KpiGoal("Reseller Revenue")>=.95
   Then 1
 When
  KpiValue("Reseller Revenue")/KpiGoal("Reseller Revenue")<.95
   And
  KpiValue("Reseller Revenue")/KpiGoal("Reseller Revenue")>=.85
   Then 0
  Else-1
End
```

This MDX expression provides the basis for evaluating the progress toward the goal. In this MDX expression, if actual reseller sales are more than 85 percent of the goal, a value of 0 is used to populate the chosen graphic. Because a gauge is the chosen graphic, the pointer in the gauge will be half-way between empty and full. If actual reseller sales are more the 90 percent, the pointer on the gauge will be three-fourths of the way between empty and full.

7. Verify that **Standard arrow** is selected in the **Trend indicator** list, and then type the following expression in the **Trend expression** box:

```
Case
 When IsEmpty
  (ParallelPeriod
   ([Date].[Calendar Date].[Calendar Year],1,
       [Date].[Calendar Date].CurrentMember))
  Then 0
 When  (
  KpiValue("Reseller Revenue") -
   (KpiValue("Reseller Revenue"),
    ParallelPeriod
     ([Date].[Calendar Date].[Calendar Year],1,
       [Date].[Calendar Date].CurrentMember))
      /
      (KpiValue ("Reseller Revenue"),
       ParallelPeriod
        ([Date].[Calendar Date].[Calendar Year],1,
         [Date].[Calendar Date].CurrentMember)))
       >=.02
  Then 1
   When(
    KpiValue("Reseller Revenue") -
     (KpiValue ( "Reseller Revenue" ),
      ParallelPeriod
       ([Date].[Calendar Date].[Calendar Year],1,
        [Date].[Calendar Date].CurrentMember))
       /
        (KpiValue("Reseller Revenue"),
         ParallelPeriod
          ([Date].[Calendar Date].[Calendar Year],1,
            [Date].[Calendar Date].CurrentMember)))
        <=.02
  Then -1
   Else 0
End
```

This MDX expression provides the basis for evaluating the trend toward achieving the defined goal.

Browsing the Cube by Using the Reseller Revenue KPI

1. On the **Build** menu of SQL Server Data Tools (SSDT), click **Deploy Analysis Service Tutorial**.

2. When deployment has successfully completed, on the toolbar of the **KPIs** tab, click the **Browser View** button, and then click **Reconnect**.

   The status and trend gauges are displayed in the **KPI Browser** pane for reseller sales based on the values for the default member of each dimension, together with the value for the value and the goal. The default member of each dimension is the All member of the All level, because you have not defined any other member of any dimension as the default member.

3. In the filter pane, select **Sales Territory** in the **Dimension** list, select **Sales Territories** in the **Hierarchy** list, select **Equal** in the **Operator** list, select the **North America** check box in the **Filter Expression** list, and then click **OK**.

4. In the next row in the **Filter** pane, select **Date** in the **Dimension** list, select **Calendar Date** in the **Hierarchy** list, select **Equal** in the **Operator** list, select the **Q3 CY 2007** check box in the **Filter Expression** list, and then click **OK**.

5. Click anywhere in the **KPI Browser** pane to update the values for the **Reseller Revenue KPI**.

Notice that the **Value**, **Goal**, and **Status** sections of the KPI reflect the values for the new time period

Defining the Product Gross Profit Margin KPI

1. Click the **Form View** button on the toolbar of the **KPIs** tab, and then click the **New KPI** button.
2. In the **Name** box, type **Product Gross Profit Margin**, and then verify that appears in the **Associated measure group** list.
3. In the **Metadata** tab in the **Calculation Tools** pane, drag the **Total GPM** measure to the **Value Expression** box.
4. In the **Goal Expression** box, type the following expression:

```
Case
    When [Product].[Category].CurrentMember Is
      [Product].[Category].[Accessories]
    Then .40
    When [Product].[Category].CurrentMember
      Is [Product].[Category].[Bikes]
    Then .12
    When [Product].[Category].CurrentMember Is
      [Product].[Category].[Clothing]
    Then .20
    When [Product].[Category].CurrentMember Is
      [Product].[Category].[Components]
    Then .10
    Else .12
End
```

5. In the **Status indicator** list, select **Cylinder**.
6. Type the following MDX expression in the **Status expression** box:

```
Case
    When KpiValue( "Product Gross Profit Margin" ) /
         KpiGoal ( "Product Gross Profit Margin" ) >= .90
    Then 1
    When KpiValue( "Product Gross Profit Margin" ) /
         KpiGoal ( "Product Gross Profit Margin" ) <  .90
         And
         KpiValue( "Product Gross Profit Margin" ) /
         KpiGoal ( "Product Gross Profit Margin" ) >= .80
    Then 0
    Else -1
End
```

   This MDX expression provides the basis for evaluating the progress toward the goal.

7. Verify that **Standard arrow** is selected in the **Trend indicator** list, and then type the following MDX expression in the **Trend expression** box:

```
Case
When IsEmpty
  (ParallelPeriod
   ([Date].[Calendar Date].[Calendar Year],1,
       [Date].[Calendar Date].CurrentMember))
  Then 0
   When VBA!Abs
    (
      KpiValue( "Product Gross Profit Margin" ) -
       (
         KpiValue ( "Product Gross Profit Margin" ),
          ParallelPeriod
          (
            [Date].[ Calendar Date].[ Calendar Year],
            1,
            [Date].[ Calendar Date].CurrentMember
          )
        ) /
        (
          KpiValue ( "Product Gross Profit Margin" ),
          ParallelPeriod
          (
            [Date].[ Calendar Date].[ Calendar Year],
            1,
            [Date].[ Calendar Date].CurrentMember
          )
        )
      ) <=.02
  Then 0
  When KpiValue( "Product Gross Profit Margin" ) -
       (
         KpiValue ( "Product Gross Profit Margin" ),
         ParallelPeriod
         (
           [Date].[ Calendar Date].[ Calendar Year],
           1,
           [Date].[ Calendar Date].CurrentMember
         )
       ) /
       (
         KpiValue ( "Product Gross Profit Margin" ),
         ParallelPeriod
         (
           [Date].[Calendar Date].[Calendar Year],
           1,
           [Date].[Calendar Date].CurrentMember
         )
       )  >.02
  Then 1
  Else -1
End
```

This MDX expression provides the basis for evaluating the trend toward achieving the defined goal.

Browsing the Cube by Using the Total Gross Profit Margin KPI

1. On the **Build** menu, click **Deploy Analysis Service Tutorial**.

2. When deployment has successfully completed, click **Reconnect** on the toolbar of the **KPIs** tab, and then click **Browser View**.

   The **Product Gross Profit Margin** KPI appears and displays the KPI value for **Q3 CY 2007** and the **North America** sales territory.

3. In the **Filter** pane, select **Product** in the **Dimension** list, select **Category** in the **Hierarchy** list, select

Equal in the **Operator** list, and then select **Bikes** in the **Filter Expression** list, and then click **OK**.

The gross profit margin for the sale of Bikes by resellers in North America in Q3 CY 2007 appears.

Next Lesson

Lesson 8: Defining Actions

Lesson 8: Defining Actions

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In this lesson, you will learn to define actions in your Analysis Services project. An action is just a Multidimensional Expressions (MDX) statement that is stored in Analysis Services and which can be incorporated into client applications and started by a user.

> **NOTE**
> Completed projects for all of the lessons in this tutorial are available online. You can jump ahead to any lesson by using the completed project from the previous lesson as a starting point. Click here to download the sample projects that go with this tutorial.

Analysis Services supports the types of actions that are described in the following table.

CommandLine	Executes a command at the command prompt
Dataset	Returns a dataset to a client application.
Drillthrough	Returns a drillthrough statement as an expression, which the client executes to return a rowset
Html	Executes an HTML script in an Internet browser
Proprietary	Performs an operation by using an interface other than those listed in this table.
Report	Submits a parameterized URL-based request to a report server and returns a report to a client application.
Rowset	Returns a rowset to a client application.
Statement	Runs an OLE DB command.
URL	Displays a dynamic Web page in an Internet browser.

Actions let users start an application or perform other steps within the context of a selected item. For more information, see Actions (Analysis Services - Multidimensional Data), Actions in Multidimensional Models

> **NOTE**
> For examples of actions, see the action examples on the Templates tab in the Calculation Tools pane or in the examples in the Adventure Works DW sample data warehouse. For more information about installing this database, see Install Sample Data and Projects for the Analysis Services Multidimensional Modeling Tutorial.

This lesson includes the following task:

Defining and Using a Drillthrough Action

In this task, you define, use, and then modify a drillthrough action through the fact dimension relationship that you defined earlier in this tutorial.

Next Lesson

Lesson 9: Defining Perspectives and Translations

See Also

Analysis Services Tutorial Scenario
Multidimensional Modeling (Adventure Works Tutorial)
Actions (Analysis Services - Multidimensional Data)
Actions in Multidimensional Models

Lesson 8-1 - Defining and Using a Drillthrough Action

11/28/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Dimensioning fact data by a fact dimension without correctly filtering the data that the query returns can cause slow query performance. To avoid this, you can define a drillthrough action that restricts the total number of rows that are returned. This will significantly improve query performance.

In the tasks in this topic, you define a drillthrough action to return order detail information for sales to customers over the Internet.

Defining the Drillthrough Action Properties

1. In Cube Designer for the Analysis Services Tutorial cube, click the **Actions** tab.

   The **Actions** tab includes several panes. On the left side of the tab are the **Action Organizer** pane and the **Calculation Tools** pane. The pane to the right of these two panes is the **Display** pane, which contains the details of the action that is selected in the **Action Organizer** pane.

   The following image shows the **Actions** tab of Cube Designer.

2. On the toolbar of the **Actions** tab, click the **New Drillthrough Action** button.

   A blank action template appears in the display pane.

3. In the **Name** box, change the name of this action to **Internet Sales Details Drillthrough Action**.

4. In the **Measure group members** list, select **Internet Sales**.

5. In the **Drillthrough Columns** box, select **Internet Sales Order Details** in the **Dimensions** list.

6. In the **Return Columns** list, select the **Item Description** and the **Order Number** check boxes, and then click **OK**. The following image shows the Action template as it should appear at this point in this procedure.

7. Expand the **Additional Properties** box, as shown in the following image.

8. In the **Maximum Rows** box, type **10**.

9. In the **Caption** box, type **Drillthrough to Order Details...**.

   These settings limit the number of rows returned and specify the caption that appears in the client application menu. The following image shows these settings in the **AdditionalProperties** box.

Using the Drillthrough Action

1. On the **Build** menu, click **Deploy Analysis Services Tutorial**.

2. When deployment has successfully completed, click the **Browser** tab in Cube Designer for the Analysis Services Tutorial cube, and then click the **Reconnect** button.

3. Start Excel.

4. Add the **Internet Sales-Sales Amount** measure to the Values area.

5. Add the **Customer Geography** user-defined hierarchy from the **Location** folder in the **Customer** dimension to the **Report Filter** area.

6. On the PivotTable, in **Customer Geography**, add a filter that selects a single customer. Expand **All Customers**, expand **Australia**, expand **Queensland**, expand **Brisbane**, expand **4000**, select the check box for **Adam Powell**, and then click **OK**.

   The total sales of products by Adventure Works Cycles to Adam Powell are displayed in the data area.

7. Right-click on the sales amount, point to **Additional Actions**, and then click **Drillthrough to Order Details**.

   The details of the orders that were shipped to Adam Powell are displayed in the **Data Sample Viewer**, as shown in the following image. However, some additional details would also be useful, such as the order date, due date, and ship date. In the next procedure, you will add these additional details.

Data returned for 'Internet Sales Details Drillthrough	Action' ([Customer].[Customer Geography].[Full Na
[$Internet Sales Order Detail].[Item Description]	[$Internet Sales Order Details].[Order Number]
Road-250 Black, 48	SO49206
Road-350W Yellow, 48	SO61522
Short-Sleeve Classic Jersey, XL	SO6152

8. Close Excel/

Modifying the Drillthrough Action

1. Open Dimension Designer for the **Internet Sales Order Details** dimension.

   Notice that only three attributes have been defined for this dimension.

2. In the **Data Source View** pane, right-click an open area, and then click **Show All Tables**.

3. On the **Format** menu, point to **Autolayout** and then click **Diagram**.

4. Locate the **InternetSales (dbo.FactInternetSales)** table by right-clicking in an open area of the **Data Source View** pane. Then click **Find Table,** click **InternetSales,** and click **OK**.

5. Create new attributes based on the following columns:

   - OrderDateKey
   - DueDateKey
   - ShipDateKey

6. Change the **Name** property for the **Order Date Key** attribute to **Order Date** Then, click the browse button for the **Name Column** property, and in the **Name Column** dialog box, select **Date** as the source table and select SimpleDate as the source column. Click **OK**.

7. Change the **Name** property for the **Due Date Key** attribute to **Due Date**, and then, by using the same method as the **Order Date Key** attribute, change the **Name Column** property for this attribute to **Date.SimpleDate (WChar)**.

8. Change the **Name** property for the **Ship Date Key** attribute to **Ship Date**, and then change the **Name Column** property for this attribute to **Date.SimpleDate (WChar)**.

9. Switch to the **Actions** tab of Cube Designer for the Analysis Services Tutorial cube.

10. In the **Drillthrough Columns** box, select the check boxes to add the following columns to the **Return Columns** list, and then click **OK**:

    - Order Date
    - Due Date
    - Ship Date

    The following image shows these columns selected.

Reviewing the Modified Drillthrough Action

1. On the **Build** menu, click **Deploy Analysis Services Tutorial**.
2. When deployment has successfully completed, switch to the **Browser** tab in Cube Designer for the Analysis Services Tutorial cube, and then click the **Reconnect** button.
3. Start Excel.
4. Recreate the PivotTable using **Internet Sales-Sales Amount** in the Values area and **Customer Geography** in the Report Filter.

   Add a filter that selects from **All Customers**, **Australia**, **Queensland**, **Brisbane**, **4000**, **Adam Powell**.
5. Click the **Internet Sales-Sales Amount** data cell, point to **Additional Actions**, and then click **Drillthrough to Order Details**.

   The details of these orders shipped to Adam Powell are displayed in a temporary worksheet. This includes item description, order number, order date, due date, and ship date information, as shown in the following image.

[Item Description]	[Order Number]	[Order Date]	[Due Date]	[Ship Date]
Road-250 Black, 48	SO49206	20070204	20070216	20070211
Road-350-W Yellow, 48	SO61522	20080105	20080117	20080112
Short-Sleeve Clasic Jersey, XL	SO61522	20080105	20080117	20080112

Next Lesson

Lesson 9: Defining Perspectives and Translations

See Also

Actions (Analysis Services - Multidimensional Data)
Actions in Multidimensional Models
Dimension Relationships
Defining a Fact Relationship
Define a Fact Relationship and Fact Relationship Properties

Lesson 9: Defining Perspectives and Translations

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In this lesson, you learn to define perspectives and translations. You can define perspectives to reduce the apparent complexity of a cube, and define translations that let users view the cube metadata in the language of their choice.

> **NOTE**
> Completed projects for all of the lessons in this tutorial are available online. You can jump ahead to any lesson by using the completed project from the previous lesson as a starting point. Click here to download the sample projects that go with this tutorial.

This lesson contains the following tasks:

Defining and Browsing Perspectives
In this task, you define and browse perspectives to simplify the view of the cube for specific users or uses.

Defining and Browsing Translations
In this task, you define and browse translations of specific metadata to certain languages.

Next Lesson

Lesson 10: Defining Administrative Roles

See Also

Analysis Services Tutorial Scenario
Multidimensional Modeling (Adventure Works Tutorial)
Perspectives
Perspectives in Multidimensional Models
Dimension Translations
Cube Translations
Translation support in Analysis Services

Lesson 9-1 - Defining and Browsing Perspectives

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A perspective can simplify the view of a cube for specific purposes. By default, users can see all of the elements in a cube to which they have permissions. What users are viewing when they view an entire Analysis Services cube is the default perspective for the cube. A view of the whole cube can be very complex for users to navigate, especially for users who only need to interact with a small part of the cube to satisfy their business intelligence and reporting requirements.

To reduce the apparent complexity of a cube, you can create viewable subsets of the cube, called *perspectives*, which show users only a part of the measure groups, measures, dimensions, attributes, hierarchies, Key Performance Indicators (KPIs), actions, and calculated members in the cube. This can be particularly useful for working with client applications that were written for a previous release of Analysis Services. These clients have no concept of display folders or perspectives, for example, but a perspective appears to older clients as if it were a cube. For more information, see Perspectives, and Perspectives in Multidimensional Models.

> **NOTE**
>
> A perspective is not a security mechanism, but instead is a tool for providing a better user experience. All security for a perspective is inherited from the underlying cube.

In the tasks in this topic, you will define several different perspectives and then browse the cube through each of these new perspectives.

Defining an Internet Sales Perspective

1. Open Cube Designer for the Analysis Services Tutorial cube, and then click the **Perspectives** tab.

   All the objects and their object types appear in the **Perspectives** pane, as shown in the following image.

2. On the toolbar of the **Perspectives** tab, click the **New Perspective** button.

   A new perspective appears in the **Perspective Name** column with a default name of **Perspective**, as shown in the following image. Notice that the check box for every object is selected; until you clear the check box for an object, this perspective is identical to the default perspective of this cube.

3. Change the perspective name to **Internet Sales**.

4. On the next row, set the DefaultMeasure to **Internet Sales-Sales Amount**.

   When users browse the cube by using this perspective, this will be the measure that the users see unless they specify some other measure.

   > **NOTE**
   >
   > You can also set the default measure for the whole Analysis Services Tutorial cube in the Properties window on the **Cube Structure** tab for the cube.

5. Clear the check box for the following objects:

   - **Reseller Sales** measure group
   - **Sales Quotas** measure group
   - **Sales Quotas 1** measure group
   - **Reseller** cube dimension
   - **Reseller Geography** cube dimension
   - **Sales Territory** cube dimension
   - **Employee** cube dimension
   - **Promotion** cube dimension
   - **Reseller Revenue** KPI
   - **Large Resellers** named set

- **Total Sales Amount** calculated member
- **Total Product Cost** calculated member
- **Reseller GPM** calculated member
- **Total GPM** calculated member
- **Reseller Sales Ratio to All Products** calculated member
- **Total Sales Ratio to All Products** calculated member

These objects do not relate to Internet sales.

> **NOTE**
> Within each dimension, you can also individually select the user-defined hierarchies and attributes that you want to appear in a perspective.

Defining a Reseller Sales Perspective

1. On the toolbar of the **Perspectives** tab, click the **New Perspective** button.
2. Change the name of the new perspective to **Reseller Sales**.
3. Set **Reseller Sales-Sales Amount** as the default measure.

   When users browse the cube by using this perspective, this measure will be the measure that the users will see unless they specify some other measure.
4. Clear the check box for the following objects:
   - **Internet Sales** measure group
   - **Internet Sales Reason** measure group
   - **Customer** cube dimension
   - **Internet Sales Order Details** cube dimension
   - **Sales Reason** cube dimension
   - **Internet Sales Details Drillthrough Action** drillthrough action
   - **Total Sales Amount** calculated member
   - **Total Product Cost** calculated member
   - **Internet GPM** calculated member
   - **Total GPM** calculated member
   - **Internet Sales Ratio to All Products** calculated member
   - **Total Sales Ratio to All Products** calculated member

   These objects do not relate to resellers sales.

Defining a Sales Summary Perspective

1. On the toolbar of the **Perspectives** tab, click the **New Perspective** button.

2. Change the name of the new perspective to **Sales Summary**.

> **NOTE**
>
> You cannot specify a calculated measure as the default measure.

3. Clear the check box for the following objects:
   - **Internet Sales** measure group
   - **Reseller Sales** measure group
   - **Internet Sales Reason** measure group
   - **Sales Quotas** measure group
   - **Sales Quotas1** measure group
   - **Internet Sales Order Details** cube dimension
   - **Sales Reason** cube dimension
   - **Internet Sales Details Drillthrough Action** drillthrough action
4. Select the check box for the following objects:
   - **Internet Sales Count** measure
   - **Reseller Sales Count** measure

Browsing the Cube Through Each Perspective

1. On the **Build** menu, click **Deploy Analysis Services Tutorial**.
2. When deployment has successfully completed, switch to the **Browser** tab, and then click the **Reconnect** button.
3. Start Excel.
4. Analyze in Excel prompts you to choose which perspective to use when browsing the model in Excel, as shown in the following image.

5. Alternatively, you can start Excel from the Windows Start menu, define a connection to the Analysis Services Tutorial database on localhost, and choose a perspective in the Data Connection wizard, as shown in the following image.

6. Select **Internet Sales** in the **Perspective** list and then review the measures and dimensions in the metadata pane.

   Notice that only those objects that are specified for the Internet Sales perspective appear.

7. In the metadata pane, expand **Measures**.

   Notice that only the **Internet Sales** measure group appears, together with the **Internet GPM** and **Internet Sales Ratio to All Products** calculated members.

8. In the model, select Excel again. Select **Sales Summary**.

   Notice that in each of these measure groups, only a single measure appears, as shown in the following image.

Next Task in Lesson

Defining and Browsing Translations

See Also

Perspectives
Perspectives in Multidimensional Models

Lesson 9-2 - Defining and Browsing Translations

11/28/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A translation is a representation of the names of Analysis Services objects in a specific language. Objects include measure groups, measures, dimensions, attributes, hierarchies, KPIs, actions, and calculated members. Translations provide server support for client applications that can support multiple languages. By using such a client, the client passes the locale identifier (LCID) to the instance of Analysis Services, which uses the LCID to determine which set of translations to use when it provides metadata for Analysis Services objects. If an Analysis Services object does not contain a translation for that language, or does not contain a translation for a specified object, the default language is used in returning the object metadata back to the client. For example, if a business user in France accesses a cube from a workstation that has a French locale setting, the business user will see the member captions and member property values in French if a French translation exists. However, if a business user in Germany accesses the same cube from a workstation that has a German locale setting, the business user will see the captions names and member property values in German. For more information, see Dimension Translations, Cube Translations, Translation support in Analysis Services.

In the tasks in this topic, you define metadata translations for a limited set of dimension objects in the Date dimension and cube objects in the Analysis Services Tutorial cube. You will then browse these dimension and cube objects to examine the metadata translations.

Specifying Translations for the Date Dimension Metadata

1. Open Dimension Designer for the **Date** dimension, and then click the **Translations** tab.

   The metadata in the default language for each dimension object appears. The default language in the Analysis Services Tutorial cube is English.

2. On the toolbar of the **Translations** tab, click the **New Translation** button.

   A list of languages appears in the **Select Language** dialog box.

3. Click **Spanish (Spain)**, and then click **OK**.

   A new column appears in which you will define the Spanish translations for the metadata objects you want to translate. In this tutorial, we will only translate a few objects just to illustrate the process.

4. On the toolbar of the **Translations** tab, click the **New Translation** button, click **French (France)** in the **Select Language** dialog box, and then click **OK**.

   Another language column appears in which you will define French translations.

5. In the row for the **Caption** object for the **Date** dimension, type **Fecha** in the **Spanish (Spain)** translation column and **Temps** in the **French (France)** translation column.

6. In the row for the **Caption** object for the **Month Name** attribute, type **Mes del Año** in the **Spanish (Spain)** translation column and **Mois d'Année** in the **French (France)** translation column.

   Notice that when you enter these translations, an ellipsis (**...**) appears. Clicking this ellipsis will enable you to specify a column in the underlying table that provides translations for each member of the attribute hierarchy.

7. Click the ellipsis (**...**) for the **Spanish (Spain)** translation for the **Month Name** attribute.

The **Attribute Data Translation** dialog box appears.

8. In the **Translation columns** list, select **SpanishMonthName**, as shown in the following image.

9. Click **OK**, and then click the ellipsis (**...**) for the **French (France)** translation for the **Month Name** attribute.

10. In the **Translation columns** list, select **FrenchMonthName**, and then click **OK**.

    The steps in this procedure illustrate the process of defining metadata translations for dimension objects and members.

Specifying Translations for the Analysis Services Tutorial Cube Metadata

1. Switch to Cube Designer for the Analysis Services Tutorial cube, and then switch to the **Translations** tab.

   The metadata in the default language for each cube object appears, as shown in the following image. The default language in the Analysis Services Tutorial cube is English.

2. On the toolbar of the **Translations** tab, click the **New Translation** button.

   A list of languages appears in the **Select Language** dialog box.

3. Select **Spanish (Spain)**, and then click **OK**.

   A new column appears in which you will define the Spanish translations for the metadata objects you want to translate. In this tutorial, we will only translate a few objects just to illustrate the process.

4. On the toolbar of the **Translations** tab, click the **New Translation** button, select **French (France)** in the **Select Language** dialog box, and then click **OK**.

   Another language column appears in which you will define French translations.

5. In the row for the **Caption** object for the **Date** dimension, type **Fecha** in the **Spanish (Spain)** translation column and **Temps** in the **French (France)** translation column.

6. In the row for the **Caption** object for the **Internet Sales** measure group, type **Ventas del Internet** in the **Spanish (Spain)** translation column and **Ventes D'Internet** in the **French (France)** translation column.

7. In the row for the **Caption** object for the Internet Sales-Sales Amount measure, type **Cantidad de las Ventas del Internet** in the **Spanish (Spain)** translation column and **Quantité de Ventes d'Internet** in the **French (France)** translation column.

   The steps in this procedure illustrate the process of defining metadata translations for cube objects.

Browsing the Cube By Using Translations

1. On the **Build** menu, click **Deploy Analysis Services Tutorial**.

2. When deployment has successfully completed, switch to the **Browser** tab, and then click **Reconnect**.

3. Remove all hierarchies and measures from the **Data** pane and select Analysis Services Tutorial in the **Perspectives** list.

4. In the metadata pane, expand **Measures** and then expand **Internet Sales**.

   Notice that the **Internet Sales-Sales Amount** measure appears in English in this measure group.

5. On the toolbar, select **Spanish (Spain)** in the **Language** list.

   Notice that the items in the metadata pane are repopulated. After the items in the metadata pane are repopulated, notice that the Internet Sales-Sales Amount measure no longer appears in the Internet Sales display folder. Instead, it appears in Spanish in a new display folder named **Ventas del Internet**, as shown in the following image.

6. In the metadata pane, right-click **Cantidad de las Ventas del Internet** and then select **Add to Query**.

7. In the metadata pane, expand **Fecha**, expand **Fecha.Calendar Date**, right-click **Fecha.Calendar Date**, and then select **Add to Filter**.

8. In the **Filter** pane, select **CY 2007** as the filter expression.

9. In the metadata pane, right-click **Mes del Ano** and select **Add to Query**.

   Notice that the month names appear in Spanish, as shown in the following image.

Dimension	Hierarchy	Operator	Filter Expression
Fecha	Fecha.Calendar Date	Equal	{ CY 2007 }
<Select dimension>			

Mes del Año	Cantidad de la...
Enero	438865.1718
Febrero	489090.3356
Marzo	485574.7923
Abril	506399.2654
Mayo	562772.5645
Junio	554799.2281
Julio	886668.84
Agosto	847413.5100...
Septiembre	1010258.13
Octubre	1080449.58
Noviembre	1196981.11
Diciembrfe	1731787.77

10. On the toolbar, select **French (France)** in the **Language** list.

    Notice that the month names now appear in French and that the measure name now also appears in French.

Next Lesson

Lesson 10: Defining Administrative Roles

See Also

Dimension Translations
Cube Translations
Translation support in Analysis Services

Lesson 10: Defining Administrative Roles

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In this lesson, you learn to define security roles for administrative tasks.

> **NOTE**
>
> Completed projects for all of the lessons in this tutorial are available online. You can jump ahead to any lesson by using the completed project from the previous lesson as a starting point. Click here to download the sample projects that go with this tutorial.

This lesson contains the following task:

Granting Process Database Permissions
In this task, you define a security role that has permissions to process the Analysis Services database, and then you test this security role.

See Also

Analysis Services Tutorial Scenario
Multidimensional Modeling (Adventure Works Tutorial)

Lesson 10 - Granting Process Database Permissions

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you install an instance of Analysis Services, all members of the Analysis Services server administrator role in that instance have server-wide permissions to perform any task within the instance of Analysis Services. By default, no other users have any permission to administer or view any objects in the instance of Analysis Services.

A member of the server administrator role can grant users administrative access on a server-wide basis by making them members of the role. A member of the server administrator role can also grant users access on a more limited basis by granting them limited or complete administrative or access permissions at the database level. Limited administrative permissions include process or read definition permissions at the database, cube, or dimension level.

In the tasks in this topic, you will define a Process Database Objects security role that grants members of the role permission to process all database objects, but no permission to view data within the database.

Defining a Process Database Objects Security Role

1. In Solution Explorer, right-click **Roles** and then click **New Role** to open the Role Designer.
2. Click the **Process database** check box.
3. In the Properties window, change the **Name** property for this new role to **Process Database Objects Role**.

4. Switch to the **Membership** tab of Role Designer and click **Add**.
5. Enter the accounts of the Windows domain users or groups who will be members of this role. Click **Check Names** to verify the account information, and then click **OK**.
6. Switch to the **Cubes** tab of Role Designer.

   Notice that members of this role have permissions to process this database, but have no permission to

access the data in the Analysis Services Tutorial cube and have no local cube/drillthrough access, as shown in the following image.

7. Switch to the **Dimensions** tab of Role Designer.

   Notice that members of this role have permissions to process all dimension objects in this database, and, by default, have read permissions to access each dimension object in the Analysis Services Tutorial database.

8. On the **Build** menu, click **Deploy Analysis Services Tutorial**.

   You have now successfully defined and deployed the Process Database Objects security role. After a cube is deployed to the production environment, the administrators of the deployed cube can add users to this role as required to delegate processing responsibilities to specific users.

> **NOTE**
>
> A completed project for Lesson 10 is available by downloading and installing the samples. For more information, see Install Sample Data and Projects for the Analysis Services Multidimensional Modeling Tutorial.

See Also

Roles and Permissions (Analysis Services)

Data Mining Tutorials (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Microsoft SQL Server Analysis Services makes it easy to create data mining solutions using wizards and integrated visualizations. Particularly if you are new to machine learning, the tools in Analysis Services are an easy way to design, train, and explore data mining models. The data in your models can be stored in a cube, relational database, or any other source support by Analysis Services. After creating a model, you can put it into production by accessing the model to create predictions using prediction multiple clients, including Integration Services and ASP.NET.

> **NOTE**
> The tutorials described here have not been updated for SQL Server 2017. You can use the tutorials created for SQL Server 2014. Functionally, there are no changes in Data Mining features for SQL Server 2017. The steps should be identical.

Tutorials

Basic Data Mining Tutorial (SQL Server 2014) - This tutorial walks you through a targeted mailing scenario. It demonstrates how to use the data mining algorithms, mining model viewers, and data mining tools that are included in Analysis Services. You will build three data mining models to answer practical business questions while learning data mining concepts and tools.

Intermediate Data Mining Tutorial (SQL Server 2014) - This tutorial contains a collection of lessons that introduce more advanced data mining concepts and techniques such as, forecasting, market basket analysis, neural networks and logistic regression, and sequence clustering.

DMX Tutorials (SQL Server 2014) - The Data Mining Extensions (DMX) query language has syntax like that of SQL but can be used to create, query, and manage predictive models stored in Analysis Services. These tutorials demonstrate how to create a new mining structure and mining models by using the DMX language, and how to create DMX prediction queries for use in applications.

See Also

Data Mining Solutions
Microsoft SQL Server Data Mining resources
Creating and Querying Data Mining Models with DMX: Tutorials (Analysis Services - Data Mining)

Install SQL Server Analysis Services

6/1/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

SQL Server Analysis Services is an analytical database server that hosts Tabular models, multidimensional cubes, and data mining models that you can access from reports, spreadsheets, and dashboards.

Analysis Services is multi-instance, which means that you can install more than one copy on a single computer, or run new and old versions side-by-side. Any instance you install runs in one of three modes, as determined during setup: Multidimensional and Data Mining, Tabular, or SharePoint. If you want to use multiple modes, you'll need a separate instance for each one.

After you install the server in a particular mode, you can use it host solutions that conform to that mode. For example, a tabular mode server is required if you want tabular model data access over the network.

Get tools and designers

SQL Server Setup no longer installs the model designers or management tools used for solution design or server administration. In this release, tools have a separate installation, which you can get from the following links:

- Download SQL Server Management Studio (SSMS)
- Download SQL Server Data Tools (SSDT)

You'll need both SSMS and SSDT to work with Analysis Services instances and data. Tools can be installed anywhere, but be sure to configure ports on the server before attempting a connection. See Configure the Windows Firewall to Allow Analysis Services Access for details.

Install using a wizard

The following list shows you which pages in the SQL Server Installation wizard are used to install Analysis Services.

1. Select **Analysis Services** from the Feature Tree in Setup.

2. On the Analysis Services Configuration page, select a mode. Tabular mode is the default..

Tabular mode uses the xVelocity in-memory analytics engine (VertiPaq), which is the default storage for tabular models. After you deploy tabular models to the server, you can selectively configure tabular solutions to use DirectQuery disk storage as an alternative to memory-bound storage.

Multidimensional and Data Mining mode use MOLAP as the default storage for models deployed to Analysis Services. After deploying to the server, you can configure a solution to use ROLAP if you want to run queries directly against the relational database rather than storing query data in an Analysis Services multidimensional database .

Memory management and IO settings can be adjusted to get better performance when using non-default storage modes. See Server Properties in Analysis Services for more information.

Command Line Setup

SQL Server Setup includes a parameter (**ASSERVERMODE**) that specifies the server mode. The following

example illustrates a command line setup that installs Analysis Services in Tabular server mode.

```
Setup.exe /q /IAcceptSQLServerLicenseTerms /ACTION=install /FEATURES=AS /ASSERVERMODE=TABULAR
/INSTANCENAME=ASTabular /INDICATEPROGRESS /ASSVCACCOUNT=<DomainName\UserName> /ASSVCPASSWORD=<StrongPassword>
/ASSYSADMINACCOUNTS=<DomainName\UserName>
```

INSTANCENAME must be less than 17 characters.

All placeholder account values must be replaced with valid accounts and password.

ASSERVERMODE is case-sensitive. All values must be expressed in upper case. The following table describes the valid values for **ASSERVERMODE**.

VALUE	DESCRIPTION
TABULAR	This is the default value. If you do not set **ASSERVERMODE**, the server is installed in Tabular mode.
MULTIDIMENSIONAL	This value is optional.
POWERPIVOT	This value is optional. In practice, if you set the **ROLE** parameter, the server mode is automatically set to 1, making **ASSERVERMODE** optional for a Power Pivot for SharePoint installation. For more information, see Install Power Pivot from the Command Prompt.

See Also

Determine the Server Mode of an Analysis Services Instance
Tabular Modeling

Install Analysis Services in Power Pivot Mode

11/28/2018 • 11 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The procedures in this topic guide you through a single server installation of a Analysis Services server in Power Pivot mode for a SharePoint deployment. The steps include running the SQL Server installation wizard as well as configuration tasks that use SharePoint Central Administration.

Background

Power Pivot for SharePoint is a collection of middle-tier and backend services that provide Power Pivot data access in a SharePoint 2016, or SharePoint 2013, farm.

- **Backend services:** If you use Power Pivot for Excel to create workbooks that contain analytical data, you must have Power Pivot for SharePoint to access that data in a server environment. You can run SQL Server Setup on a computer that has SharePoint Server installed, or on a different computer that has no SharePoint software. Analysis Services does not have any dependencies on SharePoint.

  Note: This topic describes the installation of the Analysis Services server and the backend services.

- **Middle-tier:** Enhancements to the Power Pivot experiences in SharePoint including Power Pivot Gallery, Schedule data refresh, Management dashboard, and data providers. For more information on installing and configuring the middle-tier, see the following:
  - Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2016)
  - Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2013)
  - Configure Power Pivot and Deploy Solutions (SharePoint 2016)
  - Configure Power Pivot and Deploy Solutions (SharePoint 2013)

Prerequisites

1. You must be a local administrator to run SQL Server Setup.
2. SharePoint Server enterprise edition is required for Power Pivot for SharePoint. You can also use the evaluation enterprise edition.
3. The computer must be joined to a domain in the same Active Directory forest as the Office Online Server (SharePoint 2016) or Excel Services (SharePoint 2013).
4. The Power Pivot instance name must be available. You cannot have an existing Power Pivot-named instance on the computer on which you are installing Analysis Services in Power Pivot mode.

   Note: The instance name must be POWERPIVOT.

5. Review Hardware and Software Requirements for Analysis Services Server in SharePoint Mode.
6. Review the release notes at SQL Server 2016 Release Notes.

SQL Server Edition Requirements

Business intelligence features are not all available in all editions of SQL Server 2017. For details, see Analysis Services Features Supported by the Editions of SQL Server 2016 and Editions and Components of SQL Server

2016.

Step 1: Install Power Pivot for SharePoint

In this step, you run SQL Server Setup to install an Analysis Services server in Power Pivot mode. In a subsequent step, you configure Excel Services to use this server for workbook data models.

1. Run the SQL Server Installation Wizard (Setup.exe).
2. select **Installation** in the left navigation.
3. select **New SQL Server stand-alone installation or add features to an existing installation**.
4. If you see the **Product Key** page, specify the evaluation edition or enter a product key for a licensed copy of the enterprise edition. select **Next**. For more information on editions, see Editions and Components of SQL Server 2016.
5. Review and accept the Microsoft Software License Terms of agreement, and then select **Next**.
6. If you see the **Global Rules** page, review any rules information the setup wizard displays.
7. On the **Microsoft Update** page, it is recommended you use Microsoft Update to check for updates, then select **Next**.
8. The **Install Setup Files** page runs for several minutes. Review any rule warnings or failed rules, and then select **Next**.
9. If you see another **Setup Support Rules**, review any warnings and select **Next**.

   Note: Because Windows Firewall is enabled, you see a warning to open ports to enable remote access.
10. On the **Setup Role** page, select **SQL Server Feature Installation**.

    Select **Next**.
11. On the Feature Selection page, select **Analysis Services**. This option allows you to install any of the three Analysis Services modes. You will select the mode in a later step. Select **Next**.
12. On the **Instance Configuration** page, select **Named Instance** and type **POWERPIVOT** for the instance name Click **Next**.

13. On the **Server Configuration** page, configure all of the services for Automatic **Startup Type**. Specify the desired domain account and password for **SQL Server Analysis Services**, **(1)** in the following diagram.
    - For Analysis Services, you can use a **domain user** account or **NetworkService** account. Do not use LocalSystem or LocalService accounts.
    - If you added the SQL Server Database Engine and SQL Server Agent, you can configure the

services to run under domain user accounts or under the default virtual account.

- Never provision service accounts with your own domain user account. Doing so grants the server the same permissions that you have to the resources in your network. If a malicious user compromises the server, that user is logged in under your domain credentials. The user has the permissions to download or use the same data and applications that you do.

Select **Next**.

14. If you are installing the Database Engine, the **Database Engine Configuration** page appears. In Database Engine Configuration, select **Add Current User** to grant your user account administrator permissions on the Database Engine instance.

    Select **Next**.

15. On the **Analysis Services Configuration** page, select **PowerPivot Mode** under **Server Mode**

16. On the **Analysis Services Configuration** page, select **Add Current User** to grant your user account administrative permissions. You will need administrative permission to configure the server after Setup is finished.

    - In the same page, add the Windows user account of any person who also requires administrative permissions. For example, any user who wants to connect to the Analysis Services service instance in SQL Server Management Studio to troubleshoot database connection problems must have system administrator permissions. Add the user account of any person who might need to troubleshoot or administer the server now.

    - **NOTE**

      All service applications that require access to the Analysis Services server instance need to have Analysis Services Administrative permissions. For example, add the service accounts for Excel Services, Power View, and Performance Point Services. Also, add the SharePoint farm account, which is used as the identity of the web application that hosts Central Administration.

Select **Next**.

17. On the **Error Reporting** page, select **Next**.
18. On the **Ready to Install** page, select **Install**.
19. If you see the dialog **Computer Restart Required**, select **OK**.
20. When the installation is complete, select **Close**.
21. Restart the computer.
22. If you have a firewall in your environment, review the SQL Server Books Online topic, Configure the Windows Firewall to Allow Analysis Services Access.

Verify the SQL Server Installation

Verify that the Analysis Services Service is running.

1. In Microsoft Windows click **Start**, select **All Programs**, and select the **Microsoft SQL Server 2016** group.
2. Select **SQL Server Management Studio**.
3. Connect to the Analysis Services instance, for example **[your server name]\POWERPIVOT**. If you can connect to the instance, you have verified the Service is running.

Step 2: Configure Basic Analysis Services SharePoint Integration

The following steps describe configuration changes needed so you can interact with Excel advanced data models inside a SharePoint document library. Complete these steps after you install SharePoint and SQL Server Analysis Services.

SharePoint 2016

Excel Services was removed from SharePoint 2016 , and instead uses Office Online Server for hosting Excel.

Grant Office Online Server machine account Administration Rights on Analysis Services

You do not need to complete this section if during the Analysis Services installation; you added the Office Online Server machine account as an Analysis Services administrator.

1. On the Analysis Services server, start SQL Server Management Studio and connect to the Analysis Services instance, for example `[MyServer]\POWERPIVOT`.
2. In Object Explorer, right-click the instance name and select **Properties**.

3. In the left pane, select **Security**. Add the machine account that the Office Online Server is installed on.

Register Analysis Services server with Office Online Server

You will want to perform these steps on the Office Online Server.

- Open a PowerShell command window as an administrator.
- Load the `OfficeWebApps` PowerShell module.

  ```
  Import-Module OfficeWebApps
  ```

- Add the Analysis Services server, for example `[MyServer]\POWERPIVOT`.

  ```
  New-OfficeWebAppsExcelBIServer -ServerId [MyServer]\POWERPIVOT]
  ```

SharePoint 2013

Grant Excel Services Server Administration Rights on Analysis Services

You do not need to complete this section if during the Analysis Services installation; you added the Excel Services Application service account as an Analysis Services administrator.

1. On the Analysis Services server, start SQL Server Management Studio and connect to the Analysis Services instance, for example `[MyServer]\POWERPIVOT`.
2. In Object Explorer, right-click the instance name and select **Properties**.

3. In the left pane, select **Security**. Add the domain login you configured for the Excel Services Application in step 1.

Configure Excel Services for Analysis Services integration

1. In SharePoint Central Administration, in the Application Management group, click **Manage Service Applications**.
2. Click the name of your service application, the default is **Excel Services Application**.
3. On the **Manage Excel Services Application page**, click **Data Model Settings**.
4. Click **Add Server**.
5. In **Server Name**, type the Analysis Services server name and the Power Pivot instance name. For example `MyServer\POWERPIVOT`. The Power Pivot instance name is required.

   Type a description.
6. Click **Ok**.
7. The changes will take effect in a few minutes or you can **Stop** and **Start** the service **Excel Calculation Services**. To

   Another option is to open a command prompt with administrative privileges, and type `iisreset /noforce`.

   You can verify the server is recognized by Excel Services by reviewing entries in the ULS log. You will see entries similar to the following:

```
Excel Services Application            Data Model        27           Medium                Check
Administrator Access ([ServerName]\POWERPIVOT): Pass.        f127bd9b-bae3-e0e0-9b48-3f7b5ad1eae6
Excel Services Application            Data Model        27           Medium                Check Server
Version ([ServerName]\POWERPIVOT): Pass (11.0.2809.24 >= 11.0.2800.0).         f127bd9b-bae3-e0e0-9b48-
3f7b5ad1eae6
Excel Services Application            Data Model        27           Medium                Check
Deployment Mode ([ServerName]\POWERPIVOT): Pass.            f127bd9b-bae3-e0e0-9b48-3f7b5ad1eae6
```

Step 3: Verify the Integration

The following steps walk you through creating and uploading a new workbook to verify the Analysis Services integration. You will need a SQL Server database to complete the steps.

1. **Note:** If you already have an advanced workbook with slicers or filters, you can upload it to your SharePoint document library and verify you are able to interact with the slicers and filters from the document library view.
2. Start a new workbook in Excel.
3. On the Data tab, select **From Other Sources** on the ribbon in the **Get External Data**.
4. Select **From SQL Server**.
5. In the **Data Connection Wizard**, enter the name of the SQL Server instance that has the database you want to use.
6. Under Log on credentials, verify that **Use Windows Authentication** is selected, and then select **Next**.
7. Select the database you want to use.
8. Verify that the **Connect to specific table** checkbox is selected.
9. Select the **Enable selection of multiple tables and add tables to the Excel Data Model** checkbox.
10. Select the tables you want to import.
11. Select the checkbox **Import relationships between selected tables**, and then select **Next**. Importing multiple tables from a relational database lets you work with tables that are already related. You save steps because you don't have to build the relationships manually.
12. In the **Save Data Connection File and Finish** page of the wizard,, type a dame for your connection and select **Finish**.
13. The **Import Data** dialog box will appear. Choose **PivotTable Report**, and then select **Ok**.
14. A PivotTable Field List appears in the workbook.
    On the field list, select the **All** tab
15. Add fields to the Row, Columns, and Value areas in the field list.
16. Add a slicer or a filter to the PivotTable. **Do not skip this step**. A slicer or filter is the element that will help you verify your Analysis Services installation.
17. Save the workbook to a document library in your SharePoint farm. You can also save the workbook to a file share and then upload it to the SharePoint document library.
18. Select the name of your workbook to view it in Excel Online and click the slicer or change the filter that you previously added. If a data update occurs, you know that Analysis Services is installed and available to Excel. If you open the workbook in Excel you will be using a cached copy and not using the Analysis

Services server.

Configure the Windows Firewall to Allow Analysis Services Access

Use the information in the topic Configure the Windows Firewall to Allow Analysis Services Access to determine whether you need to unblock ports in a firewall to allow access to Analysis Services or Power Pivot for SharePoint. You can follow the steps provided in the topic to configure both port and firewall settings. In practice, you should perform these steps together to allow access to your Analysis Services server.

Upgrade Workbooks and Scheduled Data Refresh

The steps required to upgrade workbooks created in previous versions of Power Pivot depend on what version of Power Pivot created the workbook. For more information, see Upgrade Workbooks and Scheduled Data Refresh (SharePoint 2013).

Beyond the Single-Server Installation - Power Pivot for Microsoft SharePoint

Web front-end (WFE) or **Middle-tier:**: To use an Analysis Services server in SharePoint mode in a larger SharePoint farm and to install additional Power Pivot features into the farm, run the installer package **spPowerPivot16.msi (SharePoint 2016), or spPowerPivot.msi (SharePoint 2013),** on each of the SharePoint servers. The spPowerPivot16.msi, or spPowerPivot.msi, installs required data providers and the Power Pivot for SharePoint 2016, or 2013, Configuration tool.

For more information on installing and configuring the middle-tier, see the following:

- Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2013)
- Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2013)
- To download the .msi, see Microsoft SQL Server 2016 Power Pivot for Microsoft SharePoint 2016
- Configure Power Pivot and Deploy Solutions (SharePoint 2013)

Redundancy and server load: Installing a second, or more Analysis Services servers in Power Pivot mode will provide redundancy of the Analysis Services server functionality. Additional servers will also spread the load across servers. For more information, see the following:

- Configure Analysis Services for processing data models in Excel Services (SharePoint 2013).
- Manage Excel Services data model settings (SharePoint 2013).

Submit feedback and contact information through SQL Server Feedback.

See Also

Migrate Power Pivot to SharePoint 2013
Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2013)
Upgrade Workbooks and Scheduled Data Refresh (SharePoint 2013)

Verify a Power Pivot for SharePoint Installation

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A Power Pivot for SharePoint instance that you install in a SharePoint farm is administered through SharePoint Central Administration. At a minimum, you can check pages in Central Administration and on SharePoint sites to verify that Power Pivot server components and features are available. However, to fully verify an installation, you must have a Power Pivot workbook that you can publish to SharePoint and access from a library. For testing purposes, you can publish a sample workbook that already contains Power Pivot data and use it to confirm that SharePoint integration is correctly configured.

Verify Central Administration Integration

To verify Power Pivot integration with Central Administration, do the following:

1. On the Start menu, click **All Programs**, open Microsoft SharePoint 2016 Products, or Microsoft SharePoint 2013 Products, and click **SharePoint 2016 Central Administration, or SharePoint 2013 Central Administration**.

2. Enter your user name and password, and then click **OK**.

   Optionally, you can modify browser settings to avoid having to enter a user name and password each time you open Central Administration. To add Central Administration as a trusted site, do the following.

   a. In Internet Explorer, on the Tools menu, click **Internet options**.

   b. On the Security tab, in the **Select a zone to view or change security settings** section, click Trusted Sites, and then click Sites.

   c. Clear the **Require server verification (https:) for all sites in this zone** checkbox.

   d. In **Add this Web site to the zone**, type the URL to your site, and then click **Add**.

   e. Click **Close**, and then click **OK**.

   > **NOTE**
   >
   > SharePoint installation documentation includes additional instructions for working around proxy server errors and for disabling Internet Explorer Enhanced Security Configuration so that you can download and install updates. For more information, see the **Perform additional tasks** section in Deploy a single server with SQL Server on the Microsoft web site.

3. In Central Administration, in System Settings, click **Manage farm features**.

4. Verify that **Power Pivot Integration Feature** is **Active**.

5. In Central Administration, in System Settings, click **Manage services on server**.

6. Verify that **SQL Server Power Pivot System Service** are started.

   In a multiple server SharePoint Farm, you may need to change the server you are viewing to validate that all of the servers you deployed Power Pivot to are running.

7. In Central Administration, in Application Management, click **Manage service applications**.

8. Click **Default Power Pivot Service Application** to open Power Pivot Management Dashboard for this application. On first use, the dashboard takes several minutes to load.

   Alternatively, click the empty space next to **Default Power Pivot Service Application** to select the row, and click **Properties** to view the configuration settings for this service application. You can modify both configuration settings and application properties to change your server configuration. For more information about these settings, see Create and Configure a Power Pivot Service Application in Central Administration.

Verify Integration at the Site Level

To verify Power Pivot integration with a SharePoint site, do the following:

1. In a browser, open the Web application you created. If you used default values, you can specify http://<your computer name> in the URL address.
2. Verify that Power Pivot data access and processing features are available in the application. You can do this by verifying the presence of Power Pivot-provided library templates:
   a. Select **Site Contents**.
   b. In the app list, you should see **Data Feed Library** and **Power Pivot Gallery**. These library templates are provided by the Power Pivot feature and will be visible in the Libraries list if the feature is integrated correctly.

Verify Data Access on the Server

To verify Power Pivot data access on the server, do the following:

1. Download the Picnic data sample that accompanies a Reporting Services tutorial. You will use the sample workbook in this download to verify Power Pivot data access. Extract the files.
2. Upload the Excel workbook (.xlsx) to Shared Documents. The workbook contains embedded Power Pivot data.
3. Click on the document to open it from the library.
4. Click on a slicer or filter at the top of the workbook. Month, color, and type are slicers in this workbook. Clicking a slicer starts a Power Pivot query and proves that your server is operational. The server will load Power Pivot data in the background and return the results.
5. Go back to the library. Select the down arrow to the right of the workbook, and then click **Launch Power View**. This step confirms that the Power View feature in Reporting Services is operational. If you did not install Reporting Services, skip this step.

   In the next step, you will connect to the server in Management Studio to verify the data is loaded and cached.

6. Start SQL Server Management Studio from the **Microsoft SQL Server 2017** program group in the Start menu. If this tool is not installed on your server, you can skip ahead to the last step to confirm the presence of cached files.
7. In Server Type, select **Analysis Services**.
8. In Server Name, enter **<server-name>\powerpivot**, where **<server-name>** is the name of the computer that has the Power Pivot for SharePoint installation.
9. Click **Connect**. This verifies that the Analysis Services server is available.

10. In Object Explorer, you can click **Databases** to view the list of Power Pivot data files that are loaded.

11. On the computer file system, check the following folder to determine whether files are cached to disk. The presence of cached files is further verification that your deployment is operational. To view the file cache, go to the <*drive*>:\Program Files\Microsoft SQL Server*nnn*\MSAS13.POWERPIVOT\OLAP\Backup\Sandboxes\Default Power Pivot Service Application folder. Each cached database is stored in its own folder, using a GUID-based naming convention to ensure a unique name.

Power Pivot Minimum-Privilege Example - SharePoint 2016

8/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes an example Power Pivot for SharePoint 2016 configuration with minimum privileges. The configuration utilizes a different account for each of the three components and each account has the minimum level of privileges.

Summary of Accounts

Power Pivot for SharePoint 2016 supports the use of the Network Service account for the Analysis Services service account. The Network Service account is not a supported scenario with SharePoint 2010. For more information on Service accounts, see Configure Windows Service Accounts and Permissions (http://msdn.microsoft.com/library/ms143504.aspx).

The following table summarizes the three accounts used in this example of a minimum privileged configuration.

SCOPE	NAME
SharePoint Administrator account	**SPAdmin**
SharePoint Farm account	**SPFarm**
Analysis Services service account	**SPsvc**

The SharePoint Administrator account (SpAdmin)

SPAdmin is a domain account you use to install and configure the farm. It is the account used to run the SharePoint Configuration Wizard and the Power Pivot Configuration Tool for SharePoint 2016.The **SPAdmin** account is the only account that requires local Administrator rights. Before running the Power Pivot Configuration tool, grant the **SPAdmin** account privileges to the SQL Server database instance where SharePoint creates content and configuration databases. To configure the SPAdmin account in a minimum privilege scenario, it should be a member of the roles **securityadmin** and **dbcreator**.

The Farm account (SPFarm)

SPFarm is a domain account that the SharePoint Timer service and the web application for Central Administration use to access the SharePoint content database. This account does not need to be a local administrator. The SharePoint configuration wizard grants the proper minimal privilege in the back-end SQL Server database.The minimum SQL Server privilege configuration is membership in the roles **securityadmin** and **dbcreator**.

The Service Account for Power Pivot Service (SPsvc)

If a new SharePoint farm is not configured before you run the Power Pivot Configuration tool, then by default the Power Pivot Configuration tool will create the following:

- Power Pivot Service application.
- Secure Store application.

The Power Pivot configuration tool configures both service applications in the default application pool. That

application pool is typically configured to run as the SPFarm account, which has access to many resources that a service account does not require. To make the environment a minimum-privileged environment, configure a new domain account to be use by the appropriate application pool and web application.

To create a new domain account SPsvc to be used as a SharePoint Service account:

1. In SharePoint Central Administration, select **Security**.
2. Select **Configure Service Accounts**
3. Select **Register new managed account**.

The **SPSvc** account has no local administrator privileges and SPsvc will not have any privileges in the SharePoint database. The only privileges SPsvc requires is administrative rights to the Power Pivot Instance of the Analysis Services.

To configure the appropriate application pool to use the SPsvc account :

1. In SharePoint Central Administration, select **Security**.
2. Select **Configure Service Accounts**.
3. Select the service application pool used by the Power Pivot Service application. Then select the SPSvc account.

To Grant access to the web application with PowerShell:

1. Run the SharePoint 2016 Management Shell with administrator privileges.
2. Run the following PowerShell code:

```
$webApp = Get-SPWebApplication "http://<servername>"
$webApp.GrantAccessToProcessIdentity("DOMAIN\<ServiceAccountName>")
```

Power Pivot Minimum-Privilege Example - SharePoint 2013

8/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes an example Power Pivot for SharePoint 2013 configuration with minimum privileges. The configuration utilizes a different account for each of the three components and each account has the minimum level of privileges.

Summary of Accounts

Power Pivot for SharePoint 2013 supports the use of the Network Service account for the Analysis Services service account. The Network Service account is not a supported scenario with SharePoint 2010. For more information on Service accounts, see Configure Windows Service Accounts and Permissions (http://msdn.microsoft.com/library/ms143504.aspx).

The following table summarizes the three accounts used in this example of a minimum privileged configuration.

SCOPE	NAME
SharePoint Administrator account	**SPAdmin**
SharePoint Farm account	**SPFarm**
Analysis Services service account	**SPsvc**

The SharePoint Administrator account (SpAdmin)

SPAdmin is a domain account you use to install and configure the farm. It is the account used to run the SharePoint Configuration Wizard and the Power Pivot Configuration Tool for SharePoint 2013.The **SPAdmin** account is the only account that requires local Administrator rights. Before running the Power Pivot Configuration tool, grant the **SPAdmin** account privileges to the SQL Server database instance where SharePoint creates content and configuration databases. To configure the SPAdmin account in a minimum privilege scenario, it should be a member of the roles **securityadmin** and **dbcreator**.

The Farm account (SPFarm)

SPFarm is a domain account that the SharePoint Timer service and the web application for Central Administration use to access the SharePoint content database. This account does not need to be a local administrator. The SharePoint configuration wizard grants the proper minimal privilege in the back-end SQL Server database.The minimum SQL Server privilege configuration is membership in the roles **securityadmin** and **dbcreator**.

The Service Account for Power Pivot Service (SPsvc)

If a new SharePoint farm is not configured before you run the Power Pivot Configuration tool, then by default the Power Pivot Configuration tool will create the following:

- Power Pivot Service application.
- Excel Services application.
- Secure Store application.

The Power Pivot configuration tool configures all three of the service applications in the default application pool. That application pool is typically configured to run as the SPFarm account, which has access to many resources that a service account does not require.To make the environment a minimum-privileged environment, configure a new domain account to be use by the appropriate application pool and web application.

To create a new domain account SPsvc to be used as a SharePoint Service account:

1. In SharePoint Central Administration, select **Security**.
2. Select **Configure Service Accounts**
3. Select **Register new managed account**.

The **SPSvc** account has no local administrator privileges and SPsvc will not have any privileges in the SharePoint database. The only privileges SPsvc requires is administrative rights to the Power Pivot Instance of the Analysis Services.

To configure the appropriate application pool to use the SPsvc account :

1. In SharePoint Central Administration, select **Security**.
2. Select **Configure Service Accounts**.
3. Select the service application pool used by the Power Pivot Service application. Then select the SPSvc account.

To Grant access to the web application with PowerShell:

1. Run the SharePoint 2013 Management Shell with administrator privileges.
2. Run the following PowerShell code:

```
$webApp = Get-SPWebApplication "http://<servername>"
$webApp.GrantAccessToProcessIdentity("DOMAIN\<ServiceAccountName>")
```

Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2016)

11/27/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Power Pivot for SharePoint 2013 is a collection of application server components and back-end services that provide Power Pivot data access in a SharePoint Server 2016 farm. The Power Pivot for SharePoint add-in (**spPowerpivot16.msi**) is an installer package used to install the application server components.

Note: This topic describes installing the Power Pivot solution files and Power Pivot for SharePoint 2016 Configuration tool. After the installation, see the following topic for information on the configuration tool and additional features, Configure Power Pivot and Deploy Solutions (SharePoint 2013).

For information on how to download **spPowerPivot16.msi**, see Microsoft® SQL Server® 2016 Power Pivot® for Microsoft SharePoint®.

Background

- **Application Server:** Power Pivot functionality in SharePoint 2016 includes using workbooks as a data source, scheduled data refresh, and the Power Pivot Management Dashboard.

  Power Pivot for SharePoint 2016 is a Microsoft Windows Installer package (**spPowerpivot16.msi**) that deploys Analysis Services client libraries and copies Power Pivot for SharePoint 2013 installation files to the computer. The installer does not deploy or configure Power Pivot features in SharePoint. The following components install by default:

  - Power Pivot for SharePoint 2016. This component includes PowerShell scripts (.ps1 files), SharePoint solution packages (.wsp), and the Power Pivot for SharePoint 2016 configuration tool to deploy Power Pivot in a SharePoint 2016 farm.
  - Microsoft OLE DB Provider for Analysis Services (MSOLAP).
  - ADOMD.NET data provider.
  - SQL Server Analysis Management Objects.

- **Backend services:** If you use Power Pivot for Excel to create workbooks that contain analytical data, you must have Office Online Server configured with a BI server running Analysis Services in Power Pivot mode to access that data in a server environment. You can run SQL Server Setup on a computer that has SharePoint Server 2016 installed, or on a different computer that has no SharePoint software. Analysis Services does not have any dependencies on SharePoint.

  For more information on installing, uninstalling, and configuring the backend services, see the following:

  - Install Analysis Services in Power Pivot Mode
  - Uninstall Power Pivot for SharePoint

Where to Install spPowerPivot16.msi?

A recommended best practice is to install **spPowerPivot16.msi** on all servers in the SharePoint farm for configuration consistency, including application servers and web-front end servers. The installer package includes

the Analysis Services data providers as well as the Power Pivot for SharePoint 2016 configuration tool. When you install **spPowerPivot16.msi** you can customize the installation by excluding individual components.

Data providers: Several SharePoint and SQL Server technologies use the Analysis Services data providers including PerformancePoint Services and Power View. Installing **spPowerPivot16.msi** on all SharePoint servers ensures the full set of Analysis Services data providers and Power Pivot connectivity is consistently available across the farm.

> **NOTE**
> You must install the Analysis Services data providers on a SharePoint 2016 server using **spPowerPivot16.msi**. Other installer packages available in the SQL Server 2017 Feature Pack are not supported because these packages do not include the SharePoint 2016 support files that the data providers require in this environment.

Configuration Tool: The Power Pivot for SharePoint 2016 configuration tool is required on only one of the SharePoint servers. However a recommended best practice in multi-server farms is to install the configuration tool on at least two servers so you have access to the configuration tool if one of the two servers is offline.

Requirements and Prerequisites

- Microsoft SharePoint Server 2016.
- **spPowerPivot16.msi** is 64-bit only, in accordance with the requirements of SharePoint products and technologies.
- A server in Power Pivot mode. Office Online Server will use the SQL Server Analysis Services instance as a Power Pivot server. Analysis Services can run on the local SharePoint server or a remote computer. It cannot be installed on the Office Online Server.
- **Permissions:** To install Power Pivot for SharePoint 2016, the current user is required to be an administrator on the computer and in the SharePoint Farm Administrators group.
- For more information on Power Pivot for SharePoint requirements and pre-requisites, go to Hardware and Software Requirements for Analysis Services Server in SharePoint Mode.

To Install Power Pivot for SharePoint

The **spPowerpivot16.msi** installer package supports both a graphical user interface and a command-line mode. Both methods of installation require that you run the .msi with administrator privileges. After the installation, see the following topic for information on the configuration tool and additional features, Configure Power Pivot and Deploy Solutions (SharePoint 2013).

User interface installation

To install Power Pivot for SharePoint 2016 with the graphical user interface, complete the following steps:

1. Run **spPowerPivot16.msi**.
2. Select **Next** on the Welcome page.
3. Review and accept the license agreement, then select **Next**.
4. On the **Feature Selection** page, all of the features are selected by default.
5. Select **Next**.
6. Select **Install** to install to finish the installation.

Command Line Installation

For a command-line installation, open a command prompt with administrative permissions, and then run the **spPowerPivot16.msi**. For example:

`Msiexec.exe /i spPowerPivot16.msi`.

To create an installation log, use the standard MsiExec logging switches. The following example creates the log file "Install_Log.txt" using the "v" verbose logging switch.

```
Msiexec.exe /i spPowerPivot16.msi /L v c:\test\Install_Log.txt
```

Quiet Command Line Installation for scripting

You can use the **/q** or **/quiet** switches for a "quiet" installation that will not display any dialogs or warnings. The quiet installation is useful if you want to script the installation of the add-in.

> **IMPORTANT**
> If you use the **/q** switch for a silent command line installation, the end-user license agreement will not be displayed. Regardless of the installation method, the use of this software is governed by a license agreement and you are responsible for complying with the license agreement.

To perform a quiet installation:

1. Open a command prompt **with administrator permissions**.
2. Run the following command:

   ```
   Msiexec.exe /i spPowerPivot16.msi /q
   ```

Command Line Installation to include specific components

The Power Pivot for SharePoint 2016 Configuration tool is not required on every SharePoint server, however it is recommended to install it on at least two servers so the configuration tool is available when you need it.

When you install the spPowerPivot16.msi, you can use the command line options to install specific items, such as the data providers and not the Power Pivot for SharePoint 2016 Configuration tool. The following command line is an example of installing all components except the configuration tool:

```
Msiexec /i spPowerPivot16.msi AGREETOLICENSE="yes" ADDLOCAL=" SQL_OLAPDM,SQL_ADOMD,SQL_AMO,SQLAS_SP_Common"
```

OPTION	DESCRIPTION
Analysis_Server_SP_addin16	Power Pivot for SharePoint 2016 Configuration
SQL_OLAPDM	Analysis Services OLE DB Provider for SQL Server 2016
SQL_ADOMD	ADOMD.NET provider
SQL_AMO	SQL Server 2016 Analysis Management Objects (AMO) provider
SQLAS_SP16_Common	Analysis Services common components for SharePoint 2016

Deploy the SharePoint Solution Files with the Power Pivot for SharePoint 2016 Configuration Tool

Three of the files copied to the hard drive by spPowerPivot16.msi are SharePoint solution files. The scope of one solution file is the Web application level while the scope of the other files is the farm level. The files are the following:

- `PowerPivot16FarmSolution.wsp`
- `PowerPivot16WebApplicationSolution.wsp`

The solution files are copied to the following folder:

`ssInstallPathTools\PowerPivotTools\SPAddinConfiguration\Resources`

Following the .msi installation, run the Power Pivot for SharePoint 2016 Configuration Tool to configure and deploy the solutions in the SharePoint farm.

To start the configuration tool:

From the Windows Start screen type "power" and in the Apps search results, select **Power Pivot for SharePoint 2016 Configuration**. Note that the search results may include two links because SQL Server setup installs separate Power Pivot configuration tools for SharePoint 2013 and SharePoint 2016. Make sure you start the Power Pivot for SharePoint 2016 Configuration tool.

Or

1. Go to **Start**, **All Programs**.
2. Select **Microsoft SQL Server 2017**.
3. Select **Configuration Tools**.
4. Select **Power Pivot for SharePoint 2016 Configuration**.

For more information on the configuration tool, see Power Pivot Configuration Tools.

Uninstall or repair the add-in

Caution

If you uninstall **spPowerPivot16.msi** the data providers and the configuration tool are uninstalled. Uninstalling the data providers will cause the server to be unable to connect to Power Pivot.

You can uninstall or repair Power Pivot for SharePoint 2016 using one of the following methods:

1. **Windows control panel:** Select **Microsoft SQL Server 2017Power Pivot for SharePoint 2016**. Select either **Uninstall** or **Repair**.
2. Run the spPowerPivot16.msi and select the **Remove** option or the **Repair** option.

Command Line: To repair or uninstall Power Pivot for SharePoint 2016 using the command line, open a command prompt **with administrator permissions** and run one of the following commands:

- To Repair, run the following command:

```
msiexec.exe /f spPowerPivot16.msi
```

OR

- To uninstall, run the following command:

```
msiexec.exe /uninstall spPowerPivot16.msi
```

Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2013)

11/27/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Power Pivot for SharePoint 2013 is a collection of application server components and back-end services that provide Power Pivot data access in a SharePoint Server 2013 farm. The Power Pivot for SharePoint add-in (**spPowerpivot.msi**) is an installer package used to install the application server components.

- The add-in is not required for SharePoint 2010 deployments.
- The add-in is not required on a single server deployment that includes SharePoint 2013 and Analysis Services in SharePoint mode. The components installed by the add-in are included when you install an Analysis Services server in SharePoint mode. For diagrams of example deployments with the add-in, see Deployment Topologies for SQL Server BI Features in SharePoint

Note: This topic describes installing the Power Pivot solution files and Power Pivot for SharePoint 2013 Configuration tool. After the installation, see the following topic for information on the configuration tool and additional features, Configure Power Pivot and Deploy Solutions (SharePoint 2013).

For information on how to download **spPowerPivot.msi**, see Microsoft® SQL Server® 2014 Power Pivot® for Microsoft SharePoint®.

Background

- **Application Server:** Power Pivot functionality in SharePoint 2013 includes using workbooks as a data source, scheduled data refresh, and the Power Pivot Management Dashboard.

  Power Pivot for SharePoint 2013 is a Microsoft Windows Installer package (**spPowerpivot.msi**) that deploys Analysis Services client libraries and copies Power Pivot for SharePoint 2013 installation files to the computer. The installer does not deploy or configure Power Pivot features in SharePoint. The following components install by default:

  - Power Pivot for SharePoint 2013. This component includes PowerShell scripts (.ps1 files), SharePoint solution packages (.wsp), and the Power Pivot for SharePoint 2013 configuration tool to deploy Power Pivot in a SharePoint 2013 farm.
  - Microsoft OLE DB Provider for Analysis Services (MSOLAP).
  - ADOMD.NET data provider.
  - SQL Server Analysis Management Objects.

- **Backend services:** If you use Power Pivot for Excel to create workbooks that contain analytical data, you must have Excel Services configured with a BI server running Analysis Services in SharePoint mode to access that data in a server environment. You can run SQL Server Setup on a computer that has SharePoint Server 2013 installed, or on a different computer that has no SharePoint software. Analysis Services does not have any dependencies on SharePoint.

  For more information on installing, uninstalling, and configuring the backend services, see the following:

  - Install Analysis Services in Power Pivot Mode

- Uninstall Power Pivot for SharePoint

Where to Install spPowerPivot.msi?

A recommended best practice is to install **spPowerPivot.msi** on all servers in the SharePoint farm for configuration consistency, including application servers and web-front end servers. The installer package includes the Analysis Services data providers as well as the Power Pivot for SharePoint 2013 configuration tool. When you install **spPowerPivot.msi** you can customize the installation by excluding individual components.

Data providers: Several SharePoint and SQL Server technologies use the Analysis Services data providers including Excel Services, PerformancePoint Services, and Power View. Installing **spPowerPivot.msi** on all SharePoint servers ensures the full set of Analysis Services data providers and Power Pivot connectivity is consistently available across the farm.

> **NOTE**
>
> You must install the Analysis Services data providers on a SharePoint 2013 server using **spPowerPivot.msi**. Other installer packages available in the SQL Server 2017 Feature Pack are not supported because these packages do not include the SharePoint 2013 support files that the data providers require in this environment.

Configuration Tool: The Power Pivot for SharePoint 2013 configuration tool is required on only one of the SharePoint servers. However a recommended best practice in multi-server farms is to install the configuration tool on at least two servers so you have access to the configuration tool if one of the two servers is offline.

Requirements and Prerequisites

- Microsoft SharePoint Server 2013.
- **spPowerPivot.msi** is 64-bit only, in accordance with the requirements of SharePoint products and technologies.
- A server in Power Pivot mode. Excel Services will use the SQL Server Analysis Services instance as a Power Pivot server. Analysis Services can run on the local or a remote computer.
- **Permissions:** To install Power Pivot for SharePoint 2013, the current user is required to be an administrator on the computer and a SharePoint Farm Administrators group.
- For more information on Power Pivot for SharePoint requirements and pre-requisites, go to Hardware and Software Requirements for Analysis Services Server in SharePoint Mode.

To Install Power Pivot for SharePoint

The **spPowerpivot.msi** installer package supports both a graphical user interface and a command-line mode. Both methods of installation require that you run the .msi with administrator privileges. After the installation, see the following topic for information on the configuration tool and additional features, Configure Power Pivot and Deploy Solutions (SharePoint 2013).

User interface installation

To install Power Pivot for SharePoint 2013 with the graphical user interface, complete the following steps:

1. Run **SpPowerPivot.msi**.
2. Click **Next** on the Welcome page.
3. Review and accept the license agreement, then click **Next**.
4. On the **Feature Selection** page, all of the features are selected by default.

5. Click **Next**.

6. Click **Install** to install to finish the installation.

Command Line Installation

For a command-line installation, open a command prompt with administrative permissions, and then run the **spPowerPivot.msi**. For example:

`Msiexec.exe /i SpPowerPivot.msi`.

To create an installation log, use the standard MsiExec logging switches. The following example creates the log file "Install_Log.txt" using the "v" verbose logging switch.

```
Msiexec.exe /i SpPowerPivot.msi /L v c:\test\Install_Log.txt
```

Quiet Command Line Installation for scripting

You can use the **/q** or **/quiet** switches for a "quiet" installation that will not display any dialogs or warnings. The quiet installation is useful if you want to script the installation of the add-in.

> **IMPORTANT**
>
> If you use the **/q** switch for a silent command line installation, the end-user license agreement will not be displayed. Regardless of the installation method, the use of this software is governed by a license agreement and you are responsible for complying with the license agreement.

To perform a quiet installation:

1. Open a command prompt **with administrator permissions**.

2. Run the following command:

```
Msiexec.exe /i spPowerPivot.msi /q
```

Command Line Installation to include specific components

The Power Pivot for SharePoint 2013 Configuration tool is not required on every SharePoint server, however it is recommended to install it on at least two servers so the configuration tool is available when you need it.

When you install the spPowerPivot.msi, you can use the command line options to install specific items, such as the data providers and not the Power Pivot for SharePoint 2013 Configuration tool. The following command line is an example of installing all components except the configuration tool:

```
Msiexec /i spPowerPivot.msi AGREETOLICENSE="yes" ADDLOCAL=" SQL_OLAPDM,SQL_ADOMD,SQL_AMO,SQLAS_SP_Common"
```

OPTION	DESCRIPTION
Analysis_Server_SP_addin	Power Pivot Configuration
SQL_OLAPDM	MSOLAP
SQL_ADOMD	ADOMD.net provider
SQL_AMO	AMO provider

OPTION	DESCRIPTION
SQLAS_SP_Common	Analysis Services common components for SharePoint 2013

Deploy the SharePoint Solution Files with the Power Pivot for SharePoint 2013 Configuration Tool

Three of the files copied to the hard drive by spPowerPivot.msi are SharePoint solution files. The scope of one solution file is the Web application level while the scope of the other files is the farm level. The files are the following:

- `PowerPivotFarmSolution.wsp`
- `PowerPivotFarm14Solution.wsp`
- `PowerPivotWebApplicationSolution.wsp`

The solution files are copied to the following folder:

`ssInstallPathTools\PowerPivotTools\SPAddinConfiguration\Resources`

Following the .msi installation, run the Power Pivot for SharePoint 2013 Configuration Tool to configure and deploy the solutions in the SharePoint farm.

To start the configuration tool:

From the Windows Start screen type "power" and in the Apps search results, click **Power Pivot for SharePoint 2013 Configuration**. Note that the search results may include two links because SQL Server setup installs separate Power Pivot configuration tools for SharePoint 2010 and SharePoint 2013. Make sure you start the Power Pivot for SharePoint 2013 Configuration tool.

Or

1. Go to **Start**, **All Programs**.
2. Click **Microsoft SQL Server 2017**.
3. Click **Configuration Tools**.
4. Click **Power Pivot for SharePoint 2013 Configuration**.

For more information on the configuration tool, see Power Pivot Configuration Tools.

Uninstall or repair the add-in

Caution

If you uninstall **spPowerPivot.msi** the data providers and the configuration tool are uninstalled. Uninstalling the data providers will cause the server to be unable to connect to Power Pivot.

You can uninstall or repair Power Pivot for SharePoint 2013 using one of the following methods:

1. **Windows control panel:** Select **Microsoft SQL Server 2017Power Pivot for SharePoint 2013**. Click either **Uninstall** or **Repair**.

2. Run the spPowerPivot.msi and select the **Remove** option or the **Repair** option.

Command Line: To repair or uninstall Power Pivot for SharePoint 2013 using the command line, open a command prompt **with administrator permissions** and run one of the following commands:

- To Repair, run the following command:

```
msiexec.exe /f spPowerPivot.msi
```

OR

- To uninstall, run the following command:

```
msiexec.exe /uninstall spPowerPivot.msi
```

Configure Power Pivot and Deploy Solutions (SharePoint 2016)

11/27/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topics describes the deployment and configuration of middle-tier enhancements to the Power Pivot features in SharePoint Server 2016 including Power Pivot Gallery, Schedule data refresh, Management Dashboard, and data providers. Run **Power Pivot for SharePoint 2016 Configuration** tool to complete the following:

- Deploy SharePoint solution files.
- Create a Power Pivot service application.
- For information on backend services and installing a Analysis Services server in Power Pivot mode, see Install Analysis Services in Power Pivot Mode.

For information on installing the Power Pivot for SharePoint 2016 Configuration tool, see Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2016).

Run Power Pivot for SharePoint 2016 configuration

Note: To complete the following steps, you must be a farm administrator. If you see an error message similar to the following:

- "The user is not a farm administrator. Please address the validation failures and try again."

Either login as the account that installed SharePoint or configure the setup account as the primary administrator of the SharePoint Central Administration Site.

1. On the **Start** menu, select **All Programs**, and then select **Microsoft SQL Server 2017**, select **Configuration Tools**, and then select **Power Pivot For SharePoint 2016 Configuration**. Tools is listed only when Power Pivot for SharePoint is installed on the local server.
2. Select **Configure or Repair Power Pivot for SharePoint** and then select **OK**.
3. The tool runs validation to verify the current state of Power Pivot and what steps are required to complete configuration. Expand the window to full size. You should see a button bar at the bottom of the window that includes **Validate**, **Run**, and **Exit** commands.
4. On the **Parameters** tab:
   a. **Default Account UserName**: Enter a domain user account for the default account. This account will be used to provision services, including the Power Pivot service application pool. Do not specify a built-in account such as Network Service or Local System. The tool blocks configurations that specify built-in accounts.
   b. **Database Server**: You can use SQL Server Database engine that is supported for the SharePoint farm.
   c. **Passphrase**: Enter a passphrase. If you are creating a new SharePoint farm, the passphrase is used whenever you add a server or application to the SharePoint farm. If the farm already exists, enter the passphrase that allows you to add a server application to the farm.

d. Click **Create Site Collection** in the left window. Note **Site URL** so you can reference it in later steps. If the SharePoint server is not already configured, then the configuration wizard defaults the web application, and site collection URLs to the root of `http://[ServerName]`. To modify the defaults review the following pages in the left window: **Create Default Web application** and **Deploy Web Application Solution**

5. Optionally, review the remaining input values used to complete each action. Click each action in the left window to see and review the details of the action. For more information about each one, see the section "Input values used to configure the server in Configure or Repair Power Pivot for SharePoint 2010 (Power Pivot Configuration Tool) in this topic.

6. Optionally, remove any actions that you do not want to process at this time. For example, if you want to configure Secure Store Service later, select **Configure Secure Store Service**, and then clear the checkbox **Include this action in the task list**.

7. select **Validate** to check whether the tool has sufficient information to process the actions in the list. If you see validation errors, click the warnings in the left pane to see details of the validation error. Correct any validation errors and then select **Validate** again.

8. Select **Run** to process all of the actions in the task list. Note that **Run** becomes available after you validate the actions. If **Run** is not enabled, select **Validate** first.

For more information, see Configure or Repair Power Pivot for SharePoint 2010 (Power Pivot Configuration Tool)

Verify Power Pivot Configuration

Services:

1. In Central Administration, in System Settings, select **Manage services on server**.

2. Verify that **SQL Server Power Pivot System Service** is started.

Farm Feature:

1. In Central Administration, in System Settings, select **Manage farm features**.

2. Verify that **Power Pivot Integration Feature** is **Active**.

Site Collection Feature:

1. Browse to your site URL that was created by the Configuration tool.

   Select **Settings**⚙, and then click **Site Settings**.

   Select **Site Collection Features**.

2. Verify that **Power Pivot Feature Integration for Site Collections** is **Active**.

Power Pivot Service Application:

1. In Central Administration, in the **Application Management**, select **Manage service applications**.

2. Verify the service application status is **started**. The default name is **Default Power Pivot Service Application**.

   Select the name of the services application to open the Power Pivot Management Dashboard for the service application opens. On first use, the dashboard takes several minutes to load.

For more information, see Verify a Power Pivot for SharePoint Installation.

Troubleshoot Issues

To assist in troubleshooting issues, it is a good idea to verify the diagnostic logging is enabled.

1. In SharePoint Central Administration, click **Monitoring** and then select **Configure usage and health data collection**.
2. Verify **Enable usage data collection** is selected.
3. Verify the following events are selected:
   - Definition of usage fields for Education telemetry
   - Power Pivot Connections
   - Power Pivot Load Data Usage
   - Power Pivot Query Usage
   - Power Pivot Unload Data Usage
4. Verify **Enable health data collection** is selected.
5. Select **OK**.

For more information on trouble shooting data refresh, see Troubleshooting Power Pivot Data Refresh (http://social.technet.microsoft.com/wiki/contents/articles/3870.troubleshooting-powerpivot-data-refresh.aspx).

For more information on the configuration tool, see Power Pivot Configuration Tools.

Configure Power Pivot and Deploy Solutions (SharePoint 2013)

11/27/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topics describes the deployment and configuration of middle-tier enhancements to the Power Pivot features in SharePoint Server 2013 including Power Pivot Gallery, Schedule data refresh, Management Dashboard, and data providers. Run **Power Pivot for SharePoint 2013 Configuration** tool to complete the following:

- Deploy SharePoint solution files.
- Create a Power Pivot service application.
- Configure an Excel Services Application to use an Analysis Services server in SharePoint mode. For information on backend services and installing a Analysis Services server in SharePoint mode, see Install Analysis Services in Power Pivot Mode.

For information on installing the Power Pivot for SharePoint 2013 Configuration tool, see Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2013)

Run Power Pivot for SharePoint 2013 configuration

Note: The SQL Server 2017 setup wizard installs two different configuration tools for Microsoft SQL Server 2016 Power Pivot for SharePoint. They each support a different version of SharePoint.

NAME	DESCRIPTION
Power Pivot for SharePoint 2013 Configuration	SharePoint 2013
Power Pivot Configuration Tool	SharePoint 2010 with SharePoint 2010 Service Pack 1 (SP1)

Note: To complete the following steps, you must be a farm administrator. If you see an error message similar to the following:

- "The user is not a farm administrator. Please address the validation failures and try again."

Either login as the account that installed SharePoint or configure the setup account as the primary administrator of the SharePoint Central Administration Site.

1. On the **Start** menu, click **All Programs**, and then click **Microsoft SQL Server 2017**, click **Configuration Tools**, and then click **Power Pivot For SharePoint 2013 Configuration**. Toold is listed only when Power Pivot for SharePoint is installed on the local server.
2. Click **Configure or Repair Power Pivot for SharePoint** and then click **OK**.
3. The tool runs validation to verify the current state of Power Pivot and what steps are required to complete configuration. Expand the window to full size. You should see a button bar at the bottom of the window that includes **Validate**, **Run**, and **Exit** commands.
4. On the **Parameters** tab:
   a. **Default Account UserName**: Enter a domain user account for the default account. This account

will be used to provision services, including the Power Pivot service application pool. Do not specify a built-in account such as Network Service or Local System. The tool blocks configurations that specify built-in accounts.

b. **Database Server**: You can use SQL Server Database engine that is supported for the SharePoint farm.

c. **Passphrase**: Enter a passphrase. If you are creating a new SharePoint farm, the passphrase is used whenever you add a server or application to the SharePoint farm. If the farm already exists, enter the passphrase that allows you to add a server application to the farm.

d. **Power Pivot Server for Excel Services**: Type the name of an Analysis Services SharePoint mode server. In a single-server deployment, it is the same as the database server. `[ServerName]\powerpivot`

e. Click **Create Site Collection** in the left window. Note **Site URL** so you can reference it in later steps. If the SharePoint server is not already configured, then the configuration wizard defaults the web application, and site collection URLs to the root of `http://[ServerName]`. To modify the defaults review the following pages in the left window: **Create Default Web application** and **Deploy Web Application Solution**

5. Optionally, review the remaining input values used to complete each action. Click each action in the left window to see and review the details of the action. For more information about each one, see the section "Input values used to configure the server in Configure or Repair Power Pivot for SharePoint 2010 (Power Pivot Configuration Tool) in this topic.

6. Optionally, remove any actions that you do not want to process at this time. For example, if you want to configure Secure Store Service later, click **Configure Secure Store Service**, and then clear the checkbox **Include this action in the task list**.

7. Click **Validate** to check whether the tool has sufficient information to process the actions in the list. If you see validation errors, click the warnings in the left pane to see details of the validation error. Correct any validation errors and then click **Validate** again.

8. Click **Run** to process all of the actions in the task list. Note that **Run** becomes available after you validate the actions. If **Run** is not enabled, click **Validate** first.

For more information, see Configure or Repair Power Pivot for SharePoint 2010 (Power Pivot Configuration Tool)

Verify Power Pivot Configuration

Services:

1. In Central Administration, in System Settings, click **Manage services on server**.

2. Verify that **SQL Server Analysis Services** and **SQL Server Power Pivot System Service** are started.

Farm Feature:

1. In Central Administration, in System Settings, click **Manage farm features**.

2. Verify that **Power Pivot Integration Feature** is **Active**.

Site Collection Feature:

1. Browse to your site URL that was created by the Configuration tool.

   Click **Settings**⚙, and then click **Site Settings**.

Click **Site Collection Features**.

2. Verify that **Power Pivot Feature Integration for Site Collections** is **Active**.

Power Pivot Service Application:

1. In Central Administration, in the **Application Management**, click **Manage service applications**.
2. Verify the service application status is **started**. The default name is **Default Power Pivot Service Application**.

   Click the name of the services application to open the Power Pivot Management Dashboard for the service application opens. On first use, the dashboard takes several minutes to load.

For more information, see Verify a Power Pivot for SharePoint Installation.

Troubleshoot Issues

To assist in troubleshooting issues, it is a good idea to verify the diagnostic logging is enabled.

1. In SharePoint Central Administration, click **Monitoring** and then click **Configure usage and health data collection**.
2. Verify **Enable usage data collection** is selected.
3. Verify the following events are selected:
   - Definition of usage fields for Education telemetry
   - Power Pivot Connects
   - Power Pivot Load Data Usage
   - Power Pivot Query Usage
   - Power Pivot Unload Data Usage
4. Verify **Enable health data collection** is selected.
5. Click **OK**.

For more information on trouble shooting data refresh, see Troubleshooting Power Pivot Data Refresh (http://social.technet.microsoft.com/wiki/contents/articles/3870.troubleshooting-powerpivot-data-refresh.aspx).

For more information on the configuration tool, see Power Pivot Configuration Tools.

Migrate Power Pivot to SharePoint 2013

12/10/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

SharePoint 2013 does not support in-place upgrade. However the procedure of **database-attach upgrade is supported**. The behavior is different from upgrading to SharePoint 2010, where a customer could choose between the two basic upgrade approaches, in-place upgrade and database-attach upgrade.

If you have a Power Pivot for SharePoint installation integrated with SharePoint 2010, you cannot upgrade in-place the SharePoint server. However you can migrate content databases and service application databases from the SharePoint 2010 farm to a SharePoint 2013 farm. This topic is an overview of the steps required to complete a database-attach upgrade and complete a migration related to Power Pivot:

Migration Overview

1	2	3	4
Prepare the SharePoint 2013 farm	Backup, copy, restore databases.	Mount content databases	Migrate Power Pivot Schedules
	SQL Server Management Studio	-SharePoint Central Administration -Windows PowerShell	-SharePoint application Pages -Windows PowerShell

Prepare the SharePoint 2013 Farm

> **TIP**
> Review the authentication method your existing web applications are configured for. SharePoint 2013 web applications default to claims-based authentication. SharePoint 2010 web applications configured for classic-mode authentication require additional steps to migrate databases from SharePoint 2010 to SharePoint 2013. If your web applications are configured for classic-mode authentication, review the SharePoint 2013 documentation.

1. Install a new SharePoint Server 2013 farm.
2. Install an instance of a SQL Server 2017 Analysis Services server in SharePoint mode. For more information, see Install Analysis Services in Power Pivot Mode.
3. Run the Power Pivot for SharePoint 2013 installation package **spPowerPivot.msi** on each server in the SharePoint farm. For more information, see Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2013).
4. In SharePoint 2013 Central Administration, configure the Excel Services service application to use the Analysis Services SharePoint mode server created in the previous step. For more information, see the "Configure Basic Analysis Services SharePoint Integration" section of Install Analysis Services in Power Pivot Mode.

Backup, Copy, Restore the Databases

The "SharePoint database-attach upgrade" process is a sequence of steps to back up, copy, and restore Power Pivot

related content and service application databases to the SharePoint 2013 farm.

1. **Set Database to read-only:** In SQL Server Management Studio, right-click the database name and click **Properties**. On the **Options** page, set the **Database read-Only** property to **True**.
2. **Back up:** Back up each content database and service application database that you want to migrate to the SharePoint 2013 farm. In SQL Server Management Studio, right-click the database name, click **Tasks**, and click **Back up**.
3. File copy the database backup files (.bak) to the desired destination server.
4. **Restore:** Restore the databases to the destination SQL Server Database Engine. This step can be completed using SQL Server Management Studio.
5. **Set Database to read-write:** Set the **Database read-Only** to **False**.

Prepare Web Applications and Mount Content Databases

For a more detailed explanation of the following procedures, see Upgrade databases from SharePoint 2010 to SharePoint 2013 (http://go.microsoft.com/fwlink/p/?LinkId=256690).

1. **Take Databases Offline:**

   Take each SharePoint 2013 content database offline, using SharePoint Central Administration. The content databases are replaced by the databases you copied over. Consider what is the best sequence for your environment. Consider taking each database offline and mount its relevant replacement database before taking the next content database offline. Another option is to take all the content databases offline as a group.

   a. In SharePoint Central Administration, click **Application Management**.

   b. Click **Manage Content Databases**.

   c. Click the name of the database.

   d. On the **Manage Content Database Settings**, set **Database status** to **Offline**.

   e. Select **Remove Content Database**. Note the warning that sites stored in the content database will no longer be accessible.

- **Mount content databases:**

  Use PowerShell cmdlets in the SharePoint 2013 Management shell to mount the migrated content database. The service application database does not need to be mounted, only the content databases:

```
Mount-SPContentDatabase "SharePoint_Content_O14-KJSP1" -DatabaseServer "[server name]\powerpivot" -
WebApplication [web application URL]
```

  For more information, see Attach or detach content databases (SharePoint Server 2010) (http://technet.microsoft.com/library/ff628582.aspx).

  Status when the step is complete: When the mount operation is complete, users can see files that were in the old content database. Therefore users can see and open the workbooks in the document library.

- **TIP**

  It is possible at this point in the migration process to create new schedules for the migrated workbooks. However, the schedules are created in the new Power Pivot service application database, and not the database you copied from the old SharePoint farm. Therefore it will not contain any of the old schedules. After you complete the following steps to use the old database and migrate the old schedules, the new schedules are not available.

Troubleshoot issues when you attempt to mount databases

This section summarizes possible issues encountered when mounting the database.

1. **Authentication errors:** If you see errors related to authentication, review what authentication mode the source web applications are using. The error could be caused by a mismatch in authentication between the SharePoint 2013 web application and the SharePoint 2010 web application. See the 1) Prepare the SharePoint 2013 Farm for more information.

2. **Missing PowerPivot.Files:** If you see errors related to missing Power Pivot .dlls, the **spPowerPivot.msi** has not been installed or the Power Pivot Configuration Tool has not been used to configure Power Pivot.

Upgrade Power Pivot Schedules

This section describes the details and options for migrating Power Pivot schedules. Schedule migration is a two-step process. First configure the Power Pivot service application to use the migrated service application database. Second, choose one of two options for schedule migration.

Configure the service application to use the migrated service application database.

In SharePoint Central Administration to configure the Power Pivot services application to use the old service application database you copied over. The Power Pivot Service upgrades the service application database to the new schema.

1. In SharePoint Central Administration, click **Manage Service Applications**.

2. Find the Power Pivot service application, for example "Default Power Pivot Service Application", click the name of the service application and click **Properties** in the SharePoint ribbon.

3. Update the database server name-instance and the database name. To the correct names for the database you backed up, copied, and restored. Once you click **Ok**, the service application database is upgraded. Errors will be in the ULS log.

Upgrade Power Pivot schedules

Configure the Power Pivot service application to migrate refresh schedules.

- **Migrate Schedules option1: SharePoint farm administrator**

  1. In the SharePoint 2013 Management Run the `Set-PowerPivotServiceApplication` cmdlet with the `-StartMigratingRefreshSchedules` switch to enable automatic on demand schedule migration .
     The following Windows PowerShell script assumes that there is only one Power Pivot service application.

     ```
     $app=Get-PowerPivotServiceApplication
     Set-PowerPivotServiceApplication $app -StartMigratingRefreshSchedules
     ```

     After the Windows PowerShell script is run, the schedules are active and the schedules will run at the

next appropriate time. However, the status one the schedule refresh page is not enabled. When the schedule runs the first time it will be migrated and on the schedule refresh page, **Enabled** will be true.

2. If you want to check the current value of the StartMigratingRefreshSchedules property, run the following PowerShell script. The Script loops through all Power Pivot service application objects and display the name and property values:

```
$apps = Get-PowerPivotServiceApplication
foreach ($app in $apps){}
Get-PowerPivotServiceApplication $appp | format-table -property
displayname,id,StartMigratingRefreshSchedules
```

Migrate Schedules option2: User updates each workbook

1. Another option to migrate schedules is to enable the schedule refresh for each workbook. Navigate to the document library that contains the workbooks.
2. Open the context menu and click **Manage Power Pivot Data Refresh**.
3. In the **schedule refresh** section, click **Enable**.
4. You can select **Also refresh as soon as possible**. This option adds one instance of the refresh to the queue as soon as you click ok. The regular refresh schedule still triggers at the appropriate time.
5. Click **Ok**. The refresh history is now visible in the refresh page, the schedule will fire at the normal time.

SQL Server 2008 R2 Power Pivot workbooks

- SQL Server 2008 R2 Power Pivot workbooks do not automatically upgrade when they are used in SQL Server 2017Power Pivot for SharePoint 2013. After you migrate a content database containing the 2008 R2 workbooks, you can use the workbooks but the schedules do not upgrade.
- For more information, see Upgrade Workbooks and Scheduled Data Refresh (SharePoint 2013).

Additional Resources

> **NOTE**
> For more information on Power Pivot and SharePoint database-attach upgrade, see the following:

- Upgrade Workbooks and Scheduled Data Refresh (SharePoint 2013).
- Overview of the upgrade process to SharePoint 2013 (http://go.microsoft.com/fwlink/p/?LinkId=256688).
- Clean up preparations before an upgrade to SharePoint 2013 (http://go.microsoft.com/fwlink/p/?LinkId=256689).
- Upgrade databases from SharePoint 2010 to SharePoint 2013 (http://go.microsoft.com/fwlink/p/?LinkId=256690).

Upgrade Workbooks and Scheduled Data Refresh (SharePoint 2013)

11/28/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic explains the user experience of workbooks created in previous Power Pivot environments and how to upgrade Power Pivot workbooks so that you can take advantage of new features introduced in this release. To learn more about new features, see What's New in Power Pivot.

> **WARNING**
> You cannot rollback upgrade for workbooks that are upgraded automatically on the server. Once a workbook is upgraded, it remains upgraded. To use a previous version, you can republish the previous workbook to SharePoint, restore a previous version, or recycle the workbook. For more information about restoring or recycling a document in SharePoint, see Plan to protect content by using recycle bins and versioning.

Overview of Upgrading Workbooks

A Power Pivot workbook is an Excel workbook that contains embedded Power Pivot data. Upgrading a workbook has two benefits:

- Use new features in Power Pivot for Excel.
- Enables scheduled data refresh for workbooks that run with a SQL Server 2012 SP1 (11.0.3x) Analysis Services server in SharePoint mode.

> **IMPORTANT**
> You cannot rollback an upgraded workbook, so be sure to make a copy of the file if you want to use it in the previous version of Power Pivot for Excel, or on a previous version of Power Pivot for SharePoint.

The following table lists the support and behavior of Power Pivot workbooks based on the environment in which the workbook was created. The behavior described includes the general user experience, the supported upgrade options to upgrade the workbook to the particular environment, and the behavior of scheduled data refresh of a workbook that has not yet been upgraded.

Workbook Behavior and Upgrade Options

CREATED IN	<	SUPPORT AND BEHAVIOR	>
	2008 R2 Power Pivot for SharePoint 2010	**2012 Power Pivot for SharePoint 2010**	**2012 SP1 Power Pivot for SharePoint 2013**

CREATED IN	<	SUPPORT AND BEHAVIOR	>
2008 R2 Power Pivot for Excel 2010	All Features	**Experience:** Users can interact with the workbook in the browser and use it as a data source for other solutions. **Upgrade:** Workbooks will auto upgrade in the document library if Auto Upgrade is enabled for the Power Pivot system Service in the SharePoint farm, **Schedule data refresh:** NOT supported. Workbook needs to be upgraded.	**Experience:** Users can interact with the workbook and use it as a data source for other solutions. **Upgrade:** Auto upgrade is not available. Users must manually upgrade their 2008 R2 workbooks to the 2012 version or to the office 2013 version. **Schedule data refresh:** NOT supported. Workbook needs to be upgraded.
2012 Power Pivot for Excel	Not supported	All Features	**Experience:** Users can interact with the workbook in the browser and use it as a data source for other solutions. Schedule data refresh is available. **Upgrade:** Auto upgrade is not supported. Users can manually upgrade their workbooks to the Office 2013 version. **Schedule data refresh:** supported.
Excel 2013	Not supported	Not supported	All Features

Upgrade to SQL Server 2012 Service Pack 1 (SP1) workbooks from 2008 R2 Workbooks

This section describes upgrading to SQL Server 2012 SP1 Power Pivot for Excel 2013 workbooks from SQL Server 2008 R2 Power Pivot for Excel 2010 workbooks.

Behavior Change: SQL Server 2008 R2 Power Pivot workbooks will not be automatically upgraded when they are used in SQL Server 2012 SP1 Power Pivot for SharePoint 2013. Therefore, scheduled data refreshes will not work for SQL Server 2008 R2 Power Pivot workbooks

2008 R2 workbooks will open in Power Pivot for SharePoint 2013, however scheduled data refreshes will not work. If you review the refresh history you will see an error message similar to the following:

"The workbook contains an unsupported Power Pivot model. The Power Pivot model in the workbook is in the SQL Server 2008 R2 Power Pivot for Excel 2010 format. Supported Power Pivot models are the following:

- SQL Server 2012 Power Pivot for Excel 2010.
- SQL Server 2012 Power Pivot for Excel 2013.

How to upgrade a workbook: The Scheduled data refresh will not work until you upgrade the workbook to a 2012 workbook. To upgrade the workbook and model it contains, complete one of the following:

- Download and open the workbook in Microsoft Excel 2010 with the SQL Server 2012 Power Pivot for Excel add-in installed.

  Open the Power Pivot window and upgrade the Power Pivot model.

  Then save the workbook and republish it to SharePoint.

- Download and open the workbook in Microsoft Excel 2013.

  Open the Power Pivot window and upgrade the Power Pivot model.

  Then save the workbook and republish it to the SharePoint server.

For more information on Changes to Analysis Services features, see Behavior Changes to Analysis Services Features in SQL Server 2016

For more information on refresh history, see View Data Refresh History (Power Pivot for SharePoint).

Upgrade to Office 2013 workbooks from Versions created by using the 2012 Power Pivot Add-In for Excel

This section describes Upgrading **to** SQL Server 2012 SP1 Power Pivot in Excel 2013 **from** SQL Server 2012 Power Pivot for Excel 2010 workbooks.

Upgrading a workbook resolves the following error that occurs when attempting scheduled data refresh on the previous workbook version workbook:

"Refresh operation for workbooks created with earlier version of Power Pivot is not available."

How to upgrade a workbook

1. Upgrade each workbook manually by opening it in Microsoft Excel 2013.
2. To upgrade the workbook and model it contains, download and open the workbook in Microsoft Excel 2013.
3. Open the Power Pivot window and upgrade the Power Pivot model.
4. Then save the workbook and republish it to the SharePoint 2013 server.

Upgrade to SQL Server 2012 workbooks from Versions created by using the 2008 R2 Power Pivot Add-In for Excel 2010

This section describes Upgrading **to** SQL Server 2012 Power Pivot for Excel 2010 **from** SQL Server 2008 R2 Power Pivot for Excel 2010 workbooks.

Upgrading a workbook resolves the following error that occurs when attempting scheduled data refresh on the previous workbook version workbook:

"Refresh operation for workbooks created with earlier version of Power Pivot is not available."

How to upgrade a workbook

There are two ways to upgrade:

1. Upgrade each workbook manually by opening it in Excel on a computer that has the SQL Server 2012 (11.x) version of Power Pivot for Excel, and then republish it to the server. When you open the workbook in the newer version of the add-in, the following internal operations occur: the data provider in the workbook data connection string is updated to MSOLAP.5, metadata is updated, and relationships are recreated to conform to a newer implementation.

2. Alternatively, a SharePoint Administrator can enable the auto-upgrade feature for the Power Pivot System Service in a SharePoint farm to automatically upgrade a SQL Server 2008 R2Power Pivot workbook when schedule data refresh runs (only workbooks that are configured for scheduled data refresh are upgraded).

> **NOTE**
> Automatic upgrade is a server configuration feature; you cannot enable or disable it for specific workbooks, libraries, or site collections.

How to configure automatic upgrade during data refresh

To use automatic upgrade, you must select the **Automatically upgrade Power Pivot workbooks to enable data refresh from the server** checkbox in the Power Pivot Configuration Tool. Within the tool, the checkbox is on the **Upgrade Power Pivot System Service** page, and on the **Create Power Pivot Service Application** page if you are configuring a new installation.

You can run the following cmdlet to verify whether automatic upgrade is enabled:

```
PS C:\Windows\system32> Get-PowerPivotSystemService
```

The output from Get-PowerPivotSystemService is a list of properties and corresponding values. You should see **WorkbookUpgradeOnDataRefresh** in the property list. It will be set to **true** if automatic upgrade is enabled. If it is **false**, continue to the next step, enabling automatic workbook upgrade.

To enable automatic workbook upgrade, run the following command:

```
PS C:\Windows\system32> Set-PowerPivotSystemService -WorkbookUpgradeOnDataRefresh:$true -Confirm:$false
```

After you upgrade the workbook, you can use scheduled data refresh and new features in the Power Pivot for Excel add-in.

Running Multiple Workbook Versions on a Newer Server

You can run older and newer versions of Power Pivot workbooks side by side on a SQL Server 2012 SP1 (11.0.3x) instance of Power Pivot for SharePoint.

Depending on how you installed the server, **you might need** to install a previous version of the Analysis Services OLE DB provider before you can access older and newer workbooks on the same server.

Note that Publishing newer version workbooks on previous SQL Server instances of Power Pivot for SharePoint is not supported. A SQL Server 2008 R2 instance will not load a workbook that you created in the SQL Server 2012 (11.x) version of Power Pivot for Excel, and a SQL Server 2012 instance will not load Office 2013 workbooks with advanced data models that you created using the SQL Server 2012 SP1 (11.0.3x) version of Power Pivot in Excel.

How to Check for MSOLAP Data Provider Information in a Power Pivot Workbook

Use the following instructions to check which OLE DB provider is used in a Power Pivot workbook. Checking the data connection information does not require the Power Pivot for Excel add-in to be installed.

1. In Excel, on the Data tab, click **Connections**. Click **Properties**.

2. On the **Definition** tab, the provider version appears at the beginning of the connection string.

   Provider=MSOLAP.5 indicates the workbook is SQL Server 2012 (11.x).

Provider=MSOLAP.4 indicates SQL Server 2008 R2.

Data Source=$Embedded$ indicates that the workbook is a Power Pivot workbook, using an embedded database.

How to Check for the Current Version of the MSOLAP Data Provider on a Local Computer

Use the following instructions to check which OLE DB provider is the current version on the server or workstation that runs Power Pivot workbooks. Knowing the current version can help you troubleshoot data connection errors after upgrading.

1. In the Registry Editor, go to HKEY_CLASSES_ROOT
2. Scroll down to MSOLAP. Verify that MSOLAP.5 is listed among the OLAP providers installed on the system. Verify that MSOLAP | CurVer is set to MSOLAP.5

See Also

Migrate Power Pivot to SharePoint 2013
Upgrade Power Pivot for SharePoint
What's New in Analysis Services
View Data Refresh History (Power Pivot for SharePoint)

Configure Analysis Services and Kerberos Constrained Delegation (KCD)

12/3/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Kerberos constrained delegation (KCD) is an authentication protocol you can configure with Windows authentication to delegate client credentials from service to service throughout your environment. KCD requires additional infrastructure, for example a Domain Controller, and additional configuration of your environment. KCD is a requirement in some scenarios that involve Analysis Services and Power Pivot data with SharePoint 2016. In SharePoint 2016, Excel Services has moved outside the SharePoint farm to a separate and new server, the **Office Online Server**. Because the Office Online Server is separate, there is an increased need for a way to delegate client credentials in the typical two hop scenarios.

Overview

KCD enables an account to impersonate another account for the purpose of providing access to resources. The impersonating account would be a service account assigned to a web application or the computer account of a web server while the impersonated account would be a user account requiring access to resources. KCD operates at the service level, so that selected services on a server can be granted access by the impersonating account, while other services on the same server, or services on other servers are denied for access.

The sections in this topic review common scenarios with Analysis Services and Power Pivot where KCD is required as well as an example server deployment with a high level summary of what you need to install and configure. See the More Information and community content section for links to more detailed information on the technologies involved such as Domain Controllers and KCD.

Scenario 1: Workbook as data source (WDS).

1 Office Online Server opens an Excel workbook and 2 detects a data connection to another workbook. Office Online Server sends a request to the Power Pivot Redirector Service 3 to open the second workbook and the data 4.

In this scenario, user credentials need to be delegated from the Office Online Server to the SharePoint Power Pivot Redirector Service in SharePoint.

Scenario 2: An Analysis Services Tabular model links to an Excel workbook

An Analysis Services Tabular model ① links to an Excel workbook which contains a Power Pivot model. In this scenario, when Analysis Services loads the Tabular model, Analysis Services detects the link to the workbook. When processing the model, Analysis Services sends a query request to SharePoint to load the workbook. In this scenario, client credentials do **not** need to be delegated from Analysis Services to SharePoint, however a client application can overwrite the data source information in an out-of-line binding. If the out-of-line binding request specifies to impersonate the current user, then the user credentials must be delegated, which requires KCD to be configured between Analysis Services and SharePoint.

Example deployment of KCD with Office Online Server and Analysis Services

This section describes an example deployment that uses four computers. The following sections summarize the key installation and configuration steps for each computer. Before you begin deployments, it is advised the computers are up to date with operating system patching and you know the computer names because they are needed in some of the configuration steps.

- Domain Controller
- SQL Server database engine and Analysis Services in Power Pivot mode. The instance of the database engine will be used for the SharePoint content databases.
- SharePoint server 2016
- Office Online Server

Domain Controller

The following is a summary of what to install for the domain controller (DC).

- **Role:** Active Directory Domain Services. For an overview, see Configuring Active Directory (AD DS) in Windows Server 2012.
- **Role:** DNS Server
- **Feature:** .NET Framework 3.5 Features / .NET Framework 3.5
- **Feature:** Remote Server Administration Tools / Role Administration Tools
- Configure Active Directory to create a new Forest and join the computers to the domain. Before trying to add other computers to the private domain, you will need to configure the client computers DNS to the DC's IP address. On the DC machine, run `ipconfig /all` to get the IPv4 and IPv6 addresses for the next step.
- It is recommended you configure both IPv4 and IPv6 addresses. You can do this in Windows control panel:
  1. Click **Network and Sharing Center**

2. Click your Ethernet connection
3. Click **Properties**
4. Click **Internet Protocol Version 6 (TCP/IPv6)**
5. Click **Properties**
6. Click **Use the following DNS server addresses**
7. Type the IP address from the ipconfig command.
8. Click the **Advanced** button , click the **DNS** tab and verify the DNS suffixes are correct.
9. Click **Append these DNS Suffixes.**
10. Repeat the steps for IPv4.

- **Note:** you can join computers to the domain from Windows Control panel, in the System settings. For more information, see How To Join Windows Server 2012 to a Domain.

2016 SQL Server Database engine and Analysis services in Power Pivot mode

The following is a summary of what to install on the SQL Server computer.

Tip In the SQL Server 2017 setup wizard, Analysis Services in Power Pivot mode is installed as part of the feature selection workflow.

1. Run the SQL Server 2017 setup wizard and from the feature selection page, click the database engine, Analysis Services, and the Management tools. In a later setup for the setup wizard you can specify the Power Pivot mode for Analysis Services.
2. For instance configuration, configure a named instance of "POWERPIVOT".
3. On the Analysis Services Configuration page, configure the Analysis Services server for **Power Pivot** mode and add the **computer name** of the Office Online Server to the list of Analysis Services server administrators. For more information, see Install Analysis Services in Power Pivot Mode.
4. Note, by default the "Computer" object type is not included in the search. Click Object Types... to add the Computers object.

5. Create the Service Principal Names (SPN) for the Analysis Services instance.

   The following are useful SPN commands:

   - List the SPN for a specific account name running the service of interest: `SetSPN -l <account-name>`
   - Set a SPN for an account name that is running the service of interest: `SetSPN -a <SPN> <account-name>`

- Delete a SPN from a specific account name running the service of interest: `SetSPN -D <SPN> <account-name>`
- Search for duplicate SPNs: `SetSPN -X`

The SPN for the PowerPivot instance will be in the form of:

```
MSSQLSvc.3/\<Fully Qualified Domain Name (FQDN)>:POWERPIVOT
MSSQLSvc.3/<NetBIOS Name>:POWERPIVOT
```

Where the FQDN and NetBIOS names are the name of the machine that the instance resides on. These SPNs will be placed on the Domain Account that is being used for the service account. If you are using Network Service, Local System, or the Service ID, you will want to place the SPN on the domain machine account. If you are using a domain user account, you will place the SPN on that account.

6. Create the SPN for the SQL Browser service on the Analysis Services machine.

   Learn more

7. **Configure constrained delegation** settings on the Analysis Services service account for any external source you will be refreshing from such as SQL Server, or Excel files. On the Analysis Services service account, we want to make sure the following are set.

   Note: If you do not see the delegation tab for the account, within Active Directory Users and Computers, it is because there is no SPN on that account. You can add a fake SPN to get it to appear such as `my/spn`.

   Trust this user for delegation to specified services only and **Use any authentication protocol**.

   This is known as constrained delegation, and is required because the windows token will originate from a Claims to Windows Token Services (C2WTS) which requires constrained delegation with protocol transitioning.

   You will also need to add the services that you will be delegating to. This will vary based on your environment.

Office Online Server

1. Install Office Online Server

2. **Configure Office Online Server** to connect to the Analysis Services server. Note, the Office Online Server computer account needs to be an administrator on the Analysis Services server. This was completed in a previous section of this topic, installing the Analysis Services server.

   a. On the Office Online Server, open a PowerShell window with administrative privileges and run the

following command

b. `New-OfficeWebAppsExcelBIServer -ServerId <AS instance name>`

c. Sample: `New-OfficeWebAppsExcelBIServer -ServerId "MTGQLSERVER-13\POWERPIVOT"`

3. **Configure Active Directory** to allow the Office Online Server computer account to impersonate users to the SharePoint service account. So, set delegation property on principal running the Application Pool for SharePoint Web Services, on the Office Online Server: The PowerShell commands in this section require the Active Directory (AD) PowerShell objects.

   a. Get the Active Directory identity of the Office Online Server

   ```
   $computer1 = Get-ADComputer -Identity [ComputerName]
   ```

   find this Principal name is by looking at Task Manager / Details / w3wp.exe's User name. For example "svcSharePoint"

   ```
   Set-ADUser svcSharePoint -PrincipalsAllowedToDelegateToAccount $computer1
   ```

   b. To verify the propery was set correctly

   ```
   Get-ADUser svcSharePoint -Properties PrincipalsAllowedToDelegateToAccount
   ```

4. **Configure constrained delegation** settings on the Office Online Server account to the Analysis Services Power Pivot instance. This should be the machine account that Office Online Server is running on. On the Office Online Service account, we want to make sure the following are set.

   Note: If you do not see the delegation tab for the account, within Active Directory Users and Computers, it is because there is no SPN on that account. You can add a fake SPN to get it to appear such as `my/spn`.

   Trust this user for delegation to specified services only and **Use any authentication protocol**.

   This is known as constrained delegation, and is required because the windows token will originate from a Claims to Windows Token Services (C2WTS) which requires constrained delegation with protocol transitioning. You will then want to allow delegation to the MSOLAPSvc.3 and MSOLAPDisco.3 SPNs that we created above.

5. Setup Claims to windows token service (C2WTS) **This is needed for scenario 1**. For more information see Claims to Windows Token Service (c2WTS) Overview.

6. **Configure constrained delegation** settings on the C2WTS service account. The settings should match what you did in step 4.

SharePoint Server 2016

The following is a summary of SharePoint Server installation.

1. Run SharePoint Pre-requisite installer

2. Run and SharePoint installation and select the **Single Server Farm** setup role.

3. Run the PowerPivot for SharePoint add-in (spPowerPivot16.msi). For more information, see Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2016)

4. Run the PowerPivot Configuration wizard. See Power Pivot Configuration Tools.
5. Connect SharePoint to the Office Online Server. (Configure_xlwac_on_SPO.ps1)
6. Configure SharePoint Authentication providers for Kerberos. **This is needed for scenario 1**. For more information, see Plan for Kerberos authentication in SharePoint 2013.

More Information and community content

Kerberos for the Busy Admin

Understanding Kerberos Double Hop

ALL things .Net and SharePoint

Resource Based Kerberos Constrained Delegation

KERBEROS PRIMER - videos

Microsoft® Kerberos Configuration Manager for SQL Server®

CheckList: Use PowerShell to Verify Power Pivot for SharePoint

1/28/2019 • 17 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

No Power Pivot for SharePoint installation or recovery operation is complete without a solid verification test pass that confirms your services and data are operational. In this article, we show you how to perform these steps using Windows PowerShell. We put each step into its own section so that you can go straight to specific tasks. For example, run the script in the Databases section of this topic to verify the name of the service application and content databases if you want to schedule them for maintenance or backup.

A full PowerShell script is included at the bottom of the topic. Use the full script as a starting point to build a custom script for auditing your full Power Pivot for SharePoint deployment.

Prepare your PowerShell environment

The steps in this section prepare your PowerShell environment. The steps may not be required, depending on how your scripting environment is currently configured.

PowerShell Permissions

Open a Powershell window or the PowerShell ISE (Integrated Scripting Environment) with **administrative privileges**. If you do not have administrative privileges when you run commands, you will see an error message similar to the following:

Get-SPLogEvent : You need to have computer **administrator privileges** to run this cmdlet.

SharePoint and Power Pivot for SharePoint Module

If you see an error message similar to the following when you run SharePoint related cmdlets, run the Add-PSSnapin command:

The term 'Get-PowerPivotSystemService' **is not recognized as the name of a cmdlet**, function, script file, or operable program. Check the spelling of the name, or if a path was included, verify that the path is correct and try again.

```
Add-PSSnapin Microsoft.Sharepoint.Powershell -EA 0
```

Windows PowerShell

For more information on the PowerShell ISE, see Introducing the Windows PowerShell ISE and Use Windows PowerShell to administer SharePoint 2013.

	You can optionally verify a majority of the components in Central Administration, using the Power Pivot management dashboard. To open the dashboard in Central Administration, click **General Application Settings**, and then click **Management Dashboard** in the **Power Pivot**. For more information on the dashboard, see Power Pivot Management Dashboard and Usage Data.

Symptoms and Recommended Actions

The following table is a list of symptoms or issues and the suggested section of this topic to consult to help you resolve the issue.

SYMPTOM	SEE SECTION
Data refresh is not running	See the section Timer Jobs and verify the **Online Power Pivot Data Refresh Timer Job** is online.
Management dashboard data is old	See the section Timer Jobs and verify the **Management Dashboard Processing Timer Job** is online.
Some portions of the Management Dashboard	If you install Power Pivot for SharePoint into a farm that has the topology of Central Administration, without Excel Services or Power Pivot for SharePoint, you must download and install the Microsoft ADOMD.NET client library if you want full access to the built-in reports in the Power Pivot management dashboard. Some reports in the dashboard use ADOMD.NET to access internal data that provides reporting data on Power Pivot query processing and server health in the farm. See the section ADOMD.Net client Library and the topic Install ADOMD.NET on Web Front-End Servers Running Central Administration.

Analysis Services Windows Service

The script in this section verifies the instance of Analysis Services in SharePoint mode. Verify the service is **running**.

```
get-service | select name, displayname, status | where {$_.Name -eq "msolap`$powerpivot"} | format-table -
property * -autosize | out-default
```

Example output

```
Name              DisplayName                                Status
----              -----------                                ------
MSOLAP$POWERPIVOT SQL Server Analysis Services (POWERPIVOT) Running
```

PowerPivotSystemService and PowerPivotEngineService

The scripts in this section verify the Power Pivot for SharePoint system services. There is one system service for a SharePoint 2013 deployment and two services for a SharePoint 2010 deployment.

PowerPivotSystemService

Verify the Status is **Online**.

```
Get-PowerPivotSystemService | select typename, status, applications, farm | format-table -property * -autosize
| out-default
```

Example output

```
TypeName                                Status Applications                          Farm
--------                                ------ ------------                          ----
SQL Server PowerPivot Service Application Online {Default PowerPivot Service Application} SPFarm
Name=SharePoint_Config_77d8ab0744a34e8aa27c806a2b8c760c
```

PowerPivotEngineService

> **NOTE**
>
> **Skip this script if** you are using SharePoint 2013. The PowerPivotEngineService is not part of a SharePoint 2013 deployment. If you run the Get-PowerPivotEngineService cmdlet on SharePoint 2013, you will see an error message similar to the following. This error message is returned even if you have run the Add-PSSnapin command described in the prerequisites section of this topic.
>
> The term 'Get-PowerPivotEngineService' is not recognized as the name of a cmdlet

In a SharePoint 2010 deployment, verify the status is **Online**.

```
Get-PowerPivotEngineService | select typename, status, name, instances, farm | format-table -property * -
autosize | out-default
```

Example output

```
TypeName  : SQL Server Analysis Services
Status    : Online
Name      : MSOLAP$POWERPIVOT
Instances : {POWERPIVOT}
Farm      : SPFarm Name=SharePoint_Config
```

Power Pivot Service Application(s) and proxies

Verify the status is **Online**. The Excel Services Application does not use a service application database and therefore the cmdlet does not return a database name. Note the database used by the Power Pivot service application so you can verify the database is online in the database section later in this topic.

Power Pivot and Excel Service Application(s)

For a SharePoint 2010 deployment, verify the status is **Online**.

```
Get-PowerPivotServiceApplication | select typename,name, status, unattendedaccount, applicationpool, farm,
database
Get-SPExcelServiceApplication | select typename, DisplayName, status
```

Example output

```
TypeName          : PowerPivot Service Application
Name              : PowerPivotServiceApplication1
Status            : Online
UnattendedAccount : PowerPivotUnattendedAccount
ApplicationPool   : SPIisWebServiceApplicationPool Name=sqlbi_serviceapp
Farm              : SPFarm Name=SharePoint_Config
Database          : GeminiServiceDatabase Name=PowerPivotServiceApplication1_19648f3f2c944e27acdc6c20aab8487a

TypeName    : Excel Services Application Web Service Application
DisplayName : Excel Services Application
Status      : Online
```

Service Application Pool

> **NOTE**
>
> The following code sample first returns the applicationpool property of the default Power Pivot for SharePoint service application. The name is parsed from the string and used to get the status of the application pool object.
>
> Verify the Status is **Online**. If the status is not Online or you see "http error" when you browse the Power Pivot site, verify the identity credentials in the IIS application pools are still correct. The IIS pool name will is the value of the ID property returned by the Get-SPServiceApplicationPool command.

```
$poolname=[string](Get-PowerPivotServiceApplication | select -property applicationpool)
$position=$poolname.lastindexof("=")
$poolname=$poolname.substring($position+1)
$poolname=$poolname.substring(0,$poolname.length-1)
Get-SPServiceApplicationPool | select name, status, processaccountname, id | where {$_.Name -eq $poolname} |
format-table -property * -autosize | out-default
```

Example output

```
Name                          Status ProcessAccountName Id
----                          ------ ------------------ -------
SharePoint Web Services System Online DOMAIN\account     89b50ec3-49e3-4de7-881a-2cec4b8b73ea
```

Tip

The application pool can also be verified on the Central Administration page **Manage Service Applications**. Click the name of the service application and then click **properties** in the ribbon.

Power Pivot and Excel Service Application proxies

Verify the Status is **Online**.

```
Get-SPServiceApplicationProxy |  select typename, status, unattendedaccount, displayname | where {$_.TypeName -
like "*powerpivot*" -or $_.TypeName -like "*excel services*"} | format-table -property * -autosize | out-
default
```

Example output

```
TypeName                                                Status UnattendedAccount           DisplayName
--------                                                ------ -----------------           -----------
PowerPivot Service Application Proxy                    Online PowerPivotUnattendedAccount
PowerPivotServiceApplication1
Excel Services Application Web Service Application Proxy Online                            Excel Services
Application
```

Databases

The following script returns the status of the service application databases and all content databases. Verify the status is **Online**.

```
Get-SPDatabase | select name, status, server, typename | where {$_.TypeName -eq "content database" -or
$_.TypeName -like "*Gemini*"} | format-table -property * -autosize | out-default
```

Example output

```
Name                                                                 Status Server
TypeName
----                                                                 ------ ------                 -----
---
DefaultPowerPivotServiceApplicationDB-38422181-2b68-4ab2-b2bb-9c00c39e5a5e Online SPServer Name=TESTSERVER
Microsoft.AnalysisServices.SPAddin.GeminiServiceDatabase
DefaultWebApplicationDB-f0db1a8e-4c22-408c-b9b9-153bd74b0312               Online TESTSERVER\POWERPIVOT
Content Database
SharePoint_Admin_3cadf0b098bf49e0bb15abd487f5c684                          Online TESTSERVER\POWERPIVOT
Content Database
```

SharePoint Features

Verify the site, web, and farm features are online.

```
Get-SPFeature | select displayname, status, scope, farm | where {$_.displayName -like "*powerpivot*"} | format-
table -property * -autosize | out-default
```

Example output

```
DisplayName     Status Scope Farm
-----------     ------ ----- ----
PowerPivotSite  Online  Site SPFarm Name=SharePoint_Config
PowerPivotAdmin Online   Web SPFarm Name=SharePoint_Config
PowerPivot      Online  Farm SPFarm Name=SharePoint_Config
```

Timer Jobs

Verify the Time Jobs are **Online**. The Power Pivot EngineService is not installed on SharePoint 2013, therefore the script will not list EngineService timer jobs in a SharePoint 2013 deployment.

```
Get-SPTimerJob | where {$_.service -like "*power*" -or $_.service -like "*mid*"} | select status, displayname,
LastRunTime, service | format-table -property * -autosize | out-default
```

Example output

```
      Status DisplayName                                                                    LastRunTime
Service
------ -----------                                                                  -----------
-------
Online Health Analysis Job (Daily, SQL Server Analysis Services, All Servers)              4/9/2014 12:00:01
AM EngineService Name=MSOLAP$POWERPIVOT
Online Health Analysis Job (Hourly, SQL Server Analysis Services, All Servers)             4/9/2014 1:00:01 PM
EngineService Name=MSOLAP$POWERPIVOT
Online Health Analysis Job (Weekly, SQL Server Analysis Services, All Servers)             4/6/2014 12:00:10
AM EngineService Name=MSOLAP$POWERPIVOT
Online PowerPivot Management Dashboard Processing Timer Job                                4/8/2014 3:45:38 AM
MidTierService
Online PowerPivot Health Statistics Collector Timer Job                                    4/9/2014 1:00:12 PM
MidTierService
Online PowerPivot Data Refresh Timer Job                                                   4/9/2014 1:09:36 PM
MidTierService
Online Health Analysis Job (Daily, SQL Server PowerPivot Service Application, All Servers)  4/9/2014 12:00:00
AM MidTierService
Online Health Analysis Job (Daily, SQL Server PowerPivot Service Application, Any Server)   4/9/2014 12:00:00
AM MidTierService
Online Health Analysis Job (Weekly, SQL Server PowerPivot Service Application, All Servers) 4/6/2014 12:00:03
AM MidTierService
Online Health Analysis Job (Weekly, SQL Server PowerPivot Service Application, Any Server)  4/6/2014 12:00:03
AM MidTierService
Online PowerPivot Setup Extension Timer Job                                                4/1/2014 1:40:31 AM
MidTierService
```

Health Rules

There are fewer rules in a SharePoint 2013 deployment. For a full list of rules for each SharePoint environment and an explanation of how to use the rules, see Configure Power Pivot Health Rules.

```
Get-SPHealthAnalysisRule | select name, enabled, summary | where {$_.summary -like "*power*"}  | format-table -
property * -autosize | out-default
```

Example output

```
Name                          Enabled Summary
----                          ------- -------
SecondaryLogonHealthRule         True PowerPivot:  Secondary Logon service (seclogon) is disabled
DataRefreshTimerJobHealthRule    True PowerPivot: The PowerPivot Data Refresh timer job is disabled.
ASUsageLoadHealthRule            True PowerPivot: The ratio of load events to connections is too high.
ASMiniDumpHealthRule             True PowerPivot: One or more minidump files were found in the Logs directory,
indicating a program crash
ASUsageCubeRule                  True PowerPivot: Usage data is not getting updated at the expected frequency.
ASADOMDNETHealthRule             True PowerPivot: ADOMD.NET is not installed on a standalone WFE that is
configured for central admin
MidTierAcctReadPermissionRule    True PowerPivot: MidTier process account should have 'Full Read' permission on
all associated SPWebApplications.
```

Windows and ULS Logs

Windows event log

The following command will search the windows event log for events related to the instance of Analysis Services in SharePoint mode. For information on disabling events or changing the event level, see Configure and View SharePoint Log Files and Diagnostic Logging (Power Pivot for SharePoint)

Service Name: MSOLAP$POWERPIVOT

Display name in Windows Services: SQL Server Analysis Services (POWERPIVOT)

```
Get-EventLog "application" | Where-Object {$_.source -like "msolap`$powerpivot*"}  |select timegenerated,
entrytype , source, message | format-table -property * -autosize | out-default
```

Example output

```
TimeGenerated           EntryType Source            Message
-------------           --------- ------            -------
4/16/2014 1:45:19 PM  Information MSOLAP$POWERPIVOT Software usage metrics are disabled.
4/16/2014 1:45:19 PM  Information MSOLAP$POWERPIVOT Service started. Microsoft SQL Server Analysis Services 64
Bit Evaluation (x64) RTM 12.0.1997.5.
4/16/2014 1:45:18 PM  Information MSOLAP$POWERPIVOT The flight recorder was started.
4/14/2014 6:45:37 PM  Information MSOLAP$POWERPIVOT Software usage metrics are disabled.
```

SharePoint ULS Log, last 48 hours

The following command will return Power Pivot messages from the ULS log that were created in the last 48 hours. Adjust the addhours parameter for your need.

```
Get-SPLogEvent -starttime(get-date).addhours(-48) | Where-Object {$_.Area -eq "powerpivot service" -and
$_.level -eq "high"} | select timestamp, area, category, eventid,level, message| format-table -property * -
autosize | out-default
```

The following variation of the command only returns log events for the **data refresh** category.

```
Get-SPLogEvent -starttime(get-date).addhours(-48) | Where-Object {$_.category -eq "data refresh" -and $_.level
-eq "high"} | select timestamp, area, category, eventid, level, correlation, message
```

Example output

```
Timestamp   : 4/14/2014 7:15:01 PM
Area        : PowerPivot Service
Category    : Data Refresh
EventID     : 43
Level       : High
Correlation : 5755879c-7cab-e097-8f80-f27895d44a77
Message     : The following error occured when working with the service application, Default PowerPivot Service
Application. Skipping the service application..

Timestamp   : 4/14/2014 7:15:02 PM
Area        : PowerPivot Service
Category    : Data Refresh
EventID     : 99
Level       : High
Correlation : 5755879c-7cab-e097-8f80-f27895d44a77
Message     : EXCEPTION: System.TimeoutException: The request channel timed out while waiting for a reply after
00:00:47.0625313. Increase the timeout value passed to
              the call to Request or increase the SendTimeout value on the Binding. The time allotted to this
operation may have been a portion of a longer timeout.
              ---> System.TimeoutException: The HTTP request to
'http://localhost:32843/SecurityTokenServiceApplication/securitytoken.svc/actas' has exceeded the
              allotted timeout of 00:00:54.5930000. The time allotted to this operation may have been a portion
of a longer timeout. ---> System.Net.WebException: The
              operation has timed out     at System.Net.HttpWebRequest.GetResponse()     at

System.ServiceModel.Channels.HttpChannelFactory`1.HttpRequestChannel.HttpChannelRequest.WaitForReply(TimeSpan
timeout...
```

MSOLAP Provider

Verify the provider MSOLAP provider. SQL Server 2012 (11.x) and SQL Server 2014 (12.x) Power Pivot require MSOLAP.5.

```
$excelApp=Get-SPExcelServiceApplication
get-spexceldataprovider -ExcelServiceApplication $excelApp |select providerid,providertype,description | where
{$_.providerid -like "msolap*" } | format-table -property * -autosize | out-default
```

Example output

```
ProviderId ProviderType Description
---------- ------------ -----------
MSOLAP     Oledb        Microsoft OLE DB Provider for OLAP Services
MSOLAP.3   Oledb        Microsoft OLE DB Provider for OLAP Services 9.0
MSOLAP.4   Oledb        Microsoft OLE DB Provider for OLAP Services 10.0
MSOLAP.5   Oledb        Microsoft OLE DB Provider for OLAP Services 11.0
```

For more information, see Install the Analysis Services OLE DB Provider on SharePoint Servers and Add MSOLAP.5 as a Trusted Data Provider in Excel Services.

ADOMD.Net client Library

```
get-wmiobject -class win32_product | Where-Object {$_.name -like "*ado*"} | select name, version, vendor |
format-table -property * -autosize | out-default
```

Example output

```
name                                                  version      vendor
----                                                  -------      ------
Microsoft SQL Server 2008 Analysis Services ADOMD.NET 10.1.2531.0  Microsoft Corporation
Microsoft SQL Server 2005 Analysis Services ADOMD.NET 9.00.1399.06 Microsoft Corporation
```

For more information, see Install ADOMD.NET on Web Front-End Servers Running Central Administration.

Health Data Collection Rules

Verify the **Status** is Online and **Enabled** is True.

```
get-spusagedefinition | select name, status, enabled, tablename, DaysToKeepDetailedData | where {$_.name -like
"powerpivot*"} | format-table -property * -autosize | out-default
```

Example output

```
Name                          Status Enabled TableName                   DaysToKeepDetailedData
----                          ------ ------- ---------                   ----------------------
PowerPivot Connections        OnlineTrue AnalysisServicesConnections  14
PowerPivot Load Data Usage    Online    True AnalysisServicesLoads                           14
PowerPivot Query Usage        Online    True AnalysisServicesRequests                        14
PowerPivot Unload Data Usage Online    True AnalysisServicesUnloads                         14
```

For more information, see Power Pivot Usage Data Collection.

Solutions

If the other components are online then you can skip verifying the solutions. If however the Health rules are missing, verify the two solutions exist and showed Verify the two Power Pivot solutions are **Online** and **Deployed**.

```
get-spsolution | select name, status, deployed, DeploymentState, DeployedServers | where {$_.Name -like
"*powerpivot*"} | format-table -property * -autosize | out-default
```

Example output SharePoint 2013

```
Name                                 Status Deployed        DeploymentState DeployedServers
----                                 ------ --------        --------------- ---------------
powerpivotfarm14solution.wsp         Online     True         GlobalDeployed {UETESTA00}
powerpivotfarmsolution.wsp           Online     True         GlobalDeployed {UETESTA00}
powerpivotwebapplicationsolution.wsp Online     True WebApplicationDeployed {UETESTA00}
```

Example output SharePoint 2010

```
Name                 Status Deployed        DeploymentState DeployedServers
----                 ------ --------        --------------- ---------------
powerpivotfarm.wsp   Online     True         GlobalDeployed {uesql11spoint2}
powerpivotwebapp.wsp Online     True WebApplicationDeployed {uesql11spoint2}
```

For more information on how to deploy SharePoint solutions, see Deploy solution packages (SharePoint Server 2010).

Manual Verification Steps

This section describes verification steps that cannot be completed with PowerShell cmdlets.

Scheduled Data Refresh: Configure the refresh schedule a workbook to **Also refresh as soon as possible**. For more information, see the "Verify Data Refresh" section of Schedule Data Refresh and Data Sources That Do Not Support Windows Authentication (Power Pivot for SharePoint).

More Resources

Web Server (IIS) Administration Cmdlets in Windows PowerShell.

PowerShell to check services, IIS sites and Application Pool status in SharePoint.

Windows PowerShell for SharePoint 2013 reference

Windows PowerShell for SharePoint Foundation 2010 reference

Manage Excel Services with Windows PowerShell (SharePoint Server 2010)

View and Read SQL Server Setup Log Files

Use the Get-EvenLog cmdlet

Full PowerShell Script

The Following script contains all of the commands from the previous sections. The script runs the commands in the same order as they are presented in this topic. The script contains some optional variations of the commands noted in this topic in case you need additional filtering. The variations are disabled with a comment character (#). The script also includes some statements for verifying Reporting Services SharePoint mode. The Reporting Services statements are disabled with a comment character (#).

```
# This script audits services related to PowerPivot for SharePoint
$starttime=Get-Date
write-host -foregroundcolor DarkGray StartTime $starttime

Write-Host  "Import the SharePoint PowerShell snappin"
Add-PSSnapin Microsoft.Sharepoint.Powershell -EA 0

#Write-Host ""
Write-Host -ForegroundColor Green "Analysis Services Windows Service"
Write-Host -ForegroundColor Green ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
get-service | select name, displayname, status | where {$_.Name -eq "msolap`$powerpivot"} | format-table -
property * -autosize | out-default

#Write-Host ""
Write-Host -ForegroundColor Green "PowerPivotEngineService and PowerPivotSystemService"
Write-Host -ForegroundColor Green ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"

Get-PowerPivotSystemService | select typename, status, applications, farm | format-table -property * -autosize
| out-default
# If needed, you can run the following to compare job definitions specific to the service against the results
of the timer job definition section
#Get-PowerPivotSystemService | select -ExpandProperty jobdefinitions | select displayname, schedule, service |
format-table -property * -autosize | out-default

Get-PowerPivotEngineService | select typename, status, name, instances, farm | format-table -property * -
autosize | out-default
# If needed, you can run the following to compare job definitions specific to the service against the results
of the timer job definition section
#Get-PowerPivotEngineService | select -ExpandProperty jobdefinitions | select displayname, schedule, service |
format-table -property * -autosize | out-default

#Write-Host ""
#Write-Host -ForegroundColor Green "Service Instances - optional if you want to associate services with the
server"
#Write-Host -ForegroundColor Green ">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>"
#Get-SPServiceInstance | select typename, status, server, service, instance | where {$_.TypeName -like
```

```
#Get-SPServiceInstance | select typename, status, server, service, instance | where {$_.TypeName -like
"*powerpivot*" -or $_.TypeName -like "*excel*" -or $_.TypeName -like "*Analysis Services*"} | format-table -
property * -autosize | out-default
#Get-PowerPivotEngineServiceInstance  | select typename, ASServername, status, server, service, instance
#Get-PowerPivotSystemServiceInstance  | select typename, ASSServerName, status, server, service, instance

#Write-Host ""
Write-Host -ForegroundColor Green "PowerPivot And Excel Service Applications"
Write-Host -ForegroundColor Green ">>>>
```

```
Write-Host -ForegroundColor Green "Windows Event Log data MSSQL$POWERPIVOT and "
Write-Host -ForegroundColor Green ">>>>
```

Deploying SQL Server 2016 PowerPivot and Power View in SharePoint 2016

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Summary: This white paper provides SharePoint administrators and architects detailed step-by-step instructions for deploying and configuring a Microsoft BI demo environment based on the Preview releases of SharePoint Server 2016, Office Online Server, and the SQL Server 2016 BI stack for SharePoint 2016. Following a brief introduction of important architecture changes and corresponding system dependencies, it outlines software and configuration requirements and a recommended deployment path to enable and verify BI capabilities in three main stages. This white paper also addresses known issues that exist in the SharePoint Server 2016 Beta 2, Office Online Server Preview and SQL Server 2016 CTP 3.1 releases and suggests appropriate workarounds. These workarounds will no longer be necessary in the final versions of the products. Check for an updated version of this white paper when deploying RTM releases.

Writer: Kay Unkroth

Technical Reviewers: Adam Saxton, Anne Zorner, Craig Guyer, Frank Weigel, Gregory Appel, Heidi Steen, Jason Haak, Kasper de Jonge, Kirk Stark, Klaus Sobel, Mike Plumley, Mike Taghizadeh, Patrick Wheeler, Riccardo Muti, Steve Hord

Published: December, 2015

Applies to: SQL Server 2016 CTP3.1, SharePoint 2016 Preview, Office Online Server Preview

To review the document, please download the Deploying SQL Server 2016 PowerPivot and Power View in SharePoint 2016 Word document.

Deploy PowerPivot and Power View - Multi-Tier SharePoint 2016 Farm

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Summary: Summary: This white paper provides SharePoint administrators and architects detailed step-by-step instructions for deploying and configuring a Microsoft BI demo environment in a SharePoint farm with multiple servers, based on the Preview releases of SharePoint Server 2016, Office Online Server, and the SQL Server 2016 BI stack for SharePoint 2016. Following a brief introduction of important architecture changes and corresponding system dependencies, it outlines software and configuration requirements and a recommended deployment path to enable and verify BI capabilities in three main stages. This white paper also addresses known issues that exist in the SharePoint Server 2016 Beta 2, Office Online Server Preview and SQL Server 2016 CTP 3.1 releases and suggests appropriate workarounds. These workarounds will no longer be necessary in the final versions of the products. Check for an updated version of this white paper when deploying RTM releases.

Writer: Kay Unkroth, Jason Haak

Technical Reviewers: Adam Saxton, Anne Zorner, Craig Guyer, Frank Weigel, Gregory Appel, Heidi Steen, Kasper de Jonge, Kirk Stark, Klaus Sobel, Mike Plumley, Mike Taghizadeh, Patrick Wheeler, Riccardo Muti, Steve Hord

Published: January, 2016

Applies to: SQL Server 2016 CTP3.1, SharePoint 2016 Preview, Office Online Server Preview

To review the document, please download the Deploying SQL Server 2016 PowerPivot and Power View in a Multi-Tier SharePoint 2016 Farm Word document.

Client libraries (data providers) used for Analysis Services connections

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services provides three client libraries, also known as **data providers**, for server and data access from tools and client applications. Tools like SSMS and SSDT, and applications like Power BI Desktop and Excel connect to Analysis Services by using these libraries. Two of the client libraries, ADOMD.NET and Analysis Services Management Objects (AMO), are managed client libraries. The Analysis Services OLE DB provider (MSOLAP DLL) is a native client library. Client libraries are the same for both SQL Server Analysis Services and Azure Analysis Services.

Where to get newer versions

The version installed on a client computer should match the major version of the server providing the data. If the server installation is newer than the data providers installed on the workstations in your network, you might need to install newer libraries.

Client libraries included in earlier SQL Server Feature Packs correspond to that SQL version; however, they may not be the latest. Connecting to Azure Analysis Services may require later versions. All versions are backward compatible.

To get the latest, see Client libraries for connecting to Azure Analysis Services.

See also

Connect to Analysis Services

Verify Analysis Services cumulative update build version

7/12/2018 • 2 minutes to read • Edit Online

Beginning with SQL Server 2017, the Analysis Services build version number and SQL Server Database Engine build version number do not match. While both Analysis Services and the Database Engine use the same installer, the build systems each use are separate.

In some cases, it may be necessary to verify if a Cumulative Update (CU) build package has been applied and Analysis Services components have been updated. You can verify by comparing the build version numbers of Analysis Services component files installed on your computer with the build version numbers for a particular CU.

Verify component file version

To verify component file version,

1. Go to SQL Server 2017 build versions.
2. In **SQL Server 2017 cumulative update (CU) builds**, click the **Knowledge Base Number** for the build you want to verify.
3. In the **Cumulative Update (#) for SQL Server 2017** article, in the **Cumulative Update package information** section, expand **Cumulative update package file information**.
4. In the **SQL Server 2017 Analysis Services** table, check the File version for the **msmdsrv.exe** component file. If the CU has been applied, the file version number should match the msmdsrv.exe file installed on your computer.

See also

Install SQL Server Servicing Updates

Update Center for Microsoft SQL Server

Post-install Configuration (Analysis Services)

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After installing Analysis Services, further configuration is required to make the server fully operational and available for general use. This section introduces the additional tasks that complete the installation. Depending on connection requirements, you might also need to configure authentication (see Connect to Analysis Services).

Later, additional work will be required once you have databases that are ready to deploy. Namely, you will need to configure role memberships on the database to grant user access to the data, design a database backup and recovery strategy, and determine whether you need a scheduled processing workload to refresh data at regular intervals. More information about database deployment and administration can be found at these links: Multidimensional Model Databases and Tabular Model Databases.

Instance Configuration

Analysis Services is a replicable service, meaning you can install multiple instances of the service on a single server. Each additional instance is installed separately as a named instance, using SQL Server Setup, and configured independently to support its intended purpose. For example, a development server might run Flight Recorder or use default values for data storage that you might otherwise change on servers supporting production workloads. Another example that calls for adjusting system configuration is installing Analysis Services instance on hardware shared by other services. When hosting multiple data-intensive applications on the same hardware, you might want to configure server properties that lower the memory thresholds to optimize available resources across all of the applications.

Post-installation Tasks

LINK	TASK DESCRIPTION
Configure the Windows Firewall to Allow Analysis Services Access	Create an inbound rule in Windows Firewall so that requests can be routed through the TCP port used by the Analysis Services instance. This task is required. No one can access Analysis Services from a remote computer until an inbound firewall rule is defined.
Grant server admin rights to an Analysis Services instance	During installation, you had to add at least one user account to the Administrator role of the Analysis Services instance. Administrative permissions are required for many routine server operations, such as processing data from external relational databases. Use the information in this topic to add or modify the membership of the Administrator role.
Configure antivirus software on computers running SQL Server	You might need to configure scanning software, such as antivirus and antispyware applications, to exclude SQL Server folders and file types. If scanning software locks a program or data file when Analysis Services needs to use it, service disruption or data corruption can occur.

LINK	TASK DESCRIPTION
Configure Service Accounts (Analysis Services)	During installation, the Analysis Services service account was provisioned, with appropriate permissions to allow controlled access to program executables and database files. As a post-installation task, you should now consider whether to allow the use of the service account when performing additional tasks. Both processing and query workloads can be executed under the service account. These operations succeed only when the service account has appropriate permissions.
Register an Analysis Services Instance in a Server Group	SQL Server Management Studio (SSMS) lets you create server groups for organizing your SQL Server instances. Scalable deployments consisting of multiple server instances are easier to manage in server groups. Use the information in this topic to organize Analysis Services instances into groups in SSMS.
Determine the Server Mode of an Analysis Services Instance	During installation, you chose a server mode that determines the type of model (multidimensional or tabular) that runs on the server. If you are unsure of the server mode, use the information in this topic to determine which mode was installed.
Rename an Analysis Services Instance	A descriptive name can help you distinguish among multiple instances having different server modes, or among instances primarily used by departments or teams in your organization. If you want to change the instance name to one that helps you better manage your installations, use the information in this topic to learn how.

Next Steps

Learn how to connect to Analysis Services from Microsoft applications or custom applications using the client libraries. Depending on your solution requirements, you might also need to configure the service for Kerberos authentication. Connections that must cross domain boundaries will require HTTP access. See Connect to Analysis Services for instructions about the next steps.

See Also

Installation for SQL Server 2016
Install Analysis Services in Multidimensional and Data Mining Mode
Install Analysis Services
Install Analysis Services in Power Pivot Mode

Configure the Windows Firewall to Allow Analysis Services Access

12/10/2018 • 15 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

An essential first step in making Analysis Services or Power Pivot for SharePoint available on the network is to determine whether you need to unblock ports in a firewall. Most installations will require that you create at least one in-bound firewall rule that allows connections to Analysis Services.

Firewall configuration requirements vary depending on how you installed Analysis Services:

- Open TCP port 2383 when installing a default instance or creating an Analysis Services failover cluster.
- Open TCP port 2382 when installing a named instance. Named instances use dynamic port assignments. As the discovery service for Analysis Services, SQL Server Browser service listens on TCP port 2382 and redirects the connection request to the port currently used by Analysis Services.
- Open TCP port 2382 when installing Analysis Services in SharePoint mode to support Power Pivot for SharePoint 2013. In Power Pivot for SharePoint 2013, the Analysis Services instance is external to SharePoint. Inbound requests to the named 'Power Pivot' instance originate from SharePoint web applications over a network connection, requiring an open port. As with other Analysis Services named instances, create an inbound rule for SQL Server Browser service on TCP 2382 to allow access to Power Pivot for SharePoint.
- For Power Pivot for SharePoint 2010, do not open ports in Windows Firewall. As an add-in to SharePoint, the service uses ports configured for SharePoint and makes only local connections to the Analysis Services instance that loads and queries Power Pivot data models.
- For Analysis Services instances running on Windows Azure Virtual Machines, use alternate instructions for configuring server access. See SQL Server Business Intelligence in Windows Azure Virtual Machines.

Although the default instance of Analysis Services listens on TCP port 2383, you can configure the server to listen on a different fixed port, connecting to the server in this format: <servername>:<portnumber>.

Only one TCP port can be used by an Analysis Services instance. On computers having multiple network cards or multiple IP addresses, Analysis Services listens on one TCP port for all IP addresses assigned or aliased to the computer. If you have specific multi-port requirements, consider configuring Analysis Services for HTTP access. You can then set up multiple HTTP endpoints on whatever ports you choose. See Configure HTTP Access to Analysis Services on Internet Information Services (IIS) 8.0.

This topic contains the following sections:

- Check port and firewall settings for Analysis Services
- Configure Windows Firewall for a default instance of Analysis Services
- Configure Windows Firewall access for a named instance of Analysis Services
- Port configuration for an Analysis Services cluster
- Port configuration for Power Pivot for SharePoint
- Use a fixed port for a default or named instance of Analysis Services

For more information about the default Windows firewall settings, and a description of the TCP ports that affect the Database Engine, Analysis Services, Reporting Services, and Integration Services, see Configure the Windows Firewall to Allow SQL Server Access.

Check port and firewall settings for Analysis Services

On the Microsoft Windows operating systems that are supported by SQL Server 2017, Windows Firewall is on by default and is blocking remote connections. You must manually open a port in the firewall to allow inbound requests to Analysis Services. SQL Server Setup does not perform this step for you.

Port settings are specified in the msmdsrv.ini file and in the General properties page of an Analysis Services instance in SQL Server Management Studio. If **Port** is set to a positive integer, the service is listening on a fixed port. If **Port** is set to 0, the service is listening on port 2383 if it is the default instance or on a dynamically assigned port if it is a named instance.

Dynamic port assignments are only used by named instances. The **MSOLAP$InstanceName** service determines which port to use when it starts up. You can determine the actual port number in use by a named instance by doing the following:

- Start Task Manager and then click **Services** to get the PID of the **MSOLAP$InstanceName**.
- Run **netstat -ao -p TCP** from the command line to view the TCP port information for that PID.
- Verify the port by using SQL Server Management Studio and connect to an Analysis Services server in this format: <IPAddress>:<portnumber>.

Although an application might be listening on a specific port, connections will not succeed if a firewall is blocking access. In order for connections to reach a named Analysis Services instance, you must unblock access to either msmdsrv.exe or the fixed port on which it is listening in the firewall. The remaining sections in this topic provide instructions for doing so.

To check whether firewall settings are already defined for Analysis Services, use Windows Firewall with Advanced Security in Control Panel. The Firewall page in the Monitoring folder shows a complete list of the rules defined for the local server.

Note that for Analysis Services, all firewall rules must be manually defined. Although Analysis Services and SQL Server Browser reserve ports 2382 and 2383, neither the SQL Server setup program nor any of the configuration tools define firewall rules that allow access to either the ports or the program executable files.

Configure Windows Firewall for a default instance of Analysis Services

The default instance of Analysis Services listens on TCP port 2383. If you installed the default instance and want to use this port, you only need to unblock inbound access to TCP port 2383 in Windows Firewall to enable remote access to the default instance of Analysis Services. If you installed the default instance but want to configure the service to listen on a fixed port, see Use a fixed port for a default or named instance of Analysis Services in this topic.

To verify whether the service is running as the default instance (MSSQLServerOLAPService), check the service name in SQL Server Configuration Manager. A default instance of Analysis Services is always listed as **SQL Server Analysis Services (MSSQLSERVER)**.

> **NOTE**
>
> Different Windows operating systems provide alternative tools for configuring Windows Firewall. Most of these tools let you choose between opening a specific port or program executable. Unless you have a reason for specifying the program executable, we recommend that you specify the port.

When specifying an inbound rule, be sure to adopt a naming convention that allows you to easily find the rules later (for example, **SQL Server Analysis Services (TCP-in) 2383**).

Windows Firewall with Advanced Security

1. On Windows 7 or Windows Vista, in Control Panel, click **System and Security**, select **Windows Firewall**, and then click **Advanced settings**. On Windows Server 2008 or 2008 R2, open Administrator Tools and click **Windows Firewall with Advanced Security**. On Windows Server 2012, open the Applications page and type **Windows Firewall**.
2. Right-click **Inbound Rules** and select **New Rule**.
3. In Rule Type, click **Port** and then click **Next**.
4. In Protocol and Ports, select **TCP** and then type **2383** in **Specific local ports**.
5. In Action, click **Allow the connection** and then click **Next**.
6. In Profile, clear any network locations that do not apply and then click **Next**.
7. In Name, type a descriptive name for this rule (for example, **SQL Server Analysis Services (tcp-in) 2383**), and then click **Finish**.
8. To verify that remote connections are enabled, open SQL Server Management Studio or Excel on a different computer and connect to Analysis Services by specifying the network name of the server in **Server name**.

> **NOTE**
> Other users will not have access to this server until you grant permissions. For more information, see Authorizing access to objects and operations (Analysis Services).

Netsh AdvFirewall Syntax

- The following command creates an inbound rule that allows incoming requests on TCP port 2383.

```
netsh advfirewall firewall add rule name="SQL Server Analysis Services inbound on TCP 2383" dir=in
action=allow protocol=TCP localport=2383 profile=domain
```

Configure Windows Firewall access for a named instance of Analysis Services

Named instances of Analysis Services can either listen on a fixed port or on a dynamically assigned port, where SQL Server Browser service provides the connection information that is current for the service at the time of the connection.

SQL Server Browser service listens on TCP port 2382. UDP is not used. TCP is the only transmission protocol used by Analysis Services.

Choose one of the following approaches to enable remote access to a named instance of Analysis Services:

- Use dynamic port assignments and SQL Server Browser service. Unblock the port used by SQL Server Browser service in Windows Firewall. Connect to the server in this format: <servername>\<instancename>.
- Use a fixed port and SQL Server Browser service together. This approach lets you connect using this format: <servername>\<instancename>, identical to the dynamic port assignment approach, except that in this case the server listens on a fixed port. In this scenario, SQL Server Browser Service provides name resolution to the Analysis Services instance listening on the fixed port. To use this approach, configure the

server to listen on a fixed port, unblock access to that port, and unblock access to the port used by SQL Server Browser service.

SQL Server Browser service is only used with named instances, never with the default instance. The service is automatically installed and enabled whenever you install any SQL Server feature as a named instance. If you choose an approach that requires SQL Server Browser service, be sure it remains enabled and started on your server.

If you cannot use SQL Server Browser service, you must assign a fixed port in the connection string, bypassing domain name resolution. Without SQL Server Browser service, all client connections must include the port number on the connection string (for example, AW-SRV01:54321).

Option 1: Use dynamic port assignments and unblock access to SQL Server Browser service

Dynamic port assignments for named instances of Analysis Services are established by the **MSOLAP$InstanceName** when the service starts. By default, the service claims the first available port number that it finds, using a different port number each time the service is restarted.

Instance name resolution is handled by the SQL Server browser service. Unblocking TCP port 2382 for SQL Server Browser service is always required if you are using dynamic port assignments with a named instance.

> **NOTE**
>
> SQL Server Browser service listens on both UDP port 1434 and TCP port 2382 for the Database Engine and Analysis Services, respectively. Even if you already unblocked UDP port 1434 for the SQL Server Browser service, you must still unblock TCP port 2382 for Analysis Services.

Windows Firewall with Advanced Security

1. On Windows 7 or Windows Vista, in Control Panel, click **System and Security**, select **Windows Firewall**, and then click **Advanced settings**. On Windows Server 2008 or 2008 R2, open Administrator Tools and click **Windows Firewall with Advanced Security**. On Windows Server 2012, open the Applications page and type **Windows Firewall**.
2. To unblock access to SQL Server Browser service, right-click **Inbound Rules** and select **New Rule**.
3. In Rule Type, click **Port** and then click **Next**.
4. In Protocol and Ports, select **TCP** and then type **2382** in **Specific local ports**.
5. In Action, click **Allow the connection** and then click **Next**.
6. In Profile, clear any network locations that do not apply and then click **Next**.
7. In Name, type a descriptive name for this rule (for example, **SQL Server Browser Service (tcp-in) 2382**), and then click **Finish**.
8. To verify that remote connections are enabled, open SQL Server Management Studio or Excel on a different computer and connect to the Analysis Services by specifying the network name of the server and the instance name in this format: <servername>\<instancename>. For example, on a server named **AW-SRV01** with a named instance of **Finance**, the server name is **AW-SRV01\Finance**.

Option 2: Use a fixed port for a named instance

Alternatively, you can assign a fixed port, and then unblock access to that port. This approach offers better auditing capability than if you allowed access to the program executable. For this reason, using a fixed port is the recommended approach for accessing any Analysis Services instance.

To assign a fixed port, follow the instructions in Use a fixed port for a default or named instance of Analysis

Services in this topic, then return to this section to unblock the port.

Windows Firewall with Advanced Security

1. On Windows 7 or Windows Vista, in Control Panel, click **System and Security**, select **Windows Firewall**, and then click **Advanced settings**. On Windows Server 2008 or 2008 R2, open Administrator Tools and click **Windows Firewall with Advanced Security**. On Windows Server 2012, open the Applications page and type **Windows Firewall**.
2. To unblock access to Analysis Services, right-click **Inbound Rules** and select **New Rule**.
3. In Rule Type, click **Port** and then click **Next**.
4. In Protocol and Ports, select **TCP** and then type the fixed port in **Specific local ports**.
5. In Action, click **Allow the connection** and then click **Next**.
6. In Profile, clear any network locations that do not apply and then click **Next**.
7. In Name, type a descriptive name for this rule (for example, **SQL Server Analysis Services on port 54321**), and then click **Finish**.
8. To verify that remote connections are enabled, open SQL Server Management Studio or Excel on a different computer and connect to the Analysis Services by specifying the network name of the server and the port number in this format: <servername>:<portnumber>.

Netsh AdvFirewall Syntax

- The following commands create inbound rules that unblock TCP 2382 for SQL Server Browser service and unblock the fixed port that you specified for the Analysis Services instance. You can run either one to allow access to a named Analysis Services instance.

  In this sample command, port 54321 is the fixed port. Be sure to replace it with the actual port in use on your system.

  ```
  netsh advfirewall firewall add rule name="SQL Server Analysis Services (tcp-in) on 54321" dir=in
  action=allow protocol=TCP localport=54321 profile=domain
  ```

  ```
  netsh advfirewall firewall add rule name="SQL Server Browser Services inbound on TCP 2382" dir=in
  action=allow protocol=TCP localport=2382 profile=domain
  ```

Use a fixed port for a default or named instance of Analysis Services

This section explains how to configure Analysis Services to listen on a fixed port. Using a fixed port is common if you installed Analysis Services as a named instance, but you can also use this approach if business or security requirements specify that you use non-default port assignments.

Note that using a fixed port will alter the connection syntax for the default instance by requiring you to append the port number to the server name. For example, connecting to a local, default Analysis Services instance listening on port 54321 in SQL Server Management Studio would require that you type localhost:54321 as the server name in the Connect to Server dialog box in Management Studio.

If you are using a named instance, you can assign a fixed port with no changes to how you specify the server name (specifically, you can use <servername\instancename> to connect to a named instance listening on a fixed port). This works only if SQL Server Browser service is running and you unblocked the port on which it is listening. SQL Server Browser service will provide redirection to the fixed port based on <servername\instancename>. As long as you open ports for both SQL Server Browser service and the named instance of Analysis Services listening on the fixed port, SQL Server Browser service will resolve the connection

to a named instance.

1. Determine an available TCP/IP port to use.

   To view a list of reserved and registered ports that you should avoid using, see Port Numbers (IANA). To view a list of ports that are already in use on your system, open a command prompt window and type **netstat -a -p TCP** to display a list of the TCP ports that are open on the system.

2. After you determine which port to use, specify the port by either editing the **Port** configuration setting in the msmdsrv.ini file or in the General properties page of an Analysis Services instance in SQL Server Management Studio.

3. Restart the service.

4. Configure Windows Firewall to unblock the TCP port you specified. Or, if you are using a fixed port for a named instance, unblock both the TCP port you specified for that instance and TCP port 2382 for SQL Server Browser service.

5. Verify by connecting locally (in Management Studio) and then remotely from a client application on another computer. To use Management Studio, connect to an Analysis Services default instance by specifying a server name in this format: <servername>:<portnumber>. For a named instance, specify the server name as <servername>\<instancename>.

Port configuration for an Analysis Services cluster

An Analysis Services failover cluster always listens on TCP port 2383, regardless of whether you installed it as a default instance or named instance. Dynamic port assignments are not used by Analysis Services when it is installed on a Windows failover cluster. Be sure to open TCP 2383 on every node running Analysis Services in the cluster. For more information about clustering Analysis Services, see How to Cluster SQL Server Analysis Services.

Port configuration for Power Pivot for SharePoint

Server architecture for Power Pivot for SharePoint is fundamentally different depending on which version of SharePoint you are using.

SharePoint 2013

In SharePoint 2013, Excel Services redirects requests for Power Pivot data models, which are subsequently loaded on an Analysis Services instance outside of the SharePoint environment. Connections follow the typical pattern, where an Analysis Services client library on a local computer sends a connection request to a remote Analysis Services instance in the same network.

Because Power Pivot for SharePoint always installs Analysis Services as a named instance, you should assume SQL Server Browser service and dynamic port assignments. As noted earlier, SQL Server Browser service listens on TCP port 2382 for connection requests sent to Analysis Services named instances, redirecting the request to the current port.

Note that Excel Services in SharePoint 2013 does not support the fixed port connection syntax, so make sure SQL Server Browser service is accessible.

SharePoint 2010

If you are using SharePoint 2010, you do not need to open ports in Windows Firewall. SharePoint opens the ports that it requires, and add-ins such as Power Pivot for SharePoint operate within the SharePoint environment. In a Power Pivot for SharePoint 2010 installation, the Power Pivot System Service has exclusive use of the local SQL Server Analysis Services (Power Pivot) service instance that is installed with it on the same computer. It uses local connections, not network connections, to access the local Analysis Services engine service

that loads, queries, and processes Power Pivot data on the SharePoint server. To request Power Pivot data from client applications, requests are routed through ports that are opened by SharePoint Setup (specifically, inbound rules are defined to allow access to SharePoint - 80, SharePoint Central Administration v4, SharePoint Web Services, and SPUserCodeV4). Because Power Pivot web services run within a SharePoint farm, the SharePoint firewall rules are sufficient for remote access to Power Pivot data in a SharePoint farm.

See Also

SQL Server Browser Service (Database Engine and SSAS)
Start, Stop, Pause, Resume, Restart the Database Engine, SQL Server Agent, or SQL Server Browser Service
Configure a Windows Firewall for Database Engine Access

Configure Service Accounts (Analysis Services)

9/28/2018 • 13 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Product-wide account provisioning is documented in Configure Windows Service Accounts and Permissions, a topic that provides comprehensive service account information for all SQL Server services, including Analysis Services. Refer to it to learn about valid account types, Windows privileges assigned by setup, file system permissions, registry permissions, and more.

This topic provides supplemental information for Analysis Services, including additional permissions necessary for tabular and clustered installations. It also covers permissions needed to support server operations. For example, you can configure processing and query operations to execute under the service account — in which case you will need to grant additional permissions to get this to work.

- Windows privileges assigned to Analysis Services
- File System Permissions assigned to Analysis Services
- Granting additional permissions for specific server operations

An additional configuration step, not documented here, is to register a Service Principal Name (SPN) for the Analysis Services instance and service account. This step enables pass-through authentication from client applications to backend data sources in double-hop scenarios. This step only applies for services configured for Kerberos constrained delegation. See Configure Analysis Services for Kerberos constrained delegation for further instructions.

Logon account recommendations

In a failover cluster, all instances of Analysis Services should be configured to use a Windows domain user account. Assign the same account to all instances. See How to Cluster Analysis Services for details.

Standalone instances should use the default virtual account, **NT Service\MSSQLServerOLAPService** for the default instance, or **NT Service\MSOLAP$***instance-name* for a named instance. This recommendation applies to Analysis Services instances in all server modes, assuming Windows Server 2008 R2 and later for the operating system, and SQL Server 2012 and later for Analysis Services.

Granting permissions to Analysis Services

This section explains the permissions that Analysis Services requires for local, internal operations, such as starting the executable, reading the configuration file, and loading databases from the data directory. If instead you're looking for guidance on setting permissions for external data access and interoperability with other services and applications, see Granting additional permissions for specific server operations further on in this topic.

For internal operations, the permission holder in Analysis Services is not the logon account, but a local Windows security group created by Setup that contains the per-service SID. Assigning permissions to the security group is consistent with previous versions of Analysis Services. Also, logon accounts can change over time, but the per-service SID and local security group are constant for the lifetime of the server installation. For Analysis Services, this makes the security group, rather than the logon account, a better choice for holding permissions. Whenever you manually grant rights to the service instance, whether file system permissions or Windows privileges, be sure to grant permissions to the local security group created for the server instance.

The name of the security group follows a pattern. The prefix is always **SQLServerMSASUser$**, followed by the

computer name, ending with the instance name. The default instance is **MSSQLSERVER**. A named instance is the name given during set up.

You can see this security group in the local security settings:

- Run compmgmt.msc | **Local Users and Groups** | **Groups** | **SQLServerMSASUser$**<server-name>**$MSSQLSERVER** (for a default instance).
- Double-click the security group to view its members.

The sole member of the group is the per-service SID. Right next to it is the logon account. The logon account name is cosmetic, there to provide context to the per-service SID. If you subsequently change the logon account and then return to this page, you'll notice that the security group and per-service SID do not change, but the logon account label is different.

Windows privileges assigned to the Analysis Services service account

Analysis Services needs permissions from the operating system for service startup and to request system resources. Requirements vary by server mode and whether the instance is clustered. If you are unfamiliar with Windows privileges, see Privileges and Privilege Constants (Windows) for details.

All instances of Analysis Services require the **Log on as a service** (SeServiceLogonRight) privilege. SQL Server Setup assigns the privilege for you on the service account specified during installation. For servers running in Multidimensional and Data Mining mode, this is the only Windows privilege required by the Analysis Services service account for standalone server installations, and it is the only privilege that Setup configures for Analysis Services. For clustered and tabular instances, additional Windows privileges must be added manually.

Failover cluster instances, in either Tabular or Multidimensional mode, must have **Increase scheduling priority** (SeIncreaseBasePriorityPrivilege).

Tabular instances use the following three additional privileges, which must be granted manually after the instance is installed.

Increase a process working set (SeIncreaseWorkingSetPrivilege)	This privilege is available to all users by default through the **Users** security group. If you lock down a server by removing privileges for this group, Analysis Services might fail to start, logging this error: "A required privilege is not held by the client." When this error occurs, restore the privilege to Analysis Services by granting it to the appropriate Analysis Services security group.
Adjust memory quotas for a process (SeIncreaseQuotaPrivilege)	This privilege is used to request more memory if a process has insufficient resources to complete its execution, subject to the memory thresholds established for the instance.

Lock pages in memory (SeLockMemoryPrivilege)	This privilege is needed only when paging is turned off entirely. By default, a tabular server instance uses the Windows paging file, but you can prevent it from using Windows paging by setting **VertiPaqPagingPolicy** to 0. **VertiPaqPagingPolicy** to 1 (default), instructs the tabular server instance to use the Windows paging file. Allocations are not locked, allowing Windows to page out as needed. Because paging is being used, there is no need to lock pages in memory. Thus, for the default configuration (where **VertiPaqPagingPolicy** = 1), you do not need to grant the **Lock pages in memory** privilege to a tabular instance. **VertiPaqPagingPolicy** to 0. If you turn off paging for Analysis Services, allocations are locked, assuming the **Lock pages in memory** privilege is granted to the tabular instance. Given this setting and the **Lock pages in memory** privilege, Windows cannot page out memory allocations made to Analysis Services when the system is under memory pressure. Analysis Services relies on the **Lock pages in memory** permission as the enforcement behind **VertiPaqPagingPolicy** = 0. Note that turning off Windows paging is not recommended. It will increase the rate of out-of-memory errors for operations that might otherwise succeed if paging were allowed. See Memory Properties for more information about **VertiPaqPagingPolicy**.

To view or add Windows privileges on the service account

1. Run GPEDIT.msc | Local Computer Policy | Computer Configuration | Windows Settings | Security Settings | Local Policies | User Rights Assignments.

2. Review existing policies that include **SQLServerMSASUser$**. This is a local security group found on computers having an Analysis Services installation. Both Windows privileges and file folder permissions are granted to this security group. Double-click **Log on as a service** policy to see how the security group is specified on your system. The full name of the security group will vary depending on whether you installed Analysis Services as a named instance. Use this security group, rather than the actual service account, when adding account privileges.

3. To add account privileges in GPEDIT, right-click **Increase a process working set** and select **Properties**.

4. Click **Add User or Group**.

5. Enter the user group for the Analysis Services instance. Remember that the service account is a member of a local security group, requiring that you prepend the local computer name as the domain of the account.

   The following list shows two examples for a default instance and named instance called "Tabular" on a machine called "SQL01-WIN12", where the machine name is the local domain.

   - SQL01-WIN12\SQL01-WIN12$SQLServerMSASUser$MSSQLSERVER
   - SQL01-WIN12\SQL01-WIN12$SQLServerMSASUser$TABULAR

6. Repeat for **Adjust memory quotas for a process**, and optionally, for **Lock pages in memory** or **Increase scheduling priority**.

> **NOTE**
> Previous versions of Setup inadvertently added the Analysis Services service account to the **Performance Log Users** group. Although this defect has been fixed, existing installations might have this unnecessary group membership. Because the Analysis Services service account does not require membership in the **Performance Log Users** group, you can remove it from the group.

File System Permissions assigned to the Analysis Services service account

> **NOTE**
> See Configure Windows Service Accounts and Permissions for a list of permissions associated with each program folder.
>
> See Configure HTTP Access to Analysis Services on Internet Information Services (IIS) 8.0 for file permission information related to IIS configuration and Analysis Services.

All file system permissions required for server operations— including permissions needed for loading and unloading databases from a designated data folder— are assigned by SQL Server Setup during installation.

The permission holder on data files, program file executables, configuration files, log files, and temporary files is a local security group created by SQL Server Setup.

There is one security group created for each instance that you install. The security group is named after the instance — either **SQLServerMSASUser$MSSQLSERVER** for the default instance, or **SQLServerMSASUser$**<servername>$<instancename> for a named instance. Setup provisions this security group with the file permissions required to perform server operations. If you check the security permissions on the \MSAS13.MSSQLSERVER\OLAP\BIN directory, you will see that the security group (not the service account or its per-service SID) is the permission holder on that directory.

The security group contains one member only: the per-service Security Identifier (SID) of the Analysis Services instance startup account. Setup adds the per-service SID to the local security group. The use of a local security group, with its SID membership, is a small but noticeable difference in how SQL Server Setup provisions Analysis Services, as compared to the Database Engine.

If you believe that file permissions are corrupted, follow these steps to verify the service is still correctly provisioned:

1. Use the Service Control command line tool (sc.exe) to obtain the SID of a default service instance.

   ```
   SC showsid MSSqlServerOlapService
   ```

   For a named instance (where the instance name is Tabular), use this syntax:

   ```
   SC showsid MSOlap$Tabular
   ```

2. Use **Computer Manager** | **Local Users and Groups** | **Groups** to inspect the membership of the SQLServerMSASUser$<servername>$<instancename> security group.

   The member SID should match the per-service SID from step 1.

3. Use **Windows Explorer** | **Program Files** | **Microsoft SQL Server** | MSASxx.MSSQLServer | **OLAP** | **bin** to verify folder Security properties are granted to the security group in step 2.

> **NOTE**
> Never remove or modify a SID. To restore a per-service SID that was inadvertently deleted, see http://support.microsoft.com/kb/2620201.

More about per-service SIDs

Every Windows account has an associated SID, but services can also have SIDs, hence referred to as per-service SIDs. A per-service SID is created when the service instance is installed, as a unique, permanent fixture of the service. The per-service SID is a local, machine-level SID generated from the service name. On a default instance, its user friendly name is NT SERVICE\MSSQLServerOLAPService.

The benefit of a per-service SID is that it allows the more widely-visible logon account to be changed arbitrarily, without affecting file permissions. For example, suppose you installed two instances of Analysis Services, a default instance and named instance, both running under the same Windows user account. While the logon account is shared, each service instance will have a unique per-service SID. This SID is distinct from the SID of the logon account. The per-service SID is used for file permissions and Windows privileges. In contrast, the logon account SID is used for authentication and authorization scenarios — different SIDS, used for different purposes.

Because the SID is immutable, file system ACLs created during service installation can be used indefinitely, regardless of how often you change the service account. As an added security measure, ACLs that specify permissions via a SID ensure that program executables and data folders are accessed only by a single instance of a service, even if other services run under the same account.

Granting additional Analysis Services permissions for specific server operations

Analysis Services executes some tasks in the security context of the service account (or logon account) that is used to start Analysis Services, and executes other tasks in the security context of the user who is requesting the task.

The following table describes additional permissions required to support tasks executing as the service account.

SERVER OPERATION	WORK ITEM	JUSTIFICATION
Remote access to external relational data sources	Create a database login for the service account	Processing refers to data retrieval from an external data source (usually a relational database), which is subsequently loaded into an Analysis Services database. One of the credential options for retrieving external data is to use the service account. This credential option works only if you create a database login for the service account and grant read permissions on the source database. See Set Impersonation Options (SSAS - Multidimensional) for more information about how the service account option is used for this task. Similarly, if ROLAP is used as the storage mode, the same impersonation options are available. In this case, the account must also have write access to the source data to process the ROLAP partitions (that is, to store aggregations).

SERVER OPERATION	WORK ITEM	JUSTIFICATION
DirectQuery	Create a database login for the service account	DirectQuery is a tabular feature used to query external datasets that are either too large to fit inside the tabular model or have other characteristics that make DirectQuery a better fit than the default in-memory storage option. One of the connection options available in DirectQuery mode is to use the service account. Once again, this option works only when the service account has a database login and read permissions on the target data source. See Set Impersonation Options (SSAS - Multidimensional) for more information about how the service account option is used for this task. Alternatively, the credentials of the current user can be used to retrieve data. In most cases this option entails a double-hop connection, so be sure to configure the service account for Kerberos constrained delegation so that the service account can delegate identities to a downstream server. For more information, see Configure Analysis Services for Kerberos constrained delegation.
Remote access to other SSAS instances	Add the service account to Analysis Services database roles defined on the remote server	Remote partitions and referencing linked objects on other remote Analysis Services instances are both system capabilities requiring permissions on a remote computer or device. When a person creates and populates remote partitions, or sets up a linked object, that operation runs in the security context of the current user. If you subsequently automate these operations, Analysis Services will access remote instances in the security context of its service account. In order to access linked objects on a remote instance of Analysis Services, the logon account must have permission to read the appropriate objects on the remote instance, such as Read access to certain dimensions. Similarly, using remote partitions requires that the service account have administrative rights on the remote instance. Such permissions are granted on the remote Analysis Services instance, using roles that associate permitted operations with a specific object. See Grant database permissions (Analysis Services) for instructions on how to grant Full Control permissions that allow processing and query operations. See Create and Manage a Remote Partition (Analysis Services) for more information about remote partitions.

SERVER OPERATION	WORK ITEM	JUSTIFICATION
Writeback	Add the service account to Analysis Services database roles defined on the remote server	When enabled in client applications, writeback is a feature of multidimensional models that allows the creation of new data values during data analysis. If writeback is enabled within any dimension or cube, the Analysis Services service account must have write permissions to the writeback table in the source SQL Server relational database. If this table does not already exist and needs to be created, the Analysis Services service account must also have create table permissions within the designated SQL Server database.
Write to a query log table in a SQL Server relational database	Create a database login for the service account and assign write permissions on the query log table	You can enable query logging to collect usage data in a database table for subsequent analysis. The Analysis Services service account must have write permissions to the query log table in the designated SQL Server database. If this table does not already exist and needs to be created, the Analysis Services logon account must also have create table permissions within the designated SQL Server database. For more information, see Improve SQL Server Analysis Services Performance with the Usage Based Optimization Wizard (Blog) and Query Logging in Analysis Services (Blog).

See Also

Configure Windows Service Accounts and Permissions
SQL Server Service Account and Per-Service SID (Blog)
SQL Server uses a service SID to provide service isolation (KB Article)
Access Token (MSDN)
Security Identifiers (MSDN)
Access Token (Wikipedia)
Access Control Lists (Wikipedia)

Grant server admin rights to an Analysis Services instance

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Members of the Server administrator role within an instance of Analysis Services have unrestricted access to all Analysis Services objects and data in that instance. A user must be a member of the Server administrator role to perform any server-wide task, such as creating or processing a database, modifying server properties, or launching a trace (other than for processing events).

Role membership is established when Analysis Services is installed. The user running the Setup program can add him or herself to the role, or add another user. You must specify at least one administrator before Setup will allow you to continue.

By default, members of the local Administrators group are also granted administrative rights in Analysis Server. Although the local group is not explicitly granted membership in the Analysis Services server administrator role, local administrators can create databases, add users and permissions, and perform any other task allowed to system administrators. The implicit granting of administrator permissions is configurable. It is determined by the **BuiltinAdminsAreServerAdmins** server property, which is set to **true** by default. You can change this property in SQL Server Management Studio. For more information, see Security Properties.

Post-installation, you can modify role membership to add any additional users who require full rights to the service. You can also manage server roles by using Analysis Management Objects (AMO). For more information, see Developing with Analysis Management Objects (AMO).

> **NOTE**
>
> Analysis Services provides a progression of increasingly granular roles for processing and querying at server, database, and object levels. See Roles and Permissions (Analysis Services) for instructions on how to use these roles.

Modify Server Role Membership

1. In SQL Server Management Studio, connect to the instance of Analysis Services, and then right-click the instance name in Object Explorer and then click **Properties**.
2. Click **Security** in the **Select a Page** pane, and then click **Add** at the bottom of the page to add one or more Windows users or groups to the server role.

Add computer accounts

You can also use SQL Server Management Studio to make a computer account a member of the Analysis Services administrators group.

1. In the **Select Users or Groups** dialog, click **Locations**.
2. Select the domain the computers that you want to add are a member of or select **Entire directory** and click **Ok**.
3. Click **Object Types**.
4. Select **Computers** and click **Ok**.

5. In the **Enter the object names to select** text box, type the name of the computer and click **Check Names** to verify the computer account is found in the current Locations. If the computer account is not found, verify the computer name and the correct domain the computer is a member of.

NT Service\SSASTelemetry account

NT Service/SSASTelemetry is a low-privileged machine account created during setup and used exclusively to run the Analysis Services implementation of the Customer Experience Improvement Program (CEIP) service. This service requires admin rights on the Analysis Services instance to run several discover commands. See Customer Experience Improvement Program for SQL Server Data Tools and Microsoft SQL Server Privacy

Statement for more information.

See Also

Authorizing access to objects and operations (Analysis Services)
Security Roles (Analysis Services - Multidimensional Data)

Features off by default (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

An instance of Analysis Services is designed to be secure by default. Therefore, features that might compromise security are disabled by default. The following features are installed in a disabled state and must specifically be enabled if you want to use them.

Feature List

To enable the following features, connect to Analysis Services using SQL Server Management Studio. Right-click the instance name and choose **Facets**. Alternatively, you can enable these features through server properties, as described in the next section.

- Ad Hoc Data Mining (OpenRowset) Queries
- Linked Objects (To)
- Linked Objects (From)
- Listen Only On Local Connections
- User Defined Functions

Server properties

Additional features that are off by default can be enabled through server properties. Connect to Analysis Services using SQL Server Management Studio. Right-click the instance name and choose **Properties**. Click **General**, and then click **Show Advanced** to display a larger property list.

- Ad Hoc Data Mining (OpenRowset) Queries
- Allow Session Mining Models (Data Mining)
- Linked Objects (To)
- Linked Objects (From)
- COM based user-defined functions
- Flight Recorder Trace Definitions (templates).
- Query logging
- Listen Only On Local Connections
- Binary XML
- Compression
- Group affinity. See Thread Pool Properties for details.

Register an Analysis Services Instance in a Server Group

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If you have a large number of Analysis Services server instances, you can create server groups in Management Studio to make server administration easier. The purpose of a server group is to provide proximity among a group of related servers within the administrative workspace. For example, suppose you are tasked with managing ten separate instances of Analysis Services. Grouping them by server mode, up-time criteria, or by department or region would allow you to view and connect to instances that share the same characteristics more easily. You can also add descriptive information that helps you remember how the server is used.

Server groups can be created in a hierarchical structure. Local Server Group is the root node. It always contains instances of Analysis Services that run on the local computer. You can add remote servers to any group, including the local group.

After you create a server group, you must use the Registered Servers pane to view and connect to the member servers. The pane filters SQL Server instances by server type (Database Engine, Analysis Services, Reporting Services, and Integration Services). You click a server type to view the server groups created for it. To connect to a specific server within group, you double-click a server in the group.

The connection information that is defined for the server, including the server name, is persisted with server registration. You cannot modify the connection information, or use the registered name when connecting to the server using other tools.

Create a Server Group and Add Registered Servers

1. In Management Studio, click Registered Servers on the View menu to open the Registered Servers pane in the workspace. By default, a local Server Group is already created. All instances of Analysis Services that are running on the local server are members.

2. Right-click Local Server Group, select New Server Group, and give the group a name.

3. Right-click the server group and select New Server Registration. Enter the network name of a local or remote server, including the instance name if the server was installed as a named instance. Optionally, you can provide a registered server name that appears in Registered Servers. This name is used in Registered Servers only. You cannot use it to rename a server, nor can you use it in a connection string. A registered server name can be more descriptive than the actual server name or include other identifying characteristics that help you distinguish this server from other servers.

Determine the Server Mode of an Analysis Services Instance

11/27/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services can be installed in one of three server modes: Multidimensional and Data Mining (default), Power Pivot for SharePoint, and Tabular. The server mode of an Analysis Services instance is determined during setup when you choose options for installing the server.

The server mode determines the type of solution that you create and deploy. If you did not install the server software and you want to know in which mode the server was installed, you can use the information in this topic to determine the mode. For more information about feature availability in a specific mode, see Comparing Tabular and Multidimensional Solutions .

If you do not want to use the server mode that you installed, you must uninstall and then reinstall the software, choosing the mode that you prefer. Alternatively, you can install an additional instance of Analysis Services on the same computer so that you have multiple instances running different modes.

Server Icons in Object Explorer

The easiest way to determine server mode is to connect to the server in SQL Server Management Studio and note the icon next to the server name in Object Explorer. The following illustration shows three instances of Analysis Services deployed in Multidimensional, Tabular, and Power Pivot modes:

Viewing DeploymentMode Property in MSMDSRV.INI File

Alternatively, you can check the **DeploymentMode** property in the msmdsrv.ini file that is included in every Analysis Services instance. The value of this property identifies the server mode. Valid values are 0 (Multidimensional), 1 (SharePoint), or 2 (Tabular). You must be an Analysis Services administrator (that is, a member of the Server role) to open the msmdsrv.ini file. This file contains structured XML. You can use Notepad or another text editor to view the file.

Caution

Do not change the value of the **DeploymentMode** property. Changing the property manually after the server is installed is not supported.

About the DeploymentMode Property

DeploymentMode property determines the operational context of an Analysis Services server instance. This property is referred to as 'server mode' in dialog boxes, messages, and documentation. This property is initialized by Setup based on how you install Analysis Services. This property should be considered internal only, always using the value specified by Setup.

Valid values for this property include the following:

VALUE	DESCRIPTION
0	This is the default value. It specifies multidimensional mode, used to service multidimensional databases that use MOLAP, HOLAP, and ROLAP storage, as well as data mining models.
1	Specifies Analysis Services instances that were installed as part of a Power Pivot for SharePoint deployment. Do not change the deployment mode property of Analysis Services instance that is part of a Power Pivot for SharePoint installation. Power Pivot data will no longer run on the server if you change the mode.
2	Specifies Tabular mode used for hosting tabular model databases that use in-memory storage or DirectQuery storage.

Each mode is exclusive of the other. A server that is configured for tabular mode cannot run Analysis Services databases that contain cubes and dimensions. If the underlying computer hardware can support it, you can install multiple instances of Analysis Services on the same computer and configure each instance to use a different deployment mode. Remember that Analysis Services is a resource intensive application. Deploying multiple instances on the same system is recommended only for high-end servers.

See Also

Install Analysis Services
Install Analysis Services in Multidimensional and Data Mining Mode
Power Pivot for SharePoint 2010 Installation
Connect to Analysis Services
Tabular Model Solutions
Multidimensional Model Solutions
Mining Models (Analysis Services - Data Mining)

Rename an Analysis Services Instance

11/27/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can rename an existing instance of Microsoft Analysis Services by using the **Rename Instance** tool, installed with Management Studio (Web install).

> **IMPORTANT**
>
> While renaming the instance, the Analysis Services Instance Rename tool runs under elevated privileges, updating the Windows service name, security accounts, and registry entries associated with that instance. To ensure that these actions are performed, be sure to run this tool as a local system administrator.

The Analysis Services Instance Rename tool does not modify the program folder that was created for the original instance. Do not modify the program folder name to match the instance you are renaming. Changing a program folder name can prevent Setup from repairing or uninstalling the installation.

> **NOTE**
>
> The Analysis Services Instance Rename tool is not supported for use in a cluster environment.

To rename an instance of Analysis Services

1. Launch the **Instance Rename** tool, **asinstancerename.exe**, from C:\Program Files (x86)\Microsoft SQL Server\130\Tools\Binn\ManagementStudio.
2. In the **Rename Instance** dialog box, in the **Instance to rename** list, select the instance that you want to rename.
3. In the **New instance name** box, enter the new name for the instance.
4. Verify that the user name and password are correct, and then click **Rename**.

   The Analysis Services instance will be stopped and restarted as part of the name change.

Post-rename checklist

1. To resume access to databases that are running on the renamed instance, you will need to manually update the data connections in Excel or other client applications. Also check any predefined connections, such as Reporting Services shared data sources, Excel ODC files, or BI Semantic Model connection files that might reference the instance you just renamed. For more information, see Connect to Analysis Services.
2. Update PowerShell scripts or AMO scripts that you routinely use to backup, synchronize, or process databases.
3. Update project properties for Analysis Services projects that you work with in SQL Server Data Tools. For tabular mode server instances, be sure to update the Workspace Server property on the model.bim file, as well as the Server property on the project.
4. Depending on how you specified the service account, you might need to update database logins or file permissions that grant data access rights to the service (for example, if you use the service account to process data or access linked objects on another server).

Updating a database login or file permissions will be necessary if you used a virtual account to provision the service. Virtual accounts are based on the instance name, so if you rename the instance, the virtual account is also updated at the same time. This means that any previous logins or permissions that you created for the previous instance are no longer valid.

The following example provides an illustration. Suppose you installed a tabular mode server as an instance named "Tabular" using the default virtual account, resulting in the following configuration:

a. Instance name = <server>\TABULAR

b. Service name = MSOLAP$TABULAR

c. Virtual account = NT Service\ MSOLAP$TABULAR

Now suppose you rename the instance to "TAB2". As a result of the name change, your configuration would now look like the following:

a. Instance name = <server>\TAB2

b. Service name = MSOLAP$TAB2

c. Virtual account = NT Service\ MSOLAP$TAB2

As you can see, database and file permissions that were previously granted to "NT Service\ MSOLAP$TABULAR" are no longer valid. To ensure that tasks and operations performed by the service run as before, you would now need to grant new database and file permissions to "NT Service\ MSOLAP$TAB2".

Tabular models

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Tabular models in Analysis Services are databases that run in-memory or in DirectQuery mode, connecting to data directly from back-end relational data sources. By using state-of-the-art compression algorithms and multi-threaded query processor, the Analysis Services Vertipaq analytics engine delivers fast access to tabular model objects and data by reporting client applications like Power BI and Excel.

While in-memory models are the default, DirectQuery is an alternative query mode for models that are either too large to fit in memory, or when data volatility precludes a reasonable processing strategy. DirectQuery achieves parity with in-memory models through support for a wide array of data sources, ability to handle calculated tables and columns in a DirectQuery model, row level security via DAX expressions that reach the back-end database, and query optimizations that result in faster throughput.

Tabular models are created in SQL Server Data Tools (SSDT) using the Tabular model project template. The project template provides a design surface for creating semantic model objects like tables, partitions, relationships, hierarchies, measures, and KPIs.

Tabular models can be deployed to Azure Analysis Services or an instance of SQL Server Analysis Services configured for Tabular server mode. Deployed tabular models can be managed in SQL Server Management Studio.

Tabular modeling documentation included here applies, in most cases, to both SQL Server Analysis Services and Azure Analysis Services. Articles specific to Azure Analysis Services are published along with other Azure documentation. To learn more, see Azure Analysis Services documentation.

Compatibility level for Analysis Services tabular models

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The *compatibility level* refers to release-specific behaviors in the Analysis Services engine. For example, DirectQuery and tabular object metadata have different implementations depending on the compatibility level. In-general, you should choose the latest compatibility level supported by your servers.

The latest compatibility level is 1400

Major features in the 1400 compatibility level include:

- New infrastructure for data connectivity and import into tabular models with support for TOM APIs and TMSL scripting. This enables support for additional data sources such as Azure Blob storage. Additional data sources will be included in future updates.
- Data transformation and data mashup capabilities by using Get Data and M expressions in SSDT.
- Measures now support a Detail Rows property with a DAX expression, enabling BI tools such as Microsoft Excel drill-down to detailed data from an aggregated report. For example, when end-users view total sales for a region and month, they can view the associated order details.
- Object-level security for table and column names, in addition to the data within them.
- Enhanced support for ragged hierarchies.
- Performance and monitoring improvements.

Supported compatibility levels by version

Compatibility level	Server version
1400	Azure Analysis Services, SQL Server 2017
1200	Azure Analysis Services, SQL Server 2017, SQL Server 2016
1103	SQL Server 2017*, SQL Server 2016, SQL Server 2014, SQL Server 2012 SP1
1100	SQL Server 2017*, SQL Server 2016, SQL Server 2014, SQL Server 2012 SP1, SQL Server 2012

* 1100 and 1103 compatibility levels are deprecated in SQL Server 2017.

Set compatibility level

When creating a new tabular model project in SQL Server Data Tools (SSDT), you can specify the compatibility level on the **Tabular model designer** dialog.

If you select the **Do not show this message again** option, all subsequent projects will use the compatibility level you specified as the default. You can change the default compatibility level in SSDT in **Tools** > **Options**.

To upgrade a tabular model project in SSDT, set the **Compatibility Level** property in the model **Properties** window. Keep in-mind, upgrading the compatibility level is irreversible.

Check compatibility level for a database in SSMS

In SSMS, right-click the database name > **Properties** > **Compatibility Level**.

Check supported compatibility level for a server in SSMS

In SSMS, right-click the server name> **Properties** > **Supported Compatibility Level**.

This property specifies the highest compatibility level of a database that will run on the server. The supported compatibility level is read-only cannot be changed.

See also

Compatibility Level of a multidimensional database
What's new in Analysis Services
Create a new tabular model project

Data sources supported in SQL Server Analysis Services tabular 1400 models

2/12/2019 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server 2017 Analysis Services Azure Analysis Services

This article describes the types of data sources that can be used with SQL Server Analysis Services (SSAS) tabular models at the 1400 compatibility level.

For SSAS tabular models at the 1200 and lower compatibility levels, see Data sources supported in SQL Server Analysis Services tabular 1200 models.

For Azure Analysis Services, see Data sources supported in Azure Analysis Services.

Cloud data sources

DATASOURCE	IN-MEMORY	DIRECTQUERY
Azure SQL Database	Yes	Yes
Azure SQL Data Warehouse	Yes	Yes
Azure Blob Storage	Yes	No
Azure Table Storage	Yes	No
Azure Cosmos DB	Yes	No
Azure Data Lake Store (Gen1)[1]	Yes	No
Azure HDInsight HDFS	Yes	No
Azure HDInsight Spark [2]	Yes	No

- ADLS Gen2 is currently not supported.
- Azure Databricks using the Spark connector is currently not supported.

Provider

In-memory and DirectQuery models connecting to Azure data sources use .NET Framework Data Provider for SQL Server.

On-premises data sources

Supported by in-memory and DirectQuery models

DATASOURCE	IN-MEMORY PROVIDER	DIRECTQUERY PROVIDER	

DATASOURCE	IN-MEMORY PROVIDER	DIRECTQUERY PROVIDER	
SQL Server	SQL Server Native Client 11.0, Microsoft OLE DB Provider for SQL Server, .NET Framework Data Provider for SQL Server	.NET Framework Data Provider for SQL Server	
SQL Server Data Warehouse	SQL Server Native Client 11.0, Microsoft OLE DB Provider for SQL Server, .NET Framework Data Provider for SQL Server	.NET Framework Data Provider for SQL Server	
Oracle	Microsoft OLE DB Provider for Oracle, Oracle Data Provider for .NET	Oracle Data Provider for .NET	
Teradata	OLE DB Provider for Teradata, Teradata Data Provider for .NET	Teradata Data Provider for .NET	

NOTE

For in-memory models, OLE DB providers can provide better performance for large-scale data. When choosing between different providers for the same data source, try the OLE DB provider first.

Supported by in-memory models only

DATABASE		
Access Database		
SQL Server Analysis Services		
IBM Informix (beta)		
JSON document		
Lines from binary		
MySQL Database		
PostgreSQL Database	Yes	No
SAP HANA	Yes	No
SAP Business Warehouse	Yes	No
Sybase Database	Yes	No

FILE	
Excel workbook	
Folder	
JSON	
Text/CSV	
XML table	

ONLINE SERVICES	
Dynamics 365	
Exhange Online	
Saleforce Objects	
Salesforce Reports	
SharePoint Online List	

OTHER	
Active Directory	
Exhange	
OData Feed	
ODBC query	
OLE DB	
SharePoint list	

See also

Data sources supported in SQL Server Analysis Services tabular 1200 models

Data sources supported in Azure Analysis Services

Data sources supported in SQL Server Analysis Services tabular 1200 models

11/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes the types of data sources that can be used with SQL Server Analysis Services tabular models at the 1200 and lower compatibility level.

For models at the 1400 compatibility levels, see Data sources supported in SQL Server Analysis Services tabular 1400 models.

For Azure Analysis Services, see Data sources supported in Azure Analysis Services.

Supported data sources for in-memory tabular models

When you install SQL Server Data Tools (SSDT), setup does not install the providers that are listed for each data source. Some providers might be installed with other applications on your computer. In other cases, you may need to download and install the provider.

Source	Versions	File type	Providers
Access databases	Microsoft Access 2010 and later.	.accdb or .mdb	ACE 14 OLE DB provider [1]
SQL Server relational databases	SQL Server 2008 and later, SQL Server Data Warehouse 2008 and later, Azure SQL Database, Azure SQL Data Warehouse, Analytics Platform System (APS) Analytics Platform System (APS) was formerly known as SQL Server Parallel Data Warehouse (PDW). Originally, connecting to PDW from Analysis Services required a special data provider. This provider was replaced in SQL Server 2012. Starting in SQL Server 2012, the SQL Server native client is used for connections to PDW/APS.	(not applicable)	OLE DB Provider for SQL Server SQL Server Native Client OLE DB Provider SQL Server Native 10.0 Client OLE DB Provider .NET Framework Data Provider for SQL Client

Oracle relational databases	Oracle 9i and later.	(not applicable)	Oracle OLE DB Provider .NET Framework Data Provider for Oracle Client .NET Framework Data Provider for SQL Server OraOLEDB MSDASQL
Teradata relational databases	Teradata V2R6 and later	(not applicable)	TDOLEDB OLE DB provider .Net Data Provider for Teradata
Informix relational databases		(not applicable)	Informix OLE DB provider
IBM DB2 relational databases	8.1	(not applicable)	DB2OLEDB
Sybase Adaptive Server Enterprise (ASE) relational databases	15.0.2	(not applicable)	Sybase OLE DB provider
Other relational databases	(not applicable)	(not applicable)	OLE DB provider or ODBC driver
Text files	(not applicable)	.txt, .tab, .csv	ACE 14 OLE DB provider [1]
Microsoft Excel files	Excel 2010 and later	.xlsx, xlsm, .xlsb, .xltx, .xltm	ACE 14 OLE DB provider [1]
Power Pivot workbook	Microsoft SQL Server 2008 and later Analysis Services	xlsx, xlsm, .xlsb, .xltx, .xltm	ASOLEDB 10.5 (used only with Power Pivot workbooks that are published to SharePoint farms that have Power Pivot for SharePoint installed)
Analysis Services cube	Microsoft SQL Server 2008 and later Analysis Services	(not applicable)	ASOLEDB 10
Data feeds (used to import data from Reporting Services reports, Atom service documents, Microsoft Azure Marketplace DataMarket, and single data feed)	Atom 1.0 format Any database or document that is exposed as a Windows Communication Foundation (WCF) Data Service (formerly ADO.NET Data Services).	`.atomsvc` for a service document that defines one or more feeds .atom for an Atom web feed document	Microsoft Data Feed Provider for Power Pivot .NET Framework data feed data provider for Power Pivot
Office Database Connection files		.odc	

Using **ACE 14 OLE DB provider** to connect to file data types **is not recommended**. If you must retain your

tabular 1200 and lower compatibility level models, export your data to a csv file type, import to SQL database, and then connect to and import from the database. However, it's recommended you upgrade to tabular 1400 compatibility level (SQL Server 2017 and later) and use **Get Data** in SSDT to select and import your file data source. Get Data uses structured data source connections provided by the Power Query data engine, which are more stable than ACE 14 OLE DB provider connections.

Supported data sources for DirectQuery models

DirectQuery is an alternative to in-memory storage mode, routing queries to and returning results directly from backend data systems rather than storing all data inside the model (and in RAM once the model is loaded). Because Analysis Services has to formulate queries in the native database query syntax, a smaller subset of data sources is supported for this mode.

DATA SOURCE	VERSIONS	PROVIDERS
Microsoft SQL Server	2008 and later	OLE DB Provider for SQL Server, SQL Server Native Client OLE DB Provider, .NET Framework Data Provider for SQL Client
Microsoft Azure SQL Database	All	OLE DB Provider for SQL Server, SQL Server Native Client OLE DB Provider, .NET Framework Data Provider for SQL Client
Microsoft Azure SQL Data Warehouse	All	SQL Server Native Client OLE DB Provider, .NET Framework Data Provider for SQL Client
Microsoft SQL Analytics Platform System (APS)	All	OLE DB Provider for SQL Server, SQL Server Native Client OLE DB Provider, .NET Framework Data Provider for SQL Client
Oracle relational databases	Oracle 9i and later	Oracle OLE DB Provider
Teradata relational databases	Teradata V2R6 and later	.Net Data Provider for Teradata

Tips for choosing data sources

Importing tables from relational databases saves you steps because *foreign key* relationships are used during import to create relationships between tables in the model designer.

Importing multiple tables, and then deleting the ones you don't need, can also save you steps. If you import tables one at a time, you might still need to create relationships between the tables manually.

Columns that contain similar data in different data sources are the basis of creating relationships within the model designer. When using heterogeneous data sources, choose tables that have columns that can be mapped to tables in other data sources that contain identical or similar data.

OLE DB providers can sometimes offer faster performance for large-scale data. When choosing between different providers for the same data source, you should try the OLE DB provider first.

See also

Data sources supported in SQL Server Analysis Services tabular 1400 models

Tabular model designer

12/7/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The tabular model designer is part of SQL Server Data Tools (SSDT), integrated with Microsoft Visual Studio, with additional project type templates specifically for developing professional tabular model solutions. SQL Server Data Tools (SSDT) is installed as a free web download. See Download SQL Server Data Tools (SSDT) for details.

Benefits

When you install SQL Server Data Tools (SSDT), new project templates for creating tabular models are added to the available project types. After creating a new tabular model project by using one of the templates, you can begin model authoring by using the tabular model designer tools and wizards.

In addition to new templates and tools for authoring professional multidimensional and tabular model solutions, the Visual Studio environment provides debugging and project lifecycle capabilities that ensure you create the most powerful BI solutions for your organization. For more information about Visual Studio, see Getting Started with Visual Studio.

Project templates

When you install SQL Server Data Tools (SSDT), the following tabular model project templates are added to the Business Intelligence project types:

Analysis Services Tabular Project
This template can be used to create a new, blank tabular model project. Compatibility levels are specified when you create the project.

Import from Server (Tabular)
This template can be used to create a new tabular model project by extracting the metadata from an existing tabular model in Analysis Services.

Older models have older compatibility levels. You can upgrade by changing the Compatibility Level property after importing the model definition.

Import from Power Pivot
This template is used for creating a new tabular model project by extracting the metadata and data from a Power Pivot for Excel file.

Windows and Menus

The SQL Server Data Tools (SSDT) tabular model authoring environment includes the following:

Designer window

The designer window is used to author tabular models by providing a visual representation of the model. When you open the Model.bim file, the model opens in the designer window. You can author a model in the designer window by using two different view modes:

Data view
The data view displays tables in a tabular, grid format. You can also define measures by using the measure grid, which can be shown for each table in Data View only.

Diagram view

The diagram view displays tables, with relationships between them, in a graphical format. Columns, measures, hierarchies, and KPIs can be filtered, and you can choose to view the model by using a defined perspective.

Most model authoring tasks can be performed in either view.

View Code window

You can view the code behind a Model.bim file when you right-click select **View Code** on the file in Solution Explorer. For Tabular models at compatibility level 1200 and later, the model definition is expressed in JSON.

Note that you will need a full version of Visual Studio that provides the JSON editor. You can download and install the free Visual Studio Community edition if you do not require the additional features in commercial editions.

Solution Explorer

The Solution Explorer window presents the active solution as a logical container for a tabular model project and its associated items. The model project (.smproj) contains only a References object (empty) and the Model.bim file. You can open project items for modification and perform other management tasks directly from this view.

Tabular model solutions typically contain only one project; however, a solution can contain other projects too, for example, Integration Services or Reporting services project. You can add any number of files provided they are not of the same type as tabular model project files and their Build Action property is set to None, or Copy to Output property is set to Do Not Copy.

To view Solution Explorer, click the **View** menu, and then click **Solution Explorer**.

Tabular Model Explorer

First available in the August 2016 release (14.0.60812.0) of SQL Server Data Tools, Tabular Model Explorer helps you navigate metadata objects in tabular models.

To show Tabular Model Explorer, click **View** > **Other Windows**, and then click **Tabular Model Explorer**.

Tabular Model Explorer organizes metadata objects into a tree structure that closely resembles the schema of a tabular model. Data Sources, Perspectives, Relationships, Roles, Tables, and Translations correspond to top-level schema objects. There are some exceptions, specifically KPIs and Measures, which technically aren't top-level objects, but child objects of the various tables in the model. However, having consolidated top-level containers for all KPIs and Measures makes it easier to work with these objects, especially if your model includes a very large number of tables. Measures are also listed under their corresponding parent tables, so you have a clear view of the actual parent-child relationships. If you select a measure in the top-level Measures container, the same measure is

also selected in the child collection under its table, and vice-versa.

Object nodes in Tabular Model Explorer are linked to appropriate menu options that until now were hiding under the Model, Table, and Column menus in Visual Studio. You can right-click an object to explore options for the object type. Not all object node types have a context menu yet, but additional options and improvements are coming in subsequent releases.

Tabular Model Explorer also offers a convenient search feature. Just type in a portion of the name in the Search box and Tabular Model Explorer narrows down the tree view to the matches.

Properties window

The Properties window lists the properties of the selected object. The following objects have properties that can be viewed and edited in the Properties window:

- Model.bim
- Table
- Column
- Measure

Project properties display only the project name and project folder in the Properties window. Projects also have additional deployment Options and deployment server settings that you can set using a modal properties dialog box. To view these properties, in **Solution Explorer**, right click the project, and then click **Properties**.

Fields in the Properties window have embedded controls that open when you click them. The type of edit control depends on the particular property. Controls include edit boxes, dropdown lists, and links to custom dialog boxes. Properties that are shown as dimmed are read-only.

To view the **Properties** window, click the **View** menu, and then click **Properties Window**.

Error List

The Error List window contains messages about the model state:

- Notifications about security best practices.
- Requirements for data processing.
- Semantic error information for calculated columns, measures, and row filters for roles. For calculated columns, you can double-click the error message to navigate to the source of the error.
- DirectQuery validation errors.

By default, the **Error List** does not appear unless an error is returned. You can, however, view the **Error List** window at any time. To view the **Error List** window, click the **View** menu, and then click **Error List**.

Output

Build and deployment information is displayed in the **Output** Window (in addition to the modal progress dialog). To view the **Output** window, click the **View** menu, and then click Output.

Menu items

When you install SQL Server Data Tools (SSDT), additional menu items specifically for authoring tabular models are added to the Visual Studio menu bar. The **Model** menu can be used to launch the Data Import Wizard, view existing connections, process workspace data, and browse the model workspace in Microsoft Excel. The **Table** menu is used to create and manage relationships between tables, create and manage measures, specify data table settings, specify calculation options, and specify other table properties. With the **Column** menu, you can add and delete columns in a table, hide and unhide columns, and specify other column properties such as data types and filters. You can build and deploy tabular model solutions on the **Build** menu. Copy/Paste functions are included on

the **Edit** menu.

In addition to these menu items, additional settings are added to the Analysis Services options on the Tools menu items.

Toolbar

The Analysis Services toolbar provides quick and easy access to the most frequently used model authoring commands.

Visual Studio integration

Source control

Analysis Services projects integrate with the selected source control plug-in. If you have configured Visual Studio to use source control, you can use check in/check out from Solution Explorer. To configure to use Team Foundation Server, see Configure Visual Studio with Team Foundation Version Control. Many third-party source control plug-ins are supported as well.

Fonts

Tabular models use the Visual Studio environment font to control the fonts in the display. It can be necessary to change this font if the default Visual Studio font does not have all of the Unicode characters you need for your language. To change fonts, click the **Tools** menu, then click **Options**, and then click **Fonts and Colors**.

Keyboard shortcuts

The Analysis Services keyboard shortcuts can be configured/remapped through the Tools->Options->Keyboard dialog. Some global Visual Studio shortcuts, such as build, save, debug, new project, etc. are supported in the tabular model designer context. Other tabular model designer specific shortcuts are in the Analysis Services context.

See also

Tabular Model Projects

Workspace database

12/7/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The tabular model workspace database, used during model authoring, is created when you create a new tabular model project in SQL Server Data Tools (SSDT).

Specifying a workspace instance

When you create a new tabular model project in SSDT, you can specify an Analysis Services instance to use while authoring your project. Beginning with the September 2016 release (14.0.60918.0) of SQL Server Data Tools, introduces two modes for specifying a workspace instance when you create a new tabular model project.

Integrated workspace - Recommended. Utilizes SSDT's own internal Analysis Services instance. Use this setting when creating a project that will be deployed to Azure Analysis Services.

Workspace server - A workspace database is created on an explicit Analysis Services instance, often on the same computer as SSDT or another computer in the same network. While you can specify an Azure Analysis Services server, it's not recommended.

Integrated workspace

With Integrated workspace, a working database is created in-memory using SSDTs own implicit Analysis Services instance. Integrated workspace mode significantly reduces the complexity of authoring tabular projects in SSDT because a separate explicit Analysis Services server is not required.

By using Integrated workspace mode, SSDT Tabular dynamically starts its own internal SSAS instance in the background and loads the database. You can add and view tables, columns, and data in the model designer. If you add additional tables, columns, relationships, etc., you're modifying the workspace database. Integrated workspace mode does not change how SSDT Tabular works with a workspace server and database. What changes is where SSDT Tabular hosts the workspace database.

You can select Integrated workspace mode when creating a new tabular model project in SSDT.

By using the Workspace Database and Workspace Server properties for model.bim, you can discover the name of the temporary database and the TCP port of the internal SSAS instance where SSDT Tabular hosts the database. You can connect to the workspace database with SSMS as long as SSDT Tabular has the database loaded. The Workspace Retention setting specifies that SSDT Tabular keeps the workspace database on disk, but no longer in memory after a model project is closed. This ensures less memory is consumed than if the model was kept in memory at all times. If you want to control these settings, set the Integrated Workspace Mode property to False and then provide an explicit workspace server. An explicit workspace server also make senses if the data you are importing into a model exceeds the memory capacity of your SSDT workstation.

> **NOTE**
> When using Integrated workspace mode, the local Analysis Services instance is 64-bit, while SSDT runs in the 32-bit environment of Visual Studio. If you're connecting to special data sources, make sure you install both the 32-bit and 64-bit versions of the corresponding data providers on your workstation. The 64-bit provider is required for the 64-bit Analysis Services instance and the 32-bit version is required for the Table Import Wizard in SSDT.

Workspace server

A workspace database is created on the Analysis Services instance, specified in the Workspace Server property, when you create a new Business Intelligence project by using one of the tabular model project templates in SQL Server Data Tools. Each tabular model project will have its own workspace database. You can use SQL Server Management Studio to view the workspace database on the Analysis Services server. The workspace database name includes the project name, followed by an underscore, followed by the username, followed by an underscore, followed by a GUID.

The workspace database resides in-memory while the tabular model project is open in SQL Server Data Tools. When you close the project, the workspace database is either kept in-memory, stored to disk and removed from memory (default), or removed from memory and not stored on disk, as determined by the Workspace Retention property. For more information about the Workspace Retention property, see Workspace Database Properties later in this topic.

After you've added data to your model project by using the Table Import Wizard or by using copy/paste, when you view the tables, columns, and data in the model designer, you are viewing the workspace database. If you add

additional tables, columns, relationships, etc. you are changing the workspace database.

When you deploy a tabular model project, the deployed model database, which is essentially a copy of the workspace database, is created on the Analysis Services server instance specified in the Deployment Server property. For more information about the Deployment Server property, see Project properties.

The model workspace database typically resides on localhost or a local named instance of an Analysis Services server. You can use a remote instance of Analysis Services to host the workspace database, however, this configuration is not recommended due to latency during data queries and other restrictions. Optimally, the instance of Analysis Services that will host the workspace databases is on the same computer as SQL Server Data Tools. Authoring model projects on the same computer as the Analysis Services instance that hosts the workspace database can improve performance.

Remote workspace databases have the following restrictions:

- Potential latency during queries.
- The Data Backup property cannot be set to **Backup to disk**.
- You cannot import data from a Power Pivot workbook when creating a new tabular model project by using the Import from Power Pivot project template.

> **IMPORTANT**
> The model's compatibility level and the Workspace Server must correspond.

> **NOTE**
> If any of the tables in your model will contain a large number of rows, consider importing only a subset of the data during model authoring. By importing a subset of the data, you can reduce processing time and consumption of workspace database server resources.

> **NOTE**
> The preview window in the Select Tables and Views page in the Table Import Wizard, Edit Table Properties dialog box, and Partition Manager dialog box show tables, columns, and rows at the data source, and may not show the same tables, columns, and rows as the workspace database.

Workspace Database Properties

Workspace database properties are included in the model properties. To view model properties, in SQL Server Data Tools, in **Solution Explorer**, click the **Model.bim** file. Model properties can be configured using the **Properties** window. Workspace database specific properties include:

> **NOTE**
> **Integrated Workspace Mode**, **Workspace Server**, **Workspace Retention**, and **Data Backup** properties have default settings applied when you create a new model project. You can change the default settings for new model projects on the **Data Modeling** page in **Analysis Server** settings in the Tools\Options dialog box. These properties, as well as others, can also be set for each model project in the **Properties** window. Changing default settings will not apply to model projects already created. For more information, see Configure default data modeling and deployment properties.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Integrated Workspace Mode	True, False	If Integrated workspace mode is selected for the workspace database when the project is created, this property will be True. If **Workspace server** mode is selected when the project is created, this property will be False.
Workspace database	Name	The name of the workspace database. This property cannot be edited when **Integrated Workspace Mode** is **True**.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Workspace Retention	Unload from memory	Specifies how a workspace database is retained after a model project is closed. A workspace database includes model metadata and imported data. In some cases, the workspace database can be very large and consume a large amount of memory. By default, when you close a model project in SQL Server Data Tools, the workspace database is unloaded from memory. When changing this setting it is important to consider your available memory resources as well as how often you plan to work on the model project. This property setting has the following options: **Keep in memory** - Specifies to keep the workspace database in memory after a model project is closed. This option will consume more memory; however, when opening a model project in SQL Server Data Tools, fewer resources are consumed and the workspace database will load faster. **Unload from memory** - Specifies to keep the workspace database on disk, but no longer in memory after a model project is closed. This option will consume less memory; however, when opening a model project in SQL Server Data Tools, the workspace database must be re-attached; additional resources are consumed and the model project will load more slowly than if the workspace database is kept in memory. Use this option when in-memory resources are limited or when working on a remote workspace database. **Delete workspace** - Specifies to delete the workspace database from memory and not keep the workspace database on disk after the model project is closed. This option will consume less memory and storage space; however, when opening a model project in SQL Server Data Tools, additional resources are consumed and the model project will load more slowly than if the workspace database is kept in memory or on-disk. Use this option when only occasionally working on model projects. The default setting for this property can be changed on the **Data Modeling** page in **Analysis Server** settings in the Tools\Options dialog box. This property cannot be edited when **Integrated Workspace Mode** is **True**.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Workspace Server	localhost	This property specifies the default server that will be used to host the workspace database while the model project is being authored in SQL Server Data Tools. All available instances of Analysis Services running on the local computer are included in the listbox. To specify a different Analysis Services server (running in Tabular mode), type the server name. The user logged on must be an Administrator on the Analysis Services server. Note that It is recommended you specify a local Analysis Services server as the workspace server. For workspace databases on a remote server, importing from Power Pivot is not supported, data cannot be backed up locally, and the user interface may experience latency during queries. The default setting for this property can be changed on the Data Modeling page in Analysis Services settings in the Tools\Options dialog box. This property cannot be edited when **Integrated Workspace Mode** is **True**.

Using SSMS to Manage the Workspace Database

You can use SQL Server Management Studio (SSMS) to connect to a Analysis Services server that hosts a workspace database. Typically, there is no management of the workspace database necessary; the exception, is to detach or delete a workspace database, which must be done from SQL Server Management Studio. Do not use SQL Server Management Studio to manage the workspace database while the project is open in the model designer. Doing so could lead to data loss.

See also

Model properties

Tabular Model Projects

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

To create a new tabular model, you must first create a new tabular model project in SQL Server Data Tools (SSDT). When you install SQL Server Data Tools, three new Business Intelligence project templates are provided for creating tabular model projects. Each project template provides a framework on which you can begin authoring your model. Topics in this section describe how to create a new tabular model project by using the project templates.

Related Tasks

TOPIC	DESCRIPTION
Create a New Tabular Model Project (Analysis Services)	This article describes how to create a new, blank tabular model project.
Import from Analysis Services	This article describes how to create a new tabular model project based on an existing model already in Analysis Services. Metadata from the existing model is extracted and used to create the new model.
Import from Power Pivot	This article describes how to create a new tabular model project by extracting the metadata and data from a Power Pivot for Excel file (.xlsx).

Create a new tabular model project

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes how to create a new, blank tabular model project in SQL Server Data Tools (SSDT). After a new model project has been created, you can begin authoring your model project by importing data from data sources.

To create a new, blank tabular model project

1. In SQL Server Data Tools, on the **File** menu, click **New**, and then click **Project**.
2. In the **New Project** dialog box, under **Installed Templates**, click **Business Intelligence**, and then click **Tabular Model Project**.
3. In **Name**, type a name for the project, then specify a location and solution name, and then click **OK**.

After your new project is created, it is important to set project and model properties. For more information, see Project properties and Model properties.

See Also

Project properties
Model properties

Import from Analysis Services

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes how to create a new tabular model project by importing the metadata from an existing tabular model by using the Import from Server project template in SQL Server Data Tools (SSDT).

Create a new model by importing metadata from an existing model in Analysis Services

You can use the Import from Server project template to create a new tabular model project by copying the metadata from an existing tabular model on an Analysis Services server. The new project will be created with the same data source connections, tables, relationships, measures, KPIs, roles, hierarchies, perspectives, and partitions as the model it was imported from. The data, however, is not copied from the existing model to the new model workspace. Once the import process has completed, and the new model project created, you must run a Process All to load the data from the data sources into the new model project workspace database.

To create a new model by importing metadata from an existing model

1. In SQL Server Data Tools, on the **File** menu, click **New**, and then click **Project**.
2. In the **New Project** dialog box, under **Installed Templates**, click **Business Intelligence**, and then click **Import from Server**.
3. In **Name**, type a name for the project, then specify a location and solution name, and then click **OK**.
4. In the **Import from Analysis Services** dialog box, in **Server Name**, specify the name of the Analysis Services server that contains the model metadata you want to import.
5. In **Database Name**, select the tabular model database that contains the model metadata you want to import, and then click **OK**.

See Also

Project properties

Import from Power Pivot

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes how to create a new tabular model project by importing the metadata and data from a Power Pivot workbook by using the Import from Power Pivot project template in SQL Server Data Tools (SSDT).

Create a new Tabular Model from a Power Pivot for Excel file

When creating a new tabular model project by importing from a Power Pivot workbook, the metadata that defines the structure of the workbook is used to create and define the structure of the tabular model project in SQL Server Data Tools. Objects such as tables, columns, measures, and relationships are retained and will appear in the tabular model project as they are in the Power Pivot workbook. No changes are made to the .xlsx workbook file.

> **NOTE**
>
> Tabular models do not support linked tables. When importing from a Power Pivot workbook that contains a linked table, linked table data is treated as copy\pasted data and stored in the Model.bim file. When viewing properties for a copy\pasted table, the **Source Data** property is disabled and the **Table Properties** dialog on the **Table** menu is disabled.
>
> There is a limit of 10,000 rows that can be added to the data embedded in the model. If you import a model from Power Pivot and see the error, "Data was truncated. Pasted tables cannot contain more than 10000 rows" you should revise the Power Pivot model by moving the embedded data into another data source, such as a table in SQL Server, and then re-import.

There are special considerations depending on whether or not the workspace database is on an Analysis Services instance on the same computer (local) as SQL Server Data Tools (SSDT) or is on a remote Analysis Services instance..

If the workspace database is on a local instance of Analysis Services, you can import both the metadata and data from the Power Pivot workbook. The metadata is copied from the workbook and used to create the tabular model project. The data is then copied from the workbook and stored in the project's workspace database (except for copy/pasted data, which is stored in the Model.bim file).

If the workspace database is on a remote Analysis Services instance, you cannot import data from a Power Pivot for Excel workbook. You can still import the workbook metadata; however, this will cause a script to be run on the remote Analysis Services instance. You should only import metadata from a trusted Power Pivot workbook. Data must be imported from sources defined in the data source connections. Copy/pasted and linked table data in the Power Pivot workbook must be copied and pasted into the tabular model project.

To create a new tabular model project from a Power Pivot for Excel file

1. In SQL Server Data Tools, on the **File** menu, click **New**, and then click **Project**.
2. In the **New Project** dialog box, under **Installed Templates**, click **Business Intelligence**, and then click **Import from Power Pivot**.
3. In **Name**, type a name for the project, then specify a location and solution name, and then click **OK**.
4. In the **Open** dialog box, select the Power Pivot for Excel file that contains the model metadata and data you want to import, and then click **Open**.

See Also

Workspace Database
Copy and Paste Data

Power View - Reporting Properties

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Power View provides intuitive ad-hoc reporting for business users such as data analysts, business decision makers, and information workers. They can easily create and interact with views of data from tabular models based on Power Pivot workbooks published in a Power Pivot Gallery, or tabular models authored by using SQL Server Data Tools (SSDT) and then deployed to SQL Server 2017 Analysis Services instances. Power View is a browser-based Silverlight application launched from SharePoint Server 2010 or later.

When authoring tabular model projects in SQL Server Data Tools, you can configure certain reporting properties unique to Power View reports. Topics in this section describe how to optimize a model to improve the reporting experience in Power View.

Related Tasks

TOPIC	DESCRIPTION
Configure Default Field Set for Power View Reports	Describes how to configure a Default Field Set; a predefined list of columns and measures that are automatically added to a Power View report canvas when the table is selected in the report field list.
Configure Table Behavior Properties for Power View Reports	Describes how to configure table behavior properties that expose detail rows at a more granular level. Setting table behavior properties changes the grouping behavior of detail rows and produces a better default placement of identifying information in tile, card, and chart layouts.

Power View - Configure Default Field Set for Reports

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A default field set is a predefined list of columns and measures that are automatically added to a Power View report canvas when the table is selected in the report field list. Tabular model authors can create a default field set to eliminate redundant steps for report authors who use the model for their reports. For example, if you know that most report authors who work with customer contact information always want to see a contact name, a primary phone number, an email address, and a company name, you can pre-select those columns so that they are always added to the report canvas when the author clicks the Customer Contact table.

> **NOTE**
> A default field set applies only to a tabular model used as a data model in Power View. Default field sets are not supported in Excel pivot reports.

Creating a Default Field Set

You can determine which fields, if any, are included by default whenever a specific table is selected in Power View. You can also determine the order in which fields appear in the list. To specify a default field set, you set report properties in the tabular model project.

To add a default field set

1. In SQL Server Data Tools (SSDT), click the table (tab) for which you are configuring a default field list.
2. In the **Properties** window, in the **Default Field Set** property, click **Click to edit**.
3. In the Default Field Set dialog, select one or more fields. You can choose any field in the table, including measures. Hold down the Shift key to select a range, or the Ctrl key to select individual fields.
4. Click **Add** to add them to the default field set.
5. Use the Up and Down buttons to specify an order to the field list. Fields will be added to the report in the order defined for the field set.
6. Repeat these steps for other tables in your workbook.

Next Step

After you create a default field set, you can further influence the report design experience by specifying default labels, default images, default group behavior, or whether rows that contain the same value are grouped together in one row or listed individually. For more information, see Configure Table Behavior Properties for Power View Reports.

Power View - Configure Table Behavior Properties for Reports

12/7/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If you are using a tabular model as a data model for Power View, you can set table behavior properties that expose detail rows at a more granular level. Setting table behavior properties changes the grouping behavior of detail rows and produces a better default placement of identifying information (such as names, photo IDs, or logo images) in tile, card, and chart layouts.

Power View differs from other reporting applications in that it will group items automatically during report design, by evaluating which columns you placed in the report field list against the presentation format you are using. In most cases, the default grouping produces an optimum result. But for some tables, primarily those that contain detail data, the default grouping behavior will sometimes group rows that should not be grouped. For such tables, you can set properties that change how groups are evaluated.

Setting table behavior properties is recommended for tables where the individual rows are of primary interest, such as employee or customer records. In contrast, tables that do not benefit from these properties include those that act as a lookup table (for example, a date table, a product category table, or a department table, where the table consists of a relatively small number of rows and columns), or summary tables containing rows that are only interesting when summarized (for example, census data that rolls up by gender, age, or geography). For lookup and summary tables, the default grouping behavior produces the best result.

> **NOTE**
> Table behavior properties only affect tabular models used as data models in Power View. Table behavior properties are not supported in Excel pivot reports.

Table behavior properties include the following:

- **Row Identifier** — specifies a column that contains only unique values, allowing that column to be used as an internal grouping key.
- **Keep Unique Rows** — specifies which columns provide values that should be treated as unique even if they are duplicates (for example, employee first name and last name, for cases where two or more employees share the same name).
- **Default Label** — specifies which column provides a display name to represent row data (for example, employee name in an employee record).
- **Default Image** — specifies which column provides an image that represents the row data (for example, a photo ID in an employee record).

> **NOTE**
> Refer to the following section to approach layout optimizations from the standpoint of a particular presentation format: Optimizing for Specific Layouts.

Opening the Table Behavior dialog box

1. In SQL Server Data Tools (SSDT), click the table (tab) for which you are configuring a default field list.
2. In the **Properties** window, in the **Table Behavior** property, click **Click to edit**.
3. In the **Table Behavior** dialog box, set the **Row Identifier**, and then specify other properties in this dialog box.

Setting the Row Identifier property

Within the table, the row identifier specifies a single column that contains only unique values and no blank values. The Row Identifier property is used to change grouping so a group is not based on a row's field composition, but rather on a fixed column that is always used to uniquely identify a row, regardless of the fields used in a particular report layout.

Setting this property changes the default grouping behavior from dynamic grouping based on the columns present on the canvas, to a fixed grouping behavior that summarizes based on the row identifier. Changing the default grouping behavior is relevant for report layouts, such as a matrix, that would otherwise group (or show subtotals) for each column in the row.

In Power View, setting a row identifier enables the following additional properties: **Keep Unique Rows** property, **Default Label** property, and **Default Image** property.

You can also use **Row Identifier** by itself, as a standalone property, to enable the following:

- Use of binary images in a report. By removing ambiguity around row uniqueness, Power View can determine how to assign default images and default labels for a given row.
- Remove unwanted subtotals from a matrix report. Default grouping at the field level creates a subtotal for each field. If you only want a single subtotal that is calculated at the row level, setting the Row Identifier produces this result.

You cannot set a Row Identifier for tables marked as date tables. For date tables, the row identifier is specified when you mark the table. For more information, see Mark as Date Table Dialog Box .

Setting the Keep Unique Rows property

This property allows you to specify which columns convey identity information (such as an employee name or a product code) in way that distinguishes one row from another. In cases where rows appear to be identical (such as two customers with the same name), the columns you specify for this property repeat in the report table.

Depending on which columns you add to a report, you might find rows that are treated as identical rows because the values in each row appear to be the same (for example two customers named Jon Yang). This might occur because other columns that provide differentiation (such as a middle name, address, or birth date) are not on the report canvas. In such a scenario, the default behavior is to group the apparently identical rows into a single row, summarizing any calculated values into a single larger result from the combined rows.

By setting the **Keep Unique Rows** property, you can designate one or more columns that should always repeat, even if there are duplicate instances, whenever you add that column to the report canvas. Calculated values associated with the row will now be allocated based on each individual row rather than rolled up to single row. When choosing columns for the **Keep Unique Rows** property, choose those containing unique or nearly unique values.

> **NOTE**
> Because the columns the end user selects can affect the grouping, which changes the filter context for expression calculations, model designers must take care to create measures that return the correct results. For more information, see Power View FAQ.

Setting a default label

This property specifies a label that appears in the navigation strip of a tile report. When used with a default image, the default label appears under the image. Without an image, the default label appears by itself. When choosing a default label, pick the column that conveys the most information about the row (for example, a name).

In a tile layout, the default label appears in the title area below an image, as defined by the Default Image property. For example, if you have a list of employees, you might tile employee information, using their photo ID as the default image and Employee Name as the default label. In a tile, the default label appears under the default image. These columns always appear in the tile, even if you do not explicitly select them in the report field list.

Setting a default image

This property specifies an image that appears in the navigation strip of a tile report, or at the front of a card. In the report, when you select the column that contains the default image, the default image will appear placed in the navigation strip of a tile report layout, or at the front of a card. A default image should be visual content. Examples include a photo ID in the employees table, a customer logo in a customer table, or a country shape in a geography table.

> **NOTE**
> Images can be sourced from URL addresses to an image file on a web server, or as binary data embedded in the workbook. If the image is based on a URL, be sure to also set the column as an image type so that Power View retrieves the image rather than displaying the URL as text data in the report.

Optimizing for Specific Layouts

This section describes the effect of setting table behavior properties from the standpoint of a particular presentation format and characteristics of the data. If you are trying to fine tune the layout of a matrix report, for example, you can use this information to understand how to improve a matrix presentation using table behavior properties in the model.

Images are missing

Properties you set in the model determine whether images are visualized in a report, or represented as text values in the report.

Category	Category Drawing
beverages	http://localhost/Shared Documents/Images/HelloWorldPicnic/rs_crescent_picnic_drawing_beverages.png
beverages	http://localhost/Shared Documents/Images/HelloWorldPicnic/rs_crescent_picnic_drawing_beverages.png
beverages	http://localhost/Shared Documents/Images/HelloWorldPicnic/rs_crescent_picnic_drawing_beverages.png
beverages	http://localhost/Shared Documents/Images/HelloWorldPicnic/rs_crescent_picnic_drawing_beverages.png
beverages	http://localhost/Shared Documents/Images/HelloWorldPicnic/rs_crescent_picnic_drawing_beverages.png
breads	http://localhost/Shared Documents/Images/HelloWorldPicnic/rs_crescent_picnic_drawing_breads.png
breads	http://localhost/Shared Documents/Images/HelloWorldPicnic/rs_crescent_picnic_drawing_breads.png
breads	http://localhost/Shared Documents/Images/HelloWorldPicnic/rs_crescent_picnic_drawing_breads.png
breads	http://localhost/Shared Documents/Images/HelloWorldPicnic/rs_crescent_picnic_drawing_breads.png
breads	http://localhost/Shared Documents/Images/HelloWorldPicnic/rs_crescent_picnic_drawing_breads.png

By default, text in the model is interpreted as text in the report. If a text column is a URL address to a report image, remember to set the **Image URL** property so that Power View retrieves the image file. For binary images, remember to set the **Row Identifier** property.

Tables are missing one or more rows

Sometimes the default grouping behavior results in an outcome that is opposite of what you intended; specifically, detail rows that are present in the model do not appear in the report. By default, Power View groups on the columns you add to the canvas. If you add **Country Name** to the report, each country appears once on the canvas, even though the underlying table might contain thousands of rows that include multiple instances of each country name. In this case, the default grouping behavior produces the correct result.

However, consider a different example where you might want multiple instances of a row to appear, because in fact the underlying rows contain data about different entities. In this example, assume you have two customers both named **Jon Yang**. Using the default grouping behavior, only one instance of **Jon Yang** will appear in the report. Moreover, because only one instance appears in the list, the measure **Annual Income** is the sum of that value for both customers.

FirstName	LastName	YearlyIncome
Jack	Yang	90,000.00
Jackson	Yang	30,000.00
Jaclyn	Yang	40,000.00
Jake	Yang	20,000.00
James	Yang	40,000.00
Jamie	Yang	30,000.00
Jason	Yang	130,000.00
Jeffery	Yang	40,000.00
Jenny	Yang	110,000.00
Jessie	Yang	40,000.00
Jon	Yang	150,000.00
Jonathan	Yang	40,000.00
Jordan	Yang	80,000.00

To change the default grouping behavior, set the **Row Identifier** and **Keep Unique Rows** properties. In **Keep Unique Rows**, choose the Last Name column so this value is repeated for a row, even if it already appears in a different row. After you change the properties and republish the workbook, you can create the same report, only this time you will see both customers named **Jon Yang**, with the **Annual Income** correctly allocated to each one.

FirstName	LastName	YearlyIncome
Jenny	Yang	110,000.00
Jenny	Ye	80,000.00
Jeremy	Young	40,000.00
Jerry	Yuan	10,000.00
Jesse	Young	60,000.00
Jessie	Yang	40,000.00
Jessie	Ye	10,000.00
Johnny	Yuan	80,000.00
Jon	Yang	90,000.00
Jon	Yang	60,000.00
Jon	Ye	40,000.00
Jon	Yuan	20,000.00
Jonathan	Yang	40,000.00

Matrix layout is too crowded

When you present a detail table in a matrix, the default grouping provides a summarized value for each column. Depending on your objectives, this might be more summarizations than you want. To change this behavior, you can set **Row Identifier**. No additional properties need to be set; just setting row identifier is sufficient to change the grouping so that summarizations are calculated for each row based on its unique row identifier.

Compare the following before and after images that show the effect of setting this property for a matrix layout.

Before: Default grouping based on fields in matrix

Name	Category	Color	MM Avg Qty Served
apple	fruits	green	26.93
apple	fruits	red	21.11
apple juice	beverages	brown	41.94
avocado	vegetables	green	26.33
bagel	breads	brown	26.44
baguette	breads	white	46.25
banana	fruits	yellow	25.21
carrot	vegetables	orange	20.82
coffee	beverages	brown	61.94
corn	vegetables	yellow	26.08
croissant	breads	brown	32.39
kiwi	fruits	green	29.95
lemon	fruits	yellow	26.93

After: Grouping on row identifier

Name	Category	Color	MM Avg Qty Served
apple	fruits	green	26.93
apple	fruits	red	21.11
apple juice	beverages	brown	41.94
avocado	vegetables	green	26.33
bagel	breads	brown	26.44
baguette	breads	white	46.25
banana	fruits	yellow	25.21
carrot	vegetables	orange	20.82
coffee	beverages	brown	61.94
corn	vegetables	yellow	26.08
croissant	breads	brown	32.39
kiwi	fruits	green	29.95
lemon	fruits	yellow	26.93

Chart showing too many items and levels on the axis

Chart reports that show detail data should use the row identifier as an axis. Without a row identifier, the axis is indeterminate, resulting in a best-guess layout that might not make sense. To change this behavior, you can set **Row Identifier**. No additional properties need to be set; just setting row identifier is sufficient to change the grouping so that summarizations are calculated for each row based on its unique row identifier.

Compare the following before and after images that show the effect of setting this property for a chart layout. It is the same report, with identical fields and presentation. The only difference is the bottom image shows a report after **Row Identifier** was set on the Items table.

Before: Default grouping based on fields in a chart

After: Grouping on row identifier (row identifier becomes the axis)

Next Steps

After you have evaluated the tables in your model and set table behavior properties on those containing detail rows that should always appear as individual items, you can further optimize the model through additional properties or settings.

Project properties

12/7/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes model project properties. Each tabular model project has deployment options and deployment server properties that specify how the project and model is deployed. For example, the server the model will be deployed to and the deployed model database name. These settings are different from model properties, which affect the model workspace database. The project properties described here are in a modal properties dialog box, different from the properties window used to display other types of properties. To display the modal project properties, in SQL Server Data Tools (SSDT), in **Solution Explorer**, right-click the project, and then click **Properties**.

Sections in this topic:

- Project properties
- Configure Deployment Options and Deployment Server property settings

Project properties

Deployment Options

PROPERTY	DEFAULT SETTING	DESCRIPTION
Processing Option	**Default**	By default, Analysis Services will determine the type of processing required when changes to objects are deployed. This generally results in the fastest deployment time. However, you can also choose to have either full processing or no processing performed with each deployment.
Transactional Deployment	**False**	Specifies whether or not the deployment of the model is transactional. By default, the deployment of all or changed objects is not transactional with the processing of those deployed objects. Deployment can succeed and persist even though processing fails. You can change this to incorporate deployment and processing in a single transaction.
Query Mode	**In-Memory**	Specifies the source from which query results are returned. For more information, see DirectQuery Mode.

Deployment Server

PROPERTY	DEFAULT SETTING	DESCRIPTION

PROPERTY	DEFAULT SETTING	DESCRIPTION
Server	**localhost**	Specifies an instance of Analysis Services. By default, models are deployed to the default instance of Analysis Services on the local computer. You can change this setting to specify a named instance on the local computer or any instance on any remote computer on which you have permission to create Analysis Services objects. Typically, Administrator permissions. The default setting for this property can be changed using the Default Deployment Server property on the Deployment page in Analysis Server settings in the Tools\Options dialog box. For more information, see Configure default data modeling and deployment properties.
Edition	**Developer**	Specifies the edition of the Analysis Services server to which the model will be deployed. The server edition defines various features that can be incorporated into the project.
Database	**Model**	Specifies the name of the Analysis Services database in which model objects will be instantiated upon deployment. This name will be specified in a data connection or an .rsds data connection file. It is recommended the name reflect the type of analysis that will be performed using the model, for example AdventureWorksSalesModel. To prevent duplicate names for deployed models, you should change the **Database** property name setting to reflect the purpose of the model. When users connect to the model as a data source, this is the name they will see.
Cube Name	**Model**	Specifies the name of the database cube as shown in a reporting client data connection.
Version	**13.0**	Version of the Analysis Services instance where the project will be deployed.

DirectQuery Options

PROPERTY	DEFAULT SETTING	DESCRIPTION

PROPERTY	DEFAULT SETTING	DESCRIPTION
Impersonation Settings	**Default**	Specifies the credentials used to connect to data sources for a model running in DirectQuery Mode. These credentials are different from impersonation credentials that are used in the default In-Memory mode. For more information, see Impersonation.

Configure Deployment Options and Deployment Server property settings

1. In SQL Server Data Tools (SSDT), in **Solution Explorer**, right-click the project, and then click **Properties**.
2. In the **Properties** window, click a property, and then type a value or click the down arrow to select a setting option.

See Also

Configure default data modeling and deployment properties
Model properties
Tabular model solution deployment

Model properties

12/7/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes tabular model properties. Each tabular model project has model properties that affect how the model you are authoring in SQL Server Development Tools is built, how it is backed up, and how the workspace database is stored. Model properties described here do not apply to models that have already been deployed.

Sections in this topic:

- Model properties
- Configure model property settings

Model properties

Advanced

PROPERTY	DEFAULT SETTING	DESCRIPTION
Build Action	Compile	This property specifies how the file relates to the build and deployment process. This property setting has the following options: **Compile** - A normal build action occurs. Definitions for model objects will be written to the .asdatabase file. **None** - The output to the .asdatabase file will be empty.
Copy to Output Directory	Do Not Copy	This property specifies the source file will be copied to the output directory. This property setting has the following options: **Do not copy** - No copy is created in the output directory. **Copy always** - A copy is always created in the output directory. **Copy if newer** - A copy is created in the output directory only if there are changes to the model.bim file.

Miscellaneous

> **NOTE**
>
> Some properties are set automatically when the model is created and cannot be changed.

> **NOTE**
> Workspace Server, Workspace Retention, and Data Backup properties have default settings applied when you create a new model project. You can change the default settings for new models on the Data Modeling page in Analysis Server settings in the Tools\Options dialog box. These properties, as well as others, can also be set for each model in the Properties window. For more information, see Configure default data modeling and deployment properties.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Collation	Default collation for the computer for which Visual Studio is installed.	The collation designator for the model.
Compatibility Level	Default or other selected when creating the project.	Applies to SQL Server 2012 Analysis Services SP1 or later. Specifies the features and settings available for this model. For more information, see Compatibility Level for Tabular models in Analysis Services.
Data Backup	Do not back up to disk	Specifies whether or not a backup of the model data is kept in a backup file. This property setting has the following options: **Back up to disk** - Specifies to keep a backup of model data on disk. When the model is saved, the data will also be saved to the backup (ABF) file. Selecting this option can cause slower model save and load times **Do not back up to disk** - Specifies to not keep a backup of model data on disk. This option will minimize save and model loading time. The default setting for this property can be changed on the Data Modeling page in Analysis Server settings in the Tools\Options dialog box.
Default filter direction	Single direction	Determines the default filter direction for new relationships.
DirectQuery Mode	Off	Specifies whether or not this model operates in DirectQuery Mode. For more information, see DirectQuery Mode.
File Name	Model.bim	Specifies the name of the .bim file. The file name should not be changed.
Full Path	Path specified when the project was created.	The model.bim file location. This property cannot be set in the Properties window.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Language	English	The default language of the model. The default language is determined by the Visual Studio language. This property cannot be set in the Properties window.
Workspace Database	The project name, followed by an underscore, followed by a GUID.	The name of the workspace database used for storing and editing the in-memory model for the selected model.bim file. This database will appear in the Analysis Services instance specified in the Workspace Server property. This property cannot be set in the Properties window. For more information, see Workspace Database.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Workspace Retention	Unload from memory	Specifies how a workspace database is retained after a model is closed. A workspace database includes model metadata, the data imported into a model, and impersonation credentials (encrypted). In some cases, the workspace database can be very large and consume a significant amount of memory. By default, workspace databases are unloaded from-memory. When changing this setting, it is important to consider your available memory resources as well as how often you plan to work on the model. This property setting has the following options: **Keep in memory** - Specifies to keep the workspace database in memory after a model is closed. This option will consume more memory; however, when opening a model, fewer resources are consumed and the workspace database will load faster. **Unload from memory** - Specifies to keep the workspace database on disk, but no longer in memory after a model is closed. This option will consume less memory; however, when opening a model, additional resources are consumed and the model will load more slowly than if the workspace database is kept in memory. Use this option when in-memory resources are limited or when working on a remote workspace database. **Delete workspace** - Specifies to delete the workspace database from memory and not keep the workspace database on disk after the model is closed. This option will consume less memory and storage space; however, when opening a model, additional resources are consumed and the model will load more slowly than if the workspace database is kept in memory or on-disk. Use this option when only occasionally working on models. The default setting for this property can be changed on the Data Modeling page in Analysis Server settings in the Tools\Options dialog box.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Workspace Server	localhost	This property specifies the default server that will be used to host the workspace database while the model is being authored. All available instances of Analysis Services running on the local computer are included in the listbox. The default setting for this property can be changed on the Data Modeling page in Analysis Server settings in the Tools\Options dialog box. Note: It is recommended that you always specify a local Analysis Services server as the workspace server. For workspaces databases on a remote server, importing from Power Pivot is not supported, data cannot be backed up locally, and the user interface may experience latency during queries.

Configure model property settings

1. In SSDT, in **Solution Explorer**, click the **Model.bim** file.
2. In the **Properties** window, click a property, and then type a value or click the down arrow to select a setting option.

See Also

Configure default data modeling and deployment properties
Project properties

Table Properties

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes tabular model table properties. The properties described here are different from those in the Edit Table Properties dialog box, which define which columns from the source are imported.

Sections in this topic:

- Table Properties
- Configure table property settings

Table Properties

Basic

PROPERTY	DEFAULT SETTING	DESCRIPTION
Connection Name	<connection name>	The name of the connection to the table's data source. To edit the connection, click the button.
Hidden	False	Specifies whether the table is hidden from reporting client field lists.
Partitions		Partitions for the table cannot be displayed in the **Properties** window. To view, create, or edit partitions, click the button to open the Partition Manager.
Source Data		Source data for the table cannot be displayed in the **Properties** window. To view or edit the source data, click the button to open the Edit Table Properties dialog box.
Table Description		A text description for the table. In Power Pivot for Excel, if an end-user places the cursor over this table in the field list, the description appears as a tooltip.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Table Name	<friendly name>	Specifies the table's friendly name. The table name can be specified when a table is imported using the Table Import Wizard or at any time after import. The table name in the model can be different from the associated table at the source. The table friendly name appears in the reporting client application field list as well as in the model database in SQL Server Management Studio.

Reporting Properties

For detailed descriptions and configuration information for reporting properties, see Power View reporting properties.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Default Field Set		
Table Behavior		

Configure table property settings

1. In the model designer, in Data View, click a table (tab), or, in Diagram View, click a table header.
2. In the **Properties** window, click on a property, and then type a value or click the button for additional configuration options.

Column properties

12/19/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes tabular model column properties.

> **NOTE**
> Some properties are not supported in all compatibility levels.

Column properties

Advanced

PROPERTY	DEFAULT SETTING	DESCRIPTION
Display Folder		A single or nested folder for organizing columns in a client application field list.

Basic

PROPERTY	DEFAULT SETTING	DESCRIPTION
Column Name		The name of the column as it is stored in the model and as it is displayed in a reporting client field list.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Data Format	Automatically determined during import.	Specifies the display format to use for the data in this column. This property has the following options: **General** **Decimal Number** **Whole Number** **Currency** **Percentage** **Scientific** After you set a data format, you can set properties that are specific to each format. For example, if you choose the **Currency** format, you can set the number of visible decimal places, choose the thousands separator, and choose the currency symbol. If the column values contain images, see **Representative Image**.
Data Type	Automatically determined during import.	Specifies the data type for all values in the column.
Description		A text description for the column. In certain reporting clients, if an end-user places the cursor over this column in the field list, the description appears as a tooltip.
Hidden	False	Specifies whether the column is hidden from reporting client field lists. Set this property to **True** to hide this column in the display. For example, columns that contain identifiers or keys are typically not useful to the end user. If you hide a column from the reporting client, the field is not suppressed in the model data. The field is still visible if you create a query against the model. A hidden column can still be used for grouping or sorting. The **Hidden** property does not provide any form of data security. To secure data, use row filters in Roles. For more information, see Roles.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Sort By Column		Specifies another column to sort the values in this column. A relationship must exist between the two columns. This value must be the name of an existing column. You cannot specify a formula or measure.
Unique		Can be set to enforce uniqueness of values in the column. Always true for calculated columns, even if uniqueness is false.

Misc.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Encoding Hints	Default	Specifies encoding to optimize processing. Value encoding can improve query performance for numeric columns typically used in aggregations. Hash encoding is for group-by columns (often dimension-table values) and foreign keys. String columns are always hash encoded.

Reporting Properties

PROPERTY	DEFAULT SETTING	DESCRIPTION
Default Image	False	Specifies which column provides an image that represents the row data (for example, a photo ID in an employee record).
Default Label	False	Specifies which column provides a display name to represent row data (for example, employee name in an employee record).
Image URL/Data Category (SP1)	False	Specifies the value in this column as a hyperlink to an image on a server. For example: `http://localhost/images/image1.jpg` .
Keep Unique Rows	False	Specifies which columns provide values that should be treated as unique even if they are duplicates (for example, employee first name and last name, for cases where two or more employees share the same name).
Row Identifier	False	Specifies a column that contains only unique values, allowing that column to be used as an internal grouping key.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Summarize By	Default	Specifies reporting client tools apply the aggregate function SUM for column calculations when this column is added to a Field list. To change the default calculation, select it from the dropdown list. This property applies only to columns of type that can be aggregated.
Table Detail Position	No Default Field Set	Specifies this column or measure can be added to a set of fields from a single table to enhance the table visualization experience in a reporting client.

Configure column property settings

1. In the model designer, in a table, select a column.
2. In the **Properties** window, click on a property, and then type a value or click the down arrow to select a setting option.

See also

Power View reporting properties
Hide or freeze columns
Add columns to a table

Configure default data modeling and deployment properties

12/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes how to configure the default compatibility level, deployment and workspace database property settings, which can be pre-defined for each new tabular model project you create in SQL Server Data Tools (SSDT). After a new project is created, these properties can still be changed depending on your particular requirements.

To configure the default Compatibility Level property setting for new model projects

1. In SQL Server Data Tools (SSDT), click the **Tools** menu, and then click **Options**.
2. In the **Options** dialog box, expand **Analysis Services Tabular Designers**, and then click **Compatibility Level**.
3. Configure the following property settings:

PROPERTY	DEFAULT SETTING	DESCRIPTION
Default compatibility level for new projects	SQL Server 2016 (1200)	This setting specifies the default compatibility level to use when creating a new Tabular model project. You can choose SQL Server 2012 (1100) if you will deploy to an Analysis Services instance without SP1 applied, or SQL Server 2012 SP1 or later if your deployment instance has SP1 applied. For more information, see Compatibility Level for Tabular models in Analysis Services.
Compatibility level options	All checked	Specifies compatibility level options for new Tabular model projects and when deploying to another Analysis Services instance.

To configure the default deployment server property setting for new model projects

1. In SQL Server Data Tools (SSDT), click the **Tools** menu, and then click **Options**.
2. In the **Options** dialog box, expand **Analysis Services Tabular Designers**, and then click **Deployment**.
3. Configure the following property settings:

PROPERTY	DEFAULT SETTING	DESCRIPTION

PROPERTY	DEFAULT SETTING	DESCRIPTION
Default deployment server	localhost	This setting specifies the default server to use when deploying a model. You can click the down arrow to browse for local network Analysis Services servers you can use or you can type the name of a remote server.

> **NOTE**
> Changes to the Default deployment Server property setting will not affect existing projects created prior to the change.

To configure the default Workspace Database property settings for new model projects

1. In SQL Server Data Tools (SSDT), click the **Tools** menu, and then click **Options**.
2. In the **Options** dialog box, expand **Analysis Services Tabular Designers**, and then click **Workspace Database**.
3. Configure the following property settings:

PROPERTY	DEFAULT SETTING	DESCRIPTION
Default workspace server	**localhost**	This property specifies the default server that will be used to host the workspace database while the model is being authored in SQL Server Data Tools (SSDT). All available instances of Analysis Services running on the local computer are included in the listbox. Note: It is recommended that you always specify a local Analysis Services server as the workspace server. For workspace databases on a remote server, importing data from Power Pivot is not supported, data cannot be backed up locally, and the user interface may experience latency during queries.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Workspace database retention after the model is closed	**Keep workspace databases on disk, but unload from memory**	Specifies how a workspace database is retained after a model is closed. A workspace database includes model metadata, the data imported into a model, and impersonation credentials (encrypted). In some cases, the workspace database can be very large and consume a significant amount of memory. By default, workspace databases are removed from memory. When changing this setting, it is important to consider your available memory resources as well as how often you plan to work on the model. This property setting has the following options: **Keep workspaces in memory** - Specifies to keep workspaces in memory after a model is closed. This option will consume more memory; however, when opening a model in SQL Server Data Tools (SSDT), fewer resources are consumed and the workspace will load faster. **Keep workspace databases on disk, but unload from memory** - Specifies to keep the workspace database on disk, but no longer in memory after a model is closed. This option will consume less memory; however, when opening a model in SQL Server Data Tools (SSDT), additional resources are consumed and the model will load more slowly than if the workspace database is kept in memory. Use this option when in-memory resources are limited or when working on a remote workspace database. **Delete workspace** - Specifies to delete workspace database from memory and not keep workspace database on disk after the model is closed. This option will consume less memory and storage space; however, when opening a model in SQL Server Data Tools (SSDT), additional resources are consumed and the model will load more slowly than if the workspace database is kept in memory or on-disk. Use this option when only occasionally working on models.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Data backup	**Keep backup of data on disk**	Specifies whether or not a backup of the model data is kept in a backup file. This property setting has the following options: **Keep backup of data on disk** - Specifies to keep a backup of model data on disk. When the model is saved, the data will also be saved to the backup (ABF) file. Selecting this option can cause slower model save and load times **Do not keep backup of databack on disk** - Specifies to not keep a backup of model data on disk. This option will minimize save and model loading time.

> **NOTE**
> Changes to default model properties will not affect properties for existing models created prior to the change.

See also

Project properties
Model properties
Compatibility level

Data sources supported in SQL Server Analysis Services tabular 1400 models

2/12/2019 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server 2017 Analysis Services Azure Analysis Services

This article describes the types of data sources that can be used with SQL Server Analysis Services (SSAS) tabular models at the 1400 compatibility level.

For SSAS tabular models at the 1200 and lower compatibility levels, see Data sources supported in SQL Server Analysis Services tabular 1200 models.

For Azure Analysis Services, see Data sources supported in Azure Analysis Services.

Cloud data sources

DATASOURCE	IN-MEMORY	DIRECTQUERY
Azure SQL Database	Yes	Yes
Azure SQL Data Warehouse	Yes	Yes
Azure Blob Storage	Yes	No
Azure Table Storage	Yes	No
Azure Cosmos DB	Yes	No
Azure Data Lake Store (Gen1)[1]	Yes	No
Azure HDInsight HDFS	Yes	No
Azure HDInsight Spark [2]	Yes	No

1 - ADLS Gen2 is currently not supported.
2 - Azure Databricks using the Spark connector is currently not supported.

Provider
In-memory and DirectQuery models connecting to Azure data sources use .NET Framework Data Provider for SQL Server.

On-premises data sources

Supported by in-memory and DirectQuery models

DATASOURCE	IN-MEMORY PROVIDER	DIRECTQUERY PROVIDER	

DATASOURCE	IN-MEMORY PROVIDER	DIRECTQUERY PROVIDER	
SQL Server	SQL Server Native Client 11.0, Microsoft OLE DB Provider for SQL Server, .NET Framework Data Provider for SQL Server	.NET Framework Data Provider for SQL Server	
SQL Server Data Warehouse	SQL Server Native Client 11.0, Microsoft OLE DB Provider for SQL Server, .NET Framework Data Provider for SQL Server	.NET Framework Data Provider for SQL Server	
Oracle	Microsoft OLE DB Provider for Oracle, Oracle Data Provider for .NET	Oracle Data Provider for .NET	
Teradata	OLE DB Provider for Teradata, Teradata Data Provider for .NET	Teradata Data Provider for .NET	

NOTE

For in-memory models, OLE DB providers can provide better performance for large-scale data. When choosing between different providers for the same data source, try the OLE DB provider first.

Supported by in-memory models only

DATABASE		
Access Database		
SQL Server Analysis Services		
IBM Informix (beta)		
JSON document		
Lines from binary		
MySQL Database		
PostgreSQL Database	Yes	No
SAP HANA	Yes	No
SAP Business Warehouse	Yes	No
Sybase Database	Yes	No

FILE	
Excel workbook	
Folder	
JSON	
Text/CSV	
XML table	

ONLINE SERVICES	
Dynamics 365	
Exhange Online	
Saleforce Objects	
Salesforce Reports	
SharePoint Online List	

OTHER	
Active Directory	
Exhange	
OData Feed	
ODBC query	
OLE DB	
SharePoint list	

See also

Data sources supported in SQL Server Analysis Services tabular 1200 models

Data sources supported in Azure Analysis Services

Data sources supported in SQL Server Analysis Services tabular 1200 models

11/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes the types of data sources that can be used with SQL Server Analysis Services tabular models at the 1200 and lower compatibility level.

For models at the 1400 compatibility levels, see Data sources supported in SQL Server Analysis Services tabular 1400 models.

For Azure Analysis Services, see Data sources supported in Azure Analysis Services.

Supported data sources for in-memory tabular models

When you install SQL Server Data Tools (SSDT), setup does not install the providers that are listed for each data source. Some providers might be installed with other applications on your computer. In other cases, you may need to download and install the provider.

Source	Versions	File type	Providers
Access databases	Microsoft Access 2010 and later.	.accdb or .mdb	ACE 14 OLE DB provider [1]
SQL Server relational databases	SQL Server 2008 and later, SQL Server Data Warehouse 2008 and later, Azure SQL Database, Azure SQL Data Warehouse, Analytics Platform System (APS) Analytics Platform System (APS) was formerly known as SQL Server Parallel Data Warehouse (PDW). Originally, connecting to PDW from Analysis Services required a special data provider. This provider was replaced in SQL Server 2012. Starting in SQL Server 2012, the SQL Server native client is used for connections to PDW/APS.	(not applicable)	OLE DB Provider for SQL Server SQL Server Native Client OLE DB Provider SQL Server Native 10.0 Client OLE DB Provider .NET Framework Data Provider for SQL Client

Oracle relational databases	Oracle 9i and later.	(not applicable)	Oracle OLE DB Provider .NET Framework Data Provider for Oracle Client .NET Framework Data Provider for SQL Server OraOLEDB MSDASQL
Teradata relational databases	Teradata V2R6 and later	(not applicable)	TDOLEDB OLE DB provider .Net Data Provider for Teradata
Informix relational databases		(not applicable)	Informix OLE DB provider
IBM DB2 relational databases	8.1	(not applicable)	DB2OLEDB
Sybase Adaptive Server Enterprise (ASE) relational databases	15.0.2	(not applicable)	Sybase OLE DB provider
Other relational databases	(not applicable)	(not applicable)	OLE DB provider or ODBC driver
Text files	(not applicable)	.txt, .tab, .csv	ACE 14 OLE DB provider [1]
Microsoft Excel files	Excel 2010 and later	.xlsx, xlsm, .xlsb, .xltx, .xltm	ACE 14 OLE DB provider [1]
Power Pivot workbook	Microsoft SQL Server 2008 and later Analysis Services	xlsx, xlsm, .xlsb, .xltx, .xltm	ASOLEDB 10.5 (used only with Power Pivot workbooks that are published to SharePoint farms that have Power Pivot for SharePoint installed)
Analysis Services cube	Microsoft SQL Server 2008 and later Analysis Services	(not applicable)	ASOLEDB 10
Data feeds (used to import data from Reporting Services reports, Atom service documents, Microsoft Azure Marketplace DataMarket, and single data feed)	Atom 1.0 format Any database or document that is exposed as a Windows Communication Foundation (WCF) Data Service (formerly ADO.NET Data Services).	`.atomsvc` for a service document that defines one or more feeds .atom for an Atom web feed document	Microsoft Data Feed Provider for Power Pivot .NET Framework data feed data provider for Power Pivot
Office Database Connection files		.odc	

[1] Using **ACE 14 OLE DB provider** to connect to file data types **is not recommended**. If you must retain your tabular 1200 and lower compatibility level models, export your data to a csv file type, import to SQL database, and then connect to and import from the database. However, it's recommended you upgrade to tabular 1400 compatibility level (SQL Server 2017 and later) and use **Get Data** in SSDT to select and import your file data source. Get Data uses structured data source connections provided by the Power Query data engine, which are more stable than ACE 14 OLE DB provider connections.

Supported data sources for DirectQuery models

DirectQuery is an alternative to in-memory storage mode, routing queries to and returning results directly from backend data systems rather than storing all data inside the model (and in RAM once the model is loaded). Because Analysis Services has to formulate queries in the native database query syntax, a smaller subset of data sources is supported for this mode.

DATA SOURCE	VERSIONS	PROVIDERS
Microsoft SQL Server	2008 and later	OLE DB Provider for SQL Server, SQL Server Native Client OLE DB Provider, .NET Framework Data Provider for SQL Client
Microsoft Azure SQL Database	All	OLE DB Provider for SQL Server, SQL Server Native Client OLE DB Provider, .NET Framework Data Provider for SQL Client
Microsoft Azure SQL Data Warehouse	All	SQL Server Native Client OLE DB Provider, .NET Framework Data Provider for SQL Client
Microsoft SQL Analytics Platform System (APS)	All	OLE DB Provider for SQL Server, SQL Server Native Client OLE DB Provider, .NET Framework Data Provider for SQL Client
Oracle relational databases	Oracle 9i and later	Oracle OLE DB Provider
Teradata relational databases	Teradata V2R6 and later	.Net Data Provider for Teradata

Tips for choosing data sources

Importing tables from relational databases saves you steps because *foreign key* relationships are used during import to create relationships between tables in the model designer.

Importing multiple tables, and then deleting the ones you don't need, can also save you steps. If you import tables one at a time, you might still need to create relationships between the tables manually.

Columns that contain similar data in different data sources are the basis of creating relationships within the model designer. When using heterogeneous data sources, choose tables that have columns that can be mapped to tables in other data sources that contain identical or similar data.

OLE DB providers can sometimes offer faster performance for large-scale data. When choosing between different providers for the same data source, you should try the OLE DB provider first.

See also

Data sources supported in SQL Server Analysis Services tabular 1400 models

Data sources supported in Azure Analysis Services

Data types supported in tabular models

11/27/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes the data types that can be used in tabular models, and discusses the implicit conversion of data types when data is calculated or used in a Data Analysis Expressions (DAX) formula.

Data types used in tabular models

When you import data or use a value in a formula, even if the original data source contains a different data type, the data is converted to one of the following data types. Values that result from formulas also use these data types.

In general, these data types are implemented to enable accurate calculations in calculated columns, and for consistency the same restrictions apply to the rest of the data in models.

Formats used for numbers, currency, dates and times should follow the format of the locale that is specified on the client used to work with model data. You can use the formatting options in the model to control the way that the value is displayed.

Data type in model	Data type in DAX	Description
Whole Number	A 64 bit (eight-bytes) integer value* Note: DAX formulas do not support data types that are too small to hold the minimum value listed in the description.	Numbers that have no decimal places. Integers can be positive or negative numbers, but must be whole numbers between -9,223,372,036,854,775,808 (-2^63) and 9,223,372,036,854,775,807 (2^63-1).
Decimal Number	A 64 bit (eight-bytes) real number* Note: DAX formulas do not support data types that are too small to hold the minimum value listed in the description.	Real numbers are numbers that can have decimal places. Real numbers cover a wide range of values: Negative values from -1.79E +308 through -2.23E -308 Zero Positive values from 2.23E -308 through 1.79E + 308 However, the number of significant digits is limited to 17 decimal digits.
Boolean	Boolean	Either a True or False value.
Text	String	A Unicode character data string. Can be strings, numbers, or dates represented in a text format.

Date	Date/time	Dates and times in an accepted date-time representation. Valid dates are all dates after March 1, 1900.
Currency	Currency	Currency data type allows values between -922,337,203,685,477.5808 to 922,337,203,685,477.5807 with four decimal digits of fixed precision.
N/A	Blank	A blank is a data type in DAX that represents and replaces SQL nulls. You can create a blank by using the BLANK function, and test for blanks by using the logical function, ISBLANK.

* If you attempt to import data that has large numeric values, import might fail with the following error:

In-memory database error: The '<column name>' column of the '<table name>' table contains a value, '1.7976931348623157e+308', which is not supported. The operation has been canceled.

This error occurs because the model designer uses that value to represent nulls. The values in the following list are synonyms to the previous mentioned null value:

Value
9223372036854775807
-9223372036854775808
1.7976931348623158e+308
2.2250738585072014e-308

Remove the value from your data and try importing again.

> **NOTE**
> You cannot import from a **varchar(max)** column that contains a string length of more than 131,072 characters.

Table data type

In addition, DAX uses a *table* data type. This data type is used by DAX in many functions, such as aggregations and time intelligence calculations. Some functions require a reference to a table; other functions return a table that can then be used as input to other functions. In some functions that require a table as input, you can specify an expression that evaluates to a table; for some functions, a reference to a base table is required. For information about the requirements of specific functions, see DAX Function Reference.

Implicit and explicit data type conversion in DAX Formulas

Each DAX function has specific requirements as to the types of data that are used as inputs and outputs. For example, some functions require integers for some arguments and dates for others; other functions require text or

tables.

If the data in the column that you specify as an argument is incompatible with the data type required by the function, DAX in many cases returns an error. However, wherever possible DAX attempts to implicitly convert the data to the required data type. For example:

- You can type a number, for example "123", as a string. DAX parses the string and attempt to specify it as a number data type.
- You can add TRUE + 1 and get the result 2, because TRUE is implicitly converted to the number 1 and the operation 1+1 is performed.
- If you add values in two columns, and one value happens to be represented as text ("12") and the other as a number (12), DAX implicitly converts the string to a number and then does the addition for a numeric result. The following expression returns 44: = "22" + 22
- If you attempt to concatenate two numbers, they are presented as strings and then concatenated. The following expression returns "1234": = 12 & 34

The following table summarizes the implicit data type conversions that are performed in formulas. In general, semantic model designer behaves like Microsoft Excel, and performs implicit conversions whenever possible when required by the specified operation.

Table of implicit data conversions

The type of conversion that is performed is determined by the operator, which casts the values it requires before performing the requested operation. These tables list the operators, and indicate the conversion that is performed on each data type in the column when it is paired with the data type in the intersecting row.

> **NOTE**
> Text data types are not included in these tables. When a number is represented as in a text format, in some cases, the model designer attempts to determine the number type and represent it as a number.

Addition (+)

Operator (+)	INTEGER	CURRENCY	REAL	Date/time
INTEGER	INTEGER	CURRENCY	REAL	Date/time
CURRENCY	CURRENCY	CURRENCY	REAL	Date/time
REAL	REAL	REAL	REAL	Date/time
Date/time	Date/time	Date/time	Date/time	Date/time

For example, if a real number is used in an addition operation in combination with currency data, both values are converted to REAL, and the result is returned as REAL.

Subtraction (-)

In the following table, the row header is the minuend (left side) and the column header is the subtrahend (right side):

Operator (-)	INTEGER	CURRENCY	REAL	Date/time

INTEGER	INTEGER	CURRENCY	REAL	REAL
CURRENCY	CURRENCY	CURRENCY	REAL	REAL
REAL	REAL	REAL	REAL	REAL
Date/time	Date/time	Date/time	Date/time	Date/time

For example, if a date is used in a subtraction operation with any other data type, both values are converted to dates, and the return value is also a date.

> **NOTE**
> Tabular models also support the unary operator, - (negative), but this operator does not change the data type of the operand.

Multiplication (*)

Operator (*)	INTEGER	CURRENCY	REAL	Date/time
INTEGER	INTEGER	CURRENCY	REAL	INTEGER
CURRENCY	CURRENCY	REAL	CURRENCY	CURRENCY
REAL	REAL	CURRENCY	REAL	REAL

For example, if an integer is combined with a real number in a multiplication operation, both numbers are converted to real numbers, and the return value is also REAL.

Division (/)

In the following table the row header is the numerator and the column header is the denominator.

Operator (/) (Row/Column)	INTEGER	CURRENCY	REAL	Date/time
INTEGER	REAL	CURRENCY	REAL	REAL
CURRENCY	CURRENCY	REAL	CURRENCY	REAL
REAL	REAL	REAL	REAL	REAL
Date/time	REAL	REAL	REAL	REAL

For example, if an integer is combined with a currency value in a division operation, both values are converted to real numbers, and the result is also a real number.

Comparison operators

Only a limited set of mixed data-type combinations for comparison operations is supported. To learn more, see DAX Operator Reference.

Handling of blanks, empty strings, and zero values

The following table summarizes the differences between DAX and in Microsoft Excel, in the way that blanks are handled:

Expression	DAX	Excel
BLANK + BLANK	BLANK	0 (zero)
BLANK +5	5	5
BLANK * 5	BLANK	0 (zero)
5/BLANK	Infinity	Error
0/BLANK	NaN	Error
BLANK/BLANK	BLANK	Error
FALSE OR BLANK	FALSE	FALSE
FALSE AND BLANK	FALSE	FALSE
TRUE OR BLANK	TRUE	TRUE
TRUE AND BLANK	FALSE	TRUE
BLANK OR BLANK	BLANK	Error
BLANK AND BLANK	BLANK	Error

For details on how a particular function or operator handles blanks, see the individual topics for each DAX function, in the section, DAX Function Reference.

Impersonation

1/21/2019 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article provides tabular model authors an understanding of how sign in credentials are used by Analysis Services when connecting to a datasource to import and process (refresh) data.

Configuring impersonation

Where, and in what context a model exists determines how impersonation information is configured. When creating a new model project, impersonation is configured in SQL Server Data Tools (SSDT) when you connect to a data source to import data. Once a model is deployed, impersonation can be configured in a model database connection string property by using SQL Server Management Studio (SSMS). For tabular models in Azure Analysis Services, you can use SSMS or the **View as: Script** mode in the browser-based designer to edit the Model.bim file in JSON.

How impersonation is used

Impersonation is the ability of a server application, such as Analysis Services, to assume the identity of a client application. Analysis Services runs using a service account, however, when the server establishes a connection to a datasource, it uses impersonation so that access checks for data import and processing can be performed.

Credentials used for impersonation are different from the credentials you are currently signed in on with. Sign in user credentials are used for particular client-side operations when authoring a model.

It is important to understand how impersonation credentials are specified and secured, as well as the difference between contexts in which both your signed on user credentials are used and when other impersonation credentials are used.

Understanding server-side credentials

When data is imported or processed, impersonation credentials are used to connect to the datasource and fetch the data. This connection is a *server-side* operation running in the context of a client application because the Analysis Services server hosting the workspace database connects to the datasource and fetches the data.

When you deploy a model to an Analysis Services server, if the workspace database is in memory when the model is deployed, the credentials are passed to the Analysis Services server to which the model is deployed. User credentials are never stored on-disk.

When a deployed model processes data from a datasource, the impersonation credentials, persisted in the in-memory database, are used to connect to the datasource and fetch the data. Because this process is handled by the Analysis Services server managing the model database, this connection is again a server-side operation.

Understanding client-side credentials

When authoring a new model or adding a datasource to an existing model, you connect to a datasource and select tables and views to be imported into the model. In the Table Import Wizard or Get Data\Query Designer preview and filter features, you see a sample of the data you import. You can also specify filters to exclude data that isn't needed in the model.

Similarly, for existing models that have already been created, you use the **Table Properties** dialog to preview and filter data imported into a table.

The preview and filter features, **Table Properties**, and **Partition Manager** dialog boxes are an in-process *client-side* operation; that is, what is done during this operation are different from how the datasource is connected to and data is fetched from the datasource; a server-side operation. The credentials used to preview and filter data are the credentials of the user currently signed on. In-effect, your credentials.

The separation of credentials used during server-side and client-side operations can lead to a mismatch in what you see and what data is fetched during an import or process (a server-side operation). If the credentials you are currently signed in with and the impersonation credentials specified are different, the data you see in the preview and filter features or the **Table Properties** dialog and the data fetched during an import or process can be different, depending on the credentials required by the datasource.

> **IMPORTANT**
> When authoring a model, ensure the credentials you are signed in with and the credentials specified for impersonation have sufficient rights to fetch the data from the datasource.

Options

When configuring impersonation, or when editing properties for an existing datasource connection, specify one of the following options:

Tabular 1400 and higher models

OPTION	DESCRIPTION
Impersonate Account	Specifies the model use a Windows user account to import or process data from the datasource. The domain and name of the user account uses the following format:**<Domain name>\<User account name>**.
Impersonate Current User	Specifies data should be accessed from the datasource using the identity of the user who sent the request. This setting applies only to DirectQuery mode.
Impersonate Identity	Specifies a username to access the datasource, but doesn't need to specify the account's password. This setting applies only when Kerberos delegation is enabled and specifies the S4U authentication should be used.
Impersonate Service Account	Specifies the model use the security credentials associated with the Analysis Services service instance that manages the model.
Impersonate Unattended Account	Specifies the Analysis Services engine should use a pre-configured unattended account to access the data.

> **IMPORTANT**
> Impersonate Current User is not supported in some environments. Impersonate Current User is not supported for tabular models deployed to Azure Analysis Services that connect to on-premises data sources. Because an Azure Analysis Services server resource is not connected to an organization's domain, client credentials cannot be authenticated against a data source server in that domain. Azure Analysis Services also does not currently integrate with (Azure) SQL Database support for single sign-on (SSO). Depending on your environment, other impersonation settings also have restrictions. When attempting to use an impersonation setting that is not supported, an error is returned.

Tabular 1200 models

OPTION	DESCRIPTION
Specific Windows user name and password	This option specifies the model use a Windows user account to import or process data from the datasource. The domain and name of the user account uses the following format:**<Domain name>\<User account name>**. When creating a new model using the Table Import Wizard, this setting is the default option.
Service Account	This option specifies the model use the security credentials associated with the Analysis Services service instance that manages the model.

Security

Credentials used with impersonation are persisted in-memory by the VertiPaq™ engine. Credentials are never written to disk. If the workspace database is not in-memory when the model is deployed, the user is prompted to enter the credentials used to connect to the datasource and fetch data.

> **NOTE**
>
> It is recommended you specify a Windows user account and password for impersonation credentials. A Windows user account can be configured to use least privileges necessary to connect to and read data from the datasource.

See also

DirectQuery mode
Data sources
Tabular model solution deployment

Import data by using a native query

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services (starting with 2017) Azure Analysis Services

For tabular 1400 models, the new Get Data experience in Visual Studio Analysis Services projects provides immense flexibility in how you can mashup your data during import. This article describes creating a connection to a datasource and then creating a native SQL query to specify data import.

In order to complete the tasks described in this article, make sure you're using the latest version of SSDT. If you're using Visual Studio 2017, make sure you've downloaded and installed the September 2017 or later Microsoft Analysis Services Projects VSIX.

Download and install SSDT

Download Microsoft Analysis Services Projects VSIX

Create a datasource connection

If you don't already have a connection to your datasource, you need to create one.

1. In Visual Studio > **Tabular Model Explorer**, right-click **Data Sources**, and then click **New Data Source**.
2. In **Get Data**, select your datasource type, and then click **Connect**. Follow any additional steps required to connect to your datasource.

Enter a query as a named expression

1. In **Tabular Model Explorer**, right-click **Expressions** > **Edit Expressions**.
2. In **Query Editor**, click **Query** > **New Query** > **Blank Query**
3. In the formula bar, type

```
= Value.NativeQuery(#"DATA SOURCE NAME", "SELECT * FROM ...")
```

4. To create a table, in **Queries**, right-click the query, and then select **Create New Table**. The new table will have the same name as the query.

Example

This native query creates an Employee table in the model that includes all columns from the Dimension.Employee table at the datasource.

```
= Value.NativeQuery(#"SQL/myserver;WideWorldImportersDW", "SELECT * FROM Dimension.Employee")
```

After importing, a table named Employees is created in the model.

See also

Value.NativeQuery
Impersonation

DirectQuery mode

12/7/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes *DirectQuery mode* for Analysis Services tabular models at the 1200 and higher compatibility levels . DirectQuery mode can be turned on for models you're designing in SSDT, or for tabular models that have already been deployed, you can change to DirectQuery mode in SSMS. Before choosing DirectQuery mode, it's important to understand both the benefits and restrictions.

Benefits

By default, tabular models use an in-memory cache to store and query data. When tabular models query data residing in-memory, even complex queries can be incredibly fast. However, there are some limitations to using cached data. Namely, large data sets can exceed available memory, and data freshness requirements can be difficult if not impossible to achieve on a regular processing schedule.

DirectQuery overcomes these limitations while also leveraging RDBMS features making query execution more efficient. With DirectQuery:

- Data is up-to-date, and there is no extra management overhead of having to maintain a separate copy of the data (in the in-memory cache). Changes to the underlying source data can be immediately reflected in queries against the data model.
- Datasets can be larger than the memory capacity of an Analysis Services server.
- DirectQuery can take advantage of provider-side query acceleration, such as that provided by memory optimized column indexes.
- Security can be enforced by the back-end database , using row-level security features from the database (alternatively, you can use row-level security in the model via DAX).
- If the model contains complex formulas that might require multiple queries, Analysis Services can perform optimization to ensure that the query plan for the query executed against the back-end database will be as efficient as possible.

Restrictions

Tabular models in DirectQuery mode have some restrictions. Before switching modes, it's important to determine whether the advantages of query execution on the backend server outweigh any reduction in functionality.

If you change the mode of an existing model in SQL Server Data Tools, the model designer will notify you of any features in your model that are incompatible with DirectQuery mode.

The following list summarizes the main feature restrictions to keep in mind:

Feature area	**Restriction**

Data sources	DirectQuery models can only use data from a single relational database of the following types: SQL Server, Azure SQL Database, Oracle, and Teradata. See Data sources supported for DirectQuery later in this article for version and provider information.
SQL stored procedures	For DirectQuery models, stored procedures cannot be specified in a SQL statement to define tables when using Data Import Wizard.
Calculated tables	Calculated tables are not supported in DirectQuery models, but calculated columns are. If you try to convert a tabular model that contains a calculated table, an error will occur stating that the model cannot contain pasted data.
Query limits	Default row limit is one million rows, which you can increase by specifying **MaxIntermediateRowSize** in the msmdsrv.ini file. See DAX Properties for details.
DAX formulas	When querying a tabular model in DirectQuery mode, Analysis Services converts DAX formulas and measure definitions into SQL statements. DAX formulas containing elements that cannot be converted into SQL syntax will return validation errors on the model. This restriction is mostly limited to certain DAX functions. For measures, DAX formulas are converted to set-based operations against the relational data store. This means that all measures created implicitly are supported. When a validation error occurs, you'll need to re-write the formula, substituting a different function, or workaround it by using derived columns in the data source. If a tabular model includes formulas containing incompatible functions will be reported when you switch to DirectQuery mode in the designer. **Note:** Some formulas in the model might validate when you switch the model to DirectQuery mode, but return different results when executed against the cache vs. the relational data store. This is because calculations against the cache use the semantics of the in-memory analytics (engine, which contains features meant to emulate the behavior of Excel, whereas queries against data stored in the relational data source use the semantics of SQL Server. To learn more, see DAX formula compatibility in DirectQuery mode.
Formula consistency	In certain cases, the same formula can return different results in a cached model compared to a DirectQuery model that uses only the relational data store. These differences are a consequence of the semantic differences between the in-memory analytics engine and SQL Server. For a complete list of compatibility issues, including functions that might return different results when the model is deployed to real-time, see DAX formula compatibility in DirectQuery mode (SQL Server Analysis Services).

MDX limitations	No relative object names. All object names must be fully qualified. No session-scope MDX statements (named sets, calculated members, calculated cells, visual totals, default members, and so forth), but you can use query-scope constructs, such as the 'WITH' clause. No tuples with members from different levels in MDX subselect clauses. No user-defined hierarchies. No native SQL queries (normally, Analysis Services supports a T-SQL subset, but not for DirectQuery models).

Data sources supported for DirectQuery

DirectQuery tabular models at compatibility level 1200 and higher are compatible with the following data sources and providers:

DATA SOURCE	VERSIONS	PROVIDERS
Microsoft SQL Server	2008 and later	OLE DB Provider for SQL Server, SQL Server Native Client OLE DB Provider, .NET Framework Data Provider for SQL Client
Microsoft Azure SQL Database	All	OLE DB Provider for SQL Server, SQL Server Native Client OLE DB Provider, .NET Framework Data Provider for SQL Client
Microsoft Azure SQL Data Warehouse	All	.NET Framework Data Provider for SQL Client
Microsoft SQL Analytics Platform System (APS)	All	OLE DB Provider for SQL Server, SQL Server Native Client OLE DB Provider, .NET Framework Data Provider for SQL Client
Oracle relational databases	Oracle 9i and later	Oracle OLE DB Provider
Teradata relational databases	Teradata V2R6 and later	.Net Data Provider for Teradata

Connecting to a data source

When designing a DirectQuery model in SSDT, connecting to a data source and selecting the tables and fields to include in your model is much the same as with in-memory models.

If you've already turned on DirectQuery but haven't yet connected to a data source, you can use the Table Import Wizard to connect to your data source, select tables and fields, specify a SQL query, and so on. The difference will be when you finish, no data is actually imported to the in-memory cache.

If you've already used Table Import Wizard to import data, but haven't yet turned on DirectQuery mode, when you do, the in-memory cache will be cleared.

Additional articles in this section

Enable DirectQuery mode in SSDT

Enable DirectQuery mode in SSMS

Add sample data to a DirectQuery model in Design Mode

Define partitions in DirectQuery models

Test a model in DirectQuery mode

DAX formula compatibility in DirectQuery mode

Enable DirectQuery mode in SSDT

12/10/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In this topic, we'll describe how to enable DirectQuery mode for a tabular model project in SQL Server Data Tools.

When you enable DirectQuery mode for a tabular model you're designing in SSDT:

- Features that are incompatible with DirectQuery mode are disabled.
- The existing model is validated. Warnings are displayed if features are incompatible with DirectQuery mode.
- If data was previously imported prior to enabling DirectQuery mode, the working model's cache is emptied.

Enable DirectQuery

In SSDT, in the **Properties** pane for the **Model.bim** file, change the property, **DirectQuery Mode**, to **On**.

If your model already has a connection to a data source and existing data, you'll be prompted to enter database credentials used to connect to the relational database. Any data already existing within the model will be removed from the in-memory cache.

If your model is partially or fully complete prior ti enabling DirectQuery mode, you might get errors about incompatible features. In Visual Studio, open the **Error List** and resolve any problems that would prevent the model from being switched to DirectQuery mode.

What's next?

You can now import data using the Table Import Wizard to get metadata for the model. You won't get rows of data, but you will get tables, columns, and relationships to use as the basis for your model.

You can create a sample partition for each table and add sample data so that you can verify model behavior as you build it. Any sample data that you add is used in **Analyze for Excel** or in other client tools that can connect to the workspace database. See Add sample data to a DirectQuery model in design mode for details.

> **TIP**
>
> Even in DirectQuery mode on an empty model, you can always view a small built-in rowset for each table. In SQL Server Data Tools, click **Table** > **Table Properties** to view the 50-row dataset.

See also

Enable DirectQuery mode in SSMS

Add sample data to a DirectQuery model in design mode

Enable DirectQuery mode in SSMS

12/7/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can change the data access properties of a tabular model that has already been deployed, enabling DirectQuery mode, where queries execute against a backend relational data source rather than cached data residing in-memory.

In SQL Server 2017, steps for DirectQuery configuration differ based on the model's compatibility level. Below you'll find steps that work for all compatibility levels.

This article assumes that you have created and validated an in-memory tabular model at compatibility level 1200 or higher, and only need to enable DirectQuery access and update connection strings. If you're starting from a lower compatibility level, you need to manually upgrade it first. See Upgrade Analysis Services for steps.

> **IMPORTANT**
>
> We recommend using SQL Server Data Tools instead of Management Studio to switch data storage modes. When you use SQL Server Data Tools to change model, and then follow that up with deployment to the server, the model and database stay in sync. Moreover, changing the storage modes in the model lets you review any validation errors that occur. When using SQL Server Management Studio as described in this article, validation errors are not reported.

Requirements

Enabling the use of Direct Query mode on a tabular model is a multistep process:

- Ensure that the model does not have features which might cause validation errors in DirectQuery mode, and then change the data storage mode on the model from in-memory to DirectQuery.

  A list of feature restrictions is documented in DirectQuery Mode.

- Review the connection string and credentials used by the deployed database to retrieve data from the backend external database. Make sure there is only one connection, and that its settings are suitable for query execution.

  Tabular databases that weren't specifically designed for DirectQuery could have multiple connections which now need to be reduced to one, as required for DirectQuery mode.

  Credentials originally used for processing data will now be used to query data. As part of DirectQuery configuration, review and possibly change the account if you use different ones for dedicated operations.

  DirectQuery mode is the only scenario in which Analysis Services does trusted delegation. If your solution calls for delegation to get user-specific query results, the account used to connect to the backend database must be allowed to delegate the identity of the user making the request, and user identities must have Read permissions on the backend database.

- As a last step, confirm that DirectQuery mode is operational through query execution.

Step 1: Check the compatibility level

Properties that define data access are different across compatibility levels. A preliminary step is to check to see what compatibility level the database is at.

1. In Management Studio, connect to the instance that has the tabular model.
2. In Object Explorer, right-click the database > **Properties** > **Compatibility Level**.

   The value will either be **SQL Server 2016 (1200)** or an earlier level like **SQL Server 2012 SP1 or later (1103)** . In the next step, follow the instructions that are valid for the compatibility level.

When you change a tabular model to DirectQuery mode, the new data storage mode takes effect immediately.

Step 2a: Switch a tabular 1200 database to DirectQuery mode

1. In Object Explorer, right-click the database > **Properties** > **Model** > **Default Mode**.
2. Set the mode to **DirectQuery**.

Valid values	**Description**
DirectQuery	Queries are executed against a backend relational database, using the data source connection defined for the model. Queries to the model are converted to native database queries and redirected to the data source. When you process a model set to DirectQuery mode, only metadata is compiled and deployed. The data itself is external to the model, residing in the database files of the operative data source.
Import	Queries are executed against the tabular database in either MDX or DAX. When you process a model set to Import mode, data is retrieved from a backend data source and stored on disk. When the database is loaded, data is copied entirely into memory for very fast table scans and queries. This is the default mode for tabular models, and it is the only mode for certain (non-relational) data sources.

Step 2b: Switch a tabular 1100-1103 database to DirectQuery mode

1. In Object Explorer, right-click the database > **Properties** > **Database** > **DirectQueryMode**.
2. Set the mode to **DirectQuery**.

   Default Mode properties consist of the following:

Valid values	**Description**
InMemory	Queries use the cached, in-memory data only.

InMemorywithDirectQuery	Queries use the cache by default, unless otherwise specified in the connection string from the client. This is a hybrid mode where partitions are individually configured to use in-memory or DirectQuery.
DirectQuery	Queries use the relational data source only.
DirectQuerywithInMemory	Queries use the relational data source by default, unless otherwise specified in the connection string from the client. This is a hybrid mode where partitions are individually configured to use in-memory or DirectQuery.

Hybrid Data Storage

When using combination of in-memory and disk-based access, the cache is still available and can be used for queries. A hybrid mode provides you with many options:

- When both the cache and the relational data source are available, you can set the preferred connection method, but ultimately the client controls which source is used, using the DirectQueryMode connection string property.
- You can configure partitions on the cache in such a way that the primary partition used for DirectQuery mode is never processed and must always reference the relational source. There are many ways to use partitions to optimize the model design and reporting experience. For more information, see Define partitions in DirectQuery models.
- After the model has been deployed, you can change the preferred connection method. For example, you might use a hybrid mode for testing, and switch the model over to **DirectQuery only** mode only after thoroughly testing any reports or queries that use the model. For more information, see Set or Change the Preferred Connection Method for DirectQuery.

Step 3: Check the connection properties on the database

Depending on how the data source connection is set up, switching to DirectQuery could change the security context of the connection. When changing the data access mode, review impersonation and connection string properties to verify the login is valid for ongoing connections to the backend database.

Review the **Configure Analysis Services for trusted delegation** section in Configure Analysis Services for Kerberos constrained delegation for background information on delegation of a user identity for DirectQuery scenarios.

1. In Object Explorer, expand **Connections** and double-click a connection to view its properties.

   For DirectQuery models, there should only be one connection defined for the database, and the data source must be relational, and of a supported database type. See Data Sources Supported.

2. **Connection string** should specify the server, database name, and the authentication method used in DirectQuery operations. If you're using SQL Server authentication, you can specify the database login here.

3. **Impersonation Info** is used for Windows authentication. Options that are valid for tabular models in DirectQuery mode include the following:

   - **Use the service account**. You can choose this option if the Analysis Services service account has read permissions on the relational database.

- **Use a specific user name and password**. Specify a Windows user account that has read permissions on the relational database.

Note that these credentials are used only for answering queries against the relational data store; they are not the same as the credentials used for processing the cache of a hybrid model.

Impersonation cannot be used when the model is used within memory only. The setting **ImpersonateCurrentUser**, is invalid unless the model is using DirectQuery mode.

Step 4: Validate DirectQuery access

1. Start a trace using either SQL Server Profiler or xEvents in Management Studio, connected to the relational database on SQL Server.

   If you are using Oracle or Teradata, use the tracing tools for those database platforms.

2. In Management Studio, enter and then execute a simple MDX query, such as `select <some measure> on 0 from model.`.

3. In the trace, you should see evidence of query execution on the relational database.

See also

Compatibility level
Data sources supported
Extended events

Add sample data to a DirectQuery model in Design Mode

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In DirectQuery mode, table partitions are used to either create sample data subsets used during model design, or create alternatives of a full data view.

When you deploy a DirectQuery tabular model, only one partition is allowed per table, and that partition by necessity must be full data view. Any additional partition is either a substitute for a full data view or sample data. In this topic, we're going to describe creating a sample partition, with a subset of data.

By default, when designing a tabular model in DirectQuery mode in SSDT, the model's working database doesn't contain any data. There is one default partition for each table, and this partition directs all queries to the data source.

You can, however, add a smaller amount of sample data to your model's working database for use at design time. Sample data is specified via a query on a sample partition used only during design. It's cached in-memory with the model. This will help you validate modeling decisions as you go without impacting the data source. You can test your modeling decisions with the sample dataset when using **Analyze in Excel** in SQL Server Data Tools (SSDT), or from other client applications that connect to your workspace database.

> **TIP**
> Even in DirectQuery mode on an empty model, you can always view a small built-in rowset for each table. In SQL Server Data Tools, click **Table** > **Table Properties** to view the 50-row dataset.

Create a sample partition

These instructions are for tabular models created at or upgraded to compatibility level 1200 or higher. Models at lower compatibility levels use different properties to get cached data. See Enable DirectQuery mode in SSMS for property descriptions.

1. In SQL Server Data Tools, in Diagram or Data View, click a fact table to open its properties page. Fact tables provide the aggregated, numeric data and measures in your model. You might have more than one.

2. Click **Table** > **Properties** to open the Partition Management dialog box.

   Notice the default partition is **(Direct Query) <table name>**. This is the full data view. Do not delete this partition. This partition will be used when the model is deployed.

3. Select the partition and then click **Copy**.

   This creates a copy of the default partition, however, this copy will contain sample data you specify in a query. For example:

4. Select the copied partition and then click the **SQL Query Editor** button to add a filter. Reduce your sample

data size while authoring your model. For example, if you selected **FactInternetSales** from AdventureWorksDW, your filter might look like the following:

```
SELECT [dbo].[FactInternetSales].* FROM [dbo].[FactInternetSales]
JOIN DimSalesTerritory as ST
ON ST.SalesTerritoryKey = FactInternetSales.SalesTerritoryKey
WHERE ST.SalesTerritoryGroup='North America';
```

5. Click **Validate** to check for syntax errors.

   Notice in DirectQuery mode, in addition to **New** , **Copy**, and **Delete** buttons on the Partitions dialog box, there is also a toggle button that alternately reads **Set as Sample** or **Set as DirectQuery**.

   Only one partition can be the DirectQuery partition. You can control it by selecting any partition defined for the table and clicking **Set as Sample**.

6. Process the table .

Define partitions in DirectQuery models

12/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This section explains how partitions are used in DirectQuery models. For more general information about partitions in tabular models, see Partitions.

> **NOTE**
> Although a table can have multiple partitions, in DirectQuery mode, only one of them can be designated for use in query execution. The single partition requirement applies to DirectQuery models at all compatibility levels.

Using Partitions in DirectQuery Mode

For each table, you must specify a single partition to use as the DirectQuery data source. If there are multiple partitions, when you switch the model to enable DirectQuery mode, by default the first partition that was created in the table is flagged as the DirectQuery partition. You can change this later by using the Partition Manager in SQL Server Data Tools (SSDT).

Why allow only a single partition in DirectQuery mode?

In tabular models (as in OLAP models), the partitions of a table are defined by SQL queries. The developer who creates the partition definition is responsible for ensuring that partitions do not overlap. Analysis Services does not check whether records belong in one or multiple partitions.

Partitions in a cached tabular model behave the same way. If you are using an in-memory model, while the cache is being accessed, DAX formulas are evaluated for each partition, and the results are combined. However, when a tabular model uses DirectQuery mode, it would be impossible to evaluate multiple partitions, combine the results, and convert that into a SQL statement for sending to the relational data store. Doing so could lead to unacceptable loss of performance, as well as potential inaccuracies as the results are aggregated.

Therefore, for queries answered in DirectQuery mode, the server uses a single partition that has been marked as the primary partition for DirectQuery access, called the *DirectQuery partition*. The SQL query specified in the definition of this partition defines the complete set of data that can be used to answer queries in DirectQuery mode.

If you do not explicitly define a partition, the engine simply issues a SQL query to the entire relational data source, performs any set-based operations dictated by the DAX formula, and returns the query results.

Change a DirectQuery Partition

Because only one partition in a table can be designated as the DirectQuery partition, by default, Analysis Services uses the first partition that was created in the table. During model project authoring, you can change the DirectQuery partition by using the Partition Manager dialog box in SQL Server Data Tools (SSDT). For deployed models, you can change the DirectQuery partition by using SQL Server Management Studio.

Change the DirectQuery partition for a tabular model project

1. In SQL Server Data Tools, in the model designer, click on the table (tab) that contains the partitioned table.

2. Click on the **Table** menu, and then click **Partitions**.

3. In **Partition Manager**, the partition that is the current Direct Query partition in indicated by the prefix **(DirectQuery)** on the partition name.

   Select a different partition from the **Partitions** list, and then click **Set as DirectQuery**. The **Set as DirectQuery** button is not enabled when the current DirectQuery partition is selected, and is not visible if the model has not been enabled for Direct Query mode.

4. If necessary, change the processing options, and then click **OK**.

Change the DirectQuery partition for a deployed tabular model

1. In SQL Server Management Studio, open the model database in Object Explorer.

2. Expand the **Tables** node, then right-click the partitioned table, and then select **Partitions**.

   The partition that is designated for use with DirectQuery mode has the prefix (DirectQuery) on the partition name.

3. To change to a different partition, click the **Direct Query** toolbar icon to open the **Set DirectQuery Partition** dialog box. The DirectQuery toolbar icon is not available on models that have not been enabled for Direct Query.

4. Choose a different partition from the **Partition Name** dropdown list, and then change processing options on the partition if necessary.

Partitions in Cached Models and in DirectQuery Models

When you configure a DirectQuery partition, you must specify processing options for the partition.

There are two processing options for the DirectQuery partition. To set this property, use the **Partition Manager** in SQL Server Data Tools (SSDT), or SQL Server Management Studio, and select the **Processing Option** property. The following table lists the values of this property, and describes the effects of each value when combined with the DirectQueryUsage property on the connection string:

CONNECTION STRING PROPERTY	PROCESSING OPTION PROPERTY	NOTES
DirectQuery	Never process this partition	When the model is using DirectQuery only, processing is never necessary. In hybrid models, you can configure the DirectQuery partition to never be processed. For example, if you are operating over a very large data set and do not want the full results added to the cache, you can specify that the DirectQuery partition include the union of results for all other partitions in the table, and then never process the union. Queries that go to the relational source will not be affected, and queries against cached data will combine data from the other partitions
DataView=Sample Applies to Tabular models using sample data views	Allow partition to be processed	If the model is using sample data, you can process the table to return a filtered dataset that provides visual cues during model design.

CONNECTION STRING PROPERTY	PROCESSING OPTION PROPERTY	NOTES
DirectQueryUsage=InMemory With DirectQuery Applies to Tabular 1100 or 1103 models running in a combination of in-memory and DirectQuery mode	Allow partition to be processed	If the model is using hybrid mode, you should use the same partition for queries against the in-memory and DirectQuery data source.

See also

Partitions

Test a model in DirectQuery mode

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Review your options for testing a tabular model in DirectQuery mode at each stage of development, starting with design.

Test in Excel

When designing your model in SSDT, you can use the **Analyze in Excel** feature to test your modeling decisions against a sample dataset in-memory, or against the relational database. When you click Analyze in Excel, a dialog box opens where you can specify options.

If your model's DirectQuery mode is on, you can specify DirectQuery connection mode, where you'll have two options:

- **Sample data view** - With this option, any queries from Excel are directed to sample partitions containing a sample dataset in-memory. This option is helpful when you want to make sure any DAX formulas you have in measures, calculated columns, or row-level security perform correctly.
- **Full data view** - With this option, any queries from Excel are sent to Analysis Services, and then on to the relational database. This option is, in-effect, fully functioning DirecQuery mode.

Other clients

When you use Analyze in Excel, an .odc connection file is created. You can use the connection string information from this file to connect to your model from other client applications. An additional parameter, DataView=Sample, is added to specify the client should connect to sample data partitions.

Monitor query execution on backend systems using xEvents or SQL Profiler

Start up a session trace, connected to the SQL Server relational database, to monitor connections coming from the Tabular model:

- Monitor Analysis Services with SQL Server Extended Events
- Use SQL Server Profiler to Monitor Analysis Services

If you're using Oracle or Teradata, use the trace monitoring tools for those systems.

DAX formula compatibility in DirectQuery mode

12/7/2018 • 17 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

For tabular 1200 and higher models in DirectQuery mode, many functional limitations in earlier versions no longer apply. For DAX formulas in-particular:

- DirectQuery now generates simpler queries, providing improved performance.
- Row level security (RLS) is now supported in DirectQuery mode.
- Calculated columns are now supported for tabular models in DirectQuery mode.

DAX functions in DirectQuery mode

In short, all DAX functions are supported for DirectQuery models. However, not all functions are supported for all formula types, and not all functions have been optimized for DirectQuery models. At the most basic level, we can put DAX functions into two camps: Optimized and Non-optimized. Let's first take a closer look at optimized functions.

Optimized for DirectQuery

These are functions that primarily return scalar or aggregate results. These functions are further divided into those that are supported in all types of formulas: measures, queries, calculated columns, row level security, and those that are supported in measure and query formulas only. These include:

SUPPORTED IN ALL DAX FORMULAS	SUPPORTED IN MEASURE AND QUERY FORMULAS ONLY
ABS	ALL
ACOS	ALLEXCEPT
ACOT	ALLNOBLANKROW
AND	ALLSELECTED
ASIN	AVERAGE
ATAN	AVERAGEA
BLANK	AVERAGEX
CEILING	CALCULATE
CONCATENATE	CALCULATETABLE
COS	COUNT
COT	COUNTA
CURRENCY	COUNTAX
DATE	COUNTROWS
DATEDIFF	COUNTX
DATEVALUE	DISTINCT
DAY	DISTINCTCOUNT
DEGREES	FILTER
DIVIDE	FILTERS
EDATE	HASONEFILTER
EOMONTH	HASONEVALUE
EXACT	ISCROSSFILTERED
EXP	ISFILTERED
FALSE	MAXA
FIND	MAXX
HOUR	MIN
IF	MINA
INT	MINX
ISBLANK	RELATEDTABLE
ISO.CEILING	STDEV.P
KEEPFILTERS	STDEV.S

SUPPORTED IN ALL DAX FORMULAS	SUPPORTED IN MEASURE AND QUERY FORMULAS ONLY
LEFT	STDEVX.P
LEN	STDEVX.S
LN	SUM
LOG	SUMX
LOG10	VALUES
LOWER	VAR.P
MAX	VAR.S
MID	VARX.P
MIN	VARX.S
MINUTE	
MOD	
MONTH	
MROUND	
NOT	
NOW	
OR	
PI	
POWER	
QUOTIENT	
RADIANS	
RAND	
RELATED	
REPT	
RIGHT	
ROUND	
ROUNDDOWN	
ROUNDUP	
SEARCH	
SECOND	
SIGN	
SIN	
SQRT	
SQRTPI	
SUBSTITUTE	
SWITCH	
TAN	
TIME	
TIMEVALUE	
TODAY	
TRIM	
TRUE	
TRUNC	
UNICODE	
UPPER	
USERNAME	
USERELATIONSHIP	
VALUE	
WEEKDAY	
WEEKNUM	
YEAR	

Non-optimized for DirectQuery

These functions have not been optimized to work with DirectQuery. These functions *are not* supported in calculated column and row-level security formulas at all. However, these functions *are supported* in measure and query formulas, albeit with uncertain performance.

We're not going to list all of the functions here. Basically, if it's not in one of the lists of optimized functions above, it's a non-optimized function for DirectQuery.

The reasons a particular function might not be optimized for DirectQuery is because the underlying relational engine cannot perform calculations equivalent to those performed by the xVelocity engine, or the formula cannot be converted to an equivalent SQL expression. In other cases, the performance of the converted expression and

the resulting calculations may be unacceptable.

To learn about all DAX functions, see the [DAX Function Reference]. (https://msdn.microsoft.com/library/ee634396.aspx)

DAX operators in DirectQuery mode

All DAX comparison and arithmetic operators are fully supported in DirectQuery mode. To learn more, see DAX Operator Reference.

Differences between in-memory and DirectQuery mode

Queries on a model deployed in DirectQuery mode can return different results than the same model deployed in in-memory mode. This is because with DirectQuery, data is queried directly from a relational data store and aggregations required by formulas are performed using the relevant relational engine (SQL, Oracle, Teradata), rather than using the xVelocity in-memory analytics engine for storage and calculation.

For example, there are differences in the way that certain relational data stores handle numeric values, dates, nulls, and so forth.

In contrast, the DAX language is intended to emulate as closely as possible the behavior of functions in Microsoft Excel. For example, when handling nulls, empty strings and zero values, Excel attempts to provide the best answer regardless of the precise data type, and therefore the xVelocity engine does the same. However, when a tabular model is deployed in DirectQuery mode and passes formulas to a relational data source, the data must be handled according to the semantics of the relational data source, which typically require distinct handling of empty strings vs. nulls. For this reason, the same formula might return a different result when evaluated against cached data and against data returned solely from the relational store.

Additionally, some functions aren't optimized for DirectQuery mode because the calculation would require the data in the current context be sent to the relational data source as a parameter. For example, measures using time-intelligence functions that reference date ranges in a calendar table. A relational data source might not have a calendar table, or at least one with .

Semantic differences

This section lists the types of semantic differences that you can expect, and describes any limitations that might apply to the usage of functions or to query results.

Comparisons

DAX in in-memory models support comparisons of two expressions that resolve to scalar values of different data types. However, models that are deployed in DirectQuery mode use the data types and comparison operators of the relational engine, and therefore might return different results.

The following comparisons will always return an error when used in a calculation on a DirectQuery data source:

- Numeric data type compared to any string data type
- Numeric data type compared to a Boolean value
- Any string data type compared to a Boolean value

In general, DAX is more forgiving of data type mismatches in in-memory models and will attempt an implicit cast of values up to two times, as described in this section. However, formulas sent to a relational data store in DirectQuery mode are evaluated more strictly, following the rules of the relational engine, and are more likely to fail.

Comparisons of strings and numbers

EXAMPLE: `"2" < 3`

The formula compares a text string to a number. The expression is **true** in both DirectQuery mode and in-memory models.

In an in-memory model, the result is **true** because numbers as strings are implicitly cast to a numerical data type for comparisons with other numbers. SQL also implicitly casts text numbers as numbers for comparison to numerical data types.

Note that this represents a change in behavior from the first version of Power Pivot, which would return **false**, because the text "2" would always be considered larger than any number.

Comparison of text with Boolean

EXAMPLE: `"VERDADERO" = TRUE`

This expression compares a text string with a Boolean value. In general, for DirectQuery or In-Memory models, comparing a string value to a Boolean value results in an error. The only exceptions to the rule are when the string contains the word **true** or the word **false**; if the string contains any of true or false values, a conversion to Boolean is made and the comparison takes place giving the logical result.

Comparison of nulls

EXAMPLE: `EVALUATE ROW("X", BLANK() = BLANK())`

This formula compares the SQL equivalent of a null to a null. It returns **true** in in-memory and DirectQuery models; a provision is made in DirectQuery model to guarantee similar behavior to in-memory model.

Note that in Transact-SQL, a null is never equal to a null. However, in DAX, a blank is equal to another blank. This behavior is the same for all in-memory models. It is important to note that DirectQuery mode uses, most of, the semantics of SQL Server; but, in this case it separates from it giving a new behavior to NULL comparisons.

Casts

There is no cast function as such in DAX, but implicit casts are performed in many comparison and arithmetic operations. It is the comparison or arithmetic operation that determines the data type of the result. For example,

- Boolean values are treated as numeric in arithmetic operations, such as TRUE + 1, or the function MIN applied to a column of Boolean values. A NOT operation also returns a numeric value.
- Boolean values are always treated as logical values in comparisons and when used with EXACT, AND, OR, &&, or ||.

Cast from string to Boolean

In in-memory and DirectQuery models, casts are permitted to Boolean values from these strings only: **""** (empty string), **"true"**, **"false"**; where an empty string casts to false value.

Casts to the Boolean data type of any other string results in an error.

Cast from string to date/time

In DirectQuery mode, casts from string representations of dates and times to actual **datetime** values behave the same way as they do in SQL Server.

Models that use the in-memory data store support a more limited range of text formats for dates than the string formats for dates that are supported by SQL Server. However, DAX supports custom date and time formats.

Cast from string to other non Boolean values

When casting from strings to non-Boolean values, DirectQuery mode behaves the same as SQL Server. For more information, see CAST and CONVERT (Transact-SQL).

Cast from numbers to string not allowed

EXAMPLE: `CONCATENATE(102,",345")`

Casting from numbers to strings is not allowed in SQL Server.

This formula returns an error in tabular models and in DirectQuery mode; however, the formula produces a result in Power Pivot.

No support for two-try casts in DirectQuery

In-memory models often attempt a second cast when the first one fails. This never happens in DirectQuery mode.

EXAMPLE: `TODAY() + "13:14:15"`

In this expression, the first parameter has type **datetime** and second parameter has type **string**. However, the casts when combining the operands are handled differently. DAX will perform an implicit cast from **string** to **double**. In in-memory models, the formula engine attempts to cast directly to **double**, and if that fails, it will try to cast the string to **datetime**.

In DirectQuery mode, only the direct cast from **string** to **double** will be applied. If this cast fails, the formula will return an error.

Math functions and arithmetic operations

Some mathematical functions will return different results in DirectQuery mode because of differences in the underlying data type or the casts that can be applied in operations. Also, the restrictions described above on the allowed range of values might affect the outcome of arithmetic operations.

Order of addition

When you create a formula that adds a series of numbers, an in-memory model might process the numbers in a different order than a DirectQuery model. Therefore, when you have many very large positive numbers and very large negative numbers, you may get an error in one operation and results in another operation.

Use of the POWER function

EXAMPLE: `POWER(-64, 1/3)`

In DirectQuery mode, the POWER function cannot use negative values as the base when raised to a fractional exponent. This is the expected behavior in SQL Server.

In an in-memory model, the formula returns -4.

Numerical overflow operations

In Transact-SQL, operations that result in a numerical overflow return an overflow error; therefore, formulas that result in an overflow also raise an error in DirectQuery mode.

However, the same formula when used in an in-memory model returns an eight-byte integer. That is because the formula engine does not perform checks for numerical overflows.

LOG functions with blanks return different results

SQL Server handles nulls and blanks differently than the xVelocity engine. As a result, the following formula returns an error in DirectQuery mode, but return infinity (-inf) in in-memory mode.

```
EXAMPLE: LOG(blank())
```

The same limitations apply to the other logarithmic functions: LOG10 and LN.

For more information about the **blank** data type in DAX, see DAX Syntax Reference.

Division by 0 and division by Blank

In DirectQuery mode, division by zero (0) or division by BLANK will always result in an error. SQL Server does not support the notion of infinity, and because the natural result of any division by 0 is infinity, the result is an error. However, SQL Server supports division by nulls, and the result must always equal null.

Rather than return different results for these operations, in DirectQuery mode, both types of operations (division by zero and division by null) return an error.

Note that, in Excel and in Power Pivot models, division by zero also returns an error. Division by a blank returns a blank.

The following expressions are all valid in in-memory models, but will fail in DirectQuery mode:

`1/BLANK`

`1/0`

`0.0/BLANK`

`0/0`

The expression `BLANK/BLANK` is a special case that returns `BLANK` in both for in-memory models, and in DirectQuery mode.

Supported numeric and date-time ranges

Formulas in in-memory tabular model are subject to the same limitations as Excel with regard to maximum allowed values for real numbers and dates. However, differences can arise when the maximum value is returned from a calculation or query, or when values are converted, cast, rounded, or truncated.

- If values of types **Currency** and **Real** are multiplied, and the result is larger than the maximum possible value, in DirectQuery mode, no error is raised, and a null is returned.
- In in-memory models, no error is raised, but the maximum value is returned.

In general, because the accepted date ranges are different for Excel and SQL Server, results can be guaranteed to match only when dates are within the common date range, which is inclusive of the following dates:

- Earliest date: March 1, 1990
- Latest date: December 31, 9999

If any dates used in formulas fall outside this range, either the formula will result in an error, or the results will not match.

Floating point values supported by CEILING

EXAMPLE: `EVALUATE ROW("x", CEILING(-4.398488E+30, 1))`

The Transact-SQL equivalent of the DAX CEILING function only supports values with magnitude of 10^19 or less. A rule of thumb is that floating point values should be able to fit into **bigint**.

Datepart functions with dates that are out of range

Results in DirectQuery mode are guaranteed to match those in in-memory models only when the date used as the argument is in the valid date range. If these conditions are not satisfied, either an error will be raised, or the formula will return different results in DirectQuery than in in-memory mode.

EXAMPLE: `MONTH(0)` or `YEAR(0)`

In DirectQuery mode, the expressions return 12 and 1899, respectively.

In in-memory models, the expressions return 1 and 1900, respectively.

EXAMPLE: `EOMONTH(0.0001, 1)`

The results of this expression will match only when the data supplied as a parameter is within the valid date range.

EXAMPLE: `EOMONTH(blank(), blank())` or `EDATE(blank(), blank())`

The results of this expression should be the same in DirectQuery mode and in-memory mode.

Truncation of time values

EXAMPLE: `SECOND(1231.04097222222)`

In DirectQuery mode, the result is truncated, following the rules of SQL Server, and the expression evaluates to 59.

In in-memory models, the results of each interim operation are rounded; therefore, the expression evaluates to 0.

The following example demonstrates how this value is calculated:

1. The fraction of the input (0.04097222222) is multiplied by 24.
2. The resulting hour value (0.98333333328) is multiplied by 60.
3. The resulting minute value is 58.9999999968.
4. The fraction of the minute value (0.9999999968) is multiplied by 60.
5. The resulting second value (59.999999808) rounds up to 60.
6. 60 is equivalent to 0.

SQL Time data type not supported

In-memory models do not support use of the new SQL **Time** data type. In DirectQuery mode, formulas that reference columns with this data type will return an error. Time data columns cannot be imported into an in-memory model.

However, sometimes the engine casts the time value to an acceptable data type, and the formula returns a result.

This behavior affects all functions that use a date column as a parameter.

Currency

In DirectQuery mode, if the result of an arithmetic operation has the type **Currency**, the value must be within the following range:

- Minimum: -922337203685477.5808
- Maximum: 922337203685477.5807

Combining currency and REAL data types

EXAMPLE: `Currency sample 1`

If **Currency** and **Real** types are multiplied, and the result is larger than 9223372036854774784 (0x7fffffffffffc00), DirectQuery mode will not raise an error.

In an in-memory model, an error is raised if the absolute value of the result is larger than 922337203685477.4784.

Operation results in an out-of-range value

EXAMPLE: `Currency sample 2`

If operations on any two currency values result in a value that is outside the specified range, an error is raised in in-memory models, but not in DirectQuery models.

Combining currency with other data types

Division of currency values by values of other numeric types can result in different results.

Aggregation functions

Statistical functions on a table with one row return different results. Aggregation functions over empty tables also behave differently in in-memory models than they do in DirectQuery mode.

Statistical functions over a table with a single row

If the table that is used as argument contains a single row, in DirectQuery mode, statistical functions such as STDEV and VARx return null.

In an in-memory model, a formula that uses STDEV or VARx over a table with a single row returns a division by zero error.

Text functions

Because relational data stores provide different text data types than does Excel, you may see different results when searching strings or working with substrings. The length of strings also can be different.

In general, any string manipulation functions that use fixed-size columns as arguments can have different results.

Additionally, in SQL Server, some text functions support additional arguments that are not provided in Excel. If the formula requires the missing argument you can get different results or errors in the in-memory model.

Operations that return a character using LEFT, RIGHT, etc. may return the correct character but in a different case, or no results

EXAMPLE: `LEFT(["text"], 2)`

In DirectQuery mode, the case of the character that is returned is always exactly the same as the letter that is stored in the database. However, the xVelocity engine uses a different algorithm for compression and indexing of values, to improve performance.

By default, the Latin1_General collation is used, which is case-insensitive but accent-sensitive. Therefore, if there are multiple instances of a text string in lower case, upper case, or mixed case, all instances are considered the same string, and only the first instance of the string is stored in the index. All text functions that operate on stored strings will retrieve the specified portion of the indexed form. Therefore, the example formula would return the same value for the entire column, using the first instance as the input.

This behavior also applies to other text functions, including RIGHT, MID, and so forth.

String length affects results

EXAMPLE: `SEARCH("within string", "sample target text", 1, 1)`

If you search for a string using the SEARCH function, and the target string is longer than the within string, DirectQuery mode raises an error.

In an in-memory model, the searched string is returned, but with its length truncated to the length of <within text>.

EXAMPLE: `EVALUATE ROW("X", REPLACE("CA", 3, 2, "California") )`

If the length of the replacement string is greater than the length of the original string, in DirectQuery mode, the formula returns null.

In in-memory models, the formula follows the behavior of Excel, which concatenates the source string and the replacement string, which returns CACalifornia.

Implicit TRIM in the middle of strings

EXAMPLE: `TRIM(" A sample sentence with leading white space")`

DirectQuery mode translates the DAX TRIM function to the SQL statement `LTRIM(RTRIM(<column>))`. As a result, only leading and trailing white space is removed.

In contrast, the same formula in an in-memory model removes spaces within the string, following the behavior of Excel.

Implicit RTRIM with use of LEN function

EXAMPLE: `LEN('string_column')`

Like SQL Server, DirectQuery mode automatically removes white space from the end of string columns: that is, it performs an implicit RTRIM. Therefore, formulas that use the LEN function can return different values if the string has trailing spaces.

In-memory supports additional parameters for SUBSTITUTE

EXAMPLE: `SUBSTITUTE([Title],"Doctor","Dr.")`

EXAMPLE: `SUBSTITUTE([Title],"Doctor","Dr.", 2)`

In DirectQuery mode, you can use only the version of this function that has three (3) parameters: a reference to a column, the old text, and the new text. If you use the second formula, an error is raised.

In in-memory models, you can use an optional fourth parameter to specify the instance number of the string to replace. For example, you can replace only the second instance, etc.

Restrictions on string lengths for REPT operations

In in-memory models, the length of a string resulting from an operation using REPT must be less than 32,767 characters.

This limitation does not apply in DirectQuery mode.

Substring operations return different results depending on character type

EXAMPLE: `MID([col], 2, 5)`

If the input text is **varchar** or **nvarchar**, the result of the formula is always the same.

However, if the text is a fixed-length character and the value for <*num_chars*> is greater than the length of the target string, in DirectQuery mode, a blank is added at the end of the result string.

In an in-memory model, the result terminates at the last string character, with no padding.

See also

DirectQuery Mode

Tables and Columns

12/7/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you have added tables and data into a model by using the Table Import Wizard, you can begin working with the tables by adding new columns of data, creating relationships between tables, defining calculations that extend the data, and filtering and sorting data in the tables for easier viewing.

Sections in this topic:

- Benefits
- Working with tables and columns
- Related Tasks

Benefits

Tables, in tabular models, provide the framework in which columns and other metadata are defined. Tables include:

Table Definition
The table definition includes the set of columns. Columns can be imported from a data source or added manually, such as with calculated columns.

Table Metadata
Relationships, measures, roles, perspectives, and pasted data are all metadata the define objects within the context of a table.

Data
Data is populated in table columns when you first import tables by using the Table Import Wizard or by creating new data in calculated columns. When data changes at the source, or when a model is removed from memory, you must run a process operation to re-populate the data into the tables.

Working with tables and columns

In the model designer, you do not create new model tables directly. A new tab is created automatically for you whenever data is imported or copied from another data source. Each tab (in the model designer) contains one table of data, which can include the following:

- A single table or view from a relational database, or from other non-relational sources, such as an Analysis Services cube.
- A tabular set of data imported from a feed or text file.
- A combination of both relational data and tabular (HTML) data copy and pasted into the table.

When you import data, each table or view, sheet, or file of data is added as a table in the model designer. You typically add data from various sources onto separate tabs, but you can combine data in a single table by using **Paste** and **Paste Append**. For more information, see Copy and Paste Data.

After you have added the data that you need, you can create additional relationships between the tables, look up or reference related values in other tables, or create derived values by adding new calculated columns.

If you are working very large data sets, you may want to filter out certain data so it is not visible. You may also

want to sort data in a different order. By using the model designer, you can use the filter, sort, and hide features to display, or not display, entire columns or certain data.

Related Tasks

TOPIC	DESCRIPTION
Add columns to a table	Describes how to add a source column to a table definition.
Delete a Column	Describes how to delete a model table column by using the model designer or by using the Table Properties dialog box.
Change table, column, or row filter mappings	Describes how to change table, column, or row filter mappings by using the table preview or SQL query editor in the Edit Table Properties dialog box.
Specify Mark as Date Table for use with Time Intelligence	Describes how to use the Mark as Date Table dialog box to specify a date table and unique identifier column. Specifying a date table and unique identifier is necessary when using time intelligence functions in DAX formulas.
Add a table	Describes how to add a table from a data source by using an existing data source connection.
Delete a Table	Describes how to delete tables in your model workspace database that you no longer need.
Rename a Table or Column	Describes how to rename a table or column to make it more identifiable in your model.
Set the Data Type of a Column	Describes how to change the data type of a column. The data type defines how data in the column is stored and presented.
Hide or freeze columns	Describes how to hide columns that you don't want to display and how to keep an area of a model visible while you scroll to another area of the model by freezing (locking) specific columns in one area.
Calculated Columns	Topics in this section describe how you can use calculated columns to add aggregated data to your model.
Filter and Sort Data	Topics in this section describe how you can filter or sort data by using controls in the model designer.

Add columns to a table

12/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes how to add columns to an existing table.

Add columns from the datasource

When using the Table Import Wizard to import data from a data source table, a new table is created in the model which includes all of the columns in the source table, or if you choose to filter out certain columns by using the Preview and Filter feature, only those columns and filtered data you select. You can also write a SQL Query that specifies only certain columns to import. You may, however, later determine a source table has additional columns that you want to add to the model table, or you need to add a calculated column with values derived from a DAX formula.

If, for example, when you initially imported from a data source, you used the Preview and Filter feature in the Table Import Wizard to select a limited number of columns from the source table, you later determine that you need to add another column that exists at the source table, but does not yet exist in the model table. Or, for example, a new AdjustedProfit column was added to the FactSales table at the data source, and you now want to add the same AdjustedProfit column and data to the Sales table in the model.

In these cases, you can use the Edit Table Properties dialog box to select columns from the source table and add them to the model table. The Edit Table Properties dialog box includes the table preview window. The table preview window displays the table at the source. Columns that are already included in the model table definition are already checked. Those columns that are not already included in the model table definition are not checked. You can add columns from the source to the model table definition by selecting the column and clicking OK. The table preview window in the Edit Table Properties dialog box provides the same view and features as the table preview window in the Preview and Filter page of the Table Import Wizard.

> **IMPORTANT**
> When adding a column to a table that contains two or more partitions, before adding the column to the table definition by using the Edit Table Properties dialog box, you must first use Partition Manager to add the column to all defined partitions. After you have added the column to the defined partitions, you can then add the same column to the table definition by using the Edit Table Properties dialog box.

> **NOTE**
> If you used a SQL Query to select tables and columns when you initially used the Table Import Wizard to import data, you must again use a SQL Query in the Edit Table Properties dialog box to add columns to the model table.

To add a column from the data source by using the Edit Table Properties dialog box

1. In the model designer, click on the table you want to add a column to, then click the **Table** menu, and then click **Table Properties**.

2. In the **Edit Table Properties** dialog box, in the table preview window, select the source column you want to add, and then click OK. Columns already included in the table definition will already be checked.

Add a calculated column

In a calculated column, a DAX formula is used to define a value for each row. For example, you can create a calculated column with a simple formula (=1) that adds a value of 1 to each row. Calculated columns can also have more complex formulas that calculate values based on other data in the model. Calculated columns are covered in more detail in other topics. For more information, see Calculated Columns.

To create a calculated column

1. In the model designer, in Data View, select the table to which you want to add a new, blank calculated column, scroll to the right-most column, or click the **Column** menu, and then click **Add Column**.

   To create a new column between two existing columns, right-click on an existing column, and then click **Insert Column**.

2. In the formula bar, type a DAX formula to add attributes for each row.

Add a blank column

You can create a named, blank column in a model table. Blank columns can be useful if you want to paste data from another source. Keep in-mind that pasted data is stored differently than imported data. For more information, see Copy and Paste Data.

To create a named, blank column

1. In the model designer, in Data View, select the table to which you want to add a blank column, scroll to the right-most column, or click the **Column** menu, and then click **Add Column**.

   To create a new column between two existing columns, right-click an existing column, and then click **Insert Column**.

2. Click on the top cell, then type a name, and then press ENTER.

See also

Edit table properties dialog box
Change table, column, or row filter mappings

Delete a Column

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes how to delete a column from a tabular model table.

Delete a Model Table Column

> **NOTE**
>
> Deleting a column from a model table does not delete the column from a partition query definition. If the column you are deleting is part of a partition, you must manually delete the column from the partition query definition. Failure to delete the column from the partition query definition will cause the column to be queried and data returned, but not populated to the model table, during processing operations. For more information, see Partitions.

To delete a model table column

- In the model designer, in the table that contains the column you want to delete, right-click on the column, and then click **Delete**.

To delete a model table column by using the Table Properties dialog box

1. In the model designer, click on the table which contains the column you want to delete, then click the **Table** menu, and then click **Table Properties**.
2. In **Column names from**, select **Model** (*to use model table column names, if different from source*).
3. In the **Edit Table Properties** dialog box, in the table preview window, uncheck the column you want to delete, and then click **OK**.

See Also

Add columns to a table
Partitions

Change table, column, or row filter mappings

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes how to change table, column, or row filter mappings by using the **Edit Table Properties** dialog box in SQL Server Data Tools.

Options in the **Edit Table Properties** dialog box are different depending on whether you originally imported data by selecting tables from a list or by using a SQL query. If you originally imported the data by selecting from a list, the **Edit Table Properties** dialog box displays the Table Preview mode. This mode displays only a subset limited to the first fifty rows of the source table. If you originally imported the data by using a SQL statement, the **Edit Table Properties** dialog box only displays a SQL statement. Using a SQL query statement, you can retrieve a subset of rows, either by designing a filter, or by manually editing the SQL statement.

If you change the source to a table that has different columns than the current table, a message is displayed warning that the columns are different. You must then select the columns that you want to put in the current table and click **Save**. You can replace the entire table by selecting the check box at the left of the table.

> **NOTE**
>
> If your table has more than one partition, you cannot use the Edit Table Properties dialog box to change row filter mappings. To change row filter mappings for tables with multiple partitions, use Partition Manager. For more information, see Partitions.

To change table, column, or row filter mappings

1. In the model designer, click the table, then click on the **Table** menu, and then click **Table Properties**.
2. In the **Edit Table Properties** dialog box, locate the column that contains the criteria you want to filter on, and then click the down arrow at the right of the column heading.
3. In the **AutoFilter** menu, do one of the following:
   - In the list of column values, select or clear one or more values to filter by, and then click OK.

     If the number of values is extremely large, individual items might not be shown in the list. Instead, you will see the message, "Too many items to show."
   - Click **Number Filters** or **Text Filters** (depending on the type of column), and then clicke of the comparison operator commands (such as Equals), or click Custom Filter. In the **Custom Filter** dialog box, create the filter, and then click **OK**.

     If you make a mistake and need to start over, click **Clear Row Filters**.

Specify Mark as Date Table for use with time-intelligence

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In order to use time-intelligence functions in DAX formulas, you must specify a date table and a unique identifier (datetime) column of the Date data type. Once a column in the date table is specified as a unique identifier, you can create relationships between columns in the date table and any fact tables.

When using time-intelligence functions, the following rules apply:

- When using DAX time-intelligence functions, never specify a datetime column from a fact table. Always create a separate date table in your model with at least one datetime column of Date data type and with unique values.
- Make sure your date table has a continuous date range.
- The datetime column in the date table should be at day granularity (without fractions of a day).
- You must specify a date table and a unique identifier column by using the **Mark the Date Table** dialog box.
- Create relationships between fact tables and columns of Date data type in the date table.

To specify a date table and unique identifier

1. In the model designer, click the date table.
2. Click the **Table** menu, then click **Date**, and then click **Mark as Date Table**
3. In the **Mark as Date Table** dialog box, in the **Date** listbox, select a column to be used as a unique identifier. This column must contain unique values and should be of Date data type. For example:

DATE
7/1/2010 12:00:00 AM
7/2/2010 12:00:00 AM
7/3/2010 12:00:00 AM
7/4/2010 12:00:00 AM
7/5/2010 12:00:00 AM

4. If necessary, create any relationships between fact tables and the date table.

See Also

Calculations
Time-intelligence Functions (DAX)

Add a table

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes how to add a table from a data source from which you have previously imported data into your model. To add a table from the same data source, you can use the existing data source connection. It is recommended you always use a single connection when importing any number of tables from a single data source.

To add a table from an existing data source

1. In SQL Server Data Tools (SSDT), click the **Model** menu, and then click **Existing Connections**.
2. On the **Existing Connections** page, select the connection to the data source that has the table you want to add, and then click **Open**.
3. On the **Select Tables and Views** page, select the table from the data source you want to add to your model.

   > **NOTE**
   > The **Select Tables and Views** page will not show tables that were previously imported as checked. If you select a table that was previously imported using this connection, and you did not give the table a different friendly name, a 1 will be appended to the friendly name. You do not need to re-select any tables that were previously imported by using this connection.

4. If necessary, use **Preview & Filter** to select only certain columns or apply filters to the data to be imported.
5. Click **Finish** to import the new table.

> **NOTE**
> When importing multiple tables at the same time from a single data source, any relationships between those tables at the source will automatically be created in the model. When adding a table later; however, you may need to manually create relationships in the model between newly added tables and the tables that were previously imported.

See also

Import Data
Delete a Table

Delete a table

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In the model designer, you can delete tables in your model workspace database that you no longer need. Deleting a table does not affect the original source data, only the data that you imported and were working with. You cannot undo the deletion of a table.

To delete a table

- Right-click the tab at the bottom of the model designer for the table that you want to delete, and then click **Delete**.

Considerations when Deleting Tables

- When you delete a table, the entire tab that the table was on is deleted.
- If any measures were associated with that table, the definition of the measure will also be deleted.
- If you created any calculated columns using that table, columns in that table are also deleted; any calculated columns in other tables that use columns from the deleted table will display an error.

See also

Tables and columns

Create a calculated table

12/19/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A *calculated table* is a computed object, based on either a DAX query or expression, derived from all or part of other tables in the same model.

A common design problem that calculated tables can solve is surfacing a role-playing dimension in a specific context so that you can expose it as a query structure in client applications. You might recall that a role-playing dimension is simply a table surfaced in multiple contexts -- a classic example is the Date table, manifested as OrderDate, ShipDate, or DueDate, depending on the foreign key relationship. By creating a calculated table for ShipDate explicitly, you get a standalone table that is available for queries, as fully operable as any other table. Another use includes configuring a filtered rowset, a subset, or superset of columns from other existing tables. This allows you to keep the original table intact while creating variations of that table to support specific scenarios.

Using calculated tables to best advantage will require that you know at least some DAX. As you work with expressions for your table, it might help to know that a calculated table contains a single partition with a DAXSource, where the expression is a DAX expression.
There is one CalculatedTableColumn for each column returned by the expression, where the SourceColumn is the name of the column returned (similar to DataColumns on non-calculated tables).

At least one table must already exist before you can create a calculated table. If you are creating a calculated table as a standalone computed table object, you can first create a table by importing from a file data source (csv, xls, xml). The file you import from can have a single column and single value. You can then hide that table.

How to create a calculated table

1. First, verify the tabular model has a compatibility level of 1200 or higher. You can check the **Compatibility Level** property on the model in SSDT.
2. Switch to the Data View. You can't create a calculated table in Diagram View.
3. Select **Table** > **New calculated table**.
4. Type or paste a DAX expression (see below for some ideas).
5. Name the table.
6. Create relationships to other tables in the model. See Create a Relationship Between Two Tables if you need help with this step.
7. Reference the table in calculations or expressions in your model or use **Analyze in Excel** for ad hoc data exploration.

Replicate a role-playing dimension

In the Formula bar, enter a DAX formula that gets a copy of another table. After the calculated table is populated, give it a descriptive name and then set up a relationship that uses the foreign key specific to the role. For example, in the Adventure Works database, you might create a calculated table for Due Date and use the DueDateKey as the basis of a relationship to the fact table.

```
=DimDate
```

Summarized or filtered dataset

In the Formula bar, enter a DAX expression that filters, summarizes, or otherwise manipulates a dataset to contain the rows you want. This example groups by sales by color and currency.

```
=SUMMARIZECOLUMNS(DimProduct[Color]
, DimCurrency[CurrencyName]
, "Sales" , SUM(FactInternetSales[SalesAmount])
)
```

Superset using columns from multiple tables

In the Formula bar, enter a DAX expression that combines columns from multiple tables. In this case, query output lists product category for each currency.

```
=CROSSJOIN(DimProductCategory, DimCurrency)
```

See also

Compatibility level
Data Analysis Expressions (DAX) in Analysis Services
Understanding DAX in tabular models

Rename a Table or Column

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can change the name of a table during the import process by typing a **Friendly Name** in the **Select Tables and Views** page of the **Table Import Wizard**. You can also change table and column names if you import data by specifying a query on the **Specify a SQL Query** page of the **Table Import Wizard**.

After you have added the data to the model, the name (or title) of a table appears on the table tab, at the bottom of the model designer. You can change the name of your table to give it a more appropriate name. You can also rename a column after the data has been added to the model. This option is especially important when you have imported data from multiple sources, and want to ensure that columns in different tables have names that are easy to distinguish.

To rename a table

1. In the model designer, right-click the tab that contains the table that you want to rename, and then click **Rename**.
2. Type the new name.

> **NOTE**
> You can edit other properties of a table, including the connection information and column mappings, by using the **Edit Table Properties** dialog box. However, you cannot change the name in this dialog box.

To rename a column

1. In the model designer, double-click the header of the column that you want to rename, or right-click the header and select **Rename Column** from the context menu.
2. Type the new name.

Naming Requirements for Columns and Tables

The following words and characters cannot be used in the name of a table or column:

- Leading or trailing spaces
- Control characters
- The following characters (which are not valid in the names of Analysis Services objects): .,;':/*|?&%$!+=()[]{}<>
- Analysis Services reserved keywords, including Multidimensional Expressions (MDX) and Data Mining Extensions (DMX) function names and operators.

Effect of Renaming on Existing Tables, Columns, and Calculations

Whenever you change the name of a table, you change the name of the underlying table object, which may contain multiple columns or measures. Any columns that are in the table, and any relationships that use the table, must be updated to use the new name in their definitions. This update happens automatically in most cases.

Any calculations that use the renamed table, or that use columns from the renamed table, must also be updated,

and the data derived from those calculations must be refreshed and recalculated. Depending on how many tables and calculations are affected, this can take some time to complete. Therefore, the best time to rename tables is either during the import process, or before you start to build complex relationships and calculations.

See Also

Tables and Columns
Import from Power Pivot
Import from Analysis Services

Set the Data Type of a Column

12/7/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you import data or paste data into a model, the model designer will automatically detect and apply data types. After you have added the data to the model, you can manually modify the data type of a column to change how data is stored. If you just want to change the format of how the data is displayed without changing the way it is stored, you can do that instead.

To change the data type or display format for a column

1. In the model designer, select the column for which you want to change the data type or display format.
2. In the column **Properties** window, do one of the following:
   - In the **Data Format** property, select a different data format.
   - In the **Data Type** property, select a different data type.

Considerations when Changing Data Types

Sometimes when you try to change the data type of a column or select a data conversion, one of the following errors might occur:

- Failed to change data type
- Failed to change column data type

These errors might occur even if the data type is available as an option in the Data Type dropdown list. This section explains the cause of these errors and how you can correct them.

Understanding Automatically Determined Data Types

When you add data to a model, the model designer checks the columns of data to see what data types each column contains. If the data in that column is consistent, is assigns the most precise data type to the column.

However, if you add data from Excel or another source that does not enforce the use of a single data type within each column, the model designer will assign a data type that accommodates all values within the column. Therefore, if a column contains numbers of different types, such as integers, long numbers, and currency, the model designer will use a decimal data type. Alternatively, if a column mixes numbers and text, the model designer will use the text data type. The model designer does not provide a data type similar to the General data type available in Excel.

Therefore, if a column contains both numbers and text values, you will not be able to convert the column to a numeric data type.

The following data types are available in business intelligence semantic models:

- **Text**
- **Decimal Number**
- **Whole Number**
- **Currency**
- **TRUE/FALSE**

- **Date**

If you find that your data has a wrong data type, or at least a different one than you wanted, you have several options:

- You can re-import the data. To do this, open the existing connection to the data source and re-import the column. Depending on the data source type, you might be able to apply a filter during import to remove problem values.
- You can create a DAX formula in a calculated column to create a new value of the desired data type. For example, the TRUNC function can be used to change a decimal number to a whole integer, or you can combine information functions and logical functions to test and convert values.

Understanding Data Conversion

If an error occurs when you select a data conversion option, it might be that the current data type of the column does not support the selected conversion. Not all conversions are allowed for all data types. For example, you can only change a column to a Boolean data type if the current data type of the column is either a number (whole or decimal) or text. Therefore, you must choose an appropriate data type for the data in the column.

After you choose an appropriate data type, the model designer will warn you about possible changes to your data, such as loss of precision, or truncation. Click OK to accept and change your data to the new data type.

If the data type is supported, but the model designer finds values that are not supported within the new data type, you will get another error, and will need to correct the data values before proceeding.

For detailed information about the data types used in business intelligence semantic models, how they are implicitly converted, and how different data types are used in formulas, see Data Types Supported.

See Also

Data Types Supported

Hide or freeze columns

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In the model designer, if there are columns that you don't want to display in a table, you can temporarily hide them. Hiding a column gives you more room on the screen to add new columns or to work with only relevant columns of data. You can hide and unhide columns from the **Column** menu in the model designer, and from the right-click menu available from each column header. To keep an area of a model visible while you scroll to another area of the model, you can lock specific columns in one area by freezing them.

> **IMPORTANT**
> The ability to hide columns is not intended to be used for data security, only to simplify and shorten the list of columns visible in the model designer or reports. To secure data, you can define security roles. Roles can limit viewable metadata and data to only those objects defined in the role. For more information, see Roles.

When you hide a column, you have the option to hide the column while you are working in the model designer or in reports. If you hide all columns, the entire table appears blank in the model designer.

> **NOTE**
> If you have many columns to hide, you can create a perspective instead of hiding and unhiding columns. A perspective is a custom view of the data that makes it easier to work with subset of related data. For more information, see Create and Manage Perspectives

To hide an individual column

1. In the model designer, select the table that contains the column that you want to hide.
2. Right-click the column, then click **Hide Columns**, and then click **From Designer and Reports**, **From Reports**, or **From Designer**.

To hide multiple columns

1. In the model designer, select the table that contains the columns that you want to hide.
2. Click on the **Columns** menu, and then click **Hide and Unhide**.
3. In the **Hide and Unhide Columns** dialog box, locate each column that you want to hide, and then deselect one or both of **In Designer** and **In Reports**.
4. Click **OK**.

To freeze columns

1. In the model designer, select the table that contains the columns that you want to freeze.
2. Select one or more columns to freeze.
3. Click on the **Columns** menu, and then click **Freeze**..

> **NOTE**
> When a column(s) is frozen, it is moved to left (or front) of the table in the designer. Unfreezing the column does not move it back to its original location.

See Also

Tables and Columns
Perspectives
Roles

Calculated Columns

12/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Calculated columns, in tabular models, enable you to add new data to your model. Instead of pasting or importing values into the column, you create a DAX formula that defines the column's row level values. The calculated column can then be used in a report, PivotTable, or PivotChart as would any other column.

Benefits

Formulas in calculated columns are much like formulas in Excel. Unlike Excel, however, you cannot create different formulas for different rows in a table; instead, the DAX formula is automatically applied to the entire column.

When a column contains a formula, the value is computed for each row. The results are calculated for the column when you enter a valid formula. Column values are then recalculated as necessary, such as when the underlying data is refreshed.

You can create calculated columns that are based on measures and other calculated columns. For example, you might create one calculated column to extract a number from a string of text, and then use that number in another calculated column.

A calculated column is based on data that you already have in an existing table, or created by using a DAX formula. For example, you might choose to concatenate values, perform addition, extract substrings, or compare the values in other fields. To add a calculated column, you must have at least one table in your model.

This example demonstrates a simple formula in a calculated column:

```
=EOMONTH([StartDate],0])
```

This formula extracts the month from the StartDate column. It then calculates the end of the month value for each row in the table. The second parameter specifies the number of months before or after the month in StartDate; in this case, 0 means the same month. For example, if the value in the StartDate column is 6/1/2001, the value in the calculated column will be 6/30/2001.

Naming a calculated column

By default, new calculated columns are added to the right of other columns in a table, and the column is automatically assigned the default name of **CalculatedColumn1**, **CalculatedColumn2**, and so forth. You can also right click a column, and then click Insert Column to create a new column between two existing columns. You can rearrange columns within the same table by clicking and dragging, and you can rename columns after they are created; however, you should be aware of the following restrictions on changes to calculated columns:

- Each column name must be unique within a table.
- Avoid names that have already been used for measures within the same model. Although it is possible for a measure and a calculated column to have the same name, if names are not unique you can get calculation errors. To avoid accidentally invoking a measure, when referring to a column always use a fully qualified column reference.

- When you rename a calculated column, any formulas that rely on the column must be updated manually. Unless you are in manual update mode, updating the results of formulas takes place automatically. However, this operation might take some time.
- There are some characters that cannot be used within the names of columns. For more information, see "Naming Requirements" in DAX Syntax Reference.

Performance of calculated columns

The formula for a calculated column can be more resource-intensive than the formula used for a measure. One reason is that the result for a calculated column is always calculated for each row in a table, whereas a measure is only calculated for the cells defined by the filter used in a report, PivotTable, or PivotChart. For example, a table with a million rows will always have a calculated column with a million results, and a corresponding effect on performance. However, a PivotTable generally filters data by applying row and column headings; therefore, a measure is calculated only for the subset of data in each cell of the PivotTable.

A formula has dependencies on the objects that are referenced in the formula, such as other columns or expressions that evaluate values. For example, a calculated column that is based on another column, or a calculation that contains an expression with a column reference, cannot be evaluated until the other column is evaluated. By default, automatic refresh is enabled in workbooks; therefore, all such dependencies can affect performance while values are updated and formulas refreshed.

To avoid performance issues when you create calculated columns, follow these guidelines:

- Rather than create a single formula that contains many complex dependencies, create the formulas in steps, with results saved to columns, so that you can validate the results and assess performance.
- Modification of data will often require that calculated columns be recalculated. You can prevent this by setting the recalculation mode to manual; however, if any values in the calculated column are incorrect the column will be grayed out until you refresh and recalculate the data.
- If you change or delete relationships between tables, formulas that use columns in those tables will become invalid.
- If you create a formula that contains a circular or self-referencing dependency, an error will occur.

Related tasks

TOPIC	DESCRIPTION
Create a Calculated Column	Tasks in this topic describe how to add a new calculated column to a table.

See also

Tables and Columns
Measures
Calculations

Create a Calculated Column

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Calculated columns allow you to add new data to your model. Instead of pasting or importing values into the column, you create a DAX formula that defines the column's row level values. The values in each row of a calculated column are calculated and populated when you create a valid formula and then click ENTER. The calculated column can then be added to a reporting or analysis application just as would any other column of data. This article describes how to create a new calculated column by using the DAX formula bar in the model designer.

To create a new calculated column

1. In the model designer, in Data View, select the table to which you want to add a calculated column, then click the **Column** menu, and then click **Add Column**.

   Add Column is highlighted over the empty rightmost column, and the cursor moves to the formula bar.

   To create a new column between two existing columns, right-click an existing column, and then click **Insert Column**.

2. In the formula bar, do one of the following:

   - Type an equal sign followed by a formula.
   - Type an equal sign, followed by a DAX function, followed by arguments and parameters as required by the function.
   - Click the function button (**fx**), then in the **Insert Function** dialog box, select a category and function, and then click **OK**. In the formula bar, type the remaining arguments and parameters as required by the function.

3. Press ENTER to accept the formula.

   The entire column is populated with the formula, and a value is calculated for each row.

> **TIP**
> You can use DAX Formula AutoComplete in the middle of an existing formula with nested functions. The text immediately before the insertion point is used to display values in the drop-down list, and all of the text after the insertion point remains unchanged.

See Also

Calculated Columns
Measures

Sort Data in a Table

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can sort data by text (A to Z or Z to A) and numbers (smallest to largest or largest to smallest) in one or more columns.

To sort the data in a table based on a text column

1. In the model designer, select a column of alphanumeric data, a range of cells in a column, or make sure that the active cell is in a table column that contains alphanumeric data, and then click the arrow in the header of the column that you want to filter by.

2. In the AutoFilter menu, do one of the following:

   - To sort in ascending alphanumeric order, click **Sort A to Z**.

   - To sort in descending alphanumeric order, click **Sort Z to A**.

   > **NOTE**
   > In some cases, data imported from another application might have leading spaces inserted before data. You must remove the leading spaces in order to correctly sort the data.

To sort the data in a table based on a numeric column

1. In the model designer, select a column of alphanumeric data, a range of cells in a column, or make sure that the active cell is in a table column that contains alphanumeric data, and then click the arrow in the header of the column that you want to filter by.

2. In the AutoFilter menu, do one of the following:

   - To sort from low numbers to high numbers, click **Sort Smallest to Largest**.

   - To sort from high numbers to low numbers, click **Sort Largest to Smallest**.

   > **NOTE**
   > If the results are not what you expected, the column might contain numbers stored as text and not as numbers. For example, negative numbers imported from some accounting systems, or a number entered with a leading ' (apostrophe), are stored as text.

See Also

Filter and Sort Data
Perspectives
Roles

Filter Data in a Table

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can apply filters when you import data to control the rows that are loaded into a table. After you have imported the data, you cannot delete individual rows. However, you can apply custom filters to control the way that rows are displayed. Rows that do not meet the filtering criteria are hidden. You can filter by one or more columns. Filters are additive, which means that each additional filter is based on the current filter and further reduces the subset of data.

> **NOTE**
>
> The filter preview window limits the number of different values displayed. If the limit is exceeded, a message is displayed.

To filter data based on column values

1. In the model designer, select a table, and then click the arrow in the header of the column that you want to filter by.
2. In the AutoFilter menu, do one of the following:
   - In the list of column values, select or clear one or more values to filter by, and then click **OK**.

     If the number of values is extremely large, individual items might not be shown in the list. Instead, you will see the message, "Too many items to show."
   - Click **Number Filters** or **Text Filters** (depending on the type of column), and then clicke of the comparison operator commands (such as **Equals**), or click **Custom Filter**. In the **Custom Filter** dialog box, create the filter, and then click **OK**.

To clear a filter for a column

1. Click the arrow in the header of the column for which you want to clear a filter.
2. Click **Clear Filter from <Column Name>**.

To clear all filters for a table

1. In the model designer, select the table for which you want to clear filters.
2. Click on the **Column** menu, and then click **Clear All Filters**.

See Also

Filter and Sort Data
Perspectives
Roles

Relationships

12/7/2018 • 11 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In tabular models, a relationship is a connection between two tables of data. The relationship establishes how the data in the two tables should be correlated. For example, a Customers table and an Orders table can be related in order to show the customer name that is associated with each order.

When using the Table Import Wizard to import from the same data source, relationships that already exist in tables (at the data source) that you choose to import will be re-created in the model. You can view relationships that were detected and re-created automatically by using the model designer in Diagram View or by using the Manage Relationships dialog box. You can also create new relationships between tables manually by using the model designer in Diagram View or by using the Create Relationship or Manage Relationships dialog box.

After relationships between tables have been defined, either automatically during import or created manually, you will be able to filter data by using related columns and look up values in related tables.

> **TIP**
> If your model contains many relationships, Diagram View can better help you better visualize and create new relationships between tables.

Benefits

A relationship is a connection between two tables of data, based on one or more columns in each table. To see why relationships are useful, imagine that you track data for customer orders in your business. You could track all the data in a single table that has a structure like the following:

CUSTOMERID	NAME	EMAIL	DISCOUNTRATE	ORDERID	ORDERDATE	PRODUCT	QUANTITY
1	Ashton	chris.ashton@contoso.com	.05	256	2010-01-07	Compact Digital	11
1	Ashton	chris.ashton@contoso.com	.05	255	2010-01-03	SLR Camera	15
2	Jaworski	michal.jaworski@contoso.com	.10	254	2010-01-03	Budget Movie-Maker	27

This approach can work, but it involves storing a lot of redundant data, such as the customer e-mail address for every order. Storage is cheap, but you have to make sure you update every row for that customer if the e-mail address changes. One solution to this problem is to split the data into multiple tables and define relationships between those tables. This is the approach used in *relational databases* like SQL Server. For example, a database that you import into a model might represent order data by using three related tables:

Customers

[CUSTOMERID]	NAME	EMAIL
1	Ashton	chris.ashton@contoso.com
2	Jaworski	michal.jaworski@contoso.com

CustomerDiscounts

[CUSTOMERID]	DISCOUNTRATE
1	.05
2	.10

Orders

[CUSTOMERID]	ORDERID	ORDERDATE	PRODUCT	QUANTITY
1	256	2010-01-07	Compact Digital	11
1	255	2010-01-03	SLR Camera	15
2	254	2010-01-03	Budget Movie-Maker	27

If you import these tables from the same database, the Table Import Wizard can detect the relationships between the tables based on the columns that are in [brackets], and can reproduce these relationships in the model designer. For more information, see Automatic Detection and Inference of Relationships in this topic. If you import tables from multiple sources, you can manually create relationships as described in Create a Relationship Between Two Tables.

Columns and keys

Relationships are based on columns in each table that contain the same data. For example, the Customers and Orders tables can be related to each other because they both contain a column that stores a customer ID. In the example, the column names are the same, but this is not a requirement. One could be CustomerID and another CustomerNumber, as long as all of the rows in the Orders table contain an ID that is also stored in the Customers table.

In a relational database, there are several types of *keys*, which are typically just columns with special properties. The following four types of keys can be used in relational databases:

- *Primary key*: uniquely identifies a row in a table, such as CustomerID in the Customers table.
- *Alternate key* (or *candidate key*): a column other than the primary key that is unique. For example, an Employees table might store an employee ID and a social security number, both of which are unique.
- *Foreign key*: a column that refers to a unique column in another table, such as CustomerID in the Orders table, which refers to CustomerID in the Customers table.
- *Composite key*: a key that is composed of more than one column. Composite keys are not supported in tabular models. For more information, see "Composite Keys and Lookup Columns" in this topic.

In tabular models, the primary key or alternate key is referred to as the *related lookup column*, or just *lookup column*. If a table has both a primary and alternate key, you can use either as the lookup column. The foreign key is referred to as the *source column* or just *column*. In our example, a relationship would be defined between CustomerID in the Orders table (the column) and CustomerID (the lookup column) in the Customers table. If you

import data from a relational database, by default the model designer chooses the foreign key from one table and the corresponding primary key from the other table. However, you can use any column that has unique values for the lookup column.

Types of relationships

The relationship between Customers and Orders is a *one-to-many relationship*. Every customer can have multiple orders, but an order cannot have multiple customers. The other types of relationships are *one-to-one* and *many-to-many*. The CustomerDiscounts table, which defines a single discount rate for each customer, is in a one-to-one relationship with the Customers table. An example of a many-to-many relationship is a direct relationship between Products and Customers, in which a customer can buy many products and the same product can be bought by many customers. The model designer does not support many-to-many relationships in the user interface. For more information, see "Many-to-Many Relationships" in this topic.

The following table shows the relationships between the three tables:

RELATIONSHIP	TYPE	LOOKUP COLUMN	COLUMN
Customers-CustomerDiscounts	one-to-one	Customers.CustomerID	CustomerDiscounts.CustomerID
Customers-Orders	one-to-many	Customers.CustomerID	Orders.CustomerID

Relationships and performance

After any relationship has been created, the model designer typically must recalculate any formulas that use columns from tables in the newly created relationship. Processing can take some time depending on the amount of data and the complexity of the relationships.

Requirements for relationships

The model designer has several requirements that must be followed when creating relationships:

Single Active Relationship between tables

Multiple relationships could result in ambiguous dependencies between tables. To create accurate calculations, you need a single path from one table to the next. Therefore, there can be only one active relationship between each pair of tables. For example, in AdventureWorks DW 2012, the table, DimDate, contains a column, DateKey, that is related to three different columns in the table, FactInternetSales: OrderDate, DueDate, and ShipDate. If you attempt to import these tables, the first relationship is created successfully, but you will receive the following error on successive relationships that involve the same column:

* Relationship: table[column 1]-> table[column 2] - Status: error - Reason: A relationship cannot be created between tables <table 1> and <table 2>. Only one direct or indirect relationship can exist between two tables.

If you have two tables and multiple relationships between them, then you will need to import multiple copies of the table that contains the lookup column, and create one relationship between each pair of tables.

There can be many inactive relationships between tables. The path to use between tables is specified by the reporting client at query time.

One relationship for each source column

A source column cannot participate in multiple relationships. If you have used a column as a source column in one relationship already, but want to use that column to connect to another related lookup column in a different table, you can create a copy of the column, and use that column for the new relationship.

It is easy to create a copy of a column that has the exact same values, by using a DAX formula in a calculated column. For more information, see Create a Calculated Column.

Unique identifier for each table

Each table must have a single column that uniquely identifies each row in that table. This column is often referred to as the primary key.

Unique lookup columns

The data values in the lookup column must be unique. In other words, the column cannot contain duplicates. In Tabular models, nulls and empty strings are equivalent to a blank, which is a distinct data value. This means that you cannot have multiple nulls in the lookup column.

Compatible data types

The data types in the source column and lookup column must be compatible. For more information about data types, see Data Types Supported.

Composite keys and lookup columns

You cannot use composite keys in a tabular model; you must always have one column that uniquely identifies each row in the table. If you try to import tables that have an existing relationship based on a composite key, the Table Import Wizard will ignore that relationship because it cannot be created in the tabular model.

If you want to create a relationship between two tables in the model designer, and there are multiple columns defining the primary and foreign keys, you must combine the values to create a single key column before creating the relationship. You can do this before you import the data, or you can do this in the model designer by creating a calculated column.

Many-to-Many relationships

Tabular models do not support many-to-many relationships, and you cannot add *junction tables* in the model designer. However, you can use DAX functions to model many-to-many relationships.

You can also try setting up a bi-directional cross filter to see if it achieves the same purpose. Sometimes the requirement of many-to-many relationship can be satisfied through cross filters that persist a filter context across multiple table relationships. See Bi-directional cross filters for tabular models in SQL Server 2016 Analysis Services for details.

Self-joins and loops

Self-joins are not permitted in tabular model tables. A self-join is a recursive relationship between a table and itself. Self-joins are often used to define parent-child hierarchies. For example, you could join an Employees table to itself to produce a hierarchy that shows the management chain at a business.

The model designer does not allow loops to be created among relationships in a model. In other words, the following set of relationships is prohibited.

Table 1, column a to Table 2, column f

Table 2, column f to Table 3, column n

Table 3, column n to Table 1, column a

If you try to create a relationship that would result in a loop being created, an error is generated.

Inference of relationships

In some cases, relationships between tables are automatically chained. For example, if you create a relationship between the first two sets of tables below, a relationship is inferred to exist between the other two tables, and a relationship is automatically established.

Products and Category -- created manually

Category and SubCategory -- created manually

Products and SubCategory -- relationship is inferred

In order for relationships to be automatically chained, the relationships must go in one direction, as shown above. If the initial relationships were between, for example, Sales and Products, and Sales and Customers, a relationship is not inferred. This is because the relationship between Products and Customers is a many-to-many relationship.

Detection of relationships when importing data

When you import from a relational data source table, the Table Import Wizard detects existing relationships in those source tables based on the source schema data. If related tables are imported, those relationships will be duplicated in the model.

Manually create relationships

While most relationships between tables in a single relational data source will be detected automatically, and created in the tabular model, there are also many instances where you must manually create relationships between model tables.

If your model contains data from multiple sources, you will likely have to manually create relationships. For example, you may import Customers, CustomerDiscounts, and Orders tables from a relational data source. Relationships existing between those tables at the source are automatically created in the model. You may then add another table from a different source, for example, you import region data from a Geography table in a Microsoft Excel workbook. You can then manually create a relationship between a column in the Customers table and a column in the Geography table.

To manually create relationships in a tabular model, you can use the model designer in Diagram View or by using the Manage Relationships dialog box. The diagram view displays tables, with relationships between them, in a graphical format. You can click a column in one table and drag the cursor to another table to easily create a relationship, in the correct order, between the tables. The Manage Relationships dialog box displays relationships between tables in a simple table format. To learn how to manually create relationships, see Create a Relationship Between Two Tables.

Duplicate values and other errors

If you choose a column that cannot be used in the relationship, a red X appears next to the column. You can pause the cursor over the error icon to view a message that provides more information about the problem. Problems that can make it impossible to create a relationship between the selected columns include the following:

PROBLEM OR MESSAGE	RESOLUTION
The relationship cannot be created because both columns selected contain duplicate values.	To create a valid relationship, at least one column of the pair that you select must contain only unique values. You can either edit the columns to remove duplicates, or you can reverse the order of the columns so that the column that contains the unique values is used as the **Related Lookup Column**.
The column contains a null or empty value.	Data columns cannot be joined to each other on a null value. For every row, there must be a value in both of the columns that are used in a relationship.

Related tasks

TOPIC	DESCRIPTION
Create a Relationship Between Two Tables	Describes how to manually create a relationship between two tables.
Delete Relationships	Describes how to delete a relationship and the ramifications of deleting relationships.
Bi-directional cross filters for tabular models in SQL Server 2016 Analysis Services	Describes two-way cross filtering for related tables. A filter context of one table relationship can be used when querying across a second table relationship if tables are related and bi-directional cross filters are define.

See also

Tables and Columns
Import Data

Create a relationship

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If the tables in your data source do not have existing relationships, or if you add new tables, you can use the tools in the model designer to create new relationships. For information about how relationships are used in tabular models, see Relationships.

Create a relationship between two tables

To create a relationship between two tables in Diagram View (Click and drag)

1. In SQL Server Data Tools, click the **Model** menu, then click **Model View**, and then click **Diagram View**.
2. Click (and hold) on a column within a table, then drag the cursor to a related lookup column in a related lookup table, and then release. The relationship will be created in the correct order automatically.

To create a relationship between two tables in Diagram View (Right-click)

1. In SQL Server Data Tools, click the **Model** menu, then click **Model View**, and then click **Diagram View**.
2. Right-click a table heading or column, and then click **Create Relationship**.
3. In the **Create Relationship** dialog box, click the down arrow for **Table**, and select a table from the dropdown list.

   In a "one-to-many" relationship, this table should be on the "many" side.
4. For **Column**, select the column that contains the data that is related to **Related Lookup Column**. The column is automatically selected if you right-clicked on a column to create the relationship.
5. For **Related Lookup Table**, select a table that has at least one column of data that is related to the table you just selected for **Table**.

   In a "one-to-many" relationship, this table should be on the "one" side, meaning that the values in the selected column do not contain duplicates. If you attempt to create the relationship in the wrong order (one-to-many instead of many-to-one), an icon will appear next to the **Related Lookup Column** field. Reverse the order to create a valid relationship.
6. For **Related Lookup Column**, select a column that has unique values that match the values in the column you selected for **Column**.
7. Click **Create**.

To create a relationship between two tables in Data View

1. In SQL Server Data Tools, click the **Table** menu, and then click **Create Relationships**.
2. In the **Create Relationship** dialog box, click the down arrow for **Table**, and select a table from the dropdown list.

   In a "one-to-many" relationship, this table should be on the "many" side.
3. For **Column**, select the column that contains the data that is related to **Related Lookup Column**.
4. For **Related Lookup Table**, select a table that has at least one column of data that is related to the table you just selected for **Table**.

In a "one-to-many" relationship, this table should be on the "one" side, meaning that the values in the selected column do not contain duplicates. If you attempt to create the relationship in the wrong order (one-to-many instead of many-to-one), an icon will appear next to the **Related Lookup Column** field. Reverse the order to create a valid relationship.

5. For **Related Lookup Column**, select a column that has unique values that match the values in the column you selected for **Column**.

6. Click **Create**.

See also

Delete relationships
Relationships

Delete relationships

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can delete existing relationships by using the model designer in Diagram View or by using the Manage Relationships dialog box. For information about how relationships are used in tabular models, see Relationships.

Considerations for deleting relationships

Keep the following issues in mind when deciding whether to delete a relationship:

- There is no way to undo the deletion of a relationship. You can re-create the relationship, but this action requires a complete recalculation of formulas in the model. Therefore, always check first before deleting a relationship that is used in formulas.
- Deleting a relationship between two tables can cause errors in formulas that reference these tables.
- The Data Analysis Expression (DAX) RELATED function uses the relationships between tables to look up related values in another table. It will return different results after the relationship is deleted. For more information, see the RELATED Function (DAX).
- In addition to changing PivotTable and formula results, both the creation and deletion of relationships will cause the workbook to be recalculated, which can take some time.

Delete Relationships

To delete a relationship by using Diagram View

1. In SQL Server Data Tools (SSDT), click the **Model** menu, then point to **Model View**, and then click **Diagram View**.
2. Right-click a relationship line between two tables, and then click **Delete**.

To delete a relationship by using the Manage Relationships dialog box

1. In SQL Server Data Tools, click the **Table** menu, and then click **Manage Relationships**.
2. In the **Manage Relationships** dialog box, select one or more relationships from the list.

   To select multiple relationships, hold down CTRL while you click each relationship.
3. Click **Delete Relationship**.
4. In the **Manage Relationships** dialog box, click **Close**.

See Also

Relationships
Create a relationship

Bi-directional cross filters in tabular models

12/7/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

New in SQL Server 2016 is a built-in approach for enabling *bi-directional cross filters* in tabular models, eliminating the need for hand-crafted DAX workarounds for propagating filter context across table relationships.

Breaking the concept down into its component parts: *cross filtering* is the ability to set a filter context on a table based on values in a related table, and *bi-directional* is the transference of a filter context to second related table on the other side of a table relationship. As the name implies, you can slice in both directions of the relationship rather than just one way. Internally, two-way filtering expands filter context to query a superset of your data.

There are two types of cross filters: One-way and two-way filtering. One-way is the traditional many-to-one filter direction between fact and dimensional tables in that relationship. Two-way is a cross-filter that enables the filter context of one relationship to be used as the filter context for another table relationship, with one table common to both relationships.

Given **DimDate** and **DimProduct** with foreign key relationships to **FactOnlineSales**, a two-way cross filter is equivalent of **FactOnlineSales-to-DimDate** plus **FactOnlineSales-to-DimProduct** used simultaneously.

Bi-directional cross filters can be an easy fix to the many-to-many query design problem that has challenged tabular and Power Pivot developers in the past. If you've used the DAX workaround for many-to-many relationships in tabular or Power Pivot models, you can try applying a two-way filter to see if it produces expected results.

When creating a bi-directional cross filter, keep the following points in mind:

- Think before you enable two-way filters.

  If you enable two-way filters everywhere, your data could be over-filtered in ways that you might not expect. You might also inadvertently introduce ambiguity by creating more than one potential query path. To avoid both issues, plan on using a combination of one-way and two-way filters.

- Do incremental testing to verify the impact of each filter change on your model. Analyze in Excel works well for incremental testing. As a best practice, periodically follow that up with tests using other reporting clients so that there are no surprises later.

> **NOTE**
> Starting in CTP 3.2, SSDT includes a default that determines whether bi-directional cross filters are attempted automatically. If you enable bi-directional filters by default, SSDT will enable two-way filtering only if the model clearly articulates one query path through a chain of table relationships.

Set the default

Single directional filters are the default. You can change the default for all new projects created in the designer, or on the model itself when the project already exists.

At the project level, the setting is evaluated when you create the project so if you change the default to bi-directional, you'll see the effects of your selection when you create the next project.

1. In SSDT, select **Tools** > **Options** > **Analysis Services Tabular Designers** > **New project settings**.
2. Set **Default filter direction** to either **Single direction** or **Both directions**.

Alternatively, you can change the default on the model.

1. In Solution Explorer, select **Model.bim** > **Properties** ,
2. Set **Default filter direction** to either **Single direction** or **Both directions**.

Walkthrough an example

The best way to appreciate the value of bi-directional cross filtering is through an example. Consider the following dataset from ContosoRetailDW, reflecting the cardinality and cross-filters that are created by default.

> **NOTE**
> By default, during data import, table relationships are created for you in many-to-one configurations derived from the foreign key and primary key relationships between the fact table and related dimension tables.

Notice that the filter direction is from dimension tables to the fact table -- promotions, products, dates, customer, and customer geography are all valid filters that successfully yield some aggregation of a measure, with the actual value varying based on the dimensions used.

Manage Relationships

Create | Edit | Delete

Active	Table 1	Cardinality	Filter Direction	Table 2
Yes	FactOnlineSales [PromotionKey]	Many to One (*:1)	<< To FactOnlineSales	DimPromotion [PromotionKey]
Yes	FactOnlineSales [ProductKey]	Many to One (*:1)	<< To FactOnlineSales	DimProduct [ProductKey]
Yes	FactOnlineSales [DateKey]	Many to One (*:1)	<< To FactOnlineSales	DimDate [Datekey]
Yes	FactOnlineSales [CustomerKey]	Many to One (*:1)	<< To FactOnlineSales	DimCustomer [CustomerKey]
Yes	DimCustomer [GeographyKey]	Many to One (*:1)	<< To DimCustomer	DimGeography [GeographyKey]

For this simple star schema, testing in Excel confirms that data slices nicely when filtering flows from dimension tables on rows and columns to aggregated data provided by a **Sum of Sales** measure located in the central **FactOnlineSales** table.

	A	B	C	D	E
1	Sum of Sales	Column Labels			
2	Row Labels	Asia	Europe	North America	Grand Total
3	**A. Datum Corporation**	**$49,524,115.17**	**$56,650,738.54**	**$73,277,092.60**	**$179,451,946.31**
4	2007	$23,400,450.11	$35,116,949.15	$45,844,578.11	$104,361,977.37
5	2008	$12,108,876.22	$11,163,448.13	$13,781,575.31	$37,053,899.65
6	2009	$14,014,788.85	$10,370,341.26	$13,650,939.19	$38,036,069.29
7	**Adventure Works**	**03,648,062.21**	**$116,881,777.06**	**$149,762,018.26**	**$370,291,857.54**
8	2007	$40,852,178.20	$60,613,375.56	$81,213,009.73	$182,678,563.48
9	2008	$29,044,034.07	$25,653,381.35	$31,781,059.77	$86,478,475.19
10	2009	$33,751,849.95	$30,615,020.16	$36,767,948.76	$101,134,818.87
11	**Contoso, Ltd**	**$192,442,421.53**	**$190,965,609.21**	**$239,022,265.49**	**$622,430,296.23**
12	2007	$60,393,606.40	$78,213,463.58	$95,883,741.75	$234,490,811.73
13	2008	$64,608,599.87	$58,815,357.39	$74,135,432.63	$197,559,389.88
14	2009	$67,440,215.26	$53,936,788.25	$69,003,091.11	$190,380,094.62

As long as measures are pulled from the fact table and the filter context terminates at the fact table, the aggregations will be filtered correctly for this model. But what happens if you want to create measures elsewhere, such as a distinct count in the products or customer table, or an average discount in the promotion table, and have an existing filter context extend to that measure.

Let's try it out by adding a distinct count from **DimProducts** to the PivotTable. Notice the repeating values for **Count Products**. At first glance, this looks like a missing table relationship, but in our model, we can see that all the relationships are fully defined and active. In this case, the repeating values occur because there is no date filter on rows in the product table.

	A	B	C
1	Row Labels	Sum of Sales	Count Products
2	**A. Datum Corporation**	**$179,451,946.31**	**132**
3	2005		132
4	2006		132
5	2007	$104,361,977.37	132
6	2008	$37,053,899.65	132
7	2009	$38,036,069.29	132
8	2010		132
9	2011		132
10	**Adventure Works**	**$370,291,857.54**	**192**
11	**Contoso, Ltd**	**$622,430,296.23**	**710**
12	**Fabrikam, Inc.**	**$448,626,054.69**	**267**
13	**Litware, Inc.**	**$297,378,189.47**	**264**
14	**Northwind Traders**	**$108,163,138.94**	**47**
15	**Proseware, Inc.**	**$223,288,734.22**	**244**
16	**Southridge Video**	**$131,026,015.56**	**192**
17	**Tailspin Toys**	**$30,588,870.06**	**144**
18	**The Phone Company**	**$124,810,502.38**	**152**
19	**Wide World Importers**	**$182,147,024.42**	**173**
20	**Grand Total**	**$2,718,202,629.81**	**2517**

After adding a two-way cross filter between **FactOnlineSales** and **DimProduct**, the rows in the product table are now correctly filtered by manufacturer and date.

	A	B	C
1	Row Labels	Sum of Sales	Count Products
2	A. Datum Corporation	$179,451,946.31	132
3	2007	$104,361,977.37	111
4	2008	$37,053,899.65	123
5	2009	$38,036,069.29	131
6	Adventure Works	$370,291,857.54	192
7	Contoso, Ltd	$622,430,296.23	710
8	Fabrikam, Inc.	$448,626,054.69	267
9	Litware, Inc.	$297,378,189.47	264
10	Northwind Traders	$108,163,138.94	47
11	Proseware, Inc.	$223,288,734.22	244
12	Southridge Video	$131,026,015.56	192
13	Tailspin Toys	$30,588,870.06	144
14	The Phone Company	$124,810,502.38	152
15	Wide World Importers	$182,147,024.42	173
16	Grand Total	$2,718,202,629.81	2517

Learn step-by-step

You can try out bi-directional cross filters by stepping through this walkthrough. To follow along, you'll need:

- SQL Server 2016 Analysis Services instance, tabular mode, latest CTP release
- SQL Server Data Tools for Visual Studio 2015 (SSDT), co-released with the latest CTP.
- ContosoRetailDW
- Read permissions on this data.
- Excel (for use with Analyze in Excel)

Create a project

1. Start SQL Server Data Tools for Visual Studio 2015.
2. Click **File** > **New** > **Project** > **Analysis Services Tabular Model**.
3. In Tabular Model Designer, set the workspace database to a SQL Server 2016 Preview Analysis Services instance in tabular server mode.
4. Verify model compatibility level is set to **SQL Server 2016 RTM (1200)** or higher.

   Click **OK** to create the project.

Add Data

1. Click **Model** > **Import from Data Source** > **Microsoft SQL Server**.
2. Specify the server, database, and authentication method.
3. Choose the ContosoRetailDW database.
4. Click **Next**.
5. On table selection, ctrl-select the following tables:
   - FactOnlineSales
   - DimCustomer
   - DimDate
   - DimGeography
   - DimPromotion

You can edit the names at this point if you want them to be more readable in the model.

Source Table	Schema	Friendly Name	Filter Details
DimChannel	dbo		
DimCurrency	dbo		
DimCustomer	dbo		
DimDate	dbo		
DimEmployee	dbo		
DimEntity	dbo		
DimGeography	dbo		
DimMachine	dbo		
DimOutage	dbo		
DimProduct	dbo		
DimProductCategory	dbo		
DimProductSubcategory	dbo	DimProductSubcategory	
DimPromotion	dbo	DimPromotion	
DimSalesTerritory	dbo		
DimScenario	dbo		
DimStore	dbo		
FactExchangeRate	dbo		
FactInventory	dbo		
FactITMachine	dbo		
FactITSLA	dbo		
FactOnlineSales	dbo		
FactSales	dbo		

6. Import the data.

If you get errors, confirm that the account used to connect to the database has a SQL Server login with read permissions on the Contoso data warehouse. On a remote connection, you might also want to check port configuration in the firewall for SQL Server.

Review default table relationships

Switch to diagram view: **Model** > **Model View** > **Diagram View**. Cardinality and active relationships are indicated visually. All of the relationships are one-to-many between any two related tables.

Alternatively, click **Table** > **Manage Relationships** to view the same information in a table layout.

Active	Table 1	Cardinality	Filter Direction	Table 2
Yes	FactOnlineSales [PromotionKey]	Many to One (*:1)	<< To FactOnlineSales	DimPromotion [PromotionKey]
Yes	FactOnlineSales [ProductKey]	Many to One (*:1)	<< To FactOnlineSales	DimProduct [ProductKey]
Yes	FactOnlineSales [DateKey]	Many to One (*:1)	<< To FactOnlineSales	DimDate [Datekey]
Yes	FactOnlineSales [CustomerKey]	Many to One (*:1)	<< To FactOnlineSales	DimCustomer [CustomerKey]
Yes	DimCustomer [GeographyKey]	Many to One (*:1)	<< To DimCustomer	DimGeography [GeographyKey]

Create measures

You'll need an aggregation to sum sale amounts by different facets of dimensional data. In **DimProduct,** you can create a measure that counts products, and then use it in an analysis of product merchandising that shows a count of how many products participated in sales for a given year, a given region, or customer type.

1. Click **Model** > **Model View** > **Diagram View**.

2. Click **FactOnlineSales**.

3. Select the **SalesAmount** column.
4. Click **Column** > **AutoSum** > **Sum** to create a measure for sales.
5. Click **DimProduct**.
6. Select the **ProductKeycolumn**.
7. Click **Column** > **AutoSum** > **DistinctCount** to create a measure for unique products.

Analyze in Excel

1. Click **Model** > **Analyze in Excel** to bring all the data together in a PivotTable.
2. Select the two measures you just created from the field list.
3. Select **Products** > **Manufacturer**.
4. Select **Date** > **Calendar Year**.

Notice that sales are broken out by year and manufacturer as expected. This is because the default filter context between **FactOnlineSales**, **DimProduct**, and **DimDate** works correctly for measures on the 'many' side of the relationship.

At the same time, you can see that product count is not picking up on the same filter context as sales. While product counts are correctly filtered by manufacturer (both manufacturer and product counts are in the same table), the date filter is not propagated to product count.

Change the cross-filter

1. Back in the model, select **Table** > **Manage Relationships**.
2. Edit the relationship between **FactOnlineSales** and **DimProduct**.

   Change the filter direction to both tables.
3. Save the settings.
4. In the workbook, click **Refresh** to re-read the model.

You should now see that both product counts and sales are filtered by the same filter context, one that includes not only manufacturers from **DimProducts** but also calendar year from **DimDate**.

Next steps

Understanding when and how a bi-directional cross filter can be a matter of trial and error to see how it works in your scenario. At times, you'll find that the built-in behaviors are not sufficient and will need to fall back on DAX computations to get the job done. In the **See Also** section, you'll find several links to additional resources on this subject.

In practical terms, cross-filtering can enable forms of data exploration typically delivered only through a many-to-many construction. Having said that, it's important to recognize that bi-directional cross-filtering is not a many-to-many construct. An actual many-to-many table configuration remains unsupported in the designer for tabular models in this release.

See also

Create and manage relationships in Power BI Desktop
A practical example of how to handle simple many-to-manay relationships in Power Pivot and tabular models
Resolving many-to-many relationships leveraging DAX cross-table filtering
Many to many revolution (SQLBI blog)

Calculations in tabular models

12/6/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you have imported data into your model, you can add calculations to aggregate, filter, extend, combine, and secure that data. Tabular models use Data Analysis Expressions (DAX), a formula language for creating custom calculations. In tabular models, the calculations you create by using DAX formulas are used in *Calculated Columns*, *Measures*, and *Row Filters*.

In this section

TOPIC	DESCRIPTION
Understanding DAX in Tabular Models	Describes the Data Analysis Expressions (DAX) formula language used to create calculations for calculated columns, measures, and row filters in tabular models.
DAX formula compatibility in DirectQuery mode	Describes the differences, lists the functions that are not supported in DirectQuery mode, and lists the functions that are supported but could return different results.
Data Analysis Expressions (DAX) Reference	This section provides detailed descriptions of DAX syntax, operators, and functions.

NOTE

Step-by-step tasks for creating calculations are not provided in this section. Because calculations are specified in calculated columns, measures, and row filters (in roles), instructions on where to create DAX formulas are provided in tasks related to those features. For more information, see Create a Calculated Column, Create and Manage Measures, and Create and Manage Roles.

NOTE

While DAX can also be used to query a tabular model, topics in this section focus specifically on using DAX formulas to create calculations.

DAX in Tabular Models

12/7/2018 • 30 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data Analysis Expressions (DAX) is a formula language used to create custom calculations in Analysis Services, Power BI Desktop, and Power Pivot in Excel. DAX formulas include functions, operators, and values to perform advanced calculations on data in tables and columns.

While DAX is used in Analysis Services, Power BI Desktop, and Power Pivot in Excel, this topic applies more to Analysis Services tabular model projects authored in SQL Server Data Tools (SSDT).

DAX formulas in calculated columns, measures, and row filters

For tabular models authored in SSDT, DAX formulas are used in calculated columns, measures, and row filters.

Calculated columns

A calculated column is a column that you add to an existing table (in the model designer) and then create a DAX formula that defines the column's values.

> **NOTE**
> Calculated columns are not supported for models that retrieve data from a relational data source using DirectQuery mode.

When a calculated column contains a valid DAX formula, values are calculated for each row as soon as the formula is entered. Values are then stored in the database. For example, in a Date table, when the formula `=[Calendar Year] & " Q" & [Calendar Quarter]` is entered into the formula bar, a value for each row in the table is calculated by taking values from the Calendar Year column (in the same Date table), adding a space and the capital letter Q, and then adding the values from the Calendar Quarter column (in the same Date table). The result for each row in the calculated column is calculated immediately and appears, for example, as **2010 Q1**. Column values are only recalculated if the data is re-processed.

For more information, see Calculated Columns.

Measures

Measures are dynamic formulas where the results change depending on context. Measures are used in reporting formats that support combining and filtering model data by using multiple attributes such as a Power BI report or Excel PivotTable or PivotChart. Measures are defined by the model author by using the measure grid (and formula bar) in the model designer in SSDT.

A formula in a measure can use standard aggregation functions automatically created by using the Autosum feature, such as COUNT or SUM, or you can define your own formula by using DAX. When you define a formula for a measure in the formula bar, a Tooltip feature shows a preview of what the results would be for the total in the current context, but otherwise the results are not immediately output anywhere. Other measure details also appear in the **Properties** pane.

The reason you cannot see the (filtered) results of the calculation immediately is because the result of a measure cannot be determined without context. To evaluate a measure requires a reporting client application that can provide the context needed to retrieve the data relevant to each cell and then evaluate the expression for each cell. That client might be an Excel PivotTable or PivotChart, a Power BI report, or an MDX query. Regardless of the reporting client, a separate query is run for each cell in the results. That is to say, each combination of row and

column headers in a PivotTable, or each selection of slicers and filters in a Power BI report, generates a different subset of data over which the measure is calculated. For example, in a measure with the formula, `Total Sales:=SUM([Sales Amount])`, when a user places the TotalSales measure in the Values window in a PivotTable, and then places the DimProductCategory column from a DimProduct table into the Filters window, the sum of Sales Amount is calculated and displayed for each product category.

Unlike calculated columns and row filters, the syntax for a measure includes the measure's name preceding the formula. In the example just provided, the name **Total Sales:** appears preceding the formula. After you have created a measure, the name and its definition appear in the reporting client application Field List and depending on perspectives and roles is available to all users of the model.

For more information, see Measures.

Row filters

Row filters define which rows in a table are visible to members of a particular role. Row filters can be created for each table in a model by using DAX formulas. Row filters are created for a particular role by using Role Manager in SSDT. Row filters can also be defined for a deployed model by using Role Properties in SQL Server Management Studio (SSMS).

In a row filter, a DAX formula, which must evaluate to a Boolean TRUE/FALSE condition, defines which rows can be returned by the results of a query by members of that particular role. Rows not included in the DAX formula cannot be returned. For example, for members of the Sales role, the Customers table with the following DAX formula, `=Customers[Country] = "USA"`, members of the Sales role will only be able to view data for customers in the USA, and aggregates, such as SUM are returned only for customers in the USA.

When you define a row filter by using DAX formula, you are creating an allowed row set. This does not deny access to other rows; rather, they are simply not returned as part of the allowed row set. Other roles can allow access to the rows excluded by the DAX formula. If a user is a member of another role, and that role's row filters allow access to that particular row set, the user can view data for that row.

Row filters apply to the specified rows as well as related rows. When a table has multiple relationships, filters apply security for the relationship that is active. Row filters will be intersected with other row filters defined for related tables.

For more information, see Roles.

DAX Data Types

You can import data into a model from many different data sources that might support different data types. When you import data into a model, the data is converted to one of the tabular model data types. When the model data is used in a calculation, the data is then converted to a DAX data type for the duration and output of the calculation. When you create a DAX formula, the terms used in the formula will automatically determine the value data type returned.

Tabular models, and DAX, support the following data types:

DATA TYPE IN MODEL	DATA TYPE IN DAX	DESCRIPTION
Whole Number	A 64 bit (eight-bytes) integer value [1, 2]	Numbers that have no decimal places. Integers can be positive or negative numbers, but must be whole numbers between -9,223,372,036,854,775,808 (-2^63) and 9,223,372,036,854,775,807 (2^63-1).

DATA TYPE IN MODEL	DATA TYPE IN DAX	DESCRIPTION
Decimal Number	A 64 bit (eight-bytes) real number [1, 2]	Real numbers are numbers that can have decimal places. Real numbers cover a wide range of values: Negative values from -1.79E +308 through -2.23E -308 Zero Positive values from 2.23E -308 through 1.79E + 308 However, the number of significant digits is limited to 17 decimal digits.
Boolean	Boolean	Either a True or False value.
Text	String	A Unicode character data string. Can be strings, numbers or dates represented in a text format.
Date	Date/time	Dates and times in an accepted date-time representation. Valid dates are all dates after March 1, 1900.
Currency	Currency	Currency data type allows values between -922,337,203,685,477.5808 to 922,337,203,685,477.5807 with four decimal digits of fixed precision.
N/A	Blank	A blank is a data type in DAX that represents and replaces SQL nulls. You can create a blank by using the BLANK function, and test for blanks by using the logical function, ISBLANK.

Tabular models also include the Table data type as the input or output to many DAX functions. For example, the FILTER function takes a table as input and outputs another table that contains only the rows that meet the filter conditions. By combining table functions with aggregation functions, you can perform complex calculations over dynamically defined data sets.

While data types are typically automatically set, it is important to understand data types and how they apply, in-particular, to DAX formulas. Errors in formulas or unexpected results, for example, are often caused by using a particular operator that cannot be used with a data type specified in an argument. For example, the formula, `= 1 & 2`, returns a string result of 12. The formula, `= "1" + "2"`, however, returns an integer result of 3.

For detailed information about data types in tabular models and explicit and implicit conversions of data types in DAX, see Data Types Supported.

DAX Operators

The DAX language uses four different types of calculation operators in formulas:

- Comparison operators to compare values and return a logical TRUE\FALSE value.

- Arithmetic operators to perform arithmetic calculations that return numeric values.
- Text concatenation operators to join two or more text strings.
- Logical operators that combine two or more expressions to return a single result.

For detailed information about operators used in DAX formulas, see DAX Operator Reference.

DAX formulas

DAX formulas are essential for creating calculations in calculated columns and measures, and securing your data by using row level filters. To create formulas for calculated columns and measures, you will use the formula bar along the top of the model designer window or the DAX Editor. To create formulas for row filters, you will use the Role Manager dialog box. Information in this section is meant to get you started with understanding the basics of DAX formulas.

Formula basics

DAX enables tabular model authors to define custom calculations in both model tables, as part of calculated columns, and as measures associated with tables, but not appearing directly in them. DAX also enables model authors to secure data, by creating calculations that return a Boolean value defining which rows in a particular or related table can be queried by member users of the associated role.

DAX formulas can be very simple or quite complex. The following table shows some examples of simple formulas that could be used in a calculated column.

Formula	Description
`=TODAY()`	Inserts today's date in every row of the column.
`=3`	Inserts the value 3 in every row of the column.
`=[Column1] + [Column2]`	Adds the values in the same row of [Column1] and [Column2] and puts the results in the calculated column of the same row.

Whether the formula you create is simple or complex, you can use the following steps when building a formula:

1. Each formula must begin with an equal sign.
2. You can either type or select a function name, or type an expression.
3. Begin to type the first few letters of the function or name you want, and AutoComplete displays a list of available functions, tables, and columns. Press TAB to add an item from the AutoComplete list to the formula.

   You can also click the **Fx** button to display a list of available functions. To select a function from the dropdown list, use the arrow keys to highlight the item, and click **OK** to add the function to the formula.
4. Supply the arguments to the function by selecting them from a dropdown list of possible tables and columns, or by typing in values.
5. Check for syntax errors: ensure that all parentheses are closed and columns, tables and values are referenced correctly.
6. Press ENTER to accept the formula.

> **NOTE**
> In a calculated column, as soon as you enter the formula and the formula is validated, the column is populated with values. In a measure, pressing ENTER saves the measure definition in the measure grid with the table. If a formula is invalid, an error will be displayed.

In this example, we will look at a more complex formula in a measure named Days in Current Quarter:

```
Days in Current Quarter:=COUNTROWS( DATESBETWEEN( 'Date'[Date], STARTOFQUARTER( LASTDATE('Date'[Date])),
ENDOFQUARTER('Date'[Date])))
```

This measure is used to create a comparison ratio between an incomplete period and the previous period. The formula must take into account the proportion of the period that has elapsed, and compare it to the same proportion in the previous period. In this case, [Days Current Quarter to Date]/[Days in Current Quarter] gives the proportion elapsed in the current period.

This formula contains the following elements:

FORMULA ELEMENT	DESCRIPTION
`Days in Current Quarter:=`	The name of the measure.
`=`	The equals sign (=) begins the formula.
`COUNTROWS`	The COUNTROWS Function (DAX) counts the number of rows in the Date table
`()`	Open and closing parenthesis specifies arguments.
`DATESBETWEEN`	The DATESBETWEEN function returns the dates between the last date for each value in the Date column in the Date table.
`'Date'`	Specifies the Date table. Tables are in single quotes.
`[Date]`	Specifies the Date column in the Date table. Columns are in brackets.
`,`	
`STARTOFQUARTER`	The STARTOFQUARTER function returns the date of the start of the quarter.
`LASTDATE`	The LASTDATE function returns the last date of the quarter.
`'Date'`	Specifies the Date table.
`[Date]`	Specifies the Date column in the Date table.
`,`	
`ENDOFQUARTER`	The ENDOFQUARTER function

FORMULA ELEMENT	DESCRIPTION
`'Date'`	Specifies the Date table.
`[Date]`	Specifies the Date column in the Date table.

Using formula AutoComplete

Both the formula bar in the model designer and the formula Row Filters window in the Role Manager dialog box provide an AutoComplete feature. AutoComplete helps you enter a valid formula syntax by providing you with options for each element in the formula.

- You can use formula AutoComplete in the middle of an existing formula with nested functions. The text immediately before the insertion point is used to display values in the drop-down list, and all of the text after the insertion point remains unchanged.
- AutoComplete does not add the closing parenthesis of functions or automatically match parentheses. You must make sure that each function is syntactically correct or you cannot save or use the formula.

Using multiple functions in a Formula

You can nest functions, meaning that you use the results from one function as an argument of another function. You can nest up to 64 levels of functions in calculated columns. However, nesting can make it difficult to create or troubleshoot formulas.

Many functions are designed to be used solely as nested functions. These functions return a table, which cannot be directly saved as a result; it must be provided as input to a table function. For example, the functions SUMX, AVERAGEX, and MINX all require a table as the first argument.

> **NOTE**
> Some limits are applied within measures on nesting of functions to ensure that performance is not affected by the many calculations required by dependencies among columns.

DAX functions

This section provides an overview of the *types* of functions supported in DAX. For more detailed information, see DAX Function Reference.

DAX provides a variety of functions you can use perform calculations using dates and times, create conditional values, work with strings, perform lookups based on relationships, and the ability to iterate over a table to perform recursive calculations. If you are familiar with Excel formulas, many of these functions will appear very similar; however, DAX formulas are different in the following important ways:

- A DAX function always references a complete column or a table. If you want to use only particular values from a table or column, you can add filters to the formula.
- If you need to customize calculations on a row-by-row basis, DAX provides functions that let you use the current row value or a related value as a kind of parameter, to perform calculations that vary by context. To understand how these functions work, see Context in DAX Formulas later in this topic.
- DAX includes many functions that return a table, rather than a value. The table is not displayed in a reporting client, but is used to provide input to other functions. For example, you can retrieve a table and then count the distinct values in it, or calculate dynamic sums across filtered tables or columns.
- DAX functions include a variety of *time-intelligence* functions. These functions let you define or select date ranges, and perform dynamic calculations based on these dates or range. For example, you can compare sums across parallel periods.

Date and time functions

The date and time functions in DAX are similar to date and time functions in Microsoft Excel. However, DAX functions are based on the **datetime** data types used by Microsoft SQL Server. For more information, see Date and Time Functions (DAX).

Filter functions

The filter functions in DAX return specific data types, look up values in related tales, and filter by related values. The lookup functions work by using tables and relationships, like a database. The filtering functions let you manipulate data context to create dynamic calculations. For more information, see Filter Functions (DAX).

Information functions

An information function looks at the cell or row that is provided as an argument and tells you whether the value matches the expected type. For example, the ISERROR function returns TRUE if the value that you reference contains an error. For more information, see Information Functions (DAX).

Logical functions

Logical functions act upon an expression to return information about the values in the expression. For example, the TRUE function lets you know whether an expression that you are evaluating returns a TRUE value. For more information, see Logical Functions (DAX).

Mathematical and trigonometric functions

The mathematical functions in DAX are very similar to the Excel mathematical and trigonometric functions. Some minor differences exist in the numeric data types used by DAX functions. For more information, see Math and Trig Functions (DAX).

Other functions

These functions perform unique actions that cannot be defined by any of the categories most other functions belong to. For more information, see Other Functions (DAX).

Statistical functions

DAX provides statistical functions that perform aggregations. In addition to creating sums and averages, or finding the minimum and maximum values, in DAX you can also filter a column before aggregating or create aggregations based on related tables. For more information, see Statistical Functions (DAX).

Text functions

The text functions in DAX are very similar to their counterparts in Excel. You can return part of a string, search for text within a string, or concatenate string values. DAX also provides functions for controlling the formats for dates, times, and numbers. For more information, see Text Functions (DAX).

Time-intelligence functions

The time-intelligence functions provided in DAX let you create calculations that use built-in knowledge about calendars and dates. By using time and date ranges in combination with aggregations or calculations, you can build meaningful comparisons across comparable time periods for sales, inventory, and so on. For more information, see Time-intelligence Functions (DAX).

Table-valued functions

There are DAX functions that output tables, take tables as input, or do both. Because a table can have a single column, table-valued functions also take single columns as inputs. Understanding how to use these table-valued functions is important for fully utilizing DAX formulas. DAX includes the following types of table-valued functions:

Filter functions - Return a column, table, or values related to the current row.

Aggregation functions - Aggregate any expression over the rows of a table.

Time-intelligence functions - Return a table of dates, or use a table of dates to calculate an aggregation.

Context in DAX formulas

Context is an important concept to understand when creating formulas using DAX. Context is what enables you to perform dynamic analysis, as the results of a formula change to reflect the current row or cell selection and also any related data. Understanding context and using context effectively are critical for building high-performing, dynamic analyses, and for troubleshooting problems in formulas.

Formulas in tabular models can be evaluated in a different context, depending on other design elements:

- Filters applied in a PivotTable or report
- Filters defined within a formula
- Relationships specified by using special functions within a formula

There are different types of context: *row context*, *query context*, and *filter context*.

Row context

Row context can be thought of as "the current row". If you create a formula in a calculated column, the row context for that formula includes the values from all columns in the current row. If the table is related to another table, the content also includes all the values from the other table that are related to the current row.

For example, suppose you create a calculated column, `=[Freight] + [Tax]`, that adds together values from two columns, Freight and Tax, from the same table. This formula automatically gets only the values from the current row in the specified columns.

Row context also follows any relationships that have been defined between tables, including relationships defined within a calculated column by using DAX formulas, to determine which rows in related tables are associated with the current row.

For example, the following formula uses the RELATED function to fetch a tax value from a related table, based on the region that the order was shipped to. The tax value is determined by using the value for region in the current table, looking up the region in the related table, and then getting the tax rate for that region from the related table.

```
= [Freight] + RELATED('Region'[TaxRate])
```

This formula gets the tax rate for the current region from the Region table and adds it to the value of the Freight column. In DAX formulas, you do not need to know or specify the specific relationship that connects the tables.

Multiple row context

DAX includes functions that iterate calculations over a table. These functions can have multiple current rows, each with its own row context. In essence, these functions let you create formulas that perform operations recursively over an inner and outer loop.

For example, suppose your model contains a **Products** table and a **Sales** table. Users might want to go through the entire sales table, which is full of transactions involving multiple products, and find the largest quantity ordered for each product in any one transaction.

With DAX you can build a single formula that returns the correct value, and the results are automatically updated any time a user adds data to the tables.

```
=MAXX(FILTER(Sales,[ProdKey]=EARLIER([ProdKey])),Sales[OrderQty])
```

For a detailed walkthrough of this formula, see the EARLIER Function (DAX).

To summarize, the EARLIER function stores the row context from the operation that preceded the current

operation. At all times, the function stores in memory two sets of context: one set of context represents the current row for the inner loop of the formula, and another set of context represents the current row for the outer loop of the formula. DAX automatically feeds values between the two loops so that you can create complex aggregates.

Query context

Query context refers to the subset of data that is implicitly retrieved for a formula. When a user places a measure or other value field into a PivotTable or into a report based on a tabular model, the engine examines the row and column headers, Slicers, and report filters to determine the context. Then, the necessary queries are run against the data source to get the correct subset of data, make the calculations defined by the formula, and then populate each cell in the PivotTable or report. The set of data that is retrieved is the query context for each cell.

> **WARNING**
>
> For a model in DirectQuery mode, the context is evaluated, and then the set operations to retrieve the correct subset of data and calculate the results are translated to SQL statements. Those statements are then directly run against the relational data store. Therefore, although the method of getting the data and calculating the results is different, the context itself does not change.

Because context changes depending on where you place the formula, the results of the formula can also change.

For example, suppose you create a formula that sums the values in the **Profit** column of the **Sales** table: `=SUM('Sales'[Profit])`. If you use this formula in a calculated column within the **Sales** table, the results for the formula will be the same for the entire table, because the query context for the formula is always the entire data set of the **Sales** table. Results will have profit for all regions, all products, all years, and so on.

However, users typically don't want to see the same result hundreds of times, but instead want to get the profit for a particular year, a particular country, a particular product, or some combination of these, and then get a grand total.

In a PivotTable, context can be changed by adding or removing column and row headers and by adding or removing Slicers. Whenever users add column or row headings to the PivotTable, they change the query context in which the measure is evaluated. Slicing and filtering operations also affect context. Therefore, the same formula, used in a measure, is evaluated in a different *query context* for each cell.

Filter context

Filter context is the set of values allowed in each column, or in the values retrieved from a related table. Filters can be applied to the column in the designer, or in the presentation layer (reports and PivotTables). Filters can also be defined explicitly by filter expressions within the formula.

Filter context is added when you specify filter constraints on the set of values allowed in a column or table, by using arguments to a formula. Filter context applies on top of other contexts, such as row context or query context.

In tabular models, there are many ways to create filter context. Within the context of clients that can consume the model, such as Power BI reports, users can create filters on the fly by adding slicers or report filters on the row and column headings. You can also specify filter expressions directly within the formula, to specify related values, to filter tables that are used as inputs, or to dynamically get context for the values that are used in calculations. You can also completely clear or selectively clear the filters on particular columns. This is very useful when creating formulas that calculate grand totals.

For more information about how to create filters within formulas, see the FILTER Function (DAX).

For an example of how filters can be cleared to create grand totals, see the ALL Function (DAX).

For examples of how to selectively clear and apply filters within formulas, see the ALLEXCEPT Function (DAX).

Determining context in formulas

When you create a DAX formula, the formula is first tested for valid syntax, and then tested to make sure the

names of the columns and tables included in the formula can be found in the current context. If any column or table specified by the formula cannot be found, an error is returned.

Context during validation (and recalculation operations) is determined as described in the preceding sections, by using the available tables in the model, any relationships between the tables, and any filters that have been applied.

For example, if you have just imported some data into a new table and it is not related to any other tables (and you have not applied any filters), the *current context* is the entire set of columns in the table. If the table is linked by relationships to other tables, the current context includes the related tables. If you add a column from the table to a report that has Slicers and maybe some report filters, the context for the formula is the subset of data in each cell of the report.

Context is a powerful concept that can also make it difficult to troubleshoot formulas. We recommend that you begin with simple formulas and relationships to see how context works. The following section provides some examples of how formulas use different types of context to dynamically return results.

Examples of context in formulas

1. The RELATED Function (DAX) function expands the context of the current row to include values in a related column. This lets you perform lookups. The example in this topic illustrates the interaction of filtering and row context.

2. The FILTER Function (DAX) function lets you specify the rows to include in the current context. The examples in this topic also illustrate how to embed filters within other functions that perform aggregates.

3. The ALL Function (DAX) function sets context within a formula. You can use it to override filters that are applied as result of query context.

4. The ALLEXCEPT Function (DAX) function lets you remove all filters except one that you specify. Both topics include examples that walk you through building formulas and understanding complex contexts.

5. The EARLIER Function (DAX) and EARLIEST Function (DAX) functions let you loop through tables by performing calculations, while referencing a value from an inner loop. If you are familiar with the concept of recursion and with inner and outer loops, you will appreciate the power that the EARLIER and EARLIEST functions provide. If you are new to these concepts, you should follow the steps in the example carefully to see how the inner and outer contexts are used in calculations.

Formulas and the tabular model

The model designer, in SSDT, is an area where you can work with multiple tables of data and connect the tables in a tabular model. Within this model, tables are joined by relationships on columns with common values (keys). The tabular model lets you link values to columns in other tables and create more interesting calculations. Just as in a relational database, you can connect many levels of related tables and use columns from any of the tables in the results.

For example, you can link a sales table, a products table, and a product categories table, and users can use various combinations of the columns in PivotTables and reports. Related fields can be used to filter connected tables, or to create calculations over subsets. (If you are not familiar with relational database and working with tables and joins, see Relationships.)

Tabular models support multiple relationships among tables. To avoid confusion or wrong results, only one relationship at a time is designated as the active relationship, but you can change the active relationship as necessary to traverse different connections in the data in calculations. The USERELATIONSHIP Function (DAX) can be used to specify one or more relationships to be used in a specific calculation.

In a tabular model, you should observe these formula design rules:

- When tables are connected by a relationship, you must ensure that the two columns used as keys have

values that match. Referential integrity is not enforced, however; therefore it is possible to have non-matching values in a key column and still create a relationship. If this happens, you should be aware that blank values or non-matching values might affect the results of formulas.

- When you link tables in your model by using relationships, you enlarge the scope, or *context*, in which your formulas are evaluated. Changes in context resulting from the addition of new tables, new relationships, or from changes in the active relationship can cause your results to change in ways that you might not anticipate. For more information, see Context in DAX Formulas earlier in this topic.

Working with tables and columns

Tables in tabular models look like Excel tables, but are different in the way they work with data and with formulas:

- Formulas work only with tables and columns, not with individual cells, range references, or arrays.
- Formulas can use relationships to get values from related tables. The values that are retrieved are always related to the current row value.
- You cannot have irregular or "ragged" data like you can in an Excel worksheet. Each row in a table must contain the same number of columns. However, you can have empty values in some columns. Excel data tables and tabular model data tables are not interchangeable.
- Because a data type is set for each column, each value in that column must be of the same type.

Referring to tables and columns in formulas

You can refer to any table and column by using its name. For example, the following formula illustrates how to refer to columns from two tables by using the *fully qualified* name:

```
=SUM('New Sales'[Amount]) + SUM('Past Sales'[Amount])
```

When a formula is evaluated, the model designer first checks for general syntax, and then checks the names of columns and tables that you provide against possible columns and tables in the current context. If the name is ambiguous or if the column or table cannot be found, you will get an error on your formula (an #ERROR string instead of a data value in cells where the error occurs). For more information about naming requirements for tables, columns, and other objects, see "Naming Requirements" in DAX Syntax Reference.

Table relationships

By creating relationships between tables, you gain the ability to look up data in another table and use related values to perform complex calculations. For example, you can use a calculated column to look up all the shipping records related to the current reseller, and then sum the shipping costs for each. In many cases, however, a relationship might not be necessary. You can use the LOOKUPVALUE function in a formula to return the value in *result_columnName* for the row that meets criteria specified in the *search_column* and *search_value* parameters.

Many DAX functions require that a relationship exist between the tables, or among multiple tables, in order to locate the columns that you have referenced and return results that make sense. Other functions will attempt to identify the relationship; however, for best results you should always create a relationship where possible. For more information, see Formulas and the Tabular Model earlier in this topic.

Updating the results of formulas (Process)

Data process and *recalculation* are two separate but related operations. You should thoroughly understand these concepts when designing a model that contains complex formulas, large amounts of data, or data that is obtained from external data sources.

Processing data is the process of updating the data in a model with new data from an external data source.

Recalculation is the process of updating the results of formulas to reflect any changes to the formulas themselves and to reflect changes in the underlying data. Recalculation can affect performance in the following ways:

- The values in a calculated column are computed and stored in the model. To update the values in the calculated column, you must process the model using one of three processing commands - Process Full, Process Data, or Process Recalc. The result of the formula must always be recalculated for the entire column, whenever you change the formula.
- The values calculated by measures are dynamically evaluated whenever a user adds the measure to a pivot table or open a report; as the user modifies the context, values returned by the measure change. The results of the measure always reflect the latest in the in-memory cache.

Processing and recalculation have no effect on row filter formulas unless the result of a recalculation returns a different value, thus making the row queryable or not queryable by role members.

For more information, see Process Data.

Troubleshooting errors in formulas

If you get an error when defining a formula, the formula might contain either a *syntactic error*, *semantic error*, or *calculation error*.

Syntactic errors are the easiest to resolve. They typically involve a missing parenthesis or comma. For help with the syntax of individual functions, see DAX Function Reference.

The other type of error occurs when the syntax is correct, but the value or the column referenced does not make sense in the context of the formula. Such semantic and calculation errors might be caused by any of the following problems:

- The formula refers to a non-existing column, table, or function.
- The formula appears to be correct, but when the data engine fetches the data it finds a type mismatch, and raises an error.
- The formula passes an incorrect number or type of parameters to a function.
- The formula refers to a different column that has an error, and therefore its values are invalid.
- The formula refers to a column that has not been processed, meaning it has metadata but no actual data to use for calculations.

In the first four cases, DAX flags the entire column that contains the invalid formula. In the last case, DAX grays out the column to indicate that the column is in an unprocessed state.

Additional resources

The Tabular Modeling (Adventure Works Tutorial) provides step-by-step instructions on how to create a tabular model that includes many calculations in calculated columns, measures, and row filters. For most formulas, a description about what the formula is meant to do is provided.

The Analysis Services Team Blog provides the latest information, tips, news and announcements.

The DAX Resource Center provides both internal and external information on DAX, including numerous DAX solutions submitted by leading Business Intelligence professionals.

See Also

Data Analysis Expressions (DAX) Reference

Measures
Calculated Columns
Roles
KPIs
Data Sources Supported

Measures

12/7/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In tabular models, a measure is a calculation created using a DAX formula for use in a reporting client. Measures are evaluated based on fields, filters, and slicers users select in the reporting client application.

Benefits

Measures can be based on standard aggregation functions, such as AVERAGE, COUNT, or SUM, or you can define your own formula by using DAX. In addition to the formula, each measure has properties defined by the measure data type, such as Name, Table Detail, Format, and Decimal Places.

When measures have been defined in a model, users can then add them to a report or PivotTable. Depending on perspectives and roles, measures appear in the Field List with their associated table, and are available to all users of the model. Measures are usually created in Fact tables; however, measures can be independent of the table it is associated with.

It is important to understand the fundamental differences between a calculated column and a measure. In a calculated column, a formula evaluates to a value for each row in the column. For example, in a FactSales table, a calculated column named TotalProfit with the following formula calculates a value for total profit for each row (one row per sale) in the FactSales table:

```
=[SalesAmount] - [TotalCost] - [ReturnAmount]
```

The TotalProfit calculated column can then be used in a reporting client just as any other column.

A measure, on the other hand, evaluates to a value based on a user selection; a filter context set in a PivotTable or report. For example, a measure in the FactSales table is created with the following formula:

```
Sum of TotalProfit: =SUM([TotalProfit])
```

A sales analyst, using Excel, wants to know the total profit for a category of products. Each product category is comprised of multiple products. The sales analyst selects the ProductCategoryName column and adds it to the Row Labels filter window of a PivotTable; a row for each product category is then displayed in the PivotTable. The user then selects the Sum of TotalProfit measure. A measure will by default be added to the Values filter window. The measure calculates the sum of total profit and displays the results for each product category. The sales analyst can then further filter the sum of total profit for each product category by using a slicer, for example, adding CalendarYear as a slicer to view the sum of total profit for each product category by year.

PRODUCTCATEGORYNAME	SUM OF TOTALPROFIT
Audio	$2,731,061,308.69
Cameras and Camcorders	$620,623,675.75
Computers	$392,999,044.59

PRODUCTCATEGORYNAME	SUM OF TOTALPROFIT
Tv and Video	$946,989,702.51
Grand Total	**$4,691,673,731.53**

Defining measures by using the measure grid

Measures are created at design time by using the measure grid in the model designer. Each table has a measure grid. By default, the measure grid is displayed below each table in the model designer. You can also choose not to view the measure grid for a particular table. To toggle the display of a table's measure grid, click the **Table** menu, and then click **Show Measure Grid**.

In the measure grid, you can create measures in the following ways:

- Click on an empty cell in the measure grid, and then type a DAX formula in the formula bar. When you click ENTER to complete the formula, the measure will then appear in the cell in the measure grid.
- Create a measure using a standard aggregation function by clicking on a column, then clicking on the AutoSum button (∑) on the toolbar, and then clicking on a standard aggregation function. Standard aggregations are: Sum, Average, Count, DistinctCount, Max, Min. Measures created using the AutoSum button will always appear in the Measure grid directly beneath the column.

By default, when using AutoSum, the name of the measure is defined by the name of the associated column followed by a colon, followed by the formula. The name can be changed in the formula bar or in the **Measure Name** property setting in the Properties window. When creating a measure by using a custom formula, you can type a name in the formula bar, followed by a colon, then followed by the formula, or you can type a name in the **Measure Name** property setting in the Properties window.

It's important to name measures carefully. The measure name will appear with the associated table in the Field List of the reporting client. A KPI will also be named according to the base measure. A measure cannot have the same name as any column in any table in the model.

> **TIP**
> You can group measures from multiple tables into one table by creating an empty table, and then moving or create new measures into it. Keep in-mind, you may need to include table names in DAX formulas when referencing columns in other tables.

If perspectives have been defined for the model, measures are not automatically added to any of those perspectives. You must manually add measures to a perspective by using the Perspectives dialog box. To learn more, see Perspectives.

Measure Properties

Each measure has properties that define it. Measure properties, along with the associated column properties can be edited in the Properties window. Measures have the following properties:

PROPERTY	DEFAULT SETTING	DESCRIPTION
Description	Blank	Description of the measure. The description will not appear with the measure in a reporting client.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Format	Automatically determined from the data type of the column referenced in the formula expression.	Format of the measure. For example, currency or percentage.
Formula	The formula entered in the formula bar when the measure was created.	The formula of the measure.
Measure Name	If AutoSum is used, the measure name will precede the column name followed by a colon. If a custom formula is entered, type a name followed by a colon, and then type the formula.	The name of the measure as it is displayed in a reporting client Field List.

Using a Measure in a KPI

A KPI (Key Performance Indicator) is defined by a *Base* value, defined by a measure, against a *Target* value, also defined by a measure or by an absolute value. A KPI also includes *Status*, a calculation where the Base value evaluates against the Target value between thresholds, displayed in graphical format. KPIs are often used by business professionals to identify trends in critical business metrics.

Any measure can serve as the Base measure of a KPI. To create a KPI, in the measure grid, right-click a measure, and then click **Create KPI**. The Key Performance Indicator dialog box appears where you can then specify a target value (defined by a measure or an absolute value) and define status thresholds, and a graphic type. To learn more, see KPIs.

Related tasks

TOPIC	DESCRIPTION
Create and Manage Measures	Describes how to create and manage measures using the measure grid in the model designer.

See Also

KPIs
Create and Manage KPIs
Calculated Columns

Create and manage measures

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A measure is a formula that is created for use in a report or Excel PivotTable (or PivotChart). Measures can be based on standard aggregation functions, such as COUNT or SUM, or you can define your own formula by using DAX. The tasks in this topic describe how to create and manage measures by using a table's measure grid.

Tasks

To create and manage measures, you will use a table's measure grid. You can view the measure grid for a table in the model designer in Data View only. You cannot create measures or view the measure grid when in Diagram View; however, you can view existing measures in Diagram View. To show the measure grid for a table, click the **Table** menu, and then click **Show Measure Grid**.

To create a measure using a standard aggregation formula

- Click on the column for which you want to create the measure, then click the **Column** menu, then point to **AutoSum**, and then click an aggregation type.

  The measure will be created automatically with a default name, followed by the formula in the first cell in the measure grid directly beneath the column.

To create a measure using a custom formula

- In the measure grid, beneath the column for which you want to create the measure, click a cell, then in the formula bar, type a name, followed by a colon (:), followed by an equals sign (=), followed by the formula. Press ENTER to accept the formula.

To edit measure properties

- In the measure grid, click a measure, and then In the **Properties** window, type or select a different property value.

To rename a measure

- In the measure grid, click on a measure, and then In the **Properties** window, in **Measure Name**, type a new name, and then click ENTER.

  You can also rename a measure in the formula bar. The measure name precedes the formula, followed by a colon.

To delete a measure

- In the measure grid, right-click a measure, and then click **Delete**.

See Also

Measures
KPIs
Calculated columns

KPIs

12/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A *KPI* (Key Performance Indicator), in a tabular model, is used to gauge performance of a value, defined by a *Base* measure, against a *Target* value, also defined by a measure or by an absolute value. This article provides tabular model authors a basic understanding of KPIs in a tabular model.

Benefits

In business terminology, a Key Performance Indicator (KPI) is a quantifiable measurement for gauging business objectives. A KPI is frequently evaluated over time. For example, the sales department of an organization may use a KPI to measure monthly gross profit against projected gross profit. The accounting department may measure monthly expenditures against revenue to evaluate costs, and a human resources department may measure quarterly employee turnover. Each is an example of a KPI. Business professionals frequently consume KPIs that are grouped together in a business scorecard to obtain a quick and accurate historical summary of business success or to identify trends.

A KPI in a tabular model includes:

Base Value
A Base value is defined by a measure that resolves to a value. This value, for example, can be an aggregate of actual Sales or a calculated measure such as Profit for a given period.

Target value
A Target value is defined by a measure that resolves to a value, or by an absolute value. For example, a target value could be the amount by which the business managers of an organization want to increase sales or profit.

Status thresholds
A Status threshold is defined by the range between a low and high threshold or by a fixed value. The Status threshold displays with a graphic to help users easily determine the status of the Base value compared to the Target value.

Example

The sales manager at Adventure Works wants to create a PivotTable that she can use to quickly display whether or not sales employees are meeting their sales quota for a given period (year). For each sales employee, she wants the PivotTable to display the actual sales amount in dollars, the sales quota amount in dollars, and a simple graphic display showing the status of whether or not each sales employee is below, at, or above their sales quota. She wants to be able to slice the data by year.

To do this, the sales manager enlists the help of her organization's BI solution developer to add a Sales KPI to the AdventureWorks Tabular Model. The sales manager will then use Excel to connect to the Adventure Works Tabular Model as a data source and create a PivotTable with the fields (measures and KPI) and slicers to analyze whether or not the sales force is meeting their quotas.

In the model, a measure on the SalesAmount column in the FactResellerSales table, which gives the actual sales amount in dollars for each sales employee is created. This measure will define the Base value of the KPI.

The Sales measure is created with the following formula:

```
Sales:=Sum(FactResellerSales[SalesAmount])
```

The SalesAmountQuota column in the FactSalesQuota table has a sales amount quota defined for each employee. The values in this column will serve as the Target measure (value) in the KPI.

The SalesAmountQuota measure is created with the following formula:

```
Target SalesAmountQuota:=Sum(FactSalesQuota[SalesAmountQuota])
```

> **NOTE**
> There is a relationship between the EmployeeKey column in the FactSalesQuota table and the EmployeeKey in the DimEmployees table. This relationship is necessary so that each sales employee in the DimEmployee table is represented in the FactSalesQuota table.

Now that measures have been created to serve as the Base value and Target value of the KPI, the Sales measure is extended to a new Sales KPI. In the Sales KPI, the Target SalesAmountQuota measure is defined as the Target value. The Status threshold is defined as a range by percentage, the target of which is 100% meaning actual sales defined by the Sales measure met the quota amount defined in the Target SalesAmoutnQuota measure. Low and High percentages are defined on the status bar, and a graphic type is selected.

The sales manager can now create a PivotTable adding the KPI's Base value, Target value, and Status to the Values field. The Employees column is added to the RowLabel field, and the CalendarYear column is added as a Slicer.

The sales manager can now slice by year the actual sales amount, sales quota amount, and status for each sales employee. She can analyze sales trends over years to determine whether or not she needs to adjust the sales quota for a sales employee.

Create and edit KPIs

To create KPIs, in the model designer, you will use the Key Performance Indicator dialog box. Since KPIs must be associated with a measure, you create a KPI by extending a measure that evaluates to a Base value, and then either creating a measure that evaluates to a Target value or by entering an absolute value. After the Base measure (value) and Target value is defined, you can then define the status threshold parameters between the Base and Target values. The status is displayed in a graphical format using selectable icons, bars, graphs, or colors. The Base and Target values, as well as the Status can then be added to a report or PivotTable as values that can be sliced against other data fields.

To view the Key Performance Indicator dialog box, in the measure grid for a table, right click a measure that will serve as the Base value, and then click **Create KPI**. After a measure has been extended to a KPI as a Base value, an icon will appear alongside the measure name in the measure grid identifying the measure as associated with a KPI.

Related Tasks

TOPIC	DESCRIPTION
Create and Manage KPIs	Describes how to create a KPI with a Base measure, a Target measure, and status thresholds.

See Also

Measures
Perspectives

Create and manage KPIs

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes how to create, edit, or delete a KPI (Key Performance Indicator) in a tabular model. To create a KPI, you select a measure that evaluates to the KPI's Base value. You then use the Key Performance Indicator dialog box to select a second measure or an absolute value that evaluates to a target value. You can then define status thresholds that measure the performance between the Base and Target measures.

Tasks

> **IMPORTANT**
>
> Before creating a KPI, you must first create a Base measure that evaluates to value. You then extend the Base measure to a KPI. How to create measures are described in another topic, Create and Manage Measures. A KPI also requires a target value. This value can be from another pre-defined measure or an absolute value. Once you have extended a Base measure to a KPI, you can then select the target value and define status thresholds in the Key Performance Indicator dialog box.

To create a KPI

1. In the measure grid, right-click the measure that will serve as the Base measure (value), and then click **Create KPI**.
2. In the **Key Performance Indicator** dialog box, in **Define target value**, select from one of the following:

   Select **Measure**, and then select a target measure from the listbox.

   Select **Absolute value**, and then type a numerical value.
3. In **Define status thresholds**, click and slide the low and high threshold values.
4. In **Select icon style**, click an image type.
5. Click on **Descriptions**, and then type descriptions for KPI, Value, Status, and Target.

> **TIP**
>
> You can use the Analyze in Excel feature to test your KPI. For more information, see Analyze in Excel.

To edit a KPI

- In the measure grid, right-click the measure that serves as the Base measure (value) of the KPI, and then click **Edit KPI Settings**.

To delete a KPI and the base measure

- In the measure grid, right-click the measure that serves as the Base measure (value) of the KPI, and then click **Delete**.

To delete a KPI, but keep the base measure

- In the measure grid, right-click the measure that serves as the Base measure (value) of the KPI, and then click **Delete KPI**.

ALT shortcuts

UI SECTION	KEY COMMAND
KPI base measure	ALT+B
KPI Status	ALT+S
Measure	ALT+M
Absolute value	ALT+A
Define status thresholds	ALT+U
Select icon style	ALT+I
Trend	ALT+T
Descriptions	ALT+D
Trend	ALT+T

See also

KPIs
Measures
Create and managemeasures

Hierarchies

12/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Hierarchies, in tabular models, are metadata that define relationships between two or more columns in a table. Hierarchies can appear separate from other columns in a reporting client field list, making them easier for client users to navigate and include in a report.

Benefits

Tables can include dozens or even hundreds of columns with unusual column names in no apparent order. This can lead to an unordered appearance in reporting client field lists, making it difficult for users to find and include data in a report. Hierarchies can provide a simple, intuitive view of an otherwise complex data structure.

For example, in a Date table, you can create a Calendar hierarchy. Calendar Year is used as the top-most parent level, with Month, Week, and Day included as child levels (Calendar Year->Month->Week->Day). This hierarchy shows a logical relationship from Calendar Year to Day. A client user can then select Calendar Year from a Field List to include all levels in a PivotTable, or expand the hierarchy, and select only particular levels to be included in the PivotTable.

Because each level in a hierarchy is a representation of a column in a table, the level can be renamed. While not exclusive to hierarchies (any column can be renamed in a tabular model), renaming hierarchy levels can make it easier for users to find and include levels in a report. Renaming a level does not rename the column it references; it simply makes the level more identifiable. In our Calendar Year hierarchy example, in the Date table in Data View, the columns: CalendarYear, CalendarMonth, CalendarWeek, and CalendarDay were renamed to Calendar Year, Month, Week, and Day to make them more easily identifiable. Renaming levels has the additional benefit of providing consistency in reports, since users will less likely need to change column names to make them more readable in PivotTables, charts, etc.

Hierarchies can be included in perspectives. Perspectives define viewable subsets of a model that provide focused, business-specific, or application-specific viewpoints of the model. A perspective, for example, could provide users a viewable list (hierarchy) of only those data items necessary for their specific reporting requirements. For more information, see Perspectives.

Hierarchies are not meant to be used as a security mechanism, but as a tool for providing a better user experience. All security for a particular hierarchy is inherited from the underlying model. Hierarchies cannot provide access to model objects to which a user does not already have access. Security for the model database must be resolved before access to objects in the model can be provided through a hierarchy. Security roles can be used to secure model metadata and data. For more information, see Roles.

Defining hierarchies

You create and manage hierarchies by using the model designer in Diagram View. Creating and managing hierarchies is not supported in the model designer in Data View. To view the model designer in Diagram View, click the **Model** menu, then point to **Model View**, and then click **Diagram View**.

To create a hierarchy, right-click a column you want to specify as the parent level, and then click **Create Hierarchy**. You can multi-select any number of columns (within a single table) to include, or you can later add columns as child levels by clicking and dragging columns to the parent level. When multiple columns are selected, columns are automatically placed in an order based on cardinality. You can change the order by clicking and

dragging a column (level) to a different order or by using Up and Down navigation controls on the context menu. When adding a column as a child level, you can use auto-detect by dragging and dropping the column onto the parent level.

A column can appear in more than one hierarchy. Hierarchies cannot include non-column objects such as measures or KPIs. A hierarchy can be based on columns from within a single table only. If you multi-select a measure along with one or more columns, or if you select columns from multiple tables, the **Create Hierarchy** command is disabled in the context menu. To add a column from a different table, use the RELATED DAX function to add a calculated column that references the column from the related table. The function uses the following syntax: `=RELATED(TableName[ColumnName])`. For more information, see RELATED Function.

By default, new hierarchies are named hierarchy1, hierarchy 2, etc. You should change hierarchy names to reflect the nature of the parent-child relationship, making them more identifiable to users.

After you have created hierarchies, you can test their efficacy by using the Analyze in Excel feature. For more information, see Analyze in Excel.

Related tasks

TASK	DESCRIPTION
Create and Manage Hierarchies	Describes how to create and manage hierarchies by using the model designer in Diagram View.

See Also

Tabular Model Designer
Perspectives
Roles

Create and manage hierarchies

12/7/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Hierarchies can be created and managed in the model designer, in Diagram View. To view the model designer in Diagram View, in SQL Server Data Tools (SSDT), click the **Model** menu, then point to **Model View**, and then click **Diagram View**.

This article includes the following tasks:

- Create a Hierarchy
- Edit a Hierarchy
- Delete a Hierarchy

Create a hierarchy

You can create a hierarchy by using the columns and table context menu. When you create a hierarchy, a new parent level displays with selected columns as child levels.

To create a hierarchy from the context menu

1. In the model designer (Diagram View), in a table window, right-click on a column, and then click **Create Hierarchy**.

   To select multiple columns, click each column, then right-click to open the context menu, and then click **Create Hierarchy**.

   A parent hierarchy level is created at the bottom of the table window and the selected columns are copied under the hierarchy as child levels.

2. Type a name for the hierarchy.

You can drag additional columns into your hierarchy's parent level, which copies the columns. Drop the child level to place it where you want it to appear in the hierarchy.

> **NOTE**
> The Create Hierarchy command is disabled in the context menu if you multi-select a measure along with one or more columns or you select columns from multiple tables.

Edit a hierarchy

You can rename a hierarchy, rename a child level, change the order of the child levels, add additional columns as child levels, remove a child level from a hierarchy, show the source name of a child level (the column name), and hide a child level if it has the same name as the hierarchy parent level.

To change the name of a hierarchy or child level

1. Right-click the hierarchy parent level or a child level, and the click **Rename**.

2. Type a new name or edit the existing name.

To change the order of a child level in a hierarchy

- Click and drag a child level into a new position in the hierarchy.
- Or, right-click a child level of the hierarchy, and then click Move Up to move the level up in the list, or click Move Down to move the level down in the list.
- Or, click a child level to select it, and then press Alt + Up Arrow to move the level up, or press Alt+Down Arrow to move the level down in the list.

To add another child level to a hierarchy

- Click and drag a column onto the parent level or into a specific location of the hierarchy. The column is copied as a child level of the hierarchy.
- Or, right-click a column, point to **Add to Hierarchy**, and then click the hierarchy.

> **NOTE**
> You can add a hidden column (a column that is hidden from reports) as a child level to the hierarchy. The child level is not hidden.

To remove a child level from a hierarchy

- Right-click a child level, and then click **Remove from Hierarchy**.
- Or, click a child level, and then press **Delete**.

> **NOTE**
> If you rename a hierarchy child level, it no longer shares the same name as the column that it is copied from. Use the **Show Source Name** command to see which column it was copied from.

To show a source name

- Right-click a hierarchy child level, and then click **Show Source Name**. The name of the column that it was copied from appears.

Delete a Hierarchy

To delete a hierarchy and remove its child levels

- Right-click the parent hierarchy level, and then click Delete Hierarchy.
- Or, click the parent hierarchy level, and then press Delete. This also removes all the child levels.

See Also

Tabular Model Designer
Hierarchies
Measures

Partitions

1/18/2019 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Partitions divide a table into logical parts. Each partition can then be processed (Refreshed) independent of other partitions. Partitions created by using the Partitions dialog box in SSDT during model authoring apply to the model workspace database. When the model is deployed, the partitions defined for the model workspace database are duplicated in the deployed model database. You can further create and manage partitions for a deployed model database by using the Partitions dialog box in SSMS. Information provided in this topic describes partitions created during model authoring by using the Partition Manager dialog box in SSDT. For information about creating and managing partitions for a deployed model, see Create and Manage Tabular Model Partitions.

Benefits

Partitions, in tabular models, divide a table into logical partition objects. Each partition can then be processed independent of other partitions. For example, a table may include certain row sets that contain data that rarely changes, but other row sets have data that changes often. In these cases, there is no need to process all of the data when you really just want to process a portion of the data. Partitions enable you to divide portions of data you need to process frequently from the data that can be processed less frequently.

Effective model design utilizes partitions to eliminate unnecessary processing and subsequent processor load on Analysis Services servers, while at the same time, making certain that data is processed and refreshed often enough to reflect the most recent data from data sources. How you implement and utilize partitions during model authoring can be very different from how partitions are implemented and utilized for deployed models. Keep in-mind that during the model authoring phase, you may be working with only a subset of the data that will ultimately be in your deployed model.

Processing partitions

For deployed models, processing occurs by using SSMS, or by running a script which includes the process command and specifies processing options and settings. When authoring models by using SSDT, you can run process operations by using a Process command from the Model menu or toolbar. A Process operation can be specified for a partition, a table, or all.

When a process operation is run, a connection to the data source is made using the data connection. New data is imported into the model tables, relationships and hierarchies are rebuilt for each table, and calculations in calculated columns and measures are re-calculated.

By further dividing a table into logical partitions, you can selectively determine what, when, and how the data in each partition is processed. When you deploy a model, processing of partitions can be completed manually using the Partitions dialog box in SSMS, or by using a script that executes a process command.

Partitions in the model workspace database

You can create new partitions, edit, merge, or delete partitions using the Partition Manager in SSDT. Depending on the compatibility level of the model you are authoring, Partition Manager provides two modes for selecting tables, rows, and columns for a partition: For tabular 1400 models, partitions are defined by using an M query, or you can use Design mode to open Query Editor. For tabular 1100, 1103, 1200 models, use Table Preview mode and SQL query mode.

Partitions in a deployed model database

When you deploy a model, the partitions for the deployed model database will appear as database objects in SSMS. You can create, edit, merge, and delete partitions for a deployed model by using the Partitions dialog box in SSMS. Managing partitions for a deployed model in SSMS is outside the scope of this topic. To learn about managing partitions in SSMS, see Create and Manage Tabular Model Partitions.

Related tasks

TOPIC	DESCRIPTION
Create and Manage Partitions in the Workspace Database	Describes how to create and manage partitions in the model workspace database by using Partition Manager in SSDT.
Process Partitions in the Workspace Database	Describes how to process (refresh) partitions in the model workspace database.

See also

DirectQuery Mode
Process Data

Create and manage partitions in the workspace database

1/18/2019 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Partitions divide a table into logical parts. Each partition can then be processed (Refreshed) independently or in parallel with other partitions. Partitions can improve scalability and manageability of large databases. By default, each table has one partition that includes all columns. Tasks in this topic describe how to create and manage partitions in the model workspace database by using the **Partition Manager** dialog box in SQL Server Data Tools

After a model has been deployed to another Analysis Services instance, database administrators can create and manage partitions in the (deployed) model by using SQL Server Management Studio. For more information, see Create and Manage Tabular Model Partitions.

> **NOTE**
> You cannot merge partitions in the model workspace database by using the Partition Manager dialog box. Partitions can be merged in a deployed model only by using SQL Server Management Studio.

Tasks

To create and manage partitions, you will use the **Partitions Manager** dialog box. To view the **Partitions Manager** dialog box, in SQL Server Data Tools, click the **Table** menu, and then click **Partitions**.

To create a new partition

1. In the model designer, select the table for which you want to define a partition.
2. Click on the **Table** menu, and then click **Partitions**.
3. In **Partition Manager**, in the **Table** listbox, verify or select the table you want to partition, and then click **New**.
4. In **Partition Name**, type a name for the partition. By default, the name of the default partition will be incrementally numbered for each new partition.
5. You can select the rows and columns to be included in the partition by using Table Preview mode or by using a SQL query created using Query Editor mode.

   To use Table Preview mode (default), click the **Table Preview** button near the upper-right corner of the preview window. Select the columns you want to include in the partition by selecting the checkbox next to the column name. To filter rows, right click a cell value, and click **Filter by Selected Cell Value**.

   To use a SQL statement, click the **Query Editor** button near the upper-right corner of the preview window, then type or paste a SQL query statement into the query window. To validate your statement, click **Validate**. To use the Query Designer, click **Design**.

To copy a partition

1. In **Partition Manager**, in the **Table** listbox, verify or select the table that contains the partition you want to copy.
2. In the **Partitions** list, select the partition you want to copy and then click **Copy**.

3. In **Partition Name**, type a new name for the partition.

To delete a partition

1. In **Partition Manager**, in the **Table** listbox, verify or select the table that contains the partition you want to delete.

2. In the **Partitions** list, select the partition you want to delete and then click **Delete**.

See also

Partitions
Process partitions in the workspace database

Process Partitions in the Workspace Database

1/18/2019 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Partitions divide a table into logical parts. Each partition can then be processed (Refreshed) independent of other partitions. The tasks in this topic describe how to process partitions in the model workspace database by using the **Process Partitions** dialog box in SQL Server Data Tools (SSDT).

After a model has been deployed to another Analysis Services instance, database administrators can create and manage partitions in the (deployed) model by using SQL Server Management Studio, by script, or by using an IS package. For more information, see Create and Manage Tabular Model Partitions.

To process a partition

1. In SQL Server Data Tools, click the **Model** menu, and then click **Process** (Refresh), and then click **Process Partitions**.

2. In the **Mode** listbox, select one of the following process modes:

MODE	DESCRIPTION
Process Default	Detects the process state of a partition object, and performs processing necessary to deliver unprocessed or partially processed partition objects to a fully processed state. Data for empty tables and partitions is loaded; hierarchies, calculated columns, and relationships are built or rebuilt.
Process Full	Processes a partition object and all the objects that it contains. When Process Full is run for an object that has already been processed, Analysis Services drops all data in the object, and then processes the object. This kind of processing is required when a structural change has been made to an object.
Process Data	Load data into a partition or a table without rebuilding hierarchies or relationships or recalculating calculated columns and measures.
Process Clear	Removes all data from a partition.
Process Add	Incrementally update partition with new data.

3. In the **Process** checkbox column, select the partitions you want to process with the selected mode, and then click **Ok**.

See Also

Partitions
Create and Manage Partitions in the Workspace Database

Perspectives in tabular models

12/6/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Perspectives, in tabular models, define viewable subsets of a model that provide focused, business-specific, or application-specific viewpoints of the model.

Benefits

Tabular models can be very complex for users to explore. A single model can represent the contents of a complete data warehouse with many tables, measures, and dimensions. Such complexity can be daunting to users who may only need to interact with a small part of the model in order to satisfy their business intelligence and reporting requirements.

In a perspective, tables, columns, and measures (including KPIs) are defined as field objects. You can select the viewable fields for each perspective. For example, a single model may contain product, sales, financial, employee, and geographic data. While a sales department requires product, sales, promotions, and geography data, they likely do not require employee and financial data. Similarly, a human resources department does not require data about sales promotions and geography.

When a user connects to a model (as a data source) with defined perspectives, the user can select the perspective they want to use. For example, when connecting to a model data source in Excel, users in Human Resources can select the Human Resources perspective on the Select Tables and Views page of the Data Connection Wizard. Only fields (tables, columns, and measures) defined for the Human Resources perspective will be visible in the PivotTable Field List.

Perspectives are not meant to be used as a security mechanism, but as a tool for providing a better user experience. All security for a particular perspective is inherited from the underlying model. Perspectives cannot provide access to model objects to which a user does not already have access. Security for the model database must be resolved before access to objects in the model can be provided through a perspective. Security roles can be used to secure model metadata and data. For more information, see Roles.

Testing perspectives

When authoring a model, you can use the Analyze in Excel feature in the model designer to test the efficacy of the perspectives you have defined. From the **Model** menu in the model designer, when you click **Analyze in Excel**, before Excel opens, the **Choose Credentials and Perspective** dialog box appears. In this dialog, you can specify the current username, a different user, a role, and a perspective with which you will use to connect to the model workspace database as a data source and view data.

Related tasks

TOPIC	DESCRIPTION
Create and Manage Perspectives	Describes how to create and manage perspectives using the Perspectives dialog box in the model designer.

See Also

Roles
Hierarchies

Create and manage perspectives

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Perspectives define viewable subsets of a model that provide focused, business-specific, or application-specific viewpoints of the model. The tasks in this topic describe how to create and manage perspectives by using the **Perspectives** dialog box in the model designer.

Tasks

To create perspectives, you will use the **Perspectives** dialog box, where you can add, edit, delete, copy, and view perspectives. To view the **Perspectives** dialog box, in SQL Server Data Tools (SSDT), click on the **Model** menu, and then click **Perspectives**.

To add a perspective

- To add a new perspective, click **New Perspective**. You can then check and uncheck field objects to be included and provide a name for the new perspective.

  If you create an empty perspective with all of the field object fields, then a user using this perspective will see an empty Field List. Perspectives should contain at least one table and column.

To edit a perspective

- To modify a perspective, check and uncheck fields in the perspective's column, which adds and removes field objects from the perspective.

To rename a perspective

- When you hover over a perspective's column header (the name of the perspective), the **Rename** button appears. To rename the perspective, click **Rename**, and then enter a new name or edit the existing name.

To delete a perspective

- When you hover over a perspective's column header (the name of the perspective), the **Delete** button appears. To delete the perspective, click the **Delete** button, and then click **Yes** in the confirmation window.

To copy a perspective

- When you hover over a perspective's column header, the **Copy** button appears. To create a copy of that perspective, click the **Copy** button. A copy of the selected perspective is added as a new perspective to the right of existing perspectives. The new perspective inherits the name of the copied perspective and a - *Copy* annotation is appended to the end of the name. For example, if a copy of the *Sales* perspective is created, the new perspective is called *Sales - Copy*.

See also

Perspectives
Hierarchies

Translations in Tabular models (Analysis Services)

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

SQL Server 2017 adds translation string support for Tabular models. A single object in the model can have multiple translations of a name or description, making it possible to support multi-language versions within the model definition.

Translated strings are for object metadata only (names and descriptions of tables and columns) that appear in a client tool like an Excel PivotTable list. To use translated strings, the client connection specifies the culture. In the **Analysis in Excel** feature, you can choose the language from a drop-down list. For other tools, you might need to specify the culture in the connection string.

This feature is not intended loading translated data into a model. If you want to load translated data values, you should develop a processing strategy that includes extracting translated strings from a data source that provides them.

A typical workflow for adding translated metadata looks like this:

- Generate an empty translation JSON file that contains placeholders for each string translation
- Add string translations to the JSON file
- Import the translations back into the model
- Build, process or deploy the model
- Connect to the model using a client application that allows an LCID on the connection string

Create an empty translation file

Use SQL Server Data Tools to add translations.

1. Click **Model** > **Translations** > **Manage Translations**.
2. Select the languages for which you are providing translations and then click **Add**.
3. Choose one or more languages from the list, depending on how you want to import the strings later.

   Your translation file can include multiple languages, but you might find translation management easier if you create one translation file per language. The translation file you're creating now will be imported in its entirety later. To vary import options by language, each language needs to be in its own file.
4. Click **Export Language File**. Provide a file name and location.

Add translations

An empty JSON translation file includes metadata for a specific language translations. Translation placeholders for object names and descriptions are specified in the **Culture** section at the end of the model definition. Translations can be added for the following:

translatedCaption	Table or column title that appears in any client application supporting visualizations of a Tabular model.
translatedDescription	Less common than captions, a description appears as model information in a modeling tool like SSDT.

Don't delete any unspecified metadata from the file. It should match the file upon which it is based. Just add the strings you want, and then save the file.

Although it might look like something you should modify, the **referenceCulture** section contains the metadata in the default culture of the model. Any changes made to the **referenceCulture** section will not be read in during import and will be ignored.

The following example shows translated captions and description for the **DimProduct** and **DimCustomer** tables.

```
frenchStrings.json*   Model.bim*
Schema: <No Schema Selected>
{
  "referenceCulture": {
  "name": "Tabular1200-AW_a249fce7-f7ea-40e0-8525-e9b3b2ce34b2",
  "id": "Tabular1200-AW_a249fce7-f7ea-40e0-8525-e9b3b2ce34b2",
  "model": ...
},
  "cultures": [
    {
  "name": "fr-CA",
  "translations": {
    "model": {
      "name": "Model",
      "translatedCaption": "",
      "translatedDescription": "",
      "tables": [
        ...,
        {
          "name": "DimCustomer",
          "translatedCaption": "Client",
          "translatedDescription": "Contient des renseignements sur les clients",
          "columns": ...
        },
        ...,
        {
          "name": "DimProduct",
          "translatedCaption": "Produit",
          "translatedDescription": "Contient des informations de produit",
          "columns": ...
        },
        ...,
```

TIP

You can use any JSON editor to open the file, but we recommend using the JSON editor in Visual Studio so that you can also use the View Code command in Solution explorer to view the Tabular model definition in SSDT. To get the JSON editor, you need a full version installation of Visual Studio 2015. The free Community edition includes the JSON editor.

Import a translation file

Translation strings that you import become a permanent part of the model definition. Once the strings are imported, the translation file is no longer referenced.

1. Click **Model** > **Translations** > **Import Translations**.
2. Find the translation file and then click **Open**.
3. Optionally, specify import options.

Overwrite existing translations	Replaces all existing captions or descriptions of the same language as the file being imported.
Ignore invalid objects	Specifies whether metadata discrepancies should be ignored or flagged as an error.
Write import results to a log file	Log files are saved to the project folder by default. The exact path to the file is provided after import concludes. The log file name is SSDT_Translations_Log_<timestamp>.

Back up translations to a JSON file before importing	Backs up an existing translation that matches the culture of the strings being imported. If the culture being imported is not present in the model, then the backup will be empty. If you need to restore this file later, you can replace the contents of the model.bim with this JSON file.

4. Click **Import**.

5. Optionally, if you generated a log file or backup, you can find the files in the project folder (for example, C:\Users\Documents\Visual Studio 2015\Projects\Tabular1200-AW\Tabular1200-AW).

6. To verify the import, follow these steps:

   - Right-click the **model.bim** file in Solution Explorer and choose **View Code**. Click **Yes** to close the design view and reopen **model.bim** in code view. If you installed a full version of Visual Studio, such as the free Community edition, the file opens in the built-in JSON editor.

   - Search for **Culture** or for a specific translated string to verify the strings you expect to see are actually there.

Connect using a locale identifier

This section describes an approach for validating the correct strings are returned from the model.

1. In Excel, connect to the Tabular model. This step assumes the model has been deployed. If the model only exists in the workspace, deploy it to an Analysis Services instance to complete the validation check.

   Alternatively, you can use the **Analyze in Excel** feature to connect to the model

2. In the Excel connection dialog box, choose the culture for which string translations exist in your model. Excel detects cultures defined in the model and populates the drop-down list accordingly.

   When you create a PivotTable, you should see the translated table and column names.

See Also

Compatibility Level for Tabular models in Analysis Services
Globalization scenarios for Analysis Services
Analyze in Excel

Roles

12/7/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Roles, in tabular models, define member permissions for a model. Members of the role can perform actions on the model as defined by the role permission. Roles defined with read permissions can also provide additional security at the row-level by using row-level filters.

For SQL Server Analysis Services, roles contain user members by Windows username or by Windows group, and permissions (read, process, administrator). For Azure Analysis Services, users must be in your Azure Active Directory and usernames and groups specified must be by organizational email address or UPN.

> **IMPORTANT**
>
> When using SSDT to create roles and add organizational users to a tabular model project that will be deployed to Azure Analysis Services, use Integrated workspace.

> **IMPORTANT**
>
> For users to connect to a deployed model by using a reporting client application, you must create at least one role with at least Read permission to which those users are members.

Information in this topic is meant for tabular model authors who define roles by using the Role Manager dialog box in SSDT. Roles defined during model authoring apply to the model workspace database. After a model database has been deployed, model database administrators can manage (add, edit, delete) role members by using SSMS. To learn about managing members of roles in a deployed database, see Tabular Model Roles.

Understanding roles

Roles are used in Analysis Services to manage model data access. There are two types of roles:

- The server role, a fixed role that provides administrator access to an Analysis Services server instance.
- Database roles, roles defined by model authors and administrators to control access to a model database and data for non-administrator users.

Roles defined for a tabular model are database roles. That is, the roles contain members consisting of users or groups that have specific permissions that define the action those members can take on the model database. A database role is created as a separate object in the database, and applies only to the database in which that role is created. Users and groups are included in the role by the model author, which, by default, has Administrator permissions on the workspace database server; for a deployed model, by an administrator.

Roles in tabular models can be further defined with row filters. Row filters use DAX expressions to define the rows in a table, and any related rows in the many direction, that a user can query. Row filters using DAX expressions can only be defined for the Read and Read and Process permissions. To learn more, see Row Filters later in this topic.

By default, when you create a new tabular model project, the model project does not have any roles. Roles can be defined by using the Role Manager dialog box in SSDT. When roles are defined during model authoring,

they are applied to the model workspace database. When the model is deployed, the same roles are applied to the deployed model. After a model has been deployed, members of the server role ([Analysis Services Administrator) and database administrators can manage the roles associated with the model and the members associated with each role by using SSMS.

Permissions

Each role has a single defined database permission (except for the combined Read and Process permission). By default, a new role will have the None permission. That is, once members are added to the role with the None permission, they cannot modify the database, run a process operation, query data, or see the database unless a different permission is granted.

A group or user can be a member of any number of roles, each role with a different permission. When a user is a member of multiple roles, the permissions defined for each role are cumulative. For example, if a user is a member of a role with the Read permission, and also a member of a role with None permission, that user will have Read permissions.

Each role can have one the following permissions defined:

PERMISSIONS	DESCRIPTION	ROW FILTERS USING DAX
None	Members cannot make any modifications to the model database schema and cannot query data.	Row filters do not apply. No data is visible to users in this role
Read	Members are allowed to query data (based on row filters) but cannot see the model database in SSMS, cannot make any changes to the model database schema, and the user cannot process the model.	Row filters can be applied. Only data specified in the row filter DAX formula is visible to users.
Read and Process	Members are allowed to query data (based on row-level filters) and run process operations by running a script or package that contains a process command, but cannot make any changes to the database. Cannot view the model database in SSMS.	Row filters can be applied. Only data specified in the row filter DAX formula can be queried.
Process	Members can run process operations by running a script or package that contains a process command. Cannot modify the model database schema. Cannot query data. Cannot query the model database in SSMS.	Row filters do not apply. No data can be queried in this role
Administrator	Members can make modifications to the model schema and can query all data in the model designer, reporting client, and SSMS.	Row filters do not apply. All data can be queried in this role.

Row filters

Row filters define which rows in a table can be queried by members of a particular role. Row filters are defined for each table in a model by using DAX formulas.

Row filters can be defined only for roles with Read and Read and Process permissions. By default, if a row filter

is not defined for a particular table, members of a role that has Read or Read and Process permission can query all rows in the table unless cross-filtering applies from another table.

Once a row filter is defined for a particular table, a DAX formula, which must evaluate to a TRUE/FALSE value, defines the rows that can be queried by members of that particular role. Rows not included in the DAX formula will cannot be queried. For example, for members of the Sales role, the Customers table with the following row filters expression, *=Customers [Country] = "USA"*, members of the Sales role, will only be able to see customers in the USA.

Row filters apply to the specified rows as well as related rows. When a table has multiple relationships, filters apply security for the relationship that is active. Row filters will be intersected with other row filers defined for related tables, for example:

TABLE	DAX EXPRESSION
Region	=Region[Country]="USA"
ProductCategory	=ProductCategory[Name]="Bicycles"
Transactions	=Transactions[Year]=2008

The net effect of these permissions on the Transactions table is that members will be allowed to query rows of data where the customer is in the USA, and the product category is bicycles, and the year is 2008. Users would not be able to query any transactions outside of the USA, or any transactions that are not bicycles, or any transactions not in 2008 unless they are a member of another role that grants these permissions.

You can use the filter, *=FALSE()*, to deny access to all rows for an entire table.

Dynamic security

Dynamic security provides a way to define row level security based on the user name of the user currently logged on or the CustomData property returned from a connection string. In order to implement dynamic security, you must include in your model a table with login (Windows user name) values for users as well as a field that can be used to define a particular permission; for example, a dimEmployees table with a login ID (domain\username) as well as a department value for each employee.

To implement dynamic security, you can use the following functions as part of a DAX formula to return the user name of the user currently logged on, or the CustomData property in a connection string:

FUNCTION	DESCRIPTION
USERNAME Function (DAX)	Returns the domain\ username of the user currently logged on.
CUSTOMDATA Function (DAX)	Returns the CustomData property in a connection string.

You can use the LOOKUPVALUE function to return values for a column in which the Windows user name is the same as the user name returned by the USERNAME function or a string returned by the CustomData function. Queries can then be restricted where the values returned by LOOKUPVALUE match values in the same or related table.

For example, using this formula:

```
='dimDepartmentGroup'[DepartmentGroupId]=LOOKUPVALUE('dimEmployees'[DepartmentGroupId],
'dimEmployees'[LoginId], USERNAME(), 'dimEmployees'[LoginId], 'dimDepartmentGroup'[DepartmentGroupId])
```

The LOOKUPVALUE function returns values for the dimEmployees[DepartmentId] column where the dimEmployees[LoginId] is the same as the LoginID of the user currently logged on, returned by USERNAME,

and values for dimEmployees[DepartmentId] are the same as values for dimDepartmentGroup[DepartmentId]. The values in DepartmentId returned by LOOKUPVALUE are then used to restrict the rows queried in the dimDepartment table, and any tables related by DepartmentId. Only rows where DepartmentId are also in the values for the DepartmentId returned by LOOKUPVALUE function are returned.

dimEmployees

LASTNAME	FIRSTNAME	LOGINID	DEPARTMENTNAME	DEPARTMENTID
Brown	Kevin	Adventure-works\kevin0	Marketing	7
Bradley	David	Adventure-works\david0	Marketing	7
Dobney	JoLynn	Adventure-works\JoLynn0	Production	4
Baretto DeMattos	Paula	Adventure-works\Paula0	Human Resources	2

dimDepartment

DEPARTMENTID	DEPARTMENTNAME
1	Corporate
2	Executive General and Administration
3	Inventory Management
4	Manufacturing
5	Quality Assurance
6	Research and Development
7	Sales and Marketing

Testing roles

When authoring a model project, you can use the Analyze in Excel feature to test the efficacy of the roles you have defined. From the **Model** menu in the model designer, when you click **Analyze in Excel**, before Excel opens, the **Choose Credentials and Perspective** dialog box appears. In this dialog, you can specify the current username, a different username, a role, and a perspective with which you will use to connect to the workspace model as a data source. To learn more, see Analyze in Excel.

Related tasks

TOPIC	DESCRIPTION

TOPIC	DESCRIPTION
Create and Manage Roles	Tasks in this topic describe how to create and manage roles by using the **Role Manager** dialog box.

See Also

Perspectives
Analyze in Excel
USERNAME Function (DAX)
LOOKUPVALUE Function (DAX)
CUSTOMDATA Function (DAX)

Create and manage roles

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Roles, in tabular models, define member permissions for a model. Roles are defined for a model project by using the Role Manager dialog box in SQL Server Data Tools (SSDT).

> **IMPORTANT**
> If you're deploying your project to Azure Analysis Services, use **Integrated Workspace** as your workspace database. To learn more, see Workspace database.

The tasks in this article describe how to create and manage roles during model authoring by using the Role Manager dialog box in SQL Server Data Tools (SSDT). For information about managing roles in a deployed model database, see Tabular Model Roles.

Tasks

To create, edit, copy, and delete roles, you will use the **Role Manager** dialog box. To view the **Role Manager** dialog box, in SQL Server Data Tools, click the **Model** menu, and then click **Role Manager**.

To create a new role

1. In SQL Server Data Tools, click the **Model** menu, and then click **Role Manager**.
2. In the **Role Manager** dialog box, click **New**.

   A new highlighted role is added to the Roles list.
3. In the **Roles** list, in the **Name** field, type a name for the role.

   By default, the name of the default role will be incrementally numbered for each new role. It is recommended you type a name that clearly identifies the member type, for example, Finance Managers or Human Resources Specialists.
4. In the **Permissions** field, click the down arrow and then select one of the following permission types:

PERMISSION	DESCRIPTION
None	Members cannot make any modifications to the model schema and cannot query data.
Read	Members are allowed to query data (based on row filters) but cannot make any changes to the model schema.
Read and Process	Members are allowed to query data (based on row-level filters) and run Process and Process All operations, but cannot make any changes to the model schema.
Process	Members can run Process and Process All operations. Cannot modify the model schema and cannot query data.

PERMISSION	DESCRIPTION
Administrator	Members can make modifications to the model schema and can query all data.

5. To enter a description for the role, click the **Description** field, and then type a description.
6. If the role you are creating has Read or Read and Process permission, you can add row filters using a DAX formula. To add row filters, click the **Row Filters** tab, then select a table, then click the **DAX Filter** field, and then type a DAX formula.
7. To add members to the role, click the **Members** tab, and then click **Add**.

> **NOTE**
>
> Role members can also be added to a deployed model by using SQL Server Management Studio. For more information, see Manage Roles by using SSMS.

8. In the **Select Users or Groups** dialog box, enter Windows user or Windows group objects as members.
9. Click **Ok**.

See also

Roles
Perspectives
Analyze in Excel
USERNAME Function (DAX)
CUSTOMDATA Function (DAX)

Object-level security

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data model security starts with effectively implementing roles and row-level filters to define user permissions on data model objects and data. Beginning with tabular 1400 models, you can also define object-level security, which includes table-level security and column-level security in the Roles object.

Table-level security

With table-level security, you can not only restrict access to table data, but also sensitive table names, helping prevent malicious users from discovering if a table exists.

Table-level security is set in the JSON-based metadata in the Model.bim, Tabular Model Scripting Language (TMSL), or Tabular Object Model (TOM). Set the **metadataPermission** property of the **tablePermissions** class in the Roles object to **none**.

In this example, the metadataPermission property of the tablePermissions class for the Product table is set to none:

```
"roles": [
  {
    "name": "Users",
    "description": "All allowed users to query the model",
    "modelPermission": "read",
    "tablePermissions": [
      {
        "name": "Product",
        "metadataPermission": "none"
      }
    ]
  }
```

Column-level security

Similar to table-level security, with column-level security you can not only restrict access to column data, but also sensitive column names, helping prevent malicious users from discovering a column.

Column-level security is set in the JSON-based metadata in the Model.bim, Tabular Model Scripting Language (TMSL), or Tabular Object Model (TOM). Set the **metadataPermission** property of the **columnPermissions** class in the Roles object to **none**.

In this example, the metadataPermission property of the columnPermissions class for the Base Rate column in the Employees table is set to none:

```
"roles": [
  {
    "name": "Users",
    "description": "All allowed users to query the model",
    "modelPermission": "read",
    "tablePermissions": [
      {
        "name": "Employee",
        "columnPermissions": [
          {
            "name": "Base Rate",
            "metadataPermission": "none"
          }
        ]
      }
    ]
  }
```

Restrictions

- Table-level security cannot be set for a model if it breaks a relationship chain. An error is generated at design time. For example, if there are relationships between tables A and B, and B and C, you cannot secure table B. If table B is secured, a query on table A cannot transit the relationships between table A and B, and B and C. In this case, a separate relationship could be configured between tables A and B.

- Row-level security and object-level security cannot be combined from different roles because it could introduce unintended access to secured data. An error is generated at query time for users who are members of such a combination of roles.
- Dynamic calculations (measures, KPIs, DetailRows) are automatically restricted if they reference a secured table or column. While there is no mechanism to explicitly secure a measure, it is possible to implicitly secure a measure by updating the expression to refer to a secured table or column.
- Relationships that reference a secured column work provided the table the column is in is not secured.

See Also

Roles
Roles object (TMSL)
Tabular Model Scripting Language (TMSL)
Tabular Object Model (TOM).

Analyze in Excel

12/6/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Analyze in Excel feature, in SSDT, provides tabular model authors a way to quickly analyze model projects during development. The Analyze in Excel feature opens Microsoft Excel, creates a data source connection to the model workspace database, and automatically adds a PivotTable to the worksheet. Workspace database objects (tables, columns, and measures) are included as fields in the PivotTable Field List. Objects and data can then be viewed within the context of the effective user or role and perspective.

This article assumes you are already familiar with Microsoft Excel, PivotTables, and PivotCharts. To learn more about using Excel, see Excel Help.

Benefits

The Analyze in Excel feature provides model authors the ability to test the efficacy of a model project by using the common data analysis application, Microsoft Excel. In order to use the Analyze in Excel feature, you must have Microsoft Office 2003 or later installed on the same computer as SSDT.

The Analyze in Excel feature opens Excel and creates a new Excel workbook (.xls). A data connection from the workbook to the model workspace database is created. A blank PivotTable is added to the worksheet and model object metadata is populated in the PivotTable Field list. You can then add viewable data and slicers to the PivotTable.

When using the Analyze in Excel feature, by default, the user account currently logged on is the effective user. This account is typically an Administrator with no view restrictions to model objects or data. You can, however, specify a different effective username or role. This allows you to browse a model project in Excel within the context of a specific user or role. Specifying the effective user includes the following options:

Current Windows User
Uses the user account with which you are currently logged on.

Other Windows User
Uses a specified Windows username rather than the user currently logged on. Using a different Windows user does not require a password. Objects and data can only be viewed in Excel within the context of the effective username. No changes to model objects or data can be made in Excel.

Role
A role is used to define user permissions on the object metadata and data. Roles are usually defined for a particular Windows user or Windows user group. Certain roles can include additional row-level filters defined in a DAX formula. When using the Analyze in Excel feature, you can optionally select a role to be used. Object metadata and data views will be restricted by the permission and filters defined for the role. For more information, see Create and Manage Roles.

In addition to the effective user or role, you can specify a perspective. Perspectives enable model authors to define particular business scenario views of model objects and data. By default, no perspective is used. In order to use a perspective with Analyze in Excel, perspectives must already be defined by using the Perspectives dialog box in SSDT. If a perspective is specified, the PivotTable Field List will contain only those objects selected in the perspective. For more information, see Create and Manage Perspectives.

Related tasks

TOPIC	DESCRIPTION
Analyze a tabular model in Excel	This article describes how to use the Analyze in Excel feature in the model designer to open Excel, create a data source connection to the model workspace database, and add a PivotTable to the worksheet.

See also

Analyze a tabular model in Excel
Roles
Perspectives

Analyze a tabular model in Excel

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Analyze in Excel feature in SQL Server Data Tools (SSDT) opens Microsoft Excel, creates a data source connection to the model workspace database, and adds a PivotTable to the worksheet. Model objects (tables, columns, measures, hierarchies, and KPIs) are included as fields in the PivotTable field list.

> **NOTE**
> To use the Analyze in Excel feature, you must have Microsoft Office 2003 or later installed on the same computer as SQL Server Data Tools. If Office is not installed on the same computer, you can use Excel on another computer and connect to the model workspace database as a data source. You can then manually add a PivotTable to the worksheet. Model objects (tables, columns, measures, and KPIs) are included as fields in the PivotTable field list.

Tasks

To analyze a tabular model project by using the Analyze in Excel feature

1. In SQL Server Data Tools, click the **Model** menu, and then click **Analyze in Excel**.
2. In the **Choose Credential and Perspective** dialog box, select one of the following credential options to connect to the model workspace data source:
   - To use the current user account, select **Current Windows User**.
   - To use a different user account, select **Other Windows User**.

     Typically, this user account will be a member of a role. No password is required. The account can only be used in the context of an Excel connection to the workspace database.
   - To use a security role, select **Role**, and then in the listbox, select one or more roles.

     Security roles must be defined using the Role Manager. For more information, see Create and Manage Roles.
3. To use a perspective, in the **Perspective** listbox, select a perspective.

   Perspectives (other than default) must be defined using the Perspectives dialog box. For more information, see Create and Manage Perspectives.

> **NOTE**
> The PivotTable Field List in Excel does not refresh automatically as you make changes to the model project in the model designer. To refresh the PivotTable Field List, in Excel, on the **Options** ribbon, click **Refresh**.

See also

Analyze in Excel

String storage and collation in tabular models

12/7/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Strings (text values) are stored in a highly compressed format in tabular models; because of this compression, you can get unexpected results when you retrieve entire or partial strings. Also, because string locale and collations are inherited hierarchically from the closest parent object, if the string language is not explicitly defined, the locale and collation of the parent can affect how each string is stored and whether the string is unique or conflated with similar strings as defined by the parent collation.

This article describes the mechanism by which strings are compressed and stored, and provides examples of how collation and language affect the results of text formulas in tabular models.

Storage

In tabular models all data is highly compressed to better fit in memory. As a consequence, all strings that can be considered lexically equivalent are stored only once. The first instance of the string is used as the canonical representation and thereafter each equivalent string is indexed to the same compressed value as the first occurrence.

The key question is: what constitutes a lexically equivalent string? Two strings are considered lexically equivalent if they can be considered as the same word. For example, in English when you search for the word **violin** in a dictionary, you might find the entry **Violin** or **violin**, depending on the editorial policy of the dictionary, but generally you consider both words equivalent, and disregard the difference in capitalization. In a tabular model, the factor that determines whether two strings are lexically equivalent is not editorial policy or even user preference, but the locale and collation order assigned to the column.

Therefore, the decision of whether uppercase and lowercase letters should be handled as the same or different depends on the collation and locale. For any particular word within that locale, the first occurrence of the word that is found within a particular column therefore serves as the canonical representation of that word and that string is stored in uncompressed format. All other strings are tested against the first occurrence, and if they match the equivalence test, they are assigned to the compressed value of the first occurrence. Later, when the compressed values are retrieved they are represented using the uncompressed value of the first occurrence of the string.

An example will help to clarify how this works. The following column "Classification - English" was extracted from a table that contains information about plants and trees. For each plant (the names of the plants are not shown here) the classification column shows the general category of plant.

CLASSIFICATION - ENGLISH
trEE
PlAnT
TREE
PLANT
Plant

CLASSIFICATION - ENGLISH
Tree
plant
tReE
tree
pLaNt
tREE

Perhaps the data came from many different sources, and so the casing and use of accents was inconsistent, and the relational database stored those differences as is. But in general the values are either **Plant** or **Tree**, just with different casing.

When these values are loaded into a tabular model that uses the default collation and sorting order for American English, case is not important, so only two values would be stored for the entire column:

CLASSIFICATION - ENGLISH
trEE
PlAnT

If you use the column, **Classification - English**, in your model, wherever you display plant classification you will see not the original values, with their various uses of upper and lower case, but only the first instance. The reason is that all the uppercase and lowercase variants of **tree** are considered equivalent in this collation and locale; therefore, only one string was preserved and the first instance of that string that is encountered by the system is the one that is saved.

WARNING

You might decide that you want to define which string will be the first to store, according to what you consider correct, but this could be very hard to so. There is no simple way to determine in advance which row should be processed first by the engine, given that all values are considered to be the same. Instead, if you need to set the standard value, you should cleanse all your strings before loading the model.

Locale and Collation Order

When comparing strings (text values), what defines equivalence is normally the cultural aspect of how such strings are interpreted. In some cultures an accent or the capitalization of a character can completely change the meaning of the string; therefore, typically such differences are considered when determining equivalency for any particular language or region.

Usually, when you use your computer it is already configured to match your own cultural expectations and linguistic behavior, and string operations such as sorting and comparing text values behaves as you would expect. The settings that control language-specific behavior are defined through the **Locale and Regional** settings in Windows. Applications read those settings and change their behavior accordingly. In some cases, an application might have a feature that allows you to change the cultural behavior of the application or the way in which strings are compared.

When you are creating a tabular model database, by default the database inherits these cultural and linguistic settings in the form of a language identifier and collation.

- The language identifier defines the character set you want to use for your strings according to your culture.
- The collation defines the ordering of the characters and their equivalence.

It is important to note that a language identifier not only identifies a language but, also the country or region where the language is used. Each language identifier also has a default collation specification. For more information about language identifiers, see Locale IDs Assigned by Microsoft. You can use the LCID Dec column to get the correct ID when manually inserting a value. For more information about the SQL concept of collations, see COLLATE (Transact-SQL). For information about the collation designators and the comparison styles for Windows collation names, see Windows Collation Name (Transact-SQL). The topic, SQL Server Collation Name (Transact-SQL), maps the Windows collation names to the names used for SQL.

Once your tabular model database has been created, all new objects in the model will inherit the language and collation attributes from the database attributes. This is true for all objects. The inheritance path begins at the object, looks at the parent for any language and collation attributes to inherit, and if none are found, continues to the top and finds the language and collation attributes at the database level. In other words, if you do not specify the language and collation attributes for an object, by default, the object inherits the attributes of its closest parent.

For columns, the language and collation attributes are inherited at creation, according to the following rules:

1. The parent dimension object is searched for language and collation attributes. If both values exist, they are copied to the column attributes; if only one exists, the other is inferred from the existing one and both are assigned; if none exist, move to next step.
2. The database object is searched using the same process described in Step 1 for dimensions; if no attributes are found, move to the next step.
3. The server object is searched using the same process described in Step 1 for dimensions; if no attributes are found, the column uses the Windows language identifier and infers the collation attribute from that value.

It is important to note that typically the language identifier and collation order in the source database has little to no effect on how values are stored in the tabular model column. The exception is if the source database transforms or filters the requested values.

Tabular model solution deployment

12/7/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After authoring a tabular model project, you must deploy it in order for users to browse the model by using a reporting client application. This article describes the various properties and methods you can use when deploying tabular model solutions in your environment.

Benefits

Deploying a tabular model creates a model database in a test, staging, or production environment. Users can then connect to the deployed model through a .bism connection file in Sharepoint or by using a data connection directly from reporting client applications such as Microsoft Excel, Power View, or a custom application. The model workspace database, created when you create a new tabular model project in SQL Server Data Tools (SSDT), and used to author the model will remain on the workspace server instance, allowing you to make changes to the model project and then re-deploying to the test, staging, or production environment when necessary.

Deploying a tabular model from SQL Server Data Tools (SSDT)

Deploying is a simple process; however, certain steps must be taken to ensure your model is deployed to the correct Analysis Services instance and with the correct configuration options.

Tabular models are defined with several deployment-specific properties. When you deploy, a connection to the Analysis Services instance specified in the **Server** property is established. A new model database with the name specified in the **Database** property is then created on that instance, if one does not already exist. Metadata from the model project's Model.bim file is used to configure objects in the model database on the deployment server. With the **Processing Option**, you can specify whether or not just the model metadata is deployed, creating the model database, or if **Default** or **Full** is specified, impersonation credentials used to connect to data sources are passed in-memory from the model workspace database to the deployed model database. Analysis Services then runs processing to populate data into the deployed model. Once the deployment process is complete, the model can then be connected to by client applications using a data connection or by using an .bism connection file in SharePoint.

Deployment properties

The project deployment options and deployment server properties specify how and where a model is deployed to a staging or production Analysis Services environment. While default property settings are defined for all model projects, depending on your particular deployment requirements, you can change these property settings for each project. For more information about setting default deployment properties, see Configure default data modeling and deployment properties.

Deployment options properties

Deployment options properties include the following:

PROPERTY	DEFAULT SETTING	DESCRIPTION

PROPERTY	DEFAULT SETTING	DESCRIPTION
Processing Option	**Default**	This property specifies the type of processing required when changes to objects are deployed. This property has the following options: **Default** - This setting specifies Analysis Services will determine the type of processing required. Unprocessed objects will be processed, and if required, recalculating attribute relationships, attribute hierarchies, user hierarchies, and calculated columns. This setting generally results in a faster deployment time than using the Full processing option. **Do Not Process** - This setting specifies only the metadata will be deployed. After deploying, it may be necessary to run a process operation on the deployed model to update and recalculate data. **Full** - This setting specifies that both the metadata is deployed and a process full operation is performed. This assures that the deployed model has the most recent updates to both metadata and data.
Transactional Deployment	**False**	This property specifies whether or not the deployment is transactional. By default, the deployment of all or changed objects is not transactional with the processing of those deployed objects. Deployment can succeed and persist even though processing fails. You can change this to incorporate deployment and processing in a single transaction.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Query Mode	**In-Memory**	This property specifies the mode in which the source from which query results are returned is running in In-Memory (cached) mode or in DirectQuery mode. This property has the following options: **DirectQuery** - This setting specifies all queries to the model should use the relational data source only. **DirectQuery with In-Memory** - This setting specifies, by default, queries should be answered by using the relational source, unless otherwise specified in the connection string from the client. **In-Memory** - This setting specifies queries should be answered by using the cache only. **In-Memory with DirectQuery** - This setting specifies, by default. queries should be answered by using the cache, unless otherwise specified in the connection string from the client. For more information, see DirectQuery Mode.

Deployment server properties

Deployment Server properties include the following:

PROPERTY	DEFAULT SETTING	DESCRIPTION
Server Set when the project is created.	**localhost**	This property, set when the project is created, specifies the Analysis Services instance by name to which the model will be deployed. By default, the model will be deployed to the default instance of Analysis Services on the local computer. However, you can change this setting to specify a named instance on the local computer or any instance on any remote computer on which you have permission to create Analysis Services objects.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Edition	The same edition as the instance in which the Workspace Server is located.	This property specifies the edition of the Analysis Services server to which the model will be deployed. The server edition defines various features that can be incorporated into the project. By default, the edition will be of the local Analysis Services server. If you specify a different Analysis Services server, for example, a production Analysis Services server, be sure to specify the edition of that Analysis Services server.
Database	**<projectname>**	This property specifies the name of the Analysis Services database in which model objects will be instantiated upon deployment. This name will also be specified in a reporting client data connection or an .bism data connection file. You can change this name at any time when you are authoring the model. If you change the name after you have deployed the model, changes that you have made after deployment will not affect the model that you previously deployed. For example, if you open a solution named **TestDB** and deploy your solution with the default model Database name Model, and then modify the solution and renamed the model Database **Sales**, the instance of Analysis Services the solutions were deployed to will display separate databases, one named Model and one named Sales.
Cube Name	**Model**	This property specifies the cube name as shown in client tools (such as Excel) and AMO (Analysis Management Objects).

DirectQuery options properties

Deployment Options properties include the following:

PROPERTY	DEFAULT SETTING	DESCRIPTION

PROPERTY	DEFAULT SETTING	DESCRIPTION
Impersonation Settings	**Default**	This property specifies the impersonation settings used when a model running in DirectQuery mode connects to data sources. Impersonation credentials are not used when querying the in-memory cache. This property setting has the following options: **Default** - This setting specifies Analysis Services will use the option specified on the Impersonation Information page when the data source connection was created by using the Table Import Wizard. **ImpersonateCurrentUser** - This setting specifies the user account of the user currently logged on will be used when connecting to all data sources.

Deployment methods

There are several methods you can use to deploy a tabular model project. Most of the deployment methods that can be used for other Analysis Services projects, such as multidimensional, can also be used to deploy tabular model projects.

METHOD	DESCRIPTION	LINK
Deploy command in SQL Server Data Tools	The Deploy command provides a simple and intuitive method to deploy a tabular model project from the SQL Server Data Tools authoring environment. **Caution** This method should not be used to deploy to production servers. Using this method can overwrite certain properties in an already deployed, existing model; for example, when using scripts or SSMS to modify properties.	Deploy From SQL Server Data Tools
Analysis Management Objects (AMO) Automation	AMO provides a programmatic interface to the complete command set for Analysis Services, including commands that can be used for solution deployment. As an approach for solution deployment, AMO automation is the most flexible, but it also requires a programming effort. A key advantage to using AMO is that you can use SQL Server Agent with your AMO application to run deployment on a preset schedule.	Developing with Analysis Management Objects (AMO)

METHOD	DESCRIPTION	LINK
XMLA	Use SQL Server Management Studio to generate an XMLA script of the metadata of an existing Analysis Services database, and then run that script on another server to recreate the initial database. XMLA scripts are easily formed in SQL Server Management Studio by defining the deployment process, then codifying it and saving it in an XMLA script. Once you have the XMLA script in a saved file, you can easily run the script according to a schedule, or embed the script in an application that connects directly to an instance of Analysis Services. You can also run XMLA Scripts on a preset basis using SQL Server Agent, but you do not have the same flexibility with XMLA Scripts as with AMO. AMO provides a larger breadth of functionality by hosting the complete spectrum of administrative commands.	Deploy Model Solutions Using XMLA
Deployment Wizard	Use the Deployment Wizard to use the XMLA output files generated by an Analysis Services project to deploy the project's metadata to a destination server. With the Deployment Wizard, you can deploy directly from the Analysis Services file, as created by the output directory by project build. The primary advantage of using the Analysis Services Deployment Wizard is convenience. Just as you can save an XMLA script for use later in SQL Server Management Studio, you can save Deployment Wizard scripts. The Deployment Wizard can be run both interactively and at the command prompt via the Deployment Utility.	Deploy Model Solutions Using the Deployment Wizard
Deployment Utility	The Deployment utility lets you start the Analysis Services deployment engine from a command prompt.	Deploy Model Solutions with the Deployment Utility

METHOD	DESCRIPTION	LINK
Synchronize Database Wizard	Use the Synchronize Database Wizard to synchronize the metadata and data between any two Analysis Services databases. The Synchronize Wizard can be used to copy both data and metadata from a source server to a destination server. If the destination server does not have a copy of the database that you want to deploy, a new database is copied to the destination server. If the destination server already has a copy of the same database, the database on the destination server is updated to use the metadata and data of the source database.	Synchronize Analysis Services Databases
Backup and Restore	Backup offers the simplest approach to transferring Analysis Services databases. From the **Backup** dialog box, you can set the options configuration, and then you can run the backup from the dialog box itself. Or, you can create a script that can be saved and run as frequently as required. Backup and restore is not used as frequently as the other deployment methods, but is a way to quickly complete a deployment with minimal infrastructure requirements.	Backup and Restore of Analysis Services Databases

Configuring the deployment server and connecting to a deployed model

After a model has been deployed, there are additional considerations for securing model data access, backups, and processing operations that can be configured on the Analysis Services server by using SQL Server Management Studio. While these properties and configuration settings are outside the scope of this topic, they are, nonetheless, very important in assuring your deployed model data is secure, kept up to date, and provide a valuable data analysis resource for users in your organization.

After a model has been deployed, and optional server settings configured, the model can be connected to by reporting client applications and used to browse and analyze the model metadata. Connecting to a deployed model database from client applications is outside the scope of this topic.

Related tasks

TASK	DESCRIPTION
Deploy From SQL Server Data Tools	Describes how to configure deployment properties and deploy a tabular model project by using the Deploy command in SQL Server Data Tools.

TASK	DESCRIPTION
Deploy Model Solutions Using the Deployment Wizard	Topics in this section describe how to use the Analysis Services Deployment Wizard to deploy both tabular and multidimensional model solutions.
Deploy Model Solutions with the Deployment Utility	Describes how to use the Analysis Services Deployment Utility to deploy tabular and multidimensional model solutions.
Deploy Model Solutions Using XMLA	Describes how to use XMLA to deploy Analysis Services tabular and multidimensional solutions.
Synchronize Analysis Services Databases	Describes how to use the Synchronize Database Wizard to synchronize the metadata and data between any two Analysis Services tabular or multidimensional databases.

See Also

Connect to a tabular model database

Deploy From SQL Server Data Tools

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use the tasks in this topic to deploy a tabular model solution by using the Deploy command in SSDT.

Configure deployment options and deployment server properties

Before you deploy your tabular model solution, you must first specify the Deployment Options and Deployment Server properties. For more information about deployment properties and settings, see Tabular model solution deployment.

To configure options and properties

1. In SSDT, in **Solution Explorer**, right-click the project name, and then click **Properties**.
2. In the **<project name> Properties** dialog, in **Deployment Options**, specify property settings if different from the default settings.

   > **NOTE**
   > For models in cached mode, **Query Mode** is always **In-Memory**.

   > **NOTE**
   > You cannot specify **Impersonation Settings** for models in DirectQuery mode.

3. In **Deployment Server**, specify the **Server** (name), **Edition**, **Database** (name), and **Cube Name** property settings, if different from the default settings, and then click **OK**.

> **NOTE**
> You can also specify the Default Deployment Server property setting so any new projects you create will automatically be deployed to the specified server. For more information, see Configure default data modeling and deployment properties.

Deploy a tabular model

To deploy a tabular model

- In SSDT, on the **Build** menu, click **Deploy <project name>**.

  The **Deploy** dialog box will appear and indicate the status of the metadata deployment and the processing (unless Processing Option property is set to Do Not Process) of each table included in the model. After the deployment process is complete, use SSMS to connect to the Analysis Services instance and verify the new model database object has been created or use a client reporting application to connect to the deployed model.

Deploy Status

The **Deploy** dialog box enables you to monitor the progress of a Deploy operation. A deploy operation can also be stopped.

Status
Indicates whether the Deploy operation was successful or not.

Details
Lists the metadata items that were deployed, the status for each metadata item, and provides a message of any issues.

Stop Deploy
Click to halt the Deploy operation. This option is useful if the Deploy operation is taking too long, or if there are too many errors.

See also

Tabular model solution deployment
Configure default data modeling and deployment properties

Manage Deployed Tabular Model Databases

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A tabular model database is an Analysis Services database that you create in SQL Server Data Tools (SSDT) using the Analysis Services Tabular model project template, where tables and relationships are the primary structures for representing data.

In This Section

Process Database, Table, or Partition (Analysis Services)

Tabular Model Roles

Tabular Model Partitions

Connect to a tabular model database

Restore from Power Pivot

See also

Tabular Model Solutions

Move an Analysis Services Database

Synchronize Analysis Services Databases

Switch an Analysis Services database between ReadOnly and ReadWrite modes

Modify or Delete an Analysis Services Database

Document and Script an Analysis Services Database

Backup and Restore of Analysis Services Databases

Attach and Detach Analysis Services Databases

Process Database, Table, or Partition (Analysis Services)

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The tasks in this topic describe how to process a tabular model database, table, or partitions manually by using the **Process <object>** dialog box in SQL Server Management Studio.

For more information about tabular model processing, see Process Data.

Tasks

To process a database

1. In SQL Server Management Studio, right-click on the database you want to process, and then click **Process Database**.

2. In the **Process Database** dialog box, in the **Mode** listbox, select one of the following process modes:

MODE	DESCRIPTION
Process Default	Detects the process state of database objects, and performs processing necessary to deliver unprocessed or partially processed objects to a fully processed state. Data for empty tables and partitions is loaded; hierarchies, calculated columns, and relationships are built or rebuilt (recalculated).
Process Full	Processes a database and all the objects that it contains. When Process Full is run for an object that has already been processed, Analysis Services drops all data in the object, and then processes the object. This kind of processing is required when a structural change has been made to an object. This option requires the most resources.
Process Clear	Removes all data from database objects.
Process Recalc	Updates and recalculates hierarchies, relationships, and calculated columns.

3. In the **Process** checkbox column, select the partitions you want to process with the selected mode, and then click **Ok**.

To process a table

1. In SQL Server Management Studio, in the tabular model database which contains the table you want to process, expand the **Tables** node, then right-click on the table you want to process, and then click **Process Table**.

2. In the **Process Table** dialog box, in the **Mode** listbox, select one of the following process modes:

MODE	DESCRIPTION
Process Default	Detects the process state of a table object, and performs processing necessary to deliver unprocessed or partially processed objects to a fully processed state. Data for empty tables and partitions is loaded; hierarchies, calculated columns, and relationships are built or rebuilt (recalculated).
Process Full	Processes a table object and all the objects that it contains. When Process Full is run for an object that has already been processed, Analysis Services drops all data in the object, and then processes the object. This kind of processing is required when a structural change has been made to an object. This option requires the most resources.
Process Data	Load data into a table without rebuilding hierarchies or relationships or recalculating calculated columns and measures.
Process Clear	Removes all data from a table and any table partitions.
Process Defrag	Defragments the auxiliary table indexes.

3. In the table checkbox column, verify the table and optionally select any additional tables you want to process, and then click **Ok**.

To process one or more partitions

1. In SQL Server Management Studio, right-click on the table that has the partitions you want to process, and then click **Partitions**.
2. In the **Partitions** dialog box, in **Partitions**, click the Process button.
3. In the **Process Partition** dialog box, in the **Mode** listbox, select one of the following process modes:

MODE	DESCRIPTION
Process Default	Detects the process state of a partition object, and performs processing necessary to deliver unprocessed or partially processed partition objects to a fully processed state. Data for empty tables and partitions is loaded; hierarchies, calculated columns, and relationships are built or rebuilt (recalculated).
Process Full	Processes a partition object and all the objects that it contains. When Process Full is run for an object that has already been processed, Analysis Services drops all data in the object, and then processes the object. This kind of processing is required when a structural change has been made to an object.
Process Data	Load data into a partition or a table without rebuilding hierarchies or relationships or recalculating calculated columns and measures.
Process Clear	Removes all data from a partition.

MODE	DESCRIPTION
Process Add	Incrementally update partition with new data.

4. In the **Process** checkbox column, select the partitions you want to process with the selected mode, and then click **Ok**.

See Also

Tabular Model Partitions
Create and Manage Tabular Model Partitions

Tabular Model Roles

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Information in this topic and related tasks is meant for tabular model database administrators who will create and manage roles for deployed tabular models by using SQL Server Management Studio. To learn about roles in tabular models, see Roles. To learn about creating and managing roles in a tabular model project by using Role Manager in SQL Server Data Tools (SSDT), see Create and Manage Roles.

Related Tasks

TASK	DESCRIPTION
Manage Roles by using SSMS	Describes how to create roles, add and remove members from a role, change permissions for a role, and add, edit, or remove row-level filters by using DAX.

Manage Roles by using SSMS

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can create, edit, and manage roles for a deployed tabular model by using SQL Server Management Studio.

Tasks in this topic:

- To create a new role
- To copy a role
- To edit a role
- To delete a role

Caution

Re-deploying a tabular model project with roles defined by using Role Manager in SQL Server Data Tools will overwrite roles defined in a deployed tabular model.

Caution

Using SQL Server Management Studio to manage a tabular model workspace database while the model project is open in SQL Server Data Tools (SSDT) may cause the Model.bim file to become corrupted. When creating and managing roles for a tabular model workspace database, use Role Manager in SQL Server Data Tools.

To create a new role

1. In SQL Server Management Studio, expand the tabular model database for which you want to create a new role, then right click on **Roles**, and then click **New Role**.
2. In the **Create Role** dialog box, in the Select a page window, click **General**.
3. In the general settings window, in the **Name** field, type a name for the role.

   By default, the name of the default role will be incrementally numbered for each new role. It is recommended you type a name that clearly identifies the member type, for example, Finance Managers or Human Resources Specialists.
4. In **Set the database permissions for this role**, select one of the following permissions options:

PERMISSION	DESCRIPTION
Full control (Administrator)	Members can make modifications to the model schema and can view all data.
Process database	Members can run Process and Process All operations. Cannot modify the model schema and cannot view data.
Read	Members are allowed to view data (based on row filters) but cannot make any changes to the model schema.

5. In the **Create Role** dialog box, in the Select a page window, click **Membership**.
6. In the membership settings window, click **Add**, and then in the **Select Users or Groups** dialog box, add the Windows users or groups you want to add as members.

7. If the role you are creating has Read permissions, you can add row filters for any table using a DAX formula. To add row filters, in the **Role Properties - <rolename>** dialog box, in **Select a page**, click on **Row Filters**.

8. In the row filters window, select a table, then click on the **DAX Filter** field, and then in the **DAX Filter - <tablename>** field, type a DAX formula.

> **NOTE**
>
> The DAX Filter - <tablename> field does not contain an AutoComplete query editor or insert function feature. To use AutoComplete when writing a DAX formula, you must use a DAX formula editor in SQL Server Data Tools.

9. Click **Ok** to save the role.

To copy a role

1. In SQL Server Management Studio, expand the tabular model database that contains the role you want to copy, then expand **Roles**, then right click on the role, and then click **Duplicate**.

To edit a role

- In SQL Server Management Studio, expand the tabular model database that contains the role you want to edit, then expand **Roles**, then right click on the role, and then click **Properties**.

  In the **Role Properties** <rolename> dialog box, you can change permissions, add or remove members, and add/edit row filters.

To delete a role

- In SQL Server Management Studio, expand the tabular model database that contains the role you want to delete, then expand **Roles**, then right click on the role, and then click **Delete**.

See Also

Roles

Tabular Model Partitions

12/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Partitions divide a table into logical parts. Each partition can then be processed (Refreshed) independent of other partitions. Partitions defined for a model during model authoring are duplicated in a deployed model. Once deployed, you can manage those partitions and create new partitions by using the **Partitions** dialog box in SQL Server Management Studio or by using a script. Information provided in this topic describes partitions in a deployed tabular model database. For more information about creating and managing partitions during model authoring, see Partitions.

Sections in this topic:

- Benefits
- Permissions
- Process Partitions
- Parallel Processing
- Related Tasks

Benefits

Effective model design utilizes partitions to eliminate unnecessary processing and subsequent processor load on Analysis Services servers, while at the same time, making certain that data is processed and refreshed often enough to reflect the most recent data from data sources.

For example, a tabular model can have a Sales table which includes sales data for the current 2011 fiscal year and each of the previous fiscal years. The model's Sales table has the following three partitions:

PARTITION	DATA FROM
Sales2011	Current fiscal year
Sales2010-2001	Fiscal years 2001, 2002, 2003, 2004, 2005, 2006. 2007, 2008, 2009, 2010
SalesOld	All fiscal years prior to the last ten years.

As new sales data is added for the current 2011 fiscal year; that data must be processed daily to accurately be reflected in current fiscal year sales data analysis, thus the Sales2011 partition is processed nightly.

There is no need to process data in the Sales2010-2001 partition nightly; however, because sales data for the previous ten fiscal years can still occasionally change because of product returns and other adjustments, it must still be processed regularly, thus data in the Sales2010-2001 partition is processed monthly. Data in the SalesOld partition never changes therefore only processed annually.

When entering the 2012 fiscal year, a new Sales2012 partition is added to the mode's Sales table. The Sales2011 partition can then be merged with the Sales2010-2001 partition and renamed to Sales2011-2002. Data from the 2001 fiscal year is eliminated from the new Sales2011-2002 partition and moved into the SalesOld partition. All partitions are then processed to reflect changes.

How you implement a partition strategy for your organization's tabular models will largely be dependent on your particular model data processing needs and available resources.

Permissions

In order to create, manage, and process partitions in SQL Server Management Studio, you must have the appropriate Analysis Services permissions defined in a security role. Each security role has one of the following permissions:

PERMISSION	ACTIONS
Administrator	Read, process, create, copy, merge, delete
Process	Read, process
Read Only	Read

To learn more about creating roles during model authoring by using SQL Server Data Tools (SSDT), see Roles. To learn more about managing role members for deployed tabular model roles by using SQL Server Management Studio, see Tabular Model Roles.

Parallel Processing

Analysis Services includes parallel processing for tables with two or more partitions, increasing processing performance. There are no configuration settings for parallel processing (see notes). Parallel processing occurs by default when you Process Table or you select multiple partitions for the same table and Process. You can still choose to process a tables partitions independently.

> **NOTE**
> To specify whether refresh operations run sequentially or in parallel, you can use the **maxParallism** property option with the Sequence command (TMSL).

> **NOTE**
> If re-encoding is detected, parallel processing can cause increased use of system resources. This is because multiple partition operations need to be interrupted and re-started with the new encoding in-parallel.

Process Partitions

Partitions can be processed (refreshed) independent of other partitions by using the **Partitions** dialog box in Management Studio or by using a script. Processing has the following options:

MODE	DESCRIPTION
Process Default	Detects the process state of a partition object, and performs processing necessary to deliver unprocessed or partially processed partition objects to a fully processed state. Data for empty tables and partitions is loaded; hierarchies, calculated columns, and relationships are built or rebuilt.

MODE	DESCRIPTION
Process Full	Processes a partition object and all the objects that it contains. When Process Full is run for an object that has already been processed, Analysis Services drops all data in the object, and then processes the object. This kind of processing is required when a structural change has been made to an object.
Process Data	Load data into a partition or a table without rebuilding hierarchies or relationships or recalculating calculated columns and measures.
Process Clear	Removes all data from a partition.
Process Add	Incrementally update partition with new data.

Related Tasks

TASK	DESCRIPTION
Create and Manage Tabular Model Partitions	Describes how to create and manage partitions in a deployed tabular model by using SQL Server Management Studio.
Process Tabular Model Partitions	Describes how to process partitions in a deployed tabular model by using SQL Server Management Studio.

Create and manage tabular model partitions

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Partitions divide a table into logical parts. Each partition can then be processed (Refreshed) independent of other partitions. Partitions defined for a model during model authoring are duplicated in a deployed model. Once deployed, you can manage those partitions by using the **Partitions** dialog box in SQL Server Management Studio or by using a script. Tasks provided in this topic describe how to create and manage partitions for a deployed model.

> **NOTE**
> Partitions in tabular models created at the 1400 compatibility level are defined using an M query statement. To learn more, see M Reference.

Tasks

To create and manage partitions for a deployed tabular model database, you will use the **Partitions** dialog box in SQL Server Management Studio. To view the **Partitions** dialog box, in SQL Server Management Studio, right-click on a table, and then click **Partitions**.

To create a new partition

1. In the **Partitions** dialog box, click the **New** button.
2. In **Partition Name**, type a name for the partition. By default, the name of the default partition will be incrementally numbered for each new partition.
3. In **Query Statement**, type or paste a SQL or M query statement that defines the columns and any clauses you want to include in the partition into the query window.
4. To validate the statement, click **Check Syntax**.

To copy a partition

1. In the **Partitions** dialog box, in the **Partitions** list, select the partition you want to copy, and then click the **Copy** button.
2. In **Partition Name**, type a new name for the partition.
3. In **Query Statement**, edit the query statement.

To merge two or more partitions

- In the **Partitions** dialog box, in the **Partitions** list, use Ctrl+click to select the partitions you want to merge, and then click the **Merge** button.

> **IMPORTANT**
> Merging partitions does not update the partition metadata. You must edit the SQL or M statement for the resulting partition to make sure processing operations process all data in the merged partition.

To delete a partition

- In the **Partitions** dialog box, in the **Partitions** list, select the partition you want to delete, and then click the **Delete** button.

See also

Tabular model partitions
Process tabular model partitions

Process Tabular Model Partitions

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Partitions divide a table into logical parts. Each partition can then be processed (Refreshed) independent of other partitions. The tasks in this topic describe how to process partitions in a model database by using the **Process Partitions** dialog box in SQL Server Management Studio.

To process a partition

1. In SQL Server Management Studio, right-click on the table that has the partitions you want to process, and then click **Partitions**.
2. In the **Partitions** dialog box, in **Partitions**, click on the Process button.
3. In the **Process Partition(s)** dialog box, in the **Mode** listbox, select one of the following process modes:

MODE	DESCRIPTION
Process Default	Detects the process state of a partition object, and performs processing necessary to deliver unprocessed or partially processed partition objects to a fully processed state. Data for empty tables and partitions is loaded; hierarchies, calculated columns, and relationships are built or rebuilt.
Process Full	Processes a partition object and all the objects that it contains. When Process Full is run for an object that has already been processed, Analysis Services drops all data in the object, and then processes the object. This kind of processing is required when a structural change has been made to an object.
Process Data	Load data into a partition or a table without rebuilding hierarchies or relationships or recalculating calculated columns and measures.
Process Clear	Removes all data from a partition.
Process Add	Incrementally update partition with new data.

4. In the **Process** checkbox column, select the partitions you want to process with the selected mode, and then click **Ok**.

See Also

Tabular Model Partitions
Create and Manage Tabular Model Partitions

Connect to a tabular model database

12/7/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you build a tabular model and deploy it to an Analysis Services tabular mode server, you need to set permissions that make it available to client applications. This article explains how to permissions and how to connect to a database from client applications.

> **NOTE**
> By default, remote connections to Analysis Services are not available until you configure the firewall. Be sure that you have opened the appropriate port if you are configuring a named or default instance for client connections. For more information, see Configure the Windows Firewall to Allow Analysis Services Access.

User permissions on the database

Users who connect to tabular databases must have membership in a database role that specifies Read access.

Roles, and sometimes role membership, are defined when a model is authored in SQL Server Data Tools (SSDT), or for deployed models, by using SQL Server Management Studio. For more information about creating roles by using Role Manager in SQL Server Data Tools, see Create and Manage Roles. For more information about creating and managing roles for a deployed model, see Tabular Model Roles.

Caution

Re-deploying a tabular model project with roles defined by using Role Manager in SQL Server Data Tools will overwrite roles defined in a deployed tabular model.

Administrative permissions on the server

For organizations that use SharePoint for hosting Excel workbooks or Reporting Services reports, additional configuration is required to make tabular model data available to SharePoint users. If you are not using SharePoint, skip this section.

Viewing Excel workbooks or Power View reports that contain tabular data requires that the account used to run Excel Services or Reporting Services has administrator permissions on the Analysis Services instance. Administrative permissions are required so that these services are trusted by the Analysis Services instance.

Grant administrative access on the server

1. In Central Administration, open the Configure service accounts page.

2. Select the service application pool used by Excel Services. It might **Service Application Pool - SharePoint Web Services System** or a custom application pool. The managed account used by Excel Services will appear on the page.

   For SharePoint farms that include Reporting Services in SharePoint mode, get the account information for the Reporting Services service application as well.

   In the following steps, you will add these accounts to the Server role on the Analysis Services instance.

3. In SQL Server Management Studio, connect to the instance of Analysis Services, right-click the server instance, and select **Properties**. In Object Explorer, right-click **Roles** and select **New Role**.

4. In the Analysis Services Properties page, click **Security**.

5. Click **Add**, and then enter the account used by Excel Services, followed by the account used by Reporting Services.

Connecting from Excel or SharePoint

Client libraries that provide access to Analysis Services databases can be used to connect to model databases that run on a tabular mode server. Libraries include the Analysis Services OLE DB provider, ADOMD.NET, and AMO.

Excel uses the OLE DB provider. If you have either MSOLAP.4 from SQL Server 2008 R2 (file name msolap100.dll, version 10.50.1600.1), or MSOLAP.5 (filename msolap110.dll) that is installed with the SQL Server 2012 (11.x) version of Power Pivot for Excel, you have a version that will connect to tabular databases.

Choose from the following approaches to connect to model databases from Excel:

- Create a data connection from within Excel, using the instructions provided in the next section.
- Create a BI semantic model connection (.bism) file in SharePoint that provides redirection to a database running on an Analysis Services tabular mode server. A BI semantic model connection file provides a right-click command that launches Excel using the model database that you specified in the connection. It will also launch Power View if Reporting Services is installed. For more information about creating and using BI semantic model connection files, see Create a BI Semantic Model Connection to a Tabular Model Database.
- Create a Reporting Services shared data source that references a tabular database as the data source. You can create the shared data source in SharePoint and use it to launch Power View.

Connect from Excel

1. In Excel, on the **Data** tab, in **Get External Data**, click **From Other Sources**.

2. Select **From Analysis Services**.

3. In **Server Name**, specify the Analysis Services instance that hosts the database. The server name is often the name of the computer that runs the server software. If the server was installed as a named instance, you must specify the name in this format: <servername>\<instancename>.

   The server instance must be configured for standalone tabular deployment and the server instance must have an inbound rule that allows access to it. For more information, see Determine the Server Mode of an Analysis Services Instance and Configure the Windows Firewall to Allow Analysis Services Access.

4. For log on credentials, choose **Use Windows Authentication** if you have read permissions to the database. Otherwise, choose **Use the Following User Name and Password**, and enter the username and password of a Windows account that has database permissions. Click **Next**.

5. Select the database. A valid selection will show a single **Model** cube for the database. Click **Next** and then click **Finish**.

After the connection is established, you can use the data to create a PivotTable or PivotChart. For more information, see Analyze in Excel.

Connect from SharePoint

If you are using Power Pivot for SharePoint, you can create a BI semantic model connection file in SharePoint that provides redirection to a database running on an Analysis Services tabular mode server. A BI semantic model connection provides an HTTP endpoint to a database. It also simplifies tabular model access for knowledge workers who routinely use documents on a SharePoint site. Knowledge workers only need to know the location of the BI semantic model connection file or its URL to access tabular model databases. Details about server location or database name are encapsulated in the BI semantic model connection. For more information about creating and

using BI semantic model connection files, see Power Pivot BI Semantic Model Connection (.bism) and Create a BI Semantic Model Connection to a Tabular Model Database.

Troubleshooting connection problems

This section provides cause and resolution steps for problems that occur while connecting to a tabular model database.

The Data Connection Wizard cannot obtain a list of databases from the specified data source.

When importing data, this Microsoft Excel error occurs when you try to use the Wizard to Connect to a tabular model database on a remote Analysis Services server, and you do not have sufficient permissions. To resolve this error, you must have user access rights on the database. Refer to the instructions provided earlier in this topic for granting user access to data.

An error occurred during an attempt to establish a connection to the external data source. The following connections failed to refresh: <model name> Sandbox

On SharePoint, this Microsoft Excel error occurs when you attempt data interaction, such as filtering data, in a PivotTable that uses model data. The error occurs because you do not have sufficient permissions on the remote Analysis Services server. To resolve this error, you must have user access rights on the database. Refer to the instructions provided earlier in this topic for granting user access to data.

An error occurred while attempting to perform this operation. Reload the workbook, and then try to perform this operation again.

On SharePoint, this Microsoft Excel error occurs when you attempt data interaction, such as filtering data, in a PivotTable that uses model data. The error occurs because Excel Services is not trusted by the Analysis Services instance on which the model data is deployed. To resolve this error, grant Excel Services administrative permission on the Analysis Services instance. Refer to the instructions provided earlier in this topic for granting administrator permissions. If the error persists, recycle the Excel Services application pool.

An error occurred during an attempt to establish a connection to the external data source used in the workbook

On SharePoint, this Microsoft Excel error occurs when you attempt data interaction, such as filtering data, in a PivotTable that uses model data. The error occurs because the user does not have sufficient SharePoint permissions on the workbook. The user must have **Read** permissions or above. **View-Only** permissions are not sufficient for data access.

See also

Tabular model solution deployment

Restore from Power Pivot

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can use the Restore from Power Pivot feature in SQL Server Management Studio to create a new Tabular model database on an Analysis Services instance (running in Tabular mode), or restore to an existing database from a Power Pivot workbook (.xlsx).

> **NOTE**
> The Import from Power Pivot project template in SQL Server Data Tools provides similar functionality. For more information, see Import from Power Pivot.

When using Restore from Power Pivot, keep the following in mind:

- In order to use Restore from Power Pivot, you must be logged on as a member of the Server Administrators role on the Analysis Services instance.
- The Analysis Services instance service account must have Read permissions on the workbook file you are restoring from.
- By default, when you restore a database from Power Pivot, the Tabular model database Data Source Impersonation Info property is set to Default, which specifies the Analysis Services instance service account. It is recommended you change the impersonation credentials to a Windows user account in Database Properties. For more information, see Impersonation.
- Data in the Power Pivot data model will be copied into an existing or new Tabular model database on the Analysis Services instance. If your Power Pivot workbook contains linked tables, they will be recreated as a table without a data source, similar to a table created using Paste To New table.

To Restore from Power Pivot

1. In SSMS, in the Active Directory instance you want to restore to, right click **Databases**, and then click **Restore from Power Pivot**.
2. In the **Restore from Power Pivot** dialog box, in **Restore Source**, in **Backup file**, click **Browse**, and then select an .abf or .xslx file to restore from.
3. In **Restore Target**, in **Restore database**, type a name for a new database or for an existing database. If you do not specify a name, the name of the workbook is used.
4. In **Storage location**, click **Browse**, and then select a location to store the database.
5. In **Options**, leave **Include security information** checked. When restoring from a Power Pivot workbook, this setting does not apply.

See Also

Tabular Model Databases
Import from Power Pivot

Multidimensional models

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

An Analysis Services multidimensional solution uses cube structures for analyzing business data across multiple dimensions. Multidimensional mode is the default server mode of Analysis Services. It includes a query and calculation engine for OLAP data, with MOLAP, ROLAP, and HOLAP storage modes to balance performance with scalable data requirements. The Analysis Services OLAP engine is an industry-leading OLAP server that works well with a broad range of BI tools. Most Analysis Services deployments are installed as classic OLAP servers.

Benefits of Using Multidimensional Solutions

The primary reason for building an Analysis Services multidimensional model is to achieve fast query performance against business data. A multidimensional model is composed of cubes and dimensions that can be annotated and extended to support complex query constructions. BI developers create cubes to support fast response times, and to provide a single data source for business reporting. Given the growing importance of business intelligence across all levels of an organization, having a single source of analytical data ensures that discrepancies are kept to a minimum, if not eliminated entirely.

Another important benefit to using Analysis Services multidimensional databases is integration with commonly used BI reporting tools such as Excel, Reporting Services, and PerformancePoint, as well as custom applications and third-party solutions.

In This Section

Multidimensional Model Solutions

Multidimensional Model Databases

Processing a multidimensional model (Analysis Services)

Roles and Permissions (Analysis Services)

Power View for Multidimensional Models

Multidimensional Model Assemblies Management

Multidimensional Model Solutions (SSAS)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In This Section

The following topics provide guidance on designing Analysis Services multidimensional database objects.

Multidimensional Model Databases
Describes how to define an Analysis Services database.

Supported Data Sources (SSAS - Multidimensional)
Describes how to define Analysis Services data source objects.

Data Source Views in Multidimensional Models
Describes how to design Analysis Services data source views.

Dimensions in Multidimensional Models
Describes how to design Analysis Services dimension objects.

Cubes in Multidimensional Models
Describes how to design Analysis Services cube objects.

Schema Generation Wizard (Analysis Services)
Describes how to design multidimensional database objects without an existing relational schema.

Analysis Services Personalization Extensions
Describes how to design Personalization Extensions for SQL Server 2008 Analysis Services.

Creating Multidimensional Models Using SQL Server Data Tools (SSDT)

5/16/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Microsoft SQL Server provides two different environments for building, deploying, and managing Analysis Services solutions: SQL Server Data Tools (SSDT) and SQL Server Management Studio. Both of these environments implement a project system. For more information about Visual Studio projects, see Projects as Containers in the MSDN Library.

- SQL Server Data Tools (SSDT) is a development environment, based on Microsoft Visual Studio 2010, used for creating and modifying business intelligence solutions. With SQL Server Data Tools (SSDT), you create Analysis Services projects that contain definitions of Analysis Services objects (cubes, dimensions, and so on), which are stored in XML files that contain Analysis Services Scripting Language (ASSL) elements. These projects are contained in solutions that can also contain projects from other SQL Server components, including SQL Server Integration Services and SQL Server Reporting Services. In SQL Server Data Tools (SSDT), you can develop Analysis Services projects as part of a solution that is independent of any particular Analysis Services instance. You can deploy the objects to an instance on a test server for testing during development, and then use the same Analysis Services project to deploy your objects to instances on one or more staging or production servers. The projects and items in a solution that includes Analysis Services, Integration Services, and Reporting Services can be integrated with source code control, such as Microsoft Visual SourceSafe. For more information about creating an Analysis Services project in SQL Server Data Tools (SSDT) using Analysis Services, see Create an Analysis Services Project (SSDT). You can also use SQL Server Data Tools (SSDT) to connect directly to an existing Analysis Services instance to create and modify Analysis Services objects, without working with a project and without storing object definitions in XML files. For more information, see Multidimensional Model Databases , and Connect in Online Mode to an Analysis Services Database.
- SQL Server Management Studio is a management and administration environment, used primarily to administer instances of Analysis Services, SQL Server, Integration Services, and Reporting Services. With SQL Server Management Studio, you can manage Analysis Services objects (perform back-ups, processing, and so on), and you can also create new objects directly on an existing Analysis Services instance by using XMLA scripts. SQL Server Management Studio provides an Analysis Server Scripts project in which you can develop and save scripts written in Multidimensional Expressions (MDX), Data Mining Extensions (DMX), and XML for Analysis (XMLA). Usually, Analysis Server Scripts projects are used for performing management tasks or re-creating objects, such as databases and cubes, on Analysis Services instances. Such projects can be saved as part of a solution and integrated with source code control. For more information about creating an Analysis Server Scripts project in SQL Server Management Studio using Analysis Services, see Analysis Services Scripts Project in SQL Server Management Studio.

Introducing Solutions, Projects, and Items

Both SQL Server Data Tools (SSDT) and SQL Server Management Studio provide projects, which are organized into solutions. A solution can contain multiple projects, and a project typically contains multiple items. A new solution is automatically generated when you create a project, and you can add additional projects as needed to an existing solution. The objects that a project contains depend on the type of the project. The items in each project container are saved as files in project folders in the file system.

SQL Server Data Tools (SSDT) contains the following projects under the Business Intelligence Projects project

type.

PROJECT	DESCRIPTION
Analysis Services Project	Contains the object definitions for a single Analysis Services database. For more information about how to create an Analysis Services project, see Create an Analysis Services Project (SSDT).
Import Analysis Services 2008 Database	Provides a wizard that you can use to create a new Analysis Services project by importing object definitions from an existing Analysis Services database.
Integration Services Project	Contains the object definitions for a set of Integration Services packages. For more information, see SQL Server Integration Services.
Report Project Wizard	Provides a wizard that guides you through the process of creating a Report project using Reporting Services. For more information, see Reporting Services (SSRS).
Report Model Project	Contains the object definitions for a Reporting Services report model. For more information, see Reporting Services (SSRS).
Report Server Project	Contains the object definitions for one or more Reporting Services reports. For more information, see Reporting Services (SSRS).

SQL Server Management Studio also contains several project types that focus on various queries or scripts, as shown in the following table.

PROJECT	DESCRIPTION
Analysis Services Scripts	Contains DMX, MDX, and XMLA scripts for Analysis Services, as well as connections to Analysis Services instances against which these scripts can be executed. For more information, see Analysis Services Scripts Project in SQL Server Management Studio.
SQL Server Compact Scripts	Contains SQL scripts for SQL Server Compact, as well as connections to SQL Server Compact instances against which these scripts can be executed.
SQL Server Scripts	Contains Transact-SQL and XQuery scripts for a SQL Server Database Engine instance, as well as connections to SQL Server Database Engine instances against which these scripts can be executed. For more information, see SQL Server Database Engine.

For more information about solutions and projects, see "Managing Solutions, Projects, and Files," either in the Microsoft Visual Studio .NET documentation or in the MSDN Library.

Choosing Between SQL Server Management Studio and SQL Server Data Tools

SQL Server Management Studio is designed for administering and configuring existing objects in SQL Server Database Engine, Analysis Services, Integration Services, and Reporting Services. SQL Server Data Tools (SSDT)

is designed for developing business intelligence solutions that include functionality from Analysis Services, Integration Services, and Reporting Services.

The following are some of the differences between SQL Server Management Studio from SQL Server Data Tools (SSDT).

- SQL Server Management Studio provides an integrated environment for connecting to instances of Analysis Services, SQL Server, and Reporting Services to configure, manage, and administer objects within an instance of Analysis Services. Through the use of scripts, you can also use SQL Server Management Studio to create or modify Analysis Services objects themselves, but SQL Server Management Studio does not provide a graphical interface for object design and definition.

- SQL Server Data Tools (SSDT) provides an integrated development environment for developing business intelligence solutions. You can use SQL Server Data Tools (SSDT) in project mode, which uses XML-based definitions of Analysis Services, Integration Services, and Reporting Services objects contained in projects and solutions. Using SQL Server Data Tools (SSDT) in project mode means that changes to Analysis Services objects in SQL Server Data Tools (SSDT) are made to these XML-based object definitions and not applied directly to an object on an Analysis Services instance until the solution is deployed. You can also use SQL Server Data Tools (SSDT) in online mode, which means connecting directly to an Analysis Services instance and working with objects in an existing database.

SQL Server Data Tools (SSDT) enhances the development of business intelligence applications because you can work on Analysis Services projects in a source-controlled, multi-user environment without requiring an active connection to an Analysis Services instance. SQL Server Management Studio provides direct access to existing objects for querying and testing, and can be used to more quickly implement previously scripted Analysis Services databases. However, once a project has been deployed into the production environment, care must be taken when working with an Analysis Services database and its objects with SQL Server Management Studio and SQL Server Data Tools (SSDT). This is to avoid overwriting changes made to objects directly in an existing database, and changes made to the Analysis Services project that originally generated the deployed solution. For more information, see Working with Analysis Services Projects and Databases During the Development Phase, and Working with Analysis Services Projects and Databases in a Production Environment.

In This Section

- Create an Analysis Services Project (SSDT)
- Configure Analysis Services Project Properties (SSDT)
- Build Analysis Services Projects (SSDT)
- Deploy Analysis Services Projects (SSDT)
- Working with Analysis Services Projects and Databases During the Development Phase
- Working with Analysis Services Projects and Databases in a Production Environment

See Also

Create an Analysis Services Project (SSDT)
Analysis Services Scripts Project in SQL Server Management Studio
Multidimensional Model Databases

Create an Analysis Services Project (SSDT)

11/28/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can define an Analysis Services project in SQL Server Data Tools (SSDT) either by using the Analysis Services Project template or by using the Import Analysis Services Database Wizard to read the contents of an Analysis Services database. If no solution is currently loaded in SQL Server Data Tools (SSDT), creating a new Analysis Services project automatically creates a new solution. Otherwise, the new Analysis Services project will be added to the existing solution. Best practices for solution development call for creating separate projects for different types of application data, using a single solution if the projects are related. For example, you might have a single solution that contains separate projects for Integration Services packages, Analysis Services databases, and Reporting Services reports that are all used by the same business application.

An Analysis Services project contains objects used in a single Analysis Services database. The deployment properties of the project specify the server and database name by which the project metadata will be deployed as instantiated objects.

This topic contains the following sections:

Create a New Project Using the Analysis Services Project Template

Create a New Project Using an Existing Analysis Services Database

Add an Analysis Services Project to an Existing Solution

Build and Deploy the Solution

Analysis Services Project Folders

Analysis Services File Types

Analysis Services Item Templates

Create a New Project Using the Analysis Services Project Template

Use these instructions to create an empty project in which you define Analysis Services objects that you can then deploy as a new Analysis Services database.

1. In SQL Server Data Tools (SSDT), click **File**, point to **New**, and click **Project**. In the **New Project** dialog box, in the **Project types** pane, select **Business Intelligence Projects**.
2. In the **New Project** dialog box, in the **Visual Studio installed templates** category, select **Analysis Services Project**.
3. In the **Name** text box, type the name of the project. The name you enter will be used as the default database name.
4. In the **Location** drop-down list, type or select the folder in which to store the files for the project, or click **Browse** to select a folder.
5. To add the new project to the existing solution, in the **Solution** drop-down list, select **Add to Solution**.

   -or-

   To create a new solution, in the **Solution** drop-down list, select **Create new Solution**. To create a new

folder for the new solution, select **Create directory for solution**. In **Solution Name**, type the name of the new solution.

6. Click **OK**.

Create a New Project Using an Existing Analysis Services Database

Use the Import Analysis Services Database Wizard to create a project based on the objects in the existing Analysis Services database. When you define an Analysis Services project based on an existing Analysis Services database, the metadata for that database will open in an Analysis Services project in SQL Server Data Tools (SSDT). These objects can then be modified within the project without affecting the original objects, and then be deployed to the same Analysis Services database if the deployment properties specify that database, or to a newly created Analysis Services database for comparison testing. Until the changes are deployed, no changes made will affect the existing Analysis Services database.

You can also use the Import Analysis Services Database template to create a project from a production database to which changes have been made directly since the original Analysis Services project was deployed.

Before you process or deploy the project, you might need to change the data provider that is specified in the data sources. If the SQL Server software you are using is newer than the software used to create the database, the data provider specified in your project might not be installed on your computer. During processing, the service account will be used to retrieve the data in your Analysis Services database. If the database is on a remote server, check whether the local service has process and read permissions on that server.

1. In SQL Server Data Tools (SSDT), click **File**, point to **New**, and click **Project**. In the **New Project** dialog box, in the **Project types** pane, select **Business Intelligence Projects**.

2. In the **New Project** dialog box, in the **Visual Studio installed templates** category, select **Import Analysis Services Database**.

3. Enter property information for the project and solution, including name and location for the files. Click **OK**.

4. On the Welcome to the **Import Analysis Services Database Wizard** page, click **Next**.

5. On the **Source Database** page, specify the server and the database from which the wizard will extract the contents and create the Analysis Services project, and then click **Next**.

   Supported databases include those created in the following versions of Analysis Services: SQL Server 2005 (9.x), SQL Server 2008, SQL Server 2008 R2, and SQL Server 2012 (11.x).

   You can either type the database name or query the server to view the existing databases on the server. If the database is on a remote server or production server, you might need to request permission to read the database. Firewall configuration settings can further restrict access to a database. If you get an error while attempting to connect to the database, check permissions and firewall settings first.

6. When the wizard finishes extracting the contents of the Analysis Services database, click **Finish** on the **Completing the Wizard** page.

7. Open the Solution Explorer window to view the contents of the project.

Add an Analysis Services Project to an Existing Solution

If you already have a solution that contains all the source files of a business application, you can add a new Analysis Services project to that solution.

Adding an existing project to a solution associates, but does not copy, the project with the solution. If the Analysis Services project was created in a different solution, the project files remain with the original solution for which it was created. This means that any changes you make to the project through either solution will operate on the

same set of source files. If this behavior is not what you intend, you should copy or move the project files to the new solution folder first, and then add the project to the solution.

1. Open the solution in SQL Server Data Tools (SSDT). In Solution Explorer, right-click the solution, point to **Add**, and then click **Existing Project** to select the project you want to add.
2. Select a .dwproj file to add to the solution.

Build and Deploy the Solution

By default, SQL Server Data Tools (SSDT) deploys a project to the default instance of Analysis Services on the local computer. You can change this deployment destination by using the **Property Pages** dialog box for the Analysis Services project to change the **Server** configuration property.

> **NOTE**
> By default, SQL Server Data Tools (SSDT) processes only objects changed by the deployment script and dependent objects when deploying a solution. You can change this functionality by using the **Property Pages** dialog box for the Analysis Services project to change the Processing Option configuration property.

Build and deploy the solution to an instance of Analysis Services for testing. Building a solution validates the object definitions and dependencies in the project and generates a deployment script. Deploying a solution uses the Analysis Services deployment engine to send the deployment script to a specified instance.

After you deploy the project, review and test the deployed database. You can then modify object definitions, build, and deploy again until the project is complete.

After the project is complete, you can use the Deployment Wizard to deploy the deployment script, generated when you build the solution, to destination instances for final testing, staging, and deployment.

Analysis Services Project Folders

An Analysis Services project contains the following folders, which are used to organize items included in the project.

FOLDER	DESCRIPTION
Data Sources	Contains data sources for an Analysis Services project. You create these objects with the Data Source Wizard and edit them in Data Source Designer.
Data Source Views	Contains data source views for an Analysis Services project. You create these objects with the Data Source View Wizard and edit them in Data Source View Designer.
Cubes	Contains cubes for an Analysis Services project. You create these objects with the Cube Wizard and edit them in Cube Designer.
Dimensions	Contains dimensions for an Analysis Services project. You create these objects with the Dimension Wizard or the Cube Wizard and edit them in Dimension Designer.
Mining Structures	Contains mining structures for an Analysis Services project. You create these objects with the Mining Model Wizard and edit them in Mining Model Designer.

FOLDER	DESCRIPTION
Roles	Contains database roles for an Analysis Services project. You create and manage roles in Role Designer.
Assemblies	Contains references to COM libraries and Microsoft .NET Framework assemblies for an Analysis Services project. You create references with the **Add Reference** dialog box.
Miscellaneous	Contains any type of file except for Analysis Services file types. Use this folder to add any miscellaneous files, such as text files that contain notes on the project.

Analysis Services File Types

A SQL Server Data Tools (SSDT) solution can contain several file types, depending on what projects you included in the solution and what items you included in each project for that solution. Typically, the files for each project in a SQL Server Data Tools (SSDT) solution are stored in the solution folder, in a separate folder for each project.

> **NOTE**
> Copying a file for an object to a project folder does not add the object to the project. You must use the **Add** command from the project's context menu in SQL Server Data Tools (SSDT) to add an existing object definition to a project.

The project folder for an Analysis Services project can contain the file types listed in the following table.

FILE TYPE	DESCRIPTION
Analysis Services project definition file (.dwproj)	Contains metadata about the items, configurations, and assembly references defined and included in the Analysis Services project.
Analysis Services project user settings (.dwproj.user)	Contains configuration information for the Analysis Services project, for a specific user.
Data source file (.ds)	Contains Analysis Services Scripting Language (ASSL) elements that define metadata for a data source.
Data source view file (.dsv)	Contains ASSL elements that define metadata for a data source view.
Cube file (.cube)	Contains ASSL elements that define metadata for a cube, including measure groups, measures, and cube dimensions.
Partition file (.partitions)	Contains ASSL elements that define metadata for the partitions of a specified cube.
Dimension file (.dim)	Contains ASSL elements that define metadata for a database dimension.
Mining structure file (.dmm)	Contains ASSL elements that define metadata for a mining structure and associated mining models.

FILE TYPE	DESCRIPTION
Database file (.database)	Contains ASSL elements that define metadata for a database, including account types, translations, and database permissions.
Database role file (.role)	Contains ASSL elements that define metadata for a database role, including role members.

Analysis Services Item Templates

If you use the **Add New Item** dialog box to add new items to an Analysis Services project, you have the option of using an item template, a predefined script or statement which demonstrates how to perform a specified action.

The item templates, listed in the following table, are available in the Analysis Services Project Items category in the **Add New Item** dialog box.

CATEGORY	ITEM TEMPLATE	DESCRIPTION
Analysis Services Project Items	Cube	Starts the Cube Wizard to add a new cube to the Analysis Services project.
	Data Source	Starts the Data Source Wizard to add a new data source to the Analysis Services project.
	Data Source View	Starts the Data Source View Wizard to add a new data source view to the Analysis Services project.
	Database Role	Adds a new database role to the Analysis Services project, and then displays Role Designer for the new database role.
	Dimension	Starts the Dimension Wizard to add a new database dimension to the Analysis Services project.
	Mining Structure	Starts the Data Mining Wizard to add a new mining structure and associated mining model to the Analysis Services project.

See Also

Configure Analysis Services Project Properties (SSDT)
Build Analysis Services Projects (SSDT)
Deploy Analysis Services Projects (SSDT)

Configure Analysis Services Project Properties (SSDT)

12/20/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In SQL Server Data Tools (SSDT), an Analysis Services project is defined with certain default properties that affect building and deploying the Analysis Services project.

To change project properties, right-click the Analysis Services project object and then click **Properties**. Alternatively, you can click **Properties** on the Project menu.

Property Description

The following table describes each project property, lists its default value, and provides information about changing its value.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Build / Deployment Server Edition	The SQL Server edition used to develop the project	Specifies the edition of the server to which projects will finally be deployed. When working with multiple developers on a project, developers need to understand the server edition to know which features to incorporate into the Analysis Services project.
Build / Deployment Server Version	The version used to develop the projects	Specifies the version of the server to which projects will finally be deployed.
Build / Outputs	/bin	The relative path for the output of the project build process
Build / Remove Passwords	True	Specifies whether known passwords will be removed from connection strings that are written to the output directory during the build process. Passwords are removed to increase security. If passwords are removed, they will need to be provided when the deployed project is processed in order for Analysis Services to access the source data.
Debugging / Start Object	<Currently Active Object>	Determines that object that will be started when you start debugging.

PROPERTY	DEFAULT SETTING	DESCRIPTION
Deployment / Deployment Mode	Deploy Changes Only	By default, only changes to project objects are deployed (provided that no other changes were made to the objects directly outside of the project). You can also choose to have all project objects deployed during each deployment. For best performance, use Deploy Changes Only.
Deployment / Processing Option	Default	By default, Analysis Services will determine the type of processing required when changes to objects are deployed. This generally results in the fastest deployment time. However, you can also choose to have either full processing or no processing performed with each deployment.
Deployment / Transactional Deployment	False	By default, the deployment of changed or all objects is not transactional with the processing of those deployed objects. Deployment can succeed and persist even though processing fails. You can change this default to incorporate deployment and processing in a single transaction.
Deployment / Target Server	localhost	By default, database objects within the Analysis Services project will be deployed to the default instance of Analysis Services on the local computer on which SQL Server Data Tools (SSDT) is being used. Change this default to specify a named instance on the local computer or any instance on any remote computer on which you have permission to create Analysis Services objects.
Deployment / Database	<project name>	By default, the name of the Analysis Services database in which the Analysis Services project objects will be instantiated upon deployment is the name of the Analysis Services project at the time it was defined. Change this property to change the name of database on the Analysis Services instance specified by the Server property.

Property Configurations

Properties are defined on a per configuration basis. Project configurations enable developers to work with an Analysis Services project with different build, debugging, and deployment settings without editing the underlying XML project files directly.

A project is initially created with a single configuration, called Development. You can create additional configurations and switch between configurations using the Configuration Manager.

Until additional configurations are created, all developers use this common configuration. However, during the various phases of project development - such as during the initial development and testing of a project - different developers will may use different data sources and deploy the project to different servers for different purposes. Configurations enable you to retain these different settings in different configuration files.

See Also

Build Analysis Services Projects (SSDT)
Deploy Analysis Services Projects (SSDT)

Build Analysis Services Projects (SSDT)

10/26/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In SQL Server Data Tools (SSDT), you build an Analysis Services project much like you build any programming project in Visual Studio. When you build the project, a set of XML files are created in the output directory. These XML files use Analysis Services Scripting Language (ASSL), which is the XML dialect the client applications including SQL Server Management Studio and SQL Server Data Tools (SSDT) use to communicate with an Analysis Services instance to create or modify Analysis Services objects. These XML files are used to deploy Analysis Services object definitions in an Analysis Services project to a specified Analysis Services instance.

Building a Project

When you build an Analysis Services project, SQL Server Data Tools (SSDT) will build a complete set of XML files in the output folder containing all of the necessary ASSL commands needed to build all of the Analysis Services database objects in the project. If the project was previously built and incremental deployment specified for the active configuration, SQL Server Data Tools (SSDT) will also build an XML file containing the ASSL commands to perform an incremental update to the deployed objects. This XML file is written to the ..\obj\<active configuration> folder for the project. Incremental builds can save time when deploying and processing a very large project or database.

> **NOTE**
> You can use the Rebuild All command to ignore the incremental deployment setting.

Building an Analysis Services project validates the object definitions in the project. The validation includes any referenced assemblies. Build errors appear in the Task List window, along with the Analysis Management Objects (AMO) error text. You can click an error to open the designer that is required to fix the error.

Successful validation does not guarantee that objects can be created on the destination server during deployment or processed successfully after deployment. The following issues can prevent successful deployment or processing after deployment:

- Security checks for the server are not performed, so locks can prevent deployment.
- Physical locations are not validated on the server.
- Details of data source views are not checked against the actual data source on the destination server.

If validation is successful, SQL Server Data Tools (SSDT) generates the XML files. After the build, the output folder will contain the files described in the following table.

FILES (IN BIN FOLDER)	DESCRIPTION
Projectname.asdatabase	Contains the ASSL elements that define metadata for the objects in the Analysis Services project in a deployment script file. This file is used by the deployment engine to deploy the objects to an Analysis Services database.

FILES (IN BIN FOLDER)	DESCRIPTION
Projectname.configsettings	Contains configuration settings using during deployment that you can modify directly or in the Analysis Services Deployment Wizard (for example, the connection string for the data sources).
Projectname.deploymenttargets	Contains the destination settings used during deployment that you can modify directly or in the Analysis Services Deployment Wizard (for example, the server and database names)
Projectname.deploymentoptions	Contain various option settings used during deployment that you can modify directly or in the Analysis Services Deployment Wizard (for example, storage locations)
Assemblyname/*dllname.*dll	Separate folders for each referenced assembly; each folder contains the DLL for the assembly, any referenced assembly, and any associated .pdb files for output debug information.

FILES (IN OBJ FOLDER)	DESCRIPTION
<Configuration Name>\LastBuilt.xml	Contains the time stamp and hash code that identifies the last time the Analysis Services project was built.

These XML files do not contain <Create> and <Alter> tags, which are constructed during deployment.

Referenced assemblies (excluding standard system and Analysis Services assemblies) are also copied to the output directory. When references are to other projects in a solution, those projects are built first, using the appropriate project configuration and build dependencies established by the project references, and then are copied to the project output folder.

See Also

Analysis Services Scripting Language (ASSL for XMLA)
Deploy Analysis Services Projects (SSDT)

Deploy Analysis Services Projects (SSDT)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

During development of an Analysis Services project in SQL Server Data Tools (SSDT), you frequently deploy the project to a development server in order to create the Analysis Services database defined by the project. This is required to test the project; for example, to browse cells in the cube, browse dimension members, or verify key performance indicators (KPIs) formulas.

Deploying a Project

You can deploy a project independently, or you can deploy all of the projects within the solution. When you deploy a project, several things happen in sequence. First, the project is built. This step creates the output files that define the Analysis Services database and its constituent objects. Next, the destination server is validated. Finally, the destination database and its objects are created on the destination server. During deployment, the deployment engine totally replaces any pre-existing database with the contents of the project unless those objects were created by the project during a previous deployment.

After an initial deployment, an IncrementalSnapshot.xml file is generated in the <Project Name>\obj folder. This file is used to determine if the database or its objects on the destination server have changed outside of the project. If so, you will be prompted to overwrite all objects in the destination database. If all changes were made within the project and the project is configured for incremental deployment, only the changes will be deployed to the destination server.

The project configuration and its associated settings determine the deployment properties that will be used to deploy the project. For a shared project, each developer can use their own configuration with their own project configuration options. For example, each developer can specify a different test server to work in isolation from other developers.

See Also

Create an Analysis Services Project (SSDT)

Create and Run an MDX Script in SQL Server Data Tools

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

To create and run an MDX Script in Analysis Services, you need to be in SQL Server Data Tools (SSDT) with a cube already created and ready for editing.

To create a Multidimensional Expressions (MDX) script

1. Open the cube for which you want to create an MDX script, and under **View**, click **Calculations**.
2. If you are not in **Script View** mode, click the **Script View** icon.
3. In the script window, under the CALCULATE command, type an MDX expression. Click the **Check Syntax** icon and verify that the expression is syntactically correct.
4. To run the MDX script, deploy and process the cube with the new MDX script changes.

See Also

The Basic MDX Script (MDX)
MDX Scripting Fundamentals (Analysis Services)
MDX Scripting Statements (MDX)

View the XML for an Analysis Services Project (SSDT)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you are working with an Analysis Services database in project mode, SQL Server Data Tools (SSDT) creates an XML definition for each object within the project folder. You can view the contents of the XML file for each object within SQL Server Data Tools (SSDT). You can also edit the XML file directly; however, this is not recommended in most circumstances as you may make changes that make the XML unreadable by SQL Server Data Tools (SSDT).

> **NOTE**
>
> You cannot view the xml code for an entire project, but rather you view the code for each object because a separate file exists for each object. The only way to view the code for an entire project is to build the project and view the ASSL code in the <project name>.asdatabase file.

To view the XML code for an object

1. Open the Analysis Services project in SQL Server Data Tools (SSDT).
2. Right-click the object in Solution Explorer and then click **View Code**.

   The XML code for the object appears in SQL Server Data Tools (SSDT).

See Also

Build Analysis Services Projects (SSDT)

Connect in Online Mode to an Analysis Services Database

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can connect directly to an existing Microsoft SQL Server Analysis Services database and directly modify objects within that database. When you connect directly to an Analysis Services database, changes to objects occur immediately and no Analysis Services project is created within SQL Server Data Tools (SSDT).

To Connect Directly to an Analysis Services Database by using SQL Server Data Tools

1. Open SQL Server Data Tools (SSDT).
2. On the **File** menu, point to **Open** and then click **Analysis Services Database**.
3. Select **Connect to existing database**.
4. Specify the server name and the database name.

   You can either type the database name or query the server to view the existing databases on the server.
5. Click **OK**.

   You can now directly edit any objects within the Analysis Services database.

See Also

Working with Analysis Services Projects and Databases During the Development Phase
Creating Multidimensional Models Using SQL Server Data Tools (SSDT)

Work with Analysis Services Projects and Databases in Development

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can develop an Analysis Services database by using SQL Server Data Tools (SSDT) in either project mode or online mode.

Single Developer

When only a single developer is developing the entire Analysis Services database and all of its constituent objects, the developer can use SQL Server Data Tools (SSDT) in either project mode or online mode at any time during the lifecycle of the business intelligence solution. In the case of a single developer, the choice of modes is not particularly critical. The maintenance of an offline project file integrated with a source control system has many benefits, such as archiving and rollback. However, with a single developer you will not have the problem of communicating changes with another developer.

Multiple Developers

When multiple developers are working on a business intelligence solution, problems will arise if the developers do not work in project mode with source control under most, if not all circumstances. Source control ensures that two developers are not making changes to the same object at the same time.

For example, suppose a developer is working in project mode and making changes to selected objects. While the developer is making these changes, suppose that another developer makes a change to the deployed database in online mode. A problem will arise when the first developer attempts to deploy his or her modified Analysis Services project. Namely, SQL Server Data Tools (SSDT) will detect that objects have changed within the deployed database and prompt the developer to overwrite the entire database, overwriting the changes of the second developer. Since SQL Server Data Tools (SSDT) has no means of resolving the changes between the Analysis Services database instance and the objects in the project about to be overwritten, the only real choice the first developer has is to discard all of his or her changes and starting anew from a new project based on the current version of the Analysis Services database.

Work with Analysis Services Projects and Databases in Production

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you have developed and deployed your Analysis Services database from your Analysis Services project to an Analysis Services instance, you must decide how you wish to make changes to objects in the deployed database. Certain changes, such changes related to security roles, partitioning, and storage settings, can be made using either SQL Server Management Studio or SQL Server Data Tools (SSDT). Other changes can only be made using SQL Server Data Tools (SSDT), either in project mode or in online mode (such as adding attributes or user-defined hierarchies).

As soon as you make a change to a deployed Analysis Services database using either SQL Server Management Studio or SQL Server Data Tools (SSDT) in online mode, the Analysis Services project that was used for deployment becomes out of date. If a developer makes any changes within the Analysis Services project and attempts to deploy the modified project, the developer will be prompted to overwrite the entire database. If the developer overwrites the entire database, it must also be processed. This issue becomes compounded if the changes made directly to the deployed database by the production staff were not communicated to the development team because they will not understand why their changes no longer appear in the Analysis Services database.

There are several ways in which you can use SQL Server Analysis Services tools to avoid the problems inherent in this situation.

- Method 1: Whenever a change is made to a production version of an Analysis Services database, use SQL Server Data Tools (SSDT) to create a new Analysis Services project based on the modified version of the Analysis Services database. This new Analysis Services project can be checked into the source control system as the master copy of the project. This method will work regardless of whether the change was made to the Analysis Services database using SQL Server Management Studio or SQL Server Data Tools (SSDT) in online mode.
- Method 2: Only make changes to the production version of an Analysis Services database using SQL Server Management Studio or SQL Server Data Tools (SSDT) in project mode. With this method, you can use options available to you in the Analysis Services Deployment Wizard to preserve changes made by SQL Server Management Studio, such as security roles and storage settings. This ensures that the design-related settings are kept in the project file (storage settings and security roles can be ignored) and the online server is used for storage settings and security roles.
- Method 3: Only make changes to the production version of an Analysis Services database using SQL Server Management Studio or SQL Server Data Tools (SSDT) in online mode. Since both tools are only working with the same online server, there are no possibilities of getting different version out of sync.

Data Sources in Multidimensional Models

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

All data that you import or load into a multidimensional model originates from an external data source. Typically, source data is from a data warehouse designed for reporting purposes, but it could come from any relational database that is accessed directly or indirectly through an intermediary, such as an SSIS package.

A **data source** object in Analysis Services specifies a direct connection to an external data source. In addition to physical location, a data source object specifies the connection string, data provider, credentials, and other properties that control connection behavior.

Information provided by the data source object is used during the following operations:

- Get schema information and other metadata used to generate data source views into a model.
- Query or load data into a model during processing.
- Run queries against multidimensional or data mining models that use ROLAP storage mode.
- Read or write to remote partitions.
- Connect to linked objects, as well as synchronize from target to source.
- Perform write back operations that update fact table data stored in a relational database.

When building a multidimensional model from the bottom up, you start by creating the data source object, and then use it to generate the next object, a **data source view**. A data source view is the data abstraction layer in the model. It is typically created after the data source object, using the schema of the source database as the basis. However, you can choose other ways to build a model, including starting with cubes and dimensions, and generating the schema that best supports your design.

Regardless of how you build it, each model requires at least one data source object that specifies a connection to source data. You can create multiple data source objects in a single model to use data from different sources or vary connection properties for specific objects.

Data source objects can be managed independently of other objects in your model. After you create a data source, you can change its properties later, and then preprocess the model to ensure the data is retrieved correctly.

Related Topics and Tasks

TOPIC	DESCRIPTION
Supported Data Sources (SSAS - Multidimensional)	Describes the types of data sources that can be used in a multidimensional model.
Create a Data Source (SSAS Multidimensional)	Explains how to add a data source object to a multidimensional model.
Delete a Data Source in Solution Explorer (SSAS Multidimensional)	Use this procedure to delete a data source object from a multidimensional model.
Set Data Source Properties (SSAS Multidimensional)	Describes each property and explains how to set each one.

TOPIC	DESCRIPTION
Set Impersonation Options (SSAS - Multidimensional)	Explains how to configure options in the Impersonation Information dialog box.

See Also

Database Objects (Analysis Services - Multidimensional Data)
Logical Architecture (Analysis Services - Multidimensional Data)
Data Source Views in Multidimensional Models
Data Sources and Bindings (SSAS Multidimensional)

Supported Data Sources (SSAS - Multidimensional)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes the types of data sources that you can use in a multidimensional model.

Supported Data Sources

You can retrieve data from the data sources in the following table. When you install SQL Server Data Tools (SSDT), setup does not install the providers that are listed for each data source. Some providers might already be installed with other applications on your computer; in other cases you will need to download and install the provider.

> **NOTE**
> Third party providers, such as the Oracle OLE DB Provider, may also be used to connect to third party databases, with support provided by those third parties.

Source	Versions	File type	Providers*
Access databases	Microsoft Access 2010, 2013, 2016	.accdb or .mdb	Microsoft Jet 4.0 OLE DB provider

SQL Server relational databases*	Microsoft SQL Server 2008, 2008 R2, 2012, 2014, 2016, Azure SQL Database, Azure SQL Data Warehouse, Microsoft Analytics Platform System (APS) Note: For more information about SQL Database on Azure.com. Note: Analytics Platform System (APS) was formerly known as SQL Server Parallel Data Warehouse (PDW). Originally, connecting to PDW from Analysis Services required a special data provider. This provider was replaced in SQL Server 2012. Starting in SQL Server 2012, the SQL Server native client is used for connections to PDW/APS. For more information about APS, see the web site Microsoft Analytics Platform System.	(not applicable)	OLE DB Provider for SQL Server SQL Server Native Client OLE DB Provider SQL Server Native 11.0 Client OLE DB Provider .NET Framework Data Provider for SQL Client
Oracle relational databases	Oracle 9i, 10g, 11g, 12g	(not applicable)	Oracle OLE DB Provider .NET Framework Data Provider for Oracle Client .NET Framework Data Provider for SQL Server OraOLEDB MSDASQL
Teradata relational databases	Teradata V2R6, V12	(not applicable)	TDOLEDB OLE DB provider .Net Data Provider for Teradata
Informix relational databases	V11.10	(not applicable)	Informix OLE DB provider
IBM DB2 relational databases	8.1	(not applicable)	DB2OLEDB
Sybase Adaptive Server Enterprise (ASE) relational databases	15.0.2	(not applicable)	Sybase OLE DB provider
Other relational databases	(not applicable)	(not applicable)	An OLE DB provider

* ODBC data sources are not supported for multidimensional solutions. Although Analysis Services itself will handle the connection, the designers in SQL Server Data Tools used for building solutions cannot connect to an ODBC data source, even when using the MSDASQL driver. If your business requirements include an ODBC data source, consider building a Tabular solution instead.

** Some features require a SQL Server relational database that runs on-premises. Specifically, writeback, and ROLAP storage require that the underlying data source is a SQL Server relational database.

See Also

Data Sources Supported
Data Sources in Multidimensional Models
Data Source Views in Multidimensional Models

Create a Data Source (SSAS Multidimensional)

11/27/2018 • 12 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In an Analysis Services multidimensional model, a data source object represents a connection to the data source from which you are processing (or importing) data. A multidimensional model must contain at least one data source object, but you can add more to combine data from several data warehouses. Use the instructions in this topic to create a data source object for your model. For more information about setting properties on this object, see Set Data Source Properties (SSAS Multidimensional).

This topic includes the following sections:

Choose a Data Provider

Set Credentials and Impersonation Options

View or Edit Connection Properties

Create a Data Source Using the Data Source Wizard

Create a Data Source Using an Existing Connection

Add Multiple Data Sources to a Model

Choose a Data Provider

You can connect using a managed Microsoft .NET Framework or native OLE DB provider. The recommended data provider for SQL Server data sources is SQL Server Native Client because it typically offers better performance.

For Oracle and other third-party data sources, check whether the third-party provides a native OLE DB provider and try that first. If you encounter errors, try one of the other .NET providers or native OLE DB providers listed in Connection Manager. Be sure that any data provider you use is installed on all computers used to develop and run the Analysis Services solution.

Set Credentials and Impersonation Options

A data source connection can sometimes use Windows authentication or an authentication service provided by the database management system, such as SQL Server authentication when connecting to SQL Azure databases. The account you specify must have a login on the remote database server and read permissions on the external database.

Windows Authentication

Connections that use Windows authentication are specified on the **Impersonation Information** tab of the Data Source Designer. Use this tab to choose the impersonation option that specifies the account under which Analysis Services runs when connecting to the external data source. Not all options can be used in all scenarios. For more information about these options and when to use them, see Set Impersonation Options (SSAS - Multidimensional).

Database Authentication

As an alternative to Windows authentication, you can specify a connection that uses an authentication service provided by the database management system. In some cases, using database authentication is required. Scenarios that call for using database authentication include using SQL Server authentication to connect to a

Windows Azure SQL Database, or accessing a relational data source that runs on a different operating system or in a non-trusted domain.

For a data source that uses database authentication, the username and password of a database login is specified on the connection string. Credentials are added to the connection string when you enter a user name and password in Connection Manager when setting up the data source connection in your Analysis Services model. Remember to specify a user identity that has read permissions to the data.

When retrieving data, the client library making the connection formulates a connection request that includes the credentials in the connection string. Windows authentication credential options in the Impersonation Information tab are not used in the connection, but can be used for other operations, such as accessing resources on the local computer. For more information, see Set Impersonation Options (SSAS - Multidimensional).

After you save the data source object in your model, the connection string and password are encrypted. For security purposes, all visible traces of the password are removed from the connection string when you subsequently view it in tools, script, or code.

> **NOTE**
>
> By default, SQL Server Data Tools (SSDT) does not save passwords with the connection string. If the password is not saved, Analysis Services prompts you to enter the password when it is needed. If you choose to save the password, the password is stored in encrypted format in the data connection string. Analysis Services encrypts password information for data sources using the database encryption key of the database that contains the data source. With encrypted connection information, you must use SQL Server Configuration Manager to change the Analysis Services service account or password or the encrypted information cannot be recovered. For more information, see SQL Server Configuration Manager.

Defining Impersonation Information for Data Mining Objects

Data mining queries may be executed in the context of the Analysis Services service account, but may also be executed in the context of the user submitting the query or in the context of a specified user. The context in which a query is executed may affect query results. For data mining **OPENQUERY** type operations, you may want the data mining query to execute in the context of the current user or in the context of a specified user (regardless of the user executing the query) rather than in the context of the service account. This enables the query to be executed with limited security credentials. If you want Analysis Services to impersonate the current user or to impersonate a specified user, select either the **Use a specific user name and password** or **Use the credentials of the current user** option.

Create a Data Source Using the Data Source Wizard

1. In SQL Server Data Tools, open the Analysis Services project or connect to the Analysis Services database in which you want to define the data source.

2. In **Solution Explorer**, right-click the **Data Sources** folder, and then click **New Data Source** to start the **Data Source Wizard**.

3. On the **Select how to define the connection** page, choose **Create a data source based on an existing or new connection** and then click **New** to open **Connection Manager**.

   New connections are created in Connection Manager. In Connection Manager, you select a provider and then specify the connection string properties used by that provider to connect to the underlying data. The exact information required depends upon the provider selected, but generally such information includes a server or service instance, information for logging on to the server or service instance, a database or file name, and other provider-specific settings. For the remainder of this procedure, we'll assume a SQL Server database connection.

4. Select the Microsoft .NET Framework or native OLE DB provider to use for the connection.

The default provider for a new connection is the Native OLE DB\SQL Server Native Client provider. This provider is used to connect to a SQL Server Database Engine instance using OLE DB. For connections to a SQL Server relational database, using Native OLE DB\SQL Server Native Client 11.0 is often faster than using alternative providers.

You can choose a different provider to access other data sources. For a list of the providers and relational databases supported by Analysis Services, see Supported Data Sources (SSAS - Multidimensional).

5. Enter the information requested by the selected provider to connect to the underlying data source. If the **Native OLE DB\SQL Server Native Client** provider is selected, enter the following information:

   a. **Server Name** is the network name of the Database Engine instance. It can be specified as the IP address, the NETBIOS name of the computer, or a fully qualified domain name. If the server is installed as a named instance, you must include the instance name (for example, <computername>\<instancename>).

   b. **Log on to the Server** specifies how the connection will be authentication. **Use Windows Authentication** uses Windows authentication. **Use SQL Server Authentication** specifies a database user login for a Windows Azure SQL databases or a SQL Server instance that supports mixed mode authentication.

   > **IMPORTANT**
   >
   > Connection Manager includes a **Save my password** checkbox for connections that use SQL Server authentication. Although the checkbox is always visible, it is not always used.
   >
   > Conditions under which Analysis Services does not use this checkbox include refreshing or processing the SQL Server relational data used in active Analysis Services database. Regardless of whether you clear or select **Save my password**, Analysis Services will always encrypt and save the password. The password is encrypted and stored in both .abf and data files. This behavior exists because Analysis Services does not support session-based password storage on the server.
   >
   > This behavior only applies to databases that a) are persisted on an Analysis Services server instance, and b) use SQL Server authentication to refresh or process relational data. It is does not apply to data source connections that you set up in SQL Server Data Tools (SSDT) that are used only for the duration of a session. While there is no way to remove a password that is already stored, you can use different credentials, or Windows authentication, to overwrite the user information that is currently stored with the database.

   c. **Select or enter a database name** or **Attach a database file** are used to specify the database.

   d. In the left side of the dialog box, click **All** to view additional settings for this connection, including all default settings for this provider.

   e. Change settings as appropriate for your environment and then click **OK**.

      The new connection appears in the **Data Connection** pane of the **Select how to define the connection** page of the Data Source Wizard.

6. Click **Next**.

7. In **Impersonation Information**, specify the Windows credentials or user identity that Analysis Services will use when connecting to the external data source. If you are using database authentication, these settings are ignored for connection purposes.

   Guidelines for choosing an impersonation option vary depending on how you are using the data source. For processing tasks, the Analysis Services service must run in the security context of its service account or a specified user account when connecting to a data source.

   - **Use a specific Windows user name and password** to specify a unique set of least privilege

credentials.

- **Use the service account** to process the data using the service identity.

The account you specify must have Read permissions on the data source.

8. Click **Next**. In **Completing the Wizard**, enter a data source name or use the default name. The default name is the name of the database specified in the connection. The **Preview** pane displays the connection string for this new data source.

9. Click **Finish**. The new data source appears in the **Data Sources** folder in Solution Explorer.

Create a Data Source Using an Existing Connection

When you work in an Analysis Services project, your data source can be based on an existing data source in your solution or can be based on an Analysis Services project. The Data Source Wizard provides several options for creating the data source object, including using an existing connection in the same project.

- Creating a data source based on an existing data source in your solution lets you define a data source that is synchronized with the existing data source. When the project containing this new data source is built, the data source settings from the underlying data source are used.

- Creating a data source based on an Analysis Services project lets you reference another Analysis Services project in the solution in the current project. The new data source uses the MSOLAP provider with its **Data Source** property and **Initial Catalog** property acquired from the **TargetServer** and **TargetDatabase** properties of the selected project. This feature is useful in solutions where you are using multiple Analysis Services projects to manage remote partitions, because the source and destination Analysis Services databases require reciprocal data sources to support remote partition storage and processing.

When you reference a data source object, you can edit that object only in the referenced object or project. You cannot edit the connection information in the data source object that contains the reference. Changes to the connection information in the referenced object or project appear in the new data source when it is built. The connection string information that appears in the data source (.ds) file in the project is synchronized when you build the project or when you clear the reference in Data Source Designer.

View or Edit Connection Properties

The connection string is formulated based on the properties you select in the Data Source Designer or the New Data Source Wizard. You can view the connection string and other properties in SQL Server Data Tools.

To edit the connection string

1. In SQL Server Data Tools, double-click the data source object in Solution Explorer.

2. Click **Edit**, and then click **All** on the left navigation pane.

3. The property grid appears, showing available properties of the data provider you are using. For more information about these properties, see the product documentation of the provider. For SQL Server native client, see Using Connection String Keywords with SQL Server Native Client.

If you have multiple data source objects in the solution and you prefer to maintain the connection string in one place, you can configure the current data source to reference the other data source object.

A *data source reference* is an association to another Analysis Services project or data source in the same solution. References provide a means to synchronize data sources between objects in a solution. The connection string information is synchronized whenever you build the project. To change the connection string for a data source that references another object, you must change the connection string of the referenced object.

You can remove the reference by clearing the check box. This ends the synchronization between the objects and lets you change the connection string in the data source.

Add Multiple Data Sources to a Model

You can create more than one data source object to support connections to additional data sources. Each data source must have columns that can be used to create relationships.

> **NOTE**
>
> If multiple data sources are defined and data is queried from multiple sources in a single query, such as for a snow-flaked dimension, you must define a data source that supports remote queries using **OpenRowset**. Typically, this will be a Microsoft SQL Server data source.

Requirements for using multiple data sources include the following:

- Designate one data source as the primary data source. The primary data source is the one used to create a data source view.
- A primary data source must support the **OpenRowset** function. For more information about this function in SQL Server, see OpenRowSet.

Use the following approach to combine data from multiple data sources:

1. Create the data sources in your model.
2. Create a data source view, using a SQL Server relational database as the data source. This is your primary data source.
3. In Data Source View Designer, using the data source view you just created, right-click anywhere in the work area and select **Add/Remove Tables**.
4. Choose the second data source and then select the tables you want to add.
5. Find and select the table you added. Right-click the table and select **New Relationship**. Choose the source and destination columns that contain matching data.

See Also

Supported Data Sources (SSAS - Multidimensional)
Data Source Views in Multidimensional Models

Delete a Data Source in Solution Explorer (SSAS Multidimensional)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can delete a data source object to permanently remove it from an Analysis Services multidimensional model project.

In Analysis Services, data sources provide the basis on which data source views are constructed, and data source views are in turn used to define dimensions, cubes, and mining structures in an Analysis Services project or database. Therefore, deleting a data source may invalidate other Analysis Services objects in an Analysis Services project. You should always review the list of dependent objects that is provided before deleting the object.

> **IMPORTANT**
>
> Data sources on which other objects depend cannot be deleted from an Analysis Services database opened by SQL Server Data Tools (SSDT) in online mode. You must delete all objects in the Analysis Services database that depend on that data source before deleting the data source. For more information about online mode, see Connect in Online Mode to an Analysis Services Database.

To delete a data source

1. In SQL Server Data Tools (SSDT), open the project or connect to the database from which you want to delete a data source.
2. In **Solution Explorer**, expand the **Data Sources** folder.
3. Right-click the data source, and then click **Delete**. The **Delete Objects** dialog box appears, showing the objects that will be invalidated if you delete the data source. Review this list carefully before clicking **OK** to delete the data source.
4. Save the project.

   After deleting a data source from an Analysis Services project, you must save the modified project or you will receive an error the next time you open the project because the underlying XML file for the deleted data source will be missing when the project attempts to load the deleted data source.

See Also

Data Sources in Multidimensional Models
Supported Data Sources (SSAS - Multidimensional)

Set Data Source Properties (SSAS Multidimensional)

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Analysis Services, a data source object specifies a connection to an external data warehouse or relational database that provides data to a multidimensional model. Properties on the data source determine the connection string, a timeout interval, maximum number of connections, and the transaction isolation level.

Set data source properties in SQL Server Data Tools

1. Double-click a data source in Solution Explorer to open Data Source Designer.
2. Click the **Impersonation Information** tab in Data Source Designer. For more information about creating a data source, see Create a Data Source (SSAS Multidimensional).

Set data source properties in Management Studio

1. Expand the database folder, open the **Data Source** folder under the database name, right-click a data source in **Object Explorer** and select **Properties**.
2. Optionally, modify the name, description, or impersonation option. For more information, see Set Impersonation Options (SSAS - Multidimensional).

Data Source properties

TERM	DEFINITION
Name, ID, Description	Name, ID, and Description are used to identify and describe the data source object in the multidimensional model. Name and Description can be specified in SQL Server Data Tools or in Management Studio after you deploy or process the solution. ID is generated when the object is created. Although you can easily modify the name and description, IDs are read-only and should not be changed. A fixed object ID preserves object dependencies and references throughout the model.
Create Timestamp	This read-only property appears in Management Studio. It displays the date and time the data source was created.
Last Schema Update	This read-only property appears in Management Studio. It displays the date and time the metadata for the data source was last updated. This value is updated when you deploy the solution.

TERM	DEFINITION
Query Timeout	Specifies how long a connection request will be attempted before it is dropped. Type the query timeout in the following format: *<Hours>:<Minutes>:<Seconds>* This property can be overruled by the **DatabaseConnectionPoolTimeoutConnection** server property. If the server property is smaller, it will be used instead of **Query Timeout**. For more information about the **Query Timeout** property, see Timeout. For more information about the server property, see OLAP Properties.
Connection String	Specifies the physical location of a database that provides data to a multidimensional model, and the data provider used for the connection. This information is provided to a client library making the connection request. The provider determines which properties can be set on the connection string. The connection string is built using the information you provide in the **Connection Manager** dialog box. You can also view and edit the connection string in Management Studio in the data source property page. For a SQL Server database, a connection string that contains **user ID** indicates database authentication; a connection containing **Integrated Security=SSPI** indicates Windows authentication. You can change the server or database name if the database was moved to a new location. Be sure to verify that the credentials currently specified for the connection are mapped to a database login.
Maximum number of connections	Specifies the maximum number of connections allowed by Analysis Services to connect to the data source. If more connections are needed, Analysis Services will wait until a connection becomes available. The default is 10. Constraining the number of connections ensures that the external data source is not overloaded with Analysis Services requests.
Isolation	Specifies the locking and row versioning behavior of SQL commands issued by a connection to a relational database. Valid values are ReadCommitted or Snapshot. The default is ReadCommitted, which specifies that data must be committed before it will be read, preventing dirty reads. Snapshot specifies that reads are from a snapshot of previously committed data. For more information about isolation level in SQL Server, see SET TRANSACTION ISOLATION LEVEL (Transact-SQL).

TERM	DEFINITION
Managed Provider	Displays the name of the managed provider, such as System.Data.SqlClient or System.Data.OracleClient, if the data source uses a managed provider. If the data source does not use a managed provider, this property displays an empty string. This property is read-only in Management Studio. To change the provider used on the connection, edit the connection string.
Impersonation Info	Specifies the Windows identity that Analysis Services runs under when connecting to a data source that uses Windows authentication. Options include using a predefined set of Windows credentials, the service account, the identity of the current user, or an inherit option that can be useful if your model contains multiple data source objects. For more information, see Set Impersonation Options (SSAS - Multidimensional). In Management Studio, the valid values list includes these values: **ImpersonateAccount** (use a specific Windows user name and password to connect to the data source). **ImpersonateServiceAccount** (use the security identity of the service account to connect to the data source). This is the default value. **ImpersonateCurrentUser** (use the security identity of the current user to connect to the data source). This option is only valid for data mining queries that retrieve data from an external data warehouse or database; do not choose it for data connections used for processing, loading, or write-back in a multidimensional database. **Inherit** or **default** (use the impersonation settings of the Analysis Services database that contains this data source object). Database properties include impersonation options.

See Also

Data Sources in Multidimensional Models
Create a Data Source (SSAS Multidimensional)

Set Impersonation Options (SSAS - Multidimensional)

11/28/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When creating a **data source** object in an Analysis Services model, one of the settings that you must configure is an impersonation option. This option determines whether Analysis Services assumes the identity of a specific Windows user account when performing local operations related to the connection, such as loading an OLE DB data provider or resolving user profile information in environments that support roaming profiles.

For connections that use Windows authentication, the impersonation option also determines the user identity under which queries execute on the external data source. For example, if you set the impersonation option to **contoso\dbuser**, queries used to retrieve data during processing will execute as **contoso\dbuser** on the database server.

This topic explains how to set impersonation options in the **Impersonation Information** dialog box when configuring a data source object.

Set impersonation options in SQL Server Data Tools

1. Double-click a data source in Solution Explorer to open Data Source Designer.
2. Click the **Impersonation Information** tab in Data Source Designer.
3. Choose an option described in Impersonation options in this topic.

Set impersonation options in Management Studio

In Management Studio, open the **Impersonation Information** dialog box by clicking the ellipsis (**...**) button for the following properties of these dialog boxes:

- **Database Properties** dialog box, through the Data Source Impersonation Info property.
- **Data Source Properties** dialog box, through the Impersonation Info property.
- **Assembly Properties** dialog box, through the Impersonation Info property.

Impersonation options

All options are available in the dialog box, but not all options are appropriate for every scenario. Use the following information to determine the best option for your scenario.

Use a specific user name and password
Select this option to have the Analysis Services object use the security credentials of a Windows user account specified in this format: <*Domain name*>****<*User account name*>.

Choose this option to use a dedicated, least-privilege Windows user identity that you have created specifically for data access purposes. For example, if you routinely create a general purpose account for retrieving data used in reports, you can specify that account here.

For multidimensional databases, the specified credentials will be used for processing, ROLAP queries, out-of-line bindings, local cubes, mining models, remote partitions, linked objects, and synchronization from target to

source.

For DMX OPENQUERY statements, this option is ignored and the credentials of the current user will be used rather than the specified user account.

Use the service account
Select this option to have the Analysis Services object use the security credentials associated with the Analysis Services service that manages the object. This is the default option. In previous releases, this was the only option you could use. You might prefer this option to monitor data access at the service level rather than individual user accounts.

In SQL Server 2017, depending on the operating system you are using, the service account might be NetworkService or a built-in virtual account created for a specific Analysis Services instance. If you choose the service account for a connection that uses Windows authentication, remember to create a database login for this account and grant read permissions, as it will be used to retrieve data during processing. For more information about the service account, see Configure Windows Service Accounts and Permissions.

> **NOTE**
> When using database authentication, you should choose the **Use the service account** impersonation option if the service is running under the dedicated virtual account for Analysis Services. This account will have permissions to access local files. If the service runs as NetworkService, a better alternative is to use a least privilege Windows user account that has **Allow log on locally** permissions. Depending on the account you provide, you might also need to grant file access permissions on the Analysis Services program folder.

For multidimensional databases, the service account credentials will be used for processing, ROLAP queries, remote partitions, linked objects, and synchronization from target to source.

For DMX OPENQUERY statements, local cubes, and mining models, the credentials of the current user will be used even if you choose the service account option. The service account option is not supported for out-of-line bindings.

> **NOTE**
> Errors can occur when processing a data mining model from a cube if the service account does not have administrator permissions on the Analysis Services instance. For more information, see Mining Structure: Issue while Processing when DataSource is OLAP Cube.

Use the credentials of the current user
Select this option to have the Analysis Services object use the security credentials of the current user for out-of-line bindings, DMX OPENQUERY, local cubes, and mining models.

With the exception of local cubes and processing using out-of-line bindings, this option is not supported for multidimensional databases.

Default or **Inherit**
The dialog box uses **Default** for the impersonation options set at the database level and **Inherit** for impersonation options set at the data source level.

Data Sources - Inherit Option

At the data source level, **Inherit** specifies that Analysis Services should use the impersonation option of the parent object. In a multidimensional model, the parent object is the Analysis Services database. Choosing the **Inherit** option lets you centrally manage the impersonation settings for this and other data sources that are part of the same database. For this option to be meaningful, choose a specific Windows user name and password at the database level. Otherwise, the combination of **Inherit** on the data source and **Default** on the database are

equivalent to using service account option.

To specify a Windows user name and password at the database level, do the following:

1. Right-click the database in Management Studio and select **Properties**.
2. In **Data Source Impersonation Info**, specify a Windows user name and password.
3. Right-click each data source and view its properties to ensure that each one is using the **Inherit** option.

For more information about default settings at the database level, see Set Multidimensional Database Properties (Analysis Services).

Databases - Default option

For multidimensional databases, **Default** means use the service account, and current user for data mining operations.

See Also

Create a Data Source (SSAS Multidimensional)
Set Data Source Properties (SSAS Multidimensional)

Data Source Views in Multidimensional Models

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A data source view (DSV) is an abstraction of a relational data source that becomes the basis of the cubes and dimensions you create in a multidimensional project. The purpose of a DSV is to give you control over the data structures used in your project, and to work independently of the underlying data sources (for example, the ability to rename or concatenate columns without directly modifying the original data source).

You can build multiple data source views in an Analysis Services project or database on one or more data sources, and construct each one to satisfy the requirements for a different solution.

Related Tasks

Defining a Data Source View (Analysis Services)

Adding or Removing Tables or Views in a Data Source View (Analysis Services)

Change Properties in a Data Source View (Analysis Services)

Define Logical Relationships in a Data Source View (Analysis Services)

Define Logical Primary Keys in a Data Source View (Analysis Services)

Define Named Calculations in a Data Source View (Analysis Services)

Define Named Queries in a Data Source View (Analysis Services)

Replace a Table or a Named Query in a Data Source View (Analysis Services)

Work with Diagrams in Data Source View Designer (Analysis Services)

Explore Data in a Data Source View (Analysis Services)

Delete a Data Source View (Analysis Services)

Refresh the Schema in a Data Source View (Analysis Services)

See Also

Schema Generation Wizard (Analysis Services)
Supported Data Sources (SSAS - Multidimensional)

Data Sources and Bindings (SSAS Multidimensional)

11/28/2018 • 13 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Cubes, dimensions, and other Analysis Services objects can be bound to a data source. A data source can be one of the following objects:

- A relational data source.
- An Analysis Services pipeline that outputs a rowset (or chaptered rowsets).

The means of expressing the data source varies by the type of data source. For example, a relational data source is distinguished by the connection string. For more information about data sources, see Data Sources in Multidimensional Models.

Regardless of the data source used, the data source view (DSV) contains the metadata for the data source. Thus, the bindings for a cube or other Analysis Services objects are expressed as bindings to the DSV. These bindings can include bindings to logical objects-objects such as views, calculated columns, and relationships that do not physically exist in the data source. Analysis Services adds a calculated column that encapsulates the expression to the DSV, and then binds the corresponding OLAP measure to that column in the DSV. For more information about DSVs, see Data Source Views in Multidimensional Models.

Each Analysis Services object binds to the data source in its own way. In addition, the data bindings for these objects and the definition of the data source can be provided inline with the definition of the databound object (for example, the dimension), or out-of-line as a separate set of definitions.

Analysis Services Data Types

The data types that are used in bindings must match the data types supported by Analysis Services. The following data types are defined in Analysis Services:

ANALYSIS SERVICES DATA TYPE	DESCRIPTION
BigInt	A 64-bit signed integer. This data type maps to the Int64 data type inside Microsoft .NET Framework and the DBTYPE_I8 data type inside OLE DB.
Bool	A Boolean value. This data type maps to the Boolean data type inside the .NET Framework and the DBTYPE_BOOL data type inside OLE DB.
Currency	A currency value ranging from -263 (or -922,337,203,685,477.5808) to 263-1 (or +922,337,203,685,477.5807) with an accuracy to a ten-thousandth of a currency unit. This data type maps to the Decimal data type inside the .NET Framework and the DBTYPE_CY data type inside OLE DB.
Date	Date data, stored as a double-precision floating-point number. The whole portion is the number of days since December 30, 1899, while the fractional portion is a fraction of a day. This data type maps to the DateTime data type inside the .NET Framework and the DBTYPE_DATE data type inside OLE DB.

ANALYSIS SERVICES DATA TYPE	DESCRIPTION
Double	A double-precision floating-point number within the range of -1.79E +308 through 1.79E +308. This data type maps to the Double data type inside the .NET Framework and the DBTYPE_R8 data type inside OLE DB.
Integer	A 32-bit signed integer. This data type maps to the Int32 data type inside the .NET Framework and the DBTYPE_I4 data type inside OLE DB.
Single	A single-precision floating-point number within the range of -3.40E +38 through 3.40E +38. This data type maps to the Single data type inside .NET Framework and the DBTYPE_R4 data type inside OLE DB.
SmallInt	A 16-bit signed integer. This data type maps to the Int16 data type inside the .NET Framework and the DBTYPE_I2 data type inside OLE DB.
TinyInt	An 8-bit signed integer. This data type maps to the SByte data type inside the .NET Framework and the DBTYPE_I1 data type inside OLE DB. Note: If a data source contains fields that are of the tinyint datatype and the AutoIncrement property is set to True, then they will be converted to integers in the data source view.
UnsignedBigInt	A 64-bit unsigned integer. This data type maps to the UInt64 data type inside .NET Framework and the DBTYPE_UI8 data type inside OLE DB.
UnsignedInt	A 32-bit unsigned integer. This data type maps to the UInt32 data type inside the .NET Framework and the DBTYPE_UI4 data type inside OLE DB.
UnsignedSmallInt	A 16-bit unsigned integer. This data type maps to the UInt16 data type inside the .NET Framework and the DBTYPE_UI2 data type inside OLE DB.
WChar	A null-terminated stream of Unicode characters. This data type maps to the String data type inside the .NET Framework and the DBTYPE_WSTR data type inside OLE DB.

All data that is received from the data source is converted to the SSAS type specified in the binding (usually during processing). An error is raised if the conversion cannot be performed (for example, String to Int). SQL Server Data Tools (SSDT) usually sets the data type in the binding to the one that best matches the source type in the data source. For example, the SQL types Date, DateTime, SmallDateTime, DateTime2, DateTimeOffset are mapped to SSAS Date, and the SQL type Time is mapped to String.

Bindings for Dimensions

Each attribute of a dimension is bound to a column in a DSV. All the attributes of a dimension must come from a single data source. However, the attributes can be bound to columns in different tables. The relationships between the tables are defined in the DSV. In the case where more than one set of relationships exists to the same table, it might be necessary to introduce a named query in the DSV to act as an 'alias' table. Expressions and filters are defined in the DSV by using named calculations and named queries.

Bindings for MeasureGroups, Measures, and Partitions

Each measure group has the following default bindings:

- The measure group is bound to a table in a DSV (for example, **MeasureGroup.Source**).
- Each measure is bound to a column in that table (for example, **Measure.ValueColumn.Source**).
- Each measure group dimension has a set of *granularity attributes* that define the granularity of the measure group. Each of these attributes must be bound to the column or columns in the fact table that contain the attribute key. (For more information about granularity attributes, see MeasureGroup Granularity Attributes later in this topic.)

These default bindings can be selectively overridden per partition. Each partition can specify a different data source, table or query name, or filter expression. The most common partitioning strategy is to override the table per partition, by using the same data source. Alternatives include applying a different filter per partition or changing the data source.

The default data source must be defined in the DSV, thereby providing the schema information, including the details of relationships. Any additional tables or queries specified at the partition level do not need to be listed in the DSV, but they must have the same schema as the default table defined for the measure group, or at least they must contain all the columns used by the measures or granularity attributes. The detailed bindings per measure and granularity attribute cannot be overridden at the partition level, and they are assumed to be to the same columns as defined for the measure group. Therefore, if the partition uses a data source that does in fact have a different schema, the **TableDefinition** query defined for the partition must result in the same schema as the schema used by the measure group.

MeasureGroup Granularity Attributes

When the granularity of a measure group matches the granularity known in the database, and there is a direct relationship from the fact table to the dimension table, the granularity attribute only needs to be bound to the appropriate foreign key column or columns on the fact table. For example, consider the following fact and dimension tables:

```
Sales(RequestedDate, OrderedProductID, ReplacementProductID, Qty)
```

```
Product(ProductID, ProductName,Category)
```

``

```
Relation: Sales.OrderedProductID -> Product.ProductID
```

```
Relation: Sales.ReplacementProductID -> Product.ProductID
```

``

If you analyze by the ordered product, for the Ordered Product on Sales dimension role, the Product granularity attribute would be bound to Sales.OrderedProductID.

However, there may be times when the **GranularityAttributes** might not exist as columns on the fact table. For example, the **GranularityAttributes** might not exist as columns in the following circumstances:

- The OLAP granularity is coarser than the granularity in the source.

- An intermediate table interposes between the fact table and the dimension table.
- The dimension key is not the same as the primary key in the dimension table.

In all such cases, the DSV must be defined so that the GranularityAttributes exist on the fact table. For example, a named query or calculated column can be introduced.

For example, in the same example tables as above, if the granularity were by Category, then a view of the Sales could be introduced:

```
SalesWithCategory(RequestedDate, OrderedProductID, ReplacementProductID, Qty, OrderedProductCategory)
```

```
SELECT Sales.*, Product.Category AS OrderedProductCategory
```

```
FROM Sales INNER JOIN Product
```

```
ON Sales.OrderedProductID = Product.ProductID
```

``

In this case, the GranularityAttribute Category is bound to SalesWithCategory.OrderedProductCategory.

Migrating from Decision Support Objects

Decision Support Objects (DSO) 8.0 allows **PartitionMeasures** to be rebound. Therefore, the migration strategy in these cases is to construct the appropriate query.

Similarly, it is not possible to rebind the dimension attributes within a partition, although DSO 8.0 allows this rebinding also. The migration strategy in these cases is to define the necessary named queries in the DSV so that the same tables and columns exist in the DSV for the partition as the tables and columns that are used for the dimension. These cases may require the adoption of the simple migration, in which the From/Join/Filter clause is mapped to a single named query rather than to a structured set of related tables. As DSO 8.0 allows PartitionDimensions to be rebound even when the partition is using the same data source, migration may also require multiple DSVs for the same data source.

In DSO 8.0, the corresponding bindings can be expressed in two different ways, depending on whether optimized schemas are employed, by binding to either the primary key on the dimension table or the foreign key on the fact table. In ASSL, the two different forms are not distinguished.

The same approach to bindings applies even for a partition using a data source that does not contain the dimension tables, because the binding is made to the foreign key column in the fact table, not to the primary key column in the dimension table.

Bindings for Mining Models

A mining model is either relational or OLAP. The data bindings for a relational mining model are considerably different than the bindings for an OLAP mining model.

Bindings for a Relational Mining Model

A relational mining model relies on the relationships defined in the DSV to resolve any ambiguity regarding which columns are bound to which data sources. In a relational mining model, the data bindings follow these rules:

- Each non-nested table column is bound to a column either on the case table or a table related to the case table (following a many-to-one or one-to-one relationship). The DSV defines the relationships between the tables.
- Each nested-table column is bound to a source table. The columns owned by the nested-table column are then bound to columns on that source table or a table related to the source table. (Again, the binding follows a many-to-one or one-to-one relationship.) The mining model bindings do not provide the join path to the nested table. Instead, the relationships defined in the DSV provide this information.

Bindings for an OLAP Mining Model

OLAP mining models do not have the equivalent of a DSV. Therefore, the data bindings must provide any disambiguation between columns and data sources. For example, a mining model can be based on the Sales cube, and columns can be based on Qty, Amount, and Product Name. Alternatively, a mining model can be based on Product, and columns can be based on Product Name, Product Color, and a nested table with Sales Qty.

In an OLAP mining model, the data bindings follow these rules:

- Each non-nested table column is bound to a measure on a cube, to an attribute on a dimension of that cube (specifying the **CubeDimension** to disambiguate in the case of dimension roles), or to an attribute on a dimension.
- Each nested table column is bound to a **CubeDimension**. That is, it defines how to navigate from a dimension to a related cube or (in the less common case of nested tables) from a cube to one of its dimensions.

Out-of-Line Bindings

Out-of-Line bindings enable you to temporarily change the existing data bindings for the duration of a command. Out-of-line bindings refer to bindings that are included in a command and are not persisted. Out-of-line bindings apply only while that particular command executes. In contrast, inline bindings are contained in the ASSL object definition, and persist with the object definition within server metadata.

ASSL allows out-of-line bindings to be specified on either a **Process** command, if it is not in a batch, or on a **Batch** command. If out-of-line bindings are specified on the **Batch** command, all bindings specified in the **Batch** command create a new binding context in which all **Process** commands of the batch run. This new binding context includes objects that are indirectly processed because of the **Process** command.

When out-of-line bindings are specified on a command, they override the inline bindings contained in the persisted DDL for the specified objects. These processed objects may include the object directly named in the **Process** command, or they may include other objects whose processing is automatically initiated as a part of the processing.

Out-of-line bindings are specified by including the optional **Bindings** collection object with the processing command. The optional **Bindings** collection contains the following elements.

PROPERTY	CARDINALITY	TYPE	DESCRIPTION
Binding	0-n	**Binding**	Provides a collection of new bindings.
DataSource	0-1	**DataSource**	Replaces **DataSource** from server that would have been used.
DataSourceView	0-1	**DataSourceView**	Replaces the **DataSourceView** from the server that would have been used.

All elements that relate to out-of-line bindings are optional. For any elements not specified, ASSL uses the specification contained in the DDL of the persisted object. Specification of **DataSource** or **DataSourceView** in the **Process** command is optional. If **DataSource** or **DataSourceView** are specified, they are not instantiated and do not persist after the **Process** command has completed.

Definition of the Out-of-line Binding Type

Inside the out-of-line **Bindings** collection, ASSL allows a collection of bindings for multiple objects, each a **Binding**. Each **Binding** has an extended object reference, which is similar to the object reference, but it can refer to minor objects as well (for example, dimension attributes and measure group attributes). This object takes the flat form typical of the **Object** element in **Process** commands, except that the <*Object*> </*Object*> tags are not present.

Each object for which the binding is specified is identified by an XML element of the form <*object*>ID (for example, **DimensionID**). After you have identified the object as specifically as possible with the form <*object*>ID, then you identify the element for which the binding is being specified, which is usually **Source**. A common case to note is that in which **Source** is a property on the **DataItem**, which is the case for column bindings in an attribute. In this case, you do not specify the **DataItem** tag; instead, you simply specify the **Source** property, as if it were directly on the column to be bound.

KeyColumns are identified by their ordering inside the **KeyColumns** collection. There it is not possible to specify, for example, only the first and third key columns of an attribute, because there is no way to indicate that the second key column is to be skipped. All of the key columns must be present in the out-of-line binding for a dimension attribute.

Translations, although they have no ID, are semantically identified by their language. Therefore, the **Translations** inside a **Binding** need to include their language identifier.

One additional element allowed within a **Binding** that does not exist directly in the DDL is **ParentColumnID**, which is used for nested tables for data mining. In this case, it is necessary to identify the parent column in the nested table for which the binding is being provided.

Defining a Data Source View (Analysis Services)

11/28/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A data source view contains the logical model of the schema used by Analysis Services multidimensional database objects-namely cubes, dimensions, and mining structures. A data source view is the metadata definition, stored in an XML format, of these schema elements used by the Unified Dimensional Model (UDM) and by the mining structures. A data source view:

- Contains the metadata that represents selected objects from one or more underlying data sources, or the metadata that will be used to generate an underlying relational data store if you are following the top-down approach to schema generation.
- Can be built over one or more data sources, letting you define multidimensional and data mining objects that integrate data from multiple sources.
- Can contain relationships, primary keys, object names, calculated columns, and queries that are not present in an underlying data source and which exist separate from the underlying data sources.
- Is not visible to or available to be queried by client applications.

A DSV is a required component of a multidimensional model. Most Analysis Services developers create a DSV during the early phases of model design, generating at least one DSV based on an external relational database that provides underlying data. However, you can also create the DSV at a later phase, generating the schema and underlying database structures after the dimensions and cubes are created. This second approach is sometimes called top-down design and is frequently used for prototyping and analysis modeling. When you use this approach, you use the Schema Generation Wizard to create the underlying data source view and data source objects based on the OLAP objects defined in an Analysis Services project or database. Regardless of how and when you create a DSV, every model must have one before you can process it.

This topic includes the following sections:

Data Source View Composition

Create a DSV Using the Data Source View Wizard

Specify Name Matching Criteria for Relationships

Add a Secondary Data Source

Data Source View Composition

A data source view contains the following items:

- A name and a description.
- A definition of any subset of the schema retrieved from one or more data sources, up to and including the whole schema, including the following:
  - Table names.
  - Column names.
  - Data types.

- Nullability.
- Column lengths.
- Primary keys.
- Primary key - foreign key relationships.

- Annotations to the schema from the underlying data sources, including the following:
  - Friendly names for tables, views, and columns.
  - Named queries that return columns from one or more data sources (that show as tables in the schema).
  - Named calculations that return columns from a data source (that show as columns in tables or views).
  - Logical primary keys (needed if a primary key is not defining in the underlying table or is not included in the view or named query).
  - Logical primary key - foreign key relationships between tables, views, and named queries.

Create a DSV Using the Data Source View Wizard

To create a DSV, run the Data Source View Wizard from Solution Explorer in SQL Server Data Tools.

> **NOTE**
> Alternatively, you can construct dimensions and cubes first, and then generate a DSV for the model using the Schema Generation wizard. For more information, see Schema Generation Wizard (Analysis Services).

1. In Solution Explorer, right-click the Data Source Views folder and then click **New Data Source View**.
2. Specify a new or existing data source object that provides connection information to an external relational database (you can only select one data source in the wizard).
3. On the same page, click **Advanced** to choose specific schemas, apply a filter, or exclude table relationship information.

   Choose Schemas

   For very large data sources containing multiple schemas, you can select which schemas to use in a comma delimited list, with no spaces.

   Retrieve Relationships

   You can purposely omit table relationship information by clearing the **Retrieve relationships** checkbox in the Advanced Data Source View Options dialog box, allowing you to manually create relationships between tables in the Data Source View Designer.

4. **Filter Available Objects**

   If the Available objects list contains a very large number of objects, you can reduce the list by applying a simple filter that species a string as selection criteria. For example, if you type **dbo** and click the **Filter** button, then only those items starting with "dbo" show up in the **Available objects** list. The filter can be a partial string (for example, "sal" returns sales and salary) but it cannot include multiple strings or operators.

5. For relational data sources that do not have table relationships defined, a **Name Matching** page appears so that you can select the appropriate name matching method. For more information, see the Specify Name

Matching Criteria for Relationships section in this topic.

Add a Secondary Data Source

When defining a data source view that contains tables, views, or columns from multiple data sources, the first data source from which you add objects to the data source view is designated as the primary data source (you cannot change the primary data source after it is defined). After defining a data source view based on objects from a single data source, you can then add objects from other data sources.

If an OLAP processing or a data mining query requires data from multiple data sources in a single query, the primary data source must support remote queries using **OpenRowset**. Typically, this will be a SQL Server data source. For example, if you design an OLAP dimension that contains attributes that are bound to columns from multiple data sources, then Analysis Services will construct an **OpenRowset** query to populate this dimension during processing. However, if an OLAP object can be populated or a data mining query resolved from a single data source, then an **OpenRowset** query will not be constructed. In certain situations, you may be able to define attribute relationships between attributes to eliminate the need for an **OpenRowset** query. For more information about attribute relationships, see Attribute Relationships, Adding or Removing Tables or Views in a Data Source View (Analysis Services) and Define Attribute Relationships.

To add tables and columns from a second data source, you double-click the DSV in Solution Explorer to open it in Data Source View Designer, and then use Add/Remove Tables dialog box to include objects from other data sources that are defined in your project. For more information, see Adding or Removing Tables or Views in a Data Source View (Analysis Services).

Specify Name Matching Criteria for Relationships

When you create a DSV, relationships are created between tables based on foreign key constraints in the data source. These relationships are required for the Analysis Services engine to construct the appropriate OLAP processing and data mining queries. Sometimes, however, a data source with multiple tables has no foreign key constraints. If a data source has no foreign key constraints, the Data Source View Wizard prompts you to define how you want the wizard to attempt to match column names from different tables.

> **NOTE**
> You are prompted to provide name matching criteria only if no foreign key relationships are detected in the underlying data source. If foreign key relationships are detected, then the detected relationships are used and you must manually define any additional relationships you want to include in the DSV, including logical primary keys. For more information, see Define Logical Relationships in a Data Source View (Analysis Services) and Define Logical Primary Keys in a Data Source View (Analysis Services).

The Data Source View Wizard uses your response to match column names and create relationships between different tables in the DSV. You can specify any one of the criteria listed in the following table.

NAME MATCHING CRITERIA	DESCRIPTION
Same name as primary key	The foreign key column name in the source table is the same as the primary key column name in the destination table. For example, the foreign key column `Order.CustomerID` is the same as the primary key column `Customer.CustomerID`.
Same name as destination table name	The foreign key column name in the source table is the same as the name of the destination table. For example, the foreign key column `Order.Customer` is the same as the primary key column `Customer.CustomerID`.

NAME MATCHING CRITERIA	DESCRIPTION

Destination table name + primary key name	The foreign key column name in the source table is the same as the destination table name concatenated with the primary key column name. A space or underscore separator is permissible. For example, the following foreign-primary key pairs all match:
	
	`Order.CustomerID` and `Customer.ID`
	
	`Order.Customer ID` and `Customer.ID`
	
	`Order.Customer_ID` and `Customer.ID`

The criteria you select changes the **NameMatchingCriteria** property setting of the DSV. This setting determines how the wizard adds related tables. When you change the data source view with Data Source View Designer, this specification determines how the designer matches columns to create relationships between tables in the DSV. You can change the **NameMatchingCriteria** property setting in Data Source View Designer. For more information, see Change Properties in a Data Source View (Analysis Services).

> **NOTE**
>
> After you complete the Data Source View Wizard, you can add or remove relationships in the schema pane of Data Source View Designer. For more information, see Define Logical Relationships in a Data Source View (Analysis Services).

See Also

Adding or Removing Tables or Views in a Data Source View (Analysis Services)
Define Logical Primary Keys in a Data Source View (Analysis Services)
Define Named Calculations in a Data Source View (Analysis Services)
Define Named Queries in a Data Source View (Analysis Services)
Replace a Table or a Named Query in a Data Source View (Analysis Services)
Work with Diagrams in Data Source View Designer (Analysis Services)
Explore Data in a Data Source View (Analysis Services)
Delete a Data Source View (Analysis Services)
Refresh the Schema in a Data Source View (Analysis Services)

Adding or Removing Tables or Views in a Data Source View (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you have created a data source view (DSV) in SQL Server Data Tools (SSDT), you can modify it in Data Source View Designer by adding or removing tables and columns, including tables and columns from another data source.

To open the DSV in Data Source View Designer, you double-click the DSV in Solution Explorer. Once you open the DSV, you can use the **Add/Remove Tables** command on the button bar or menu to modify or extend the DSV. You can also work with the objects in the diagram. For example, you can select an object and then use the Delete key on your keyboard to remove an object.

> **WARNING**
>
> Use caution when removing a table. Removing a table deletes all the associated columns and relationships from the DSV and invalidates all objects bound to that table.

Selecting Tables or Views to Add or Remove

Using the **Add/Remove Tables** dialog box, you can move tables or views between the **Available objects** and **Included objects** lists. The **Available objects** list initially includes any tables or views in the primary data source that are not already in the data source view. If the primary data source supports the **OPENROWSET** function, you can also add tables or views from other data sources in the project or database.

Adding or removing a table to the DSV also adds or removes the table to the currently selected diagram in the DSV. For more information on diagrams, see Work with Diagrams in Data Source View Designer (Analysis Services).

After you move a table to the **Included objects** list in the **Add/Remove Tables** dialog box, you can add all related tables. This operation adds tables according to foreign key constraints in the data source, if such constraints exist. If foreign key constraints do not exist, you can use the **NameMatchingCriteria** property of the data source view to determine relationships by specifying a criterion for matching column names in tables to generate likely relationships. If the **NameMatchingCriteria**property is specified for the data source view, click **Add Related Tables** to add tables from the data source that have matching column names. For more information about setting the **NameMatchingCriteria** property, see Data Source Views in Multidimensional Models.

> **NOTE**
>
> Adding or removing objects in a data source view does not affect the underlying data source.

See Also

Data Source Views in Multidimensional Models
Work with Diagrams in Data Source View Designer (Analysis Services)

Explore Data in a Data Source View (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can use the **Explore Data** dialog box in Data Source View Designer in SQL Server Data Tools (SSDT) to browse data for a table, view, or named query in a data source view (DSV). When you explore the data in Data Source View Designer, you can view the contents of each column of data in a selected table, view, or named query. Viewing the actual contents assists you in determining whether all columns are needed, if named calculations are required to increase user friendliness and usability, and whether existing named calculations or named queries return the anticipated values.

To view data, you must have an active connection to the data source or sources for the selected object in the DSV. Any named calculations in a table are also sent in the query.

The data returned in a tabular format that you can sort and copy. Click on the column headers to re-sort the rows by that column. You can also highlight data in the grid and press Ctrl-C to copy the selection to the clipboard.

You can also control the sample method and the sample count. By default, the top 5000 rows are returned.

To browse data or change sampling options

1. In SQL Server Data Tools (SSDT), open the project or connect to the database that contains the data source view in which you want to browse data.

2. In Solution Explorer, expand the **Data Source Views** folder, and then double-click the data source view.

3. Right-click the table, view, or named query that contains the data you want to view, and then click **Explore Data**.

   The data source underlying the table, view, or named query in the data source view are queries, and the results appear in the **Explore <object name> Table** tab.

4. On the **Explore <object name> Table** toolbar, click the **Sampling options** icon.

   The **Data Exploration Options** dialog box opens. In this dialog box you can specify the sampling method (fewer or more records than the default sampling size of 5000 rows), or sample count.

5. Click **OK** or **Cancel** as appropriate.

6. To resample the data, click **Resample Data** on the **Explore <object name> Table** toolbar.

See Also

Data Source Views in Multidimensional Models

Work with Diagrams in Data Source View Designer (Analysis Services)

11/28/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A data source view (DSV) diagram is a visual representation of the objects in a DSV. You can work with the diagram interactively to add, hide, delete or modify specific objects. You can also create multiple diagrams on the same DSV to focus attention on a subset of the objects.

To change the area of the diagram that appears in the diagram pane, click the four-headed arrow in the lower right corner of the pane, and then drag the selection box over the thumbnail diagram until you select the area that is to appear in the diagram pane.

This topic includes the following sections:

Add a Diagram

Edit or Delete a Diagram

Find Tables in a Diagram

Arrange Objects in a Diagram

Preserve object arrangement

Add a Diagram

DSV diagrams are created automatically when you create the DSV. After the DSV exists, you can create additional diagrams, remove them, or hide specific objects to create a more manageable representation of the DSV.

To create a new diagram, right-click anywhere in the **Diagram Organizer** pane, click **New Diagram**.

When you initially define a data source view (DSV) in an Analysis Services project, all tables and views added to the data source view are added to the <All Tables> diagram. This diagram appears in the Diagram Organizer pane in Data Source View Designer, the tables in this diagram (and their columns and relationships) are listed in the Tables pane, and the tables in this diagram (and their columns and relationships) are displayed graphically in the schema pane. However, as you add tables, views and named queries to the <All Tables> diagram, the sheer number of objects in this diagram makes it difficult to visualize relationships-particularly as multiple fact tables are added to the diagram, and dimension tables relate to multiple fact tables.

To reduce the visual clutter when you only want to view a subset of the tables in the data source view, you can define sub-diagrams (simply called diagrams) consisting of selected subsets of the tables, views, and named queries in the data source view. You can use diagrams to group items in the data source view according to business or solution needs.

You can group related tables and named queries in separate diagrams for business purposes, and to make it easier to understand a data source view that contains many tables, views, and named queries. The same table or named query can be included in multiple diagrams except for the <All Tables> diagram. In the <All Tables> diagram, all objects that are contained in the data source view are shown exactly once.

Edit or Delete a Diagram

When working with a diagram, pay close attention to the commands used for adding and removing objects. For example, deleting an object from a diagram will delete it from the DSV. If you only want to delete it from the diagram, use **Hide Table** instead.

Although you can hide objects individually, bringing them back using the Show Related Tables command returns all related objects to the diagram. To control which objects are returned to the workspace, drag them from the Tables pane instead.

Find Tables in a Diagram

If your schema is large, scrolling to a particular table in the **Diagram** pane may be difficult. However, the following tools make it easy to find a table in a diagram.

- Scroll the list of tables in the **Tables** pane.

  To include a table in the currently displayed diagram, drag the table from the **Tables** pane to the diagram pane.

  To center the display on a table that is already included in the diagram, select the table in the **Tables** pane.

- Table locator in **Diagram** pane-The table locator is a 4-way arrow icon located at the intersection of the vertical and horizontal scroll bars in the lower right corner of the **Diagram** pane. It opens a thumbnail representation of the current diagram in the Diagram pane. You can use this thumbnail to change the view in the Diagram pane to any location on the diagram.

- Use the **Find Table** dialog box- Right-click on open area in the Diagram pane and click **Find Table**. Or, click the **Find Table** command on the toolbar or the **Data Source View** menu.

  You can type strings and wildcard characters in the Filter box to view subsets of the tables in the diagram.

Arrange Objects in a Diagram

Although Data Source View Designer can define multiple diagrams to make a DSV more understandable, diagrams that contain dozens of tables can be hard to read and manually rearranging table layouts is a tedious process. The Data Source View Designer can automatically rearrange tables in the current diagram using either a rectangular or diagonal layout based on the relationships between tables in the current diagram.

- In a rectangular layout, the relationship lines are drawn between tables instead of between columns. Relationship lines are drawn horizontally and vertically between tables.

- In a diagonal layout, relationship lines are drawn as directly as possible between related columns in tables. A relationship to multiple columns attaches to the first related column in the table. If the columns in a table are not visible, the lines are drawn to the top of the table.

Preserve object arrangement

After manually arranging tables the way you want, adding more tables to the diagram can result in a diagram refresh that removes any recent modifications you made to the object layout.

This behavior is more likely to occur when you add a table, causing the Diagram Organizer to move other tables in order to accommodate the new one. It then redraws the diagram to ensure all tables and connection lines represented correctly. At this point, any manual adjustments to the placement of specific objects might be lost.

To avoid this problem, add all the tables first, before making any final adjustments. Objects should now retain their position in the diagram when you open it again later.

See Also

Data Source Views in Multidimensional Models
Data Source View Designer (Analysis Services - Multidimensional Data)

Refresh the Schema in a Data Source View (Analysis Services)

11/28/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After defining a data source view (DSV) in an Analysis Services project or database, the schema in an underlying data source may change. These changes are not automatically detected or updated in a development project. Moreover, if you deployed the project to a server, you will now encounter processing errors if Analysis Services can no longer connect to the external data source.

To update the DSV so that it matches the external data source, you can refresh the DSV in Business Intelligence Development Studio (BIDS). Refreshing the DSV detects changes to the external data sources upon which the DSV is based, and builds a change list that enumerates the additions or deletions in the external data source. You can then apply the set of changes to the DSV that will realign it to the underlying data source. Note that additional work is often required to further update the cubes and dimensions in the project that use the DSV.

This topic includes the following sections:

Changes Supported in Refresh

Refresh a DSV in SQL Server Data Tools

Changes Supported in Refresh

DSV Refresh can include any of the following the actions:

- Deletion of tables, columns, and relationships
- Addition of columns and relationships, as applied to tables that are already included in the DSV
- Addition of new unique constraints. If a logical primary key exists for a table in the DSV and a physical key is added to the table in the data source, the logical key is removed and replaced by the physical key.

Refresh never adds new tables to a DSV. If you want to add a new table, you must add it manually. For more information, see Adding or Removing Tables or Views in a Data Source View (Analysis Services).

Refresh a DSV in SQL Server Data Tools

To refresh a DSV, double-click the DSV from Solution Explorer in SQL Server Data Tools (SSDT). This launches the DSV designer. Then click the Refresh Data Source View button in the designer or choose **Refresh** from the Data Source View menu.

During refresh, Analysis Services queries all underlying relational data sources to determine whether there have been changes in tables/views which are included in the DSV. If connections can be established to all underlying data sources and there have been any changes, you will see them in the **Refresh Data Source View** dialog box.

The dialog box lists tables, columns, constraints, and relationships that will be deleted or added in the DSV. The report also lists any named query or calculation that cannot be successfully prepared. The affected objects are listed in a tree view with columns and relationships nested under tables and the type of change (deletion or addition) indicated for each object. The standard data source view object icons indicate the type of object affected.

Refresh is based completely on the names of the underlying objects. Therefore, if an underlying object is renamed in the data source, Data Source View Designer treats the renamed object as two separate operations-a deletion and an addition. In this case, you may have to manually add the renamed object back to the data source view. You may also have to re-create relationships or logical primary keys.

> **IMPORTANT**
>
> If you are aware that a table has been renamed in a data source, you may want to use the **Replace Table** command to replace the table with the renamed table before you refresh the data source view. For more information, see Replace a Table or a Named Query in a Data Source View (Analysis Services).

After you examine the report, you can either accept the changes or cancel the update to reject any changes. All changes must be accepted or rejected together. You cannot choose individual items in the list. You can also save a report of the changes.

See Also

Data Source Views in Multidimensional Models

Replace a Table or a Named Query in a Data Source View (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Data Source View Designer, you can replace a table, view, or named query in a data source view (DSV) with a different table or view from the same or a different data source, or with a named query defined in the DSV. When you replace a table, all other objects in an Analysis Services database or project that have references to the table continue to reference the table because the object ID for the table in the DSV does not change. Any relationships that are still relevant (based on name and column-type matching) are retained. In contrast, if you delete and then add a table, references and relationships are lost and must be recreated.

To replace a table with another table, you must have an active connection to the source data in Data Source View Designer in project mode.

You most frequently replace a table in the data source view with another table in the data source. However, you can also replace a named query with a table. For example, you previously replaced a table with a named query, and now want to revert to a table.

> **IMPORTANT**
>
> If you rename a table in a data source, follow the steps for replacing a table and specify the renamed table as the source of the corresponding table in the DSV before you refresh a DSV. Completing the replacement and renaming process preserves the table, the table's references, and the table's relationships in the DSV. Otherwise, when you refresh the DSV, a renamed table in data source is interpreted as being deleted. For more information, see Refresh the Schema in a Data Source View (Analysis Services).

Replace a table with a named query

1. In SQL Server Data Tools (SSDT), open the project or connect to the database that contains the data source view in which you want to replace a table or a named query.
2. In Solution Explorer, expand the **Data Source Views** folder, and then double-click the data source view.
3. Open the **Create Named Query** dialog box. In either **Tables** or **Diagram** pane, right-click the table that you wish to replace, point to **Replace Table** and then click **With New Named Query**.
4. In the **Create Named Query** dialog box, define the named query and then click **OK**.
5. Save the modified data source view.

Replace a table or named query with a table

1. In SQL Server Data Tools (SSDT), open the project or connect to the database that contains the data source view in which you want to replace a table or a named query.
2. In Solution Explorer, expand the **Data Source Views** folder, and then double-click the data source view.
3. Open the **Replace Table with Other Table** dialog box. In either **Tables** or **Diagram** pane, right-click the table or named query that you wish to replace, point to **Replace Table** and then click **With Other Table**.

4. In the **Replace Table with Other Table** dialog box:
    a. In the **DataSource** drop-down list box, select the desired data source
    b. Select the table with which you wish to replace the table or named query
5. Click **OK**.
6. Save the modified data source view.

See Also

Data Source Views in Multidimensional Models

Define Logical Relationships in a Data Source View (Analysis Services)

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Data Source View Wizard and Data Source View Designer automatically define relationships between tables added to a data source view (DSV) based on underlying database relationships or name matching criteria you specify.

In cases where you are working with data from multiple data sources, you may need to manually define logical relationships in the DSV to supplement those relationships that are defined automatically. Relationships are required in Analysis Services to identify fact and dimension tables, to construct queries for retrieving data and metadata from underlying data sources, and to take advantage of advanced business intelligence features.

You can define the following types of relationships in Data Source View Designer:

- A relationship from one table to another table in the same data source.
- A relationship from one table to itself, as in a parent-child relationship.
- A relationship from one table in a data source to another table in a different data source.

> **NOTE**
> The relationships defined in a DSV are logical and may not reflect the actual relationships defined in the underlying data source. You can create relationships in Data Source View Designer that do not exist in the underlying data source, and remove relationships created by Data Source View Designer from existing foreign key relationships in the underlying data source.

Relationships are directed. For every value in the source column, there is a corresponding value in the destination column. In a data source view diagram, such as the diagrams displayed in the **Diagram** pane, an arrow on the line between two tables indicates the direction of the relationship.

This topic includes the following sections:

To add a relationship between tables, named queries, or views

To view or modify a relationship in the Diagram pane

To view or modify a relationship in the Tables pane

To add a relationship between tables, named queries, or views

1. In SQL Server Data Tools (SSDT), open the project or connect to the database that contains the data source view in which you wish to add a logical relationship.
2. In Solution Explorer, expand the **Data Source Views** folder, then double-click the data source view to open it in **Data Source View Designer**.
3. Right-click the table, named query or view to which you want to add a relationship in either the **Tables** pane and then click **New Relationship**.

> **NOTE**
> To locate a table, view or named query, you can use the **Find Table** option by either clicking the **Data Source View** menu or right-clicking in an open area of the **Tables** or **Diagram** panes.

4. In the **Specify Relationship** dialog box, do the following:

   a. Select the appropriate table, named query, or view in the **Source (foreign key) table** list.

   b. Select the appropriate table, named query, or view in the **Destination (primary key) table** lists.

   c. Select columns from the **Source Columns** and **Destination Columns** lists to create a relationship between the two tables.

      If SQL Server Data Tools (SSDT) detects, by sampling the data in the underlying table, view, or named query, that you have defined the relationship in the wrong direction (from the primary key to the foreign key rather than from the foreign key to the primary key), you will be prompted to reverse the order. To quickly reverse the order, click **Reverse**.

      If SQL Server Data Tools (SSDT) detects that a relationship already exists for the columns you have selected, you will be prompted. You cannot define duplicate relationships.

   d. Optionally, in the **Description** box, type a description for the relationship.

To view or modify a relationship in the Diagram pane

- In the **Diagram** pane in **Data Source View Designer**, right-click the relationship that you want to view and click **Edit Relationship** (or simply double-click the relationship arrow). Use the **Edit Relationship** dialog box to modify the relationship.

To view or modify a relationship in the Tables pane

1. In the **Tables** pane in **Data Source View Designer**, locate and then expand the table, view or named query containing the relationship that you wish to view or modify.

2. Expand the **Relationships** folder. The relationships between the selected table, view or named query and other tables, views and named queries appear with the relationship column listed.

3. Right-click the relationship you want to modify, and then click **Edit Relationship**.

See Also

Data Source Views in Multidimensional Models

Define Logical Primary Keys in a Data Source View (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Data Source View Wizard and Data Source View Designer automatically define a primary key for a table that is added to a data source view based on underlying database table.

Occasionally, you may need to manually define a primary key in the data source view. For example, for performance or design reasons, tables in a data source may not have explicitly defined primary key columns. Named queries and views may also omit the primary key column for a table. If a table, view, or named query does not have a physical primary key defined, you can manually define a logical primary key on the table, view or named query in Data Source View Designer.

Set a Logical Primary Key

Primary keys are required in Analysis Services to uniquely identify records in a table, identify key columns in dimension tables and to support relationships between tables, views and named queries. These relationships are used to construct queries for retrieving data and metadata from underlying data sources, and to take advantage of advanced business intelligence features.

Any column can be used for the logical primary key, including a named calculation. When you create a logical primary key, a unique constraint is created in the data source view and marked as a primary key constraint. Any other existing logical primary key specified in the selected table is deleted.

1. In SQL Server Data Tools (SSDT), open the project or connect to the database that contains the data source view in which you wish to set a logical primary key.

2. In Solution Explorer, expand the **Data Source Views** folder, then double-click the data source view.

   To locate a table or view, you can use the **Find Table** option by either clicking the **Data Source View** menu or right-clicking in an open area of the **Tables** or **Diagram** panes.

3. In either the **Tables** or the **Diagram** pane, right-click the column or columns that you want to use to define a logical primary key, and then click **Set Logical Primary Key**.

   The option to set a logical primary key is available only for tables that do not have a primary key.

   Notice that after you set the key, a key icon now identifies the primary key columns.

See Also

Data Source Views in Multidimensional Models
Define Named Calculations in a Data Source View (Analysis Services)

Define Named Calculations in a Data Source View (Analysis Services)

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A named calculation is a SQL expression represented as a calculated column. This expression appears and behaves as a column in the table. A named calculation lets you extend the relational schema of existing tables or views in a data source view without modifying the tables or views in the underlying data source. Consider the following examples:

- Create a single named calculation that is derived from multiple columns in a fact table (for example, creating Tax Amount by multiplying a tax rate by a sales price).
- Construct a user friendly name for a dimension member.
- As a query performance enhancement, create a named calculation in the DSV instead of creating a calculated member in a cube. Named calculations are calculated during processing whereas calculated members are calculated at query time.

Creating Named Calculations

> **NOTE**
> You cannot add a named calculation to a named query, nor can you base a named query on a table that contains a named calculation.

When you create a named calculation, you specify a name, the SQL expression, and, optionally, a description of the calculation. The SQL expression can refer to other tables in the data source view. After the named calculation is defined, the expression in a named calculation is sent to the provider for the data source and validated as the following SQL statement in which `<Expression>` contains the expression that defines the named calculation.

```
SELECT
   <Table Name in Data Source>.*,
   <Expression> AS <Column Name>
FROM
   <Table Name in Data Source> AS <Table Name in Data Source View>
```

The data type of the column is determined by the data type of the scalar value returned by the expression. If the provider does not find any errors in the expression, the column is added to the table.

Columns referenced in the expression should not be qualified or should be qualified by the table name only. For example, to refer to the SaleAmount column in a table, `SaleAmount` or `Sales.SaleAmount` is valid, but `dbo.Sales.SaleAmount` generates an error.

The expression is not automatically enclosed between parentheses. Therefore, if an expression, such as a SELECT statement, requires parentheses, you must type the parentheses in the **Expression** box. For example, the following expression is valid only if you type the parentheses.

```
(SELECT Description FROM Categories WHERE Categories.CategoryID = CategoryID)
```

Add or Edit a Named Calculation

1. In SQL Server Data Tools (SSDT), open the project or connect to the database that contains the data source view in which you wish to define a named calculation.

2. In Solution Explorer, expand the **Data Source Views** folder, then double-click the data source view.

3. Right-click the table in which you wish to define the named calculation in either the **Tables** or the **Diagram** pane, and then click **New Named Calculation**. Be sure to right-click on the table name, and not on an attribute. The menu should look like the following:

> **NOTE**
>
> To locate a table or view, you can use the **Find Table** option by either clicking the **Data Source View** menu or right-clicking in an open area of the **Tables** or **Diagram** panes.

4. In the **Create Named Calculations** dialog box, do the following:

   - In the **Column name** text box, type the name of the new column.
   - In the **Description** text box type a description for the new column.
   - In the **Expression** text box, type the expression that yields the content of the new column in the SQL dialect appropriate for the data provider.

5. Click **OK**.

   The named calculation column appears as the last column in the data source view table. A calculator symbol indicates that the column contains a named calculation.

Delete a Named Calculation

When you attempt to delete a named calculation, you are prompted with a list of the objects defined in the project or database that will be invalidated by the deletion. Review the list carefully before deleting the calculation.

See Also

Define Named Queries in a Data Source View (Analysis Services)

Define Named Queries in a Data Source View (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A named query is a SQL expression represented as a table. In a named query, you can specify an SQL expression to select rows and columns returned from one or more tables in one or more data sources. A named query is like any other table in a data source view (DSV) with rows and relationships, except that the named query is based on an expression.

A named query lets you extend the relational schema of existing tables in DSV without modifying the underlying data source. For example, a series of named queries can be used to split up a complex dimension table into smaller, simpler dimension tables for use in database dimensions. A named query can also be used to join multiple database tables from one or more data sources into a single data source view table.

Creating a Named Query

> **NOTE**
> You cannot add a named calculation to a named query, nor can you base a named query on a table that contains a named calculation.

When you create a named query, you specify a name, the SQL query returning the columns and data for the table, and optionally, a description of the named query. The SQL expression can refer to other tables in the data source view. After the named query is defined, the SQL query in a named query is sent to the provider for the data source and validated as a whole. If the provider does not find any errors in the SQL query, the column is added to the table.

Tables and columns referenced in the SQL query should not be qualified or should be qualified by the table name only. For example, to refer to the SaleAmount column in a table, `SaleAmount` or `Sales.SaleAmount` is valid, but `dbo.Sales.SaleAmount` generates an error.

Note When defining a named query that queries a SQL Server 2000 (8.x) or SQL Server 7.0 data source, a named query that contains a correlated subquery and a GROUP BY clause will fail. For more information, see Internal Error with SELECT Statement Containing Correlated Subquery and GROUP BY in the Microsoft Knowledge Base.

Add or Edit a Named Query

1. In SQL Server Data Tools (SSDT), open the project or connect to the database that contains the data source view in which you want to add a named query.
2. In Solution Explorer, expand the **Data Source Views** folder, then double-click the data source view.
3. In the **Tables** or **Diagram** pane, right-click an open area and then click **New Named Query**.
4. In the **Create Named Query** dialog box, do the following:
    a. In the **Name** text box, type a query name.

b. Optionally, in the **Description** text box, type a description for the query.

c. In the **Data Source** list box, select the data source against which the named query will execute.

d. Type the query in the bottom pane, or use the graphical query building tools to create a query.

> **NOTE**
>
> The query-building user interface (UI) depends on the data source. Instead of getting a graphical UI, you can get a generic UI, which is text-based. You can accomplish the same things with these different UIs, but you must do so in different ways. For more information, see Create or Edit Named Query Dialog Box (Analysis Services - Multidimensional Data).

5. Click **OK**. An icon showing two overlapping tables appears in the table header to indicate that the table has been replaced by a named query.

See Also

Data Source Views in Multidimensional Models
Define Named Calculations in a Data Source View (Analysis Services)

Delete a Data Source View (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If you are no longer using a data source view (DSV) in an OLAP project, you can delete it from the project in SQL Server Data Tools (SSDT).

Deleting a DSV is permanent. You cannot restore a deleted DSV to a Analysis Services project or database.

DSVs that other objects depend on cannot be deleted from an Analysis Services database opened by SQL Server Data Tools (SSDT) in online mode. To delete a DSV from a project that is connected o a database running on a server, you must first delete all objects in the Analysis Services database that depend on that DSV before deleting the DSV itself.

Deleting a DSV will invalidate other Analysis Services objects that depend on it, so before you delete the DSV, you will see the list of objects that will become invalid once the DSV is removed. Review this list carefully to be sure that it does not include objects you still expect to use.

See Also

Data Source Views in Multidimensional Models
Change Properties in a Data Source View (Analysis Services)

Change Properties in a Data Source View (Analysis Services)

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After defining a data source view using the Data Source View Wizard and adding tables, views, named calculations, and named queries to the data source view, you may want to change properties related to:

- Data source view matching criteria
- Advanced data source view options
- Object names
- Object metadata

You can also view object metadata retrieved from the data source that you cannot modify.

Viewing or Changing Data Source View Properties

Data source view properties, other than a description of the data source view, are set by the Data Source View Wizard when you initially define the data source view. The following table lists and describes the properties of a data source view.

> **NOTE**
> The Properties pane shows properties for the .dsv file as well as the DSV object. To view the properties of the object, double-click it in Solution Explorer. The Properties will update to reflect the properties you see in the following table.

PROPERTY	DESCRIPTION
Data Source	Specifies the data source within the data source view whose properties you are viewing
Description	Specifies a description of the data source view
Name	Specifies the name of the data source view that appears in Solution Explorer or the Analysis Services database. You can change the data source view name here or in Solution Explorer.
NameMatchingCriteria	The name matching criteria for the data source. The default is (none) if primary key - foreign key relationships were detected by the Data Source View Wizard. Regardless of whether this property was set by the Data Source View Wizard, you can specify a value here. If database relationships exist and you specify name matching criteria, both will be used to infer relationships between existing tables and newly added tables.

PROPERTY	DESCRIPTION
RetrieveRelationships	Specifies whether relationships are retrieved from the database. Default is True.
SchemaRestriction	Specifies the restrictions, if any, on the schemas retrieved from a data source. By default, no schema restrictions exist.

Viewing or Changing DataTable Properties

DataTable properties are the properties of tables, views, and named queries in a data source view. These properties are set when any of these objects is added to the data source view. The following table lists and describes the properties of **DataTable** objects in a data source view.

PROPERTY	DESCRIPTION
AllowChangesDuringGeneration	Specifies whether the Schema Generation Wizard has permission to overwrite a data source view table during regeneration. This property will only exist on tables initially generated by the Schema Generation Wizard. For more information, see Understanding Incremental Generation.
DataSource	Specifies the data source for the object. You cannot edit this property.
Description	Specifies the description for the table, view or named query. If the underlying database table or view had a description stored as an extended property, this value appears. You can edit this property.
FriendlyName	Specifies a name for the table or view that is easier for users to understand or more relevant to the subject area. By default, the **FriendlyName** property of a table or view is the same as the **Name** property of the table or view. The **FriendlyName** property is used by OLAP and data mining objects when defining object names based on tables or views. You can edit this property.
Name	Specifies the name of the underlying table or view, or the name of the named query. The **Name** property is used by OLAP and data mining objects when defining object names based on named queries. This property is only editable for named queries.
QueryDefinition	Specifies the named query definition. This property is only applicable to named queries and is not directly editable. To edit this property, you edit the named query itself.
Schema	Specify the database schema applicable to the table, view or named query. This property is not editable.
TableType	Specifies the type of table for the table, view or named query. This property is not editable.

Viewing or Changing DataColumn Properties

DataColumn properties are the properties of columns in tables, views, and named queries in a data source view. These properties are set when any of these objects is added to the data source view, either from the underlying table or view, from a named query, or as defined by a named calculation. The following table lists and describes the properties of **DataColumn** objects in a data source view.

PROPERTY	DESCRIPTION
AllowNull	Specifies the nullability property of the column based on the column in the underlying table, value or named query. This property is not editable.
DataType	Specifies the data type of the column based on the column in the underlying table, value or named query. This property is not directly editable. However, if you need to change the data type of a column from a table or view, replace the table with a named query that converts the column to the desired data type.
DateTimeMode	Specifies the date serialization format for **DateTime** columns. The default value is **UnspecifiedLocal**. This property is editable.
Description	Specifies the description for the column. If the underlying database column had a description stored as an extended property, this value appears. You can edit this property.
FriendlyName	Specifies the name for a column from a table or view that is easier for users to understand or more relevant to the subject area. By default, the **FriendlyName** property of a column from a table or view is the same as the **Name** property of the column. The **FriendlyName** property is used by OLAP and data mining objects when defining attributes based on columns from tables or views. You can edit this property.
Length	Specifies the maximum length of the column, based on the data in the column in the underlying table or view.
Name	Specifies the name of the underlying column, or the name of the named calculation. The **Name** property is used by OLAP and data mining objects when defining attributes based on named calculations. This property is only editable for named calculations.

See Also

Data Source Views in Multidimensional Models
Work with Diagrams in Data Source View Designer (Analysis Services)

Schema Generation Wizard (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

SQL Server Data Tools (SSDT) supports two methods of working with relational schemas when defining OLAP objects within an Analysis Services project or database. Generally, you will define OLAP objects based on a logical data model constructed in a data source view within an Analysis Services project or database. This data source view is defined based on schema elements from one or more relational data sources, as customized in the data source view.

Alternatively, you can define OLAP objects first, and then generate a data source view, a data source, and the underlying relational database schema that supports these OLAP objects. This relational database is referred to as the subject area database.

This approach is sometimes called top-down design and is frequently used for prototyping and analysis modeling. When you use this approach, you use the Schema Generation Wizard to create the underlying data source view and data source objects based on the OLAP objects defined in an Analysis Services project or database.

This is an iterative approach. You will most likely rerun the wizard multiple times as you change the design of the dimensions and cubes. Each time you run the wizard, it incorporates the changes into the underlying objects and, as much as is possible, preserves the data contained in the underlying databases.

The schema that is generated is a SQL Server relational database engine schema. The wizard does not generate schemas for other relational database products.

The data that is populated in the subject area database is added separately, using whatever tools and techniques you use to populate a SQL Server relational database. In most cases, the data is preserved when you rerun the wizard, but there are exceptions. For example, some data must be dropped if you delete a dimension or an attribute that contains data. If the Schema Generation Wizard must drop some data because of a schema change, you receive a warning before the data is dropped and can then cancel the regeneration.

As a general rule, any change that you make to an object that was originally generated by the Schema Generation Wizard is overwritten when the Schema Generation Wizard subsequently regenerates that object. The primary exception to this rule is when you add columns to a table that the Schema Generation Wizard generated. In this case, the Schema Generation Wizard preserves the columns that you added to the table, as well as the data in these columns.

In this section

The following table lists additional topics that explain how to work with the Schema Generation Wizard.

TOPIC	DESCRIPTION
Use the Schema Generation Wizard (Analysis Services)	Describes how to generate the schema for the subject area and staging area databases.
Understanding the Database Schemas	Describes the schema that is generated for the subject area and staging area databases.
Understanding Incremental Generation	Describes the incremental generation capabilities of the Schema Generation Wizard.

See Also

Data Source Views in Multidimensional Models
Data Sources in Multidimensional Models
Supported Data Sources (SSAS - Multidimensional)

Use the Schema Generation Wizard (Analysis Services)

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Schema Generation Wizard requires a limited amount of information during the generation phase. Most of the information that the Schema Generation Wizard requires for generating relational schemas is extracted from the Analysis Services cubes and dimensions that you already created in the project. Additionally, you can customize how the subject area database schema is generated and how objects in the schema are named.

Start the Wizard

You can open the Schema Generation Wizard from SQL Server Data Tools (SSDT) in several different ways:

- Right-click the Analysis Services project object and then click **Generate Relational Schema** from the context menu.
- Click the Analysis Services project object, and then click **Generate Relational Schema** from the **Database** menu.
- Start the wizard from within the Dimension Wizard by clicking the **Generate Schema Now** check box on the last page of the wizard.

Step 1: Specify Targets

You must specify the data source view (DSV) in which you want the Schema Generation Wizard to generate the schema for the subject area database. Although you can select an existing DSV, typically you create a new one based on a data source. You can create the data source based on an existing or a new connection, or based on another object. The Schema Generation Wizard generates the schema for the subject area database in the database referenced by the data source, as well as the data source view. The Schema Generation Wizard does not create the subject area database itself; instead, the wizard creates the relational schema to support the cubes and dimensions in an existing database that you specify.

When the Schema Generation Wizard generates the underlying objects, it binds the Analysis Services dimensions and cubes to the generated tables and columns by using data source view-style bindings.

> **NOTE**
> To unbind Analysis Services dimensions and cubes from previously generated objects, delete the data source view to which the Analysis Services cubes and dimensions are bound, and then define a new data source view for the cubes and dimensions by using the Schema Generation Wizard.

Step 3: Specify Schema Options for the Subject Area Database

The Schema Generation Wizard provides a number of options for defining the schema that is generated for the subject area database. You can specify these options on the **Subject Area Database Schema Options** page of the wizard.

Specifying the Schema Owner

You can specify the owner of the schema by setting the value of **Owning schema** to a valid string. The default owner of the schema is the Analysis Services project, but you can specify any desired schema owner.

Specifying Primary Keys, Indexes, and Constraints

The Schema Generation Wizard by default creates a primary key constraint in each dimension table in the subject area database. The primary key corresponds to the attribute that is designated as the key attribute in the corresponding Analysis Services dimension. This constraint improves processing performance in most environments, with minimal cost. Logical primary keys are always created in the data source view, even if you choose not to create the primary key in the subject area database. To define primary key constraints on dimension tables, select **Create primary keys on dimension tables**.

The wizard by default also creates indexes on the foreign key columns in each fact table. These indexes improve processing performance in most environments. Performance is typically improved because the processing queries that Analysis Services generates to retrieve new data from the subject area database typically include a significant number of join statements between the fact table and the dimension tables. To define indexes on the foreign key columns in each fact table, select **Create indexes**.

Finally, the wizard by default enforces referential integrity between the fact table and each of the dimension tables. If you choose not to enforce referential integrity, the Schema Generation Wizard still creates these relationships in the database and the data source view. To enforce referential integrity, select **Enforce referential integrity**.

Preserving Data for Incremental Generation

The Schema Generation Wizard by default attempts to preserve data when the database schema is regenerated. If the Schema Generation Wizard has to delete any rows because of a schema change, you receive a warning before the rows are deleted. For example, rows may have to be deleted to solve referential integrity issues because you dropped a dimension or because a data type changed when you changed a dimension attribute. To preserve data when the database schema is regenerated, select **Preserve data on regeneration**.

Step 4: Specify Naming Conventions

You can define the naming conventions that the Schema Generation Wizard uses when generating certain objects in the subject area database on the **Specify Naming Conventions** page of the wizard. For more information about the options available on the **Specify Naming Conventions** page, see Specify Naming Conventions (Schema Generation Wizard) (Analysis Services - Multidimensional Data).

Understanding the Database Schemas

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Schema Generation Wizard generates a denormalized relational schema for the subject area database based on the dimensions and measure groups in Analysis Services. The wizard generates a relational table for each dimension to store dimension data, which is called a dimension table, and a relational table for each measure group to store fact data, which is called a fact table. The wizard ignores linked dimensions, linked measure groups, and server time dimensions when it generates these relational tables.

Validation

Before it begins to generate the underlying relational schema, the Schema Generation Wizard validates the Analysis Services cubes and dimensions. If the wizard detects errors, it stops and reports the errors to the Task List window in SQL Server Data Tools (SSDT). Examples of errors that prevent generation include the following:

- Dimensions that have more than one key attribute.
- Parent attributes that have different data types than the key attributes.
- Measure groups that do not have measures.
- Degenerate dimensions or measures that are improperly configured.
- Surrogate keys that are improperly configured, such as multiple attributes using the **ScdOriginalID** attribute type or an attribute using the **ScdOriginalID** that is not bound to a column using the integer data type.

Dimension Tables

For each dimension, the Schema Generation Wizard generates a dimension table to be included in the subject area database. The structure of the dimension table depends on the choices made while designing the dimension on which it is based.

Columns

The wizard generates one column for the bindings associated to each attribute in the dimension on which the dimension table is based, such as the bindings for the **KeyColumns**, **NameColumn**, **ValueColumn**, **CustomRollupColumn**, **CustomRollupPropertiesColumn**, and **UnaryOperatorColumn** properties of each attribute.

Relationships

The wizard generates a relationship between the column for each parent attribute and the primary key of the dimension table.

The wizard also generates a relationship to the primary key in each additional dimension table defined as a referenced dimension in the cube, if applicable.

Constraints

The wizard generates a primary key constraint, by default, for each dimension table based on the key attribute of the dimension. If the primary key constraint is generated, a separate name column is generated by default. A logical primary key is created in the data source view even if you decide not to create the primary key in the database.

> **NOTE**
> An error occurs if more than one key attribute is specified in the dimension on which the dimension table is based.

Translations
The wizard generates a separate table to hold the translated values for any attribute that requires a translation column. The wizard also creates a separate column for each of the required languages.

Fact Tables

For each measure group in a cube, the Schema Generation Wizard generates a fact table to be included in the subject area database. The structure of the fact table depends on the choices made while designing the measure group on which it is based, and the relationships established between the measure group and any included dimensions.

Columns
The wizard generates one column for each measure, except for measures that use the **Count** aggregation function. Such measures do not require a corresponding column in the fact table.

The wizard also generates one column for each granularity attribute column of each regular dimension relationship on the measure group, and one or more columns for the bindings associated to each attribute of a dimension that has a fact dimension relationship to the measure group on which this table is based, if applicable.

Relationships
The wizard generates one relationship for each regular dimension relationship from the fact table to the dimension table's granularity attribute. If the granularity is based on the key attribute of the dimension table, the relationship is created in the database and in the data source view. If the granularity is based on another attribute, the relationship is created only in the data source view.

If you chose to generate indexes in the wizard, a non-clustered index is generated for each of these relationship columns.

Constraints
Primary keys are not generated on fact tables.

If you chose to enforce referential integrity, referential integrity constraints are generated between dimension tables and fact tables where applicable.

Translations
The wizard generates a separate table to hold the translated values for any property in the measure group that requires a translation column. The wizard also creates a separate column for each of the required languages.

Data Type Conversion and Default Lengths

Schema Generation Wizard ignores data types in all cases except for columns that use the SQL Server **wchar** data type. The **wchar** data size translates directly to the **nvarchar** data type. However, if the specified length of a column using the **wchar** size is larger than 4000 bytes, the Schema Generation Wizard generates an error.

If a data item, such as the binding for an attribute, has no specified length, the default length listed in the following table is used for the column.

DATA ITEM	DEFAULT LENGTH (BYTES)
KeyColumn	50

DATA ITEM	DEFAULT LENGTH (BYTES)
NameColumn	50
CustomRollupColumn	3000
CustomRollupPropertiesColumn	500
UnaryOperatorColumn	1

See Also

Understanding Incremental Generation
Manage Changes to Data Source Views and Data Sources

Understanding Incremental Generation

5/16/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Following the initial schema generation, you can change cube and dimension definitions by using SQL Server Data Tools (SSDT), and then rerun the Schema Generation Wizard. The wizard updates the schema in the subject area database and in the associated data source view to reflect the changes, and retaining the data that currently exists in the tables to be regenerated, to the extent possible. If you changed the tables after the initial generation, the Schema Generation Wizard preserves those changes when possible by using the following rules:

- If a table was previously generated by the wizard, the table is overwritten. You can prevent a table that was generated by the wizard from being overwritten by changing the **AllowChangesDuringGeneration** property for the table in the data source view to **false**. When you take control of a table, the table is treated like any other user-defined table and is not affected during regeneration. After you remove a table from generation, you can later change the **AllowChangesDuringGeneration** property for the table in the data source view to **true** and reopen the table for changes by the wizard. For more information, see Change Properties in a Data Source View (Analysis Services).
- If a table was added to the data source view or to the underlying database by something other than the wizard, the table is not overwritten.

When the Schema Generation Wizard regenerates tables that were previously generated in the subject area database, you can choose to have the wizard preserve existing data in those tables.

Supporting Data Preservation

As a general rule, the Schema Generation Wizard preserves data that is stored in the tables that it generated. In addition, if you add columns to tables that the wizard generated, the wizard preserves that data as well. You can use this capability to add or modify your dimensions and cubes and then regenerate the underlying objects without having to reload the data stored in the underlying tables.

> **NOTE**
> If you are loading data from delimited text files, you can also choose whether the Schema Generation Wizard overwrites these files and the data contained in them during regeneration. Text files are either overwritten completely or not at all. The Schema Generation Wizard does not partially overwrite these files. By default, these files are not overwritten.

Partial Preservation

The Schema Generation Wizard cannot preserve existing data under some circumstances. The following table provides examples of situations in which the wizard cannot preserve all the existing data in the underlying tables during regeneration.

TYPE OF DATA CHANGE	TREATMENT
Incompatible data type change	The Schema Generation Wizard uses standard SQL Server data type conversions, whenever possible, to convert existing data from one data type to another. However, when you change an attribute's data type to a type that is incompatible with the existing data, the wizard drops the data for the affected column.

TYPE OF DATA CHANGE	TREATMENT
Referential integrity errors	If you change a dimension or cube that contains data and the change causes a referential integrity error during regeneration, the Schema Generation Wizard drops all data in the foreign key table. The data that is dropped is not limited to the column that caused the foreign key constraint violation or to the rows that contain the referential integrity errors. For example, if you change the dimension key to an attribute that has non-unique or null data, all existing data in the foreign key table is dropped. Furthermore, dropping all the data in one table can have a cascading effect and can cause other referential integrity violations.
Deleted attribute or dimension	If you delete an attribute from a dimension, the Schema Generation Wizard deletes the column that is mapped to the deleted attribute. If you delete a dimension, the wizard deletes the table that is mapped to the deleted dimension. In these cases, the wizard drops the data that is contained in the deleted column or table.

The Schema Generation Wizard issues a warning before it drops any data so that you can cancel the wizard without losing any data. However, the Schema Generation Wizard is not able to differentiate between anticipated data loss and unanticipated data loss. When you run the wizard, a dialog box lists the tables and columns that contain data that will be dropped. You can either have the wizard continue and drop the data, or you can cancel the wizard and revise the changes you made to the tables and columns.

Supporting Cube and Dimension Changes

When you change the properties of dimensions and cubes, the Schema Generation Wizard regenerates the appropriate objects in the underlying subject area database, as well as in the related data source view, as described in the following table.

Deleting an object, such as a dimension, cube, or attribute.
The Schema Generation Wizard deletes the underlying objects to which the deleted object is mapped. If you add columns to a table that the wizard generated, the new columns do not prevent that table from being deleted. Deleting an object causes the data stored in the underlying objects to be dropped, and may also cause other data to be dropped if referential integrity errors occur.

Renaming an object, such as a dimension, cube, or attribute.
The Schema Generation Wizard renames the underlying objects to which the renamed object is mapped. The wizard also renames all affected objects, such as primary keys. Existing data stored in the underlying objects is preserved.

Modifying an object, such as changing its data type.
The Schema Generation Wizard modifies the underlying objects to which the changed object is mapped. Existing data stored in the underlying objects in the databases is preserved, unless the new data type is incompatible with the existing data.

Adding a new object, such as a dimension, cube, or attribute.
The Schema Generation Wizard adds underlying objects to which the new object is mapped.

If the Schema Generation Wizard cannot make the required change because of the presence of a user object in the subject area database (because the Database Engine returns an error), the Schema Generation Wizard fails and displays the error returned by the Database Engine. For example, if you create a primary key constraint or a non-clustered index on a table after the wizard generated the table, the Schema Generation Wizard does not drop that table because it did not create the constraint or the index.

Supporting Schema Changes

When you change the properties of the tables or columns in the subject area database or in the associated data source view, the Schema Generation Wizard treats the changes as described in the following table.

Deleting a table or a column generated by the Schema Generation Wizard.
If you delete a table or a column generated by the Schema Generation Wizard, the wizard regenerates the deleted table. The wizard provides no warning that the deleted table or column will be regenerated.

Changing the properties of a table or column generated by the Schema Generation Wizard.
If you modify the properties of a table or a column generated by the Schema Generation Wizard, the wizard regenerates the changed table without the change. For example, if you change the data type or nullability of a column, or the filegroup of a table generated by the Schema Generation Wizard, the change does not survive the regeneration. The wizard provides no warning that the changed object will be regenerated without the change.

Adding a column to a table generated by the Schema Generation Wizard or adding a table to the subject area database or staging area database.
If you add a column to a table generated by the Schema Generation Wizard, the wizard preserves the additional column, along with any data stored in it, during regeneration. However, if you add a table to the subject area database or the staging area database, the Schema Generation Wizard does not incorporate the new table. The added column, or the added table, is not reflected in the Analysis Services project, the Analysis Services database, the DTS packages, the data source view, or any other place in the schema that is generated.

Supporting Data Source and Data Source View Changes

When the Schema Generation Wizard is rerun, it reuses the same data source and data source view that it used for the original generation. If you add a data source or a data source view, the wizard does not use it. If you delete the original data source or data source view after the initial generation, you must run the wizard from the beginning. All previous settings in the wizard are also deleted. Any existing objects in an underlying database that were bound to a deleted data source or data source view are treated as user-created objects the next time you run the Schema Generation Wizard.

If the data source view does not reflect the actual state of the underlying database at the time of generation, the Schema Generation Wizard may encounter errors when it generates the schemas for the subject area database and the staging area database. For example, if the data source view specifies that the data type for a column is set to **int**, but the data type for the column is actually set to **string**, the Schema Generation Wizard sets the data type for the foreign key to **int** to match the data source view and then fails when it creates the relationship because the actual data type is **string**.

On the other hand, if you change the data source connection string to a different database from the previous generation, no error is generated. The new database is used, and no change is made to the previous database.

See Also

Manage Changes to Data Source Views and Data Sources
Schema Generation Wizard (Analysis Services)

Manage Changes to Data Source Views and Data Sources

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When the Schema Generation Wizard is rerun, it reuses the same data source and data source view that it used for the original generation. If you add a data source or a data source view, the wizard does not use it. If you delete the original data source or data source view after the initial generation, you must run the wizard from the beginning. All previous settings in the wizard are also deleted. Any existing objects in an underlying database that were bound to a deleted data source or data source view are treated as user-created objects the next time you run the Schema Generation Wizard.

If the data source view does not reflect the actual state of the underlying database at the time of generation, the Schema Generation Wizard may encounter errors when it generates the schema for the subject area database. For example, if the data source view specifies that the data type for a column is **int**, but data type for the column is actually **string**, the Schema Generation Wizard sets the data type for the foreign key to **INT** to match the data source view and then fails when it creates the relationship because the actual type is **string**.

On the other hand, if you change the data source connection string to a different database from the previous generation, no error is generated. The new database is used, and no change is made to the previous database.

See Also

Understanding Incremental Generation

Dimensions in Multidimensional Models

12/6/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A database dimension is a collection of related objects, called attributes, which can be used to provide information about fact data in one or more cubes. For example, typical attributes in a product dimension might be product name, product category, product line, product size, and product price. These objects are bound to one or more columns in one or more tables in a data source view. By default, these attributes are visible as attribute hierarchies and can be used to understand the fact data in a cube. Attributes can be organized into user-defined hierarchies that provide navigational paths to assist users when browsing the data in a cube.

Cubes contain all the dimensions on which users base their analyses of fact data. An instance of a database dimension in a cube is called a cube dimension and relates to one or more measure groups in the cube. A database dimension can be used multiple times in a cube. For example, a fact table can have multiple time-related facts, and a separate cube dimension can be defined to assist in analyzing each time-related fact. However, only one time-related database dimension needs to exist, which also means that only one time-related relational database table needs to exist to support multiple cube dimensions based on time.

Defining Dimensions, Attributes, and Hierarchies

The simplest method for defining database and cube dimensions, attributes, and hierarchies is to use the Cube Wizard to create dimensions at the same time that you define the cube. The Cube Wizard will create dimensions based on the dimension tables in the data source view that the wizard identifies or that you specify for use in the cube. The wizard then creates the database dimensions and adds them to the new cube, creating cube dimensions.

When you create a cube, you can also add to the new cube any dimensions that already exist in the database. These dimensions may have been previously defined for another cube or by the Dimension Wizard. After a database dimension has been defined, you can modify and configure the database dimension in Dimension Designer. You can also customize the cube dimension, to a limited extent, in Cube Designer.

> **NOTE**
>
> You can also design and configure dimensions, attributes, and hierarchies programmatically by using either XMLA or Analysis Management Objects (AMO). For more information, see Analysis Services Scripting Language (ASSL for XMLA) and Developing with Analysis Management Objects (AMO).

In This Section

The following table describes the topics in this section.

Define Database Dimensions
Describes how to modify and configure a database dimension by using Dimension Designer.

Dimension Attribute Properties Reference
Describes how to define, modify, and configure a database dimension attribute by using Dimension Designer.

Define Attribute Relationships
Describes how to define, modify, and configure an attribute relationship by using Dimension Designer.

Create User-Defined Hierarchies

Describes how to define, modify, and configure a user-defined hierarchy of dimension attributes by using Dimension Designer.

Use the Business Intelligence Wizard to Enhance Dimensions

Describes how to enhance a database dimension by using the Business Intelligence Wizard.

See Also

Cubes in Multidimensional Models

Create a Dimension Using the Dimension Wizard

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can create a new dimension by using the Dimension Wizard in SQL Server Data Tools (SSDT).

To create a new dimension

1. In **Solution Explorer**, right-click **Dimensions**, and then click **New Dimension**.
2. On the **Select Creation Method** page of the Dimension Wizard, select **Use an existing table**, and then click **Next**.

   > **NOTE**
   > You might occasionally have to create a dimension without using an existing table. For more information, see Create a Dimension by Generating a Non-Time Table in the Data Source and Create a Time Dimension by Generating a Time Table.

3. On the **Specify Source Information** page, do the following procedures:

   a. In the **Data source view** list, select a data source view.

   b. In the **Main table** list, select the main dimension table.

   c. In the **Key columns** list, review the key columns that the wizard has automatically selected based on the primary key that is defined in the main dimension table. To change this default setting, specify the key columns that link the dimension table to the fact table.

   d. In the **Name column** drop-down list, review the name column that the wizard has automatically selected.

   This default name is appropriate when the column contains descriptive information. However, you might want to specify a name that contains values that are more meaningful for the end user. For example, if a product category attribute in a Products dimension uses the **ProductCategoryKey** column as its key column, you can specify the **ProductCategoryName** column as its name column.

   If the **Key columns** list contains multiple key columns, you must specify a name column that provides the member values for the key attribute. To do this, you can create a named calculation in the data source view and use that as the name column.

   e. Click **Next**.

4. On the **Select Related Tables** page, select the related tables that you want to include in your dimension, and then click **Next**.

   > **NOTE**
   > The **Select Related Tables** page appears if the main dimension table that you specified has relationships to other dimension tables.

5. On the **Select Dimension Attributes** page, select the attributes that you want to include in the dimension, and then click **Next**.

Optionally, you can change the attribute names, enable or disable browsing, and specify the attribute type.

> **NOTE**
>
> To make the **Enable Browsing** and **Attribute Type** fields of an attribute active, the attribute must be selected for inclusion in the dimension.

6. On the **Define Account Intelligence** page, in the **Built-In Account Types** column, select the account type, and then click **Next**.

   The account type must correspond to the account type of the source table that is listed in the **Source Table Account Types** column.

   > **NOTE**
   >
   > The **Define Account Intelligence** page appears if you defined an **Account Type** dimension attribute on the **Select Dimension Attributes** page of the wizard.

7. On the **Completing the Wizard** page, enter a name for the new dimension and review the dimension structure. If you want to make changes, click **Back**; otherwise, click **Finish**.

   > **NOTE**
   >
   > You can use Dimension Designer after you complete the Dimension Wizard to add, remove, and configure attributes and hierarchies in the dimension.

See Also

Create a Dimension by Using an Existing Table

Create a Dimension by Using an Existing Table

5/16/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, you can use the Dimension Wizard in SQL Server Data Tools (SSDT) to create a dimension from an existing table. You do this by selecting the **Use an existing table** option on the **Select Creation Method** page of the wizard. If you select this option, the wizard bases the dimension structure on the dimension tables, their columns, and any relationships between those columns in an existing data source view. The wizard samples the data in the source table and related tables. It uses this data to define attribute columns that are based on the columns in the dimension tables, and to define hierarchies of attributes (called *user-defined* hierarchies). After you use the Dimension Wizard to create your dimension, you can use Dimension Designer to add, remove, and configure attributes and hierarchies in the dimension.

When you are using an existing table to create a dimension, the Dimension Wizard guides you through the following:

- Specifying the source information
- Selecting related tables
- Selecting dimension attributes
- Defining account intelligence

> **NOTE**
> For the step-by-step instructions that correspond to the information presented in this topic, see Create a Dimension Using the Dimension Wizard.

Specifying the Source Information

You specify the source information on the **Specify Source Information** page. You begin this process by selecting the data source view that contains the table on which you want the dimension to be based. You then specify the main dimension table for the dimension that you are defining. The main dimension table is the table that is directly linked to the fact table. For example, specify a Product table as the main table for a Products dimension, or an Employee table for an Employees dimension. The wizard automatically selects a key column that is based on the primary key in the data source view. However, you can change the key column as appropriate. The key column determines the members of the dimension. For example, you would define ProductKey as the key column for a Product dimension.

Optionally, you can define a column that contains the member name. By default, the member name that will be displayed to users is the value from the key column. The values in a key column, such as ProductID or EmployeeID, are often unique, system-generated keys that are meaningless to the user. You can often provide more meaningful information to the user if you change the name that users see to a corresponding value in some other column in the dimension. For example, you can define a member name column that contains product or employee names. If you change the member name, users see a more descriptive name, but queries still use the key column values to correctly distinguish members that share the same name. If you specify a composite key for the key column, you must also specify the column that provides the member values for the key attribute. For more information about how to configure attribute properties, see Dimension Attribute Properties Reference.

Selecting Related Tables

> **NOTE**
> The wizard skips this step if the main dimension table has no relationships defined in the data source view to other dimension tables.

If you are building a snowflake dimension, you specify the related tables from which additional attributes will be defined on the **Select Related Tables** page. For example, you are building a customer dimension in which you want to define a customer geography table. In this case, you might define a geography table as a related table.

Selecting Dimension Attributes

After selecting the dimension tables, you use the **Select Dimension Attributes** page to select the attributes that you want to include in the dimension from these tables. All of the underlying columns from all these tables are available as potential dimension attributes. The dimension key attribute must be selected and enabled for browsing.

By default, the wizard sets the type of an attribute to **Regular**. However, you might want to map specific attributes to a different attribute type that better represents the data. For example, the dbo.DimAccount table in the Adventure Works DW sample database contains an AccountCodeAlternateKey column that provides the account number. Instead of setting the type to **Regular** for this attribute, you might want to map this attribute to the **Account Number** type.

> **NOTE**
> If the dimension type and standard attribute types are not set when you create the dimension, use the Business Intelligence Wizard to set these values after you create the dimension. For more information, see Add Dimension Intelligence to a Dimension or (for an Accounts type dimension) Add Account Intelligence to a Dimension.

The wizard automatically sets the dimension type based on the attribute types that are specified. The attribute types specified in the wizard set the **Type** property for the attributes. The **Type** property settings for the dimension and its attributes provide information about the contents of a dimension to server and client applications. In some cases, these **Type** property settings only provide guidance for client applications and are optional. In other cases, as for Accounts, Time, or Currency dimensions, these **Type** property settings determine specific server-based behavior and might be required to implement certain cube behavior.

For more information about dimension and attribute types, see Dimension Types, Configure Attribute Types.

Defining Account Intelligence

> **NOTE**
> The Dimension Wizard displays this step only if you defined an **Account Type** dimension attribute on the **Select Dimension Attributes** page of the wizard.

You use the **Define Account Intelligence** page to create an Account type dimension. If you are creating an Account type dimension, you have to map standard account types supported by Analysis Services to members of the account type attribute in the dimension. The server uses these mappings to provide separate aggregation functions and aliases for each type of account data.

To map these account types, the wizard provides a table with the following columns:

- The **Source Table Account Types** column lists account types from the data source table.
- The **Built-In Account Types** column lists the corresponding standard account types supported by the server. If the source data uses standard names, the wizard automatically maps the source type to the server type, and populates the **Built-In Account Types** column with this information. If the server does not map the account types or you want to change the mapping, select a different type from the list in the **Built-In Account Types** column.

> **NOTE**
> If the account types are not mapped when the wizard creates an Accounts dimension, use the Business Intelligence Wizard to configure these mappings after you create the dimension. For more information, see Add Account Intelligence to a Dimension.

Completing the Wizard

The wizard scans dimension tables to detect relationships. The wizard will create attribute relationships between key attributes in snowflake dimensions automatically.

The wizard also automatically detects if a parent-child relationship exists in the dimension. A parent-child relationship exists when a parent attribute references members of the key attribute of the dimension. This relationship defines hierarchical relationships and aggregation paths between leaf members of the dimension. For more information about parent-child hierarchies, see Attributes in Parent-Child Hierarchies.

On the **Completing the Wizard** page, you complete the wizard by typing a name for the new dimension and reviewing the dimension structure.

See Also

Create a Dimension by Generating a Non-Time Table in the Data Source
Create a Time Dimension by Generating a Time Table
Dimension Attribute Properties Reference
Create a Time Dimension by Generating a Time Table
Create a Dimension by Generating a Non-Time Table in the Data Source

Create a Time Dimension by Generating a Time Table

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, you can use the Dimension Wizard in SQL Server Data Tools (SSDT) to create a time dimension when no time table is available in the source database. You do this by selecting one of the following options on the **Select Creation Method** page:

- **Generate a time table in the data source** Select this option when you have permission to create objects in the underlying data source. The wizard will then generate a time table and store this table in the data source. The wizard then creates the time dimension from this time table.
- **Generate a time table on the server** Select this option when you do not have permission to create objects in the underlying data source. The wizard will then generate and store a table on the server instead of in the data source. (The dimension created from a time table on the server is called a *server time dimension*.) The wizard then creates the server time dimension from this table.

When you create a time dimension, you specify the time periods, and also the start and end dates for the dimension. The wizard uses the specified time periods to create the time attributes. When you process the dimension, Analysis Services generates and stores the data that is required to support the specified dates and periods. The wizard uses the attributes created for a time dimension to recommend hierarchies for the dimension. The hierarchies reflect the relationships between different time periods and take account of different calendars. For example, in a standard calendar hierarchy, a Weeks level appears under a Years level but not under a Months level, because weeks divide evenly into years but not into months. In contrast, in a manufacturing or reporting calendar hierarchy, weeks divide evenly months, so a Weeks level appears under a Months level.

Define Time Periods

You use the **Define Time Periods** page of the wizard to specify the range of dates that you want to include in the dimension. For example, you might select a range that starts on January 1 of the earliest year in your data and that ends one or two years beyond the current year (to allow for future transactions). Transactions that are outside the range either do not appear or appear as unknown members in the dimension, depending on the **UnknownMemberVisible** property setting for the dimension. You can also change the first day of the week used by your data (the default is Sunday).

Select the time periods to use when the wizard creates the hierarchies that apply to your data, such as Years, Half Years, Quarters, Trimesters, Months, Ten Days, Weeks, or Date. You must always select at least the Date time period. The Date attribute is the key attribute for the dimension, so the dimension cannot function without it.

Next to **Language for time member names**, select the language to be used to label the members of the dimension.

After you create a time dimension that is based on a range of dates, you can use Dimension Designer to add or remove time attributes. Because the Date attribute is the key attribute for the dimension, you cannot remove it from the dimension. To hide the Date attribute from users, you can change the **AttributeHierarchyVisible** property on the attribute to **False**.

Select Calendars

The standard (Gregorian) 12-month calendar, starting on January 1 and ending on December 31, is always included when you create a time dimension. On the **Select Calendars** page of the wizard, you can specify additional calendars on which to base hierarchies in the dimension. For descriptions of the calendar types, see Create a Date type Dimension.

Depending on which time periods you select on the **Define Time Periods** page of the wizard, the calendar selections determine attributes that are created in the dimension. For example, if you select the **Year** and **Quarter** time periods on the **Define Time Periods** page of the wizard and select **Fiscalcalendar** on the **Select Calendars** page, the FiscalYear, FiscalQuarter, and FiscalQuarterOfYear attributes are created for the fiscal calendar.

The wizard also creates calendar-specific hierarchies that are composed of the attributes that are created for the calendar. For every calendar, each level in every hierarchy rolls up into the level above it. For example, in the standard 12-month calendar, the wizard creates a hierarchy of Years and Weeks or Years and Months. However, weeks are not contained evenly within months in a standard calendar, so there is no hierarchy of Years, Months, and Weeks. In contrast, weeks in a reporting or manufacturing calendar are evenly divided into months, so in these calendars weeks roll up into months.

Completing the Dimension Wizard

On the **Completing the Wizard** page, review the attributes and hierarchies created by the wizard, and then name the time dimension. Click **Finish** to complete the wizard and create the dimension. After you complete the dimension, you can change it by using Dimension Designer.

See Also

Data Source Views in Multidimensional Models
Create a Date type Dimension
Database Dimension Properties
Dimension Relationships
Create a Dimension by Generating a Non-Time Table in the Data Source

Create a Dimension by Generating a Non-Time Table in the Data Source

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, you can use the Dimension Wizard in SQL Server Data Tools (SSDT) to create a dimension without using an existing data source. You do this by selecting the **Generate a non-time table in the data source** option of the **Select Creation Method** page of the wizard. To create a new dimension table in the underlying data source, you must have permission to create objects in the underlying data source. When defining a dimension without a predefined data source view, you can either define the dimension from scratch or use a dimension template.

The Dimension Wizard provides sample dimension templates from which you can build a common dimension type. You can select from the following types of dimensions:

- Account
- Customer
- Date
- Department
- Destination Currency
- Employee
- Geography
- Internet Sales Order Details
- Organization
- Product
- Promotion
- Reseller Sales Order Details
- Reseller
- Sales Channel
- Sales Reason
- Sales Summary Order Details
- Sales Territory
- Scenario
- Source Currency

Each of the standard templates supports attributes that you can choose to include in the dimension. You can also add your own template files for dimensions that are commonly used with your data. The dimension templates are located in the following folder:

```
C:\Program Files\Microsoft SQL Server\100\Tools\Templates\olap\1033\Dimension Templates
```

You can use Dimension Designer after you complete the Dimension Wizard to add, remove, and configure attributes and hierarchies in the dimension.

When you are creating a non-time dimension without using a data source, the Dimension Wizard guides you through the steps of specifying the dimension type, and identifying the key attribute and slowly changing dimensions.

Specify Dimension Type

On the **Specify Dimension Type** page of the Dimension Wizard, you can specify the dimension type. If you are building the dimension based on a template, the dimension type is defined for you. On this page, you can also select standard attributes for the specified dimension type if any are available.

If you selected a template that corresponds to a dimension type, this page is populated with the options for that dimension type. If you did not select a template, or if there is no corresponding dimension type, the default dimension type is **Regular**. If a dimension type is not already selected, select the most appropriate type for the dimension that you are creating. If no appropriate type is listed for **Dimension type**, use **Regular**.

When you select a dimension type, the wizard lists the attribute types that apply to this dimension under **Dimension attributes**. To select an attribute type, select the **Include** check box next to the attribute type, and type the name for the attribute under **Dimension Attribute**. The default name is the same as **Attribute Type**.

Identify Key Attribute and Changing Dimensions

On the **Specify Dimension Key and Type** page, select the attribute that you want to be the key attribute for the dimension. The key attribute typically corresponds to the primary key column in the main dimension table, and it typically indexes the leaf members of the dimension.

If you selected a template, and a key attribute is defined in the template, that attribute is the default key attribute. If you selected a template but no key attribute is defined in the template, the default is the first attribute in the list. The list contains all the attributes that you selected on the **Specify Dimension Type** page. You can select any one of the attributes that you selected on the **Specify Dimension Type** page to be the key attribute. If you did not select any attributes, the wizard automatically creates a key attribute and names it by concatenating the dimension name and "ID".

Finally, specify whether this dimension is a changing dimension. Members in a changing dimension move over time to different locations in the hierarchy. The wizard generates additional columns and creates attributes that correspond to those columns. These columns will let users query the dimension in such a way as to factor in the change. Any packages that you subsequently create with the Schema Generation Wizard manage surrogate keys based on slowly changing dimension characteristics of the dimension.

When you select the **This is a changing dimension** check box, the Dimension Wizard defines the attributes indicated in the following table:

ATTRIBUTE	TYPE
SCD OriginalID	SCDOriginalID
SCD End Date	SCDEndDate
SCD Start Date	SCDStartDate
SCD Status	SCDStatus

By default, the **This is a changing dimension** check box is selected if you use a template that has these slowly changing dimension attributes defined. If you clear the check box, the slowly changing dimension attributes are removed from the dimension.

You can use Dimension Designer to configure properties for a slowly changing dimension.

Completing the Dimension Wizard

On the **Completing the Wizard** page, type a name for the new dimension and view the dimension structure. Select the **Generate schema now** check box to start the Schema Generation Wizard after you click **Finish**. In most cases, you should not select this check box if you plan to create additional objects. If you do not select this check box, you can use Dimension Designer to generate the schema later.

See Also

Create a Time Dimension by Generating a Time Table
Create a Time Dimension by Generating a Time Table

Define Database Dimensions

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use Dimension Designer in SQL Server Data Tools (SSDT) to configure an existing database dimension in a Microsoft SQL Server Analysis Services project or database. You can use Dimension Designer to do the following:

- Configure the dimension-level properties.
- Add and configure dimension attributes.
- Define and configure user-defined hierarchies.
- Define and configure relationships between attributes.
- Define dimension translations.
- For processed dimensions, you can browse the dimension structure and view data.

After you modify a dimension, attribute, or hierarchy, you must process that dimension to view the changes. When working in project mode, you deploy the changes to the Analysis Services instance before processing.

For more information about how to open a dimension in Dimension Designer, see Modify or Delete a Database Dimension in Solution Explorer.

Dimension Designer has three different tabs, which are described in the following table.

TAB	DESCRIPTION
Dimension Structure	Use this tab to work with the structure of a dimension-to examine or create the data source view schema for a dimension, to work with attributes, and to organize attributes in user-defined hierarchies.
Attribute Relationships	Use this tab to create, modify, or delete the attribute relationships of a dimension.
Translations	Use this tab to add and edit translations of dimension metadata in different languages.
Browser	Use this tab to examine the members of a processed dimension and to review translations of dimension metadata.

The following topics describe the tasks that you can perform in Dimension Designer.

Dimension Attribute Properties Reference
Describes how to define and configure a dimension attribute.

Create User-Defined Hierarchies
Describes how to define and configure a user-defined hierarchy.

Define Attribute Relationships
Describes how to define and configure an attribute relationship.

Use the Business Intelligence Wizard to Enhance Dimensions

Describes how to use the Business Intelligence Wizard to enhance a dimension.

Database Dimensions - Configure the (All) Level for Attribute Hierarchies

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, the (All) level is an optional, system-generated level. It contains only one member whose value is the aggregation of the values of all members in the immediately subordinate level. This member is called the All member. It is a system-generated member that is not contained in the dimension table. Because the member in the (All) level is at the top of the hierarchy, the member's value is the consolidated aggregation of the values of all members in the hierarchy. The All member often serves as the default member of a hierarchy.

The presence of an (All) level in an attribute hierarchy depends on the **IsAggregatable** property setting for the attribute and the presence of an (All) level in a user-defined hierarchy depends on the **IsAggregatable** property of the attribute at the top-most level of user-defined hierarchy. If the **IsAggregatable** property is set to **True**, an (All) level will exist. A hierarchy has no (All) level if the **IsAggregatable** property is set to **False**.

Establishing the Topmost Level

If the **IsAggregatable** property is set to **False** on the source attribute of a level in a hierarchy, then no aggregatable level can appear in the hierarchy above that level. A non-aggregatable level must be the topmost level of any hierarchy or the **IsAggregatable** property of the source attributes for any levels above it must also be set to **False**.

All Member and (All) Level

The single member of the (All) level is called the All member. The **AttributeAllMemberName** property on a dimension specifies the name of the All member for attributes in a dimension. The **AllMemberName** property on a hierarchy specifies the name of the All member for the hierarchy.

See Also

Define a Default Member

Database Dimensions - Modify or Delete a Database Dimension in Solution Explorer

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can modify or delete a dimension by using Dimension Designer in SQL Server Data Tools (SSDT). When you modify a dimension, your changes are not available to users until you process the dimension. For more information, see Processing a multidimensional model (Analysis Services).

To modify a dimension in SQL Server Data Tools

1. In Solution Explorer, right-click the dimension that you want to modify, and then click **Open**.
2. On the **Dimension Structure** tab, in the **Attributes**, **Hierarchies**, or **Data Source View** panes, click the item that you want to modify, and then make the changes.

   For more information about the types of changes that you can make, see Processing a multidimensional model (Analysis Services).

To delete a dimension in SQL Server Data Tools

- In Solution Explorer, right-click the dimension that you want to delete, and then click **Delete**.

See Also

Dimensions in Multidimensional Models

Database Dimensions - Browse Dimension Data in Dimension Designer

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can browse the data in a dimension by using the **Browser** view in Dimension Designer, which is accessed from SQL Server Data Tools (SSDT).

In order for dimension data to be displayed in the **Browser** view, the dimension must be in a processed state. For more information, see Processing a multidimensional model (Analysis Services).

To browse a dimension in SQL Server Data Tools

1. Open the dimension that you want to browse, and click the **Browser** tab.
2. In the **Hierarchy** drop-down list, choose the hierarchy for which you want to browse data.
3. Optionally, if your dimension includes translations, in the **Language** drop-down list, choose the language in which you want to display the data.

> **NOTE**
>
> You must have a translation defined in the dimension for the selected language.

See Also

Introduction to Dimensions (Analysis Services - Multidimensional Data)
Dimensions in Multidimensional Models

Database Dimensions - BI Wizard in Dimension Designer

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, you create custom rollup operators and custom member formulas and configure dimensions for writeback by using the Business Intelligence Wizard or Dimension Designer. You can start the Business Intelligence Wizard from Solution Explorer or from Cube Designer in SQL Server Data Tools (SSDT). Dimension Designer can be accessed from SQL Server Data Tools (SSDT).

To start the Business Intelligence Wizard

- Open the dimension to which you want to add business intelligence, and on the toolbar, click the **Add Business Intelligence** button. Alternatively, you can right-click the dimension in either Object Explorer or Solution Explorer, and then click **Add Business Intelligence**.

Database Dimensions - Create a Date type Dimension

11/28/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, a time dimension is a dimension type whose attributes represent time periods, such as years, semesters, quarters, months, and days. The periods in a time dimension provide time-based levels of granularity for analysis and reporting. The attributes are organized in hierarchies, and the granularity of the time dimension is determined largely by the business and reporting requirements for historical data. For example, most financial and sales data in business intelligence applications use a monthly or quarterly granularity.

Typically, cubes in Analysis Services incorporate a time dimension in one form or another. A cube may include more than one time dimension, or several hierarchies from the same time dimension, depending on the granularity of the data and the reporting requirements. However, not all cubes require a time dimension. Some OLAP applications, such as activity-based costing, do not require a time dimension, because .costing in an activity-based dimension is based on activity instead of time.

Dimension Structure

The dimension structure for a time dimension depends on how the underlying data source stores the time period information. This difference in storage produces two basic types of time dimensions:

Time dimension
Time dimensions are similar to other dimensions in that a dimension table supplies the attributes for the dimension. Each column in the dimension main table defines an attribute for a specific time period.

Like other dimensions, the fact table has a foreign key relationship to the dimension table for the time dimension. The key attribute for a time dimension is based either on an integer key or on the lowest level of detail, such as the date, that appears in the dimension main table.

Server time dimension
If you do not have a dimension table to which to bind time-related attributes, you can have Analysis Services define a server time dimension based on time periods. To define the hierarchies, levels, and members represented by the server time dimension, you select standard time periods when you create the dimension.

Attributes in a server time dimension have a special time-attribute binding. Analysis Services uses the attribute types that are related to dates, such as Year, Month, or Day, to define members of attributes in a time dimension.

After you include the server time dimension in a cube, you set the relationship between the measure group and the server time dimension by specifying a relationship on the **Define Dimension Usage** page of the Cube Wizard.

Calendars

In a time dimension or server time dimension time period attributes are grouped together in hierarchies. Such hierarchies are generally referred to as calendars.

Business intelligence applications frequently require multiple calendar definitions. For example, a human resource department may track employees by using a *standard* calendar-a twelve-month Gregorian calendar starting on January 1 and ending on December 31st. However, that same human resource department may track expenditures by using a *fiscal* calendar-a twelve-month calendar that defines the fiscal year used by the organization.

You can construct these different calendars manually in Dimension Designer. However, the Dimension Wizard provides several hierarchy templates that can be used to automatically generate several types of calendars when you create a time dimension or server time dimension. The following table describes the various calendars that can be generated by the Dimension Wizard.

CALENDAR	DESCRIPTION
Standard calendar	A twelve-month Gregorian calendar starting on January 1 and ending on December 31st. Regardless of whether you use the Dimension Wizard to create a time dimension or a server time dimension, the wizard generates a hierarchy for a standard calendar after you define the attributes that represent the time periods for the dimension. If you use the Dimension Wizard to create a server time dimension, you can adjust the starting date of the standard calendar to start on a day other than January 1.
Fiscal calendar	A twelve-month fiscal calendar. When you select this calendar, specify the starting day and month for the fiscal year used by your organization. Note: This calendar is only available if you use the Dimension Wizard to create a server time dimension.
Reporting calendar (or marketing calendar)	A twelve-month reporting calendar that includes two months of four weeks and one month of five weeks in a repeated three-month (quarterly) pattern. When you select this calendar, specify the starting day and month and the three-month pattern of 4-4-5, 4-5-4, or 5-4-4 weeks, where each digit represents the number of weeks in a month. Note: This calendar is only available if you use the Dimension Wizard to create a server time dimension.
Manufacturing calendar	A calendar that uses 13 periods of four weeks, divided into three quarters of three periods and one quarter of four periods. When you select this calendar, you specify the starting week (between 1 and 4) and month for the manufacturing year used by your organization, and also identify which quarter contains four periods. Note: This calendar is only available if you use the Dimension Wizard to create a server time dimension.
ISO 8601 Calendar	The International Organization for Standardization (ISO) Representation of Dates and Time standard calendar (8601). This calendar has an integral number of 7-day weeks. The new year may start several days before or after the start of the new year, based on the Gregorian calendar. The first week of this calendar is determined by the first week of the Gregorian calendar which contains a Thursday. Therefore, the first day of this week, the Sunday, may actually fall within the previous year. Note: This calendar is only available if you use the Dimension Wizard to create a server time dimension.

When you create a server time dimension, and specify the time periods and the calendars to use in that dimension, the Dimension Wizard adds the attributes for the time periods that are appropriate for each specified calendar. For

example, if you create a server time dimension that uses years as the time period and includes both fiscal and reporting calendars, the wizard will then add FiscalYear and ReportingYears attributes, together with the standard Years attribute, to the dimension. A server time dimension will also have attributes for combinations of selected time periods, such as a DayOfWeek attribute for a dimension that contains Days and Weeks. The Dimension Wizard creates a calendar hierarchy by combining attributes that belong to a single calendar type. For example, a fiscal calendar hierarchy may contain the following levels: Fiscal Year, Fiscal Half Year, Fiscal Quarter, Fiscal Month, and Fiscal Day.

Adding Time Intelligence with the Business Intelligence Wizard

After you have defined a time dimension and added that dimension to a cube, you can use the Business Intelligence Wizard to add time intelligence functionality, such as period-to-date, period-to-period, and rolling average measures. For more information, see Define Time Intelligence Calculations using the Business Intelligence Wizard.

> **NOTE**
>
> You cannot use the Business Intelligence Wizard to add time intelligence to server time dimensions. The Business Intelligence Wizard adds a hierarchy to support time intelligence, and this hierarchy must be bound to a column of the time dimension table. Server time dimensions do not have a corresponding time dimension table and therefore cannot support this additional hierarchy.

See Also

Create a Time Dimension by Generating a Time Table
Business Intelligence Wizard F1 Help
Dimension Types

Database Dimensions - Finance Account of parent-child type

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, an account type dimension is a dimension whose attributes represent a chart of accounts for financial reporting purposes.

An account dimension lets you selectively manage aggregation behavior across accounts over time. An account dimension also lets use a standard mechanism to resolve most of the nonstandard aggregation issues typically encountered in business intelligence solutions that handle financial data. If you did not have such a standard mechanism, resolving these nonstandard aggregation issues would require custom rollup formulas, calculated members, or Multidimensional Expressions (MDX) scripts.

To identify a dimension as an account dimension, set the **Type** property of the dimension to **Accounts**.

Dimension Structure

An account dimension contains, at least, two attributes:

- A key attribute-an attribute that identifies individual accounts in the dimension table for the account dimension.
- An account attribute-a parent attribute that describes how accounts are hierarchically arranged in the account dimension.

  To identify an attribute as an account attribute, set the **Type** property of the attribute to **Account** and the **Usage** property to **Parent**.

Account dimensions can optionally contain the following attributes:

- An account type attribute-an attribute that defines the account type for each account in the dimension. The member names of the account type attribute map to the account types defined for the Analysis Services database or project, and indicate the aggregation function used by Analysis Services for those accounts. You can also use unary operators or custom rollup formulas to determine aggregation behavior for account attributes, but account types let you to easily apply consistent behavior to a chart of accounts without requiring changes to the underlying relational database.

  To identify an account type attribute, set the **Type** property of the attribute to **AccountType**.
- An account name attribute-an attribute that is used for reporting purposes. You identify an account name attribute by setting the **Type** property of the attribute to **AccountName**.
- An account number attribute-an attribute that is used for reporting purposes. You identify an account number attribute by setting the **Type** property of the attribute to **AccountNumber**.

For more information about attribute types, see Configure Attribute Types.

Adding Account Intelligence with the Business Intelligence Wizard

After you have defined an account dimension and added that dimension to a cube, you can use the Business Intelligence Wizard to adding account intelligence functionality, such as identifying and mapping account types, to

the dimension. For more information, see Add Account Intelligence to a Dimension.

See Also

Attributes and Attribute Hierarchies
Business Intelligence Wizard F1 Help
Dimension Types

Database Dimensions - Create a Currency type Dimension

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, a currency type dimension is a dimension whose attributes represent a list of currencies for financial reporting purposes.

A currency dimension lets you add currency conversion capabilities to a cube in Analysis Services. To add currency conversion to a cube, you use the Business Intelligence Wizard define a Multidimensional Expressions (MDX) script command that converts currency measures to values that are appropriate for the locale of the client application. To create this MDX script, the Business Intelligence Wizard needs the following information:

- A currency dimension that represents source currencies. (This dimension is the source currency dimension.)
- A measure group that represents the exchange rates that will be used.

From this information, the Business Intelligence Wizard will design a currency conversion process that identifies the appropriate destination currency dimension (the currency dimension that represents destination currencies). Depending on the number of currency conversions that your business intelligence solution requires, the Business Intelligence Wizard can define multiple destination currency dimensions. For more information about defining currency conversions, see Currency Conversions (Analysis Services).

To identify a dimension as a currency dimension, set the **Type** property of the dimension to **Currency**.

Dimension Structure

A currency dimension contains, at least, a key attribute identifying individual currencies in the dimension table for the currency dimension. The value of the key attribute is different in both source and destination currency dimensions:

- To identify an attribute as the key attribute of a source currency dimension, set the **Type** property of the attribute to **CurrencySource**.
- To identify an attribute as the destination currency dimension, set the **Type** property of the attribute to **CurrencyDestination**.

For reporting purposes, both source and destination currency dimensions can optionally contain the following attributes:

- A currency name attribute.

  To identify an attribute as a currency name attribute, set the **Type** property of the attribute to **CurrencyName**.

- A currency source attribute.

  To identify an attribute as a currency source attribute, set the **Type** property of the attribute to **CurrencySource**.

- A currency International Standards Organization (ISO) code.

  To identify an attribute as a currency ISO code attribute, set the **Type** property of the attribute to

CurrencyIsoCode.

For more information about attribute types, see Configure Attribute Types.

Defining Account Intelligence with the Business Intelligence Wizard

After you have defined an account dimension and added that dimension to a cube, you can use the Business Intelligence Wizard to add account intelligence functionality, such as identifying and mapping account types, to the dimension. For more information, see Add Account Intelligence to a Dimension.

See Also

Attributes and Attribute Hierarchies
Business Intelligence Wizard F1 Help
Dimension Types

Dimension Attribute Properties Reference

5/16/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, there are many properties that determine how dimensions and dimension attributes function. The following table lists and describes each of these attribute properties.

PROPERTY	DESCRIPTION
AttributeHierarchyDisplayFolder	Identifies the folder in which to display the associated attribute hierarchy to end users.
AttributeHierarchyEnabled	Determines whether an attribute hierarchy is generated by Analysis Services for the attribute. If the attribute hierarchy is not enabled, the attribute cannot be used in a user-defined hierarchy and the attribute hierarchy cannot be referenced in Multidimensional Expressions (MDX) statements.
AttributeHierarchyOptimizedState	Determines the level of optimization applied to the attribute hierarchy. By default, an attribute hierarchy is **FullyOptimized**, which means that Analysis Services builds indexes for the attribute hierarchy to improve query performance. The other option, **NotOptimized**, means that no indexes are built for the attribute hierarchy. Using **NotOptimized** is useful if the attribute hierarchy is used for purposes other than querying, because no additional indexes are built for the attribute. Other uses for an attribute hierarchy can be helping to order another attribute.
AttributeHierarchyOrdered	Determines whether the associated attribute hierarchy is ordered. The default value is **True**. However, if an attribute hierarchy will not be used for querying, you can save processing time by changing the value of this property to **False**.
AttributeHierarchyVisible	Determines whether the attribute hierarchy is visible to client applications. The default value is **True**. However, if an attribute hierarchy will not be used for querying, you can save processing time by changing the value of this property to **False**.
CustomRollupColumn	Specifies the column that defines a custom rollup formula.
CustomRollupPropertiesColumn	Specifies the column that contains the properties of a custom rollup formula.
DefaultMember	Specifies a Multidimensional Expressions (MDX) expression that defines the default measure for the attribute.
Description	Contains the description of the attribute.
DiscretizationBucketCount	Contains the number of buckets into which to discretize.

PROPERTY	DESCRIPTION
DiscretizationMethod	Defines the method to use for discretization.
EstimatedCount	Specifies the estimated number of members in the attribute. Until you run the Aggregation Design Wizard, the default value is zero. Either you can allow the wizard to count the number of records or you can enter an estimated value. Enter a value manually if you know the number of members and want to save the time that is required to query the database for the count. If you are working with a test subset of your production data, you can use the counts of your production data so that the aggregation design will be optimized for the production data instead of the test data.
GroupingBehavior	A user defined value that provides a hint to client applications on how to group attributes.
ID	Contains the unique identifier (ID) of the dimension.
InstanceSelection	Provides a hint to client applications about how a list of items should be displayed, based on the expected number of items in the list. The available options are as follows: **None** No hint is provided to the client application. This is the default value. **DropDown** The number of items is small enough to display in a drop-down list. **List** The number of items is too large for a drop-down **list**, but does not require filtering. **FilteredList** The number of items is large enough to require users to filter the items to be displayed. **MandatoryFilter** The number of items is so large that the display must always be filtered.
IsAggregatable	Specifies whether the values of the attribute members can be aggregated. The default value is **True**, which means that the attribute hierarchy contains an (All) level. If the value for this property is **False**, the attribute hierarchy does not contain an (All) level.
KeyColumns	Contains the column or columns that represent the key for the attribute, which is the column in the underlying relational table in the data source view to which the attribute is bound. The value of this column for each member is displayed to users unless a value is specified for the **NameColumn** property.
MemberNamesUnique	Determines whether member names in the attribute hierarchy must be unique.

PROPERTY	DESCRIPTION
MembersWithData	Used by parent attributes to determine whether to display data members for non-leaf members in the parent attribute. This property value is only used when the value of the **Usage** property is set to Parent. This means that a parent-child hierarchy has been defined. The available options are as follows: **NonLeafDataHidden** Non-leaf data is hidden. **NonLeafDataVisible** Non-leaf data is visible.
MembersWithDataCaption	Provides a template string that is used by parent attributes to create captions for system-generated data members in the parent attribute. This property value is only used when the value of the **Usage** property is set to Parent. This means that a parent-child hierarchy has been defined.
Name	Contains the user-friendly name of the attribute.
NameColumn	Identifies the column that provides the name of the attribute that is displayed to users, instead of the value in the key column for the attribute. This column is used when the key column value for an attribute member is cryptic or not otherwise useful to the user, or when the key column is based on a composite key. The **NameColumn** property is not used in parent-child hierarchies; instead, the **NameColumn** property for child members is used as the member names in a parent-child hierarchy.
NamingTemplate	Defines how levels are named in a parent-child hierarchy constructed from the parent attribute. This property value is only used when the value of the **Usage** property is set to Parent. This means that a parent-child hierarchy has been defined.
OrderBy	Describes how to order the members that are contained in the attribute hierarchy. The default value is Name, which specifies that ordering of the attribute members is based on the value of the **NameColumn** property, if any. Otherwise, members are ordered by the value of the key column. The available options are as follows: **NameColumn** Order by the value of the **NameColumn** property. **Key** Order by the value of the key column of the attribute member. **AttributeKey** Order by the value of the member key of a specified attribute, which must have an attribute relationship to the attribute. **AttributeName** Order by the value of the member name of a specified attribute, which must have an attribute relationship to the attribute.
OrderByAttribute	Identifies the attribute by which to order the members of the attribute hierarchy.

PROPERTY	DESCRIPTION
RootMemberIf	Determines how the root or topmost members of a parent-child hierarchy are identified. This property value is only used when the value of the **Usage** property is set to Parent. This means that a parent-child hierarchy has been defined. The default value is **ParentIsBlankSelfOrMissing**, which means that only members that meet one or more of the conditions described for **ParentIsBlank**, **ParentIsSelf**, or **ParentIsMissing** are treated as root members. The following values are also available: **ParentIsBlank** Only members with a null, a zero, or an empty string in the key column or columns are treated as root members. **ParentIsSelf** Only members with themselves as parents are treated as root members. **ParentIsMissing** Only members with parents that cannot be found are treated as root members.
Type	Contains the type of the attribute. For more information, see Configure Attribute Types.
UnaryOperatorColumn	Specifies the column that provides unary operators. It is a binding of DataItem type that defines the details of a column providing a unary operator.
Usage	Describes how an attribute is used. The available options are as follows: **Regular** The attribute is a regular attribute. This is the default value. **Key** The attribute is a key attribute. **Parent** The attribute is a parent attribute.
ValueColumn	Identifies the column that provides the value of the attribute. If the **NameColumn** element of the attribute is specified, the same **DataItem** values are used as default values for the **ValueColumn** element. If the **NameColumn** element of the attribute is not specified and the **KeyColumns** collection of the attribute contains a single **KeyColumn** element representing a key column with a string data type, the same **DataItem** values are used as default values for the **ValueColumn** element.

NOTE

For more information about how to set values for the **KeyColumn** property when you are working with null values and other data integrity issues, see Handling Data Integrity Issues in Analysis Services 2005.

NOTE

The default member on an attribute is used to evaluate expressions when a member from the hierarchy is not explicitly included in a query. The default member for an attribute is specified by the **DefaultMember** property on the attribute. Whenever a hierarchy from a dimension is included in a query, all the default members from attributes corresponding to levels in the hierarchy are ignored. If no hierarchy of a dimension is included in a query, then default members are used for all the attributes in the dimension. For more information on default members, see Define a Default Member.

See Also

Attributes and Attribute Hierarchies

Attribute Properties - Rename an Attribute

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The name of an attribute in Microsoft SQL Server Analysis Services is defined by its **Name** property. To rename an attribute, right-click the attribute in the **Attributes** pane of the **Dimension Structure** tab of Dimension Designer in SQL Server Data Tools (SSDT), and then click **Rename**. If the **Attributes** pane is in grid view, click the attribute's name and edit it directly in the grid. You can also select an attribute and set its **Name** property in the **Properties** window.

For more information about how to rename an attribute, see Bind an attribute to a Key column.

See Also

Dimension Attribute Properties Reference

Attribute Properties - Add an Attribute to a Dimension

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can add an attribute to a dimension either automatically or manually in Microsoft SQL Server Analysis Services.

To create an attribute automatically, on the **Dimension Structure** tab of Dimension Designer in SQL Server Data Tools (SSDT), select the column that you want to map to an attribute, and then drag that column from the **Data Source View** pane to the **Attributes** pane. This creates an attribute that is mapped to the column, and assigns the same name to the attribute as the name of the column. If an attribute that uses that name already exists, Analysis Services adds an ordinal number suffix, starting with "1" for the first duplicate name.

To create an attribute manually, set the **Attributes** pane to grid view. In the name column of the last row in the grid, click the **<new attribute>** item. Type a name for the new attribute. This creates an attribute that is not mapped to a column and that has default settings for all its properties. You can set the mapping in the **Properties** window of SQL Server Data Tools by configuring the **KeyColumns** property for the new attribute.

For more information, see Define a New Attribute Automatically, Bind an Attribute to a Name Column, and Modify the KeyColumn Property of an Attribute.

See Also

Dimension Attribute Properties Reference

Attribute Properties - Remove an Attribute from a Dimension

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

To remove an attribute from a dimension in Microsoft SQL Server Analysis Services, right-click the attribute in the **Attributes** pane of the **Dimension Structure** tab of Dimension Designer in SQL Server Data Tools (SSDT), and then click **Delete**. In the **Delete Objects** dialog box, click **OK**. This removes the attribute from the dimension, but does not affect the data source view for the dimension.

See Also

Dimensions in Multidimensional Models
Dimension Attribute Properties Reference

Attribute Properties - Configure Attribute Types

5/16/2018 • 12 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, attribute types help classify an attribute in terms of business functionality. There are many attribute types, most of which are used by client applications to display or support an attribute. However, some attribute types also have specific meaning to Analysis Services. For example, some attribute types identify attributes that represent time periods in various calendars for time dimensions.

Setting Attribute Types

The value of the **Type** property for an attribute determines the attribute type for that attribute. Several wizards in Analysis Services set attribute types when defining dimensions or attributes. These Analysis Services wizards also set attribute types when the wizards add functionality to dimensions. For example, the Business Intelligence Wizard applies several attribute types to attributes in a dimension when the wizard adds account intelligence to identify attributes that contain the names, codes, numbers, and structure of accounts in the dimension. The Business Intelligence Wizard also consumes attribute types, such as for currency conversion. For more information, see Create a Currency type Dimension.

The following tables list the attribute types available in Analysis Services. These tables separate attribute types into the following categories:

TERM	DEFINITION
General attribute types	These values are available to all attributes, and exist only to enable classification of attributes for client application purposes.
Account dimension attribute types	These values identify an attribute that belongs to an account dimension. For more information about account dimension, see Create a Finance Account of parent-child type Dimension.
Currency dimension attribute type	These values identify an attribute that belongs to a currency dimension. For more information about currency dimensions, see Create a Currency type Dimension.
Slowly changing dimension attributes	These values identify an attribute that belongs to a slowly changing dimension.
Time dimension attributes	These values identify an attribute that belongs to a time dimension. For more information about time dimensions, see Create a Date type Dimension.

General Attribute Types

ATTRIBUTE TYPE VALUE	DESCRIPTION
Address	Represents an address.
AddressBuilding	Represents a building identifier for an address.

ATTRIBUTE TYPE VALUE	DESCRIPTION
AddressCity	Represents a city for an address.
AddressCountry	Represents a country or region for an address.
AddressFax	Represents a fax telephone number.
AddressFloor	Represents a floor identifier for an address.
AddressHouse	Represents a house number for an address.
AddressPhone	Represents a telephone number.
AddressQuarter	Represents a quarter for an address.
AddressRoom	Represents a room identifier for an address.
AddressStateOrProvince	Represents a state or province for an address.
AddressStreet	Represents the street for an address.
AddressZip	Represents a ZIP Code or Postal Code for an address.
BomResource	Represents a resource for a bill of materials (BOM).
Caption	Represents a caption.
CaptionAbbreviation	Represents an abbreviation.
CaptionDescription	Represents a description.
Channel	Represents a channel.
City	Represents a city.
Company	Represents a company.
Continent	Represents a continent.
Country	Represents a country or region.
County	Represents a county.
CustomerGroup	Represents a group of customers.
CustomerHousehold	Represents a household of customers.
Customers	Represents a customer.
DateCanceled	Represents a cancellation date.

ATTRIBUTE TYPE VALUE	DESCRIPTION
DateDuration	Represents a duration.
DateEnded	Represents an end date.
DateModified	Represents a modification date.
DateStart	Represents a start date.
DeletedFlag	Indicates whether a member is or should be deleted (in terms of business functionality.) Note: Analysis Services does not use this attribute type to determine whether a member should be deleted. Instead, this attribute type is intended to be used by client applications for display purposes only.
FormattingColor	Represents the color used in formatting.
FormattingFont	Represents the font used in formatting.
FormattingFontEffects	Represents the font effects used in formatting.
FormattingFontSize	Represents the font size used in formatting.
FormattingOrder	Represents the order used in formatting.
FormattingSubtotal	Represents a subtotal.
GeoBoundaryBottom	Represents the bottommost value of a geographic boundary.
GeoBoundaryFront	Represents the value at the front of a geographic boundary.
GeoBoundaryLeft	Represents the leftmost value of a geographic boundary.
GeoBoundaryPolygon	Represents the polygon definition of a geographic boundary.
GeoBoundaryRear	Represents the rearmost value of a geographic boundary.
GeoBoundaryRight	Represents the rightmost value of a geographic boundary.
GeoBoundaryTop	Represents the topmost value of a geographic boundary.
GeoCentroidX	Represents an X-axis centroid for a geographic region.
GeoCentroidY	Represents a Y-axis centroid for a geographic region.
GeoCentroidZ	Represents a Z-axis centroid for a geographic region.
ID	Represents an identifier (ID) or key.
Image	Represents an image in an undefined graphic format.

ATTRIBUTE TYPE VALUE	DESCRIPTION
ImageBmp	Represents an image in bitmap graphic format.
ImageGif	Represents an image in Graphics Interchange Format (GIF) graphic format.
ImageJpg	Represents an image in Joint Photographic Experts Group (JPEG) graphic format.
ImagePng	Represents an image in Portable Network Graphics (PNG) graphic format.
ImageTiff	Represents an image in Tagged Image File Format (TIFF) graphic format.
OrganizationalUnit	Represents an organizational unit.
OrgTitle	Represents an organizational title.
PercentOwnership	Represents a percent of ownership.
PercentVoteRight	Represents a percent of voting rights.
Person	Represents a person.
PersonContact	Represents contact information for a person.
PersonDemographic	Represents demographic information for a person.
PersonFirstName	Represents the first name of a person.
PersonFullName	Represents the full name of a person.
PersonLastName	Represents the surname (last name) of a person.
PersonMiddleName	Represents the middle name of a person.
PhysicalColor	Represents a color.
PhysicalDensity	Represents density.
PhysicalDepth	Represents depth.
PhysicalHeight	Represents height.
PhysicalSize	Represents a size.
PhysicalVolume	Represents volume.
PhysicalWeight	Represents weight.
PhysicalWidth	Represents width.

ATTRIBUTE TYPE VALUE	DESCRIPTION
Point	Represents a point.
PostalCode	Represents a postal code.
Product	Represents a product.
ProductBrand	Represents a product brand.
ProductCategory	Represents a product category.
ProductGroup	Represents a product group.
ProductSKU	Represents a product stock keeping unit (SKU).
Project	Represents a project.
ProjectCode	Represents a project code.
ProjectCompletion	Represents the completion status of a project.
ProjectEndDate	Represents a project end date.
ProjectName	Represents a project name.
ProjectStartDate	Represents a project start date.
Promotion	Represents a promotion.
QtyRangeHigh	Represents the highest value of a range of quantities.
QtyRangeLow	Represents the lowest value of a range of quantities.
Quantitative	Represents a quantitative attribute.
Rate	Represents a rate.
RateType	Represents a rate type.
Region	Represents a customer-defined region.
Regular	Represents a regular attribute.
RelationToParent	Represents a relation to a parent.
Representative	Represents a representative.
Scenario	Represents a scenario.
Sequence	Represents a sequence attribute.

ATTRIBUTE TYPE VALUE	DESCRIPTION
ShortCaption	Represents a short caption.
StateOrProvince	Represents a state or province.
Utility	Represents a utility.
Version	Represents a version.
WebHtml	Represents HTML content.
WebMailAlias	Represents an e-mail alias.
WebUrl	Represents a URL address.
WebXmlOrXsl	Represents XML or XSL content.

Account Dimension Attribute Types

ATTRIBUTE TYPE VALUE	DESCRIPTION
Account	Represents the parent of an account. This attribute type is typically applied to the parent attribute of an account dimension.
AccountName	Represents the name of an account. This attribute type is typically applied to the key attributes of an account dimension.
AccountNumber	Represents the number of an account.
AccountType	Represents the type of an account. This attribute type identifies the aggregation function of an account member in an account type dimension in the Analysis Services database.

Currency Dimension Attribute Types

ATTRIBUTE TYPE VALUE	DESCRIPTION
CurrencyDestination	Represents the destination currency of a currency exchange. This attribute type is typically applied to the key attribute of a reporting dimension, for use in currency conversion. For more information about currency conversion, see Currency Conversions (Analysis Services).
CurrencyIsoCode	Represents the International Standards Organization (ISO) code of a currency. For more information about currency conversion, see Currency Conversions (Analysis Services).
CurrencyName	Represents the name of a currency. For more information about currency conversion, see Currency Conversions (Analysis Services).

ATTRIBUTE TYPE VALUE	DESCRIPTION
CurrencySource	Represents the source currency of a currency exchange. This attribute type is typically applied to the key attribute of a currency dimension, for use in currency conversion. For more information about currency conversion, see Currency Conversions (Analysis Services).

Slowly Changing Dimension Attribute Types

ATTRIBUTE TYPE VALUE	DESCRIPTION
ScdEndDate	Represents the effective end date for a member in a slowly changing dimension.
ScdOriginalID	Represents the original identifier for a member in a slowly changing dimension.
ScdStartDate	Represents the effective start date for a member in a slowly changing dimension.
ScdStatus	Represents the effective status of a member in a slowly changing dimension.

Time Dimension Attribute Types

ATTRIBUTE TYPE VALUE	DESCRIPTION
Date	Represents a date. This attribute type is typically applied to the key attribute of a time dimension or server time dimension.
DayOfHalfYear	Represents the day ordinal of a half-year.
DayOfMonth	Represents the day ordinal of a month.
DayOfQuarter	Represents the day ordinal of a quarter.
DayOfTenDays	Represents the day ordinal of a ten-day period.
DayOfTrimester	Represents the day ordinal of a trimester.
DayOfWeek	Represents the day ordinal of a week.
DayOfYear	Represents the day ordinal of a year.
Days	Represents days.
FiscalDate	Represents a date in a fiscal calendar.
FiscalDayOfHalfYear	Represents the day ordinal of a half-year in a fiscal calendar.
FiscalDayOfMonth	Represents the day ordinal of a month in a fiscal calendar.

ATTRIBUTE TYPE VALUE	DESCRIPTION
FiscalDayOfQuarter	Represents the day ordinal of a quarter in a fiscal calendar.
FiscalDayOfTrimester	Represents the day ordinal of a trimester in a fiscal calendar.
FiscalDayOfWeek	Represents the day ordinal of a week in a fiscal calendar.
FiscalDayOfYear	Represents the day ordinal of a year in a fiscal calendar.
FiscalHalfYears	Represents half-years in a fiscal calendar.
FiscalHalfYearOfYear	Represents the half-year ordinal of a year in a fiscal calendar.
FiscalMonths	Represents months in a fiscal calendar.
FiscalMonthOfHalfYear	Represents the month ordinal of a half-year in a fiscal calendar.
FiscalMonthOfQuarter	Represents the month ordinal of a quarter in a fiscal calendar.
FiscalMonthOfTrimester	Represents the month ordinal of a trimester in a fiscal calendar.
FiscalMonthOfYear	Represents the month ordinal of a year in a fiscal calendar.
FiscalQuarters	Represents quarters in a fiscal calendar.
FiscalQuarterOfHalfYear	Represents the quarter ordinal of a half-year in a fiscal calendar.
FiscalQuarterOfYear	Represents the quarter ordinal of a year in a fiscal calendar.
FiscalTrimesters	Represents trimesters in a fiscal calendar.
FiscalTrimesterOfYear	Represents the trimester ordinal of a year in a fiscal calendar.
FiscalWeeks	Represents weeks in a fiscal calendar.
FiscalWeekOfHalfYear	Represents the week ordinal of a half-year in a fiscal calendar.
FiscalWeekOfMonth	Represents the week ordinal of a month in a fiscal calendar.
FiscalWeekOfQuarter	Represents the week ordinal of a quarter in a fiscal calendar.
FiscalWeekOfTrimester	Represents the week ordinal of a trimester in a fiscal calendar.
FiscalWeekOfYear	Represents the week ordinal of a year in a fiscal calendar.
FiscalYears	Represents years in a fiscal calendar.
HalfYears	Represents half-years.

ATTRIBUTE TYPE VALUE	DESCRIPTION
HalfYearOfYear	Represents the half-year ordinal of a year.
Hours	Represents hours.
IsHoliday	Indicates whether a date is a holiday.
ISO8601Date	Represents a date in an ISO 8601 calendar.
ISO8601DayOfWeek	Represents the day ordinal of a week in an ISO 8601 calendar.
ISO8601DayOfYear	Represents the day ordinal of a year in an ISO 8601 calendar.
ISO8601Weeks	Represents weeks in an ISO 8601 calendar.
ISO8601WeekOfYear	Represents the week ordinal of a year in an ISO 8601 calendar.
ISO8601Years	Represents years in an ISO 8601 calendar.
IsPeakDay	Indicates whether a date is a peak day.
IsWeekDay	Indicates whether a date is a weekday.
IsWorkingDay	Indicates whether a date is a working day.
ManufacturingDate	Represents a date in a manufacturing calendar.
ManufacturingDayOfHalfYear	Represents the day ordinal of a half-year in a manufacturing calendar.
ManufacturingDayOfMonth	Represents the day ordinal of a month in a manufacturing calendar.
ManufacturingDayOfQuarter	Represents the day ordinal of a quarter in a manufacturing calendar.
ManufacturingDayOfTrimester	Represents the day ordinal of a trimester in a manufacturing calendar.
ManufacturingDayOfWeek	Represents the day ordinal of a week in a manufacturing calendar.
ManufacturingDayOfYear	Represents the day ordinal of a year in a manufacturing calendar.
ManufacturingHalfYears	Represents half-years in a manufacturing calendar.
ManufacturingHalfYearOfYear	Represents the half-year ordinal of a year in a manufacturing calendar.
ManufacturingMonths	Represents months in a manufacturing calendar.

ATTRIBUTE TYPE VALUE	DESCRIPTION
ManufacturingMonthOfHalfYear	Represents the month ordinal of a half-year in a manufacturing calendar.
ManufacturingMonthOfQuarter	Represents the month ordinal of a quarter in a manufacturing calendar.
ManufacturingMonthOfTrimester	Represents the month ordinal of a trimester in a manufacturing calendar.
ManufacturingMonthOfYear	Represents the month ordinal of a year in a manufacturing calendar.
ManufacturingQuarters	Represents quarters in a manufacturing calendar.
ManufacturingQuarterOfHalfYear	Represents the quarter ordinal of a half-year in a manufacturing calendar.
ManufacturingQuarterOfYear	Represents the quarter ordinal of a year in a manufacturing calendar.
ManufacturingWeeks	Represents weeks in a manufacturing calendar.
ManufacturingWeekOfHalfYear	Represents the week ordinal of a half-year in a manufacturing calendar.
ManufacturingWeekOfMonth	Represents the week ordinal of a month in a manufacturing calendar.
ManufacturingWeekOfQuarter	Represents the week ordinal of a quarter in a manufacturing calendar.
ManufacturingWeekOfTrimester	Represents the week ordinal of a trimester in a manufacturing calendar.
ManufacturingWeekOfYear	Represents the week ordinal of a year in a manufacturing calendar.
ManufacturingYears	Represents years in a manufacturing calendar.
Minutes	Represents minutes.
Months	Represents months.
MonthOfHalfYear	Represents the month ordinal of a half-year.
MonthOfQuarter	Represents the month ordinal of a quarter.
MonthOfTrimester	Represents the month ordinal of a trimester.
MonthOfYear	Represents the month ordinal of a year.
Quarters	Represents quarters.

ATTRIBUTE TYPE VALUE	DESCRIPTION
QuarterOfHalfYear	Represents the quarter ordinal of a half-year.
QuarterOfYear	Represents the quarter ordinal of a year.
ReportingDate	Represents a date in a reporting calendar.
ReportingDayOfHalfYear	Represents the day ordinal of a half-year in a reporting calendar.
ReportingDayOfMonth	Represents the day ordinal of a month in a reporting calendar.
ReportingDayOfQuarter	Represents the day ordinal of a quarter in a reporting calendar.
ReportingDayOfTrimester	Represents the day ordinal of a trimester in a reporting calendar.
ReportingDayOfWeek	Represents the day ordinal of a week in a reporting calendar.
ReportingDayOfYear	Represents the day ordinal of a year in a reporting calendar.
ReportingHalfYears	Represents half-years in a reporting calendar.
ReportingHalfYearOfYear	Represents the half-year ordinal of a year in a reporting calendar.
ReportingMonths	Represents months in a reporting calendar.
ReportingMonthOfHalfYear	Represents the month ordinal of a half-year in a reporting calendar.
ReportingMonthOfQuarter	Represents the month ordinal of a quarter in a reporting calendar.
ReportingMonthOfTrimester	Represents the month ordinal of a trimester in a reporting calendar.
ReportingMonthOfYear	Represents the month ordinal of a year in a reporting calendar.
ReportingQuarters	Represents quarters in a reporting calendar.
ReportingQuarterOfHalfYear	Represents the quarter ordinal of a half-year in a reporting calendar.
ReportingQuarterOfYear	Represents the quarter ordinal of a year in a reporting calendar.
ReportingTrimesters	Represents trimesters in a reporting calendar.
ReportingTrimesterOfYear	Represents the trimester ordinal of a year in a reporting calendar.

ATTRIBUTE TYPE VALUE	DESCRIPTION
ReportingWeeks	Represents weeks in a reporting calendar.
ReportingWeekOfHalfYear	Represents the week ordinal of a half-year in a reporting calendar.
ReportingWeekOfMonth	Represents the week ordinal of a month in a reporting calendar.
ReportingWeekOfQuarter	Represents the week ordinal of a quarter in a reporting calendar.
ReportingWeekOfTrimester	Represents the week ordinal of a trimester in a reporting calendar.
ReportingWeekOfYear	Represents the week ordinal of a year in a reporting calendar.
ReportingYears	Represents years in a reporting calendar.
Seconds	Represents seconds.
TenDayOfHalfYear	Represents the ten-day period ordinal of a half-year.
TenDayOfMonth	Represents the ten-day period ordinal of a month.
TenDayOfQuarter	Represents the ten-day period ordinal of a quarter.
TenDayOfTrimester	Represents the ten-day period ordinal of a trimester.
TenDayOfYear	Represents the ten-day period ordinal of a year.
TenDays	Represents ten-day periods.
Trimesters	Represents trimesters.
TrimesterOfYear	Represents the trimester ordinal of a year.
UndefinedTime	Represents an undefined time period.
WeekOfYear	Represents the week ordinal of a year.
Weeks	Represents weeks.
WinterSummerSeason	Indicates whether the date is part of the winter/summer season.
Years	Represents years.

See Also

Attributes and Attribute Hierarchies
Dimension Attribute Properties Reference

Attribute Properties - Group Attribute Members

11/15/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A member group is a system-generated collection of consecutive dimension members. In Microsoft SQL Server Analysis Services, members of an attribute can be grouped into a number of member groups through a process called discretization. A level in a hierarchy contains either member groups or members, but not both. When business users browse a level that contains member groups, they see the names and cell values of the member groups. The members generated by Analysis Services to support member groups are called grouping members, and look like ordinary members.

The **DiscretizationMethod** property on an attribute controls how the members are grouped.

DISCRETIZATIONMETHOD SETTING	DESCRIPTION
None	Displays the members.
Automatic	Selects the method that best represents the data: either the **EqualAreas** method or the **Clusters** method.
EqualAreas	Tries to divide the members in the attribute into groups that contain an equal number of members.
Clusters	Tries to divide the members in the attribute into groups by sampling the training data, initializing to a number of random points, and running several iterations of the Expectation-Maximization (EM) clustering algorithm. This method is useful because it works on any distribution curve, but is more expensive in terms of processing time.

The **DiscretizationNumber** property on attributes specifies the number of groups to display. If the property is set to the default value of 0, Analysis Services determines the number of groups by either sampling or reading the data, depending on the setting of the **DiscretizationMethod** property.

The sort order of members in the member groups is controlled by using the **OrderBy** property of the attribute. Based on this sort order, the members in a member group are ordered consecutively.

A common use for member groups is to drill down from a level with few members to a level with many members. To enable users to drill down between levels, change the **DiscretizationMethod** property on the attribute for the level that contains numerous members from **None** to one of the discretization methods described in the previous table. For example, a Client dimension contains a Client Name attribute hierarchy with 500,000 members. You can rename this attribute Client Groups and set the **DiscretizationMethod** property to **Automatic** to display member groups on the attribute hierarchy member level.

To drill down to individual clients in each group, you can create another Client Name attribute hierarchy bound to the same table column. Then, create a new user hierarchy based on the two attributes. The top level would be based on the Client Groups attribute and the lower level would be based on the Client Name attribute. The **IsAggregatable** property would be **True** on both attributes. The user could then expand the (All) level on the hierarchy to view the group members, and expand the group members to view the leaf members of the hierarchy. To hide either group or client level, you could set the **AttributeHierarchyVisible** property to **False** on the corresponding attribute.

Naming Template

Member group names are generated automatically when the member groups are created. Unless you specify a naming template, the default naming template is used. You can change this method of naming by specifying a naming template in the **Format** option for the **NameColumn** property of an attribute. Different naming templates can be redefined for every language specified in the **Translations** collection of the column binding that was used for the **NameColumn** property of the attribute.

The **Format** setting uses the following string expression to define the naming template:

```
<Naming template> ::= <First definition> [;<Intermediate definition>;<Last definition>]
```

```
<First definition> ::= <Name expression>
```

```
<Intermediate definition> ::= <Name expression>
```

```
<Last definition> ::= <Name expression>
```

The `<First definition>` parameter applies only to the first or only member group generated by the discretization method. If the optional parameters `<Intermediate definition>` and `<Last definition>` are not provided, the `<First definition>` parameter is used for all measure groups generated for that attribute.

The `<Last definition>` parameter applies only to the last member group generated by the discretization method.

The `<Intermediate bucket name>` parameter applies to every member group other than the first or last member group generated by the discretization method. If two or fewer member groups are generated, this parameter is ignored.

The `<Bucket name>` parameter is a string expression that can incorporate a set of variables to represent member or member group information as part of the name of the member group:

VARIABLE	DESCRIPTION
%{First bucket member}	The member name of the first member to be included in the current member group.
%{Last bucket member}	The member name of the last member to be included in the current member group.
%{Previous bucket last member}	The member name of the last member assigned to the previous member group.
%{Next bucket first member}	The member name of the first member to be assigned to the next member group.
%{Bucket Min}	The minimum value of the members to be assigned to the current member group.
%{Bucket Max}	The maximum value of the members to be assigned to the current member group.
%{Previous Bucket Max}	The maximum value of the members to be assigned to the previous member group.
%{Next Bucket Min}	The minimum value of the members to be assigned to the next member group.

The default naming template is `"%{First bucket member} - %{Last bucket member}"`, to provide compatibility with earlier versions of Analysis Services.

> **NOTE**
> To include a semicolon (;) as a literal character in the naming template, prefix it with the percent (%) character.

Example

The following string expression could be used to classify the Yearly Income attribute of the Customer dimension in the Adventure Works DW Multidimensional 2012 sample Analysis Services database, where the Yearly Income attribute uses member grouping:

"Less than %{Next Bucket Min};Between %{First bucket member} and %{Last bucket member};Greater than %{Previous Bucket Max}"

Adding New Members to Existing Member Groups

If new members are added to the dimension, they are assigned to the appropriate member groups by comparing the value of the member against the current member group layout.

If a member is inserted between the last member of the previous member group and the first member of the next member group, the new member becomes the last member of the previous member group.

Updating a Dimension with Discretized Attributes

When you process a dimension, a discretized attribute is rediscretized only with a full update (ProcessFull). To rediscretize an attribute, you must perform a full update of the dimension. If the dimension table of a discretized attribute is updated and you process the dimension with an incremental update (ProcessAdd), the discretized attribute is not rediscretized. The names and children of the new buckets remain the same. For more information about processing dimensions, see Processing Analysis Services Objects.

Usage Limitations

- You cannot create member groups in the topmost or bottommost level of a hierarchy. However, if you need to do this, you can add a level in such a way that the level in which you want to create member groups is no longer the top or bottom level. You can hide the added level by setting its **Visible** property to **False**.
- You cannot create member groups in two consecutive levels of a hierarchy.
- Member groups are not supported for dimensions that use the ROLAP storage mode.
- If the dimension table of a dimension that contains member groups is updated, and the dimension is subsequently fully processed, a new set of member groups is generated. The names and children of the new member groups may be different from the old member groups.

See Also

Attributes and Attribute Hierarchies

Attribute Properties - Define Custom Member Formulas

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can define a Multidimensional Expressions (MDX) expression, called a custom member formula, to supply the values for the members of a specified attribute. A column in a table from a data source view provides, for each member in an attribute, the expression used to supply the value for that member.

Custom member formulas determine the cell values that are associated with members and override the aggregate functions of measures. Custom member formulas are written in MDX. Each custom member formula applies to a single member. Custom member formulas are stored in the dimension table or in another table that has a foreign key relationship with the dimension table.

The **CustomRollupColumn** property on an attribute specifies the column that contains custom member formulas for members of the attribute. If a row in the column is empty, the cell value for the member is returned normally. If the formula in the column is not valid, a run-time error occurs whenever a cell value that uses the member is retrieved.

Before you can specify custom member formulas for an attribute, make sure that the dimension table that contains the attribute, or a directly related table, has a string column to store the custom member formulas. If this is the case, you can either set the **CustomRollupColumn** property on an attribute manually or use the Set Custom Member Formula enhancement of the Business Intelligence Wizard to enable a custom member formula on an attribute. For more information about how to use this enhancement, see Set Custom Member Formulas for Attributes in a Dimension.

Evaluating Custom Member Formulas

Custom member formulas differ from calculated members. Custom member formulas apply to members that exist in dimension tables, and only provide the value of the member. In contrast, calculated members are not stored in dimension tables, and calculated member expressions define both data and metadata for additional members included in a dimension or hierarchy.

Custom member formulas override the aggregate functions associated with measures. For example, before a custom member formula is specified, a measure using the **Sum** aggregate function has the following values for the following members of the Time dimension:

- 2003: 2100
  - Quarter 1: 700
  - Quarter 2: 500
  - Quarter 3: 100
  - Quarter 4: 800
- 2004: 1500
  - Quarter 1: 600
  - Quarter 2: 200

- Quarter 3: 300
- Quarter 4: 400

With a custom member formula, the value of the member is instead supplied by the custom rollup formula. For example, the following custom member formula can be used to supply the value for the Quarter 4 child member of the 2004 member in the Time dimension as 450.

```
Time.[Quarter 3] * 1.5
```

Custom member formulas are stored in a column of the dimension table. You enable custom rollup formulas by setting the **CustomRollupColumn** property on an attribute.

To apply a single MDX expression to all members of an attribute, create a named calculation on the dimension table that returns an MDX expression as a literal string. Then, specify the named calculation with the **CustomRollupColumn** property setting on the attribute that you want to configure. A named calculation is a column in a data source view table that returns row values defined by a SQL expression. For more information about constructing named calculations, see Define Named Calculations in a Data Source View (Analysis Services)

> **NOTE**
> To apply an MDX expression to members of a particular level instead of to members of all levels based on a particular attribute, you can define the expression as an MDX script on the level. For more information, see MDX Scripting Fundamentals (Analysis Services).

If you use both calculated members and custom rollup formulas for members of an attribute, you should be aware of the order of evaluation. Calculated members are resolved before custom rollup formulas are resolved.

See Also

Attributes and Attribute Hierarchies
Set Custom Member Formulas for Attributes in a Dimension

Attribute Properties - Define a Default Member

11/28/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The default member of an attribute hierarchy is used to evaluate expressions when an attribute hierarchy is not included in a query. The default member is ignored whenever a query includes an attribute hierarchy or user hierarchy that contains the attribute that sources the attribute hierarchy. This is because the member specified in the query is used.

The default member for an attribute hierarchy is set by specifying an attribute member as the **DefaultMember** property value for the attribute hierarchy. You can set this property on the Dimension Structure tab in Dimension Designer, or in the cube's calculation script on the Calculation tab in Cube Designer in SQL Server Data Tools (SSDT). You can also specify the **DefaultMember** property for a security role (overriding the default member set on the dimension) on the Dimension Data tab when defining dimension security. To avoid name resolution problems, define the default member in the cube's MDX script in the following situations: if the cube refers to a database dimension more than once, if the dimension in the cube has a different name than the dimension in the database, or if you want to have different default members in different cubes.

The default member on an attribute is used to evaluate expressions when an attribute is not included in a query. The default member for an attribute is specified by the **DefaultMember** property on the attribute. Whenever a hierarchy from a dimension is included in a query, all the default members from attributes corresponding to levels in the hierarchy are ignored. If no hierarchy of a dimension is included in a query, default members are used for all the attributes in the dimension.

Resolving the Default Member When No Default Member is Specified

If no default member is specified for an attribute hierarchy, and the attribute hierarchy is aggregatable (the **IsAggregatable** property on the attribute is set to **True**), the (All) member is the default member. If no default member is specified and the attribute hierarchy is not aggregatable (the **IsAggregatable** property on the attribute is set to **False**), a default member is selected from the attribute hierarchy's top level.

Specifying the Default Member

Every attribute in a dimension in Microsoft SQL Server Analysis Services has a default member, which you can specify by using the **DefaultMember** property for an attribute. This setting is used to evaluate expressions if an attribute is not included in a query. If a query specifies a hierarchy in a dimension, the default members for the attributes in the hierarchy are ignored. If a query does not specify a hierarchy in a dimension, the **DefaultMember** settings for dimension attributes take effect.

If the **DefaultMember** setting for an attribute is blank and its **IsAggregatable** property is set to **True**, the default member is the All member. If the **IsAggregatable** property is set to **False**, the default member is the first member of the first visible level.

The **DefaultMember** setting for an attribute applies to every hierarchy in which the attribute participates. You cannot use different settings for different hierarchies in a dimension. For example, if the [1998] member is the default member for a [Year] attribute, this setting applies to every hierarchy in the dimension. The **DefaultMember** setting in this case cannot be [1998] in one hierarchy and [1997] in a different hierarchy.

If you define a default member for a particular level in a hierarchy that does not aggregate naturally, you must define default members in all levels above that level in the hierarchy. For example, in the hierarchy All-Countries-

Climate, you cannot define a default member for Climate unless you define a default member for Countries. Failing to do so creates query-time errors.

When levels in a hierarchy aggregate naturally, you can define a default member for any attribute in the hierarchy without regard to other attributes in the hierarchy. For example, in the hierarchy Country-Province-City, you can define a default member for City such as [City].[Montreal] without defining the default member for State or for Country.

See Also

Configure the (All) Level for Attribute Hierarchies

Attribute Properties - View Attributes in Dimension Designer

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Attributes are created on dimension objects. You can view and configure attributes by using Dimension Designer in Microsoft SQL Server Analysis Services. The **Attributes** pane of the **Dimension Structure** tab of Dimension Designer lists the attributes that are in a dimension. Use this pane to add, remove, or configure attributes. You can also select attributes to use as a level in a new hierarchy or to add as a level in an existing hierarchy.

To view the attributes in a dimension, open Dimension Designer for the dimension. The **Attributes** pane of the **Dimension Structure** tab of the designer shows the attributes that are in the dimension. You can switch between a list, tree, or grid view by pointing to **Show Attributes in** on the **Dimension** menu of SQL Server Data Tools (SSDT) and then clicking one of the following commands:

1. Show Attributes in a **List**. Displays the attributes in list format. Right-click an attribute to delete it from the list, to rename the attribute, or to change the usage of the attribute. Use this view for building hierarchies. Attribute information and member properties are not visible.

2. Show Attributes in a **Tree**. Display the attributes in tree format, with the dimension as the top-level node in the tree. Expand an attribute to view attribute relationships for it or to create a new attribute relationship, by performing the following actions:

   - Click the dimension, an attribute, or a member property to view its properties in the **Properties** window.

   - Right-click an attribute or a member property to delete it from the list, to rename it, or to change its usage.

   Use this view for viewing and creating member properties. You can also use this view to build hierarchies.

3. Show Attributes in a **Grid**. Display the attributes in grid format. The grid displays the following columns:

   - **Name** shows the value of the **Name** property. Type a different name to change the setting.

   - **Usage** specifies whether this is a Regular, Key, Parent, or AccountType attribute. Click a value in this column to select a different setting.

   - **Type** specifies the business intelligence category for the attribute. Click this cell to select a different setting.

   - **Key Column** shows the OLE DB data type for the **KeyColumn** property on the attribute. This column cannot be changed.

   - **Name Column** indicates whether the **NameColumn** property setting on the attribute is the same column as the setting for the **KeyColumn** property. This column cannot be changed.

   Click any row in the grid to view properties for that attribute.

   Use this view for creating and configuring attributes.

In SQL Server Data Tools, the icons shown in the following table mark attributes according to their usage.

ICON	ATTRIBUTE USAGE
	Regular or AccountType
	Key
	Parent

See Also

Dimension Attribute Properties Reference

Attribute Properties - Define a New Attribute Automatically

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can create a new attribute in a dimension by using drag-and-drop editing in the Dimension Designer.

To create a new attribute automatically

1. In Dimension Designer, open the dimension in which you want to create the attribute.
2. On the **Dimension Structure** tab, in the **Data Source View** pane, in the table to which you want to bind the attribute, select the column, and then drag the column to the **Attributes** pane.

   Analysis Services creates the new attribute that has the same name as the column to which it is bound. If there are multiple attributes that use the same column, Analysis Services appends a number to the attribute name.

See Also

Dimensions in Multidimensional Models
Dimension Attribute Properties Reference

View Attributes in Dimension Designer

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This procedure describes how to change the way that attributes are displayed in the Dimension Designer.

To change the format of the Attributes pane in Dimension Designer

1. In Dimension Designer, open the dimension that contains the attributes you want to work with, and click the Dimension Structure tab.
2. Click anywhere in the **Attributes** pane to set the focus to the **Attributes**pane.
3. Click the **Dimension**menu, or, right-click the **Attributes** pane.
4. Point to **Show Attributes In**, and then click **Tree**, **Grid**, or **List**.

Attribute Properties - Modify the KeyColumn Property

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can modify the **KeyColumns** property of an attribute. For example, you might want to specify a composite key instead of a single key as the key for the attribute.

To modify the KeyColumns property of an attribute

1. In SQL Server Data Tools (SSDT), open the project in which you want to modify the **KeyColumns** property.
2. Open Dimension Designer by doing one of the following procedures:
   - In **Solution Explorer**, right-click the dimension in the **Dimensions** folder, and then either click **Open** or **View Designer**.

     -or-
   - In Cube Designer, on the **Cube Structure** tab, expand the cube dimension in the **Dimensions** pane and click **Edit <dimension>**.
3. On the **Dimension Structure** tab, in the **Attributes** pane, click the attribute whose **KeyColumns** property you want to modify.
4. In the **Properties** window, click the value for the **KeyColumns** property.
5. Click the browse **(...)** button that appears in the value cell of the property box.

   The **Key Columns** dialog box opens.
6. To remove an existing key column, in the **Key Columns** list, select the column, and then click the **<** button.
7. To add a key column, in the **Available Columns** list, select the column, and then click the **>** button.

   > **NOTE**
   > If you define multiple key columns, the order in which those columns appear in the **Key Columns** list affects the display order. For example, a month attribute has two key columns: month and year. If the year column appears in the list before the month column, Analysis Services will sort by year and then by month. If the month column appears before the year column, Analysis Services will sort by month and then by year.

8. To change the order of key columns, select a column, and then click the **Up** or **Down** button.

See Also

Dimension Attribute Properties Reference

Attribute Properties - Bind an Attribute to a Key Column

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This procedure describes how to change the setting for the **Name** property of an attribute in a dimension.

To bind an attribute to a key column

1. In Dimension Designer, open the dimension that contains the attribute you want to rename.
2. In the **Attributes** pane of the Dimension Builder view, change the display format to either **Tree** or **List**. For more information about how to change the format of the Attributes pane, see View Attributes in a Tree, List or Grid in Dimension Designer.
3. Right-click the attribute you want to configure, and then click **Rename**.
4. Type the new name.

> **NOTE**
>
> You can also set the **Name** property of a selected attribute in the **Properties** window of SQL Server Data Tools (SSDT).

Attribute Properties - Bind an Attribute to a Name Column

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This procedure describes how to bind an attribute to a name column manually by using the **Attributes** pane in the Dimension Designer and by using the **Object Binding** dialog box.

To bind an attribute to a name column

1. In Dimension Designer, open the dimension in which you want to create the attribute.
2. On the **Dimension Structure** tab, in the **Attributes** pane, right-click the attribute that you want to configure, and then click **Properties**.
3. In the **Properties** window, locate the **NameColumn** property, and then select **(new)**.
4. In the **Object Binding** dialog box, for **Binding type**, select **Column binding**.
5. In the **Source column** list, select the column to which the attribute will be bound, and then click **OK**.

Attribute Properties - Set Usage Property

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can set the usage for an attribute by using the **Dimension Structure** view in Dimension Designer, which is accessed from SQL Server Data Tools (SSDT).

When you set usage for an attribute, your changes do not take effect until you process the dimension. For more information, see Processing a multidimensional model (Analysis Services).

To set usage for an attribute

1. In Solution Explorer, right-click a dimension, and click **Open**.

   The **Dimension Structure** view opens by default.

2. In **Attributes**, right-click the attribute for which you which you want to set usage, point to **Set Attribute Usage**, and then click one of the following options:

   - **Regular**
   - **Key**
   - **Parent**

See Also

Attributes and Attribute Hierarchies
Add an Attribute to a Dimension

Attribute Properties - Define Member Groups

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If an attribute has a large number of members, you can choose to group those members into buckets, reducing the number of members that users see when they browse the data in a hierarchy. You can also determine the number of buckets in which the members are groups and set a naming scheme for the buckets. For more information, see Group Attribute Members (Discretization).

You group members by setting the **DiscretizationMethod** property, which is accessed through the **Properties** window in SQL Server Data Tools (SSDT).

When you create member groups, your changes are not available to users until you process the dimension. For more information, see Processing a multidimensional model (Analysis Services).

To create member groups

1. Open the dimension that you want to work with. The **Dimension Structure** tab opens by default.
2. In **Attributes**, right-click the attribute whose members you want to group, and then click **Properties**.
3. From the drop-down list next to **DiscretizationMethod**, select a method by which to group the members. For more information about **DiscretizationMethod** settings, see Group Attribute Members (Discretization).

Parent-Child Dimension

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A parent-child hierarchy is a hierarchy in a standard dimension that contains a parent attribute. A parent attribute describes a *self-referencing relationship*, or *self-join*, within a dimension main table. Parent-child hierarchies are constructed from a single parent attribute. Only one level is assigned to a parent-child hierarchy, because the levels present in the hierarchy are drawn from the parent-child relationships between members associated with the parent attribute. The position of a member in a parent-child hierarchy is determined by the **KeyColumns** and **RootMemberIf** properties of the parent attribute, whereas the position of a member in a level is determined by the **OrderBy** property of the parent attribute. For more information about attribute properties, see Attributes and Attribute Hierarchies.

Because of parent-child relationships between levels in a parent-child hierarchy, some nonleaf members can also have data derived from underlying data sources, in addition to data aggregated from child members.

Dimension Schema

The dimension schema of a parent-child hierarchy depends on a self-referencing relationship present on the dimension main table. For example, the following diagram illustrates the **DimOrganization** dimension main table in the **AdventureWorksDW2012** sample database.

In this dimension table, the **ParentOrganizationKey** column has a foreign key relationship with the **OrganizationKey** primary key column. In other words, each record in this table can be related through a parent-child relationship with another record in the table. This kind of self-join is generally used to represent organization entity data, such as the management structure of employees in a department.

Hierarchies and Levels

Dimensions that do not have a parent-child relationship construct hierarchies by grouping and ordering attributes. These dimensions derive the level names for their hierarchies from the attribute names.

However, parent-child dimensions construct parent-child hierarchies by examining the data that the dimension main table contains, and then evaluating the parent-child relationships between the records in the table. For more information about parent-child hierarchies, see User Hierarchies.

Parent-child hierarchies do not derive the names for the levels in a parent-child hierarchy from the attributes that are used to create the hierarchy. Instead, these dimensions create level names automatically by using a naming template-a string expression you can specify at the level of the parent attribute that controls how the attribute generates the attribute hierarchy. For more information about how to set the naming template for a parent attribute, see Attributes and Attribute Hierarchies.

Data Members

Typically, leaf members in a dimension contain data derived directly from underlying data sources, whereas nonleaf members contain data derived from aggregations performed on child members.

However, parent-child hierarchies might have some nonleaf members whose data is derived from underlying data sources, in addition to data aggregated from child members. For these nonleaf members in a parent-child hierarchy, special system-generated child members can be created that contain the underlying fact table data. Referred to as *data members*, these special child members contain a value that is directly associated with a nonleaf member and is independent of the summary value calculated from the descendants of the nonleaf member. For more information about data members, see Attributes in Parent-Child Hierarchies.

See Also

Attributes in Parent-Child Hierarchies
Database Dimension Properties

Parent-Child Dimension Attributes

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, a general assumption is usually made about the content of members in a dimension. Leaf members contain data derived directly from underlying data sources; nonleaf members contain data derived from aggregations performed on child members.

In a parent-child hierarchy, however, some nonleaf members may also have data derived from underlying data sources, in addition to data aggregated from child members. For these nonleaf members in a parent-child hierarchy, special system-generated child members are created that contain the underlying fact table data. Referred to as *data members*, they contain a value directly associated with a nonleaf member that is independent of the summary value calculated from the descendants of the nonleaf member.

Data members are available only to dimensions with parent-child hierarchies, and are visible only if allowed by the parent attribute. You can use Dimension Designer to control the visibility of data members. To expose data members, set the **MembersWithData** property for the parent attribute to **NonLeafDataVisible.** To hide data members contained by the parent attribute, set the **MembersWithData** property on the parent attribute to **NonLeafDataHidden**.

This setting does not override the normal aggregation behavior for nonleaf members; the data member is always included as a child member for the purposes of aggregation. However, a custom rollup formula can be used to override the normal aggregation behavior. The Multidimensional Expressions (MDX) DataMember function gives you the ability to access the value of the associated data member regardless of the value of the **MembersWithData** property.

The **MembersWithDataCaption** property of the parent attribute provides Analysis Services with the naming template used to generate member names for data members.

Using Data Members

Data members are useful when aggregating measures along organizational dimensions that have parent-child hierarchies. For example, the following diagram shows a dimension that has three levels, representing the gross sales volume of products. The first level shows the gross sales volume for all salespersons. The second level contains the gross sales volume for all sales staff grouped by sales manager, and the third level contains the gross sales volume for all sales staff grouped by salesperson.

Ordinarily, the value of the Sales Manager 1 member would be derived by aggregating the values of the Salesperson 1 and Salesperson 2 members. However, because Sales Manager 1 also can sell products, that member may also contain data derived from the fact table, because there may be gross sales associated with Sales Manager 1.

Moreover, the individual commissions for each sales staff member can vary. In this case, two different scales are used to compute commissions for the individual gross sales of the sales managers, as opposed to the total of gross sales generated by their salespersons. Therefore, it is important to be able to access the underlying fact

table data for nonleaf members. The MDX **DataMember** function can be used to retrieve the individual gross sales volume of the Sales Manager 1 member, and a custom rollup expression can be used to exclude the data member from the aggregated value of the Sales Manager 1 member, providing the gross sales volume of the salespersons associated with that member.

See Also

Dimension Attribute Properties Reference
Parent-Child Dimensions

Parent-Child Dimension Attributes - Custom Rollup Operators

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Custom rollup operators provide a simple way to control how member values are rolled up into parent values in a parent-child hierarchy. In a dimension containing a parent-child relationship, you specify a column that contains unary operators that specify rollup for all noncalculated members of the parent attribute. The unary operator is applied to members whenever the values of the parent members are evaluated.

The unary operators are stored in column defined by the **UnaryOperatorColumn** property of the parent attribute, and they are applied to each member of the attribute. The column specified by this property can reside either in the dimension table or in a table related to the dimension table by a foreign key in the dimension table.

Custom rollup operators provide functionality similar to, but more simplified than, custom member formulas. A custom member formula uses Multidimensional Expressions (MDX) expressions to determine how the members are rolled up. In contrast, a custom rollup operator uses a simple unary operator to determine how the value of a member affects the parent. Custom member formulas of the preceding level in a dimension override a level's custom rollup operator.

Custom Rollup Precedence

In terms of precedence, the custom rollup operators of the source attribute for a level in a hierarchy override the custom member formulas of the previous level. However, the custom member formulas of the preceding level override the custom rollup operators of a level.

See Also

Define Custom Member Formulas
Unary Operators in Parent-Child Dimensions

Parent-Child Dimension Attributes - Unary Operators

11/28/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In a dimension that contains a parent-child relationship in Microsoft SQL Server Analysis Services, you specify a unary (or custom rollup) operator column that determines the custom rollup for all noncalculated members of the parent attribute. The unary operator is applied to members whenever the values of the parent members are evaluated. The **UnaryOperatorColumn** on a parent attribute (**Usage**=Parent) specifies the column of a table in the data source view that contains unary operators. Values for the custom rollup operators that are stored in this column are applied to each member of the attribute.

You can create and specify a named calculation on a dimension table in the data source view as a unary operator column. The simplest expression, such as '+', returns the same operator for all members. But you can use any expression as long as it returns an operator for every member.

You can change the **UnaryOperatorColumn** property setting manually on a parent attribute or use the Define Custom Aggregation enhancement of the Business Intelligence Wizard to replace the default aggregation that is associated with members of a dimension. For more information about how to use the Business Intelligence Wizard to perform this configuration, see Add a Custom Aggregation to a Dimension.

The default setting for the **UnaryOperatorColumn** property on a parent attribute is (none), which disables the custom rollup operators. The following table lists the unary operators and describes how they behave when they are applied to a level.

UNARY OPERATOR	DESCRIPTION
+ (plus sign)	The value of the member is added to the aggregate value of the sibling members that occur before the member. This operator is the default operator if no unary operator column is defined for an attribute.
- (minus sign)	The value of the member is subtracted from the aggregate value of the sibling members that occur before the member.
* (asterisk)	The value of the member is multiplied by the aggregate value of the sibling members that occur before the member.
/ (slash mark)	The value of the member is divided by the aggregate value of the sibling members that occur before the member.
~ (tilde)	The value of the member is ignored.

Blank values and any other values not found in the table are treated the same as the plus sign (+) unary operator. There is no operator precedence, so the order of members as stored in the unary operator column determines the order of evaluation. To change the order of evaluation, create a new attribute, set its **Type** property to **Sequence**, and then assign sequence numbers that correspond to the order of evaluation in its **Source Column** property. You must also order members of the attribute by that attribute. For information about how to use the Business Intelligence Wizard to order members of an attribute, see Define the Ordering for a Dimension.

You can use the **UnaryOperatorColumn** property to specify a named calculation that returns a unary operator as a literal character for all members of the attribute. This could be as simple as typing a literal character such as `'*'`

in the named calculation. This would replace the default operator, the plus sign (+), with the multiplication operator, the asterisk (*), for all members of the attribute. For more information, see Define Named Calculations in a Data Source View (Analysis Services).

In the **Browser** tab of Dimension Designer, you can view the unary operators next to each member in a hierarchy. You can also change the unary operators when you work with a write-enabled dimension. If the dimension is not write-enabled, you must use a tool to modify the data source directly.

See Also

Dimension Attribute Properties Reference
Custom Rollup Operators in Parent-Child Dimensions
Start the Business Intelligence Wizard in Dimension Designer

User-Defined Hierarchies - Create

12/6/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

SQL Server Analysis Services lets you create user-defined hierarchies. A hierarchy is a collection of levels based on attributes. For example, a time hierarchy might contain the Year, Quarter, Month, Week, and Day levels. In some hierarchies, each member attribute uniquely implies the member attribute above it. This is sometimes referred to as a natural hierarchy. A hierarchy can be used by end users to browse cube data. Define hierarchies by using the Hierarchies pane of Dimension Designer in SQL Server Data Tools (SSDT).

When you create a user-defined hierarchy, the hierarchy might become *ragged*. A ragged hierarchy is where the logical parent member of at least one member is not in the level immediately above the member. If you have a ragged hierarchy, there are settings that control whether the missing members are visible and how to display the missing members. For more information, see Ragged Hierarchies.

See Also

User Hierarchy Properties
Level Properties - user hierarchies
Parent-Child Dimensions

User-Defined Hierarchies - Ragged Hierarchies

11/28/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A ragged hierarchy is a user-defined hierarchy that has an uneven number of levels. Common examples include an organizational chart where a high-level manager has both departmental managers and non-managers as direct reports, or geographic hierarchies composed of Country-Region-City, where some cities lack a parent State or Province, such as Washington D.C., Vatican City, or New Delhi.

For most hierarchies in a dimension, each level has the same number of members above it as any other member at the same level. A ragged hierarchy is different in that the logical parent of at least one member is not in the level immediately above the member. When this occurs, the hierarchy descends to different levels for different drilldown paths. In a client application, this can make drilldown paths unnecessarily complicated.

Client applications vary in how well they handle a ragged hierarchy. If ragged hierarchies exist in your model, be prepared to do a little extra work to get the rendering behavior you expect.

As a first step, check the client application to see how it handles the drilldown path. For example, Excel repeats the parent names as placeholders for missing values. To see this behavior yourself, build a PivotTable using the Sales Territory dimension in the Adventure Works multidimensional model. In a PivotTable having the Sales Territory attributes Group, Country, and Region, you will see that countries missing a region value will get a placeholder, in this case a repeat of the parent above it (Country name). This behavior derives from the MDX Compatibility=1 connection string property that is fixed within Excel. If the client does not naturally provide the drill-down behaviors you are looking for, you can set properties in the model to change at least some of those behaviors.

This topic contains the following sections:

- Approaches for modifying drilldown navigation in a ragged hierarchy
- Set HideMemberIf to hide members in a regular hierarchy
- Set MDX Compatibility to determine how placeholders are represented in client applications

Approaches for modifying drilldown navigation in a ragged hierarchy

The presence of a ragged hierarchy becomes a problem when drilldown navigation does not return expected values or is perceived as awkward to use. To fix navigation problems that result from ragged hierarchies, consider these options:

- Use a regular hierarchy but set the **HideMemberIf** property on each level to specify whether a missing level is visualized to the user. When setting **HideMemberIf**, you should also set **MDXCompatibility** on the connection string to override default navigation behaviors. Instructions for setting these properties are in this topic.
- Create a parent-child hierarchy that explicitly manages the level members. For an illustration of the technique, see Ragged Hierarchy in SSAS (blog post). For more information in Books Online, see Parent-Child Dimensions. Downsides to creating a parent-child hierarchy are that you can only have one per dimension, and you typically incur a performance penalty when calculating aggregations for intermediate members.

If your dimension contains more than one ragged hierarchy, you should use the first approach, setting **HideMemberIf**. BI Developers with practical experience in working with ragged hierarchies go further in

advocating for additional changes in the physical data tables, creating separate tables for each level. See Martin Mason's the SSAS Financial Cube-Part 1a-Ragged Hierarchies (blog) for details about this technique.

Set HideMemberIf to hide members in a regular hierarchy

In a ragged dimension's table, the logically missing members can be represented in different ways. The table cells can contain nulls or empty strings, or they can contain the same value as their parent to serve as a placeholder. The representation of placeholders is determined by the placeholder status of child members, as determined by the **HideMemberIf** property, and the **MDX Compatibility** connection string property for the client application.

For client applications that support the display of ragged hierarchies, you can use these properties to hide logically missing members.

1. In SSDT, double-click a dimension to open it in Dimension Designer. The first tab, Dimension Structure, shows attribute hierarchies in the Hierarchies pane.
2. Right-click a member within the hierarchy and select **Properties**. Set **HideMemberIf** to one of the values described below.

HIDEMEMBERIF SETTING	DESCRIPTION
Never	Level members are never hidden. This is the default value.
OnlyChildWithNoName	A level member is hidden when it is the only child of its parent and its name is null or an empty string.
OnlyChildWithParentName	A level member is hidden when it is the only child of its parent and its name is the same as the name of its parent.
NoName	A level member is hidden when its name is empty.
ParentName	A level member is hidden when its name is identical to that of its parent.

Set MDX Compatibility to determine how placeholders are represented in client applications

After setting **HideMemberIf** on a hierarchy level, you should also set the **MDX Compatibility** property in the connection string sent from the client application. The **MDX Compatibility** setting determines whether the **HideMemberIf** is used.

MDX COMPATIBILITY SETTING	DESCRIPTION	USAGE
1	Show a placeholder value.	This is the default used by Excel, SSDT, and SSMS. It instructs the server to return placeholder values when drilling down empty levels in a ragged hierarchy. If you click the placeholder value, you can continue down to get to the child (leaf) nodes. Excel owns the connection string used to connect to Analysis Services, and it always sets **MDX Compatibility** to 1 on each new connection. This behavior preserves backward compatibility.

MDX COMPATIBILITY SETTING	DESCRIPTION	USAGE
2	Hide a placeholder value (either a null value or a duplicate of the parent level), but show other levels and nodes having relevant values.	**MDX Compatibility**=2 is typically viewed as the preferred setting in terms of ragged hierarchies. A Reporting Services report and some third-party client applications can persist this setting.

See Also

Create User-Defined Hierarchies
User Hierarchies
Parent-Child Dimensions
Connection String Properties (Analysis Services)

User-Defined Hierarchies - Add or Delete a User-Defined Hierarchy

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You add a user-defined hierarchy to or remove a user-defined hierarchy from a dimension on the **Dimension Structure** tab in Dimension Designer in SQL Server Data Tools (SSDT).

When you add a user-defined hierarchy, it is not available to users until it is instantiated in an Analysis Services instance and the dimension is processed. For more information, see Multidimensional Model Databases and Processing a multidimensional model (Analysis Services).

To add a user-defined hierarchy to a dimension

1. In SQL Server Data Tools (SSDT), open the appropriate Analysis Services project, and then open the dimension with which you want to work.

   The dimension opens in Dimension Designer on the **Dimension Structure** tab.

2. From the **Attributes** pane, drag an attribute that you want to use in the user-defined hierarchy to the **Hierarchies** pane.

3. Drag additional attributes to form levels in the user-defined hierarchy.

   The order in which attributes are listed in the hierarchy is important. For example, in a time hierarchy, years should appear higher in the hierarchy list than days.

4. Optionally, rearrange the levels in the user-defined hierarchy by dragging them to the correct positions.

5. Optionally, modify properties of the user-defined hierarchy or its levels.

   For example, you might want to specify a name for the user-defined hierarchy, rename one or more of its levels, and define a custom name for the All level. For more information, see User Hierarchy Properties, and Level Properties - user hierarchies.

   > **NOTE**
   >
   > By default, a user-defined hierarchy is just a path that enables users to drill down for information. However, if relationships exist between levels, you can increase query performance by configuring attribute relationships between levels. For more information, see Attribute Relationships and Define Attribute Relationships.

To remove a user-defined hierarchy from a dimension

- On the **Dimension Structure** tab, click the user-defined hierarchy that you want to remove in the **Hierarchies** pane. On the toolbar, click **Delete**.
  - or -
- Right-click the user-defined hierarchy that you want to remove in the **Hierarchies** pane and then click **Delete**.
  - or -
- Drag the user-defined hierarchy off of the design surface.

See Also

User Hierarchies
Create User-Defined Hierarchies

Attribute Relationships - Define

12/6/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, attributes are the fundamental building block of a dimension. A dimension contains a set of attributes that are organized based on attribute relationships.

For each table included in a dimension, there is an attribute relationship that relates the table's key attribute to other attributes from that table. You create this relationship when you create the dimension.

An attribute relationship provides the following advantages:

- Reduces the amount of memory needed for dimension processing. This speeds up dimension, partition, and query processing.
- Increases query performance because storage access is faster and execution plans are better optimized.
- Results in the selection of more effective aggregates by the aggregation design algorithms, provided that user-defined hierarchies have been defined along the relationship paths.

Attribute Relationship Considerations

When the underlying data supports it, you should also define unique attribute relationships between attributes. To define unique attribute relationships, use the **Attribute Relationships** tab of Dimension Designer.

Any attribute that has an outgoing relationship must have a unique key relative to its related attribute. In other words, a member in a source attribute must identify one and only one member in a related attribute. For example, consider the relationship, City -> State. In this relationship, the source attribute is City and the related attribute is State. The source attribute is the "many" side and the related side is the "one" side of the many-to-one relationship. The key for the source attribute would be City + State. For more information, see Create, Modify, or Delete an Attribute Relationship.

For more information about properties of an attribute relationship, see Configure Attribute Relationship Properties.

> **NOTE**
>
> Defining attribute relationships incorrectly can cause invalid query results.

See Also

Attribute Relationships

Attribute Relationships - Configure Attribute Properties

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The following table lists and describes the properties of an attribute relationship.

PROPERTY	DESCRIPTION
Attribute	Contains the name of the attribute.
Cardinality	Indicates the cardinality of the relationship. Values are Many, for a many to one relationship, or One, for a one to one relationship. Default value is Many. In Microsoft SQL Server Analysis Services, the cardinality property has no effect - its use is reserved for a future implementation.
Name	Contains the friendly name of the attribute.
RelationshipType	Indicates whether member relationships change over time. Values are Rigid, which means that relationships between members do not change over time, or Flexible, which means that relationships between members change over time. Default is Flexible. If you define a relationship as flexible, aggregations are dropped and recomputed as part of an incremental update (they will not be dropped if only new members are added). If you define a relationship as rigid, Analysis Services retains aggregations when the dimension is incrementally updated. If a relationship that is defined as rigid actually changes, Analysis Services generates an error during incremental processing. Specifying the appropriate relationships and relationship properties increases query and processing performance.
Visible	Determines the visibility of the attribute relationship. Values are True or False. Default is True.

Attribute Relationships - Create, Modify, or Delete Relationship

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can create, modify, or delete an attribute relationship between attributes in a dimension by using the **Attribute Relationships** tab of Dimension Designer in SQL Server Data Tools (SSDT).

To create an Attribute Relationship

1. In Dimension Designer, open the dimension that contains the attributes that you want to create an attribute relationship between.
2. On the **Attribute Relationships** tab, right-click an attribute in the diagram or in the **Attributes** pane, and then select **New Attribute Relationship**.

   > **NOTE**
   > To display the **Attributes** pane, click **Show List Views** on the toolbar.

3. In the **Create Attribute Relationship** dialog box, select a source attribute and a related attribute.
4. In the **Relationship type** list, select a relationship type, and then click **OK**.

To modify an Attribute Relationship

1. In Dimension Designer, open the dimension that contains the attribute relationship that you want to modify.
2. Click the **Attribute Relationships** tab.
3. Right-click the attribute relationship in the diagram or in the **Attribute Relationships** pane, and select **Edit Attribute Relationship**.

   > **NOTE**
   > To display the **Attribute Relationships** pane, click **Show List Views** on the toolbar.

4. In the **Edit Attribute Relationship** dialog box, select a source attribute and a related attribute.
5. In the **Relationship type** list, select a relationship type, and then click **OK**.

To delete an Attribute Relationship

1. In Dimension Designer, open the dimension that contains the attribute relationship that you want to delete.
2. On the **Attribute Relationships** tab, right-click the attribute relationship in the diagram or in the **Attribute Relationships** pane, and then select **Delete**.
3. In the **Delete Objects** dialog box, click **OK**.

See Also

Attribute Relationships

Attribute Relationships - Define the Relationship Type

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You define the relationship type of an attribute relationship by using the **Attribute Relationships** tab in Dimension Designer, which can be accessed from SQL Server Data Tools (SSDT).

To set the relationship type of an attribute relationship

1. In Dimension Designer, open the dimension with which you want to work, and then click on the **Attribute Relationships** tab.

2. In the diagram or in the **Attribute Relationships** pane, right-click the attribute relationship, click **Relationship Type**, and then click either **Flexible** or **Rigid**.

   > **NOTE**
   >
   > To display the **Attribute Relationships** pane, click **Show List Views** on the toolbar.

   In a flexible relationship, relationships between members change over time. In a rigid relationship, relationships between members do not change over time.

See Also

Define Attribute Relationships

Attribute Relationships - Arrange Shapes in the Diagram

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can change the layout of the shapes on the **Attribute Relationships** tab of the Dimension Structure view in Dimension Designer in SQL Server Data Tools (SSDT).

To arrange shapes in the attribute relationship diagram

1. In Dimension Designer, open the dimension that contains the attributes that you want to create an attribute relationship between.

2. On the **Attribute Relationships** tab, in the diagram, click a shape to select it, and then grab the shape by the top edge and drag it to the new location.

   You can also click **Arrange Shapes** to automatically arrange all shapes according to the layout algorithm that Dimension Designer uses.

See Also

Attribute Relationships
Define Attribute Relationships

BI Wizard - Add Account Intelligence to a Dimension

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Add the account intelligence enhancement to a cube or a dimension to assign standard account classifications, such as income and expense, to members of an account attribute. This enhancement also identifies account types (such as Asset and Liability) and assigns the appropriate aggregation to each account type. Microsoft SQL Server Analysis Services can use the classifications to aggregate accounts over time.

> **NOTE**
> Account intelligence is available only for dimensions that are based on existing data sources. For dimensions that were created without using a data source, you must run the Schema Generation Wizard to create a data source view before adding account intelligence.

You apply account intelligence to a dimension that specifies account information (for example, account name, account number, and account type). To add account intelligence, you use the Business Intelligence Wizard, and select the **Define account intelligence** option on the **Choose Enhancement** page. This wizard then guides you through the steps of selecting a dimension to which you want to apply account intelligence, identifying account attributes in the selected account dimension, and then mapping account types in the dimension table to account types recognized by Analysis Services.

Selecting a Dimension

On the first **Define Account Intelligence** page of the wizard, you specify the dimension to which you want to apply account intelligence. The account intelligence enhancement added to this selected dimension will result in changes to the dimension. These changes will be inherited by all cubes that include the selected dimension.

Specifying Account Attributes

On the **Configure Dimension Attributes** page of the wizard, you specify the account attributes in the selected account dimension. First, in the **Include** column, select the check box next to each of the account attribute types that you want to map to a dimension attribute in the dimension. Then, in the **Dimension Attribute** column, expand the drop-down list, and select the attribute in the dimension that corresponds to the selected attribute type. Selecting the attribute from the list sets the attribute **Type** property for the account attributes.

Mapping Account Types

The second **Define Account Intelligence** page maps account type values from the dimension table to account types recognized by Analysis Services. This page appears only if you included the **Account Type** dimension attribute in the dimension. To include the **Account Type** dimension, on the **Define Account Intelligence Settings** page of the wizard, select the check box next to **Account Type**, and then select appropriate attribute.

On the second **Define Account Intelligence** page, there are two columns:

- The **Source Table Account Types** column lists the account types that the wizard obtains from the dimension table.
- The **Server Account Types** column identifies the corresponding account type that Analysis Services recognizes. The following table lists the account types that Analysis Services recognizes and the default

aggregation for each of these types. The selections are made automatically if the dimension table uses the same account type names as Analysis Services uses.

SERVER ACCOUNT TYPE	AGGREGATION	DESCRIPTION
Statistical	**None**	A calculated ratio of something, or a count of something that does not aggregate over time. This type of account does not convert across currencies with conversion rules.
Liability	**LastNonEmpty**	The money or value of things owed at a specific time. This account type does not accumulate over time, and therefore, does not aggregate naturally over time. For example, the Year amount is the value of the last month with data. This type of account converts across currencies with the End of Period rate.
Asset	**LastNonEmpty**	The money or value of things held at a specific time. This account type accumulates over time, and therefore does not aggregate naturally over time. For example, the Year amount is the value of the last month with data. This type of account converts across currencies with the End of Period rate.
Balance	**LastNonEmpty**	The count of something at a specified time. This account type accumulates but does not aggregate naturally over time. For example, the Year amount is the value of the last month with data.
Flow	**Sum**	An incremental count of something. This account type aggregates as a **Sum** over time but does not convert with currency conversion rules.
Expense	**Sum**	The money or value of things spent. This account type aggregates as a **Sum** over time and converts across currencies with an average rate.
Income	**Sum**	The money or value of things received. This account type aggregates as a **Sum** over time and converts across currencies with an average rate.

> **NOTE**
>
> If appropriate, you can map more than one account type in the dimension to a particular server account type.

To change the default aggregations mapped to each account type for a database, you can use the Database

Designer.

1. In Solution Explorer, right-click the Analysis Services project and click **Edit Database**.
2. In the **Account Type Mapping** box, select an account type in the **Name**.
3. In the **Alias** text box, type an alias for this account type.
4. In the **Aggregation Function** drop down list box, change the default Aggregation Function for this account type.

BI Wizard - Add Dimension Intelligence to a Dimension

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Add the dimension intelligence enhancement to a cube or a dimension to specify a standard business type for a dimension. This enhancement also specifies the corresponding types for dimension attributes. Client applications can use these type specifications when analyzing data.

To add dimension intelligence, you use the Business Intelligence Wizard, and select the **Define dimension intelligence** option on the **Choose Enhancement** page. This wizard then guides you through the steps of selecting a dimension to which you want to apply dimension intelligence and identifying the attributes for the selected dimension.

Selecting a Dimension

On the first **Set Dimension Intelligence Options** page of the wizard, you specify the dimension to which you want to apply dimension intelligence. The dimension intelligence enhancement added to this selected dimension will result in changes to the dimension. These changes will be inherited by all cubes that include the selected dimension.

> **NOTE**
>
> If you select **Account** as the dimension, you will be specifying account intelligence for the dimension. For more information, see Add Account Intelligence to a Dimension.

Specifying Dimension Attributes

On the **Define Dimension Intelligence** page, in **Dimension Type** list, the selection that you make sets the dimension's **Type** property. The **Type** property setting provides information to servers and client applications about the contents of a dimension. Some settings only provide guidance for client applications; these settings are optional. Other settings, such as Accounts or Time, determine specific behaviors and may be required to implement particular business intelligence enhancements. For example, the SQL Server Management Studio uses the dimension type to identify a Currency dimension and set the appropriate currency conversion rules. The default setting for the **Dimension Type** is **Regular**, which makes no assumptions about the contents of the dimension.

After selecting the dimension type, in **Dimension attributes**, in the **Include** column, select the check box next to each standard attribute type for which there is a corresponding attribute in the dimension. Finally, in the **Dimension Attribute** column, expand the drop-down list, and select the attribute in the dimension that corresponds to the selected attribute type. Selecting the attribute from the list sets the attribute **Type** property for the attributes.

For example, you want to add dimension intelligence to an Accounts dimension. In **Dimension Type**, you select **Accounts**. Then, if the dimension has **Account Type** and **Account Description** attributes, in the **Include** column, you select the check boxes for the **Account Name** and **Account Type** account types. In the **Dimension Attribute** column, you then associate these account types with the **Account Description** and **Account Type** attributes, respectively, in the dimension.

See Also

Define Time Intelligence Calculations using the Business Intelligence Wizard

BI Wizard - Add a Custom Aggregation to a Dimension

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Add a custom aggregation enhancement to a cube or dimension to replace the default aggregations that are associated with a dimension member with a different unary operator. This enhancement specifies a unary operator column in the dimension table that defines rollup for members in a parent-child hierarchy. The unary operator acts on the parent attribute in a parent-child hierarchy.

> **NOTE**
> A custom aggregation is available only for dimensions that are based on existing data sources. For dimensions that were created without using a data source, you must run the Schema Generation Wizard to create a data source view before adding the custom aggregation.

To add a custom aggregation, you use the Business Intelligence Wizard, and select the **Specify a unary operator** option on the **Choose Enhancement** page. This wizard then guides you through the steps of selecting a dimension to which you want to apply a custom aggregation and identifying the custom aggregation.

> **NOTE**
> Before you run the Business Intelligence Wizard to add a custom aggregation, make sure that the dimension that you want to enhance contains a parent-child attribute hierarchy. For more information, see Parent-Child Dimensions.

Selecting a Dimension

On the first **Specify a Unary Operator** page of the wizard, you specify the dimension to which you want to apply a custom aggregation. The custom aggregation added to this selected dimension will result in changes to the dimension. These changes will be inherited by all cubes that include the selected dimension.

Adding Custom Aggregation (Unary Operator)

On the second **Specify a Unary Operator** page, you specify the parent attribute that you want for the custom aggregation and the source column in the dimension table for the unary operator. **Parent attribute** lists attributes that have their **Usage** property set to **Parent**. If there is more than one parent attribute, choose the parent attribute that corresponds to the parent-child relationship that you want to use. If there is no parent attribute listed, then the dimension does not have a valid parent-child hierarchy.

In **Source column**, you select the string column that contains the unary operators. (This selection sets the **UnaryOperatorColumn** property on the parent attribute.) The dimension table should also have a string column that specifies the unary rollup operator. The string values in this column should contain valid aggregation operators. If a row is empty, the corresponding member is calculated normally. If the formula in a column is not valid, a run-time error occurs when a cell value that uses the member is retrieved. For more information, see Unary Operators in Parent-Child Dimensions.

BI Wizard - Custom Member Formulas for Attributes in a Dimension

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Add a custom member formula enhancement to a cube or dimension to replace the default aggregation that is associated with a dimension member with the results of a Multidimensional Expressions (MDX) expression. (This enhancement sets the **CustomRollupColumn** property on a specified attribute in a dimension.)

> **NOTE**
> A custom member formula is available only for dimensions that are based on existing data sources. For dimensions that were created without using a data source, you must run the Schema Generation Wizard to create a data source view before adding a custom member formula.

To add a custom member formula, you use the Business Intelligence Wizard, and select the **Create a custom member formula** option on the **Choose Enhancement** page. This wizard then guides you through the steps of selecting a dimension to which you want to apply a custom member formula and enabling the custom member formula.

Selecting a Dimension

On the first **Create a Custom Member Formula** page of the wizard, you specify the dimension to which you want to apply a custom member formula. The custom member formula enhancement added to this selected dimension will result in changes to the dimension. These changes will be inherited by all cubes that include the selected dimension.

Enabling a Custom Member Formula

On the second **Create a Custom Member Formula** page, you associate the source column that contains the custom member formula with one or more attributes in the dimension. In the **Attribute** column, select the check box next to the attribute that you want to associate with the custom member formula column. After you select each attribute, the wizard displays the **Select a Column** dialog box. In this dialog box, click the column in the dimension table that contains the formula. If you want to change a selection after you close the **Select a Column** dialog box, click the **Source Column** cell that you want to change, and then click the ellipsis (**...**) to open the **Select a Column** dialog box again.

See Also

Use the Business Intelligence Wizard to Enhance Dimensions

BI Wizard - Define the Ordering for a Dimension

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Add the attribute ordering enhancement to a cube or dimension to specify how the members of an attribute are ordered. Members can be ordered by the name or the key of the attribute, or by the name or the key of another attribute (based on an attribute relationship). By default, members are ordered by the name. This enhancement changes the **OrderBy** and **OrderByAttributeID** property settings for attributes in a dimension.

To add attribute ordering, you use the Business Intelligence Wizard, and select the **Specify attribute ordering** option on the **Choose Enhancement** page. This wizard then guides you through the steps of selecting a dimension to which you want to apply attribute ordering and specifying how to order the attributes for the selected dimension.

Selecting a Dimension

On the first **Specify Attribute Ordering** page of the wizard, you specify the dimension to which you want to apply attribute ordering. The attribute ordering enhancement added to this selected dimension will result in changes to the dimension. These changes will be inherited by all cubes that include the selected dimension.

Specifying Ordering

On the second **Specify Attribute Ordering** page of the wizard, you specify how all the attributes in the dimension will be ordered.

In the **Ordering Attribute** column, you can change the attribute used to do the ordering. If the attribute that you want to use to order members is not in the list, scroll down the list, and then select **<New attribute...>** to open the **Select a Column** dialog box, where you can select a column in a dimension table. Selecting a column by using the **Select a Column** dialog box creates an additional attribute with which to order members of an attribute.

In the **Criteria** column, you can then select whether to order the members of the attribute by either **Key** or **Name**.

BI Wizard - Enable Dimension Writeback

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Add the dimension writeback enhancement to a cube or dimension to allow users to manually modify the dimension structure and members. Updates to a write-enabled dimension are recorded directly in the dimension table. This enhancement changes the **WriteEnabled** property setting for a dimension.

To add dimension writeback, you use the Business Intelligence Wizard, and select the **Enable dimension writeback** option on the **Choose Enhancement** page. This wizard then guides you through the steps of selecting a dimension to which you want to apply dimension writeback and setting this option for the selected dimension.

> **NOTE**
>
> Writeback is supported for SQL Server relational databases and data marts only.

Selecting a Dimension

On the first **Enable Dimension Writeback** page of the wizard, you specify the dimension to which you want to apply dimension writeback. The dimension writeback enhancement added to this selected dimension will result in changes to the dimension. These changes will be inherited by all cubes that include the selected dimension.

Setting Dimension Writeback Capability

On the second **Enable Dimension Writeback** page of the wizard, you actually set the **Enable writeback in the dimension** option. Selecting this option automatically sets the **WriteEnabled** property of the dimension to **True**. Clearing this option automatically sets the property to **False**.

Remarks

When you create a new member, you must include every attribute in a dimension. You cannot insert a member without specifying a value for the key attribute of the dimension. Therefore, creating members is subject to any constraints (such as non-null key values) that are defined on the dimension table. You should also consider columns optionally specified by dimension properties, such as columns specified in the **CustomRollupColumn**, **CustomRollupPropertiesColumn** or the **UnaryOperatorColumn** dimension properties.

> **WARNING**
>
> If you use SQL Azure as a data source to perform writeback into an Analysis Services database, the operation fails. This is by design, because the provider option that enables multiple active result sets (MARS) is not turned on by default.
>
> The workaround is to add the following setting in the connection string, to support MARS and to enable writeback:
>
> `"MultipleActiveResultSets=True"`
>
> For more information see Using Multiple Active Result Sets (MARS).

See Also

Write-Enabled Dimensions

Configure String Storage for Dimensions and Partitions

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can reconfigure string storage to accommodate very large strings in dimension attributes or partitions that exceed the 4 GB file size limit for string stores. If your dimensions or partitions include string stores of this size, you can work around the file size constraint by changing the **StringStoresCompatibilityLevel** property at the dimension or partition level, for local as well as linked (local or remote) objects.

Note that you can increase string storage on just those objects that require additional capacity. In most multidimensional models, string data is associated with dimensions. However, partitions that contain distinct count measures on top of strings can also benefit from this setting. Because the setting is for strings, numeric data is unaffected.

Valid values for this property include the following:

VALUE	DESCRIPTION
1050	Specifies the default string storage architecture, subject to a 4 GB maximum file size per store.
1100	Specifies larger string storage, supports up to 4 billion unique strings per store.

> **IMPORTANT**
> Changing the string storage settings of an object requires that you reprocess the object itself and any dependent object. Processing is required to complete the procedure.

This topic contains the following sections:

- About String Stores
- Prerequisites
- Step 1: Set the StringStoreCompatiblityLevel Property in SQL Server Data Tools
- Step 2: Process the Objects

About String Stores

String storage configuration is optional, which means that even new databases that you create use the default string store architecture that is subject to the 4 GB maximum file size. Using the larger string storage architecture has a small but noticeable impact on performance. You should use it only if your string storage files are close to or at the maximum 4 GB limit.

> **NOTE**
> This setting does not apply to data mining models. Currently, it is still possible to encounter the GB file size limitation on models containing data mining structures.

In an Analysis Services multidimensional database, strings are stored separately from numeric data to allow for optimizations based on characteristics of the data. String data is typically found in dimension attributes that represent names or descriptions. It is also possible to have string data in distinct count measures. String data can also be used in keys.

You can identify a string store by its file extension (for example, asstore, .bstore, .ksstore, or .string files). By default, each of these files is subject to a maximum 4 GB limit. In SQL Server 2012 (11.x), you can override the maximum file size by specifying an alternative storage mechanism that allows a string store to grow as needed.

In contrast with the default string storage architecture which limits the size of the physical file, larger string storage is based on a maximum number of strings. The maximum limit for larger string storage is 4 billion unique strings or 4 billion records, whichever occurs first. Larger string storage creates records of an even size, where each record is equal to a 64K page. If you have very long strings that do not fit in a single record, your effective limit will be less than 4 billion strings.

Prerequisites

You must have a SQL Server 2012 (11.x) or later version of Analysis Services.

Dimensions and partitions must use MOLAP storage.

The database compatibility level must be set to 1100. If you created or deployed a database using SQL Server Data Tools and the SQL Server 2012 (11.x) or later version of Analysis Services, the database compatibility level is already set to 1100. If you moved a database created in an earlier version of Analysis Services to ssSQL11 or later, you must update the compatibility level. For databases that you are moving, but not redeploying, you can use SQL Server Management Studio to set the compatibility level. For more information, see Compatibility Level of a Multidimensional Database (Analysis Services).

Step 1: Set the StringStoreCompatiblityLevel Property in SQL Server Data Tools

1. Using SQL Server Data Tools (SSDT), open the project that contains the dimensions or partitions you want to modify.
2. To change the string storage for dimensions, open Solution Explorer. Double-click the dimension for which you are modifying string storage.
3. In Dimension Designer, in the Attributes pane, make sure that the parent node of the dimension is selected (for example, if the dimension is Customers, select Customers and not one of the child attributes).
4. In the Properties pane, in the Advanced section, set **StringStoresCompatibilityLevel** to **1100**. Repeat for other dimensions that require larger storage, otherwise leave the remaining dimensions at the **1050** value.
5. For partitions, open a cube from Solution Explorer.
6. Click the Partitions tab.
7. Expand the partition, select the partition that requires extra storage capacity, and then modify the **StringStoresCompatibilityLevel** property.
8. Save the file.

Step 2: Process the Objects

The new storage architecture will be used after you process the objects. Processing objects also proves you have successfully resolved storage constraint problem because the error that previously reported a string store overflow condition should no longer occur.

- In Solution Explorer, right-click the dimension or you just modified and select **Process**.

You must use the Process Full option on each object that is using the new string store architecture. Before processing, be sure to run an impact analysis on the dimension to check whether dependent objects also require reprocessing.

See Also

Tools and Approaches for Processing (Analysis Services)
Processing Options and Settings (Analysis Services)
Partition Storage Modes and Processing
Dimension Storage

Define Linked Dimensions

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A linked dimension is based on a dimension created and stored in another Analysis Services database of the same version and compatibility level. By using a linked dimension, you can create, store, and maintain a dimension on one database, while making it available to users of multiple databases. To users, a linked dimension appears like any other dimension.

Linked dimensions are read-only. If you want to modify the dimension or create new relationships, you must change the source dimension, then delete and recreate the linked dimension and its relationships. You cannot refresh a linked dimension to pick up changes from the source object.

All related measure groups and dimensions must come from the same source database. You cannot create new relationships between local measure groups and the linked dimensions you add to your cube. After linked dimensions and measure groups have been added to the current cube, the relationships between them must be maintained in their source database.

> **NOTE**
>
> Because refresh is not available, most Analysis Services developers copy dimensions rather than link them. You can copy dimensions across projects within the same solution. For more information, see Refresh of a linked dimension in SSAS.

Prerequisites

The source database that provides the dimension and the current database that uses it must be at the same version and compatibility level. For more information, see Compatibility Level of a Multidimensional Database (Analysis Services).

The source database must be deployed and online. Servers that publish or consume linked objects must be configured to allow the operation (see below).

The dimension you want to use cannot itself be a linked dimension.

Configure server to allow linked objects

1. In SQL Server Management Studio, connect to an Analysis Services server. In Object Explorer, right-click the server name and select **Facets**.
2. Set **LinkedObjectsLinksFromOtherInstancesEnabled** to **True** to allow the server to issue requests for linked objects that reside in databases running on other instances.
3. Set **LinkedObjectsLinksToOtherInstances** to **True** to allow the server to request data for linked on databases running on other instances.

Create a linked dimension in SQL Server Data Tools

1. Start the wizard. In SQL Server Data Tools (SSDT), right-click the **Dimensions** folder in an Analysis Services database or project, and then click **New Linked Dimension**.
2. Connect to the Analysis Services database that provides the dimension. On the **Select a Data Source** page

of the Linked Object Wizard, choose the Analysis Services data source or create a new one.

3. On the **Select Objects** page of the wizard, choose the dimensions you want to link to in the remote database.

4. On the **Completing the Wizard** page, you can preview the linked objects. If you link a dimension that has the same name as one that already exists, an ordinal number (starting with '1' for the first duplicated name) is appended to the name. When you complete the wizard, the dimension is added to the **Dimensions** folder.

Create a New Data Source Connection to an Analysis Services Database

Use the New Data Source wizard to add to your project connection information about the Analysis Services database that provides the dimension. You can start the wizard by clicking **New Data Source** in the Select a Data Source page of the Linked Objects wizard.

1. In the Data Source Wizard, on the Select how to define the connection page, click **New**.

2. In Connection Manager, verify that the provider is set to **Native OLE DB\Microsoft OLE DB Provider for Analysis Services 11.0**.

3. Enter the name of the server (use *servername\instancename* for a named instance) or type **localhost** to connect to an Analysis Services server that is running on the same computer.

4. Use Windows authentication for the connection.

5. In **Initial catalog**, click the down arrow to select a database on this server.

6. On the Data Source Wizard, click **Next** to continue.

7. On the Impersonation Information page, click **Use the service account**. Click **Next**, and then finish the wizard. The connection you just defined will be selected in the Linked Objects Wizard.

Next Steps

You cannot change the structure of a linked dimension, so you cannot view it with the **Dimension Structure** tab of Dimension Designer. After processing the linked dimension, you can view it with the **Browser** tab. You can also change its name and create a translation for the name.

See Also

Compatibility Level of a Multidimensional Database (Analysis Services)
Linked Measure Groups
Dimension Relationships

Cubes in Multidimensional Models

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A cube is a multidimensional structure that contains information for analytical purposes; the main constituents of a cube are dimensions and measures. Dimensions define the structure of the cube that you use to slice and dice over, and measures provide aggregated numerical values of interest to the end user. As a logical structure, a cube allows a client application to retrieve values, of measures, as if they were contained in cells in the cube; cells are defined for every possible summarized value. A cell, in the cube, is defined by the intersection of dimension members and contains the aggregated values of the measures at that specific intersection.

Benefits of Using Cubes

A cube provides a single place where all related data, for analysis, is stored.

Components of Cubes

A cube is composed of:

ELEMENT	DESCRIPTION
Dimensions	Dimensions in Multidimensional Models
Measures and Measure Groups	Create Measures and Measure Groups in Multidimensional Models
Partitions	Partitions in Multidimensional Models
Perspectives	Perspectives in Multidimensional Models
Hierarchies	Create User-Defined Hierarchies
Actions	Actions in Multidimensional Models
Key Performance Indicators (KPI)	Key Performance Indicators (KPIs) in Multidimensional Models
Calculations	Calculations in Multidimensional Models
Translations	Translations in Multidimensional Models (Analysis Services)

Related Tasks

TOPIC	DESCRIPTION
Create a Cube Using the Cube Wizard	Describes how to use the Cube Wizard to define a cube, dimensions, dimension attributes, and user-defined hierarchies.

TOPIC	DESCRIPTION
Create Measures and Measure Groups in Multidimensional Models	Describes how to define measure groups.
Calculations in Multidimensional Models	Describes how to define and configure a calculation in an MDX script.
Actions in Multidimensional Models	Describes how to define and configure an action.
Perspectives in Multidimensional Models	Describes how to define and configure a perspective.
Defining Stored Procedures	Describes how to work with stored procedures.

Create a Cube Using the Cube Wizard

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can create a new cube by using the Cube Wizard in SQL Server Data Tools (SSDT).

To create a new cube

1. In **Solution Explorer**, right-click **Cubes**, and then click **New Cube**.
2. On the **Select Creation Method** page of the Cube Wizard, select **Use existing tables**, and then click **Next**.

   > **NOTE**
   > You might occasionally have to create a cube without using existing tables. To create an empty cube, select **Create an empty cube**. To generate tables, select **Generate tables in the data source**.

3. On the **Select Measure Group Tables** page, do the following procedures:
   a. In the **Data source view** list, select a data source view.
   b. In the **Measure group tables** list, select the tables that will be used to create measure groups.
   c. Click **Next**.
4. On the **Select Measures** page, select the measures that you want to include in the cube, and then click **Next**.

   Optionally, you can change the names of the measures and measure groups.

5. On the **Select Existing Dimensions** page, select the existing dimensions to include in the cube, and then click **Next**.

   > **NOTE**
   > The **Select Existing Dimensions** page appears if dimensions already exist in the database for any of the selected measure groups.

6. On the **Select New Dimensions** page, select the new dimensions to create, and then click **Next**.

   > **NOTE**
   > The **Select New Dimensions** page appears if any tables would be good candidates for dimension tables, and the tables have not already been used by existing dimensions.

7. On the **Select Missing Dimension Keys** page, select a key for the dimension, and then click **Next**.

> **NOTE**
>
> The **Select Missing Dimension Keys** page appears if any of the dimension tables that you specified do not have a key defined.

8. On the **Completing the Wizard** page, enter a name for the new cube and review the cube structure. If you want to make changes, click **Back**; otherwise, click **Finish**.

> **NOTE**
>
> You can use Cube Designer after you complete the Cube Wizard to configure the cube. You can also use Dimension Designer to add, remove, and configure attributes and hierarchies in the dimensions that you created.

Create a Cube from a template without using a Data Source View

11/28/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Select **Build the cube without using a data source** on the first page of the Cube Wizard to create a cube without using a data source view. You can later use the Schema Generation Wizard to generate the relational schema for the data source view based on the structure of the cube and possibly other Analysis Services objects. For more information about generating a schema, see Schema Generation Wizard (Analysis Services).

Selecting the Build Method

In the Cube Wizard, on the **Select Build Method** page, click **Build the cube without using a data source**. To build the cube by using an existing cube template, select the **Use a cube template** check box. . If you do not select to use a template, you must set the options manually.

Cube templates contain predefined measures, measure groups, dimensions, hierarchies, and attributes. If you select a template, the wizard uses the object definitions in the templates as the basis for setting options in the following pages. Analysis Services is installed with several templates for standard cubes. The server administrator can also add cube or dimension templates that are designed specifically for your organization's data.

Selecting Dimensions

Use the **Select Dimensions** page of the wizard to add existing dimensions to the cube. This page appears only if there are already shared dimensions without a data source in the project or database. It does not list dimensions that have a data source.

To add existing dimensions, select one or more dimensions in the **Shared dimensions** list and click the right arrow (**>**) button to move them to the **Cube dimensions** list. Click the double-arrow (**>>**) button to move all the dimensions in the list.

Defining New Measures

Use the **Define New Measures** page of the wizard to specify the measures and measure groups in the new cube. The measure groups you specify here will correspond to fact tables in the generated schema. The measures you specify here will correspond to numeric non-key columns in the tables.

If you use a template to create the cube, measures in the template are listed in grid format under **Select measures from template**. The check box next to every measure in the list is initially selected. You can clear the check box next to any measure that you do not want to include in the cube. To add or remove all the measures in the list, select or clear the check box on the title bar of the grid.

You can add measures to the cube in the list under **Add new measures**. To add a new measure, click the first empty cell in the **Measure Name** column (which displays **Add new measure**). Specify a measure name, measure group, data type, and aggregation for each new measure. To delete a measure from the **Add new measures** list, click the delete icon (**X**). If you do not use a template, **Add new measures** is the only list on this page of the wizard.

Both the **Select measures from template** grid and the **Add new measures** grid display values under the columns described in the following table. You can click a value in either list to change it.

COLUMN	DESCRIPTION
Measure Name	A value in this column defines the name of a measure in the cube. Click a value in this column to type a name. Click **Add new measure** in this column to create a new measure. This column sets the **Name** property on the measure object.
Measure Group	The name of the measure group that contains the measure. Click this value to either choose or type a name. If you delete all the measures that belong to a particular measure group, the measure group is removed as well. This column sets the **Name** property for the measure group object.
Data Type	The data type for the measure. Click this value to change the data type. The default when you create a measure is **Single**. This column sets the **DataType** property on the measure object.
Aggregation	The standard aggregation for the measure. Click this cell to specify one of the standard aggregations for the measure (or **None**). The default when you create a measure is **Sum**. This column sets the **AggregationFunction** property on the measure object.

Defining New Dimensions

Use the **Define New Dimensions** page of the wizard to specify the dimensions in the new cube.

If you use a template to create the cube, the grid under **Select dimensions from template** displays the dimensions in the template. You can clear the check box next to any dimension to remove it from the cube. Clear the check box on the title bar of the grid to remove all the dimensions listed. If you do not use a template, this grid lists only the Time dimension.

You can add dimensions to the cube in the grid under **Add new dimensions**. To add a dimension, click the cell in the **Name** column that contains the text **Add new dimension**, and then type a name for the dimension. To remove a row from the list, click the delete icon (**X**).

Both the **Select dimensions from template** grid and the **Add new dimensions** grid display values under the columns described in the following table. You can click a value in either list to change it.

COLUMN	DESCRIPTION
Type	Displays the dimension type for a template dimension. Click this cell to change the dimension type for a dimension. This column sets the **Type** property for the dimension object.
Name	Displays the dimension name. Click this cell to type a different name. This value sets the **Name** property for the dimension object.
SCD	Specifies that this is a slowly changing dimension (SCD). Selecting this check box adds the SCD Start Date, End Date Original ID, and Status attributes to the dimension. **SCD** is selected by default if you use a template to create the cube and the wizard detects these four attribute types in a template dimension.

COLUMN	DESCRIPTION
Attributes	Displays the attributes that are to be created for the dimension. Each attribute name in the list is preceded by the dimension name. This list is read-only. You can edit the attributes by using Dimension Designer after you complete the wizard.

Defining Time Periods

Use the **Define Time Periods** page of the wizard to specify the range of dates you want to include in the dimension. For example, you might choose a range starting on January 1 of the earliest year in your data and extending years past your most current transaction. Transactions that are outside the range either do not appear or appear as unknown members in the dimension, depending on the **UnknownMemberVisible** property setting for the dimension. The **UnknownMemberName** property specifies the caption for the unknown member. You can also change the first day of the week used by your data (the default is Sunday).

> **NOTE**
>
> The **Define Time Periods** page appears only if you include a time dimension in your cube on the **Define New Dimensions** page of the wizard.

Select the time periods (**Year**, **Half Year**, **Quarter**, **Trimester**, **Month**, **Ten Days**, **Week**, and **Date**) that you want to include in your schema. You must select the Date time period; the Date attribute is the key attribute for the dimension, so the dimension cannot function without it. You can also change the language used to label the members of the dimension.

The time periods you select create corresponding time attributes in the new time dimension. The wizard also adds related attributes that do not appear in the list. For example, when you select **Year** and **Half Year** time intervals, the wizard creates Day of Year, Day of Half Year, and Half Years of Year attributes, in addition to Year and Half Year attributes.

After you finish creating the cube, you can use Dimension Designer to add or remove time attributes. Because the Date attribute is the key attribute for the dimension, you cannot remove it. To hide the Date attribute from users, you can change the **AttributeHierarchyVisible** property to **False**.

All the available time periods appear in the Time Periods pane of Dimension Designer. (This pane replaces the **Data Source View** pane for dimensions that are based on dimension tables.) You can change the range of dates for a dimension by changing the **Source** (time binding) property setting for the dimension. Because this is a structural change, you must reprocess the dimension and any cubes that use it before browsing the data.

Specifying Additional Calendars

On the **Specify Additional Calendars** page of the wizard, select calendars on which to base hierarchies in the dimension. You can choose any of the following calendars.

CALENDAR	DESCRIPTION
Fiscal calendar	A twelve-month fiscal calendar. If you select this calendar, specify the starting day and month for the fiscal year used by your organization.

CALENDAR	DESCRIPTION
Reporting (or marketing) calendar	A twelve-month reporting calendar that includes two months of four weeks and one month of five weeks in a recurring three-month (quarterly) pattern. If you select this calendar, specify the starting day and month and the three-month pattern of 4-4-5, 4-5-4, or 5-4-4 weeks, where each digit represents the number of weeks in a month.
Manufacturing calendar	A calendar that uses 13 periods of four weeks, divided into three quarters of four periods and one quarter of five periods. If you select this calendar, specify the starting week (between 1 and 4) and month for the manufacturing year, and the quarter with extra periods.
ISO 8601 Calendar	The International Organization for Standardization (ISO) Representation of Dates and Time standard calendar (8601). This calendar has an integral number of 7-day weeks. To avoid splitting a week, this calendar starts a new year up to several days before or after January 1.

The calendar and settings you select determine the attributes that are created in the dimension. For example, if you select the **Year** and **Quarter** time periods on the **Define Time Periods** page of the wizard and the **Fiscal calendar** on this page, the FiscalYear, FiscalQuarter, FiscalQuarterOfYear attributes are created for the fiscal calendar.

The wizard also creates calendar-specific hierarchies composed of the attributes that are created for the calendar. For every calendar, each level in every hierarchy rolls up into the level above it. For example, in the standard 12-month calendar, the wizard creates a hierarchy of Years and Weeks or Years and Months. However, weeks are not contained evenly within months in a standard calendar, so there is no hierarchy of Years, Months, and Weeks. In contrast, weeks in a reporting or manufacturing calendar are evenly divided into months, so in these calendars weeks do roll up into months.

Defining Dimension Usage

Use the **Define Dimension Usage** page of the wizard to specify which cube measures are aggregated by each dimension in the wizard. The **Dimension usage** grid on this page lists the dimensions as rows and measure groups as columns. Select the check box for any dimension and measure group combination in which the dimension aggregates the measures of that measure group.

Completing the Cube Wizard

On the **Completing the Wizard** page, review the structure of the new cube and in the **Cube name** box, type a name for it. Optionally, select the **Generate schema now** check box to start the schema generation wizard. In most cases, you should not select this check box if you plan to create additional objects. You can also use Cube Designer to generate the schema later.

Create a Cube using a Data Source View

5/16/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use this method of building a new cube if you intend to use an existing data source view. With this method, you specify the data source view and select fact and dimension tables that you want to use in the data source view. You then choose the dimensions and measures that you want to include in the cube.

To create a cube with a data source, in Solution Explorer, right-click **Cubes** and select **New Cube**. The Cube Wizard opens.

Selecting the Build Method

On the **Select Build Method** page of the wizard, click **Build the cube using a data source**.

If you select the **Auto build** check box, the wizard analyzes the data source view to configure the cube and its dimensions for you. The wizard identifies fact and dimension tables, selects measures to include in the cube, and builds hierarchies. On each page of the wizard, you can examine and change the choices that the wizard makes when **Auto build** is selected. If you do not select **Auto build**, you make all these choices manually.

If you select **Auto build**, you can click **Finish** on any page of the wizard to jump to the last page and accept the default configurations for any remaining pages of the wizard. On the last page of the wizard, you can review the structure of the cube before finishing the wizard.

If you do not select **Auto build**, you must select the fact and dimension tables yourself. The wizard builds any dimensions that you choose to create, but you must use Dimension Designer to manually build user-defined hierarchies in the dimensions. This requirement may not make any difference if you have already created the dimensions you want to use in the cube before running the Cube Wizard.

Selecting the Data Source View

If you use an existing data source to create a cube, the first step is to specify the data source view on which to base the cube. On the **Select Data Source View** page of the wizard, select an existing data source view. In the preview pane, you can view the tables in a selected data source view. To display the schema for any selected data source view, click **Browse**.

If the data source view you want to use is not listed, in the Cube Wizard, click **Cancel**, and open the Data Source View Wizard. You can also click **Add New Item** on the **File** menu to add an existing data source view from another database (or other location). For more information about creating data source views, see Data Source Views in Multidimensional Models.

> **NOTE**
>
> A data source view must contain at least one table to be listed on this page. You cannot create a cube based on a data source view that does not have any tables.

Identify Fact and Dimension Tables

In the Cube Wizard, use the **Identify Fact and Dimension Tables** page of the wizard to select the fact and dimension tables required to create the cube. If you selected the **Auto build** check box to create the cube, the fact

or dimension tables the wizard detects are selected when this page first appears. If the wizard detects a table that is both a fact table and a dimension table, both columns are selected. If the wizard detects a table that is neither, neither column is selected. If you do not need a table for your cube design, clear the **Fact** and **Dimension** check boxes.

If you did not select the **Auto build** check box, you must make all the selections manually. Use the **Tables** tab or the **Diagram** tab:

- The **Tables** tab lists the tables in table format. Select the check box in the **Fact** column or the **Diagram** column.
- The **Diagram** tab displays the data source view schema. Tables are color coded to indicate fact or dimension. Click any table in the schema, and then click **Fact** or **Dimension** to select or clear the setting on that table. Use the **Zoom** button to change the magnification.

> **NOTE**
>
> On the **Diagram** tab, you can enlarge or maximize the wizard window to view the schema.

If there is a time dimension table in the data source view, select it in the **Time dimension table** list. If there are none, leave **<None>** selected. This is the default item on the list. Selecting a table as the time dimension table also selects it as a dimension table on the **Tables** and **Diagram** tabs.

Defining Time Periods

If you specified a time dimension table while selecting table types, use the **Define Time Periods** page of the wizard to specify the columns in the table that correspond to standard time periods. Look for the standard periods under **Time Property Name**. For each row that has a corresponding column in the time dimension table, choose the correct column under **Time Table Columns**. The wizard uses the associations you specify to create attributes and suggest time hierarchies that make sense for your data. These associations also set the **Type** property for the corresponding attributes in the new time dimension. The wizard then creates a time dimension based on a time dimension table.

After you create the cube, you can use the Business Intelligence Wizard to add time intelligence enhancements to the cube. These enhancements include period-to-date, rolling average, and period-to-period views.

Selecting Dimensions

Use the **Select Dimensions** page of the wizard to add existing dimensions to the cube. This page appears only if there are already shared dimensions that correspond to dimension tables in the new cube.

To add existing dimensions, select one or more dimensions in the **Shared dimensions** list and click the right arrow (**>**) button to move them to the **Cube dimensions** list. Click the double-arrow (**>>**) button to move all the dimensions in the list.

If an existing dimension does not appear in the list and you think it should, you can click **Back** and change the table type settings for one or more tables. An existing dimension must also be related to at least one of the fact tables in the cube to appear in the **Shared dimensions** list.

Selecting Measures

Use the **Select Measures** page of the wizard to select the measures you want to include in the cube. Every table marked as a fact table appears as a measure group in this list, and every numeric non-key column appears as a measure on the list. By default every measure in every measure group is selected. You can clear the check box next to measures that you do not want to include in the cube. To remove all the measures of a measure group from the

cube, clear the **Measure Groups/Measures** check box.

The names of measures listed under **Measure Groups/Measures** reflect column names. You can click the cell that contains a name to edit the name.

To view data for any measure, right-click any of the measure rows in the list, and then click **View sample data**. This opens the **Data Sample Viewer** and displays up to the first 1000 records from the corresponding fact table.

Reviewing New Dimensions

Use the **Review New Dimensions** page of the wizard to review the structures of any dimensions created by the wizard. The dimensions are listed on this page of the wizard in the **New dimensions** tree view. You can review the dimensions in the following ways:

- Expand any dimension to view its attributes and hierarchies.
- Expand the **Attributes** folder under any dimension to view the attributes in the dimension.
- Expand the **Hierarchy** folder under any dimension to view the hierarchies in the dimension.
- Expand any hierarchy to view its levels.

> **NOTE**
> You can enlarge or maximize the wizard window to view the tree better.

To remove any object in the tree from the cube, clear the check box next to it. Clearing the check box next to an object also removes all the objects underneath it. Dependencies between objects are enforced, so if you remove an attribute, any hierarchy levels dependent on the attribute are also removed. For example, clearing a check box next to a hierarchy clears the check boxes next to all the levels in the hierarchy and removes the levels as well as the hierarchies. The key attribute for a dimension cannot be removed.

You can rename any dimension, attribute, hierarchy or level either by clicking the name or by right-clicking the name and then on the shortcut menu clicking **Rename <object>**, where **<object>** is **Dimension**, **Attribute**, or **Level**.

There is not necessarily a one-to-one relationship between the number of dimension tables defined on the **Identify Fact and Dimension Tables** page of the wizard and the number of dimensions listed on this page of the wizard. Depending on relationships between tables in the data source view, the wizard can use two or more tables to build a dimension (for example, as required by a snowflake schema).

Completing the Cube Wizard

On the **Completing the Wizard** page of the wizard, you can view the measure groups, measures, and dimensions in the new cube. In the **Cube name** box, type a name for the cube. Then, if you are satisfied with the cube, click **Finish**. Click **Back** to go back to any previous page of the wizard and make changes.

Browse data and metadata in Cube

11/28/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use the **Browser** tab of Cube Designer to browse cube data. You can use this view to examine the structure of a cube and to check data, calculation, formatting, and security of database objects. You can quickly examine a cube as end users see it in reporting tools or other client applications. When you browse cube data, you can view different dimensions, drill down into members, and slice through dimensions.

Before you browse a cube, you must process it and the reconnect to it. After you process it, open the **Browser** tab of Cube Designer. Click the Reconnect button on the toolbar to refresh the connection.

The **Browser** tab has three panes - the Metadata pane, the Filter pane, and the Data pane. Use the Metadata pane to examine the structure of the cube in tree format. Use the Filter pane at the top of the **Browser** tab to define any subcube you want to browse. Use the Data pane to view the result set and drill down through dimension hierarchies.

Setting up the Browser

To prepare to browse a cube, you can specify a perspective or translation that you want to use. You add measures and dimensions to the Data pane and specify any filters in the Filter pane.

Specifying a Perspective

Use the **Perspective** list to choose a perspective that is defined for the cube. Perspectives are defined in the **Perspectives** tab of Cube Designer. To switch to a different perspective, select any perspective in the list.

Specifying a Translation

Use the **Language** list to choose a translation that is defined for the cube. Translations are defined in the **Translations** tab of Cube Designer. The **Browser** tab initially shows captions for the default language, which is specified by the **Language** property for the cube. To switch to a different language, select any language in the list.

Formatting the Data Pane

You can format captions and cells in the Data pane. Right-click the selected cells or captions that you want to format, and then click **Commands and Options**. Depending on the selection, the **Commands and Options** dialog box displays settings for **Format**, **Filter and Group**, **Report**, and **Behavior**.

Setting up the Data

Adding or Removing Measures

Drag the measures you want to browse from the Metadata pane to the details area of the Data pane, which is the large empty pane on the lower right side of the workspace. As you drag additional measures, they are added as columns. A vertical line indicates where each additional measure will drop. Dragging the **Measures** folder drops all the measures into the details area.

To remove a measure from the details area, either drag it out of the Data pane, or right-click it and then click **Delete** on the shortcut menu.

Adding or Removing Dimensions

Drag the hierarchies or dimensions to the row or filter areas.

If you want to work with multiple dimensions, use Analyze in Excel. Analyze in Excel is a shortcut that starts Excel if it is installed on the same computer as SQL Server Data Tools. Excel will open a workbook that contains an existing connection to the current database and a PivotTable Field List that is preloaded with measures and dimensions. For more information, see Analyze in Excel (Browser Tab, Cube Designer) (Analysis Services - Multidimensional Data).

To remove a dimension, either drag it out of the Data pane, or right-click it and then click **Delete** on the shortcut menu.

Adding or Removing Filters

You can use either of two methods to add a filter:

- Expand a dimension in the Metadata pane, and then drag a hierarchy to the Filter pane.

  - or -

- In the **Dimension** column of the **Filter** pane, click **<Select dimension>** and select a dimension from the list, then click **<Select hierarchy>** in the **Hierarchy** column and select a hierarchy from the list.

After you specify the hierarchy, specify the operator and filter expression. The following table describes the operators and filter expressions.

OPERATOR	FILTER EXPRESSION	DESCRIPTION
Equal	One or more members	Values must be equal to a specified member. (Provides multiple member selection for attribute hierarchies, other than parent-child hierarchies, and single member selection for other hierarchies.)
Not Equal	One or more members	Values must not equal a specified member. (Provides multiple member selection for attribute hierarchies, other than parent-child hierarchies, and single member selection for other hierarchies.)
In	One or more named sets	Values must be in a specified named set. (Supported for attribute hierarchies only.)
Not In	One or more named sets	Values must not be in a specified named set. (Supported for attribute hierarchies only.)
Range (Inclusive)	One or two delimiting members of a range	Values must be between or equal to the delimiting members. If the delimiting members are equal or only one member is specified, no range is imposed and all values are permitted. (Supported only for attribute hierarchies. The range must be on one level of a hierarchy. Unbounded ranges are not currently supported.)

OPERATOR	FILTER EXPRESSION	DESCRIPTION
Range (Exclusive)	One or two delimiting members of a range	Values must be between the delimiting members. If the delimiting members are the equal or only one member is specified, values must be either greater than or less than the delimiting member. (Supported only for attribute hierarchies. The range must be on one level of a hierarchy. Unbounded ranges are not currently supported.)
MDX	An MDX expression returning a member set	Values must be in the calculated member set. Selecting this option displays the **Calculated Member Builder** dialog box for creating MDX expressions.

For user-defined hierarchies, in which multiple members may be specified in the filter expression, all the specified members must be at the same level and share the same parent. This restriction does not apply for parent-child hierarchies.

Working with Data

Drilling Down into a Member

To drill down into a particular member, click the plus sign (+) next to the member or double-click the member.

Slicing Through Cube Dimensions

To filter the cube data, click the drop-down list box on the top-level column heading. You can expand the hierarchy and select or clear a member on any level to show or hide it and all its descendants. Clear the check box next to **(All)** to clear all the members in the hierarchy.

After you have set this filter on dimensions, you can toggle it on or off by right-clicking anywhere in the Data pane and clicking **Auto Filter**.

Filtering Data

You can use the filter area to define a subcube on which to browse. You can add a filter by either clicking a dimension in the Filter pane or by expanding a dimension in the Metadata pane and then dragging a hierarchy to the Filter pane. Then specify an **Operator** and **Filter Expression**.

Performing Actions

A triangle marks any heading or cell in the Data pane for which there is an action. This might apply for a dimension, level, member, or cube cell. Move the pointer over the heading object to see a list of available actions. Click the triangle in the cell to display a menu and start the associated process.

For security, the **Browser** tab only supports the following actions:

- URL
- Rowset
- Drillthrough

Command Line, Statement, and Proprietary actions are not supported. URL actions are only as safe as the Web page to which they link.

Viewing Member Properties and Cube Cell Information

To view information about a dimension object or cell value, move the pointer over the cell.

Showing or Hiding Empty Cells

You can hide empty cells in the data grid by right-clicking anywhere in the Data pane and clicking **Show Empty Cells**.

View the Cube Schema

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The **Data Source View** pane of the **Cube Structure** tab in **Cube Designer** displays the cube schema. The schema is the set of tables from which the measures and dimensions for a cube are derived. Every cube schema consists of one or more fact tables and one or more dimension tables on which the measures and dimensions in the cube are based.

The **Data Source View** pane of the **Cube Structure** tab displays a diagram of the data source view on which the cube is based. This diagram is a subset of the main diagram of the data source view. You can hide and show tables in the **Data Source View** pane and view any existing diagrams. However, you cannot make changes (such as adding new relationships or named queries) to the underlying schema. To make changes to the schema, use Data Source View Designer.

When you create a cube, the diagram displayed in the **Data Source View** pane of the **Cube Structure** tab is initially the same as the **Show All Tables** diagram in the data source view for the project or database. You can replace this diagram with any existing diagram in the data source view and make adjustments in the **Data Source View** pane.

While you work with the diagram in **Cube Designer**, commands that act on the tab or on any selected object in the tab are available on the **Data Source View** menu. You can also right-click the background of the diagram or any object in the diagram to use commands that act on the diagram or selected object. You can:

- Switch between diagram and tree formats.
- Arrange, find, show, and hide tables.
- Show friendly names.
- Switch layouts.
- Change the magnification.
- View properties.

Additionally, you can perform the actions listed in the following table:

TO	DO THIS
Use a diagram from the data source view of the cube	On the **Data Source View** menu, point to **Copy Diagram from**, and then click the data source view diagram you want to use. - or - Right-click the background of the **Data Source View** pane, point to **Copy Diagram from**, and then click the diagram in the data source view that you want. This method creates an independent copy of the diagram, so any changes you make on the **Cube Builder** tab do not appear in the original diagram.

TO	DO THIS
Show only the tables that are used in the cube	On the **Data Source View** menu, click **Show Only Used Tables**. - or - Right-click the background of the **Data Source View** pane, and then click **Show Only Used Tables**.
Edit the data source view schema	On the **Data Source View** menu, click **Edit Data Source View**. - or - Right-click the background of the **Data Source View** pane, and then click **Edit Data Source View**.

Define Cube Dimension Properties

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A cube dimension is an instance of a database dimension within a cube. A database dimension can be used in multiple cubes, and multiple cube dimensions can be based on a single database dimension. The following table describes the properties of a cube dimension.

PROPERTY	DESCRIPTION
AllMemberAggregationUsage	Controls how aggregations are designed by the Aggregation Designer in Microsoft SQL Server Analysis Services. This property can have the following values: **Full**: Every aggregation for the cube must include the All member. **None**: No aggregation for the cube can include the All member. This is the default value. **Unrestricted**: No restrictions are placed on the Aggregation Designer. **Default**: The same functionality as Unrestricted.
Description	Provides a descriptive name for the level.
DimensionID	Contains the unique identifier (ID) of the database dimension.
HierarchyUniqueNameStyle	Determines how unique names are generated for hierarchies that are contained within the cube dimension. This property can have the following values: **IncludeDimensionName**: The name of the dimension is included as part of the name of the hierarchy. This is the default value. **ExcludeDimensionName**: The name of the dimension is not included as part of the name of the hierarchy.
ID	Contains the unique identifier of the cube dimension.
MemberUniqueNameStyle	Determines how unique names are generated for members of hierarchies contained within the cube dimension. This property can have the following values: **Native**: Analysis Services automatically determines the unique names of members. This is the default value. **NamePath**: Analysis Services generates a compound name consisting of the name of each level and the caption of the member.

PROPERTY	DESCRIPTION
Name	Contains the friendly name of the cube dimension. By default, the name of a cube dimension is the same as the name of the database dimension, unless another cube dimension of the same name is already defined.
Visible	Determines whether the cube dimension is visible. The default value is **True**.

See Also

Dimensions (Analysis Services - Multidimensional Data)

Define Cube Hierarchy Properties

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Cube hierarchy properties enable you to specify unique settings for user-defined hierarchies in cube dimensions based on the same database dimension. The following table describes the properties of a cube hierarchy.

PROPERTY	DESCRIPTION
Enabled	Determines whether the hierarchy is enabled for the cube dimension.
HierarchyID	Contains the unique identifier (ID) of the hierarchy.
OptimizedState	Determines the level of optimization that is applied to the hierarchy. This property can have the following values: **FullyOptimized**: The instance builds indexes for the hierarchy to improve query performance. This is the default value. **NotOptimized**: The instance does not build additional indexes.
Visible	Determines the visibility of the cube hierarchy. The default value is **True**.

See Also

User Hierarchies

Define Cube Attribute Properties

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Cube attribute properties enable you to specify unique settings for dimension attributes in cube dimensions based on the same database dimension. The following table describes the properties of a cube attribute.

PROPERTY	DESCRIPTION
AggregationUsage	Specifies how the Aggregation Design Wizard will design aggregations for the attribute. The default value is **Default**. This property can have the following values: **Default**: The Aggregation Design Wizard applies a default rule based on the type of attribute (Full for keys, Unrestricted for others). **None**: No aggregation for the cube should include this attribute. **Unrestricted**: No restrictions are placed on the Aggregation Design Wizard **Full**: Every aggregation for the cube must include this attribute.
AttributeHierarchyEnabled	Identifies whether the attribute hierarchy is enables on this cube dimension. This allows attribute hierarchies to be disabled on specific cubes or dimension roles. This setting has no effect if the underlying attribute hierarchy is disabled. Default value is **True**.
OptimizedState	Indicates whether the attribute hierarchy is optimized on this cube dimension. This allows attribute hierarchies to be optimized on specific cubes or dimension roles. This setting has no effect if the underlying attribute hierarchy is not optimized. The default value is **FullyOptimized**. This property can have the following values: **FullyOptimized**: The instance builds indexes for the hierarchy to improve query performance. This is the default value. **NotOptimized**: The instance does not build additional indexes.
AttributeHierarchyVisible	Indicates whether the attribute hierarchy is visible on this cube dimension. This allows attribute hierarchies to be visible on specific cubes or dimension roles. This setting has no effect if the underlying attribute hierarchy is not visible. The default value is **True**.
AttributeID	Contains the unique identifier (ID) of the attribute.

See Also

Define Cube Dimension Properties
Define Cube Hierarchy Properties

Define a Regular Relationship and Regular Relationship Properties

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you define a new cube dimension or a new measure group, Analysis Services will try to detect if a regular relationship exists and then set the dimension usage setting to **Regular**. You can view or edit a regular dimension relationship on the **Dimension Usage** tab of Cube Designer.

When you define the relationship of a cube dimension to a measure group, you also specify the granularity attribute for that relationship. The granularity attribute defines the lowest level of detail available in the cube for that dimension, which is generally the key attribute for the dimension. However, sometimes you may want to set the granularity of a particular cube dimension in a particular measure group to a different grain. For example, you may want to set the granularity attribute for the Time dimension to the Month attribute instead of to the Day attribute, if you are using a Sales Quotas or a Budget measure group. When you specify the granularity attribute to be an attribute other than the key attribute, you must guarantee that all other attributes in the dimension are directly or indirectly linked to this other attribute through attribute relationships. If not, Analysis Services will be unable to aggregate data correctly.

Define a Referenced Relationship and Referenced Relationship Properties

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A reference dimension relationship is defined on the **Dimension Usage** tab of Cube Designer. The reference dimension relationship is defined by specifying the following:

- The intermediate dimension to which to join. This can be a regular dimension or another reference dimension.
- A reference dimension attribute that defines the lowest level that the dimension is available for aggregation with regard to the measure group.
- The (foreign key) attribute in the intermediate dimension that corresponds to the reference dimension attribute.

Notice that the column that links the reference dimension to the fact table, which is generally the key attribute in the reference dimension, must also be defined as an attribute in the intermediate dimension. When you create a chain of reference dimensions, start by creating the regular relationship between the first dimension in the chain and the measure group. Then create each additional referenced relationship in order. A referenced relationship can only be made to a dimension that has an existing relationship to the measure group.

When you create a reference dimension relationship, the dimension attribute link is materialized by default. Materializing a dimension attribute link causes the value of the link between the fact table and the reference dimension for each row to be materialized, or stored, in the MOLAP structure for the dimension during processing. This will have a minor effect on processing performance and storage requirements, but will improve query performance.

In a reference dimension, granularity is specified by identifying the attribute that defines the relationship between a reference dimension and the measure group corresponding to the main table of the dimension. When multiple reference dimensions are chained together, the references define the relationship from the outermost dimension to the measure group.

Define a Fact Relationship and Fact Relationship Properties

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you define a new cube dimension or a new measure group, Analysis Services will try to detect if a fact dimension relationship exists and then set the dimension usage setting to **Fact**. You can view or edit a fact dimension relationship on the **Dimension Usage** tab of Cube Designer. The fact relationship between a dimension and a measure group has the following constraints:

- A cube dimension can have only one fact relationship to a particular measure group.
- A cube dimension can have separate fact relationships to multiple measure groups.
- The granularity attribute for the relationship must be the key attribute (such as Transaction Number) for the dimension. This creates a one-to-one relationship between the dimension and facts in the fact table.

Define a Many-to-Many Relationship and Many-to-Many Relationship Properties

5/16/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic explains many-to-many dimensions in Analysis Services, including when to use them and how to create them.

Introduction

Analysis Services supports many-to-many dimensions, allowing for more complex analytics than what can be described in a classic star schema. In a classic star schema, all dimensions have a one-to-many relationship with a fact table. Each fact joins to one dimension member; a single dimension member is associated with many facts.

Many-to-many removes this modeling restriction by enabling a fact (such as an account balance) to be associated with multiple members of the same dimension (the balance of a joint account can be attributed to two or more owners of a joint account).

Conceptually, a many-to-many dimensional relationship in Analysis Services is equivalent to many-to-many relationships in a relational model, supporting the same kinds of scenarios. Common examples of many-to-many include:

- Students are enrolled in many courses; each course has many students.
- Doctors have many patients; patients have many doctors.
- Customers have many bank accounts; bank accounts might belong to more than one customer.
- In Adventure Works, many customers have many reasons for ordering a product, and a sales reason can be associated with many orders.

Analytically, the problem that a many-to-many relationship solves is accurate representation of a count or sum relative to the dimensional relationship (usually by eliminating double-counts when performing calculations for a specific dimension member). An example is necessary to clarify this point. Consider a product or service that belongs to more than one category. If you were counting the number of services by category, you would want a service belonging to both categories to be included in each one. At the same time, you would not want to overstate the number of services that you provide. By specifying the many-to-many dimensional relationship, you are more likely to get the correct results back when querying by category or by service. However, thorough testing is always necessary to ensure that this is the case.

Structurally, creating a many-to-many dimensional relationship is similar to how you might create many-to-many in a relational data model. Whereas a relational model uses a *junction table* to store row associations, a multidimensional model uses an *intermediate measure group*. Intermediate measure group is the term we use to refer to a table that maps members from different dimensions.

Visually, a many-to-many dimensional relationship is not indicated in a cube diagram. Instead, use the Dimension Usage tab to quickly identify any many-to-many relationships in a model. A many-to-many relationship is indicated by the following icon.

Click the button to open the Define Relationship dialog box to verify the relationship type is many-to-many, and to

view which intermediate measure group is used in the relationship.

In subsequent sections, you will learn how to set up a many-to-many dimension and test model behaviors. If you would rather review additional information or try tutorials first, see **Learn More** at the end of this article.

Create a many-to-many dimension

A simple many-to-many relationship includes two dimensions having a many-to-many cardinality, an intermediate measure group for storing member associations, and a fact measure group containing measurable data, such as sum of total sales or the balance of a bank account.

Dimensions in a many-to-many relationship might have correspondent tables in the DSV, where each dimension in the model is based on an existing table in a data source. Conversely, the dimensions in your model might derive from fewer or different physical tables in the DSV. Using Sales Reasons and Sales Orders as a case in point, the Adventure Works sample cube demonstrates a many-to-many relationship using dimensions that exist as model-only data structures, without physical counterparts in the DSV. The Sales Order dimension is based on a fact table, rather than a dimension table, in the underlying data source.

The next procedure assumes that you already know which entities participate in the many-to-many relationship. See **Learn More** for further study.

To illustrate the steps used to create a many-to-many relationship, this procedure re-creates one of the many-to-many relationships in the Adventure Works sample cube. If you have the source data (that is, the Adventure Works sample data warehouse) installed on a relational database engine instance, you can follow these steps.

Step 1: Verify DSV relationships

1. In SQL Server Data Tools, in a multidimensional project, create a data source to the Adventure Works DW 2012 relational data warehouse, hosted on a SQL Server Database Engine instance.

2. Create a Data Source View using the following existing tables:

   - FactInternetSales
   - FactInternetSalesReason
   - DimSalesReason

3. Verify that all of the tables you plan to use in the many-to-many relationships are related in the DSV through primary key relationships. This is a requirement for establishing a link to the intermediate measure group in a subsequent step.

   > **NOTE**
   > If the underlying data source does not provide primary and foreign key relationships, you can create the relationships manually in the DSV. For more information, see Define Logical Relationships in a Data Source View (Analysis Services).

   The following example confirms that the tables used in this procedure are linked using primary keys.

Step 2: Create dimensions and measure groups

1. In SQL Server Data Tools, in a multidimensional project, right-click **Dimensions** and select **New Dimension**.

2. Create a new dimension based on existing table, **DimSalesReason**. Accept all of the default values when specifying the source.

   For attributes, select all.

3. Create a second dimension based on existing table, Fact Internet Sales. Although this is a fact table, it contains Sales Order information. We'll use it to build a Sales Order dimension.

4. In Specify Source Information, you will see a warning that indicates a Name column must be specified. Choose **SalesOrderNumber** as the Name.

5. On the next page of the wizard, choose the attributes. In this example, you can select just **SalesOrderNumber**.

6. Rename the dimension to **Dim Sales Orders**, so that you have a consistent naming convention for the dimensions.

7. Right-click **Cubes** and select **New Cube**.

8. In measure group tables, choose **FactInternetSales** and **FactInternetSalesReason**.

   You are choosing **FactInternetSales** because it contains the measures you want to use in the cube. You are choosing **FactInternetSalesReason** because it is the intermediate measure group, providing member association data that relates sales orders to sales reasons.

9. Choose measures for each fact table.

   To simplify your model, clear all the measures, and then select just **Sales Amount** and **Fact Internet Sales Count** at the bottom of the list. The **FactInternetSalesReason** only has one measure, so it is selected for you automatically.

10. In the dimension list, you should see **Dim Sales Reason** and **Dim Sales Orders**.

    In the Select New Dimensions page, the wizard prompts you to create a new dimension for **Fact Internet Sales Dimension**. You do not need this dimension, so you can clear it from the list.

11. Name the cube and click **Finish**.

Step 3: Define Many-to-Many relationship

1. In cube designer, click Dimension Usage tab. Notice that there is already a many-to-many relationship between **Dim Sales Reason** and **Fact Internet Sales**. Recall that the following icon indicates a many-to-many relationship.

2. Click on the intersection cell between **Dim Sales Reason** and **Fact Internet Sales**, and then click the button to open the Define Relationship dialog box.

   You can see that this dialog box is used to specify a many-to-many relationship. If you were adding dimensions that had a regular relationship instead, you would use this dialog box to change it to many-to-many.

3. Deploy the project to an Analysis Services multidimensional instance. In the next step, you will browse the cube in Excel to verify its behaviors.

Testing Many-to-Many

When you define a many-to-many relationship in a cube, testing is imperative to ensure queries return expected results. You should test the cube using the client application tool that will be used by end-users. In this next procedure, you will use Excel to connect to the cube and verify query results.

Browse the cube in Excel

1. Deploy the project and then browse the cube to confirm the aggregations are valid.

2. In Excel, click **Data** | **From Other Sources** | **From Analysis Services**. Enter the name of the server, choose the database and cube.

3. Create a PivotTable that uses the following:

   - **Sales Amount** as the Value
   - **Sales Reason Name** on Columns
   - **Sales Order Number** on Rows

4. Analyze the results. Because we are using sample data, the initial impression is that all sales orders have identical values. However, if you scroll down, you begin to see data variation.

   Part way down, you can find the sales amount and sales reasons for order number **SO5382**. Grand total of this particular order is **539.99**, and the purchase reasons attributed to this order include Promotion, Other and Price.

	A	B	C	D	E	F	G	H	I
1	Sales Amount	Column Labe							
2	Row Labels	Manufacturer	On Promotion	Other	Price	Quality	Review	Television	Grand Total
11130	SO53824				63.5				63.5
11131	SO53825		539.99	539.99	539.99				539.99
11132	SO53825		8.99	8.99	8.99				8.99
11133	SO53825		4.99	4.99	4.99				4.99
11134	SO53826		539.99	539.99	539.99				539.99
11135	SO53826		4.99	4.99	4.99				4.99
11136	SO53826		8.99	8.99	8.99				8.99

   Notice that the Sales Amount is correctly calculated for the order; it is **539.99** for the entire order. Although **539.99** is indicated for each reason, that value is not summed for all three reasons, erroneously inflating our grand total.

   Why put a sales amount under each sales reason in the first place? The answer is that it allows us to identify the amount of sales we can attribute to each reason.

5. Scroll to the bottom of the worksheet. It is now easy to see that Price is the most important reason for customer purchases, relative to other reasons as well as the grand total.

Column Label							
Manufacturer	On Promotion	Other	Price	Quality	Review	Television	Ad Grand Total
$ 5,998,122.10	$ 6,361,828.95	$ 248,483.34	$ 10,975,842.56	$ 5,549,896.77	$ 1,694,881.98	$ 27,475.82	$ 29,358,677.22

Tips for handling unexpected query results

1. Hide measures in the intermediate measure group, such as the count, that do not return meaningful results in a query. This prevents people from trying to use aggregations producing meaningless data. To hide a measure, set **Visibility** to **False** on the attribute in dimension designer.

2. Create perspectives to use a subset of measures and dimensions that support the analytical experience you want to provide. Possibly, a cube that contains many measure groups and dimensions do not work well together in all cases. By isolating the dimensions and measure groups that you intend to be used together, you ensure a more predictable outcome.

3. Always remember to deploy and reconnect after changing a model. In Excel, use the Refresh button on the PivotTable Analyze ribbon.

4. Avoid using linked measure groups in multiple many-to-many relationships, especially when those

relationships are in different cubes. Doing so can result in ambiguous aggregations. For more information, see Incorrect Amounts for Linked Measures in Cubes containing Many-to-Many Relationships.

Learn more

Use the following links to get additional information that helps you master the concepts.

The many-to-many Revolution 2.0

Tutorial: Many-to-many dimension example for SQL Server Analysis Services

See Also

Dimension Relationships
Install Sample Data and Projects for the Analysis Services Multidimensional Modeling Tutorial
Deploy Analysis Services Projects (SSDT)
Perspectives in Multidimensional Models

Measures and Measure Groups

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A cube includes *measures* in *measure groups*, business logic, plus a collection of dimensions that give context for evaluating the numerical data that a measure provides. Both measures and measure groups are an essential component of a cube. A cube cannot exist without at least one of each.

This topic describes Measures and Measure Groups. It also contains the following table, with links to procedural steps for creating and configuring measures and measure groups.

LINK	DESCRIPTION
Create Measures and Measure Groups in Multidimensional Models	Choose from one of several approaches for creating measures and measure groups.
Configure Measure Properties	If you used the Cube Wizard to start your cube, you might need to change the aggregation method, apply a data format, set the visibility of the measure in client applications, or possibly add a measure expression to manipulate the data before values are aggregated.
Configure Measure Group Properties	In a multidimensional model, a measure group equates to a fact table in the source data warehouse. Properties on a measure group allow you to specify caching behaviors, storage, and processing directives that operate collectively at the measure group level. Partition configuration is partly determined by the properties you set on measure group objects.
Use Aggregate Functions	Understand the aggregation methods that can be assigned to a measure.
Define Semiadditive Behavior	Semiadditive behavior refers to aggregations that are valid for some dimensions but not others. A common example is a bank account balance. You might want to aggregate balances by customer and region, but not time. For example, you would not want to add balances from the same account over consecutive days. To define semiadditive behavior, use the Add Business Intelligence Wizard.
Linked Measure Groups	Repurpose an existing measure group in other cubes in the same database or in different Analysis Services databases.

Measures

A measure represents a column that contains quantifiable data, usually numeric, that can be aggregated. Measures represent some aspect of organizational activity, expressed in monetary terms (such as revenue, margins, or costs) or as counts (inventory levels, number of employees, customers, or orders), or as a more complex calculation that incorporates business logic.

Every cube must have at least one measure, but most have many, sometimes numbering in the hundreds. Structurally, a measure is often mapped to a source column in a fact table, with the column providing the values

used to load the measure. Alternatively, you can also define a measure using MDX.

Measures are context-sensitive, operating on numeric data in a context that is determined by whichever dimension members happen to be included in the query. For example, a measure that calculates **Reseller Sales** will be backed by a **Sum** operator, and it will add the sales amounts for each dimension member included in the query. Whether the query specifies individual products, rolls up to a category, or is sliced by time or geography, the measure should produce an operation that is valid for the dimensions included in the query.

In this example **Reseller Sales** aggregates to various levels along the **Sales Territory** hierarchy.

Measure

Reseller Sales Amount	Column Labels				
Row Labels	CY 2005	CY 2006	CY 2007	CY 2008	Grand Total
Europe		$1,698,880.94	$5,632,816.55	$3,538,837.31	$10,870,534.80
North America	$8,065,435.31	$22,445,548.71	$25,722,421.91	$11,752,320.88	$67,985,726.81
Canada	$1,513,359.46	$4,822,999.20	$5,651,305.43	$2,390,261.51	$14,377,925.60
United States	$6,552,075.85	$17,622,549.51	$20,071,116.48	$9,362,059.37	$53,607,801.21
Central	$951,240.65	$2,625,639.72	$3,005,591.43	$1,323,536.38	$7,906,008.18
Northeast	$568,545.52	$2,443,901.73	$2,863,937.85	$1,056,456.93	$6,932,842.01
Northwest	$1,689,790.14	$3,471,099.54	$4,640,535.06	$2,633,651.25	$12,435,076.00
Southeast	$1,448,921.51	$2,815,903.10	$2,429,279.90	$1,173,311.72	$7,867,416.23
Southwest	$1,893,578.02	$6,266,005.43	$7,131,772.25	$3,175,103.09	$18,466,458.79
Pacific			$847,430.96	$746,904.41	$1,594,335.38
Grand Total	$8,065,435.31	$24,144,429.65	$32,202,669.43	$16,038,062.60	$80,450,596.98

Dimensions and Attributes

Measures produce valid results when the fact table that contains the numeric source data also contains pointers to dimension tables that are used in the query. Using the Reseller Sales example, if each row storing a sales amount also stores a pointer to a product table, a date table, or a sales territory table, then queries that include members from those dimension will resolve correctly.

What happens if the measure is unrelated to the dimensions used in query? Typically, Analysis Services will show the default measure, and the value will be the same for all members. In this example, **Internet Sales**, which measure direct sales placed by customers using the online catalog, has no relationship to the sales organization.

Internet Sales Amount	Column Labels					
Row Labels	CY 2010	CY 2011	CY 2012	CY 2013	CY 2014	Grand Total
AdventureWorks Cycle	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22
European Operations	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22
France	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22
Germany	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22
North America Operations	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22
Canadian Division	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22
USA Operations	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22
Central Division	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22
Northeast Division	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22
Northwest Division	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22
Southeast Division	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22
Southwest Division	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22
Pacific Operations	$43,421.04	$7,075,525.93	$5,842,485.20	$16,351,550.34	$45,694.72	$29,358,677.22

To minimize the chances of encountering these behaviors in a client application, you could build multiple cubes or perspectives within the same database, and ensure that each cube or perspective contains only related objects. The relationships you need to check are between the measure group (mapped to the fact table) and the dimensions.

Measure Groups

In a cube, measures are grouped by their underlying fact tables into measure groups. Measure groups are used to associate dimensions with measures. Measure groups are also used for measures that have distinct count as their

aggregation behavior. Placing each distinct count measure into its own measure group optimizes aggregation processing.

A simple MeasureGroup object is composed of basic information like the group name, storage mode, and processing mode. It also contains its constituent parts; the measures, dimensions, and partitions that form the composition of the measure group.

See Also

Cubes in Multidimensional Models
Create Measures and Measure Groups in Multidimensional Models

Create Measures and Measure Groups in Multidimensional Models

5/16/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A *measure* is an aggregation of numeric data values, such as sum, count, minimum, maximum, average, or a custom MDX expression that you create. A *measure group* is a container for one or more measures. All measures exist in a measure group, even if there is only one measure. A cube must have at least one measure and measure group.

This topic includes the following sections:

- Approaches for creating measures
- Components of a measure
- Modeling measures and measure groups on facts and fact tables
- Granularity of a measure group

Approaches for creating measures

Measures can be a static element of the cube, created at design time, always present whenever the cube is accessed. But you can also define a measure as a *calculated member* by using a MDX to provide a calculated value for a measure based on other measures in the cube. A calculated member can be scoped to session or user.

To create a measure or a measure group, use one of these approaches:

Cube Wizard	Run the Cube Wizard in SQL Server Data Tools (SSDT) to create a cube. In Solution Explorer, right-click **Cubes** and choose **New Cube**. See Multidimensional Modeling (Adventure Works Tutorial) if you need help with these steps. When you create a cube based on tables from an existing data warehouse, definitions for the measures and measure group materialize as part of the cube creation process. In the wizard, you'll choose which facts and fact tables to use as the basis for the measure and measure group objects in your cube.
New Measure dialog	Assuming the cube already exists in SQL Server Data Tools (SSDT), double-click the cube name in Solution Explorer to open it in Cube Designer. In the Measures pane, right-click the top node to create a new measure group, or new measures, by specifying a source table, column, and aggregation type. Using this approach requires that you choose the aggregation method from a fixed list of prebuilt functions. See Use Aggregate Functions for a discussion of the more commonly used aggregations.

Calculated member	Calculated members add flexibility and analysis capability to a cube in Analysis Services because you can control when and how they are created. Sometimes you only need a measure temporarily, for the duration of a user session, or in Management Studio as part of an investigation. In SQL Server Data Tools (SSDT), open the Calculations tab to create a new calculated member. Choose this approach when basing a measure on an MDX expression. See these topics for more information: Building Measures in MDX, Calculations, Calculations in Multidimensional Models and MDX Scripting Fundamentals (Analysis Services).
MDX or XMLA	In SQL Server Management Studio, you can execute MDX or XMLA to alter a database to include a new calculated measure. This approach is useful for ad hoc testing of data, after the solution is deployed to a server. See Document and Script an Analysis Services Database.

Components of a measure

A measure is an object with properties. In addition to its name, a measure must have an aggregation type and a source column or an expression used to load the measure with data. You can modify the measure definition by setting its properties.

source	Most measures come from numeric columns in fact tables in an external data warehouse, such as the Sales Amount column in the Internet Sales and Reseller Sales tables in the AdventureWorks data warehouse, but you can also create new measures based entirely on calculations that you define. Attribute columns from dimension tables can be used to define measures, but such measures are typically semiadditive or nonadditive in terms of their aggregation behavior. For more information about semiadditive behavior, see Define Semiadditive Behavior.
aggregation	By default, measures are summed along each dimension. However, the **AggregateFunction** property lets you modify this behavior. See Use Aggregate Functions for a list.
Properties	See Configure Measure Properties for additional property descriptions.

Modeling measures and measure groups on facts and fact tables

Before you run a wizard, it helps to understand the modeling principles behind measure definition.

Measures and measure groups are the multidimensional objects that represent facts and fact tables in an external data warehouse. In most cases, measures and measure groups will be based on objects in a data source view, which in turn are created from the underlying data warehouse.

The following diagram represents the **FactSalesQuota** fact table and the two dimension tables associated with it, **DimTime** and **DimEmployee**. In the Adventure Works sample cube, these tables are used as the basis of the

Sales Quotas measure group, and the Time and Employee dimensions.

The fact table contains two basic types of columns: attribute columns and measure columns.

- Attribute columns are used to create foreign key relationships to dimension tables, so that the quantifiable data in the measure columns can be organized by the data contained in the dimension tables. Attribute columns are also used to define the granularity of a fact table and its measure group.
- Measure columns define the measures contained by a measure group.

When you run the Cube Wizard, the foreign keys are filtered out. In the list of remaining columns to choose from, you will see measure columns, plus attribute columns that are not identified as a foreign key. In the **FactSalesQuote** example, the wizard will offer **CalendarYear** and **CalendarQuarter** in addition to **SalesAmountQuota**. Only the **SalesAmountQuota** measure column will result in a workable measure for your multidimensional model. The other date-based columns exist to qualify each quota amount. You should exclude the other columns, **CalendarYear** and **CalendarQuarter**, from the measure list in the Cube Wizard (or remove them from the measure group later in the designer).

The point to take away from this discussion is that not all columns offered by the wizard are useful as a measure. Rely on your understanding of the data and how it will be used when deciding which columns to use as measures. Remember that you can right-click a table in the data source view to explore the data, which can help you identify which columns to use as measures. See Explore Data in a Data Source View (Analysis Services) for more information.

> **NOTE**
>
> Not all measures are derived directly from a value stored in a column of the fact table. For example, the **Sales Person Count** measure defined in the **Sales Quota** measure group of the Adventure Works sample cube is actually based on the count of unique values (or distinct count) in the **EmployeeKey** column of the **FactSalesQuota** fact table.

Granularity of a measure group

Measure groups have an associated granularity that refers to the level of detail supported by a fact table. The granularity is set through the foreign key relationship to a dimension.

For example, the **FactSalesQuota** fact table has a foreign key relationship with the **DimEmployee** table, each

record in the **FactSalesQuota** table is related to a single employee, and thus the granularity of the measure group as viewed from the Employee dimension is at the individual employee level.

The granularity of a measure group can never be set finer than the lowest level of the dimension from which the measure group is viewed, but the granularity can be made coarser by using additional attributes. For example, the **FactSalesQuota** fact table uses three columns, **TimeKey**, **CalendarYear**, and **CalendarQuarter**, to establish the granularity of the relationship with the **DimTime** table. As a result, the granularity of the measure group as viewed from the Time dimension is by calendar quarter, and not by day, which is the lowest level of the Time dimension.

You can specify the granularity of a measure group with relation to a specific dimension by using the **Dimension Usage** tab of the Cube Designer. For more information about dimension relationships, see Dimension Relationships.

See Also

Cubes in Multidimensional Models
Measures and Measure Groups

Configure Measure Group Properties

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Measures groups have properties that enable you to define how measure groups function.

Measure Group Properties

Measure group properties determine behaviors for the entire measure group and set the default behaviors for certain properties of measures within a measure group.

PROPERTY	DEFINITION
AggregationPrefix	Applies to ROLAP storage. Assigns a common prefix to the indexed views in SQL Server, used to store aggregations for the partitions associated with this measure group.
DataAggregation	This property is reserved for future use and currently has no effect. Therefore, it is recommended that you do not modify this setting.
Description	You can use this property to document the measure group.
ErrorConfiguration	Configurable error handling settings for handling of duplicate keys, unknown keys, null keys, error limits, action upon error detection, and the error log file. See Error Configuration for Cube, Partition, and Dimension Processing (SSAS - Multidimensional).
EstimatedRows	Specifies the estimated number of rows in the fact table.
EstimatedSize	Specifies the estimated size (in bytes) of the measure group.
ID	Specifies the identifier of the object.
IgnoreUnrelatedDimensions	Determines whether unrelated dimensions are forced to their top level when members of dimensions that are unrelated to the measure group are included in a query. Default setting is **True**.
Name	Name of the measure. This property is read-only.
ProactiveCaching	Configurable error handling settings for handling of duplicate keys, unknown keys, null keys, error limits, action upon error detection, and the error log file.
ProcessingMode	Indicates whether indexing and aggregating should occur during or after processing. Options are Regular and LazyAggregations. LazyAggregations can be used to run aggregation as a background task.

PROPERTY	DEFINITION
ProcessingPriority	Determines the processing priority of the cube during background operations, such as lazy aggregations and indexing. The default value is **0**.
StorageLocation	The file system storage location for the measure group. If none is specified, the location is inherited from the cube that contains the measure group.
StorageMode	The storage mode for the measure group. Available values are MOLAP, ROLAP, or HOLAP.
Type	Specifies the type of the measure group.

Configure Measure Properties

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Measures have properties that enable you to define how the measures function and to control how the measures appear to users.

You can set properties in SQL Server Data Tools (SSDT) when creating or editing a cube or measure. You can also set them programmatically, using MDX or AMO. See Create Measures and Measure Groups in Multidimensional Models or CREATE MEMBER Statement (MDX) or Programming AMO OLAP Basic Objects for details.

Measure Properties

Measures inherit certain properties from the measure group of which they are a member, unless those properties are overridden at the measure level. Measure properties determine how a measure is aggregated, its data type, the name that is displayed to the user, the display folder in which the measure will appear, its format string, any measure expression, the underlying source column, and its visibility to users.

PROPERTY	DEFINITION
AggregateFunction	Required. Determines how measures are aggregated. **Sum** is the default aggregation. For more information, see Use Aggregate Functions for a description of each function.
DataType	Required. Specifies the data type of the column in the underlying fact table to which the measure is bound. This value is inherited from the source column by default.
Description	Provides a description of the measure, which may be exposed in client applications.
DisplayFolder	Specifies the folder in which the measure will appear when users connect to the cube. When a cube has many measures, you can use display folders to categorize the measures and improve the user browsing experience.
FormatString	You can select the format that is used to display measure values to users by using the **FormatString** property of the measure. Although a list of display formats is provided in SQL Server Data Tools (SSDT), you can specify many additional formats that are not in the list. You can specify any named or user-defined format that is valid in Microsoft Visual Basic.
ID	Required. Displays the unique identifier (ID) of the measure. This property is read-only.

PROPERTY	DEFINITION
MeasureExpression	Specifies a constrained MDX expression defining the value of the measure. The expression is evaluated at the leaf level before being aggregated, and allows for weighting of a value. For example, in currency conversion where a sales amount is weighted by the exchange rate.
Name	Required. Specifies the name of the measure.
Source	Required. Specifies the column in the data source view to which the measure is bound. See Data Sources and Bindings (SSAS Multidimensional).
Visible	Determines the visibility of the measure in client applications.

See Also

Configure Measure Group Properties
Modifying Measures

Use Aggregate Functions

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When a dimension is used to slice a measure, the measure is summarized along the hierarchies contained in that dimension. The summation behavior depends on the aggregate function specified for the measure. For most measures containing numeric data, the aggregate function is **Sum**. The value of the measure will sum to different amounts depending on which level of the hierarchy is active.

In Analysis Services, every measure that you create is backed by an aggregation function that determines the measure's operation. Predefined aggregation types include **Sum**, **Min**, **Max**, **Count**, **Distinct Count**, and several other more specialized functions. Alternatively, if you require aggregations based on complex or custom formulas, you can build an MDX calculation in lieu of using a prebuilt aggregation function. For example, if you want to define a measure for a percentage value, you would do that in MDX, using a calculated measure. See CREATE MEMBER Statement (MDX).

Measures that are created via the Cube Wizard are assigned an aggregation type as part of the measure definition. The aggregation type is always **Sum**, assuming the source column contains numeric data. **Sum** is assigned regardless of the source column's data type. For example, if you used the Cube Wizard to create measures, and you pulled in all columns from a fact table, you will notice that all of the resulting measures have an aggregation of **Sum**, even if the source is a date time column. Always review the pre-assigned aggregation methods for measures created via the wizard to make sure the aggregation function is suitable.

You can assign or change the aggregation method in the either the cube definition, via SQL Server Data Tools - Business Intelligence, or via MDX. See Create Measures and Measure Groups in Multidimensional Models or Aggregate (MDX) for further instructions.

Aggregate Functions

Analysis Services provides functions to aggregate measures along the dimensions that are contained in measure groups. The *additivity* of an aggregation function determines how the measure is aggregated across all the dimensions in the cube. Aggregation functions fall into three levels of additivity:

Additive
An additive measure, also called a fully additive measure, can be aggregated along all the dimensions that are included in the measure group that contains the measure, without restriction.

Semiadditive
A semiadditive measure can be aggregated along some, but not all, dimensions that are included in the measure group that contains the measure. For example, a measure that represents the quantity available for inventory can be aggregated along a geography dimension to produce a total quantity available for all warehouses, but the measure cannot be aggregated along a time dimension because the measure represents a periodic snapshot of quantities available. Aggregating such a measure along a time dimension would produce incorrect results. See Define Semiadditive Behavior for details.

Nonadditive
A nonadditive measure cannot be aggregated along any dimension in the measure group that contains the measure. Instead, the measure must be individually calculated for each cell in the cube that represents the measure. For example, a calculated measure that returns a percentage, such as profit margin, cannot be aggregated from the percentage values of child members in any dimension.

The following table lists the aggregation functions in Analysis Services, and describes both the additivity and expected output of the function.

AGGREGATION FUNCTION	ADDITIVITY	RETURNED VALUE
Sum	Additive	Calculates the sum of values for all child members. This is the default aggregation function.
Count	Additive	Retrieves the count of all child members.
Min	Semiadditive	Retrieves the lowest value for all child members.
Max	Semiadditive	Retrieves the highest value for all child members.
DistinctCount	Nonadditive	Retrieves the count of all unique child members. For more details, see About Distinct Count Measures in the next section.
None	Nonadditive	No aggregation is performed, and all values for leaf and nonleaf members in a dimension are supplied directly from the fact table for the measure group that contains the measure. If no value can be read from the fact table for a member, the value for that member is set to null.
ByAccount	Semiadditive	Calculates the aggregation according to the aggregation function assigned to the account type for a member in an account dimension. If no account type dimension exists in the measure group, treated as the **None** aggregation function. For more information about account dimensions, see Create a Finance Account of parent-child type Dimension.
AverageOfChildren	Semiadditive	Calculates the average of values for all non-empty child members.
FirstChild	Semiadditive	Retrieves the value of the first child member.
LastChild	Semiadditive	Retrieves the value of the last child member.
FirstNonEmpty	Semiadditive	Retrieves the value of the first non-empty child member.

AGGREGATION FUNCTION	ADDITIVITY	RETURNED VALUE
LastNonEmpty	Semiadditive	Retrieves the value of the last non-empty child member.

About Distinct Count Measures

A measure with an **Aggregate Function** property value of **Distinct Count** is called a distinct count measure. A distinct count measure can be used to count occurrences of a dimension's lowest-level members in the fact table. Because the count is distinct, if a member occurs multiple times, it is counted only once. A distinct count measure is always placed in a dedicated measure group. Putting a distinct count measure into its own measure group is a best practice that has been built into the designer as a performance optimization technique.

Distinct count measures are commonly used to determine for each member of a dimension how many distinct, lowest-level members of another dimension share rows in the fact table. For example, in a Sales cube, for each customer and customer group, how many distinct products were purchased? (That is, for each member of the Customers dimension, how many distinct, lowest-level members of the Products dimension share rows in the fact table?) Or, for example, in an Internet Site Visits cube, for each site visitor and site visitor group, how many distinct pages on the Internet site were visited? (That is, for each member of the Site Visitors dimension, how many distinct, lowest-level members of the Pages dimension share rows in the fact table?) In each of these examples, the second dimension's lowest-level members are counted by a distinct count measure.

This kind of analysis need not be limited to two dimensions. In fact, a distinct count measure can be separated and sliced by any combination of dimensions in the cube, including the dimension that contains the counted members.

A distinct count measure that counts members is based on a foreign key column in the fact table. (That is, the measure's **Source Column** property identifies this column.) This column joins the dimension table column that identifies the members counted by the distinct count measure.

See Also

Measures and Measure Groups
MDX Function Reference (MDX)
Define Semiadditive Behavior

Define Semiadditive Behavior

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Semiadditive measures, which do not uniformly aggregate across all dimensions, are very common in many business scenarios. Every cube that is based on snapshots of balances over time exhibits this problem. You can find these snapshots in applications dealing with securities, account balances, budgeting, human resources, insurance policies and claims, and many other business domains.

Add semiadditive behavior to a cube to define an aggregation method for individual measures or members of the account type attribute. If the cube contains an account dimension, you can automatically set semiadditive behavior based on the account type.

To add semiadditive behavior, open a cube in Cube Designer and choose **Add Business Intelligence** from the Cube menu. In the Business Intelligence Wizard, select the **Define semiadditive behavior** option on the **Choose Enhancement** page. This wizard then guides you through the steps of identifying which measures have semiadditive behavior.

With the exception of LastChild which is available in the standard edition, semi-additive behaviors are only available in the business intelligence or enterprise editions.

Define Semiadditive Behavior

On the **Define Semiadditive Behavior** page of the wizard, you select how to define semiadditivity by selecting one of the following options:

Turn off semiadditive behavior
Removes semiadditive behavior from a cube in which semiadditive behavior was previously defined. This selection resets a measure to **SUM** if it is set to any of the following aggregation function types:

- By Account
- Average of Children
- First Child
- Last Child
- Last Nonempty Child
- First Nonempty Child
- None

This option does not change measures with a regular aggregation function: **Sum**, **Min**, **Max**, **Count**, or **Distinct****Count**.

The wizard has detected the 'Account" account dimension, which contains semiadditive members. The server will aggregate members of this dimension according to the semiadditive behavior specified for each account type.
Causes the system to set all measures from a measure group dimensioned by an Account type dimension to the By Account aggregation function and the server will aggregate members of the dimension according to the semiadditive behavior specified for each account type.

> **NOTE**
> This option is selected by default if the wizard detects an Account type dimension.

Define semiadditive behavior for individual measures

Selects the semiadditive behavior of each measure individually. The default setting is **SUM** (fully additive).

> **NOTE**
> This option is selected by default if the wizard does not detect an Account type dimension.

For each measure, you can select from the types of semiadditive functionality described in the following table.

SEMIADDITIVE FUNCTION	DESCRIPTION
Average of Children	The aggregation of a member is the average of its children.
ByAccount	The system reads the semiadditive behavior specified for the account type.
Count	The aggregation is a count of members.
Distinct Count	The aggregation is a count of unique members.
First Child	The member value is evaluated as the value of its first child along the time dimension.
FirstNonEmpty	The member value is evaluated as the value of its first child along the time dimension that contains data.
LastChild	The member value is evaluated as the value of its last child along the time dimension.
LastNonEmpty	The member value is evaluated as the value of its last child along the time dimension that contains data.
Max	The standard maximum aggregation function is applied.
Min	The standard minimum aggregation function is applied.
None	No aggregation is applied.
Sum	The standard summation function is applied.

Any existing semiadditive behavior is overwritten when you complete the wizard.

Linked Measure Groups

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A linked measure group is based on another measure group in a different cube within the same database or a different Analysis Services database. You might use a linked measure group if you want to reuse a set of measures, and the corresponding data values, in multiple cubes.

Microsoft recommends that the original and linked measure groups reside in solutions that run on the same server. Linking to a measure group on a remote server is scheduled for deprecation in a future release (see Deprecated Analysis Services Features in SQL Server 2016).

> **IMPORTANT**
> Linked measure groups are read-only. To pick up the latest changes, you must delete and recreate all linked measure groups based on the modified source object. For this reason, copy and pasting measure groups between projects is an alternative approach that you should consider in case future modifications to the measure group are required.

Usage Limitations

As noted previously, an important constraint to using linked measures is an inability to customize a linked measure directly. Modifications to the data type, format, data binding, and visibility, as well as membership of the items in the measure group itself, are all changes that must be made in the original measure group.

Operationally, linked measure groups are identical to other measure groups when accessed by client applications, and are queried in the same manner as other measure groups.

When you query a cube that contains a linked measure group, the link is established and resolved during the first calculation pass of the destination cube. Because of this behavior, any calculations that are stored in the linked measure group cannot be resolved before the query is evaluated. In other words, calculated measures and calculated cells must be recreated in the destination cube rather than inherited from the source cube.

The following list summarizes usage limitations.

- You cannot create a linked measure group from another linked measure group.
- You cannot add or remove measures in a linked measure group. Membership is defined only in the original measure group.
- Writeback is not supported in linked measure groups.
- Linked measure groups cannot be used in multiple many-to-many relationships, especially when those relationships are in different cubes. Doing so can result in ambiguous aggregations. For more information, see Incorrect Amounts for Linked Measures in Cubes containing Many-to-Many Relationships.

The measures contained in a linked measure group can be directly organized only along linked dimensions retrieved from the same Analysis Services database. However, you can use calculated members to relate information from linked measure groups to the other, non-linked dimensions in your cube. You can also use an indirect relationship, such as a reference or many-to-many relationship, to relate non-linked dimensions to a linked measure group.

Create or modify a linked measure

Use SQL Server Data Tools to create a linked measure group.

1. Finalize any modifications to the original measure group now, in the source cube, so that you don't have to recreate the linked measure groups later in subsequent cubes. You can rename a linked object, but you cannot change any other properties.
2. In Solution Explorer, double-click the cube to which you are adding the linked measure group. This step opens the cube in Cube Designer.
3. In Cube Designer, in either the Measures pane or Dimensions pane, right-click anywhere in either pane, then select **New Linked Object**. This starts the Linked Object Wizard.
4. On the first page, specify the data source. This step establishes the location of the original measure group. The default is the current cube in the current database, but you can also choose a different Analysis Services database.
5. On the next page, choose the measure group or dimension you want to link. Dimensions and Cube objects, such as measure groups, are listed separately. Only those objects not already present in the current cube are available.
6. Click **Finish** to create the linked object. Linked objects appear in the Measures and Dimensions pane, indicated by the link icon.

Secure a linked measure

After the link has been defined, access to the measures in a linked measure group, is managed in the same manner as access to other measure groups. A linked object appears alongside its non-linked counterparts in Role Designer. For more information on managing security for a measure group, see Grant cube or model permissions (Analysis Services).

In order to define or use a linked measure group, the Windows service account for the Analysis Services instance must belong to an Analysis Services database role that has **ReadDefinition** and **Read** access rights on the source Analysis Services instance to the source cube and measure group, or must belong to the Analysis Services Administrators role for the source Analysis Services instance.

See Also

Define Linked Dimensions

Partitions in Multidimensional Models

12/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Analysis Services, a *partition* provides the physical storage of fact data loaded into a measure group. A single partition is created for each measure group automatically, but it is common to create additional partitions that further segment the data, resulting in more efficient processing and faster query performance.

Processing is more efficient because partitions can be processed independently and in parallel, on one or more servers. Queries run faster because each partition can be configured to have storage modes and aggregation optimizations that result in shorter response times. For example, choosing MOLAP storage for partitions containing newer data is typically faster than ROLAP. Likewise, if you partition by date, partitions containing newer data can have more optimizations than partitions containing older data that is accessed less frequently. Note that varying storage and aggregation design by partition will have a negative impact on future merge operations. Be sure to consider whether merging is an essential component of your partition management strategy before optimizing individual partitions.

> **NOTE**
>
> Support for multiple partitions is available in the business intelligence edition and enterprise edition. The standard edition does not support multiple partitions. For more information, see Features Supported by the Editions of SQL Server 2016.

Important considerations when designing a partitioning strategy

The integrity of a cube's data relies on the data being distributed among the partitions of the cube such that no data is duplicated among the partitions. When data is summarized from the partitions, any data elements that are present in more than one partition will be summarized as if they were different data elements. This can result in incorrect summaries and erroneous data provided to the end user. For example, if a sales transaction for Product X is duplicated in the fact tables for two partitions, summaries of Product X sales can include a double accounting of the duplicated transaction.

Partitions can be merged; you can use this feature in your overall storage and data update strategy. Partitions can be merged only if they have the same storage mode and aggregation design. To create partitions that are candidates for later merging, you can copy the aggregation design of another partition when you create partitions. You can also edit a partition after it has been created to copy the aggregation design of another partition. Merging partitions must also be performed carefully to avoid duplication of data in the resulting partition, which can cause cube data to be inaccurate.

Local Partitions

Local partitions are partitions that are defined, processed, and stored on one server. If you have large measure groups in a cube, you might want to partition them out so that processing occurs in parallel across the partitions. The advantage is that parallel processing provides faster execution. Because one partition processing job does not have to finish before another starts, they can run in parallel. For more information, see Create and Manage a Local Partition (Analysis Services).

Remote Partitions

Remote partitions are partitions that are defined on one server, but are processed and stored on another. If you

want to distribute storage of your data and metadata across multiple servers, use remote partitions. Ordinarily, when you transition from development to production, the size of data under analysis grows several times over. With such large chunks of data, one possible alternative is to distribute that data over multiple computers. This is not just because one computer cannot hold all the data, but because you will want more than one computer processing the data in parallel. For more information, see Create and Manage a Remote Partition (Analysis Services).

Aggregations

Aggregations are precalculated summaries of cube data that help enable Analysis Services to provide rapid query responses. You can control the number of aggregations created for a measure group by setting limits on storage, performance gains, or arbitrarily stopping the aggregation build process after it has been running for a while. More aggregations are not necessarily better. Every new aggregation comes at a cost, both in terms of disk space and processing time. We recommend creating aggregations for a thirty percent performance gain, and then raising the number only if testing or experience warrants it.For more information, see Designing Aggregations (Analysis Services - Multidimensional).

Partition Merging and Editing

If two partitions use the same aggregation design, you can merge those two partitions into one. For example, if you have an inventory dimension that is partitioned by month, then at the end of each calendar month, you can merge that month partition with the existing year-to-date partition. This way, the current month partition can be processed and analyzed quickly, while the rest of the year in months only has to be reprocessed when merged. That reprocess requires longer processing time, and can be run less frequently. For more information about managing the partition merging process, see Merge Partitions in Analysis Services (SSAS - Multidimensional). To edit cube partitions by using the **Partitions** tab in Cube Designer, see Edit or Delete Partitions (Analyisis Services - Multidimensional).

Related Topics

TOPIC	DESCRIPTION
Create and Manage a Local Partition (Analysis Services)	Contains information about how to partition data using filters or different fact tables without duplicating data.
Set Partition Storage (Analysis Services - Multidimensional)	Describes how to configure storage for partitions.
Edit or Delete Partitions (Analyisis Services - Multidimensional)	Describes how to view and edit partitions.
Merge Partitions in Analysis Services (SSAS - Multidimensional)	Contains information about how to merge partitions that have different fact tables or different data slices without duplicating data.
Set Partition Writeback	Provides instructions on write-enabling a partition.
Create and Manage a Remote Partition (Analysis Services)	Describes how to create and manage a remote partition.

Create and Manage a Local Partition (Analysis Services)

5/16/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can create additional partitions for a measure group to improve processing performance. Having multiple partitions allows you to allocate fact data across a corresponding number of physical data files on local as well as remote servers. In Analysis Services, partitions can be processed independently and in parallel, giving you more control over processing workloads on the server.

Partitions can be created in SQL Server Data Tools during model design, or after the solution is deployed using SQL Server Management Studio or XMLA. We recommend that you choose one approach only. If you alternate between tools, you might find that changes made to a deployed database in SQL Server Management Studio are overwritten when you subsequently redeploy the solution from SQL Server Data Tools.

Before you start

Check whether you have either the business intelligence edition or enterprise edition. Standard edition does not support multiple partitions. To check the edition, right-click the server node in SQL Server Management Studio and choose **Reports** | **General**. For more information about feature availability, see Features Supported by the Editions of SQL Server 2016.

From the outset, it's important to understand that partitions must share the same aggregation design if you want to merge them later. Partitions can be merged only if they have identical aggregation designs and storage modes.

> **TIP**
> Explore the data in Data Source View (DSV) to understand the range and depth of the data you are partitioning. For example, if partitioning by date, you can sort on a date column to determine the upper and lower bounds of each partition.

Choose an approach

The most important consideration when creating partitions is to segment the data so that there are no duplicate rows. Data must be stored in one, and only one, partition to avoid double counting any rows. As such, it's common to partition by DATE so that you can define clear boundaries between each partition.

You can use either technique to distribute the fact data across multiple partitions. The following techniques can be used to segment the data.

TECHNIQUE	RECOMMENDATIONS

TECHNIQUE	RECOMMENDATIONS
Use SQL queries to segment fact data	Partitions can be sourced from SQL queries. During processing, the SQL query is to retrieve the data. The query's WHERE clause provides the filter that segments the data for each partition. Analysis Services generates the query for you, but you must fill in the WHERE clause to properly segment the data. The primary advantage of this approach is the ease with which you can partition data from a single source table. If all of the source data originates from a large fact table, you can build queries that filter that data into discrete partitions, without having to create additional data structures in the Data Source View (DSV). One disadvantage is that using queries will break the binding between the partition and the DSV. If you later update the DSV in the Analysis Services project, such as adding columns to the fact table, you must manually edit the queries for each partition to include the new column. The second approach, discussed next, does not have this disadvantage.
Use tables in the DSV to segment fact data	You can bind a partition to a table, named query, or view in the DSV. As the basis of a partition, all three are functionally equivalent. The entire table, named query, or view provides all of the data to a single partition. Using a table, view, or named query places all of the data selection logic in the DSV, which can be easier to manage and maintain over time. An important advantage to this approach is that table bindings are preserved. If you update the source table later, you do not have to modify the partitions that use it. Secondly, all of the tables, named queries and views exist in a common work space, making updates more convenient than having to open and edit partition queries individually.

Option 1: Filter a Fact Table for Multiple Partitions

To create multiple partitions, you begin by modifying the **Source** property of the default partition. By default, a measure group is created using a single partition that is bound to a single table in the DSV. Before you can add more partitions, you must first modify the original partition to contain just a portion of the fact data. You can then proceed to create additional partitions for storing the remainder of the data.

Construct your filters such that data is not duplicated among the partitions. A partition's filter specifies which data in the fact table is used in the partition. It is important that the filters for all partitions in a cube extract mutually exclusive datasets from the fact table. The same fact data might be double-counted if it appears in multiple partitions.

1. In SQL Server Data Tools, in Solution Explorer, double-click the cube to open it in Cube Designer, and then click the **Partitions** tab.

2. Expand the measure group for which are adding partitions. By default, each measure group has one partition, bound to a fact table in the DSV.

3. In the Source column, click the browse (. .) button to open the Partition Source dialog box.

4. In Binding Type, select **Query Binding**. The SQL query that selects the data appears automatically.

5. In the WHERE clause at the bottom, add a filter that segments data for this partition.

   Examples of WHERE clause syntax include `WHERE OrderDateKey >= '20060101'` or `WHERE OrderDateKey BETWEEN '20051001' AND '20051201'`. For other examples, see WHERE (Transact-SQL).

   Notice that the following filters are mutually exclusive within each set:

Set 1:	"SaleYear" = 2012 "SaleYear" = 2013
Set 2:	"Continent" = 'NorthAmerica' "Continent" = 'Europe' "Continent" = 'SouthAmerica'
Set 3:	"Country" = 'USA' "Country" = 'Mexico' ("Country" <> 'USA' AND "Country" <> 'Mexico')

6. Click **Check** to check for syntax errors, and then click **OK**.

7. Repeat the previous steps to create the remaining partitions, modifying the WHERE clause each time to select the next data slice.

8. Deploy the solution or process the partition to load the data. Be sure to process all partitions.

9. Browse the cube to verify the correct data is returned.

After you have a measure group that uses multiple measure groups, you can create additional partitions in SQL Server Management Studio. Under a measure group, right-click the Partitions folder and select **New Partitions** to start the wizard.

> **NOTE**
>
> Instead of filtering data in a partition, you can use the same query to create a name query in the DSV, and then base the partition on the named query.

Option 2: Use Tables, Views, or Named Queries

If the DSV already organizes facts into individual tables (for example, by year or quarter), you can create partitions based on an individual table, where each partition has its own data source table. This is essentially how measure groups are partitioned by default but in the case of multiple partitions, you break the original partition into multiple partitions, and map each new partition to the data source table providing the data.

Views and named queries are functional equivalent to tables, in that all three objects are defined in the DSV and bound to a partition using the Table Binding option in the Partition Source dialog box. You can create a view or named query to generate the data segment needed for each partition. For more information, see Define Named Queries in a Data Source View (Analysis Services).

> **IMPORTANT**
> When you create mutually exclusive named queries for partitions in a DSV, ensure that the combined data for the partitions includes all data from a measure group that you want to include in the cube. Make sure that you do not leave a default partition based on the entire table for the measure group, or else the query based partitions will overlap the query based on the complete table.

1. Create one or more named queries to use as the partition source. For more information, see Define Named Queries in a Data Source View (Analysis Services).

   The named query must be based on the fact table associated with the measure group. For example, if you are partitioning the FactInternetSales measure group, the named queries in the DSV must specify the FactInternetSales table in the FROM statement.

2. In SQL Server Data Tools, in Solution Explorer, double-click the cube to open it in Cube Designer, and then click the **Partitions** tab.
3. Expand the measure group for which are adding partitions.
4. Click **New Partition** to start the Partition Wizard. If you created the named queries using the fact table bound to the measure group, you should see each of the named queries you created in the previous step.
5. In Specify Source Information, choose one of the named queries you created in a previous step. If you do not see any named queries, go back to the DSV and check the FROM statement.
6. Click **Next** to accept the default values for each subsequent page.
7. On the last page, Completing the Wizard, give the partition a descriptive name.
8. Click **Finish**.
9. Repeat the previous steps to create the remaining partitions, choosing a different named query each time to select the next data slice.
10. Deploy the solution or process the partition to load the data. Be sure to process all partitions.
11. Browse the cube to verify the correct data is returned.

Next Step

When you create mutually exclusive queries for partitions, ensure that the combined partition data includes all data you want to include in the cube.

As a final step, you normally want to remove the default partition that was based on the table itself (if it still exists), or else the query based partitions will overlap the query based on the complete table.

See Also

Partitions (Analysis Services - Multidimensional Data)
Remote Partitions
Merge Partitions in Analysis Services (SSAS - Multidimensional)

Create and Manage a Remote Partition (Analysis Services)

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When partitioning a measure group, you can configure a secondary database on a remote Analysis Services instance as partition storage.

Remote partitions for a cube (called the master database) are stored in a dedicated Analysis Services database on the remote instance of Analysis Services (called the secondary database).

A dedicated secondary database can store remote partitions for one and only one master database, but the master database can use multiple secondary databases, as long as all the secondary databases are on the same remote instance of Analysis Services. Dimensions in a database dedicated to remote partitions are created as linked dimensions.

Prerequisites

Before you create a remote partition, the following conditions must be met:

- You must have a second Analysis Services instance and dedicated database to store the partitions. The secondary database is single-purpose; it provides storage of remote partitions for a master database.
- Both server instances must be the same version. Both databases should be the same functional level.
- Both instances must be configured for TCP connections. Analysis Services does not support creation of remote partitions by using the HTTP protocol.
- Firewall settings on both computers must be set to accept outside connections. For information about setting the firewall, see Configure the Windows Firewall to Allow Analysis Services Access.
- The service account for the instance running the master database must have administrative access to the remote instance of Analysis Services. If the service account changes, you must update permissions on both the server and database.
- You must be an Analysis Services administrator on both computers.
- You must ensure your disaster recovery plan accommodates backup and restore of the remote partitions. Using remote partitions can complicate backup and restore operations. Be sure to test your plan thoroughly to be sure you can restore the necessary data.

Configure remote partitions

Two separate computers that are running an instance of SQL Server Analysis Services are each required to create a remote partition arrangement that designates one computer as the master server and the other computer as the subordinate server.

The following procedure assumes that you have two server instances, with a cube database deployed on the master server. For the purposes of this procedure, the cube database is referred to as db-master. The storage database containing remote partitions is referred to as db-storage.

You will use both SQL Server Management Studio and SQL Server Data Tools (SSDT) to complete this

procedure.

> **NOTE**
> Remote partitions can be merged only with other remote partitions. If you are using a combination of local and remote partitions, an alternative approach is to create new partitions that include the combined data, deleting the partitions you no longer use.

Specify valid server names for cube deployment (in SSDT)

1. On the master server: In Solution Explorer, right-click the solution name and select **Properties**. In the **Properties** dialog box, click **Configuration Properties**, then click **Deployment**, and then click **Server** and set the master server's name.

2. On the subordinate server: In Solution Explorer, right-click the solution name and select **Properties**. In the **Properties** dialog box, click **Configuration Properties**, then click **Deployment**, and then click **Server** and set the subordinate server's name.

Create and deploy a secondary database (in SSDT)

1. On the subordinate server: Create a new Analysis Services project for the storage database.

2. On the subordinate server: In Solution Explorer, create a new data source pointing to the cube database, db-master. Use the provider **Native OLE DB\Microsoft OLE DB Provider for Analysis Services 11.0**.

3. On the subordinate server: Deploy the solution.

Enable features (in SSMS)

1. On the subordinate server: In SQL Server Management Studio, right-click your connected Analysis Services instance in Object Explorer and select **Properties**. Set both **Feature\LinkToOtherInstanceEnabled** and **Feature\LinkFromOtherInstanceEnabled** to **True**.

2. On the subordinate server: Restart the server by right-clicking the server name in Object Explorer and selecting **Restart**.

3. On the master server: In SQL Server Management Studio, right-click your connected Analysis Services instance in Object Explorer and select **Properties**. Set both **Feature\LinkToOtherInstanceEnabled** and **Feature\LinkFromOtherInstanceEnabled** to **True**.

4. On the master server: To restart the server, right-click the server name in Object Explorer and select **Restart**.

Set the MasterDataSourceID database property on the remote server (in SSMS)

1. On the subordinate server: Right-click the storage database, db-storage, point to **Script Database as | ALTER To | New Query Editor Window**.

2. Add **MasterDataSourceID** to the XMLA, and then specify the cube database, db-master, ID as the value. The XMLA should look similar to the following.

```
<Alter ObjectExpansion="ExpandFull" xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
<Object>
   <DatabaseID>DB-Storage</DatabaseID>
</Object>
<ObjectDefinition>
   <Database xmlns:xsd="http://www.w3.org/2001/XMLSchema" 400"
      <ID>DB-Storage</ID>
      <Name>DB-StorageB</Name>
      <ddl200:CompatibilityLevel>1100</ddl200:CompatibilityLevel>
      <Language>1033</Language>
      <Collation>Latin1_General_CI_AS</Collation>
      <DataSourceImpersonationInfo>
<ImpersonationMode>ImpersonateAccount</ImpersonationMode>
         <Account>*********</Account>
      </DataSourceImpersonationInfo>
      <MasterDataSourceID>DB-Master</MasterDataSourceID>
   </Database>
</ObjectDefinition>
</Alter>
```

3. Press F5 to execute the script.

Set up the remote partition (in SSDT)

1. On the master server: Open the cube in Cube Designer and click **Partitions** tab. Expand the measure group. Click **New Partition** if the measure group is already configured for multiple partitions, or click the browse (. .) button in the Source column to edit the existing partition.

2. In the Partition Wizard, in **Specify Source Information**, select the original Data Source View and fact table.

3. If using a query binding, provide a WHERE clause that segments the data for the new partition you are creating.

4. In **Processing and Storage Locations**, under **Processing Location**, choose **Remote Analysis Services data source** and click **New** to create a new data source that points to your subordinate database, db-storage.

   > **NOTE**
   > If you get an error indicating the data source does not exist in the collection, you must open the project of the storage database, db-storage, and create a data source that points to the master database, db-master.

5. On the master server: Right-click the cube name in Solution Explorer, select **Process** and fully process the cube.

Administering remote partitions

Analysis Services supports both parallel and sequential processing of remote partitions. The master database, where the partitions were defined, coordinates the transactions among all the instances that participate in processing the partitions of a cube. Processing reports are then sent to all instances that processed a partition.

A cube that contains remote partitions can be administered together with its partitions on a single instance of Analysis Services. However, the metadata for the remote partition can be viewed and updated only on the instance of Analysis Services where the partition and its parent cube were defined. The remote partition cannot be viewed or updated on the remote instance of Analysis Services.

NOTE

Although databases dedicated to storage of remote partitions are not exposed to schema rowsets, applications that use Analysis Management Objects (AMO) can still discover a dedicated database by using the XML for Analysis Discover command. Any CREATE or DELETE command that is sent directly to a dedicated database by using a TCP or HTTP client will succeed, but the server will return a warning indicating that the action may damage this closely managed database.

See Also

Partitions (Analysis Services - Multidimensional Data)

Change a partition source to use a different fact table

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a partition for a cube, you can choose to use a different fact table. Different tables may be from a single data source view, from different data source views, or from different data sources. A data source view may also contain different tables from more than one data source.

All fact tables and dimensions for a cube's partitions must have the same structure as the fact table and dimensions of the cube. For example, different fact tables can have the same structure but contain data for different years or different product lines.

You specify a fact table on the **Specify Source Information** page of the Partition Wizard. On this page, in the Partition Source group next to **Look in**, specify a data source or data source view in which to look. Next, click **Find Tables**, and then, under **Available Tables**, select the table you want to use.

When you use different fact tables, ensure that no data is duplicated among the partitions. For example, if one fact table contains transactions for 2012 only, and another fact table contains transactions for 2013 only, these tables contain independent data. Similarly, fact tables for distinct product lines or distinct geographical areas are independent.

It is possible, but not recommended, to use different fact tables that contain duplicated data. In this case, you must use filters in the partitions to ensure that data used by one partition is not used by any other partition. For more information, see Create and Manage a Local Partition (Analysis Services).

See Also

Create and Manage a Local Partition (Analysis Services)

Designing Aggregations (Analysis Services - Multidimensional)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Aggregations are precalculated summaries of cube data that help enable Analysis Services to provide rapid query responses.

To set storage options and design aggregations for a partition, use the Aggregation Design Wizard. The wizard operates on a single partition of a measure group at a time so that you can select different options and designs for each partition. The wizard takes you through steps to configure storage and design aggregation for a partition. For more information about configuring storage, see.

Select a method for controlling the number of aggregations the wizard will design, and then let the wizard design the aggregations.

The goal is to design the optimal number of aggregations. This number should not only provide satisfactory response time, but also prevent excessive partition size. A greater number of aggregations produces faster response times but it also requires more storage space and may take longer to compute. Moreover, as the wizard designs more and more aggregations, earlier aggregations produce considerably larger performance gains than later aggregations. Reduction in less useful aggregations increases performance as well. You can control the number of aggregations the wizard designs by one of the following methods available in the wizard:

- Specify a storage space limit for the aggregations.
- Specify a performance gain limit.
- Stop the wizard manually when the displayed performance versus size curve starts to level off at an acceptable performance gain.
- Choose not to design aggregations.

To design storage, the wizard must be able to connect to Analysis Services on the target server. The wizard will display an error message if Analysis Services is not running on the target server or if the storage design process is otherwise unable to connect to the target server.

The final step of the wizard allows you to process or defer processing. Processing creates the aggregations you design with the wizard, while deferring processing saves the designed aggregations for future processing, thus allowing design activities to continue without processing. Depending on the size of the partition, processing may take considerable time. If you choose, you can interrupt processing a partition.

See Also

Aggregations and Aggregation Designs

Edit or Delete Partitions (Analyisis Services - Multidimensional)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Cube partitions are modified using the **Partitions** tab in Cube Designer in SQL Server Data Tools. The **Partitions** tab lists the partitions for all of the measure groups in a cube. It also lists the writeback partitions that have writeback enabled.

To edit the partitions for any measure group, expand the measure group on the **Partitions** tab. Partitions for a measure group are listed by ordinal number in a table format with the columns listed in the following table.

Settings for a linked measure group must be edited in the source cube.

Deleting partitions occurs automatically when you merge a source partition into a destination partition. The partition specified as the Source is deleted after the merge is completed. You can also delete partitions manually in SQL Server Management Studio or in the Partitions tab in SQL Server Data Tools. Right-click and choose **Delete**. Remember that deleting a partition also deletes data and aggregations. As a precaution, make sure you have a recent back up of the database in case you need to reverse this step later.

> **NOTE**
>
> Alternatively, you can use XMLA scripts that automate tasks for building, merging, and deleting partitions. XMLA script can be created and executed in SQL Server Management Studio, or in custom SSIS packages that runs as a scheduled task. For more information, see Automate Analysis Services Administrative Tasks with SSIS.

Partition source

Specifies the source table or named query for the partition. To change the source table, click the cell and then click the browse (**...**) button.

If the partition is based on a query, click the browse (**...**) button to edit the query. This edits the **Source** property for the partition. For more information, see Change a partition source to use a different fact table.

You can specify a table in the data source view that has the same structure as the original source table (in the external data source from which the data is retrieved). The source can be in any of the data sources or data source views of the cube database.

Storage settings

In Cube Designer, on the Partitions tab, you can click **Storage Settings** to pick one of the standard settings for MOLAP, ROLAP, or HOLAP storage, or to configure custom settings for the storage mode and proactive caching. The default is MOLAP because it delivers the fastest query performance. For more information about each setting,

see Set Partition Storage (Analysis Services - Multidimensional).

Storage can be configured separately for each partition of each measure group in a cube. You can also configure the default storage settings for a cube or measure group. Storage is configured on the **Partitions** tab in the Cube Wizard.

See Also

Create and Manage a Local Partition (Analysis Services)
Designing Aggregations (Analysis Services - Multidimensional)
Merge Partitions in Analysis Services (SSAS - Multidimensional)

Merge Partitions in Analysis Services (SSAS - Multidimensional)

5/16/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can merge partitions in an existing Analysis Services database to consolidate fact data from multiple partitions of the same measure group.

Common scenarios

Requirements

Update the Partition Source after merging partitions

Special considerations for partitions segmented by fact table or named query

How to merge partitions using SSMS

How to merge partitions using XMLA

Common scenarios

The single most common configuration for partition use involves the separation of data across the dimension of time. The granularity of time associated with each partition varies depending on the business requirements specific to the project. For example, segmentation might be by years, with the most recent year divided by months, plus a separate partition for the active month. The active month partition regularly takes on new data.

When the active month is completed, that partition is merged back into the months in the year-to-date partition and the process continues. At the end of the year, a whole new year partition has been formed.

As this scenario demonstrates, merging partitions can become a routine task performed on a regular basis, providing a progressive approach for consolidating and organizing historical data.

Requirements

Partitions can be merged only if they meet all of the following criteria:

- They have the same measure group.
- They have the same structure.
- They must be in a processed state.
- They have the same storage modes.
- They contain identical aggregation designs.
- They share the same string store compatibility level (applies to partitioned distinct count measure groups only).

If the target partition is empty (that is, it has an aggregation design but no aggregations), merge will drop the aggregations for the source partitions. You must run Process Index, Process Full, or Process Default on the partition to build the aggregations.

Remote partitions can be merged only with other remote partitions that are defined with the same remote instance of Analysis Services.

> **NOTE**
> If you are using a combination of local and remote partitions, an alternative approach is to create new partitions that include the combined data, deleting the partitions you no longer use.

To create a partition that is a candidate for future merging, when you create the partition in the Partition Wizard, you can choose to copy the aggregation design from another of the cube's partitions. This ensures that these partitions have the same aggregation design. When they are merged, the aggregations of the source partition are combined with the aggregations in the target partition.

Update the Partition Source after merging partitions

Partitions are segmented by query, such as the WHERE clause of a SQL query used to process the data, or by a table or named query that provides data to the partition. The **Source** property on the partition indicates whether the partition is bound to a query or a table.

When you merge partitions, the contents of the partitions are consolidated, but the **Source** property is not updated to reflect the additional scope of the partition. This means if you subsequently reprocess a partition that retains its original **Source**, you will get incorrect data from that partition. The partition will erroneously aggregate data at the parent level. The following example illustrates this behavior.

The Problem

Suppose you have a cube containing information about three soft drink products. It has three partitions that use the same fact table. These partitions are segmented by product. Partition 1 contains data about [ColaFull], Partition 2 contains data about [ColaDecaf], and Partition 3 contains data about [ColaDiet]. If Partition 3 is merged into Partition 2, the data in the resulting partition (Partition 2) is correct and the cube data is accurate. However, when Partition 2 is processed, its content may be determined by the parent of the members at the product level. This parent, [SoftDrinks], also includes [ColaFull], the product in Partition 1. Processing Partition 2 loads the partition with data for all soft drinks, including [ColaFull]. The cube then contains duplicate data for [ColaFull] and returns incorrect data to end users.

The Solution

The solution is to update the **Source** property, adjusting either the WHERE clause or named query, or manually merging data from the underlying fact tables, to ensure that subsequent processing is accurate given the expanded scope of the partition.

In this example, after merging Partition 3 into Partition 2, you can provide a filter such as ("Product" = 'ColaDecaf' OR "Product" = 'ColaDiet') in the resulting Partition 2 to specify that only data about [ColaDecaf] and [ColaDiet] be extracted from the fact table, and the data pertaining to [ColaFull] be excluded. Alternatively, you can specify filters for Partition 2 and Partition 3 when they are created, and these filters will be combined during the merger process. In either case, after the partition is processed the cube does not contain duplicate data.

The Conclusion

After you merge partitions, always check the **Source** to verify the filter is correct for the merged data. If you started with a partition that included historical data for Q1, Q2, and Q3, and you now merge Q4, you must adjust the filter to include Q4. Otherwise, subsequent processing of the partition will yield erroneous results. It will not be correct for Q4.

Special considerations for partitions segmented by fact table or named

query

In addition to queries, partitions can also be segmented by table or named query. If the source partition and target partition use the same fact table in a data source or data source view, the **Source** property is valid after merging partitions. It specifies the fact table data that is appropriate to the resulting partition. Because the facts that are required for the resulting partition are present in the fact table, no modification to the **Source** property is necessary.

Partitions using data from multiple fact tables or named queries require additional work. You must manually merge the facts from the fact table of the source partition into the fact table of the target partition.

Alternatively, you can change the source for the merged partition to a named query that returns the contents of two separate fact tables. If this manual step is not performed, the fact table does not contain complete information.

For the same reason, partitions obtaining segmented data from named queries also require updating. The combined partition must now have a named query that returns the combined result set that was previously obtained from the separate named queries.

Partition storage considerations: MOLAP

When MOLAP partitions are merged, the facts stored in the multidimensional structures of the partitions are also merged. This results in an internally complete and consistent partition. However, the facts stored in MOLAP partitions are copies of facts in the fact table. When the partition is subsequently processed, the facts in the multidimensional structure are deleted (only for full and refresh) and data is copied from the fact table as specified by the data source and filter for the partition. If the source partition uses a different fact table from the target partition, the fact table of the source partition must be manually merged with the fact table of the target partition to ensure that a complete set of data is available when the resulting partition is processed. This applies as well if the two partitions were based on different named queries.

> **IMPORTANT**
> A merged MOLAP partition with an incomplete fact table contains an internally merged copy of fact table data and operates correctly until it is processed.

Partition storage considerations: HOLAP and ROLAP Partitions

When HOLAP or ROLAP partitions that have different fact tables are merged, the fact tables are not automatically merged. Unless the fact tables are manually merged, only the fact table associated with the target partition is available to the resulting partition. Facts associated with the source partition are not available for drilldown in the resulting partition, and when the partition is processed, aggregations do not summarize data from the unavailable table.

> **IMPORTANT**
> A merged HOLAP or ROLAP partition with an incomplete fact table contains accurate aggregations, but incomplete facts. Queries that refer to missing facts return incorrect data. When the partition is processed, aggregations are computed only from available facts.

The absence of unavailable facts might not be noticed unless a user attempts to drill down to a fact in the unavailable table or executes a query that requires a fact from the unavailable table. Because aggregations are combined during the merge process, queries whose results are based only on aggregations return accurate data, whereas other queries may return inaccurate data. Even after the resulting partition is processed, the missing data

from the unavailable fact table may not be noticed, especially if it represents only a small portion of the combined data.

Fact tables can be merged before or after merging the partitions. However, the aggregations will not accurately represent the underlying facts until both operations have been completed. It is recommended that you merge HOLAP or ROLAP partitions that access different fact tables when users are not connected to the cube that contains these partitions.

How to merge partitions using SSMS

> **IMPORTANT**
>
> Before merging partitions, first copy the data filter information (often, the WHERE clause for filters based on SQL queries). Later, after the merge is completed, you should update the Partition Source property of the partition containing the accumulated fact data.

1. In Object Explorer, expand the **Measure Groups** node of the cube containing the partitions that you want to merge, expand **Partitions**, right-click the partition that is the target or destination of the merge operation. For example, if you are moving quarterly fact data to a partition that stores annual fact data, select the partition that contains the annual fact data.
2. Click **Merge Partitions** to open the **Merge Partition <partition name>** dialog box.
3. Under **Source Partitions**, select the check box next to each source partition that you want to merge with the target partition, and click **OK**.

   > **NOTE**
   >
   > Source partitions are immediately deleted after the source is merged into the target partition. Refresh the Partitions folder to update its contents after the merge is completed.

4. Right-click the partition containing the accumulated data and select **Properties**.
5. Open the **Source** property and modify the WHERE clause so that it includes the partition data you just merged. Recall that the **Source** property is not updated automatically. If you reprocess without first updating the **Source**, you might not get all of the expected data.

How to merge partitions using XMLA

See this topic for information, Merging Partitions (XMLA).

See Also

Processing Analysis Services Objects
Partitions (Analysis Services - Multidimensional Data)
Create and Manage a Local Partition (Analysis Services)
Create and Manage a Remote Partition (Analysis Services)
Set Partition Writeback
Write-Enabled Partitions
Configure String Storage for Dimensions and Partitions

Set Partition Storage (Analysis Services - Multidimensional)

5/16/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Microsoft SQL Server Analysis Services provides several standard storage configurations for storage modes and caching options. These provide commonly used configurations for update notification, latency, and rebuilding data.

You can specify partition storage in the Partitions tab of the cube in SQL Server Data Tools, or on the partition property page in SQL Server Management Studio.

Guidelines for choosing a storage mode

For a large measure group, it's common practice to configure storage differently for different partitions. Consider the following guidelines:

- Use real-time ROLAP for current data that is being continuously updated.
- Use proactive caching with low latency or medium latency for partitions based on data sources that are being updated less frequently.
- Use automatic MOLAP for data sources of which users require high performance but can afford some latency of the data.
- Use scheduled MOLAP for data sources for which users need to be able to continuously access the data but see changes only periodically.
- Use MOLAP storage without proactive caching for partitions that are changing infrequently or not at all; for partitions for which users do not need to browse the most recent data; and if the data does not have to be continuously available to users during any necessary updates and processing.

These are general guidelines, and careful analysis and testing may be necessary to develop the best possible storage scheme for your data. You may also manually configure storage settings for a partition if none of the standard configurations meet your needs.

Storage Settings Descriptions

STANDARD STORAGE SETTING	DESCRIPTION
Real Time ROLAP	OLAP is in real time. Detail data and aggregations are stored in relational format. The server listens for notifications when data changes and all queries reflect the current state of the data (zero latency). This setting would typically be used for a data source with very frequent and continuous updates when the very latest data is always required by users. Depending on the types of queries generated by client applications, this method is liable to give the slowest response times.

STANDARD STORAGE SETTING	DESCRIPTION
Real Time HOLAP	OLAP is in real time. Detail data is stored in a relational format while aggregations are stored in a multidimensional format. The server listens for notifications when data changes and refreshes the multidimensional OLAP (MOLAP) aggregations as needed. No MOLAP cache is created. Whenever the data source is updated, the server switches to real-time relational OLAP (ROLAP) until the aggregations are refreshed. All queries reflect the current state of the data (zero latency). This setting would typically be used for a data source with frequent and continuous updates (but not so frequent as to require real-time ROLAP) and users always require the latest data. This method normally provides better overall performance than ROLAP storage. Users can get MOLAP performance from this setting if the data source stays silent long enough.
Low Latency MOLAP	Detail data and aggregations are stored in multidimensional format. The server listens for notifications of changes to the data and switches to real-time ROLAP while MOLAP objects are reprocessed in a cache. A silence interval of at least 10 seconds is required before updating the cache. There is an override interval of 10 minutes if the silence interval is not attained. Processing occurs automatically as data changes with a target latency of 30 minutes after the first change. This setting would typically be used for a data source with frequent updates when query performance is somewhat more important than always providing the most current data. This setting automatically processes MOLAP objects whenever required after the latency interval. Performance is slower while the MOLAP objects are being reprocessed.
Medium Latency MOLAP	Detail data and aggregations are stored in multidimensional format. The server listens for notifications of changes to the data and switches to real-time ROLAP while MOLAP objects are reprocessed in cache. A silence interval of at least 10 seconds is required before updating the cache. There is an override interval of 10 minutes if the silence interval is not attained. Processing occurs automatically as data changes with a target latency of four hours. This setting is typically used for a data source with frequent (or less frequent) updates when query performance is more important than always providing the most current data. This setting automatically processes MOLAP objects whenever required after the latency interval. Performance is slower while the MOLAP objects are being reprocessed.

STANDARD STORAGE SETTING	DESCRIPTION
Automatic MOLAP	Detail data and aggregations are stored in multidimensional format. The server listens for notifications but retains the current MOLAP cache while it builds a new one. The server never switches to real-time OLAP, and queries may be stale while the new cache is built. A silence interval of at least 10 seconds is required before creating the new MOLAP cache. There is an override interval of 10 minutes if the silence interval is not attained. Processing occurs automatically as data changes with a target latency of two hours. This setting is typically used for a data source when query performance is of key importance. This setting automatically processes MOLAP objects whenever required after the latency interval. Queries do not return the most recent data while the new cache is being built and processed.
Scheduled MOLAP	Detail data and aggregations are stored in a multidimensional format. The server does not receive notifications when data changes. Processing occurs automatically every 24 hours. This setting is typically used for a data source when only daily updates are required. Queries are always against data in the MOLAP cache, which is not discarded until a new cache is built and its objects are processed.
MOLAP	Proactive caching is not enabled. Detail data and aggregations are stored in multidimensional format. The server does not receive notifications when data changes. Processing must either be scheduled or performed manually. This setting is typically used for a data source in which periodic updates are unnecessary for the client applications but for which high performance is critical. MOLAP storage without proactive caching provides the best possible performance if your applications do not require the most recent data. It does require downtime to process updated objects, although downtime can be minimized by updating and processing cubes on a staging server and using database synchronization to copy the updated and processed MOLAP objects to the production server.

Custom Storage Options

Instead of using one of the standard storage settings, you can manually configure storage and proactive caching. Before you create custom storage settings, you may want to first click the **Standard settings** option and move the slider to the standard setting that most closely matches the configuration you want to use. Then, to create a custom configuration, click the **Custom settings** option and click **Options**.

- You can specify whether changes in a data source trigger updates to the cache. To allow for a tolerable level of churn, you can specify a minimum silence interval after updates to the data source. You can also specify a silence interval override that updates the cache after a specified period if the interval between changes to the data source never reaches the minimum.
- You can specify whether to drop the outdated cache when updates occur. If you select this option, then when the specified latency is exceeded, the server switches to real-time relational OLAP (ROLAP) while it updates

the cache. If you do not select this option, the server continues to query the stale multidimensional OLAP (MOLAP) cache while it builds the new one.

You can specify the latency interval that must occur between changes and dropping an outdated cache. This is the amount of time users may continue browsing data in an outdated cache before it is dropped. If changes occur and the cache is still being updated or processed at the end of this interval, queries will be redirected to ROLAP.

- You can schedule forced updates of the cache if you want to periodically update the cached MOLAP objects regardless of changes to the data source. Real-time OLAP benefits vary with the size of your database and the latency period assigned by frequency of source data changes. You want users sending queries to a cache as often as possible, not to ROLAP.

If you select the **Apply settings to dimensions** check box, the same storage settings are applied to the dimensions related to the measure group. The dimension values are initially the same as the partition values.

See Also

Partitions in Multidimensional Models

Set the Partition Slice Property (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A data slice is an important optimization feature that helps direct queries to the data of the appropriate partitions. Explicitly setting the Slice property can improve query performance by overriding default slices generated for MOLAP and HOLAP partitions. Additionally, the Slice property provides an extra validation check when processing the partition.

You can specify a data slice after you create a partition, but before processing it, using the Slice property. On the Partitions tab, expand a measure group, right-click a partition, and select **Properties**.

For an explanation of data slice benefits, see Set the Slice on your SSAS Cube Partition.

Defining a Slice

Valid values for a slice property are an MDX member, set, or tuple. The following examples illustrate valid slice syntax:

SLICE	MEMBER, SET OR TUPLE
[Date].[Calendar].[Calendar Year].&[2010]	Specify this slice on a partition containing facts from year 2010 (assuming the model includes a Date dimension with Calendar Year hierarchy, where 2010 is a member). Although the partition source WHERE clause or table might already filter by 2010, specifying the Slice provides an additional check during processing, as well as more targeted scans during query execution.
{ [Sales Territory].[Sales Territory Country].&[Australia], [Sales Territory].[Sales Territory Country].&[Canada] }	Specify this slice on a partition containing facts that include sales territory information. A slice can be an MDX set consisting of two or more members.
[Measures].[Sales Amount Quota] > '5000'	This slice shows an MDX expression.

A data slice of a partition should reflect, as closely as possible, the data in the partition. For example, if a partition is limited to 2012 data, the partition's data slice should specify the 2012 member of the Time dimension. It is not always possible to specify a data slice that reflects the exact contents of a partition. For example, if a partition contains data for only January and February, but the levels of the Time dimension are Year, Quarter, and Month, the Partition Wizard cannot select both the January and February members. In such cases, select the parent of the members that reflect the partition's contents. In this example, select Quarter 1.

NOTE

Note that dynamic MDX functions (such as Generate (MDX) or Except (MDX)) are not supported in the Slice property for partitions. You must define the slice by using explicit tuples or member references.

For example, rather than using the range operator (:) to define a range, you would need to enumerate each member by the specific years.

If you need to define a complex slice, we recommend that you define the tuples in the slice by using an XMLA Alter script. Then, you can use either the ascmd command-line tool or the Analysis Services Execute DDL Task in Integration Services to run the script and create the specified set of members immediately before you process the partition.

See Also

Create and Manage a Local Partition (Analysis Services)

Set Partition Writeback

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If you write-enable a measure group, end users can change cube data while they browse it, where changes are saved in a separate table called a writeback table, not in the cube data or source data. End users who browse a write-enabled partition see the net effect of all changes in the writeback table for the partition.

You can browse or delete writeback data. You can also convert writeback data to a partition. On a write-enabled partition, you can use cube roles to grant read/write access to users and groups of users, and to limit access to specific cells or groups of cells in the partition.

For a short video introduction to writeback, see Excel 2010 Writeback to Analysis Services. A more detailed exploration of this feature is available through this blog post series, Building a Writeback Application with Analysis Services (blog).

> **NOTE**
> Writeback is supported for SQL Server relational databases and data marts only, and only for Analysis Services multidimensional models.

How to write-enable a partition

You can write-enable a partition's measure groups by write-enabling the partition itself in Cube Designer in SQL Server Data Tools or SQL Server Management Studio.

- In Cube Designer, in the Partitions tab, right-click a partition and choose **Writeback Settings**.
- In Management Studio, expand the database | cube | measure group, and then right-click **Writeback** and choose **Enable Writeback**.

Writeback is only supported for measures that use the SUM aggregation. In the AdventureWorks sample database, you can use the Sales Targets measure group to test writeback behaviors.

When you write-enable a partition, you specify a table name and a data source for storing the write-back table. Any subsequent changes to the measure group are recorded in this table.

Browse writeback data in a partition

You can browse the contents of a cube's writeback table in the **Browse Data** dialog box, which you can access by right-clicking a write-enabled partition on the **Partitions** tab in Cube Designer.

Delete writeback data or disable writeback

Deleting writeback data clears the writeback cache; as soon as that data is deleted, additional writeback work is performed on a clean slate. Disabling writeback for a cube partition simply turns off writeback for that partition.

Convert writeback data to a partition

You can convert the data in a partition's writeback table to a partition. This procedure causes the writeback table to become the new partition's fact table.

Caution

Incorrect use of partitions can result in inaccurate cube data. For more information, see Create and Manage a Local Partition (Analysis Services).

Converting the writeback data table to a partition also write-disables the partition. All unrestricted read/write policies and read/write permissions for the partition's cells are disabled, and end users will not be able to change displayed cube data. (End users with disabled unrestricted read/write policies or disabled read/write permissions will still be able to browse the cube.) Read and read-contingent permissions are not affected.

To convert writeback data to a partition, use the **Convert to Partition** dialog box, which is accessed by right-clicking the writeback table for a write-enabled partition in SQL Server Management Studio. You specify a name for the partition and whether to design the aggregation for the partition later or at the same time that you create it. To create the aggregation at the same time you choose the partition, you must choose to copy the aggregation design from an existing partition. This is normally, but not necessarily, the current writeback partition. You can also choose to process the partition at the same time that you create it.

See Also

Write-Enabled Partitions
Enabling Write-back to an OLAP Cube at Cell Level in Excel 2010
Enabling and Securing Data Entry with Analysis Services Writeback

Calculations in Multidimensional Models

12/6/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use the **Calculations** tab of Cube Designer to create calculated members, named sets, and other Multidimensional Expressions (MDX) calculations.

The **Calculations** tab has the following three panes:

- The **Script Organizer** pane lists the calculations in a cube. Use this pane to create, organize, and select calculations for editing.
- The **Calculation Tools** pane provides metadata, functions, and templates with which to create calculations.
- The Calculation Expressions pane supports a form view and a script view.

Creating a New Calculation

To create a new calculation, on the **Calculations** tab of Cube Designer, on the **Cube** menu, click either **New Calculated Member**, **New Named Set**, or **New Script Command**, depending on the type of calculation you want to create. You can also click any of the corresponding buttons on the toolbar, or right-click anywhere in the **Script Organizer** pane and then click one of the commands on the shortcut menu. This action adds a new calculation to the **Script Organizer** pane and displays fields for it in the calculations form in the Calculations Expressions pane. If you create a new script, this action opens the Script view in the Calculation Expressions pane. For more information about building the three types of calculations, see Create Calculated Members, Create Named Sets, and Define Assignments and Other Script Commands.

Editing Scripts

Edit scripts in the Calculation Expressions pane of the **Calculations** tab. The Calculation Expressions pane has two views, Script view and Form view. The Form view shows the expressions and properties for a single command. When you edit an MDX script, an expression box fills the entire Form view.

The Script view provides a code editor in which to edit the scripts. While the Calculation Expressions pane is in Script view, the **Script Organizer** pane is hidden. The Script view provides color coding, parenthesis matching, auto-complete, and MDX code regions. The MDX code regions can be collapsed or expanded to facilitate editing.

You can switch between the Form view and the Script view by clicking the **Cube** menu, pointing to **Show Calculations in**, and then clicking **Script** or **Form**. You can also click either **Form view** or **Script view** on the toolbar.

Changing Solve Order

Calculations are solved in the order listed in the **Script Organizer** pane. You can reorder the calculations by right-clicking any calculation and then clicking **Move Up** or **Move Down** on the shortcut menu. You can also click a calculation and then click the **Move Up** or **Move Down** button on the toolbar.

You can override this order manually for calculated cells and calculated members. For more information about pass order and solve order, see Understanding Pass Order and Solve Order (MDX).

Deleting a Calculation

To delete an existing calculation, on the **Calculations** tab, in the **Script Organizer** pane, select the calculation to be deleted, and then click **Delete** on the **Edit** menu or click the **Delete** icon on the toolbar. You can also right-click the calculation in the **Script Organizer** pane and select **Delete** from the short-cut menu.

Create Calculated Members

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can create customized measures or dimension members, called calculated members, by combining cube data, arithmetic operators, numbers, and functions. For example, you can create a calculated member named Euros that converts dollars to euros by multiplying an existing dollar measure by a conversion rate. Euros can then be displayed to end users in a separate row or column.

Calculated member definitions are stored, but their values exist only in memory. In the preceding example, values in marks are displayed to end users but are not stored as cube data.

You create calculated members in cubes. To create a calculated member, in Cube Designer, on the **Calculations** tab, click the **New Calculated Member** icon on the toolbar. This command displays a form to specify the following options for the calculated member:

Name
Select the name of the calculated member. This name appears as the column or row heading for the calculated member values when end users browse the cube.

Parent hierarchy
Select the parent hierarchy to include in the calculated member. Hierarchies are descriptive categories of a dimension by which the numeric data (that is, measures) in a cube can be separated for analysis. In tabular browsers, hierarchies provide the column and row headings displayed to end users when they browse data in a cube. (In graphical browsers, they provide other types of descriptive labels, but they have the same function as in tabular browsers.) A calculated member provides a new heading (or label) in the parent dimension you select.

Alternatively, you can include the calculated member in the measures instead of in a dimension. This option also provides a new column or row heading, but it is attached to measures in the browser.

Parent member
Click **Change** to select a parent member to include the calculated member. This option is unavailable if you select a one-level hierarchy or MEASURES as the parent dimension.

Hierarchies are divided into levels that contain members. Each member produces a heading. While browsing data in a cube, end users can drill down from a selected heading to previously undisplayed subordinate headings. The heading for the calculated member is added at the level directly below the parent member you select.

Expression
Specify the expression that produces the values of the calculated member. This expression can be written in Multidimensional Expressions (MDX). The expression can contain any of the following:

- Data expressions that represent cube components such as dimensions, levels, measures, and so on
- Arithmetic operators
- Numbers
- Functions

You can drag or copy cube components from the **Metadata** tab of the **Calculation Tools** pane to quickly add them to an expression.

> **IMPORTANT**
> Any calculated member that is to be used in the value expression of another calculated member must be created before the calculated member that uses it.

Format String
Specifies the format of cell values that are based on the calculated member. This property accepts the same values as the **Display Format** property for measures. For more information about display formats, see Configure Measure Properties.

Visible
Determines whether the calculated member is visible or hidden when cube metadata is retrieved. If the calculated member is hidden, it can still be used in MDX expressions, statements, and scripts, but it is not displayed as a selectable object in client user interfaces.

Non-empty Behavior
Stores the names of measures used to resolve NON EMPTY queries in MDX. If this property is blank, the calculated member must be evaluated repeatedly to determine whether a member is empty. If this property contains the name of one or more measures, the calculated member is treated as empty if all the specified measures are empty. This property is an optimization hint to Analysis Services to return only non-NULL records. Returning only non-NULL records improves the performance of MDX queries that utilize the NON EMPTY operator or the NonEmpty function, or that require the calculation of cell values. For best performance with cell calculations, specify only a single member when possible.

Color Expressions
Specifies MDX expressions that dynamically set the foreground and background colors of cells based on the value of the calculated member. This property is ignored if unsupported by client applications.

Font Expressions
Specifies MDX expressions that dynamically set the font, font size, and font attributes for cells based on the value of the calculated member. This property is ignored if unsupported by client applications.

You can copy or drag cube components from the **Metadata** tab of the **Calculation Tools** pane to the **Expression** box in the Calculation Expressions pane. You can copy or drag functions from the **Functions** tab in the **Calculation Tools** pane to the **Expression** box in the Calculation Expressions pane.

Addressing Calculated Members

When you create a calculated member on the **Calculations** tab of **Cube Designer**, you specify the parent hierarchy in which the calculated member is stored. The parent hierarchy determines how a calculated member can be addressed, according to the following rules:

- If a calculated member is created in the measures dimension, the calculated member is addressable in that dimension.

See Also

Calculations in Multidimensional Models

Create Named Sets

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A named set is a set of dimension members or a set expression that is created for reuse, for example in Multidimensional Expressions (MDX) queries. You can create named sets by combining cube data, arithmetic operators, numbers, and functions. For example, you can create a named set called Top Ten Factories that contains the ten members of the Factories dimension that have the highest values for the Production measure. Top Ten Factories can then be used in queries by end users. For example, an end user can place Top Ten Factories on one axis and the Measures dimension, including Production, on another axis. For more information, see Calculations in Multidimensional Models, and Building Named Sets in MDX (MDX).

To create a named set, use the **New Named Set** command on the **Calculations** tab of Cube Designer. This command can be invoked on the **Cube** menu on the **Calculations** tab toolbar. This command displays a form to specify the following options for the named set:

Name
Select the name of the named set. This name appears to end users when they browse the cube.

Expression
Specify the expression that produces the named set. This expression can be written in MDX. The expression can contain any of the following:

- Data expressions that represent cube components such as dimensions, levels, measures, and so on.
- Arithmetic operators.
- Numbers.
- Functions.

You can copy or drag cube components from the **Metadata** tab of the **Calculation Tools** pane to the **Expression** box in the **Named Set Form Editor** pane. You can copy or drag functions from the **Functions** tab in the **Calculation Tools** pane to the **Expression** box in the **Named Set Form Editor** pane.

> **IMPORTANT**
> If you create the set expression by explicitly naming the members in the set, enclose the list of members in a pair of braces ({}).

See Also

Calculations in Multidimensional Models

Define Assignments and Other Script Commands

12/6/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

On the **Calculations** tab of Cube Designer, click the **New ScriptCommand** icon on the toolbar to create an empty script. When you create a new script, it initially is displayed with a blank title in the **Script Organizer** pane of the Calculations tab. The characters you type in the Calculation Expressions pane will be visible as the name of the item in **Script Organizer**. Therefore, you may want to type a commented name on the first line to more easily identify the script in the **Script Organizer** pane. For more information, see Introduction to MDX Scripting in Microsoft SQL Server 2005.

> **IMPORTANT**
>
> When you initially switch to the **Calculations** tab of Cube Designer, the **Script Organizer** pane contains a single script with a CALCULATE command. The CALCULATE command controls the aggregation of the cells in the cube and should be edited only if you intend to manually specify how the cube is to be aggregated.

You can use the Calculation Expressions pane to build an expression in Multidimensional Expressions (MDX) syntax. While you build the expression, you can drag or copy cube components, functions, and templates from the **Calculation Tools** pane to the Calculation Expressions pane. Doing so adds the script for the item to the Calculation Expressions pane in the location that you drop or paste it. Replace arguments and their delimiters (« and ») with the appropriate values.

> **IMPORTANT**
>
> When writing an expression that contains multiple statements using the Calculation Expressions pane, ensure that all lines except the last line of the MDX script end with a semi-colon (;). Calculations are concatenated into a single MDX script, and each script has a semi-colon appended to it to ensure that the MDX script compiles correctly. If you add a semi-colon to the last line of the script in the Calculation Expressions pane, the cube will build and deploy correctly but you will be unable to run queries against it.

See Also

Calculations in Multidimensional Models

Define Time Intelligence Calculations using the Business Intelligence Wizard

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The time intelligence enhancement is a cube enhancement that adds time calculations (or time views) to a selected hierarchy. This enhancement supports the following categories of calculations:

- Period to date.
- Period over period growth.
- Moving averages.
- Parallel period comparisons.

You apply time intelligence to cubes that have a time dimension. (A time dimension is a dimension whose **Type** property is set to **Time**). Additionally, the time attributes of that dimension must also have the appropriate setting (such as, Years or Months) for their **Type** property. The **Type** property of both the dimension and its attributes will be set correctly if you use the Dimension Wizard to create the time dimension.

To add time intelligence to a cube, you use the Business Intelligence Wizard, and select the **Define time intelligence** option on the **Choose Enhancement** page. This wizard then guides you through the steps of selecting a hierarchy to which you are adding time intelligence and specifying which members in the hierarchy will have time intelligence applied to them. On the last page of the wizard, you can see the changes that will be made to the Microsoft SQL Server Analysis Services database to add the selected time intelligence.

Selecting a Time Hierarchy

On the **Choose Target Hierarchy and Calculations** page, you select the time hierarchy to which the time enhancement applies. You can apply the time enhancement to only one time hierarchy every time that you run the Business Intelligence Wizard. If you want to apply the enhancement to more than one time hierarchy, you run the wizard again.

After you select a time hierarchy, in the **Available time calculations** list, you select the calculations that apply to the hierarchy. The calculations that are listed depend on the levels in the hierarchy, and on the **Type** property setting for the attribute for each level. For example, a Years hierarchy supports Year to Date and Year Over Year Growth, but a Quarters hierarchy does not.

> **NOTE**
>
> The Timeintelligence.xml template file defines the time calculations that are listed in **Available time calculations**. If the listed calculations do not meet your needs, you can either change the existing calculations or add new calculations to the Timeintelligence.xml file.

> **TIP**
>
> To view a description for a calculation, use the up and down arrows to highlight that calculation.

Apply Time Views to Members

On the **Define Scope of Calculations** page, you specify the members to which the new time views apply. You can apply the new time views to one of the following objects:

- **Members of an account dimension** On the **Define Scope of Calculations** page, the **Available measures** list includes account dimensions. Account dimensions have their **Type** properties set to **Accounts**. If you have an accounts dimension but that dimension does not appear in the **Available measures** list, you can use the Business Intelligence Wizard to apply the account intelligence to that dimension. For more information, see Add Account Intelligence to a Dimension.
- **Measures** Instead of specifying an account dimension, you can specify the measures to which the time views apply. In this case, select the views to which selected time calculations apply. For example, assets and liabilities are year-to-date data; therefore, you do not apply a Year-to-Date calculation to assets or liabilities measures.

Viewing the Time Intelligence Enhancement

On the final page of the Business Intelligence Wizard, you can view the changes that will be made to the Analysis Services database. For a time intelligence enhancement, the wizard will change the selected time dimension, associated data source view, and associated cube as described in the following table.

OBJECT	CHANGE
Time dimension	Adds an attribute for each calculation (or view).
Data source view	Adds a calculated column in the time table for each new attribute in the time dimension.
Cube	Adds a calculated member that defines the Multidimensional Expressions (MDX) code to perform the calculation.

See Also

Create Calculated Members

Key Performance Indicators (KPIs) in Multidimensional Models

10/26/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In business terminology, a Key Performance Indicator (KPI) is a quantifiable measurement for gauging business success.

In Analysis Services, a KPI is a collection of calculations that are associated with a measure group in a cube that are used to evaluate business success. Typically, these calculations are a combination of Multidimensional Expressions (MDX) expressions or calculated members. KPIs also have additional metadata that provides information about how client applications should display the results of the KPI's calculations.

A KPI handles information about a goal set, the actual formula of the performance recorded in the cube, and measurement to show the trend and the status of the performance. AMO is used to define the formulas and other definitions about the values of a KPI. A query interface, such as ADOMD.NET, is used by the client application to retrieve and expose the KPI values to the end user. For more information see Developing with ADOMD.NET.

A simple Kpi object is composed of: basic information, the goal, the actual value achieved, a status value, a trend value, and a folder where the KPI is viewed. Basic information includes the name and description of the KPI. The goal is an MDX expression that evaluates to a number. The actual value is an MDX expression that evaluates to a number. The status and trend value are MDX expressions that evaluate to a number. The folder is a suggested location for the KPI to be presented to the client.

In business terminology, a Key Performance Indicator (KPI) is a quantifiable measurement for gauging business success. A KPI is frequently evaluated over time. For example, the sales department of an organization may use monthly gross profit as a KPI, but the human resources department of the same organization may use quarterly employee turnover. Each is an example of a KPI. Business executives frequently consume KPIs that are grouped together in a business scorecard to obtain a quick and accurate historical summary of business success.

In Microsoft SQL Server Analysis Services, a KPI is a collection of calculations, which are associated with a measure group in a cube, that are used to evaluate business success. Typically, these calculations are a combination of Multidimensional Expressions (MDX) expressions and calculated members. KPIs also have additional metadata that provides information about how client applications should display the results of a KPI's calculation.

One key advantage of KPIs in Analysis Services is that they are server-based KPIs that are consumable by different client applications. A server-based KPI presents a single version of the truth, compared to separate versions of the truth from separate client applications. Moreover, performing the sometimes complex calculations on the server instead of on each client computer may have performance benefits.

Common KPI Terms

The following table provides definitions for common KPI terms in Analysis Services.

TERM	DEFINITION
Goal	An MDX numeric expression or a calculation that returns the target value of the KPI.

TERM	DEFINITION
Value	An MDX numeric expression that returns the actual value of the KPI.
Status	An MDX expression that represents the state of the KPI at a specified point in time. The status MDX expression should return a normalized value between -1 and 1. Values equal to or less than -1 will be interpreted as "bad" or "low." A value of zero (0) is interpreted as "acceptable" or "medium." Values equal to or greater than 1 will be interpreted as "good" or "high." An unlimited number of intermediate values can optionally be returned and can be used to display any number of additional states, if supported by the client application.
Trend	An MDX expression that evaluates the value of the KPI over time. The trend can be any time-based criterion that is useful in a specific business context. The trend MDX expression enables a business user to determine whether the KPI is improving over time or degrading over time.
Status indicator	A visual element that provides a quick indication of the status for a KPI. The display of the element is determined by the value of the MDX expression that evaluates status.
Trend indicator	A visual element that provides a quick indication of the trend for a KPI. The display of the element is determined by the value of the MDX expression that evaluates trend.
Display folder	The folder in which the KPI will appear when a user is browsing the cube.
Parent KPI	A reference to an existing KPI that uses the value of the child KPI as part of computation of the parent KPI. Sometimes, a single KPI will be a computation that consists of the values for other KPIs. This property facilitates the correct display of the child KPIs underneath the parent KPI in client applications.
Current time member	An MDX expression that returns the member that identifies the temporal context of the KPI.
Weight	An MDX numeric expression that assigns a relative importance to a KPI. If the KPI is assigned to a parent KPI, the weight is used to proportionally adjust the results of the child KPI value when calculating the value of the parent KPI.

Parent KPIs

An organization may track different business metrics at different levels. For example, only two or three KPIs may be used to gauge business success for the whole company, but these company-wide KPIs may be based on three or four other KPIs tracked by the business units throughout the company. Also, business units in a company may use different statistics to calculate the same KPI, the results of which are rolled up to the company-wide KPI.

Analysis Services lets you define a parent-child relationship between KPIs. This parent-child relationship lets the results of the child KPI be used to calculate the results of the parent KPI. Client applications can also use this relationship to appropriately display parent and child KPIs.

Weights

Weights can also be assigned to child KPIs. Weights enable Analysis Services to proportionally adjust the results of the child KPI when calculating the value of the parent KPI.

Retrieving and Displaying KPIs

The display of KPIs depends on the implementation of the client application. For example, selecting **Browser View** on the toolbar on the **KPIs** tab of Cube Designer demonstrates one possible client implementation, with graphics used to display status and trend indicators, display folders used to group KPIs, and child KPIs displayed under parent KPIs.

You can use MDX functions to retrieve individual sections of the KPI, such as the value or goal, for use in MDX expressions, statements, and scripts.

Actions in Multidimensional Models

10/26/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

An action is an end user-initiated operation upon a selected cube or portion of a cube. The operation can start an application with the selected item as a parameter, or it can retrieve information about the selected item. For more information about actions, see Actions (Analysis Services - Multidimensional Data).

Use the **Actions** tab of Cube Designer to build actions for a cube. Specify the following:

Name

Select a name that identifies the action.

Action Target

Select the object to which the action is attached. Generally, in client applications, the action is displayed when end users select the target object; however, the client application determines which end-user operation displays actions. For **Target type**, select from the following objects:

- Attribute members
- Cells
- Cube
- Dimension members
- Hierarchy
- Hierarchy members
- Level
- Level members

After you select the target object type, under **Target object**, select the cube object of the designated type.

Condition (Optional)

Specify an optional Multidimensional Expressions (MDX) expression that resolves to a Boolean value. If the value is **True**, the action is performed on the specified target. If the value is **False**, the action is not performed.

Action Content

Select the type of action. The following table summarizes the available types.

TYPE	DESCRIPTION
Data Set	Retrieves a dataset.
Proprietary	Performs an operation by using an interface other than those listed in this table.
Row Set	Retrieves a rowset.
Statement	Runs an OLE DB command.

TYPE	DESCRIPTION
URL	Displays a variable page in an Internet browser.

For **Action Expression**, specify the parameters that are passed when the action is run. The syntax must evaluate to a string, and you must include an expression written in MDX. For example, your MDX expression can indicate a part of the cube that is included in the syntax. MDX expressions are evaluated before the parameters are passed. Also, MDX Builder is available to help you build MDX expressions.

Additional Properties

Select the property. The following table summarizes the available properties.

PROPERTY	DESCRIPTION
Invocation	Specifies how the action is run. Interactive, the default, specifies that the action is run when a user accesses an object. The possible settings are: Batch Interactive On Open
Application	Describes the application of the action.
Description	Describes the action.
Caption	Provides a caption that is displayed for the action. If the caption is MDX, specify **True** for **Caption is MDX**.
Caption is MDX	Specify **True** if the caption is MDX or **False** if it is not.

> **NOTE**
> You must use Analysis Services Scripting Language (ASSL) or Analysis Management Objects (AMO) to define HTML and Command Line action types. For more information, see Action Element (ASSL), Type Element (Action) (ASSL), and Programming AMO OLAP Advanced Objects.

Creating a Reporting Action

The report server responds to URL-based requests for reports. To create a reporting action, on the **Cube** menu, click **New Reporting Action**. The following options are specific to a reporting action.

Report Server

The properties described in the following table are specified for the report server.

PROPERTY	DESCRIPTION
Server name	The name of the computer running report server.
Server path	The path exposed by report server.
Report format	HTML5, HTML3, Excel, or PDF.

> **NOTE**
> In SQL Server 2017, you can specify Transport Layer Security (https:) in the server name property.

Parameters (Optional)
The parameters are sent to the server as part of the URL string when the action is created. They include **Parameter Name** and **Parameter Value**, which is an MDX expression.

The report server URL is constructed as follows:

```
http://
host
/
virtualdirectory
/Path&
parametername1
=
parametervalue1
& ...
```

For example:

```
http://localhost/ReportServer/Sales/YearlySalesByCategory?rs:Command=Render&Region=West
```

Creating a Drillthrough Action

A drillthrough action is defined by a rowset action, which is returned to the client application as a drillthrough statement. The action target is a member of a measure group. To create a new drillthrough action, on the **Cube** menu, click **New Drillthrough Action**. The following options are specific to a drillthrough action:

Drillthrough Columns
Select one or more dimensions and, for each dimension, the drillthrough columns returned to the client application by the action.

See Also

Cubes in Multidimensional Models

Actions (Analysis Services - Multidimensional Data)

10/26/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Actions can be of different types and they have to be created accordingly. Actions can be:

- Drillthrough actions, which return the set of rows that represents the underlying data of the selected cells of the cube where the action occurs.
- Reporting actions, which return a report from Reporting Services that is associated with the selected section of the cube where the action occurs.
- Standard actions, which return the action element (URL, HTML, DataSet, RowSet, and other elements) that is associated with the selected section of the cube where the action occurs.

A query interface, such as ADOMD.NET, is used by the client application to retrieve and expose the actions to the end user. For more information see Developing with ADOMD.NET.

A simple Action object is composed of: basic information, the target where the action is to occur, a condition to limit the action scope, and the type. Basic information includes the name of the action, the description of the action, the caption suggested for the action, and others.

The target is the actual location in the cube where the action is to occur. The target is composed of a target type and a target object. Target type represents the kind of object, in the cube, where the action is to be enabled. Target type could be level members, cells, hierarchy, hierarchy members, or others. The target object is a specific object of the target type; if the target type is hierarchy, then the target object is any one of the defined hierarchies in the cube.

The condition is a **Boolean** MDX expression that is evaluated at the action event. If the condition evaluates to **true**, then the action is executed. Otherwise, the action is not executed.

The type is the kind of action to be executed. Action is an abstract class, therefore, to use it you have to use any one of the derived classes. Two kinds of actions are predefined: drillthrough and reporting. These have corresponding derived classes: DrillThroughAction and ReportAction. Other actions are covered in the StandardAction class.

In Microsoft SQL Server Analysis Services, an action is a stored MDX statement that can be presented to and employed by client applications. In other words, an action is a client command that is defined and stored on the server. An action also contains information that specifies when and how the MDX statement should be displayed and handled by the client application. The operation that is specified by the action can start an application, using the information in the action as a parameter, or can retrieve information based on criteria supplied by the action.

Actions enable business users to act upon the outcomes of their analyses. By saving and reusing actions, end users can go beyond traditional analysis, which typically ends with presentation of data, and initiate solutions to discovered problems and deficiencies, thereby extending the business intelligence application beyond the cube. Actions can transform the client application from a sophisticated data rendering tool into an integral part of the enterprise's operational system. Instead of focusing on sending data as input to operational applications, end users can "close the loop" on the decision-making process. This ability to transform analytical data into decisions is crucial to the successful business intelligence application.

For example, a business user browsing a cube notices that the current stock of a certain product is low. The client application provides to the business user a list of actions, all related to low product stock value, that are retrieved from the Analysis Services database, The business user selects the Order action for the member of the cube that

represents the product. The Order action initiates a new order by calling a stored procedure in the operational database. This stored procedure generates the appropriate information to send to the order entry system.

You can exercise flexibility when you create actions: for example, an action can launch an application, or retrieve information from a database. You can configure an action to be triggered from almost any part of a cube, including dimensions, levels, members, and cells, or create multiple actions for the same portion of a cube. You can also pass string parameters to the launched applications and specify the captions displayed to end users as the action runs.

> **IMPORTANT**
> In order for a business user to use actions, the client application employed by the business user must support actions.

Types of Actions

The following table lists the types of actions that are included in Analysis Services:

ACTION TYPE	DESCRIPTION
CommandLine	Executes a command at the command prompt
Dataset	Returns a dataset to a client application.
Drillthrough	Returns a drillthrough statement as an expression, which the client executes to return a rowset
Html	Executes an HTML script in an Internet browser
Proprietary	Performs an operation by using an interface other than those listed in this table.
Report	Submits a parameterized URL-based request to a report server and returns a report to a client application.
Rowset	Returns a rowset to a client application.
Statement	Runs an OLE DB command.
URL	Displays a dynamic Web page in an Internet browser.

Resolving and Executing Actions

When a business user accesses the object for which the command object is defined, the statement associated with the action is automatically resolved, which makes it available to the client application, but the action is not automatically executed. The action is executed only when the business user performs the client-specific operation that initiates the action. For example, a client applications might display a list of actions as a pop-up menu when the business user right-clicks on a particular member or cell.

See Also

Actions in Multidimensional Models

Add a Standard Action

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You add an action to a database by using the Actions view in Cube Designer. That view can be accessed from SQL Server Data Tools (SSDT). After you create an action, it becomes available to users after you reprocess the relevant cube. For more information, see Processing Analysis Services Objects.

To create an action

1. Open the cube for which you want to create an action, and then click the **Actions** tab.
2. On the toolbar, click the **New Action** icon, and then in the expression pane, do the following:
   - In **Name**, type a name for the action.
   - From the **Target type** drop-down list, select the type of object to which you want to attach the action. The object you select in **Target Type** determines the objects that are available and the type of selection that you can make in **Target Object**. The following table lists valid **Target Object** selections for each target type.

IF YOU SELECT THE FOLLOWING TARGET TYPE	MAKE THE FOLLOWING SELECTION IN TARGET OBJECT
Attribute Members	The only valid selection is a single attribute hierarchy. The target type of the action will be all members of the attribute wherever they appear (that is, the action will apply to user-defined hierarchies as well).
Cells	All cells is the only selection available. If you choose **Cells** as a target type, you can type an expression in **Condition** to restrict the cells with which the action is associated.
Cube	CURRENTCUBE is the only selection available. The action is associated with the current cube.
Dimension members	Select a single dimension. The action will be associated with all members of the dimension.
Hierarchy	Select a single hierarchy. The action will be associated with the hierarchy object only. Attribute hierarchies appear in the list only if their **AttributeHierarchyEnabled** and **AttributeHierarchyVisible** properties are set to **True**.
Hierarchy members	Select a single hierarchy. The action will be associated with all members of the selected hierarchy. Attribute hierarchies appear in the list only if their **AttributeHierarchyEnabled** and **AttributeHierarchyVisible** properties are set to **True**.
Level	Select a single level. The action will be associated with the level object only.

IF YOU SELECT THE FOLLOWING TARGET TYPE	MAKE THE FOLLOWING SELECTION IN TARGET OBJECT
Level members	Select a single level. The action will be associated with all members of the selected level.

- In **Target object**, click the arrow at the right of the text box, and in the tree view that opens, click the object to which you want to attach the action, and then click **OK**.
- (Optional.) In **Condition**, create an MDX expression to limit the target of the action. You can either type the expression manually, or you can drag items from the **Metadata** and **Functions** tabs.
- From the **Type** drop-down list, select the type of action that you want to create. The following table lists the types of actions that are available.

TYPE	DESCRIPTION
Dataset	Retrieves a dataset.
Proprietary	Performs an operation using an interface other than those listed in this table.
Rowset	Retrieves a rowset.
Statement	Runs an OLE DB command.
URL	Displays a Web page in an Internet browser.

- In **Action expression**, create an expression that defines the action. The expression must evaluate to a string. You can either type the expression manually, or you can drag items from the **Metadata** and **Functions** tabs.

3. (Optional.) Expand **Additional Properties**, and then perform one of the following steps:

- From the **Invocation** drop-down list, specify how the action is invoked. The following table describes the available options for invoking an action.

OPTION	DESCRIPTION
Interactive	The action is triggered by user interaction.
Batch	The action runs as a batch operation.
On open	The action runs when a user opens the cube.

- In **Application**, type the name of the application that is associated with the action. For example, if you create an action that takes a user to a particular Web site, the application associated with the action should be Microsoft Internet Explorer or another Web browser.

> **NOTE**
> Proprietary actions are not returned to the server unless the client application explicitly restricts the schema rowset to return only actions that match the name specified in **Application**.

- In **Action Content**, if you are using the URL type, enclose the Internet address in quotes, For example, "http://www.adventure-works.com".

- In **Description**, type a description for the action.
- In **Caption**, type a caption or an MDX expression that evaluates to a caption. This caption is displayed to end users when the action is initiated. If you do not specify a caption, the name of the action is used instead.
- From the **Caption is MDX** drop-down list, specify whether the caption is MDX. This field indicates to the server whether to evaluate the contents of **Caption** as an MDX expression.

Test an Action

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You test an action by using the Browser view in Cube Designer. Cube Designer can be accessed from SQL Server Data Tools (SSDT). After you create the action, you must process the cube before the action can be tested. For more information, see Processing a multidimensional model (Analysis Services).

To test an action

1. Open the cube that contains the action that you want to test, then under **View**, click **Browser**.
2. Drag the target object to either **Drop Column Fields Here** or **Drop Row Fields Here**.
3. In the view pane, right-click the object to which the action is attached (for example, a cell). The action name will appear in the context menu.
4. Click the action name to test the action.

Use a Template to Create an Action

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The **Actions** view in Cube Designer contains a number of templates that you can use to create common actions. Cube Designer can be accessed from SQL Server Data Tools (SSDT).

> **NOTE**
>
> After you create the action, it will become available to users after you update or reprocess the cube. For more information, see Processing a multidimensional model (Analysis Services).

To use a template to create an action

- Open the cube for which you want to create an action, and under **Design**, click the **Actions** tab.
- Under **Calculation Tools**, click the **Templates** tab, expand the node containing the type of action you want to create, and then double-click the template you want to use.
- Fill in the required information to complete building the action based on the template.

See Also

Actions (Analysis Services - Multidimensional Data)

Perspectives in Multidimensional Models

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A perspective is a subset of a cube created for a particular application or group of users. The cube itself is the default perspective. A perspective is exposed to a client as a cube. When a user views a perspective, it appears like another cube. Any changes made to cube data through writeback in the perspective are to the original cube. For more information about the views in Analysis Services, see Perspectives.

Use the **Perspectives** tab in Cube Designer to create or modify perspectives in a cube. The first column of the **Perspectives** tab is the **Cube Objects** column, which lists all the objects in the cube. This corresponds to the default perspective for the cube, which is the cube itself.

Creating or Deleting Perspectives

You can add a perspective to the **Perspectives** tab by clicking **New Perspective** on the **Cube** menu. You can also click the **New Perspective** button on the toolbar, or right-click anywhere in the pane and click **New Perspective** on the shortcut menu. You can add any number of perspectives to a cube.

To remove a perspective, first click any cell in the column for the perspective that you want to delete. Then, on the **Cube** menu, click **Delete Perspective**. You can also click the **Delete Perspective** button on the toolbar, or right-click any cell in the perspective you want to delete, and then click **Delete Perspective** on the shortcut menu.

Renaming Perspectives

The first row of each perspective shows its name. When you create a perspective, the name is initially Perspective (followed by an ordinal number, starting with 1, if there is already a perspective called Perspective). You can click the name to edit it.

Hiding Objects from a Perspective

To hide an object from a perspective, clear the check box in the row that corresponds to the object in the column for the perspective. The cube objects that can be hidden from a perspective are:

- Measure groups
- Measures
- Dimensions
- Hierarchies
- Named sets
- KPIs
- Actions
- Calculated members

To view any object, expand the category (**Measure Groups**, **Dimensions**, **KPIs**, **Calculations**, or **Actions**) for any object type under **Cube Objects**. To view hierarchies or attributes in a dimension, first expand a dimension, and then expand the **Hierarchies** or **Attributes** row. To view measures in a measure group, expand the measure

group.

See Also

Cubes in Multidimensional Models

Translations in Multidimensional Models (Analysis Services)

10/26/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can define translations in SQL Server Data Tools (SSDT) by using the appropriate designer for the Analysis Services object to be translated. Defining a translation creates a **Translation** object associated with the appropriate Analysis Services object that has the specified explicit literal values, in the specified language, for the properties of the associated Analysis Services object.

Elements of a multi-lingual data model

A data model used in a multi-lingual solution needs more than translated labels (field names and descriptions). It also needs to provide data values that are articulated in various language scripts. Achieving a multi-lingual solution requires that you have individual attributes, bound to columns in an external database that return the data.

Adventure Works sample databases (multidimensional as well as the relational data warehouse) demonstrate the translation capabilities of Analysis Services. The sample model includes translated captions and descriptions. The sample relational data warehouse contains columns of translated values that provide localized attribute members in the model.

To view translated data values available to the model:

1. Open the Adventure Works multidimensional model in the designer.
2. In Solution Explorer, open Data Source Views and double-click Adventure Works DW<version>.dsv.
3. Find dimDate, dimProduct, dimProductCategory, or dimProductSubcateogry. All of these dimensions contain attributes for translated members for month, day of week, product name, category name, and so on.
4. Right-click any field and select **Explore Data**. You will see English, Spanish, and French translations of each member.

Formats for date, time, and currency are not implemented through translations. To dynamically provide culturally specific formats based on the client's locale, use the Currency Conversion Wizard and **FormatString** property. See Currency Conversions (Analysis Services) and FormatString Element (ASSL) for details.

Lesson 9: Defining Perspectives and Translations in the Analysis Services Tutorial will walk you through the steps for creating and testing translations.

Defining Translations

Add translations to a cube

You can add translations to the cube, measure groups, measures, cube dimension, perspectives, KPIs, actions, named sets, and calculated members.

1. In Solution Explorer, double-click the cube name to open cube designer.
2. Click the **Translations** tab. All objects that support translations are listed in this page.

3. For each object, specify the target language (resolves internally to an LCID), translated caption, and translated description. The language list is consistent throughout Analysis Services, whether you are setting the server language in Management Studio, or adding a translation override on a single attribute.

   Remember that you cannot change the collation. A cube essentially uses one collation, even if you're supporting multiple languages through translated captions (there is an exception for dimension attributes, discussed below). If the languages won't sort properly under the shared collation, you will need to make copies of the cube just to accommodate your collation requirements.

4. Build and deploy the project.

5. Connect to the database using a client application, such as Excel, modifying the connection string to use the locale identifier. See Globalization Tips and Best Practices (Analysis Services) for details.

Add translations to a dimension and attributes

You can add translations to database dimensions, attributes, hierarchies, and levels within a hierarchy.

Translated captions are added to the model manually using your keyboard or copy-paste, but for dimension attribute members, you can obtain translated values from an external database. Specifically, the **CaptionColumn** property of an attribute can be bound to a column in a data source view.

At the attribute level, you can override collation settings, for example you might want to adjust width-sensitivity or use a binary sort for a specific attribute. In Analysis Services, collation is exposed where data bindings are defined. Because you are binding a dimension attribute translation to a different source column in the DSV, a collation setting is available so that you can specify the collation used by the source column. See Set or Change the Column Collation for details about column collation in the relational database.

1. In Solution Explorer, double-click the dimension name to open dimension designer.

2. Click the **Translations** tab. All dimension objects that support translations are listed in this page.

   For each object, specify target language (resolves to an LCID), translated caption, and translated description. The language list is consistent throughout Analysis Services, whether you are setting the server language in Management Studio, or adding a translation override on a single attribute.

3. To bind an attribute to a column providing translated values:

   a. Still in Dimension Designer | **Translations**, add a new translation. Choose the language. A new column appears on the page to accept the translated values.

   b. Place the cursor in an empty cell adjacent to one of the attributes. The attribute cannot be the key, but all other attributes are viable choices. You should see a small button with a dot in it. Click the button to open the **Attribute Data Translation Dialog Box**.

   c. Enter a translation for the caption. This is used as a data label in the target language, for example as a field name in a PivotTable field list.

   d. Choose the source column that provides the translated values of attribute members. Only pre-existing columns in the table or query bound to the dimension, are available. If the column does not exist, you need to modify the data source view, dimension, and cube to pick up the column.

   e. Choose the collation and sort order, if applicable.

4. Build and deploy the project.

5. Connect to the database using a client application, such as Excel, modifying the connection string to use the locale identifier. See Globalization Tips and Best Practices (Analysis Services) for details.

Add a translation of the database name

At the database level, you can add translations for the database name and description. The translated database name might be visible on client connections that specify the LCID of the language, but that depends on the tool. For example, viewing the database in Management Studio will not show the translated name, even if you specify the locale identifier on the connection. The API used by Management Studio to connect to Analysis Services does not read the **Language** property.

1. In Solution Explorer, right-click project name | **Edit Database** to open the database designer.
2. In Translations, specify target language (resolves to an LCID), translated caption, and translated description. The language list is consistent throughout Analysis Services, whether you are setting the server language in Management Studio, or adding a translation override on a single attribute.
3. In the Properties page of the database, set **Language** to the same LCID you specified for the translation. Optionally, set the **Collation** as well if the default no longer makes sense.
4. Build and deploy the database.

Deleting Translation Objects

You can right-click a translation object in the dimension or cube designer to permanently remove it. You cannot restore or recycle a deleted object, so be sure to review the list of affected objects before continuing.

Resolving Translations

If a client application requests information in a specified language identifier, the Analysis Services instance attempts to resolve data and metadata for Analysis Services objects to the closest possible language identifier. If the client application does not specify a default language, or specifies the neutral locale identifier (0) or process default language identifier (1024), then Analysis Services uses the default language for the instance to return data and metadata for Analysis Services objects.

If the client application specifies a language identifier other than the default language identifier, the instance iterates through all available translations for all available objects. If the specified language identifier matches the language identifier of a translation, Analysis Services returns that translation. If a match cannot be found, Analysis Services attempts to use one of the following methods to return translations with a language identifier closest to the specified language identifier:

- For the following language identifiers, Analysis Services attempts to use an alternate language identifier if a translation for the specified language identifier is not defined:

SPECIFIED LANGUAGE IDENTIFIER	ALTERNATE LANGUAGE IDENTIFIER
3076 - Chinese (Hong Kong SAR, PRC)	1028 - Chinese (Taiwan)
5124 - Chinese (Macao SAR)	1028 - Chinese (Taiwan)
1028 - Chinese (Taiwan)	Default language
4100 - Chinese (Singapore)	2052 - Chinese (PRC)
2074 - Croatian	Default language
3098 - Croatian (Cyrillic)	Default language

- For all other specified language identifiers, Analysis Services extracts the primary language of the specified language identifier and retrieves the language identifier indicated by Windows as the best match for the

primary language. If a translation for the best match language identifier cannot be found, or if the specified language identifier is the best match for the primary language, then the default language is used.

See Also

Globalization scenarios for Analysis Services
Languages and Collations (Analysis Services)

Multidimensional Model Solution Deployment

10/26/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you have completed the development of an Analysis Services project, you can deploy the database to an Analysis Services server. Analysis Services provides six possible deployment methods that can be used to move the database to a test or production server. The methods are listed here in order of advantage: AMO Automation, XMLA, Deployment Wizard, Deployment Utility, Synchronize Wizard, Backup and Restore.

This topic includes the following sections:

Deployment Methods

Deployment considerations

Related Tasks

Deployment Methods

METHOD	DESCRIPTION	LINK
Analysis Management Objects (AMO) Automation	AMO provides a programmatic interface to the complete command set for Analysis Services, including commands that can be used for solution deployment. As an approach for solution deployment, AMO automation is the most flexible, but it also requires a programming effort. A key advantage to using AMO is that you can use SQL Server Agent with your AMO application to run deployment on a preset schedule.	Developing with Analysis Management Objects (AMO)

METHOD	DESCRIPTION	LINK
XMLA	Use SQL Server Management Studio to generate an XMLA script of the metadata of an existing Analysis Services database, and then run that script on another server to recreate the initial database. XMLA scripts are easily formed in SQL Server Management Studio by defining the deployment process, then codifying it and saving it in an XMLA script. Once you have the XMLA script in a saved file, you can easily run the script according to a schedule, or embed the script in an application that connects directly to an instance of Analysis Services. You can also run XMLA Scripts on a preset basis using SQL Server Agent, but you do not have the same flexibility with XMLA Scripts as with AMO. AMO provides a larger breadth of functionality by hosting the complete spectrum of administrative commands.	Deploy Model Solutions Using XMLA
Deployment Wizard	Use the Deployment Wizard to use the XMLA output files generated by an Analysis Services project to deploy the project's metadata to a destination server. With the Deployment Wizard, you can deploy directly from the Analysis Services file, as created by the output directory by project build. The primary advantage of using the Analysis Services Deployment Wizard is convenience. Just as you can save an XMLA script for use later in SQL Server Management Studio, you can save Deployment Wizard scripts. The Deployment Wizard can be run both interactively and at the command prompt via the Deployment Utility.	Deploy Model Solutions Using the Deployment Wizard
Deployment Utility	The Deployment utility lets you start the Analysis Services deployment engine from a command prompt.	Deploy Model Solutions with the Deployment Utility

METHOD	DESCRIPTION	LINK
Synchronize Database Wizard	Use the Synchronize Database Wizard to synchronize the metadata and data between any two Analysis Services databases. The Synchronize Wizard can be used to copy both data and metadata from a source server to a destination server. If the destination server does not have a copy of the database that you want to deploy, a new database is copied to the destination server. If the destination server already has a copy of the same database, the database on the destination server is updated to use the metadata and data of the source database.	Synchronize Analysis Services Databases
Backup and Restore	Backup offers the simplest approach to transferring Analysis Services databases. From the **Backup** dialog box, you can set the options configuration, and then you can run the backup from the dialog box itself. Or, you can create a script that can be saved and run as frequently as required. Backup and restore is not used as frequently as the other deployment methods, but is a way to quickly complete a deployment with minimal infrastructure requirements.	Backup and Restore of Analysis Services Databases

Deployment considerations

Before you deploy an Analysis Services project, consider which of these questions apply to your solution and then review the related link to learn ways of addressing the issue:

CONSIDERATION	LINK TO MORE INFORMATION
What hardware and software resources are required for this solution?	Requirements and Considerations for Analysis Services Deployment
How will you deploy related objects that are outside the scope of the Analysis Services project, such as Integration Services packages, reports, or relational database schemas?	
How will you load and update the data in the deployed Analysis Services database? How will you update the metadata (such as calculations) in the deployed Analysis Services database?	Deployment Methods in this topic.
Do you want to give users access to Analysis Services data through the Internet?	Configure HTTP Access to Analysis Services on Internet Information Services (IIS) 8.0

CONSIDERATION	LINK TO MORE INFORMATION
Do you want to provide continuous query access to Analysis Services data?	Requirements and Considerations for Analysis Services Deployment
Do you want to deploy objects in a distributed environment by using linked objects or remote partitions?	Create and Manage a Local Partition (Analysis Services), Create and Manage a Remote Partition (Analysis Services) and Linked Measure Groups.
How will you secure the Analysis Services data?	Authorizing access to objects and operations (Analysis Services)

Related Tasks

Requirements and Considerations for Analysis Services Deployment

Deploy Model Solutions Using XMLA

Deploy Model Solutions Using the Deployment Wizard

Deploy Model Solutions with the Deployment Utility

Requirements and Considerations for Analysis Services Deployment

11/28/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The performance and availability of a solution depends on many factors, including the capabilities of the underlying hardware, the topology of your server deployment, the characteristics of your solution (for example, having partitions distributed across multiple servers or using ROLAP storage that requires direct access to the relational engine), service level agreements, and the complexity of your data model.

Memory and Processor Requirements

Analysis Services needs more memory and processor resources in the following cases:

- When processing large or complex cubes. These require more memory and processor resources than small or simple cubes.
- When the number of cubes within a single database increases.
- When the number of databases within a single instance of Analysis Services increases.
- When the number of instances of Analysis Services on a single computer increases.
- When the number of users who are accessing Analysis Services resources simultaneously increases.

The amount of memory and processor resources that are available to Analysis Services varies depending on the edition of SQL Server, operating system, hardware capability, and whether you are using virtual or physical processors. For more information, follow these links:

Hardware and Software Requirements for Installing SQL Server 2016

Compute Capacity Limits by Edition of SQL Server

Features Supported by the Editions of SQL Server 2016

Maximum Capacity Specifications (Analysis Services)

Disk Space Requirements

Different aspects of your Analysis Services installation and the tasks related to object processing require different amounts of disk space. The following list describes these requirements.

Cubes
Cubes that have large fact tables require more disk space than cubes that have small fact tables. Similarly, although to a lesser extent, cubes that have many large dimensions require more disk space than cubes that have fewer dimension members. Generally, you can expect that an Analysis Services database will require approximately 20 percent of the amount of space required for the same data stored in the underlying relational database.

Aggregations
Aggregations require additional space proportional to aggregations added-the more aggregations there are, the more space is required. If you avoid creating unneeded aggregations, the additional disk space that is needed for aggregations typically should not exceed approximately 10 percent of the size of the data that is stored in the

underlying relational database.

Data Mining
By default, mining structures cache to disk the dataset with which they are trained. To remove this cached data from the disk, you can use the **Process Clear Structure** processing option on the mining structure object. For more information, see Processing Requirements and Considerations (Data Mining).

Object Processing
During processing, Analysis Services stores copies of the objects it is processing in the processing transaction on disk until the processing is finished. When the processing is finished, the processed copies of the objects replace the original objects. Therefore, you must provide sufficient additional disk space for a second copy of each object to be processed. For example, if you plan to process a whole cube in a single transaction, you need sufficient hard disk space to store a second copy of the whole cube.

Availability Considerations

In an Analysis Services environment, a cube or mining model may be unavailable for querying because of a hardware or software failure. A cube also may be unavailable because it needs to be processed.

Providing Availability in the Event of Hardware or Software Failures

Hardware or software may fail for various reasons. However, maintaining availability of your Analysis Services installation is not only about troubleshooting the source of those failures, but also about providing alternative resources that enable the user to continue using the system if a failure occurs. Clustering and load balancing servers are typically used to provide the alternative resources that are necessary to maintain availability when hardware or software failures occur.

To provide availability in the event of a hardware or software failure, consider deploying Analysis Services into a failover cluster. In a failover cluster, if the primary node fails for any reason or if it must be rebooted, Microsoft Windows Clustering fails over to a secondary node. After the failover, which occurs very quickly, when users run query they are accessing the instance of Analysis Services that is running on the secondary node. For more information about failover clusters, see Windows Server Technologies: Failover Clusters.

Another solution for availability issues is to deploy your Analysis Services project onto two or more production servers. You can then use the Network Load Balancing (NLB) feature of Windows servers to combine the production servers into a single cluster. In an NLB cluster, if a server in the cluster is unavailable due to hardware or software issues, the NLB service directs user queries to those servers that are still available.

Providing Availability While Processing Structural Changes

Certain changes to a cube can cause the cube to be unavailable until it is processed. For example, if you make structural changes to a dimension in a cube, even if you reprocess the dimension, each cube that uses the modified dimension must also be processed. Until you process those cubes, users cannot query them, nor can they query any mining models that are based on a cube that has the modified dimension.

To provide availability while you process structural changes that may affect one or more cubes in an Analysis Services project, consider incorporating a staging server and using the Synchronize Database Wizard. This feature lets you update data and metadata on a staging server, and then to perform an online synchronization of the production server and the staging server. For more information, see Synchronize Analysis Services Databases.

To transparently process incremental updates to source data, enable proactive caching. Proactive caching updates cubes with new source data without requiring manual processing and without affecting the availability of cubes. For more information, see Proactive Caching (Partitions).

Scalability Considerations

Multiple instances of Microsoft SQL Server and Analysis Services on the same computer may cause performance

issues. To solve these issues, one option may be to increase the processor, memory, and disk resources on the server. However, you may also need to scale the instances of SQL Server and Analysis Services across multiple computers.

Scaling Analysis Services Across Multiple Computers

There are several ways to scale an installation of Analysis Services across multiple computers. These options are described in the following list.

- If there are multiple instances of Analysis Services on a single computer, you can move one or more instances to another computer.
- If there are multiple Analysis Services databases on a single computer, you can move one or more of the databases onto its own instance of Analysis Services on a separate computer.
- If one or more relational databases provide data to an Analysis Services database, you can move these databases to a separate computer. Before you move the databases, consider the network speed and bandwidth that exist between the Analysis Services database and its underlying databases. If the network is slow or congested, moving the underlying databases to a separate computer will degrade processing performance.
- If processing affects query performance, but you can't process during times of reduced query load, consider moving your processing tasks to a staging server and then performing an online synchronization of the production server and the staging server. For more information, see Synchronize Analysis Services Databases. You can also distribute processing across multiple instances of Analysis Services by using remote partitions. Processing remote partitions uses the processor and memory resources on the remote server, instead of the resources on the local computer. For information on remote partitions management, see Create and Manage a Remote Partition (Analysis Services).
- If query performance is poor but you cannot increase the processor and memory resources on the local server, consider deploying an Analysis Services project onto two or more production servers. Then you can use Network Load Balancing (NLB) to combine the servers into a single cluster. In an NLB cluster, queries are automatically distributed across all the servers in the NLB cluster.

Deploy Model Solutions Using XMLA

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In SQL Server Management Studio, the **CREATE To** option of the **Script Database As** command creates an XML script of an entire Microsoft SQL Server Analysis Services database or one of its constituent objects. The resulting script can then be run on another computer to recreate the schema (metadata) of the Analysis Services database. The script generates the entire database, and there is no mechanism for incrementally updating already deployed objects when using the script. After running the script and deploying the database, the newly created database must be processed before users can browse it.

For more information about the **Script Database As** command, see Document and Script an Analysis Services Database.

Modifying Object Properties in the XML Script

When using the **Script Database As** command, you cannot modify specific properties (such as the database name, data source connection strings, and security settings) of the database objects. These properties must either be manually modified in the script after the script has been generated or modified in the deployed database after running the script.

> **IMPORTANT**
>
> The XML script will not contain the password if this is specified in either the connection string for a data source or for impersonation purposes. Since the password is required for processing purposes in this scenario, you will either need to add this manually to the XML script before it executes or add it after the XML script executes.

See Also

Deploy Model Solutions Using the Deployment Wizard
Synchronize Analysis Services Databases

Deploy Model Solutions Using the Deployment Wizard

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Analysis Services Deployment Wizard uses JSON output files generated from a Microsoft SQL Server Analysis Services project as input files. These input files are easily modifiable to customize the deployment of an Analysis Services project. The generated deployment script can then either be immediately run or saved for later deployment.

> **NOTE**
> The Analysis Services Deployment Wizard/Utility is installed with SQL Server Managment Studio (SSMS). Be sure you're using the latest version. If running from the command prompt, by default, the latest version of the deployment wizard is installed to C:\Program Files (x86)\Microsoft SQL Server\140\Tools\Binn\ManagementStudio.

You can deploy by using the wizard as discussed here. You can also automate deployment or use the Synchronize capability. If the deployed database is large, consider using partitions on target systems. You can automate partition creation and population by using Tabular Object Model (TOM), Tabular Model Scriting Language (TMSL), and Analysis Management Objects (AMO).

> **IMPORTANT**
> Neither the output files nor the deployment script will contain the user id or password if these are specified in either the connection string for a data source or for impersonation purposes. Since these are required for processing purposes in this scenario, you will add this information manually. If the deployment will not include processing, you can add this connection and impersonation information as needed after deployment. If the deployment will include processing, you can either add this information within the wizard or in the deployment script after it is saved.

In This Section

The following topics describe how to work with the Analysis Services Deployment Wizard, the input files, and the deployment script:

TOPIC	DESCRIPTION
Running the Analysis Services Deployment Wizard	Describes the various ways in which you can run the Analysis Services Deployment Wizard.
Understanding the Input Files Used to Create the Deployment Script	Describes which files the Analysis Services Deployment Wizard uses as input values, what each of those files contains, and provides links to topics that describe how to modify the values in each of the input files.
Understanding the Analysis Services Deployment Script	Describes what the deployment script contains and how the script runs.

See Also

Deploy Model Solutions Using XMLA
Synchronize Analysis Services Databases
Understanding the Input Files Used to Create the Deployment Script
Deploy Model Solutions with the Deployment Utility

Running the Analysis Services Deployment Wizard

11/28/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Analysis Services Deployment Wizard can be run the following ways:

- **Interactively** When run interactively, the Analysis Services Deployment Wizard generates a deployment script based on the input files, as modified interactively by user input. The wizard applies any user modifications only to the deployment script. The wizard does not modify the input files. For more information about the input files, see Understanding the Input Files Used to Create the Deployment Script.
- **From the command prompt** When run at the command prompt, the Analysis Services Deployment Wizard generates a deployment script based upon the switches that you use to run the wizard. The wizard may any one of the following: prompt you for user input and modify input files based on that input; run a silent, unattended deployment using the input files as is; or create a deployment script that you can use later.

Running the Analysis Services Deployment Wizard Interactively

When you run interactively, the Analysis Services Deployment Wizard reads the values from the input files and presents this information to you. You can modify these input values-such as deployment destination, configuration settings, deployment options, and connection string passwords-or leave them as is. If you change any input values, the wizard uses these changes when generating the deployment script. However, the wizard does not make any changes to the values in the input file.

> **NOTE**
> If you want to have the Analysis Services Deployment Wizard modify the input values, run the wizard at the command prompt and set the wizard to run in answer file mode.

After you review the input values and make any wanted changes, the wizard generates the deployment script. You can run this deployment script immediately on the destination server or save the script for later use.

To run the Analysis Services Deployment Wizard interactively

- Click **Start** > **Microsoft SQL Server** > **Deployment Wizard**.

  -or-

- In the **Projects** folder of the Analysis Services project, double-click the <project name>.asdatabase file.

  > **NOTE**
  > If you cannot find the .asdatabase file, try using Search and specify *.asdatabase. Or, you may need to build the project in SSDT.

Running the Analysis Services Deployment Wizard at the Command Prompt

The Analysis Services Deployment Wizard can also be run at the command prompt. When you run the wizard at the command prompt, you provide the full path to the .asdatabase file and run the wizard in one of the following

modes:

Answer file mode
In answer file mode, the wizard lets you interactively modify the input files that were originally generated when the Analysis Services project was built in SQL Server Data Tools (SSDT). The wizard saves these modified input files before generating the deployment script. The modified input files become the new starting point the next time that the wizard is run.

To run the wizard in answer file mode, use the **/a** switch.

Silent mode
In silent mode, the wizard runs a silent, unattended deployment based on the information resident in the input files.

To run the wizard in silent mode, use the **/s** switch. When you run the wizard in silent mode, messages are output to the console or to a log file if one is provided.

Output mode
In output mode, the wizard generates a deployment script for later execution based on the input files.

To run the wizard in output mode, use the **/o** switch and provide an output file name.

For more information about these command line switches, see Deploy Model Solutions with the Deployment Utility.

The following procedure describes how to run the Analysis Services Deployment Wizard at the command prompt.

To run the Analysis Services Deployment Wizard at the command prompt

1. Open a command prompt and navigate to the C:\Program Files (x86)\Microsoft SQL Server\140\Tools\Binn\ManagementStudio
2. Type **Microsoft.AnalysisServices.Deployment.exe** followed by the switches that correspond to the mode in which you want to run the wizard.

See Also

Understanding the Analysis Services Deployment Script
Deploy Model Solutions Using the Deployment Wizard

Deployment Script Files - Input Used to Create Deployment Script

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you build a Microsoft SQL Server Analysis Services project, SQL Server Data Tools (SSDT) generates files for the project. SQL Server Data Tools (SSDT) puts these files in the Output folder of the Analysis Services project. By default output is out in the \Bin folder. The following table lists the XML files that SQL Server Data Tools (SSDT) creates.

FILE	DESCRIPTION
<*project name*>.asdatabase	An XMLA file for Multidimensional or 1100/1103 Tabular model projects, or a JSON file for Tabular 1200 and higher model projects. Contains the declarative definitions for all the Analysis Services objects in the project.
<*project name*>.deploymenttargets	Contains the name of the Analysis Services instance and database in which the Analysis Services objects will be created.
<*project name*>.configsettings	Contains environment specific settings, such as data source connection information and object storage locations. Settings in this file override settings in the <*project name*>.asdatabase file.
<*project name*>.deploymentoptions	Contains deployment options, such as whether deployment is transactional and whether deployed objects should be processed after deployment.

NOTE

SQL Server Data Tools (SSDT) never stores passwords in its project files.

Modifying the Input Files

Modifying the values in the input files, or the values retrieved from the input files, makes it possible to change the deployment destination, the configuration settings, and deployment options without editing the whole <*project name*>.asdatabase file (or a whole script file if you generate a script from an existing Analysis Services database). Being able to modify individual files lets you easily create different deployment scripts for different purposes.

The following topics explain how to modify values in the various input files:

- Specifying the Installation Target
- Specifying Partition and Role Deployment Options
- Specifying Configuration Settings for Solution Deployment
- Specifying Processing Options

See Also

Running the Analysis Services Deployment Wizard
Understanding the Analysis Services Deployment Script

Deployment Script Files - Specifying the Installation Target

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Analysis Services Deployment Wizard reads the installation target information from the <*project name*>.deploymenttargets file. SQL Server Data Tools (SSDT) creates this file when you build the Microsoft SQL Server Analysis Services project. SQL Server Data Tools (SSDT) uses the database and server specified on the **Deployment** page of the <*project name*> **Properties Pages** dialog box to create the <*project name*>.targets file.

Modifying the Installation Target for Deployment

In some situations, you may need to deploy an Analysis Services project to a database or Analysis Services instance that is different than the one specified on the **Deployment** page. For example, you may want to deploy the project to a server for testing before deployment, and then deploy it to a production server after testing is finished. You may also want to deploy a completed and tested project to multiple production servers in a Network Load Balancing cluster, or to a staging server and a production server.

To deploy an Analysis Services project to a different database or Analysis Services instance, you can change the installation target in the input file by using one of the methods described in the following procedure.

To change the installation target after the input files have been generated

- Run the Analysis Services Deployment Wizard interactively. On the **Installation Target** page, specify a new destination for the Analysis Services instance and database.

  -or-

- Run the Analysis Services Deployment Wizard at the command prompt and set the wizard to run in answer file mode. For more information about answer file mode, see Running the Analysis Services Deployment Wizard.

  -or-

- Modify the <*project name*>.deploymenttargets file by using any text editor.

See Also

Specifying Partition and Role Deployment Options
Specifying Configuration Settings for Solution Deployment
Specifying Processing Options

Deployment Script Files - Partition and Role Deployment Options

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Analysis Services Deployment Wizard reads the partition and role deployment options from the <*project name*>.deploymentoptions file. SQL Server Data Tools (SSDT) creates this file when you build the Microsoft SQL Server Analysis Services project. SQL Server Data Tools (SSDT) uses the partition and role deployment options of the current project when the <*project name*>.deploymentoptions file is created. For more information about configuration settings, see Understanding the Input Files Used to Create the Deployment Script.

Reviewing the Partition and Role Deployment Options

The deployment options in the <*project name*>.deploymentoptions file include the following:

Partition deployment options

The <*project name*>.deploymentoptions file specifies whether existing partitions in the destination database are retained or overwritten (default). If existing partitions are retained, only new partitions will be deployed, and the partitions and aggregation designs on all existing measure groups are left unchanged.

> **NOTE**
>
> If the measure group in which the partition exists is deleted, the partition is automatically deleted.

Role deployment options

The <*project name*>.deploymentoptions file specifies one of the following role deployment options:

- Existing roles and role members in the destination database are retained, and only new roles and role members are deployed.
- All existing roles and members in the destination database are replaced by the roles and members being deployed.
- Existing roles and role members in the destination database are retained, and no new roles are deployed.
- **Note** When existing roles and members are retained, the permissions associated with those roles are reset to none. Security permissions are contained by the objects they secure, not by the security roles with which they are associated. For more information about how to work with this behavior by using the Analysis Service Deployment Wizard, see 'Retain Roles and Members' in the Microsoft Knowledge Base.

Modifying the Partition and Role Deployment Options

You may have to deploy the Analysis Services project using different partition and role options than those stored in the <*project name*>.deploymentoptions file. For example, you may want to retain existing partitions, roles, and role members, instead of replacing all existing partitions, roles, and members as indicated in the <*project name*>.deploymentoptions file.

To modify the deployment of partitions and roles in an Analysis Services project, you cannot change the partition and roles settings within the project because the <*project name*> **Properties Pages** dialog box in SQL Server Data Tools (SSDT) does not display these options. If you want to change the deployment options for roles and

partitions, you must change this information within the <*project name*>.deploymentoptions file itself. The following procedure describes how to change the partition and role deployment options within the <*project name*>.deploymentoptions file.

To change the deployment of partitions or roles after the input files have been generated

- Run the Analysis Services Deployment Wizard interactively, and on the **Partition and Role Deployment Options** page, specify new deployment options for the partitions and roles.

  -or-

- Run the Analysis Services Deployment Wizard at the command prompt, and set the wizard to run in answer file mode. (For more information about answer file mode, see Running the Analysis Services Deployment Wizard.)

  -or-

- Open the <*project name*>.deploymentoptions in any text editor and manually change the options. The options for PartitionDeployment are DeployPartitions, RetainPartitions. The options for RoleDeployment are DeployRolesAndMembers, DeployRolesRetainMembers, RetainRoles.

See Also

Specifying the Installation Target
Specifying Configuration Settings for Solution Deployment
Specifying Processing Options

Deployment Script Files - Solution Deployment Config Settings

11/28/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Analysis Services Deployment Wizard reads the partition and role deployment options that you use in the deployment script from the <*project name*>.configsettings file. SQL Server Data Tools (SSDT) creates this file when you build the Microsoft SQL Server Analysis Services project. SQL Server Data Tools (SSDT) uses the configuration settings of the current project to create the <*project name*>.configsettings file.

Reviewing the Configuration Settings for Deployment

The following are the configuration settings stored in the <*project name*>.configsettings file:

- **Data Source Connection Strings** These are the connection strings for each data source based on the values specified in the Analysis Services project. The user id and password are always removed from the connection string before the remainder of the string is stored in this file. However, if the Deployment Wizard is deploying directly to an Analysis Services instance, you can add the appropriate user id and password information within the wizard to enable a successful processing of the deployment database. This connection information will not be stored in the deployment script itself if one is saved by the Deployment Wizard.
- **Impersonation Accounts** This setting specifies the user name that Analysis Services uses to run statements in each data source. If no impersonation account is specified, Analysis Services uses its logon account to run statements. If the Analysis Services logon account is granted permissions directly in the data source, all database administrators in all databases in the Analysis Services instance have access to the data source through the logon account. If a user account and password is specified, this information is always removed before the impersonation information is stored in this file. However, if the Deployment Wizard is deploying directly to an Analysis Services instance, you can add the appropriate user id and password information within the wizard to enable a successful processing of the deployment database. This impersonation information will not be stored in the deployment script itself if one is saved by the Deployment Wizard.
- **Key Error Log Files** This setting specifies the file name and path of the key error log file for each cube, measure group, partition, and dimension in the database.
- **Storage Locations** This setting specifies the storage location for each cube, measure group, and partition in the database. If no value is provided for an object, the Analysis Services Deployment Wizard uses the default location for the object. For example, partitions use the location for the measure group, measure groups use the location for the cube, and cubes use the default location for objects on the Analysis Services instance. The storage location can be a local or a Universal Naming Convention (UNC) path.
- **Report Server** This setting specifies the report server and folder location for each report action defined in each cube in the database.

Modifying the Configuration Settings for Deployment

In some cases, you may need to deploy the Analysis Services project using different configuration settings than those stored in the <*project name*>.configsettings file. For example, you may want to change the connection string to one or more data sources, or specify storage locations for specific partitions or measure groups.

To modify the deployment of partitions and roles in an Analysis Services project, you must change this information within the <*project name*>.configsettings file, as described in the procedure below. You cannot change the partition and roles settings within the project because the <*project name*> **Properties Pages** dialog box in SQL Server Data Tools (SSDT) does not display these options.

> **NOTE**
>
> Configuration settings can apply to all objects or only to newly created objects. Apply configuration settings to newly created objects only when you are deploying additional objects to a previously deployed Analysis Services database and do not want to overwrite existing objects. To specify whether configuration settings apply to all objects or only to newly created ones, set this option in the <*project name*>.deploymentoptions file. For more information, see Specifying Partition and Role Deployment Options.

To change configuration settings after the input files have been generated

- Run the Analysis Services Deployment Wizard interactively, and on the **Configuration Settings** page, specify the configuration setting for the objects being deployed.

  -or-

- Run the Analysis Services Deployment Wizard at the command prompt and set the wizard to run in answer file mode. For more information about answer file mode, see Running the Analysis Services Deployment Wizard.

  -or-

- Modify the <*project name*>.configsettings file by using any text editor.

See Also

Specifying the Installation Target
Specifying Partition and Role Deployment Options
Specifying Processing Options

Deployment Script Files - Specifying Processing Options

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Analysis Services Deployment Wizard reads the processing options from the *<project name>*.deploymentoptions file. SQL Server Data Tools (SSDT) creates this file when you build the Microsoft SQL Server Analysis Services project. SQL Server Data Tools (SSDT) uses the processing options specified on the **Deployment** page of *<project name>* **Properties Pages** dialog box to create the *<project name>*.deploymentoptions file.

Reviewing the Processing Options for Deployment

The configuration settings stored within the *<project name>*.deploymentoptions file are as follows:

- **Processing Method** This setting controls whether the deployed objects are processed after deployment and the type of processing that will be performed. There are three processing options:
  - Default processing (default)
  - Full processing
  - None
- **Writeback Table Options** If writeback is enabled in the Analysis Services project, this setting defines how writeback is handled. There are three writeback table options:
  - By default, if a writeback table exists, it will be used. If a writeback table does not exist, a new writeback table will be created.
  - If a writeback table already exists, the deployment fails. If a writeback table does not exist, a new writeback table will be created.
  - Regardless of whether a writeback table already exists, a new writeback table will be created. In this case, the Analysis Services Deployment Wizard will delete any existing table and replace it with a new writeback table.
- **Transactional Deployment** This setting controls whether the deployment of metadata changes and process commands occurs in a single transaction or in separate transactions.
  - If this option is **True** (default), Analysis Services deploys all metadata changes and all process commands within a single transaction.
  - If this option is **False**, Analysis Services deploys the metadata changes in a single transaction, and deploys each processing command in its own transaction.

Modifying the Processing Options for Deployment

However, you may need to deploy the Analysis Services project using different processing options than those stored in the *<project name>*.deploymentoptions file. For example, you may want to have all objects fully processed, or processed using the default processing option, or have no processing occur. If the cubes or dimensions are write-enabled, you can specify whether a new or existing writeback table be used.

To modify the processing options used during deployment, you can either edit and rebuild the project, or change the processing options in the input file by using one of the methods as described in the following procedure.

To change processing options after the input files have been generated

- Run the Analysis Services Deployment Wizard interactively. On the **Processing Options** page, specify the processing options for the project being deployed.

  -or-

- Run the Analysis Services Deployment Wizard at the command prompt and set the wizard to run in answer file mode. For more information about answer file mode, see Running the Analysis Services Deployment Wizard.

  -or-

- Modify the <*project name*>.deploymentoptions file by using any text editor.

See Also

Specifying the Installation Target
Specifying Partition and Role Deployment Options
Specifying Configuration Settings for Solution Deployment

Understanding the Analysis Services Deployment Script

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The XMLA deployment script generated by the Analysis Services Deployment Wizard consists of two sections:

- The first part of the deployment script contains the commands required to create, alter, or delete the appropriate Microsoft SQL Server objects in the destination database. By default, the input files generated by the Analysis Services project are based on an incremental deployment. As a result, the XMLA deployment script will only affect those objects that were changed or deleted.
- The second part of the deployment script contains the commands required to process only the objects created or altered on the destination server (the Process Default option) or to fully process the destination database. You can also choose to have the deployment script contain no processing commands.

The entire deployment script can execute in a single transaction or in multiple transactions. If the script executes in multiple transactions, the first part of the script executes as a single transaction, and each object is processed in its own transaction.

> **IMPORTANT**
>
> The Analysis Services Deployment Wizard only deploys objects into a single Analysis Services database. It does not deploy any server level objects or data.

See Also

Running the Analysis Services Deployment Wizard
Understanding the Input Files Used to Create the Deployment Script

Deploy Model Solutions with the Deployment Utility

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The **Microsoft.AnalysisServices.Deployment** utility lets you start the Microsoft SQL Server Analysis Services deployment engine from the command prompt. As input file, the utility uses the XML output files generated by building an Analysis Services project in SQL Server Data Tools (SSDT). The input files are easily modifiable to customize the deployment of an Analysis Services project. The generated deployment script can then either be immediately run or saved for later deployment.

> **NOTE**
> The Analysis Services Deployment Wizard/Utility is installed with SQL Server Managment Studio (SSMS). Be sure you're using the latest version. By default, the latest version of the deployment wizard is installed to C:\Program Files (x86)\Microsoft SQL Server\140\Tools\Binn\ManagementStudio.

Syntax

```
Microsoft.AnalysisServices.Deployment [ASdatabasefile]
    {[/s[:logfile]] | [/a] | [[/o[:output_script_file]] [/d]]}
```

Arguments

ASdatabasefile
The full path of the folder in which the Analysis Services deployment script (.asdatabase) file is located. This file is generated when you deploy a project in SQL Server Data Tools. It is located in the project bin folder. The .asdatabase file contains the object definitions to be deployed is located. If not specified, the current folder is used.

/s
Runs the utility in silent mode and not display any dialog boxes. For more information about modes, see the section, Modes, later in this topic.

logfile
The full path and file name of the log file. Trace events will be logged to the specified log file. If the log file already exists, the file's content will be replaced.

/a
Runs the utility in answer mode. All responses made during the wizard part of the utility should be written back to the input files, but no changes will actually be made to the deployment targets.

/o
Runs the utility in output mode. Deployment will not occur, but the XML for Analysis (XMLA) script that would ordinarily be sent to the deployment targets is instead saved to the specified output script file. If *output_script_file* is not specified, the utility tries to use the output script file specified in the deployment options (.deploymentoptions) input file. If an output script file is not specified in the deployment options input file, an error occurs.

For more information about modes, see the section, Modes, later in this topic.

output_script_file
The full path and file name of the output script file.

/d
If the **/o** argument is used, specifies that the utility should not connect to the target instance. Because no connection is made to the deployment targets, the output script is generated based only on information retrieved from the input files.

> **NOTE**
> The **/d** argument is used only in output mode. This argument is ignored if specified in answer or silent mode. For more information about modes, see the section, Modes, later in this topic.

Remarks

The **Microsoft.AnalysisServices.Deployment** utility takes a set of files that provide the object definitions, deployment targets, deployment options, and configuration settings, and tries to deploy the object definitions to the specified deployment targets, using the specified deployment options and configuration settings. This utility can provide a user interface when invoked in answer file or output mode. For more information about how to use the user interface supplied for this utility to create answer files, see Deploy Model Solutions Using the Deployment Wizard.

The utility is located in the \Program files (x86)\Microsoft SQL Server\140\Binn\ManagementStudio folder.

Modes

The utility can be run in the modes listed in the following table.

MODE	DESCRIPTION
Silent mode	No user interface is displayed and all information needed for deployment is supplied by the input files. No progress is displayed by the utility in silent mode. Instead, an optional log file can be used to capture progress and error information for later review.
Answer mode	The Deployment Wizard user interface is displayed and user responses are saved to the specified input files for later deployment. Deployment does not occur in answer mode. The only purpose of answer mode is to capture user responses
Output mode	No user interface is displayed and all information needed for deployment is supplied by the input files. However, unlike silent mode, the output from the utility is written to an output script file, not sent to the deployment targets indicated in the input files. Unless the **/d** argument is specified, the utility connects with each deployment target to compare metadata while generating the output script file.

Back to Arguments

Examples

The following example shows how to deploy an Analysis Services project in silent mode, logging progress and

error messages for later review:

```
Microsoft.AnalysisServices.Deployment.exe
```

```
<drive>:\My Documents\Visual Studio 2010\Projects\AdventureWorksProject\Project1\bin
```

```
/s: C:\ My Documents\Visual Studio 2010\Projects\AdventureWorksProject\Project1\bin\deployment.log
```

See Also

Command Prompt Utility Reference (Database Engine)

Multidimensional Model Databases (SSAS)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

An Analysis Services database is a collection of data sources, data source views, cubes, dimensions, and roles. Optionally, an Analysis Services database can include structures for data mining, and custom assemblies that provide a way for you to add user-defined functions to the database.

Cubes are the fundamental query objects in Analysis Services. When you connect to an Analysis Services database via a client application, you connect to a cube within that database. A database might contain multiple cubes if you are reusing dimensions, assemblies, roles, or mining structures across multiple contexts.

You can create and modify a Analysis Services database programmatically or by using one of these interactive methods:

- Deploy an Analysis Services project from SQL Server Data Tools (SSDT) to a designated instance of Analysis Services. This process creates an Analysis Services database, if a database with that name does not already exist within that instance, and instantiates the designed objects within the newly created database. When working with an Analysis Services database in SQL Server Data Tools, changes made to objects in the Analysis Services project take effect only when the project is deployed to an Analysis Services instance.
- Create an empty Analysis Services database within an instance of Analysis Services, by using either SQL Server Management Studio or SQL Server Data Tools (SSDT), and then connect directly to that database using SQL Server Data Tools (SSDT) and create objects within it (rather than within a project). When working with an Analysis Services database in this manner, changes made to objects take effect in the database to which you are connecting when the changed object is saved.

SQL Server Data Tools (SSDT) uses integration with source control software to support multiple developers working with different objects within an Analysis Services project at the same time. A developer can also interact with an Analysis Services database directly, rather than through an Analysis Services project, but the risk of this is that the objects in an Analysis Services database can become out of sync with the Analysis Services project that was used for its deployment. After deployment, you administer an Analysis Services database by using SQL Server Management Studio. Certain changes can also be made to an Analysis Services database by using SQL Server Management Studio, such as to partitions and roles, which can also cause the objects in an Analysis Services database to become out of sync with the Analysis Services project that was used for its deployment.

Related Tasks

Attach and Detach Analysis Services Databases

Backup and Restore of Analysis Services Databases

Document and Script an Analysis Services Database

Modify or Delete an Analysis Services Database

Move an Analysis Services Database

Rename a Multidimensional Database (Analysis Services)

Compatibility Level of a Multidimensional Database (Analysis Services)

Set Multidimensional Database Properties (Analysis Services)

Synchronize Analysis Services Databases

Switch an Analysis Services database between ReadOnly and ReadWrite modes

See Also

Connect in Online Mode to an Analysis Services Database
Create an Analysis Services Project (SSDT)
Querying Multidimensional Data with MDX

Attach and Detach Analysis Services Databases

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

There are often situations when an Analysis Services database administrator (dba) wants to take a database offline for a period, and then bring that database back online on the same server instance, or on a different one. These situations are often driven by business needs, such as moving the database to a different disk for better performance, gaining room for database growth, or to upgrade a product. For all those cases and more, the **Attach** and **Detach** commands enable the Analysis Services dba to take the database offline and bring it back online with little effort.

Attach and Detach commands

The **Attach** command enables you to bring online a database that was taken offline. You can attach the database to the original server instance, or to another instance. When you attach a database the user can specify the **ReadWriteMode** setting for the database. The **Detach** command enables you to take offline a database from the server.

Attach and Detach Usage

The **Attach** command is used to bring online an existing database structure. If the database is attached in **ReadWrite** mode, it can be attached only one time to a server instance. However, if the database is attached in **ReadOnly** mode, it can be attached multiple times to different server instances. However, the same database cannot be attached more than one time to the same server instance. An error is raised when an attempt is made to attach the same database more than one time, even if the data has been copied to separate folders.

> **IMPORTANT**
>
> If a password was required to detach the database, the same password is required to attach the database.

The **Detach** command is used to take offline an existing database structure. When a database is detached, you should provide a password to protect confidential metadata.

> **IMPORTANT**
>
> To protect the content of the data files, you should use an access control list for the folder, subfolders, and data files.

When you detach a database, the server follows these steps.

DETACHING A READ/WRITE DATABASE	DETACHING A READ-ONLY DATABASE
1) The server issues a request for a CommitExclusive Lock on the database 2) The server waits until all ongoing transactions are either committed or rolled back 3) The server builds all the metadata that it must have to detach the database 4) The database is marked as deleted 5) The server commits the transaction	1) The database is marked as deleted 2) The server commits the transaction Note: The detaching password cannot be changed for a read-only database. An error is raised if the password parameter is provided for an attached database that already contains a password.

The **Attach** and **Detach** commands must be executed as single operations. They cannot be combined with other operations in the same transaction. Also, the **Attach** and **Detach** commands are atomic transactional commands. This means the operation will either succeed or fail. No database will be left in an uncompleted state.

> **IMPORTANT**
>
> Server or database administrator privileges are required to execute the **Detach** command.

> **IMPORTANT**
>
> Server administrator privileges are required to execute the **Attach** command.

See Also

Detach
Move an Analysis Services Database
Database ReadWriteModes
Switch an Analysis Services database between ReadOnly and ReadWrite modes
Detach Element
Attach Element

Backup and Restore of Analysis Services Databases

12/6/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services includes backup and restore so that you can recover a database and its objects from a particular point in time. Backup and restore is also a valid technique for migrating databases to upgraded servers, moving databases between servers, or deploying a database to a production server. For the purposes of data recovery, if you do not already have a backup plan and your data is valuable, you should design and implement a plan as soon as possible.

The backup and restore commands are performed on a deployed Analysis Services database. For your projects and solutions in SQL Server Data Tools (SSDT), you should use source control to ensure you can recover specific versions of your source files, and then create a data recovery plan for the repository of the source control system you are using.

For a full backup that includes source data, you have to back up the database which contains detail data. Specifically, if you are using ROLAP or DirectQuery database storage, detail data is stored in an external SQL Server relational database that is distinct from the Analysis Services database. Otherwise, if all objects are tabular or multidimensional, the Analysis Services backup will include both the metadata and source data.

One clear benefit of automating backup is that the data snapshot will always be as up-to-date as the automated frequency of backup specifies. Automated schedulers ensure that backups are not forgotten. Restoring a database can also be automated, and can be a good way to replicate data, but be sure to back up the encryption key file on the instance you replicate to. The synchronization feature is dedicated to replication of Analysis Services databases, but only for the data that is out of date. All of the features mentioned here can be implemented through the user interface, by way of XML/A commands or programmatically run through AMO.

This topic includes the following sections:

- Preparing for Backup
- Backing Up a Multidimensional or a Tabular Database
- Restoring an Analysis Services Database

Prerequisites

You must have administrative permissions on the Analysis Services instance or Full Control (Administrator) permissions on the database you are backing up.

Restore location must be an Analysis Services instance that is the same version, or a newer version, as the instance from which the backup was taken. Although you cannot restore a database from a SQL Server 2017 instance to an earlier version of Analysis Services, it is common practice to restore an older version database, such as SQL Server 2012, on a newer SQL Server 2017 instance.

Restore location must be the same server type. Tabular databases can only be restored to Analysis Services running in tabular mode. Multidimensional databases require an instance running in multidimensional mode.

Preparing for Backup

Use the following checklist to prepare for backup:

- Check the location where the backup file will be stored. If you are using a remote location, you must

specify it as a UNC folder. Verify that you can access the UNC path.

- Check the permissions on the folder to ensure that the Analysis Services service account has Read/Write permissions on the folder.
- Check for sufficient disk space on the target server.
- Check for existing files of the same name. If a file of the same name already exists, backup will fail unless you specify options to overwrite the file.

Backing Up a Multidimensional or a Tabular Database

Administrators can back up an Analysis Services database to a single Analysis Services backup file (.abf), regardless of size of the database. For step by step instructions, see How to Backup an Analysis Services Database (TechMantra) and Automate Backup an Analysis Services Database (TechMantra).

> **NOTE**
> Power Pivot for SharePoint, used for loading and querying Power Pivot data models in a SharePoint environment, loads its models from SharePoint content databases. These content databases are relational and run on the SQL Server relational database engine. As such, there is no Analysis Services backup and restore strategy for Power Pivot data models. If you have a disaster recovery plan in place for SharePoint content, that plan encompasses the Power Pivot data models stored in the content databases.

Remote Partitions

If the Analysis Services database contains remote partitions, the remote partitions should also be backed up. When you back up a database with remote partitions, all the remote partitions on each remote server are backed up to a single file on each of those remote servers respectively. Therefore, if you want to create those remote backups off their respective host computers, you will have to manually copy those files to the designated storage areas.

Contents of a backup file

Backing up an Analysis Services database produces a backup file whose contents vary depending upon the storage mode used by the database objects. This difference in backup content results from the fact that each storage mode actually stores a different set of information within an Analysis Services database. For example, multidimensional hybrid OLAP (HOLAP) partitions and dimensions store aggregations and metadata in the Analysis Services database, while relational OLAP (ROLAP) partitions and dimensions only store metadata in the Analysis Services database. Because the actual contents of an Analysis Services database vary based on the storage mode of each partition, the contents of the backup file also vary. The following table associates the contents of the backup file to the storage mode used by the objects.

STORAGE MODE	CONTENTS OF BACKUP FILE
Multidimensional MOLAP partitions and dimensions	Metadata, source data, and aggregations
Multidimensional HOLAP partitions and dimensions	Metadata and aggregations
Multidimensional ROLAP partitions and dimensions	Metadata
Tabular In-Memory Models	Metadata and source data
Tabular DirectQuery Models	Metadata only

> **NOTE**
> Backing up an Analysis Services database does not back up the data in any underlying data sources, such as a relational database. Only the contents of the Analysis Services database are backed up.

When you back up an Analysis Services database, you can choose from the following options:

- Whether to compress all database backups. The default is to compress backups.
- Whether to encrypt the contents of the backup files and require a password before the file can be unencrypted and restored. By default, the backed up data is not encrypted.

> **IMPORTANT**
> For each backup file, the user who runs the backup command must have permission to write to the backup location specified for each file. Also, the user must have one of the following roles: a member of a server role for the Analysis Services instance, or a member of a database role with Full Control (Administrator) permissions on the database to be backed up.

For more information about backing up an Analysis Services database, see Backup Options.

Restoring an Analysis Services Database

Administrators can restore an Analysis Services database from one or more backup files.

> **NOTE**
> If a backup file is encrypted, you must provide the password specified during backup before you can use that file to restore an Analysis Services database.

During restoration, you have the following options:

- You can restore the database using the original database name, or you can specify a new database name.
- You can overwrite an existing database. If you choose to overwrite the database, you must expressly specify that you want to overwrite the existing database.
- You can choose whether to restore existing security information or skip security membership information.
- You can choose to have the restore command change the restoration folder for each partition being restored. Local partitions can be restored to any folder location that is local to the Analysis Services instance to which the database is being restored. Remote partitions can be restored to any folder on any server, other than the local server; remote partitions cannot become local.

> **IMPORTANT**
> For each backup file, the user who runs the restore command must have permission to read from the backup location specified for each file. To restore an Analysis Services database that is not installed on the server, the user must also be a member of the server role for that Analysis Services instance. To overwrite an Analysis Services database, the user must have one of the following roles: a member of the server role for the Analysis Services instance, or a member of a database role with Full Control (Administrator) permissions on the database to be restored.

> **NOTE**
> After restoring an existing database, the user who restored the database might lose access to the restored database. This loss of access can occur if, at the time that the backup was performed, the user was not a member of the server role or was not a member of the database role with Full Control (Administrator) permissions.

For more information about restoring an Analysis Services database, see Restore Options.

See Also

Backing Up, Restoring, and Synchronizing Databases (XMLA)

Backup Options

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

There are many ways to back up your Microsoft SQL Server Analysis Services databases and they all require that you have server administrator and database administrator permissions. You can open the **Backup** dialog box in SQL Server Management Studio, select the appropriate options configuration, and then run the backup from the dialog box itself. Or, you can create a script using the settings already specified in the file; the script can then be saved and run as frequently as required.

Backup and Synchronize

If the database is located on a remote instance of Analysis Services, you can use the synchronization feature to back up the database to the local instance. Development builds of a database can be moved into production in this manner. You can also use the conventional, file based, backup and restore to move the development build into production, but synchronization provides additional functionality. For example, you can have security settings which are different for the development and production computers; synchronization will provide you the option to maintain those settings and synchronize all objects other than roles. Also, synchronization typically does an incremental update of those objects which are different for the source and destination computers. This kind of incremental backup is not available using the backup/restore feature. For more information, see Synchronize Analysis Services Databases.

> **IMPORTANT**
>
> The Analysis Services service account must have permission to write to the backup location specified for each file. Also, the user must have one of the following roles: administrator role on the Analysis Services instance, or a member of a database role with Full Control (Administrator) permissions on the database to be backed up.

See Also

Backup Database Dialog Box (Analysis Services - Multidimensional Data)
Backup and Restore of Analysis Services Databases
Backup Element (XMLA)
Backing Up, Restoring, and Synchronizing Databases (XMLA)

Restore Options

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

There are many ways to restore your Microsoft SQL Server Analysis Services databases, all of which require that you have administrator permissions for both the server computer and the Analysis Services database. To restore an Analysis Services database, you can open the **Restore Database** dialog box in SQL Server Management Studio, select the appropriate options configuration and then run the restore from the dialog box. Or, you can create a script using the settings already specified in the file; the script can then be saved and run as often as needed. In this way, the restore is completed by using XMLA, as described in the next section.

Restoring Databases Using XMLA

The XMLA Restore command is a way to automate the restore process by running a restore based on an .abf file. The Restore command has a number of properties that can be set to specify security definitions, where remote partitions should be stored, and the relocation of relational OLAP (ROLAP) objects. For more information, see Restore Element (XMLA).

> **IMPORTANT**
>
> For each backup file, the user who runs the restore command must have permission to read from the backup location specified for each file. To restore an Analysis Services database that is not installed on the server, the user must also be a member of the server role for that Analysis Services instance. To overwrite an Analysis Services database, the user must have one of the following roles: a member of the server role for the Analysis Services instance, or a member of a database role with Full Control (Administrator) permissions on the database to be restored.

> **NOTE**
>
> After restoring an existing database, the user who restored the database might lose access to the restored database. This loss of access can occur if, at the time that the backup was performed, the user was not a member of the server role or was not a member of the database role with Full Control (Administrator) permissions.

See Also

Restore Database Dialog Box (Analysis Services - Multidimensional Data)
Backup and Restore of Analysis Services Databases
Restore Element (XMLA)
Backing Up, Restoring, and Synchronizing Databases (XMLA)

Document and Script an Analysis Services Database

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After an Analysis Services database is deployed, you can use SQL Server Management Studio to output the metadata of the database, or of an object contained in the database, as an XML for Analysis (XMLA) script. You can output this script to a new **XMLA Query Editor** window, to a file, or to the Clipboard. For more information about XMLA, see Analysis Services Scripting Language (ASSL for XMLA).

The generated XMLA script uses Analysis Services Scripting Language (ASSL) elements to define the objects contained by the script. If you generated a CREATE script, the resulting XMLA script contains an XMLA **Create** command and ASSL elements that can be used to create the entire Analysis Services database structure on an instance. If you generated an ALTER script, the resulting XMLA script contains an XMLA **Alter** command and ASSL elements that can be used to restore the structure of an existing Analysis Services database to the state of the database at the time the script was created.

You can use the generated XMLA script from an Analysis Services database in many ways, including:

- To maintain a backup script that allows you to recreate all the database objects and permissions.
- To create or update database development code.
- To create a test or development environment from an existing schema.

See Also

Modify or Delete an Analysis Services Database
Alter Element (XMLA)
Create Element (XMLA)

Modify or Delete an Analysis Services Database

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can change the name and description of an Analysis Services database before deployment in SQL Server Data Tools (SSDT) and after deployment in SQL Server Management Studio. You can also adjust additional settings on an Analysis Services database, depending on the environment.

> **NOTE**
> You cannot change database properties using SQL Server Data Tools (SSDT) in online mode.

Modifying Databases Using SQL Server Management Studio

Once an Analysis Services database is deployed, you can use SQL Server Management Studio to change the impersonation mode used by Analysis Services when connecting to data sources contained by the database. The impersonation mode allows you to specify the security context used by Analysis Services when attempting to connect to a data source for processing, browsing, or drillthrough purposes.

Modifying Databases Using SQL Server Data Tools

You can use SQL Server Data Tools (SSDT) in project mode to modify the translations for the caption and description of an Analysis Services project used to define a database. For more information about using translations in an Analysis Services database, see Globalization scenarios for Analysis Services.

You can also set the aliases and aggregation functions associated with account types used by account attributes in dimensions contained by the database. Aliases allow you to select the business-specific terminology used by your organization for the account types in a chart of accounts. The account types are used by members of an account attribute to indicate how to aggregate measures over each member by using the aggregation functions specified for each account type contained in the database. For more information about account attributes, see Attributes and Attribute Hierarchies.

Deleting Databases

Deleting an existing Analysis Services database deletes the database and all cubes, dimensions, and mining models in the database. You can only delete an existing Analysis Services database in SQL Server Management Studio.

To delete an Analysis Services database

1. Connect to an Analysis Services instance.
2. In **Object Explorer**, expand the node for the connected Analysis Services instance and ensure that the object to be deleted is visible.
3. Right-click the object to be deleted and select **Delete**.
4. In the **Delete Object** dialog box, click **OK**.

See Also

Move an Analysis Services Database

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

There are often situations when an Analysis Services database administrator (dba) wants to move a multidimensional or tabular model database to a different location. These situations are often driven by business needs, such as moving the database to a different disk for better performance, gaining room for database growth, or to upgrade a product.

A database can be moved in many ways. This document explains the following common scenarios:

- Interactively using SSMS
- Programmatically using AMO
- By script using XMLA

All scenarios require the user to access the database folder and to use a method for moving the files to the desired final destination.

> **NOTE**
> Detaching a database without assigning a password to it leaves the database in an unsecured state. We recommend assigning a password to the database to protect confidential information. Also, the corresponding access security should be applied to the database folder, sub-folders, and files to prevent unauthorized access to them.

Procedures

Moving a database interactively using SSMS

1. Locate the database to be moved in the left or right pane of SSMS.
2. Right-click on the database and select **Detach...**
3. Assign a password to the database to be detached, then click **OK** to execute the detach command.
4. Use any operating system mechanism or your standard method for moving files to move the database folder to the new location.
5. Locate the **Databases** folder in the left or right pane of SSMS.
6. Right-click on the **Databases** folder and select **Attach...**
7. In the **folder** text box, type the new location of the database folder. Alternatively, you can use the browse button (**...**) to locate the database folder.
8. Select the **ReadWrite** mode for the database.
9. Type the password used in step 3 and click **OK** to execute the attach command.

Moving a database programmatically using AMO

1. In your C# application, adapt the following sample code and complete the indicated tasks.

```
private void MoveDb(Server server, string dbName,
```

```
string dbInitialLocation, string dbFinalLocation,
```

```
string dbPassword, ReadWriteMode dbReadWriteMode)
{
//Verify dbInitialLocation exists before continuing
if (server.Databases.ContainsName(dbName))
{
Database db;
//Save current cursor and change cursor to Cursors.WaitCursor
db = server.Databases[dbName];
db.Detach(dbPassword);
//Add your own code to copy the database files to the destination where you intend to attach the database
//Verify dbFinalLocation exists before continuing
server.Attach(dbFinalLocation, dbReadWriteMode, dbPassword);
//Restore cursor to its original
}
}
```

1. In your C# application, invoke `MoveDb()` with the necessary parameters.
2. Compile and execute your code to move the database.

Moving a database by script using XMLA

1. Open a new XMLA tab in SSMS.
2. Copy the following script template for XMLA

```
<Detach xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
<Object>
<DatabaseID>%dbName%</DatabaseID>
<Password>%password%</Password>
</Object>
</Detach>
```

1. Replace `%dbName%` with the name of the database and `%password%` with the password. The % characters are part of the template and must be removed.
2. Execute the XMLA command.
3. Use any operating system mechanism or your standard method for moving files to move the database folder to the new location.
4. Copy the following script template for XMLA in a new XMLA tab

```
<Attach xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
<Folder>%dbFolder%</Folder>
```

```
<ReadWriteMode xmlns="http://schemas.microsoft.com/analysisservices/2008/engine/100">%ReadOnlyMode%
</ReadWriteMode>
```

```
</Attach>
```

1. Replace `%dbFolder%` with the complete UNC path of the database folder, `%ReadOnlyMode%` with the corresponding value **ReadOnly** or **ReadWrite**, and `%password%` with the password. The % characters are part of the template and must be removed.

2. Execute the XMLA command.

See Also

Attach
Detach
Attach and Detach Analysis Services Databases
Database Storage Location
Database ReadWriteModes
Attach Element
Detach Element
ReadWriteMode Element
DbStorageLocation Element

Rename a Multidimensional Database (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The manner in which you change the name of a Microsoft SQL Server Analysis Services database depends upon how you connect to the Analysis Services database. To change the name of an existing database, you must connect in online mode. To change the name of the database into which objects in an Analysis Services project will be instantiated, you must connect in project mode.

To change the database name in online mode

1. Using SQL Server Data Tools, connect directly to the Analysis Services database.
2. In Solution Explorer, right-click the database and then click **Edit Database**.
3. In the **Database name** text box, change the database name.
4. Click **Save** or **Save All** on the toolbar, click **Save Selected Items** or **Save All** on the **File** menu, or close the **Database Designer** and then click **Save** when prompted.

   The database name is updated in the Analysis Services instance and the database object in Solution Explorer is refreshed.

To change the database name in project mode

1. Open the Analysis Services project.
2. In Solution Explorer, right-click the Analysis Services project and then click **Properties**.
3. In the **Property Pages** dialog box, click **Deployment** in the **Configuration Properties** section.
4. Change the **Database** property to the new database name.

   The next time the Analysis Services project is deployed, it will be deployed to this new database name. If this database already exists, it will be overwritten.

To change the database name using SQL Server Management Studio

- Right-click the Analysis Services database and edit the Name property.

See Also

Server Properties in Analysis Services
Set Multidimensional Database Properties (Analysis Services)
Configure Analysis Services Project Properties (SSDT)
Deploy Analysis Services Projects (SSDT)

Compatibility Level of a Multidimensional Database (Analysis Services)

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Analysis Services, the database compatibility level property determines the functional level of a database. Compatibility levels are unique to each model type. For example, a compatibility level of **1100** has a different meaning depending on whether the database is multidimensional or tabular.

This topic describes compatibility level for multidimensional databases only. For more information about tabular solutions, see Compatibility Level for Tabular models in Analysis Services.

> **NOTE**
> Tabular models have additional database compatibility levels that are not applicable to multidimensional models. Compatibility level **1103** does not exist for multidimensional models. See What is new for the Tabular model in SQL Server 2012 SP1 and compatibility level for more information about **1103** for tabular solutions.

Compatibility Levels for multidimensional databases

Currently, the only multidimensional database behavior that varies by functional level is string storage architecture. By raising the database compatibility level, you can override the 4 gigabyte maximum limit for string storage of measures and dimensions.

For a multidimensional database, valid values for the **CompatibilityLevel** property include the following:

SETTING	DESCRIPTION
1050	This value is not visible in script or tools, but it corresponds to databases created in SQL Server 2005 (9.x), SQL Server 2008, or SQL Server 2008 R2. Any database that does not have **CompatibilityLevel** explicitly set is implicitly running at the **1050** level.
1100	This is the default value for new databases that you create in SQL Server 2012 (11.x) or SQL Server 2017. You can also specify it for databases created in earlier versions of Analysis Services to enable the use of features that are supported only at this compatibility level (namely, increased string storage for dimension attributes or distinct count measures that contain string data). Databases that have a **CompatibilityLevel** set to **1100** get an additional property, **StringStoresCompatibilityLevel**, that lets you choose alternative string storage for partitions and dimensions.

> **WARNING**
> Setting the database compatibility to a higher level is irreversible. After you increase the compatibility level to **1100**, you must continue to run the database on newer servers. You cannot rollback to **1050**. You cannot attach or restore an **1100** database on a server version that is earlier than SQL Server 2012 (11.x) or SQL Server 2017.

Prerequisites

Database compatibility levels are introduced in SQL Server 2012 (11.x). You must have SQL Server 2012 (11.x)Analysis Services or higher to view or set the database compatibility level.

The database cannot be a local cube. Local cubes do not support the **CompatibilityLevel** property.

The database must have been created in a previous release (SQL Server 2008 R2 or earlier) and then attached or restored to a SQL Server 2012 (11.x)Analysis Services or higher server. Databases deployed to SQL Server 2012 are already at **1100** and cannot be downgraded to run at a lower level.

Determine the existing database compatibility level for a multidimensional database

The only way to view or modify the database compatibility level is through XMLA. You can view or modify the XMLA script that specifies your database in SQL Server Management Studio.

If you search the XMLA definition of a database for the property **CompatibilityLevel** and it does not exist, you most likely have a database at the **1050** level.

Instructions for viewing and modifying the XMLA script are provided in the next section.

Set the database compatibility level in SQL Server Management Studio

1. Before raising the compatibility level, backup the database in case you want to reverse your changes later.
2. Using SQL Server Management Studio, connect to the SQL Server 2017Analysis Services server that hosts the database.
3. Right-click the database name, point to **Script Database as**, point to **ALTER to**, and then select **New Query Editor Window**. An XMLA representation of the database will open in a new window.
4. Copy the following XML element:

   ```
   <ddl200:CompatibilityLevel>1100</ddl200:CompatibilityLevel>
   ```

5. Paste it after the `</Annotations>` closing element and before the `<Language>` element. The XML should look similar to the following example:

   ```
   </Annotations>
   <ddl200:CompatibilityLevel>1100</ddl200:CompatibilityLevel>
   <Language>1033</Language>
   ```

6. Save the file.
7. To run the script, click **Execute** on the Query menu or press F5.

Supported Operations that Require the Same Compatibility Level

The following operations require that the source databases share the same compatibility level.

1. Merging partitions from different databases is supported only if both databases share the same compatibility level.
2. Using linked dimensions from another database requires the same compatibility level. For example, if you want to use a linked dimension from a SQL Server 2008 R2 database in a SQL Server 2012 (11.x) database, you must port the SQL Server 2008 R2 database to a SQL Server 2012 (11.x) server and set the compatibility level to **1100**.
3. Synchronizing servers is only supported for servers that share the same version and database compatibility level.

Next Steps

After you increase the database compatibility level, you can set the **StringStoresCompatibilityLevel** property in SQL Server Data Tools. This increases string storage for measures and dimensions. For more information about this feature, see Configure String Storage for Dimensions and Partitions.

See Also

Backing Up, Restoring, and Synchronizing Databases (XMLA)

Set Multidimensional Database Properties (Analysis Services)

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

There are a number of Microsoft SQL Server Analysis Services database properties that you can configure in the SQL Server Data Tools (SSDT) database designer.

In this designer, you can perform the following types of tasks:

- If you are connected to the Analysis Services database in online mode, you can change the name of the Analysis Services database. If you are working in project mode, you can change the database name for the next deployment of the project. For more information, see Rename a Multidimensional Database (Analysis Services) and Configure Analysis Services Project Properties (SSDT).
- You can provide a description of the database that can be presented to users. You can also view the name of the database, but cannot change it. To change the database name, you must edit the properties of the project.
- You can provide translations for the database name and description for one or more languages. For more information, see Cube Translations, Dimension Translations, and Translation support in Analysis Services.
- You can view and modify default account type mappings. Account type mappings are used when one or more measures use the *ByAccount* aggregation function. For each account type, you can specify an alias and modify the default aggregation function associated with the account type. For more information modifying the default aggregation, see Define Semiadditive Behavior.

Database Properties

In addition to the above, there are a number of properties of a database that you can configure in the Properties window.

PROPERTY	DESCRIPTION
Aggregation Prefix	The common prefix that can be used for aggregation names for all of the partitions in a database. For more information, see AggregationPrefix Element (ASSL).
Collation	When the Analysis Services project is deployed to an Analysis Services instance, the database will inherit from the Collation server property unless a different value is provided here.
DataSourceImpersonationInfo	Specifies the default impersonation mode for all data source objects in the database. This is the mode that the Analysis Services service uses when processing objects, synchronizing servers, and executing the OpenQuery and SystemOpenSchema data mining statements.

PROPERTY	DESCRIPTION
Estimated Size	Provides an estimated size of the database files on disk. If data is stored in multiple locations, this estimate will be limited to just the data files stored under the database folder. **EstimatedSize** can also be used as a basis for estimating memory. Typically, memory requirements are larger than the size of data on disk due to additional data structures that are created when the database is loaded into memory. To further estimate memory requirements, you can also use Task Manager to look at the Analysis Services process memory before and after processing the database and observe the memory utilized as a method for understanding the memory requirements of the database.
Language	When the Analysis Services project is deployed to an Analysis Services instance, the database will inherit from the Language server property unless a different value is provided here.
MasterDataSource ID	Used with remote partitions. For more information, see Remote Partitions.

See Also

Database Properties Dialog Box (SSAS - Multidimensional)
Configure Analysis Services Project Properties (SSDT)

Database Storage Location

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

There are often situations when an Analysis Services database administrator (dba) wants a certain database to reside outside of the server data folder. These situations are often driven by business needs, such as improving performance or expanding storage. For these situations, the **DbStorageLocation** database property enables the Analysis Services dba to specify the database location in a local disk or network device.

DbStorageLocation database property

The **DbStorageLocation** database property specifies the folder where Analysis Services creates and manages all the database data and metadata files. All metadata files are stored at the **DbStorageLocation** folder, with the exception of the database metadata file, which is stored in the server data folder. There are two important considerations when setting the value of **DbStorageLocation** database property:

- The **DbStorageLocation** database property must be set to an existing UNC folder path or an empty string. An empty string is the default for the server data folder. If the folder does not exist, an error will be raised when you execute a **Create**, **Attach**, or **Alter** command.
- The **DbStorageLocation** database property cannot be set to point to the server data folder or any one of its subfolders. If the location points to the server data folder or any one of its subfolders, an error will be raised when you execute a **Create**, **Attach**, or **Alter** command.

> **IMPORTANT**
>
> We recommend that set your UNC path to use a Storage Area Network (SAN), iSCSI-based network, or a locally attached disk. Any UNC path to a network share or any high latency remote storage solution leads to an unsupported installation.

DbStorageLocation compared to StorageLocation

DbStorageLocation specifies the folder where all the database data and metadata files reside, whereas **StorageLocation** specifies the folder where one or more partitions of a cube reside. **StorageLocation** can be set independently of **DbStorageLocation**. This is an Analysis Services dba decision based on the expected results, and many times the usage of one property or the other will overlap.

DbStorageLocation Usage

The **DbStorageLocation** database property is used as part of a **Create** database command in a **Detach**/**Attach** database commands sequence, in a **Backup**/**Restore** database commands sequence, or in a **Synchronize** database command. Changing the **DbStorageLocation** database property is considered a structural change in the database object. This means that all metadata must be recreated and the data reprocessed.

> **IMPORTANT**
>
> You should not change the database storage location by using an **Alter** command. Instead, we recommend that you use a sequence of **Detach**/**Attach** database commands (see Move an Analysis Services Database, Attach and Detach Analysis Services Databases).

See Also

DbStorageLocation
Attach and Detach Analysis Services Databases
Move an Analysis Services Database
DbStorageLocation Element
Create Element (XMLA)
Attach Element
Synchronize Element (XMLA)

Database ReadWriteModes

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

There are often situations when an Analysis Services database administrator (dba) wants to change a read/write database to a read-only database, or vice versa. These situations are often driven by business needs, such as sharing the same database folder among several servers for scaling out a solution and improving performance. For these situations, the **ReadWriteMode** database property enables the Analysis Services dba to easily change the database operating mode.

ReadWriteMode database property

The **ReadWriteMode** database property specifies whether the database is in read/write mode or in read-only mode. These are the only two possible values of the property. When the database is in read-only mode, no changes or updates can be applied to the database. However, when the database is in read/write mode, changes and updates can occur. The **ReadWriteMode** database property is defined as a read-only property; it can only be set through an **Attach** command.

When a database is in read-only mode, certain restrictions are in place that affect the ordinary set of allowed operations to the database. See the following table for the restricted operations.

READONLY MODE	RESTRICTED OPERATIONS
XML/A commands Note: An error is raised when you execute any one of these commands.	**Create** **Alter** **Delete** **Process** **MergePartitions** **DesignAggregations** **CommitTransaction** **Restore** **Synchronize** **Insert** **Update** **Drop** Note: Cell writeback is allowed in databases set to read-only; however, the changes cannot be committed.

READONLY MODE	RESTRICTED OPERATIONS
MDX statements Note: An error is raised when you execute any one of these statements.	**COMMIT TRAN** **CREATE SESSION CUBE** **ALTER CUBE** **ALTER DIMENSION** **CREATE DIMENSION MEMBER** **DROP DIMENSION MEMBER** **ALTER DIMENSION** Note: Excel users cannot use the grouping feature in Pivot tables, because that feature is internally implemented by using **CREATE SESSION CUBE** commands.
DMX statements Note: An error is raised when you execute any one of these statements.	**CREATE [SESSION] MINING STRUCTURE** **ALTER MINING STRUCTURE** **DROP MINING STRUCTURE** **CREATE [SESSION] MINING MODEL** **DROP MINING MODEL** **IMPORT** **SELECT INTO** **INSERT** **UPDATE** **DELETE**
Background operations	Any background operations that would modify the database are disabled. This includes lazy processing and proactive caching.

ReadWriteMode Usage

The **ReadWriteMode** database property is to be used as part of an **Attach** database command. The **Attach** command allows the database property to be set to either **ReadWrite** or **ReadOnly**. The **ReadWriteMode** database property value cannot be updated directly because the property is defined as read-only. Databases are created with the **ReadWriteMode** property set to **ReadWrite**. A database cannot be created in read-only mode.

To switch the **ReadWriteMode** database property between **ReadWrite** and **ReadOnly**, you must issue a sequence of **Detach/Attach** commands.

All database operations, with the exception of **Attach**, keep the **ReadWriteMode** database property in its current state. For example, operations like **Alter**, **Backup**, **Restore**, and **Synchronize** preserve the **ReadWriteMode** value.

> **NOTE**
> Local cubes can be created from a read-only database.

See Also

Detach
Attach and Detach Analysis Services Databases
Move an Analysis Services Database
Detach Element
Attach Element

Synchronize Analysis Services Databases

10/26/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services includes a database synchronization feature that makes two Analysis Services databases equivalent by copying the data and metadata a database on a source server to a database on a destination server. Use the Synchronize Database feature to accomplish any of the following tasks:

- Deploy a database from a staging server onto a production server.
- Update a database on a production server with the changes made to the data and metadata in a database on a staging server.
- Generate XMLA script that can be run in the future to synchronize the databases.
- In distributed workloads where cubes and dimensions are processed on multiple servers, use database synchronization to merge the changes into a single database.

Database synchronization is initiated on the destination server, pulling data and metadata into a database copy on the source server. If the database does not exist, it will be created. Synchronization is a one-way, one-time operation that concludes once the database is copied. It does not provide real-time parity between the databases.

You can re-sync databases that already exist on source and destination servers to pull the latest changes from a staging server into a production database. Files on the two servers will be compared for changes and those that are different will be updated. An existing database on a destination server remains available while synchronization occurs in the background. Users can continue to query the destination database while synchronization is in progress. After synchronization finishes, Analysis Services automatically switches the users to the newly copied data and metadata, and drops the old data from the destination database.

To synchronize databases, run the Synchronize Database Wizard to immediately synchronize the databases, or use it to generate a synchronization script that you can run later. Either approach can be used to increase the availability and scalability of your Analysis Services databases and cube.

> **NOTE**
> The following whitepapers, written for previous versions of Analysis Services, still apply to scalable multidimensional solutions built using SQL Server 2012. For more information, see Scale-Out Querying with Analysis Services and Scale-Out Querying for Analysis Services with Read-Only Databases

Prerequisites

On the destination (or target) server from which you initiate database synchronization, you must be a member of the Analysis Services server administrator role. On the source server, your Windows user account must have Full Control permissions on the source database. If you are synchronizing database interactively, remember that synchronization runs under the security context of your Windows user identity. If your account is denied access to specific objects, those objects will be excluded from the operation. For more information about server administrator roles and database permissions, see Grant server admin rights to an Analysis Services instance and Grant database permissions (Analysis Services).

TCP port 2383 must be open on both servers to allow remote connections between default instances. For more information about creating an exception in Windows Firewall, see Configure the Windows Firewall to Allow

Analysis Services Access.

Both the source and destination servers must be the same version and service pack. Because model metadata is also synchronized, to ensure compatibility the build number for both servers should be the same. The edition of each installation must support database synchronization. In SQL Server 2017, database synchronization is supported in enterprise, developer, and business intelligence editions. For more information about features in each edition, see Editions and Supported Features for SQL Server 2016.

The server deployment mode must be identical on each server. If the database you are synchronizing is multidimensional, both source and destination servers must be configured for multidimensional server mode. For more information about deployment modes, see Determine the Server Mode of an Analysis Services Instance.

Turn off lazy aggregation processing if you are using it on the source server. Aggregations that are processing in the background can interfere with database synchronization. For more information about setting this server property, see OLAP Properties.

> **NOTE**
>
> Database size is a factor in determining whether synchronization is a suitable approach. There are no hard requirements, but if synchronization is too slow, consider synchronizing multiple servers in parallel, as described in this technical paper: Analysis Services Synchronization Best Practices.

Synchronize Database Wizard

Use the Synchronize Database Wizard to perform one-way synchronization from a source to a destination database, or to generate a script that specifies a database synchronization operation. You can synchronize both local and remote partitions during the synchronization process and choose whether to include roles.

The Synchronize Database Wizard guides you through the following steps:

- Select the source instance and database from which to synchronize.
- Select storage locations for local partitions on the destination instance.
- Select storage locations for remote partitions on other destination instances.
- Select the level of security and membership information to copy from the source instance and database to the destination instance.
- Select whether to synchronize immediately or to save the XML for Analysis (XMLA) **Synchronize** command generated by the Synchronize Database Wizard to a script file for later synchronization.

By default, the wizard synchronizes all data and metadata, other than membership in existing security groups. You can also copy all security settings or ignore all security settings when synchronizing the data and metadata.

Run the wizard

1. In SQL Server Management Studio, connect to the instance of Analysis Services that will run the destination database. For example, if you are deploying a database to a production server, you would run the wizard on the production server.
2. In Object Explorer, right-click the **Databases** folder, then click **Synchronize**.
3. Specify the source server and source database. In the Select Database to Synchronize page, in **Source Server** and **Source Database**, type the name of the source server and source database. For example, if you are deploying from a test environment to a production server, the source is the database on the staging server.

Destination Server displays the name of the Analysis Services instance with which the data and metadata from the database selected in **Source database** is synchronized.

Synchronization will occur for source and destination databases that have the same name. If the destination server already has a database that shares the same name as the source database, the destination database will be updated with the metadata and data of the source. If the database does not exist, it will be created on the destination server.

4. Optionally, change location for local partition. Use the **Specify Locations for Local Partitions** page to indicate where local partitions should be stored on the destination server.

> **NOTE**
> This page appears only if at least one local partition exists in the specified database.

If a set of partitions are installed on drive C of the source server, the wizard lets you copy this set of partitions to a different location on the destination server. If you do not change the default locations, the wizard deploys the measure group partitions within each cube on the source server to the same locations on the destination server. Similarly, if the source server uses remote partitions, the same remote partitions will be used on the destination server.

The **Locations** option displays a grid listing the source folder, destination folder, and estimated size of the local partitions to be stored on the destination instance. The grid contains the following columns:

Source Folder
Displays the folder name on the source Analysis Services instance that contains the local partition. If the column contains the value "(Default)", the default location for the source instance contains the local partition.

Destination Folder
Displays the folder name on the destination Analysis Services instance into which the local partition is to be synchronized. If the column contains the value, "(Default)", the default location for the destination instance contains the local partition.

Click the ellipsis (**...**) button to display the **Browse for Remote Folder** dialog box and specify a folder on the destination instance into which the local partitions stored in the selected location should be synchronized.

> **NOTE**
> This column cannot be changed for local partitions stored in the default location for the source instance.

Size
Displays the estimated size of the local partition.

The **Partitions in selected location** option displays a grid that describes the local partitions stored in the location on the source Analysis Services instance specified in the **Source Folder** column of the selected row in **Locations**.

Cube
Displays the name of the cube that contains the partition.

Measure Group
Displays the name of the measure group in the cube that contains the partition.

Partition Name

Displays the name of the partition.

Size(Mb)
Displays the size in megabytes (MB) of the partition.

5. Optionally, change location for remote partitions.Use the **Specify Locations for Remote Partitions** page to indicate if remote partitions managed by the specified database on the source server should be synchronized, and to specify a destination Analysis Services instance and database into which the selected remote partitions should be stored.

> **NOTE**
> This page appears only if at least one remote partition is managed by the specified database on the source Analysis Services instance.

The **Locations** option displays a grid that lists details about locations in which remote partitions for the source database are stored, including source and destination information and the storage size used by each location, available from the selected database. The grid contains the following columns:

Sync
Select to include a location that contains the remote partitions during synchronization.

> **NOTE**
> If this option is not selected for a location, remote partitions that are contained in that location will not be synchronized.

Source Server
Displays the name of the Analysis Services instance that contains remote partitions.

Source Folder
Displays the folder name on the Analysis Services instance that contains remote partitions. If the column contains the value "(Default)", the default location for the instance displayed in **Source Server** contains remote partitions.

Destination Server
Displays the name of the Analysis Services instance into which the remote partitions stored in the location specified in **Source Server** and **Source Folder** should be synchronized.

Click the ellipsis (**...**) button to display the **Connection Manager** dialog box and specify an Analysis Services instance into which the remote partitions stored in the selected location should be synchronized.

Destination Folder
Displays the folder name on the destination Analysis Services instance into which the remote partition is to be synchronized. If the column contains the value, "(Default)", the default location for the destination instance should contain the remote partition.

Click the ellipsis (**...**) button to display the **Browse for Remote Folder** dialog box and specify a folder on the destination instance into which the remote partitions stored in the selected location should be synchronized.

Size
Displays the estimated size of remote partitions stored in the location.

The **Partitions in selected location** displays a grid that describes the remote partitions stored in the location on the source Analysis Services instance specified in the **Source Folder** column of the selected

row in **Locations**. The grid contains the following columns:

Cube
Displays the name of the cube that contains the partition.

Measure Group
Displays the name of the measure group in the cube that contains the partition.

Partition Name
Displays the name of the partition.

Size(Mb)
Displays the size in megabytes (MB) of the partition.

6. Specify whether user permission information should be included, and whether compression should be used. By default, the wizard compresses all data and metadata prior to copying the files to the destination server. This option results in a faster file transmission. Files are uncompressed once they reach the destination server.

   Copy all
   Select to include security definitions and membership information during synchronization.

   Skip membership
   Select to include security definitions, but exclude membership information, during synchronization.

   Ignore all
   Select to ignore the security definition and membership information currently in the source database. If a destination database is created during synchronization, no security definitions or membership information will be copied. If the destination database already exists and has roles and memberships, that security information will be preserved.

7. Choose the synchronization method. You can synchronize immediately or generate a script that is saved to a file. By default, the file is saved with an .xmla extension and placed in your Documents folder.

8. Click **Finish** to synchronize. After verifying the options on the **Completing the Wizard** page, click **Finish** again.

Next Steps

If you did not synchronize roles or membership, remember to specify user access permissions now on the destination database.

See Also

Synchronize Element (XMLA)
Deploy Model Solutions Using XMLA
Deploy Model Solutions Using the Deployment Wizard

Switch an Analysis Services database between ReadOnly and ReadWrite modes

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services database administrators can change the read/write mode of a Tabular or Multidimensional database as part of larger effort that distributes a query workload among multiple query-only servers.

A database mode can be switched in several ways. This document explains the following common scenarios:

- Interactively using SQL Server Management Studio
- Programmatically using AMO
- Script using XMLA or TMSL

Switch the read/write mode of a database interactively using Management Studio

1. In Object Explorer, right-click the database and select **Properties**.

   Note the location. An empty database storage location indicates that the database folder is located in the server data folder.

2. Right-click the database and select **Detach...**
3. Assign a password to the database to be detached, and then click **OK** to execute the detach command.
4. In Object Explorer, right-click the **Databases** folder and select **Attach...**
5. In the **folder** text box, type the original location of the database folder. Alternatively, you can use the browse button (**...**) to locate the database folder.
6. Select the read/write mode for the database.
7. Type the password and click **OK** to execute the attach command.

Switch the read/write mode to a database programmatically using AMO

In your C# application, invoke `SwitchReadWrite()` with the necessary parameters. Compile and execute your code to move the database.

```
private void SwitchReadWrite(Server server, string dbName, ReadWriteMode dbReadWriteMode)
{
    if (server.Databases.ContainsName(dbName))
    {
        Database db;
        string databaseLocation;
        db = server.Databases[dbName];
        databaseLocation = db.DbStorageLocation;

              if (databaseLocation == null)
            {
                 string dataDir = server.ServerProperties["DataDir"].Value;
                 string dataDir = server.ServerProperties["DataDir"].Value;
                 string dataDir = server.ServerProperties["DataDir"].Value;

    String[] possibleFolders = Directory.GetDirectories(dataDir, string.Concat(dbName,"*"),
SearchOption.TopDirectoryOnly);

   if (possibleFolders.Length > 1)
         {
         List<String> sortedFolders = new List<string>(possibleFolders.Length);
         sortedFolders.AddRange(possibleFolders);
         sortedFolders.Sort();
         databaseLocation = sortedFolders[sortedFolders.Count - 1];
         }
         else
         {
         databaseLocation = possibleFolders[0];
          }
        }
    db.Detach();
    server.Attach(databaseLocation, dbReadWriteMode);
    }
}
```

Switch the read/write mode to a database by script using XMLA

The following instructions apply to Multidimensional databases and Tabular databases at compatibility mode 1050, 1100, or 1103.

1. In Object Explorer, right-click the database and select **Properties**.

   Note the location. An empty database storage location indicates that the database folder is located in the server data folder.

2. Right-click the database and select **Detach...**

3. Open a new XMLA tab in Management Studio.

4. Copy the following script template for XMLA:

```
<Detach xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
   <Object>
      <DatabaseID>%dbName%</DatabaseID>
      <Password>%password%</Password>
   </Object>
</Detach>
```

5. Replace `%dbName%` with the name of the database and `%password%` with the password. The % characters are part of the template and must be removed.

6. Execute the XMLA command.

7. Copy the following script template for XMLA in a new XMLA tab

```
<Attach xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
   <Folder>%dbFolder%</Folder>
   <ReadWriteMode xmlns="http://schemas.microsoft.com/analysisservices/2008/engine/100">%ReadOnlyMode%
</ReadWriteMode>
</Attach>
```

8. Replace `%dbFolder%` with the complete UNC path of the database folder, `%ReadOnlyMode%` with the corresponding value **ReadOnly** or **ReadWrite**, and `%password%` with the password. The % characters are part of the template and must be removed.

9. Execute the XMLA command.

See Also

Detach
High availability and Scalability in Analysis Services
Attach and Detach Analysis Services Databases
Database Storage Location
Database ReadWriteModes
Attach Element
Detach Element
ReadWriteMode Element
DbStorageLocation Element

Processing a multidimensional model (Analysis Services)

12/7/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Processing is the step, or series of steps, in which Analysis Services loads data from a relational data source into a multidimensional model. For objects that use MOLAP storage, data is saved on disk in the database file folder. For ROLAP storage, processing occurs on demand, in response to an MDX query on an object. For objects that use ROLAP storage, processing refers to updating the cache before returning query results.

By default, processing occurs when you deploy a solution to the server. You can also process all or part of a solution, either ad hoc using tools such as Management Studio or SQL Server Data Tools, or on a schedule using Integration Services and SQL Server Agent. When making a structural change to the model, such as removing a dimension or changing its compatibility level, you will need to process again to synchronize the physical and logical aspects of the model.

This topic includes the following sections:

Prerequisites

Choosing a tool or approach

Processing Objects

Reprocessing Objects

Prerequisites

- Processing requires administrative permissions on the Analysis Services instance. If you are processing interactively from SQL Server Data Tools or Management Studio, you must be a member of the server administrator role on the Analysis Services instance. For processing that runs unattended, for example using an SSIS package that you schedule through SQL Server Agent, the account used to run the package must be a member of the server administrator role. For more information about setting administrator permissions, see Grant server admin rights to an Analysis Services instance.
- The account used to retrieve data is specified in the data source object, either as an impersonation option if you are using Windows authentication, or as the user name on the connection string if using database authentication. The account must have read permissions on relational data sources used by the model.
- The project or solution must be deployed before you can process any objects.

  Initially, during the early stages of model development, deployment and processing occur together. However, you can set options to process the model later, after you deploy the solution. For more information about deployment, see Deploy Analysis Services Projects (SSDT).

Choosing a tool or approach

You can process objects interactively using a client application such as SQL Server Data Tools or Management Studio, or a scripted operation that run as a SQL Server Agent job or SSIS package.

How you process a database varies considerably depending on whether the model is in active development or

in production. Once a model is deployed to a production server, processing must be tightly controlled to ensure the integrity and availability of multidimensional data. Because objects are interdependent, processing typically has a cascading effect across the model as other objects are also processed or unprocessed in tandem. If some objects are left in an unprocessed state, queries for that data will not resolve, breaking any reports or applications that use it. When developing a strategy for processing a production database, consider using script or SSIS packages that you have debugged and tested to avoid operator error or overlooked steps.

For more information, see Tools and Approaches for Processing (Analysis Services).

Processing Objects

Processing affects the following Analysis Services objects: measure groups, partitions, dimensions, cubes, mining models, mining structures, and databases. When an object contains one or more objects, processing the highest-level object causes a cascade of processing all the lower-level objects. For example, a cube typically contains one or more measure groups (each of which contains one or more partitions) and dimensions. Processing a cube causes processing of all the measure groups within the cube and the constituent dimensions that are currently in an unprocessed state. For more information about processing Analysis Services objects, see Processing Analysis Services Objects.

While the processing job is working, the affected Analysis Services objects can be accessed for querying. The processing job works inside a transaction and the transaction can be committed or rolled back. If the processing job fails, the transaction is rolled back. If the processing job succeeds, an exclusive lock is put on the object when changes are being committed, which means the object is temporarily unavailable for query or processing. During the commit phase of the transaction, queries can still be sent to the object, but they will be queued until the commit is completed.

During a processing job, whether an object is processed, and how it will be processed, depends on the processing option that is set for that object. For more information about the specific processing options that can be applied to each object, see Processing Options and Settings (Analysis Services).

Reprocessing Objects

Cubes that contain unprocessed elements have to be reprocessed before they can be browsed. Cubes in Analysis Services contain measure groups and partitions that must be processed before the cube can be queried. Processing a cube causes Analysis Services to process constituent dimensions of the cube if those dimensions are in an unprocessed state. After an object has been processed the first time, it must be reprocessed either partially or in full whenever one of the following situations occurs:

- The structure of the object changes, such as dropping a column in a fact table.
- The aggregation design for the object changes.
- The data in the object needs to be updated.

When you process objects in Analysis Services, you can select a processing option, or you can enable Analysis Services to determine the appropriate type of processing. The processing methods made available differ from one object to another, and are based on the type of object. Additionally, the methods available are based on what changes have occurred to the object since it was last processed. If you enable Analysis Services to automatically select a processing method, it will use the method that returns the object to a fully processed state in the least time. For more information, see Processing Options and Settings (Analysis Services).

See Also

Logical Architecture (Analysis Services - Multidimensional Data)
Database Objects (Analysis Services - Multidimensional Data)

Processing Options and Settings (Analysis Services)

5/16/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you process objects in Microsoft SQL Server Analysis Services, you can select a processing option to control the type of processing that occurs for each object. Processing types differ from one object to another, and by changes that have occurred to the object since it was last processed. If you enable Analysis Services to automatically select a processing method, it will use the method that returns the object to a fully processed state in the least time.

Processing settings let you control the objects that are processed, and the methods that are used to process those objects. Some processing settings are primarily used for batch processing jobs. For more information about batch processing, see Batch Processing (Analysis Services).

> **NOTE**
> This topic applies to multidimensional and data mining solutions. For information about tabular solutions, see Process Database, Table, or Partition (Analysis Services).

Processing Options

The following table describes the processing methods that are available in Analysis Services, and identifies the objects for which each method is supported.

MODE	APPLIES TO	DESCRIPTION
Process Default	Cubes, databases, dimensions, measure groups, mining models, mining structures, and partitions.	Detects the process state of database objects, and performs processing necessary to deliver unprocessed or partially processed objects to a fully processed state. If you change a data binding, Process Default will do a Process Full on the affected object.
Process Full	Cubes, databases, dimensions, measure groups, mining models, mining structures, and partitions.	Processes an Analysis Services object and all the objects that it contains. When Process Full is executed against an object that has already been processed, Analysis Services drops all data in the object, and then processes the object. This kind of processing is required when a structural change has been made to an object, for example, when an attribute hierarchy is added, deleted, or renamed.
Process Clear	Cubes, databases, dimensions, measure groups, mining models, mining structures, and partitions.	Drops the data in the object specified and any lower-level constituent objects. After the data is dropped, it is not reloaded.

MODE	APPLIES TO	DESCRIPTION
Process Data	Dimensions, cubes, measure groups, and partitions.	Processes data only without building aggregations or indexes. If there is data is in the partitions, it will be dropped before re-populating the partition with source data.
Process Add	Dimensions, measure groups, and partitions Note: **Process Add** is not available for dimension processing in Management Studio, but you can write XMLA script performs this action.	For dimensions, adds new members and updates dimension attribute captions and descriptions. For measure groups and partitions, adds newly available fact data and process only to the relevant partitions.
Process Update	Dimensions	Forces a re-read of data and an update of dimension attributes. Flexible aggregations and indexes on related partitions will be dropped.
Process Index	Cubes, dimensions, measure groups, and partitions	Creates or rebuilds indexes and aggregations for all processed partitions. For unprocessed objects, this option generates an error. Processing with this option is needed if you turn off Lazy Processing.
Process Structure	Cubes and mining structures	If the cube is unprocessed, Analysis Services will process, if it is necessary, all the cube's dimensions. After that, Analysis Services will create only cube definitions. If this option is applied to a mining structure, it populates the mining structure with source data. The difference between this option and the Process Full option is that this option does not iterate the processing down to the mining models themselves.
Process Clear Structure	Mining structures	Removes all training data from a mining structure.

Processing Settings

The following table describes the processing settings that are available for use when you create a process operation.

PROCESSING OPTION	DESCRIPTION	OPTION VALUE

PROCESSING OPTION	DESCRIPTION	OPTION VALUE
Parallel	Used for batch processing. This setting causes Analysis Services to fork off processing tasks to run in parallel inside a single transaction. If there is a failure, the result is a roll-back of all changes. You can set the maximum number of parallel tasks explicitly, or let the server decide the optimal distribution. The Parallel option is useful for speeding up processing.	
Sequential (Transaction Mode)	Controls the execution behavior of the processing job. Two options are available. When you process using **One Transaction**, all changes are committed after the processing job succeeds. This means that all Analysis Services objects affected by a particular processing job remain available for queries until the commit process. This makes the objects temporarily unavailable. Using **Separate Transactions** causes all objects that are affected by a process in processing job to be taken unavailable for queries as soon as that process succeeds.	**One Transaction**. The processing job runs as a transaction. If all processes inside the processing job succeed, all changes by the processing job are committed. If one process fails, all changes by the processing job are rolled back. **One Transaction** is the default value. **Separate Transactions**. Each process in the processing job runs as a stand-alone job. If one process fails, only that process is rolled back and the processing job continues. Each job commits all process changes at the end of the job.
Writeback Table Option	Controls how writeback tables are handled during processing. This option applies to writeback partitions in a cube.	**Use Existing**. Uses the existing writeback table. This is default value. **Create**. Creates a new writeback table and causes the process to fail if one already exists. **Create Always**. Creates a new writeback table even if one already exists. An existing table is deleted and replaced.

PROCESSING OPTION	DESCRIPTION	OPTION VALUE
Process Affected Objects	Controls the object scope of the processing job. An affected object is defined by object dependency. For example, partitions are dependent on the dimensions that determine aggregation, but dimensions are not dependent on partitions. **False** is the default setting.	**False**. The job processes the objects explicitly named in the job and all dependent objects. For example, if the processing job contains only dimensions, Analysis Services processes just those objects explicitly identified in the job. If the processing job contains partitions, partition processing automatically invokes processing of affected dimensions. **True**. The job processes the objects explicitly named in the job, all dependent objects, and all objects affected by the objects being processed without changing the state of the affected objects. For example, if the processing job contains only dimensions, Analysis Services also processes all partitions affected by the dimension processing for partitions that are currently in a processed state. Affected partitions that are currently in an unprocessed state are not processed. However, because partitions are dependent on dimensions, if the processing job contains only partitions, partition processing automatically invokes processing of affected dimensions, even when the dimension is currently in an unprocessed state.
Dimension Key Errors	Determines the action taken by Analysis Services when errors occur during processing. When you select **Use custom error configuration**, you can select values for the following actions to control error-handling behavior. When you select Use default error configuration, Analysis Services uses the error configuration that is set for each object being processed. If an object is set to use default configuration settings, Analysis Services uses the default settings that are listed for each option.	
	Key error action. If a key value does not yet exist in a record, one of these actions is selected to occur:	**Convert to unknown**. The key is interpreted as an unknown member. This is the default setting. **Discard record**. The record is discarded.

PROCESSING OPTION	DESCRIPTION	OPTION VALUE
	Processing error limit. Controls the number of errors processed by selecting one of these options:	**Ignore errors count**. This will enable processing to continue regardless of the number of errors. **Stop on error**. With this option, you control two additional settings. **Number of errors** lets you limit processing to the occurrence of a specific number of errors. **On error action** lets you determine the action when **Number of errors** is reached. You can select **Stop processing**, which causes the processing job to fail and roll back any changes, or **Stop logging**, which enables processing to continue without logging errors. **Stop on error** is the default setting with **Number of errors** set to **0** and **On error action** set to **Stop processing**.
	The following error conditions. You can set the option value to control specific error-handling behavior. When you select **Use default error configuration**, Analysis Services uses the error configuration that is set for each object being processed. If an object is set to use default configuration settings, Analysis Services uses the default settings that are listed for each option.	**Key not found**. Occurs when a key value exists in a partition but does not exist in the corresponding dimension. The default setting is **Report and continue**. Other settings are **Ignore error** and **Report and stop**. **Duplicate key**. Occurs when more than one key value exists in a dimension. The default setting is **Ignore error**. Other settings are **Report and continue** and **Report and stop**. **Null key converted to unknown**. Occurs when a key value is null and the **Key error action** is set to **Convert to unknown**. The default setting is **Ignore error**. Other settings are **Report and continue** and **Report and stop**. **Null key not allowed**. Occurs when **Key error action** is set to **Discard record**. The default setting is **Report and continue**. Other settings are **Ignore error** and **Report and stop**.

See Also

Processing a multidimensional model (Analysis Services)

Processing Analysis Services Objects

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Processing affects the following Microsoft SQL Server Analysis Services object types: Analysis Services databases, cubes, dimensions, measure groups, partitions, and data mining structures and models. For each object, you can specify the level of processing for the object, or you can specify the Process Default option to enable Analysis Services to automatically select the optimal level of processing. For more information about the different levels of processing for each object, see Processing Options and Settings (Analysis Services).

You should be aware of the consequences of processing behavior in order to reduce the occurrence of negative repercussions. For example, fully processing a dimension automatically sets all partitions dependent on that dimension to an unprocessed state. This causes affected cubes to become unavailable for query until the dependent partitions are processed.

This topic includes the following sections:

Processing a Database

Processing a Dimension

Processing a Cube

Processing a Measure Group

Processing a Partition

Processing Data Mining Structures and Models

Processing a Database

In Analysis Services, a database contains objects but not data. When you process a database, you direct the server to recursively process those objects that store data in the model, such as dimensions, partitions, mining structures, and mining models.

When you process a database, some or all partitions, dimensions, and mining models that the database contains are processed. The actual processing type varies depending on the state of each object and the processing option that you select. For more information, see Processing Options and Settings (Analysis Services).

Processing a Cube

A cube can be thought of as a wrapper object for measure groups and partitions. A cube is made of dimensions in addition to one or more measures, which are stored in partitions. Dimensions define how data is laid out in the cube. When you process a cube, an SQL query is issued to retrieve values from the fact table to populate each member in the cube with appropriate measure values. For any specific path to a node in the cube, there is a value or a calculable value.

When you process a cube, Analysis Services processes any unprocessed dimensions in the cube, and some or all partitions within the measure groups in the cube. The specifics depend on the state of the objects when processing starts and the processing option that you select. For more information about processing options, see Processing Options and Settings (Analysis Services).

Processing a cube creates machine-readable files that store relevant fact data. If there are aggregations created,

they are stored in aggregation data files. The cube is then available for browsing from the Object Explorer in Management Studio or Solution Explorer in SQL Server Data Tools

Processing a Dimension

When you process a dimension, Analysis Services formulates and runs queries against dimension tables to return information that is required for processing.

COUNTRY	SALES REGION	STATE
United States	West	California
United States	West	Oregon
United States	West	Washington

The processing itself turns the tabular data into usable hierarchies. These hierarchies are fully articulated member names that are internally represented by unique numeric paths. The following example is a text representation of a hierarchy.

[United States]
[United States].[West]
[United States].[West].[California]
[United States].[West].[Oregon]
[United States].[West].[Washington]

Dimension processing does not create or update calculated members, which are defined at the cube level. Calculated members are affected when the cube definition is updated. Also, dimension processing does not create or update aggregations. However, dimension processing can cause aggregations to be dropped. Aggregations are created or updated only during partition processing.

When you process a dimension, be aware that the dimension might be used in several cubes. When you process the dimension, those cubes are marked as unprocessed and become unavailable for queries. To process both the dimension and the related cubes at the same time, use the batch processing settings. For more information, see Batch Processing (Analysis Services).

Processing a Measure Group

When you process a measure group, Analysis Services processes some or all partitions within the measure group, and any unprocessed dimensions that participate in the measure group. Specifics of the processing job depend on the processing option that you select. You can process one or more measure groups in Analysis Services without affecting other measure groups in a cube.

> **NOTE**
> You can process individual measure groups programmatically, or by using Management Studio. You cannot process individual measure groups in SQL Server Data Tools; however, you can process by partition.

Processing a Partition

Effective administration of Analysis Services involves the practice of partitioning data. Partition processing is unique because it involves consideration of hard disk use and space constraints, combined with data structure limitations imposed by Analysis Services. To keep query response times fast and processing throughput high, you have to regularly create, process, and merge partitions. It is very important to recognize and manage against the chance of integrating redundant data during partition merging. For more information, see Merge Partitions in Analysis Services (SSAS - Multidimensional).

When you process a partition, Analysis Services processes the partition and any unprocessed dimensions that exist in the partition, depending on the processing option that you select. Using partitions offers several advantages for processing. You can process a partition without affecting other partitions in a cube. Partitions are useful for storing data that is subject to cell writeback. Writeback is a feature that enables the user to perform what-if analysis by writing new data back into the partition to see the effect of projected changes. A writeback partition is required if you use the cell writeback feature of Analysis Services. Processing partitions in parallel is useful because Analysis Services uses the processing power more efficiently and can significantly reduce total processing time. You can also process partitions sequentially.

Processing Data Mining Structures and Models

A mining structure defines the data domain from which data-mining models will be built. One mining structure can contain more than one mining model. You can process a mining structure separately from its associated mining models. When you process a mining structure separately, it is populated with the training data from your data source.

When a data mining model is processed, the training data passes through the mining model algorithms, trains the model using the data mining algorithm, and builds the content. For more information about the data mining model object, see Mining Structures (Analysis Services - Data Mining).

For more information about processing mining structures and models, see Processing Requirements and Considerations (Data Mining).

See Also

Tools and Approaches for Processing (Analysis Services)
Batch Processing (Analysis Services)
Processing a multidimensional model (Analysis Services)

Tools and Approaches for Processing (Analysis Services)

10/26/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Processing is an operation in which Analysis Services queries a relational data source and populates Analysis Services objects using that data.

As an Analysis Services system administrator, you can execute and monitor the processing of Analysis Services objects using these approaches:

- Run Impact Analysis to understand object dependencies and scope of operations
- Process individual objects in SQL Server Management Studio
- Process individual or multiple objects in SQL Server Data Tools (SSDT)
- Run Impact Analysis to review a list of related objects that will be unprocessed as result of the current action.
- Generate and run a script in an Analysis Services XMLA Query window in Management Studio to process individual or multiple objects
- Use Analysis Services PowerShell cmdlets
- Use control flows and tasks in SSIS packages
- Monitor processing with SQL Server Profiler
- Program a custom solution using AMO. For more information, see Programming AMO OLAP Basic Objects.

Processing is a highly configurable operation, controlled by a set of processing options that determine whether full or incremental processing occurs at the object level. For more information about processing options and objects, see Processing Options and Settings (Analysis Services) and Processing Analysis Services Objects.

> **NOTE**
> This topic describes the tools and approaches for processing multidimensional models. For more information about processing tabular models, see Process Database, Table, or Partition (Analysis Services) and Process Data.

Processing objects in SQL Server Management Studio

1. Start Management Studio and connect to Analysis Services.
2. Right-click the Analysis Services object you want to process and then click **Process**. You can process data at any of these levels:
   - Databases
   - Cubes
   - Measure Groups or individual partitions in the measure group
   - Dimensions

- Mining Models
- Mining Structures

Analysis Services objects are hierarchical. If you choose database, processing can occur for all of the objects contained in the database. Whether processing actually occurs will vary depending on the process option you select and the state of the object. Specifically, if an object is unprocessed, processing its parent will result in that object getting processed. For more information about object dependencies, see Processing Analysis Services Objects.

3. In the **Process** dialog box, in **Process Options**, use the default value provided or select a different option from the list. For more information about each option, see Processing Options and Settings (Analysis Services).
4. Click **Impact Analysis** to identify and optionally process dependent objects that are affected if the objects listed in the Process dialog box are processed.
5. Optionally, click **Change Settings** to modify the processing order, processing behavior relative to specific types of errors, and other settings.
6. Click **OK**.

   The Process Progress dialog box provides ongoing status for each command. If a status message is truncated, you can click **View Details** to read the entire message.

Processing Objects in SQL Server Data Tools

1. Start SQL Server Data Tools (SSDT) and open a project that has been deployed.
2. In Solution Explorer, under the deployed project, expand the **Dimensions** folder.
3. Right-click a dimension, and then click **Process**. You can right-click multiple dimensions to process multiple objects at once. For more information, see Batch Processing (Analysis Services).
4. In the **Process Dimension** dialog box, in the **Process Options** column under **Object list**, verify that the option for this column is **Process Full**. If it is not, under **Process Options**, click the option, and select **Process Full** from the drop-down list.
5. Click **Run**.
6. When processing is finished, click **Close**.

Run Impact Analysis to identify object dependencies and scope of operations

1. Before you process an Analysis Services object in either SQL Server Data Tools or Management Studio, you can analyze the effect on related objects by clicking **Impact Analysis** in one of the **Process Objects** dialog boxes.
2. Right-click a dimension, cube, measure group, or partition to open a **Process Objects** dialog box.
3. Click **Impact Analysis**. Analysis Services scans the model and reports on reprocessing requirements for objects that are related to the one you selected for processing.

Processing objects using XMLA

1. Start Management Studio and connect to Analysis Services.
2. Right-click the object to be processed and then click **Process**.
3. In the **Process** dialog box, select the process option you want to use. Modify any other settings. Run Impact

Analysis to identify any changes you might need to make.

4. Click **Script** on the **Process Objects** screen.

   This generates an XMLA script and opens an Analysis Services XMLA Query window.

5. Close the dialog box. The script contains the processing command and options that were specified in the dialog box.

6. Optionally, you can continue adding to the script if you want to process additional objects in the same batch. To continue, repeat the previous steps, appending the generated script so that you have a single script for all processing operations. To view an example, see Schedule SSAS Administrative Tasks with SQL Server Agent.

7. From the menu bar, click **Query**, and then click **Execute**.

Processing objects using PowerShell

1. Starting in this release of SQL Server, you can use Analysis Services PowerShell cmdlets to process objects. The following cmdlets can be run interactively or in script:

   - Invoke-ProcessCube cmdlet
   - Invoke-ProcessDimension cmdlet
   - Invoke-ProcessPartition cmdlet
   - Invoke-ASCmd cmdlet, which can be used to execute XMLA, MDX, or DMX script that includes processing commands.

Monitoring object processing using SQL Server Profiler

1. Connect to an Analysis Services instance in SQL Server Profiler.

2. In Events Selection, click **Show all events** to add all events to the list.

3. Choose the following events:

   - **Command Begin** and **Command End** to show when processing starts and stops
   - **Error** to capture any errors
   - **Progress Report Begin**, **Progress Report Current**, and **Progress Report End** to report on process status and show the SQL queries used to retrieve the data
   - **Execute MDX Script Begin** and **Execute MDX Script End** to show the cube calculations
   - Optionally, add lock events if you are diagnosing performance problems related to processing

Process Analysis Services objects using Integration Services

1. In Integration Services, create a package that uses the Analysis Services Processing Task to automatically populate objects with new data when you make regular updates to your source relational database.

2. In the **SSIS Toolbox**, double-click **Analysis Services Processing** to add it to the package.

3. Edit the task to specify a connection to the database, which objects to process, and the process option. For more information about how to implement this task, see Analysis Services Processing Task.

See Also

Processing a multidimensional model (Analysis Services)

Batch Processing (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Analysis Services, you can use the Batch command to send multiple processing commands to the server in a single request. Batch processing gives you a way to control which objects are to be processed, and in what order. Also, a batch can run as a series of stand-alone jobs, or as a transaction in which the failure of one process causes a rollback of the complete batch.

Batch processing maximizes data availability by consolidating and reducing the amount of time taken to commit changes. When you fully process a dimension, any partition using that dimension is marked as unprocessed. As a result, cubes that contain the unprocessed partitions are unavailable for browsing. You can address this with a batch processing job by processing the dimensions together with the affected partitions. Running the batch processing job as a transaction makes sure that all objects included in the transaction remain available for queries until all processing is completed. As the transaction commits the changes, locks are put on the affected objects, making the objects temporarily unavailable, but overall the amount of time used to commit the changes is less than if you processed objects individually.

The procedures in this topic show the steps for fully processing dimensions and partitions. Batch processing can also include other processing options, such as incremental processing. For these procedures to work correctly, you should use an existing Analysis Services database that contains at least two dimensions and one partition.

This topic includes the following sections:

Batch Processing in SQL Server Data Tools

Batch Processing using XMLA in Management Studio

Batch Processing in SQL Server Data Tools

Before objects can be processed in SQL Server Data Tools, the project that contains the objects must be deployed. For more information, see Deploy Analysis Services Projects (SSDT).

1. Open SQL Server Data Tools.
2. Open a project that has been deployed.
3. In Solution Explorer, under the deployed project, expand the **Dimensions** folder.
4. Holding the Ctrl key, click each dimension listed in the **Dimensions** folder.
5. Right-click the selected dimensions, and then click **Process**.
6. Holding the Ctrl key, click each dimension listed in the **Object list**.
7. Right-click the selected dimensions and select **Process Full**.
8. To customize the batch process job, click **Change Settings.**
9. Under **Processing options**, mark the following settings:
   - **Processing Order** is set to **Sequential**, and **Transaction mode** is set to **One Transaction**.
   - **Writeback Table Option** is set to **Use existing**.
   - Under **Affected Objects**, select the **Process affected objects** check box.

10. Click the **Dimension key errors** tab. Verify that **Use default error configuration** is selected.
11. Click **OK** to close the **Change Settings** screen.
12. Click **Run** in the **Process Objects** screen to start the processing job.
13. When the **Status** box shows **Process succeeded**, click **Close**.
14. Click **Close** on the **Process Objects** screen.

Batch Processing using XMLA in Management Studio

You can create an XMLA script that performs batch processing. Start by generating an XMLA script in Management Studio for each object, and then combine them into a single XMLA Query that you run interactively or inside a scheduled task.

For step by step instructions, see **Example 2** in Schedule SSAS Administrative Tasks with SQL Server Agent

See Also

Processing a multidimensional model (Analysis Services)

Remote Processing (Analysis Services)

8/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can run scheduled or unattended processing on a remote Analysis Services instance, where the processing request originates from one computer but executes on another computer on the same network.

Prerequisites

- If you are running different versions of SQL Server on each computer, the client libraries must match the version of the Analysis Services instance that is processing the model.
- On the remote server, **Allow remote connections to this computer** must be enabled, and the account issuing the processing request must be listed as an allowed user.
- Windows firewall rules must be configured to allow inbound connections to Analysis Services. Verify you can connect to the remote Analysis Services instance using SQL Server Management Studio. See Configure the Windows Firewall to Allow Analysis Services Access.
- Resolve any existing local processing errors before attempting remote processing. Verify that when the processing request is local, data can be successfully retrieved from the external relational data source. See Set Impersonation Options (SSAS - Multidimensional) for instructions on specifying credentials used to retrieve data.

On-demand remote processing

Analysis Services accepts processing requests from user or application accounts that have Analysis Services administrator permissions. If you are an administrator, verify that you can connect to the remote instance and process the database manually over the remote connection.

1. On the computer that will be used to schedule processing, start SQL Server Management Studio and connect to the remote Analysis Services instance.
2. Right-click the database, select **Process**, point to **Script** and choose **Script Action to New Query Window**. Commands used to invoke processing will appear in the query window.
3. Click **OK** to begin processing.

   Successful completion of this task provides an XMLA query that you can embed in a scheduled job. It also confirms there are no connection problems.

Schedule remote processing using SQL Server Agent Service

By default, SQL Server Agent service runs under a virtual account, with network connections made using the machine account. To avoid having to give a machine account administrative rights on the remote Analysis Services instance, you should change the SQL Server Agent service account to run as a least-privilege domain user account.

Be sure to grant all of the necessary permissions, including giving the account **sysadmin** rights on the Database Engine instance providing the service.

Use the following links to set permissions:

- Configure SQL Server Agent

- SQL Server Agent Components suggests alternative fixed server roles if granting **sysadmin** permissions is not possible.

After account permissions are configured, continue with these steps.

Grant the SQL Server Agent account administrator permission on SSAS

1. Using Management Studio, connect to the remote Analysis Services instance.
2. Right-click the server name, click P**roperties**, and then click **Security**.
3. Click **Add** to add the SQL Server Agent account.

Create the Job

1. In Management Studio, connect to the local Database Engine instance. SQL Server Agent is the last item in Object Explorer. If necessary, start the service.
2. Right-click **Job**s, click **New Job** and then enter a name.
3. In Steps, click **New** and then enter a name.
4. In Type, choose **SQL Server Analysis Services Command**.
5. In Server, enter the name of the remote Analysis Services instance.
6. In Command, paste the XMLA command for processing the database. This is the XMLA script that you generated in the verification step for on-demand remote processing. Click **OK** to save the job.

Start SQL Server Profiler

1. On the remote computer, start SQL Server Profiler. Connect to the Analysis Services instance, and click **Run** to start the trace using the default events.

   You will use SQL Server Profiler to monitor the processing events as they occur.

2. Optionally, you can set trace properties to send the trace to a file or table in a database.

Run the job

1. On the computer used to run the job, verify that the job can perform the basic operation. In Object Explorer, under SQL Server Agent, expand **Jobs**, right-click the job you just created, and click **Start Job at Step**. The job starts immediately. You can monitor progress in SQL Server Profiler.
2. As a final step, modify the job to run on a schedule that you define, adding any alerts or notifications necessary to administer the job. You might also want to refine the processing script, or create multiple steps in the job to process objects independently.

See Also

SQL Server Agent Components
Schedule SSAS Administrative Tasks with SQL Server Agent
Batch Processing (Analysis Services)
Processing a multidimensional model (Analysis Services)
Processing Objects (XMLA)

Error Configuration for Cube, Partition, and Dimension Processing

5/16/2018 • 12 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Error configuration properties on cube, partition, or dimension objects determine how the server responds when data integrity errors occur during processing. Duplicate keys, missing keys, and null values in a key column typically trigger such errors, and while the record causing the error will not be added to the database, you can set properties that determine what happens next. By default, processing stops. However, during cube development, you might want processing to continue when errors occur so that you can test cube behaviors with imported data, even if it is incomplete.

This topic includes the following sections:

- Execution order
- Default behaviors
- Error Configuration Properties
- Where to set Error Configuration properties
- Missing keys (KeyNotFound)
- Null foreign keys in a fact table (KeyNotFound)
- Null keys in a dimension
- Duplicate keys resulting inconsistent relationships (KeyDuplicate)
- Change the error limit or error limit action
- Set the error log path
- Next step

Execution order

The server always executes **NullProcessing** rules before **ErrorConfiguration** rules for each record. This is important to understand because null processing properties that convert nulls to zeroes can subsequently introduce duplicate key errors when two or more error records have zero in a key column.

Default behaviors

By default, processing stops at the first error implicating a key column. This behavior is controlled by an error limit that specifies zero as the number of allowed errors and the Stop Processing directive that tells the server to stop processing when the error limit is reached.

Records triggering an error, due to null or missing or duplicate values, are either converted to the unknown member or discarded. Analysis Services will not import data that violates data integrity constraints.

- Conversion to unknown member occurs by default, due to the **ConvertToUnknown** setting for **KeyErrorAction**. Records allocated to unknown member are quarantined in the database as evidence of a problem that you might want to investigate after processing is finished.

Unknown members are excluded from query workloads, but they will be visible in some client applications if the **UnknownMember** is set to **Visible**.

If you want to track how many nulls were converted to the unknown member, you can modify the **NullKeyConvertedToUnknown** property to report these errors to the log or in the Processing window.

- Discard occurs when you manually set the **KeyErrorAction** property to **DiscardRecord**.

Through error configuration properties, you can determine how the server responds when an error occurs. Options include stop processing immediately, continue processing but stop logging, or continue both processing and logging of errors. Defaults vary based on the severity of the error.

The error count keeps track of how many errors occur. When you set an upper limit, server response changes when that limit is reached. By default, the server stops processing after the limit is reached. The default limit is 0, causing processing to stop on the first error that is counted.

High impact errors, such as a missing key or null value in a key field, should be addressed quickly. By default, these errors adhere to **ReportAndContinue** server behaviors, where the server catches the error, adds it to the error count, and then continues processing (except the error limit is zero, so processing stops immediately).

Other errors are generated but not counted or logged by default (this is the **IgnoreError** setting) because the error does not necessarily pose a data integrity problem.

Error counts are affected by null processing settings. For dimension attributes, null processing options determine how the server responds when null values are encountered. By default, nulls in a numeric column are converted to zeroes, while nulls in a string column are processed as blank strings. You can override **NullProcessing** properties to catch null values before they turn into **KeyNotFound** or **KeyDuplicate** errors. See Null keys in a dimension for details.

Errors are logged in the Process dialog box but not saved. You can specify a key error log file name to collect errors in a text file.

Error Configuration Properties

There are nine error configuration properties. Five are used to determine server response when a specific error occurs. The other four are scoped to error configuration workloads, such as how many errors to allow, what to do when that limit is reached, whether to collect errors in a log file.

Server response to specific errors

PROPERTY	DEFAULT	OTHER VALUES
CalculationError Occurs when initializing error configuration.	**IgnoreError** neither logs nor counts the error; processing continues as long as the error count is under the maximum limit.	**ReportAndContinue** logs and counts the error. **ReportAndStop** reports the error and stops processing immediately, regardless of the error limit.

PROPERTY	DEFAULT	OTHER VALUES
KeyNotFound Occurs when a foreign key in a fact table does not have a matching primary key in a related dimension table (for example, a Sales fact table has a record with a product ID that doesn't exist in the Product dimension table). This error can occur during partition processing, or dimension processing of snowflaked dimensions.	**ReportAndContinue** logs and counts the error.	**ReportAndStop** reports the error and stops processing immediately, regardless of the error limit. **IgnoreError** neither logs nor counts the error; processing continues as long as the error count is under the maximum limit. Records that trigger this error are converted to the unknown member by default, but you can change the **KeyErrorAction** property to discard them instead.
KeyDuplicate Occurs when duplicate attribute keys are found in a dimension. In most cases, it is acceptable to have duplicate attribute keys, but this error informs you of the duplicates so that you can check the dimension for design flaws that might lead to inconsistent relationships between attributes.	**IgnoreError** neither logs nor counts the error; processing continues as long as the error count is under the maximum limit.	**ReportAndContinue** logs and counts the error. **ReportAndStop** reports the error and stops processing immediately, regardless of the error limit.
NullKeyNotAllowed Occurs when **NullProcessing** = **Error** is set on a dimension attribute or when null values exists in an attribute key column used to uniquely identify a member.	**ReportAndContinue** logs and counts the error.	**ReportAndStop** reports the error and stops processing immediately, regardless of the error limit. **IgnoreError** neither logs nor counts the error; processing continues as long as the error count is under the maximum limit. Records that trigger this error are converted to the unknown member by default, but you can set the **KeyErrorAction** property to discard them instead.
NullKeyConvertedToUnknown Occurs when null values are subsequently converted to the unknown member. Setting **NullProcessing** = **ConvertToUnknown** on a dimension attribute will trigger this error.	**IgnoreError** neither logs nor counts the error; processing continues as long as the error count is under the maximum limit.	If you consider this error to be informational, keep the default. Otherwise, you can choose **ReportAndContinue** to report the error to the Processing window and count the error towards the error limit. **ReportAndStop** reports the error and stops processing immediately, regardless of the error limit.

General Properties

PROPERTY	VALUES
KeyErrorAction	This is the action taken by the server when a **KeyNotFound** error occurs. Valid responses to this error include **ConvertToUnknown** or **DiscardRecord**.

PROPERTY	VALUES
KeyErrorLogFile	This is a user-defined filename that must have a .log file extension, located in a folder on which the service account has read-write permissions. This log file will only contain errors generated during processing. Use the Flight Recorder if you require more detailed information.
KeyErrorLimit	This is the maximum number of data integrity errors that the server will allow before failing the processing. A value of -1 indicates that there is no limit. The default is 0, which means processing stops after the first error occurs. You can also set it to a whole number.
KeyErrorLimitAction	This is the action taken by the server when the number of key errors has reached the upper limit. With **Stop Processing**, processing terminates immediately. With **Stop Logging**, processing continues but errors are no longer reported or counted.

Where to set Error Configuration properties

Use the property pages in either SQL Server Management Studio after the database is deployed, or in the model project in SQL Server Data Tools. The same properties are found in both tools. You can also set error configuration properties in the msmdrsrv.ini file to change server defaults for error configuration, and in **Batch** and **Process** commands if processing runs as a scripted operation.

You can set error configuration on any object that can be processed as a standalone operation.

SQL Server Management Studio

1. In Object Explorer, rick-click **Properties** one of these objects: dimension, cube, or partition.
2. In Properties, click **Error Configuration**.

SQL Server Data Tools

1. In Solution Explorer, double-click a dimension or cube. **ErrorConfiguration** appears in Properties in the pane below.
2. Alternatively, for a single dimension only, right-click the dimension in Solution Explorer, choose **Process**, and then choose **Change Settings** in the Process Dimension dialog box. Error configuration options appear on the Dimension Key Errors tab.

Missing keys (KeyNotFound)

Records with a missing key value cannot be added to the database, not even when errors are ignored or the error limit is unlimited.

The server produces the **KeyNotFound** error during partition processing, when a table in fact record contains a foreign key value, but the foreign key has no corresponding record in a related dimension table. This error also occurs when processing related or snowflaked dimension tables, where a record in one dimension specifies a foreign key that doesn't exist in the related dimension.

When a **KeyNotFound** error occurs, the offending record is allocated to the unknown member. This behavior is controlled through the **Key Action**, set to **ConvertToUnknown**, so that you can view the allocated records for further investigation.

Null foreign keys in a fact table (KeyNotFound)

By default, a null value in a foreign key column of a fact table is converted to zero. Assuming zero is not a valid foreign key value, the **KeyNotFound** error will be logged and counted towards the error limit that is zero by default.

To allow processing to continue, you can handle the null before it is converted and checked for errors. To do this, set **NullProcessing** to **Error**.

Set NullProcessing property on a measure

1. In SQL Server Data Tools, in Solution Explorer, double-click the cube to open it in Cube Designer.
2. Right-click a measure in the Measures pane and choose **Properties**.
3. In Properties, expand **Source** to view **NullProcessing** property. It is set to **Automatic** by default, which for OLAP items, converts nulls to zeroes for fields containing numeric data.
4. Change the value to **Error** to exclude any records having a null value, avoiding the null-to-numeric (zero) conversion. This modification lets you avoid duplicate key errors related to multiple records having zero in the key column, and also avoid **KeyNotFound** errors when a zero-valued foreign key has no primary key equivalent in a related dimension table.

Null keys in a dimension

To continue processing when null values are found in foreign keys in a snowflaked dimension, handle the null values first by setting **NullProcessing** on the **KeyColumn** of the dimension attribute. This discards or converts the record, before the **KeyNotFound** error can occur.

You have two options for handling nulls on dimension attribute:

- Set **NullProcessing**=**UnknownMember** to allocate records with null values to the unknown member. This produces the **NullKeyConvertedToUnknown** error, which is ignored by default.
- Set **NullProcessing**=**Error** to exclude records with null values. This produces the **NullKeyNotAllowed** error, which is logged and counts toward the key error limit. You can set error configuration property on **Null Key Not Allowed** to **IgnoreError** to allow processing to continue.

Nulls can be problem for non-key fields, in that MDX queries return different results depending on whether null is interpreted as zero or empty. For this reason, Analysis Services provides null processing options that let you predefine the conversion behavior you want. See Defining the Unknown Member and Null Processing Properties and NullProcessing for details.

Set NullProcessing property on a dimension attribute

1. In SQL Server Data Tools, in Solution Explorer, double-click the dimension to open it in Dimension Designer.
2. Right-click an attribute in the Attributes pane and choose **Properties**.
3. In Properties, expand **KeyColumns** to view **NullProcessing** property. It is set to **Automatic** by default, which converts nulls to zeroes for fields containing numeric data. Change the value to either **Error** or **UnknownMember**.

   This modification removes the underlying conditions that trigger **KeyNotFound** by either discarding or converting the record before it is checked for errors.

   Depending on error configuration, either of these actions can result in an error that is reported and counted. You might need to adjust additional properties, such as setting **KeyNotFound** to **ReportAndContinue** or **KeyErrorLimit** to a non-zero value, to allow processing to continue when these errors are reported and counted.

Duplicate keys resulting inconsistent relationships (KeyDuplicate)

By default, the presence of a duplicate key does not stop processing, but the error is ignored and the duplicate record is excluded from the database.

To change this behavior, set **KeyDuplicate** to **ReportAndContinue** or **ReportAndStop** to report the error. You can then examine the error to determine potential flaws in dimension design.

Change the error limit or error limit action

You can raise the error limit to allow more errors through during processing. There is no guidance for raising the error limit; the appropriate value will vary depending on your scenario. Error limits are specified as **KeyErrorLimit** in **ErrorConfiguration** properties in SQL Server Data Tools, or as **Number of Errors** in the Error Configuration tab for properties of dimensions, cubes, or measure groups in SQL Server Management Studio.

Once the error limit is reached, you can specify that processing stops or that logging stops. For example, suppose you set the action to **StopLogging** on an error limit of 100. On the 101st error, processing continues, but errors are no longer logged or counted. Error limit actions are specified as **KeyErrorLimitAction** in **ErrorConfiguration** properties in SQL Server Data Tools, or as **On error action** in the Error Configuration tab for properties of dimensions, cubes, or measure groups in SQL Server Management Studio.

Set the error log path

You can specify a file to store key-related error messages that are reported during processing. By default, errors are visible during interactive processing in the Process window and then discarded when you close the window or session. The log will only contain error information related to keys, identical to the errors you see reported in the processing dialog boxes.

Errors will be logged to a text file and must have .log file extension. The file will be empty unless errors occur. By default, a file will be created in the DATA folder. You can specify another folder as long as the Analysis Services service account can write to that location.

Next step

Decide whether errors will stop processing or be ignored. Remember that only the error is ignored. The record that caused the error is not ignored; it is either discarded or converted to unknown member. Records that violate data integrity rules are never added to the database. By default, processing stops when the first error occurs, but you can change this by raising the error limit. In cube development, it can be useful to relax error configuration rules, allowing processing to continue, so that there is data to test with.

Decide whether to change default null processing behaviors. By default, nulls in a string column are processed as empty values, while nulls in a numeric column are processed as zero. See Defining the Unknown Member and Null Processing Properties for instructions on setting null processing on an attribute.

See Also

Log Properties
Defining the Unknown Member and Null Processing Properties

Roles and Permissions (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services provides a role-based authorization model that grants access to operations, objects, and data. All users who access an Analysis Services instance or database must do so within the context of a role.

As an Analysis Services system administrator, you are in charge of granting membership to the **server administrator role** that conveys unrestricted access to operations on the server. This role has fixed permissions and cannot be customized. By default, members of the local Administrators group are automatically Analysis Services system administrators.

Non-administrative users who query or process data are granted access through **database roles**. Both system administrators and database administrators can create the roles that describe different levels of access within a given database, and then assign membership to every user who requires access. Each role has a customized set of permissions for accessing objects and operations within a particular database. You can assign permissions at these levels: database, interior objects such as cubes and dimensions (but not perspectives), and rows.

It is common practice to create roles and assign membership as separate operations. Often, the model designer adds roles during the design phase. This way, all role definitions are reflected in the project files that define the model. Role membership is typically rolled out later as the database moves into production, usually by database administrators who create scripts that can be developed, tested, and run as an independent operation.

All authorization is predicated on a valid Windows user identity. Analysis Services uses Windows authentication exclusively to authenticate user identities. Analysis Services provides no proprietary authentication method.See Authentication methodologies supported by Analysis Services.

> **IMPORTANT**
>
> Permissions are additive for each Windows user or group, across all roles in the database. If one role denies a user or group permission to perform certain tasks or view certain data, but another role grants this permission to that user or group, the user or group will have permission to perform the task or view the data.

In this section

- Authorizing access to objects and operations (Analysis Services)
- Grant database permissions (Analysis Services)
- Grant cube or model permissions (Analysis Services)
- Grant process permissions (Analysis Services)
- Grant read definition permissions on object metadata (Analysis Services)
- Grant permissions on a data source object (Analysis Services)
- Grant permissions on data mining structures and models (Analysis Services)
- Grant permissions on a dimension (Analysis Services)
- Grant custom access to dimension data (Analysis Services)

- Grant custom access to cell data (Analysis Services)

See Also

Create and Manage Roles

Authorizing access to objects and operations (Analysis Services)

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Non-administrative user access to cubes, dimensions, and mining models within an Analysis Services database is granted through membership in one or more database roles. Analysis Services administrators create these database roles, granting Read or Read/Write permissions on Analysis Services objects, and then assigning Microsoft Windows users and groups to each role.

Analysis Services determines the effective permissions for a specific Windows user or group by combining the permissions that are associated with each database role to which the user or group belongs. As a result, if one database role does not give a user or group permission to view a dimension, measure, or attribute, but a different database role does give that user or group permission, the user or group will have permission to view the object.

> **IMPORTANT**
>
> Members of the Analysis Services Server Administrator role and members of a database role having Full Control (Administrator) permissions can access all data and metadata in the database and need no additional permissions to view specific objects. Moreover, members of the Analysis Services server role cannot be denied access to any object in any database, and members of an Analysis Services database role that has Full Control (Administrator) permissions within a database cannot be denied access to any object within that database. Specialized administrative operations, such as processing, can be authorized through separate roles having less permission. See Grant process permissions (Analysis Services) for details.

List roles defined for your database

Administrators can run a simple DMV query in SQL Server Management Studio to get a list of all roles defined on the server.

1. In SSMS, right-click a database and select **New Query** | **MDX**.
2. Type the following query and press F5 to execute:

   ```
   Select * from $SYSTEM.DBSCHEMA_CATALOGS
   ```

   Results include the database name, description, role name, and the date last modified. Using this information as a starting point, you can proceed to individual databases to check membership and permissions of a specific role.

Top-down overview of Analysis Services authorization

This section covers the basic workflow for configuring permissions.

Step 1: Server Administration

As a first step, decide who will have administrator rights at the server level. During installation, the local administrator who installs SQL Server is required to specify one or more Windows accounts as the Analysis Services server administrator. Server administrators have all possible permissions on a server, including the

permission to view, modify, and delete any object on the server, or view associated data. After installation is complete, a server administrator can add or remove accounts to change membership of this role. See Grant server admin rights to an Analysis Services instance for details about this permission level.

Step 2: Database Administration

Next, after a tabular or multidimensional solution is created, it is deployed to the server as a database. A server administrator can delegate database administration tasks by defining a role that has Full Control permissions for the database in question. Members of this role can process or query objects in the database, as well as create additional roles for accessing cubes, dimensions, and other objects within the database itself. See Grant database permissions (Analysis Services) for more information.

Step 3: Enable cube or model access for query and processing workloads

By default, only server and database administrators have access to cubes or tabular models. Making these data structures available to other people in your organization requires additional role assignments that map Windows user and group accounts to cubes or models, along with permissions that specify **Read** privileges. See Grant cube or model permissions (Analysis Services) for details.

Processing tasks can be isolated from other administrative functions, allowing server and database administrators to delegate this task to other people, or to configure unattended processing by specifying service accounts that run scheduling software. See Grant process permissions (Analysis Services) for details.

> **NOTE**
> Users do not require any permissions to the relational tables in the underlying relational database from which Analysis Services loads its data, and do not require any file level permissions on the computer on which the instance of Analysis Services is running.

Step 4 (Optional): Allow or deny access to interior cube objects

Analysis Services provides security settings for setting permissions on individual objects, including dimension members and cells within a data model. For details, see Grant custom access to dimension data (Analysis Services) and Grant custom access to cell data (Analysis Services).

You can also vary permissions based on user identity. This is often referred to as dynamic security, and is implemented using the UserName (MDX) function

Best practices

To better manage permissions, we suggest an approach similar to the following:

1. Create roles by function (for example, dbadmin, cubedeveloper, processadmin) so that whoever maintains the roles can see at glance what the role allows. As noted elsewhere, you can define roles in the model definition, thus preserving those roles over subsequent solution deployments.

2. Create a corresponding Windows security group in Active Directory, and then maintain the security group in Active Directory to insure it contains the appropriate individual accounts. This places the responsibility of security group membership on security specialists who are already have proficiency with the tools and processes used for account maintenance in your organization.

3. Generate scripts in SQL Server Management Studio so that you can quickly replicate role assignments whenever the model is redeployed from its source files to a server. See Grant cube or model permissions (Analysis Services) for details on how to quickly generate a script.

4. Adopt a naming convention that reflects the scope and membership of the role. Role names are only visible in design and administration tools, so use a naming convention that makes sense to your cube

security specialists. For example, **processadmin-windowsgroup1** indicates read access, plus processing rights, to people in your organization whose individual Windows user accounts are members of the **windowsgroup1** security group.

Including account information can help you keep track of which accounts are used in various roles. Because roles are additive, the combined roles associated with **windowsgroup1** make up the effective permission set for people belonging to that security group.

5. Cube developers will require Full Control permissions for models and databases under development, but only need Read permissions once a database is rolled out to a production server. Remember to develop role definitions and assignments for all scenarios, including development, test, and production deployments.

Using an approach like this one minimizes churn to role definitions and role membership in the model, and provides visibility into role assignments that makes cube permissions easier to implement and maintain.

See Also

Grant server admin rights to an Analysis Services instance
Roles and Permissions (Analysis Services)
Authentication methodologies supported by Analysis Services

Grant database permissions (Analysis Services)

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If you are approaching Analysis Services database administration with a background in relational databases, the first thing you need to understand is that, in terms of data access, the database is not the primary securable object in Analysis Services.

The primary query structure in Analysis Services is a cube (or a tabular model), with user permissions set on those particular objects. Contrasted with the relational database engine — where database logins and user permissions (often **db_datareader**) are set on the database itself — an Analysis Services database is mostly a container for the main query objects in a data model. If your immediate objective is to enable data access for a cube or tabular model, you can bypass database permissions for now and go straight to this topic: Grant cube or model permissions (Analysis Services).

Database permissions in Analysis Services enable administrative functions; broadly, as is the case with the Full Control database permission, or of a more granular nature if you are delegating processing operations. Permission levels for an Analysis Services database are specified on the **General** pane of the **Create Role** dialog box, shown in the following illustration and described below.

There are no logins in Analysis Services. You simply create roles and assign Windows accounts in the **Membership** pane. All users, including administrators, connect to Analysis Services using a Windows account.

There are three types of permissions specified at the database level.

Full Control (Administrator) — Full Control is an all-encompassing permission that conveys broad powers over an Analysis Services database, such as the ability to query or process any object within the database, and manage role security. Full Control is synonymous with database administrator status. When you select **Full Control**, the **Process Database** and **Read Definition** permissions are also selected and cannot be removed.

> **NOTE**
>
> Server administrators (members of the Server Administrator role) also have implicit Full Control over every database on the server.

Process Database — This permission is used to delegate processing at the database level. As an administrator, you can offload this task by creating a role that allows another person or service to invoke processing operations for any object in the database. Alternatively, you can also create roles that enable processing on specific objects. See Grant process permissions (Analysis Services) for more information.

Read Definition — This permission grants the ability to read object metadata, minus the ability to view associated data. Typically this permission is used in roles created for dedicated processing, adding the ability to use tools such as SQL Server Data Tools or SQL Server Management Studio to process a database interactively. Without **Read Definition**, the **Process Database** permission is effective only in scripted scenarios. If you plan to automate processing, perhaps through SSIS or another scheduler, you probably want to create a role that has **Process Database** without **Read Definition**. Otherwise, consider combining the two properties together in the same role to support both unattended and interactive processing via SQL Server tools that visualize the data model in a user interface.

Full Control (Administrator) permissions

In Analysis Services, a database administrator is any Windows user identity assigned to a role that includes Full Control (Administrator) permissions. A database administrator can perform any task within the database, including:

- Process objects
- Read data and metadata for all objects in the database, including cubes, dimensions, measure groups, perspectives and data mining models
- Create or modify database roles by adding users or permissions, including adding users to roles also having Full Control permissions
- Delete database roles or role membership
- Register assemblies (or stored procedures) for the database.

Notice that a database administrator cannot add or delete databases on the server, or grant administrator rights to other databases on the same server. That privilege belongs to server administrators alone. See Grant server admin rights to an Analysis Services instance for more information about this permission level.

Because all roles are user-defined, we recommend that you create a role dedicated for this purpose (for example, a role named "dbadmin"), and then assign Windows user and group accounts accordingly.

Create roles in SSMS

1. In SQL Server Management Studio, connect to the instance of Analysis Services, open the **Databases** folder, select a database, and right-click **Roles** | **New Role**.
2. In the **General** pane, enter a name, such as DBAdmin.
3. Select the **Full Control (Administrator)** check box for the cube. Notice that **Process Database** and **Read Definition** are selected automatically. Both of these permissions are always included in roles that include **Full Control**.
4. In the **Membership** pane, enter the Windows user and group accounts that connect to Analysis Services using this role.
5. Click **OK** to finish creating the role.

Process database

When defining a role that grants database permissions, you can skip **Full Control** and choose just **Process Database**. This permission, set at the database level, allows processing on all objects within the database. See

Grant process permissions (Analysis Services)

Read definition

Like **Process Database**, setting **Read Definition** permissions at the database level has a cascading effect on other objects within the database. If you want to set Read Definition permissions at a more granular level, you must clear Read Definition as a database property in the General pane. See Grant read definition permissions on object metadata (Analysis Services) for more information.

See Also

Grant server admin rights to an Analysis Services instance
Grant process permissions (Analysis Services)

Grant cube or model permissions (Analysis Services)

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A cube or tabular model is the primary query object in an Analysis Services data model. When connecting to multidimensional or tabular data from Excel for ad hoc data exploration, users typically start by selecting a specific cube or tabular model as the data structure behind the Pivot report object. This topic explains how to grant the necessary permissions for cube or tabular data access.

By default, no one except a Server Administrator or Database Administrator has permission to query cubes in a database. Cube access by a non-administrator requires membership in a role created for the database containing the cube. Membership is supported for Windows user or group accounts, defined in either Active Directory or on the local computer. Before you start, identify which accounts will be assigned membership in the roles you are about to create.

Having **Read** access to a cube also conveys permissions on the dimensions, measure groups, and perspectives within it. Most administrators will grant read permissions at the cube level and then restrict permissions on specific objects, on associated data, or by user identity.

To preserve role definitions over successive solution deployments, a best practice is to define roles in SQL Server Data Tools as an integral part of the model, and then have a database administrator assign role memberships in SQL Server Management Studio after the database is published. But you can use either tool for both tasks. To simplify the exercise, we'll use SQL Server Management Studio for both role definition and membership.

> **NOTE**
> Only server administrators, or database administrators having Full Control permissions, can deploy a cube from source files to a server, or create roles and assign members. See Grant server admin rights to an Analysis Services instance and Grant database permissions (Analysis Services) for details about these permission levels.

Step 1: Create the role

1. In SSMS, connect to Analysis Services. See Connect from client applications (Analysis Services) if you need help with this step.
2. Open the **Databases** folder in Object Explorer, and select a database.
3. Right-click **Roles** and choose **New Role**. Notice that roles are created at the database level and apply to objects within it. You cannot share roles across databases.
4. In the **General** pane, enter a name, and optionally, a description. This pane also contains several database permissions, such as Full Control, Process Database, and Read Definition. None of these permissions are needed for querying a cube or tabular model. See Grant database permissions (Analysis Services) for more information about these permissions.
5. Continue to the next step after entering a name and optional description.

Step 2: Assign Membership

1. In the **Membership** pane, click **Add** to enter the Windows user or group accounts that will be accessing the cube using this role. Analysis Services only supports Windows security identities. Notice that you are not creating database logins in this step. In Analysis Services, users connect through Windows accounts.
2. Continue to the next step, setting cube permissions.

Notice that we are skipping the Data Source pane. Most regular consumers of Analysis Services data do not need permissions on the data source object. See Grant permissions on a data source object (Analysis Services) for details on when you might set this permission.

Step 3: Set Cube Permissions

1. In the **Cubes** pane, select a cube, and then click **Read** or **Read/Write** access.

   Read access is sufficient for most operations. **Read/Write** is used only for writeback, not processing. See Set Partition Writeback for more information about this capability.

   Notice that you can select multiple cubes, as well as other objects available in the Create Role dialog box. Granting permissions to a cube authorizes access to the dimensions and perspectives associated with the cube. It's not necessary to manually add objects already represented in the cube.

   If you need to vary authorization by objects or user, for example to make certain measures unavailable, you can allow or deny access atomically on specific objects, even on cells. See Grant custom access to dimension data (Analysis Services) and Grant custom access to cell data (Analysis Services) for details.

2. At this point, after you click **OK**, all members of this role have access to the cubes, at the permission levels you specified.

   Notice that on the **Cubes** pane, you can grant users permission to create local cubes from a server cube via **Drillthrough and Local Cube**, or allow drillthrough only, via the **Drillthrough** permission.

   Finally, this pane lets you grant **Process Database** rights on the cube to give all members of this role the ability to process data for this cube. Because processing is typically a restricted operation, we recommend that you leave that task to the administrators, or define separate roles specifically for that task. See Grant process permissions (Analysis Services) for more information about processing permission best practices.

Step 4: Test

1. Use Excel to test cube access permissions. You can also use SQL Server Management Studio, following the same technique described next — running the application as a non-administrator user.

   > **NOTE**
   > If you are an Analysis Services administrator, administrator permissions will be combined with roles having lesser permissions, making it difficult to test role permissions in isolation. To simplify testing, we suggest that you open a second instance of SSMS, using the account assigned to the role you are testing.

2. Hold-down the Shift key and right-click the **Excel** shortcut to access the **Run as different user** option. Enter one of the Windows user or group accounts having membership in this role.

3. When Excel opens, use the Data tab to connect to Analysis Services. Because you are running Excel as a different Windows user, the **Use Windows Authentication** option is the right credential type to use when testing roles. See Connect from client applications (Analysis Services) if you need help with this step.

   If you get errors on the connection, check the port configuration for Analysis Services and verify that the server accepts remote connections. See Configure the Windows Firewall to Allow Analysis Services Access for port configuration.

Step 5: Script role definition and assignments

1. As a final step, you should generate a script that captures the role definition you just created.

   Redeploying a project from SQL Server Data Tools will overwrite any roles or role memberships that are not defined inside the project. The quickest way to rebuild roles and role membership after redeployment is through script.

2. In SSMS, navigate to the **Roles** folder and right-click an existing role.
3. Select **Script Role as** | **CREATE TO** | **file**.
4. Save the file with an .xmla file extension. To test the script, delete the current role, open the file in SSMS, and press F5 to execute the script.

Next step

You can refine cube permissions to restrict access to cell or dimension data. See Grant custom access to dimension data (Analysis Services) and Grant custom access to cell data (Analysis Services) for details.

See Also

Authentication methodologies supported by Analysis Services
Grant permissions on data mining structures and models (Analysis Services)
Grant permissions on a data source object (Analysis Services)

Grant process permissions (Analysis Services)

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

As an administrator, you can create a role dedicated to Analysis Services processing operations, allowing you to delegate that particular task to other users, or to applications used for unattended scheduled processing. Process permissions can be granted at the database, cube, dimension, and mining structure levels. Unless you are working with a very large cube or tabular database, we recommend granting processing rights at the database level, inclusive of all objects, including those having dependencies on each other.

Permissions are granted through roles that associate objects with permissions and Windows user or group accounts. Remember that permissions are additive. If one role grants permission to process a cube, while a second role gives the same user permission to process a dimension, the permissions from the two different roles combine to give the user permission to both process the cube and process the specified dimension within that database.

> **IMPORTANT**
>
> A user whose role only has Process permissions will be unable to use SQL Server Management Studio or SQL Server Data Tools (SSDT) to connect to Analysis Services and process objects. These tools require the **Read Definition** permission to access object metadata. Without the ability to use either tool, XMLA script must be used to execute a processing operation.
>
> We suggest you also grant **Read Definition** permissions for testing purposes. A user having both **Read Definition** and **Process Database** permissions can process objects in SQL Server Management Studio, interactively. See Grant read definition permissions on object metadata (Analysis Services) for details.

Set processing permissions at the database level

This section explains how to enable processing by non-administrators, for all cubes, dimensions, mining structures, and mining models in the database.

1. In SQL Server Management Studio, connect to the instance of Analysis Services, open the Databases folder, and select a database.
2. Right-click **Roles** | **New Role**. Enter a name and description.
3. In the **General** pane, select the **Process Database** check box. Additionally, select **Read Definition** to also enable interactive processing through one of the SQL Server tools, such as SQL Server Management Studio.
4. In the **Membership** pane, add the Windows user and group accounts having permission to process any object in this database.
5. Click **OK** to complete the role definition.

Set processing permissions on individual objects

You can set processing permissions on individual cubes, dimensions, data mining structures or models.

Processing can fail if you inadvertently exclude objects that need to be processed together (for example, if you enable processing on a cube, but not on its related dimensions). Because it can be easy to miss object

dependencies, thorough testing is essential when setting processing permissions on individual objects.

1. In SQL Server Management Studio, connect to the instance of Analysis Services, open the Databases folder, and select a database.
2. Right-click **Roles** | **New Role**. Enter a name and description.
3. In the **General** pane, clear the **Process Database** check box. Database permissions override the ability to set permissions on lower-level objects by making role options grayed out or un-selectable.

   Technically, no database permissions are needed for dedicated processing roles. But without **Read Definition** at the database level, you cannot view the database in SQL Server Management Studio, making testing more difficult.
4. Select individual objects to process:
   - In the **Cubes** pane, select the **Process** check box for each cube.
   - In the **Dimensions** pane, select **All database dimensions**, and then **Process** check box for each dimension. Or, select all rows, then use shift-click to toggle the check box selections.
5. In the **Membership** pane, add the Windows user and group accounts having permission to process these objects.
6. Click **OK** to complete the role definition.

Test processing

1. Hold down the shift-key and right-click SQL Server Management Studio, select **Run as a different user** and connect to the instance of Analysis Services using a Windows account assigned to the role you are testing.
2. Open the Databases folder, and select a database. You will only see the databases that are visible to the roles for which your account has membership.
3. Right-click a cube or dimension and select **Process**. Choose a processing option. Test all of the options, for all combinations of objects. If errors occur due to missing objects, add the objects to the role.

Set processing permissions on a data mining structure

You can create a role granting permission to process data mining structures. This includes the processing of all mining models.

Drill Through and **Read Definition** permissions used for browsing a mining model and structure are atomic and can be added to the same role, or separated out into a different role.

1. In SQL Server Management Studio, connect to the instance of Analysis Services, open the Databases folder, and select a database.
2. Right-click **Roles** | **New Role**. Enter a name and description. In the **General** pane, make sure that the database permission check boxes are clear. Database permissions will override the ability to set permissions on lower-level objects by making role options grayed out or un-selectable.
3. In the **Mining Structures** pane, select the **Process** check box for each mining structure.
4. In the **Membership** pane, add the Windows user and group accounts having permission to process any object in this database.
5. Click **OK** to complete the role definition.

See Also

Process Database, Table, or Partition (Analysis Services)
Processing a multidimensional model (Analysis Services)
Grant database permissions (Analysis Services)
Grant read definition permissions on object metadata (Analysis Services)

Grant read definition permissions on object metadata (Analysis Services)

10/26/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Permission to read an object definition, or metadata, on selected objects lets an administrator grant permission to view object information, without also granting permission to modify the object's definition, modify the object's structure, or view the actual data for the object. **Read Definition** permissions can be granted at the database, data source, dimension, mining structure, and mining model levels. If you require **Read Definition** permissions for a cube, you must enable **Read Definition** for the database.Remember that permissions are additive. For example, one role grants permission to read the metadata for a cube, while a second role grants the same user permission to read the metadata for a dimension. The permissions from the two different roles combine to give the user permission to both read metadata for the cube and the metadata for the dimension within that database.

> **NOTE**
>
> Permission to read a database's metadata is the minimum permission required to connect to an Analysis Services database using either Management Studio or SQL Server Data Tools. A user who has permission to read metadata can also use the DISCOVER_XML_METADATA schema rowset to query the object and view its metadata. For more information, see DISCOVER_XML_METADATA Rowset.

Set read definition permissions on a database

Granting permission to read database metadata also grants permission to read the metadata of all objects in the database.

We suggest that you include the **Read Definition** permission at the database level whenever you are setting up roles for dedicated processing. Having **Read Definition** allows non-administrators to view a model's object hierarchy in SQL Server Management Studio and navigate to individual objects for subsequent processing.

1. In SQL Server Management Studio, connect to the instance of Analysis Services, expand **Roles** for the appropriate database in Object Explorer, and then click a database role (or create a new database role).
2. On the **General** tab, select the **Read Definition** option.
3. In the **Membership** pane, enter the Windows user and group accounts that connect to Analysis Services using this role.
4. Click **OK** to finish creating the role.

Set read definition permissions on individual objects

1. In SQL Server Management Studio, connect to the instance of Analysis Services, open the **Databases** folder, select a database, expand **Roles** for the appropriate database in Object Explorer, and then click a database role (or create a new database role).
2. In the **General** pane, clear the database permission for **Read Definition**. This step removes permission inheritance so that you can set permissions on individual objects.
3. Select the object for which you are specifying **Read Definition** properties:

- In the **Data Sources** pane, click the **Read Definition** check box for that data source. Role members can view the connection string to the data source, including the server name and possibly the user name. This permission is available in case you want to provide connection string information, without also granting permission to modify the connection string or view the definitions of any other objects.
- In the **Dimensions** pane, click the **Read Definition** check box for that dimension. Experienced analysts and developers may need to view the definition without permission to modify it or view the definitions of other objects (such as other dimensions, cube objects, or mining structures and models).
- In the Mining Structures pane, click the **Read Definition** check box for data mining structures or models. **Read Definition** is required for browsing the data model. See Grant permissions on data mining structures and models (Analysis Services) for details.

4. In the **Membership** pane, enter the Windows user and group accounts that connect to Analysis Services using this role.
5. Click **OK** to finish creating the role.

See Also

Grant database permissions (Analysis Services)
Grant process permissions (Analysis Services)

Grant permissions on a dimension (Analysis Services)

10/26/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Dimension security is used to set permissions on a dimension object, not its data. Typically, allowing or denying access to processing operations is the main objective when setting permissions on a dimension.

But perhaps your objective is not to control processing operations, but rather data access to a dimension, or the attributes and hierarchies it contains. For example, a company with regional sales divisions might want to make sales performance information off limits to those outside the division. To allow or deny access to portions of dimension data for different constituents, you can set permissions on dimension attributes and dimension members. Notice that you cannot deny access to an individual dimension object itself, only to its data. If your immediate goal is to allow or deny access to members in a dimension, including access rights to individual attribute hierarchies, see Grant custom access to dimension data (Analysis Services) for more information.

The remainder of this topic covers permissions that you can set on the dimension object itself, including:

- Read or Read/Write permissions (you can only choose from Read or Read/Write; specifying "none" is not an option). As noted, if your goal is to restrict access to dimension data, see Grant custom access to dimension data (Analysis Services) for details.
- Processing permissions (do this when scenarios require a processing strategy that calls for custom permissions on individual objects)
- Read definition permissions (typically you would do this to support interactive processing in a tool, or to provide visibility into a model. Read definition lets you see the structure of a dimension, without permission to its data or the ability to modify its definition).

When defining roles for a dimension, available permissions vary depending on whether the object is a standalone database dimension —internal to the database but external to a cube— or a cube dimension.

> **NOTE**
>
> By default, permissions on a database dimension are inherited by a cube dimension. For example, if you enable **Read/Write** on a Customer database dimension, the Customer cube dimension inherits **Read/Write** in the context of the current role. You can clear inherited permissions if you want to override a permission setting.

Set permissions on a database dimension

Database dimensions are standalone objects within a database, allowing for dimension reuse within the same model. Consider a DATE database dimension that is used multiple times in a model, as Order Date, Ship Date, and Due Date cube dimensions. Because cubes and database dimensions are peer objects in a database, you can set processing permissions independently on each object.

1. In SQL Server Management Studio, connect to the instance of Analysis Services, expand **Roles** for the appropriate database in Object Explorer, and then click a database role (or create a new database role).
2. In the **Dimensions** pane, the dimension set should be set to **All database dimensions**.

   By default, permissions are set to **Read**.

   Although **Read/Write** is available, we recommend that you do not use this permission. **Read/Write** is used

for dimension writeback scenarios, which have been deprecated. See Deprecated Analysis Services Features in SQL Server 2016.

Optionally, you can set **Read Definition** and **Process** permissions on individual dimension objects, as long as those permissions are not already set at the database level. See Grant process permissions (Analysis Services) and Grant read definition permissions on object metadata (Analysis Services) for details.

Set permissions on a cube dimension

Cube dimensions are database dimensions that have been added to a cube. As such, they are structurally dependent on associated measure groups. Although you can process these objects atomically, in terms of authorization, it makes sense to treat the cube and cube dimensions as a single entity.

1. In SQL Server Management Studio, connect to the instance of Analysis Services, expand **Roles** for the appropriate database in Object Explorer, and then click a database role (or create a new database role).

2. In the **Dimensions** pane, change the dimension set to <cube-name> **cube dimensions**.

   By default, permissions are inherited from a corresponding database dimension. Clear the **Inherit** check box to alter permissions from **Read** to **Read/Write**. Before using **Read/Write**, be sure to read the note in the previous section.

> **IMPORTANT**
>
> If you configure database role permissions by using Analysis Management Objects (AMO), any reference to a cube dimension in a cube's DimensionPermission attribute severs the permission inheritance from the database's DimensionPermission attribute. For more information about AMO, see Developing with Analysis Management Objects (AMO).

See Also

Roles and Permissions (Analysis Services)
Grant cube or model permissions (Analysis Services)
Grant permissions on data mining structures and models (Analysis Services)
Grant custom access to dimension data (Analysis Services)
Grant custom access to cell data (Analysis Services)

Grant custom access to dimension data (Analysis Services)

5/16/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After enabling read access to a cube, you can set additional permissions that explicitly allow or deny access to dimension members (including measures contained in the Measures Dimension containing all of the measures used in a cube). For example, given multiple categories of resellers, you might want to set permissions to exclude data for a specific business type. The following illustration is a before-and-after effect of denying access to the Warehouse business type in the Reseller dimension.

Row Labels	Reseller Sales Amount	Reseller Total Product Cost	Reseller Gross Profit
Specialty Bike Shop	$6,756,166.18	$6,728,368.53	$27,797.65
Value Added Reseller	$34,967,517.33	$34,520,042.39	$447,474.93
Warehouse	$38,726,913.48	$38,731,703.45	($4,789.98)
Grand Total	**$80,450,596.98**	**$79,980,114.38**	**$470,482.60**

Row Labels	Reseller Sales Amount	Reseller Total Product Cost	Reseller Gross Profit
Specialty Bike Shop	$6,756,166.18	$6,728,368.53	$27,797.65
Value Added Reseller	$34,967,517.33	$34,520,042.39	$447,474.93
Grand Total	**$80,450,596.98**	**$79,980,114.38**	**$470,482.60**

By default, if you can read data from an Analysis Services cube, you automatically have read permissions on all measures and dimension members associated with that cube. While this behavior might be sufficient for many scenarios, sometimes security requirements call for a more segmented authorization strategy, with varying levels of access for different users, on the same dimension.

You can restrict access by choosing which members to allow (AllowedSet) or deny (DeniedSet) access. You do this by either selecting or deselecting dimension members to include or exclude from the role.

Basic dimension security is the easiest; you simply select which dimension attributes and attribute hierarchies to include or exclude in the role. Advanced security is more complex and requires expertise in MDX scripting. Both approaches are described below.

> **NOTE**
>
> The following instructions assume a client connection that issues queries in MDX. If the client uses DAX, such as Power View in Power BI, then dimension security is not evident in the query results. See Understanding Power View for Multidimensional Models for more information.

Prerequisites

Not all measures or dimension members can be used in custom access scenarios. A connection will fail if a role restricts access to a default measure or member, or restricts access to measures that are part of measure expressions.

Check for obstructions to dimension security: default measures, default members, and measures used in measure expressions

1. In SQL Server Management Studio, right-click a cube and select **Script Cube as** | **ALTER To** | **New Query Editor Window**.

2. Search for **DefaultMeasure**. You should find one for the cube, and one for each perspective. When defining dimension security, avoid restricting access to default measures.

3. Next, search for **MeasureExpression**. A measure expression is a measure, based on a calculation, where the calculation often includes other measures. Verify that the measure you want to restrict is not used in an expression. Alternatively, go ahead and restrict access, just make sure to also exclude all references to that measure throughout the cube.

4. Finally, search for **DefaultMember**. Make a note of any attributes that serve as a default member of an attribute. Avoid putting restrictions on those attributes when setting up dimension security.

Basic dimension security

1. In SQL Server Management Studio, connect to the instance of Analysis Services, expand **Roles** for the appropriate database in Object Explorer, and then click a database role (or create a new database role).

   The role should already have read access to the cube. See Grant cube or model permissions (Analysis Services) if you need help with this step.

2. On **Dimension Data** | **Basic**, select the dimension for which you are setting permissions.

3. Choose the attribute hierarchy. Not all attributes will be available. Only those attributes having **AttributeHierarchyEnabled** appear in the **Attribute Hierarchy** list.

4. Choose which members to allow or deny access. Allowing access, through the **Select all members** option, is the default. We suggest you keep this default and then clear individual members that should not be visible to the Windows user and group accounts in the **Memberships** pane via this role. The advantage is that new members added in future processing operations are automatically available to people connecting through this role.

   Alternatively, you can **Deselect all members** to revoke access overall, and then pick which members to allow. In future processing operations, new members are not visible until you manually edit dimension data security to allow access to them.

5. Optionally, click **Advanced** to enable **Visual Totals** for this attribute hierarchy. This option recalculates aggregations based on the members available through the role.

   > **NOTE**
   >
   > When applying permissions that trim dimension members, aggregated totals are not recalculated automatically. Suppose the **All** member of an attribute hierarchy returns a count of 200 before permissions are applied. After applying permissions that deny access to some members, **All** still returns 200, even though the member values visible to the user are much less. To avoid confusing the consumers of your cube, you can configure the **All** member be the aggregate of just those members to which role members, rather than the aggregate of all of the members of the attribute hierarchy. To invoke this behavior, you can enable **Visual Totals** on the **Advanced** tab when configuring dimension security. Once enabled, the aggregate is calculated at query time rather than retrieved from pre-calculated aggregations. This can have a noticeable effect on query performance, so use it only when necessary.

Hiding measures

In Grant custom access to cell data (Analysis Services), it was explained that fully hiding all visual aspects of a measure, and not just its cell data, requires permissions on dimension members. This section explains how to deny access to the object metadata of a measure.

1. On **Dimension Data** | **Basic**, scroll down the Dimension list until you reach cube dimensions, and then

select **Measures Dimension**.

2. From the list of measures, clear the check box for measures that should not appear to users connecting via this role.

> **NOTE**
> Check the Prerequisites to learn how to identify measures that can break role security.

Advanced dimension security

If you have MDX expertise, another approach is to write MDX expressions that set the criteria for which members are allowed or denied access. Click **Create Role** | **Dimension Data** | **Advanced** to provide the script.

You can the MDX Builder to write the MDX statement. See MDX Builder (Analysis Services - Multidimensional Data) for details. The **Advanced** tab has the following options:

Attribute
Select the attribute for which you want to manage member security.

Allowed member set
The AllowedSet can resolve to no members (default), all members, or some members. If you allow access to an attribute and do not define any members of the allowed set, access to all members is granted. If you allow access to an attribute and define a specific set of attribute members, only the explicitly allowed members are visible.

Creating an AllowedSet has a ripple effect when the attribute participates in a multi-level hierarchy. For example, suppose a role allows access to Washington state (assume a scenario where the role is granting permissions to a company's Washington state sales division). For people connecting through this role, queries that include ancestors (United States) or descendants (Seattle and Redmond) will only see members in a chain including Washington state. Because other states are not explicitly allowed, the effect will be the same as if they were denied.

> **NOTE**
> If you define an empty set ({}) of attribute members, no members of the attribute will be visible to the database role. The absence of an allowed set is not interpreted as an empty set.

Denied member set
The DeniedSet property can resolve to no members, all members (default), or some attribute members. When the denied set contains only a specific set of attribute members, the database role is denied access only to those specific members, as well as descendants if the attribute is in a multi-level hierarchy. Consider the Washington state sales division example. If Washington is placed in the DeniedSet, people connecting through this role will see all other states except Washington and its descendent attributes.

Recall from the previous section that the denied set is a fixed collection. If processing subsequently introduces new members that should also be denied access, you will need to edit this role to add those members to the list.

Default member
The DefaultMember property determines the data set returned to a client when an attribute is not explicitly included in a query. When the attribute is not explicitly included, Analysis Services uses one of the following default members for the attribute:

- If the database role defines a default member for the attribute, Analysis Services uses this default member.
- If the database role does not define a default member for the attribute, Analysis Services uses the default

member that is defined for the attribute itself. The default member for an attribute, unless you specify otherwise, is the **All** member (unless the attribute is defined as non-aggregatable).

For example, suppose a database role specifies **Male** as the default member for the **Gender** attribute. Unless a query both explicitly includes the **Gender** attribute and specifies a different member for this attribute, Analysis Services would return a data set that included only male customers. For more information about setting the default member, see Define a Default Member.

Enable Visual Total

The VisualTotals property indicates whether the aggregated cell values that are displayed are calculated according to all cell values or only according to the cell values that are visible to the database role.

By default, the VisualTotals property is disabled (set to **False**). This default setting maximizes performance because Analysis Services can quickly calculate the total of all cell values, instead of having to spend time selecting which cells values to calculate.

However, having the VisualTotals property disabled could create a security issue if a user can use the aggregated cell values to deduce values for attribute members to which the user's database role does not have access. For example, Analysis Services uses the values for three attribute members to calculate an aggregated cell value. The database role has access to view two of these three attribute members. Using the aggregated cell value, a member of this database role would be able to deduce the value for the third attribute member.

Setting VisualTotals property to **True** can eliminate this risk. When you enable the VisualTotals property, a database role can only view aggregated totals for dimension members to which the role has permission.

Check

Click to test the MDX syntax defined on this page.

See Also

Grant cube or model permissions (Analysis Services)
Grant custom access to cell data (Analysis Services)
Grant permissions on data mining structures and models (Analysis Services)
Grant permissions on a data source object (Analysis Services)

Grant custom access to cell data (Analysis Services)

11/28/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Cell security is used to allow or deny access to measure data within a cube. The following illustration shows a combination of allowed and denied measures in a PivotTable, when connected as a user whose role only allows access to certain measures. In this example, **Reseller Sales Amount** and **Reseller Total Product Cost** are the only measures available through this role. All other measures are implicitly denied (the steps used to get this result are provided below in the next section, Allow access to specific measures).

Row Labels	Internet Order Count	Internet Gross Profit	Reseller Average Unit Price	Reseller Sales Amount	Reseller Total Product Cost
⊞Specialty Bike Shop	#N/A	#N/A	#N/A	$6,756,166.18	$6,728,368.53
⊞Value Added Reseller	#N/A	#N/A	#N/A	$34,967,517.33	$34,520,042.39
⊞Warehouse	#N/A	#N/A	#N/A	$38,726,913.48	$38,731,703.45
Grand Total	**#N/A**	**#N/A**	**#N/A**	**$80,450,596.98**	**$79,980,114.38**

Cell permissions apply to data inside the cell, and not to its metadata. Notice how the cell is still visible in the results of a query, displaying a value of **#N/A** instead of the actual cell value. The **#N/A** value appears in the cell unless the client application translates the value, or another value is specified by setting the Secured Cell Value property in the connection string.

To hide the cell entirely, you have to limit the members-dimensions, dimension attributes, and dimension attribute members-that are viewable. For more information, see Grant custom access to dimension data (Analysis Services).

As an administrator, you can specify whether role members have read, read contingent, or read/write permissions on cells within a cube. Putting permissions on a cell is the lowest level of security allowed, so before you start applying permissions at this level, it's important to keep a few facts in mind:

- Cell-level security cannot expand rights that have been restricted at a higher level. An example: if a role denies access to dimension data, cell-level security cannot override the denied set. Another example: consider a role with **Read** permission on a cube and **Read/Write** permission on a cell — the cell data permission will not be **Read/Write**; it will be **Read**.
- Custom permissions often need to be coordinated between dimension members and cells within the same role. For example, suppose you want to deny access to several discount-related measures for different combinations of resellers. Given **Resellers** as dimension data and **Discount Amount** as a measure, you would need to combine within the same role permissions on both the measure (using the instructions in this topic), as well as on dimension members. See Grant custom access to dimension data (Analysis Services) for details on setting dimension permissions.

Cell-level security is specified through MDX expressions. Because a cell is a tuple (that is, an intersection point across potentially multiple dimensions and measures), it is necessary to use MDX to identify specific cells.

Allow access to specific measures

You can use cell security to explicitly choose which measures are available. Once you specifically identify which members are allowed, all other measures become unavailable. This is perhaps the simplest scenario to implement via MDX script, as the following steps illustrate.

1. In SQL Server Management Studio connect to the instance of Analysis Services, select a database, open the **Roles** folder, and then click a database role (or create a new database role). Membership should already be specified, and the role should have **Read** access to the cube. See Grant cube or model

permissions (Analysis Services) if you need help with this step.

2. In **Cell Data**, check the cube selection to be sure you have chosen the right one, and then select **Enable read permissions**.

   If you select just this check box, and do not provide an MDX expression, the effect is the same as denying access to all cells in the cube. This is because the default allowed set is an empty set whenever Analysis Services resolves a subset of cube cells.

3. Enter the following MDX expression.

```
(Measures.CurrentMember IS [Measures].[Reseller Sales Amount]) OR (Measures.CurrentMember IS
[Measures].[Reseller Total Product Cost])
```

   This expression explicitly identifies which measures are visible to users. No other measures will be available to users connecting through this role. Notice that CurrentMember (MDX) sets the context and is followed by the measure that is allowed. The effect of this expression is, if the current member includes either the **Reseller Sales Amount** or the **Reseller Total Product Cost**, show the value. Otherwise, deny access. The expression has multiple parts, with each part enclosed in parentheses. The **OR** operator is used to specify multiple measures.

Deny access to specific measures

The following MDX expression, also specified in **Create Role** | **Cell Data** | **Allow reading of cube content**, has the opposite effect, making certain measures unavailable. In this example, **Discount Amount** and **Discount Percentage** are made unavailable using the **NOT** and **AND** operators. All other measures will be visible to users connecting through this role.

```
(NOT Measures.CurrentMember IS [Measures].[Discount Amount]) AND (NOT Measures.CurrentMember IS [Measures].
[Discount Percentage])
```

In Excel, cell-security is evident in the following illustration:

A	B	C	D
Row Labels	Reseller Sales Amount	Discount Amount	Discount Percentage
Specialty Bike Shop	$6,756,166.18	#N/A	#N/A
Value Added Reseller	$34,967,517.33	#N/A	#N/A
Warehouse	$38,726,913.48	#N/A	#N/A
Grand Total	$80,450,596.98	#N/A	#N/A

Set Read permissions on calculated measures

Permissions on a calculated measure can be set independently of its constituent parts. Skip ahead to the next section on Read-Contingent if you want to coordinate permissions between a calculated measure and its dependent measures.

To understand how Read permissions work for a calculated measure, consider **Reseller Gross Profit** in AdventureWorks. It's derived from **Reseller Sales Amount** and **Reseller Total Product Cost** measures. As long as a role has Read permission on **Reseller Gross Profit** cells, this measure is viewable even if permissions are expressly denied on the other measures. As a demonstration, copy the following MDX expression into **Create Role** | **Cell Data** | **Allow reading of cube content**.

```
(NOT Measures.CurrentMember IS [Measures].[Reseller Sales Amount])
AND (NOT Measures.CurrentMember IS [Measures].[Reseller Total Product Cost])
```

In Excel, connect to the cube using the current role, and choose all three measures to see the effects of cell

security. Notice that measures in the denied set are unavailable, but the calculated measure is visible to the user.

Row Labels	Reseller Sales Amount	Reseller Total Product Cost	Reseller Gross Profit
⊞Specialty Bike Shop	#N/A	#N/A	$27,797.65
⊞Value Added Reseller	#N/A	#N/A	$447,474.93
⊞Warehouse	#N/A	#N/A	($4,789.98)
Grand Total	**#N/A**	**#N/A**	**$470,482.60**

Set Read-Contingent permissions on calculated measures

Cell-security offers an alternative, Read-Contingent, for setting permissions on the related cells participating in a calculation. Consider again the **Reseller Gross Profit** example. When you enter the same MDX expression provided in the previous section, placed this time into the second text area of the **Create Role** | **Cell data** dialog box (in the text area below **Allow reading of cell content contingent on cell security**), the result is apparent when viewed in Excel. Because **Reseller Gross Profit** is contingent upon **Reseller Sales Amount** and **Reseller Total Product Cost**, gross profit is now inaccessible because its constituent parts are inaccessible.

> **NOTE**
>
> What happens if you set both the Read and Read-Contingent permissions on a cell within the same role? The role will provide Read permissions on the cell, and not Read-Contingent.

Recall from previous sections that selecting just the **Enable read-contingent permissions** checkbox, without providing any MDX expression, denies access to all cells in the cube. This is because the default allowed set is an empty set whenever Analysis Services resolves a subset of cube cells.

Set Read/Write permissions on a cell

Read/write permissions on a cell are used to enable writeback, provided that members have read/write permissions to the cube itself. Permissions that are granted at the cell level cannot be greater than the permissions that are granted at the cube level. See Set Partition Writeback for details.

See Also

MDX Builder (Analysis Services - Multidimensional Data)
The Basic MDX Script (MDX)
Grant process permissions (Analysis Services)
Grant permissions on a dimension (Analysis Services)
Grant custom access to dimension data (Analysis Services)
Grant cube or model permissions (Analysis Services)

Grant permissions on a data source object (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Typically, most users of Analysis Services do not require access to the data sources that underlie an Analysis Services project. Users ordinarily just query the data within an Analysis Services database. However, in the context of data mining, such as performing predictions based on a mining model, a user has to join the learned data of a mining model with user-provided data. To connect to the data source that contains the user-provided data, the user uses a Data Mining Extensions (DMX) query that contains either the OPENQUERY (DMX) and OPENROWSET (DMX) clause.

To execute a DMX query that connects to a data source, the user must have access to the data source object within the Analysis Services database. By default, only Server Administrators or Database Administrators have access to data source objects. This means that a user cannot access a data source object unless an administrator grants permissions.

> **IMPORTANT**
>
> For security reasons, the submission of DMX queries by using an open connection string in the OPENROWSET clause is disabled.

Set Read permissions to a data source

A database role can be granted either no access permissions on a data source object or read permissions.

1. In SQL Server Management Studio, connect to the instance of Analysis Services, expand **Roles** for the appropriate database in Object explorer, and then click a database role (or create a new database role).
2. In the **Data Source Access** pane, locate the data source object in the **Data Source** list, and then select the **Read** in the **Access** list for the data source. If this option is unavailable, check the **General** pane to see if Full Control is selected. Full Control is already providing permission, you cannot override permissions on the data source.

Working With the Connection String Used by a Data Source Object

The data source object contains the connection string that is used to connect to the underlying data source. This connection string can specify one of the following:

- **Specify a user name and password**

  If the connection string that a data source object uses specifies a user name and password, you may want to create multiple data source objects, each with different user accounts. Creating multiple data source objects lets users access certain data source objects and prevents those users from accessing other data source objects. These other data source objects can be used by Analysis Services itself for processing objects, such as cubes and mining models.

- **Specify Windows Authentication**

  If the connection string that a data source object uses specifies Windows Authentication, Analysis Services

must be able to impersonate the client. If the data source is on a remote computer, the two computers must be trusted for impersonation by using Kerberos authentication, or the query will typically fail. See Configure Analysis Services for Kerberos constrained delegation for more information.

If the client does not allow for impersonation (through the Impersonation Level property in OLE DB and other client components), Analysis Services will try to make an anonymous connection to the underlying data source. Anonymous connections to remote data sources rarely succeed, as most data sources do not accept anonymous connections).

See Also

Data Sources in Multidimensional Models
Connection String Properties (Analysis Services)
Authentication methodologies supported by Analysis Services
Grant custom access to dimension data (Analysis Services)
Grant cube or model permissions (Analysis Services)
Grant custom access to cell data (Analysis Services)

Grant permissions on data mining structures and models (Analysis Services)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

By default, only an Analysis Services server administrator has permissions to view data mining structures or mining models in the database. Follow the instructions below to grant permissions to non-administrator users.

Set permissions to access a mining structure

1. In SSMS, connect to Analysis Services. See Connect from client applications (Analysis Services) if you need help with the steps.
2. Open the **Databases** folder, and select a database in Object Explorer.
3. Right-click **Roles** and choose **New Role**.
4. In the General page, enter a name, and optionally, a description. The page also contains several database permissions, such as Full Control, Process Database, and Read Definition. None of these permissions are needed for data mining access. See Grant database permissions (Analysis Services) for more information about database permissions.
5. In the **Mining Structure** pane, select **Read** or **Read/Write** for each data mining structure.
6. In the **Membership** pane, enter the Windows user and group accounts that connect to Analysis Services using this role.
7. Click **OK** to finish creating the role.

Set permissions to access a mining model

For a data mining model, a role can have either **Read** or **Read/Write** permissions, as well as **Drillthrough** and **Read Definition** permissions that allow viewing and browsing the underlying data.

Note If you enable drillthrough on both the mining structure and the mining model, any user who is a member of a role that has drillthrough permissions on the mining model and the mining structure can also view columns in the mining structure, even if those columns are not included in the mining model. Therefore, to protect sensitive information, you should set up the data source view to mask personal information, and allow drillthrough access on the mining structure only when necessary.

To grant read or read/write permissions to a database role, a user must be a member of the Analysis Services server role or a member of an Analysis Services database role that has Full Control (Administrator) permissions.

1. In SQL Server Management Studio, connect to the instance of Analysis Services, expand **Roles** for the appropriate database in Object Explorer, and then click a database role (or create a new database role).
2. In the **Mining Structure** pane, locate the mining model in the **Mining Models** list, and then select **Read**, **Read/Write**, **Drill Through**, or **Browse** for that mining model.
3. In the **Membership** pane, enter the Windows user and group accounts that connect to Analysis Services using this role.
4. Click **OK** to finish creating the role.

To use a data source in a drillthrough query that uses the Data Mining Extensions (DMX) OPENQUERY clause, the database role also needs read/write permission on the appropriate data source object. For more information, see Grant permissions on a data source object (Analysis Services) and OPENQUERY (DMX).

> **NOTE**
>
> By default, the submission of DMX queries by using OPENROWSET is disabled.

See Also

Grant server admin rights to an Analysis Services instance
Grant cube or model permissions (Analysis Services)
Grant custom access to dimension data (Analysis Services)
Grant custom access to cell data (Analysis Services)

Power View for Multidimensional Models

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Microsoft SQL Server 2016 Analysis Services multidimensional models support ad-hoc data analysis, exploration, and visualization by using Microsoft Power View.

In this section

- Understanding Power View for Multidimensional Models
- Create a Report Data Source
- Create a Power View Report with a Multidimensional Data Source

Understanding Power View for Multidimensional Models

11/28/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article describes the Power View for Multidimensional Models feature in SQL Server, and provides important information for BI professionals and administrators who intend to implement Power View for Multidimensional Models in their organization.

Multidimensional models provide industry leading OLAP data modeling, storage, and analysis solutions. Multidimensional models in SQL Server support ad-hoc data analysis, exploration, and visualization by using Microsoft Power View.

Power View is a thin web client that launches in the browser from a shared Report Data Source (.rsds) file in a SharePoint library. The Report Data Source acts as a bridge between the client and the back-end data source. The back-end data source can be a Power Pivot workbook in SharePoint, a Tabular model on an Analysis Services server running in Tabular mode, or a Multidimensional model on an Analysis Services server running in Multidimensional mode. Power View reports can then be saved to a SharePoint library or gallery and shared with other members in your organization.

Power View for Multidimensional Models architecture

Prerequisites

Server Requirements

- Microsoft SQL Server 2016 Analysis Services running in Multidimensional mode.
- Microsoft SQL Server 2016 Reporting Services Add-in for Microsoft SharePoint Server 2010 or later Enterprise Edition.

Client Requirements

- Power View client functionality requires Microsoft Silverlight 5. For more information, see Browser Support for Reporting Services and Power View.

Features

Native support for Power View

With this release, multidimensional models support analysis and visualization by using Power View in SharePoint mode. No special configuration of your multidimensional models is necessary. There are however some

differences in how multidimensional model objects are displayed in Power View compared to other client tools such as Microsoft Excel and Microsoft Performance Point. This release does not support analysis and visualization of multidimensional models by using Power View in Excel.

Native support for DAX queries

With this release, multidimensional models support DAX queries and functions in addition to more traditional MDX queries. Some DAX functions, such as PATH, are not applicable in multidimensional modeling. For a better understanding of DAX and how it compares to MDX, see Data Analysis Expressions and MDX.

Multidimensional to tabular object mapping

Analysis Services provides a tabular model metadata representation of a multidimensional model. Objects in a multidimensional model are represented as tabular objects in Power View and in CSDL out with BI annotations.

Object mapping summary

MULTIDIMENSIONAL OBJECT	TABULAR OBJECT
Cube	Model
Cube Dimension	Table
Dimension Attributes (Key(s), Name)	Column
Measure Group	Table
Measure	Measure
Measure without Measure Group	In a table named Measures
Measure Group Cube Dimension Relationship	Relationship
Perspective	Perspective
KPI	KPI
User/Parent-Child Hierarchies	Hierarchy
Display Folder	Display Folder

Measures, measure groups, and KPIs

> **NOTE**
> Some images and text in this article refer to the Adventure Works Multidimensional Model for SQL Server 2012 sample database.

Measure groups in a multidimensional cube are seen in the Power View Field List as tables with the sigma (∑) sign.

Measure groups in the Power View Field List

Measures within a measure group appear as measures. If there are calculated measures that do not have an associated measure group, they will be grouped under a special table called Measures.

To help simplify more complex multidimensional models, model authors can define a set of measures or KPIs in a cube to be located within a display folder. Power View can show display folders and the measures and KPIs in them.

Measures and KPIs in a measure group

Measures as variants

Measures in multidimensional models are variants. This means the measures are not strongly typed and can have different data types. For example, in the image below, the Amount measure in the Financial Reporting table by default is Currency data type, but also has a string value "NA" for the sub-total of "Statistical Accounts", which is String data type. Power View recognizes certain measures as variants and shows the correct values and formatting in the different visualizations.

Measure as variant

Account Level 01	Account Level 02	Amount
Balance Sheet	Assets	$13,740,731.00
	Liabilities and Owners Equity	$13,740,731.00
	Total	**$0.00**
Net Income	Operating Profit	$16,728,234.50
	Other Income and Expense	$50,017.50
	Taxes	$4,168,749.00
	Total	**$12,609,503.00**
Statistical Accounts	Headcount	188
	Units	5,305
	Average Unit Price	$12,182.18
	Square Footage	276,500
	Total	**NA**

Implicit measures

Tabular models provide users the ability to create *implicit* measures such as count, sum, or average on fields. For multidimensional models, because dimension attribute data is stored differently, querying implicit measures can take a long time. Because of this, implicit measures are not available in Power View.

Dimensions, attributes, and hierarchies

Cube dimensions are exposed as tables in tabular metadata. In the Power View Field List, dimension attributes are shown as columns within display folders. The dimension attributes that have the AttributeHierarchyEnabled property set to false; for example: Birth Date attribute in Customer dimension, or AttributeHierarchyVisible property set to false will not appear in the Power View Field List. Multi-level hierarchies or user hierarchies; for example Customer Geography in the Customer dimension, are exposed as hierarchies in the Power View Field List. Hidden UnknownMembers of a dimension attribute are exposed in DAX Queries and in Power View.

Dimension, attributes and hierarchies in SQL Server Data Tools (SSDT) and Power View Field List

Dimension attribute type

Multidimensional models support associating dimension attributes with specific dimension attribute types. The image below shows the Geography dimension where the City, State-Province, Country and Postal Code dimension attributes have geography types associated with them. These are exposed in the tabular metadata. Power View recognizes the metadata enabling users to create map visualizations. This is indicated by the map icon next to the City, Country, Postal Code and State-Province columns in the Geography table in the Power View Field List.

Dimension attribute geography types in SSDT and Power View Field List

Dimension calculated members

Multidimensional models support calculated members for child of All with a single real member. Additional constraints while exposing this type of calculated member are:

- Must be a single real member when the dimension has more than one attribute.
- An attribute containing calculated members cannot be the key attribute of the dimension unless it is the only attribute.
- An attribute containing calculated members cannot be a parent-child attribute.

Calculated members of user hierarchies are not exposed in Power View; however, end-users will still be able to connect to a cube containing calculated members on user hierarchies.

The image below shows a Power View report for a cube that contains time-intelligence calculated members on dimension attribute "Fiscal Date Calculations" in the Date dimension.

Power View report with calculated members

Default members

Multidimensional models support default members for dimension attributes. The default member is used by Analysis Services when aggregating data for a query. The default member of a dimension attribute is exposed as default value or filter for the corresponding column in the tabular metadata.

Power View behaves much the same as Excel PivotTables when attributes are applied. When a user adds a column to a Power View visualization (table, matrix, or chart) that contains a default value, the default value will not be applied and all available values are shown. If the user adds the column to Filters, the default value will be applied.

Dimension security

Multidimensional models support dimension and cell level security through roles. A user connecting to a cube by using Power View is authenticated and evaluated for appropriate permissions. When dimension security is applied, the respective dimension members will not be seen by the user in Power View; however if a user has a cell security permission defined where certain cells are restricted, then that user cannot connect to the cube with Power View. In some cases, users can see aggregate data when portions of that data are calculated from secured data.

Non-aggregatable attributes/hierarchies

In a multidimensional model, attributes of a dimension can have the IsAggregatable property set to false. This means the model author has specified client applications should not aggregate the data across hierarchies (attribute or multi-level) when they query the data. In Power View, this dimension attribute is exposed as a column for which sub-totals are not available. In the image below, you can see an example of a non-aggregatable hierarchy: Accounts. The top-most level of the Accounts parent-child hierarchy is non-aggregatable while other levels are aggregatable. In a matrix visualization of the Accounts hierarchy (first two levels), you see sub-totals for Account Level 02 but not for the top-most level, Account Level 01.

Non-aggregatable hierarchy in Power View

Account Level 01	Account Level 02	Amount
Balance Sheet	Assets	$13,740,731.00
	Liabilities and Owners Equity	$13,740,731.00
	Total	**$0.00**
Net Income	Operating Profit	$16,728,234.50
	Other Income and Expense	$50,017.50
	Taxes	$4,168,749.00
	Total	**$12,609,503.00**
Statistical Accounts	Headcount	188
	Units	5,305
	Average Unit Price	$12,182.18
	Square Footage	276,500
	Total	**NA**

Images

Power View provides the ability to render images. In multidimensional models, one of the ways you can provide images to Power View is to expose columns containing URLs (Uniform Resource Locator) of the images. With this release, Analysis Services supports tagging dimension attributes as type ImageURL. This data type is then provided to Power View in the tabular metadata. Power View can then download and display the images specified in the URLs within visualizations.

ImageURL dimension attribute type in SSDT

Parent-child hierarchies

Multidimensional models support parent-child hierarchies, which are exposed as a hierarchy in the tabular metadata. Each level of the parent-child hierarchy is exposed as a hidden column. The key attribute of the parent-child dimension is not exposed in the tabular metadata.

Parent-child hierarchies in Power View

Perspectives and translations

Perspectives are views of cubes where only certain dimensions or measure groups are visible in client tools. You can specify a perspective name as a value to the Cube connection string property. For example, in the following connection string, 'Direct Sales' is a perspective in the multidimensional model:

```
Data Source=localost;Initial Catalog=AdventureWorksDW-MD;Cube='Direct Sales'
```

Cubes can have metadata and data translations specified for various Languages within the model. In order to see the translations (data and metadata) you need to add the optional "Locale Identifier" property to the connection string in the RSDS file as shown below.

```
Data Source=localost;Initial Catalog=AdventureWorksDW-MD;Cube='Adventure Works'; Locale Identifier=3084
```

When Power View connects to a multidimensional model with an .rsds file that has a Locale Identifier, and if a corresponding translation is contained in the cube, users will see the translations in Power View.

For more information, see Create a Report Data Source.

Power View pinned filters

Power View reports can contain multiple views. With this release, the *Pin Filter* feature for both tabular and multidimensional models provides the ability to create filters that apply across all views in a report. The image below shows the Pin filter toggle button for a view filter. By default, a view filter is unpinned and applies only to that view. Pinning a view filter applies it to all views; unpinning it removes it from other views.

Pinned Filters

Unsupported features

Power View in Excel 2013 - does not support connecting to and creating reports for multidimensional models. However, **Power View in Excel 2016** does support connecting to and creating reports for multidimensional models. To learn more, see Power View and OLAP in Excel 2016

Actions - are not supported in Power View reports or in DAX queries against a multidimensional model.

Named sets - in multidimensional models, are not supported in Power View or in DAX queries against a multidimensional model.

> **NOTE**
> Unsupported Actions and Named sets do not prevent users from connecting to and exploring multidimensional models using Power View.

Cell level security - is not supported in Power View reports.

CSDLBI Annotations

Multidimensional cube metadata is exposed as an Entity Data Model (EDM) based conceptual model by Conceptual Schema Definition Language with Business Intelligence annotations (CSDLBI).

Multidimensional metadata is represented as a tabular model namespace in a CSDLBI document, or CSDL out, when a DISCOVER_CSDL_METADATA request is sent to the Analysis Services instance.

Sample DISCOVER_CSDL_METADATA request

```
<Envelopexmlns="http://schemas.xmlsoap.org/soap/envelope/">
   <Body>
      <Discoverxmlns="urn:schemas-microsoft-com:xml-analysis">
         <RequestType>DISCOVER_CSDL_METADATA</RequestType>
         <Restrictions>
            <RestrictionList>
              <CATALOG_NAME>"catalogname"<CATALOG_NAME>
            </RestrictionList>
         </Restrictions>
         <Properties>
            <PropertyList>
            </PropertyList>
         </Properties>
      </Discover>
   </Body>
</Envelope>
```

DISCOVER_CSDL_METADATA request has the following restrictions:

NAME	REQUIRED	DESCRIPTION
CATALOG_NAME	Yes	The catalog\database name.
PERSPECTIVE_NAME	Yes, if the cube contains more than one perspective. Optional if there is only one cube or there is a default perspective.	The cube name or perspective name in the multidimensional database.
VERSION	Yes	CSDL version requested by client. Multidimensional features and constructs are supported in version 2.0.

The return CSDL out document represents the model as a namespace, containing entities, associations, and properties.

For more detailed information about CSDLBI annotations for tabular models, see Technical Reference for BI Annotations to CSDL on MSDN, and [MS-CSDLBI]: Conceptual Schema Definitions File Format with Business Intelligence Annotations.

Client Help on Office.com

The following articles are provided on Office.com to help users learn about how Multidimensional Model objects appear in Power View and how to create a sample report:

Understanding Multidimensional Model Objects in Power View

Explore the Adventure Works Multidimensional Model by using Power View

Create a Report Data Source

11/27/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In order for Power View to connect to a multidimensional model, you must create a shared report data source definition, also known as an .rsds file, in a SharePoint library. The .rsds file specifies the name of an Analysis Services server instance, connection type, connection string, and credentials used to connect to the multidimensional model. When a user clicks on the .rsds, a new blank Power View report (an .rdlx file) opens in the browser.

In order to create an .rsds connection, you must have SQL Server 2012 (or later) Reporting Services and the Reporting Services add-in for SharePoint 2010 or SharePoint 2013 installed.

Create a Report Data Source (.rsds) connection to a multidimensional model

Before you begin, you need to know:

- The name of the Analysis Services server instance running in Multidimensional mode.
- The name of the multidimensional database you want to connect to.
- The name of the cube, if there is more than one.
- (Optional) Perspective name.
- (Optional) Locale Identifier.

To create a shared Report Data Source (.rsds) file (SharePoint 2010)

1. Click the **Documents** tab on the library ribbon.
2. Click **New Document** > **Report Data Source**.

   > **NOTE**
   > If you do not see the **Report Data Source** item on the menu, the report data source content type has not been enabled for this library. For more information, see Add Reporting Services Content Types to a SharePoint Library.

3. On the **Data Source Properties** page, in **Name**, type a name for the connection .rsds file.
4. In **Data Source Type**, select **Microsoft BI Semantic Model for Power View**.
5. In **Connection String**, specify the Analysis Services server name, database name, cube name, and any optional settings.

   Connection String:

   ```
   Data source=<servername>;initial catalog=<multidimensionaldatabasename>-ee;cube='<cubename>'
   ```

   > **NOTE**
   > If there is more than one cube, you must specify a cube name.

(Optional) Cubes can have perspectives that provide users a select view where only certain dimensions and/or measure groups are visible in the client. To specify a perspective, enter the perspective name as a value to the Cube property:

```
Data source=<servername>;initial catalog=<multidimensionaldatabasename>-ee;cube='<perspectivename>'
```

(Optional) Cubes can have metadata and data translations specified for various languages within the model. In order to see the translations (data and metadata) you need to add the "Locale Identifier" property to the connection string:

```
Data source=<servername>;initial catalog=<multidimensionaldatabasename>-ee;cube='<cubename>'; Locale
Identifier=<identifier number>
```

6. In **Credentials**, specify how the report server obtains credentials to access the external data source.

   - Select **Windows authentication (integrated)** if you want to access the data using the credentials of the user who opened the report. Do not select this option if the SharePoint site or farm uses forms authentication or connects to the report server through a trusted account. Do not select this option if you want to schedule subscription or data processing for this report. This option works best when Kerberos authentication is enabled for your domain, or when the data source is on the same computer as the report server. If Kerberos authentication is not enabled, Windows credentials can only be passed to one other computer. This means that if the external data source is on another computer, requiring an additional connection, you will get an error instead of the data you expect.

   - Select **Prompt for credentials** if you want the user to enter his or her credentials each time he or she runs the report. Do not select this option if you want to schedule subscription or data processing for this report.

   - Select **Stored credentials** if you want to access the data using a single set of credentials. The credentials are encrypted before they are stored. You can select options that determine how the stored credentials are authenticated. Select Use as Windows credentials if the stored credentials belong to a Windows user account. Select **Set execution context to this account** if you want to set the execution context on the database server.

   - Select **Credentials are not required** if you specify credentials in the connection string, or if you want to run the report using a least-privilege account.

7. Click **Test Connection** to validate.

8. Select **Enable this data source** if you want the data source to be active. If the data source is configured but not active, users will receive an error message when they attempt to create a report.

Create a Power View Report with a Multidimensional Data Source

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Creating a Power View report based on a multidimensional model is no different than creating a report based on a Power Pivot workbook or an Analysis Services Tabular model. Power View reports are created from a Report Data Source connection file (.rsds) in a SharePoint library. For more information about how to create an .rsds, see Create a Report Data Source.

Before you begin, you need to know:

- The name and location of the SharePoint library that contains a shared report data source connection (.rsds) to a multidimensional model.

Create a new, blank Power View report

- In the SharePoint library, click the arrow next to the .rsds shared report data source connection (the .rsds that connects to the multidimensional model), and then click **Create Power View Report**.

Multidimensional Model Assemblies Management

12/10/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Microsoft SQL Server Analysis Services supplies lots of intrinsic functions for use with the Multidimensional Expressions (MDX) and Data Mining Extensions (DMX) languages, designed to accomplish everything from standard statistical calculations to traversing members in a hierarchy. But, as with any other complex and robust product, there is always the need to extend the functionality of such a product further.

Therefore, Analysis Services lets you add assemblies to an Analysis Services instance or database. Assemblies let you create external, user-defined functions using any common language runtime (CLR) language, such as Microsoft Visual Basic .NET or Microsoft Visual C#. You can also use Component Object Model (COM) Automation languages such as Microsoft Visual Basic or Microsoft Visual C++.

> **IMPORTANT**
>
> COM assemblies might pose a security risk. Due to this risk and other considerations, COM assemblies were deprecated in SQL Server 2008 Analysis Services (SSAS). COM assemblies might not be supported in future releases.

Assemblies let you extend the business functionality of MDX and DMX. You build the functionality that you want into a library, such as a dynamic link library (DLL) and add the library as an assembly to an instance of Analysis Services or to an Analysis Services database. The public methods in the library are then exposed as user-defined functions to MDX and DMX expressions, procedures, calculations, actions, and client applications.

An assembly with new procedures and functions can be added to the server. You can use assemblies to enhance or add custom functionality that is not provided by the server. By using assemblies, you can add new functions to Multidimensional Expressions (MDX), Data Mining Extensions (DMX), or stored procedures. Assemblies are loaded from the location where the custom application is run, and a copy of the assembly binary file is saved along with the database data in the server. When an assembly is removed, the copied assembly is also removed from the server.

Assemblies can be of two different types: COM and CLR. CLR assemblies are assemblies developed in .NET Framework programming languages such as C#, Visual Basic .NET, managed C++. COM assemblies are COM libraries that must be registered in the server

Assemblies can be added to Server or Database objects. Server assemblies can be called by any user connected to the server or any object in the server. Database assemblies can be called only by Database objects or users connected to the database.

A simple Assembly object is composed of basic information (Name and Id), file collection, and security specifications.

The file collection refers to the loaded assembly files and their corresponding debugging (.pdb) files, if the debugging files were loaded with the assembly files. Assembly files are loaded from the location where the application defined the files to, and a copy is saved in the server along with the data. The copy of the assembly file is used to load the assembly every time the service is started.

Security specifications include the permission set and the impersonation used to run the assembly.

Calling User-Defined Functions

Calling a user-defined function in an assembly is performed just like calling an intrinsic function, except that you must use a fully qualified name. For example, a user-defined function that returns a type expected by MDX is included in an MDX query, as shown in the following example:

```
Select MyAssembly.MyClass.MyStoredProcedure(a, b, c) on 0 from Sales
```

User-defined functions can also be called using the CALL keyword. You must use the CALL keyword for user-defined functions which return recordsets or void values, and you cannot use the CALL keyword if the user-defined function depends on an object in the context of the MDX or DMX statement or script, such as the current cube or data mining model. A common use for a function called outside an MDX or DMX query is to use the AMO object model to perform administrative functions. If, for example, you wanted to use the function `MyVoidProcedure(a, b, c)` in an MDX statement, the following syntax would be employed:

```
Call MyAssembly.MyClass.MyVoidProcedure(a, b, c)
```

Assemblies simplify database development by enabling common code to be developed once and stored in a single location. Client software developers can create libraries of functions for Analysis Services and distribute them with their applications.

Assemblies and user-defined functions can duplicate the function names of the Analysis Services function library or of other assemblies. As long as you call the user-defined function by using its fully qualified name, Analysis Services will use the correct procedure. For security purposes, and to eliminate the chance of calling a duplicate name in a different class library, Analysis Services requires that you use only fully qualified names for stored procedures.

To call a user-defined function from a specific CLR assembly, the user-defined function is preceded by the assembly name, full class name, and procedure name, as demonstrated here:

AssemblyName.*FullClassName*.*ProcedureName*(*Argument1*, *Argument2*, ...)

For backward compatibility with earlier versions of Analysis Services, the following syntax is also acceptable:

AssemblyName!*FullClassName*!*ProcedureName*(*Argument1*, *Argument2*, ...)

If a COM library supports multiple interfaces, the interface ID can also be used to resolve the procedure name, as demonstrated here:

AssemblyName!*InterfaceID*!*ProcedureName*(*Argument1*, *Argument2*, ...)

Security

Security for assemblies is based on the .NET Framework security model, which is a code-access security model. .NET Framework supports a code-access security mechanism that assumes that the runtime can host both fully trusted and partially trusted code. The resources that are protected by .NET Framework code access security are typically wrapped by managed code which demands the corresponding permission before enabling access to the resource. The demand for the permission is satisfied only if all the callers (at the assembly level) in the call stack have the corresponding resource permission.

For assemblies, permission for execution is passed with the **PermissionSet** property on the **Assembly** object. The permissions that managed code receives are determined by the security policy in effect. There are already three levels of policy in effect in a non- Analysis Services hosted environment: enterprise, computer and user. The effective list of permissions that code receives is determined by the intersection of the permissions obtained by these three levels.

Analysis Services supplies a host-level security policy level to the CLR while hosting it; this policy is an additional

policy level below the three policy levels that are always in effect. This policy is set for every application domain that is created by Analysis Services.

The Analysis Services host-level policy is a combination of Analysis Services fixed policy for system assemblies and user-specified policy for user assemblies. The user-specified piece of the Analysis Services host policy is based on the assembly owner specifying one of three permission buckets for each assembly:

PERMISSION SETTING	DESCRIPTION
Safe	Provides internal computation permission. This permission bucket does not assign permissions to access any of the protected resources in the .NET Framework. This is the default permission bucket for an assembly if none is specified with the **PermissionSet** property.
ExternalAccess	Provides the same access as the **Safe** setting, with the additional ability to access external system resources. This permission bucket does not offer security guarantees (although it is possible to secure this scenario), but it does give reliability guarantees.
Unsafe	Provides no restrictions. No security or reliability guarantees can be made for managed code running under this permission set. Any permission, even a custom permission included by the administrator, is granted to code running at this level of trust.

When CLR is hosted by Analysis Services, the stack-walk based permission check stops at the boundary with native Analysis Services code. Any managed code in Analysis Services assemblies always falls into one of the three permission categories listed earlier.

COM (or unmanaged) assembly routines do not support the CLR security model.

Impersonation

Whenever managed code accesses any resource outside Analysis Services, Analysis Services follows the rules associated with the **ImpersonationMode** property setting of the assembly to make sure that the access occurs in an appropriate Windows security context. Because assemblies using the **Safe** permission setting cannot access resources outside Analysis Services, these rules are applicable only for assemblies using the **ExternalAccess** and **Unsafe** permission settings.

- If the current execution context corresponds to Windows Authenticated login and is the same as the context of the original caller (that is, there is no EXECUTE AS in the middle), Analysis Services will impersonate the Windows Authenticated login before accessing the resource.
- If there is an intermediate EXECUTE AS that changed the context from that of the original caller), the attempt to access external resource will fail.

The **ImpersonationMode** property can be set to **ImpersonateCurrentUser** or **ImpersonateAnonymous**. The default setting, **ImpersonateCurrentUser**, runs an assembly under the current user's network login account. If the **ImpersonateAnonymous** setting is used, the execution context is corresponds to the Windows login user account IUSER_*servername* on the server. This is the Internet guest account, which has limited privileges on the server. An assembly running in this context can only access limited resources on the local server.

Application Domains

Analysis Services does not expose application domains directly. Because of a set of assemblies running in the same application domain, application domains can discover each other at execution time by using the **System.Reflection** namespace in the .NET Framework or in some other way, and can call into them in late-bound

manner. Such calls will be subject to the permission checks used by Analysis Services authorization-based security.

You should not rely on finding assemblies in the same application domain, because the application domain boundary and the assemblies that go into each domain are defined by the implementation.

See Also

Setting Security for Stored Procedures
Defining Stored Procedures

Data Mining (SSAS)

1/9/2019 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

> **IMPORTANT**
>
> Data mining is deprecated in SQL Server Analysis Services 2017. Documentation is not updated for deprecated features. To learn more, see Analysis Services backward compatibility (SQL 2017).

SQL Server has been a leader in predictive analytics since the 2000 release, by providing data mining in Analysis Services. The combination of Integration Services, Reporting Services, and SQL Server Data Mining provides an integrated platform for predictive analytics that encompasses data cleansing and preparation, machine learning, and reporting. SQL Server Data Mining includes multiple standard algorithms, including EM and K-means clustering models, neural networks, logistic regression and linear regression, decision trees, and naive bayes classifiers. All models have integrated visualizations to help you develop, refine, and evaluate your models. Integrating data mining into business intelligence solution helps you make intelligent decisions about complex problems.

Benefits of Data Mining

Data mining (also called predictive analytics and machine learning) uses well-researched statistical principles to discover patterns in your data. By applying the data mining algorithms in Analysis Services to your data, you can forecast trends, identify patterns, create rules and recommendations, analyze the sequence of events in complex data sets, and gain new insights.

In SQL Server 2017, data mining is powerful, accessible, and integrated with the tools that many people prefer to use for analysis and reporting.

Key Data Mining Features

SQL Server Data Mining provides the following features in support of integrated data mining solutions:

- Multiple data sources: You can use any tabular data source for data mining, including spreadsheets and text files. You can also easily mine OLAP cubes created in Analysis Services. However, you cannot use data from an in-memory database.
- Integrated data cleansing, data management, and reporting: Integration Services provides tools for profiling and cleansing data. You can build ETL processes for cleaning data in preparation for modeling, and ssISnoversion also makes it easy to retrain and update models.
- Multiple customizable algorithms: In addition to providing algorithms such as clustering, neural networks, and decisions trees, SQL Server Data Mining supports development of your own custom plug-in algorithms.
- Model testing infrastructure: Test your models and data sets using important statistical tools as cross-validation, classification matrices, lift charts, and scatter plots. Easily create and manage testing and training sets.
- Querying and drillthrough: SQL Server Data Mining provides the DMX language for integrating prediction queries into applications. You can also retrieve detailed statistics and patterns from the models, and drill

through to case data.

- Client tools: In addition to the development and design studios provided by SQL Server, you can use the Data Mining Add-ins for Excel to create, query, and browse models. Or, create custom clients, including Web services.
- Scripting language support and managed API: All data mining objects are fully programmable. Scripting is possible through MDX, XMLA, or the PowerShell extensions for Analysis Services. Use the Data Mining Extensions (DMX) language for fast querying and scripting.
- Security and deployment: Provides role-based security through Analysis Services, including separate permissions for drillthrough to model and structure data. Easy deployment of models to other servers, so that users can access the patterns or perform predictions

In This Section

The topics in this section introduce the principal features of SQL Server Data Mining and related tasks.

- Data Mining Concepts
- Data Mining Algorithms (Analysis Services - Data Mining)
- Mining Structures (Analysis Services - Data Mining)
- Mining Models (Analysis Services - Data Mining)
- Testing and Validation (Data Mining)
- Data Mining Queries
- Data Mining Solutions
- Data Mining Tools
- Data Mining Architecture
- Security Overview (Data Mining)

See Also

SQL Server R Services

Data Mining Concepts

1/9/2019 • 13 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

> **IMPORTANT**
> Data mining is deprecated in SQL Server Analysis Services 2017. Documentation is not updated for deprecated features. To learn more, see Analysis Services backward compatibility (SQL 2017).

Data mining is the process of discovering actionable information from large sets of data. Data mining uses mathematical analysis to derive patterns and trends that exist in data. Typically, these patterns cannot be discovered by traditional data exploration because the relationships are too complex or because there is too much data.

These patterns and trends can be collected and defined as a *data mining model*. Mining models can be applied to specific scenarios, such as:

- **Forecasting**: Estimating sales, predicting server loads or server downtime
- **Risk and probability**: Choosing the best customers for targeted mailings, determining the probable break-even point for risk scenarios, assigning probabilities to diagnoses or other outcomes
- **Recommendations**: Determining which products are likely to be sold together, generating recommendations
- **Finding sequences**: Analyzing customer selections in a shopping cart, predicting next likely events
- **Grouping**: Separating customers or events into cluster of related items, analyzing and predicting affinities

Building a mining model is part of a larger process that includes everything from asking questions about the data and creating a model to answer those questions, to deploying the model into a working environment. This process can be defined by using the following six basic steps:

1. Defining the Problem
2. Preparing Data
3. Exploring Data
4. Building Models
5. Exploring and Validating Models
6. Deploying and Updating Models

The following diagram describes the relationships between each step in the process, and the technologies in Microsoft SQL Server that you can use to complete each step.

The process illustrated in the diagram is cyclical, meaning that creating a data mining model is a dynamic and iterative process. After you explore the data, you may find that the data is insufficient to create the appropriate mining models, and that you therefore have to look for more data. Alternatively, you may build several models and then realize that the models do not adequately answer the problem you defined, and that you therefore must redefine the problem. You may have to update the models after they have been deployed because more data has become available. Each step in the process might need to be repeated many times in order to create a good model.

Microsoft SQL Server Data Mining provides an integrated environment for creating and working with data mining models. This environment includes SQL Server Development Studio, which contains data mining algorithms and query tools that make it easy to build a comprehensive solution for a variety of projects, and SQL Server Management Studio, which contains tools for browsing models and managing data mining objects. For more information, see Creating Multidimensional Models Using SQL Server Data Tools (SSDT).

For an example of how the SQL Server tools can be applied to a business scenario, see the Basic Data Mining Tutorial.

Defining the Problem

The first step in the data mining process, as highlighted in the following diagram, is to clearly define the problem, and consider ways that data can be utilized to provide an answer to the problem.

This step includes analyzing business requirements, defining the scope of the problem, defining the metrics by which the model will be evaluated, and defining specific objectives for the data mining project. These tasks translate into questions such as the following:

- What are you looking for? What types of relationships are you trying to find?
- Does the problem you are trying to solve reflect the policies or processes of the business?
- Do you want to make predictions from the data mining model, or just look for interesting patterns and associations?

- Which outcome or attribute do you want to try to predict?
- What kind of data do you have and what kind of information is in each column? If there are multiple tables, how are the tables related? Do you need to perform any cleansing, aggregation, or processing to make the data usable?
- How is the data distributed? Is the data seasonal? Does the data accurately represent the processes of the business?

To answer these questions, you might have to conduct a data availability study, to investigate the needs of the business users with regard to the available data. If the data does not support the needs of the users, you might have to redefine the project.

You also need to consider the ways in which the results of the model can be incorporated in key performance indicators (KPI) that are used to measure business progress.

Preparing Data

The second step in the data mining process, as highlighted in the following diagram, is to consolidate and clean the data that was identified in the Defining the Problem step.

Data can be scattered across a company and stored in different formats, or may contain inconsistencies such as incorrect or missing entries. For example, the data might show that a customer bought a product before the product was offered on the market, or that the customer shops regularly at a store located 2,000 miles from her home.

Data cleaning is not just about removing bad data or interpolating missing values, but about finding hidden correlations in the data, identifying sources of data that are the most accurate, and determining which columns are the most appropriate for use in analysis. For example, should you use the shipping date or the order date? Is the best sales influencer the quantity, total price, or a discounted price? Incomplete data, wrong data, and inputs that appear separate but in fact are strongly correlated all can influence the results of the model in ways you do not expect.

Therefore, before you start to build mining models, you should identify these problems and determine how you will fix them. For data mining typically you are working with a very large dataset and cannot examine every transaction for data quality; therefore, you might need to use some form of data profiling and automated data cleansing and filtering tools, such as those supplied in Integration Services, Microsoft SQL Server 2012 Master Data Services, or SQL Server Data Quality Services to explore the data and find the inconsistencies. For more information, see these resources:

- Integration Services in Business Intelligence Development Studio
- Master Data Services Overview (MDS)
- Data Quality Services

It is important to note that the data you use for data mining does not need to be stored in an Online Analytical Processing (OLAP) cube, or even in a relational database, although you can use both of these as data sources. You can conduct data mining using any source of data that has been defined as an Analysis Services data source. These can include text files, Excel workbooks, or data from other external providers. For more information, see Supported Data Sources (SSAS - Multidimensional).

Exploring Data

The third step in the data mining process, as highlighted in the following diagram, is to explore the prepared data.

You must understand the data in order to make appropriate decisions when you create the mining models. Exploration techniques include calculating the minimum and maximum values, calculating mean and standard deviations, and looking at the distribution of the data. For example, you might determine by reviewing the maximum, minimum, and mean values that the data is not representative of your customers or business processes, and that you therefore must obtain more balanced data or review the assumptions that are the basis for your expectations. Standard deviations and other distribution values can provide useful information about the stability and accuracy of the results. A large standard deviation can indicate that adding more data might help you improve the model. Data that strongly deviates from a standard distribution might be skewed, or might represent an accurate picture of a real-life problem, but make it difficult to fit a model to the data.

By exploring the data in light of your own understanding of the business problem, you can decide if the dataset contains flawed data, and then you can devise a strategy for fixing the problems or gain a deeper understanding of the behaviors that are typical of your business.

You can use tools such as Master Data Services to canvass available sources of data and determine their availability for data mining. You can use tools such as SQL Server Data Quality Services, or the Data Profiler in Integration Services, to analyze the distribution of your data and repair issues such as wrong or missing data.

After you have defined your sources, you combine them in a Data Source view by using the Data Source View Designer in SQL Server Data Tools. For more information, see Data Source Views in Multidimensional Models. This designer also contains some several tools that you can use to explore the data and verify that it will work for creating a model. For more information, see Explore Data in a Data Source View (Analysis Services).

Note that when you create a model, Analysis Services automatically creates statistical summaries of the data contained in the model, which you can query to use in reports or further analysis. For more information, see Data Mining Queries.

Building Models

The fourth step in the data mining process, as highlighted in the following diagram, is to build the mining model or models. You will use the knowledge that you gained in the Exploring Data step to help define and create the models.

You define the columns of data that you want to use by creating a mining structure. The mining structure is linked to the source of data, but does not actually contain any data until you process it. When you process the mining structure, Analysis Services generates aggregates and other statistical information that can be used for analysis. This information can be used by any mining model that is based on the structure. For more information about how mining structures are related to mining models, see Logical Architecture (Analysis Services - Data Mining).

Before the structure and model is processed, a data mining model too is just a container that specifies the columns used for input, the attribute that you are predicting, and parameters that tell the algorithm how to process the data. Processing a model is often called *training*. Training refers to the process of applying a specific mathematical algorithm to the data in the structure in order to extract patterns. The patterns that you find in the training process depend on the selection of training data, the algorithm you chose, and how you have configured the algorithm. SQL Server 2017 contains many different algorithms, each suited to a different type of task, and each creating a different type of model. For a list of the algorithms provided in SQL Server 2017, see Data Mining Algorithms (Analysis Services - Data Mining).

You can also use parameters to adjust each algorithm, and you can apply filters to the training data to use just a subset of the data, creating different results. After you pass data through the model, the mining model object contains summaries and patterns that can be queried or used for prediction.

You can define a new model by using the Data Mining Wizard in SQL Server Data Tools, or by using the Data Mining Extensions (DMX) language. For more information about how to use the Data Mining Wizard, see Data Mining Wizard (Analysis Services - Data Mining). For more information about how to use DMX, see Data Mining Extensions (DMX) Reference.

It is important to remember that whenever the data changes, you must update both the mining structure and the mining model. When you update a mining structure by reprocessing it, Analysis Services retrieves data from the source, including any new data if the source is dynamically updated, and repopulates the mining structure. If you have models that are based on the structure, you can choose to update the models that are based on the structure, which means they are retrained on the new data, or you can leave the models as is. For more information, see Processing Requirements and Considerations (Data Mining).

Exploring and Validating Models

The fifth step in the data mining process, as highlighted in the following diagram, is to explore the mining models that you have built and test their effectiveness.

Before you deploy a model into a production environment, you will want to test how well the model performs. Also, when you build a model, you typically create multiple models with different configurations and test all models to see which yields the best results for your problem and your data.

Analysis Services provides tools that help you separate your data into training and testing datasets so that you can accurately assess the performance of all models on the same data. You use the training dataset to build the model, and the testing dataset to test the accuracy of the model by creating prediction queries. This partitioning can be done automatically while building the mining model. For more information, see Testing and Validation (Data Mining).

You can explore the trends and patterns that the algorithms discover by using the viewers in Data Mining Designer in SQL Server Data Tools. For more information, see Data Mining Model Viewers. You can also test how well the models create predictions by using tools in the designer such as the lift chart and classification matrix. To verify whether the model is specific to your data, or may be used to make inferences on the general population, you can use the statistical technique called *cross-validation* to automatically create subsets of the data and test the model against each subset. For more information, see Testing and Validation (Data Mining).

If none of the models that you created in the Building Models step perform well, you might have to return to a previous step in the process and redefine the problem or reinvestigate the data in the original dataset.

Deploying and Updating Models

The last step in the data mining process, as highlighted in the following diagram, is to deploy the models that performed the best to a production environment.

After the mining models exist in a production environment, you can perform many tasks, depending on your needs. The following are some of the tasks you can perform:

- Use the models to create predictions, which you can then use to make business decisions. SQL Server provides the DMX language that you can use to create prediction queries, and Prediction Query Builder to help you build the queries. For more information, see Data Mining Extensions (DMX) Reference.

- Create content queries to retrieve statistics, rules, or formulas from the model. For more information, see Data Mining Queries.
- Embed data mining functionality directly into an application. You can include Analysis Management Objects (AMO), which contains a set of objects that your application can use to create, alter, process, and delete mining structures and mining models. Alternatively, you can send XML for Analysis (XMLA) messages directly to an instance of Analysis Services. For more information, see Development (Analysis Services - Data Mining).
- Use Integration Services to create a package in which a mining model is used to intelligently separate incoming data into multiple tables. For example, if a database is continually updated with potential customers, you could use a mining model together with Integration Services to split the incoming data into customers who are likely to purchase a product and customers who are likely to not purchase a product. For more information, see Typical Uses of Integration Services.
- Create a report that lets users directly query against an existing mining model. For more information, see Reporting Services in SQL Server Data Tools (SSDT).
- Update the models after review and analysis. Any update requires that you reprocess the models. For more information, see Processing Data Mining Objects.
- Update the models dynamically, as more data comes into the organization, and making constant changes to improve the effectiveness of the solution should be part of the deployment strategy. For more information, see Management of Data Mining Solutions and Objects

See Also

Data Mining Solutions
Data Mining Tools

Data Mining Algorithms (Analysis Services - Data Mining)

5/16/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

An *algorithm* in data mining (or machine learning) is a set of heuristics and calculations that creates a model from data. To create a model, the algorithm first analyzes the data you provide, looking for specific types of patterns or trends. The algorithm uses the results of this analysis over many iterations to find the optimal parameters for creating the mining model. These parameters are then applied across the entire data set to extract actionable patterns and detailed statistics.

The mining model that an algorithm creates from your data can take various forms, including:

- A set of clusters that describe how the cases in a dataset are related.
- A decision tree that predicts an outcome, and describes how different criteria affect that outcome.
- A mathematical model that forecasts sales.
- A set of rules that describe how products are grouped together in a transaction, and the probabilities that products are purchased together.

The algorithms provided in SQL Server Data Mining are the most popular, well-researched methods of deriving patterns from data. To take one example, K-means clustering is one of the oldest clustering algorithms and is available widely in many different tools and with many different implementations and options. However, the particular implementation of K-means clustering used in SQL Server Data Mining was developed by Microsoft Research and then optimized for performance with Analysis Services. All of the Microsoft data mining algorithms can be extensively customized and are fully programmable, using the provided APIs. You can also automate the creation, training, and retraining of models by using the data mining components in Integration Services.

You can also use third-party algorithms that comply with the OLE DB for Data Mining specification, or develop custom algorithms that can be registered as services and then used within the SQL Server Data Mining framework.

Choosing the Right Algorithm

Choosing the best algorithm to use for a specific analytical task can be a challenge. While you can use different algorithms to perform the same business task, each algorithm produces a different result, and some algorithms can produce more than one type of result. For example, you can use the Microsoft Decision Trees algorithm not only for prediction, but also as a way to reduce the number of columns in a dataset, because the decision tree can identify columns that do not affect the final mining model.

Choosing an Algorithm by Type

SQL Server Data Mining includes the following algorithm types:

- **Classification algorithms** predict one or more discrete variables, based on the other attributes in the dataset.
- **Regression algorithms** predict one or more continuous numeric variables, such as profit or loss, based on other attributes in the dataset.

- **Segmentation algorithms** divide data into groups, or clusters, of items that have similar properties.
- **Association algorithms** find correlations between different attributes in a dataset. The most common application of this kind of algorithm is for creating association rules, which can be used in a market basket analysis.
- **Sequence analysis algorithms** summarize frequent sequences or episodes in data, such as a series of clicks in a web site, or a series of log events preceding machine maintenance.

However, there is no reason that you should be limited to one algorithm in your solutions. Experienced analysts will sometimes use one algorithm to determine the most effective inputs (that is, variables), and then apply a different algorithm to predict a specific outcome based on that data. SQL Server Data Mining lets you build multiple models on a single mining structure, so within a single data mining solution you could use a clustering algorithm, a decision trees model, and a Naïve Bayes model to get different views on your data. You might also use multiple algorithms within a single solution to perform separate tasks: for example, you could use regression to obtain financial forecasts, and use a neural network algorithm to perform an analysis of factors that influence forecasts.

Choosing an Algorithm by Task

To help you select an algorithm for use with a specific task, the following table provides suggestions for the types of tasks for which each algorithm is traditionally used.

EXAMPLES OF TASKS	MICROSOFT ALGORITHMS TO USE
Predicting a discrete attribute: Flag the customers in a prospective buyers list as good or poor prospects. Calculate the probability that a server will fail within the next 6 months. Categorize patient outcomes and explore related factors.	Microsoft Decision Trees Algorithm Microsoft Naive Bayes Algorithm Microsoft Clustering Algorithm Microsoft Neural Network Algorithm
Predicting a continuous attribute: Forecast next year's sales. Predict site visitors given past historical and seasonal trends. Generate a risk score given demographics.	Microsoft Decision Trees Algorithm Microsoft Time Series Algorithm Microsoft Linear Regression Algorithm
Predicting a sequence: Perform clickstream analysis of a company's Web site. Analyze the factors leading to server failure. Capture and analyze sequences of activities during outpatient visits, to formulate best practices around common activities.	Microsoft Sequence Clustering Algorithm

EXAMPLES OF TASKS	MICROSOFT ALGORITHMS TO USE
Finding groups of common items in transactions: Use market basket analysis to determine product placement. Suggest additional products to a customer for purchase. Analyze survey data from visitors to an event, to find which activities or booths were correlated, to plan future activities.	Microsoft Association Algorithm Microsoft Decision Trees Algorithm
Finding groups of similar items: Create patient risk profiles groups based on attributes such as demographics and behaviors. Analyze users by browsing and buying patterns. Identify servers that have similar usage characteristics.	Microsoft Clustering Algorithm Microsoft Sequence Clustering Algorithm

Related Content

The following table provides links to learning resources for each of the data mining algorithms that are provided in SQL Server Data Mining:

Basic algorithm description	Explains what the algorithm does and how it works, and outlines possible business scenarios where the algorithm might be useful.
	Microsoft Association Algorithm Microsoft Clustering Algorithm Microsoft Decision Trees Algorithm Microsoft Linear Regression Algorithm Microsoft Logistic Regression Algorithm Microsoft Naive Bayes Algorithm Microsoft Neural Network Algorithm Microsoft Sequence Clustering Algorithm Microsoft Time Series Algorithm
Technical reference	Provides technical detail about the implementation of the algorithm, with academic references as necessary. Lists the parameters that you can set to control the behavior of the algorithm and customize the results in the model. Describes data requirements and provides performance tips if possible.

	Microsoft Association Algorithm Technical Reference Microsoft Clustering Algorithm Technical Reference Microsoft Decision Trees Algorithm Technical Reference Microsoft Linear Regression Algorithm Technical Reference Microsoft Logistic Regression Algorithm Technical Reference Microsoft Naive Bayes Algorithm Technical Reference Microsoft Neural Network Algorithm Technical Reference Microsoft Sequence Clustering Algorithm Technical Reference Microsoft Time Series Algorithm Technical Reference
Model content	Explains how information is structured within each type of data mining model, and explains how to interpret the information stored in each of the nodes.
	Mining Model Content for Association Models (Analysis Services - Data Mining) Mining Model Content for Clustering Models (Analysis Services - Data Mining) Mining Model Content for Decision Tree Models (Analysis Services - Data Mining) Mining Model Content for Linear Regression Models (Analysis Services - Data Mining) Mining Model Content for Logistic Regression Models (Analysis Services - Data Mining) Mining Model Content for Naive Bayes Models (Analysis Services - Data Mining) Mining Model Content for Neural Network Models (Analysis Services - Data Mining) Mining Model Content for Sequence Clustering Models (Analysis Services - Data Mining) Mining Model Content for Time Series Models (Analysis Services - Data Mining)
Data mining queries	Provides multiple queries that you can use with each model type. Examples include content queries that let you learn more about the patterns in the model, and prediction queries to help you build predictions based on those patterns.

	Association Model Query Examples Clustering Model Query Examples Decision Trees Model Query Examples Linear Regression Model Query Examples Logistic Regression Model Query Examples Naive Bayes Model Query Examples Neural Network Model Query Examples Sequence Clustering Model Query Examples Time Series Model Query Examples

Related Tasks

TOPIC	DESCRIPTION
Determine the algorithm used by a data mining model	Query the Parameters Used to Create a Mining Model
Create a Custom Plug-In Algorithm	Plugin Algorithms
Explore a model using an algorithm-specific viewer	Data Mining Model Viewers
View the content of a model using a generic table format	Browse a Model Using the Microsoft Generic Content Tree Viewer
Learn about how to set up your data and use algorithms to create models	Mining Structures (Analysis Services - Data Mining) Mining Models (Analysis Services - Data Mining)

See Also

Data Mining Tools

Microsoft Association Algorithm

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Association algorithm is an algorithm that is often used for recommendation engines. A recommendation engine recommends items to customers based on items they have already bought, or in which they have indicated an interest. The Microsoft Association algorithm is also useful for market basket analysis.

Association models are built on datasets that contain identifiers both for individual cases and for the items that the cases contain. A group of items in a case is called an *itemset*. An association model consists of a series of itemsets and the rules that describe how those items are grouped together within the cases. The rules that the algorithm identifies can be used to predict a customer's likely future purchases, based on the items that already exist in the customer's shopping cart. The following diagram shows a series of rules in an itemset.

Rule
Road Bottle Cage = Existing, Cycling Cap = Existing -> Water Bottle = Existing
Mountain-200 = Existing, Mountain Tire Tube = Existing -> HL Mountain Tire = Existing
Mountain-200 = Existing, Water Bottle = Existing -> Mountain Bottle Cage = Existing
Touring-1000 = Existing, Water Bottle = Existing -> Road Bottle Cage = Existing
Road-750 = Existing, Water Bottle = Existing -> Road Bottle Cage = Existing
Touring Tire = Existing, Sport-100 = Existing -> Touring Tire Tube = Existing

As the diagram illustrates, the Microsoft Association algorithm can potentially find many rules within a dataset. The algorithm uses two parameters, support and probability, to describe the itemsets and rules that it generates. For example, if X and Y represent two items that could be in a shopping cart, the support parameter is the number of cases in the dataset that contain the combination of items, X and Y. By using the support parameter in combination with the user-defined parameters, *MINIMUM_SUPPORT* and *MAXIMUM_SUPPORT*, the algorithm controls the number of itemsets that are generated. The probability parameter, also named *confidence*, represents the fraction of cases in the dataset that contain X and that also contain Y. By using the probability parameter in combination with the *MINIMUM_PROBABILITY* parameter, the algorithm controls the number of rules that are generated.

Example

The Adventure Works Cycle company is redesigning the functionality of its Web site. The goal of the redesign is to increase sell-through of products. Because the company records each sale in a transactional database, they can use the Microsoft Association algorithm to identify sets of products that tend to be purchased together. They can then predict additional items that a customer might be interested in, based on items that are already in the customer's shopping basket.

How the Algorithm Works

The Microsoft Association algorithm traverses a dataset to find items that appear together in a case. The algorithm then groups into itemsets any associated items that appear, at a minimum, in the number of cases that are specified by the *MINIMUM_SUPPORT* parameter. For example, an itemset could be "Mountain 200=Existing, Sport 100=Existing", and could have a support of 710. The algorithm then generates rules from the itemsets. These rules are used to predict the presence of an item in the database, based on the presence of other specific items that the algorithm identifies as important. For example, a rule could be "if Touring 1000=existing and Road bottle cage=existing, then Water bottle=existing", and could have a probability of 0.812. In this example, the algorithm identifies that the presence in the basket of the Touring 1000 tire and the water bottle cage predicts that a water bottle would also likely be in the basket.

For a more detailed explanation of the algorithm, together with a list of parameters for customizing the behavior of the algorithm and controlling the results in the mining model, see Microsoft Association Algorithm Technical Reference.

Data Required for Association Models

When you prepare data for use in an association rules model, you should understand the requirements for the particular algorithm, including how much data is needed, and how the data is used.

The requirements for an association rules model are as follows:

- **A single key column** Each model must contain one numeric or text column that uniquely identifies each record. compound keys not permitted.
- **A single predictable column** An association model can have only one predictable column. Typically it is the key column of the nested table, such as the filed that lists the products that were purchased. The values must be discrete or discretized.
- **Input columns** . The input columns must be discrete. The input data for an association model often is contained in two tables. For example, one table might contain customer information while another table contains customer purchases. You can input this data into the model by using a nested table. For more information about nested tables, see Nested Tables (Analysis Services - Data Mining).

For more detailed information about the content types and data types supported for association models, see the Requirements section of Microsoft Association Algorithm Technical Reference.

Viewing an Association Model

To explore the model, you can use the **Microsoft Association Viewer**. When you view an association model, Analysis Services presents the correlations from different angles so that you can better understand the relationships and rules that were found in the data. The **Itemset** pane in the viewer provides a detailed breakdown of the most common combinations, or itemsets. The **Rules** pane presents a list of rules that have been generalized from the data, adds calculations of probability, and ranks the rules by relative importance. the dependency network viewer lets you visually explore how individual different items are connected. For more information, see Browse a Model Using the Microsoft Cluster Viewer.

If you want to find out more detail about any of the itemsets and rules, you can browse the model in the Microsoft Generic Content Tree Viewer. The content stored for the model includes the support for each itemset, a score for each rule, and other statistics. For more information, see Mining Model Content for Association Models (Analysis Services - Data Mining).

Creating Predictions

After the model has been processed, you can use the rules and itemsets to make predictions. In an association model, a prediction tells you what item is likely to occur given the presence of the specified item, and the prediction can include such information as the probability, the support, or the importance. For examples of how to create queries against an association model, see Association Model Query Examples.

For general information about how to create a query against a data mining model, see Data Mining Queries.

Performance

The process of creating itemsets and counting correlations can be time-consuming. Although the Microsoft Association Rules algorithm uses optimization techniques to save space and make processing faster, you should know that performance issues can occur under conditions such as the following:

- Data set is large with many individual items.
- Minimum itemset size is set too low.

To minimize processing time and reduce the complexity of the itemsets, you might try grouping related items by categories before you analyze the data.

Remarks

- Does not support the use of Predictive Model Markup Language (PMML) to create mining models.
- Supports drillthrough.
- Supports the use of OLAP mining models.
- Supports the creation of data mining dimensions.

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Browse a Model Using the Microsoft Association Rules Viewer
Mining Model Content for Association Models (Analysis Services - Data Mining)
Microsoft Association Algorithm Technical Reference
Association Model Query Examples

Microsoft Association Algorithm Technical Reference

11/28/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Association Rules algorithm is a straightforward implementation of the well-known Apriori algorithm.

Both the Microsoft Decision Trees algorithm and the Microsoft Association Rules algorithm can be used to analyze associations, but the rules that are found by each algorithm can differ. In a decision trees model, the splits that lead to specific rules are based on information gain, whereas in an association model, rules are based completely on confidence. Therefore, in an association model, a strong rule, or one that has high confidence, might not necessarily be interesting because it does not provide new information.

Implementation of the Microsoft Association Algorithm

The Apriori algorithm does not analyze patterns, but rather generates and then counts *candidate itemsets*. An item can represent an event, a product, or the value of an attribute, depending on the type of data that is being analyzed.

In the most common type of association model Boolean variables, representing a Yes/No or Missing/Existing value, are assigned to each attribute, such as a product or event name. A market basket analysis is an example of an association rules model that uses Boolean variables to represent the presence or absence of particular products in a customer's shopping basket.

For each itemset, the algorithm then creates scores that represent support and confidence. These scores can be used to rank and derive interesting rules from the itemsets.

Association models can also be created for numerical attributes. If the attributes are continuous, the numbers can be *discretized, or* grouped in buckets. The discretized values can then be handled either as Booleans or as attribute-value pairs.

Support, Probability, and Importance

Support, which issometimes referred to as *frequency*, means the number of cases that contain the targeted item or combination of items. Only items that have at least the specified amount of support can be included in the model.

A *frequent itemset* refers to a collection of items where the combination of items also has support above the threshold defined by the MINIMUM_SUPPORT parameter. For example, if the itemset is {A,B,C} and the MINIMUM_SUPPORT value is 10, each individual item A, B, and C must be found in at least 10 cases to be included in the model, and the combination of items {A,B,C} must also be found in at least 10 cases.

Note You can also control the number of itemsets in a mining model by specifying the maximum length of an itemset, where length means the number of items.

By default, the support for any particular item or itemset represents a count of the cases that contain that item or items. However, you can also express MINIMUM_SUPPORT as a percentage of the total cases in the data set, by typing the number as a decimal value less than 1. For example, if you specify a MINIMUM_SUPPORT value of 0.03, it means that at least 3% of the total cases in the data set must contain this item or itemset for inclusion in the model. You should experiment with your model to determine whether using a count or percentage makes more sense.

In contrast, the threshold for rules is expressed not as a count or percentage, but as a probability, sometimes

referred to as *confidence*. For example, if the itemset {A,B,C} occurs in 50 cases, but the itemset {A,B,D} also occurs in 50 cases, and the itemset {A,B} in another 50 cases, it is obvious that {A,B} is not a strong predictor of {C}. Therefore, to weight a particular outcomes against all known outcomes, Analysis Services calculates the probability of the individual rule (such as If {A,B} Then {C}) by dividing the support for the itemset {A,B,C} by the support for all related itemsets.

You can restrict the number of rules that a model produces by setting a value for MINIMUM_PROBABILITY.

For each rule that is created, Analysis Services outputs a score that indicates its *importance*, which is alsoreferred to as *lift*. Lift Importance is calculated differently for itemsets and rules.

The importance of an itemset is calculated as the probability of the itemset divided by the compound probability of the individual items in the itemset. For example, if an itemset contains {A,B}, Analysis Services first counts all the cases that contain this combination A and B, and divides that by the total number of cases, and then normalizes the probability.

The importance of a rule is calculated by the log likelihood of the right-hand side of the rule, given the left-hand side of the rule. For example, in the rule `If {A} Then {B}`, Analysis Services calculates the ratio of cases with A and B over cases with B but without A, and then normalizes that ratio by using a logarithmic scale.

Feature Selection

The Microsoft Association Rules algorithm does not perform any kind of automatic feature selection. Instead, the algorithm provides parameters that control the data that is used by the algorithm. This might include limits on the size of each itemset, or setting the maximum and minimum support required to add an itemset to the model.

- To filter out items and events that are too common and therefore uninteresting, decrease the value of MAXIMUM_SUPPORT to remove very frequent itemsets from the model.
- To filter out items and itemsets that are rare, increase the value of MINIMUM_SUPPORT.
- To filter out rules, increase the value of MINIMUM_PROBABILITY.

Customizing the Microsoft Association Rules Algorithm

The Microsoft Association Rules algorithm supports several parameters that affect the behavior, performance, and accuracy of the resulting mining model.

Setting Algorithm Parameters

You can change the parameters for a mining model at any time by using the Data Mining Designer in SQL Server Data Tools (SSDT). You can also change parameters programmatically by using the AlgorithmParameters collection in AMO, or by using the MiningModels Element (ASSL) in XMLA. The following table describes each parameter.

> **NOTE**
> You cannot change the parameters in an existing model by using a DMX statement; you must specify the parameters in the DMX CREATE MODEL or ALTER STRUCTURE... ADD MODEL when you create the model.

MAXIMUM_ITEMSET_COUNT

Specifies the maximum number of itemsets to produce. If no number is specified, the default value is used.

The default is 200000.

> **NOTE**
> Itemsets are ranked by support. Among itemsets that have the same support, ordering is arbitrary.

MAXIMUM_ITEMSET_SIZE
Specifies the maximum number of items that are allowed in an itemset. Setting this value to 0 specifies that there is no limit to the size of the itemset.

The default is 3.

> **NOTE**
> Decreasing this value can potentially reduce the time that is required for creating the model, because processing of the model stops when the limit is reached.

MAXIMUM_SUPPORT
Specifies the maximum number of cases that an itemset has for support. This parameter can be used to eliminate items that appear frequently and therefore potentially have little meaning.

If this value is less than 1, the value represents a percentage of the total cases. Values greater than 1 represent the absolute number of cases that can contain the itemset.

The default is 1.

MINIMUM_ITEMSET_SIZE
Specifies the minimum number of items that are allowed in an itemset. If you increase this number, the model might contain fewer itemsets. This can be useful if you want to ignore single-item itemsets, for example.

The default is 1.

> **NOTE**
> You cannot reduce model processing time by increasing the minimum value, because Analysis Services must calculate probabilities for single items anyway as part of processing. However, by setting this value higher you can filter out smaller itemsets.

MINIMUM_PROBABILITY
Specifies the minimum probability that a rule is true.

For example, if you set this value to 0.5, it means that no rule with less than fifty percent probability can be generated.

The default is 0.4.

MINIMUM_SUPPORT
Specifies the minimum number of cases that must contain the itemset before the algorithm generates a rule.

If you set this value to less than 1, the minimum number of cases is calculated as a percentage of the total cases.

If you set this value to a whole number greater than 1, specifies the minimum number of cases is calculated as a count of cases that must contain the itemset. The algorithm might automatically increase the value of this parameter if memory is limited.

The default is 0.03. This means that to be included in the model, an itemset must be found in at least 3% of cases.

OPTIMIZED_PREDICTION_COUNT
Defines the number of items to be cached for optimizing prediction.

The default value is 0. When the default is used, the algorithm will produce as many predictions as requested in the query.

If you specify a nonzero value for *OPTIMIZED_PREDICTION_COUNT*, prediction queries can return at most the

specified number of items, even if you request additional predictions. However, setting a value can improve prediction performance.

For example, if the value is set to 3, the algorithm caches only 3 items for prediction. You cannot see additional predictions that might be equally probable to the 3 items that are returned.

Modeling Flags

The following modeling flags are supported for use with the Microsoft Association Rules algorithm.

NOT NULL
Indicates that the column cannot contain a null. An error will result if Analysis Services encounters a null during model training.

Applies to the mining structure column.

MODEL_EXISTENCE_ONLY
Means that the column will be treated as having two possible states: **Missing** and **Existing**. A null is a missing value.

Applies to the mining model column.

Requirements

An association model must contain a key column, input columns, and a single predictable column.

Input and Predictable Columns

The Microsoft Association Rules algorithm supports the specific input columns and predictable columns that are listed in the following table. For more information about the meaning of content types in a mining model, see Content Types (Data Mining).

COLUMN	CONTENT TYPES
Input attribute	Cyclical, Discrete, Discretized, Key, Table, Ordered
Predictable attribute	Cyclical, Discrete, Discretized, Table, Ordered

> **NOTE**
> Cyclical and Ordered content types are supported, but the algorithm treats them as discrete values and does not perform special processing.

See Also

Microsoft Association Algorithm
Association Model Query Examples
Mining Model Content for Association Models (Analysis Services - Data Mining)

Mining Model Content for Association Models (Analysis Services - Data Mining)

5/16/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes mining model content that is specific to models that use the Microsoft Association Rules algorithm. For an explanation of general and statistical terminology related to mining model content that applies to all model types, see Mining Model Content (Analysis Services - Data Mining).

Understanding the Structure of an Association Model

An association model has a simple structure. Each model has a single parent node that represents the model and its metadata, and each parent node has a flat list of itemsets and rules. The itemsets and rules are not organized in trees, they are ordered with itemsets first and rules next as shown in the following diagram.

Each itemset is contained in its own node (NODE_TYPE = 7). The *node* includes the definition of the itemset, the number of cases that contain this itemset, and other information.

Each rule is also contained in its own node (NODE_TYPE = 8). A *rule* describes a general pattern for how items are associated. A rule is like an IF-THEN statement. The left-hand side of the rule shows an existing condition or set of conditions. The right-hand side of the rule shows the item in your data set that is usually associated with the conditions on the left side.

Note If you want to extract either the rules or the itemsets, you can use a query to return only the node types that you want. For more information, see Association Model Query Examples.

Model Content for an Association Model

This section provides detail and examples only for those columns in the mining model content that are relevant for association models.

For information about the general-purpose columns in the schema rowset, such as MODEL_CATALOG and MODEL_NAME, see Mining Model Content (Analysis Services - Data Mining).

MODEL_CATALOG
Name of the database where the model is stored.

MODEL_NAME
Name of the model.

ATTRIBUTE_NAME
The names of the attributes that correspond to this node.

NODE_NAME

The name of the node. For an association model, this column contains the same value as NODE_UNIQUE_NAME.

NODE_UNIQUE_NAME
The unique name of the node.

NODE_TYPE
A association model outputs only the following node types:

NODE TYPE ID	TYPE
1 (Model)	Root or parent node.
7 (Itemset)	An itemset, or collection of attribute-value pairs. Examples: `Product 1 = Existing, Product 2 = Existing` or `Gender = Male`.
8 (Rule)	A rule defining how items relate to each other. Example: `Product 1 = Existing, Product 2 = Existing -> Product 3 = Existing` .

NODE_CAPTION
A label or a caption associated with the node.

Itemset node A comma-separated list of items.

Rule node Contains the left and right-hand sides of the rule.

CHILDREN_CARDINALITY
Indicates the number of children of the current node.

Parent node Indicates the total number of itemsets plus rules.

> **NOTE**
> To get a breakdown of the count for itemsets and rules, see the NODE_DESCRIPTION for the root node of the model.

Itemset or rule node Always 0.

PARENT_UNIQUE_NAME
The unique name of the node's parent.

Parent node Always NULL.

Itemset or rule node Always 0.

NODE_DESCRIPTION
A user-friendly description of the contents of the node.

Parent node Includes a comma-separated list of the following information about the model:

ITEM	DESCRIPTION
ITEMSET_COUNT	Count of all itemsets in model.
RULE_COUNT	Count of all rules in model.
MIN_SUPPORT	The minimum support found for any single itemset. **Note** This value might differ from the value that you set for the *MINIMUM _SUPPORT* parameter.
MAX_SUPPORT	The maximum support found for any single itemset. **Note** This value might differ from the value that you set for the *MAXIMUM_SUPPORT* parameter.
MIN_ITEMSET_SIZE	The size of the smallest itemset, represented as a count of items. A value of 0 indicates that the **Missing** state was treated as an independent item. **Note** The default value of the *MINIMUM_ITEMSET_SIZE* parameter is 1.
MAX_ITEMSET_SIZE	Indicates the size of the largest itemset that was found. **Note** This value is constrained by the value that you set for the *MAX_ITEMSET_SIZE* parameter when you created the model. This value can never exceed that value; however, it can be less. The default value is 3.
MIN_PROBABILITY	The minimum probability detected for any single itemset or rule in the model. Example: 0.400390625 **Note** For itemsets, this value is always greater than the value that you set for the *MINIMUM_PROBABILITY* parameter when you created the model.
MAX_PROBABILITY	The maximum probability detected for any single itemset or rule in the model. Example: 1 **Note** There is no parameter to constrain maximum probability of itemsets. If you want to eliminate items that are too frequent, use the *MAXIMUM_SUPPORT* parameter instead.
MIN_LIFT	The minimum amount of lift that is provided by the model for any itemset. Example: 0.14309369632511 Note: Knowing the minimum lift can help you determine whether the lift for any one itemset is significant.

ITEM	DESCRIPTION
MAX_LIFT	The maximum amount of lift that is provided by the model for any itemset. Example: 1.95758227647523 **Note** Knowing the maximum lift can help you determine whether the lift for any one itemset is significant.

Itemset node Itemset nodes contain a list of the items, displayed as a comma-separated text string.

Example:

```
Touring Tire = Existing, Water Bottle = Existing
```

This means touring tires and water bottles were purchased together.

Rule node Rule nodes contains a left-hand and right-hand side of the rule, separated by an arrow.

Example: `Touring Tire = Existing, Water Bottle = Existing -> Cycling cap = Existing`

This means that if someone bought a touring tire and a water bottle, they were also likely to buy a cycling cap.

NODE_RULE
An XML fragment that describes the rule or itemset that is embedded in the node.

Parent node Blank.

Itemset node Blank.

Rule node The XML fragment includes additional useful information about the rule, such as support, confidence, and the number of items, and the ID of the node that represents the left-hand side of the rule.

MARGINAL_RULE
Blank.

NODE_PROBABILITY
A probability or confidence score associated with the itemset or rule.

Parent node Always 0.

Itemset node Probability of the itemset.

Rule node Confidence value for the rule.

MARGINAL_PROBABILITY
Same as NODE_PROBABILITY.

NODE_DISTRIBUTION
The table contains very different information, depending on whether the node is an itemset or a rule.

Parent node Blank.

Itemset node Lists each item in the itemset together with a probability and support value. For example, if the itemset contains two products, the name of each product is listed, together with the count of cases that include each product.

Rule node Contains two rows. The first row shows the attribute from the right-hand side of the rule, which is the predicted item, together with a confidence score.

The second row is unique to association models; it contains a pointer to the itemset on the right-hand side of the rule. The pointer is represented in the ATTRIBUTE_VALUE column as the ID of the itemset that contains only the

right-hand item.

For example, if the rule is `If {A,B} Then {C}`, the table contains the name of the item `{C}`, and the ID of the node that contains the itemset for item C.

This pointer is useful because you can determine from the itemset node how many cases in all include the right-hand side product. The cases that are subject to the rule `If {A,B} Then {C}` are a subset of the cases listed in the itemset for `{C}`.

NODE_SUPPORT
The number of cases that support this node.

Parent node Number of cases in the model.

Itemset node Number of cases that contains all items in the itemset.

Rule node The number of cases that contain all items included in the rule.

MSOLAP_MODEL_COLUMN
Contains different information depending on whether the node is an itemset or rule.

Parent node Blank.

Itemset node Blank.

Rule node The ID of the itemset that contains the items in the left-hand side of the rule. For example, if the rule is `If {A,B} Then {C}`, this column contains the ID of the itemset that contains only `{A,B}`.

MSOLAP_NODE_SCORE
Parent node Blank.

Itemset node Importance score for the itemset.

Rule node Importance score for the rule.

> **NOTE**
> Importance is calculated differently for itemsets and rules. For more information, see Microsoft Association Algorithm Technical Reference.

MSOLAP_NODE_SHORT_CAPTION
Blank.

See Also

Mining Model Content (Analysis Services - Data Mining)
Microsoft Association Algorithm
Association Model Query Examples

Association Model Query Examples

11/28/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a query against a data mining model, you can create either a content query, which provides details about the rules and itemsets discovered during analysis, or you can create a prediction query, which uses the associations discovered in the data to make predictions. For an association model, predictions typically are based on rules, and can be used to make recommendations, whereas queries on content typically explore the relationship among itemsets. You can also retrieve metadata about the model.

This section explains how to create these kinds of queries for models that are based on the Microsoft Association Rules algorithm.

Content Queries

Getting model metadata data by using DMX

Getting metadata from the schema rowset

Retrieving the original parameters for the model

Retrieving a list of itemsets and products

Returning the top 10 itemsets

Prediction Queries

Predicting associated items

Determining confidence for related itemsets

Finding Information about the Model

All mining models expose the content learned by the algorithm according to a standardized schema, which is named the mining model schema rowset. You can create queries against the mining model schema rowset either by using Data Mining Extensions (DMX) statements, or by using Analysis Services stored procedures. In SQL Server 2017, you can also query the schema rowsets directly as system tables, by using a SQL-like syntax.

Sample Query 1: Getting Model Metadata by Using DMX

The following query returns basic metadata about the association model, `Association`, such as the name of the model, the database where the model is stored, and the number of child nodes in the model. This query uses a DMX content query to retrieve the metadata from the parent node of the model:

```
SELECT MODEL_CATALOG, MODEL_NAME, NODE_CAPTION,
NODE_SUPPORT, [CHILDREN_CARDINALITY], NODE_DESCRIPTION
FROM Association.CONTENT
WHERE NODE_TYPE = 1
```

> **NOTE**
>
> You must enclose the name of the column, CHILDREN_CARDINALITY, in brackets to distinguish it from the MDX reserved keyword of the same name.

Example results:

MODEL_CATALOG	Association Test
MODEL_NAME	Association
NODE_CAPTION	Association Rules Model
NODE_SUPPORT	14879
CHILDREN_CARDINALITY	942
NODE_DESCRIPTION	Association Rules Model; ITEMSET_COUNT=679; RULE_COUNT=263; MIN_SUPPORT=14; MAX_SUPPORT=4334; MIN_ITEMSET_SIZE=0; MAX_ITEMSET_SIZE=3; MIN_PROBABILITY=0.400390625; MAX_PROBABILITY=1; MIN_LIFT=0.14309369632511; MAX_LIFT=1.95758227647523

For a definition of what these columns mean in an association model, see Mining Model Content for Association Models (Analysis Services - Data Mining).

Return to Top

Sample Query 2: Getting Additional Metadata from the Schema Rowset

By querying the data mining schema rowset, you can find the same information that is returned in a DMX content query. However, the schema rowset provides some additional columns, such as the date the model was last processed, the mining structure, and the name of the column used as the predictable attribute.

```
SELECT MODEL_CATALOG, MODEL_NAME, SERVICE_NAME, PREDICTION_ENTITY,
MINING_STRUCTURE, LAST_PROCESSED
FROM $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = 'Association'
```

Example results:

MODEL_CATALOG	Adventure Works DW Multidimensional 2012
MODEL_NAME	Association
SERVICE_NAME	Association Rules Model
PREDICTION_ENTITY	v Assoc Seq Line Items
MINING_STRUCTURE	Association
LAST_PROCESSED	9/29/2007 10:21:24 PM

Return to Top

Sample Query 3: Retrieving Original Parameters for Model

The following query returns a single column that contains details about the parameter settings that were used

when the model was created.

```
SELECT MINING_PARAMETERS
from $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = 'Association'
```

Example results:

MAXIMUM_ITEMSET_COUNT=200000,MAXIMUM_ITEMSET_SIZE=3,MAXIMUM_SUPPORT=1,MINIMUM_SUPPORT=9.40923449156529E-04,MINIMUM_IMPORTANCE=-999999999,MINIMUM_ITEMSET_SIZE=0,MINIMUM_PROBABILITY=0.4

Return to Top

Finding Information about Rules and Itemsets

There are two common uses of an association model: to discover information about frequent itemsets, and to extract details about particular rules and itemsets. For example, you might want to extract a list of rules that were scored as being especially interesting, or create a list of the most common itemsets. You retrieve such information by using a DMX content query. You can also browse this information by using the **Microsoft Association Viewer**.

Sample Query 4: Retrieving List of Itemsets and Products

The following query retrieves all of the itemsets, together with a nested table that lists the products included in each itemset. The NODE_NAME column contains the unique ID of the itemset within the model, whereas the NODE_CAPTION provides a text description of the items. In this example, the nested table is flattened, so that an itemset that contains two products generates two rows in the results. You can omit the FLATTENED keyword if your client supports hierarchical data.

```
SELECT FLATTENED NODE_NAME, NODE_CAPTION,
NODE_PROBABILITY, NODE_SUPPORT,
(SELECT ATTRIBUTE_NAME FROM NODE_DISTRIBUTION) as PurchasedProducts
FROM Association.CONTENT
WHERE NODE_TYPE = 7
```

Example results:

NODE_NAME	37
NODE_CAPTION	Sport-100 = Existing
NODE_PROBABILITY	0.291283016331743
NODE_SUPPORT	4334
PURCHASEDPRODUCTS.ATTRIBUTE_NAME	v Assoc Seq Line Items(Sport-100)

Return to Top

Sample Query 5: Returning Top 10 Itemsets

This example demonstrates how to use some of the grouping and ordering functions that DMX provides by default. The query returns the top 10 itemsets when ordered by the support for each node. Note that you do not need to explicitly group the results, as you would in Transact-SQL; however, you can use only one aggregate

function in each query.

```
SELECT TOP 10 (NODE_SUPPORT),NODE_NAME, NODE_CAPTION
FROM Association.CONTENT
WHERE NODE_TYPE = 7
```

Example results:

NODE_SUPPORT	4334
NODE_NAME	37
NODE_CAPTION	Sport-100 = Existing

Return to Top

Making Predictions using the Model

An association rules model is often used to generate recommendations, which are based on correlations discovered in the itemsets. Therefore, when you create a prediction query based on an association rules model, you are typically using the rules in the model to make guesses based on new data. PredictAssociation (DMX) is the function that returns recommendations, and has several arguments that you can use to customize the query results.

Another example of where queries on an association model might be useful is to return the confidence for various rules and itemsets so that you can compare the effectiveness of different cross-sell strategies. The following examples illustrate how to create such queries.

Sample Query 6: Predicting Associated Items

This example uses the Association model created in the Intermediate Data Mining Tutorial (Analysis Services - Data Mining). It demonstrates how to create a prediction query that tells you what products to recommend to a customer who has purchased a particular product. This type of query, where you provide values to the model in a **SELECT...UNION** statement, is called a singleton query. Because the predictable model column that corresponds to the new values is a nested table, you must use one **SELECT** clause to map the new value to the nested table column, `[Model]`, and another **SELECT** clause to map the nested table column to the case-level column, `[v Assoc Seq Line Items]`. Adding the keyword INCLUDE-STATISTICS to the query lets you see the probability and support for the recommendations.

```
SELECT PredictAssociation([Association].[vAssocSeqLineItems],INCLUDE_STATISTICS, 3)
FROM [Association]
NATURAL PREDICTION JOIN
(SELECT
(SELECT 'Classic Vest' as [Model])
AS [v Assoc Seq Line Items])
AS t
```

Example results:

MODEL	$SUPPORT	$PROBABILITY	$ADJUSTEDPROBABILITY
Sport-100	4334	0.291283	0.252696
Water Bottle	2866	0.19262	0.175205

MODEL	$SUPPORT	$PROBABILITY	$ADJUSTEDPROBABILITY
Patch kit	2113	0.142012	0.132389

Return to Top

Sample Query 7: Determining Confidence for Related Itemsets

Whereas rules are useful for generating recommendations, itemsets are more interesting for deeper analysis of the patterns in the data set. For example, if you were not satisfied with the recommendation that are returned by the previous sample query, you could examine other itemsets that contain Product A, to can get a better idea of whether Product A is an accessory that people tend to buy with all kinds of products, or whether A is strongly correlated with purchases of particular products. The easiest way to explore these relationships is by filtering the itemsets in the Microsoft Association Viewer; however, you can retrieve the same information with a query.

The following sample query returns all itemsets that include the Water Bottle item, including the single item Water bottle.

```
SELECT TOP 100 FROM
(
SELECT FLATTENED NODE_CAPTION, NODE_SUPPORT,
(SELECT ATTRIBUTE_NAME from NODE_DISTRIBUTION
WHERE ATTRIBUTE_NAME = 'v Assoc Seq Line Items(Water Bottle)') as D
FROM Association.CONTENT
WHERE NODE_TYPE = 7
) AS Items
WHERE [D.ATTRIBUTE_NAME] <> NULL
ORDER BY NODE_SUPPORT DESC
```

Example results:

NODE_CAPTION	NODE_SUPPORT	D.ATTRIBUTE_NAME
Water Bottle = Existing	2866	v Assoc Seq Line Items(Water Bottle)
Mountain Bottle Cage = Existing, Water Bottle = Existing	1136	v Assoc Seq Line Items(Water Bottle)
Road Bottle Cage = Existing, Water Bottle = Existing	1068	v Assoc Seq Line Items(Water Bottle)
Water Bottle = Existing, Sport-100 = Existing	734	v Assoc Seq Line Items(Water Bottle)

This query returns both the rows from the nested table that match the criteria, and all the rows from the outside or case table. Therefore, you must add a condition that eliminates the case table rows that have a null value for the target attribute name.

Return to Top

Function List

All Microsoft algorithms support a common set of functions. However, the Microsoft Association algorithm supports the additional functions listed in the following table.

Prediction Function	Usage
IsDescendant (DMX)	Determines whether one node is a child of another node in the neural network graph.
IsInNode (DMX)	Indicates whether the specified node contains the current case.
PredictAdjustedProbability (DMX)	Returns the weighted probability.
PredictAssociation (DMX)	Predicts membership in an associative dataset.
PredictHistogram (DMX)	Returns a table of values related to the current predicted value.
PredictNodeId (DMX)	Returns the Node_ID for each case.
PredictProbability (DMX)	Returns probability for the predicted value.
PredictSupport (DMX)	Returns the support value for a specified state.
PredictVariance (DMX)	Returns variance for the predicted value.

See Also

Microsoft Association Algorithm
Microsoft Association Algorithm Technical Reference
Mining Model Content for Association Models (Analysis Services - Data Mining)

Microsoft Clustering Algorithm

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Clustering algorithm is a *segmentation* or *clustering* algorithm that iterates over cases in a dataset to group them into clusters that contain similar characteristics. These groupings are useful for exploring data, identifying anomalies in the data, and creating predictions.

Clustering models identify relationships in a dataset that you might not logically derive through casual observation. For example, you might easily guess that people who commute to their jobs by bicycle do not typically live a long distance from where they work. The algorithm, however, can find other characteristics about bicycle commuters that are not as obvious. In the following diagram, cluster A represents data about people who tend to drive to work, while cluster B represents data about people who tend to ride bicycles to work.

A = Commuters who drive to work
B = Commuters who bicycle to work

The clustering algorithm differs from other data mining algorithms, such as the Microsoft Decision Trees algorithm, in that you do not have to designate a predictable column to be able to build a clustering model. The clustering algorithm trains the model strictly from the relationships that exist in the data and from the clusters that the algorithm identifies.

Example

Consider a group of people who share similar demographic information and who buy similar products from the Adventure Works company. This group of people represents a cluster of data. Several such clusters may exist in a database. By observing the columns that make up a cluster, you can more clearly see how records in a dataset are related to one another.

How the Algorithm Works

The Microsoft Clustering algorithm first identifies relationships in a dataset and generates a series of clusters based on those relationships. A scatter plot is a useful way to visually represent how the algorithm groups data, as shown in the following diagram. The scatter plot represents all the cases in the dataset, and each case is a point on the graph. The clusters group points on the graph and illustrate the relationships that the algorithm identifies.

After first defining the clusters, the algorithm calculates how well the clusters represent groupings of the points, and then tries to redefine the groupings to create clusters that better represent the data. The algorithm iterates through this process until it cannot improve the results more by redefining the clusters.

You can customize the way the algorithm works by selecting a specifying a clustering technique, limiting the maximum number of clusters, or changing the amount of support required to create a cluster. For more

information, see Microsoft Clustering Algorithm Technical Reference. this algorithm includes two popular clustering methods: K-means clustering and the Expectation Maximization method.

Data Required for Clustering Models

When you prepare data for use in training a clustering model, you should understand the requirements for the particular algorithm, including how much data is needed, and how the data is used.

The requirements for a clustering model are as follows:

- **A single key column** Each model must contain one numeric or text column that uniquely identifies each record. Compound keys are not allowed.
- **Input columns** Each model must contain at least one input column that contains the values that are used to build the clusters. You can have as many input columns as you want, but depending on the number of values in each column, the addition of extra columns can increase the time it takes to train the model.
- **Optional predictable column** The algorithm does not need a predictable column to build the model, but you can add a predictable column of almost any data type. The values of the predictable column can be treated as input to the clustering model, or you can specify that it be used for prediction only. For example, if you want to predict customer income by clustering on demographics such as region or age, you would specify income as **PredictOnly** and add all the other columns, such as region or age, as inputs.

For more detailed information about the content types and data types supported for clustering models, see the Requirements section of Microsoft Clustering Algorithm Technical Reference.

Viewing a Clustering Model

To explore the model, you can use the **Microsoft Cluster Viewer**. When you view a clustering model, Analysis Services shows you the clusters in a diagram that depicts the relationships among clusters, and also provides a detailed profile of each cluster, a list of the attributes that distinguish each cluster from the others, and the characteristics of the entire training data set. For more information, see Browse a Model Using the Microsoft Cluster Viewer.

If you want to know more detail, you can browse the model in the Microsoft Generic Content Tree Viewer. The content stored for the model includes the distribution for all values in each node, the probability of each cluster, and other information. For more information, see Mining Model Content for Clustering Models (Analysis Services - Data Mining).

Creating Predictions

After the model has been trained, the results are stored as a set of patterns, which you can explore or use to make predictions.

You can create queries to return predictions about whether new data fits into the clusters that were discovered, or to obtain descriptive statistics about the clusters.

For information about how to create queries against a data mining model, see Data Mining Queries. For examples of how to use queries with a clustering model, see Clustering Model Query Examples.

Remarks

- Supports the use of Predictive Model Markup Language (PMML) to create mining models.
- Supports drillthrough.
- Supports the use of OLAP mining models and the creation of data mining dimensions.

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Microsoft Clustering Algorithm Technical Reference
Mining Model Content for Clustering Models (Analysis Services - Data Mining)
Clustering Model Query Examples

Microsoft Clustering Algorithm Technical Reference

5/16/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This section explains the implementation of the Microsoft Clustering algorithm, including the parameters that you can use to control the behavior of clustering models. It also provides guidance about how to improve performance when you create and process clustering models.

For additional information about how to use clustering models, see the following topics:

- Mining Model Content for Clustering Models (Analysis Services - Data Mining)
- Clustering Model Query Examples

Implementation of the Microsoft Clustering Algorithm

The Microsoft Clustering algorithm provides two methods for creating clusters and assigning data points to the clusters. The first, the *K-means* algorithm, is a hard clustering method. This means that a data point can belong to only one cluster, and that a single probability is calculated for the membership of each data point in that cluster. The second method, the *Expectation Maximization* (EM) method, is a *soft clustering* method. This means that a data point always belongs to multiple clusters, and that a probability is calculated for each combination of data point and cluster.

You can choose which algorithm to use by setting the *CLUSTERING_METHOD* parameter. The default method for clustering is scalable EM.

EM Clustering

In EM clustering, the algorithm iteratively refines an initial cluster model to fit the data and determines the probability that a data point exists in a cluster. The algorithm ends the process when the probabilistic model fits the data. The function used to determine the fit is the log-likelihood of the data given the model.

If empty clusters are generated during the process, or if the membership of one or more of the clusters falls below a given threshold, the clusters with low populations are reseeded at new points and the EM algorithm is rerun.

The results of the EM clustering method are probabilistic. This means that every data point belongs to all clusters, but each assignment of a data point to a cluster has a different probability. Because the method allows for clusters to overlap, the sum of items in all the clusters may exceed the total items in the training set. In the mining model results, scores that indicate support are adjusted to account for this.

The EM algorithm is the default algorithm used in Microsoft clustering models. This algorithm is used as the default because it offers multiple advantages in comparison to k-means clustering:

- Requires one database scan, at most.
- Will work despite limited memory (RAM).
- Has the ability to use a forward-only cursor.
- Outperforms sampling approaches.

The Microsoft implementation provides two options: scalable and non-scalable EM. By default, in scalable EM, the first 50,000 records are used to seed the initial scan. If this is successful, the model uses this data only. If the

model cannot be fit using 50,000 records, an additional 50,000 records are read. In non-scalable EM, the entire dataset is read regardless of its size. This method might create more accurate clusters, but the memory requirements can be significant. Because scalable EM operates on a local buffer, iterating through the data is much faster, and the algorithm makes much better use of the CPU memory cache than non-scalable EM. Moreover, scalable EM is three times faster than non-scalable EM, even if all the data can fit in main memory. In the majority of cases, the performance improvement does not lead to lower quality of the complete model.

For a technical report that describes the implementation of EM in the Microsoft Clustering algorithm, see Scaling EM (Expectation Maximization) Clustering to Large Databases.

K-Means Clustering

K-means clustering is a well-known method of assigning cluster membership by minimizing the differences among items in a cluster while maximizing the distance between clusters. The "means" in k-means refers to the *centroid* of the cluster, which is a data point that is chosen arbitrarily and then refined iteratively until it represents the true mean of all data points in the cluster. The "k" refers to an arbitrary number of points that are used to seed the clustering process. The k-means algorithm calculates the squared Euclidean distances between data records in a cluster and the vector that represents the cluster mean, and converges on a final set of k clusters when that sum reaches its minimum value.

The k-means algorithm assigns each data point to exactly one cluster, and does not allow for uncertainty in membership. Membership in a cluster is expressed as a distance from the centroid.

Typically, the k-means algorithm is used for creating clusters of continuous attributes, where calculating distance to a mean is straightforward. However, the Microsoft implementation adapts the k-means method to cluster discrete attributes, by using probabilities. For discrete attributes, the distance of a data point from a particular cluster is calculated as follows:

1 - P(data point, cluster)

> **NOTE**
>
> The Microsoft Clustering algorithm does not expose the distance function used in computing k-means, and measures of distance are not available in the completed model. However, you can use a prediction function to return a value that corresponds to distance, where distance is computed as the probability of a data point belonging to the cluster. For more information, see ClusterProbability (DMX).

The k-means algorithm provides two methods of sampling the data set: non-scalable K-means, which loads the entire data set and makes one clustering pass, or scalable k-means, where the algorithm uses the first 50,000 cases and reads more cases only if it needs more data to achieve a good fit of model to data.

Updates to the Microsoft Clustering Algorithm in SQL Server 2008

In SQL Server 2008, the default configuration of the Microsoft clustering algorithm was changed to use the internal parameter, NORMALIZATION = 1. Normalization is performed using z-score statistics, and assumes normal distribution. The intent of this change in the default behavior is to minimize the effect of attributes that might have large magnitudes and many outliers. However, z-score normalization may alter the clustering results on distributions that are not normal (such as uniform distributions). To prevent normalization and obtain the same behavior as the K-means clustering algorithm in SQL Server 2005, you can use the **Parameter Settings** dialog box to add the custom parameter, NORMALIZATION, and set its value to 0.

> **NOTE**
>
> The NORMALIZATION parameter is an internal property of the Microsoft Clustering algorithm and is not supported. In general, the use of normalization is recommended in clustering models to improve model results.

Customizing the Microsoft Clustering Algorithm

The Microsoft Clustering algorithm supports several parameters that affect the behavior, performance, and accuracy of the resulting mining model.

Setting Algorithm Parameters

The following table describes the parameters that can be used with the Microsoft Clustering algorithm. These parameters affect both the performance and accuracy of the resulting mining model.

CLUSTERING_METHOD
Specifies the clustering method for the algorithm to use. The following clustering methods are available:

ID	METHOD
1	Scalable EM
2	Non-scalable EM
3	Scalable K-Means
4	Non-scalable K-Means.

The default is 1 (scalable EM).

CLUSTER_COUNT
Specifies the approximate number of clusters to be built by the algorithm. If the approximate number of clusters cannot be built from the data, the algorithm builds as many clusters as possible. Setting the CLUSTER_COUNT to 0 causes the algorithm to use heuristics to best determine the number of clusters to build.

The default is 10.

CLUSTER_SEED
Specifies the seed number that is used to randomly generate clusters for the initial stage of model building.

By changing this number, you can change the way that the initial clusters are built, and then compare models that have been built using different seeds. If the seed is changed but the clusters that are found do not change greatly , the model can be considered relatively stable.

The default is 0.

MINIMUM_SUPPORT
Specifies the minimum number of cases that are required to build a cluster. If the number of cases in the cluster is lower than this number, the cluster is treated as empty and discarded.

If you set this number too high, you may miss valid clusters.

> **NOTE**
> If you use EM, which is the default clustering method, some clusters may have a support value that is lower than the specified value. This is because each case is evaluated for its membership in all possible clusters, and for some clusters there may be only minimal support.

The default is 1.

MODELLING_CARDINALITY
Specifies the number of sample models that are constructed during the clustering process.

Reducing the number of candidate models can improve performance at the risk of missing some good candidate models.

The default is 10.

STOPPING_TOLERANCE
Specifies the value that is used to determine when convergence is reached and the algorithm is finished building the model. Convergence is reached when the overall change in cluster probabilities is less than the ratio of the STOPPING_TOLERANCE parameter divided by the size of the model.

The default is 10.

SAMPLE_SIZE
Specifies the number of cases that the algorithm uses on each pass if the CLUSTERING_METHOD parameter is set to one of the scalable clustering methods. Setting the SAMPLE_SIZE parameter to 0 will cause the whole dataset to be clustered in a single pass. Loading the entire dataset in a single pass can cause memory and performance issues.

The default is 50000.

MAXIMUM_INPUT_ATTRIBUTES
Specifies the maximum number of input attributes that the algorithm can handle before it invokes feature selection. Setting this value to 0 specifies that there is no maximum number of attributes.

Increasing the number of attributes can significantly degrade performance.

The default is 255.

MAXIMUM_STATES
Specifies the maximum number of attribute states that the algorithm supports. If an attribute has more states than the maximum, the algorithm uses the most popular states and ignores the remaining states.

Increasing the number of states can significantly degrade performance.

The default is 100.

Modeling Flags

The algorithm supports the following modeling flags. You define modeling flags when you create the mining structure or mining model. The modeling flags specify how values in each column are handled during analysis.

MODELING FLAG	DESCRIPTION
MODEL_EXISTENCE_ONLY	The column will be treated as having two possible states: Missing and Existing. A null is a missing value. Applies to mining model column.
NOT NULL	The column cannot contain a null. An error will result if Analysis Services encounters a null during model training. Applies to mining structure column.

Requirements

A clustering model must contain a key column and input columns. You can also define input columns as being predictable. Columns set to **Predict Only** are not used to build clusters. The distribution of these values in the clusters are calculated after the clusters are built.

Input and Predictable Columns

The Microsoft Clustering algorithm supports the specific input columns and predictable columns that are listed in the following table. For more information about what the content types mean when used in a mining model, see Content Types (Data Mining).

COLUMN	CONTENT TYPES
Input attribute	Continuous, Cyclical, Discrete, Discretized, Key, Table, Ordered
Predictable attribute	Continuous, Cyclical, Discrete, Discretized, Table, Ordered

> **NOTE**
> Cyclical and Ordered content types are supported, but the algorithm treats them as discrete values and does not perform special processing.

See Also

Microsoft Clustering Algorithm
Clustering Model Query Examples
Mining Model Content for Clustering Models (Analysis Services - Data Mining)

Mining Model Content for Clustering Models (Analysis Services - Data Mining)

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes mining model content that is specific to models that use the Microsoft Clustering algorithm. For a general explanation of mining model content for all model types, see Mining Model Content (Analysis Services - Data Mining).

Understanding the Structure of a Clustering Model

A clustering model has a simple structure. Each model has a single parent node that represents the model and its metadata, and each parent node has a flat list of clusters (NODE_TYPE = 5). This organization is shown in the following image.

Each child node represents a single cluster and contains detailed statistics about the attributes of the cases in that cluster. This includes a count of the number of cases in the cluster, and the distribution of values that distinguish the cluster from other clusters.

> **NOTE**
> You do not need to iterate through the nodes to get a count or description of the clusters; the model parent node also counts and lists the clusters.

The parent node contains useful statistics that describe the actual distribution of all the training cases. These statistics are found in the nested table column, NODE_DISTRIBUTION. For example, the following table shows several rows from the NODE_DISTRIBUTION table that describe the distribution of customer demographics for the clustering model, `TM_Clustering`, that you create in the Basic Data Mining Tutorial:

ATTRIBUTE_NAME	ATRIBUTE_VALUE	SUPPORT	PROBABILITY	VARIANCE	VALUE_TYPE
Age	Missing	0	0	0	1 (Missing)
Age	44.9016152716593	12939	1	125.663453102554	3 (Continuous)
Gender	Missing	0	0	0	1 (Missing)
Gender	F	6350	0.490764355823479	0	4 (Discrete)

ATTRIBUTE_NAME	ATRIBUTE_VALUE	SUPPORT	PROBABILITY	VARIANCE	VALUE_TYPE
Gender	M	6589	0.509235644176521	0	4 (Discrete)

From these results, you can see that there were 12939 cases used to build the model, that the ratio of males to females was about 50-50, and that the mean age was 44. The descriptive statistics vary depending on whether the attribute being reported is a continuous numeric data type, such as age, or a discrete value type, such as gender. The statistical measures *mean* and *variance* are computed for continuous data types, whereas *probability* and *support* are computed for discrete data types.

> **NOTE**
> The variance represents the total variance for the cluster. When the value for variance is small, it indicates that most values in the column were fairly close to the mean. To obtain the standard deviation, calculate the square root of the variance.

Note that for each of the attributes there is a **Missing** value type that tells you how many cases had no data for that attribute. Missing data can be significant and affects calculations in different ways, depending on the data type. For more information, see Missing Values (Analysis Services - Data Mining).

Model Content for a Clustering Model

This section provides detail and examples only for those columns in the mining model content that are relevant for clustering models.

For information about the general-purpose columns in the schema rowset, such as MODEL_CATALOG and MODEL_NAME, see Mining Model Content (Analysis Services - Data Mining).

MODEL_CATALOG
Name of the database where the model is stored.

MODEL_NAME
Name of the model.

ATTRIBUTE_NAME
Always blank in clustering models because there is no predictable attribute in the mode.

NODE_NAME
Always same as NODE_UNIQUE_NAME.

NODE_UNIQUE_NAME
A unique identifier for the node within the model. This value cannot be changed.

NODE_TYPE
A clustering model outputs the following node types:

NODE ID AND NAME	DESCRIPTION
1 (Model)	Root node for model.
5 (Cluster)	Contains a count of cases in the cluster, the characteristics of cases in the cluster, and statistics that describe the values in the cluster.

NODE_CAPTION
A friendly name for display purposes. When you create a model, the value of NODE_UNIQUE_NAME is

automatically used as the caption. However, you can change the value for NODE_CAPTION to update the display name for the cluster, either programmatically or by using the viewer.

> **NOTE**
> When you reprocess the model, all name changes will be overwritten by the new values. You cannot persist names in the model, or track changes in cluster membership between different versions of a model.

CHILDREN_CARDINALITY
An estimate of the number of children that the node has.

Parent node Indicates the number of clusters in the model.

Cluster nodes Always 0.

PARENT_UNIQUE_NAME
The unique name of the node's parent.

Parent node Always NULL

Cluster nodes Usually 000.

NODE_DESCRIPTION
A description of the node.

Parent node Always **(All)**.

Cluster nodes A comma-separated list of the primary attributes that distinguish the cluster from other clusters.

NODE_RULE
Not used for clustering models.

MARGINAL_RULE
Not used for clustering models.

NODE_PROBABILITY
The probability associated with this node. **Parent node** Always 1.

Cluster nodes The probability represents the compound probability of the attributes, with some adjustments depending on the algorithm used to create the clustering model.

MARGINAL_PROBABILITY
The probability of reaching the node from the parent node. In a clustering model, the marginal probability is always the same as the node probability.

NODE_DISTRIBUTION
A table that contains the probability histogram of the node.

Parent node See the Introduction to this topic.

Cluster nodes Represents the distribution of attributes and values for cases that are included in this cluster.

NODE_SUPPORT
The number of cases that support this node. **Parent node** Indicates the number of training cases for the entire model.

Cluster nodes Indicates the size of the cluster as a number of cases.

Note If the model uses K-Means clustering, each case can belong to only one cluster. However, if the model uses EM clustering, each case can belong to different cluster, and the case is assigned a weighted distance for each

cluster to which it belongs. Therefore, for EM models the sum of support for an individual cluster is greater than support for the overall model.

MSOLAP_MODEL_COLUMN
Not used for clustering models.

MSOLAP_NODE_SCORE
Displays a score associated with the node.

Parent node The Bayesian Information Criterion (BIC) score for the clustering model.

Cluster nodes Always 0.

MSOLAP_NODE_SHORT_CAPTION
A label used for display purposes. You cannot change this caption.

Parent node The type of model: Cluster model

Cluster nodes The name of the cluster. Example: Cluster 1.

Remarks

Analysis Services provides multiple methods for creating a clustering model. If you do not know which method was used to create the model that you are working with, you can retrieve the model metadata programmatically, by using an ADOMD client or AMO, or by querying the data mining schema rowset. For more information, see Query the Parameters Used to Create a Mining Model.

> **NOTE**
> The structure and content of the model stay the same, regardless of which clustering method or parameters you use.

See Also

Mining Model Content (Analysis Services - Data Mining)
Data Mining Model Viewers
Microsoft Clustering Algorithm
Data Mining Queries

Clustering Model Query Examples

5/16/2018 • 14 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a query against a data mining model, you can retrieve metadata about the model, or create a content query that provides details about the patterns discovered in analysis. Alternatively, you can create a prediction query, which uses the patterns in the model to make predictions for new data. Each type of query will provide different information. For example, a content query might provide additional details about the clusters that were found, whereas a prediction query might tell you in which cluster a new data point is most likely to belong.

This section explains how to create queries for models that are based on the Microsoft Clustering algorithm.

Content Queries

Getting Model Metadata by Using DMX

Retrieving Model Metadata from the Schema Rowset

Returning a Cluster or a List of Clusters

Returning Attributes for a Cluster

Returning a Cluster Profile Using System Stored Procedures

Finding Discriminating Factors for a Cluster

Returning Cases that Belong to a Cluster

Prediction Queries

Predicting Outcomes from a Clustering Model

Determining Cluster Membership

Returning All Possible Clusters with Probability and Distance

Finding Information about the Model

All mining models expose the content learned by the algorithm according to a standardized schema, the mining model schema rowset. You can create queries against the mining model schema rowset by using Data Mining Extension (DMX) statements. In SQL Server 2017, you can also query the schema rowsets directly as system tables.

Return to Top

Sample Query 1: Getting Model Metadata by Using DMX

The following query returns basic metadata about the clustering model, `TM_Clustering`, that you created in the Basic Data Mining Tutorial. The metadata available in the parent node of a clustering model includes the name of the model, the database where the model is stored, and the number of child nodes in the model. This query uses a DMX content query to retrieve the metadata from the parent node of the model:

```
SELECT MODEL_CATALOG, MODEL_NAME, NODE_CAPTION,
NODE_SUPPORT, [CHILDREN_CARDINALITY], NODE_DESCRIPTION
FROM TM_Clustering.CONTENT
WHERE NODE_TYPE = 1
```

NOTE

You must enclose the name of the column, CHILDREN_CARDINALITY, in brackets to distinguish it from the Multidimensional Expressions (MDX) reserved keyword of the same name.

Example results:

MODEL_CATALOG	TM_Clustering
MODEL_NAME	Adventure Works DW
NODE_CAPTION	Cluster Model
NODE_SUPPORT	12939
CHILDREN_CARDINALITY	10
NODE_DESCRIPTION	All

For a definition of what these columns mean in a clustering model, see Mining Model Content for Clustering Models (Analysis Services - Data Mining).

Return to Top

Sample Query 2: Retrieving Model Metadata from the Schema Rowset

By querying the data mining schema rowset, you can find the same information that is returned in a DMX content query. However, the schema rowset provides some additional columns. These include the parameters that were used when the model was created, the date and time that the model was last processed, and the owner of the model.

The following example returns the date the model was created, modified, and last processed, together with the clustering parameters that were used to build the model, and the size of the training set. This information can be useful for documenting the model, or for determining which of the clustering options were used to create an existing model.

```
SELECT MODEL_NAME, DATE_CREATED, LAST_PROCESSED, PREDICTION_ENTITY, MINING_PARAMETERS
from $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = 'TM_Clustering'
```

Example results:

MODEL_NAME	TM_Clustering
DATE_CREATED	10/12/2007 7:42:51 PM

LAST_PROCESSED	10/12/2007 8:09:54 PM
PREDICTION_ENTITY	Bike Buyer
MINING_PARAMETERS	CLUSTER_COUNT=10, CLUSTER_SEED=0, CLUSTERING_METHOD=1, MAXIMUM_INPUT_ATTRIBUTES=255, MAXIMUM_STATES=100, MINIMUM_SUPPORT=1, MODELLING_CARDINALITY=10, SAMPLE_SIZE=50000, STOPPING_TOLERANCE=10

Return to Top

Finding Information about Clusters

The most useful content queries on clustering models generally return the same type of information that you can browse by using the **Cluster Viewer**. This includes cluster profiles, cluster characteristics, and cluster discrimination. This section provides examples of queries that retrieve this information.

Sample Query 3: Returning a Cluster or List of Clusters

Because all clusters have a node type of 5, you can easily retrieve a list of the clusters by querying the model content for only the nodes of that type. You can also filter the nodes that are returned by probability or by support, as shown in this example.

```
SELECT NODE_NAME, NODE_CAPTION ,NODE_SUPPORT, NODE_DESCRIPTION
FROM TM_Clustering.CONTENT
WHERE NODE_TYPE = 5 AND NODE_SUPPORT > 1000
```

Example results:

NODE_NAME	002
NODE_CAPTION	Cluster 2
NODE_SUPPORT	1649

NODE_DESCRIPTION	English Education=Graduate Degree , 32 <=Age <=48 , Number Cars Owned=0 , 35964.0771121808 <=Yearly Income <=97407.7163393957 , English Occupation=Professional , Commute Distance=2-5 Miles , Region=North America , Bike Buyer=1 , Number Children At Home=0 , Number Cars Owned=1 , Commute Distance=0-1 Miles , English Education=Bachelors , Total Children=1 , Number Children At Home=2 , English Occupation=Skilled Manual , Marital Status=S , Total Children=0 , House Owner Flag=0 , Gender=F , Total Children=2 , Region=Pacific

The attributes that define the cluster can be found in two columns in the data mining schema rowset.

- The NODE_DESCRIPTION column contains a comma-separated list of attributes. Note that the list of attributes might be abbreviated for display purposes.
- The nested table in the NODE_DISTRIBUTION column contains the full list of attributes for the cluster. If your client does not support hierarchical rowsets, you can return the nested table by adding the FLATTENED keyword before the SELECT column list. For more information about the use of the FLATTENED keyword, see SELECT FROM <model>.CONTENT (DMX).

Return to Top

Sample Query 4: Returning Attributes for a Cluster

For every cluster, the **Cluster Viewer** displays a profile that lists the attributes and their values. The viewer also displays a histogram that shows the distribution of values for the whole population of cases in the model. If you are browsing the model in the viewer, you can easily copy the histogram from the Mining Legend and then paste it to Excel or a Word document. You can also use the Cluster Characteristics pane of the viewer to graphically compare the attributes of different clusters.

However, if you must obtain values for more than one cluster at a time, it is easier to query the model. For example, when you browse the model, you might notice that the top two clusters differ with regard to one attribute, `Number Cars Owned`. Therefore, you want to extract the values for each cluster.

```
SELECT TOP 2 NODE_NAME,
(SELECT ATTRIBUTE_VALUE, [PROBABILITY] FROM NODE_DISTRIBUTION WHERE ATTRIBUTE_NAME = 'Number Cars Owned')
AS t
FROM [TM_Clustering].CONTENT
WHERE NODE_TYPE = 5
```

The first line of the code specifies that you want only the top two clusters.

> **NOTE**
> By default, the clusters are ordered by support. Therefore, the NODE_SUPPORT column can be omitted.

The second line of the code adds a sub-select statement that returns only certain columns from the nested table column. Furthermore, it restricts the rows from the nested table to those related to the target attribute, `Number Cars Owned`. To simplify the display, the nested table is aliased.

NOTE

The nested table column, `PROBABILITY`, must be enclosed in brackets because it is also the name of a reserved MDX keyword.

Example results:

NODE_NAME	T.ATTRIBUTE_VALUE	T.PROBABILITY
001	2	0.829207754
001	1	0.109354156
001	3	0.034481552
001	4	0.013503302
001	0	0.013453236
001	Missing	0
002	0	0.576980023
002	1	0.406623939
002	2	0.016380082
002	3	1.60E-05
002	4	0
002	Missing	0

Return to Top

Sample Query 5: Return a Cluster Profile Using System Stored Procedures

As a shortcut, rather than writing your own queries by using DMX, you can also call the system stored procedures that Analysis Services uses to work with clusters. The following example illustrates how to use the internal stored procedures to return the profile for a cluster with the ID of 002.

```
CALL System.Microsoft.AnalysisServices.System.DataMining.Clustering.GetClusterProfiles('TM_Clustering",
'002',0.0005
```

Similarly, you can use a system stored procedure to return the characteristics of a specific cluster, as shown in the following example:

```
CALL
System.Microsoft.AnalysisServices.System.DataMining.Clustering.GetClusterCharacteristics('TM_Clustering",
'009',0.0005
```

Example results:

ATTRIBUTES	VALUES	FREQUENCY	SUPPORT
Number Children at Home	0	0.999999829076798	899
Region	North America	0.999852875241508	899
Total Children	0	0.993860958572323	893

> **NOTE**
> The data mining system stored procedures are for internal use and Microsoft reserves the right to change them as needed. For production use, we recommend that you create queries by using DMX, AMO, or XMLA.

Return to Top

Sample Query 6: Find Discriminating Factors for a Cluster

The **Cluster Discrimination** tab of the **Cluster Viewer** enables you to easily compare a cluster with another cluster, or compare a cluster with all remaining cases (the complement of the cluster).

However, creating queries to return this information can be complex, and you might need some additional processing on the client to store the temporary results and compare the results of two or more queries. As a shortcut, you can use the system stored procedures.

The following query returns a single table that indicates the primary discriminating factors between the two clusters that have the node IDs of 009 and 007. Attributes with positive values favor cluster 009, whereas attributes with negative values favor cluster 007.

```
CALL
System.Microsoft.AnalysisServices.System.DataMining.Clustering.GetClusterDiscrimination('TM_Clustering','009'
,'007',0.0005,true)
```

Example results:

ATTRIBUTES	VALUES	SCORE
Region	North America	100
English Occupation	Skilled Manual	94.9003803898654
Region	Europe	-72.5041051379789
English Occupation	Manual	-69.6503163202722

This is the same information that is presented in the chart of the **Cluster Discrimination** viewer if you select Cluster 9 from the first drop-down list and Cluster 7 from the second drop-down list. To compare cluster 9 with its complement, you use the empty string in the second parameter, as shown in the following example:

```
CALL
System.Microsoft.AnalysisServices.System.DataMining.Clustering.GetClusterDiscrimination('TM_Clustering','009'
,'',0.0005,true)
```

> **NOTE**
> The data mining system stored procedures are for internal use and Microsoft reserves the right to change them as needed. For production use, we recommend that you create queries by using DMX, AMO, or XMLA.

Return to Top

Sample Query 7: Returning Cases that Belong to a Cluster

If drillthrough has been enabled on the mining model, you can create queries that return detailed information about the cases used in the model. Moreover, if drillthrough has been enabled on the mining structure, you can include columns from the underlying structure by using the StructureColumn (DMX) function.

The following example returns two columns that were used in the model, Age and Region, and one more column, First Name, that was not used in the model. The query returns only cases that were classified into Cluster 1.

```
SELECT [Age], [Region], StructureColumn('First Name')
FROM [TM_Clustering].CASES
WHERE IsInNode('001')
```

To return the cases that belong to a cluster, you must know the ID of the cluster. You can obtain the ID of the cluster by browsing the model in one of the viewers. Or, you can rename a cluster for easier reference, after which you could use the name in place of an ID number. However, know that the names that you assign to a cluster will be lost if the model is reprocessed.

Return to Top

Making Predictions using the Model

Although clustering is typically used for describing and understanding data, the Microsoft implementation also lets you make prediction about cluster membership, and return probabilities associated with the prediction. This section provides examples of how to create prediction queries on clustering models. You can make predictions for multiple cases, by specifying a tabular data source, or you can provide new values on at a time by creating a singleton query. For clarity the examples in this section are all singleton queries.

For more information about how to create prediction queries using DMX, see Data Mining Query Tools.

Return to Top

Sample Query 8: Predicting Outcomes from a Clustering Model

If the clustering model you create contains a predictable attribute, you can use the model to make predictions about outcomes. However, the model handles the predictable attribute differently depending on whether you set the predictable column to **Predict** or **PredictOnly**. If you set the usage of the column to **Predict**, the values for that attribute are added to the clustering model and appear as attributes in the finished model. However, if you set the usage of the column to **PredictOnly**, the values are not used to create clusters. Instead, after the mode is completed, the clustering algorithm creates new values for the **PredictOnly** attribute based on the clusters to which each case belongs.

The following query provides a single new case to the model, where the only information about the case is the age and gender. The SELECT statement specifies the predictable attribute/value pair that you are interested in, and the PredictProbability (DMX) function tells you the probability that a case with those attributes will have the targeted outcome.

```
SELECT
  [TM_Clustering].[Bike Buyer], PredictProbability([Bike Buyer],1)
FROM
  [TM_Clustering]
NATURAL PREDICTION JOIN
(SELECT 40 AS [Age],
  'F' AS [Gender]) AS t
```

Example of results when usage is set to **Predict**:

BIKE BUYER	EXPRESSION
1	0.592924735740338

Example of results when the usage is set to **PredictOnly** and the model is reprocessed:

BIKE BUYER	EXPRESSION
1	0.55843544003102

In this example, the difference in the model is not significant. However, sometimes it can be important to detect differences between the actual distribution of values and what the model predicts. The PredictCaseLikelihood (DMX) function is useful in this scenario, because it tells you how likely a case is, given the model.

The number that is returned by the PredictCaseLikelihood function is a probability, and therefore is always between 0 and 1, with a value of .5 representing random outcome. Therefore, a score less than .5 means that the predicted case is unlikely, given the model, and a score over.5 indicates that the predicted case is more likely than not to fit the model.

For example, the following query returns two values that characterize the likelihood of a new sample case. The non-normalized value represents the probability given the current model. When you use the NORMALIZED keyword, the likelihood score that is returned by the function is adjusted by dividing "probability with the model" by "probability without the model".

```
SELECT
PredictCaseLikelihood(NORMALIZED) AS [NormalizedValue], PredictCaseLikelihood(NONNORMALIZED) AS
[NonNormalizedValue]
FROM
  [TM_Clustering_PredictOnly]
NATURAL PREDICTION JOIN
(SELECT 40 AS [Age],
  'F' AS [Gender]) AS t
```

Example results:

NORMALIZEDVALUE	NONNORMALIZEDVALUE
5.56438372679893E-11	8.65459953145182E-68

Note that the numbers in these results are expressed in scientific notation.

Return to Top

Sample Query 9: Determining Cluster Membership

This example uses the Cluster (DMX) function to return the cluster to which the new case is most likely to belong, and uses the ClusterProbability (DMX) function to return the probability for membership in that cluster.

```
SELECT Cluster(), ClusterProbability()
FROM
  [TM_Clustering]
NATURAL PREDICTION JOIN
(SELECT 40 AS [Age],
  'F' AS [Gender],
  'S' AS [Marital Status]) AS t
```

Example results:

$CLUSTER	EXPRESSION
Cluster 2	0.397918596951617

Note By default, the **ClusterProbability** function returns the probability of the most likely cluster. However, you can specify a different cluster by using the syntax `ClusterProbability('cluster name')`. If you do this, be aware that the results from each prediction function are independent of the other results. Therefore, the probability score in the second column could refer to a different cluster than the cluster named in the first column.

Return to Top

Sample Query 10: Returning All Possible Clusters with Probability and Distance

In the previous example, the probability score was not very high. To determine if there is a better cluster, you can use the PredictHistogram (DMX) function together with the Cluster (DMX) function to return a nested table that includes all possible clusters, together with the probability that the new case that belongs to each cluster. The FLATTENED keyword is used to change the hierarchical rowset into a flat table for easier viewing.

```
SELECT FLATTENED PredictHistogram(Cluster())
From
  [TM_Clustering]
NATURAL PREDICTION JOIN
(SELECT 40 AS [Age],
  'F' AS [Gender],
  'S' AS [Marital Status])
```

EXPRESSION.$CLUSTER	EXPRESSION.$DISTANCE	EXPRESSION.$PROBABILITY
Cluster 2	0.602081403048383	0.397918596951617
Cluster 10	0.719691686785675	0.280308313214325
Cluster 4	0.867772590378791	0.132227409621209
Cluster 5	0.931039872200985	0.0689601277990149
Cluster 3	0.942359230072167	0.0576407699278328
Cluster 6	0.958973668972756	0.0410263310272437
Cluster 7	0.979081275926724	0.0209187240732763
Cluster 1	0.999169044818624	0.000830955181376364
Cluster 9	0.999831227795894	0.000168772204105754

EXPRESSION.$CLUSTER	EXPRESSION.$DISTANCE	EXPRESSION.$PROBABILITY
Cluster 8	1	0

By default, the results are ranked by probability. The results tell you that, even though the probability for Cluster 2 is fairly low, Cluster 2 is still the best fit for the new data point.

Note The additional column, `$DISTANCE`, represents the distance from the data point to the cluster. By default, the Microsoft Clustering Algorithm uses scalable EM clustering, which assigns multiple clusters to each data point and ranks the possible clusters. However, if you create your clustering model using the K-means algorithm, only one cluster can be assigned to each data point, and this query would return only one row. Understanding these differences is necessary to interpret the results of the PredictCaseLikelihood (DMX) function. For more information about the differences between EM and K-means clustering, see Microsoft Clustering Algorithm Technical Reference.

Return to Top

Function List

All Microsoft algorithms support a common set of functions. However, models that are built by using the Microsoft Clustering algorithm support the additional functions that are listed in the following table.

Prediction Function	Usage
Cluster (DMX)	Returns the cluster that is most likely to contain the input case.
ClusterDistance (DMX)	Returns the distance of the input case from the specified cluster, or if no cluster is specified, the distance of the input case from the most likely cluster. Returns the probability that the input case belongs to the specified cluster.
ClusterProbability (DMX)	Returns the probability that the input case belongs to the specified cluster.
IsDescendant (DMX)	Determines whether one node is a child of another node in the model.
IsInNode (DMX)	Indicates whether the specified node contains the current case.
PredictAdjustedProbability (DMX)	Returns the weighted probability.
PredictAssociation (DMX)	Predicts membership in an associative dataset.
PredictCaseLikelihood (DMX)	Returns the likelihood that an input case will fit in the existing model.
PredictHistogram (DMX)	Returns a table of values related to the current predicted value.
PredictNodeId (DMX)	Returns the Node_ID for each case.

PredictProbability (DMX)	Returns probability for the predicted value.
PredictStdev (DMX)	Returns the predicted standard deviation for the specified column.
PredictSupport (DMX)	Returns the support value for a specified state.
PredictVariance (DMX)	Returns the variance of a specified column.

For the syntax of specific functions, see Data Mining Extensions (DMX) Function Reference.

See Also

Data Mining Queries
Microsoft Clustering Algorithm Technical Reference
Microsoft Clustering Algorithm

Microsoft Decision Trees Algorithm

5/16/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Decision Trees algorithm is a classification and regression algorithm for use in predictive modeling of both discrete and continuous attributes.

For discrete attributes, the algorithm makes predictions based on the relationships between input columns in a dataset. It uses the values, known as states, of those columns to predict the states of a column that you designate as predictable. Specifically, the algorithm identifies the input columns that are correlated with the predictable column. For example, in a scenario to predict which customers are likely to purchase a bicycle, if nine out of ten younger customers buy a bicycle, but only two out of ten older customers do so, the algorithm infers that age is a good predictor of bicycle purchase. The decision tree makes predictions based on this tendency toward a particular outcome.

For continuous attributes, the algorithm uses linear regression to determine where a decision tree splits.

If more than one column is set to predictable, or if the input data contains a nested table that is set to predictable, the algorithm builds a separate decision tree for each predictable column

Example

The marketing department of the Adventure Works Cycles company wants to identify the characteristics of previous customers that might indicate whether those customers are likely to buy a product in the future. The AdventureWorks2012 database stores demographic information that describes previous customers. By using the Microsoft Decision Trees algorithm to analyze this information, the marketing department can build a model that predicts whether a particular customer will purchase products, based on the states of known columns about that customer, such as demographics or past buying patterns.

How the Algorithm Works

The Microsoft Decision Trees algorithm builds a data mining model by creating a series of splits in the tree. These splits are represented as *nodes*. The algorithm adds a node to the model every time that an input column is found to be significantly correlated with the predictable column. The way that the algorithm determines a split is different depending on whether it is predicting a continuous column or a discrete column.

The Microsoft Decision Trees algorithm uses *feature selection* to guide the selection of the most useful attributes. Feature selection is used by all SQL Server Data Mining algorithms to improve performance and the quality of analysis. Feature selection is important to prevent unimportant attributes from using processor time. If you use too many input or predictable attributes when you design a data mining model, the model can take a very long time to process, or even run out of memory. Methods used to determine whether to split the tree include industry-standard metrics for *entropy* and Bayesian networks*.* For more information about the methods used to select meaningful attributes and then score and rank the attributes, see Feature Selection (Data Mining).

A common problem in data mining models is that the model becomes too sensitive to small differences in the training data, in which case it said to be *overfitted* or *over-trained*. An overfitted model cannot be generalized to other data sets. To avoid overfitting on any particular set of data, the Microsoft Decision Trees algorithm uses techniques for controlling the growth of the tree. For a more in-depth explanation of how the Microsoft Decision Trees algorithm works, see Microsoft Decision Trees Algorithm Technical Reference.

Predicting Discrete Columns

The way that the Microsoft Decision Trees algorithm builds a tree for a discrete predictable column can be demonstrated by using a histogram. The following diagram shows a histogram that plots a predictable column, Bike Buyers, against an input column, Age. The histogram shows that the age of a person helps distinguish whether that person will purchase a bicycle.

The correlation that is shown in the diagram would cause the Microsoft Decision Trees algorithm to create a new node in the model.

As the algorithm adds new nodes to a model, a tree structure is formed. The top node of the tree describes the breakdown of the predictable column for the overall population of customers. As the model continues to grow, the algorithm considers all columns.

Predicting Continuous Columns

When the Microsoft Decision Trees algorithm builds a tree based on a continuous predictable column, each node contains a regression formula. A split occurs at a point of non-linearity in the regression formula. For example, consider the following diagram.

In a standard regression model, you would attempt to derive a single formula that represents the trend and relationships for the data as a whole. However, a single formula might do a poor job of capturing the discontinuity in complex data. Instead, the Microsoft Decision Trees algorithm looks for segments of the tree that are largely linear and creates separate formulas for these segments. By breaking up the data into different segments, the model can do a much better job of approximating the data.

The following diagram represents the tree diagram for the model in the scatterplot above. To predict the outcome, the model provides two different formulas: one for the left branch, with the formula y = .5x x 5, and one for the right branch, with the formula y = .25x + 8.75. The point where the two lines come together in the scatterplot is the point of non-linearity, and is the point where a node in a decision tree model would split.

This is a simple model with only two linear equations; therefore, the split in the tree is immediately after the **All** node. However, a split can occur at any level of the tree. That means that in a tree containing multiple levels and nodes, where each node is characterized by a different collection of attributes, a formula might be shared across multiple nodes, or apply only to a single node. For example, you might get one formula for a node defined as "customers over a certain age and income", and another in a node that represents "customers who commute long distances". To see the formula for an individual node or segment, just click the node.

Data Required for Decision Tree Models

When you prepare data for use in a decision trees model, you should understand the requirements for the particular algorithm, including how much data is needed, and how the data is used.

The requirements for a decision tree model are as follows:

- **A single key column** Each model must contain one numeric or text column that uniquely identifies each record. Compound keys are not permitted.
- **A predictable column** Requires at least one predictable column. You can include multiple predictable attributes in a model, and the predictable attributes can be of different types, either numeric or discrete. However, increasing the number of predictable attributes can increase processing time.
- **Input columns** Requires input columns, which can be discrete or continuous. Increasing the number of input attributes affects processing time.

For more detailed information about the content types and data types supported for decision tree models, see the Requirements section of Microsoft Decision Trees Algorithm Technical Reference.

Viewing a Decision Trees Model

To explore the model, you can use the **Microsoft Tree Viewer**. If your model generates multiple trees, you can select a tree and the viewer shows you a breakdown of how the cases are categorized for each predictable attribute. You can also view the interaction of the trees by using the dependency network viewer. For more information, see Browse a Model Using the Microsoft Tree Viewer.

If you want to know more detail about any branch or node in the tree, you can also browse the model by using the Microsoft Generic Content Tree Viewer. The content stored for the model includes the distribution for all values in each node, probabilities at each level of the tree, and regression formulas for continuous attributes. For more information, see Mining Model Content for Decision Tree Models (Analysis Services - Data Mining).

Creating Predictions

After the model has been processed, the results are stored as a set of patterns and statistics, which you can use to explore relationships or make predictions.

For examples of queries to use with a decision trees model, see Decision Trees Model Query Examples.

For general information about how to create queries against mining models, see Data Mining Queries.

Remarks

- Supports the use of Predictive Model Markup Language (PMML) to create mining models.

- Supports drillthrough.
- Supports the use of OLAP mining models and the creation of data mining dimensions.

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Microsoft Decision Trees Algorithm Technical Reference
Decision Trees Model Query Examples
Mining Model Content for Decision Tree Models (Analysis Services - Data Mining)

Microsoft Decision Trees Algorithm Technical Reference

5/16/2018 • 11 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Decision Trees algorithm is a hybrid algorithm that incorporates different methods for creating a tree, and supports multiple analytic tasks, including regression, classification, and association. The Microsoft Decision Trees algorithm supports modeling of both discrete and continuous attributes.

This topic explains the implementation of the algorithm, describes how to customize the behavior of the algorithm for different tasks, and provides links to additional information about querying decision tree models.

Implementation of the Decision Trees Algorithm

The Microsoft Decision Trees algorithm applies the Bayesian approach to learning causal interaction models by obtaining approximate posterior distributions for the models. For a detailed explanation of this approach, see the paper on the Microsoft Research site, by Structure and Parameter Learning.

The methodology for assessing the information value of the *priors* needed for learning is based on the assumption of *likelihood equivalence*. This assumption says that data should not help to discriminate network structures that otherwise represent the same assertions of conditional independence. Each case is assumed to have a single Bayesian prior network and a single measure of confidence for that network.

Using these prior networks, the algorithm then computes the relative *posterior probabilities* of network structures given the current training data, and identifies the network structures that have the highest posterior probabilities.

The Microsoft Decision Trees algorithm uses different methods to compute the best tree. The method used depends on the task, which can be linear regression, classification, or association analysis. A single model can contain multiple trees for different predictable attributes. Moreover, each tree can contain multiple branches, depending on how many attributes and values there are in the data. The shape and depth of the tree built in a particular model depends on the scoring method and other parameters that were used. Changes in the parameters can also affect where the nodes split.

Building the Tree

When the Microsoft Decision Trees algorithm creates the set of possible input values, it performs *feature selection* to identify the attributes and values that provide the most information, and removes from consideration the values that are very rare. The algorithm also groups values into *bins*, to create groupings of values that can be processed as a unit to optimize performance.

A tree is built by determining the correlations between an input and the targeted outcome. After all the attributes have been correlated, the algorithm identifies the single attribute that most cleanly separates the outcomes. This point of the best separation is measured by using an equation that calculates information gain. The attribute that has the best score for information gain is used to divide the cases into subsets, which are then recursively analyzed by the same process, until the tree cannot be split any more.

The exact equation used to evaluate information gain depends on the parameters set when you created the algorithm, the data type of the predictable column, and the data type of the input.

Discrete and Continuous Inputs

When the predictable attribute is discrete and the inputs are discrete, counting the outcomes per input is a matter of creating a matrix and generating scores for each cell in the matrix.

However, when the predictable attribute is discrete and the inputs are continuous, the input of the continuous columns are automatically discretized. You can accept the default and have Analysis Services find the optimum number of bins, or you can control the manner in which continuous inputs are discretized by setting the DiscretizationMethod and DiscretizationBucketCount properties. For more information, see Change the Discretization of a Column in a Mining Model.

For continuous attributes, the algorithm uses linear regression to determine where a decision tree splits.

When the predictable attribute is a continuous numeric data type, feature selection is applied to the outputs as well, to reduce the possible number of outcomes and build the model faster. You can change the threshold for feature selection and thereby increase or decrease the number of possible values by setting the MAXIMUM_OUTPUT_ATTRIBUTES parameter.

For a more detained explanation about how the Microsoft Decision Trees algorithm works with discrete predictable columns, see Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. For more information about how the Microsoft Decision Trees algorithm works with a continuous predictable column, see the appendix of Autoregressive Tree Models for Time-Series Analysis.

Scoring Methods and Feature Selection

The Microsoft Decision Trees algorithm offers three formulas for scoring information gain: Shannon's entropy, Bayesian network with K2 prior, and Bayesian network with a uniform Dirichlet distribution of priors. All three methods are well established in the data mining field. We recommend that you experiment with different parameters and scoring methods to determine which provides the best results. For more information about these scoring methods, see Feature Selection.

All Analysis Services data mining algorithms automatically use feature selection to improve analysis and reduce processing load. The method used for feature selection depends on the algorithm that is used to build the model. The algorithm parameters that control feature selection for a decision trees model are MAXIMUM_INPUT_ATTRIBUTES and MAXIMUM_OUTPUT.

ALGORITHM	METHOD OF ANALYSIS	COMMENTS
Decision Trees	Interestingness score Shannon's Entropy Bayesian with K2 Prior Bayesian Dirichlet with uniform prior (default)	If any columns contain non-binary continuous values, the interestingness score is used for all columns, to ensure consistency. Otherwise, the default or specified method is used.
Linear Regression	Interestingness score	Linear Regression only uses interestingness, because it only supports continuous columns.

Scalability and Performance

Classification is an important data mining strategy. Generally, the amount of information that is needed to classify the cases grows in direct proportion to the number of input records. This limits the size of the data that can be classified. The Microsoft Decision Trees algorithm using uses the following methods to resolve these problems, improve performance, and eliminate memory restrictions:

- Feature selection to optimize the selection of attributes.
- Bayesian scoring to control tree growth.

- Optimization of binning for continuous attributes.
- Dynamic grouping of input values to determine the most important values.

The Microsoft Decision Trees algorithm is fast and scalable, and has been designed to be easily parallelized, meaning that all processors work together to build a single, consistent model. The combination of these characteristics makes the decision-tree classifier an ideal tool for data mining.

If performance constraints are severe, you might be able to improve processing time during the training of a decision tree model by using the following methods. However, if you do so, be aware that eliminating attributes to improve processing performance will change the results of the model, and possibly make it less representative of the total population.

- Increase the value of the COMPLEXITY_PENALTY parameter to limit tree growth.
- Limit the number of items in association models to limit the number of trees that are built.
- Increase the value of the MINIMUM_SUPPORT parameter to avoid overfitting.
- Restrict the number of discrete values for any attribute to 10 or less. You might try grouping values in different ways in different models.

> **NOTE**
> You can use the data exploration tools available in SQL Server 2017 Integration Services (SSIS) to visualize the distribution of values in your data and group your values appropriately before beginning data mining. For more information, see Data Profiling Task and Viewer. You can also use the Data Mining Add-ins for Excel 2007, to explore, group and relabel data in Microsoft Excel.

Customizing the Decision Trees Algorithm

The Microsoft Decision Trees algorithm supports parameters that affect the performance and accuracy of the resulting mining model. You can also set modeling flags on the mining model columns or mining structure columns to control the way that data is processed.

> **NOTE**
> The Microsoft Decision Trees algorithm is available in all editions of SQL Server; however, some advanced parameters for customizing the behavior of the Microsoft Decision Trees algorithm are available for use only in specific editions of SQL Server. For a list of features that are supported by the editions of SQL Server, see Features Supported by the Editions of SQL Server 2012 (http://go.microsoft.com/fwlink/?linkid=232473).

Setting Algorithm Parameters

The following table describes the parameters that you can use with the Microsoft Decision Trees algorithm.

COMPLEXITY_PENALTY
Controls the growth of the decision tree. A low value increases the number of splits, and a high value decreases the number of splits. The default value is based on the number of attributes for a particular model, as described in the following list:

- For 1 through 9 attributes, the default is 0.5.
- For 10 through 99 attributes, the default is 0.9.
- For 100 or more attributes, the default is 0.99.

FORCE_REGRESSOR

Forces the algorithm to use the specified columns as regressors, regardless of the importance of the columns as calculated by the algorithm. This parameter is only used for decision trees that are predicting a continuous attribute.

> **NOTE**
> By setting this parameter, you force the algorithm to try to use the attribute as a regressor. However, whether the attribute is actually used as a regressor in the final model depends on the results of analysis. You can find out which columns were used as regressors by querying the model content.

[Available only in some editions of SQL Server]

MAXIMUM_INPUT_ATTRIBUTES
Defines the number of input attributes that the algorithm can handle before it invokes feature selection.

The default is 255.

Set this value to 0 to turn off feature selection.

[Available only in some editions of SQL Server]

MAXIMUM_OUTPUT_ATTRIBUTES
Defines the number of output attributes that the algorithm can handle before it invokes feature selection.

The default is 255.

Set this value to 0 to turn off feature selection.

[Available only in some editions of SQL Server]

MINIMUM_SUPPORT
Determines the minimum number of leaf cases that is required to generate a split in the decision tree.

The default is 10.

You may need to increase this value if the dataset is very large, to avoid overtraining.

SCORE_METHOD
Determines the method that is used to calculate the split score. The following options are available:

ID	NAME
1	Entropy
3	Bayesian with K2 Prior
4	Bayesian Dirichlet Equivalent (BDE) with uniform prior (default)

The default is 4, or BDE.

For an explanation of these scoring methods, see Feature Selection.

SPLIT_METHOD
Determines the method that is used to split the node. The following options are available:

ID	NAME
1	**Binary:** Indicates that regardless of the actual number of values for the attribute, the tree should be split into two branches.
2	**Complete:** Indicates that the tree can create as many splits as there are attribute values.
3	**Both:** Specifies that Analysis Services can determine whether a binary or complete split should be used to produce the best results.

The default is 3.

Modeling Flags

The Microsoft Decision Trees algorithm supports the following modeling flags. When you create the mining structure or mining model, you define modeling flags to specify how values in each column are handled during analysis. For more information, see Modeling Flags (Data Mining).

MODELING FLAG	DESCRIPTION
MODEL_EXISTENCE_ONLY	Means that the column will be treated as having two possible states: **Missing** and **Existing**. A null is a missing value. Applies to mining model columns.
NOT NULL	Indicates that the column cannot contain a null. An error will result if Analysis Services encounters a null during model training. Applies to mining structure columns.

Regressors in Decision Tree Models

Even if you do not use the Microsoft Linear Regression algorithm, any decision tree model that has continuous numeric inputs and outputs can potentially include nodes that represent a regression on a continuous attribute.

You do not need to specify that a column of continuous numeric data represents a regressor. The Microsoft Decision Trees algorithm will automatically use the column as a potential regressor and partition the dataset into regions with meaningful patterns even if you do not set the REGRESSOR flag on the column.

However, you can use the FORCE_REGRESSOR parameter to guarantee that the algorithm will use a particular regressor. This parameter can be used only with the Microsoft Decision Trees and Microsoft Linear Regression algorithms. When you set the modeling flag, the algorithm will try to find regression equations of the form `a*C1 + b*C2 + ...` to fit the patterns in the nodes of the tree. The sum of the residuals is calculated, and if the deviation is too great, a split is forced in the tree.

For example, if you are predicting customer purchasing behavior using **Income** as an attribute, and set the REGRESSOR modeling flag on the column, the algorithm will first try to fit the **Income** values by using a standard regression formula. If the deviation is too great, the regression formula is abandoned and the tree will be split on another attribute. The decision tree algorithm will then try to fit a regressor for income in each of the branches after the split.

Requirements

A decision tree model must contain a key column, input columns, and at least one predictable column.

Input and Predictable Columns

The Microsoft Decision Trees algorithm supports the specific input columns and predictable columns that are listed in the following table. For more information about what the content types mean when used in a mining model, see Content Types (Data Mining).

COLUMN	CONTENT TYPES
Input attribute	Continuous, Cyclical, Discrete, Discretized, Key, Ordered, Table
Predictable attribute	Continuous, Cyclical, Discrete, Discretized, Ordered, Table

> **NOTE**
> Cyclical and Ordered content types are supported, but the algorithm treats them as discrete values and does not perform special processing.

See Also

Microsoft Decision Trees Algorithm
Decision Trees Model Query Examples
Mining Model Content for Decision Tree Models (Analysis Services - Data Mining)

Mining Model Content for Decision Tree Models (Analysis Services - Data Mining)

5/16/2018 • 18 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes mining model content that is specific to models that use the Microsoft Decision Trees algorithm. For a general explanation of mining model content for all model types, see Mining Model Content (Analysis Services - Data Mining). It is important to remember that The Microsoft Decision Trees algorithm is a hybrid algorithm that can create models with very different functions: a decision tree can represent associations, rules, or even linear regression. The structure of the tree is essentially the same, but how you interpret the information will depend on the purpose for which you created the model.

Understanding the Structure of a Decision Trees Model

A decision trees model has a single parent node that represents the model and its metadata. Underneath the parent node are independent trees that represent the predictable attributes that you select. For example, if you set up your decision tree model to predict whether customers will purchase something, and provide inputs for gender and income, the model would create a single tree for the purchasing attribute, with many branches that divide on conditions related to gender and income.

However, if you then add a separate predictable attribute for participation in a customer rewards program, the algorithm will create two separate trees under the parent node. One tree contains the analysis for purchasing, and another tree contains the analysis for the customer rewards program. If you use the Decision Trees algorithm to create an association model, the algorithm creates a separate tree for each product that is being predicted, and the tree contains all the other product combinations that contribute towards selection of the target attribute.

> **NOTE**
>
> If your model includes multiple trees, you can view only one tree at a time in the **Microsoft Tree Viewer**. However, in the **Generic Content Tree Viewer** , all trees in the same model are displayed at the same time.

The tree for each predictable attribute contains information that describes how the input columns that you choose affect the outcome of that particular predictable attribute. Each tree is headed by a node (NODE_TYPE = 9) that contains the predictable attribute, followed by a series of nodes (NODE_TYPE = 10) that represent the input attributes. An attribute corresponds to either a case-level column or values of nested table columns, which are generally the values in the **Key** column of the nested table.

Interior and leaf nodes represent split conditions. A tree can split on the same attribute multiple times. For example, the **TM_DecisionTree** model might split on [Yearly Income] and [Number of Children], and then split again on [Yearly Income] further down the tree.

The Microsoft Decision Trees algorithm can also contain linear regressions in all or part of the tree. If the attribute that you are modeling is a continuous numeric data type, the model can create a regression tree node

(NODE_TYPE = 25) wherever the relationship between the attributes can be modeled linearly. In this case, the node contains a regression formula.

However, if the predictable attribute has discrete values, or if numeric values have been bucketed or discretized, the model always creates a classification tree (NODE_TYPE =2). A classification tree can have multiple branches or interior tree nodes (NODE_TYPE =3) for each value of the attribute. However, the split is not necessarily on each value of the attribute.

The Microsoft Decision Trees algorithm does not allow continuous data types as inputs; therefore, if any columns have a continuous numeric data type, the values are discretized. The algorithm performs its own discretization at the point of a split for all continuous attributes.

> **NOTE**
> Analysis Services automatically chooses a method for bucketing continuous attributes; however, you can control how continuous values in the inputs are discretized by setting the content type of the mining structure column to **Discretized** and then setting the DiscretizationBucketCount or DiscretizationMethod property.

Top

Model Content for a Decision Trees Model

This section provides details and examples only for those columns in the mining model content that have particular relevance for decision trees models. For information about general-purpose columns in the schema rowset, and explanations of mining model terminology, see Mining Model Content (Analysis Services - Data Mining).

MODEL_CATALOG
Name of the database where the model is stored.

MODEL_NAME
Name of the model.

ATTRIBUTE_NAME
Name of the attribute that corresponds to this node.

NODE_NAME
Always same as NODE_UNIQUE_NAME.

NODE_UNIQUE_NAME
A unique identifier for the node within the model. This value cannot be changed.

For decision tree models, the unique names follow the following convention, which does not apply to all algorithms:

The child nodes of any particular node will all have the same hexadecimal prefix, followed by another hexadecimal number that represents the sequence of the child node within the parent. You can use the prefixes to infer a path.

NODE_TYPE
In decision tree models, the following types of nodes are created:

NODE TYPE	DESCRIPTION
1 (Model)	Root node for model.

NODE TYPE	DESCRIPTION
2 (Tree)	Parent node for classification trees in the model. Labeled **"All"**.
3 (Interior)	Head of interior branch, found within in a classification tree or regression tree.
4 (Distribution)	Leaf node, found within a classification tree or regression tree.
25 (Regression tree)	Parent node for regression tree within the model. Labeled as **"All"**.

NODE_CAPTION
A friendly name for display purposes.

When you create a model, the value of NODE_UNIQUE_NAME is automatically used as the caption. However, you can change the value for NODE_CAPTION to update the display name for the cluster, either programmatically or by using the viewer. The caption is automatically generated by the model. The content of the caption depends on the type of model, and the node type.

In a decision trees model, the NODE_CAPTION and the NODE_DESCRIPTION have different information, depending on the level in the tree. For more information and examples, see Node Caption and Node Description.

CHILDREN_CARDINALITY
An estimate of the number of children that the node has.

Parent node Indicates the number of predictable attributes that were modeled. A tree is created for each predictable attribute.

Tree node The **All** node for each tree tells you how many values were used for the target attribute.

- If the target attribute is discrete, the value equals the number of distinct values plus 1 for the **Missing** state.
- If the predictable attribute is continuous, the value tells you how many buckets were used to model the continuous attribute.

Leaf nodes Always 0.

PARENT_UNIQUE_NAME
The unique name of the node's parent. NULL is returned for any nodes at the root level.

NODE_DESCRIPTION
A description of the node.

In a decision trees model, the NODE_CAPTION and the NODE_DESCRIPTION have different information, depending on the level in the tree.

For more information and examples, see Node Caption and Node Description.

NODE_RULE
An XML description of the rule that describes the path to the current node from its immediate parent node.

For more information and examples, see Node Rule and Marginal Rule.

MARGINAL_RULE
An XML description of the rule that describes the path from the model parent node to the current node.

For more information, see Node Rule and Marginal Rule.

NODE_PROBABILITY
The probability associated with this node.

For more information, see Probability.

MARGINAL_PROBABILITY
The probability of reaching the node from the parent node.

For more information, see Probability.

NODE_DISTRIBUTION
A table that contains the probability histogram of the node. The information in this table differs depending on whether the predictable attribute is a continuous or discrete variable.

Model root node This table is empty.

(All) node Contains a summary for the model as a whole.

Interior node Contains aggregated statistics for its leaf nodes.

Leaf node Contains support and probability for the predicted outcomes given all the conditions in the path leading to the current leaf node.

Regression node Contains regression formula that represents the relationship between the inputs and the predictable attribute.

For more information, see Node Distribution for Discrete Attributes and Node Distribution for Continuous Attributes.

NODE_SUPPORT
The number of cases that support this node.

MSOLAP_MODEL_COLUMN
Indicates the column that contains the predictable attribute.

MSOLAP_NODE_SCORE
Displays a score associated with the node. For more information, see Node Score.

MSOLAP_NODE_SHORT_CAPTION
A label used for display purposes.

Remarks

A decision trees model does not have a separate node that stores statistics for the entire model, unlike the marginal statistics node found in a Naive Bayes or neural network model. Instead, the model creates a separate tree for each predictable attribute, with an (All) node at the top of the tree. Each tree is independent of the others. If your model contains only one predictable attribute, there is only one tree, and therefore only one (All) node.

Each tree that represents an output attribute is additionally subdivided into interior branches (NODE_TYPE = 3) that represent splits. Each of these trees contains statistics about the distribution of the target attribute. In addition, each leaf node (NODE_TYPE = 4) contains statistics that describe input attributes and their values, together with the number of cases in support of each attribute-value pair. Therefore, in any branch of a decision tree, you can view the probabilities or the distribution of data easily without having to query the source data. Each level of the tree necessarily represents the sum of its immediate child nodes.

For examples of how to retrieve these statistics, see Decision Trees Model Query Examples.

Top

Example of Decision Tree Structure

To understand how a decision tree works, consider an example, such as the AdventureWorks bike buyer scenario. Assuming that the predictable attribute is customer purchases, the decision trees algorithm tries to find one column of data, among all the inputs that you provided, that most effectively detects the customers that are likely to purchase a bike and those who are unlikely to buy a bike. For example, the model might find that Age is the best indicator of purchasing behavior. Specifically, that the customers over the age of 30 are very likely to purchase a bike, and all other customers are unlikely to make a purchase. In this scenario, the model creates a *split* on the Age attribute. That means that the tree divides into two branches, one containing customers over the age of 30, and the other containing customers under 30. The new branches are represented in the model structure as two new interior trees (NODE_TYPE = 3).

For each branch, the model continues to look for additional attributes to use in differentiating customers. If there is insufficient evidence in the data to continue creating subgroups of customers, the model stops building the tree. The model will also stop building the tree whenever the number of cases in the node is too small to continue, regardless of how good the split is, or if the value is null or missing. By stopping the growth of the tree early, you prevent the model from training too closely to one particular set of data.

Each interior tree node contains leaf nodes that provide a breakdown of the outcomes given the current classification results. For example, you might have an interior node that represents Age >= 30 and Gender = Male. The node for this group shows you how many customers in this category purchased or did not purchase something. For example, the classification might contain the following tree splits:

INTERIOR TREE	SPLIT
Age >= 30	Age >= 30 and Gender = Male
	Age >= 30 and Gender = Female
Age < 30	Age < 30 and Gender = Male
	Age < 30 and Gender = Female

When you use a decision tree model for prediction, the model takes the attributes that you provide to it as arguments and follows the path of the attributes down through the tree. In general, all predictions go to a leaf, and the interior nodes are used only for classification.

A leaf node always has a NODE_TYPE of 4 (Distribution) and contains a histogram that tells the probability of each outcome (purchase or not purchase) given the attributes you provide. For example, if you ask for a prediction for a new customer who is a male over 60, the model will look up the corresponding node (Age > 30 and Gender = Male) and then return the probability for the outcome that you specify. These probabilities are stored in the NODE_DISTRIBUTION table for the node.

If the predictable attribute is a continuous number, the algorithm tries to create a regression formula that models the relationship between the predictable attribute and the inputs.

Top

Node Caption and Node Description

In a decision tree model, the node caption and node description contain similar information. However, the node description is more complete and contains more information as you move closer to the leaf nodes. Both the node caption and node description are localized strings.

NODE_CAPTION	Displays the attribute that distinguishes that particular node relative to the parent node. The node caption defines a sub-segment of the population based the split condition. For example, if the split was on [Age] and it was a three-way split, the node captions for the three child nodes might be "[Age] < 40", "40 <= [Age] < 50", "[Age] >= 50".
NODE_DESCRIPTION	Contains a full list of the attributes that distinguish that node from other nodes, starting from the model parent node. For example, Product name = Apple and Color = Red.

Top

Node Rule and Marginal Rule

The NODE_RULE and MARGINAL_RULE columns contain the same information as the NODE_CAPTION and NODE_DESCRIPTION columns, but represent the information as XML fragments. The node rule is an XML version of the full path, whereas the marginal rule indicates the most recent split.

The attribute represented by the XML fragment can be either simple or complex. A simple attribute contains the name of the model column, and the value of the attribute. If the model column contains a nested table, the nested table attribute is represented as a concatenation of the table name, the key value, and the attribute.

> **NOTE**
>
> SQL Server Analysis Services supports version 2.0 of the PMML standard, with extensions to support the use of nested table. If your data contains nested tables and you generate a PMML version of the model, all elements in the model that include the predicates are marked as an extension.

Top

Node Distribution for Discrete Attributes

In a decision trees model, the NODE_DISTRIBUTION table contains useful statistics. However, the type of statistics depends on whether the tree predicts a discrete or continuous attribute. This section describes the meaning of the node distribution statistics for discrete attributes.

Attribute Name and Attribute Value

In a classification tree, the attribute name always contains the name of the predictable column. This value tells you what the tree predicts. Because a single tree always represents a single predictable attribute, this value is repeated throughout the tree.

For a discrete data type, the attribute value field lists the possible values of the predictable column, plus the **Missing** value.

Support

The support value for each node tells you how many cases are included in this node. At the (All) level, you should see the complete count of cases that were used to train the model. For each split in the tree, the support value is the count of cases that were grouped into that node of the tree. The sum of cases in the leaf nodes necessarily equals the count of cases in the parent node of the tree.

For nodes that represent continuous attributes, the presence of nulls in the data might lead to some counterintuitive results. For example, if there are m cases, a mean value would be calculated as sum(all cases)/n, where n is a number less than m, and m-n indicates the count of cases with missing values. Support is also represented as n.

Probability

The probability associated with each node tells you the probability that any case in the whole data set would end up in this particular node. Probability scores are computed both for the tree as a whole, and for the immediate split.

For example, the following table shows a very simple model, with 100 cases.

INTERIOR TREE	CASES	LEAF NODE	CASES	PROBABILITY RELATIVE TO PARENT NODE	PROBABILITY RELATIVE TO TOP NODE
Age >= 30	60	Age >= 30 and Gender = Male	50	50/60 = .83	50/100 = .5
		Age >= 30 and Gender = Female	10	10/60 = .16	10/100 = .10
Age < 30	40	Age < 30 and Gender = Male	30	30/40 = .75	30/100 = .30
		Age < 30 and Gender = Female	10	10/40 = .25	10/100 = .10

A small adjustment is made in all models to account for possible missing values. For continuous attributes, each value or range of values is represented as a state (for example, Age <30, Age = 30, and Age >30) and the probabilities are calculated as follows: state exists (value = 1), some other state exists (value = 0), state is **Missing**. For more information about how probabilities are adjusted to represent missing values, see Missing Values (Analysis Services - Data Mining).

The probabilities for each node are calculated almost directly from the distribution, as follows:

Probability = (support for state + support for prior state) / (node support plus the prior node support)

Analysis Services uses probabilities for each node to compare the stored probability with the prior probability to determine whether the path from the parent to the child node indicates a strong inference.

When making predictions, the probability of the distribution must be balanced with the probability of the node, to smoothen the probabilities. For example, if a split in the tree separates cases by a ratio of 9000/1000, the tree is very unbalanced. As a result, a prediction coming from the small branch should not carry the same weight as a prediction coming from a branch with many cases.

Variance

Variance is a measure of how scattered values in a sample are, given an expected distribution. For discrete values, the variance is 0 by definition.

For information about how variance is calculated for continuous values, see Mining Model Content for Linear Regression Models (Analysis Services - Data Mining).

Value Type

The value type column provides information about the meaning of the numeric value provided in the other columns in the NODE_DISTRIBUTION table. You can use the value type in queries to retrieve specific rows from the nested tables. For examples, see Decision Trees Model Query Examples.

Of the types in the MiningValueType enumeration, the following are used in classification trees.

VALUE TYPE	DESCRIPTION
1 (Missing)	Indicates a count, probability, or other statistic related to missing values.

VALUE TYPE	DESCRIPTION
4 (Discrete)	Indicates a count, probability, or other statistic related to a discrete or discretized value.

If the model includes a continuous predictable attribute, the tree might also contain value types that are unique to regression formulas. For a list of the value types that are used in regression trees, see Mining Model Content for Linear Regression Models (Analysis Services - Data Mining).

Node Score

The node score represents slightly different information at each level of the tree. In general, the score is a numeric value that tells you how good a split was achieved by splitting on the condition. The value is represented as a double, where a higher value is better.

By definition, the model node and all leaf nodes have a node score of 0.

For the (All) node that represents the top of each tree, the MSOLAP_NODE_SCORE column contains the best split score in the whole tree.

For all other nodes in the tree (except leaf nodes), the score for each node represents the best split score for the current node, minus the split score for the parent node. Typically, the split score for a parent node should always be better than the split score on any one of its child nodes. That is because a decision trees model ideally splits on the most important attributes first.

There are many ways of calculating a score for a split, depending on the algorithm parameter you choose. A discussion of how the scores are calculated for each of the scoring methods is beyond the scope of this topic. For more information, see "Learning Bayesian Networks: The Combination of Knowledge and Statistical Data", on the Microsoft Research Web site.

> **NOTE**
> If you create a decision trees model that has both continuous and discrete predictable attributes, you will see completely different scores in the (All) nodes that represent each tree type. Each model should be considered independently, and the methods used for scoring regression are completely different from those used for scoring classification. The node score values cannot be compared.

Top

Regression Nodes within a Decision Tree Model

If a decision trees model contains a predictable attribute with continuous numeric data, the Microsoft Decision Trees algorithm seeks to find areas in the data where the relationship between the predicted state and the input variables is linear. If the algorithm is successful in finding a linear relationship, it creates a special tree (NODE_TYPE = 25) that represents a linear regression. These regression tree nodes are more complex than nodes that represent discrete values.

In general, a regression maps the changes in the continuous dependent (predictable variable) as a function of changes in the inputs. If the dependent variable has any continuous inputs, and the relationship between the input and predicted value is stable enough to be computed as a line graph, the node for the regression contains a formula.

However, if the relationship between the input and predicted value is *nonlinear*, a split is created instead, just like a standard decision tree. For example, assume that A is the predictable attribute, and B and C are the inputs, where C is a continuous value type. If the relationship between A and C is fairly stable in parts of the data, but unstable in others, the algorithm will create splits to represent the different areas of the data.

SPLIT CONDITION	RESULT IN NODE
if n < 5	Relationship can be expressed as equation 1
if n between 5 and 10	No equation
if n > 10	Relationship can be expressed as equation 2

For more information about regression nodes, see Mining Model Content for Linear Regression Models (Analysis Services - Data Mining).

See Also

Mining Model Content (Analysis Services - Data Mining)
Data Mining Model Viewers
Data Mining Queries
Microsoft Decision Trees Algorithm

Decision Trees Model Query Examples

5/16/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a query against a data mining model, you can create a content query, which provides details about the patterns discovered in analysis, or you can create a prediction query, which uses the patterns in the model to make predictions for new data. For example, a content query for a decision trees model might provide statistics about the number of cases at each level of the tree, or the rules that differentiate between cases. Alternatively, a prediction query maps the model to new data in order to generate recommendations, classifications, and so forth. You can also retrieve metadata about the model by using a query.

This section explains how to create queries for models that are based on the Microsoft Decision Trees algorithm.

Content Queries

Retrieving Model Parameters from the Data Mining Schema Rowset

Getting Details about Trees in the Model by Using DMX

Retrieving Subtrees from the Model

Prediction Queries

Returning Predictions with Probabilities

Predicting Associations from a Decision Trees Model

Retrieving a Regression Formula from a Decision Trees Model

Finding Information about a Decision Trees Model

To create meaningful queries on the content of a decision trees model, you should understand the structure of the model content, and which node types store what kind of information. For more information, see Mining Model Content for Decision Tree Models (Analysis Services - Data Mining).

Sample Query 1: Retrieving Model Parameters from the Data Mining Schema Rowset

By querying the data mining schema rowset, you can find metadata about the model, such as when it was created, when the model was last processed, the name of the mining structure that the model is based on, and the name of the column used as the predictable attribute. You can also return the parameters that were used when the model was first created.

```
select MINING_PARAMETERS
from $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = 'TM_Decision Tree'
```

Sample results:

MINING_PARAMETERS

COMPLEXITY_PENALTY=0.5, MAXIMUM_INPUT_ATTRIBUTES=255,MAXIMUM_OUTPUT_ATTRIBUTES=255,MINIMUM_SUPPORT=10,SCORE_METHOD=4,SPLIT_METHOD=3,FORCE_REGRESSOR=

Sample Query 2: Returning Details about the Model Content by Using DMX

The following query returns some basic information about the decision trees that were created when you built the model in the Basic Data Mining Tutorial. Each tree structure is stored in its own node. Because this model contains a single predictable attribute, there is only one tree node. However, if you create an association model by using the Decision Trees algorithm, there might be hundreds of trees, one for each product.

This query returns all the nodes of type 2, which are the top level nodes of a tree that represents a particular predictable attribute.

> **NOTE**
> The column, CHILDREN_CARDINALITY, must be enclosed in brackets to distinguish it from the MDX reserved keyword of the same name.

```
SELECT MODEL_NAME, NODE_NAME, NODE_CAPTION,
NODE_SUPPORT, [CHILDREN_CARDINALITY]
FROM TM_DecisionTrees.CONTENT
WHERE NODE_TYPE = 2
```

Example results:

MODEL_NAME	NODE_NAME	NODE_CAPTION	NODE_SUPPORT	CHILDREN_CARDINALITY
TM_DecisionTree	000000001	All	12939	5

What do these results tell you? In a decision trees model, the cardinality of a particular node tells you how many immediate children that node has. The cardinality for this node is 5, meaning that the model divided the target population of potential bike buyers into 5 subgroups.

The following related query returns the children for these five subgroups, together with the distribution of attributes and values in the child nodes. Because statistics such as support, probability, and variance are stored in the nested table, NODE_DISTRIBUTION, this example uses the `FLATTENED` keyword to output the nested table columns.

> **NOTE**
> The nested table column, SUPPORT, must be enclosed in brackets to distinguish it from the reserved keyword of the same name.

```
SELECT FLATTENED NODE_NAME, NODE_CAPTION,
(SELECT ATTRIBUTE_NAME, ATTRIBUTE_VALUE, [SUPPORT]
FROM NODE_DISTRIBUTION) AS t
FROM TM_DecisionTree.CONTENT
WHERE [PARENT_UNIQUE_NAME] = '000000001'
```

Example results:

NODE_NAME	NODE_CAPTION	T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE	SUPPORT
00000000100	Number Cars Owned = 0	Bike Buyer	Missing	0

NODE_NAME	NODE_CAPTION	T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE	SUPPORT
00000000100	Number Cars Owned = 0	Bike Buyer	0	1067
00000000100	Number Cars Owned = 0	Bike Buyer	1	1875
00000000101	Number Cars Owned = 3	Bike Buyer	Missing	0
00000000101	Number Cars Owned = 3	Bike Buyer	0	678
00000000101	Number Cars Owned = 3	Bike Buyer	1	473

From these results, you can tell that of the customers who bought a bike ([Bike Buyer] = 1), 1067 customers had 0 cars and 473 customers had 3 cars.

Sample Query 3: Retrieving Subtrees from the Model

Suppose you wanted to discover more about the customers who did buy a bike. You can view additional detail for any of the sub-trees by using the IsDescendant (DMX) function in the query, as shown in the following example. The query returns the count of bike purchasers by retrieving leaf nodes (NODE_TYPE = 4) from the tree that contains customers who are over 42 years of age. The query restricts rows from the nested table to those where Bike Buyer = 1.

```
SELECT FLATTENED NODE_NAME, NODE_CAPTION,NODE_TYPE,
(
SELECT [SUPPORT] FROM NODE_DISTRIBUTION WHERE ATTRIBUTE_NAME = 'Bike Buyer' AND ATTRIBUTE_VALUE = '1'
) AS t
FROM TM_DecisionTree.CONTENT
WHERE ISDESCENDANT('0000000010001')
AND NODE_TYPE = 4
```

Example results:

NODE_NAME	NODE_CAPTION	T.SUPPORT
000000001000100	Yearly Income >= 26000 and < 42000	266
00000000100010100	Total Children = 3	75
0000000010001010100	Number Children At Home = 1	75

Making Predictions using a Decision Trees Model

Because decision trees can be used for various tasks, including classification, regression, and even association, when you create a prediction query on a decision tree model you have many options available to you. You must understand the purpose for which the model was created to understand the results of prediction. The following query samples illustrate three different scenarios:

- Returning a prediction for a classification model, together with the probability of the prediction being correct, and then filtering the results by the probability;

- Creating a singleton query to predict associations;
- Retrieving the regression formula for a part of a decision tree where the relationship between the input and output is linear.

Sample Query 4: Returning Predictions with Probabilities

The following sample query uses the decision tree model that was created in the Basic Data Mining Tutorial. The query passes in a new set of sample data, from the table dbo.ProspectiveBuyers in AdventureWorks2012 DW, to predict which of the customers in the new data set will purchase a bike.

The query uses the prediction function PredictHistogram (DMX), which returns a nested table that contains useful information about the probabilities discovered by the model. The final WHERE clause of the query filters the results to return only those customers who are predicted as likely to buy a bike, with a probability greater than 0%.

```
SELECT
  [TM_DecisionTree].[Bike Buyer],
  PredictHistogram([Bike Buyer]) as Results
From
  [TM_DecisionTree]
PREDICTION JOIN
  OPENQUERY([Adventure Works DW Multidimensional 2012],
    'SELECT
      [FirstName],
      [LastName],
      [MaritalStatus],
      [Gender],
      [YearlyIncome],
      [TotalChildren],
      [NumberChildrenAtHome],
      [HouseOwnerFlag],
      [NumberCarsOwned]
    FROM
      [dbo].[ProspectiveBuyer]
    ') AS t
ON
  [TM_DecisionTree].[First Name] = t.[FirstName] AND
  [TM_DecisionTree].[Last Name] = t.[LastName] AND
  [TM_DecisionTree].[Marital Status] = t.[MaritalStatus] AND
  [TM_DecisionTree].[Gender] = t.[Gender] AND
  [TM_DecisionTree].[Yearly Income] = t.[YearlyIncome] AND
  [TM_DecisionTree].[Total Children] = t.[TotalChildren] AND
  [TM_DecisionTree].[Number Children At Home] = t.[NumberChildrenAtHome] AND
  [TM_DecisionTree].[House Owner Flag] = t.[HouseOwnerFlag] AND
  [TM_DecisionTree].[Number Cars Owned] = t.[NumberCarsOwned]
WHERE [Bike Buyer] = 1
AND PredictProbability([Bike Buyer]) >'.05'
```

By default, Analysis Services returns nested tables with the column label, **Expression**. You can change this label by aliasing the column that is returned. If you do this, the alias (in this case, **Results**) is used as both the column heading and as the value in the nested table. You must expand the nested table to see the results.

Example results with **Bike Buyer** = 1:

BIKE BUYER	$SUPPORT	$PROBABILITY	$ADJUSTEDPROBABILITY	$VARIANCE	$STDEV
1	2540	0.634849242045644	0.013562168281562	0	0

BIKE BUYER	$SUPPORT	$PROBABILITY	$ADJUSTEDPROBABILITY	$VARIANCE	$STDEV
0	1460	0.364984174579377	0.00661336932550915	0	0
	0	0.000166583374979177	0.000166583374979177	0	0

If your provider does not support hierarchical rowsets, such as those shown here, you can use the FLATTENED keyword in the query to return the results as a table that contains nulls in place of the repeated column values. For more information, see Nested Tables (Analysis Services - Data Mining) or Understanding the DMX Select Statement.

Sample Query 5: Predicting Associations from a Decision Trees Model

The following sample query is based on the Association mining structure. To follow along with this example, you can add a new model to this mining structure, and select Microsoft Decision Trees as the algorithm. For more information on how to create the Association mining structure, see Lesson 3: Building a Market Basket Scenario (Intermediate Data Mining Tutorial).

The following sample query is a singleton query, which you can create easily in SQL Server Data Tools (SSDT) by choosing fields and then selecting values for those fields from a drop-down list.

```
SELECT PredictAssociation([DT_Association].[v Assoc Seq Line Items],3)
FROM
  [DT_Association]
NATURAL PREDICTION JOIN
(SELECT (SELECT 'Patch kit' AS [Model]) AS [v Assoc Seq Line Items]) AS t
```

Expected results:

MODEL
Mountain-200
Mountain Tire Tube
Touring Tire Tube

The results tell you the three best products to recommend to customers who have purchased the Patch Kit product. You can also provide multiple products as input when you make recommendations, either by typing in values, or by using the **Singleton Query Input** dialog box and adding or removing values. The following sample query shows how the multiple values are provided, upon which to make a prediction. Values are connected by a UNION clause in the SELECT statement that defines the input values.

```
SELECT PredictAssociation([DT_Association].[v Assoc Seq Line Items],3)
From
  [DT_Association]
NATURAL PREDICTION JOIN
(SELECT (SELECT 'Racing Socks' AS [Model]
  UNION SELECT 'Women''s Mountain Shorts' AS [Model]) AS [v Assoc Seq Line Items]) AS t
```

Expected results:

MODEL
Long-Sleeve Logo Jersey
Mountain-400-W
Classic Vest

Sample Query 6: Retrieving a Regression Formula from a Decision Trees Model

When you create a decision tree model that contains a regression on a continuous attribute, you can use the regression formula to make predictions, or you can extract information about the regression formula. For more information about queries on regression models, see Linear Regression Model Query Examples.

If a decision trees model contains a mixture of regression nodes and nodes that split on discrete attributes or ranges, you can create a query that returns only the regression node. The NODE_DISTRIBUTION table contains the details of the regression formula. In this example, the columns are flattened and the NODE_DISTRIBUTION table is aliased for easier viewing. However, in this model, no regressors were found to relate Income with other continuous attributes. In such cases, Analysis Services returns the mean value of the attribute and the total variance in the model for that attribute.

```
SELECT FLATTENED NODE_DISTRIBUTION AS t
FROM DT_Predict. CONTENT
WHERE NODE_TYPE = 25
```

Example results:

T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE	T.SUPPORT	T.PROBABILITY	T.VARIANCE	T.VALUETYPE
Yearly Income	Missing	0	0.000457142857142857	0	1
Yearly Income	57220.8876687257	17484	0.999542857142857	1041275619.52776	3
	57220.8876687257	0	0	1041216662.54387	11

For more information about the value types and the statistics used in regression models, see Mining Model Content for Linear Regression Models (Analysis Services - Data Mining).

List of Prediction Functions

All Microsoft algorithms support a common set of functions. However, the Microsoft Decision Trees algorithm supports the additional functions listed in the following table.

Prediction Function	Usage
IsDescendant (DMX)	Determines whether one node is a child of another node in the model.

IsInNode (DMX)	Indicates whether the specified node contains the current case.
PredictAdjustedProbability (DMX)	Returns the weighted probability.
PredictAssociation (DMX)	Predicts membership in an associative dataset.
PredictHistogram (DMX)	Returns a table of values related to the current predicted value.
PredictNodeId (DMX)	Returns the Node_ID for each case.
PredictProbability (DMX)	Returns probability for the predicted value.
PredictStdev (DMX)	Returns the predicted standard deviation for the specified column.
PredictSupport (DMX)	Returns the support value for a specified state.
PredictVariance (DMX)	Returns the variance of a specified column.

For a list of the functions that are common to all Microsoft algorithms, see General Prediction Functions (DMX). For the syntax of specific functions, see Data Mining Extensions (DMX) Function Reference.

See Also

Data Mining Queries
Microsoft Decision Trees Algorithm
Microsoft Decision Trees Algorithm Technical Reference
Mining Model Content for Decision Tree Models (Analysis Services - Data Mining)

Microsoft Linear Regression Algorithm

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Linear Regression algorithm is a variation of the Microsoft Decision Trees algorithm that helps you calculate a linear relationship between a dependent and independent variable, and then use that relationship for prediction.

The relationship takes the form of an equation for a line that best represents a series of data. For example, the line in the following diagram is the best possible linear representation of the data.

Each data point in the diagram has an error associated with its distance from the regression line. The coefficients a and b in the regression equation adjust the angle and location of the regression line. You can obtain the regression equation by adjusting a and b until the sum of the errors that are associated with all the points reaches its minimum.

There are other kinds of regression that use multiple variables, and also nonlinear methods of regression. However, linear regression is a useful and well-known method for modeling a response to a change in some underlying factor.

Example

You can use linear regression to determine a relationship between two continuous columns. For example, you can use linear regression to compute a trend line from manufacturing or sales data. You could also use the linear regression as a precursor to development of more complex data mining models, to assess the relationships among data columns.

Although there are many ways to compute linear regression that do not require data mining tools, the advantage of using the Microsoft Linear Regression algorithm for this task is that all the possible relationships among the variables are automatically computed and tested. You do not have to select a computation method, such as solving for least squares. However, linear regression might oversimplify the relationships in scenarios where multiple factors affect the outcome.

How the Algorithm Works

The Microsoft Linear Regression algorithm is a variation of the Microsoft Decision Trees algorithm. When you select the Microsoft Linear Regression algorithm, a special case of the Microsoft Decision Trees algorithm is invoked, with parameters that constrain the behavior of the algorithm and require certain input data types. Moreover, in a linear regression model, the whole data set is used for computing relationships in the initial pass, whereas a standard decision trees model splits the data repeatedly into smaller subsets or trees.

Data Required for Linear Regression Models

When you prepare data for use in a linear regression model, you should understand the requirements for the particular algorithm. This includes how much data is needed, and how the data is used. The requirements for this model type are as follows:

- **A single key column** Each model must contain one numeric or text column that uniquely identifies each record. Compound keys are not permitted.
- **A predictable column** Requires at least one predictable column. You can include multiple predictable attributes in a model, but the predictable attributes must be continuous numeric data types. You cannot use a datetime data type as a predictable attribute even if the native storage for the data is numeric.
- **Input columns** Input columns must contain continuous numeric data and be assigned the appropriate data type.

For more information, see the Requirements section of Microsoft Linear Regression Algorithm Technical Reference.

Viewing a Linear Regression Model

To explore the model, you use the **Microsoft Tree Viewer**. The tree structure for a linear regression model is very simple, with all the information about the regression equation contained in a single node. For more information, see Browse a Model Using the Microsoft Tree Viewer.

If you want to know more detail about the equation, you can also view the coefficients and other details by using the Microsoft Generic Content Tree Viewer.

For a linear regression model, the model content includes metadata, the regression formula, and statistics about the distribution of input values. For more information, see Mining Model Content for Linear Regression Models (Analysis Services - Data Mining).

Creating Predictions

After the model has been processed, the results are stored as a set of statistics together with the linear regression formula, which you can use to compute future trends. For examples of queries to use with a linear regression model, see Linear Regression Model Query Examples.

For general information about how to create queries against mining models, see Data Mining Queries.

In addition to creating a linear regression model by selecting the Microsoft Linear Regression algorithm, if the predictable attribute is a continuous numeric data type, you can create a decision tree model that contains regressions. In this case, the algorithm will split the data when it finds appropriate separation points, but for some regions of data, will create a regression formula instead. For more information about regression trees within a decision trees model, see Mining Model Content for Decision Tree Models (Analysis Services - Data Mining).

Remarks

- Does not support the use of Predictive Model Markup Language (PMML) to create mining models.
- Does not support the creation of data mining dimensions.
- Supports drillthrough.
- Supports the use of OLAP mining models.

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Microsoft Linear Regression Algorithm Technical Reference
Linear Regression Model Query Examples
Mining Model Content for Linear Regression Models (Analysis Services - Data Mining)

Microsoft Linear Regression Algorithm Technical Reference

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Linear Regression algorithm is a special version of the Microsoft Decision Trees algorithm that is optimized for modeling pairs of continuous attributes. This topic explains the implementation of the algorithm, describes how to customize the behavior of the algorithm, and provides links to additional information about querying models.

Implementation of the Linear Regression Algorithm

The Microsoft Decision Trees algorithm can be used for many tasks: linear regression, classification, or association analysis. To implement this algorithm for the purpose of linear regression, the parameters of the algorithm are controlled to restrict the growth of the tree and keep all data in the model in a single node. In other words, although linear regression is based on a decision tree, the tree contains only a single root and no branches: all data resides in the root node.

To accomplish this, the algorithm's *MINIMUM_LEAF_CASES* parameter is set to be greater than or equal to the total number of cases that the algorithm uses to train the mining model. With the parameter set in this way, the algorithm will never create a split, and therefore performs a linear regression.

The equation that represents the regression line takes the general form of y = ax + b, and is known as the regression equation. The variable Y represents the output variable, X represents the input variable, and a and b are adjustable coefficients. You can retrieve the coefficients, intercepts, and other information about the regression formula by querying the completed mining model. For more information, see Linear Regression Model Query Examples.

Scoring Methods and Feature Selection

All Analysis Services data mining algorithms automatically use feature selection to improve analysis and reduce processing load. The method used for feature selection in linear regression is the interestingness score, because the model supports only supports continuous columns. For reference, the following table shows the difference in feature selection for the Linear Regression algorithm and the Decision Trees algorithm.

ALGORITHM	METHOD OF ANALYSIS	COMMENTS
Linear Regression	Interestingness score	Default. Other feature selection methods that are available with the Decision Trees algorithm apply to discrete variables only and therefore are not applicable to linear regression models.

ALGORITHM	METHOD OF ANALYSIS	COMMENTS
Decision Trees	Interestingness score Shannon's Entropy Bayesian with K2 Prior Bayesian Dirichlet with uniform prior (default)	If any columns contain non-binary continuous values, the interestingness score is used for all columns, to ensure consistency. Otherwise, the default or specified method is used.

The algorithm parameters that control feature selection for a decision trees model are MAXIMUM_INPUT_ATTRIBUTES and MAXIMUM_OUTPUT.

Customizing the Linear Regression Algorithm

The Microsoft Linear Regression algorithm supports parameters that affect the behavior, performance, and accuracy of the resulting mining model. You can also set modeling flags on the mining model columns or mining structure columns to control the way that data is processed.

Setting Algorithm Parameters

The following table lists the parameters that are provided for the Microsoft Linear Regression algorithm.

PARAMETER	DESCRIPTION
MAXIMUM_INPUT_ATTRIBUTES	Defines the number of input attributes that the algorithm can handle before it invokes feature selection. Set this value to 0 to turn off feature selection. The default is 255.
MAXIMUM_OUTPUT_ATTRIBUTES	Defines the number of output attributes that the algorithm can handle before it invokes feature selection. Set this value to 0 to turn off feature selection. The default is 255.
FORCE_REGRESSOR	Forces the algorithm to use the indicated columns as regressors, regardless of the importance of the columns as calculated by the algorithm.

Modeling Flags

The Microsoft Linear Regression algorithm supports the following modeling flags. When you create the mining structure or mining model, you define modeling flags to specify how values in each column are handled during analysis. For more information, see Modeling Flags (Data Mining).

MODELING FLAG	DESCRIPTION
NOT NULL	Indicates that the column cannot contain a null. An error will result if Analysis Services encounters a null during model training. Applies to mining structure columns.

MODELING FLAG	DESCRIPTION
REGRESSOR	Indicates that the column contains continuous numeric values that should be treated as potential independent variables during analysis. Applies to mining model columns. Note: Flagging a column as a regressor does not ensure that the column will be used as a regressor in the final model.

Regressors in Linear Regression Models

Linear regression models are based on the Microsoft Decision Trees algorithm. However, even if you do not use the Microsoft Linear Regression algorithm, any decision tree model can contain a tree or nodes that represent a regression on a continuous attribute.

You do not need to specify that a continuous column represents a regressor. The Microsoft Decision Trees algorithm will partition the dataset into regions with meaningful patterns even if you do not set the REGRESSOR flag on the column. The difference is that when you set the modeling flag, the algorithm will try to find regression equations of the form `a*C1 + b*C2 + ...` to fit the patterns in the nodes of the tree. The sum of the residuals is calculated, and if the deviation is too great, a split is forced in the tree.

For example, if you are predicting customer purchasing behavior using income as an attribute, and set the REGRESSOR modeling flag on the [Income] column, the algorithm would first try to fit the values by using a standard regression formula. If the deviation is too great, the regression formula is abandoned and the tree would be split on some other attribute. The decision tree algorithm would then try to fit a regressor for income in each of the branches after the split.

You can use the FORCED_REGRESSOR parameter to guarantee that the algorithm will use a particular regressor. This parameter can be used with the Microsoft Decision Trees and Microsoft Linear Regression algorithms.

Requirements

A linear regression model must contain a key column, input columns, and at least one predictable column.

Input and Predictable Columns

The Microsoft Linear Regression algorithm supports the specific input columns and predictable columns that are listed in the following table. For more information about what the content types mean when used in a mining model, see Content Types (Data Mining).

COLUMN	CONTENT TYPES
Input attribute	Continuous, Cyclical, Key, Table, and Ordered
Predictable attribute	Continuous, Cyclical, and Ordered

> **NOTE**
>
> **Cyclical** and **Ordered** content types are supported, but the algorithm treats them as discrete values and does not perform special processing.

See Also

Microsoft Linear Regression Algorithm
Linear Regression Model Query Examples

Mining Model Content for Linear Regression Models (Analysis Services - Data Mining)

10/11/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes mining model content that is specific to models that use the Microsoft Linear Regression algorithm. For a general explanation of mining model content for all model types, see Mining Model Content (Analysis Services - Data Mining).

Understanding the Structure of a Linear Regression Model

A linear regression model has an extremely simple structure. Each model has a single parent node that represents the model and its metadata, and a regression tree node (NODE_TYPE = 25) that contains the regression formula for each predictable attribute.

Linear regression models use the same algorithm as Microsoft Decision Trees, but different parameters are used to constrain the tree, and only continuous attributes are accepted as inputs. However, because linear regression models are based on the Microsoft Decision Trees algorithm, linear regression models are displayed by using the Microsoft Decision Tree Viewer. For information, see Browse a Model Using the Microsoft Tree Viewer.

The next section explains how to interpret information in the regression formula node. This information applies not only to linear regression models, but also to decision trees models that contain regressions in a portion of the tree.

Model Content for a Linear Regression Model

This section provides detail and examples only for those columns in the mining model content that have particular relevance for linear regression.

For information about general-purpose columns in the schema rowset, see Mining Model Content (Analysis Services - Data Mining).

MODEL_CATALOG
Name of the database where the model is stored.

MODEL_NAME
Name of the model.

ATTRIBUTE_NAME
Root node: Blank

Regression node: The name of the predictable attribute.

NODE_NAME
Always same as NODE_UNIQUE_NAME.

NODE_UNIQUE_NAME
A unique identifier for the node within the model. This value cannot be changed.

NODE_TYPE
A linear regression model outputs the following node types:

NODE TYPE ID	TYPE	DESCRIPTION
25	Regression tree root	Contains the formula that describes the relationship between the input and output variable.

NODE_CAPTION
A label or a caption associated with the node. This property is primarily for display purposes.

Root node: Blank

Regression node: All.

CHILDREN_CARDINALITY
An estimate of the number of children that the node has.

Root node: Indicates the number of regression nodes. One regression node is created for each predictable attribute in the model.

Regression node: Always 0.

PARENT_UNIQUE_NAME
The unique name of the node's parent. NULL is returned for any nodes at the root level.

NODE_DESCRIPTION
A description of the node.

Root node: Blank

Regression node: All.

NODE_RULE
Not used for linear regression models.

MARGINAL_RULE
Not used for linear regression models.

NODE_PROBABILITY
The probability associated with this node.

Root node: 0

Regression node: 1

MARGINAL_PROBABILITY
The probability of reaching the node from the parent node.

Root node: 0

Regression node: 1

NODE_DISTRIBUTION
A nested table that provides statistics about the values in the node.

Root node: 0

Regression node: A table that contains the elements used to build the regression formula. A regression node contains the following value types:

VALUETYPE
1 (Missing)
3 (Continuous)
7 (Coefficient)
8 (Score Gain)
9 (Statistics)
11 (Intercept)

NODE_SUPPORT
The number of cases that support this node.

Root node: 0

Regression node: Count of training cases.

MSOLAP_MODEL_COLUMN
Name of predictable attribute.

MSOLAP_NODE_SCORE
Same as NODE_PROBABILITY

MSOLAP_NODE_SHORT_CAPTION
Label used for display purposes.

Remarks

When you create a model by using the Microsoft Linear Regression algorithm, the data mining engine creates a special instance of a decision trees model and supplies parameters that constrain the tree to contain all the training data in a single node. All continuous inputs are flagged and evaluated as potential regressors, but only those regressors that fit the data are retained as regressors in the final model. The analysis produces either a single regression formula for each regressor or no regression formula at all.

You can view the complete regression formula in the **Mining Legend**, by clicking the **(All)** node in the Microsoft Tree Viewer.

Also, when you create a decision trees model that includes a continuous predictable attribute, sometimes the tree has regression nodes that share the properties of regression tree nodes.

Node Distribution for Continuous Attributes

Most of the important information in a regression node is contained in the NODE_DISTRIBUTION table. The following example illustrates the layout of the NODE_DISTRIBUTION table. In this example, the Targeted Mailing mining structure has been used to create a linear regression model that predicts customer income based on age. The model is for the purpose of illustration only, because it can be built easily using the existing AdventureWorks2012 sample data and mining structure.

ATTRIBUTE_NAME	ATTRIBUTE_VALUE	SUPPORT	PROBABILITY	VARIANCE	VALUETYPE

ATTRIBUTE_NAME	ATTRIBUTE_VALUE	SUPPORT	PROBABILITY	VARIANCE	VALUETYPE
Yearly Income	Missing	0	0.000457142857142857	0	1
Yearly Income	57220.8876687257	17484	0.999542857142857	1041275619.52776	3
Age	471.687717702463	0	0	126.969442359327	7
Age	234.680904692439	0	0	0	8
Age	45.4269617936399	0	0	126.969442359327	9
	35793.5477381267	0	0	1012968919.28372	11

The NODE_DISTRIBUTION table contains multiple rows, each grouped by a variable. The first two rows are always value types 1 and 3, and describe the target attribute. The succeeding rows provide details about the formula for a particular *regressor*. A regressor is an input variable that has a linear relationship with the output variable. You can have multiple regressors, and each regressor will have a separate row for the coefficient (VALUETYPE = 7), score gain (VALUETYPE = 8), and statistics (VALUETYPE = 9). Finally, the table has a row that contains the intercept of the equation (VALUETYPE = 11).

Elements of the Regression Formula

The nested NODE_DISTRIBUTION table contains each element of the regression formula in a separate row. The first two rows of data in the example results contain information about the predictable attribute, **Yearly Income**, which models the dependent variable. The SUPPORT column shows the count of cases in support of the two states of this attribute: either a **Yearly Income** value was available, or the **Yearly Income** value was missing.

The VARIANCE column tells you the computed variance of the predictable attribute. *Variance* is a measure of how scattered the values are in a sample, given an expected distribution. Variance here is calculated by taking the average of the squared deviation from the mean. The square root of the variance is also known as standard deviation. Analysis Services does not provide the standard deviation but you can easily calculate it.

For each regressor, three rows are output. They contain the coefficient, score gain, and regressor statistics.

Finally, the table contains a row that provides the intercept for the equation.

Coefficient

For each regressor, a coefficient (VALUETYPE = 7) is calculated. The coefficient itself appears in the ATTRIBUTE_VALUE column, whereas the VARIANCE column tells you the variance for the coefficient. The coefficients are calculated so as to maximize linearity.

Score Gain

The score gain (VALUETYPE = 8) for each regressor represents the interestingness score of the attribute. You can use this value to estimate the usefulness of multiple regressors.

Statistics

The regressor statistic (VALUETYPE = 9) is the mean for the attribute for cases that have a value. The ATTRIBUTE_VALUE column contains the mean itself, whereas the VARIANCE column contains the sum of deviations from the mean.

Intercept

Normally, the *intercept* (VALUETYPE = 11) or *residual* in a regression equation tells you the value of the predictable attribute, at the point where the input attribute, is 0. In many cases, this might not happen, and could lead to counterintuitive results.

For example, in a model that predicts income based on age, it is useless to learn the income at age 0. In real-life, it is typically more useful to know about the behavior of the line with respect to average values. Therefore, SQL Server Analysis Services modifies the intercept to express each regressor in a relationship with the mean.

This adjustment is difficult to see in the mining model content, but is apparent if you view the completed equation in the **Mining Legend** of the **Microsoft Tree Viewer**. The regression formula is shifted away from the 0 point to the point that represents the mean. This presents a view that is more intuitive given the current data.

Therefore, assuming that the mean age is around 45, the intercept (VALUETYPE = 11) for the regression formula tells you the mean income.

See Also

Mining Model Content (Analysis Services - Data Mining)
Microsoft Linear Regression Algorithm
Microsoft Linear Regression Algorithm Technical Reference
Linear Regression Model Query Examples

Linear Regression Model Query Examples

5/16/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a query against a data mining model, you can create a content query, which provides details about the patterns discovered in analysis, or you can create a prediction query, which uses the patterns in the model to make predictions for new data. For example, a content query might provide additional details about the regression formula, while a prediction query might tell you if a new data point fits the model. You can also retrieve metadata about the model by using a query.

This section explains how to create queries for models that are based on the Microsoft Linear Regression algorithm.

> **NOTE**
> Because linear regression is based on a special case of the Microsoft Decision Trees algorithm, there are many similarities, and some decision tree models that use continuous predictable attributes can contain regression formulas. For more information, see Microsoft Decision Trees Algorithm Technical Reference.

Content queries

Using the Data Mining Schema Rowset to determine parameters used for a model

Using DMX to return the regression formula for the model

Returning only the coefficient for the model

Prediction queries

Predicting income using a singleton query

Using prediction functions with a regression model

Finding Information about the Linear Regression Model

The structure of a linear regression model is extremely simple: the mining model represents the data as a single node, which defines the regression formula. For more information, see Mining Model Content for Logistic Regression Models (Analysis Services - Data Mining).

Return to Top

Sample Query 1: Using the Data Mining Schema Rowset to Determine Parameters Used for a Model

By querying the data mining schema rowset, you can find metadata about the model. This might include when the model was created, when the model was last processed, the name of the mining structure that the model is based on, and the name of the column designated as the predictable attribute. You can also return the parameters that were used when the model was first created.

```
SELECT MINING_PARAMETERS
FROM $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = 'TM_PredictIncome'
```

Sample results:

MINING_PARAMETERS
COMPLEXITY_PENALTY=0.9, MAXIMUM_INPUT_ATTRIBUTES=255, MAXIMUM_OUTPUT_ATTRIBUTES=255, MINIMUM_SUPPORT=10, SCORE_METHOD=4, SPLIT_METHOD=3, FORCE_REGRESSOR=

> **NOTE**
> The parameter setting, "`FORCE_REGRESSOR =` ", indicates that the current value for the FORCE_REGRESSOR parameter is null.

Return to Top

Sample Query 2: Retrieving the Regression Formula for the Model

The following query returns the mining model content for a linear regression model that was built by using the same Targeted Mailing data source that was used in the Basic Data Mining Tutorial. This model predicts customer income based on age.

The query returns the contents of the node that contains the regression formula. Each variable and coefficient is stored in a separate row of the nested NODE_DISTRIBUTION table. If you want to view the complete regression formula, use the Microsoft Tree Viewer, click the **(All)** node, and open the **Mining Legend**.

```
SELECT FLATTENED NODE_DISTRIBUTION as t
FROM LR_PredictIncome.CONTENT
```

> **NOTE**
> If you reference individual columns of the nested table by using a query such as `SELECT <column name> from NODE_DISTRIBUTION`, some columns, such as **SUPPORT** or **PROBABILITY**, must be enclosed in brackets to distinguish them from reserved keywords of the same name.

Expected results:

T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE	T.SUPPORT	T.PROBABILITY	T.VARIANCE	T.VALUETYPE
Yearly Income	Missing	0	0.000457142857142857	0	1
Yearly Income	57220.8876687257	17484	0.999542857142857	1041275619.52776	3
Age	471.687717702463	0	0	126.969442359327	7

T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE	T.SUPPORT	T.PROBABILITY	T.VARIANCE	T.VALUETYPE
Age	234.680904692439	0	0	0	8
Age	45.4269617936399	0	0	126.969442359327	9
	35793.5477381267	0	0	1012968919.28372	11

In comparison, in the **Mining Legend**, the regression formula appears as follows:

```
Yearly Income = 57,220.919 + 471.688 * (Age - 45.427)
```

You can see that in the **Mining Legend**, some numbers are rounded; however, the NODE_DISTRIBUTION table and the **Mining Legend** essentially contain the same values.

The values in the VALUETYPE column tell you what kind of information is contained in each row, which is useful if you are processing the results programmatically. The following table shows the value types that are output for a linear regression formula.

VALUETYPE
1 (Missing)
3 (Continuous)
7 (Coefficient)
8 (Score Gain)
9 (Statistics)
7 (Coefficient)
8 (Score Gain)
9 (Statistics)
11 (Intercept)

For more information about the meaning of each value type for regression models, see Mining Model Content for Linear Regression Models (Analysis Services - Data Mining).

Return to Top

Sample Query 3: Returning Only the Coefficient for the Model

By using the VALUETYPE enumeration, you can return only the coefficient for the regression equation, as shown in the following query:

```
SELECT FLATTENED MODEL_NAME,
    (SELECT ATTRIBUTE_VALUE, VALUETYPE
     FROM NODE_DISTRIBUTION
     WHERE VALUETYPE = 11)
AS t
FROM LR_PredictIncome.CONTENT
```

This query returns two rows, one from the mining model content, and the row from the nested table that contains the coefficient. The ATTRIBUTE_NAME column is not included here because it is always blank for the coefficient.

MODEL_NAME	T.ATTRIBUTE_VALUE	T.VALUETYPE
LR_PredictIncome		
LR_PredictIncome	35793.5477381267	11

Return to Top

Making Predictions from a Linear Regression Model

You can build prediction queries on linear regression models by using the Mining Model Prediction tab in Data Mining Designer. The prediction query builder is available in both SQL Server Management Studio and SQL Server Data Tools (SSDT).

> **NOTE**
> You can also create queries on regression models by using the SQL Server 2005 (9.x) Data Mining Add-ins for Excel or the SQL Server 2008 Data Mining Add-ins for Excel. Even though the Data Mining Add-ins for Excel do not create regression models, you can browse and query any mining model that is stored on an instance of Analysis Services.

Return to Top

Sample Query 4: Predicting Income using a Singleton Query

The easiest way to create a single query on a regression model is by using the **Singleton Query Input** dialog box. For example, you can build the following DMX query by selecting the appropriate regression model, choosing **Singleton Query**, and then typing **20** as the value for **Age**.

```
SELECT [LR_PredictIncome].[Yearly Income]
From   [LR_PredictIncome]
NATURAL PREDICTION JOIN
(SELECT 20 AS [Age]) AS t
```

Sample results:

YEARLY INCOME
45227.302092176

Return to Top

Sample Query 5: Using Prediction Functions with a Regression Model

You can use many of the standard prediction functions with linear regression models. The following example illustrates how to add some descriptive statistics to the prediction query results. From these results, you can see

that there is considerable deviation from the mean for this model.

```
SELECT
  ([LR_PredictIncome].[Yearly Income]) as [PredIncome],
  (PredictStdev([LR_PredictIncome].[Yearly Income])) as [StDev1]
From
  [LR_PredictIncome]
NATURAL PREDICTION JOIN
(SELECT 20 AS [Age]) AS t
```

Sample results:

YEARLY INCOME	STDEV1
45227.302092176	31827.1726561396

Return to Top

List of Prediction Functions

All Microsoft algorithms support a common set of functions. However, the Microsoft Linear Regression algorithm supports the additional functions listed in the following table.

Prediction Function	Usage
IsDescendant (DMX)	Determines whether one node is a child of another node in the model.
IsInNode (DMX)	Indicates whether the specified node contains the current case.
PredictHistogram (DMX)	Returns a predicted value, or set of values, for a specified column.
PredictNodeId (DMX)	Returns the Node_ID for each case.
PredictStdev (DMX)	Returns standard deviation for the predicted value.
PredictSupport (DMX)	Returns the support value for a specified state.
PredictVariance (DMX)	Returns the variance of a specified column.

For a list of the functions that are common to all Microsoft algorithms, see Data Mining Algorithms (Analysis Services - Data Mining). For more information about how to use these functions, see Data Mining Extensions (DMX) Function Reference.

See Also

Microsoft Linear Regression Algorithm
Data Mining Queries
Microsoft Linear Regression Algorithm Technical Reference
Mining Model Content for Linear Regression Models (Analysis Services - Data Mining)

Microsoft Logistic Regression Algorithm

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Logistic regression is a well-known statistical technique that is used for modeling binary outcomes.

There are various implementations of logistic regression in statistics research, using different learning techniques. The Microsoft Logistic Regression algorithm has been implemented by using a variation of the Microsoft Neural Network algorithm. This algorithm shares many of the qualities of neural networks but is easier to train.

One advantage of logistic regression is that the algorithm is highly flexible, taking any kind of input, and supports several different analytical tasks:

- Use demographics to make predictions about outcomes, such as risk for a certain disease.
- Explore and weight the factors that contribute to a result. For example, find the factors that influence customers to make a repeat visit to a store.
- Classify documents, e-mail, or other objects that have many attributes.

Example

Consider a group of people who share similar demographic information and who buy products from the Adventure Works company. By modeling the data to relate to a specific outcome, such as purchase of a target product, you can see how the demographic information contributes to someone's likelihood of buying the target product.

How the Algorithm Works

Logistic regression is a well-known statistical method for determining the contribution of multiple factors to a pair of outcomes. The Microsoft implementation uses a modified neural network to model the relationships between inputs and outputs. The effect of each input on the output is measured, and the various inputs are weighted in the finished model. The name logistic regression comes from the fact that the data curve is compressed by using a logistic transformation, to minimize the effect of extreme values. For more information about the implementation, and how to customize the algorithm, see Microsoft Logistic Regression Algorithm Technical Reference.

Data Required for Logistic Regression Models

When you prepare data for use in training a logistic regression model, you should understand the requirements for the particular algorithm, including how much data is needed, and how the data is used.

The requirements for a logistic regression model are as follows:

A single key column Each model must contain one numeric or text column that uniquely identifies each record. Compound keys are not allowed.

Input columns Each model must contain at least one input column that contains the values that are used as factors in analysis. You can have as many input columns as you want, but depending on the number of values in each column, the addition of extra columns can increase the time it takes to train the model.

At least one predictable column The model must contain at least one predictable column of any data type, including continuous numeric data. The values of the predictable column can also be treated as inputs to the

model, or you can specify that it be used for prediction only. Nested tables are not allowed for predictable columns, but can be used as inputs.

For more detailed information about the content types and data types supported for logistic regression models, see the Requirements section of Microsoft Logistic Regression Algorithm Technical Reference.

Viewing a Logistic Regression Model

To explore the model, you can use the Microsoft Neural Network Viewer, or the Microsoft Generic Content Tree Viewer.

When you view the model by using the Microsoft Neural Network Viewer, Analysis Services shows you the factors that contribute to a particular outcome, ranked by their importance. You can choose an attribute and values to compare. For more information, see Browse a Model Using the Microsoft Neural Network Viewer.

If you want to know more, you can browse the model details by using the Microsoft Generic Content Tree Viewer. The model content for a logistic regression model includes a marginal node that shows you the all the inputs used for the model, and subnetworks for the predictable attributes. For more information, see Mining Model Content for Logistic Regression Models (Analysis Services - Data Mining).

Creating Predictions

After the model has been trained, you can create queries against the model content to get the regression coefficients and other details, or you can use the model to make predictions.

- For general information about how to create queries against a data mining model, see Data Mining Queries.
- For examples of queries on a logistic regression model, see Clustering Model Query Examples.

Remarks

- Does not support drillthrough. This is because the structure of nodes in the mining model does not necessarily correspond directly to the underlying data.
- Does not support the creation of data mining dimensions.
- Supports the use of OLAP mining models.
- Does not support the use of Predictive Model Markup Language (PMML) to create mining models.

See Also

Mining Model Content for Logistic Regression Models (Analysis Services - Data Mining)
Microsoft Logistic Regression Algorithm Technical Reference
Logistic Regression Model Query Examples

Microsoft Logistic Regression Algorithm Technical Reference

11/28/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Logistic Regression algorithm is a variation of the Microsoft Neural Network algorithm, where the *HIDDEN_NODE_RATIO* parameter is set to 0. This setting will create a neural network model that does not contain a hidden layer, and that therefore is equivalent to logistic regression.

Implementation of the Microsoft Logistic Regression Algorithm

Suppose the predictable column contains only two states, yet you still want to perform a regression analysis, relating input columns to the probability that the predictable column will contain a specific state. The following diagram illustrates the results you will obtain if you assign 1 and 0 to the states of the predictable column, calculate the probability that the column will contain a specific state, and perform a linear regression against an input variable.

The x-axis contains values of an input column. The y-axis contains the probabilities that the predictable column will be one state or the other. The problem with this is that the linear regression does not constrain the column to be between 0 and 1, even though those are the maximum and minimum values of the column. A way to solve this problem is to perform logistic regression. Instead of creating a straight line, logistic regression analysis creates an "S" shaped curve that contains maximum and minimum constraints. For example, the following diagram illustrates the results you will achieve if you perform a logistic regression against the same data as used for the previous example.

Notice how the curve never goes above 1 or below 0. You can use logistic regression to describe which input columns are important in determining the state of the predictable column.

Feature Selection

Feature selection is used automatically by all Analysis Services data mining algorithms to improve analysis and reduce processing load. The method used for feature selection in a logistic regression model depends on the data type of the attribute. Because logistic regression is based on the Microsoft Neural Network algorithm, it uses a subset of the feature selection methods that apply to neural networks. For more information, see Feature Selection (Data Mining).

Scoring Inputs

Scoring in the context of a neural network model or logistic regression model means the process of converting the values that are present in the data into a set of values that use the same scale and therefore can be compared to each other. For example, suppose the inputs for Income range from 0 to 100,000 whereas the inputs for [Number of Children] range from 0 to 5. This conversion process allows you to compare the importance of each input regardless of the difference in values.

For each state that appears in the training set, the model generates an input. For discrete or discretized inputs, an additional input is created to represent the Missing state, if the missing state appears at least once in the training set. For continuous inputs, at most two input nodes are created: one for Missing values, if present in the training data, and one input for all existing, or non-null, values. Each input is scaled to a numeric format using the z-score normalization method, `(x - μ)\StdDev`.

During z-score normalization, the mean (μ) and standard deviation are obtained over the complete training set.

Continuous values

Value is present: `(X - μ)/σ` (X is the actual value being encoded)

Value is absent: `- μ/σ` (negative mu divided by sigma)

Discrete values

`μ = p` (the prior probability of a state)

StdDev `= sqrt(p\(1-p))`

Value is present: `\(1 - μ)/σ` (One minus mu divided by sigma)

Value is absent: `(- μ)/σ` (negative mu divided by sigma)

Understanding Logistic Regression Coefficients

There are various methods in the statistical literature for performing logistic regression, but an important part of all methods is assessing the fit of the model. A variety of goodness-to-fit statistics have been proposed, among them odds ratios and covariate patterns. A discussion of how to measure the fit of a model is beyond the scope of this topic; however, you can retrieve the value of the coefficients in the model and use them to design your own measures of fit.

> **NOTE**
> The coefficients that are created as part of a logistic regression model do not represent odds ratios and should not be interpreted as such.

The coefficients for each node in the model graph represent a weighted sum of the inputs to that node. In a logistic regression model, the hidden layer is empty; therefore, there is only one set of coefficients, which is stored in the output nodes. You can retrieve the values of the coefficients by using the following query:

```
SELECT FLATTENED [NODE_UNIQUE NAME],
(SELECT ATTRIBUTE_NAME< ATTRIBUTE_VALUE
FROM NODE_DISTRIBUTION) AS t
FROM <model name>.CONTENT
WHERE NODE_TYPE = 23
```

For each output value, this query returns the coefficients and an ID that points back to the related input node. It also returns a row that contains the value of the output and the intercept. Each input X has its own coefficient (Ci), but the nested table also contains a "free" coefficient (Co), calculated according to the following formula:

`F(X) = X1*C1 + X2*C2 + ... +Xn*Cn + X0`

Activation: `exp(F(X)) / (1 + exp(F(X)) )`

For more information, see Logistic Regression Model Query Examples.

Customizing the Logistic Regression Algorithm

The Microsoft logistic regression algorithm supports several parameters that affect the behavior, performance, and accuracy of the resulting mining model. You can also modify the behavior of the model by setting modeling flags on the columns used as input.

Setting Algorithm Parameters

The following table describes the parameters that can be used with the Microsoft Logistic Regression algorithm.

HOLDOUT_PERCENTAGE
Specifies the percentage of cases within the training data used to calculate the holdout error. HOLDOUT_PERCENTAGE is used as part of the stopping criteria while training the mining model.

The default is 30.

HOLDOUT_SEED
Specifies a number to use to seed the pseudo-random generator when randomly determining the holdout data. If HOLDOUT_SEED is set to 0, the algorithm generates the seed based on the name of the mining model, to guarantee that the model content remains the same during reprocessing.

The default is 0.

MAXIMUM_INPUT_ATTRIBUTES
Defines the number of input attributes that the algorithm can handle before it invokes feature selection. Set this value to 0 to turn off feature selection.

The default is 255.

MAXIMUM_OUTPUT_ATTRIBUTES
Defines the number of output attributes that the algorithm can handle before it invokes feature selection. Set this value to 0 to turn off feature selection.

The default is 255.

MAXIMUM_STATES
Specifies the maximum number of attribute states that the algorithm supports. If the number of states that an attribute has is larger than the maximum number of states, the algorithm uses the most popular states of the attribute and ignores the remaining states.

The default is 100.

SAMPLE_SIZE
Specifies the number of cases to be used to train the model. The algorithm provider uses either this number or

the percentage of total of cases that are not included in the holdout percentage as specified by the HOLDOUT_PERCENTAGE parameter, whichever value is smaller.

In other words, if HOLDOUT_PERCENTAGE is set to 30, the algorithm will use either the value of this parameter, or a value that is equal to 70 percent of the total number of cases, whichever is smaller.

The default is 10000.

Modeling Flags

The following modeling flags are supported for use with the Microsoft Logistic Regression algorithm.

NOT NULL
Indicates that the column cannot contain a null. An error will result if Analysis Services encounters a null during model training.

Applies to mining structure columns.

MODEL_EXISTENCE_ONLY
Means that the column will be treated as having two possible states: **Missing** and **Existing**. A null is a missing value.

Applies to mining model column.

Requirements

A logistic regression model must contain a key column, input columns, and at least one predictable column.

Input and Predictable Columns

The Microsoft Logistic Regression algorithm supports the specific input column content types, predictable column content types, and modeling flags that are listed in the following table. For more information about what the content types mean when used in a mining model, see Content Types (Data Mining).

COLUMN	CONTENT TYPES
Input attribute	Continuous, Discrete, Discretized, Key, Table
Predictable attribute	Continuous, Discrete, Discretized

See Also

Microsoft Logistic Regression Algorithm
Linear Regression Model Query Examples
Mining Model Content for Logistic Regression Models (Analysis Services - Data Mining)
Microsoft Neural Network Algorithm

Mining Model Content for Logistic Regression Models

5/16/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes mining model content that is specific to models that use the Microsoft Logistic Regression algorithm. For an explanation of how to interpret statistics and structure shared by all model types, and general definitions of terms related to mining model content, see Mining Model Content (Analysis Services - Data Mining).

Understanding the Structure of a Logistic Regression Model

A logistic regression model is created by using the Microsoft Neural Network algorithm with parameters that constrain the model to eliminate the hidden node. Therefore, the overall structure of a logistic regression model is almost identical to that of a neural network: each model has a single parent node that represents the model and its metadata, and a special marginal statistics node (NODE_TYPE = 24) that provides descriptive statistics about the inputs used in the model.

Additionally, the model contains a subnetwork (NODE_TYPE = 17) for each predictable attribute. Just like in a neural network model, each subnetwork always contains two branches: one for the input layer, and another branch that contains the hidden layer (NODE_TYPE = 19) and the output layer (NODE_TYPE = 20) for the network. The same subnetwork may be used for multiple attributes if they are specified as predict-only. Predictable attributes that are also inputs may not appear in the same subnetwork.

However, in a logistic regression model, the node that represents the hidden layer is empty, and has no children. Therefore the model contains nodes that represent individual outputs (NODE_TYPE = 23) and individual inputs (NODE_TYPE = 21) but no individual hidden nodes.

By default, a logistic regression model is displayed in the **Microsoft Neural Network Viewer**. With this custom viewer, you can filter on input attributes and their values, and graphically see how they affect the outputs. The tooltips in the viewer show you the probability and lift associated with each pair of inputs and output values. For more information, see Browse a Model Using the Microsoft Neural Network Viewer.

To explore the structure of the inputs and subnetworks, and to see detailed statistics, you can use the Microsoft Generic Content Tree viewer. You can click on any node to expand it and see the child nodes, or view the weights and other statistics contained in the node.

Model Content for a Logistic Regression Model

This section provides detail and examples only for those columns in the mining model content that have particular relevance for logistic regression. The model content is almost identical to that of a neural network model, but descriptions that apply to neural network models may be repeated in this table for convenience.

For information about general-purpose columns in the schema rowset, such as MODEL_CATALOG and MODEL_NAME, that are not described here, or for explanations of mining model terminology, see Mining Model Content (Analysis Services - Data Mining).

MODEL_CATALOG
Name of the database where the model is stored.

MODEL_NAME
Name of the model.

ATTRIBUTE_NAME
The names of the attribute that corresponds to this node.

NODE	CONTENT
Model root	Blank
Marginal statistics	Blank
Input layer	Blank
Input node	Input attribute name
Hidden layer	Blank
Output layer	Blank
Output node	Output attribute name

NODE_NAME
The name of the node. Currently, this column contains the same value as NODE_UNIQUE_NAME, though this may change in future releases.

NODE_UNIQUE_NAME
The unique name of the node.

For more information about how the names and IDs provide structural information about the model, see the section, Using Node Names and IDs.

NODE_TYPE
A logistic regression model outputs the following node types:

NODE TYPE ID	DESCRIPTION
1	Model.
17	Organizer node for the subnetwork.
18	Organizer node for the input layer.

NODE TYPE ID	DESCRIPTION
19	Organizer node for the hidden layer. The hidden layer is empty.
20	Organizer node for the output layer.
21	Input attribute node.
23	Output attribute node.
24	Marginal statistics node.

NODE_CAPTION

A label or a caption associated with the node. In logistic regression models, always blank.

CHILDREN_CARDINALITY

An estimate of the number of children that the node has.

NODE	CONTENT
Model root	Indicates the count of child nodes, which includes at least 1 network, 1 required marginal node, and 1 required input layer. For example, if the value is 5, there are 3 subnetworks.
Marginal statistics	Always 0.
Input layer	Indicates the number of input attribute-values pairs that were used by the model.
Input node	Always 0.
Hidden layer	In a logistic regression model, always 0.
Output layer	Indicates the number of output values.
Output node	Always 0.

PARENT_UNIQUE_NAME

The unique name of the node's parent. NULL is returned for any nodes at the root level.

For more information about how the names and IDs provide structural information about the model, see the section, Using Node Names and IDs.

NODE_DESCRIPTION

A user-friendly description of the node.

NODE	CONTENT
Model root	Blank
Marginal statistics	Blank
Input layer	Blank

NODE	CONTENT
Input node	Input attribute name
Hidden layer	Blank
Output layer	Blank
Output node	If the output attribute is continuous, contains the name of the output attribute. If the output attribute is discrete or discretized, contains the name of the attribute and the value.

NODE_RULE
An XML description of the rule that is embedded in the node.

NODE	CONTENT
Model root	Blank
Marginal statistics	Blank
Input layer	Blank
Input node	An XML fragment containing the same information as the NODE_DESCRIPTION column.
Hidden layer	Blank
Output layer	Blank
Output node	An XML fragment containing the same information as the NODE_DESCRIPTION column.

MARGINAL_RULE
For logistic regression models, always blank.

NODE_PROBABILITY
The probability associated with this node. For logistic regression models, always 0.

MARGINAL_PROBABILITY
The probability of reaching the node from the parent node. For logistic regression models, always 0.

NODE_DISTRIBUTION
A nested table that contains statistical information for the node. For detailed information about the contents of this table for each node type, see the section, Understanding the NODE_DISTRIBUTION Table, in Mining Model Content for Neural Network Models (Analysis Services - Data Mining).

NODE_SUPPORT
For logistic regression models, always 0.

> **NOTE**
> Support probabilities are always 0 because the output of this model type is not probabilistic. The only thing that is meaningful for the algorithm is the weights; therefore, the algorithm does not compute probability, support, or variance.

To get information about the support in the training cases for specific values, see the marginal statistics node.

MSOLAP_MODEL_COLUMN
Node	Content
Model root	Blank
Marginal statistics	Blank
Input layer	Blank
Input node	Input attribute name.
Hidden layer	Blank
Output layer	Blank
Output node	Input attribute name.

MSOLAP_NODE_SCORE
In logistic regression models, always 0.

MSOLAP_NODE_SHORT_CAPTION
In logistic regression models, always blank.

Using Node Names and IDs

The naming of the nodes in a logistic regression model provides additional information about the relationships between nodes in the model. The following table shows the conventions for the IDs that are assigned to nodes in each layer.

NODE TYPE	CONVENTION FOR NODE ID
Model root (1)	00000000000000000.
Marginal statistics node (24)	10000000000000000
Input layer (18)	30000000000000000
Input node (21)	Starts at 60000000000000000
Subnetwork (17)	20000000000000000
Hidden layer (19)	40000000000000000
Output layer (20)	50000000000000000
Output node (23)	Starts at 80000000000000000

You can use these IDs to determine how output attributes are related to specific input layer attributes, by viewing the NODE_DISTRIBUTION table of the output node. Each row in that table contains an ID that points back to a specific input attribute node. The NODE_DISTRIBUTION table also contains the coefficient for that input-output pair.

See Also

Microsoft Logistic Regression Algorithm
Mining Model Content for Neural Network Models (Analysis Services - Data Mining)
Logistic Regression Model Query Examples
Microsoft Logistic Regression Algorithm Technical Reference

Logistic Regression Model Query Examples

5/16/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a query against a data mining model, you can create a content query, which provides details about the patterns discovered in analysis, or you can create a prediction query, which uses the patterns in the model to make predictions using new data.

This section explains how to create queries for models that are based on the Microsoft Logistic Regression algorithm.

Content Queries

Retrieving Model Parameters by Using the Data Mining Schema Rowset

Finding Additional Detail about the Model by Using DMX

Prediction Queries

Making Predictions for a Continuous Value

Making Predictions for a Discrete Value

Getting Information about the Logistic Regression Model

Logistic regression models are created by using the Microsoft Neural Network algorithm with a special set of parameters; therefore, a logistic regression model has some of the same information as a neural networks model, but is less complex. To understand the structure of the model content, and which node types store what kind of information, see Mining Model Content for Logistic Regression Models (Analysis Services - Data Mining).

To follow along in the query scenarios, you can create a logistic regression model as described in the following section of the Intermediate Data Mining Tutorial: Lesson 5: Building Neural Network and Logistic Regression Models (Intermediate Data Mining Tutorial).

You can also use the mining structure, Targeted Mailing, from the Basic Data Mining Tutorial.

```
ALTER MINING STRUCTURE [Targeted Mailing]
ADD MINING MODEL [TM_Logistic Regression]
([Customer Key],
[Age],
[Bike Buyer] PREDICT,
[Yearly Income] PREDICT,
[Commute Distance],
[English Education],
Gender,
[House Owner Flag],
[Marital Status],
[Number Cars Owned],
[Number Children At Home],
[Region],
[Total Children]
)
USING Microsoft_Logistic_Regression
```

Sample Query 1: Retrieving Model Parameters by Using the Data Mining Schema Rowset

By querying the data mining schema rowset, you can find metadata about the model, such as when it was created, when the model was last processed, the name of the mining structure that the model is based on, and the name of the column used as the predictable attribute. The following example returns the parameters that were used when the model was first created, together with the name and type of the model, and the date that it was created.

```
SELECT MODEL_NAME, SERVICE_NAME, DATE_CREATED, MINING_PARAMETERS
FROM $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = 'Call Center_LR'
```

Sample results:

MODEL_NAME	SERVICE_NAME	DATE_CREATED	MINING_PARAMETERS
Call Center_LR	Microsoft_Logistic_Regression	04/07/2009 20:38:33	HOLDOUT_PERCENTAGE=30, HOLDOUT_SEED=1, MAXIMUM_INPUT_ATTRIBUTES=255, MAXIMUM_OUTPUT_ATTRIBUTES=255, MAXIMUM_STATES=100, SAMPLE_SIZE=10000

Sample Query 2: Finding Additional Detail about the Model by Using DMX

The following query returns some basic information about the logistic regression model. A logistic regression model is similar to a neural network model in many ways, including the presence of a marginal statistic node (NODE_TYPE = 24) that describes the values used as inputs. This example query uses the Targeted Mailing model, and gets the values of all the inputs by retrieving them from the nested table, NODE_DISTRIBUTION.

```
SELECT FLATTENED NODE_DISTRIBUTION AS t
FROM [TM_Logistic Regression].CONTENT
```

Partial results:

T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE	T.SUPPORT	T.PROBABILITY	T.VARIANCE	T.VALUETYPE
Age	Missing	0	0	0	1
Age	45.43491192	17484	1	126.9544114	3
Bike Buyer	Missing	0	0	0	1
Bike Buyer	0	8869	0.507263784	0	4
Bike Buyer	1	8615	0.492736216	0	4
Commute Distance	Missing	0	0	0	1
Commute Distance	5-10 Miles	3033	0.173472889	0	4

The actual query returns many more rows; however, this sample illustrates the type of information that is provided about the inputs. For discrete inputs, each possible value is listed in the table. For continuous-value

inputs such as Age, a complete listing is impossible, so the input is discretized as a mean. For more information about how to use the information in the marginal statistics node, see Mining Model Content for Logistic Regression Models (Analysis Services - Data Mining).

> **NOTE**
> The results have been flattened for easier viewing, but you can return the nested table in a single column if your provider supports hierarchical rowsets.

Prediction Queries on a Logistic Regression Model

You can use the Predict (DMX) function with every kind of mining model to provide new data to the model and make predictions based on the new values. You can also use functions to return additional information about the prediction, such as the probability that a prediction is correct. This section provides some examples of prediction queries on a logistic regression model.

Sample Query 3: Making Predictions for a Continuous Value

Because logistic regression supports the use of continuous attributes for both input and prediction, it is easy to create models that correlate various factors in your data. You can use prediction queries to explore the relationship among these factors.

The following query sample is based on the Call Center model, from the Intermediate Tutorial, and creates a singleton query that predicts service grade for the Friday AM shift. The PredictHistogram (DMX) function returns a nested table that provides statistics relevant to understanding the validity of the predicted value.

```
SELECT
  Predict([Call Center_LR].[Service Grade]) as Predicted ServiceGrade,
  PredictHistogram([Call Center_LR].[Service Grade]) as [Results],
FROM
  [Call Center_LR]
NATURAL PREDICTION JOIN
(SELECT 'Friday' AS [Day Of Week],
  'AM' AS [Shift]) AS t
```

Sample results:

PREDICTED SERVICE GRADE	SERVICE GRADE	$SUPPORT	$PROBABILITY	$ADJUSTEDPROBABILITY	$VARIANCE	$STDEV
0.102601830123659	0.102601830123659	83.0232558139535	0.988372093023256	0	0.00120552660600087	0.034720694203902
		0.976744186046512	0.0116279069767442	0.0116279069767442	0	0

For more information about the probability, support, and standard deviation values in the nested NODE_DISTRIBUTION table, see Mining Model Content for Logistic Regression Models (Analysis Services - Data Mining).

Sample Query 4: Making Predictions for a Discrete Value

Logistic regression is typically used in scenarios where you want to analyze the factors that contribute to a binary outcome. Although the model used in the tutorial predicts a continuous value, **ServiceGrade**, in a real-life scenario you might want to set up the model to predict whether service grade met some discretized target value. Alternatively, you could output the predictions using a continuous value but later group the predicted outcomes

into **Good**, **Fair**, or **Poor**.

The following sample illustrates how to change the way that the predictable attribute is grouped. To do this, you create a copy of the mining structure and then change the discretization method of the target column so that the values are grouped rather than continuous.

The following procedure describes how to change the grouping of Service Grade values in the Call Center data.

To create a discretized version of the Call Center mining structure and models

1. In SQL Server Data Tools (SSDT), in Solution Explorer, expand **Mining Structures**.
2. Right-click Call Center.dmm and select **Copy**.
3. Right click **Mining Structures** and select **Paste**. A new mining structure iss added, named Call Center 1.
4. Right-click the new mining structure and select **Rename**. Type the new name, **Call Center Discretized**.
5. Double-click the new mining structure to open it in the designer. Notice that the mining models have all been copied as well, and all have the extension 1. Leave the names as is for now.
6. In the **Mining Structure** tab, right-click the column for Service Grade, and select **Properties**.
7. Change the **Content** property from **Continuous** to **Discretized**. Change the **DiscretizationMethod** property to **Clusters**. For Discretization BucketCount, type **3**.

   > **NOTE**
   > These parameters are just used for illustrating the process, and do not necessarily produce a valid model,

8. From the **Mining Model** menu, select **Process structure and all models**.

The following sample query is based on this discretized model, and predicts the service grade for the specified day of the week, together with the probabilities for each predicted outcome.

```
SELECT
  (PredictHistogram([Call Center_LR 1].[Service Grade])) as [Predictions]
FROM
  [Call Center_LR 1]
NATURAL PREDICTION JOIN
(SELECT 'Saturday' AS [Day Of Week]) AS t
```

Expected results:

Predictions:

SERVICE GRADE	$SUPPORT	$PROBABILITY	$ADJUSTEDPROBABILITY	$VARIANCE	$STDEV
0.10872718383125	35.7246504770641	0.425293458060287	0.0170168360030293	0	0
0.05855769230625	31.7098880800703	0.377498667619885	0.020882020060454	0	0
0.170169491525	15.6109159883202	0.185844237956192	0.0661386571386049	0	0
	0.954545454545455	0.0113636363636364	0.0113636363636364	0	0

Note that the predicted outcomes have been grouped into three categories as specified; however, these groupings are based on the clustering of actual values in the data, not arbitrary values that you might set as business goals.

List of Prediction Functions

All Microsoft algorithms support a common set of functions. However, the Microsoft Logistic Regression algorithm supports the additional functions listed in the following table.

Prediction Function	Usage
IsDescendant (DMX)	Determines whether one node is a child of another node in the model.
PredictAdjustedProbability (DMX)	Returns the adjusted probability of a specified state.
PredictHistogram (DMX)	Returns a predicted value, or set of values, for a specified column.
PredictProbability (DMX)	Returns the probability for a specified state.
PredictStdev (DMX)	Returns standard deviation for the predicted value.
PredictSupport (DMX)	Returns the support value for a specified state.
PredictVariance (DMX)	Returns the variance of a specified column.

For a list of the functions that are common to all Microsoft algorithms, see General Prediction Functions (DMX). For the syntax of specific functions, see Data Mining Extensions (DMX) Function Reference.

> **NOTE**
> For neural network and logistic regression models, the PredictSupport (DMX) function returns a single value that represents the size of the training set for the entire model.

See Also

Data Mining Queries
Microsoft Logistic Regression Algorithm
Microsoft Logistic Regression Algorithm Technical Reference
Mining Model Content for Logistic Regression Models (Analysis Services - Data Mining)
Lesson 5: Building Neural Network and Logistic Regression Models (Intermediate Data Mining Tutorial)

Microsoft Naive Bayes Algorithm

11/27/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Naive Bayes algorithm is a classification algorithm based on Bayes' theorems, and can be used for both exploratory and predictive modeling. The word naïve in the name Naïve Bayes derives from the fact that the algorithm uses Bayesian techniques but does not take into account dependencies that may exist.

This algorithm is less computationally intense than other Microsoft algorithms, and therefore is useful for quickly generating mining models to discover relationships between input columns and predictable columns. You can use this algorithm to do initial exploration of data, and then later you can apply the results to create additional mining models with other algorithms that are more computationally intense and more accurate.

Example

As an ongoing promotional strategy, the marketing department for the Adventure Works Cycle company has decided to target potential customers by mailing out fliers. To reduce costs, they want to send fliers only to those customers who are likely to respond. The company stores information in a database about demographics and response to a previous mailing. They want to use this data to see how demographics such as age and location can help predict response to a promotion, by comparing potential customers to customers who have similar characteristics and who have purchased from the company in the past. Specifically, they want to see the differences between those customers who bought a bicycle and those customers who did not.

By using the Microsoft Naive Bayes algorithm, the marketing department can quickly predict an outcome for a particular customer profile, and can therefore determine which customers are most likely to respond to the fliers. By using the Microsoft Naive Bayes Viewer in SQL Server Data Tools (SSDT), they can also visually investigate specifically which input columns contribute to positive responses to fliers.

How the Algorithm Works

The Microsoft Naive Bayes algorithm calculates the probability of every state of each input column, given each possible state of the predictable column.

To understand how this works, use the Microsoft Naive Bayes Viewer in SQL Server Data Tools (SSDT) (as shown in the following graphic) to visually explore how the algorithm distributes states.

Here, the Microsoft Naive Bayes Viewer lists each input column in the dataset, and shows how the states of each column are distributed, given each state of the predictable column.

You would use this view of the model to identify the input columns that are important for differentiating between states of the predictable column.

For example, in the row for Commute Distance shown here, the distribution of input values is visibly different for buyers vs. non-buyers. What this tells you is that the input, Commute Distance = 0-1 miles, is a potential predictor.

The viewer also provides values for the distributions, so you can see that for customers who commute from one to two miles to work, the probability of them buying a bike is 0.387, and the probability that they will not buy a bike is 0.287. In this example, the algorithm uses the numeric information, derived from customer characteristics (such as commute distance), to predict whether a customer will buy a bike.

For more information about using the Microsoft Naive Bayes Viewer, see Browse a Model Using the Microsoft Naive Bayes Viewer.

Data Required for Naive Bayes Models

When you prepare data for use in training a Naive Bayes model, you should understand the requirements for the algorithm, including how much data is needed, and how the data is used.

The requirements for a Naive Bayes model are as follows:

- **A single key column** Each model must contain one numeric or text column that uniquely identifies each record. Compound keys are not allowed.
- **Input columns** In a Naive Bayes model, all columns must be either discrete, or the values must have been binned. For information about how to discretize (bin) columns, see Discretization Methods (Data Mining).
- **Variables must be independent.** For a Naive Bayes model, it is also important to ensure that the input attributes are independent of each other. This is particularly important when you use the model for

prediction. If you use two columns of data that are already closely related, the effect would be to multiply the influence of those columns, which can obscure other factors that influence the outcome.

Conversely, the ability of the algorithm to identify correlations among variables is useful when you are exploring a model or dataset, to identify relationships among inputs.

- **At least one predictable column** The predictable attribute must contain discrete or discretized values.

  The values of the predictable column can be treated as inputs. This practice can be useful when you are exploring a new dataset, to find relationships among the columns.

Viewing the Model

To explore the model, you can use the **Microsoft Naive Bayes Viewer**. The viewer shows you how the input attributes relate to the predictable attribute. The viewer also provides a detailed profile of each cluster, a list of the attributes that distinguish each cluster from the others, and the characteristics of the entire training data set. For more information, see Browse a Model Using the Microsoft Naive Bayes Viewer.

If you want to know more detail, you can browse the model in the Microsoft Generic Content Tree Viewer (Data Mining). For more information about the type of information stored in the model, see Mining Model Content for Naive Bayes Models (Analysis Services - Data Mining).

Making Predictions

After the model has been trained, the results are stored as a set of patterns, which you can explore or use to make predictions.

You can create queries to return predictions about how new data relates to the predictable attribute, or you can retrieve statistics that describe the correlations found by the model.

For information about how to create queries against a data mining model, see Data Mining Queries. For examples of how to use queries with a Naive Bayes model, see Naive Bayes Model Query Examples.

Remarks

- Supports the use of Predictive Model Markup Language (PMML) to create mining models.
- Supports drillthrough.
- Does not support the creation of data mining dimensions.
- Supports the use of OLAP mining models.

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Feature Selection (Data Mining)
Naive Bayes Model Query Examples
Mining Model Content for Naive Bayes Models (Analysis Services - Data Mining)
Microsoft Naive Bayes Algorithm Technical Reference

Microsoft Naive Bayes Algorithm Technical Reference

11/27/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Naive Bayes algorithm is a classification algorithm provided by Microsoft SQL Server Analysis Services for use in predictive modeling. The algorithm calculates the conditional probability between input and predictable columns, and assumes that the columns are independent. This assumption of independence leads to the name Naive Bayes.

Implementation of the Microsoft Naive Bayes Algorithm

This algorithm is less computationally intense than other Microsoft algorithms, and therefore is useful for quickly generating mining models to discover relationships between input columns and predictable columns. The algorithm considers each pair of input attribute values and output attribute values.

A description of the mathematical properties of Bayes Theorem is beyond the scope of this documentation; for more information, see the paper by Microsoft Research titled Learning Bayesian Networks: The Combination of Knowledge and Statistical Data.

For a description of how probabilities in all models are adjusted to account for potential missing values, see Missing Values (Analysis Services - Data Mining).

Feature Selection

The Microsoft Naive Bayes algorithm performs automatic feature selection to limit the number of values that are considered when building the model. For more information, see Feature Selection (Data Mining).

ALGORITHM	METHOD OF ANALYSIS	COMMENTS
Naive Bayes	Shannon's Entropy Bayesian with K2 Prior Bayesian Dirichlet with uniform prior (default)	Naive Bayes only accepts discrete or discretized attributes; therefore, it cannot use the interestingness score.

The algorithm is designed to minimize processing time and efficiently select the attributes that have the greatest importance; however, you can control the data that is used by the algorithm by setting parameters as follows:

- To limit the values that are used as inputs, decrease the value of MAXIMUM_INPUT_ATTRIBUTES.
- To limit the number of attributes analyzed by the model, decrease the value of MAXIMUM_OUTPUT_ATTRIBUTES.
- To limit the number of values that can be considered for any one attribute, decrease the value of MINIMUM_STATES.

Customizing the Naive Bayes Algorithm

The Microsoft Naive Bayes algorithm supports several parameters that affect the behavior, performance, and accuracy of the resulting mining model. You can also set modeling flags on the model columns to control how data is processed, or set flags on the mining structure to specify how missing values or nulls should be handled.

Setting Algorithm Parameters

The Microsoft Naive Bayes algorithm supports several parameters that affect the performance and accuracy of the resulting mining model. The following table describes each parameter.

MAXIMUM_INPUT_ATTRIBUTES
Specifies the maximum number of input attributes that the algorithm can handle before it invokes feature selection. Setting this value to 0 disables feature selection for input attributes.

The default is 255.

MAXIMUM_OUTPUT_ATTRIBUTES
Specifies the maximum number of output attributes that the algorithm can handle before it invokes feature selection. Setting this value to 0 disables feature selection for output attributes.

The default is 255.

MINIMUM_DEPENDENCY_PROBABILITY
Specifies the minimum dependency probability between input and output attributes. This value is used to limit the size of the content that is generated by the algorithm. This property can be set from 0 to 1. Larger values reduce the number of attributes in the content of the model.

The default is 0.5.

MAXIMUM_STATES
Specifies the maximum number of attribute states that the algorithm supports. If the number of states that an attribute has is greater than the maximum number of states, the algorithm uses the attribute's most popular states and treats the remaining states as missing.

The default is 100.

Modeling Flags

The Microsoft Decision Trees algorithm supports the following modeling flags. When you create the mining structure or mining model, you define modeling flags to specify how values in each column are handled during analysis. For more information, see Modeling Flags (Data Mining).

MODELING FLAG	DESCRIPTION
MODEL_EXISTENCE_ONLY	Means that the column will be treated as having two possible states: Missing and Existing. A null is a missing value. Applies to mining model column.
NOT NULL	Indicates that the column cannot contain a null. An error will result if Analysis Services encounters a null during model training. Applies to mining structure column.

Requirements

A Naive Bayes tree model must contain a key column, at least one predictable attribute, and at least one input attribute. No attribute can be continuous; if your data contains continuous numeric data, it will be ignored or discretized.

Input and Predictable Columns

The Microsoft Naive Bayes algorithm supports the specific input columns and predictable columns that are listed in the following table. For more information about what the content types mean when used in a mining model,

see Content Types (Data Mining).

COLUMN	CONTENT TYPES
Input attribute	Cyclical, Discrete, Discretized, Key, Table, and Ordered
Predictable attribute	Cyclical, Discrete, Discretized, Table, and Ordered

> **NOTE**
> Cyclical and Ordered content types are supported, but the algorithm treats them as discrete values and does not perform special processing.

See Also

Microsoft Naive Bayes Algorithm
Naive Bayes Model Query Examples
Mining Model Content for Naive Bayes Models (Analysis Services - Data Mining)

Mining Model Content for Naive Bayes Models (Analysis Services - Data Mining)

5/16/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes mining model content that is specific to models that use the Microsoft Naive Bayes algorithm. For an explanation of how to interpret statistics and structure shared by all model types, and general definitions of terms related to mining model content, see Mining Model Content (Analysis Services - Data Mining).

Understanding the Structure of a Naive Bayes Model

A Naive Bayes model has a single parent node that represents the model and its metadata, and underneath that parent node, any number of independent trees that represent the predictable attributes that you selected. In addition to trees for the attributes, each model contains one marginal statistics node (NODE_TYPE = 26) that provides descriptive statistics about the set of training cases. For more information, see Information in the Marginal Statistics Node.

For each predictable attribute and value, the model outputs a tree that contains information describing how the various input columns affected the outcome of that particular predictable. Each tree contains the predictable attribute and its value (NODE_TYPE = 9), and then a series of nodes that represent the input attributes (NODE_TYPE = 10). Because the input attributes typically have multiple values, each input attribute (NODE_TYPE = 10) may have multiple child nodes (NODE_TYPE = 11), each for a specific state of the attribute.

> **NOTE**
>
> Because a Naive Bayes model does not permit continuous data types, all the values of the input columns are treated as discrete or discretized. You can specify how a value is discretized. For more information, Change the Discretization of a Column in a Mining Model.

Model Content for a Naive Bayes Model

This section provides detail and examples only for those columns in the mining model content that have particular relevance for Naive Bayes models.

For information about general-purpose columns in the schema rowset, such as MODEL_CATALOG and MODEL_NAME, that are not described here, or for explanations of mining model terminology, see Mining Model Content (Analysis Services - Data Mining).

MODEL_CATALOG
Name of the database where the model is stored.

MODEL_NAME
Name of the model.

ATTRIBUTE_NAME
The names of the attributes that correspond to this node.

Model root The name of the predictable attribute.

Marginal statistics Not applicable

Predictable attribute The name of the predictable attribute.

Input attribute The name of the input attribute.

Input attribute state The name of the input attribute only. To get the state, use MSOLAP_NODE_SHORT_CAPTION.

NODE_NAME
The name of the node.

This column contains the same value as NODE_UNIQUE_NAME.

For more information about node naming conventions, see Using Node Names and IDs.

NODE_UNIQUE_NAME
The unique name of the node. The unique names are assigned according to a convention that provides information about the relationships among the nodes. For more information about node naming conventions, see Using Node Names and IDs.

NODE_TYPE
A Naive Bayes model outputs the following node types:

NODE TYPE ID	DESCRIPTION
26 (NaiveBayesMarginalStatNode)	Contains statistics that describes the entire set of training cases for the model.
9 (Predictable attribute)	Contains the name of the predictable-attribute.
10 (Input attribute)	Contains the name of an input attribute column, and child nodes that contains the values for the attribute.
11 (Input attribute state)	Contains the values or discretized values of all input attributes that were paired with a particular output attribute.

NODE_CAPTION
The label or a caption associated with the node. This property is primarily for display purposes.

Model root blank

Marginal statistics blank

Predictable attribute The name of the predictable attribute.

Input attribute The name of the predictable attribute and the current input attribute. Ex:

Bike Buyer -> Age

Input attribute state The name of the predictable attribute and the current input attribute, plus the value of the input. Ex:

Bike Buyer -> Age = Missing

CHILDREN_CARDINALITY
The number of children that the node has.

Model root Count of predictable attributes in the model plus 1 for the marginal statistics node.

Marginal statistics By definition has no children.

Predictable attribute Count of the input attributes that were related to the current predictable attribute.

Input attribute Count of the discrete or discretized values for the current input attribute.

Input attribute state Always 0.

PARENT_UNIQUE_NAME
The unique name of the parent node. For more information about relating parent and child nodes, see Using Node Names and IDs.

NODE_DESCRIPTION
The same as the node caption.

NODE_RULE
An XML representation of the node caption.

MARGINAL_RULE
The same as the node rule.

NODE_PROBABILITY
The probability associated with this node.

Model root Always 0.

Marginal statistics Always 0.

Predictable attribute Always 1.

Input attribute Always 1.

Input attribute state A decimal number that represents the probability of the current value. Values for all input attribute states under the parent input attribute node sum to 1.

MARGINAL_PROBABILITY
The same as the node probability.

NODE_DISTRIBUTION
A table that contains the probability histogram for the node. For more information, see NODE_DISTRIBUTION Table.

NODE_SUPPORT
The number of cases that support this node.

Model root Count of all cases in training data.

Marginal statistics Always 0.

Predictable attribute Count of all cases in training data.

Input attribute Count of all cases in training data.

Input attribute state Count of cases in training data that contain only this particular value.

MSOLAP_MODEL_COLUMN
A label used for display purposes. Usually the same as ATTRIBUTE_NAME.

MSOLAP_NODE_SCORE
Represents the importance of the attribute or value within the model.

Model root Always 0.

Marginal statistics Always 0.

Predictable attribute Always 0.

Input attribute Interestingness score for the current input attribute in relation to the current predictable attribute.

Input attribute state Always 0.

MSOLAP_NODE_SHORT_CAPTION
A text string that represents the name or the value of a column.

Model root Blank

Marginal statistics Blank

Predictable attribute The name of the predictable attribute.

Input attribute The name of the input attribute.

Input attribute state The value or discretized value of the input attribute.

Using Node Names and IDs

The naming of the nodes in a Naive Bayes model provides additional information about the type of node, to make it easier to understand the relationships among the information in the model. The following table shows the convention for the IDs that are assigned to different node types.

NODE TYPE	CONVENTION FOR NODE ID
Model root (1)	Always 0.
Marginal statistics node (26)	An arbitrary ID value.
Predictable attribute (9)	Hexadecimal number beginning with 10000000 Example: 100000001, 10000000b
Input attribute (10)	A two-part hexadecimal number where the first part is always 20000000, and the second part starts with the hexadecimal identifier of the related predictable attribute. Example: 20000000b00000000 In this case, the related predictable attribute is 10000000b.

NODE TYPE	CONVENTION FOR NODE ID
Input attribute state (11)	A three-part hexadecimal number where the first part is always 30000000, the second part starts with the hexadecimal identifier of the related predictable attribute, and the third part represents the identifier of the value. Example: 30000000b00000000200000000 In this case, the related predictable attribute is 10000000b.

You can use the IDs to relate input attributes and states to a predictable attribute. For example, the following query returns the names and captions for nodes that represent the possible combinations of input and predictable attributes for the model, `TM_NaiveBayes`.

```
SELECT NODE_NAME, NODE_CAPTION
FROM TM_NaiveBayes.CONTENT
WHERE NODE_TYPE = 10
```

Expected results:

NODE_NAME	NODE_CAPTION
20000000000000001	Bike Buyer -> Commute Distance
20000000000000002	Bike Buyer -> English Education
20000000000000003	Bike Buyer -> English Occupation
20000000000000009	Bike Buyer -> Marital Status
2000000000000000a	Bike Buyer -> Number Children At Home
2000000000000000b	Bike Buyer -> Region
2000000000000000c	Bike Buyer -> Total Children

You can then use the IDs of the parent nodes to retrieve the child nodes. The following query retrieves the nodes that contain values for the `Marital Status` attribute, together with the probability of each node.

```
SELECT NODE_NAME, NODE_CAPTION, NODE_PROBABILITY
FROM TM_NaiveBayes.CONTENT
WHERE NODE_TYPE = 11
AND [PARENT_UNIQUE_NAME] = '20000000000000009'
```

> **NOTE**
> The name of the column, PARENT_UNIQUE_NAME, must be enclosed in brackets to distinguish it from the reserved keyword of the same name.

Expected results:

NODE_NAME	NODE_CAPTION	NODE_PROBABILITY
300000000000000000900000000	Bike Buyer -> Marital Status = Missing	0
300000000000000000900000001	Bike Buyer -> Marital Status = S	0.457504004
300000000000000000900000002	Bike Buyer -> Marital Status = M	0.542495996

NODE_DISTRIBUTION Table

The nested table column, NODE_DISTRIBUTION, typically contains statistics about the distribution of values in the node. In a Naive Bayes model, this table is populated only for the following nodes:

NODE TYPE	CONTENT OF NESTED TABLE
Model root (1)	Blank.
Marginal statistics node (24)	Contains summary information for all predictable attributes and input attributes, for entire set of training data.
Predictable attribute (9)	Blank.
Input attribute (10)	Blank.
Input attribute state (11)	Contains statistics that describe the distribution of values in the training data for this particular combination of a predictable value and input attribute value.

You can use the node IDs or node captions to retrieve increasing levels of detail. For example, the following query retrieves specific columns from the NODE_DISTRIBUTION table for only those input attribute nodes that are related to the value, `'Marital Status = S'`.

```
SELECT FLATTENED NODE_CAPTION,
(SELECT ATTRIBUTE_NAME, ATTRIBUTE_VALUE, [SUPPORT], [PROBABILITY], VALUETYPE
FROM NODE_DISTRIBUTION) as t
FROM TM_NaiveBayes.content
WHERE NODE_TYPE = 11
AND NODE_CAPTION = 'Bike Buyer -> Marital Status = S'
```

Expected results:

NODE_CAPTION	T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE	T.SUPPORT	T.PROBABILITY	T.VALUETYPE
Bike Buyer -> Marital Status = S	Bike Buyer	Missing	0	0	1
Bike Buyer -> Marital Status = S	Bike Buyer	0	3783	0.472934117	4
Bike Buyer -> Marital Status = S	Bike Buyer	1	4216	0.527065883	4

NODE_CAPTION	T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE	T.SUPPORT	T.PROBABILITY	T.VALUETYPE

In these results, the value of the SUPPORT column tells you the count of customers with the specified marital status who purchased a bike. The PROBABILITY column contains the probability of each attribute value, as calculated for this node only. For general definitions of terms used in the NODE_DISTRIBUTION table, see Mining Model Content (Analysis Services - Data Mining).

Information in the Marginal Statistics Node

In a Naive Bayes model, the nested table for the marginal statistics node contains the distribution of values for the entire set of training data. For example, the following table contains a partial list of the statistics in the nested NODE_DISTRIBUTION table for the model, `TM_NaiveBayes`:

ATTRIBUTE_NAME	ATTRIBUTE_VALUE	SUPPORT	PROBABILITY	VARIANCE	VALUETYPE
Bike Buyer	Missing	0	0	0	1
Bike Buyer	0	8869	0.507263784	0	4
Bike Buyer	1	8615	0.492736216	0	4
Marital Status	Missing	0	0	0	1
Marital Status	S	7999	0.457504004	0	4
Marital Status	M	9485	0.542495996	0	4
Total Children	Missing	0	0	0	1
Total Children	0	4865	0.278254404	0	4
Total Children	3	2093	0.119709449	0	4
Total Children	1	3406	0.19480668	0	4

The [Bike Buyer] column is included because the marginal statistics node always contains a description of the predictable attribute and its possible values. All other columns that are listed represent input attributes, together with the values that were used in the model. Values can only be missing, discrete or discretized.

In a Naive Bayes model, there can be no continuous attributes; therefore, all numeric data is represented as either discrete (VALUE_TYPE = 4) or discretized (VALUE_TYPE = 5).

A **Missing** value (VALUE_TYPE = 1) is added to every input and output attribute to represent potential values that were not present in the training data. You must be careful to distinguish between "missing" as a string and the default **Missing** value. For more information, see Missing Values (Analysis Services - Data Mining).

See Also

Mining Model Content (Analysis Services - Data Mining)
Data Mining Model Viewers

Naive Bayes Model Query Examples

5/16/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a query against a data mining model, you can create either a content query, which provides details about the patterns discovered in analysis, or you can create a prediction query, which uses the patterns in the model to make predictions for new data. You can also retrieve metadata about the model by using a query against the data mining schema rowset. This section explains how to create these queries for models that are based on the Microsoft Naive Bayes algorithm.

Content Queries

Getting model metadata by using DMX

Retrieving a summary of training data

Finding more information about attributes

Using system stored procedures

Prediction Queries

Predicting outcomes using a singleton query

Getting predictions with probability and support values

Predicting associations

Finding Information about a Naive Bayes Model

The model content of a Naive Bayes model provides aggregated information about the distribution of values in the training data. You can also retrieve information about the metadata of the model by creating queries against the data mining schema rowsets.

Sample Query 1: Getting Model Metadata by Using DMX

By querying the data mining schema rowset, you can find metadata for the model. This might include when the model was created, when the model was last processed, the name of the mining structure that the model is based on, and the name of the columns used as the predictable attribute. You can also return the parameters that were used when the model was created.

```
SELECT MODEL_CATALOG, MODEL_NAME, DATE_CREATED, LAST_PROCESSED,
SERVICE_NAME, PREDICTION_ENTITY, FILTER
FROM $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = 'TM_NaiveBayes_Filtered'
```

Sample results:

MODEL_CATALOG	AdventureWorks
MODEL_NAME	TM_NaiveBayes_Filtered

DATE_CREATED	3/1/2008 19:15
LAST_PROCESSED	3/2/2008 20:00
SERVICE_NAME	Microsoft_Naive_Bayes
PREDICTION_ENTITY	Bike Buyer,Yearly Income
FILTER	[Region] = 'Europe' OR [Region] = 'North America'

The model used for this example is based on the Naive Bayes model you create in the Basic Data Mining Tutorial, but was modified by adding a second predictable attribute and applying a filter to the training data.

Sample Query 2: Retrieving a Summary of Training Data

In a Naive Bayes model, the marginal statistics node stores aggregated information about the distribution of values in the training data. This summary is convenient and saves you from having to create SQL queries against the training data to find the same information.

The following example uses a DMX content query to retrieve the data from the node (NODE_TYPE = 24). Because the statistics are stored in a nested table, the FLATTENED keyword is used to make the results easier to view.

```
SELECT FLATTENED MODEL_NAME,
(SELECT ATTRIBUTE_NAME, ATTRIBUTE_VALUE, [SUPPORT], [PROBABILITY], VALUETYPE FROM NODE_DISTRIBUTION) AS t
FROM TM_NaiveBayes.CONTENT
WHERE NODE_TYPE = 26
```

> **NOTE**
> You must enclose the name of the columns, SUPPORT and PROBABILITY, in brackets to distinguish them from the Multidimensional Expressions (MDX) reserved keywords of the same names.

Partial results:

MODEL_NAME	T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE	T.SUPPORT	T.PROBABILITY	T.VALUETYPE
TM_NaiveBayes	Bike Buyer	Missing	0	0	1
TM_NaiveBayes	Bike Buyer	0	8869	0.507263784	4
TM_NaiveBayes	Bike Buyer	1	8615	0.492736216	4
TM_NaiveBayes	Gender	Missing	0	0	1
TM_NaiveBayes	Gender	F	8656	0.495081217	4
TM_NaiveBayes	Gender	M	8828	0.504918783	4

For example, these results tell you the number of training cases for each discrete value (VALUETYPE = 4), together with the computed probability, adjusted for missing values (VALUETYPE = 1).

For a definition of the values provided in the NODE_DISTRIBUTION table in a Naive Bayes model, see Mining Model Content for Naive Bayes Models (Analysis Services - Data Mining). For more information about how support and probability calculations are affected by missing values, see Missing Values (Analysis Services - Data Mining).

Sample Query 3: Finding More Information about Attributes

Because a Naive Bayes model often contains complex information about the relationships among different attributes, the easiest way to view these relationships is to use the Microsoft Naive Bayes Viewer. However, you can create DMX queries to return the data.

The following example shows how to return information from the model about a particular attribute, `Region`.

```
SELECT NODE_TYPE, NODE_CAPTION,
NODE_PROBABILITY, NODE_SUPPORT, MSOLAP_NODE_SCORE
FROM TM_NaiveBayes.CONTENT
WHERE ATTRIBUTE_NAME = 'Region'
```

This query returns two types of nodes: the node that represents the input attribute (NODE_TYPE = 10), and nodes for each value of the attribute (NODE_TYPE = 11). The node caption is used to identify the node, rather than the node name, because the caption shows both the attribute name and attribute value.

NODE_TYPE	NODE_CAPTION	NODE_PROBABILITY	NODE_SUPPORT	MSOLAP_NODE_SCORE	NODE_TYPE
10	Bike Buyer -> Region	1	17484	84.51555875	10
11	Bike Buyer -> Region = Missing	0	0	0	11
11	Bike Buyer -> Region = North America	0.508236102	8886	0	11
11	Bike Buyer -> Region = Pacific	0.193891558	3390	0	11
11	Bike Buyer -> Region = Europe	0.29787234	5208	0	11

Some of the columns stored in the nodes are the same that you can get from the marginal statistics nodes, such as the node probability score and the node support values. However, the MSOLAP_NODE_SCORE is a special value provided only for the input attribute nodes, and indicates the relative importance of this attribute in the model. You can see much the same information in the Dependency Network pane of the viewer; however, the viewer does not provide scores.

The following query returns the importance scores of all attributes in the model:

```
SELECT NODE_CAPTION, MSOLAP_NODE_SCORE
FROM TM_NaiveBayes.CONTENT
WHERE NODE_TYPE = 10
ORDER BY MSOLAP_NODE_SCORE DESC
```

Sample results:

NODE_CAPTION	MSOLAP_NODE_SCORE
Bike Buyer -> Total Children	181.3654836
Bike Buyer -> Commute Distance	179.8419482
Bike Buyer -> English Education	156.9841928
Bike Buyer -> Number Children At Home	111.8122599
Bike Buyer -> Region	84.51555875
Bike Buyer -> Marital Status	23.13297354
Bike Buyer -> English Occupation	2.832069191

By browsing the model content in the Microsoft Generic Content Tree Viewer, you will get a better idea of what statistics might be interesting. Some simple examples were demonstrated here; more often you may need to execute multiple queries or store the results and process them on the client.

Sample Query 4: Using System Stored Procedures

In addition to writing your own content queries, you can use some Analysis Services system stored procedures to explore the results. To use a system stored procedure, prefix the stored procedure name with the CALL keyword:

```
CALL GetPredictableAttributes ('TM_NaiveBayes')
```

Partial results:

ATTRIBUTE_NAME	NODE_UNIQUE_NAME
Bike Buyer	100000001

> **NOTE**
> These system stored procedures are for internal communication between the Analysis Services server and the client and should only be used for convenience when developing and testing mining models. When you create queries for a production system, you should always write your own queries by using DMX.

For more information about Analysis Services system stored procedures, see Data Mining Stored Procedures (Analysis Services - Data Mining).

Using a Naive Bayes Model to Make Predictions

The Microsoft Naive Bayes algorithm is typically used less for prediction than it is for exploration of relationships among the input and predictable attributes. However, the model supports the use of prediction functions for both prediction and association.

Sample Query 5: Predicting Outcomes using a Singleton Query

The following query uses a singleton query to provide a new value and predict, based on the model, whether a customer with these characteristics is likely to buy a bike. The easiest way to create a singleton query on a regression model is by using the **Singleton Query Input** dialog box. For example, you can build the following DMX query by selecting the `TM_NaiveBayes` model, choosing **Singleton Query**, and selecting values from the

dropdown lists for `[Commute Distance]` and `Gender`.

```
SELECT
  Predict([TM_NaiveBayes].[Bike Buyer])
FROM
  [TM_NaiveBayes]
NATURAL PREDICTION JOIN
(SELECT '5-10 Miles' AS [Commute Distance],
  'F' AS [Gender]) AS t
```

Example results:

EXPRESSION
0

The prediction function returns the most likely value, in this case, 0, which means this type of customer is unlikely to purchase a bike.

Sample Query 6: Getting Predictions with Probability and Support Values

In addition to predicting an outcome, you often want to know how strong the prediction is. The following query uses the same singleton query as the previous example, but adds the prediction function, PredictHistogram (DMX), to return a nested table that contains statistics in support of the prediction.

```
SELECT
  Predict([TM_NaiveBayes].[Bike Buyer]),
  PredictHistogram([TM_NaiveBayes].[Bike Buyer])
FROM
  [TM_NaiveBayes]
NATURAL PREDICTION JOIN
(SELECT '5-10 Miles' AS [Commute Distance],
  'F' AS [Gender]) AS t
```

Example results:

BIKE BUYER	$SUPPORT	$PROBABILITY	$ADJUSTEDPROBABILITY	$VARIANCE	$STDEV
0	10161.5714	0.581192599	0.010530981	0	0
1	7321.428768	0.418750215	0.008945684	0	0
	0.999828444	5.72E-05	5.72E-05	0	0

The final row in the table shows the adjustments to support and probability for the missing value. Variance and standard deviation values are always 0, because Naive Bayes models cannot model continuous values.

Sample Query 7: Predicting Associations

The Microsoft Naive Bayes algorithm can be used for association analysis, if the mining structure contains a nested table with the predictable attribute as the key. For example, you could build a Naive Bayes model by using the mining structure created in Lesson 3: Building a Market Basket Scenario (Intermediate Data Mining Tutorial) of the data mining tutorial. The model used in this example was modified to add information about income and customer region in the case table.

The following query example shows a singleton query that predicts products that are related to purchases of the product, `'Road Tire Tube'`. You might use this information to recommend products to a specific type of customer.

```
SELECT   PredictAssociation([Association].[v Assoc Seq Line Items])
FROM [Association_NB]
NATURAL PREDICTION JOIN
(SELECT 'High' AS [Income Group],
  'Europe' AS [Region],
  (SELECT 'Road Tire Tube' AS [Model])
AS [v Assoc Seq Line Items])
AS t
```

Partial results:

MODEL
Women's Mountain Shorts
Water Bottle
Touring-3000
Touring-2000
Touring-1000

Function List

All Microsoft algorithms support a common set of functions. However, the Microsoft Naive Bayes algorithm supports the additional functions that are listed in the following table.

Prediction Function	Usage
IsDescendant (DMX)	Determines whether one node is a child of another node in the model.
Predict (DMX)	Returns a predicted value, or set of values, for a specified column.
PredictAdjustedProbability (DMX)	Returns the weighted probability.
PredictAssociation (DMX)	Predicts membership in an associative dataset.
PredictNodeId (DMX)	Returns the Node_ID for each case.
PredictProbability (DMX)	Returns probability for the predicted value.
PredictSupport (DMX)	Returns the support value for a specified state.

To see the syntax of specific functions, see Data Mining Extensions (DMX) Function Reference.

See Also

Microsoft Naive Bayes Algorithm Technical Reference
Microsoft Naive Bayes Algorithm
Mining Model Content for Naive Bayes Models (Analysis Services - Data Mining)

Microsoft Neural Network Algorithm

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Neural Network algorithm is an implementation of the popular and adaptable neural network architecture for machine learning. The algorithm works by testing each possible state of the input attribute against each possible state of the predictable attribute, and calculating probabilities for each combination based on the training data. You can use these probabilities for both classification or regression tasks, to predict an outcome based on some input attributes. A neural network can also be used for association analysis.

When you create a mining model using the Microsoft Neural Network algorithm, you can include multiple outputs, and the algorithm will create multiple networks. The number of networks contained in a single mining model contains depends on the number of states (or attribute values) in the input columns, as well as the number of predictable columns that the mining model uses and the number of states in those columns.

Example

The Microsoft Neural Network algorithm is useful for analyzing complex input data, such as from a manufacturing or commercial process, or business problems for which a significant quantity of training data is available but for which rules cannot be easily derived by using other algorithms.

Suggested scenarios for using the Microsoft Neural Network algorithm include the following:

- Marketing and promotion analysis, such as measuring the success of a direct mail promotion or a radio advertising campaign
- Predicting stock movement, currency fluctuation, or other highly fluid financial information from historical data
- Analyzing manufacturing and industrial processes
- Text mining
- Any prediction model that analyzes complex relationships between many inputs and relatively fewer outputs

How the Algorithm Works

The Microsoft Neural Network algorithm creates a network that is composed of up to three layers of nodes (sometimes called *neurons*). These layers are the *input layer*, the *hidden layer*, and the *output layer*.

Input layer: Input nodes define all the input attribute values for the data mining model, and their probabilities.

Hidden layer: Hidden nodes receive inputs from input nodes and provide outputs to output nodes. The hidden layer is where the various probabilities of the inputs are assigned weights. A weight describes the relevance or importance of a particular input to the hidden node. The greater the weight that is assigned to an input, the more important the value of that input is. Weights can be negative, which means that the input can inhibit, rather than favor, a specific result.

Output layer: Output nodes represent predictable attribute values for the data mining model.

For a detailed explanation of how the input, hidden, and output layers are constructed and scored, see Microsoft Neural Network Algorithm Technical Reference.

Data Required for Neural Network Models

A neural network model must contain a key column, one or more input columns, and one or more predictable columns.

Data mining models that use the Microsoft Neural Network algorithm are heavily influenced by the values that you specify for the parameters that are available to the algorithm. The parameters define how data is sampled, how data is distributed or expected to be distributed in each column, and when feature selection is invoked to limit the values that are used in the final model.

For more information about setting parameters to customize model behavior, see Microsoft Neural Network Algorithm Technical Reference.

Viewing a Neural Network Model

To work with the data and see how the model correlates inputs with outputs, you can use the **Microsoft Neural Network Viewer**. With this custom viewer, you can filter on input attributes and their values, and see graphs that show how they affect the outputs. Tooltips in the viewer show the probability and lift associated with each pair of input and output values. For more information, see Browse a Model Using the Microsoft Neural Network Viewer.

The easiest way to explore the structure of the model is to use the **Microsoft Generic Content Tree Viewer**. You can view the inputs, outputs, and networks created by the model, and click on any node to expand it and see statistics related to the input, output, or hidden layer nodes. For more information, see Browse a Model Using the Microsoft Generic Content Tree Viewer.

Creating Predictions

After the model has been processed, you can use the network and the weights stored within each node to make predictions. A neural network model supports regression, association, and classification analysis, Therefore, the meaning of each prediction might be different. You can also query the model itself, to review the correlations that were found and retrieve related statistics. For examples of how to create queries against a neural network model, see Neural Network Model Query Examples.

For general information about how to create a query on a data mining model, see Data Mining Queries.

Remarks

- Does not support drillthrough or data mining dimensions. This is because the structure of the nodes in the mining model does not necessarily correspond directly to the underlying data.
- Does not support the creation of models in Predictive Model Markup Language (PMML) format.
- Supports the use of OLAP mining models.
- Does not support the creation of data mining dimensions.

See Also

Microsoft Neural Network Algorithm Technical Reference
Mining Model Content for Neural Network Models (Analysis Services - Data Mining)
Neural Network Model Query Examples
Microsoft Logistic Regression Algorithm

Microsoft Neural Network Algorithm Technical Reference

11/28/2018 • 13 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Neural Network uses a *Multilayer Perceptron* network, also called a *Back-Propagated Delta Rule network*, composed of up to three layers of neurons, or *perceptrons*. These layers are an input layer, an optional hidden layer, and an output layer.

A detailed discussion of Multilayer Perceptron neural networks is outside the scope of this documentation. This topic explains the basic implementation of the algorithm, including the method used to normalize input and output values, and feature selection methods used to reduce attribute cardinality. This topic describes the parameters and other settings that can be used to customize the behavior of the algorithm, and provides links to additional information about querying the model.

Implementation of the Microsoft Neural Network Algorithm

In a Multilayer Perceptron neural network, each neuron receives one or more inputs and produces one or more identical outputs. Each output is a simple non-linear function of the sum of the inputs to the neuron. Inputs pass forward from nodes in the input layer to nodes in the hidden layer, and then pass from the hidden layer to the output layer; there are no connections between neurons within a layer. If no hidden layer is included, as in a logistic regression model, inputs pass forward directly from nodes in the input layer to nodes in the output layer.

There are three types of neurons in a neural network that is created with the Microsoft Neural Network algorithm:

Input neurons

Input neurons provide input attribute values for the data mining model. For discrete input attributes, an input neuron typically represents a single state from the input attribute. This includes missing values, if the training data contains nulls for that attribute. A discrete input attribute that has more than two states generates one input neuron for each state, and one input neuron for a missing state, if there are any nulls in the training data. A continuous input attribute generates two input neurons: one neuron for a missing state, and one neuron for the value of the continuous attribute itself. Input neurons provide inputs to one or more hidden neurons.

Hidden neurons

Hidden neurons receive inputs from input neurons and provide outputs to output neurons.

Output neurons

Output neurons represent predictable attribute values for the data mining model. For discrete input attributes, an output neuron typically represents a single predicted state for a predictable attribute, including missing values. For example, a binary predictable attribute produces one output node that describes a missing or existing state, to indicate whether a value exists for that attribute. A Boolean column that is used as a predictable attribute generates three output neurons: one neuron for a true value, one neuron for a false value, and one neuron for a missing or existing state. A discrete predictable attribute that has more than two states generates one output neuron for each state, and one output neuron for a missing or existing state. Continuous predictable columns generate two output neurons: one neuron for a missing or existing state, and one neuron for the value of the continuous column itself. If more than 500 output neurons are generated by reviewing the set of predictable columns, Analysis Services generates a new network in the mining model to represent the additional output

neurons.

A neuron receives input from other neurons, or from other data, depending on which layer of the network it is in. An input neuron receives inputs from the original data. Hidden neurons and output neurons receive inputs from the output of other neurons in the neural network. Inputs establish relationships between neurons, and the relationships serve as a path of analysis for a specific set of cases.

Each input has a value assigned to it, called the *weight*, which describes the relevance or importance of that particular input to the hidden neuron or the output neuron. The greater the weight that is assigned to an input, the more relevant or important the value of that input. Weights can be negative, which implies that the input can inhibit, rather than activate, a specific neuron. The value of each input is multiplied by the weight to emphasize the importance of an input for a specific neuron. For negative weights, the effect of multiplying the value by the weight is to deemphasize the importance.

Each neuron has a simple non-linear function assigned to it, called the *activation function*, which describes the relevance or importance of a particular neuron to that layer of a neural network. Hidden neurons use a *hyperbolic tangent* function (tanh) for their activation function, whereas output neurons use a *sigmoid* function for activation. Both functions are nonlinear, continuous functions that allow the neural network to model nonlinear relationships between input and output neurons.

Training Neural Networks

Several steps are involved in training a data mining model that uses the Microsoft Neural Network algorithm. These steps are heavily influenced by the values that you specify for the algorithm parameters.

The algorithm first evaluates and extracts training data from the data source. A percentage of the training data, called the *holdout data*, is reserved for use in assessing the accuracy of the network. Throughout the training process, the network is evaluated immediately after each iteration through the training data. When the accuracy no longer increases, the training process is stopped.

The values of the *SAMPLE_SIZE* and *HOLDOUT_PERCENTAGE* parameters are used to determine the number of cases to sample from the training data and the number of cases to be put aside for the holdout data. The value of the *HOLDOUT_SEED* parameter is used to randomly determine the individual cases to be put aside for the holdout data.

> **NOTE**
> These algorithm parameters are different from the HOLDOUT_SIZE and HOLDOUT_SEED properties, which are applied to a mining structure to define a testing data set.

The algorithm next determines the number and complexity of the networks that the mining model supports. If the mining model contains one or more attributes that are used only for prediction, the algorithm creates a single network that represents all such attributes. If the mining model contains one or more attributes that are used for both input and prediction, the algorithm provider constructs a network for each attribute.

For input and predictable attributes that have discrete values, each input or output neuron respectively represents a single state. For input and predictable attributes that have continuous values, each input or output neuron respectively represents the range and distribution of values for the attribute. The maximum number of states that is supported in either case depends on the value of the *MAXIMUM_STATES* algorithm parameter. If the number of states for a specific attribute exceeds the value of the *MAXIMUM_STATES* algorithm parameter, the most popular or relevant states for that attribute are chosen, up to the maximum number of states allowed, and the remaining states are grouped as missing values for the purposes of analysis.

The algorithm then uses the value of the *HIDDEN_NODE_RATIO* parameter when determining the initial number of neurons to create for the hidden layer. You can set *HIDDEN_NODE_RATIO* to 0 to prevent the creation of a hidden layer in the networks that the algorithm generates for the mining model, to treat the neural

network as a logistic regression.

The algorithm provider iteratively evaluates the weight for all inputs across the network at the same time, by taking the set of training data that was reserved earlier and comparing the actual known value for each case in the holdout data with the network's prediction, in a process known as *batch learning*. After the algorithm has evaluated the entire set of training data, the algorithm reviews the predicted and actual value for each neuron. The algorithm calculates the degree of error, if any, and adjusts the weights that are associated with the inputs for that neuron, working backward from output neurons to input neurons in a process known as *backpropagation*. The algorithm then repeats the process over the entire set of training data. Because the algorithm can support many weights and output neurons, the conjugate gradient algorithm is used to guide the training process for assigning and evaluating weights for inputs. A discussion of the conjugate gradient algorithm is outside the scope of this documentation.

Feature Selection

If the number of input attributes is greater than the value of the *MAXIMUM_INPUT_ATTRIBUTES* parameter, or if the number of predictable attributes is greater than the value of the *MAXIMUM_OUTPUT_ATTRIBUTES* parameter, a feature selection algorithm is used to reduce the complexity of the networks that are included in the mining model. Feature selection reduces the number of input or predictable attributes to those that are most statistically relevant to the model.

Feature selection is used automatically by all Analysis Services data mining algorithms to improve analysis and reduce processing load. The method used for feature selection in neural network models depends on the data type of the attribute. For reference, the following table shows the feature selection methods used for neural network models, and also shows the feature selection methods used for the Logistic Regression algorithm, which is based on the Neural Network algorithm.

ALGORITHM	METHOD OF ANALYSIS	COMMENTS
Neural Network	Interestingness score Shannon's Entropy Bayesian with K2 Prior Bayesian Dirichlet with uniform prior (default)	The Neural Networks algorithm can use both entropy-based and Bayesian scoring methods, as long as the data contains continuous columns. Default.
Logistic Regression	Interestingness score Shannon's Entropy Bayesian with K2 Prior Bayesian Dirichlet with uniform prior (default)	Because you cannot pass a parameter to this algorithm to control feature election behavior, the defaults are used. Therefore, if all attributes are discrete or discretized, the default is BDEU.

The algorithm parameters that control feature selection for a neural network model are MAXIMUM_INPUT_ATTRIBUTES, MAXIMUM_OUTPUT_ATTRIBUTES, and MAXIMUM_STATES. You can also control the number of hidden layers by setting the HIDDEN_NODE_RATIO parameter.

Scoring Methods

Scoring is a kind of normalization, which in the context of training a neural network model means the process of converting a value, such as a discrete text label, into a value that can be compared with other types of inputs and weighted in the network. For example, if one input attribute is Gender and the possible values are Male and Female, and another input attribute is Income, with a variable range of values, the values for each attribute are not directly comparable, and therefore must be encoded to a common scale so that the weights can be

computed. Scoring is the process of normalizing such inputs to numeric values: specifically, to a probability range. The functions used for normalization also help to distribute input value more evenly on a uniform scale so that extreme values do not distort the results of analysis.

Outputs of the neural network are also encoded. When there is a single target for output (that is, prediction), or multiple targets that are used for prediction only and not for input, the model create a single network and it might not seem necessary to normalize the values. However, if multiple attributes are used for input and prediction, the model must create multiple networks; therefore, all values must be normalized, and the outputs too must be encoded as they exit the network.

Encoding for inputs is based on summing each discrete value in the training cases, and multiplying that value by its weight. This is called a *weighted sum*, which is passed to the activation function in the hidden layer. A z-score is used for encoding, as follows:

Discrete values

`μ = p` - the prior probability of a state

`StdDev = sqrt(p(1-p))`

Continuous values

Value present= `1 - μ/σ`

No existing value= `-μ/σ`

After the values have been encoded, the inputs go through weighted summing, with network edges as weights.

Encoding for outputs uses the sigmoid function, which has properties that make it very useful for prediction. One such property is that, regardless of how the original values are scaled, and regardless of whether values are negative or positive, the output of this function is always a value between 0 and 1, which is suited for estimating probabilities. Another useful property is that the sigmoid function has a smoothing effect, so that as values move farther away from point of inflection, the probability for the value moves towards 0 or 1, but slowly.

Customizing the Neural Network Algorithm

The Microsoft Neural Network algorithm supports several parameters that affect the behavior, performance, and accuracy of the resulting mining model. You can also modify the way that the model processes data by setting modeling flags on columns, or by setting distribution flags to specify how values within the column are handled.

Setting Algorithm Parameters

The following table describes the parameters that can be used with the Microsoft Neural Network algorithm.

HIDDEN_NODE_RATIO
Specifies the ratio of hidden neurons to input and output neurons. The following formula determines the initial number of neurons in the hidden layer:

HIDDEN_NODE_RATIO * SQRT(Total input neurons * Total output neurons)

The default value is 4.0.

HOLDOUT_PERCENTAGE
Specifies the percentage of cases within the training data used to calculate the holdout error, which is used as part of the stopping criteria while training the mining model.

The default value is 30.

HOLDOUT_SEED
Specifies a number that is used to seed the pseudo-random generator when the algorithm randomly determines

the holdout data. If this parameter is set to 0, the algorithm generates the seed based on the name of the mining model, to guarantee that the model content remains the same during reprocessing.

The default value is 0.

MAXIMUM_INPUT_ATTRIBUTES
Determines the maximum number of input attributes that can be supplied to the algorithm before feature selection is employed. Setting this value to 0 disables feature selection for input attributes.

The default value is 255.

MAXIMUM_OUTPUT_ATTRIBUTES
Determines the maximum number of output attributes that can be supplied to the algorithm before feature selection is employed. Setting this value to 0 disables feature selection for output attributes.

The default value is 255.

MAXIMUM_STATES
Specifies the maximum number of discrete states per attribute that is supported by the algorithm. If the number of states for a specific attribute is greater than the number that is specified for this parameter, the algorithm uses the most popular states for that attribute and treats the remaining states as missing.

The default value is 100.

SAMPLE_SIZE
Specifies the number of cases to be used to train the model. The algorithm uses either this number or the percentage of total of cases not included in the holdout data as specified by the HOLDOUT_PERCENTAGE parameter, whichever value is smaller.

In other words, if HOLDOUT_PERCENTAGE is set to 30, the algorithm will use either the value of this parameter, or a value equal to 70 percent of the total number of cases, whichever is smaller.

The default value is 10000.

Modeling Flags

The following modeling flags are supported for use with the Microsoft Neural Network algorithm.

NOT NULL
Indicates that the column cannot contain a null. An error will result if Analysis Services encounters a null during model training.

Applies to mining structure columns.

MODEL_EXISTENCE_ONLY
Indicates that the model should only consider whether a value exists for the attribute or if a value is missing. The exact value does not matter.

Applies to mining model columns.

Distribution Flags

The following distribution flags are supported for use with the Microsoft Neural Network algorithm. The flags are used as hints to the model only; if the algorithm detects a different distribution it will use the found distribution, not the distribution provided in the hint.

Normal
Indicates that values within the column should be treated as though they represent the normal, or Gaussian, distribution.

Uniform
Indicates that values within the column should be treated as though they are distributed uniformly; that is, the

probability of any value is roughly equal, and is a function of the total number of values.

Log Normal
Indicates that values within the column should be treated as though distributed according to the *log normal* curve, which means that the logarithm of the values is distributed normally.

Requirements

A neural network model must contain at least one input column and one output column.

Input and Predictable Columns

The Microsoft Neural Network algorithm supports the specific input columns and predictable columns that are listed in the following table.

COLUMN	CONTENT TYPES
Input attribute	Continuous, Cyclical, Discrete, Discretized, Key, Table, and Ordered
Predictable attribute	Continuous, Cyclical, Discrete, Discretized, and Ordered

> **NOTE**
> Cyclical and Ordered content types are supported, but the algorithm treats them as discrete values and does not perform special processing.

See Also

Microsoft Neural Network Algorithm
Mining Model Content for Neural Network Models (Analysis Services - Data Mining)
Neural Network Model Query Examples

Mining Model Content for Neural Network Models (Analysis Services - Data Mining)

5/16/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes mining model content that is specific to models that use the Microsoft Neural Network algorithm. For an explanation of how to interpret statistics and structure shared by all model types, and general definitions of terms related to mining model content, see Mining Model Content (Analysis Services - Data Mining).

Understanding the Structure of a Neural Network Model

Each neural network model has a single parent node that represents the model and its metadata, and a marginal statistics node (NODE_TYPE = 24) that provides descriptive statistics about the input attributes. The marginal statistics node is useful because it summarizes information about inputs, so that you do not need to query data from the individual nodes.

Underneath these two nodes, there are at least two more nodes, and might be many more, depending on how many predictable attributes the model has.

- The first node (NODE_TYPE = 18) always represents the top node of the input layer. Beneath this top node, you can find input nodes (NODE_TYPE = 21) that contain the actual input attributes and their values.
- Successive nodes each contain a different *subnetwork* (NODE_TYPE = 17). Each subnetwork always contains a hidden layer (NODE_TYPE = 19), and an output layer (NODE_TYPE = 20) for that subnetwork.

The information in the input layer is straightforward: the top node for each input layer (NODE_TYPE = 18) serves as an organizer for a collection of input nodes (NODE_TYPE = 21). The content of the input nodes is described in the following table.

Each subnetwork (NODE_TYPE = 17) represents the analysis of the influence of the input layer on a particular predictable attribute. If there are multiple predictable outputs, there are multiple subnetworks. The hidden layer for each subnetwork contains multiple hidden nodes (NODE_TYPE = 22) that contain details about the weights for each transition that ends in that particular hidden node.

The output layer (NODE_TYPE = 20) contains output nodes (NODE_TYPE = 23) that each contain distinct values of the predictable attribute. If the predictable attribute is a continuous numeric data type, there is only one output node for the attribute.

> **NOTE**
> The logistic regression algorithm uses a special case of a neural network that has only one predictable outcome and potentially many inputs. Logistic regression does not use a hidden layer.

The easiest way to explore the structure of the inputs and subnetworks is to use the **Microsoft Generic Content Tree viewer**. You can click any node to expand it and see the child nodes, or view the weights and other statistics that is contained in the node.

To work with the data and see how the model correlates inputs with outputs, use the **Microsoft Neural Network Viewer**. By using this custom viewer, you can filter on input attributes and their values, and graphically see how they affect the outputs. The tooltips in the viewer show you the probability and lift associated with each pair of inputs and output values. For more information, see Browse a Model Using the Microsoft Neural Network Viewer.

Model Content for a Neural Network Model

This section provides detail and examples only for those columns in the mining model content that have particular relevance for neural networks. For information about general-purpose columns in the schema rowset, such as MODEL_CATALOG and MODEL_NAME, that are not described here, or for explanations of mining model terminology, see Mining Model Content (Analysis Services - Data Mining).

MODEL_CATALOG
Name of the database where the model is stored.

MODEL_NAME
Name of the model.

ATTRIBUTE_NAME
The names of the attributes that correspond to this node.

NODE	CONTENT
Model root	Blank
Marginal statistics	Blank
Input layer	Blank
Input node	Input attribute name
Hidden layer	Blank
Hidden node	Blank
Output layer	Blank
Output node	Output attribute name

NODE_NAME
The name of the node. This column contains the same value as NODE_UNIQUE_NAME.

NODE_UNIQUE_NAME
The unique name of the node.

For more information about how the names and IDs provide structural information about the model, see the section, Using Node Names and IDs.

NODE_TYPE

A neural network model outputs the following node types:

NODE TYPE ID	DESCRIPTION
1	Model.
17	Organizer node for the subnetwork.
18	Organizer node for the input layer.
19	Organizer node for the hidden layer.
20	Organizer node for the output layer.
21	Input attribute node.
22	Hidden layer node
23	Output attribute node.
24	Marginal statistics node.

NODE_CAPTION

A label or a caption associated with the node. In neural network models, always blank.

CHILDREN_CARDINALITY

An estimate of the number of children that the node has.

NODE	CONTENT
Model root	Indicates the count of child nodes, which includes at least 1 network, 1 required marginal node, and 1 required input layer. For example, if the value is 5, there are 3 subnetworks.
Marginal statistics	Always 0.
Input layer	Indicates the number of input attribute-values pairs that were used by the model.
Input node	Always 0.
Hidden layer	Indicates the number of hidden nodes that were created by the model.
Hidden node	Always 0.
Output layer	Indicates the number of output values.
Output node	Always 0.

PARENT_UNIQUE_NAME

The unique name of the node's parent. NULL is returned for any nodes at the root level.

For more information about how the names and IDs provide structural information about the model, see the section, Using Node Names and IDs.

NODE_DESCRIPTION

A user-friendly description of the node.

NODE	CONTENT
Model root	Blank
Marginal statistics	Blank
Input layer	Blank
Input node	Input attribute name
Hidden layer	Blank
Hidden node	Integer that indicates the sequence of the hidden node in the list of hidden nodes.
Output layer	Blank
Output node	If the output attribute is continuous, contains the name of the output attribute. If the output attribute is discrete or discretized, contains the name of the attribute and the value.

NODE_RULE

An XML description of the rule that is embedded in the node.

NODE	CONTENT
Model root	Blank
Marginal statistics	Blank
Input layer	Blank
Input node	An XML fragment that contains the same information as the NODE_DESCRIPTION column.
Hidden layer	Blank
Hidden node	Integer that indicates the sequence of the hidden node in the list of hidden nodes.
Output layer	Blank
Output node	An XML fragment that contains the same information as the NODE_DESCRIPTION column.

MARGINAL_RULE

For neural network models, always blank.

NODE_PROBABILITY
The probability associated with this node. For neural network models, always 0.

MARGINAL_PROBABILITY
The probability of reaching the node from the parent node. For neural network models, always 0.

NODE_DISTRIBUTION
A nested table that contains statistical information for the node. For detailed information about the contents of this table for each node type, see the section, Understanding the NODE_DISTRIBUTION Table.

NODE_SUPPORT
For neural network models, always 0.

> **NOTE**
> Support probabilities are always 0 because the output of this model type is not probabilistic. Only the weights are meaningful for the algorithm; therefore, the algorithm does not compute probability, support, or variance.

To get information about the support in the training cases for specific values, see the marginal statistics node.

MSOLAP_MODEL_COLUMN

Node	Content
Model root	Blank
Marginal statistics	Blank
Input layer	Blank
Input node	Input attribute name.
Hidden layer	Blank
Hidden node	Blank
Output layer	Blank
Output node	Input attribute name.

MSOLAP_NODE_SCORE
For a neural network model, always 0.

MSOLAP_NODE_SHORT_CAPTION
For neural network models, always blank.

Remarks

The purpose of training a neural network model is to determine the weights that are associated with each transition from an input to a midpoint, and from a midpoint to an endpoint. Therefore, the input layer of the model principally exists to store the actual values that were used to build the model. The hidden layer stores the weights that were computed, and provides pointers back to the input attributes. The output layer stores the predictable values, and also provides pointers back to the midpoints in the hidden layer.

Using Node Names and IDs

The naming of the nodes in a neural network model provides additional information about the type of node, to make it easier to relate the hidden layer to the input layer, and the output layer to the hidden layer. The following table shows the convention for the IDs that are assigned to nodes in each layer.

NODE TYPE	CONVENTION FOR NODE ID
Model root (1)	00000000000000000.
Marginal statistics node (24)	10000000000000000
Input layer (18)	30000000000000000
Input node (21)	Starts at 60000000000000000
Subnetwork (17)	20000000000000000
Hidden layer (19)	40000000000000000
Hidden node (22)	Starts at 70000000000000000
Output layer (20)	50000000000000000
Output node (23)	Starts at 80000000000000000

You can determine which input attributes are related to a specific hidden layer node by viewing the NODE_DISTRIBUTION table in the hidden node (NODE_TYPE = 22). Each row of the NODE_DISTRIBUTION table contains the ID of an input attribute node.

Similarly, you can determine which hidden layers are related to an output attribute by viewing the NODE_DISTRIBUTION table in the output node (NODE_TYPE = 23). Each row of the NODE_DISTRIBUTION table contains the ID of a hidden layer node, together with the related coefficient.

Interpreting the Information in the NODE_DISTRIBUTION Table

The NODE_DISTRIBUTION table can be empty in some nodes. However, for input nodes, hidden layer nodes, and output nodes, the NODE_DISTRIBUTION table stores important and interesting information about the model. To help you interpret this information, the NODE_DISTRIBUTION table contains a VALUETYPE column for each row that tells you whether the value in the ATTRIBUTE_VALUE column is Discrete (4), Discretized (5), or Continuous (3).

Input Nodes

The input layer contains a node for each value of the attribute that was used in the model.

Discrete attribute: The input node stores only the name of the attribute and its value in the ATTRIBUTE_NAME and ATTRIBUTE_VALUE columns. For example, if [Work Shift] is the column, a separate node is created for each value of that column that was used in the model, such as AM and PM. The NODE_DISTRIBUTION table for each node lists only the current value of the attribute.

Discretized numeric attribute: The input node stores the name of the attribute, and the value, which can be a range or a specific value. All values are represented by expressions, such as '77.4 - 87.4' or ' < 64.0' for the value of the [Time Per Issue]. The NODE_DISTRIBUTION table for each node lists only the current value of the attribute.

Continuous attribute: The input node stores the mean value of the attribute. The NODE_DISTRIBUTION table for each node lists only the current value of the attribute.

Hidden Layer Nodes

The hidden layer contains a variable number of nodes. In each node, the NODE_DISTRIBUTION table contains

mappings from the hidden layer to the nodes in the input layer. The ATTRIBUTE_NAME column contains a node ID that corresponds to a node in the input layer. The ATTRIBUTE_VALUE column contains the weight associated with that combination of input node and hidden layer node. The last row in the table contains a coefficient that represents the weight of that hidden node in the hidden layer.

Output Nodes

The output layer contains one output node for each output value that was used in the model. In each node, the NODE_DISTRIBUTION table contains mappings from the output layer to the nodes in the hidden layer. The ATTRIBUTE_NAME column contains a node ID that corresponds to a node in the hidden layer. The ATTRIBUTE_VALUE column contains the weight associated with that combination of output node and hidden layer node.

The NODE_DISTRIBUTION table has the following additional information, depending on whether the type of the attribute:

Discrete attribute: The final two rows of the NODE_DISTRIBUTION table contain a coefficient for the node as a whole, and the current value of the attribute.

Discretized numeric attribute: Identical to discrete attributes, except that the value of the attribute is a range of values.

Continuous attribute: The final two rows of the NODE_DISTRIBUTION table contain the mean of the attribute, the coefficient for the node as a whole, and the variance of the coefficient.

See Also

Microsoft Neural Network Algorithm
Microsoft Neural Network Algorithm Technical Reference
Neural Network Model Query Examples

Neural Network Model Query Examples

11/28/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a query against a data mining model, you can create a content query, which provides details about the patterns discovered in analysis, or a prediction query, which uses the patterns in the model to make predictions for new data. For example, a content query for a neural network model might retrieve model metadata such as the number of hidden layers. Alternatively, a prediction query might suggest classifications based on an input and optionally provide probabilities for each classification.

This section explains how to create queries for models that are based on the Microsoft Neural Network algorithm.

Content queries

Getting Model Metadata by Using DMX

Retrieving Model Metadata from the Schema Rowset

Retrieving the Input Attributes for the Model

Retrieving Weights from the Hidden Layer

Prediction queries

Creating a Singleton Prediction

Finding Information about a Neural Network Model

All mining models expose the content learned by the algorithm according to a standardized schema, the mining model schema rowset. This information provides details about the model and includes the basic metadata, structures discovered in analysis, and parameters that are used when processing. You can create queries against the model content by using Data Mining Extension (DMX) statements.

Sample Query 1: Getting Model Metadata by Using DMX

The following query returns some basic metadata about a model that was built by using the Microsoft Neural Network algorithm. In a neural network model, the parent node of the model contains only the name of the model, the name of the database where the model is stored, and the number of child nodes. However, the marginal statistics node (NODE_TYPE = 24) provides both this basic metadata and some derived statistics about the input columns used in the model.

The following sample query is based on the mining model that you create in the Intermediate Data Mining Tutorial, named `Call Center Default NN`. The model uses data from a call center to explore possible correlations between staffing and the number of calls, orders, and issues. The DMX statement retrieves data from the marginal statistics node of the neural network model. The query includes the FLATTENED keyword, because the input attribute statistics of interest are stored in a nested table, NODE_DISTRIBUTION. However, if your query provider supports hierarchical rowsets you do not need to use the FLATTENED keyword.

```
SELECT FLATTENED MODEL_CATALOG, MODEL_NAME,
(    SELECT ATTRIBUTE_NAME, ATTRIBUTE_VALUE,
     [SUPPORT], [PROBABILITY], VALUETYPE
     FROM NODE_DISTRIBUTION
) AS t
FROM [Call Center Default NN].CONTENT
WHERE NODE_TYPE = 24
```

NOTE

You must enclose the name of the nested table columns SUPPORT and PROBABILITY in brackets to distinguish them from the reserved keywords of the same name.

Example results:

MODEL_CATALOG	MODEL_NAME	T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE	T.SUPPORT	T.PROBABILITY	T.VALUETYPE
Adventure Works DW Multidimensional 2012	Call Center NN	Average Time Per Issue	Missing	0	0	1
Adventure Works DW Multidimensional 2012	Call Center NN	Average Time Per Issue	< 64.7094100096	11	0.407407407	5

For a definition of what the columns in the schema rowset mean in the context of a neural network model, see Mining Model Content for Neural Network Models (Analysis Services - Data Mining).

Sample Query 2: Retrieving Model Metadata from the Schema Rowset

You can find the same information that is returned in a DMX content query by querying the data mining schema rowset. However, the schema rowset provides some additional columns. The following sample query returns the date that the model was created, the date it was modified, and the date that the model was last processed. The query also returns the predictable columns, which are not easily available from the model content, and the parameters that were used to build the model. This information can be useful for documenting the model.

```
SELECT MODEL_NAME, DATE_CREATED, LAST_PROCESSED, PREDICTION_ENTITY, MINING_PARAMETERS
from $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = 'Call Center Default NN'
```

Example results:

MODEL_NAME	Call Center Default NN
DATE_CREATED	1/10/2008 5:07:38 PM
LAST_PROCESSED	1/10/2008 5:24:02 PM

PREDICTION_ENTITY	Average Time Per Issue, Grade Of Service, Number Of Orders
MINING_PARAMETERS	HOLDOUT_PERCENTAGE=30, HOLDOUT_SEED=0, MAXIMUM_INPUT_ATTRIBUTES=255, MAXIMUM_OUTPUT_ATTRIBUTES=255, MAXIMUM_STATES=100, SAMPLE_SIZE=10000, HIDDEN_NODE_RATIO=4

Sample Query 3: Retrieving the Input Attributes for the Model

You can retrieve the input attribute-value pairs that were used to create the model by querying the child nodes (NODE_TYPE = 20) of the input layer (NODE_TYPE = 18). The following query returns a list of input attributes from the node descriptions.

```
SELECT NODE_DESCRIPTION
FROM [Call Center Default NN].CONTENT
WHERE NODE_TYPE = 2
```

Example results:

NODE_DESCRIPTION
Average Time Per Issue=64.7094100096 - 77.4002099712
Day Of Week=Fri.
Level 1 Operators

Only a few representative rows from the results are shown here. However, you can see that the NODE_DESCRIPTION provides slightly different information depending on the data type of the input attribute.

- If the attribute is a discrete or discretized value, the attribute and either its value or its discretized range are returned.
- If the attribute is a continuous numeric data type, the NODE_DESCRIPTION contains only the attribute name. However, you can retrieve the nested NODE_DISTRIBUTION table to obtain the mean, or return the NODE_RULE to obtain the minimum and maximum values of the numeric range.

The following query shows how to query the nested NODE_DISTRIBUTION table to return the attributes in one column, and their values in another column. For continuous attributes, the value of the attribute is represented by its mean.

```
SELECT FLATTENED
(SELECT ATTRIBUTE_NAME, ATTRIBUTE_VALUE
FROM NODE_DISTRIBUTION) as t
FROM [Call Center Default NN -- Predict Service and Orders].CONTENT
WHERE NODE_TYPE = 21
```

Example results:

T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE
Average Time Per Issue	64.7094100096 - 77.4002099712
Day Of Week	Fri.
Level 1 Operators	3.2962962962963

The minimum and maximum range values are stored in the NODE_RULE column, and are represented as an XML fragment, as shown in the following example:

```
<NormContinuous field="Level 1 Operators">
  <LinearNorm orig="2.83967303681711" norm="-1" />
  <LinearNorm orig="3.75291955577548" norm="1" />
</NormContinuous>
```

Sample Query 4: Retrieving Weights from the Hidden Layer

The model content of a neural network model is structured in a way that makes it easy to retrieve details about any node in the network. Moreover, the ID numbers of the nodes provide information that helps you identify relationships among the node types.

The following query demonstrates how to retrieve the coefficients that are stored under a particular node of the hidden layer. The hidden layer consists of an organizer node (NODE_TYPE = 19), which contains only metadata, and multiple child nodes (NODE_TYPE = 22), which contain the coefficients for the various combinations of attributes and values. This query returns only the coefficient nodes.

```
SELECT FLATTENED TOP 1 NODE_UNIQUE_NAME,
(SELECT ATTRIBUTE_NAME, ATTRIBUTE_VALUE, VALUETYPE
FROM NODE_DISTRIBUTION) as t
FROM  [Call Center Default NN -- Predict Service and Orders].CONTENT
WHERE NODE_TYPE = 22
AND [PARENT_UNIQUE_NAME] = '40000000200000000' FROM [Call Center Default NN].CONTENT
```

Example results:

NODE_UNIQUE_NAME	T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE	T.VALUETYPE
70000000200000000	6000000000000000a	-0.178616518	7
70000000200000000	6000000000000000b	-0.267561918	7
70000000200000000	6000000000000000c	0.11069497	7
70000000200000000	6000000000000000d	0.123757712	7
70000000200000000	6000000000000000e	0.294565343	7
70000000200000000	6000000000000000f	0.22245318	7
70000000200000000		0.188805045	7

The partial results shown here demonstrate how the neural network model content relates the hidden node to the input nodes.

- The unique names of nodes in the hidden layer always begin with 70000000.
- The unique names of nodes in the input layer always begin with 60000000.

Thus, these results tell you that the node denoted by the ID 70000000200000000 had six different coefficients (VALUETYPE = 7) passed to it. The values of the coefficients are in the ATTRIBUTE_VALUE column. You can determine exactly which input attribute the coefficient is for by using the node ID in the ATTRIBUTE_NAME column. For example, the node ID 6000000000000000a refers to input attribute and value, `Day of Week = 'Tue.'` You can use the node ID to create a query, or you can browse to the node by using the Microsoft Generic Content Tree Viewer.

Similarly, if you query the NODE_DISTRIBUTION table of the nodes in the output layer (NODE_TYPE = 23), you can see the coefficients for each output value. However, in the output layer, the pointers refer back to the nodes of the hidden layer. For more information, see Mining Model Content for Neural Network Models (Analysis Services - Data Mining).

Using a Neural Network Model to Make Predictions

The Microsoft Neural Network algorithm supports both classification and regression. You can use prediction functions with these models to provide new data and create either singleton or batch predictions.

Sample Query 5: Creating a Singleton Prediction

The easiest way to build a prediction query on a neural network model is to use the Prediction Query Builder, available on the **Mining Prediction** tab of Data Mining Designer in both SQL Server Management Studio and SQL Server Data Tools (SSDT). You can browse the model in the Microsoft Neural Network Viewer to filter attributes of interest and view trends, and then switch to the **Mining Prediction** tab to create a query and predict new values for those trends.

For example, you can browse the call center model to view correlations between the order volumes and other attributes. To do this, open the model in the viewer, and for **Input**, select **<All>**. Next, for **Output**, select **Number of Orders**. For **Value 1**, select the range that represents the most orders, and for **Value 2**, select the range that represents the fewest orders. You can then see at a glance all the attributes that the model correlates with order volume.

By browsing the results in the viewer, you find that certain days of the week have low order volumes, and that an increase in the number of operators seems to be correlated with higher sales. You could then use a prediction query on the model to test a "what if" hypothesis and ask if increasing the number of level 2 operators on a low-volume day would increase orders. To do this, create a query such as the following:

```
SELECT Predict([Call Center Default NN].[Number of Orders]) AS [Predicted Orders],
PredictProbability([Call Center Default NN].[Number of Orders]) AS [Probability]
FROM [Call Center Default NN]
NATURAL PREDICTION JOIN
(SELECT 'Tue.' AS [Day of Week],
13 AS [Level 2 Operators]) AS t
```

Example results:

PREDICTED ORDERS	PROBABILITY
364	0.9532...

The predicted sales volume is higher than the current range of sales for Tuesday, and the probability of the prediction is very high. However, you might want to create multiple predictions by using a batch process to test a variety of hypotheses on the model.

> **NOTE**
> The Data Mining Add-Ins for Excel 2007 provide logistic regression wizards that make it easy to answer complex questions, such as how many Level Two Operators would be needed to improve service grade to a target level for a specific shift. The data mining add-ins are a free download, and include wizards that are based on the neural network and/or logistic regression algorithms. For more information, see the Data Mining Add-ins for Office 2007 Web site.

List of Prediction Functions

All Microsoft algorithms support a common set of functions. There are no prediction functions that are specific to the Microsoft Neural Network algorithm; however, the algorithm supports the functions that are listed in the following table.

Prediction Function	Usage
IsDescendant (DMX)	Determines whether one node is a child of another node in the neural network graph.
PredictAdjustedProbability (DMX)	Returns the weighted probability.
PredictHistogram (DMX)	Returns a table of values related to the current predicted value.
PredictVariance (DMX)	Returns variance for the predicted value.
PredictProbability (DMX)	Returns probability for the predicted value.
PredictStdev (DMX)	Returns the standard deviance for the predicted value.
PredictSupport (DMX)	For neural network and logistic regression models, returns a single value that represents the size of the training set for the entire model.

For a list of the functions that are common to all Microsoft algorithms, see Algorithm Reference (Analysis Services - Data Mining). For the syntax of specific functions, see Data Mining Extensions (DMX) Function Reference.

See Also

Microsoft Neural Network Algorithm
Microsoft Neural Network Algorithm Technical Reference
Mining Model Content for Neural Network Models (Analysis Services - Data Mining)
Lesson 5: Building Neural Network and Logistic Regression Models (Intermediate Data Mining Tutorial)

Microsoft Sequence Clustering Algorithm

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Sequence Clustering algorithm is a unique algorithm that combines sequence analysis with clustering. You can use this algorithm to explore data that contains events that can be linked in a *sequence*. The algorithm finds the most common sequences, and performs clustering to find sequences that are similar. The following examples illustrate the types of sequences that you might capture as data for machine learning, to provide insight about common problems or business scenarios:

- Clickstreams or click paths generated when users navigate or browse a Web site
- Logs that list events preceding an incident, such as a hard disk failure or server deadlock
- Transaction records that describe the order in which a customer adds items to a online shopping cart
- Records that follow customer or patient interactions over time, to predict service cancellations or other poor outcomes

This algorithm is similar in many ways to the Microsoft Clustering algorithm. However, instead of finding clusters of cases that contain similar attributes, the Microsoft Sequence Clustering algorithm finds clusters of cases that contain similar paths in a sequence.

Example

The Adventure Works Cycles web site collects information about what pages site users visit, and about the order in which the pages are visited. Because the company provides online ordering, customers must log in to the site. This provides the company with click information for each customer profile. By using the Microsoft Sequence Clustering algorithm on this data, the company can find groups, or clusters, of customers who have similar patterns or sequences of clicks. The company can then use these clusters to analyze how users move through the Web site, to identify which pages are most closely related to the sale of a particular product, and to predict which pages are most likely to be visited next.

How the Algorithm Works

The Microsoft Sequence Clustering algorithm is a hybrid algorithm that combines clustering techniques with Markov chain analysis to identify clusters and their sequences. One of the hallmarks of the Microsoft Sequence Clustering algorithm is that it uses sequence data. This data typically represents a series of events or transitions between states in a dataset, such as a series of product purchases or Web clicks for a particular user. The algorithm examines all transition probabilities and measures the differences, or distances, between all the possible sequences in the dataset to determine which sequences are the best to use as inputs for clustering. After the algorithm has created the list of candidate sequences, it uses the sequence information as an input for clustering using Expectation maximization (EM).

For a detailed description of the implementation, see Microsoft Sequence Clustering Algorithm Technical Reference.

Data Required for Sequence Clustering Models

When you prepare data for use in training a sequence clustering model, you should understand the requirements for the particular algorithm, including how much data is needed, and how the data is used.

The requirements for a sequence clustering model are as follows:

- **A single key column** A sequence clustering model requires a key that identifies records.
- **A sequence column** For sequence data, the model must have a nested table that contains a sequence ID column. The sequence ID can be any sortable data type. For example, you can use a Web page identifier, an integer, or a text string, as long as the column identifies the events in a sequence. Only one sequence identifier is allowed for each sequence, and only one type of sequence is allowed in each model.
- **Optional non sequence attributes** The algorithm supports the addition of other attributes that are not related to sequencing. These attributes can include nested columns.

For example, in the example cited earlier of the Adventure Works Cycles Web site, a sequence clustering model might include order information as the case table, demographics about the specific customer for each order as non-sequence attributes, and a nested table containing the sequence in which the customer browsed the site or put items into a shopping cart as the sequence information.

For more detailed information about the content types and data types supported for sequence clustering models, see the Requirements section of Microsoft Sequence Clustering Algorithm Technical Reference.

Viewing a Sequence Clustering Model

The mining model that this algorithm creates contains descriptions of the most common sequences in the data. To explore the model, you can use the **Microsoft Sequence Cluster Viewer**. When you view a sequence clustering model, Analysis Services shows you clusters that contain multiple transitions. You can also view pertinent statistics. For more information, see Browse a Model Using the Microsoft Sequence Cluster Viewer.

If you want to know more detail, you can browse the model in the Microsoft Generic Content Tree Viewer. The content stored for the model includes the distribution for all values in each node, the probability of each cluster, and details about the transitions. For more information, see Mining Model Content for Sequence Clustering Models (Analysis Services - Data Mining).

Creating Predictions

After the model has been trained, the results are stored as a set of patterns. You can use the descriptions of the most common sequences in the data to predict the next likely step of a new sequence. However, because the algorithm includes other columns, you can use the resulting model to identify relationships between sequenced data and inputs that are not sequential. For example, if you add demographic data to the model, you can make predictions for specific groups of customers. Prediction queries can be customized to return a variable number of predictions, or to return descriptive statistics.

For information about how to create queries against a data mining model, see Data Mining Queries. For examples of how to use queries with a sequence clustering model, see Sequence Clustering Model Query Examples.

Remarks

- Does not support the use of Predictive Model Markup Language (PMML) to create mining models.
- Supports drillthrough.
- Supports the use of OLAP mining models and the creation of data mining dimensions.

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Microsoft Sequence Clustering Algorithm Technical Reference

Sequence Clustering Model Query Examples
Browse a Model Using the Microsoft Sequence Cluster Viewer

Microsoft Sequence Clustering Algorithm Technical Reference

11/27/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Sequence Clustering algorithm is a hybrid algorithm that uses Markov chain analysis to identify ordered sequences, and combines the results of this analysis with clustering techniques to generate clusters based on the sequences and other attributes in the model. This topic describes the implementation of the algorithm, how to customize the algorithm, and special requirements for sequence clustering models.

For more general information about the algorithm, including how to browse and query sequence clustering models, see Microsoft Sequence Clustering Algorithm.

Implementation of the Microsoft Sequence Clustering Algorithm

The Microsoft Sequence Clustering model uses Markov models to identify sequences and determine the probability of sequences. A Markov model is a directed graph that stores the transitions between different states. The Microsoft Sequence Clustering algorithm uses n-order Markov chains, not a Hidden Markov model.

The number of orders in a Markov chain tells you how many states are used to determine the probability of the current states. In a first-order Markov model, the probability of the current state depends only on the previous state. In a second-order Markov chain, the probability of a state depends on the previous two states, and so forth. For each Markov chain, a transition matrix stores the transitions for each combination of states. As the length of the Markov chain increases, the size of the matrix also increases exponentially, and the matrix becomes extremely sparse. Processing time also increases proportionally.

It might be helpful to visualize the chain by using the example of clickstream analysis, which analyzes visits to Web pages on a site. Each user creates a long sequence of clicks for each session. When you create a model to analyze user behavior on a Web site, the data set used for training is a sequence of URLs, converted to a graph that includes the count of all instances of the same click path. For example, the graph contains the probability that the user moves from page 1 to page 2 (10%), the probability that the user moves from page 1 to page 3 (20%), and so forth. When you put all the possible paths and pieces of the paths together, you obtain a graph that might be much longer and more complex than any single observed path.

By default, the Microsoft Sequence Clustering algorithm uses the Expectation Maximization (EM) method of clustering. For more information, see Microsoft Clustering Algorithm Technical Reference.

The targets of clustering are both the sequential and nonsequential attributes. Each cluster is randomly selected using a probability distribution. Each cluster has a Markov chain that represents the complete set of paths, and a matrix that contains the sequence state transitions and probabilities. Based on the initial distribution, Bayes rule is used to calculate the probability of any attribute, including a sequence, in a specific cluster.

The Microsoft Sequence Clustering algorithm supports the additional of nonsequential attributes to the model. This means that these additional attributes are combined with the sequence attributes to create clusters of cases with similar attributes, just like in a typical clustering model.

A sequence clustering model tends to create many more clusters than a typical clustering model. Therefore, the Microsoft Sequence Clustering algorithm performs *cluster decomposition*to separate clusters based on sequences and other attributes.

Feature Selection in a Sequence Clustering Model

Feature selection is not invoked when building sequences; however, feature selection applies at the clustering stage.

MODEL TYPE	FEATURE SELECTION METHOD	COMMENTS
Sequence Clustering	Not used	Feature selection is not invoked; however, you can control the behavior of the algorithm by setting the value of the parameters MINIMUM_SUPPORT and MINIMUM_PROBABILIITY.
Clustering	Interestingness score	Although the clustering algorithm may use discrete or discretized algorithms, the score of each attribute is calculated as a distance and is continuous; therefore the interestingness score is used.

For more information, see Feature Selection.

Optimizing Performance

The Microsoft Sequence Clustering algorithm supports various ways to optimize processing:

- Controlling the number of clusters generated, by setting a value for the CLUSTER_COUNT parameter.
- Reducing the number of sequences included as attributes, by increasing the value of the MINIMUM_SUPPORT parameter. As a result, rare sequences are eliminated.
- Reducing complexity before processing the model, by grouping related attributes.

In general, you can optimize the performance of an n-order Markov chain mode in several ways:

- Controlling the length of the possible sequences.
- Programmatically reducing the value of n.
- Storing only probabilities that exceed a specified threshold.

A complete discussion of these methods is beyond the scope of this topic.

Customizing the Sequence Clustering Algorithm

The Microsoft Sequence Clustering algorithm supports parameters that affect the behavior, performance, and accuracy of the resulting mining model. You can also modify the behavior of the completed model by setting modeling flags that control the way the algorithm processes training data.

Setting Algorithm Parameters

The following table describes the parameters that can be used with the Microsoft Sequence Clustering algorithm.

CLUSTER_COUNT
Specifies the approximate number of clusters to be built by the algorithm. If the approximate number of clusters cannot be built from the data, the algorithm builds as many clusters as possible. Setting the CLUSTER_COUNT parameter to 0 causes the algorithm to use heuristics to best determine the number of clusters to build.

The default is 10.

> **NOTE**
> Specifying a non-zero number acts as a hint to the algorithm, which proceeds with the goal of finding the specified number, but may end up finding more or less.

MINIMUM_SUPPORT
Specifies the minimum number of cases that is required in support of an attribute to create a cluster.

The default is 10.

MAXIMUM_SEQUENCE_STATES
Specifies the maximum number of states that a sequence can have.

Setting this value to a number greater than 100 may cause the algorithm to create a model that does not provide meaningful information.

The default is 64.

MAXIMUM_STATES
Specifies the maximum number of states for a non-sequence attribute that the algorithm supports. If the number of states for a non-sequence attribute is greater than the maximum number of states, the algorithm uses the attribute's most popular states and treats the remaining states as **Missing**.

The default is 100.

Modeling Flags

The following modeling flags are supported for use with the Microsoft Sequence Clustering algorithm.

NOT NULL
Indicates that the column cannot contain a null. An error will result if Analysis Services encounters a null during model training.

Applies to the mining structure column.

MODEL_EXISTENCE_ONLY
Means that the column will be treated as having two possible states: **Missing** and **Existing**. A null is treated as a **Missing** value.

Applies to the mining model column.

For more information about the use of Missing values in mining models, and how missing values affect probability scores, see Missing Values (Analysis Services - Data Mining).

Requirements

The case table must have a case ID column. Optionally the case table can contain other columns that store attributes about the case.

The Microsoft Sequence Clustering algorithm requires sequence information, stored as a nested table. The nested table must have a single Key Sequence column. A **Key Sequence** column can contain any type of data that can be sorted, including string data types, but the column must contain unique values for each case. Moreover, before you process the model, you must ensure that both the case table and the nested table are sorted in ascending order on the key that relates the tables.

> **NOTE**
> If you create a model that uses the Microsoft Sequence algorithm but do not use a sequence column, the resulting model will not contain any sequences, but will simply cluster cases based on other attributes that are included in the model.

Input and Predictable Columns

The Microsoft Sequence Clustering algorithm supports the specific input columns and predictable columns that are listed in the following table. For more information about what the content types mean when used in a mining model, see Content Types (Data Mining).

COLUMN	CONTENT TYPES
Input attribute	Continuous, Cyclical, Discrete, Discretized, Key, Key Sequence, Table, and Ordered
Predictable attribute	Continuous, Cyclical, Discrete, Discretized, Table, and Ordered

Remarks

- Use the PredictSequence (DMX) function for Prediction of Sequences. For more information about the editions of SQL Server that support Sequence Prediction, see Features Supported by the Editions of SQL Server 2012 (http://go.microsoft.com/fwlink/?linkid=232473).
- The Microsoft Sequence Clustering algorithm does not support using the Predictive Model Markup Language (PMML) to create mining models.
- The Microsoft Sequence Clustering algorithm supports drillthrough, the use of OLAP mining models, and the use of data mining dimensions.

See Also

Microsoft Sequence Clustering Algorithm
Sequence Clustering Model Query Examples
Mining Model Content for Sequence Clustering Models (Analysis Services - Data Mining)

Mining Model Content for Sequence Clustering Models

5/16/2018 • 12 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes mining model content that is specific to models that use the Microsoft Sequence Clustering algorithm. For an explanation of general and statistical terminology related to mining model content that applies to all model types, see Mining Model Content (Analysis Services - Data Mining).

Understanding the Structure of a Sequence Clustering Model

A sequence clustering model has a single parent node (NODE_TYPE = 1) that represents the model and its metadata. The parent node, which is labeled **(All)**, has a related sequence node (NODE_TYPE = 13) that lists all the transitions that were detected in the training data.

The algorithm also creates a number of clusters, based on the transitions that were found in the data and any other input attributes included when creating the model, such as customer demographics and so forth. Each cluster (NODE_TYPE = 5) contains its own sequence node (NODE_TYPE = 13) that lists only the transitions that were used in generating that specific cluster. From the sequence node, you can drill down to view the details of individual state transitions (NODE_TYPE = 14).

For an explanation of sequence and state transitions, with examples, see Microsoft Sequence Clustering Algorithm.

Model Content for a Sequence Clustering Model

This section provides additional information about columns in the mining model content that have particular relevance for sequence clustering.

MODEL_CATALOG
Name of the database where the model is stored.

MODEL_NAME
Name of the model.

ATTRIBUTE_NAME
Always blank.

NODE_NAME
The name of the node. Currently the same value as NODE_UNIQUE_NAME.

NODE_UNIQUE_NAME

The unique name of the node.

NODE_TYPE
A sequence clustering model outputs the following node types:

NODE TYPE ID	DESCRIPTION
1 (Model)	Root node for model
5 (Cluster)	Contains a count of transitions in the cluster, a list of the attributes, and statistics that describe the values in the cluster.
13 (Sequence)	Contains a list of transitions included in the cluster.
14 (Transition)	Describes a sequence of events as a table in which the first row contains the starting state, and all other rows contain successive states, together with support and probability statistics.

NODE_GUID
Blank.

NODE_CAPTION
A label or a caption associated with the node for display purposes.

You can rename the cluster captions while you are using the model; however, the new name is not persisted if you close the model.

CHILDREN_CARDINALITY
An estimate of the number of children that the node has.

Model root Cardinality value equals the number of clusters plus one. For more information, see Cardinality.

Cluster nodes Cardinality is always 1, because each cluster has a single child node, which contains the list of sequences in the cluster.

Sequence nodes Cardinality indicates the number of transitions that are included in that cluster. For example, the cardinality of the sequence node for the model root tells you how many transitions were found in the entire model.

PARENT_UNIQUE_NAME
The unique name of the node's parent.

NULL is returned for any nodes at the root level.

NODE_DESCRIPTION
Same as node caption.

NODE_RULE
Always blank.

MARGINAL_RULE
Always blank.

NODE_PROBABILITY
Model root Always 0.

Cluster nodes The adjusted probability of the cluster in the model. The adjusted probabilities do not sum to 1, because the clustering method used in sequence clustering permits partial membership in multiple clusters.

Sequence nodes Always 0.

Transition nodes Always 0.

MARGINAL_PROBABILITY
Model root Always 0.

Cluster nodes The same value as NODE_PROBABILITY.

Sequence nodes Always 0.

Transition nodes Always 0.

NODE_DISTRIBUTION
A table that contains probabilities and other information. For more information, see NODE_DISTRIBUTION Table.

NODE_SUPPORT
The number of transitions that support this node. Therefore, if there are 30 examples of sequence "Product A followed by Product B" in the training data, the total support is 30.

Model root Total number of transitions in the model.

Cluster nodes Raw support for the cluster, meaning the number of training cases that contribute cases to this cluster.

Sequence nodes Always 0.

Transition nodes Percentage of cases in the cluster that represent a specific transition. Can be 0, or can have a positive value. Calculated by taking the raw support for the cluster node, and multiplying by the probability of the cluster.

From this value, you can tell how many training cases contributed to the transition.

MSOLAP_MODEL_COLUMN
Not applicable.

MSOLAP_NODE_SCORE
Not applicable.

MSOLAP_NODE_SHORT_CAPTION
Same as NODE_DESCRIPTION.

Understanding Sequences, States and Transitions

A sequence clustering model has a unique structure that combines two kinds of objects with very different types of information: the first are clusters, and the second are state transitions.

The clusters created by sequence clustering are like the clusters created by the Microsoft Clustering algorithm. Each cluster has a profile and characteristics. However, in sequence clustering, each cluster additionally contains a single child node that lists the sequences in that cluster. Each sequence node contains multiple child nodes that describe the state transitions in detail, with probabilities.

There are almost always more sequences in the model than you can find in any single case, because the sequences can be chained together. Microsoft Analysis Services stores pointers from one state to the other so that you can count the number of times each transition happens. You can also find information about how many times the sequence occurred, and measure its probability of occurring as compared to the entire set of observed states.

The following table summarizes how information is stored in the model, and how the nodes are related.

NODE	HAS CHILD NODE	NODE_DISTRIBUTION TABLE
Model root	Multiple cluster nodes Node with sequences for entire model	Lists all products in the model, with support and probability. Because the clustering method permits partial membership in multiple clusters, support and probability can have fractional values. That is, instead of counting a single case once, each case can potentially belong to multiple clusters. Therefore, when the final cluster membership is determined, the value is adjusted by the probability of that cluster.
Sequence node for model	Multiple transition nodes	Lists all products in the model, with support and probability. Because the number of sequences is known for the model, at this level, calculations for support and probability are straightforward: Support = count of cases Probability = raw probability of each sequence in model. All probabilities should sum to 1.
Individual cluster nodes	Node with sequences for that cluster only	Lists all products in a cluster, but provides support and probability values only for products that are characteristic of the cluster. Support represents the adjusted support value for each case in this cluster. Probability values are adjusted probability.
Sequence nodes for individual clusters	Multiple nodes with transitions for sequences in that cluster only	Exactly the same information as in individual cluster nodes.
Transitions	No children	Lists transitions for the related first state. Support is an adjusted support value, indicating the cases that take part in each transition. Probability is the adjusted probability, represented as a percentage.

NODE_DISTRIBUTION Table

The NODE_DISTRIBUTION table provides detailed probability and support information for the transitions and sequences for a specific cluster.

A row is always added to the transition table to represent possible **Missing** values. For information about what the **Missing** value means, and how it affects calculations, see Missing Values (Analysis Services - Data Mining).

The calculations for support and probability differ depending on whether the calculation applies to the training cases or to the finished model. This is because the default clustering method, Expectation Maximization (EM), assumes that any case can belong to more than one cluster. When calculating support for the cases in the model, it is possible to use raw counts and raw probabilities. However, the probabilities for any particular sequence in a cluster must be weighted by the sum of all possible sequence and cluster combinations.

Cardinality

In a clustering model, the cardinality of the parent node generally tells you how many clusters are in the model. However, a sequence clustering model has two kinds of nodes at the cluster level: one kind of node contains clusters, and the other kind of node contains a list of sequences for the model as a whole.

Therefore, to learn the number of clusters in the model, you can take the value of NODE_CARDINALITY for the (All) node and subtract one. For example, if the model created 9 clusters, the cardinality of the model root is 10. This is because the model contains 9 cluster nodes, each with its own sequence node, plus one additional sequence node labeled cluster 10, which represents the sequences for the model.

Walkthrough of Structure

An example might help clarify how the information is stored, and how you can interpret it. For example, you can find the largest order, meaning the longest observed chain in the underlying **AdventureWorksDW2012** data, by using the following query:

```
USE AdventureWorksDW2012
SELECT DISTINCT OrderNumber, Count(*)
FROM vAssocSeqLineItems
GROUP BY OrderNumber
ORDER BY Count(*) DESC
```

From these results, you find that the order numbers 'SO72656', 'SO58845', and 'SO70714' contain the largest sequences, with eight items each. By using the order IDs, you can view the details of a particular order to see which items were purchased, and in what order.

ORDERNUMBER	LINENUMBER	MODEL
SO58845	1	Mountain-500
SO58845	2	LL Mountain Tire
SO58845	3	Mountain Tire Tube
SO58845	4	Fender Set - Mountain
SO58845	5	Mountain Bottle Cage
SO58845	6	Water Bottle
SO58845	7	Sport-100
SO58845	8	Long-Sleeve Logo Jersey

However, some customers who purchase the Mountain-500 might purchase different products. You can view all the products that follow the Mountain-500 by viewing the list of sequences in the model. The following procedures walk you through viewing these sequences by using the two viewers provided in Analysis Services:

To view related sequences by using the Sequence Clustering viewer

1. In Object Explorer, right-click the [Sequence Clustering] model, and select Browse.
2. In the Sequence Clustering viewer, click the **State Transitions** tab.
3. In the **Cluster** dropdown list, ensure that **Population (All)** is selected.
4. Move the slider bar at the left of the pane all the way to the top, to show all links.
5. In the diagram, locate **Mountain-500**, and click the node in the diagram.
6. The highlighted lines point to the next states (the products that were purchased after the Mountain-500) and the numbers indicate the probability. Compare these to the results in the generic model content viewer.

To view related sequences by using the generic model content viewer

1. In Object Explorer, right-click the [Sequence Clustering] model, and select Browse.
2. In the viewer dropdown list, select the **Microsoft Generic Content Tree Viewer**.
3. In the **Node caption** pane, click the node named **Sequence level for cluster 16**.
4. In the Node details pane, find the NODE_DISTRIBUTION row, and click anywhere in the nested table.

   The top row is always for the Missing value. This row is sequence state 0.
5. Press the down arrow key, or use the scroll bars, to move down through the nested table until you see the row, Mountain-500.

   This row is sequence state 20.

   > **NOTE**
   > You can obtain the row number for a particular sequence state programmatically, but if you are just browsing, it might be easier to simply copy the nested table into an Excel workbook.
6. Return to the Node caption pane, and expand the node, **Sequence level for cluster 16**, if it is not already expanded.
7. Look among its child nodes for **Transition row for sequence state 20**. Click the transition node.
8. The nested NODE_DISTRIBUTION table contains the following products and probabilities. Compare these to the results in the **State Transition** tab of the Sequence Clustering viewer.

The following table shows the results from the NODE_DISTRIBUTION table, together with the rounded probability values that are displayed in the graphical viewer.

PRODUCT	SUPPORT (NODE_DISTRIBUTION TABLE)	PROBABILITY (NODE_DISTRIBUTION) TABLE)	PROBABILITY (FROM GRAPH)
Missing	48.447887	0.138028169	(not shown)
Cycling Cap	10.876056	0.030985915	0.03
Fender Set - Mountain	80.087324	0.228169014	0.23
Half-Finger Gloves	0.9887324	0.002816901	0.00
Hydration Pack	0.9887324	0.002816901	0.00
LL Mountain Tire	51.414085	0.146478873	0.15

PRODUCT	SUPPORT (NODE_DISTRIBUTION TABLE)	PROBABILITY (NODE_DISTRIBUTION) TABLE)	PROBABILITY (FROM GRAPH)
Long-Sleeve Logo Jersey	2.9661972	0.008450704	0.01
Mountain Bottle Cage	87.997183	0.250704225	0.25
Mountain Tire Tube	16.808451	0.047887324	0.05
Short-Sleeve Classic Jersey	10.876056	0.030985915	0.03
Sport-100	20.76338	0.05915493	0.06
Water Bottle	18.785915	0.053521127	0.25

Although the case that we initially selected from the training data contained the product 'Mountain-500' followed by 'LL Mountain Tire', you can see that there are many other possible sequences. To find detailed information for any particular cluster, you must repeat the process of drilling down from the list of sequences in the cluster to the actual transitions for each state, or product.

You can jump from the sequence listed in one particular cluster, to the transition row. From that transition row, you can determine which product is next, and jump back to that product in the list of sequences. By repeating this process for each first and second state you can work through long chains of states.

Using Sequence Information

A common scenario for sequence clustering is to track user clicks on a Web site. For example, if the data were from records of customer purchases on the Adventure Works e-commerce Web site, the resulting sequence clustering model could be used to infer user behavior, to redesign the e-commerce site to solve navigation problems, or to promote sales.

For example, analysis might show that users always follow a particular chain of products, regardless of demographics. Also, you might find that users frequently exit the site after clicking on a particular product. Given that finding, you might ask what additional paths you could provide to users that would induce users to stay on the Web site.

If you do not have additional information to use in classifying your users, then you can simply use the sequence information to collect data about navigation to better understand overall behavior. However, if you can collect information about customers and match that information with your customer database, you can combine the power of clustering with prediction on sequences to provide recommendations that are tailored to the user, or perhaps based on the path of navigation to the current page.

Another use of the extensive state and transition information compiled by a sequence clustering model is to determine which possible paths are never used. For example, if you have many visitors going to pages 1-4, but visitors never continue on to page 5, you might investigate whether there are problems that prevent navigation to page 5. You can do this by querying the model content, and comparing it against a list of possible paths. Graphs that tell you all the navigation paths in a Web site can be created programmatically, or by using a variety of site analysis tools.

To find out how to obtain the list of observed paths by querying the model content, and to see other examples of queries on a sequence clustering model, see Sequence Clustering Model Query Examples.

See Also

Mining Model Content (Analysis Services - Data Mining)

Microsoft Sequence Clustering Algorithm
Sequence Clustering Model Query Examples

Sequence Clustering Model Query Examples

5/16/2018 • 13 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a query against a data mining model, you can create either a content query, which provides details about the information stored in the model, or you can create a prediction query, which uses the patterns in the model to make predictions based on new data that you provide. For a sequence clustering model, content queries typically provide additional details about the clusters that were found, or the transitions within those clusters. You can also retrieve metadata about the model by using a query.

Prediction queries on a sequence clustering model typically make recommendations based either on the sequences and transitions, on non-sequence attributes that were included in the model, or on a combination of sequence and non-sequence attributes.

This section explains how to create queries for models that are based on the Microsoft Sequence Clustering algorithm. For general information about creating queries, see Data Mining Queries.

Content Queries

Using the Data Mining Schema Rowset to return model parameters

Getting a list of sequences for a state

Using system stored procedures

Prediction Queries

Predict next state or states

Finding Information about the Sequence Clustering Model

To create meaningful queries on the content of a mining model, you must understand the structure of the model content, and which node types store what kind of information. For more information, see Mining Model Content for Sequence Clustering Models (Analysis Services - Data Mining).

Sample Query 1: Using the Data Mining Schema Rowset to Return Model Parameters

By querying the data mining schema rowset, you can find various kinds of information about the model, including basic metadata, the date and time that the model was created and last processed, the name of the mining structure that the model is based on, and the column used as the predictable attribute.

The following query returns the parameters that were used to build and train the model, `[Sequence Clustering]`. You can create this model in Lesson 5 of the Basic Data Mining Tutorial.

```
SELECT MINING_PARAMETERS
from $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = 'Sequence Clustering'
```

Example results:

MINING_PARAMETERS
CLUSTER_COUNT=15,MINIMUM_SUPPORT=10,MAXIMUM_STATES=100,MAXIMUM_SEQUENCE_STATES=64

Note that this model was built by using the default value of 10 for CLUSTER_COUNT. When you specify a non-zero number of clusters for CLUSTER_COUNT, the algorithm treats this number as a hint for the approximate number of clusters to find. However, in the process of analysis, the algorithm may find more or fewer clusters. In this case, the algorithm found that 15 clusters best fit the training data. Therefore, the list of parameter values for the completed model reports the count of clusters as determined by the algorithm, not the value passed in when creating the model.

How does this behavior differ from letting the algorithm determine the best number of clusters? As an experiment, you can create another clustering model that uses this same data, but set CLUSTER_COUNT to 0. When you do this, the algorithm detects 32 clusters. Therefore, by using the default value of 10 for CLUSTER_COUNT, you constrain the number of results.

The value of 10 is used by default because reducing the number of clusters makes it easier for most people to browse and understand groupings in the data. However, each model and set of data is different. You may wish to experiment with different numbers of clusters to see which parameter value yields the most accurate model.

Sample Query 2: Getting a List of Sequences for a State

The mining model content stores the sequences that are found in the training data as a first state coupled with a list of all related second states. The first state is used as the label for the sequence, and the related second states are called transitions.

For example, the following query returns the complete list of first states in the model, before the sequences are grouped into clusters. You can get this list by returning the list of sequences (NODE_TYPE = 13) that have the model root node as parent (PARENT_UNIQUE_NAME = 0). The FLATTENED keyword makes the results easier to read.

> **NOTE**
>
> The name of the columns, PARENT_UNIQUE_NAME, Support, and Probability must be enclosed in brackets to distinguish them from the reserved keywords of the same name.

```
SELECT FLATTENED NODE_UNIQUE_NAME,
(SELECT ATTRIBUTE_VALUE AS [Product 1],
[Support] AS [Sequence Support],
[Probability] AS [Sequence Probability]
FROM NODE_DISTRIBUTION) AS t
FROM [Sequence Clustering].CONTENT
WHERE NODE_TYPE = 13
AND [PARENT_UNIQUE_NAME] = 0
```

Partial results:

NODE_UNIQUE_NAME	PRODUCT 1	SEQUENCE SUPPORT	SEQUENCE PROBABILITY
1081327	Missing	0	#######
1081327	All-Purpose Bike Stand	17	0.00111
1081327	Bike Wash	64	0.00418
1081327	(rows 4-36 omitted)		
1081327	Women's Mountain Shorts	506	0.03307

The list of sequences in the model is always sorted alphabetically in ascending order. The ordering of the

sequences is important because you can find the related transitions by looking at the order number of the sequence. The **Missing** value is always transition 0.

For example, in the previous results, the product "Women's Mountain Shorts" is the sequence number 37 in the model. You can use that information to view all of the products that were ever purchased after "Women's Mountain Shorts."

To do this, first, you reference the value returned for NODE_UNIQUE_NAME in the previous query, to get the ID of the node that contains all sequences for the model. You pass this value to the query as the ID of the parent node, to get only the transitions included in this node, which happens to contain a list of al sequences for the model. However, if you wanted to see the list of transitions for a specific cluster, you could pass in the ID of the cluster node, and see only the sequences associated with that cluster.

```
SELECT NODE_UNIQUE_NAME
FROM [Sequence Clustering].CONTENT
WHERE NODE_DESCRIPTION = 'Transition row for sequence state 37'
AND [PARENT_UNIQUE_NAME] = '1081327'
```

Example results:

NODE_UNIQUE_NAME
1081365

The node represented by this ID contains a list of the sequences that follow the "Women's Mountain Shorts" product, together with the support and probability values.

```
SELECT FLATTENED
(SELECT ATTRIBUTE_VALUE AS Product2,
[Support] AS [P2 Support],
[Probability] AS [P2 Probability]
FROM NODE_DISTRIBUTION) AS t
FROM [Sequence Clustering].CONTENT
WHERE NODE_UNIQUE_NAME = '1081365'
```

Example results:

T.PRODUCT2	T.P2 SUPPORT	T.P2 PROBABILITY
Missing	230.7419	0.456012
Classic Vest	8.16129	0.016129
Cycling Cap	60.83871	0.120235
Half-Finger Gloves	30.41935	0.060117
Long-Sleeve Logo Jersey	86.80645	0.171554
Racing Socks	28.93548	0.057185
Short-Sleeve Classic Jersey	60.09677	0.118768

Note that support for the various sequences related to Women's Mountain Shorts is 506 in the model. The support values for the transitions also add up to 506. However, the numbers are not whole numbers, which

seems a bit odd if you expect support to simply represent a count of cases that contain each transition. However, because the method for creating clusters calculates partial membership, the probability of any transition within a cluster must be weighted by its probability of belonging to that particular cluster.

For example, if there are four clusters, a particular sequence might have a 40% chance of belonging to cluster 1, a 30% chance of belonging to cluster 2, a 20% chance of belonging to cluster 3, and a 10% chance of belonging to cluster 4. After the algorithm determines the cluster that the transition is mostly likely to belong to, it weights the probabilities within the cluster by the cluster prior probability.

Sample Query 3: Using System Stored Procedures

You can see from these query samples that the information stored in the model is complex, and that you might need to create multiple queries to get the information that you need. However, the Microsoft Sequence Clustering viewer provides a powerful set of tools for graphically browsing the information contained in a sequence clustering model, and you can also use the viewer to query and drill down into the model.

In most cases, the information that is presented in the Microsoft Sequence Clustering viewer is created by using Analysis Services system stored procedures to query the model. You can write Data Mining Extensions (DMX) queries against the model content to retrieve the same information, but the Analysis Services system stored procedures provide a convenient shortcut when for exploration or for testing models.

> **NOTE**
>
> System stored procedures are used for internal processing by the server and by the clients that Microsoft provides for interacting with the Analysis Services server. Therefore, Microsoft reserves the right to change them at any time. Although they are described here for your convenience, we do not support their use in a production environment. To ensure stability and compatibility in a production environment, you should always write your own queries by using DMX.

This section provides some samples of how to use the system stored procedures to create queries against a sequence clustering model:

Cluster Profiles and Sample Cases

The Cluster Profiles tab shows you a list of the clusters in the model, the size of each cluster, and a histogram that indicates the states included in the cluster. There are two system stored procedures that you can use in queries to retrieve similar information:

- `GetClusterProfile` returns the characteristics of the cluster, with all the information that is found in the NODE_DISTRIBUTION table for the cluster.
- `GetNodeGraph` returns nodes and edges that can be used to construct a mathematical graph representation of the clusters, corresponding to what you see on the first tab of the Sequence Clustering view. The nodes are clusters, and the edges represent weights or strength.

The following example illustrates how to use the system stored procedure, `GetClusterProfiles`, to return all of the clusters in the model, with their respective profiles. This stored procedure executes a series of DMX statements that return the complete set of profiles in the model. However, to use this stored procedure you must know the address of the model.

```
CALL System.Microsoft.AnalysisServices.System.DataMining.Clustering.GetClusterProfiles('Sequence Clustering',
2147483647, 0)
```

The following example illustrates how to retrieve the profile for a specific cluster, Cluster 12, by using the system stored procedure `GetNodeGraph`, and specifying the cluster ID, which is usually the same as the number in the cluster name.

```
CALL System.Microsoft.AnalysisServices.System.DataMining.Clustering.GetNodeGraph('Sequence
Clustering','12',0)
```

If you omit the cluster ID, as shown in the following query, `GetNodeGraph` returns an ordered, flattened list of all cluster profiles:

```
CALL System.Microsoft.AnalysisServices.System.DataMining.Clustering.GetNodeGraph('Sequence Clustering','',0)
```

The **Cluster Profile** tab also displays a histogram of model sample cases. These sample cases represent the idealized cases for the model. These cases are not stored in the model the same way that the training data is; you must use a special syntax to retrieve the sample cases for the model.

```
SELECT * FROM [Sequence Clustering].SAMPLE_CASES WHERE IsInNode('12')
```

For more information, see SELECT FROM <model>.SAMPLE_CASES (DMX).

Cluster Characteristics and Cluster Discrimination

The **Cluster Characteristics** tab summarizes the main attributes of each cluster, ranked by probability. You can find out how many cases belong to a cluster, and what the distribution of cases is like in the cluster: Each characteristic has certain support. To see the characteristics of a particular cluster, you must know the ID of the cluster.

The following examples uses the system stored procedure, `GetClusterCharacteristics`, to return all the characteristics of Cluster 12 that have a probability score over the specified threshold of 0.0005.

```
CALL System.Microsoft.AnalysisServices.System.DataMining.Clustering.GetClusterCharacteristics('Sequence
Clustering','12',0.0005)
```

To return the characteristics of all clusters, you can leave the cluster ID empty.

```
CALL System.Microsoft.AnalysisServices.System.DataMining.Clustering.GetClusterCharacteristics('Sequence
Clustering','',0.0005)
```

The following example calls the system stored procedure `GetClusterDiscrimination` to compare the characteristics of Cluster 1 and Cluster 12.

```
CALL System.Microsoft.AnalysisServices.System.DataMining.Clustering.GetClusterDiscrimination('Sequence
Clustering','1','12',0.0005,true)
```

If you want to write your own query in DMX to compare two clusters, or compare a cluster with its complement, you must first retrieve one set of characteristics, and then retrieve the characteristics for the specific cluster that you are interested in, and compare the two sets. This scenario is more complicated and typically requires some client processing.

States and Transitions

The **State Transitions** tab of the Microsoft Sequence Clustering performs complicated queries on the back end to retrieve and compare the statistics for different clusters. To reproduce these results requires a more complex query and some client processing.

However, you can use the DMX queries described in Example 2 of the section, Content Queries, to retrieve probabilities and states for sequences or for individual transitions.

Using the Model to Make Predictions

Prediction queries on a sequence clustering model can use many of the prediction functions that are used with other clustering models. In addition, you can use the special prediction function, PredictSequence (DMX), to make recommendations or to predict next states.

Sample Query 4: Predict Next State or States

You can use the PredictSequence (DMX) function to predict the next most likely state, given a value. You can also predict multiple next states: for example, you can return a list of the top three products that a customer is likely to purchase, to present a list of recommendations.

The following sample query is a singleton prediction query that returns the top five predictions, together with their probability. Because the model includes a nested table, you must use the nested table, `[v Assoc Seq Line Items]`, as the column reference when making predictions. Also, when you supply values as input, you must join both the case table and the nested table columns, as shown by the nested SELECT statements.

```
SELECT FLATTENED PredictSequence([v Assoc Seq Line Items], 7)
FROM [Sequence Clustering]
NATURAL PREDICTION JOIN
(SELECT  (SELECT 1 as [Line Number],
   'All-Purpose Bike Stand' as [Model]) AS [v Assoc Seq Line Items])
AS t
```

Example results:

EXPRESSION.$SEQUENCE	EXPRESSION.LINE NUMBER	EXPRESSION.MODEL
1		Cycling Cap
2		Cycling Cap
3		Sport-100
4		Long-Sleeve logo Jersey
5		Half-Finger Gloves
6		All-Purpose Bike Stand
7		All-Purpose Bike Stand

There are three columns in the results, even though you might only expect one column, because the query always returns a column for the case table. Here the results are flattened; otherwise, the query would return a single column that contains two nested table columns.

The column $sequence is a column returned by default by the `PredictSequence` function to order the prediction results. The column, `[Line Number]`, is required to match the sequence keys in the model, but the keys are not output.

Interestingly, the top predicted sequences after All-Purpose Bike Stand are Cycling Cap and Cycling Cap. This is not an error. Depending on how the data is presented to the customer, and how it is grouped when training the model, it is very possible to have sequences of this kind. For example, a customer might purchase a cycling cap (red) and then another cycling cap (blue), or purchase two in a row if there were no way to specify quantity.

The values in rows 6 and 7 are placeholders. When you reach the end of the chain of possible transitions, rather than terminating the prediction results, the value that was passed as an input is added to the results. For example, if you increased the number of predictions to 20, the values for rows 6-20 would all be the same, All-Purpose Bike Stand.

Function List

All Microsoft algorithms support a common set of functions. However, the Microsoft Sequence Clustering algorithm supports the additional functions that are listed in the following table.

Prediction Function	Usage
Cluster (DMX)	Returns the cluster that is most likely to contain the input case
ClusterDistance (DMX)	Returns the distance of the input case from the specified cluster, or if no cluster is specified, the distance of the input case from the most likely cluster. This function can be used with any kind of clustering model (EM, K-Means, etc.), but the results differ depending on the algorithm.
ClusterProbability (DMX)	Returns the probability that the input case belongs to the specified cluster.
IsInNode (DMX)	Indicates whether the specified node contains the current case.
PredictAdjustedProbability (DMX)	Returns the adjusted probability of a specified state.
PredictAssociation (DMX)	Predicts associative membership.
PredictCaseLikelihood (DMX)	Returns the likelihood that an input case will fit in the existing model.
PredictHistogram (DMX)	Returns a table that represents a histogram for the prediction of a given column.
PredictNodeId (DMX)	Returns the Node_ID of the node to which the case is classified.
PredictProbability (DMX)	Returns the probability for a specified state.
PredictSequence (DMX)	Predicts future sequence values for a specified set of sequence data.
PredictStdev (DMX)	Returns the predicted standard deviation for the specified column.
PredictSupport (DMX)	Returns the support value for a specified state.
PredictVariance (DMX)	Returns the variance of a specified column.

For a list of the functions that are common to all Microsoft algorithms, see General Prediction Functions (DMX). For the syntax of specific functions, see Data Mining Extensions (DMX) Function Reference.

See Also

Data Mining Queries
Microsoft Sequence Clustering Algorithm Technical Reference
Microsoft Sequence Clustering Algorithm
Mining Model Content for Sequence Clustering Models (Analysis Services - Data Mining)

Microsoft Time Series Algorithm

5/16/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Time Series algorithm provides multiple algorithms that are optimized for forecasting continuous values, such as product sales, over time. Whereas other Microsoft algorithms, such as decision trees, require additional columns of new information as input to predict a trend, a time series model does not. A time series model can predict trends based only on the original dataset that is used to create the model. You can also add new data to the model when you make a prediction and automatically incorporate the new data in the trend analysis.

The following diagram shows a typical model for forecasting sales of a product in four different sales regions over time. The model that is shown in the diagram shows sales for each region plotted as red, yellow, purple, and blue lines. The line for each region has two parts:

- Historical information appears to the left of the vertical line and represents the data that the algorithm uses to create the model.
- Predicted information appears to the right of the vertical line and represents the forecast that the model makes.

The combination of the source data and the prediction data is called a *series*.

An important feature of the Microsoft Time Series algorithm is that it can perform cross prediction. If you train the algorithm with two separate, but related, series, you can use the resulting model to predict the outcome of one series based on the behavior of the other series. For example, the observed sales of one product can influence the forecasted sales of another product. Cross prediction is also useful for creating a general model that can be applied to multiple series. For example, the predictions for a particular region are unstable because the series lacks good quality data. You could train a general model on an average of all four regions, and then apply the model to the individual series to create more stable predictions for each region.

Example

The management team at Adventure Works Cycles wants to predict monthly bicycle sales for the coming year. The company is especially interested in whether the sale of one bike model can be used to predict the sale of another model. By using the Microsoft Time Series algorithm on historical data from the past three years, the company can produce a data mining model that forecasts future bike sales. Additionally, the company can perform cross predictions to see whether the sales trends of individual bike models are related.

Each quarter, the company plans to update the model with recent sales data and update their predictions to model recent trends. To correct for stores that do not accurately or consistently update sales data, they will create a general prediction model, and use that to create predictions for all regions.

How the Algorithm Works

In SQL Server 2005 (9.x), the Microsoft Time Series algorithm used a single auto-regressive time series method, named ARTXP. The ARTXP algorithm was optimized for short-term predictions, and therefore, excelled at predicting the next likely value in a series. Beginning in SQL Server 2008, the Microsoft Time Series algorithm added a second algorithm, ARIMA, which was optimized for long-term prediction. For a detailed explanation about the implementation of the ARTXP and ARIMA algorithms, see Microsoft Time Series Algorithm Technical Reference.

By default, the Microsoft Time Series algorithm uses a mix of the algorithms when it analyzes patterns and making predictions. The algorithm trains two separate models on the same data: one model uses the ARTXP algorithm, and one model uses the ARIMA algorithm. The algorithm then blends the results of the two models to yield the best prediction over a variable number of time slices. Because ARTXP is best for short-term predictions, it is weighted more heavily at the beginning of a series of predictions. However, as the time slices that you are predicting move further into the future, ARIMA is weighted more heavily.

You can also control the mix of algorithms to favor either short- or long-term prediction in the times series. Beginning in SQL Server 2008 Standard, you can specify that which algorithm to use:

- Use only ARTXP for short-term prediction.
- Use only ARIMA for long-term prediction.
- Use the default blending of the two algorithms.

Beginning in SQL Server 2008 Enterprise, you can also customize how the Microsoft Time Series algorithm blends the models for prediction. When you use a mixed model, the Microsoft Time Series algorithm blends the two algorithms in the following way:

- Only ARTXP is always used for making the first couple of predictions.
- After the first couple of predictions, a combination of ARIMA and ARTXP is used.
- As the number of prediction steps increases, predictions rely more heavily on ARIMA until ARTXP is no longer used.
- You control the mixing point, the rate at which the weight of ARTXP is decreased, and the weight of ARIMA is increased by setting the PREDICTION_SMOOTHING parameter.

Both algorithms can detect seasonality in data at multiple levels. For example, your data might contain monthly cycles nested within yearly cycles. To detect these seasonal cycles, you can either provide a periodicity hint or specify that the algorithm should automatically detect periodicity.

In addition to periodicity, there are several other parameters that control the behavior of the Microsoft Time Series algorithm when it detects periodicity, makes predictions, or analyzes cases. For information about how to set algorithm parameters, see Microsoft Time Series Algorithm Technical Reference.

Data Required for Time Series Models

When you prepare data for use in training any data mining model, make sure that you understand the requirements for the particular model and how the data is used.

Each forecasting model must contain a case series, which is the column that specifies the time slices or other series over which change occurs. For example, the data in the previous diagram shows the series for historical and forecasted bicycle sales over a period of several months. For this model, each region is a series, and the date column contains the time series, which is also the case series. In other models, the case series can be a text field or some identifier such as a customer ID or transaction ID. However, a time series model must always use a date, time, or some other unique numeric value for its case series.

The requirements for a time series model are as follows:

- **A single key time column** Each model must contain one numeric or date column that is used as the case series, which defines the time slices that the model will use. The data type for the key time column can be either a datetime data type or a numeric data type. However, the column must contain continuous values, and the values must be unique for each series. The case series for a time series model cannot be stored in two columns, such as a Year column and a Month column.
- **A predictable column** Each model must contain at least one predictable column around which the algorithm will build the time series model. The data type of the predictable column must have continuous values. For example, you can predict how numeric attributes, such as income, sales, or temperature, change over time. However, you cannot use a column that contains discrete values, such as purchasing status or level of education, as the predictable column.
- **An optional series key column** Each model can have an additional key column that contains unique values that identify a series. The optional series key column must contain unique values. For example, a single model can contain sales for many product models, as long as there is only one record for each product name for every time slice.

You can define input data for the Microsoft Time Series model in several different ways. However, because the format of the input cases affects the definition of the mining model, you must consider your business needs and prepare your data accordingly. The following two examples illustrate how the input data affects the model. In both examples, the completed mining model contains patterns for four distinct series:

- Sales for Product A
- Sales for Product B
- Volume for Product A
- Volume for Product B

In both examples, you can predict new future sales and volume for each product. You cannot predict new values for product or for time.

Example 1: Time Series Data Set with Series Represented as Column Values

This example uses the following table of input cases:

TIMEID	PRODUCT	SALES	VOLUME
1/2001	A	1000	600
2/2001	A	1100	500
1/2001	B	500	900
2/2001	B	300	890

The TimeID column in the table contains a time identifier, and has two entries for each day. The TimeID column becomes the case series. Therefore, you would designate this column as the key time column for the time series model.

The Product column defines a product in the database. This column contains the product series. Therefore, you would designate this column as a second key for the time series model.

The Sales column describes the gross profits of the specified product for one day, and the Volume column describes the quantity of the specified product that remains in the warehouse. These two columns contain the

data that is used to train the model. Both Sales and Volume can be predictable attributes for each series in the Product column.

Example 2: Time Series Data Set with Each Series in Separate Column

Although this example uses basically the same input data as the first example, the input data is structured differently, as shown in the following table:

TIMEID	A_SALES	A_VOLUME	B_SALES	B_VOLUME
1/2001	1000	600	500	900
2/2001	1100	500	300	890

In this table, the TimeID column still contains the case series for the time series model, which you designate as the key time column. However, the previous Sales and Volume columns are now split into two columns and each of those columns are preceded by the product name. As a result, only a single entry exists for each day in the TimeID column. This creates a time series model that would contain four predictable columns: A_Sales, A_Volume, B_Sales, and B_Volume.

Furthermore, because you have separated the products into different columns, you do not have to specify an additional series key column. All the columns in the model are either a case series column or a predictable column.

Viewing a Time Series Model

After the model has been trained, the results are stored as a set of patterns, which you can explore or use to make predictions.

To explore the model, you can use the Time Series Viewer. The viewer includes a chart that displays future predictions, and a tree view of the periodic structures in the data.

If you want to know more about how the predictions are calculated, you can browse the model in the Microsoft Generic Content Tree Viewer. The content stored for the model includes details such as the periodic structures detected by the ARIMA and ARTXP algorithms, the equation used to blend the algorithms, and other statistics.

Creating Time Series Predictions

By default, when you view a time series model, Analysis Services shows you five predictions for the series. However, you can create queries to return a variable number of predictions, and you can extra columns to the predictions to return descriptive statistics. For information about how to create queries against a time series model, see Time Series Model Query Examples. For examples of how to use Data Mining Extensions (DMX) to make time series predictions, see PredictTimeSeries (DMX).

When using the Microsoft Time Series algorithm to make predictions, you should consider the following additional restrictions and requirements:

- Cross-prediction is only available when you use a mixed model, or when you use a model based solely on the ARTXP algorithm. If you use a model based only on the ARIMA algorithm, cross-prediction is not possible.
- A time series model can make predictions that differ, sometimes significantly, depending on the 64-bit operating system that the server uses. These differences occur due to the way that an Itanium-based system represents and handles numbers for floating-point arithmetic, which differs from the way that an x64-based system does these calculations. Because prediction results can be specific to the operating system, we recommend that you evaluate models on the same operating system that you will use in production.

Remarks

- Does not support using the Predictive Model Markup Language (PMML) to create mining models.
- Supports the use of OLAP mining models.
- Does not support the creation of data mining dimensions.
- Supports drillthrough.

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Browse a Model Using the Microsoft Time Series Viewer
Microsoft Time Series Algorithm Technical Reference
Time Series Model Query Examples
Mining Model Content for Time Series Models (Analysis Services - Data Mining)

Microsoft Time Series Algorithm Technical Reference

11/28/2018 • 14 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Time Series algorithm includes two separate algorithms for analyzing time series:

- The ARTXP algorithm, which was introduced in SQL Server 2005 (9.x), is optimized for predicting the next likely value in a series.
- The ARIMA algorithm was added in SQL Server 2008 to improve accuracy for long-term prediction.

By default, Analysis Services uses each algorithm separately to train the model and then blends the results to yield the best prediction for a variable number of predictions. You can also choose to use just one of the algorithms, based on your data and prediction requirements. In SQL Server 2008 Enterprise, you can also customize the cut-off point that controls the blend of algorithms during prediction.

This topic provides additional information about how each algorithm is implemented, and how you can customize the algorithm by setting parameters to fine-tune the analysis and prediction results.

Implementation of the Microsoft Time Series Algorithm

Microsoft Research developed the original ARTXP algorithm that was used in SQL Server 2005, basing the implementation on the Microsoft Decision Trees algorithm. Therefore, the ARTXP algorithm can be described as an autoregressive tree model for representing periodic time series data. This algorithm relates a variable number of past items to each current item that is being predicted. The name ARTXP derives from the fact that the autoregressive tree method (an ART algorithm) is applied to multiple unknown prior states. For a detailed explanation of the ARTXP algorithm, see Autoregressive Tree Models for Time-Series Analysis.

The ARIMA algorithm was added to the Microsoft Time Series algorithm in SQL Server 2008 to improve long-term prediction. It is an implementation of the process for computing autoregressive integrated moving averages that was described by Box and Jenkins. The ARIMA methodology makes it possible to determine dependencies in observations taken sequentially in time, and can incorporate random shocks as part of the model. The ARIMA method also supports multiplicative seasonality. Readers who want to learn more about the ARIMA algorithm are encouraged to read the seminal work by Box and Jenkins; this section is intended to provide specific details about how the ARIMA methodology has been implemented in the Microsoft Time Series algorithm.

By default, the Microsoft Time Series algorithm uses both methods, ARTXP and ARIMA, and blends the results to improve prediction accuracy. If you want to use only a specific method, you can set the algorithm parameters to use only ARTXP or only ARIMA, or to control how the results of the algorithms are combined. Note that the ARTXP algorithm supports cross-prediction, but the ARIMA algorithm does not. Therefore, cross-prediction is available only when you use a blend of algorithms, or when you configure the model to use only ARTXP.

Understanding ARIMA Difference Order

This section introduces some terminology needed to understand the ARIMA model, and discusses the specific implementation of *differencing* in the Microsoft Time Series algorithm. For a full explanation of these terms and concepts, we recommend a review of Box and Jenkins.

- A term is a component of a mathematical equation. For example, a term in a polynomial equation might include a combination of variables and constants.
- The ARIMA formula that is included in the Microsoft Time Series algorithm uses both *autoregressive* and *moving average* terms.
- Time series models can be *stationary* or *nonstationary*. *Stationary models* are those that revert to a mean, though they might have cycles, whereas *nonstationary* models do not have a focus of equilibrium and are subject to greater variance or change introduced by *shocks*, or external variables.
- The goal of *differencing* is to make a time series stabilize and become stationary.
- The *order of difference* represents the number of times that the difference between values is taken for a time series.

The Microsoft Time Series algorithm works by taking values in a data series and attempting to fit the data to a pattern. If the data series is are not already stationary, the algorithm applies an order of difference. Each increase in the order of difference tends to make the time series more stationary.

For example, if you have the time series (z1, z2, ..., zn) and perform calculations using one order of difference, you obtain a new series (y1, y2,...., yn-1), where `yi = zi+1-zi`. When the difference order is 2, the algorithm generates another series `(x1, x2, ..., xn-2)`, based on the y series that was derived from the first order equation. The correct amount of differencing depends on the data. A single order of differencing is most common in models that show a constant trend; a second order of differencing can indicate a trend that varies with time.

By default, the order of difference used in the Microsoft Time Series algorithm is -1, meaning that the algorithm will automatically detect the best value for the difference order. Typically, that best value is 1 (when differencing is required), but under certain circumstances the algorithm will increase that value to a maximum of 2.

The Microsoft Time Series algorithm determines the optimal ARIMA difference order by using the autoregression values. The algorithm examines the AR values and sets a hidden parameter, ARIMA_AR_ORDER, representing the order of the AR terms. This hidden parameter, ARIMA_AR_ORDER, has a range of values from -1 to 8. At the default value of -1, the algorithm will automatically select the appropriate difference order.

Whenever the value of ARIMA_AR_ORDER is greater than 1, the algorithm multiplies the time series by a polynomial term. If one term of the polynomial formula resolves to a root of 1 or close to 1, the algorithm attempts to preserve the stability of the model by removing the term and increasing the difference order by 1. If the difference order is already at the maximum, the term is removed and the difference order does not change.

For example, if the value of AR = 2, the resulting AR polynomial term might look like this: `1 - 1.4B + .45B^2 = (1- .9B) (1- 0.5B)`. Note the term `(1- .9B)` which has a root of about 0.9. The algorithm eliminates this term from the polynomial formula but cannot increase the difference order by one because it is already at the maximum value of 2.

It is important to note that the only way that you can **force** a change in difference order is to use the unsupported parameter, ARIMA_DIFFERENCE_ORDER. This hidden parameter controls how many times the algorithm performs differencing on the time series, and can be set by typing a custom algorithm parameter. However, we do not recommend that you change this value unless you are prepared to experiment and are familiar with the calculations involved. Also note that there is currently no mechanism, including hidden parameters, to let you control the threshold at which the increase in difference order is triggered.

Finally, note that the formula described above is the simplified case, with no seasonality hints. If seasonality hints are provided, then a separate AR polynomial term is added to the left of the equation for each seasonality hint, and the same strategy is applied to eliminate terms that might destabilize the differenced series.

Customizing the Microsoft Time Series Algorithm

The Microsoft Time Series algorithm supports the following parameters that affect the behavior, performance, and accuracy of the resulting mining model.

> **NOTE**
>
> The Microsoft Time Series algorithm is available in all editions of SQL Server; however, some advanced features, including parameters for customizing the time series analysis, are supported only in specific editions of SQL Server. For a list of features that are supported by the editions of SQL Server, see Features Supported by the Editions of SQL Server.

Detection of Seasonality

Both ARIMA and ARTXP algorithms support detection of seasonality or periodicity. Analysis Services uses Fast Fourier transformation to detect seasonality before training. However, you can affect seasonality detection, and the results of time series analysis, by setting algorithm parameters.

- By changing the value of *AUTODETECT_SEASONALITY*, you can influence the number of possible time segments that are generated.
- By setting a value or multiple values for *PERIODICITY_HINT*, you can provide the algorithm with information about expected cycles in the data and potentially increase the accuracy of detection.

> **NOTE**
>
> Both the ARTXP and ARIMA algorithms are very sensitive to seasonality hints. Therefore, providing the wrong hint can adversely affect results.

Choosing an Algorithm and Specifying the Blend of Algorithms

By default, or when you select the MIXED option, Analysis Services combines the algorithms and assigns them equal weight. However, in Enterprise Edition, you can specify a particular algorithm, or you can customize the proportion of each algorithm in the results by setting a parameter that weights the results towards either the short or the long-term prediction. By default, the *FORECAST_METHOD* parameter is set to MIXED, and Analysis Services uses both algorithms and then weights their values to maximize the strengths of each algorithm.

- To control the choice of algorithm, you set the *FORECAST_METHOD* parameter.
- If you want to use cross-prediction, you must use either the ARTXP or the MIXED option because ARIMA does not support cross-prediction.
- Set the *FORECAST_METHOD* to ARTXP if you want to favor short-term prediction.
- Set the *FORECAST_METHOD* to ARIMA if you want to improve long-term prediction.

In Enterprise Edition, you can also customize how Analysis Services mixes the combination of the ARIMA and ARTXP algorithms. You can control both the starting point for the mixture, and the rate of change by setting the *PREDICTION_SMOOTHING* parameter:

- If you set *PREDICTION_SMOOTHING* to 0, the model uses ARTXP only.
- If you set *PREDICTION_SMOOTHING* to 1, the model uses ARIMA only.
- If you set *PREDICTION_SMOOTHING* to a value between 0 and 1, the model weights the ARTXP algorithm as an exponentially decreasing function of the prediction steps. At the same time, the model also weights the ARIMA algorithm as the 1-complement of the ARTXP weight. The model uses normalization and a stabilization constant to smooth the curves.

In general, if you predict up to 5 time slices, ARTXP is almost always the better choice. However, as you

increase the number of time slices to predict, ARIMA typically performs better.

The following diagram illustrates how the model blends the algorithms when *PREDICTION_SMOOTHING* is set to the default value, 0.5. ARIMA and ARTXP are weighted equally at first, but as the number of prediction steps increases, ARIMA is weighed more heavily.

In contrast, the following diagram illustrates the blending of the algorithms when *PREDICTION_SMOOTHING* is set to 0.2. For step 0, the model weights ARIMA as 0.2 and ARTXP as 0.8. Thereafter, the weight of ARIMA exponentially increases and the weight of ARTXP exponentially decreases.

Setting Algorithm Parameters

The following table describes the parameters that can be used with the Microsoft Time Series algorithm.

PARAMETER	DESCRIPTION
AUTO_DETECT_PERIODICITY	Specifies a numeric value between 0 and 1 that detects periodicity. The default is 0.6. If the value is closer to 0, periodicity is detected only for strongly periodic data. Setting this value closer to 1 favors the discovery of many patterns that are almost periodic and the automatic generation of periodicity hints. Note: Dealing with many periodicity hints will likely lead to significantly longer model training times, but more accurate models.

PARAMETER	DESCRIPTION
COMPLEXITY_PENALTY	Controls the growth of the decision tree. The default is 0.1. Decreasing this value increases the chance of a split. Increasing this value decreases the chance of a split. Note: This parameter is only available in some editions of SQL Server.
FORECAST_METHOD	Specifies which algorithm to use for analysis and prediction. Possible values are ARTXP, ARIMA, or MIXED. The default is MIXED.
HISTORIC_MODEL_COUNT	Specifies the number of historic models that will be built. The default is 1. Note: This parameter is only available in some editions of SQL Server.
HISTORICAL_MODEL_GAP	Specifies the time lag between two consecutive historic models. The default is 10. The value represents a number of time units, where the unit is defined by the model. For example, setting this value to g causes historic models to be built for data that is truncated by time slices at intervals of g, 2*g, 3*g, and so on. Note: This parameter is only available in some editions of SQL Server.
INSTABILITY_SENSITIVITY	Controls the point at which prediction variance exceeds a certain threshold, after which the ARTXP algorithm suppresses predictions. The default value is 1. Note: This parameter does not apply to models that use ARIMA only. The default value of 1 provides the same behavior as in SQL Server 2005 (9.x). Analysis Services monitors the normalized standard deviation for each prediction. As soon as this value exceeds the threshold for any prediction, the time series algorithm returns a NULL and stops the prediction process. A value of 0 stops instability detection. This means that you can create an infinite number of predictions, regardless of the variance. Note: This parameter can only be modified in SQL Server Enterprise. In SQL Server Standard, Analysis Services uses only the default value of 1.
MAXIMUM_SERIES_VALUE	Specifies the maximum value to use for predictions. This parameter is used, together with *MINIMUM_SERIES_VALUE*, to constrain the predictions to some expected range. For example, you can specify that the predicted sales quantity for any day should never exceed the number of products in inventory. Note: This parameter is only available in some editions of SQL Server.

PARAMETER	DESCRIPTION
MINIMUM_SERIES_VALUE	Specifies the minimum value that can be predicted. This parameter is used, together with *MAXIMUM_SERIES_VALUE*, to constrain the predictions to some expected range. For example, you can specify that the predicted sales quantity should never be a negative number. Note: This parameter is only available in some editions of SQL Server.
MINIMUM_SUPPORT	Specifies the minimum number of time slices that are required to generate a split in each time series tree. The default is 10.
MISSING_VALUE_SUBSTITUTION	Specifies how gaps in historical data are filled. By default, gaps in data are not allowed. The following table lists the possible values for this parameter: **Previous**: Repeats the value from the previous time slice. **Mean**: Uses a moving average of time slices used in training. Numeric constant: Uses the specified number to replace all missing values. **None**: Default. Replaces missing values with values plotted along the curve of the trained model. Note that if your data contains multiple series, the series cannot have ragged edges. That is, all series should have the same start and end points. Analysis Services also uses the value of this parameter to fill gaps in new data when you perform a **PREDICTION JOIN** on time series model.
PERIODICITY_HINT	Provides a hint to the algorithm as to the periodicity of the data. For example, if sales vary by year, and the unit of measurement in the series is months, the periodicity is 12. This parameter takes the format of {n [, n]}, where n is any positive number. The n in the brackets [] is optional and can be repeated as frequently as needed. For example, to provide multiple periodicity hints for data supplied monthly, you might enter {12, 3, 1} to detect patterns for the year, quarter, and month. However, periodicity has a strong effect on model quality. If the hint that you give differs from the actual periodicity, your results can be adversely affected. The default is {1}. Note that the braces are required. Also, this parameter has a string data type. Therefore, if you type this parameter as part of a Data Mining Extensions (DMX) statement, you must enclose the number and braces in quotation marks.

PARAMETER	DESCRIPTION
PREDICTION_SMOOTHING	Specifies how the model should be mixed to optimize forecasting. You can type any value between 0 and 1 or use one of the following values: 0: Specifies that prediction uses ARTXP only. Forecasting is optimized for fewer predictions. 1: Specifies that prediction uses ARIMA only. Forecasting is optimized for many predictions. 0.5: Default. Specifies that for prediction both algorithms should be used and the results blended. When doing prediction smoothing, use the *FORECAST_METHOD* parameter to control training. Note that this parameter is only available in some editions of SQL Server.

Modeling Flags

The Microsoft Time Series algorithm supports the following modeling flags. When you create the mining structure or mining model, you define modeling flags to specify how values in each column are handled during analysis. For more information, see Modeling Flags (Data Mining).

MODELING FLAG	DESCRIPTION
NOT NULL	Indicates that the column cannot contain a null. An error will result if Analysis Services encounters a null during model training. Applies to mining structure columns.
MODEL_EXISTENCE_ONLY	Means that the column will be treated as having two possible states: Missing and Existing. A null is a missing value. Applies to mining model columns.

Requirements

A time series model must contain a key time column that contains unique values, input columns, and at least one predictable column.

Input and Predictable Columns

The Microsoft Time Series algorithm supports the specific input column content types, predictable column content types, and modeling flags that are listed in the following table.

COLUMN	CONTENT TYPES
Input attribute	Continuous ,Key, Key Time, and Table
Predictable attribute	Continuous, Table

> **NOTE**
> Cyclical and Ordered content types are supported, but the algorithm treats them as discrete values and does not perform special processing.

See Also

Microsoft Time Series Algorithm
Time Series Model Query Examples
Mining Model Content for Time Series Models (Analysis Services - Data Mining)

Mining Model Content for Time Series Models (Analysis Services - Data Mining)

11/28/2018 • 25 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

All mining models use the same structure to store their content. This structure is defined according to the data mining content schema rowset. However, within that standard structure, the nodes that contain information are arranged in different ways to represent various kinds of trees. This topic describes how the nodes are organized, and what each node means, for mining models that are based on the Microsoft Time Series algorithm.

For an explanation of general mining model content that applies to all model types, see Mining Model Content (Analysis Services - Data Mining).

When reviewing this topic, you might find it useful follow along by browsing the contents of a time series model. You can create a time series model by completing the Basic Data Mining tutorial. The model you create in the tutorial is a mixed model that trains data by using both the ARIMA and ARTXP algorithms. For information about how to view the contents of a mining model, see Data Mining Model Viewers.

Understanding the Structure of a Time Series Model

A time series model has a single parent node that represents the model and its metadata. Underneath that parent node, there are one or two time series trees, depending on the algorithm that you used to create the model.

If you create a mixed model, two separate trees are added to the model, one for ARIMA and one for ARTXP. If you choose to use only the ARTXP algorithm or only the ARIMA algorithm, you will have a single tree that corresponds to that algorithm. You specify which algorithm to use by setting the FORECAST_METHOD parameter. For more information about whether to use ARTXP, ARIMA, or a mixed model, see Microsoft Time Series Algorithm.

The following diagram shows an example of a time series data mining model that was created with the default settings, to create a mixed model. So that you can more easily compare the differences between the two models, here the ARTXP model is shown on the left side of the diagram and the ARIMA model is shown in the right side of the diagram. Whereas ARTXP is a tree-like structure that splits into smaller and smaller branches, the structure created by the ARIMA algorithm is more like a pyramid built upwards from smaller components.

The important point to remember is that information is arranged within the ARIMA and ARTXP trees in completely different ways, and you should consider the two trees as related only at the root node. Although the two representations are presented in one model for convenience, they should be treated as two independent models. ARTXP represents an actual tree structure, but ARIMA does not.

When you use the Microsoft Generic Model Content Tree Viewer to view a model that uses both ARIMA and ARTXP, the nodes for the ARTXP and ARIMA models are all presented as child nodes of the parent time series model. However, you can easily tell them apart by the labels applied to the nodes.

- The first set of nodes is labeled (All), and represents the results of analysis by the ARTXP algorithm.
- The second set of nodes is labeled ARIMA, and represents the results of analysis by the ARIMA algorithm.

> **WARNING**
>
> The name (All) on the ARTXP tree is retained only for backward compatibility. Prior to SQL Server 2008, the Time Series algorithm used a single algorithm for analysis, the ARTXP algorithm.

The following sections explain how the nodes are arranged within each of these model types.

Structure of an ARTXP Model

The ARTXP algorithm creates a model similar to a decision trees model. It groups predictable attributes and splits them whenever significant differences are found. Therefore, each ARTXP model contains a separate branch for each predictable attribute. For example, the Basic Data Mining tutorial creates a model that predicts the amount of sales for several regions. In this case, **[Amount]** is the predictable attribute and a separate branch is created for each region. If you had two predictable attributes, **[Amount]** and **[Quantity]**, a separate branch would be created for each combination of an attribute and a region.

The top node for the ARTXP branch contains the same information that is in a decision tree root node. This includes the number of children for that node (CHILDREN_CARDINALITY), the number of cases that meet the conditions of this node (NODE_SUPPORT), and a variety of descriptive statistics (NODE_DISTRIBUTION).

If the node does not have any children, this means that no significant conditions were found that would justify dividing the cases into further subgroups. The branch ends at this point and the node is termed a *leaf node*. The leaf node contains the attributes, coefficients, and values that are the building blocks of the ARTXP formula.

Some branches may have additional splits, similar to a decision trees model. For example, the branch of the tree that represents sales for the Europe region splits into two branches. A split occurs when a condition is found that causes a significant difference between the two groups. The parent node tells you the name of the attribute that

caused the split, such as [Amount], and how many cases there are in the parent node. The leaf nodes provide more detail: the value of the attribute, such as [Sales] >10,000 vs. [Sales] < 10,000), the number of cases that support each condition, and the ARTXP formula.

> **NOTE**
> If you want to view the formulas, you can find the complete regression formula at the leaf node level, but not in an intermediate or root node.

Structure of an ARIMA Model

The ARIMA algorithm creates a single piece of information for each combination of a data series (such as **[Region]**) and a predictable attribute (such as **[Sales Amount]**)-the equation that describes the change of the predictable attribute over time.

The equation for each series is derived from multiple components, one for each periodic structure that was found in the data. For example, if you have sales data that is collected on a monthly basis, the algorithm might detect monthly, quarterly, or yearly periodic structures.

The algorithm outputs a separate set of parent and child nodes for each periodicity it finds. The default periodicity is 1, for a single time slice, and is automatically added into all models. You can specify possible periodic structures by entering multiple values in the PERIODICITY_HINT parameter. However, if the algorithm does not detect a periodic structure, it will not output results for that hint.

Each periodic structure that is output in the model content contains the following component nodes:

- A node for the *autoregressive order* (AR)
- A node for the *moving average* (MA)

For information about the meaning of these terms, see Microsoft Time Series Algorithm.

The *difference order* is an important part of the formula, and is represented in the equation. For more information about how the difference order is used, see Microsoft Time Series Algorithm Technical Reference.

Model Content for Time Series

This section provides detail and examples only for those columns in the mining model content that have particular relevance for time series models.

For information about general-purpose columns in the schema rowset, such as MODEL_CATALOG and MODEL_NAME, or for explanations of mining model terminology, see Mining Model Content (Analysis Services - Data Mining).

MODEL_CATALOG
Name of the database where the model is stored.

MODEL_NAME
Name of the model.

ATTRIBUTE_NAME
The predictable attribute for the data series represented in the node. (The same value as for MSOLAP_MODEL_COLUMN.)

NODE_NAME
The name of the node.

Currently, this column contains the same value as NODE_UNIQUE_NAME, although this might change in future releases.

NODE_UNIQUE_NAME

The unique name of the node. The model parent node is always named **TS**.

ARTXP: Each node is represented by TS followed by a hexadecimal numeric value. The order of the nodes is unimportant.

For example, the ARTXP nodes directly under the TS tree might be numbered TS00000001-TS0000000b.

ARIMA: Each node in an ARIMA tree is represented by TA followed by a hexadecimal numeric value. The child nodes contain the unique name of the parent node followed by another hexadecimal number indicating the sequence within the node.

All ARIMA trees are structured exactly the same. Each root contains the nodes and naming convention exemplified in the following table:

ARIMA NODE ID AND TYPE	EXAMPLE OF NODE NAME
ARIMA Root (27)	TA0000000b
ARIMA Periodic Structure (28)	TA0000000b00000000
ARIMA Auto Regressive (29)	TA0000000b000000000
ARIMA Moving Average (30)	TA0000000b000000001

NODE_TYPE

A time series model outputs the following node types, depending on the algorithm.

ARTXP:

NODE TYPE ID	DESCRIPTION
1 (Model)	Time series
3 (Interior)	Represents an interior branch within an ARTXP time series tree.
16 (Time series tree)	Root of ARTXP tree that corresponds to a predictable attribute and series.
15 (Time series)	Leaf node in ARTXP tree.

ARIMA:

NODE TYPE ID	DESCRIPTION
27 (ARIMA Root)	The top node of an ARIMA tree.
28 (ARIMA Periodic Structure)	Component of an ARIMA tree that describes a single periodic structure.
29 (ARIMA Autoregressive)	Contains a coefficient for a single periodic structure.
30 (ARIMA Moving Average)	Contains a coefficient for a single periodic structure.

NODE_CAPTION

A label or caption that is associated with the node.

This property is primarily for display purposes.

ARTXP: Contains the split condition for the node, displayed as a combination of attribute and value range.

ARIMA: Contains the short form of the ARIMA equation.

For information about the format of the ARIMA equation, see Mining Legend for ARIMA.

CHILDREN_CARDINALITY
The number of direct children that the node has.

PARENT_UNIQUE_NAME
The unique name of the node's parent. NULL is returned for any nodes at the root level.

NODE_DESCRIPTION
A description in text of the rules, splits, or formulas in the current node.

ARTXP: For more information, see Understanding the ARTXP Tree.

ARIMA: For more information, see Understanding the ARIMA Tree.

NODE_RULE
An XML description of the rules, splits, or formulas in the current node.

ARTXP: The NODE_RULE generally corresponds to the NODE_CAPTION.

ARIMA: For more information, see Understanding the ARIMA Tree.

MARGINAL_RULE
An XML description of the split or content that is specific to that node.

ARTXP: The MARGINAL_RULE generally corresponds to the NODE_DESCRIPTION.

ARIMA: Always blank; use NODE_RULE instead.

NODE_PROBABILITY
ARTXP: For tree nodes, always 1. For leaf nodes, the probability of reaching the node from the model root node.

ARIMA: Always 0.

MARGINAL_PROBABILITY
ARTXP: For tree nodes, always 1. For leaf nodes, the probability of reaching the node from the immediate parent node.

ARIMA: Always 0.

NODE_DISTRIBUTION
A table that contains the probability histogram of the node. In a time series model, this nested table contains all the components required to assemble the actual regression formula.

For more information about the node distribution table in an ARTXP tree, see Understanding the ARTXP Tree.

For more information about the node distribution table in an ARIMA tree, see Understanding the ARIMA Tree.

If you wish to see all the constants and other components composed into a readable format, use the Time Series Viewer, click the node, and open the **Mining Legend**.

NODE_SUPPORT
The number of cases that support this node.

ARTXP: For the **(All)** node, indicates the total number of time slices included in the branch.

For terminal nodes, indicates the number of time slices that are included in the range that is described by NODE_CAPTION. The number of time slices in the terminal nodes always sums to the NODE_SUPPORT value of the branch **(All)** node.

ARIMA: A count of cases that support the current periodic structure. The value for support is repeated in all nodes of the current periodic structure.

MSOLAP_MODEL_COLUMN
The predictable attribute for the data series represented in the node. (The same value as for ATTRIBUTE_NAME.)

MSOLAP_NODE_SCORE
A numeric value that characterizes the information value of the tree or split.

ARTXP: Value is always 0.0 for nodes without a split. For nodes with a split, the value represents the interestingness score of the split.

For more information about scoring methods, see Feature Selection (Data Mining).

ARIMA: The Bayesian Information Criterion (BIC) score of the ARIMA model. The same score is set on all the ARIMA nodes related to the equation.

MSOLAP_NODE_SHORT_CAPTION
ARTXP: Same information as the NODE_DESCRIPTION.

ARIMA: Same information as the NODE_CAPTION: that is, the short form of the ARIMA equation.

Understanding the ARTXP Tree

The ARTXP model clearly separates the areas of the data that are linear from the areas of the data that split on some other factor. Wherever the changes in the predictable attribute can be directly represented as a function of the independent variables, a regression formula is calculated to represent that relationship

For example, if there is a direct correlation between time and sales for most of the data series, each series would be contained within a time series tree (NODE_TYPE =16) that has no child nodes for each data series, only a regression equation. However, if the relationship is not linear, an ARTXP time series tree can split on conditions into child nodes, just like a decision tree model. By viewing the model content in the **Microsoft Generic Content Tree Viewer** you can see where the splits occur, and how it affects the trend line.

To better understand this behavior, you can review the time series model created in the Basic Data Mining Tutorial. This model, based on the AdventureWorks data warehouse, does not use particularly complex data. Therefore, there are not many splits in the ARTXP tree. However, even this relatively simple model illustrates three different kinds of splits:

- The [Amount] trend line for the Pacific region splits on the time key. A split on the time key means that there is a change in the trend at a certain point in time. The trend line was linear only up to a certain point, and then the curve assumed a different shape. For example, one time series might continue until August 6, 2002, and another time series start after that date.
- The [Amount] trend line for the North America region splits on another variable. In this case, the trend for North America splits based on the value for the same model in the Europe region. In other words, the algorithm detected that when the value for Europe changes, the value for North America A also changes.
- The trend line for Europe region splits on itself.

What does each split mean? Interpreting the information conveyed by the model content is an art that requires a deep understanding of the data and its meaning in the business context.

- The apparent link between the trends for the North America and Europe regions may signify only that the data series for Europe has more entropy, which causes the trend for the North America to appear weaker.

Or, there might be no significant difference in the scoring for the two, and the correlation could be accidental, based simply on computing Europe before computing North America. However, you might want to review the data and make sure whether the correlation is false, or investigate to see if some other factor might involved.

- The split on the time key means that there is a statistically significant change in the gradient of the line. This might have been caused by mathematical factors such as the support for each range, or the calculations of entropy required for the split. Thus, this split might not be interesting in terms of the model's meaning in the real world. However, when you review the time period indicated in the split, you might find interesting correlations that are not represented in the data, such a sales promotion or other event that began at that time and that may have affected the data.

If the data contained other attributes, you would very likely see more interesting examples of branching in the tree. For example, if you tracked weather information and used that as an attribute for analysis, you might see multiple splits in the tree that represent the complex interaction of sales and weather.

In short, data mining is useful for providing hints about where potentially interesting phenomena occur, but further investigation and the expertise of the business users is necessary to accurately interpret the worth of the information in context.

Elements of the ARTXP Time Series Formula

To view the complete formula for an ARTXP tree or branch, we recommend that you use the **Mining Legend** of the Microsoft Time Series Viewer, which presents all of the constants in a readable format.

- View the Formula for a Time Series Model (Data Mining)

The following section presents a sample equation and explains the basic terms.

Mining Legend for an ARTXP Formula

The following example shows the ARTXP formula for one part of the model, as displayed in the **Mining Legend**. To view this formula, open the [Forecasting] model that you created in the Basic Data Mining Tutorial in the Microsoft Time Series viewer, click the **Model** tab, and select the tree for the R250: Europe data series.

To view the equation used for this example, click the node that represents the date series on or after 7/5/2003.

Example of tree node equation:

```
Quantity = 21.322 -0.293 * Quantity(R250 North America,-7) + 0.069 * Quantity(R250 Europe,-1) + 0.023 *
Quantity(R250 Europe,-3) -0.142 * Quantity(R750 Europe,-8)
```

In this case, the value 21.322 represents the value that is predicted for Quantity as a function of the following elements of the equation.

For example, one element is `Quantity(R250 North America,-7)`. This notation means the quantity for the North America region at `t-7`, or seven time slices before the current time slice. The value for this data series is multiplied by the coefficient -0.293. The coefficient for each element is derived during the training process and is based on trends in the data.

There are multiple elements in this equation because the model has calculated that the quantity of the R250 model in the Europe region is dependent on the values of several other data series.

Model Content for an ARTXP Formula

The following table shows the same information for the formula, using the contents of the relevant node as displayed in the Microsoft Generic Content Tree Viewer (Data Mining).

ATTRIBUTE_NAME	ATTRIBUTE_VALUE	SUPPORT	PROBABILITY	VARIANCE	VALUETYPE
Quantity(R250 Europe,y-intercept)	21.3223433563772	11	0	1.65508795539661	11 (Intercept)
Quantity(R250 Europe,-1)	0.0691694140876526	0	0	0	7 (Coefficient)
Quantity(R250 Europe,-1)	20.6363635858123	0	0	182.380682874818	9 (Statistics)
Quantity(R750 Europe,-8)	-0.1421203048299	0	0	0	7 (Coefficient)
Quantity(R750 Europe,-8)	22.5454545333019	0	0	104.362130048408	9 (Statistics)
Quantity(R250 Europe,-3)	0.0234095979448281	0	0	0	7 (Coefficient)
Quantity(R250 Europe,-3)	24.8181818883176	0	0	176.475304989169	9 (Statistics)
Quantity(R250 North America,-7)	-0.292914186039869	0	0	0	7 (Coefficient)
Quantity(R250 North America,-7)	10.36363640433	0	0	701.882534898676	9 (Statistics)

As you can see from comparing these examples, the mining model content contains the same information that is available in the **Mining Legend**, but with additional columns for *variance* and *support*. The value for support indicates the count of cases that support the trend described by this equation.

Using the ARTXP Time Series Formula

For most business users, the value of the ARTXP model content is that it combines both a tree view and a linear representation of the data.

- If the changes in the predictable attribute can be represented as a linear function of the independent variables, the algorithm will automatically compute the regression equation and output that series in a separate node
- Whenever the relationship cannot be expressed as a linear correlation, the time series branches like a decision tree.

By browsing the model content in the Microsoft Time Series Viewer you can see where the split occurs, and how it affects the trend line.

If a direct correlation exists between time and sales for any part of the data series, the easiest way to get the formula is to copy the formula from the **Mining Legend**, and then paste it into a document or presentation to help explain the model. Alternatively, you could extract the mean, coefficient, and other information from the NODE_DISTRIBUTION table for that tree and use it to compute extensions of the trend. If the entire series exhibits a consistent linear relationship, the equation is contained in the (All) node. If there is any branching in the tree, the equation is contained in the leaf node.

The following query returns all the ARTXP leaf nodes from a mining model, together with the nested table, NODE_DISTRIBUTION, which contains the equation.

```
SELECT MODEL_NAME, ATTRIBUTE_NAME, NODE_NAME,
NODE_CAPTION,
(SELECT ATTRIBUTE_NAME, ATTRIBUTE_VALUE, [VARIANCE], VALUETYPE
FROM NODE_DISTRIBUTION) as t
FROM Forecasting.CONTENT
WHERE NODE_TYPE = 15
```

Understanding the ARIMA Tree

Each structure in an ARIMA model corresponds to a *periodicity* or *periodic structure*. A periodic structure is a pattern of data that repeats throughout the data series. Some minor variation in the pattern is allowed, within statistical limits. Periodicity is measured according to the default time units that were used in the training data. For example, if the training data provides sales data for each day, the default time unit is one day, and all periodic structures are defined as a specified number of days.

Each period that is detected by the algorithm gets its own structure node. For example, if you are analyzing daily sales data, the model might detect periodic structures that represent weeks. In this case, the algorithm will create two periodic structures in the finished model: one for the default daily period, denoted as {1}, and one for weeks, indicated by {7}.

For example, the following query returns all the ARIMA structures from a mining model.

```
SELECT MODEL_NAME, ATTRIBUTE_NAME, NODE_NAME, NODE_CAPTION
FROM Forecasting.CONTENT
WHERE NODE_TYPE = 27
```

Example results:

MODEL_NAME	ATTRIBUTE_NAME	NODE_NAME	NODE_TYPE	NODE_CAPTION
Forecasting	M200 Europe:Quantity	TA00000000	27	ARIMA (1,0,1)
Forecasting	M200 North America:Quantity	TA00000001	27	ARIMA (1,0,4) X (1,1,4)(6)
Forecasting	M200 Pacific:Quantity	TA00000002	27	ARIMA (2,0,8) X (1,0,0)(4)
Forecasting	M200 Pacific:Quantity	TA00000002	27	ARIMA (2,0,8) X (1,0,0)(4)
Forecasting	R250 Europe:Quantity	TA00000003	27	ARIMA (1,0,7)
Forecasting	R250 North America:Quantity	TA00000004	27	ARIMA (1,0,2)
Forecasting	R250 Pacific:Quantity	TA00000005	27	ARIMA (2,0,2) X (1,1,2)(12)

MODEL_NAME	ATTRIBUTE_NAME	NODE_NAME	NODE_TYPE	NODE_CAPTION
Forecasting	R750 Europe:Quantity	TA00000006	27	ARIMA (2,1,1) X (1,1,5)(6)
Forecasting	T1000 Europe:Quantity	TA00000009	27	ARIMA (1,0,1)
Forecasting	T1000 North America:Quantity	TA0000000a	27	ARIMA (1,1,1)
Forecasting	T1`000 Pacific:Quantity	TA0000000b	27	ARIMA (1,0,3)

From these results, which you can also browse by using the Microsoft Generic Content Tree Viewer (Data Mining), you can tell at a glance which series are completely linear, which have multiple periodic structures, and what the discovered periodicities are.

For example, the short form of the ARIMA Equation for the M200 Europe series tells you that only the default, or daily, cycle was detected. The short form of the equation is provided in the NODE_CAPTION column.

However, for the M200 North America series, an additional periodic structure was found. The node TA00000001 has two child nodes, one with the equation, (1,0,4), and one with the equation, (1,1,4)(6). These equations are concatenated and presented in the parent node.

For each periodic structure, the model content also provides the *order* and the *moving average* as child nodes. For example, the following query retrieves the child nodes of one of the nodes listed in the previous example. Notice that the column, PARENT_UNIQUE_NAME, must be enclosed in brackets to distinguish it from the reserved keyword of the same name.

```
SELECT *
FROM Forecasting.CONTENT
WHERE [PARENT_UNIQUE_NAME] = ' TA00000001'
```

Because this is an ARIMA tree, not an ARTXP tree, you cannot use the IsDescendant (DMX) function to return the child nodes of this periodic structure. Instead, you can use the attribute and node types to filter the results and return the child nodes that provide more detail about how the equation was built, including the moving averages and difference order.

```
SELECT MODEL_NAME, ATTRIBUTE_NAME, NODE_UNIQUE_NAME,
NODE_TYPE,  NODE_CAPTION
FROM Forecasting.CONTENT
WHERE [MSOLAP_MODEL_COLUMN] ='M200 North America:Quantity'
AND (NODE_TYPE = 29 or NODE_TYPE = 30)
```

Example results:

MODEL_NAME	ATTRIBUTE_NAME	NODE_UNIQUE_NAME	NODE_TYPE	NODE_CAPTION
Forecasting	M200 North America:Quantity	TA00000001000000010	29	ARIMA {1,0.961832044807041}

MODEL_NAME	ATTRIBUTE_NAME	NODE_UNIQUE_NAME	NODE_TYPE	NODE_CAPTION
Forecasting	M200 North America:Quantity	TA00000001000000011	30	ARIMA {1,-3.51073103693271E-02,2.15731642954099,-0.220314343327742,-1.33151478258758}
Forecasting	M200 North America:Quantity	TA00000001000000000	29	ARIMA {1,0.643565911081657}
Forecasting	M200 North America:Quantity	TA00000001000000001	30	ARIMA {1,1.45035399809581E-02,-4.40489283927752E-02,-0.19203901352577,0.242202497643993}

These examples illustrate that the further you drill down into the ARIMA tree, the more detail is revealed, but the important information is combined and presented in the parent node as well.

Time Series Formula for ARIMA

To view the complete formula for any ARIMA node, we recommend that you use the **Mining Legend** of the Microsoft Time Series Viewer, which presents the autoregressive order, moving averages, and other elements of the equation already composed in a consistent format.

- View the Formula for a Time Series Model (Data Mining)

This section presents a sample equation and explains the basic terms.

Mining Legend for ARIMA Formula

The following example shows the ARIMA formula for one part of the model, as displayed in the Mining Legend. To view this formula, open the **Forecasting** model by using the **Microsoft Time Series viewer**, click the **Model** tab, select the tree for the **R250: Europe** data series, and then click the node that represents the date series on or after 7/5/2003. The mining legend composes all of the constants in a readable format, shown in this example:

ARIMA equation:

```
ARIMA ({1,1},0,{1,1.49791920964142,1.10640053499397,0.888873034670339,-5.05429403071953E-02,-
0.905265316720334,-0.961908900643379,-0.649991020901922}) Intercept:56.8888888888889
```

This equation is the long ARIMA format, which includes the values of the coefficients and the intercept. The short format for this equation would be {1,0,7}, where 1 indicates the period as a count of time slices, 0 indicates the term difference order, and 7 indicates the number of coefficients.

> **NOTE**
>
> A constant is calculated by Analysis Services for computing variance, but the constant itself is not displayed anywhere in the user interface. However, you can view the variance for any point in the series as a function of this constant if you select **Show Deviations,** in **Chart** view. The Tooltip for each data series shows the variance for a specific predicted point.

Model Content for ARIMA Formula

An ARIMA model follows a standard structure, with different information contained in nodes of different types. To view the model content for the ARIMA model, change the viewer to the **Microsoft Generic Content Tree**

Viewer, and then expand the node that has the attribute name, **R250 Europe: Quantity**.

An ARIMA model for a data series contains the basic periodic equation in four different formats, which you can choose from depending on the application.

NODE_CAPTION: Displays the short format of the equation. The short format tells you how many periodic structures are represented, and how many coefficients they have. For example, if the short format of the equation is `{4,0,6}`, the node represents one periodic structure with 6 coefficients. If the short format is something like `{2,0,8} x {1,0,0}(4)`, the node contains two periodic structures.

NODE DESCRIPTION: Displays the long format of the equation, which is also the form of the equation that appears in the **Mining Legend**. The long form of the equation is similar to the short form, except that the actual values of the coefficients are displayed instead of being counted.

NODE_RULE: Displays an XML representation of the equation. Depending on the node type, the XML representation can include single or multiple periodic structures. The following table illustrates how XML nodes are rolled up to higher levels of the ARIMA model.

NODE TYPE	XML CONTENT
27 (ARIMA Root)	Includes all periodic structures for the data series, and the content of all child nodes for each periodic structure.
28 (ARIMA Periodic Structure)	Defines a single periodic structure, including its autoregressive term node and its moving average coefficients.
29 (ARIMA Autoregressive)	Lists the terms for a single periodic structure.
30 (ARIMA Moving Average)	Lists the coefficients for a single periodic structure.

NODE_DISTRIBUTION: Displays terms of the equation in a nested table, which you can query to obtain specific terms. The node distribution table follows the same hierarchical structure as the XML rules. That is, the root node of the ARIMA series (NODE_TYPE = 27) contains the intercept value and the periodicities for the complete equation, which can include multiple periodicities, whereas the child nodes contain only information specific to a certain periodic structure or to the child nodes of that periodic structure.

NODE TYPE	ATTRIBUTE	VALUE TYPE
27 (ARIMA Root)	Intercept Periodicity	11
28 (ARIMA Periodic Structure)	Periodicity Auto Regressive order Difference order Moving average order	12 13 15 14
29 (ARIMA Autoregressive)	Coefficient (complement of coefficient)	7

NODE TYPE	ATTRIBUTE	VALUE TYPE
30 (ARIMA Moving Average)	Value at t Value at t-1 ... Value at t-n	7

The value for the *moving average order* indicates the number of moving averages in a series. Generally the moving average is calculated `n-1` times if there are `n` terms in a series, but the number can be reduced for easier computation.

The value for *autoregressive order* indicates the number of autoregressive series.

The value for *difference order* indicates how many times the series are compared, or differenced.

For an enumeration of the possible value types, see Microsoft.AnalysisServices.AdomdServer.MiningValueType.

Using the ARIMA Tree Information

If you use predictions that are based on the ARIMA algorithm in a business solution, you might want to paste the equation into a report to demonstrate the method that was used to create the prediction. You can use the caption to present the formulas in short format, or the description to present the formulas in long format.

If you are developing an application that uses time series predictions, you might find it useful to obtain the ARIMA equation from the model content and then make your own predictions. To obtain the ARIMA equation for any particular output, you can query the ARIMA root for that particular attribute directly, as shown in the previous examples.

If you know the ID of the node that contains the series you want, you have two options to retrieve the components of the equation:

- Nested table format: Use a DMX query or query via OLEDB client.
- XML representation: Use an XML query.

Remarks

It can be difficult to retrieve information from an ARTXP tree, because information for each split is in a different place within the tree. Therefore, with an ARTXP model, you must get all the pieces and then do some processing to reconstitute the complete formula. Retrieving an equation from an ARIMA model is easier because the formula has been made available throughout the tree. For information about how to create a query to retrieve this information, see Time Series Model Query Examples.

See Also

Mining Model Content (Analysis Services - Data Mining)
Microsoft Time Series Algorithm
Time Series Model Query Examples
Microsoft Time Series Algorithm Technical Reference

Time Series Model Query Examples

11/28/2018 • 12 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a query against a data mining model, you can create either a content query, which provides details about the patterns discovered in analysis, or you can create a prediction query, which uses the patterns in the model to make predictions for new data. For example, a content query for a time series model might provide additional details about the periodic structures that were detected, while a prediction query might give you predictions for the next 5-10 time slices. You can also retrieve metadata about the model by using a query.

This section explains how to create both kinds of queries for models that are based on the Microsoft Time Series algorithm.

Content Queries

Retrieving Periodicity Hints for the Model

Retrieving the Equation for an ARIMA Model

Retrieving the Equation for an ARTxp Model

Prediction Queries

Understanding When to Replace and When to Extend Time Series Data

Making Predictions with EXTEND_MODEL_CASES

Making Predictions with REPLACE_MODEL_CASES

Missing Value Substitution in Time Series Models

Getting Information about a Time Series Model

A model content query can provide basic information about the model, such as the parameters that were used when the model was created, the time the model was last processed. The following example illustrates the basic syntax for querying the model content by using the data mining schema rowsets.

Sample Query 1: Retrieving Periodicity Hints for the Model

You can retrieve the periodicities that were found within the time series by querying the ARIMA tree or the ARTXP tree. However, the periodicities in the completed model might not be the same as the periods that you specified as hints when you created the model. To retrieve the hints that were supplied as parameters when you created the model, you can query the mining model content schema rowset by using the following DMX statement:

```
SELECT MINING_PARAMETERS
FROM $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = '<model name>'
```

Partial results:

MINING_PARAMETERS
COMPLEXITY_PENALTY=0.1,MINIMUM_SUPPORT=10,PERIODICITY_HINT={1,3},....

The default periodicity hint is {1} and appears in all models; this sample model was created with an additional hint that might not be present in the final model.

> **NOTE**
> The results have been truncated here for readability.

Sample Query 2: Retrieving the Equation for an ARIMA Model

You can retrieve the equation for an ARIMA model by querying any node in an individual tree. Remember that each tree within an ARIMA model represents a different periodicity, and if there are multiple data series, each data series will have its own set of periodicity trees. Therefore, to retrieve the equation for a specific data series you must identify the tree first.

For example, the TA prefix tells you that the node is part of an ARIMA tree, whereas the TS prefix is used for ARTXP trees. You can find all ARIMA root trees by querying the model content for nodes with a NODE_TYPE value of 27. You can also use the value of ATTRIBUTE_NAME to find the ARIMA root node for a particular data series. This query example finds the ARIMA nodes that represent quantities sold of the R250 model in the Europe region.

```
SELECT NODE_UNIQUE_NAME
FROM Forecasting.CONTENT
WHERE ATTRIBUTE_NAME = 'R250 Europe: Quantity"
AND NODE_TYPE = 27
```

By using this node ID, you can retrieve details about the ARIMA equation for this tree. The following DMX statement retrieves the short form of the ARIMA equation for the data series. It also retrieves the intercept from the nested table, NODE_DISTRIBUTION. In this example, the equation is obtained by referencing the node unique ID TA00000007. However, you may have to use a different node ID, and you may obtain slightly different results from your model.

```
SELECT FLATTENED NODE_CAPTION as [Short equation],
(SELECT ATTRIBUTE_NAME, ATTRIBUTE_VALUE
FROM NODE_DISTRIBUTION) as t
FROM Forecasting.CONTENT
WHERE NODE_NAME = 'TA00000007'
```

Example results:

SHORT EQUATION	T.ATTRIBUTE_NAME	T.ATTRIBUTE_VALUE
ARIMA (2,0,7)x(1,0,2)(12)	R250 Europe:Quantity(Intercept)	15.24....
ARIMA (2,0,7)x(1,0,2)(12)	R250 Europe:Quantity(Periodicity)	1
ARIMA (2,0,7)x(1,0,2)(12)	R250 Europe:Quantity(Periodicity)	12

For more information about how to interpret this information, see Mining Model Content for Time Series Models (Analysis Services - Data Mining).

Sample Query 3: Retrieving the Equation for an ARTXP Model

For an ARTxp model, different information is stored at each level of the tree. For more information about the structure of an ARTxp model, and how to interpret the information in the equation, see Mining Model Content for Time Series Models (Analysis Services - Data Mining).

The following DMX statement retrieves information a part of the ARTxp tree that represents the quantity of sales for the R250 model in Europe.

> **NOTE**
> The name of the nested table column, VARIANCE, must be enclosed in brackets to distinguish it from the reserved keyword of the same name. The nested table columns, PROBABILITY and SUPPORT, are not included because they are blank in most cases.

```
SELECT NODE_CAPTION as [Split information],
(SELECT ATTRIBUTE_NAME, ATTRIBUTE_VALUE,
   [VARIANCE]
   FROM NODE_DISTRIBUTION) AS t
FROM Forecasting.CONTENT
WHERE NODE_ATTRIBUTE_NAME = 'R250 Europe:Quantity'
AND NODE_TYPE = 15
```

For more information about how to interpret this information, see Mining Model Content for Time Series Models (Analysis Services - Data Mining).

Creating Predictions on a Time Series Model

Beginning in SQL Server 2008 Enterprise, you can add new data to a time series model and automatically incorporate the new data into the model. You add new data to a time series mining model in one of two ways:

- Use a **PREDICTION JOIN** to join data in an external source to the training data.
- Use a singleton prediction query to provide data one slice at a time. For information about how to create a singleton prediction query, see Data Mining Query Tools.

Understanding the Behavior of Replace and Extend Operations

When you add new data to a time series model, you can specify whether to extend or replace the training data:

- **Extend:** When you extend a data series, Analysis Services adds the new data at the end of the existing training data. The number of training cases also increases.

  Extending the model cases is useful for continuously updating the model with new data. For example, if you want to make the training set grow over time, you would simply extend the model.

  To extend the data, you create a **PREDICTION JOIN** on a time series model, specify the source of the new data, and use the **EXTEND_MODEL_CASES** argument.

- **Replace:** When you replace the data in the data series, Analysis Services keeps the trained model, but uses the new data values to replace some or all of the existing training cases. Therefore, the size of the training data never changes, but the cases themselves are continually being replaced with newer data. If you supply enough new data, you can replace the training data with a completely new series.

  Replacing the model cases is useful when you want to train a model on one set of cases and then apply that model to a different data series.

  To replace the data, you create a **PREDICTION JOIN** on a time series model, specify the source of the new data, and use the **REPLACE_MODEL_CASES** argument.

> **NOTE**
> You cannot make historical predictions when you add new data.

Regardless of whether you extend or replace the training data, predictions always begin at the time stamp that ends the original training set. In other words, if your new data contains n time slices, and you request predictions for time steps 1 through n, the predictions will coincide with the same period as the new data, and you will not get any new predictions.

To get new predictions for time periods that do not overlap with the new data, you must either start predictions at the time slice n+1, or make sure that you request additional time slices.

For example, assume that the existing model has six months' worth of data. You want to extend this model by adding the sales figures from the last three months. At the same time, you want to make a prediction about the next three months. To obtain only the new predictions when you add the new data, specify the starting point as time slice 4, and the ending point as time slice 7. You could also request a total of six predictions, but the time slices for the first three would overlap with the new data just added.

For query examples and more information about the syntax for using **REPLACE_MODEL_CASES** and **EXTEND_MODEL_CASES**, see PredictTimeSeries (DMX).

Making Predictions with EXTEND_MODEL_CASES

Prediction behavior differs depending on whether you extend or replace the model cases. When you extend a model, the new data is attached to the end of the series and the size of the training set increases. However, the time slices used for prediction queries always start at the end of the original series. Therefore, if you add three new data points and request six predictions, the first three predictions returned overlap with the new data. In this case, Analysis Services returns the actual new data points instead of making a prediction, until all the new data points are used up. Then, Analysis Services makes predictions based on the composite series.

This behavior lets you add new data, and then show your actual sales figures in the prediction chart, instead of seeing projections.

For example, to add three new data points and make three new predictions, you would do the following:

- Create a **PREDICTION JOIN** on a time series model, and specify the source of three months' of new data.
- Request predictions for six time slices. To do this, specify 6 time slices, where the starting point is time slice 1, and the ending point is time slice 7. This is true only for EXTEND_MODEL_CASES.
- To get only the new predictions, you specify the starting point as 4 and the ending point as 7.
- You must use the argument **EXTEND_MODEL_CASES**.

  The actual sales figures are returned for the first three time slices, and predictions based on the extended model are returned for the next three time slices.

Making Predictions with REPLACE_MODEL_CASES

When you replace the cases in a model, the size of the model stays the same but Analysis Services replaces the individual cases in the model. This is useful for cross-prediction and scenarios in which maintaining the training data set at a consistent size is important.

For example, one of your stores has insufficient sales data. You could create a general model by averaging sales for all stores in a particular region and then training a model. Then, to make predictions for the store without sufficient sales data, you create a **PREDICTION JOIN** on the new sales data for just that store. When you do this, Analysis Services keeps the patterns derived from the regional model, but replaces the existing training cases with the data from the individual store. As a result, your prediction values will be closer to the trend lines

for the individual store.

When you use the **REPLACE_MODEL_CASES** argument, Analysis Services continually adds new cases to the end of the case set and deletes a corresponding number from the beginning of the case set. If you add more new data than was in the original training set, Analysis Services discards the earliest data. If you supply sufficient new values, the predictions can be based on completely new data.

For example, you trained your model on a case data set that contained 1000 rows. You then add 100 rows of new data. Analysis Services drops the first 100 rows from the training set and adds the 100 rows of new data to the end of the set for a total of 1000 rows. If you add 1100 rows of new data, only the most recent 1000 rows are used.

Here is another example. To add three new month's worth of data and make three new predictions, you would do the following actions:

- Create a **PREDICTION JOIN** on a time series model and use the **REPLACE_MODEL_CASE** argument.
- Specify the source of three months' of new data. This data might be from a completely different source than the original training data.
- Request predictions for six time slices. To do this, specify 6 time slices, or specify the starting point as time slice 1, and the ending point as time slice 7.

> **NOTE**
> Unlike **EXTEND_MODEL_CASES**, you cannot return the same values that you added as input data. All six values returned are predictions that are based on the updated model, which includes both old and new data.

> **NOTE**
> With REPLACE_MODEL_CASES, starting at timestamp 1 you get new predictions based on the new data, which replaces the old training data.

For query examples and more information about the syntax for using **REPLACE_MODEL_CASES** and **EXTEND_MODEL_CASES**, see PredictTimeSeries (DMX).

Missing Value Substitution in Time Series Models

When you add new data to a time series model by using a **PREDICTION JOIN** statement, the new dataset cannot have any missing values. If any series is incomplete, the model must supply the missing values by using either a null, a numeric means, a specific numeric mean, or a predicted value. If you specify **EXTEND_MODEL_CASES**, Analysis Services replaces the missing values with predictions based on the original model. If you use **REPLACE_MODEL_CASES**, Analysis Services replaces the missing values with the value that you specify in the *MISSING_VALUE_SUBSTITUTION* parameter.

List of Prediction Functions

All Microsoft algorithms support a common set of functions. However, the Microsoft Time Series algorithm supports the additional functions, listed in the following table.

Prediction Function	Usage

Lag (DMX)	Returns a number of time slices between the date of the current case and the last date of the training set. A typical use of this function is to identify recent training cases so that you can retrieve detailed data about the cases.
PredictNodeId (DMX)	Returns the node ID for the specified predictable column. A typical use of this function is to identify the node that generated a particular predicted value so that you can review the cases associated with the node, or retrieve the equation and other details.
PredictStdev (DMX)	Returns the standard deviation of the predictions in the specified predictable column. This function replaces the INCLUDE_STATISTICS argument, which is not supported for time series models.
PredictVariance (DMX)	Returns the variance of the predictions for the specified predictable column. This function replaces the INCLUDE_STATISTICS argument, which is not supported for time series models.
PredictTimeSeries (DMX)	Returns predicted historical values or future predicted values for a time series. You can also query time series models by using the general prediction function, Predict (DMX).

For a list of the functions that are common to all Microsoft algorithms, see General Prediction Functions (DMX). For the syntax of specific functions, see Data Mining Extensions (DMX) Function Reference.

See Also

Data Mining Queries
Microsoft Time Series Algorithm
Microsoft Time Series Algorithm Technical Reference
Mining Model Content for Time Series Models (Analysis Services - Data Mining)

Plugin Algorithms

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In addition to the algorithms that Microsoft SQL Server Analysis Services provides, there are many other algorithms that you can use for data mining. Accordingly, Analysis Services provides a mechanism for "plugging in" algorithms that are created by third parties. As long as the algorithms follow certain standards, you can use them within Analysis Services just as you use the Microsoft algorithms. Plugin algorithms have all the capabilities of algorithms that SQL Server Analysis Services provides.

For a full description of the interfaces that Analysis Services uses to communicate with plugin algorithms, see the samples for creating a custom algorithm and custom model viewer that are published on CodePlex Web site.

Algorithm Requirements

To plug an algorithm into Analysis Services, you must implement the following COM interfaces:

IDMAlgorithm
Implements an algorithm that produces models, and implements the prediction operations of the resulting models.

IDMAlgorithmNavigation
Enables browsers to access the content of the models.

IDMPersist
Enables the models that the algorithm trains to be saved and loaded by Analysis Services.

IDMAlgorithmMetadata
Describes the capabilities and input parameters of the algorithm.

IDMAlgorithmFactory
Creates instances of the objects that implement the algorithm interface, and provides Analysis Services with access to the algorithm-metadata interface.

Analysis Services uses these COM interfaces to communicate with plugin algorithms. Although plugin algorithms that you use must support the Microsoft OLE DB for Data Mining specification, they do not have to support all the data mining options in the specification. You can use the MINING_SERVICES schema rowset to determine the capabilities of an algorithm. This schema rowset lists the data mining support options for each plugin algorithm provider.

You must register new algorithms before you use them with Analysis Services. To register an algorithm, include the following information in the .ini file of the instance of Analysis Services on which you want to include the algorithms:

- The algorithm name
- ProgID (this is optional and will only be included for plugin algorithms)
- A flag that indicates whether the algorithm is enabled or not

The following code sample illustrates how to register a new algorithm:

```
<ConfigurationSettings>
```

```
...
```

```
<DataMining>
...
<Algorithms>
...
<Sample_Plugin_Algorithm>
<Enabled>1</Enabled>
<ProgID>Microsoft.DataMining.SamplePlugInAlgorithm.Factory</ProgID>
</Sample_PlugIn_Algorithm>
...
</Algorithms>
...
</DataMining>
...
</ConfigurationSettings>
```

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
DMSCHEMA_MINING_SERVICES Rowset

Mining Structures (Analysis Services - Data Mining)

5/16/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The mining structure defines the data from which mining models are built: it specifies the source data view, the number and type of columns, and an optional partition into training and testing sets. A single mining structure can support multiple mining models that share the same domain. The following diagram illustrates the relationship of the data mining structure to the data source, and to its constituent data mining models.

The mining structure in the diagram is based on a data source that contains multiple tables or views, joined on the CustomerID field. One table contains information about customers, such as the geographical region, age, income and gender, while the related nested table contains multiple rows of additional information about each customer, such as products the customer has purchased. The diagram shows that multiple models can be built on one mining structure, and that the models can use different columns from the structure.

Model 1 Uses CustomerID, Income, Age, Region, and filters the data on Region.

Model 2 Uses CustomerID, Income, Age, Region and filters the data on Age.

Model 3 Uses CustomerID, Age, Gender, and the nested table, with no filter.

Because the models use different columns for input, and because two of the models additionally restrict the data that is used in the model by applying a filter, the models might have very different results even though they are based on the same data. Note that the CustomerID column is required in all models because it is the only available column that can be used as the case key.

This section explains the basic architecture of data mining structures: how you define a mining structure, how you populate it with data, and how you use it to create models. For more information about how to manage or export existing data mining structures, see Management of Data Mining Solutions and Objects.

Defining a Mining Structure

Setting up a data mining structure includes the following steps:

- Define a data source.
- Select columns of data to include in the structure (not all columns need to be added to the model) and defining a key.

- Define a key for the structure, including the key for the bested table, if applicable.
- Specify whether the source data should be separate into a training set and testing set. This step is optional.
- Process the structure.

These steps are described in more detail in the following sections.

Data Sources for Mining Structures

When you define a mining structure, you use columns that are available in an existing data source view. A data source view is a shared object that lets you combine multiple data sources and use them as a single source. The original data sources are not visible to client applications, and you can use the properties of the data source view to modify data types, create aggregations, or alias columns.

If you build multiple mining models from the same mining structure, the models can use different columns from the structure. For example, you can create a single structure and then build separate decision tree and clustering models from it, with each model using different columns and predicting different attributes.

Moreover, each model can use the columns from the structure in different ways. For example, your data source view might contain an Income column, which you can bin in different ways for different models.

The data mining structure stores the definition of the data source and the columns in it in the form of *bindings* to the source data. For more information about data source bindings, see Data Sources and Bindings (SSAS Multidimensional). However, note that you can also create a data mining structure without binding it to a specific data source by using the DMX CREATE MINING STRUCTURE (DMX) statement.

Mining Structure Columns

The building blocks of the mining structure are the mining structure columns, which describe the data that the data source contains. These columns contain information such as data type, content type, and how the data is distributed. The mining structure does not contain information about how columns are used for a specific mining model, or about the type of algorithm that is used to build a model; this information is defined in the mining model itself.

A mining structure can also contain nested tables. A nested table represents a one-to-many relationship between the entity of a case and its related attributes. For example, if the information that describes the customer resides in one table, and the customer's purchases reside in another table, you can use nested tables to combine the information into a single case. The customer identifier is the entity, and the purchases are the related attributes. For more information about when to use nested tables, see Nested Tables (Analysis Services - Data Mining).

To create a data mining model in SQL Server Data Tools (SSDT), you must first create a data mining structure. The Data Mining wizard walks you through the process of creating a mining structure, choosing data, and adding a mining model.

If you create a mining model by using Data Mining Extensions (DMX), you can specify the model and the columns in it, and DMX will automatically create the required mining structure. For more information, see CREATE MINING MODEL (DMX).

For more information, see Mining Structure Columns.

Dividing the Data into Training and Testing Sets

When you define the data for the mining structure, you can also specify that some of the data be used for training, and some for testing. Therefore, it is no longer necessary to separate your data in advance of creating a data mining structure. Instead, while you create your model, you can specify that a certain percentage of the data be held out for testing, and the rest used for training, or you can specify a certain number of cases to use as the test data set. The information about the training and testing data sets is cached with the mining structure,

and as a result, the same test set can be used with all models that are based on that structure.

For more information, see Training and Testing Data Sets.

Enabling Drillthrough

You can add columns to the mining structure even if you do not plan to use the column in a specific mining model. This is useful if, for example, you want to retrieve the e-mail addresses of customers in a clustering model, without using the e-mail address during the analysis process. To ignore a column during the analysis and prediction phase, you add it to the structure but do not specify a usage for the column, or set the usage flag to Ignore. Data flagged in this way can still be used in queries if drillthrough has been enabled on the mining model, and if you have the appropriate permissions. For example, you could review the clusters resulting from analysis of all customers, and then use a drillthrough query to get the names and e-mail addresses of customers in a particular cluster, even though those columns of data were not used to build the model.

For more information, see Drillthrough Queries (Data Mining).

Processing Mining Structures

A mining structure is just a metadata container until it is processed. When you process a mining structure, Analysis Services creates a cache that stores statistics about the data, information about how any continuous attributes are discretized, and other information that is later used by mining models. The mining model itself does not store this summary information, but instead references the information that was cached when the mining structure was processed. Therefore, you do not need to reprocess the structure each time you add a new model to an existing structure; you can process just the model.

You can opt to discard this cache after processing, if the cache is very large or you want to remove detailed data. If you do not want the data to be cached, you can change the **CacheMode** property of the mining structure to **ClearAfterProcessing**. This will destroy the cache after any models are processed. Setting the **CacheMode** property to **ClearAfterProcessing** will disable drillthrough from the mining model.

However, after you destroy the cache, you will not be able to add new models to the mining structure. If you add a new mining model to the structure, or change the properties of existing models, you would need to reprocess the mining structure first. For more information, see Processing Requirements and Considerations (Data Mining).

Viewing Mining Structures

You cannot use viewers to browse the data in a mining structure. However, in SQL Server Data Tools (SSDT), you can use the **Mining Structure** tab of Data Mining Designer to view the structure columns and their definitions. For more information, see Data Mining Designer.

If you want to review the data in the mining structure, you can create queries by using Data Mining Extensions (DMX). For example, the statement `SELECT * FROM <structure>.CASES` returns all the data in the mining structure. To retrieve this information, the mining structure must have been processed, and the results of processing must be cached.

The statement `SELECT * FROM <model>.CASES` returns the same columns, but only for the cases in that particular model. For more information, see SELECT FROM <structure>.CASES and SELECT FROM <model>.CASES (DMX).

Using Data Mining Models with Mining Structures

A data mining model applies a mining model algorithm to the data that is represented by a mining structure. A mining model is an object that belongs to a particular mining structure, and the model inherits all the values of the properties that are defined by the mining structure. The model can use all the columns that the mining structure contains or a subset of the columns. You can add multiple copies of a structure column to a structure. You can also add multiple copies of a structure column to a model, and then assign different names, or *aliases*,

to each structure column in the model. For more information about aliasing structure columns, see Create an Alias for a Model Column and Mining Model Properties.

For more information about the architecture of data mining models, see Mining Models (Analysis Services - Data Mining).

Related Tasks

Use the links provided her to learn more about how to define, manage, and use mining structures.

TASKS	LINKS
Work with relational mining structures	Create a New Relational Mining Structure Add a Nested Table to a Mining Structure
Work with mining structures based on OLAP cubes	Create a New OLAP Mining Structure
Work with columns in a mining structure	Add Columns to a Mining Structure Remove Columns from a Mining Structure
Change or query mining structure properties and data	Change the Properties of a Mining Structure
Work with the underlying data sources and update source data	Edit the Data Source View used for a Mining Structure Process a Mining Structure

See Also

Database Objects (Analysis Services - Multidimensional Data)
Mining Models (Analysis Services - Data Mining)

Mining Structure Columns

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You define the columns in a mining structure when you create the mining structure, by choosing columns of external data and then specifying how the data is to be used for modeling. Therefore, mining structure columns are more than copies of data from a data source: they define how the data from the source is to be used by the mining model. You can assign properties that determine how the data is discretized, properties that describe how the data values are distributed

Mining structure columns are designed to be flexible and extensible, because each algorithm that you use to build a mining model may use different columns from the structure to interpret the data. Rather than have one set of data for each model, you can use a single mining structure and use the columns in it to customize the data for each model.

Defining Mining Structure Columns

The basic data types and content types that define structure columns are derived from the data source that you use to create the structure. You can change these settings within the mining structure, and you can also set modeling flags and set the distribution for continuous columns.

The definition of a mining structure column must contain the following information:

- **ID**: The unique name of the column, often the same as the name. This cannot be changed after you create the mining structure, whereas the name can be changed.
- **Name**: A name or alias for the column.
- **Content**: An enumeration that describes whether the data is discrete or continuous.
- **Type**: An enumeration that indicates the general data type.
- **Distribution**: An enumeration that describes the expected distribution of values. A distribution is included if the column is continuous.
- **Modeling flags**: An enumeration that indicates how to handle missing values and so forth. Modeling flags can also be defined on the mining model, but the model flags are different than the flags used on structure columns.
- **Bindings**: Properties that specify the source data.

Third-party algorithms may also include custom properties that can be defined on the mining structure column.

For more information about the data mining structure and the data mining model, see Mining Structures (Analysis Services - Data Mining).

Related Content

See the following topics for more information about how to define and use mining structure columns.

TOPIC	LINKS
Describes the data types that you can use to define a mining structure column.	Data Types (Data Mining)
Describes the content types that are available for each type of data that is contained in a mining structure column. Content types are dependent on data type. The content type is assigned at the model level, and determines how the column data is used by the model.	Content Types (Data Mining)
Introduces the concept of nested tables, and explains how nested tables can be added to the data source as mining structure columns.	Classified Columns (Data Mining)
Lists and explains the distribution properties that you can set on a mining structure column to specify the expected distribution of values in the column.	Column Distributions (Data Mining)
Explains the concept of discretization (sometimes referred to as *binning*) and describes the methods that Analysis Services provides for discretizing continuous numeric data.	Discretization Methods (Data Mining)
Describes the modeling flags that you can set on a mining structure column.	Modeling Flags (Data Mining)
Describes classified columns, which are a special type of column that you can use to relate one mining structure column to another.	Classified Columns (Data Mining)
Learn to add and modify mining structure columns.	Mining Structure Tasks and How-tos

See Also

Mining Structures (Analysis Services - Data Mining)
Mining Model Columns

Data Types (Data Mining)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a mining model or a mining structure in Microsoft SQL Server Analysis Services, you must define the data types for each of the columns in the mining structure. The data type tells the analysis engine whether the data in the data source is numerical or text, and how the data should be processed. For example, if your source data contains numerical data, you can specify whether the numbers be treated as integers or by using decimal places.

Analysis Services supports the following data types for mining structure columns:

DATA TYPE	SUPPORTED CONTENT TYPES
Text	Cyclical, Discrete, Discretized, Key Sequence, Ordered, Sequence
Long	Continuous, Cyclical, Discrete, Discretized, Key, Key Sequence, Key Time, Ordered, Sequence, Time Classified
Boolean	Cyclical, Discrete, Ordered
Double	Continuous, Cyclical, Discrete, Discretized, Key, Key Sequence, Key Time, Ordered, Sequence, Time Classified
Date	Continuous, Cyclical, Discrete, Discretized, Key, Key Sequence, Key Time, Ordered

> **NOTE**
> The Time and Sequence content types are only supported by third-party algorithms. The Cyclical and Ordered content types are supported, but most algorithms treat them as discrete values and do not perform special processing.

The table also shows the *content types* supported for each data type.

The content type is specific to data mining and lets you customize the way that data is processed or calculated in the mining model. For example, even if your column contains numbers, you might need to model them as discrete values. If the column contains numbers, you can also specify that they be binned, or discretized, or specify that the model handle them as continuous values. Thus, the content type can have a huge effect on the model.. For a list of all the content types, see Content Types (Data Mining).

> **NOTE**
> In other machine learning systems, you might encounter the terms *nominal data*, *factors* or *categories*, *ordinal data*, or *sequence data*. In general, these correspond to content types. In SQL Server, the data type specifies only the value type for storage, not its usage in the model.

Specifying a Data Type

If you create the mining model directly by using Data Mining Extensions (DMX), you can define the data type for each column as you define the model, and Analysis Services will create the corresponding mining structure with the specified data types at the same time. If you create the mining model or mining structure by using a wizard, Analysis Services will suggest a data type, or you can choose a data type from a list.

Changing a Data Type

If you change the data type of a column, you must always reprocess the mining structure and any mining models that are based on that structure. Sometimes if you change the data type, that column can no longer be used in a particular model. In that case, Analysis Services will either raise an error when you reprocess the model, or will process the model but leave out that particular column.

See Also

Content Types (Data Mining)
Content Types (DMX)
Data Mining Algorithms (Analysis Services - Data Mining)
Mining Structures (Analysis Services - Data Mining)
Data Types (DMX)
Mining Model Columns
Mining Structure Columns

Nested Tables (Analysis Services - Data Mining)

5/16/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In SQL Server Analysis Services, data must be fed to a data mining algorithm as a series of cases that are contained within a case table. However, not all cases can be described by a single row of data. For example, a case might be derived from two tables: one table that contains customer information, and another table that contains customer purchases. A single customer in the customer information table might have multiple items in the customer purchases table, which makes it difficult to describe the data by using a single row. Analysis Services provides a unique method for handling these cases, by using *nested tables*. The concept of a nested table is demonstrated in the following illustration.

In this diagram, the first table, which is the parent table, contains information about customers, and associates a unique identifier for each customer. The second table, the child table, contains the purchases for each customer. The purchases in the child table are related to the parent table by the unique identifier, the **CustomerKey** column. The third table in the diagram shows the two tables combined.

A nested table is represented in the case table as a special column that has a data type of **TABLE**. For any particular case row, this kind of column contains selected rows from the child table that pertain to the parent table.

The data in a nested table can be used for prediction or for input, or for both. For example, you might have two nested table columns in a model: one nested table column might contain a list of the products that a customer has purchased, while the other nested table column contains information about the customer's hobbies and interests, possibly obtained from a survey. In this scenario, you could use the customer's hobbies and interests as an input for analyzing purchasing behavior, and predicting likely purchases.

Joining Case Tables and Nested Tables

In order to create a nested table, the two source tables must contain a defined relationship so that the items in one table can be related to the other table. In SQL Server Data Tools (SSDT), you can define this relationship in the data source view.

> **NOTE**
>
> The **CustomerKey** field is the relational key that is used to link the case table and the nested table within the data source view definition, and to establish the relationship of the columns within the mining structure. However, typically you should not use this relational key in mining models built on that structure. Usually it is best to omit the relational key column from the mining model if it serves only to join the tables and does not provide information that is interesting for analysis.

You can create nested tables programmatically by either using Data Mining Extensions (DMX) or Analysis Management Objects (AMO), or you can use the Data Mining Wizard and Data Mining Designer in SQL Server Data Tools (SSDT).

Using Nested Table Columns in a Mining Model

In the case table, the key is often a customer ID, a product name, or date in a series: data that uniquely identifies a row in the table. . However, in nested tables, the key is typically not the relational key (or foreign key) but rather the column that represents the attribute that you are modeling.

For example, if the case table contains orders, and the nested table contains items in the order, you would be interested in modeling the relationship between items stored in the nested table across multiple orders, which are stored in the case table. Therefore, although the **Items** nested table is joined to the **Orders** case table by the relational key **OrderID**, you should not use **OrderID** as the nested table key. Instead, you would select the **Items** column as the nested table key, because that column contains the data that you want to model. In most cases, you can safely ignore **OrderID** in the mining model, because the relationship between the case table and the nested table has already been established by the data source view definition.

When you choose a column to use as the nested table key, you must ensure that the values in that column are unique for each case. For example, if the case table represents customers and the nested table represents items purchased by the customer, you must ensure that no item is listed more than one time per customer. If a customer has purchased the same item more than one time, you might want to create a different view that has a column that aggregates the count of purchases for each unique product.

How you decide to handle duplicate values in a nested table depends on the mining model that you are creating and the business problem that you are solving. In some scenarios you might not care how many times a customer has purchased a particular product, but want to check for the existence of at least one purchase. In other scenarios, the quantity and sequence of purchases might be very important.

If the order of items is important, you might need an additional column that indicates the sequence. When you use the sequence clustering algorithm to create a model, you must choose an additional *key sequence* column to represent the order of the items. The key sequence column is a special kind of nested key that is used only in sequence clustering models, and requires a unique numeric data type. For example, integers and dates can both be used as a key sequence column, but all sequence values must be unique. In addition to the key sequence column, a sequence clustering model also has a nested table key that represents the attribute that is being modeled, such as the products that have been purchased.

Using Non-Key Nested Columns from a Nested Table

After you have defined the join between the case table and the nested table, and you have chosen a column that contains interesting and unique attributes to use as the nested table key, you can include other columns from the nested table to use as input to the model. All columns from the nested table can be used for input, prediction and input, or for prediction only.

For example, if the nested table contains the columns **Product**, **ProductQuantity**, and **ProductPrice**, you might choose **Product** as the nested table key, but add **ProductQuantity** to the mining structure to use as input.

Filtering Nested Table Data

In SQL Server 2017, you can create filters on the data that is used to train or test a data mining model. A filter can be used to affect the composition of the model, or to test the model on a subset of cases. Filters can also be applied to nested tables. However, there are limitations on the syntax that can be used with nested tables.

Often when you apply a filter to a nested table you are testing for the existence or nonexistence of an attribute. For example, you can apply a filter that restricts the cases used in the model to only those cases that have a specified value in the nested table. Or, you could restrict the cases used in the model to customers who have not purchased

a particular item.

When you create filters on a nested table, you can also use operators such as greater than or less than. For example, you could restrict the cases used in the model to customers who had purchased at least n units of the target product. The ability to filter on nested table attributes provides great flexibility for customizing models.

For more information about how to create and use model filters, see Filters for Mining Models (Analysis Services - Data Mining).

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Mining Structures (Analysis Services - Data Mining)

Column Distributions (Data Mining)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, you can define column distributions in a mining structure, to affect how algorithms process the data in those columns when you create mining models. For some algorithms, it is useful to define the distribution of any continuous columns before you process the model, if the columns are known to contain common distributions of values. If you do not define the distributions, the resulting mining models may produce less accurate predictions than if the distributions were defined, because the algorithms will have less information from which to interpret the data.

The algorithms that are available in Analysis Services support the following distribution types:

Normal

The values for the continuous column form a histogram with a normal distribution.

Log Normal

The values for the continuous column form a histogram, where the curve is elongated at the upper end and is skewed toward the lower end.

Uniform

The values for the continuous column form a flat curve, in which all values are equally likely.

For more information about the algorithms that Analysis Services provides, see Data Mining Algorithms (Analysis Services - Data Mining).

See Also

Content Types (Data Mining)

Mining Structures (Analysis Services - Data Mining)
Discretization Methods (Data Mining)
Distributions (DMX)
Mining Structure Columns

Discretization Methods (Data Mining)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Some algorithms that are used to create data mining models in SQL Server Analysis Services require specific content types in order to function correctly. For example, the Microsoft Naive Bayes algorithm cannot use continuous columns as input and cannot predict continuous values. Additionally, some columns may contain so many values that the algorithm cannot easily identify interesting patterns in the data from which to create a model.

In these cases, you can discretize the data in the columns to enable the use of the algorithms to produce a mining model. *Discretization* is the process of putting values into buckets so that there are a limited number of possible states. The buckets themselves are treated as ordered and discrete values. You can discretize both numeric and string columns.

There are several methods that you can use to discretize data. If your data mining solution uses relational data, you can control the number of buckets to use for grouping data by setting the value of the DiscretizationBucketCount property. The default number of buckets is 5.

If your data mining solution uses data from an Online Analytical Processing (OLAP) cube, the data mining algorithm automatically computes the number of buckets to generate by using the following equation, where n is the number of distinct values of data in the column:

```
Number of Buckets = sqrt(n)
```

If you do not want Analysis Services to calculate the number of buckets, you can use the DiscretizationBucketCount property to manually specify the number of buckets.

The following table describes the methods that you can use to discretize data in Analysis Services.

DISCRETIZATION METHOD	DESCRIPTION
AUTOMATIC	Analysis Services determines which discretization method to use.
CLUSTERS	The algorithm divides the data into groups by sampling the training data, initializing to a number of random points, and then running several iterations of the Microsoft Clustering algorithm using the Expectation Maximization (EM) clustering method. The **CLUSTERS** method is useful because it works on any distribution curve. However, it requires more processing time than the other discretization methods. This method can only be used with numeric columns.

DISCRETIZATION METHOD	DESCRIPTION
EQUAL_AREAS	The algorithm divides the data into groups that contain an equal number of values. This method is best used for normal distribution curves, but does not work well if the distribution includes a large number of values that occur in a narrow group in the continuous data. For example, if one-half of the items have a cost of 0, one-half the data will occur under a single point in the curve. In such a distribution, this method breaks the data up in an effort to establish equal discretization into multiple areas. This produces an inaccurate representation of the data.

Remarks

- You can use the **EQUAL_AREAS** method to discretize strings.
- The **CLUSTERS** method uses a random sample of 1000 records to discretize data. Use the **EQUAL_AREAS** method if you do not want the algorithm to sample data.

See Also

Content Types (Data Mining)
Content Types (DMX)
Data Mining Algorithms (Analysis Services - Data Mining)
Mining Structures (Analysis Services - Data Mining)
Data Types (Data Mining)
Mining Structure Columns
Column Distributions (Data Mining)

Classified Columns (Data Mining)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you define a classified column, you create a relationship between the current column and another column in the mining structure. The data in the mining structure column that you designate as the classified column contains categorical information that describes the values in another column in the mining structure.

For example, suppose you have two columns with numerical data: one column, [Yearly Purchases], contains the total yearly purchases per customer for a specific calendar year, and the other column, [Standard Deviations], contains the standard deviations for those values. In this case you could designate the [Yearly Purchases] column as the classified column, and the model would be able to use this relationship in analysis.

> **NOTE**
> The algorithms provided in Analysis Services do not support the use of classified columns; this feature is provided for use in creating custom algorithms.

Defining a Classified Column

The data type of a classified column must be either **Long** or **Double**.

The following list describes the content types that Analysis Services supports for classified columns.

PROBABILITY
The value in the column is the probability of the associated value, and is a number between 0 and 1.

VARIANCE
The value in the column is the variance of the associated value.

STDEV
The value in the column is the standard deviation of the associated value.

PROBABILITY_VARIANCE
The value in the column is the variance of the probability for the associated value.

PROBABILITY_STDEV
The value in the column is the standard deviation of the probability for the associated value.

SUPPORT
The value in the column is the weight, or case replication factor, of the associated value.

See Also

Content Types (Data Mining)
Mining Structures (Analysis Services - Data Mining)
Data Types (Data Mining)

Drillthrough on Mining Structures

10/26/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Drillthrough means the ability to query either a mining model or a mining structure and get detailed data that is not exposed in the model.

SQL Server 2017 provides two different options for drilling through into case data. You can drill through to the data that were used to build the mining model, or you can drill through to the source data in the mining structure.

Drillthrough to Model Cases vs. Drillthrough to Structure

Drilling through to **model cases** is useful for finding additional details about rules, patterns or clusters in a model.

In contrast, **drillthrough to structure** data is intended to provide access to information that was not made available in the model. For example, if you have the appropriate permissions, you might want to find out which rows of data were used for training the model and which were used for testing.

You can also view attributes of the data that were not used in analysis, provided they have been included in the structure definition. For example, often mining structures support many different kinds of models, and some structure columns might have been excluded from a model because the data type was incompatible or the data was not useful for analysis. For example, you would not use customer contact information in a clustering model, even if the data was included in the structure, but by enabling drillthrough you gain access to this information without running separate queries against the data source.

Enabling Drillthrough to Structure Data

To use drillthrough on the mining structure, the following conditions must be met:

- Drillthrough on the model must also be enabled. By default, drillthrough of both kinds is disabled. To enable drillthrough in the Data Mining Wizard, select the option to enable drillthrough to model cases on the final page of the wizard. You can also add the ability to drillthrough on a model later by changing the **AllowDrillthrough** property.
- If you create the mining structure by using DMX, use the WITH DRILLTHROUGH clause. For more information, see CREATE MINING STRUCTURE (DMX).
- Drillthrough works by retrieving information about the training cases that was cached when you processed the mining structure. Therefore, if you clear the cached data after processing the structure by changing the MiningStructureCacheMode property to **ClearAfterProcessing**, drillthrough will not work. To enable drillthrough to structure columns, you must change the MiningStructureCacheMode property to **KeepTrainingCases** and then reprocess the structure.
- Verify that both the mining structure and the mining model have the AllowDrillThrough property set to **True**. Moreover, you must be a member of a role that has drillthrough permissions on both the structure and the model.

Security Issues for Drillthrough

Drillthrough permissions are set separately on the structure and model. The model permission lets you drill through from the model, even if you do not have permissions on the structure. Drillthrough permissions on the structure provide the additional ability to include structure columns in drillthrough queries from the model, by

using the StructureColumn (DMX) function.

For information about how to create roles and assign permissions in Analysis Services, see Role Designer (Analysis Services - Multidimensional Data).

> **NOTE**
> If you enable drillthrough on both the mining structure and the mining model, any user who is a member of a role that has drillthrough permissions on the mining model can also view columns in the mining structure, even if those columns are not included in the mining model. Therefore, to protect sensitive data, you should set up the data source view to mask personal information, and allow drillthrough access on the mining structure only when necessary.

Related Tasks

See the following topics for more information about how to use drillthrough with mining models.

Use drillthrough to structure from the mining model viewers	Use Drillthrough from the Model Viewers
See examples of drillthrough queries for specific model types.	Data Mining Queries
Get information about permissions that apply to specific mining structures and mining models.	Grant permissions on data mining structures and models (Analysis Services)

See Also

Drillthrough on Mining Models

Properties for Mining Structure and Structure Columns

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can set or change the properties for a mining structure and for its associated columns and nested tables by using the **Mining Structure** tab of Data Mining Designer. Properties that you set in this tab are propagated to each mining model that is associated with the structure.

> **NOTE**
> If you change the value of any property in the mining structure, even metadata such as a name or description, the mining structure and its models must be reprocessed before you can view or query the model.

Properties of Mining Structures and Mining Structure Columns

The following table describes the properties for the mining structure and the mining structure columns that are specific to data mining, and that you can view or configure in the **Mining Structure** tab. To view or configure these properties, right-click an element in the tree view, and then click **Properties**.

- To view the properties of the structure, click the mining structure heading.
- To view the properties of a column or a nested table, click the column name.

Properties of the Mining Structure

PROPERTY	DESCRIPTION
CacheMode	Specifies whether cases used in training should be cached or discarded after training is completed. **Note:** This property must be set to **KeepTrainingCases** to enable drillthrough and holdout.
Collation	Specifies the default collation for the column. If a collation is not specified, the collation of the server is used.
Description	Describes the mining structure. As a best practice, the description should state the purpose and composition of the data in the structure.
ErrorConfiguration (default)	Specifies options for special handling of errors, if any.
HoldoutMaxCases	Specifies the maximum number of structure cases that can be reserved as a test data set. If values are specified for both **HoldoutMaxCases** and **HoldoutPercent**, the conditions are combined. **Note:** To set this property, CacheMode must be set to **KeepTrainingCases**.

PROPERTY	DESCRIPTION
HoldoutPercent	Specifies the percentage of the structure cases to reserve as a test data set. If values are specified for both **HoldoutMaxCases** and **HoldoutPercent**, the conditions are combined. **Note:** To set this property, CacheMode must be set to **KeepTrainingCases**.
HoldoutSeed	Specifies a seed to initialize partitioning of the holdout test set, to ensure that the test data set can be re-created. **Note:** To set this property, CacheMode must be set to **KeepTrainingCases**.
ID	Displays the unique identifier of the mining structure. The name that you assigned to the mining structure when you created the structure is used as the ID. If you later change the name by typing a new value for the **Name** property, the new name is used as an alias only; the ID does not change.
Language	Specifies the language for the captions in the mining structure.
Name	Specifies the name or alias of the mining structure. If you change the value for the Name property, the new name is used as a caption or alias only; the identifier for the mining structure does not change.
Source	Displays the name of the data source, and the type of data source.

Properties of the Mining Structure Columns

PROPERTY	DESCRIPTION
ClassifiedColumns	Identifies the column that a classified column describes.
Content	The content type of the column.
Description	Describes the column. As a best practice, the description of the column should provide information about how the data in the column has been derived or altered for data mining.
DiscretizationBucketCount	Displays the number of buckets in the discretized column. Enabled only if the content type is set to **Discretized**. This property is read-only.
DiscretizationMethod	Displays the method that was used to discretize the column. Enabled only if the content type is set to **Discretized**. This property is read-only.
Distribution	Specifies the distribution of content in the column.

PROPERTY	DESCRIPTION
ID	Displays the identifier of the column. If you change the value of the Name property of the column, it does not affect the value of the ID property.
IsKey	Indicates whether the column is a key column.
KeyColumns	Contains the definition of a column that is the key or is part of the key for an attribute.
ModelingFlags	Sets additional parameters that are made available by the algorithm.
Name	The name of the column.
NameColumn	Identifies the column that provides the name of the parent element.
Source	Displays the source of the column. For relational data sources, the value is always **(none)**. For structures based on an OLAP cube, the value is the MDX statement that defines the slice used as the source for the nested table.
SourceMeasureGroup	Displays the source of the measure group. For relational data sources, the value is always **(none)**. For structures based on an OLAP cube, the value is the MDX statement that defines the slice used as the source for the nested table.
Type	The data type for the content in the column.

For more information about setting or changing properties, see Mining Structure Tasks and How-tos.

See Also

Create a Relational Mining Structure
Mining Structure Columns

Mining Structure Tasks and How-tos

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The **Mining Structure** tab of Data Mining Designer in SQL Server Data Tools (SSDT) contains tools that you can use to create, edit, and process a mining structure.

In This Section

- Create a New Relational Mining Structure
- Create a New OLAP Mining Structure
- Add Columns to a Mining Structure
- Remove Columns from a Mining Structure
- Add a Nested Table to a Mining Structure
- Change the Properties of a Mining Structure
- Edit the Data Source View used for a Mining Structure
- Process a Mining Structure

See Also

Designing and Implementing How-to Topics (Analysis Services - Data Mining)

Create a New Relational Mining Structure

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use the Data Mining Wizard to create a new mining structure, using data from a relational database or other source, and then save the structure and any related models to a Microsoft SQL Server Analysis Services database.

To create a relational mining structure

1. In Solution Explorer, right-click the **Mining Structures** folder in an Analysis Services project, and then click **New Mining Structure**.

   The Data Mining Wizard opens.

2. On the **Welcome to the Data Mining Wizard** page, click **Next**.

3. On the **Select the Definition Method** page, select **From existing relational database or data warehouse**, and then click **Next**.

4. On the **Select the Data Mining Technique** page, select the data mining algorithm that you want to use, and then click **Next**.

   For more information about data mining algorithms, see Data Mining Algorithms (Analysis Services - Data Mining).

5. On the **Select Data Source View** page, under **Available data source views**, click the data source view that you want to use, and then click **Next**.

   For more information about creating a data source view, see Data Source Views in Multidimensional Models.

6. On the **Specify Table Types** page, under **Input tables**, select a case table and a nested table.

7. On the **Specify the Training Data** page, under **Mining model structure**, select the key, input, and predictable columns.

   After you select the predictable column, you can click the **Suggest** button to open the **Suggest Related Columns** dialog box. You can accept the suggested columns by clicking **OK** in this dialog box to include the selected columns in the mining structure, or you can change the selections in the **Input** column first, and then click **OK**. To ignore the suggestions, click **Cancel**.

8. Click **Next**.

9. On the **Specify Columns' Content and Data Type** page, under **Mining model structure**, you can adjust the content type and data type for each column.

   > **NOTE**
   >
   > You can click **Detect** to automatically detect whether a column contains continuous or discrete data. After you click this button, the column content and data types will be updated in the **Content Type** and **Data Type** columns. For more information about content types and data types, see Content Types (Data Mining) and Data Types (Data Mining).

10. Click **Next**.

11. On the **Completing the Wizard** page, provide a name for the mining structure and the related initial mining model that will be created, and then click **Finish**.

See Also

Mining Structure Tasks and How-tos

Create a New OLAP Mining Structure

11/27/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can use the Data Mining Wizard in Microsoft SQL Server Analysis Services to create a mining structure that uses data from a multidimensional model. Mining models that are based on OLAP cubes can use the column and values in fact tables, dimensions, and measure groups as attributes for analysis.

To create a new OLAP mining structure

1. In Solution Explorer in SQL Server Data Tools (SSDT), right-click the **Mining Structures** folder in an Analysis Services project, and then click **New Mining Structure** to open the Data Mining Wizard.

2. On the **Welcome to the Data Mining Wizard** page, click **Next**.

3. On the **Select the Definition Method** page, select **From existing cube**, and then click **Next**.

   If you get an error with the message, Unable to retrieve a list of supported data mining algorithms, open the **Project Properties** dialog box and verify that you have specified the name of an Analysis Services instance that supports multidimensional models. You cannot create mining models on an instance of Analysis Services that supports tabular modeling.

4. On the **Create the Data Mining Structure** page, decide whether you will create a mining structure only, or a mining structure plus one related mining model. Generally it is easier to create a mining model at the same time, so that you can be prompted to include necessary columns.

   If you will create a mining model, select the data mining algorithm that you want to use, and then click **Next**. For more information about how to choose an algorithm, see Data Mining Algorithms (Analysis Services - Data Mining).

5. On the **Select the Source Cube Dimension** page, under **Select a Source Cube Dimension**, locate the dimension that contains the majority of your case data.

   For example, if you are trying to identify customer groupings, you might choose the Customer dimension; if you are trying to analyze purchases across transactions, you might choose the Internet Sales Order Details dimension. You are not restricted to using only the data in this dimension, but it should contain important attributes to use in analysis.

   Click **Next**.

6. On the **Select the Case Key** page, under **Attributes**, select the attribute that will be the key of the mining structure, and then click **Next**.

   Typically the attribute that you use as key for the mining structure is also a key for the dimension and will be pre-selected.

7. On the **Select Case Level Columns** page, under **Related Attributes and Measures**, select the attributes and measures that contain values you want to add to the mining structure as case data. Click **Next**.

8. On the **Specify Mining Model Column Usage** page, under **Mining model structure**, first set the predictable column, and then choose columns to use as inputs.

   - Select the checkbox in the leftmost column to include the data in the mining structure. You can include columns in the structure that you will use for reference, but not use them for analysis.

- Select the checkbox in the **Input** column to use the attribute as a variable in analysis.
- Select the checkbox in the **Predict** column only for predictable attributes.

Note that columns you have designated as keys cannot be used for input or prediction.

Click **Next**.

9. On the **Specify Mining Model Column Usage** page, you can also add and remove nested tables to the mining structure, using **Add Nested Tables** and **Nested Tables**.

   In an OLAP mining model, a nested table is another set of data within the cube that has a one-to-many relationship with the dimension that represents the case attributes. Therefore, when the dialog box opens, it pre-selects measure groups that are already related to the dimension you selected as the case table. At this point, you would choose a different dimension that contains additional information useful for analysis.

   For example, if you are analyzing customers, you would use the [Customer] dimension as the case table. For the nested table, you might add the reason customers cited when making a purchase, which is included in the [Sales Reason] dimension.

   If you add nested data, you must specify two additional columns:

   - The key of the nested table: This should be pre-selected on the page, **Select Nested Table Key**.
   - The attributes or attributes to use for analysis: The page, **Select Nested Table Columns**, provides a list of measures and attributes in the nested table selection.
     - For each attribute that you include in the model, check the box in the left column.
     - If you want to use the attribute for analysis only, check **Input**.
     - If you want to include the column as one of the predictable attributes for the model, select **Predict**.
     - Any item that you include in the structure but do not specify as an input or predictable attribute is added to the structure with the flag **Ignore**; this means that the data is processed when you build the model but is not used in analysis, and is available only for drillthrough. This can be handy if you want to include details such as customer names but don't want to use them in analysis.

   Click **Finish** to close the part of the wizard that works with nested tables. You can repeat the process to add multiple nested columns.

10. On the **Specify Columns' Content and Data Type** page, under **Mining model structure**, set the content type and data type for each column.

    > **NOTE**
    > OLAP mining models do not support using the **Detect** feature to automatically detect whether a column contains continuous or discrete data.

    Click **Next**.

11. On the **Slice Source Cube** page, you can filter the data that is used to create the mining structure.

    Slicing the cube lets you restrict the data that is used to build the model. For example, you could build separate models for each region by slicing on the Geography hierarchy and

    - **Dimension**: Choose a related dimension from the dropdown list.

- **Hierarchy**: Select the level of the dimension hierarchy at which you want to apply the filter. For example, if you are slicing by the [Geography] dimension, you would choose a hierarchy level such as [Region Country Name] .
- **Operator**: Choose an operator from the list.
- **Filter Expression**: Type a value or expression to serve as the filter condition, or use the dropdown list to select a value from the list of members at the specified level of the hierarchy.

  For example, if you selected [Geography] as the dimension and [Region Country Name] as the hierarchy level, the dropdown list contains all the valid countries that you can use as a filter condition. You can make multiple selections. As a result, the data in the mining structure will be limited to cube data from these geographical areas.
- **Parameters**: Ignore this check box. This dialog box supports multiple cube filtering scenarios and this option is not relevant to building a mining structure.

Click **Next**.

12. On the **Split data into training and testing sets** page, specify a percentage of the mining structure data to reserve for testing, or specify the maximum number of test cases. Click **Next**.

    If you specify both values, the limits are combined to use whichever is lowest.
13. On the **Completing the Wizard** page, provide a name for the new OLAP mining structure and the initial mining model.
14. Click **Finish**.
15. On the **Completing the Wizard** page, you also have the option to create a mining model dimension and/or a cube using the mining model dimension. These options are supported only for models built using the following algorithms:

    - Microsoft Clustering algorithm
    - Microsoft Decision Trees algorithm
    - Microsoft Association Rules algorithm

    Create mining model dimension: Select this check box and provide a type name for the mining model dimension. When you use this option, a new dimension is created within the original cube that was used to build the mining structure. You can use this dimension to drill down and conduct further analysis. Because the dimension is located within the cube, the dimension is automatically mapped to the case data dimension.

    Create cube using mining model dimension: Select this check box, and provide a name for the new cube. When you use this option, a new cube is created that contains both the existing dimensions that were used in building the structure, and the new data mining dimension that contains the results from the model.

See Also

Mining Structure Tasks and How-tos

Add Columns to a Mining Structure

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use Data Mining Designer in SQL Server Data Tools (SSDT) to add columns to a mining structure after you have defined it in the Data Mining Wizard. You can add any column that exists in the data source view that was used to define the mining structure.

> **NOTE**
> You can add multiple copies of columns to a mining structure; however, you should avoid using more than one instance of the column within the same model, to avoid false correlations between the source and the derived column.

To add a column to a mining structure

1. Select the **Mining Structure** tab in Data Mining Designer.
2. Right-click the mining structure and select **Add a Column**.

   The **Select a Column** dialog box opens.
3. Under **Source table**, select the table in the data source view where the column resides.
4. Under **Source column**, select the column that you want to add to the mining structure.
5. Click **OK**.

> **NOTE**
> If you add a column that already exists, a copy will be included in the structure, and the name appended with a "1". You can change the name of the copied column to something more descriptive by typing a new name in the **Name** property of the mining structure column.

See Also

Mining Structure Tasks and How-tos

Remove Columns from a Mining Structure

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can use Data Mining Designer to remove columns from a mining structure after the structure has already been created. Reasons to remove a mining structure column might include the following:

- The mining structure contains multiple copies of a column and you want to avoid the use of duplicate data in a model.
- The data should be protected, but drillthrough has been enabled.
- The data is unused in modeling and should not be processed.

Deleting a column from the mining structure does not change the column in the data source view or in the external data; only metadata is deleted. However, when you change the columns used in a mining structure, you must reprocess the structure and any models based on it.

To remove a column from the mining structure

1. Select the **Mining Structure** tab in Data Mining Designer.
2. Expand the tree for the mining structure, to show all the columns.
3. Right-click the column that you want to delete, and then select **Delete**.
4. In the **Delete Objects** dialog box, click **OK**.

See Also

Mining Structure Tasks and How-tos

Add a Nested Table to a Mining Structure

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use Data Mining Designer to add a nested table to a mining structure after it has been created by the Data Mining Wizard.

To add a nested table to a mining structure

1. Select the **Mining Structure** tab in Data Mining Designer.
2. Right-click the mining structure to which you want to add a table column.
3. Select **Add a Nested Table**.

   The **Select a Nested Table Key Column** dialog box opens.
4. Under **Nested table**, select the table that you want to nest in the mining structure.
5. Under **Source column**, select the key column for the nested table.
6. Click **OK**.

   A new table column that contains the key column is added to the mining structure. For information about how to add additional columns, see Add Columns to a Mining Structure.

See Also

Mining Structure Tasks and How-tos

Change the Properties of a Mining Structure

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

There are two kinds of properties on a mining structure, both of which can be modified:

- Properties of the mining structure that affect the entire structure
- Properties on individual columns in the structure

Note that some properties are dependent on other property settings. For example, you cannot set properties that control binning behavior (such as DiscretizationMethod or DiscretizationBucketCount) until you have set the data type of the column to **Discretized**.

For more information about mining structure properties, see Mining Structure Columns.

To change the properties of a mining structure

1. On the **Mining Structure** tab in Data Mining Designer, right-click either the mining structure or a column in the mining structure, and then select **Properties**.

   The **Properties** window opens on the right side of the screen, if it was not already visible.

2. In the **Properties** window, select the value that corresponds to the property that you want to change, and then enter the new value.

   The new value will take effect when you select a different element in the designer.

See Also

Mining Structure Tasks and How-tos

Edit the Data Source View used for a Mining Structure

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can open a data source view from within Data Mining Designer so that you can modify it.

To access Data Source View Designer from the Mining Structure tab in Data Mining Designer

- Right-click in the **Data Source View** pane and select **Edit Data Source View**.

  Data Source View Designer opens in a new tab in SQL Server Data Tools (SSDT).

See Also

Mining Structure Tasks and How-tos

Process a Mining Structure

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Before you can browse or work with the mining models that are associated with a mining structure, you have to deploy the Analysis Services project and process the mining structure and mining models. Also, if you make a change to the mining structure or mining models, you will be prompted to redeploy and process them. Processing the structure in the **Mining Structure** tab of Data Mining Designer in SQL Server Data Tools (SSDT) processes all the associated models.

You can process a mining structure by using these tools:

- SQL Server Data Tools (SSDT)
- SQL Server Management Studio
- XMLA: Process command

For information about how to process individual models, see Process a Mining Model.

To process a mining structure and all associated mining models using SQL Server Data Tools

1. Select **Process Mining Structure and All Models** from the **Mining Model** menu item in SQL Server Data Tools.

   If you made changes to the structure, you will be prompted to redeploy the structure before processing the models. Click **Yes**.

2. Click **Run** in the **Processing Mining Structure - <structure>** dialog box.

   The **Process Progress** dialog box opens to display the details of model processing.

3. Click **Close** in the **Process Progress** dialog box after the models have completed processing.

4. Click **Close** in the **Processing Mining Structure - <structure>** dialog box.

See Also

Mining Structure Tasks and How-tos

Mining Models (Analysis Services - Data Mining)

11/28/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A *mining model* is created by applying an algorithm to data, but it is more than an algorithm or a metadata container: it is a set of data, statistics, and patterns that can be applied to new data to generate predictions and make inferences about relationships.

This section explains what a data mining model is and what it can be used for: the basic architecture of models and structures, the properties of mining models, and ways to create and work with mining models.

Mining Model Architecture

Defining Data Mining Models

Mining Model Properties

Mining Model Columns

Processing Mining Models

Viewing and Querying Mining Models

Mining Model Architecture

A data mining model gets data from a mining structure and then analyzes that data by using a data mining algorithm. The mining structure and mining model are separate objects. The mining structure stores information that defines the data source. A mining model stores information derived from statistical processing of the data, such as the patterns found as a result of analysis.

A mining model is empty until the data provided by the mining structure has been processed and analyzed. After a mining model has been processed, it contains metadata, results, and bindings back to the mining structure.

The metadata specifies the name of the model and the server where it is stored, as well as a definition of the model, including the columns from the mining structure that were used in building the model, the definitions of any filters that were applied when processing the model, and the algorithm that was used to analyze the data. All these choices-the data columns and their data types, filters, and algorithm-have a powerful influence on the results of analysis.

For example, you can use the same data to create multiple models, using perhaps a clustering algorithm, decision tree algorithm, and Naïve Bayes algorithm. Each model type creates different set of patterns, itemsets, rules, or formulas, which you can use for making predictions. Generally each algorithm analyses the data in a different way, so the *content* of the resulting model is also organized in different structures. In one type of model, the data

and patterns might be grouped in *clusters*; in another type of model, data might be organized into trees, branches, and the rules that divide and define them.

The model is also affected by the data that you train it on: even models trained on the same mining structure can yield different results if you filter the data differently or use different seeds during analysis. However, the actual data is not stored in the model-only summary statistics are stored, with the actual data residing in the mining structure. If you have created filters on the data when you trained the model, the filter definitions are saved with the model object as well.

The model does contain a set of bindings, which point back to the data cached in the mining structure. If the data has been cached in the structure and has not been cleared after processing, these bindings enable you to drill through from the results to the cases that support the results. However, the actual data is stored in the structure cache, not in the model.

Mining Model Architecture

Defining Data Mining Models

You create a data mining model by following these general steps:

- Create the underlying mining structure and include the columns of data that might be needed.
- Select the algorithm that is best suited to the analytical task.
- Choose the columns from the structure to use in the model, and specify how they should be used-which column contains the outcome you want to predict, which columns are for input only, and so forth.
- Optionally, set parameters to fine-tune the processing by the algorithm.
- Populate the model with data by *processing* the structure and model.

Analysis Services provides the following tools to help you manage your mining models:

- The Data Mining Wizard helps you create a structure and related mining model. This is the easiest method to use. The wizard automatically creates the required mining structure and helps you with the configuration of the important settings.
- A DMX CREATE MODEL statement can be used to define a model. The required structure is automatically created as part of the process; therefore, you cannot reuse an existing structure with this method. Use this method if you already know exactly which model you want to create, or if you want to script models.
- A DMX ALTER STRUCTURE ADD MODEL statement can be used to add a new mining model to an existing structure. Use this method if you want to experiment with different models that are based on the same data set.

You can also create mining models programmatically, by using AMO or XML/A, or by using other clients such as the Data Mining Client for Excel. For more information, see the following topics:

Mining Model Architecture

Mining Model Properties

Each mining model has properties that define the model and its metadata. These include the name, description, the date the model was last processed, permissions on the model, and any filters on the data that is used for training.

Each mining model also has properties that are derived from the mining structure, and that describe the columns of data used by the model. If any column used by the model is a nested table, the column can also have a separate filter applied.

In addition, each mining model contains two special properties: Algorithm and Usage.

- **Algorithm property** Specifies the algorithm that is used to create the model. The algorithms that are available depend on the provider that you are using. For a list of the algorithms that are included with SQL Server Analysis Services, see Data Mining Algorithms (Analysis Services - Data Mining). The **Algorithm** property applies to the mining model and can be set only one time for each model. You can change the algorithm later but some columns in the mining model might become invalid if they are not supported by the algorithm that you choose. You must always reprocess the model following a change to this property.

- **Usage property** Defines how each column is used by the model. You can define the column usage as **Input**, **Predict**, **Predict Only**, or **Key**. The **Usage** property applies to individual mining model columns and must be set individually for every column that is included in a model. If the structure contains a column that you do not use in the model, the usage is set to **Ignore**. Examples of data that you might include in the mining structure but not use in analysis might be customer names or e-mail addresses. This way you can query them later without having to include them during the analysis phase.

You can change the value of mining model properties after you create a mining model. However, any change, even to the name of the mining model, requires that you reprocess the model. After you reprocess the model, you might see different results.

Mining Model Architecture

Mining Model Columns

The mining model contains columns of data that are obtained from the columns defined in the mining structure. You can choose which columns from the mining structure to use in the model, and you can create copies of the mining structure columns and then rename them or change their usage. As part of the model building process, you must also define the usage of the column by the model. That includes such information as whether the column is a key, whether it is used for prediction, or whether it can be ignored by the algorithm.

While you are building a model, rather than automatically adding every column of data that is available, it is recommended that you review the data in the structure carefully and include in the model only those columns that make sense for analysis. For example, you should avoid including multiple columns that repeat the same data, and you should avoid using columns that have mostly unique values. If you think a column should not be used, you do not need to delete it from the mining structure or mining model; instead, you can just set a flag on the column that specifies that it should be ignored when building the model. This means that the column will remain in the mining structure, but will not be used in the mining model. If you have enabled drillthrough from the model to the mining structure, you can retrieve the information from the column later.

Depending on which algorithm you choose, some columns in the mining structure might be incompatible with certain model types, or might give you poor results. For example, if your data contains continuous numeric data, such as an Income column, and your model requires discrete values, you might need to convert the data to discrete ranges or remove it from the model. In some cases the algorithm will automatically convert or bin the data for you, but the results might not always be what you want or expect. Consider making additional copies of the column and trying out different models. You can also set flags on the individual columns to indicate where special processing is required. For example, if your data contains nulls, you can use a modeling flag to control handling. If you want a particular column to be considered as a regressor in a model you can do that with a modeling flag.

After you have created the model, you can make changes such as adding or removing columns, or changing the name of the model. However, any change, even only to the model metadata, requires that you reprocess the model.

Mining Model Architecture

Processing Mining Models

A data mining model is an empty object until it is processed. When you process a model, the data that is cached by the structure is passed through a filter, if one has been defined in the model, and is analyzed by the algorithm. The algorithm computes a set of summary statistics that describes the data, identifies the rules and patterns within the data, and then uses these rules and patterns to populate the model.

After it has been processed, the mining model contains a wealth of information about the data and the patterns found through analysis, including statistics, rules, and regression formulas. You can use the custom viewers to browse this information, or you can create data mining queries to retrieve this information and use it for analysis and presentation.

Mining Model Architecture

Viewing and Querying Mining Models

After you have processed a model, you can explore it by using the custom viewers that are provided in SQL Server Data Tools (SSDT) and SQL Server Management Studio. For

You can also create queries against the mining model either to make predictions, or to retrieve model metadata or the patterns created by the model. You create queries by using Data Mining Extensions (DMX).

Related Content

TOPICS	LINKS
Learn how to build mining structures that can support multiple mining models. Learn about the usage of columns in models.	Mining Structure Columns Mining Model Columns Content Types (Data Mining)
Learn about different algorithms, and how the choice of algorithm affects the model content.	Mining Model Content (Analysis Services - Data Mining) Data Mining Algorithms (Analysis Services - Data Mining)
Learn now you can set properties on the model that affects its composition and behavior.	Mining Model Properties Modeling Flags (Data Mining)
Learn about the programmable interfaces for data mining.	Developing with Analysis Management Objects (AMO) Data Mining Extensions (DMX) Reference
Learn how to use the custom data mining viewers in Analysis Services.	Data Mining Model Viewers
View examples of the different types of queries that you can use against data mining models.	Data Mining Queries

Related Tasks

Use the following links to get more specific information about working with data mining models

TASK	LINK
Add and delete mining models	Add a Mining Model to an Existing Mining Structure Delete a Mining Model from a Mining Structure
Work with mining model columns	Exclude a Column from a Mining Model Create an Alias for a Model Column Change the Discretization of a Column in a Mining Model Specify a Column to Use as Regressor in a Model
Alter model properties	Change the Properties of a Mining Model Apply a Filter to a Mining Model Delete a Filter from a Mining Model Enable Drillthrough for a Mining Model View or Change Algorithm Parameters
Copy. move, or manage models	Make a Copy of a Mining Model Copy a View of a Mining Model EXPORT (DMX) IMPORT (DMX)
Populate models with data, or update data in a model	Process a Mining Model
Work with OLAP models	Create a Data Mining Dimension

See Also

Database Objects (Analysis Services - Multidimensional Data)

Mining Model Columns

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A data mining model applies a mining model algorithm to the data that is represented by a mining structure. Like the mining structure, the mining model contains columns. A mining model is contained within the mining structure, and inherits all the values of the properties that are defined by the mining structure. The model can use all the columns that the mining structure contains or a subset of the columns.

You can define two additional pieces of information on a mining model column: usage, and modeling flags.

- **Usage** is a property that defines how the model uses the column. Columns can be used as input columns, key columns, or predictable columns.
- **Modeling flags** provide the algorithm with additional information about the data that is defined in the case table, so that the algorithm can build a more accurate model. You can define modeling flags programmatically by using the Data Mining Extensions (DMX) language, or in **Data Mining Designer** in SQL Server Data Tools (SSDT).

The following list describes the modeling flags that you can define on a mining model column.

MODEL_EXISTENCE_ONLY
Indicates that the presence of the attribute is more important than the values that are in the attribute column. For example, consider a case table that contains a list of order items that are associated with a particular customer. The table data includes the product type, ID, and cost of each item. For modeling purposes, the fact that the customer purchased a particular order item may be more important than the cost of the order item itself. In this case, the cost column should be marked as **MODEL_EXISTENCE_ONLY**.

REGRESSOR
Indicates that the algorithm can use the specified column in the regression formula of regression algorithms. This flag is supported by the Microsoft Decision Trees and Microsoft Time Series algorithms.

For more information about setting the usage property and defining modeling flags programmatically with DMX, see CREATE MINING MODEL (DMX). For more information about setting the usage property and defining modeling flags in SQL Server Data Tools (SSDT), see Moving Data Mining Objects.

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Mining Structures (Analysis Services - Data Mining)
Change the Properties of a Mining Model
Exclude a Column from a Mining Model
Mining Structure Columns

Content Types (Data Mining)

5/16/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, you can define the both the physical data type for a column in a mining structure, and a logical content type for the column when used in a model,

The *data type* determines how algorithms process the data in those columns when you create mining models. Defining the data type of a column gives the algorithm information about the type of data in the columns, and how to process the data. Each data type in Analysis Services supports one or more content types for data mining.

The *content type* describes the behavior of the content that the column contains. For example, if the content in a column repeats in a specific interval, such as days of the week, you can specify the content type of that column as cyclical.

Some algorithms require specific data types and specific content types to be able to function correctly. For example, the Microsoft Naive Bayes algorithm cannot use continuous columns as input, and cannot predict continuous values. Some content types, such as Key Sequence, are used only by a specific algorithm. For a list of the algorithms and the content types that each supports, see Data Mining Algorithms (Analysis Services - Data Mining).

The following list describes the content types that are used in data mining, and identifies the data types that support each type.

Discrete

Discrete means that the column contains a finite number of values with no continuum between values. For example, a gender column is a typical discrete attribute column, in that the data represents a specific number of categories.

The values in a discrete attribute column cannot imply ordering, even if the values are numeric. Moreover, even if the values used for the discrete column are numeric, fractional values cannot be calculated. Telephone area codes are a good example of discrete data that is numeric.

The **Discrete** content type is supported by all data mining data types.

Continuous

Continuous means that the column contains values that represent numeric data on a scale that allows interim values. Unlike a discrete column, which represents finite, countable data, a continuous column represents scalable measurements, and it is possible for the data to contain an infinite number of fractional values. A column of temperatures is an example of a continuous attribute column.

When a column contains continuous numeric data, and you know how the data should be distributed, you can potentially improve the accuracy of the analysis by specifying the expected distribution of values. You specify the column distribution at the level of the mining structure. Therefore, the setting applies to all models that are based on the structure, For more information, see Column Distributions (Data Mining).

The **Continuous** content type is supported by the following data types: **Date**, **Double**, and **Long**.

Discretized

Discretization is the process of putting values of a continuous set of data into buckets so that there are a limited number of possible values. You can discretize only numeric data.

Thus, the *discretized* content type indicates that the column contains values that represent groups, or buckets, of values that are derived from a continuous column. The buckets are treated as ordered and discrete values.

You can discretize your data manually, to ensure that you get the buckets you want, or you can use the discretization methods provided in SQL Server Analysis Services. Some algorithms perform discretization automatically. For more information, see Change the Discretization of a Column in a Mining Model.

The **Discretized** content type is supported by the following data types: **Date**, **Double**, **Long**, and **Text**.

Key

The *key* content type means that the column uniquely identifies a row. In a case table, typically the key column is a numeric or text identifier. You set the content type to **key** to indicate that the column should not be used for analysis, only for tracking records.

Nested tables also have keys, but the usage of the nested table key is a little different. You set the content type to **key** in a nested table if the column is the attribute that you want to analyze. The values in the nested table key must be unique for each case but there can be duplicates across the entire set of cases.

For example, if you are analyzing the products that customers purchase, you would set content type to key for the **CustomerID** column in the case table, and set content type to key again for the **PurchasedProducts** column in the nested table.

> **NOTE**
>
> Nested tables are available only if you use data from an external data source that has been defined as an Analysis services data source view.

This content type is supported by the following data types: **Date**, **Double**, **Long**, and **Text**.

Key Sequence

The *key sequence* content type can only be used in sequence clustering models. When you set content type to **key sequence**, it indicates that the column contains values that represent a sequence of events. The values are ordered, but do not have to be an equal distance apart.

This content type is supported by the following data types: **Double**, **Long**, **Text**, and **Date**.

Key Time

The *key time* content type can only be used in time series models. When you set content type to **key time**, it indicates that the values are ordered and represent a time scale.

This content type is supported by the following data types: **Double**, **Long**, and **Date**.

Table

The *table* content type indicates that the column contains another data table, with one or more columns and one or more rows. For any particular row in the case table, this column can contain multiple values, all related to the parent case record. For example, if the main case table contains a listing of customers, you could have several columns that contain nested tables, such as a **ProductsPurchased** column, where the nested table lists

products bought by this customer in the past, and a **Hobbies** column that lists the interests of the customer.

The data type of this column is always **Table**.

Cyclical

The *cyclical* content type means that the column contains values that represent a cyclical ordered set. For example, the numbered days of the week is a cyclical ordered set, because day number one follows day number seven.

Cyclical columns are considered both ordered and discrete in terms of content type.

This content type is supported by all the data mining data types in Analysis Services. However, most algorithms treat cyclical values as discrete values and do not perform special processing.

Ordered

The *Ordered* content type also indicates that the column contains values that define a sequence or order. However, in this content type the values used for ordering do not imply any distance or magnitude relationship between values in the set. For example, if an ordered attribute column contains information about skill levels in rank order from one to five, there is no implied information in the distance between skill levels; a skill level of five is not necessarily five times better than a skill level of one.

Ordered attribute columns are considered to be discrete in terms of content type.

This content type is supported by all the data mining data types in Analysis Services. However, however, most algorithms treat ordered values as discrete values and do not perform special processing.

Classified

In addition to the preceding content types that are in common use with all models, for some data types you can use classified columns to define content types. For more information about classified columns, see Classified Columns (Data Mining).

See Also

Content Types (DMX)
Data Types (Data Mining)
Data Types (DMX)
Change the Properties of a Mining Structure
Mining Structure Columns

Mining Model Properties

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Mining models have the following kinds of properties:

- Properties that are inherited from the mining structure that define the data type and content type of the data used by the model;
- Properties that are related to the algorithm used to create the mining model, including any customer parameters;
- Properties that define a filter on the model used to train the model.

The properties of a mining model are initially defined when you create the model; however, you can alter most properties later, including the algorithm parameters, filters, and column usage properties. You change properties by using the **Mining Models** tab of Data Mining Designer, or by using AMO or XMLA.

Whenever you change any property of a model, you must reprocess the model for the changes to be reflected in the model. Reprocessing is required even if the change only involves metadata, such as adding a column alias or description.

Properties of Models

The following table describes the properties that are specific to mining models. Additionally, there are properties that you can set on individual columns in the mining

PROPERTY	DESCRIPTION
Algorithm	Sets the algorithm type for the mining model.
AlgorithmParameters	Sets values for algorithm parameters that are available for each algorithm type.
Filter	Sets a filter that is applied to the data that is used for training and testing the mining model. The filter definition is stored with the model and can be used optionally when you create prediction queries, or when you test the accuracy of the model. The model filter is not optional when training the model.
Name	Sets the name of the mining model.
AllowDrillThrough	Specifies whether drill through is enabled on the mining model.

Properties of Model Columns

You can set the following data mining-specific properties for each column in a mining model. You can set these properties to a different value for each mining model in a mining structure.

PROPERTY	DESCRIPTION
Description	Describes the purpose of the mining column.
Name	Sets the name of the mining model column. You can type a new name, to provide an alias for the mining model column.
ModelingFlags	Sets any algorithm-specific flags for the column.
SourceColumnID	Indicates the name of the mining structure column on which the model column is based. This property is read-only.
Usage	Sets how the column will be used by the mining model.

See Also

Mining Model Columns
Mining Structures (Analysis Services - Data Mining)
Mining Model Tasks and How-tos
Change the Properties of a Mining Model
Data Mining Tools
Create a Relational Mining Structure
Create an Alias for a Model Column

Missing Values (Analysis Services - Data Mining)

5/16/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Handling *missing values* correctly is an important part of effective modeling. This section explains what missing values are, and describes the features provided in Analysis Services to work with missing values when building data mining structures and mining models.

Definition of Missing Values in Data Mining

A missing value can signify a number of different things. Perhaps the field was not applicable, the event did not happen, or the data was not available. It could be that the person who entered the data did not know the right value, or did not care if a field was not filled in.

However, there are many data mining scenarios in which missing values provide important information. The meaning of the missing values depends largely on context. For example, a missing value for the date in a list of invoices has a meaning substantially different from the lack of a date in column that indicates an employee hire date. Generally, Analysis Services treats missing values as informative and adjusts the probabilities to incorporate the missing values into its calculations. By doing so, you can ensure that models are balanced and do not weight existing cases too heavily.

Therefore, Analysis Services provides two distinctly different mechanisms for managing and calculating missing values. The first method controls the handling of nulls at the level of the mining structure. The second method differs in implementation for each algorithm, but generally defines how missing values are processed and counted in models that permit null values.

Specifying Handling of Nulls

In your data source, missing values might be represented in many ways: as nulls, as empty cells in a spreadsheet, as the value N/A or some other code, or as an artificial value such as 9999. However, for purposes of data mining, only nulls are considered missing values. If your data contains placeholder values instead of nulls, they can affect the results of the model, so you should replace them with nulls or infer correct values if possible. There are a variety of tools that you can use to infer and fill in appropriate values, such as the Lookup transformation or the Data Profiler task in SQL Server Integration Services, or the Fill By Example tool provided in the Data Mining Add-Ins for Excel.

If the task that you are modeling specifies that a column must never have missing values, you should apply the **NOT_NULL** modeling flag to the column when you define the mining structure. This flag indicates that processing should fail if a case does not have an appropriate value. If this error occurs when processing a model, you can log the error and take steps to correct the data that is supplied to the model.

Calculation of the Missing State

To the data mining algorithm, missing values are informative. In case tables, **Missing** is a valid state like any other. Moreover, a data mining model can use other values to predict whether a value is missing. In other words, the fact that a value is missing is not an error.

When you create a mining model, a **Missing** state is automatically added to the model for all discrete columns. For example, if the input column [Gender] contains two possible values, Male and Female, a third value is automatically added to represent the **Missing** value, and the histogram that shows the distribution of all values

for the column always includes a count of the cases with **Missing** values. If the Gender column is not missing any values, the histogram shows that the Missing state is found in 0 cases.

The rationale for including the **Missing** state by default becomes clear when you consider that your data might not have examples of all possible values, and you would not want the model to exclude the possibility just because there was no example in the data. For example, if sales data for a store showed that all customers who purchased a certain product happened to be women, you would not want to create a model that predicts that only women could purchase the product. Instead, Analysis Services adds a placeholder for the extra unknown value, called **Missing**, as a way of accommodating possible other states.

For example, the following table shows the distribution of values for the (All) node in the decision tree model created for the Bike Buyer tutorial. In the example scenario, the [Bike Buyer] column is the predictable attribute, where 1 indicates "Yes" and 0 indicates "No".

VALUE	CASES
0	9296
1	9098
Missing	0

This distribution shows that about half of the customers have purchased a bike, and half have not. This particular data set is very clean; therefore, every case has a value in the [Bike Buyer] column, and the count of **Missing** values is 0. However, if any case had a null in the [Bike Buyer] field, Analysis Services would count that row as a case with a **Missing** value.

If the input is a continuous column, the model tabulates two possible states for the attribute: **Existing** and **Missing**. In other words, either the column contains a value of some numeric data type, or it contains no value. For cases that have a value, the model calculates mean, standard deviation, and other meaningful statistics. For cases that have no value, the model provides a count of the **Missing** vales and adjusts predictions accordingly. The method for adjusting the prediction differs depending on the algorithm and is described in the following section.

> **NOTE**
> For attributes in a nested table, missing values are not informative. For example, if a customer has not purchased a product, the nested **Products** table would not have a row corresponding to that product, and the mining model would not create an attribute for the missing product. However, if you are interested in customers who have not purchased certain products, you can create a model that is filtered on the non-existence of the products in the nested table, by using a NOT EXISTS statement in the model filter. For more information, see Apply a Filter to a Mining Model.

Adjusting Probability for Missing States

In addition to counting values, Analysis Services calculates the probability of any value across the data set. The same is true for the **Missing** value. For example, the following table shows the probabilities for the cases in the previous example:

VALUE	CASES	PROBABILITY
0	9296	50.55%
1	9098	49.42%

VALUE	CASES	PROBABILITY
Missing	0	0.03%

It may seem odd that the probability of the **Missing** value is calculated as 0.03%, when the number of cases is 0. In fact, this behavior is by design, and represents an adjustment that lets the model handle unknown values gracefully.

In general, probability is calculated as the favorable cases divided by all possible cases. In this example, the algorithm computes the sum of the cases that meet a particular condition ([Bike Buyer] = 1, or [Bike Buyer] = 0), and divides that number by the total count of rows. However, to account for the **Missing** cases, 1 is added to the number of all possible cases. As a result, the probability for the unknown case is no longer zero, but a very small number, indicating that the state is merely improbable, not impossible.

The addition of the small **Missing** value does not change the outcome of the predictor; however, it enables better modeling in scenarios where the historical data does not include all possible outcomes.

> **NOTE**
> Data mining providers differ in the way they handle missing values. For example, some providers assume that missing data in a nested column is sparse representation, but that missing data in a non-nested column is missing at random.

If you are certain that all outcomes are specified in your data and want to prevent probabilities from being adjusted, you should set the NOT_NULL modeling flag on the column in the mining structure.

> **NOTE**
> Each algorithm, including custom algorithms that you may have obtained from a third-party plug-in, can handle missing values differently.

Special Handling of Missing Values in Decision Tree Models

The Microsoft Decision Trees algorithm calculates probabilities for missing values differently than in other algorithms. Instead of just adding 1 to the total number of cases, the decision trees algorithm adjusts for the **Missing** state by using a slightly different formula.

In a decision tree model, the probability of the **Missing** state is calculated as follows:

StateProbability = (NodePriorProbability)* (StateSupport + 1) / (NodeSupport + TotalStates)

The Decision Trees algorithm provides an additional adjustment that helps the algorithm compensate for the presence of filters on the model, which may result in many states to be excluded during training.

In SQL Server 2017, if a state is present during training but just happens to have zero support in a certain node, the standard adjustment is made. However, if a state is never encountered during training, the algorithm sets the probability to exactly zero. This adjustment applies not only to the **Missing** state, but also to other states that exist in the training data but have zero support as result of model filtering.

This additional adjustment results in the following formula:

StateProbability = 0.0 if that state has 0 support in the training set

ELSE StateProbability = (NodePriorProbability)* (StateSupport + 1) / (NodeSupport + TotalStatesWithNonZeroSupport)

The net effect of this adjustment is to maintain the stability of the tree.

Related Tasks

The following topics provide more information about how to handle missing values.

TASKS	LINKS
Add flags to individual model columns to control handling of missing values	View or Change Modeling Flags (Data Mining)
Set properties on a mining model to control handling of missing values	Change the Properties of a Mining Model
Learn how to specify modeling flags in DMX	Modeling Flags (DMX)
Alter the way that the mining structure handles missing values	Change the Properties of a Mining Structure

See Also

Mining Model Content (Analysis Services - Data Mining)
Modeling Flags (Data Mining)

Feature Selection (Data Mining)

5/16/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Feature selection is an important part of machine learning. Feature selection refers to the process of reducing the inputs for processing and analysis, or of finding the most meaningful inputs. A related term, *feature engineering* (or *feature extraction*), refers to the process of extracting useful information or features from existing data.

Why Do Feature Selection?

Feature selection is critical to building a good model for several reasons. One is that feature selection implies some degree of *cardinality reduction*, to impose a cutoff on the number of attributes that can be considered when building a model. Data almost always contains more information than is needed to build the model, or the wrong kind of information. For example, you might have a dataset with 500 columns that describe the characteristics of customers; however, if the data in some of the columns is very sparse you would gain very little benefit from adding them to the model, and if some of the columns duplicate each other, using both columns could affect the model.

Not only does feature selection improve the quality of the model, it also makes the process of modeling more efficient. If you use unneeded columns while building a model, more CPU and memory are required during the training process, and more storage space is required for the completed model. Even if resources were not an issue, you would still want to perform feature selection and identify the best columns, because unneeded columns can degrade the quality of the model in several ways:

- Noisy or redundant data makes it more difficult to discover meaningful patterns.
- If the data set is high-dimensional, most data mining algorithms require a much larger training data set.

During the process of feature selection, either the analyst or the modeling tool or algorithm actively selects or discards attributes based on their usefulness for analysis. The analyst might perform feature engineering to add features, and remove or modify existing data, while the machine learning algorithm typically scores columns and validates their usefulness in the model.

In short, feature selection helps solve two problems: having too much data that is of little value, or having too little data that is of high value. Your goal in feature selection should be to identify the minimum number of columns from the data source that are significant in building a model.

How Feature Selection Works in SQL Server Data Mining

Feature selection is always performed before the model is trained. With some algorithms, feature selection techniques are "built-in" so that irrelevant columns are excluded and the best features are automatically discovered. Each algorithm has its own set of default techniques for intelligently applying feature reduction. However, you can also manually set parameters to influence feature selection behavior.

During automatic feature selection, a score is calculated for each attribute, and only the attributes that have the best scores are selected for the model. You can also adjust the threshold for the top scores. SQL Server Data Mining provides multiple methods for calculating these scores, and the exact method that is applied in any model depends on these factors:

- The algorithm used in your model
- The data type of the attribute
- Any parameters that you may have set on your model

Feature selection is applied to inputs, predictable attributes, or to states in a column. When scoring for feature selection is complete, only the attributes and states that the algorithm selects are included in the model-building process and can be used for prediction. If you choose a predictable attribute that does not meet the threshold for feature selection the attribute can still be used for prediction, but the predictions will be based solely on the global statistics that exist in the model.

> **NOTE**
>
> Feature selection affects only the columns that are used in the model, and has no effect on storage of the mining structure. The columns that you leave out of the mining model are still available in the structure, and data in the mining structure columns will be cached.

Feature Selection Scores

SQL Server Data Mining supports these popular and well-established methods for scoring attributes. The specific method used in any particular algorithm or data set depends on the data types, and the column usage.

- The *interestingness* score is used to rank and sort attributes in columns that contain nonbinary continuous numeric data.
- *Shannon's entropy* and two *Bayesian* scores are available for columns that contain discrete and discretized data. However, if the model contains any continuous columns, the interestingness score will be used to assess all input columns, to ensure consistency.

Interestingness score

A feature is interesting if it tells you some useful piece of information. However, *interestingness* can be measured in many ways. *Novelty* might be valuable for outlier detection, but the ability to discriminate between closely related items, or *discriminating weight*, might be more interesting for classification.

The measure of interestingness that is used in SQL Server Data Mining is *entropy-based*, meaning that attributes with random distributions have higher entropy and lower information gain; therefore, such attributes are less interesting. The entropy for any particular attribute is compared to the entropy of all other attributes, as follows:

Interestingness(Attribute) = - (m - Entropy(Attribute)) * (m - Entropy(Attribute))

Central entropy, or m, means the entropy of the entire feature set. By subtracting the entropy of the target attribute from the central entropy, you can assess how much information the attribute provides.

This score is used by default whenever the column contains nonbinary continuous numeric data.

Shannon's Entropy

Shannon's entropy measures the uncertainty of a random variable for a particular outcome. For example, the entropy of a coin toss can be represented as a function of the probability of it coming up heads.

Analysis Services uses the following formula to calculate Shannon's entropy:

$H(X) = -\sum P(xi) \log(P(xi))$

This scoring method is available for discrete and discretized attributes.

Bayesian with K2 Prior

SQL Server Data Mining provides two feature selection scores that are based on Bayesian networks. A Bayesian network is a *directed* or *acyclic* graph of states and transitions between states, meaning that some states are always prior to the current state, some states are posterior, and the graph does not repeat or loop. By definition, Bayesian networks allow the use of prior knowledge. However, the question of which prior states to use in calculating probabilities of later states is important for algorithm design, performance, and accuracy.

The K2 algorithm for learning from a Bayesian network was developed by Cooper and Herskovits and is often used in data mining. It is scalable and can analyze multiple variables, but requires ordering on variables used as input. For more information, see Learning Bayesian Networks by Chickering, Geiger, and Heckerman.

This scoring method is available for discrete and discretized attributes.

Bayesian Dirichlet Equivalent with Uniform Prior

The Bayesian Dirichlet Equivalent (BDE) score also uses Bayesian analysis to evaluate a network given a dataset. The BDE scoring method was developed by Heckerman and is based on the BD metric developed by Cooper and Herskovits. The Dirichlet distribution is a multinomial distribution that describes the conditional probability of each variable in the network, and has many properties that are useful for learning.

The Bayesian Dirichlet Equivalent with Uniform Prior (BDEU) method assumes a special case of the Dirichlet distribution, in which a mathematical constant is used to create a fixed or uniform distribution of prior states. The BDE score also assumes likelihood equivalence, which means that the data cannot be expected to discriminate equivalent structures. In other words, if the score for If A Then B is the same as the score for If B Then A, the structures cannot be distinguished based on the data, and causation cannot be inferred.

For more information about Bayesian networks and the implementation of these scoring methods, see Learning Bayesian Networks.

Feature Selection Methods per Algorithm

The following table lists the algorithms that support feature selection, the feature selection methods used by the algorithm, and the parameters that you set to control feature selection behavior:

ALGORITHM	METHOD OF ANALYSIS	COMMENTS
Naive Bayes	Shannon's Entropy Bayesian with K2 Prior Bayesian Dirichlet with uniform prior (default)	The Microsoft Naïve Bayes algorithm accepts only discrete or discretized attributes; therefore, it cannot use the interestingness score. For more information about this algorithm, see Microsoft Naive Bayes Algorithm Technical Reference.
Decision trees	Interestingness score Shannon's Entropy Bayesian with K2 Prior Bayesian Dirichlet with uniform prior (default)	If any columns contain non-binary continuous values, the interestingness score is used for all columns, to ensure consistency. Otherwise, the default feature selection method is used, or the method that you specified when you created the model. For more information about this algorithm, see Microsoft Decision Trees Algorithm Technical Reference.

ALGORITHM	METHOD OF ANALYSIS	COMMENTS
Neural network	Interestingness score Shannon's Entropy Bayesian with K2 Prior Bayesian Dirichlet with uniform prior (default)	The Microsoft Neural Networks algorithm can use both Bayesian and entropy-based methods, as long as the data contains continuous columns. For more information about this algorithm, see Microsoft Neural Network Algorithm Technical Reference.
Logistic regression	Interestingness score Shannon's Entropy Bayesian with K2 Prior Bayesian Dirichlet with uniform prior (default)	Although the Microsoft Logistic Regression algorithm is based on the Microsoft Neural Network algorithm, you cannot customize logistic regression models to control feature selection behavior; therefore, feature selection always default to the method that is most appropriate for the attribute. If all attributes are discrete or discretized, the default is BDEU. For more information about this algorithm, see Microsoft Logistic Regression Algorithm Technical Reference.
Clustering	Interestingness score	The Microsoft Clustering algorithm can use discrete or discretized data. However, because the score of each attribute is calculated as a distance and is represented as a continuous number, the interestingness score must be used. For more information about this algorithm, see Microsoft Clustering Algorithm Technical Reference.
Linear regression	Interestingness score	The Microsoft Linear Regression algorithm can only use the interestingness score, because it only supports continuous columns. For more information about this algorithm, see Microsoft Linear Regression Algorithm Technical Reference.

ALGORITHM	METHOD OF ANALYSIS	COMMENTS
Association rules Sequence clustering	Not used	Feature selection is not invoked with these algorithms. However, you can control the behavior of the algorithm and reduce the size of input data if necessary by setting the value of the parameters MINIMUM_SUPPORT and MINIMUM_PROBABILIITY. For more information, see Microsoft Association Algorithm Technical Reference and Microsoft Sequence Clustering Algorithm Technical Reference.
Time series	Not used	Feature selection does not apply to time series models. For more information about this algorithm, see Microsoft Time Series Algorithm Technical Reference.

Feature Selection Parameters

In algorithms that support feature selection, you can control when feature selection is turned on by using the following parameters. Each algorithm has a default value for the number of inputs that are allowed, but you can override this default and specify the number of attributes. This section lists the parameters that are provided for managing feature selection.

MAXIMUM_INPUT_ATTRIBUTES

If a model contains more columns than the number that is specified in the *MAXIMUM_INPUT_ATTRIBUTES* parameter, the algorithm ignores any columns that it calculates to be uninteresting.

MAXIMUM_OUTPUT_ATTRIBUTES

Similarly, if a model contains more predictable columns than the number that is specified in the *MAXIMUM_OUTPUT_ATTRIBUTES* parameter, the algorithm ignores any columns that it calculates to be uninteresting.

MAXIMUM_STATES

If a model contains more cases than are specified in the *MAXIMUM_STATES* parameter, the least popular states are grouped together and treated as missing. If any one of these parameters is set to 0, feature selection is turned off, affecting processing time and performance.

In addition to these methods for feature selection, you can improve the ability of the algorithm to identify or promote meaningful attributes by setting *modeling flags* on the model or by setting *distribution flags* on the structure. For more information about these concepts, see Modeling Flags (Data Mining) and Column Distributions (Data Mining).

See Also

Customize Mining Models and Structure

Modeling Flags (Data Mining)

11/28/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can use modeling flags in SQL Server Analysis Services to provide additional information to a data mining algorithm about the data that is defined in a case table. The algorithm can use this information to build a more accurate data mining model.

Some modeling flags are defined at the level of the mining structure, whereas others are defined at the level of the mining model column. For example, the **NOT NULL** modeling flag is used with mining structure columns. You can define additional modeling flags on the mining model columns, depending on the algorithm you use to create the model.

> **NOTE**
> Third-party plug-ins might have other modeling flags, in addition to those pre-defined by Analysis Services.

List of Modeling Flags

The following list describes the modeling flags that are supported in Analysis Services. For information about modeling flags that are supported by specific algorithms, see the technical reference topic for the algorithm that was used to create the model.

NOT NULL

Indicates that the values for the attribute column should never contain a null value. An error will result if Analysis Services encounters a null value for this attribute column during the model training process.

MODEL_EXISTENCE_ONLY

Indicates that the column will be treated as having two states: **Missing** and **Existing**. If the value is **NULL**, it is treated as Missing. The MODEL_EXISTENCE_ONLY flag is applied to the predictable attribute and is supported by most algorithms.

In effect, setting the MODEL_EXISTENCE_ONLY flag to **True** changes the representation of the values such that there are only two states: **Missing** and **Existing**. All the non-missing states are combined into a single **Existing** value.

A typical use for this modeling flag would be in attributes for which the **NULL** state has an implicit meaning, and the explicit value of the **NOT NULL** state might not be as important as the fact that the column has any value. For example, a [DateContractSigned] column might be **NULL** if a contract was never signed and **NOT NULL** if the contract was signed. Therefore, if the purpose of the model is to predict whether a contract will be signed, you can use the MODEL_EXISTENCE_ONLY flag to ignore the exact date value in the **NOT NULL** cases and distinguish only between cases where a contract is **Missing** or **Existing**.

> **NOTE**
> Missing is a special state used by the algorithm, and differs from the text value "Missing" in a column. For more information, see Missing Values (Analysis Services - Data Mining).

REGRESSOR

Indicates that the column is a candidate for used as a regressor during processing. This flag is defined on a

mining model column, and can only be applied to columns that have a continuous numeric data type. For more information about the use of this flag, see the section in this topic, Uses of the REGRESSOR Modeling Flag.

Viewing and Changing Modeling Flags

You can view the modeling flags associated with a mining structure column or model column in Data Mining Designer by viewing the properties of the structure or model.

To determine which modeling flags have been applied to the current mining structure, you can create a query against the data mining schema rowset that returns the modeling flags for just the structure columns, by using a query like the following:

```
SELECT COLUMN_NAME, MODELING_FLAG
FROM $system.DMSCHEMA_MINING_STRUCTURE_COLUMNS
WHERE STRUCTURE_NAME = '<structure name>'
```

You can add or change the modeling flags used in a model by using the Data Mining Designer and editing the properties of the associated columns. Such changes require that the structure or model be reprocessed.

You can specify modeling flags in a new mining structure or mining model by using DMX, or by using AMO or XMLA scripts. However, you cannot change the modeling flags used in an existing mining model and structure by using DMX. You must create a new mining model by using the syntax, `ALTER MINING STRUCTURE....ADD MINING MODEL`.

Uses of the REGRESSOR Modeling Flag

When you set the REGRESSOR modeling flag on a column, you are indicating to the algorithm that the column contains potential regressors. The actual regressors that are used in the model are determined by the algorithm. A potential regressor can be discarded if it does not model the predictable attribute.

When you build a model by using the Data Mining wizard, all continuous input columns are flagged as possible regressors. Therefore, even if you do not explicitly set the REGRESSOR flag on a column, the column might be used as a regressor in the model.

You can determine the regressors that were actually used in the processed model by performing a query against the schema rowset for the mining model, as shown in the following example:

```
SELECT COLUMN_NAME, MODELING_FLAG
FROM $system.DMSCHEMA_MINING_COLUMNS
WHERE MODEL_NAME = '<model name>'
```

Note If you modify a mining model and change the content type of a column from continuous to discrete, you must manually change the flag on the mining column and then reprocess the model.

Regressors in Linear Regression Models

Linear regression models are based on the Microsoft Decision Trees algorithm. Even if you do not use the Microsoft Linear Regression algorithm, any decision tree model can contain a tree or nodes that represents a regression on a continuous attribute.

Therefore, in these models you do not need to specify that a continuous column represents a regressor. The Microsoft Decision Trees algorithm will partition the dataset into regions with meaningful patterns even if you do not set the REGRESSOR flag on the column. The difference is that when you set the modeling flag, the algorithm will try to find regression equations of the following form to fit the patterns in the nodes of the tree.

a*C1 + b*C2 + ...

Then, the sum of the residuals is calculated, and if the deviation is too great, a split is forced in the tree.

For example, if you are predicting customer purchasing behavior using **Income** as an attribute, and set the REGRESSOR modeling flag on the column, the algorithm would first try to fit the **Income** values by using a standard regression formula. If the deviation is too great, the regression formula is abandoned and the tree would be split on some other attribute. The decision tree algorithm would then try fit a regressor for income in each of the branches after the split.

You can use the FORCE_REGRESSOR parameter to guarantee that the algorithm will use a particular regressor. This parameter can be used with the Decision Trees algorithm and Linear Regression algorithm.

Related Tasks

Use the following links to learn more about using modeling flags.

TASK	TOPIC
Edit modeling flags by using the Data Mining Designer	View or Change Modeling Flags (Data Mining)
Specify a hint to the algorithm to recommend likely regressors	Specify a Column to Use as Regressor in a Model
See the modeling flags supported by specific algorithms (in the Modeling Flags section for each algorithm reference topic)	Data Mining Algorithms (Analysis Services - Data Mining)
Learn more about mining structure columns and the properties that you can set on them	Mining Structure Columns
Learn about mining model columns and modeling flags that can be applied at the model level	Mining Model Columns
See syntax for working with modeling flags in DMX statements	Modeling Flags (DMX)
Understand missing values and how to work with them	Missing Values (Analysis Services - Data Mining)
Learn about managing models and structures and setting usage properties	Moving Data Mining Objects

Mining Model Content (Analysis Services - Data Mining)

11/28/2018 • 22 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you have designed and processed a mining model using data from the underlying mining structure, the mining model is complete and contains *mining model content*. You can use this content to make predictions or analyze your data.

Mining model content includes metadata about the model, statistics about the data, and patterns discovered by the mining algorithm. Depending on the algorithm that was used, the model content may include regression formulas, the definitions of rules and itemsets, or weights and other statistics.

Regardless of the algorithm that was used, mining model content is presented in a standard structure. You can browse the structure in the Microsoft Generic Content Tree Viewer, provided in SQL Server Data Tools (SSDT), and then switch to one of the custom viewers to see how the information is interpreted and displayed graphically for each model type. You can also create queries against the mining model content by using any client that supports the MINING_MODEL_CONTENT schema rowset. For more information, see Data Mining Query Tasks and How-tos.

This section describes the basic structure of the content provided for all kinds of mining models. It describes the node types that are common to all mining model content, and provides guidance on how to interpret the information.

Structure of Mining Model Content

Nodes in the Model Content

Mining Model Content by Algorithm Type

Tools for Viewing Mining Model Content

Tools for Querying Mining Model Content

Structure of Mining Model Content

The content of each model is presented as a series of *nodes*. A node is an object within a mining model that contains metadata and information about a portion of the model. Nodes are arranged in a hierarchy. The exact arrangement of nodes in the hierarchy, and the meaning of the hierarchy, depends on the algorithm that you used. For example, if you create a decision trees model, the model can contain multiple trees, all connected to the model root; if you create a neural network model, the model may contain one or more networks, plus a statistics node.

The first node in each model is called the *root node*, or the *model parent* node. Every model has a root node (NODE_TYPE = 1). The root node typically contains some metadata about the model, and the number of child nodes, but little additional information about the patterns discovered by the model.

Depending on which algorithm you used to create the model, the root node has a varying number of child nodes. Child nodes have different meanings and contain different content, depending on the algorithm and the depth and complexity of the data.

Nodes in Mining Model Content

In a mining model, a node is a general-purpose container that stores a piece of information about all or part of the model. The structure of each node is always the same, and contains the columns defined by the data mining schema rowset. For more information, see DMSCHEMA_MINING_MODEL_CONTENT Rowset.

Each node includes metadata about the node, including an identifier that is unique within each model, the ID of the parent node, and the number of child nodes that the node has. The metadata identifies the model to which the node belongs, and the database catalog where that particular model is stored. Additional content provided in the node differs depending on the type of algorithm you used to create the model, and might include the following:

- Count of cases in the training data that supports a particular predicted value.
- Statistics, such as mean, standard deviation, or variance.
- Coefficients and formulas.
- Definition of rules and lateral pointers.
- XML fragments that describe a portion of the model.

List of Mining Content Node Types

The following table lists the different types of nodes that are output in data mining models. Because each algorithm processes information differently, each model generates only a few specific kinds of nodes. If you change the algorithm, the type of nodes may change. Also, if you reprocess the model, the content of each node may change.

> **NOTE**
> If you use a different data mining service, or if you create your own plug-in algorithms, additional custom node types may be available.

NODE_TYPE ID	NODE LABEL	NODE CONTENTS
1	Model	Metadata and root content node. Applies to all model types.
2	Tree	Root node of a classification tree. Applies to decision tree models.
3	Interior	Interior split node in a tree. Applies to decision tree models.
4	Distribution	Terminal node of a tree. Applies to decision tree models.
5	Cluster	Cluster detected by the algorithm. Applies to clustering models and sequence clustering models.
6	Unknown	Unknown node type.

NODE_TYPE ID	NODE LABEL	NODE CONTENTS
7	ItemSet	Itemset detected by the algorithm. Applies to association models or sequence clustering models.
8	AssociationRule	Association rule detected by the algorithm. Applies to association models or sequence clustering models.
9	PredictableAttribute	Predictable attribute. Applies to all model types.
10	InputAttribute	Input attribute. Applies to decision trees and Naïve Bayes models.
11	InputAttributeState	Statistics about the states of an input attribute. Applies to decision trees and Naïve Bayes models.
13	Sequence	Top node for a Markov model component of a sequence cluster. Applies to sequence clustering models.
14	Transition	Markov transition matrix. Applies to sequence clustering models.
15	TimeSeries	Non-root node of a time series tree. Applies to time series models only.
16	TsTree	Root node of a time series tree that corresponds to a predictable time series. Applies to time series models, and only if the model was created using the MIXED parameter.
17	NNetSubnetwork	One sub-network. Applies to neural network models.
18	NNetInputLayer	Group that contains the nodes of the input layer. Applies to neural network models.
19	NNetHiddenLayer	Groups that contains the nodes that describe the hidden layer. Applies to neural network models.
21	NNetOutputLayer	Groups that contains the nodes of the output layer. Applies to neural network models.
21	NNetInputNode	Node in the input layer that matches an input attribute with the corresponding states. Applies to neural network models.

NODE_TYPE ID	NODE LABEL	NODE CONTENTS
22	NNetHiddenNode	Node in the hidden layer. Applies to neural network models.
23	NNetOutputNode	Node in the output layer. This node will usually match an output attribute and the corresponding states. Applies to neural network models.
24	NNetMarginalNode	Marginal statistics about the training set. Applies to neural network models.
25	RegressionTreeRoot	Root of a regression tree. Applies to linear regression models and to decision trees models that contains continuous input attributes.
26	NaiveBayesMarginalStatNode	Marginal statistics about the training set. Applies to Naïve Bayes models.
27	ArimaRoot	Root node of an ARIMA model. Applies to only those time series models that use the ARIMA algorithm.
28	ArimaPeriodicStructure	A periodic structure in an ARIMA model. Applies to only those time series models that use the ARIMA algorithm.
29	ArimaAutoRegressive	Autoregressive coefficient for a single term in an ARIMA model. Applies to only those time series models that use the ARIMA algorithm.
30	ArimaMovingAverage	Moving average coefficient for a single term in an ARIMA model. Applies to only those time series models that use the ARIMA algorithm.
1000	CustomBase	Starting point for custom node types. Custom node types must be integers greater in value than this constant. Applies to models created by using custom plug-in algorithms.

Node ID, Name, Caption and Description

The root node of any model always has the unique ID (**NODE_UNIQUE_NAME**) of 0. All node IDs are assigned automatically by Analysis Services and cannot be modified.

The root node for each model also contains some basic metadata about the model. This metadata includes the Analysis Services database where the model is stored (**MODEL_CATALOG**), the schema

(**MODEL_SCHEMA**), and the name of the model (**MODEL_NAME**). However, this information is repeated in all the nodes of the model, so you do not need to query the root node to get this metadata.

In addition to a name used as the unique identifier, each node has a *name* (**NODE_NAME**). This name is automatically created by the algorithm for display purposes and cannot be edited.

> **NOTE**
> The Microsoft Clustering algorithm allows users to assign friendly names to each cluster. However, these friendly names are not persisted on the server, and if you reprocess the model, the algorithm will generate new cluster names.

The *caption* and *description* for each node are automatically generated by the algorithm, and serve as labels to help you understand the content of the node. The text generated for each field depends on the model type. In some cases, the name, caption, and description may contain exactly the same string, but in some models, the description may contain additional information. See the topic about the individual model type for details of the implementation.

> **NOTE**
> Analysis Services server supports the renaming of nodes only if you build models by using a custom plug-in algorithm that implements renaming,. To enable renaming, you must override the methods when you create the plug-in algorithm.

Node Parents, Node Children, and Node Cardinality

The relationship between parent and child nodes in a tree structure is determined by the value of the PARENT_UNIQUE_NAME column. This value is stored in the child node and tells you the ID of the parent node. Some examples follow of how this information might be used:

- A PARENT_UNIQUE_NAME that is NULL means that the node is the top node of the model.
- If the value of PARENT_UNIQUE_NAME is 0, the node must be a direct descendant of the top node in the model. This is because the ID of the root node is always 0.
- You can use functions within a Data Mining Extensions (DMX) query to find descendants or parents of a particular node. For more information about using functions in queries, see Data Mining Queries.

Cardinality refers to the number of items in a set. In the context of a processed mining model, cardinality tells you the number of children in a particular node. For example, if a decision tree model has a node for [Yearly Income], and that node has two child nodes, one for the condition [Yearly Income] = High and one for the condition, [Yearly Income] = Low, the value of CHILDREN_CARDINALITY for the [Yearly Income] node would be 2.

> **NOTE**
> In Analysis Services, only the immediate child nodes are counted when calculating the cardinality of a node. However, if you create a custom plug-in algorithm, you can overload CHILDREN_CARDINALITY to count cardinality differently. This may be useful, for example, if you wanted to count the total number of descendants, not just the immediate children.

Although cardinality is counted in the same way for all models, how you interpret or use the cardinality value differs depending on the model type. For example, in a clustering model, the cardinality of the top node tells you the total number of clusters that were found. In other types of models, cardinality may

always have a set value depending on the node type. For more information about how to interpret cardinality, see the topic about the individual model type.

> **NOTE**
> Some models, such as those created by the Microsoft Neural Network algorithm, additionally contain a special node type that provides descriptive statistics about the training data for the entire model. By definition, these nodes never have child nodes.

Node Distribution

The NODE_DISTRIBUTION column contains a nested table that in many nodes provides important and detailed information about the patterns discovered by the algorithm. The exact statistics provided in this table change depending on the model type, the position of the node in the tree, and whether the predictable attribute is a continuous numeric value or a discrete value; however, they can include the minimum and maximum values of an attribute, weights assigned to values, the number of cases in a node, coefficients used in a regression formula, and statistical measures such as standard deviation and variance. For more information about how to interpret node distribution, see the topic for the specific type of model type that you are working with.

> **NOTE**
> The NODE_DISTRIBUTION table may be empty, depending on the node type. For example, some nodes serve only to organize a collection of child nodes, and it is the child nodes that contain the detailed statistics.

The nested table, NODE_DISTRIBUTION, always contains the following columns. The content of each column varies depending on the model type. For more information about specific model types, see Mining Model Content by Algorithm Type.

ATTRIBUTE_NAME
Content varies by algorithm. Can be the name of a column, such as a predictable attribute, a rule, an itemset, or a piece of information internal to the algorithm, such as part of a formula.

This column can also contain an attribute-value pair.

ATTRIBUTE_VALUE
Value of the attribute named in ATTRIBUTE_NAME.

If the attribute name is a column, then in the most straightforward case, the ATTRIBUTE_VALUE contains one of the discrete values for that column.

Depending on how the algorithm processes values, the ATTRIBUTE_VALUE can also contain a flag that tells you whether a value exists for the attribute (**Existing**), or whether the value is null (**Missing**).

For example, if your model is set up to find customers who have purchased a particular item at least once, the ATTRIBUTE_NAME column might contain the attribute-value pair that defines the item of interest, such as `Model = 'Water bottle'`, and the ATTRIBUTE_VALUE column would contain only the keyword **Existing** or **Missing**.

SUPPORT
Count of the cases that have this attribute-value pair, or that contain this itemset or rule.

In general, for each node, the support value tells you how many cases in the training set are included in the current node. In most model types, support represents an exact count of cases. Support values are useful because you can view the distribution of data within your training cases without having to query the training data. The Analysis Services server also uses these stored values to calculate stored probability

versus prior probability, to determine whether inference is strong or weak.

For example, in a classification tree, the support value indicates the number of cases that have the described combination of attributes.

In a decision tree, the sum of support at each level of a tree sums to the support of its parent node. For example, if a model containing 1200 cases is divided equally by gender, and then subdivided equally by three values for Income-Low, Medium, and High-the child nodes of node (2), which are nodes (4), (5) and (6), always sum to the same number of cases as node (2).

NODE ID AND NODE ATTRIBUTES	SUPPORT COUNT
(1) Model root	1200
(2) Gender = Male	600
(3) Gender = Female	600
(4) Gender = Male and Income = High	200
(5) Gender = Male and Income = Medium	200
(6) Gender = Male and Income = Low	200
(7) Gender = Female and Income = High	200
(8) Gender = Female and Income = Medium	200
(9) Gender = Female and Income = Low	200

For a clustering model, the number for support can be weighted to include the probabilities of belonging to multiple clusters. Multiple cluster membership is the default clustering method. In this scenario, because each case does not necessarily belong to one and only one cluster, support in these models might not add up to 100 percent across all clusters.

PROBABILITY
Indicates the probability for this specific node within the entire model.

Generally, probability represents support for this particular value, divided by the total count of cases within the node (NODE_SUPPORT).

However, probability is adjusted slightly to eliminate bias caused by missing values in the data.

For example, if the current values for [Total Children] are 'One' and 'Two', you want to avoid creating a model that predicts that it is impossible to have no children, or to have three children. To ensure that missing values are improbable, but not impossible, the algorithm always adds 1 to the count of actual values for any attribute.

Example:

Probability of [Total Children = One] = [Count of cases where Total Children = One] + 1/[Count of all cases] + 3

Probability of [Total Children = Two]= [Count of cases where Total Children = Two] +1/[Count of all cases] +3

> **NOTE**
> The adjustment of 3 is calculated by adding 1 to the total number of existing values, n.

After adjustment, the probabilities for all values still add up to 1. The probability for the value with no data (in this example, [Total Children = 'Zero', 'Three', or some other value]), starts at a very low non-zero level, and rises slowly as more cases are added.

VARIANCE
Indicates the variance of the values within the node. By definition, variance is always 0 for discrete values. If the model supports continuous values, variance is computed as σ (sigma), using the denominator n, or the number of cases in the node.

There are two definitions in general use to represent standard deviation (**StDev**). One method for calculating standard deviation takes into account bias, and another method computes standard deviation without using bias. In general, Microsoft data mining algorithms do not use bias when computing standard deviation.

The value that appears in the NODE_DISTRIBUTION table is the actual value for all discrete and discretized attributes, and the mean for continuous values.

VALUE_TYPE
Indicates the data type of the value or an attribute, and the usage of the value. Certain value types apply only to certain model types:

VALUE_TYPE ID	VALUE LABEL	VALUE TYPE NAME
1	Missing	Indicates that the case data did not contain a value for this attribute. The **Missing** state is calculated separately from attributes that have values.
2	Existing	Indicates that the case data contains a value for this attribute.
3	Continuous	Indicates that the value of the attribute is a continuous numeric value and therefore can be represented by a mean, together with variance and standard deviation.
4	Discrete	Indicates a value, either numeric or text, that is treated as discrete. **Note** Discrete values can also be missing; however, they are handled differently when making calculations. For information, see Missing Values (Analysis Services - Data Mining).
5	Discretized	Indicates that the attribute contains numeric values that have been discretized. The value will be a formatted string that describes the discretization buckets.

VALUE_TYPE ID	VALUE LABEL	VALUE TYPE NAME
6	Existing	Indicates that the attribute has continuous numeric values and that values have been supplied in the data, vs. values that are missing or inferred.
7	Coefficient	Indicates a numeric value that represents a coefficient. A coefficient is a value that is applied when calculating the value of the dependent variable. For example, if your model creates a regression formula that predicts income based on age, the coefficient is used in the formula that relates age to income.
8	Score gain	Indicates a numeric value that represents score gain for an attribute.
9	Statistics	Indicates a numeric value that represents a statistic for a regressor.
10	Node unique name	Indicates that the value should not be handled not as numeric or string, but as the unique identifier of another content node in a model. For example, in a neural network model, the IDs provide pointers from nodes in the output layer to nodes in the hidden layer, and from nodes in the hidden layer to nodes in the input layer.
11	Intercept	Indicates a numeric value that represents the intercept in a regression formula.
12	Periodicity	Indicates that the value denotes a periodic structure in a model. Applies only to time series models that contain an ARIMA model. Note: The Microsoft Time Series algorithm automatically detects periodic structures based on the training data. As a result, the periodicities in the final model may include periodicity values that you did not provide as a parameter when creating the model.

VALUE_TYPE ID	VALUE LABEL	VALUE TYPE NAME
13	Autoregressive order	Indicates that the value represents the number of autoregressive series. Applies to time series models that use the ARIMA algorithm.
14	Moving average order	Represents a value that represents the number of moving averages in a series. Applies to time series models that use the ARIMA algorithm.
15	Difference order	Indicates that the value represents a value that indicates how many times the series is differentiated. Applies to time series models that use the ARIMA algorithm.
16	Boolean	Represents a Boolean type.
17	Other	Represents a custom value defined by the algorithm.
18	Prerendered string	Represents a custom value that the algorithm renders as a string. No formatting was applied by the object model.

The value types are derived from the ADMOMD.NET enumeration. For more information, see Microsoft.AnalysisServices.AdomdServer.MiningValueType.

Node Score

The meaning of the node score differs depending on the model type, and can be specific to the node type as well. For information about how NODE_SCORE is calculated for each model and node type, see Mining Model Content by Algorithm Type.

Node Probability and Marginal Probability

The mining model schema rowset includes the columns NODE_PROBABILITY and MARGINAL_PROBABILITY for all model types. These columns contain values only in nodes where a probability value is meaningful. For example, the root node of a model never contains a probability score.

In those nodes that do provide probability scores, the node probability and marginal probabilities represent different calculations.

- **Marginal probability** is the probability of reaching the node from its parent.
- **Node probability** is the probability of reaching the node from the root.
- **Node probability** is always less than or equal to **marginal probability**.

For example, if the population of all customers in a decision tree is split equally by gender (and no values are missing), the probability of the child nodes should be .5. However, suppose that each of the nodes for gender is divided equally by income levels-High, Medium, and Low. In this case the MARGINAL_PROBABILITY score for each child node should always be .33 but the NODE_PROBABILTY

value will be the product of all probabilities leading to that node and thus always less than the MARGINAL_PROBABILITY value.

LEVEL OF NODE/ATTRIBUTE AND VALUE	MARGINAL PROBABILITY	NODE PROBABILITY
Model root All target customers	1	1
Target customers split by gender	.5	.5
Target customers split by gender, and split again three ways by income	.33	.5 * .33 = .165

Node Rule and Marginal Rule

The mining model schema rowset also includes the columns NODE_RULE and MARGINAL_RULE for all model types. These columns contain XML fragments that can be used to serialize a model, or to represent some part of the model structure. These columns may be blank for some nodes, if a value would be meaningless.

Two kinds of XML rules are provided, similar to the two kinds of probability values. The XML fragment in MARGINAL_RULE defines the attribute and value for the current node, whereas the XML fragment in NODE_RULE describes the path to the current node from the model root.

Mining Model Content by Algorithm Type

Each algorithm stores different types of information as part of its content schema. For example, the Microsoft Clustering Algorithm generates many child nodes, each of which represents a possible cluster. Each cluster node contains rules that describe characteristics shared by items in the cluster. In contrast, the Microsoft Linear Regression algorithm does not contain any child nodes; instead, the parent node for the model contains the equation that describes the linear relationship discovered by analysis.

The following table provides links to topics for each type of algorithm.

- **Model content topics:** Explain the meaning of each node type for each algorithm type, and provide guidance about which nodes are of most interest in a particular model type.
- **Querying topics:** Provide examples of queries against a particular model type and guidance on how to interpret the results.

ALGORITHM OR MODEL TYPE	MODEL CONTENT	QUERYING MINING MODELS
Association rules models	Mining Model Content for Association Models (Analysis Services - Data Mining)	Association Model Query Examples
Clustering models	Mining Model Content for Decision Tree Models (Analysis Services - Data Mining)	Clustering Model Query Examples
Decision trees model	Mining Model Content for Decision Tree Models (Analysis Services - Data Mining)	Decision Trees Model Query Examples

ALGORITHM OR MODEL TYPE	MODEL CONTENT	QUERYING MINING MODELS
Linear regression models	Mining Model Content for Linear Regression Models (Analysis Services - Data Mining)	Linear Regression Model Query Examples
Logistic regression models	Mining Model Content for Logistic Regression Models (Analysis Services - Data Mining)	Linear Regression Model Query Examples
Naïve Bayes models	Mining Model Content for Naive Bayes Models (Analysis Services - Data Mining)	Naive Bayes Model Query Examples
Neural network models	Mining Model Content for Neural Network Models (Analysis Services - Data Mining)	Neural Network Model Query Examples
Sequence clustering	Mining Model Content for Sequence Clustering Models (Analysis Services - Data Mining)	Sequence Clustering Model Query Examples
Time series models	Mining Model Content for Time Series Models (Analysis Services - Data Mining)	Time Series Model Query Examples

Tools for Viewing Mining Model Content

When you browse or explore a model in SQL Server Data Tools (SSDT), you can view the information in the **Microsoft Generic Content Tree Viewer**, which is available in both SQL Server Data Tools (SSDT) and SQL Server Management Studio.

The Microsoft Generic Content Viewer displays the columns, rules, properties, attributes, nodes, and other content from the model by using the same information that is available in the content schema rowset of the mining model. The content schema rowset is a generic framework for presenting detailed information about the content of a data mining model. You can view model content in any client that supports hierarchical rowsets. The viewer in SQL Server Data Tools (SSDT) presents this information in an HTML table viewer that represents all models in a consistent format, making it easier to understand the structure of the models that you create. For more information, see Browse a Model Using the Microsoft Generic Content Tree Viewer.

Tools for Querying Mining Model Content

To retrieve mining model content, you must create a query against the data mining model.

The easiest way to create a content query is to execute the following DMX statement in SQL Server Management Studio:

```
SELECT * FROM [<mining model name>].CONTENT
```

For more information, see Data Mining Queries.

You can also query the mining model content by using the data mining schema rowsets. A schema rowset is a standard structure that clients use to discover, browse, and query information about mining structures and models. You can query the schema rowsets by using XMLA, Transact-SQL, or DMX statements.

In SQL Server 2017, you can also access the information in the data mining schema rowsets by opening a connection to the Analysis Services instance and querying the system tables. For more information, see Data Mining Schema Rowsets (SSAs).

See Also

Microsoft Generic Content Tree Viewer (Data Mining)
Data Mining Algorithms (Analysis Services - Data Mining)

Drillthrough on Mining Models

10/10/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Drillthrough means the ability to query either a mining model or a mining structure and get detailed data that is not exposed in the model.

SQL Server 2017 provides two different options for drilling through into case data. You can drill through to the cases that were used to build the data, or you can drill through to the cases in the mining structure.

Drillthrough to Model Cases vs. Drillthrough to Structure

Drilling through to **model cases** is useful for finding additional details about rules, patterns or clusters in a model. For example, you would not use customer contact information for analysis in a clustering model, even if the data was available, by using drillthrough, you can gain access to that information from the model.

In contrast, **drillthrough to structure** data is intended to provide access to information that was not made available in the model. For example, some structure columns might have been excluded from a model because the data type was incompatible or the data was not useful for analysis.

Enabling Drillthrough on a Model

To use drillthrough on a mining model, the following conditions must be met:

- It is possible to configure drillthrough on just the model cases and not on the mining structure, but not vice versa. In other words, drillthrough must be enabled on the mining model to permit drillthrough to the mining structure.
- Drillthrough on both model and structure is disabled by default. If you use the Data Mining Wizard, the option to enable drillthrough to the model cases is on the final page of the wizard.
- You can add the ability to drill through on an existing mining model, but if you do, the model must be reprocessed before you can drill through to the data.
- Drillthrough will not work unless the cache that was created during the training process has been preserved. For more information about the properties that control caching, see Drillthrough on Mining Structures.

Models that Support Drillthrough

If a mining model has been configured to allow drillthrough, and if you have the appropriate permissions, when you browse the model, you can click on a node in the appropriate viewer and retrieve detailed information about the cases in that particular node.

Not all models support drillthrough; it depends on the algorithm that was used to create the model. The following table lists the types of models that do not support drillthrough, or support drillthrough with limitations. If the model type is not listed here, drillthrough is supported.

ALGORITHM NAME	SUPPORT FOR DRILLTHROUGH

ALGORITHM NAME	SUPPORT FOR DRILLTHROUGH
Microsoft Naïve Bayes algorithm	Not supported. These algorithms do not assign cases to specific nodes in the content.
Microsoft Neural Network algorithm	Not supported. These algorithms do not assign cases to specific nodes in the content.
Microsoft Logistic Regression algorithm	Not supported. These algorithms do not assign cases to specific nodes in the content.
Microsoft Linear Regression algorithm	Supported. However, because the model creates a single node, **All**, drillthrough returns all the training cases for the model. If the training set is large, loading the results may take a very long time.
Microsoft Time Series algorithm	Supported. However, you cannot drill through to structure or case data by using the **Mining Model Viewer** in Data Mining Designer. You must create a DMX query instead. Also, you cannot drill through to specific nodes, or write a DMX query to retrieve cases in specific nodes of a time series model. You can retrieve case data from either the model or the structure by using other criteria, such as date or attribute values. If you wish to view details of the ARTXP and ARIMA nodes created by the Microsoft Time Series algorithm, it might be easier to use the Microsoft Generic Content Tree Viewer (Data Mining).

Related Tasks

See the following topics for more information about how to use drillthrough with mining models.

TASKS	LINKS
Use drillthrough in the mining model viewers	Use Drillthrough from the Model Viewers
Retrieve case data for a model by using drillthrough	Drill Through to Case Data from a Mining Model
Enable drillthrough on an existing mining model	Enable Drillthrough for a Mining Model
See examples of drillthrough queries for specific model types.	Data Mining Queries
Enable drillthrough in the Mining Model Wizard	Completing the Wizard (Data Mining Wizard).

See Also

Drillthrough on Mining Structures

Filters for Mining Models (Analysis Services - Data Mining)

11/27/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data-based model filtering helps you create mining models that use subsets of data in a mining structure. Filtering gives you flexibility when you design your mining structures and data sources, because you can create a single mining structure, based on a comprehensive data source view. You can then create filters to use only a part of that data for training and testing a variety of models, instead of building a different structure and related model for each subset of data.

For example, you define the data source view on the Customers table and related tables. Next, you define a single mining structure that includes all the fields you need. Finally, you create a model that is filtered on a particular customer attribute, such as Region. You can then easily make a copy of that model, and change just the filter condition to generate a new model based on a different region.

Some real-life scenarios where you might benefit from this feature include the following:

- Creating separate models for discrete values such as gender, regions, and so forth. For example, a clothing store might use customer demographics to build separate models by gender, even though the sales data comes from a single data source for all customers.
- Experimenting with models by creating and then testing multiple groupings of the same data, such as ages 20-30 vs. ages 20-40 vs. ages 20-25.
- Specifying complex filters on nested table contents, such as requiring that a case be included in the model only if the customer has purchased at least two of a particular item.

This section explains how to build, use, and manage filters on mining models.

Creating Model Filters

You can create and apply filters in the following ways:

- Using the **Mining Models** tab in Data Mining Designer to build conditions with the help of filter editor dialog boxes.
- Typing a filter expression directly into the **Filter** property of the mining model.
- Setting filter conditions on a model programmatically, by using AMO.

Creating Model Filters using Data Mining Designer

You filter a model in Data Mining Designer by changing the **Filter** property of the mining model. You can either type a filter expression directly into the **Properties** pane, or you can open a filter dialog box to build conditions.

There are two filter dialog boxes. The first lets you create conditions that are applied to the case table. If the data source contains multiple tables, first you select a table, and then you select a column and specify operators and conditions that apply to that column. You can link multiple conditions by using **AND**/**OR** operators. The operators that are available for defining values depend on whether the column contains discrete or continuous values. For example, with continuous values, you can use **greater than** and **less than** operators. However, for discrete values, you can only use **= (equal to)**, **!= (not equal to)**, and **is null** operators.

> **NOTE**
> The **LIKE** keyword is not supported. If you want to include multiple discrete attributes, you must create separate conditions and link them by using the **OR** operator.

If the conditions are complex, you can use the second filter dialog box to work with one table at a time. When you close the second filter dialog box, the expression is evaluated and then combined with filter conditions that have been set on other columns in the case table.

Creating Filters on Nested Tables

If the data source view contains nested tables, you can use the second filter dialog box to build conditions on the rows in the nested tables.

For example, if your case table is related to customers, and the nested table shows the products that a customer has purchased, you can create a filter for customers who have purchased particular items by using the following syntax in the nested table filter: `[ProductName]='Water Bottle' OR ProductName='Water Bottle Cage'`.

You can also filter on the existence of a particular value in the nested table by using the **EXISTS** or **NOT EXISTS** keywords and a subquery. This lets you create conditions such as `EXISTS (SELECT * FROM Products WHERE ProductName='Water Bottle')`. The `EXISTS SELECT(<subquery>)` returns **true** if the nested table contains at least one row that includes the value, `Water Bottle`.

You can combine conditions on the case table with conditions on the nested table. For example, the following syntax includes a condition on the case table (`Age > 30`), a subquery on the nested table (`EXISTS (SELECT * FROM Products)`), and multiple conditions on the nested table (`WHERE ProductName='Milk' AND Quantity>2`)).

```
(Age > 30 AND EXISTS (SELECT * FROM Products WHERE ProductName='Milk'  AND Quantity>2) )
```

When you have finished building the filter, the filter text is evaluated by Analysis Services, translated to a DMX expression, and then saved with the model.

For instructions on how to use the filter dialog boxes in SQL Server Data Tools (SSDT), see Apply a Filter to a Mining Model.

Managing Mining Model Filters

Data-based model filtering greatly simplifies the task of managing mining structures and mining models, because you can easily create multiple models that are based on the same structure. You can also quickly make copies of existing mining models and then change only the filter condition. However, filters can lead to some confusion.

Here are some frequently asked questions about how to manage and interpret filters on mining models:

How can I tell whether a filter is being used?

There are multiple ways to determine whether a filter is applied to a model:

- In the designer, click the **Mining Models** tab, open **Properties**, and view the **Filter** property of the mining model.
- The DMV, DMSCHEMA_MINING_MODELS, outputs a column that contains the text of the filter. You can use the following query on a DMV to return the names of models and their filters:

```
SELECT MODEL_NAME, [FILTER]
FROM $SYSTEM.DMSCHEMA_MINING_MODELS
```

- You can get the value of the Filter property of the MiningModel object in AMO, or inspect the Filter element in XMLA.

You might also establish a naming convention for models to reflect the contents of the filter. This can make it easier to tell related models apart.

How can I save a filter?

The filter expression is saved as a script that is stored with the associated mining model or nested table. If you delete the filter text, it can only be restored by manually re-creating the filter expression. Therefore, if you create complex filter expressions, you should create a backup copy of the filter text.

Why can't I see any effects from the filter?

Whenever you change or add a filter expression, you must reprocess the structure and model before you can view the effects of the filter.

Why do I see filtered attributes in prediction query results?

When you apply a filter to a model, it affects only the selection of cases used for training the model. The filter does not affect the attributes known to the model, or change or suppress data that is present in the data source. As a result, queries against the model can return predictions for other types of cases, and dropdown lists of values used by the model might show attribute values excluded by the filter.

For example, suppose you train the [Bike Buyer] model using only cases involving women aged 20-30. You can still run a prediction query that predicts the probability of a man buying a bike, or predict the outcome for a woman aged 30-40. That is because the attributes and values present in the data source define what is theoretically possible, while the cases define the occurrences used for training. However, these queries would return very small probabilities, because the training data does not contain any cases with the target values.

If you need to completely hide or anonymize attribute values in the model, there are various options:

- Filter the incoming data as part of the definition of the data source view, or in the relational data source.
- Mask or encode the attribute value.
- Collapse excluded values into a category as part of the mining structure definition.

Related Resources

For more information about filter syntax, and examples of filter expressions, see Model Filter Syntax and Examples (Analysis Services - Data Mining).

For information about how to use model filters when you are testing a mining model, see Choose an Accuracy Chart Type and Set Chart Options.

See Also

Model Filter Syntax and Examples (Analysis Services - Data Mining)
Testing and Validation (Data Mining)

Model Filter Syntax and Examples (Analysis Services - Data Mining)

11/27/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This section provides detailed information about the syntax for model filters, together with sample expressions.

Filter syntax

Filters on case attributes

Filters on nested table attributes

Filters on multiple nested table attributes

Filters attributes missing in nested table

Filters on multiple nested table values

Filters on nested table attributes and EXISTS

Filter combinations

Filters on dates

Filter Syntax

Filter expressions generally are equivalent to the content of a WHERE clause. You can connect multiple conditions by using the logical operators **AND**, **OR**, and **NOT**.

In nested tables, you can also use the **EXISTS** and **NOT EXISTS** operators. An **EXISTS** condition evaluates to **true** if the subquery returns at least one row. This is useful in cases where you want to restrict the model to cases that contain a particular value in the nested table: for example, customers who have purchased an item at least once.

A **NOT EXISTS** condition evaluates to **true** if the condition specified in the subquery does not exist. An example is when you want to restrict the model to customers who have never purchased a particular item.

The general syntax is as follows:

```
<filter>::=<predicate list>  | ( <predicate list> )
<predicate list>::= <predicate> | [<logical_operator> <predicate list>]
<logical_operator::= AND| OR
<predicate>::= NOT <predicate>|( <predicate> ) <avPredicate> | <nestedTablePredicate> | ( <predicate> )
<avPredicate>::= <columnName> <operator> <scalar> | <columnName> IS [NOT] NULL
<operator>::= = | != | <> | > | >= | < | <=
<nestedTablePredicate>::= EXISTS (<subquery>)
<subquery>::=SELECT * FROM <columnName>[ WHERE  <predicate list> ]
```

filter
Contains one or more predicates, connected by logical operators.

predicate list
One or more valid filter expressions, separated by logical operators.

columnName
The name of a mining structure column.

logical operator
AND, **OR**, **NOT**

avPredicate
Filter expression that can be applied to scalar mining structure column only. An *avPredicate* expression can be used in both model filters or nested table filters.

An expression that uses any of the following operators can only be applied to a continuous column. :

- **<** (less than)
- **>** (greater than)
- **>=** (greater than or equal to)
- **<=** (less than or equal to)

> **NOTE**
> Regardless of the data type, these operators cannot be applied to a column that has the type **Discrete**, **Discretized**, or **Key**.

An expression that uses any of the following operators can be applied to a continuous, discrete, discretized, or key column:

- **=** (equals)
- **!=** (not equal to)
- **IS NULL**

If the argument, *avPredicate*, applies to a discretized column, the value used in the filter can be any value in a specific bucket.

In other words, you do not define the condition as `AgeDisc = '25-35'`, but instead compute and then use a value from that interval.

Example: `AgeDisc = 27` means any value in the same interval as 27, which in this case is 25-35.

nestedTablePredicate
Filter expression that applies to a nested table. Can be used in model filters only.

The sub-query argument of the argument, *nestedTablePredicate*, can only apply to a table mining structure column

subquery
A SELECT statement followed by a valid predicate or list of predicates.

All the predicates must be of the type described in *avPredicates*. Furthermore, the predicates can refer only to columns that are included in the current nested table, identified by the argument, *columnName*.

Limitations on Filter Syntax

The following restrictions apply to filters:

- A filter can contain only simple predicates. These include mathematical operators, scalars, and column names.
- User-defined functions are not supported in the filter syntax.

- Non-Boolean operators, such as the plus or minus signs, are not supported in the filter syntax.

Examples of Filters

The following examples demonstrate the use of filters applied to a mining model. If you create the filter expression by using SQL Server Data Tools (SSDT), in the **Property** window and the **Expression** pane of the filter dialog box, you would see only the string that appears after the WITH FILTER keywords. Here, the definition of the mining structure is included to make it easier to understand the column type and usage.

Example 1: Typical Case-Level Filtering

This example shows a simple filter that restricts the cases used in the model to customers whose occupation is architect and whose age is over 30.

```
ALTER MINING STRUCTURE MyStructure  ADD MINING MODEL MyModel_1
(
CustomerId,
Age,
Occupation,
MaritalStatus PREDICT
)
WITH FILTER (Age > 30 AND Occupation='Architect')
```

Example 2: Case-Level Filtering using Nested Table Attributes

If your mining structure contains nested tables, you can either filter on the existence of a value in a nested table, or filter on nested table rows that contain a specific value. This example restricts the cases used for the model to customers over the age of 30 who made at least one purchase that included milk.

As this example shows, it is not necessary that the filter use only columns that are included in the model. The nested table **Products** is part of the mining structure, but is not included in the mining model. However, you can still filter on values and attributes in the nested table. To view the details of these cases, drillthrough must be enabled.

```
ALTER MINING STRUCTURE MyStructure  ADD MINING MODEL MyModel_2
(
CustomerId,
Age,
Occupation,
MaritalStatus PREDICT
)
WITH DRILLTHROUGH,
FILTER (Age > 30 AND EXISTS (SELECT * FROM Products WHERE ProductName='Milk')
)
```

Example 3: Case-Level Filtering on Multiple Nested Table Attributes

This example shows a three-part filter: a condition applies to the case table, another condition to an attribute in the nested table, and another condition on a specific value in one of the nested table columns.

The first condition in the filter, `Age > 30`, applies to a column in the case table. The remaining conditions apply to the nested table.

The second condition, `EXISTS (SELECT * FROM Products WHERE ProductName='Milk'`, checks for the presence of at least one purchase in the nested table that included milk. The third condition, `Quantity>=2`, means that the customer must have purchased at least two units of milk in a single transaction.

```
ALTER MINING STRUCTURE MyStructure  ADD MINING MODEL MyModel_3
(
CustomerId,
Age,
Occupation,
MaritalStatus PREDICT,
Products PREDICT
(
ProductName KEY,
Quantity
)
)
FILTER (Age > 30 AND EXISTS (SELECT * FROM Products WHERE ProductName='Milk'  AND Quantity >= 2)
)
```

Example 4: Case-Level Filtering On Absence of Nested Table Attributes

This example shows how to limit cases to customer who did not purchase a specific item, by filtering on the absence of an attribute in the nested table. In this example, the model is trained using customers over the age of 30 who have never bought milk.

```
ALTER MINING STRUCTURE MyStructure  ADD MINING MODEL MyModel_4
(
CustomerId,
Age,
Occupation,
MaritalStatus PREDICT,
Products PREDICT
(
ProductName
)
)
FILTER (Age > 30 AND NOT EXISTS (SELECT * FROM Products WHERE ProductName='Milk') )
```

Example 5: Filtering on Multiple Nested Table Values

The purpose of the example is to show nested table filtering. The nested table filter is applied after the case filter, and only restricts nested table rows.

This model could contain multiple cases with empty nested tables because EXISTS is not specified.

```
ALTER MINING STRUCTURE MyStructure  ADD MINING MODEL MyModel_5
(
CustomerId,
Age,
Occupation,
MaritalStatus PREDICT,
Products PREDICT
(
ProductName KEY,
Quantity
) WITH FILTER(ProductName='Milk' OR ProductName='bottled water')
)
WITH DRILLTHROUGH
```

Example 6: Filtering on Nested Table Attributes and EXISTS

In this example, the filter on the nested table restricts the rows to those that contain either milk or bottled water. Then, the cases in the model are restricted by using an **EXISTS** statement. This makes sure that the nested table is not empty.

```
ALTER MINING STRUCTURE MyStructure  ADD MINING MODEL MyModel_6
(
CustomerId,
Age,
Occupation,
MaritalStatus PREDICT,
Products PREDICT
(
ProductName KEY,
Quantity
) WITH FILTER(ProductName='Milk' OR ProductName='bottled water')
)
FILTER (EXISTS (Products))
```

Example 7: Complex Filter Combinations

The scenario for this model resembles that of Example 4, but is far more complex. The nested table, **ProductsOnSale**, has the filter condition `(OnSale)` meaning that the value of **OnSale** must be **true** for the product listed in **ProductName**. Here, **OnSale** is a structure column.

The second part of the filter, for **ProductsNotOnSale**, repeats this syntax, but filters on products for which the value of **OnSale** is **not true** `(!OnSale)`.

Finally, the conditions are combined and one additional restriction is added to the case table. The result is to predict purchases of products in the **ProductsNotOnSale** list, based on the cases that are included in the **ProductsOnSale** list, for all customers over the age of 25.

`ALTER MINING STRUCTURE MyStructure ADD MINING MODEL MyModel_7`

`(`

`CustomerId,`

`Age,`

`Occupation,`

`MaritalStatus,`

`ProductsOnSale`

`(`

`ProductName KEY`

`) WITH FILTER(OnSale),`

`ProductsNotOnSale PREDICT ONLY`

`(`

`ProductName KEY`

`) WITH FILTER(!OnSale)`

`)`

`WITH DRILLTHROUGH,`

`FILTER (EXISTS (ProductsOnSale) AND EXISTS(ProductsNotOnSale) AND Age > 25)`

Example 8: Filtering on Dates

You can filter input columns on dates, as you would any other data. Dates contained in a column of type date/time

are continuous values; therefore, you can specify a date range by using operators such as greater than (>) or less than (<). If your data source does not represent dates by a Continuous data type, but as discrete or text values, you cannot filter on a date range, but must specify individual discrete values.

However, you cannot create a filter on the date column in a time series model if the date column used for the filter is also the key column for the model. That is because, in time series models and sequence clustering models, the date column might be handled as type **KeyTime** or **KeySequence**.

If you need to filter on continuous dates in a time series model, you can create a copy of the column in the mining structure, and filter the model on the new column.

For example, the following expression represents a filter on a date column of type **Continuous** that has been added to the Forecasting model.

```
=[DateCopy] > '12:31:2003:00:00:00'
```

> **NOTE**
> Note that any extra columns that you add to the model might affect the results. Therefore, if you do not want the column to be used in computation of the series, you should add the column only to the mining structure, and not to the model. You can also set the model flag on the column to **PredictOnly** or to **Ignore**. For more information, see Modeling Flags (Data Mining).

For other model types, you can use dates as input criteria or filter criteria just like you would in any other column. However, if you need to use a specific level of granularity that is not supported by a **Continuous** data type, you can create a derived value in the data source by using expressions to extract the unit to use in filtering and analysis.

> **WARNING**
> When you specify a dates as filter criteria, you must use the following format, regardless of the date format for the current OS: `mm/dd/yyyy`. Any other format results in an error.

For example, if you want to filter your call center results to show only weekends, you can create an expression in the data source view that extracts the weekday name for each date, and then use that weekday name value for input or as a discrete value in filtering. Just remember that repeating values can affect the model, so you should use only one of the columns, not the date column plus the derived value.

See Also

Filters for Mining Models (Analysis Services - Data Mining)
Testing and Validation (Data Mining)

Mining Model Tasks and How-tos

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use the **Mining Models** tab of Data Mining Designer in SQL Server Data Tools (SSDT) to manage and process mining models in a mining structure.

In This Section

- Add a Mining Model to an Existing Mining Structure
- Delete a Mining Model from a Mining Structure
- Exclude a Column from a Mining Model
- Create an Alias for a Model Column
- Change the Discretization of a Column in a Mining Model
- View or Change Modeling Flags (Data Mining)
- Specify a Column to Use as Regressor in a Model
- Change the Properties of a Mining Model
- Apply a Filter to a Mining Model
- Delete a Filter from a Mining Model
- Enable Drillthrough for a Mining Model
- View or Change Algorithm Parameters
- Make a Copy of a Mining Model
- Process a Mining Model
- Create a Data Mining Dimension

See Also

Mining Structure Tasks and How-tos
Mining Models (Analysis Services - Data Mining)
Data Mining Concepts

Add a Mining Model to an Existing Mining Structure

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can add more mining models to a mining structure, after you add the initial model. Each model must contain columns that exist in the structure, but you can define the usage of the columns differently for each mining model. For more information about how to define mining models columns, see Mining Model Columns.

To add a mining model to the structure

1. From the **Mining Model** menu item in SQL Server Data Tools (SSDT), select **New Mining Model**.

   The **New Mining Model** dialog box opens.

2. Under **Model name**, enter a name for the new mining model.

3. Under **Algorithm name**, select the algorithm that the mining model will be built from.

4. Click **OK**.

A new mining model appears in the **Mining Models** tab. The model uses the default columns that exist in the structure. For information about how to modify the columns, see Change the Properties of a Mining Model.

See Also

Mining Model Tasks and How-tos

Delete a Mining Model from a Mining Structure

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can delete mining models by using Data Mining Designer, by using SQL Server Management Studio, or by using DMX statements.

Delete a mining model using SQL Server Data Tools

1. Select the **Mining Models** tab in SQL Server Data Tools (SSDT).
2. Right-click the model that you want to delete, and select **Delete**.

   The **Delete Objects** dialog box opens.
3. Click **OK**.

Delete a mining model using SQL Server Management Studio

1. In SQL Server Management Studio, open the Analysis Services database that contains the model.
2. Expand **Mining Structures**, and then expand **Mining Models**.
3. Right-click the model you want to delete, and select **Delete**.

   Deleting the model does not delete the training data, only the metadata and any patterns created when you trained the model.

Delete a mining model using DMX

- DROP MINING MODEL (DMX)

See Also

Mining Model Tasks and How-tos

Exclude a Column from a Mining Model

11/27/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a new mining model, you may not want to use all the columns that exist in the mining structure on which the model is based. For example, you might have added a customer name column for drillthrough, but don't want to use it for modeling. Or, you might decide to create multiple copies of a column with different discretizations, and use only one of the copies in each model, and ignore the rest. You could also selectively add input columns in several different models to see how the added variable affects the output column.

You do not need to create a new mining structure for each combination of columns; instead, you can simply flag a column as not being used in a particular model.

To exclude a column from a mining model

1. In the **Mining Models** tab of Data Mining Designer in SQL Server Data Tools (SSDT), select the cell that corresponds to the column you want to exclude, under the appropriate mining model.

2. Select **Ignore** from the drop-down list box.

See Also

Mining Model Tasks and How-tos

Create an Alias for a Model Column

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In SQL Server 2017, you can create an alias for a model column. This might be useful when the mining structure name is too long to easily work with, or when you want to rename the column to be more descriptive of its contents or usage in the model. For example, if you make a copy of a structure column and then discretize the column differently for a particular model, you can rename the column to reflect the content more accurately.

To create an alias for a model column, you use the **Properties** pane and set the Name property of the column.

On the **Mining Models** tab of Data Mining Designer, the alias appears enclosed in parentheses next to the column usage label.

For information about how to set the properties of a mining model, see Mining Model Columns.

To add an alias to a mining model column

1. In the **Mining Models** tab in Data Mining Designer, right-click the cell in the mining model for the mining column that you want to change, and then select **Properties**.
2. In the **Properties** window on the right side of the screen, click the cell next to the Name property and delete the current value. Type a new name for the column.

See Also

Mining Model Tasks and How-tos
Mining Model Properties

Change the Discretization of a Column in a Mining Model

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services automatically discretizes values-that is to say, it bins data in numeric column-in certain scenarios. For example, if your data contains continuous numeric data and you create a decision tree model, each column of continuous data will be automatically binned, depending on the distribution of the data. If you want to control how the data is discretized, you must change the properties on the mining structure column that control how the data is used in the model.

For general information about how to set the properties in a mining model, see Mining Model Columns.

To display the properties for a mining model column

1. In the **Mining Models** tab in Data Mining Designer, right-click the column header that contains the name of the mining model, or the row in the grid that contains the name of the mining algorithm, and then select **Properties**.

   The **Properties** window displays the properties that are associated with the mining model as a whole.

2. In the **Structure** column near the left side of the screen, click the column that contains the continuous numeric data you want to discretize.

   The **Properties** window changes to display just the properties associated with that column.

To change the discretization method

1. In the **Mining Properties** window, click the text box next to **Content**, and select **Discretized** from the dropdown list.

   The DiscretizationBucketCount and DiscretizationMethod properties are now enabled.

2. In the **Properties** window, click the text box next to DiscretizationMethod and select one of the following values: **Automatic**, **EqualAreas**, or **Cluster**.

   > **NOTE**
   >
   > If the column usage is set to **Ignore**, the **Properties** window for the column is blank.

   The new value will take effect when you select a different element in the designer.

3. In the **Properties** window, click the text box next to DiscretizationBucketCount and type a numeric value.

   > **NOTE**
   >
   > If you change these properties, the structure must be reprocessed, along with any models that you want to use the new setting.

See Also

Mining Model Tasks and How-tos

View or Change Modeling Flags (Data Mining)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Modeling flags are properties that you set on a mining structure column or mining model columns to control how the algorithm processes the data during analysis.

In Data Mining Designer, you can view and modify the modeling flags associated with a mining structure or mining column by viewing the properties of the structure or model. You can also set modeling flags by using DMX, AMO, or XMLA.

This procedure describes how to change the modeling flags in the designer.

View or change the modeling flag for a structure column or model column

1. In SQL Server Design Studio, open Solution Explorer, and then double-click the mining structure.
2. To set the NOT NULL modeling flag, click the **Mining Structure** tab. To set the REGRESSOR or MODEL_EXISTENCE_ONLY flags, click the **Mining Model** tab.
3. Right-click the column you want to view or change, and select **Properties**.
4. To add a new modeling flag, click the text box next to the **ModelingFlags** property, and select the check box or check boxes for the modeling flags you want to use.

   Modeling flags are displayed only if they are appropriate for the column data type.

   > **NOTE**
   >
   > After you change a modeling flag, you must reprocess the model.

Get the modeling flags used in the model

- In SQL Server Management Studio, open a DMX Query window, and type a query like the following:

```
SELECT COLUMN_NAME, CONTENT_TYPE, MODELING_FLAG
FROM $system.DMSCHEMA_MINING_COLUMNS
WHERE MODEL_NAME = 'Forecasting'
```

See Also

Mining Model Tasks and How-tos
Modeling Flags (Data Mining)

Specify a Column to Use as Regressor in a Model

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A linear regression model represents the value of the predictable attribute as the result of a formula that combines the inputs in such a way that the data is fitted as a closely as possible to an estimated regression line. The algorithm accepts only numeric values as inputs, and automatically detects the inputs that provide the best fit.

However, you can specify that a column be included as a regressor by adding the FORCE_REGRESSOR parameter to the model and specifying the regressors to use. You might want to do this in cases where the attribute has meaning even if the effect is too small to be detected by the model, or when you want to ensure that the attribute is included in the formula.

The following procedure describes how to create a simple linear regression model, using the same sample data that is used for the neural networks tutorial. The model is not necessarily robust, but demonstrates how to use the Data Mining Designer to customize a linear regression model.

How to create a simple linear regression model

1. In SQL Server Data Tools (SSDT), in **Solution Explorer**, expand **Mining Structures**.
2. Double-click Call Center.dmm to open it in the designer.
3. From the **Mining Model** menu, select **New Mining Model**.
4. For the algorithm, select **Microsoft Linear Regression**. For the name, type **Call Center Regression**.
5. In the **Mining Models** tab, change the column usage as follows. All columns not in the following list should be set to **Ignore**, if they are not already.

   FactCallCenterID**Key**

   ServiceGrade**PredictOnly**

   Total Operators**Input**

   AverageTimePerIssue**Input**
6. From the **Mining Model** menu, select **Set Model Parameters**.
7. For the parameter, FORCE_REGRESSOR, in the **Value** column, type the column names enclosed in brackets and separated by a comma, as follows:

   ```
   [Average Time Per Issue],[Total Operators]
   ```

   > **NOTE**
   > The algorithm will automatically detect which columns are the best regressors. You only need to force regressors when you want to ensure that a column is included in the final formula.

8. From the **Mining Model** menu, select **Process Model**.

   In the viewer, the model is represented a single node containing the regression formula. You can view the formula in the **Mining Legend**, or you can extract the coefficients for the formula by using queries.

See Also

Microsoft Linear Regression Algorithm
Data Mining Queries
Microsoft Linear Regression Algorithm Technical Reference
Mining Model Content for Linear Regression Models (Analysis Services - Data Mining)

Change the Properties of a Mining Model

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Some mining model properties apply to the model as a whole, and other model properties apply to individual columns. Examples of properties that apply to the entire model would be the **Drillthrough** property, which specifies whether the case data should be available for querying, and the **Description** property. Properties that apply to the column include **Usage** and **ModelingFlags**, which control how data in the column is used within the model.

The following model properties have advanced editors that you can use to create expressions or configure complex model properties. The following properties provide:

- **Filter** property: Opens the Data Set Filter or Model Filter Dialog Box.
- **AlgorithmParameters** property: Opens the Algorithm Parameters Dialog Box (Mining Models View).

For information about how to set the properties in a mining model, see Mining Model Columns.

To change the properties of a mining model

1. In the **Mining Models** tab in Data Mining Designer, right-click either the column header that contains the name of the mining model, or the row in the grid that contains the name of the mining algorithm, and then select **Properties**.
2. In the **Properties** window on the right side of the screen, highlight the value that corresponds to the property that you want to change, and enter the new value.

   The new value will take effect when you select a different element in the designer.

To change the properties of a mining model column

1. In the **Mining Models** tab in Data Mining Designer, right-click the cell in the grid at the intersection of the mining structure column and the mining model, and then select **Properties**.
2. In the **Properties** window on the right side of the screen, highlight the value that corresponds to the property that you want to change, and enter the new value.

   > **NOTE**
   >
   > If the column usage is set to **Ignore**, the **Properties** window for the column is blank.

   The new value will take effect when you select a different element in the designer.

See Also

Mining Model Tasks and How-tos

Apply a Filter to a Mining Model

11/28/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If your mining structure contains a nested table, you can apply a filter to the case table, the nested table, or both.

The following procedure demonstrates how to create both kinds of filters: case filters, and filters on the nested table rows.

The condition on the case table restricts customers to those with income between 30000 and 40000. The condition on the nested table restricts the customers to those who did not purchase a particular item.

The complete filter condition created in this example is as follows:

```
[Income] > '30000'
AND  [Income] < '40000'
AND EXISTS (SELECT * FROM [<nested table name>]
WHERE [Model] <> 'Water Bottle' )
```

To create a case filter on a mining model

1. In SQL Server Data Tools (SSDT), in Solution Explorer, click the mining structure that contains the mining model you want to filter.
2. Click the **Mining Models** tab.
3. Select the model, and right-click to open the shortcut menu.

   -or-

   Select the model. Then, on the **Mining Model** menu, select **Set Model Filter**.
4. In the **Model Filter** dialog box, click the top row in the grid, in the **Mining Structure Column** text box.
5. If the data source contains a single flat table, the drop-down list displays only the names of the columns in that table.

   If the mining structure contains multiple tables, the list shows the names of the source tables. The column names do not display until a table has been selected.

   If the mining structure contains a case table and a nested table, the drop-down list shows columns from the case table, and the name of the nested table.
6. Select a column from the drop-down list.

   The icon at the left side of the text box changes to indicate that the selected item is a table or a column.
7. Click the **Operator** text box and select an operator from the list. The valid operators change depending on the data type of the column you selected.
8. Click the **Value** text box, and type a value in the box.

   For example, select **Income** as the column, select the greater than operator (>), and then type **30000**.
9. Click the next row in the grid.

   The filter condition that you created is automatically added to the Expression text box. For example,

```
[Income] > '30000'
```

10. Click the **AND/OR** text box in the next row of the grid to add a condition.

    For example, to create a BETWEEN condition, select **AND** from the drop-down list of logical operands.

11. Select an operator and type a value as described in Steps 7 and 8.

    For example, select **Income** as the column again, select the less than operator (<), and then type **40000**.

12. Click the next row in the grid.

13. The filter condition in the Expression text box is automatically updated to include the new condition. The completed expression is as follows: `[Income] > '30000'AND [Income] < '40000'`

To add a filter on the nested table in a mining model

1. In the **<name>Model Filter** Dialog box, click an empty row in the grid under **Mining Structure Column**.

2. Select the name of the nested table from the drop-down list.

   The icon at the left side of the text box changes to indicate that the selected item is the name of a table.

3. Click the **Operator** text box and select **Contains** or **Not Contain**.

   These are the only conditions available for the nested table in the **Model Filter** dialog box, because you are restricting the case table to only those cases that contain a certain value in the nested table. You will set the value for the condition on the nested table in the next step.

4. Click the **Value** box, and then click the **(...)** button to build an expression.

   The **<name>Filter** dialog box opens. This dialog box can set conditions only on the current table, which in this case is the nested table.

5. Click the **Mining Structure Column** box and select a column name from the dropdown lists of nested table columns.

6. Click **Operator** and select an operator from the list of valid operators for the column.

7. Click **Value** and type a value.

   For example, for **Mining Structure Column,** select **Model**. For **Operator**, select **<>**, and type the value **Water Bottle**. This condition creates the following filter expression:

```
EXISTS (SELECT * FROM [<nested table name>] WHERE [Model] <> 'Water Bottle' )
```

> **NOTE**
> Because the number of nested table attributes is potentially unlimited, Analysis Services does not supply a list of possible values from which to select. You must type the exact value. Also, you cannot use a LIKE operator in a nested table.

1. Add more conditions as necessary, combining the conditions by selecting **AND** or **OR** in the **AND/OR** box at the left side of the **Conditions** grid. Click **OK**.

2. In the **Model Filter** dialog box, review the conditions that you created by using the **Filter** dialog box. The conditions for the nested table are appended to the conditions for the case table, and the complete set of filter conditions is displayed in the **Expression** text box.

3. Optionally, click **Edit Query** to manually change the filter expression.

> **NOTE**
> If you change any part of a filter expression manually, the grid will be disabled and thereafter you must work with the filter expression in text edit mode only. To restore grid editing mode, you must clear the filter expression and start over.

See Also

Filters for Mining Models (Analysis Services - Data Mining)
Mining Model Tasks and How-tos
Delete a Filter from a Mining Model

Delete a Filter from a Mining Model

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create a filter on a mining model, you can create models on a subset of the data in the data source view. Filters are also useful for testing the accuracy of the model on a subset of the original data.

However, you must delete the filter if you want to view the complete set of cases again. This procedure describes how to remove conditions on a filter, or delete the filter completely.

To delete a condition from a filter on a mining model

1. In SQL Server Data Tools (SSDT), in Solution Explorer, click the mining structure that contains the mining model you want to filter.
2. Click the **Mining Models** tab.
3. Select the model, and right-click to open the shortcut menu.

   -or-

   Select the model. On the **Mining Model** menu, select **Set Model Filter**.
4. In the **Model Filter** dialog box, right-click the row in the grid that contains the condition you want to delete.
5. Select **Delete**.

To clear the filter on a mining model in the Filter Editor dialog box

- In the **Filter Editor** dialog box, right-click any row in the grid, and select **Delete All**.

Working with Model Filters Using the Properties Window

If you want to delete the whole filter, you do not need to open the filter editor dialog boxes. The filter conditions that you created are available in the **Filter** property of the mining model.

> **NOTE**
> You can view the properties of a mining model in SQL Server Data Tools (SSDT), but not in SQL Server Management Studio.

To clear the filter on a mining model in Solution Explorer

1. In Solution Explorer, click the mining model that contains the filter.
2. In the **Properties** window, right-click the filter text in the **Filter** property, and select **Select All**.
3. Press the Backspace or Delete key.

See Also

Drill Through to Case Data from a Mining Model
Mining Model Tasks and How-tos
Filters for Mining Models (Analysis Services - Data Mining)

Enable Drillthrough for a Mining Model

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If you have enabled drillthrough for a mining model, when you browse the model you can retrieve detailed information about the cases that were used to create the model. To view this information, you must have the necessary permissions, and the structure must have already been processed.

Permissions For a user to drill through to either model data or structure data, the user must be a member of a role that has AllowDrillThrough permissions on the mining model or mining structure. Drillthrough permissions are set separately on the structure and model.

- Drillthrough permissions on the model enable you to drill through from the model, even if you do not have permissions on the structure.
- Drillthrough permissions on the structure provide the additional ability to include structure columns in drillthrough queries from the model, by using the StructureColumn (DMX) function. You can also query the training and test cases in the structure by using the SELECT... FROM <structure>.CASES syntax.

Caching of training cases Drillthrough works by retrieving information about the training cases in the mining structure. This information is cached when the structure is processed. Therefore, if you choose to clear all cached data by changing the MiningStructureCacheMode property to **ClearAfterProcessing**, drillthrough will not work.

> **NOTE**
> If the training cases have not been cached, you must change the MiningStructureCacheMode property to **KeepTrainingCases** and then reprocess the model before you can view the case data.

For more information, see Drillthrough Queries (Data Mining).

To enable drillthrough on a mining model

1. In SQL Server Data Tools (SSDT), on the **Mining Models** tab of Data Mining Designer, right-click the name of the mining model on which you want to enable drillthrough, and select **Properties**.
2. In the **Properties** windows, click **AllowDrillthrough**, and select **True**.
3. In the **Mining Models** tab, right-click the model, and select **Process Model**.

To enable caching for a mining structure

1. In SQL Server Data Tools (SSDT), on the **Mining Structure** tab of Data Mining Designer, right-click the name of the mining structure.
2. Open the **Properties** window.
3. In the **Properties** window, locate the **CacheMode** property, and select **KeepTrainingCases** from the list.
4. On the **Database** menu, select **Process**.

See Also

Drillthrough Queries (Data Mining)

View or Change Algorithm Parameters

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can change the parameters provided with the algorithms that you use to build data mining models to customize the results of the model.

The algorithm parameters provided in Microsoft SQL Server Analysis Services change much more than just properties on the model: they can be used to fundamentally alter the way that data is processed, grouped, and displayed. For example, you can use algorithm parameters to do the following:

- Change the method of analysis, such as the clustering method.
- Control feature selection behavior.
- Specify the size of itemsets or the probability of rules.
- Control branching and depth of decision trees.
- Specify a seed value or the size of the internal holdout set used for model creation.

The parameters provided for each algorithm vary greatly; for a list of the parameters that you can set for each algorithm, see the technical reference topics in this section: Data Mining Algorithms (Analysis Services - Data Mining).

Change an algorithm parameter

1. On the **Mining Models** tab of Data Mining Designer in SQL Server Data Tools (SSDT), right-click the algorithm type of the mining model for which you want to tune the algorithm, and select **Set Algorithm Parameters**.

   The **Algorithm Parameters** dialog box opens.

2. In the **Value** column, set a new value for the algorithm that you want to change.

   If you do not enter a value in the **Value** column, Analysis Services uses the default parameter value. The **Range** column describes the possible values that you can enter.

3. Click **OK**.

   The algorithm parameter is set with the new value. The parameter change will not be reflected in the mining model until you reprocess the model.

View the parameters used in an existing model

1. In SQL Server Management Studio, open a DMX Query window.

2. Type a query like this one:

```
select MINING_PARAMETERS
from $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = '<model name>'
```

See Also

Process a Mining Model

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In the Mining Models tab of Data Mining Designer in SQL Server Data Tools (SSDT), you can either process a specific mining model that is associated with a mining structure or you can process all the models that are associated with the structure.

You can process a mining model by using the following tools:

- SQL Server Data Tools (SSDT)
- SQL Server Management Studio

You can also use an XMLA Process command. For more information, see Tools and Approaches for Processing (Analysis Services).

Process a single mining model using SQL Server Data Tools

1. On the **Mining Models** tab of Data Mining Designer, select a mining model from the one or more columns of models in the grid.
2. On the **Mining Model** menu, select **Process Model**.

   If you made changes to the mining structure, you will be prompted to redeploy the structure before processing the model. Click **Yes**.
3. In the **Processing Mining Model - <model>** dialog box, click **Run**.

   The **Process Progress** dialog box opens to show the details of model processing.
4. After the model has successfully completed processing, click **Close** in the **Process Progress** dialog box.
5. Click **Close** in the **Processing Mining Model - <model>** dialog box.

Only the mining structure and the selected mining model have been processed.

Process all mining models that are associated with a mining structure

1. On the **Mining Models** tab of Data Mining Designer, select **Process Mining Structure and All Models** from the **Mining Model** menu.
2. If you made changes to the mining structure, you will be prompted to redeploy the structure before processing the models. Click **Yes**.
3. In the **Processing Mining Structure - <structure>** dialog box, click **Run**.
4. The **Process Progress** dialog box opens to show the details of model processing.
5. After the models have successfully completed processing, click **Close** in the **Process Progress** dialog box.
6. Click **Close** in the **Processing <model>** dialog box.

The mining structure and all the associated mining models have been processed.

See Also

Mining Model Tasks and How-tos

Make a Copy of a Mining Model

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Creating a copy of a mining model is useful when you want to quickly create several mining models that are based on the same data. After you copy the model, you can then edit the new copy by changing parameters or adding a filter.

For example, if you have a Customers table that is linked to a table of purchases, you could create copies to generate separate mining models for each customer demographic, filtering on attributes such as age or region.

For information about how to copy the content of the model (such as the graphical representation or the model patterns) to the Clipboard for use in other programs, see Copy a View of a Mining Model.

To create a related mining model

1. In SQL Server Data Tools (SSDT), in Solution Explorer, click the mining structure that contains the mining model.
2. Click the **Mining Models** tab.
3. Select the model, and right-click to open the shortcut menu.

   -or-

   Select the model. On the **Mining Model** menu, select **New Mining Model**.
4. Type a name for the new mining model, and select an algorithm. Click **OK**.

To edit the filter on the copied mining model

1. Select the mining model.
2. In the **Properties** window, click the text box for the **Filter** property, and the click the build **(...)** button.
3. Change the filter conditions.

   For more information about how to use the filter editor dialog boxes, see Apply a Filter to a Mining Model.
4. In the **Properties** window, in the **AlgorithmParameters** text box, click **Setalgorithm parameters**, and change algorithm parameters, if desired.
5. Click **OK**.

See Also

Filters for Mining Models (Analysis Services - Data Mining)
Mining Model Tasks and How-tos
Delete a Filter from a Mining Model

Create a Data Mining Dimension

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If your mining structure is based on an OLAP cube, you can create a dimension that contains the content of the mining model. You can then incorporate the dimension back into the source cube.

You can also browse the dimension, use it to explore the model results, or query the dimension using MDX.

To create a data mining dimension

1. In Data Mining Designer in SQL Server Data Tools, select either the **Mining Structure** tab or the **Mining Models** tab.
2. From the **Mining Model** menu, select **Create a Data Mining Dimension**.

   The **Create Data Mining Dimension** dialog box opens.
3. In the **Model name** list of the **Create Data Mining Dimension** dialog box, select an OLAP mining model.
4. In the **Dimension name** box, enter a name for the new data mining dimension.
5. If you want to create a cube that includes the new data mining dimension, select **Create cube**. After you select **Create cube**, you can enter a new name for the cube.
6. Click **OK**.

   The data mining dimension is created and is added to the **Dimensions** folder in Solution Explorer. If you selected **Create cube**, a new cube is also created and is added to the **Cubes** folder.

See Also

Mining Structure Tasks and How-tos

Testing and Validation (Data Mining)

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Validation is the process of assessing how well your mining models perform against real data. It is important that you validate your mining models by understanding their quality and characteristics before you deploy them into a production environment.

This section introduces some basic concepts related to model quality, and describes the strategies for model validation that are provided in Microsoft Analysis Services. For an overview of how model validation fits into the larger data mining process, see Data Mining Solutions.

Methods for Testing and Validation of Data Mining Models

There are many approaches for assessing the quality and characteristics of a data mining model.

- Use various measures of statistical validity to determine whether there are problems in the data or in the model.
- Separate the data into training and testing sets to test the accuracy of predictions.
- Ask business experts to review the results of the data mining model to determine whether the discovered patterns have meaning in the targeted business scenario

All of these methods are useful in data mining methodology and are used iteratively as you create, test, and refine models to answer a specific problem. No single comprehensive rule can tell you when a model is good enough, or when you have enough data.

Definition of Criteria for Validating Data Mining Models

Measures of data mining generally fall into the categories of accuracy, reliability, and usefulness.

Accuracy is a measure of how well the model correlates an outcome with the attributes in the data that has been provided. There are various measures of accuracy, but all measures of accuracy are dependent on the data that is used. In reality, values might be missing or approximate, or the data might have been changed by multiple processes. Particularly in the phase of exploration and development, you might decide to accept a certain amount of error in the data, especially if the data is fairly uniform in its characteristics. For example, a model that predicts sales for a particular store based on past sales can be strongly correlated and very accurate, even if that store consistently used the wrong accounting method. Therefore, measurements of accuracy must be balanced by assessments of reliability.

Reliability assesses the way that a data mining model performs on different data sets. A data mining model is reliable if it generates the same type of predictions or finds the same general kinds of patterns regardless of the test data that is supplied. For example, the model that you generate for the store that used the wrong accounting method would not generalize well to other stores, and therefore would not be reliable.

Usefulness includes various metrics that tell you whether the model provides useful information. For example, a data mining model that correlates store location with sales might be both accurate and reliable, but might not be useful, because you cannot generalize that result by adding more stores at the same location. Moreover, it does not answer the fundamental business question of why certain locations have more sales. You might also find that a model that appears successful in fact is meaningless, because it is based on cross-correlations in the data.

Tools for Testing and Validation of Mining Models

Analysis Services supports multiple approaches to validation of data mining solutions, supporting all phases of the data mining test methodology.

- Partitioning data into testing and training sets.
- Filtering models to train and test different combinations of the same source data.
- Measuring *lift* and *gain*. A *lift chart* is a method of visualizing the improvement that you get from using a data mining model, when you compare it to random guessing.
- Performing *cross-validation* of data sets
- Generating *classification matrices*. These charts sort good and bad guesses into a table so that you can quickly and easily gauge how accurately the model predicts the target value.
- Creating *scatter plots* to assess the fit of a regression formula.
- Creating *profit charts* that associate financial gain or costs with the use of a mining model, so that you can assess the value of the recommendations.

These metrics do not aim to answer the question of whether the data mining model answers your business question; rather, these metrics provide objective measurements that you can use to assess the reliability of your data for predictive analytics, and to guide your decision of whether to use a particular iterate on the development process.

The topics in this section provide an overview of each method and walk you through the process of measuring the accuracy of models that you build using SQL Server Data Mining.

Related Topics

TOPICS	LINKS
Learn how to set up a testing data set using a wizard or DMX commands	Training and Testing Data Sets
Learn how to test the distribution and representativeness of the data in a mining structure	Cross-Validation (Analysis Services - Data Mining)
Learn about the accuracy chart types provided.	Lift Chart (Analysis Services - Data Mining) Profit Chart (Analysis Services - Data Mining) Scatter Plot (Analysis Services - Data Mining)
Learn how to create a classification matrix, sometimes called a confusion matrix, for assessing the number of true and false positives and negatives.	Classification Matrix (Analysis Services - Data Mining)

See Also

Data Mining Tools
Data Mining Solutions
Testing and Validation Tasks and How-tos (Data Mining)

Training and Testing Data Sets

10/26/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Separating data into training and testing sets is an important part of evaluating data mining models. Typically, when you separate a data set into a training set and testing set, most of the data is used for training, and a smaller portion of the data is used for testing. Analysis Services randomly samples the data to help ensure that the testing and training sets are similar. By using similar data for training and testing, you can minimize the effects of data discrepancies and better understand the characteristics of the model.

After a model has been processed by using the training set, you test the model by making predictions against the test set. Because the data in the testing set already contains known values for the attribute that you want to predict, it is easy to determine whether the model's guesses are correct.

Creating Test and Training Sets for Data Mining Structures

In SQL Server 2017, you separate the original data set at the level of the mining structure. The information about the size of the training and testing data sets, and which row belongs to which set, is stored with the structure, and all the models that are based on that structure can use the sets for training and testing.

You can define a testing data set on a mining structure in the following ways:

- Using the Data Mining Wizard to divide the mining structure when you create it.
- Modifying structure properties in the **Mining Structure** tab of the Data Mining Designer.
- Creating and modifying structures programmatically by using Analysis Management Objects (AMO) or XML Data Definition Language (DDL).

Using the Data Mining Wizard to Divide a Mining Structure

By default, after you have defined the data sources for a mining structure, the Data Mining Wizard will divide the data into two sets: one with 70 percent of the source data, for training the model, and one with 30 percent of the source data, for testing the model. This default was chosen because a 70-30 ratio is often used in data mining, but with Analysis Services you can change this ratio to suit your requirements.

You can also configure the wizard to set a maximum number of training cases, or you can combine the limits to allow a maximum percentage of cases up to a specified maximum number of cases. When you specify both a maximum percentage of cases and a maximum number of cases, Analysis Services uses the smaller of the two limits as the size of the test set. For example, if you specify 30 percent holdout for the testing cases, and the maximum number of test cases as 1000, the size of the test set will never exceed 1000 cases. This can be useful if you want to ensure that the size of your test set stays consistent even if more training data is added to the model.

If you use the same data source view for different mining structures, and want to ensure that the data is divided in roughly the same way for all mining structures and their models, you should specify the seed that is used to initialize random sampling. When you specify a value for **HoldoutSeed**, Analysis Services will use that value to begin sampling. Otherwise, sampling uses a hashing algorithm on the name of the mining structure to create the seed value.

> **NOTE**
> If you create a copy of the mining structure by using the **EXPORT** and **IMPORT** statements, the new mining structure will have the same training and testing data sets, because the export process creates a new ID but uses the same name. However, if two mining structures use the same underlying data source but have different names, the sets that are created for each mining structure will be different.

Modifying Structure Properties to Create a Test Data Set

If you create and process a mining structure, and then later decide that you want to set aside a test data set, you can modify the properties of the mining structure. To change the way that data is partitioned, edit the following properties:

PROPERTY	DESCRIPTION
HoldoutMaxCases	Specifies the maximum number of cases to include in the testing set.
HoldoutMaxPercent	Specifies the number of cases to include in the testing set as a percentage of the complete data set. To have no data set, you would specify 0.
HoldoutSeed	Specifies an integer value to use as seed when randomly selecting data for the partitions. This value does not affect the number of cases in the training set; instead, it ensures that the partition can be repeated.

If you add or change a test data set to an existing structure, you must reprocess the structure and all associated models. Also, because dividing the source data causes the model to be trained on a different subset of the data, you might see different results from your model.

Specifying Holdout Programmatically

You can define testing and training data sets on a mining structure by using DMX statements, AMO, or XML DDL. The ALTER MINING STRUCTURE statement does not support the use of holdout parameters.

- **DMX** In the Data Mining Extensions (DMX) language, the CREATE MINING STRUCTURE statement has been extended to include a WITH HOLDOUT clause..
- **ASSL** You can either create a new mining structure, or add a testing data set to an existing mining structure, by using the Analysis Services Scripting Language (ASSL)..
- **AMO** You can also view and modify holdout data sets by using AMO..

You can view information about the holdout data set in an existing mining structure by querying the data mining schema rowset. You can do this by making a DISCOVER ROWSET call, or you can use a DMX query.

Retrieving Information about Holdout Data

By default, all information about the training and test data sets is cached, so that you can use existing data to train and then test new models. You can also define filters to apply to the cached holdout data so that you can evaluate the model on subsets of the data.

The way that cases are divided into training and testing data sets depends on the way that you configure holdout, and the data that you provide. If you want to determine the number of cases used for training or for testing, or if you want to find additional details about the cases included in the training and test sets, you can query the model structure by creating a DMX query. For example, the following query returns the cases that were used in the training set of the model.

```
SELECT * from <structure>.CASES WHERE IsTrainingCase()
```

To retrieve only the test cases, and additionally filter the test cases on one of the columns in the mining structure, use the following syntax:

```
SELECT * from <structure>.CASES WHERE IsTestCase() AND <structure column name> = '<value>'
```

Limitations on the Use of Holdout Data

- To use holdout, the MiningStructureCacheMode property of the mining structure must be set to the default value, **KeepTrainingCases**. If you change the **CacheMode** property to **ClearAfterProcessing**, and then reprocess the mining structure, the partition will be lost.
- You cannot remove data from a time series model; therefore, you cannot separate the source data into training and testing sets. If you begin to create a mining structure and model and choose the Microsoft Time Series algorithm, the option to create a holdout data set is disabled. Use of holdout data is also disabled if the mining structure contains a KEY TIME column at either the case or nested table level.
- It is possible to inadvertently configure the holdout data set such that the complete data set is used for testing, and no data remains for training. However, if you do so, Analysis Services will raise an error so that you can correct the problem. Analysis Services also warns you when the structure is processed if more than 50 percent of the data has been held out for testing.
- In most cases, the default holdout value of 30 provides a good balance between training and testing data. There is no simple way to determine how large the data set should be to provide sufficient training, or how sparse the training set can be and still avoid overfitting. However, after you have built a model, you can use cross-validation to assess the data set with respect to a particular model.
- In addition to the properties listed in the previous table, a read-only property, **HoldoutActualSize**, is provided in AMO and XML DDL. However, because the actual size of a partition cannot be determined accurately until after the structure has been processed, you should check whether the model has been processed before you retrieve the value of the **HoldoutActualSize** property.

Related Content

TOPICS	LINKS
Describes how filters on a model interact with training and testing data sets.	Filters for Mining Models (Analysis Services - Data Mining)
Describes how the use of training and testing data affects cross-validation.	Cross-Validation (Analysis Services - Data Mining)
Provides information on the programmatic interfaces for working with training and testing sets in a mining structure.	AMO Concepts and Object Model MiningStructure Element (ASSL)
Provides DMX syntax for creating holdout sets.	CREATE MINING STRUCTURE (DMX)
Retrieve information about cases in the training and testing sets.	Data Mining Schema Rowsets Data Mining Schema Rowsets (SSAs)

See Also

Data Mining Tools
Data Mining Concepts
Data Mining Solutions
Testing and Validation (Data Mining)

Lift Chart (Analysis Services - Data Mining)

11/27/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A *lift chart* graphically represents the improvement that a mining model provides when compared against a random guess, and measures the change in terms of a *lift* score. By comparing the lift scores for different models, you can determine which model is best. You can also determine the point at which the model's predictions become less useful. For example, by reviewing the lift chart, you might realize that a promotional campaign is likely to be effective against only 30% of your customers, and use that figure to limit the scope of the campaign.

In SQL Server Data Mining, the lift chart can compare the accuracy of multiple models that have the same predictable attribute. You can also assess the accuracy of prediction either for a single outcome (a single value of the predictable attribute), or for all outcomes (all values of the specified attribute).

A profit chart is a related chart type that contains the same information as a lift chart, but also displays the projected increase in profit that is associated with using each model.

Understanding the Lift Chart

It can be hard to understand lift charts in the abstract. Therefore, to illustrate the use of the lift chart tools and the information in the chart, this section presents a scenario in which a lift chart is used to estimate the response to a targeted mailing campaign.

The marketing department in this scenario knows that a 10 percent response rate is more or less typical of mailing campaigns. They have a list of 10,000 potential customers stored in a table in the database. Based on the typical response rate, they could normally expect only about 1,000 of the potential customers to respond. However, the money budgeted for the project is not enough to reach all 10,000 customers in the database, and they want to improve their response rate. Assume for this scenario that their budget allows them to mail an advertisement to only 5,000 customers. The marketing department has two options:

- Randomly select 5,000 customers to target.
- Use a mining model to target the 5,000 customers who are most likely to respond.

By using a lift chart, you can compare the expected results of both options. For example, if the company randomly selected 5,000 customers, they might expect to receive only 500 responses, based on the typical response rate. This scenario is what the *random* line in the lift chart represents. However, if the marketing department used a mining model to target their mailing, they could expect a better response rate because the model would identify those customers who are most likely to respond. If the model were perfect, it would create predictions that are never wrong, and the company could expect to receive 1,000 responses by sending the mailing just to the 1,000 potential customers recommended by the model. This scenario is what the *ideal* line in the lift chart represents.

The reality is that the mining model most likely falls between these two extremes; between a random guess and a perfect prediction. Any improvement from the random guess is considered to be lift.

When you create a lift chart, you can target a specific value and measure lift only for that outcome, or you can create a general assessment of the model that measures lifts for all possible outcomes. These selections affect the final chart, as described in the following sections.

Back to Top

Lift Chart with Target Value

The following chart shows a lift chart for the **Targeted Mailing** model that you create in the Basic Data Mining Tutorial. In this chart, the target attribute is [Bike Buyer] and the target value is 1, meaning that the customer is predicted to buy a bike. The lift chart thus shows the improvement the model provides when identifying these potential customers.

This chart contains multiple models based on the same data. One of these models has been customized to target specific customers. You can customize a model by adding filters on the data used to train the mode. This filter restricts the cases used in both training and evaluation to customers who are under the age of 30. Notice that one effect of filtering is that the basic model and the filtered model use different data sets, and therefore the number of cases used for evaluation in the lift chart is different as well. This point is important to remember when you interpret the prediction results and other statistics.

The x-axis of the chart represents the percentage of the test dataset that is used to compare the predictions. The y-axis of the chart represents the percentage of predicted values.

The diagonal straight line, shown here in blue, appears in every chart. It represents the results of random guessing, and is the baseline against which to evaluate lift. For each model that you add to a lift chart, you get two additional lines: one line shows the ideal results for the training data set if you could create a model that always predicted perfectly, and the second line shows the actual lift, or improvement in results, for the model.

In this example, the ideal line for the filtered model is shown in dark blue, and the line for actual lift in yellow. You can tell from the chart that the ideal line peaks at around 40 percent, meaning that if you had a perfect model, you could reach 100 percent of your targeted customers by sending a mailing to only 40% of the total population. The actual lift for the filtered model when you target 40 percent of the population is between 60 and 70 percent, meaning you could reach 60-70 percent of your targeted customers by sending the mailing to 40 percent of the total customer population.

The **Mining Legend** contains the actual values at any point on the curves. You can change the place that is measured by clicking the vertical gray bar and moving it. In the chart, the gray line has been moved to 30 percent, because this is the point where both the filtered and unfiltered models appear to be most effective, and after this point the amount of lift declines.

The **Mining Legend** also contains scores and statistics that help you interpret the chart. These results represent the accuracy of the model at the gray line, which in this scenario is positioned to include 30 percent of the overall test cases.

SERIES AND MODEL	SCORE	TARGET POPULATION	PREDICT PROBABILITY
Targeted mailing all	0.71	47.40%	61.38%
Targeted mailing under 30	0.85	51.81%	46.62%
Random guess model		31.00%	
Ideal model for: Targeted mailing all		62.48%	
Ideal model for: Targeted mailing under 30		65.28%	

Back to Top

Interpreting the Results

From these results, you can see that, when measured at 30 percent of all cases, the general model, [Targeted mailing all], can predict the bike buying behavior of 47.40% of the target population. In other words, if you sent out a targeted mailing to only 30 percent of the customers in your database, you could reach slightly less than half of your target audience. If you used the filtered model, you could get slightly better results, and reach about 51 percent of your targeted customers.

The value for **Predict probability** represents the threshold required to include a customer among the "likely to buy" cases. For each case, the model estimates the accuracy of each prediction and stores that value, which you can use to filter out or to target customers. For example, to identify the customers from the basic model who are likely buyers, you would use a query to retrieve cases with a Predict probability of at least 61 percent. To get the customers targeted by the filtered model, you would create query that retrieved cases that met all the criteria: age and a **PredictProbability** value of at least 46 percent.

It is interesting to compare the models. The filtered model appears to capture more potential customers, but when you target customers with a prediction probability score of 46 percent, you also have a 53 percent chance of sending a mailing to someone who will not buy a bike. Therefore, if you were deciding which model is better, you would want to balance the greater precision and smaller target size of the filtered model against the selectiveness of the basic model.

The value for **Score** helps you compare models by calculating the effectiveness of the model across a normalized population. A higher score is better, so in this case you might decide that targeting customers under 30 is the most effective strategy, despite the lower prediction probability.

How is the Score Calculated?

The score is calculated as the geometric mean score of all the points constituting a scatter plot in which the x-axis contains the actual values, the y-axis contains the predicted value, and each point has an associated probability.

The statistical meaning of any individual point score is the predictive lift for the model measured at that point. The average of all points constitutes the score for the model.

Back to Top

Lift Chart for Model with No Target Value

If you do not specify the state of the predictable column, you create the type of chart shown in the following diagram. This chart shows how the model performs for all states of the predictable attribute. For example, this chart would tell you how well the model predicts both customers who are likely to buy a bike, and those who are unlikely to buy a bike.

The x-axis is the same as in the chart with the predictable column specified, but the y-axis now represents the

percentage of predictions that are correct. Therefore, the ideal line is the diagonal line, which shows that at 50 percent of the data, the model correctly predicts 50% of the cases, the maximum that can be expected.

You can click in the chart to move the vertical gray bar, and the **Mining Legend** displays the percentage of cases overall, and the percentage of cases that were predicted correctly. For example, if you position the gray slider bar at the 50 percent mark, the **Mining Legend** displays the following accuracy scores. These figures are based on the TM_Decision Tree model created in the Basic Data Mining Tutorial.

SERIES, MODEL	SCORE	TARGET POPULATION	PREDICT PROBABILITY
TM_Decision Tree	0.77	40.50%	72.91%
Ideal model		50.00%	

This table tells you that, at 50 percent of the population, the model that you created correctly predicts 40 percent of the cases. You might consider this a reasonably accurate model. However, remember that this particular model predicts all values of the predictable attribute. Therefore, the model might be accurate in predicting that 90 percent of customers will not buy a bike.

Back to Top

Restrictions on Lift Charts

Lift charts require that the predictable attribute be a discrete value. In other words, you cannot use lift charts to measure the accuracy of models that predict continuous numeric values.

The prediction accuracy for all discrete values of the predictable attribute is shown in a single line. If you want to see prediction accuracy lines for any individual value of the predictable attribute, you must create a separate lift chart for each targeted value.

You can add multiple models to a lift chart, as long as the models all have the same predictable attribute. Models that do not share the attribute will be unavailable for selection in the **Input** tab.

You cannot display time series models in a lift chart or profit chart. A common practice for measuring the accuracy of time series predictions is to reserve a portion of historical data and compare that data to the predictions. For more information, see Microsoft Time Series Algorithm.

Related Content

Back to Top

See Also

Testing and Validation (Data Mining)

Profit Chart (Analysis Services - Data Mining)

11/27/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A profit chart displays the estimated profitability associated with using a mining model. For example, let's assume your model predicts which customers a company should contact in a business scenario. In that case, you would add to the profit chart information about the cost of conducting the targeted mailing campaign. Then, in the completed chart, you can see the estimated profit if customers are correctly targeted, as compared to randomly contacting customers.

Build a Profit Chart

A profit chart is similar to a lift chart. You start by creating a lift chart, and then add in the cost and profit information.

To build a profit chart, you must have an existing model.

For this example, we used the Targeted Mailing decision tree model. The model identifies customers who are likely to buy a bike. You can apply the **Profit Chart** to determine how many of your customers to target to maximize your profit.

If you don't have the sample model, you can create it using the Basic Data Mining Tutorial.

1. Open the mining accuracy chart builder.
   - In SQL Server Management Studio, right-click the model, and select **View Lift Chart**.
   - In SQL Server Data Tools, open the project in which you created the model, and click the **Mining Accuracy Chart** tab.
2. In **the Input Selection** tab, select the model and choose the predictable attribute value.

   For this particular scenario, you are interested only in the profitability of accurately predicting one value: [Bike Buyer] =1.

   However, there are other scenarios in which you are equally interested in predicting false values correctly. For example, the cost of a false positive on a medical diagnostic test can be significant and needs to be factored into the profitability of the prediction, as does the cost of false negatives. In such scenarios, you would measure all outcomes.
3. Select a data set for testing. For this example, select the testing data set.
4. Now click the **Lift Chart** tab.

   A lift chart is automatically generated.
5. To change this to a profit chart, select **Profit Chart** from the **Chart type** list.
6. As soon as you choose profit chart as the chart type, the **Profit Chart Setting** dialog box automatically opens.

   This dialog box helps you specify the costs and benefits associated with a targeted mailing campaign. For the chart shown in these examples, we used the following values:

SETTING	VALUE	COMMENTS
Population	20,000	Set the value for the total target population Your database might contain many customers, but to save on mailing expenses you might choose to target only the 20,000 customers who are most likely to respond. You can get this list by running a prediction query and sorting by the probability output by the predictive model.
Fixed cost	500	Enter the one-time cost of setting up a targeted mailing campaign for 20,000 people. This might include printing, or the cost of setting up an e-mail campaign.
Individual cost	3	Enter the per-unit cost for the targeted mailing campaign. This amount will be multiplied by a number equal to or less than 20000, depending on how many customers the model predicts are good prospects.
Revenue per individual	400	Enter a value that represents the amount of profit or income that can be expected from a successful result. In this case, we'll assume that mailing a catalog results in purchase of accessories or bikes averaging $400. This amount will be used to project the total profit associated with high probability cases.

7. After you have set the required parameters, click **OK**.
8. The chart updates to show the profit curve.

Understanding the Profit Chart

The following diagram shows the chart that was based on these parameters. The Y-axis of the chart represents the profit, while the X-axis represents the percentage of the customers who were contacted by the targeted mailing campaign.

As shown here, you can use a profit chart to compare multiple models, as long as they all predict the same discrete attribute.

Notice the gray vertical line in the chart. As you click and drag the line, the ToolTip display the percentage of the target population that is included under the curve at that point.

The **Mining Legend** is also updated as you drag the line, to display the percentage value, a profit score, and the predict probability that is associated with the population percentage at the vertical gray line.

For example, if you were using this model to decide who to send your promotional material to, you might decide to target 25% of the population, based on the predict probabilities, However, the area under the profit curve of the chart is greatest between 40 and 70 percent, indicating that by mailing to more people, you can maximize your return, even if a smaller overall percentage responds.

Saving Charts

When you create an accuracy chart or profit chart, no objects are created on the server. Instead, queries are executed against an existing model and the results are rendered in the viewer. If you need to save the results, you must copy either the chart or the results to Excel or another file.

Related Content

The following topics contain more information about how you can build and use accuracy charts.

TOPICS	LINKS
Provides a walkthrough of how to create a lift chart for the Targeted Mailing model.	Basic Data Mining Tutorial Testing Accuracy with Lift Charts (Basic Data Mining Tutorial)
Explains related chart types.	Lift Chart (Analysis Services - Data Mining) Classification Matrix (Analysis Services - Data Mining) Scatter Plot (Analysis Services - Data Mining)
Describes cross-validation for mining models and mining structures.	Cross-Validation (Analysis Services - Data Mining)
Describes steps for creating lift charts and other accuracy charts.	Testing and Validation Tasks and How-tos (Data Mining)

See Also

Testing and Validation (Data Mining)
Testing Accuracy with Lift Charts (Basic Data Mining Tutorial)

Classification Matrix (Analysis Services - Data Mining)

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A *classification matrix* sorts all cases from the model into categories, by determining whether the predicted value matched the actual value. All the cases in each category are then counted, and the totals are displayed in the matrix. The classification matrix is a standard tool for evaluation of statistical models and is sometimes referred to as a *confusion matrix*.

The chart that is created when you choose the **Classification Matrix** option compares actual to predicted values for each predicted state that you specify. The rows in the matrix represent the predicted values for the model, whereas the columns represent the actual values. The categories used in analysis are *false positive*, *true positive*, *false negative*, and *true negative*

A classification matrix is an important tool for assessing the results of prediction because it makes it easy to understand and account for the effects of wrong predictions. By viewing the amount and percentages in each cell of this matrix, you can quickly see how often the model predicted accurately.

This section explains how to create a classification matrix and how to interpret the results.

Understanding the Classification Matrix

Consider the model that you created as part of the Basic Data Mining Tutorial. The [TM_DecisionTree] model is used to help create a targeted mailing campaign, and can be used to predict which customers are most likely to buy a bike. To test this expected usefulness of this model, you use a data set for which the values of the outcome attribute, [Bike Buyer], are already known. Typically, you would use the testing data set that you set aside when you created the mining structure that is used for training the model.

There are only two possible outcomes: yes (the customer is likely to buy a bike), and no (the customer will likely not purchase a bike). Therefore, the resulting classification matrix is relatively simple.

Interpreting the Results

The following table shows the classification matrix for the TM_DecisionTree model. Remember that for this predictable attribute, 0 means No and 1 means Yes.

PREDICTED	0 (ACTUAL)	1 (ACTUAL)
0	362	144
1	121	373

The first result cell, which contains the value 362, indicates the number of *true positives* for the value 0. Because 0 indicates that the customer did not purchase a bike, this statistic tells you that model predicted the correct value for non bike-buyers in 362 cases.

The cell directly underneath that one, which contains the value 121, tells you the number of *false positives*, or how many times the model predicted that someone would buy a bike when actually they did not.

The cell that contains the value 144 indicates the number of *false positives* for the value 1. Because 1 means that the customer did purchase a bike, this statistic tells you that in 144 cases, the model predicted someone would not

buy a bike when in fact they did.

Finally, the cell that contains the value 373 indicates the number of true positives for the target value of 1. In other words, in 373 cases the model correctly predicted that someone would buy a bike.

By summing the values in cells that are diagonally adjacent, you can determine the overall accuracy of the model. One diagonal tells you the total number of accurate predictions, and the other diagonal tells you the total number of erroneous predictions.

Using Multiple Predictable Values

The [Bike Buyer] case is especially easy to interpret because there are only two possible values. When the predictable attribute has multiple possible values, the classification matrix adds a new column for each possible actual value and then counts the number of matches for each predicted value. The following table shows the results on a different model, where three values (0, 1, 2) are possible.

PREDICTED	0 (ACTUAL)	1 (ACTUAL)	2 (ACTUAL)
0	111	3	5
1	2	123	17
2	19	0	20

Although the addition of more columns makes the report look more complex, the additional detail can be very useful when you want to assess the cumulative cost of making the wrong prediction. To create sums on the diagonals or to compare the results for different combinations of rows, you can click the **Copy** button provided in the **Classification Matrix** tab and paste the report into Excel. Alternatively, you can use a client such as the Data Mining Client for Excel, which supports SQL Server 2005 (9.x) and later versions, to create a classification report directly in Excel that includes both counts and percentages. For more information, see SQL Server Data Mining.

Restrictions on the Classification Matrix

A classification matrix can be used only with discrete predictable attributes.

Although you can add multiple models when selecting models on the **Input Selection** tab of the **Mining Accuracy Chart** designer, the **Classification Matrix** tab will display a separate matrix for each model.

Related Content

The following topics contain more information about how you can build and use classification matrices and other charts.

TOPICS	LINKS
Provides a walkthrough of how to create a lift chart for the Targeted Mailing model.	Basic Data Mining Tutorial Testing Accuracy with Lift Charts (Basic Data Mining Tutorial)
Explains related chart types.	Lift Chart (Analysis Services - Data Mining) Profit Chart (Analysis Services - Data Mining) Scatter Plot (Analysis Services - Data Mining)

TOPICS	LINKS
Describes uses of cross-validation for mining models and mining structures.	Cross-Validation (Analysis Services - Data Mining)
Describes steps for creating lift charts and other accuracy charts.	Testing and Validation Tasks and How-tos (Data Mining)

See Also

Testing and Validation (Data Mining)

Scatter Plot (Analysis Services - Data Mining)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A *scatter plot* graphs the actual values in your data against the values predicted by the model. The scatter plot displays the actual values along the X-axis, and displays the predicted values along the Y-axis. It also displays a line that illustrates the perfect prediction, where the predicted value exactly matches the actual value. The distance of a point from this ideal 45-degree angle line indicates how well or how poorly the prediction performed.

Understanding the Scatter Plot

Consider a model in which the marketing department predicts daily sales based on the number of clicks on a link sent in a promotional e-mail. Because both the number of clicks and the amount of sales are continuous numeric values, you can graph the number of clicks as the independent variable and the sales as the dependent variable. When you do so, the straight line shows the expected linear relationship, and the points scattered around that line show how the actual data diverges from the expected. This analysis tells you at a glance how closely a set of results is correlated with a particular input, and how much variation there is from the ideal model

Interpreting the Results

The following diagram shows an example of a scatter plot, created for the scenario just described.

You can pause the mouse on any point scattered around the line to view the predicted and actual values in a tooltip. There is no **Mining Legend** for a scatter plot; however, the chart itself contains a legend that displays the score associated with the model. For more information about interpreting the score, see Mining Model Content for Linear Regression Models (Analysis Services - Data Mining).

You can copy the visual representation of the chart to the Clipboard, but not the underlying data or the formula. If you want to view the regression formula for the line, you can create a content query against the model. For more information, see Linear Regression Model Query Examples.

Restrictions on Scatter Plots

A scatter plot can only be created if the model that you choose on the **Input Selection** tab contains a continuous predictable attribute. You do not have to make any additional selections; the scatter plot chart type is automatically displayed in the **Lift Chart** tab based on model and attribute type.

Although time series models predict continuous numbers, you cannot measure the accuracy of a time series model by using a scatter plot. There are other methods that you can use, such as reserving a portion of the historical data. For more information, see Time Series Model Query Examples.

Related Content

The following topics contain more information about how you can build and use scatter plots and related accuracy charts.

TOPICS	LINKS
Provides a walkthrough of how to create a lift chart for the Targeted Mailing model.	Basic Data Mining Tutorial Testing Accuracy with Lift Charts (Basic Data Mining Tutorial)
Explains related chart types.	Lift Chart (Analysis Services - Data Mining) Profit Chart (Analysis Services - Data Mining) Classification Matrix (Analysis Services - Data Mining)
Describes uses of cross-validation for mining models and mining structures.	Cross-Validation (Analysis Services - Data Mining)
Describes steps for creating lift charts and other accuracy charts.	Testing and Validation Tasks and How-tos (Data Mining)

See Also

Testing and Validation (Data Mining)

Cross-Validation (Analysis Services - Data Mining)

11/28/2018 • 11 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Cross-validation is a standard tool in analytics and is an important feature for helping you develop and fine-tune data mining models. You use cross-validation after you have created a mining structure and related mining models to ascertain the validity of the model. Cross-validation has the following applications:

- Validating the robustness of a particular mining model.
- Evaluating multiple models from a single statement.
- Building multiple models and then identifying the best model based on statistics.

This section describes how to use the cross-validation features provided for data mining, and how to interpret the results of cross-validation for either a single model or for multiple models based on a single data set.

Overview of Cross-Validation Process

Cross-validation consists of two phases, training and result generation. These phases include the following steps:

- You select a target mining structure.
- You specify the models you want to test. This step is optional; you can test just the mining structure as well.
- You specify the parameters for testing the trained models.
  - The predictable attribute, predicted value, and accuracy threshold.
  - The number of folds into which to partition the structure or model data.
- Analysis Services creates and trains as many models as there are folds.
- Analysis Services returns a set of accuracy metrics for each fold in each model, or for the data set as a whole.

Configuring Cross-Validation

You can customize the way that cross-validation works to control the number of cross-sections, the models that are tested, and the accuracy bar for predictions. If you use the cross-validation stored procedures, you can also specify the data set that is used for validating the models. This wealth of choices means that you can easily produce many sets of different results that must then be compared and analyzed.

This section provides information to help you configure cross-validation appropriately.

Setting the Number of Partitions

When you specify the number of partitions, you determine how many temporary models will be created. For each partition, a cross-section of the data is flagged for use as the test set, and a new model is created by training on the remaining data not in the partition. This process is repeated until Analysis Services has created and tested the specified number of models. The data that you specified as being available for cross-validation is distributed evenly among all partitions.

The example in the diagram illustrates the usage of data if three folds are specified.

In the scenario in the diagram, the mining structure contains a holdout data set that is used for testing, but the test data set has not been included for cross-validation. As a result, all the data in the training data set, 70 percent of the data in the mining structure, is used for cross-validation. The cross-validation report shows the total number of cases used in each partition.

You can also specify the amount of data that is used during cross-validation, by specifying the number of overall cases to use. The cases are distributed evenly across all folds.

For mining structures stored in an instance of SQL Server Analysis Services, the maximum value that you can set for the number of folds is 256, or the number of cases, whichever is less. If you are using a session mining structure, the maximum number of folds is 10.

> **NOTE**
> As you increase the number of folds, the time required to perform cross-validation increases accordingly, because a model must be generated and tested for each fold. You may experience performance problems if the number of folds is too high.

Setting the Accuracy Threshold

The state threshold lets you set the accuracy bar for predictions. For each case, the model calculates the *predict probability*, meaning the probability that the predicted state is correct. If the predict probability exceeds the accuracy bar, the prediction is counted as correct; if not, the prediction is counted as incorrect. You control this value by setting **State Threshold** to a number between 0.0 and 1.0, where numbers closer to 1 indicate a strong level of confidence in the predictions, and numbers closer to 0 indicate that the prediction is less likely to be true. The default value for state threshold is NULL, which means that the predicted state with the highest probability is considered the target value.

You should be aware that the setting for state threshold affects measures of model accuracy. For example, assume that you have three models that you want to test. All are based on the same mining structure and all predict the column [Bike Buyer]. Moreover, you want to predict a single value of 1, meaning "yes, will buy." The three models return predictions with predict probabilities of 0.05, 0.15, and 0.8. If you set the state threshold to 0.10, two of the predictions are counted as correct. If you set the state threshold to 0.5, only one model is counted as having returned a correct prediction. If you use the default value, null, the most probable prediction is counted as correct. In this case, all three predictions would be counted as correct.

> **NOTE**
> You can set a value of 0.0 for the threshold, but the value is meaningless, because every prediction will be counted as correct, even those with zero probability. Be careful not to accidentally set **State Threshold** to 0.0.

Choosing Models and Columns to Validate

When you use the **Cross Validation** tab in Data Mining Designer, you must first select the predictable column from a list. Typically, a mining structure can support many mining models, not all of which use the same predictable column. When you run cross-validation, only those models that use the same predictable column can be included in the report.

To choose a predictable attribute, click **Target Attribute** and select the column from the list. If the target attribute is a nested column, or a column in a nested table, you must type the name of the nested column using the format <Nested Table Name>(key).<Nested Column>. If the only column used from the nested table is the key column, you can use <Nested Table Name>(key).

After you select the predictable attribute, Analysis Services automatically tests all models that use the same predictable attribute. If the target attribute contains discrete values, after you have selected the predictable column, you can optionally type a target state, if there is a specific value that you want to predict.

The selection of the target state affects the measures that are returned. If you specify a target attribute-that is, a column name-and do not pick a specific value that you want the model to predict, by default the model will be evaluated on its prediction of the most probable state.

When you use cross-validation with clustering models, there is no predictable column; instead, you select **#Cluster** from the list in the **Target Attribute** list box. After you have selected this option, other options that are not relevant to clustering models, such as **Target State**, are disabled. Analysis Services will then test all clustering models that are associated with the mining structure.

Tools for Cross-Validation

You can use cross-validation from the Data Mining Designer, or you can perform cross-validation by running stored procedures.

If you use the Data Mining Designer tools to perform cross-validation, you can configure the training and accuracy results parameters in a single dialog box. This makes it easier to set up and view results. You can measure the accuracy of all mining models that are related to a single mining structure and then immediately view the results in an HTML report. However, the stored procedures offer some advantages, such as added customizations and the ability to script the process.

Cross-Validation in Data Mining Designer

You can perform cross-validation by using the **Cross-Validation** tab of the Mining Accuracy Chart view in either SQL Server Management Studio or SQL Server Development Studio.

To see an example of how to create a cross-validation report using the user interface, see Create a Cross-Validation Report.

Cross-Validation Stored Procedures

For advanced users, cross-validation is also available in the form of fully parameterized system stored procedures. You can run the stored procedures by connecting to an instance from SQL Server Management Studio, or from any managed code application.

The stored procedures are grouped by mining model type. One set of stored procedures works with clustering models only. The other set of stored procedures works with other mining models.

For each type of mining model, clustered or non-clustered, the stored procedures perform cross-validation in two separate phases.

Partition data and generate metrics for partitions

For the first phase, you call a system stored procedure that creates as many partitions as you specify within the data set, and returns accuracy results for each partition. For each metric, Analysis Services then calculates the mean and standard deviation for the partitions.

- SystemGetCrossValidationResults (Analysis Services - Data Mining)
- SystemGetClusterCrossValidationResults (Analysis Services - Data Mining)

Generate metrics for entire data set

In the second phase, you call a different set of stored procedures. These stored procedures do not partition the data set, but generate accuracy results for the specified data set as a whole. If you have already partitioned and processed a mining structure, you can call this second set of stored procedures to get just the results.

- SystemGetAccuracyResults (Analysis Services - Data Mining)
- SystemGetClusterAccuracyResults (Analysis Services - Data Mining)

Defining the Testing Data

When you run the cross-validation stored procedures that calculate accuracy (SystemGetAccuracyResults or SystemGetClusterAccuracyResults), you can specify the source of the data that is used for testing during cross-validation. This option is not available in the user interface.

You can specify as a testing data source any of the following options:

- Use only the training data.
- Include an existing testing data set.
- Use only the testing data set.
- Apply existing filters to each model.
- Any combination of the training set, testing set, and model filters.

To specify a testing data source, you provide an integer value for the **DataSet** parameter of the stored procedure. For a list of the argument values, see the Remarks section of the relevant stored procedure reference topic.

If you perform cross-validation by using the **Cross-Validation** report in the Data Mining Designer, you cannot change the data set that is used. By default, the training cases for each model are used. If a filter is associated with a model, the filter is applied.

Results of Cross-Validation

If you use the Data Mining Designer, these results are displayed in a grid-like Web viewer. If you use the cross-validation stored procedures, these same results are returned as a table.

The report contains two types of measures: aggregates that indicate the variability of the data set when divided into folds, and model-specific measures of accuracy for each fold. The following topics provide more information about these metrics:

Cross-Validation Formulas

Lists all the measures by test type. Describes in general how the measures can be interpreted.

Measures in the Cross-Validation Report

Describes the formulas for calculating each measure, and lists the type of attribute that each measure can be applied to.

Restrictions on Cross-Validation

If you perform cross-validation by using the cross-validation report in SQL Server Development Studio, there are some limitations on the models that you can test and the parameters you can set.

- By default, all models associated with the selected mining structure are cross-validated. You cannot specify the model or a list of models.
- Cross-validation is not supported for models that are based on the Microsoft Time Series algorithm or the Microsoft Sequence Clustering algorithm.
- The report cannot be created if your mining structure does not contain any models that can be tested by cross-validation.
- If the mining structure contains both clustering and non-clustering models and you do not choose the **#Cluster** option, results for both types of models are displayed in the same report, even though the attribute, state, and threshold settings might not be appropriate for the clustering models.
- Some parameter values are restricted. For example, a warning is displayed if the number of folds is more than 10, because generating so many models might cause the report to display slowly.

If you are testing multiple mining models, and the models have filters, each model is filtered separately. You cannot add a filter to a model or change the filter for a model during cross-validation.

Because cross-validation by defaults tests all mining models that are associated with a structure, you may receive inconsistent results if some models have a filter and others do not. To ensure that you compare only those models that have the same filter, you should use the stored procedures and specify a list of mining models. Or, use only the mining structure test set with no filters to ensure that a consistent set of data is used for all models.

If you perform cross-validation by using the stored procedures, you have the additional option of choosing the source of testing data. If you perform cross-validation by using the Data Mining Designer, you must use the testing data set that is associated with the model or structure, if any. Generally, if you want to specify advanced settings, you should use the cross-validation stored procedures.

Cross-validation cannot be used with time series or sequence clustering models. Specifically, no model that contains a KEY TIME column or a KEY SEQUENCE column can be included in cross-validation.

Related Content

See the following topics for more information about cross-validation, or information about related methods for testing mining models, such as accuracy charts.

TOPICS	LINKS
Describes how to set cross-validation parameters in SQL Server Development Studio.	Cross-Validation Tab (Mining Accuracy Chart View)
Describes the metrics that are provided by cross-validation	Cross-Validation Formulas
Explains the cross-validation report format and defines the statistical measures provided for each model type.	Measures in the Cross-Validation Report
Lists the stored procedures for computing cross-validation statistics.	Data Mining Stored Procedures (Analysis Services - Data Mining)
Describes how to create a testing data set for mining structures and related models.	Training and Testing Data Sets

TOPICS	LINKS
See examples of other accuracy chart types.	Classification Matrix (Analysis Services - Data Mining) Lift Chart (Analysis Services - Data Mining) Profit Chart (Analysis Services - Data Mining) Scatter Plot (Analysis Services - Data Mining)
Describes steps for creating various accuracy charts.	Testing and Validation Tasks and How-tos (Data Mining)

See Also

Testing and Validation (Data Mining)

Measures in the Cross-Validation Report

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

During cross-validation, Analysis Services divides the data in a mining structure into multiple cross-sections and then iteratively tests the structure and any associated mining models. Based on this analysis, it outputs a set of standard accuracy measures for the structure and each model.

The report contains some basic information about the number of folds in the data and the amount of data in each fold, and a set of general metrics that describe data distribution. By comparing the general metrics for each cross-section, you can assess the reliability of the structure or model.

Analysis Services also displays a set of detailed measures for mining models. These measures depend on the model type and on the type of attribute that is being analyzed: for example, whether it is discrete or continuous.

This section provides a list of the measures that are contained in the **Cross-Validation** report, and what they mean. For details on how each measure is calculated, see Cross-Validation Formulas.

List of Measures in the Cross-Validation Report

The following table lists the measures that appear in the cross-validation report. The measures are grouped by *test type*, which is provided in the left-hand column of the following table. The right-hand column lists the name of the measure as it appears in the report, and provides a brief explanation of what it means.

TEST TYPE	MEASURES AND DESCRIPTIONS
Clustering	Measures that apply to clustering models
	Case likelihood: This measure usually indicates how likely it is that a case belongs to a particular cluster. For cross-validation, the scores are summed and then divided by the number of cases, so here the score is an average case likelihood.
Classification	Measures that apply to classification models
	True Positive/True Negative/False Positive/False Negative: Count of rows or values in the partition where the predicted state matches the target state, and the predict probability is greater than the specified threshold. Cases that have missing values for the target attribute are excluded, meaning the counts of all values might not add up.
	Pass/Fail: Count of rows or values in the partition where the predicted state matches the target state, and where the predict probability value is greater than 0.
Likelihood	Likelihood measures apply to multiple model types.

TEST TYPE	MEASURES AND DESCRIPTIONS
	Lift: The ratio of the actual prediction probability to the marginal probability in the test cases. Rows that have missing values for the target attribute are excluded. This measure generally shows how much the probability of the target outcome improves when the model is used.
	Root Mean Square Error: Square root of the mean error for all partition cases, divided by the number of cases in the partition, excluding rows that have missing values for the target attribute. RMSE is a popular estimator for predictive models. The score averages the residuals for each case to yield a single indicator of model error.
	Log score: The logarithm of the actual probability for each case, summed, and then divided by the number of rows in the input dataset, excluding rows that have missing values for the target attribute. Because probability is represented as a decimal fraction, log scores are always negative numbers. A number closer to 0 is a better score. Whereas raw scores can have very irregular or skewed distributions, a log score is similar to a percentage.
Estimation	Measures that apply only to estimation models, which predict a continuous numeric attribute.
	Root Mean Square Error: Average error when the predicted value is compared to the actual value. RMSE is a popular estimator for predictive models. The score averages the residuals for each case to yield a single indicator of model error.
	Mean Absolute Error: Average error when predicted values are compared to actual values, calculated as the mean of the absolute sum of errors. Mean absolute error is useful for understanding how close overall the predictions were to actual values. A smaller score means predictions were more accurate.
	Log Score: The logarithm of the actual probability for each case, summed, and then divided by the number of rows in the input dataset, excluding rows that have missing values for the target attribute. Because probability is represented as a decimal fraction, log scores are always negative numbers. A number closer to 0 is a better score. Whereas raw scores can have very irregular or skewed distributions, a log score is similar to a percentage.

TEST TYPE	MEASURES AND DESCRIPTIONS
Aggregates	Aggregate measures provide an indication of the variance in the results for each partition.
	Mean: Average of the partition values for a particular measure.
	Standard Deviation: Average of the deviation from the mean for a specific measure, across all the partitions in a model. For cross-validation, a higher value for this score implies substantial variation between the folds.

See Also

Testing and Validation (Data Mining)

Cross-Validation Formulas

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you generate a cross-validation report, it contains accuracy measures for each model, depending on the type of mining model (that is, the algorithm that was used to create the model), the data type of the predictable attribute, and the predictable attribute value, if any.

This section lists the measures used in the cross-validation report and describes the method of calculation.

For a breakdown of the accuracy measures by model type, see Measures in the Cross-Validation Report.

Formulas used for Cross-Validation Measures

> **NOTE**
>
> **Important:** These measures of accuracy are computed for each target attribute. For each attribute you can specify or omit a target value. If a case in the data set does not have any value for the target attribute, the case is treated as having a special value called the *missing value*. Rows that have missing values are not counted when computing the accuracy measure for a particular target attribute. Note that because the scores are computed for each attribute individually, if values are present for the target attribute but missing for other attributes, it does not affect the score for the target attribute.

MEASURE	APPLIES TO	IMPLEMENTATION
True positive	Discrete attribute, value is specified	Count of cases that meet these conditions: Case contains the target value. Model predicted that case contains the target value.
True Negative	Discrete attribute, value is specified	Count of cases that meet these conditions: Case does not contain the target value. Model predicted that case does not contain the target value.
False positive	Discrete attribute, value is specified	Count of cases that meet these conditions: Actual value is equal to target value. Model predicted that case contains the target value.

MEASURE	APPLIES TO	IMPLEMENTATION
False Negative	Discrete attribute, value is specified	Count of cases that meet these conditions: Actual value not equal to target value. Model predicted that case does not contain the target value.
Pass/fail	Discrete attribute, no specified target	Count of cases that meet these conditions: Pass if the predicted state with the highest probability is the same as the input state and probability is greater than the value of **State Threshold**. Otherwise, fail.
Lift	Discrete attribute. Target value can be specified but is not required.	The mean log likelihood for all rows with values for the target attribute, where log likelihood for each case is calculated as Log(ActualProbability/MarginalProbability). To compute the mean, the sum of the log likelihood values is divided by the number of rows in the input dataset, excluding rows with missing values for the target attribute. Lift can be either a negative or positive value. A positive value means an effective model that outperforms the random guess.
Log score	Discrete attribute. Target value can be specified but is not required.	Log of the actual probability for each case, summed, and then divided by the number of rows in the input dataset, excluding rows with missing values for the target attribute. Because probability is represented as a decimal fraction, log scores are always negative numbers. A score that is closer to 0 is a better score.
Case likelihood	Cluster	Sum of the cluster likelihood scores for all cases, divided by the number of cases in the partition, excluding rows with missing values for the target attribute.
Mean absolute error	Continuous attribute	Sum of the absolute error for all cases in the partition, divided by the number of cases in the partition.
Root mean square error	Continuous attribute	Square root of the mean squared error for the partition.

MEASURE	APPLIES TO	IMPLEMENTATION
Root mean squared error	Discrete attribute. Target value can be specified but is not required.	Square root of the mean of the squares of complement of the probability score, divided by the number of cases in the partition, excluding rows with missing values for the target attribute.
Root mean squared error	Discrete attribute, no specified target.	Square root of the mean of the squares of complement of the probability score, divided by the number of cases in the partition, excluding cases with missing values for the target attribute.

See Also

Testing and Validation (Data Mining)
Cross-Validation (Analysis Services - Data Mining)

Testing and Validation Tasks and How-tos (Data Mining)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can use the **Mining Accuracy Chart** tab of Data Mining Designer in SQL Server Data Tools (SSDT) to compare the predictive accuracy of the mining models in your mining structure.

You can create four kinds of charts:

- Lift chart
- Profit chart
- Classification matrix
- Cross-validation report

The first three charts use the **Input Selection** tab to define the data that is used for generating the chart.

The Cross-validation chart is created by using additional inputs, available on the **Cross-Validation** tab. For more information, see Cross-Validation (Analysis Services - Data Mining).

For more information about how to use the mining accuracy chart, see Testing and Validation (Data Mining).

In This Section

- Create a Lift Chart, Profit Chart, or Classification Matrix
- Create a Cross-Validation Report
- Choose and Map Model Testing Data
- Apply Filters to Model Testing Data
- Choose the Column to Use for Testing a Mining Model
- Using Nested Table Data as an Input for an Accuracy Chart

Create a Lift Chart, Profit Chart, or Classification Matrix

11/28/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can create an accuracy chart for an Analysis Services data mining model in five basic steps:

- Select the mining structure that contains the mining models that you want to compare.
- Select the mining models to add to the chart.
- Specify a source of testing data to use in generating the chart.
- Choose the chart type.
- Configure the chart options.

These basic steps are the same for the lift chart, profit chart, and classification matrix. The following procedures outline the steps to configure the basic chart options for these chart types. For information about how to create a cross-validation report, see Measures in the Cross-Validation Report.

Open the mining structure in the Accuracy Chart Designer

1. Open the Data Mining Designer in SQL Server Data Tools (SSDT).
2. In Solution Explorer, double-click the structure that contains the mining model or models.
3. Click the **Mining Accuracy Chart** tab.

Select mining models for inclusion in the chart

1. On the **Mining Accuracy Chart** tab of Data Mining Designer in SQL Server Data Tools (SSDT), click the **Input Selection** tab.

   The list displays all models in the current structure that have the same predictable attribute.

2. Select the **Show box** for each model that you want to include in the chart.
3. Click the **Predictable Column Name** text box, and select the name of a predictable column from the list. All models that you put in one chart must have the same predictable column.
4. If you compare two models and the predictable columns have different values or different data types, clear the **Synchonize prediction columns and values** box to force a comparison.

   > **NOTE**
   >
   > If the **Synchonize prediction columns and values** box is selected, Analysis Services analyzes the data in the predictable columns of the model and the test data, and attempts to find the best match. Therefore, do not clear the box unless absolutely necessary to force a comparison of the columns.

5. Click the **Predict Value** text box, and select a value from the list. If the predictable column is a continuous data type, you must type a value in the text box.

   For more information, see Choose the Column to Use for Testing a Mining Model.

Select testing data

1. On the **Input Selection** tab of the **Mining Accuracy Chart** tab, specify the source of the data that you will use to generate the chart by selecting one of the options in the group, **Select data set to be used for accuracy chart**.

   - Select the option, **Use Mining Model test cases**, if you want to use the subset of cases that is defined by the intersection of the mining structure test cases and any filters that may have been applied during model creation.
   - Select the option, **Use mining structure test cases**, to use the full set of testing cases that were defined as part of the mining structures holdout data set.
   - Select the option, **Specify a different data set**, if you want to use external data. The data set must be available as a data source view. Click the browse (**...**) button to choose the data tables to use for the accuracy chart. For more information, see Choose and Map Model Testing Data.

     If you are using an external data set, you can optionally filter the input data set. For more information, see Apply Filters to Model Testing Data.

> **NOTE**
>
> You cannot create a filter on the model test cases or the mining structure test cases on the **Input Selection** tab. To create a filter on the mining model, modify the Filter property of the model. For more information, see Apply a Filter to a Mining Model.

Configure chart settings and generate the chart

1. In the **Mining Accuracy Chart** tab, click the tab for the chart you want to create.
2. For a **lift chart**, click the **Lift Chart** tab. The chart is automatically generated based on the model, predictable attributes, and input data that you just selected.
3. For a **classification matrix**, click the **Classification Matrix** tab. No further settings are needed; the chart is automatically generated based on the input data and model that you selected.
4. For a **profit chart**, first click the **Lift Chart** tab. Then, from the **Chart type** drop-down list, select **Profit chart**.

   Enter the following settings in the **Profit Chart Settings** dialog box.

   Population
   The number of cases from the data set that you want to use when creating the lift chart.

   The model always chooses the cases in order of decreasing probability; that is, if you are assessing potential customers and you choose a number that represents only half the records in your customer database, the model will measure accuracy on the subset of cases that best fit your model.

   This is because when you use the model to generate a mailing or create a campaign, you will use the prediction probability associated with each case to target only the customers who have the highest probability of making a positive response.

   Fixed Cost
   The fixed cost that is associated with the business problem.

   If this were for a targeted mailing solution, the fixed cost might represent a printer setup fee that covers the initial cost of preparing the promotional mailing.

   This cost applies one time to the entire target population.

   Individual Cost

Costs that are in addition to the fixed cost, that can be associated with each customer contact. For example, you might enter the postage cost for a promotional mailing or the cost of making telephone calls.

This cost must be the same for the entire target population. Each value is multiplied by the number of cases that are targeted.

Revenue Per Individual
The amount of revenue that is associated with each successful sale.

See Also

Lift Chart (Analysis Services - Data Mining)
Classification Matrix (Analysis Services - Data Mining)

Create a Cross-Validation Report

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic walks you through creation of a cross-validation report using the Accuracy Chart tab in Data Mining Designer. For general information about what a cross-validation report looks like, and the statistical measures it includes, see Cross-Validation (Analysis Services - Data Mining).

A cross-validation report is fundamentally different from an accuracy chart, such as a lift chart or classification matrix.

- Cross-validation assesses the overall distribution of data that is used in a model or structure; therefore, you do not specify a testing data set. Cross-validation always uses only the original data that was used to train the model or the mining structure.
- Cross-validation can only be performed with respect to a single predictable outcome. If the structure supports models that have different predictable attributes, you must create separate reports for each predictable output.
- Only models that are related to the currently selected structure are available for cross-validation.
- If the structure that is currently selected supports a combination of clustering and non-clustering models, when you click **Get Results**, the cross-validation stored procedure will automatically load models that have the same predicted column, and ignore clustering models that do not share the same predictable attribute.
- You can create a cross-validation report on a clustering model that does not have a predictable attribute only if the mining structure does not support any other predictable attributes.

Select a mining structure

1. Open the Data Mining Designer in SQL Server Data Tools (SSDT).
2. In Solution Explorer, open the database that contains the structure or model for which you want to create a report.
3. Double-click the mining structure to open the structure and its related models in Data Mining Designer.
4. Click the **Mining Accuracy Chart** tab.
5. Click the **Cross Validation** tab.

Set cross-validation options

1. On the **Cross Validation** tab, for **Fold Count**, click the down arrow to select a number between 1 and 10. The default value is 10.

   The **Fold Count** represents the number of partitions that will be created within the original data set. If you set Fold Count to 1, the training set will be used without partitioning.

2. For **Target Attribute**, click the down arrow, and select a column from the list. If the model is a clustering model, select **#Cluster** to indicate that the model does not have a predictable attribute. Note that the value, **#Cluster**, is available only when the mining structure does not support other types of predictable attributes.

   You can select only one predictable attribute per report. By default, all related models that have the same predictable attribute are included in the report.

3. For **Max Cases**, type a number that is large enough to provide a representative sample of data when the data is split among the specified number of folds. If the number is greater than the count of cases in the model training set, all cases will be used.

   If the training data set is very large, setting the value for **Max Cases** limits the total number of cases processed, and lets the report finish faster. However, you should not set **Max Cases** too low or there may be insufficient data for cross-validation.

4. Optionally, for **Target State**, type the value of the predictable attribute that you want to model. For example, if the column [Bike Buyer] has two possible values, 1 (Yes) and 2 (No), you can enter the value 1 to assess the accuracy of the model for just the desired outcome.

   > **NOTE**
   >
   > If you do not enter a value, the **Target Threshold** option is not available, and the model is assessed for all possible values of the predictable attribute.

5. Optionally, for **Target Threshold**, type a decimal number between 0 and 1 to specify the minimum probability that a prediction must have to be counted as accurate.

   For additional tips about how to set probability thresholds, see Measures in the Cross-Validation Report.

6. Click **Get Results**.

Print the cross-validation report

1. Right-click the completed report on the **Cross Validation** tab.
2. In the shortcut menu, select **Print** or **Print Preview** to review the report first.

Create a copy of the report in Microsoft Excel

1. Right-click the completed report on the **Cross Validation** tab.
2. In the shortcut menu, select **Select All**.
3. Right-click the selected text, and select **Copy**.
4. Paste the selection into an open Excel workbook. If you use the **Paste** option, the report is pasted into Excel as HTML, which preserves row and column formatting. If you paste the report by using the **Paste Special** options for text or Unicode text, the report is pasted in row-delimited format.

See Also

Measures in the Cross-Validation Report

Choose and Map Model Testing Data

11/28/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

To create an accuracy chart in Analysis Services, you must choose the data that will be used to test the model, and map the data to the model.

By default, Analysis Services will use the mining model testing data, provided that you created a holdout data set when you built the mining structure. Creating a holdout test set is the easiest way to test models that are based on the same mining structure, because the column names and data types will always match the model, and you can be reasonably assured that the distribution of the data is similar. Also, the designer will automatically create the relationships between the input and model columns.

Alternatively, you can specify an external source of data. For external data, there are some additional requirements:

- The external data set must be defined as a data source view in an instance of Analysis Services.
- The external data set must at least contain one column that can be mapped to the predictable column in the mining model. You can choose to ignore some columns.
- You cannot add new columns or map columns in a different data source view. The data source view that you select must contain all the columns that you need for the prediction query.
- If the external column names exactly match those in the model, the designer will map them for you. If the mappings are wrong, you can change them, or delete and create new mappings for existing columns.
- If you use an external data source, you can apply filters to restrict the testing data to a relevant subset of cases.
- Even when you use the holdout test set, you should be aware that filters can cause differences between the testing data associated with a mining structure and the mining model test cases.

This topic describes how to choose and map the testing data:

Select input tables to test the accuracy of a mining model

Map model columns to the columns in the testing data

Change the way that columns in the testing data are mapped to the model

To select input tables to test the accuracy of a mining model

1. In Data Mining Designer in SQL Server Data Tools (SSDT), double-click the mining structure that contains the models you want to chart.
2. Select the **Mining Accuracy Chart** tab.
3. On the **Input Selection Tab** of the **Mining Accuracy Chart** view, select one of the following options:

   Use mining model test cases

   Use mining structure test cases

   Specify a different data set

4. If you selected **Specify a different data set**, optionally click **Open Filter Editor** to create filter conditions

on the input data set. Click **OK**.

5. Click the **Lift Chart** tab or the **Classification Matrix** tab to automatically build the chart by using the testing data you specified.

To map model columns to the columns in the testing data

1. Double-click the mining structure that contains the models you want to chart, to open the structure and models in Data Mining Designer.
2. Select the **Mining Accuracy Chart** tab, and then select the **Input Selection** tab.
3. In the **Input Selection** tab, under **Select data set to be used for Accuracy Chart**, select **Specify a different data set**.
4. Click the browse button **(...)** to open a dialog box and build the definition of the external data set.
5. In the **Select Mining Structure** dialog box, select the mining structure that contains the models you want to work with, and then click **OK**.
6. On the **Select Input Table(s)** table of the **Mining Accuracy Chart** tab, click **Select Case Table** to open the **Select Table** dialog box.
7. In the **Select Table** dialog box, select a data source from the **Data Source** list. Choose a table that contains the data that you want to use in the prediction queries to determine the accuracy of the models.
8. In the **Table/View Name** box, select the table that contains the data that you want to use to test the models.
9. Edit the mappings, if necessary. Columns in the mining structure are automatically mapped to the columns with the same name in the input table. To manually create mappings, click a column in the **Select Input Table(s)** table and drag it onto the corresponding column in the **Mining Structure** table. To delete a mapping, click the line that links the column in the **Mining Structure** table to the mapped column in the **Select Input Table(s)** table, and then press DELETE.
10. Click **OK**.

To modify the way input data is mapped to the model

1. In Data Mining Designer, double-click the structure that contains the models you want to chart.
2. Select the **Mining Accuracy Chart** tab.
3. Click the **Input Selection** tab.
4. In **Select data set to be used for Accuracy Chart**, select the option **Specify a different data set**.
5. Click the browse button **(...)** to open a dialog box and build the definition of the external data source.
6. In the **Specify Column Mapping** dialog box, click **Select Case Table**.
7. In the Select Table dialog box, Select a data source view from the list, and select the table that contains the case data. Click **OK**.
8. If the tables you need are not available, close the dialog box and create a new data source view that contains the table. For information about how to create a data source view, see Defining a Data Source View (Analysis Services).
9. If the mining model contains a nested table, click **Select Nested Table**, and select the nested table from the list of tables in the data source view. Click **OK**.

10. Select the join line of the mapping you want to modify, and select **Modify Connections**.

    The **Modify Mapping** dialog box opens. In the table in this dialog box, **Mining Structure Column** lists each column that the selected mining structure contains, and **Table Column** lists the columns from input tables that are mapped to columns in the mining structure.

11. Under **Table Column**, select the row that corresponds to the row under **Mining Structure Column** for which you want to modify a relationship. Select a new column from the list, or select the blank entry from the list to delete the column.

12. Click **OK**.

    The new column mappings are displayed in the **Specify Column Mapping** dialog box. You can remove a mapping by selecting the line between the columns and pressing the DELETE key. You can create a new connection by selecting a column in the **Mining Structure** table and dragging it to the corresponding column in the **SelectInput Table(s)** table.

See Also

Testing and Validation Tasks and How-tos (Data Mining)

Apply Filters to Model Testing Data

11/28/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you specify an external data source to use in testing a model, you can optionally apply a filter to restrict the input data. For example, you might want to test the model specifically for predictions on customers in a certain income range.

For example, in the Adventure Works targeted mailing scenario, you can create a filter expression like the following one on ProspectiveBuyer, which is the table that contains the testing data, and restrict testing cases by income range:

```
[YearlyIncome] = '50000'
```

The behavior of filters is slightly different, depending on whether you are filtering model training data or a testing data set:

- When you define a filter on a testing data set, you create a WHERE clause on the incoming data. If you are filtering an input data set used for evaluating a model, the filter expression is translated to a Transact-SQL statement and applied to the input table when the chart is created. As a result, the number of test cases can be greatly reduced.
- When you apply a filter to a mining model, the filter expression that you create is translated to a Data Mining Extensions (DMX) statement, and applied to the individual model. Therefore, when you apply a filter to a model, only a subset of the original data is used to train the model. This can cause problems if you filter the training model with one set of criteria, so that the model is tuned to a certain set of data, and then test the model with another set of criteria.
- If you defined a testing data set when you created the structure, the model cases used for training include only those cases that are in the mining structure training set **and** which meet the conditions of the filter. Thus, when you are testing a model and select the option **Use mining model test cases**, the testing cases will include only the cases that are in the mining structure test set and which meet the conditions of the filter. However, if you did not define a holdout data set, the model cases used for testing include all the cases in the data set that meet the filter conditions
- Filter conditions that you apply on a model also affect drillthrough queries on the model cases.

In summary, when you test multiple models, even if all the models are based on the same mining structure, you must be aware that the models potentially use different subsets of data for training and testing. This can have the following effects on accuracy charts:

- The total number of cases in the testing sets can vary among the models being tested.
- The percentages for each model may not align in the chart, if the models use different subsets of training data or testing data.

To determine whether a model contains a predefined filter that might affect results, you can look for the **Filter** property in the **Property** pane, or you can query the model by using the data mining schema rowsets. For example, the following query returns the filter text for the specified model:

```
SELECT [FILTER] FROM $system.DMSCHEMA_MINING_MODELS WHERE MODEL_NAME = 'name of model'
```

> **WARNING**
> If you want to remove the filter from an existing mining model, or change the filter conditions, you must reprocess the mining model.

For more information about the kinds of filters you can apply, and how filter expressions are evaluated, see Model Filter Syntax and Examples (Analysis Services - Data Mining).

Create a filter on external testing data

1. Double-click the mining structure that contains the model you want to test, to open Data Mining Designer.
2. Select the **Mining Accuracy Chart** tab, and then select the **Input Selection** tab.
3. On the **Input Selection** tab, under **Select data set to be used for Accuracy Chart**, select the option **Specify a different data set**.
4. Click the browse button **(...)** to open a dialog box and choose the external data set.
5. Choose the case table, and add a nested table if necessary. Map columns in the model to columns in the external data set as necessary. Close the **Specify Column Mapping** dialog box to save the source table definition.
6. Click **Open Filter Editor** to define a filter for the data set.

   The **Data Set Filter** dialog box opens. If the structure contains a nested table, you can create a filter in two parts. First, set conditions on the case table by using the **Data Set Filter** dialog box, and then set conditions on the nested rows by using the **Filter** dialog box.
7. In the **Data Set Filter** dialog box, click the top row in the grid, under **Mining Structure Column**, and select a table or column from the list.

   If the data source view contains multiple tables, or a nested table, you must first select the table name. Otherwise, you can select columns from the case table directly.

   Add a new row for each column that you want to filter.
8. Use **Operator**, and **Value** columns to define how the column is filtered.

   Note Type values without using quotation marks.
9. Click the **And/Or** text box and select a logical operator to define how multiple conditions are combine.
10. Optionally, click the browse button **(...)** at the right of the **Value** text box to open the **Filter** dialog box and set conditions on the nested table or on the individual case table columns.
11. Verify that the completed filter conditions are correct by viewing the text in the **Expression** pane.
12. Click **OK**.

    The filter condition is applied to the data source when you create the accuracy chart.

See Also

Choose and Map Model Testing Data
Using Nested Table Data as an Input for an Accuracy Chart
Choose an Accuracy Chart Type and Set Chart Options

Choose the Column to Use for Testing a Mining Model

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Before you can measure the accuracy of a mining model, you must decide which outcome it is that you want to assess. Most data mining models require that you choose at least one column to use as the predictable attribute when you create the model. Therefore, when you test the accuracy of the model, you generally must select that attribute to test.

The following list describes some additional considerations for choosing the predictable attribute to use in testing:

- Some types of data mining models can predict multiple attributes-such as neural networks, which can explore the relationships between many attributes.
- Other types of mining models-such as clustering models-do not necessarily have a predictable attribute. Clustering models cannot be tested unless they have a predictable attribute.
- To create a scatter plot or measure the accuracy of a regression model requires that you choose a continuous predictable attribute as the outcome. In that case, you cannot specify a target value. If you are creating anything other than a scatter plot, the underlying mining structure column must also have a content type of **Discrete** or **Discretized**.
- If you choose a discrete attribute as the predictable outcome, you can also specify a target value, or you can leave the **Predict Value** field blank. If you include a **Predict Value**, the chart will measure only the model's effectiveness at predicting the target value. If you do not specify a target outcome, the model is measured for its accuracy in predicting all outcomes.
- If you want to include multiple models and compare them in a single accuracy chart, all models must use the same predictable column.
- When you create a cross-validation report, Analysis Services will automatically analyze all models that have the same predictable attribute.
- When the option, **Synchronize Prediction columns and Values**, is selected, Analysis Services automatically chooses predictable columns that have the same names and matching data types. If your columns do not meet these criteria, you can turn off this option and manually choose a predictable column. You might need to do this if you are testing the model with an external data set that has different columns than the model. However, if you choose a column with the wrong the type of data you will either get an error or bad results.

Specify the outcome to predict

1. Double-click the mining structure to open it in Data Mining Designer.
2. Select the **Mining Accuracy Chart** tab.
3. Select the **Input Selection** tab.
4. On the **Input Selection** tab, under **Predictable Column Name**, select a predictable column for each model that you include in the chart.

   The mining model columns that are available in the **Predictable Column Name** box are only those with the usage type set to **Predict** or **Predict Only**.

5. If you want to determine the lift for a model, you must select a specific outcome value to measure, for by choosing from the **Predict Value** list.

See Also

Choose and Map Model Testing Data
Choose an Accuracy Chart Type and Set Chart Options

Choose an Accuracy Chart Type and Set Chart Options

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

SQL Server Analysis Services provides multiple methods for determining the validity of your mining models. The type of accuracy chart that you can create for each model or structure depends on these factors:

- The type of algorithm that was used to create the model
- The data type of the predictable attribute
- The number of predictable attributes to measure

This topic provides an overview of the different accuracy charts.

Note Charts and their definitions are not saved. If you close the window that contains a chart, you must create the chart again.

Accuracy Chart Types

Depending on the chart type that you choose, you have the ability to further configure options, to browse the chart, or copy the chart to the Clipboard and work with the data in Excel.

Lift chart

After you have configured the options for the models and the testing data, click the **Lift Chart** tab to view the results. You can also copy the chart to the Clipboard, or view details of individual trend lines or data points in the Mining Legend.

Profit Chart

After you have configured the options for the models and the testing data, click the **Lift Chart** tab, select **Profit Chart** from the **Chart Type** list to set profit chart options, and then click **OK** to view the results.

You can use the **Profit Chart Settings** dialog box as many times as you want to try different cost options and redisplay the chart. The Mining Legend contains detailed information about the estimated profit for each model. You can also copy the chart and the contents of the Mining Legend to the Clipboard to work with it in Excel.

Scatter Plot

If you have selected the appropriate type of model, when you click the **Lift Chart** tab, the chart type is automatically set to **Scatter Plot** and a scatter plot is displayed. No further configuration is possible. You can also copy the chart to the Clipboard and paste the chart as a graphic into Excel or another application.

Classification Matrix

For a classification matrix, use the **Input Selection** tab to choose the models and the testing data, and then click the **Classification Matrix** tab to view the results.

The contents of a classification matrix are the same for all model types, and cannot be configured. You can copy the data in the chart to the Clipboard and then work with it in Excel.

Cross-Validation Report

For a cross-validation report, after you have selected a mining structure or mining model in Solution Explorer, click the **Cross Validation** tab, configure all relevant options, and then click **Get Results** to generate the report. No further configuration is possible.

The format of the cross-validation report is the same for all model types, and cannot be configured. However, the content of the report differs depending on the type of model that you are analyzing, and the data type of the predictable attribute. You can also copy the results of the report to the Clipboard and work with the data in Excel.

Using Nested Table Data as an Input for an Accuracy Chart

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you test the accuracy of a mining model by using external data, if the mining model contains nested tables, the external data must also contain a case table and an associated nested table.

This topic describes how to work with nested tables used for model testing, how to map nested and case tables in the mode and in the external data, and how to apply a filter to a nested table.

When working with nested tables, keep in mind these tips:

- If you select the option **Use mining model test cases** or **Use mining structure test cases**, you do not need to specify a case table or nested table. With those options, the definition of the test data is stored in the mining structure and the test data is automatically selected when you create the accuracy chart.
- If a relationship already exists between the case and nested table in the data source, the columns in the mining structure are automatically mapped to the columns that have the same name in the input table.
- You cannot select a nested table until you have specified the case table. Also, you cannot specify a nested table as an input unless the mining model also uses a case table and nested table structure.

Use a nested table as input to an accuracy chart

1. Double-click the mining structure to open it in Data Mining Designer.
2. Select the **Mining Accuracy Chart** tab, and then select the **Input Selection** tab.
3. In **Select data set to be used for accuracy chart**, select the option **Specify a different data set**.
4. Click the browse button **(...)** to choose the external data set from a list of data source views on the current server.
5. Click **Select Case Table**. In the **Select Table** dialog box, choose the table from the data source view that contains the case data, and then click **OK**.
6. Click **Select Nested Table**. In the **Select Table** dialog box, select the table that contains the nested data, and then click **OK**.
7. Click **OK**.

   If you need to modify the relationship between the nested table and the case table, click **Modify Join** to open the **Create Relationship** dialog box.

See Also

Choose and Map Model Testing Data
Apply Filters to Model Testing Data

Data Mining Queries

5/16/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data mining queries are useful for many purposes. You can:

- Apply the model to new data, to make single or multiple predictions. You can provide input values as parameters, or in a batch.
- Get a statistical summary of the data used for training.
- Extract patterns and rules, or generate a profile of the typical case representing a pattern in the model.
- Extract regression formulas and other calculations that explain patterns.
- Get the cases that fit a particular pattern.
- Retrieve details about individual cases used in the model, including data not used in analysis.
- Retrain a model by adding new data, or perform cross-prediction.

This section provides an overview of the information you need to get started with data mining queries. It describes the types of queries you can create against data mining objects, introduces the query tools and query languages, and provides links to examples of queries that you can create against models that were built using the algorithms provided in SQL Server Data Mining.

Understanding Data Mining Queries

Query Tools and Interfaces

Queries for Different Model Types

Requirements

Understanding Data Mining Queries

Analysis Services Data Mining supports the following types of queries:

- Prediction Queries (Data Mining)

  Queries that make inferences based on patterns in the model, and from input data.

- Content Queries (Data Mining)

  Queries that return metadata, statistics, and other information about the model itself.

- Drillthrough Queries (Data Mining)

  Queries that can retrieve the underlying case data for the model, or even data from the structure that was not used in the model.

- Data Definition Queries (Data Mining)

  Queries that do not return information from the model, but rather are used to build models and structures or to update the data in a model or structure.

Before you create queries, we recommend that you familiarize yourself with the differences between models created with each of the data mining algorithms provided by SQL Server.

- Browse and explore each model type by using the custom data mining viewers that are provided for each algorithm type. For more information, see Mining Model Viewer Tasks and How-tos.
- Review the model content for each model type, by using the **Microsoft Generic Content Tree Viewer**. To interpret this information, refer to Mining Model Content (Analysis Services - Data Mining).

Query Tools and Interfaces

You can build data mining queries interactively by using one of the query tools provided by SQL Server. The graphical Prediction Query Builder is provided in both SQL Server Data Tools (SSDT) and SQL Server Management Studio. If you have not used the Prediction Query Builder before, we recommend that you follow the steps in the Basic Data Mining Tutorial to familiarize yourself with the interface. For q quick overview of the steps, see Create a Query using the Create a Prediction Query Using the Prediction Query Builder.

The Prediction Query Builder is helpful for starting queries that you will customize later. You can easily add data sources and map them to columns, and then switch to DMX view and customize the query by adding a WHERE clause or other functions.

Once you are familiar with data mining models and how to build queries, you can also write queries directly by using Data Mining Extensions (DMX). DMX is a query language that is similar to Transact-SQL, and that you can use from many different clients. DMX is the tool of choice for creating both custom predictions and complex queries. For an introduction to DMX, see Creating and Querying Data Mining Models with DMX: Tutorials (Analysis Services - Data Mining).

DMX editors are provided in both SQL Server Data Tools (SSDT) and SQL Server Management Studio. You can also use the Prediction Query Builder to start your queries, then change the view to the text editor and copy the DMX statement to another client. For more information, see Data Mining Query Tools.

You can compose DMX statements programmatically and send them from your client to the Analysis Services server by using AMO or XMLA. However, DMX is the language that you must use to create queries against a mining model.

You can also query the metadata, statistics, and some content of the model by using Dynamic Management Views (DMVs) that are based on the data mining schema rowsets. These DMVs make it easy to retrieve information about the model by typing SELECT statements; however, you cannot create predictions. For more information about DMVs supported by Analysis Services, see Use Dynamic Management Views (DMVs) to Monitor Analysis Services.

Finally, you can create data mining queries for use in Integration Services packages, by using the Data Mining Query Task, or the Data Mining Query Transformation. The control flow task supports multiple types of DMX queries, whereas the data flow transformation supports only queries that work with data in the data flow, meaning queries that use the PREDICTION JOIN syntax.

Queries for Different Model Types

The algorithm that was used when the model was created greatly influences the type of information that you can get from a data mining query. The reason for the differences is that each algorithm processes the data in a different way, and stores different kinds of patterns. For example, some algorithms create clusters; others create trees. Therefore, you might need to use specialized prediction and query functions, depending on the type of model that you are working with.

The following list provides a summary of the functions that you can use in queries:

- **General prediction functions:** The **Predict** function is polymorphic, meaning it works with all model types. This function will automatically detect the type of model you are working with and prompt you for additional parameters. For more information, see Predict (DMX).

  > **WARNING**
  > Not all models are used to make predictions. For example, you can create a clustering model that does not have a predictable attribute. However, even if a model does not have a predictable attribute, you can create prediction queries that return other types of useful information from the model.

- **Custom prediction functions:** Each model type provides a set of prediction functions designed for working with the patterns created by that algorithm.

  For example, the **Lag** function is provided for time series models, to let you view the historical data used for the model. For clustering models, functions such as **ClusterDistance** are more meaningful.

  For more information about the functions that are supported for each model type, see the following links:

Association Model Query Examples	Microsoft Naive Bayes Algorithm
Clustering Model Query Examples	Neural Network Model Query Examples
Decision Trees Model Query Examples	Sequence Clustering Model Query Examples
Linear Regression Model Query Examples	Time Series Model Query Examples
Logistic Regression Model Query Examples	

  You can also call VBA functions, or create your own functions. For more information, see Functions (DMX).

- **General statistics:** There are a number of functions that can be used with almost any model type, which return a standard set of descriptive statistics, such as standard deviation.

  For example, the **PredictHistogram** function returns a table that lists all the states of the specified column.

  For more information, see General Prediction Functions (DMX).

- **Custom statistics:** Additional supporting functions are provided for each model type, to generate statistics that are relevant to the specific analytical task.

  For example, when you are working with a clustering model, you can use the function, **PredictCaseLikelihood**, to return the likelihood score associated with a certain case and cluster. However, if you created a linear regression model, you would be more interested in retrieving the coefficient and intercept, which you can do using a content query.

- **Model content functions:** The *content* of all models is represented in a standardized format that lets you retrieve information with a simple query. You create queries on the model content by using DMX. You can also get some type of model content by using the data mining schema rowsets.

In the model content, the meaning of each row or node of the table that is returned differs depending on the type of algorithm that was used to build the model, as well as the data type of the column. For more information, see Content Queries (Data Mining).

Requirements

Before you can create a query against a model, the data mining model must have been processed. Processing of Analysis Services objects requires special permissions. For more information on processing mining models, see Processing Requirements and Considerations (Data Mining).

To execute queries against a data mining model requires different levels of permissions, depending on the type of query that you run. For example, drillthrough to case or structure data typically requires additional permissions which can be set on the mining structure object or mining model object.

However, if your query uses external data, and includes statements such as OPENROWSET or OPENQUERY, the database that you are querying must enable these statements, and you must have permission on the underlying database objects.

For more information on the security contexts required to run data mining queries, see Security Overview (Data Mining)

In This Section

The topics in this section introduce each type of data mining query in more detail, and provide links to detailed examples of how to create queries against data mingin models.

Prediction Queries (Data Mining)

Content Queries (Data Mining)

Drillthrough Queries (Data Mining)

Data Definition Queries (Data Mining)

Data Mining Query Tools

Related Tasks

Use these links to learn how to create and work with data mining queries.

TASKS	LINKS
View tutorials and walkthroughs on data mining queries	Lesson 6: Creating and Working with Predictions (Basic Data Mining Tutorial) Time Series Prediction DMX Tutorial
Use data mining query tools in SQL Server Management Studio and SQL Server Data Tools (SSDT)	Create a DMX Query in SQL Server Management Studio Create a Prediction Query Using the Prediction Query Builder Apply Prediction Functions to a Model Manually Edit a Prediction Query

TASKS	LINKS
Work with external data used in prediction queries	Choose and Map Input Data for a Prediction Query Choose and Map Input Data for a Prediction Query
Work with the results of queries	View and Save the Results of a Prediction Query
Use DMX and XMLA query templates provided in Management Studio	Create a Singleton Prediction Query from a Template Create a Data Mining Query by Using XMLA Use Analysis Services Templates in SQL Server Management Studio
Learn more about content queries and see examples	Create a Content Query on a Mining Model Query the Parameters Used to Create a Mining Model Content Queries (Data Mining)
Set query options and troubleshoot query permissions and problems	Change the Time-out Value for Data Mining Queries
Use the data mining components in Integration Services	Data Mining Query Task Data Mining Query Transformation

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Mining Model Content (Analysis Services - Data Mining)

Prediction Queries (Data Mining)

11/28/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The goal of a typical data mining project is to use the mining model to make predictions. For example, you might want to predict the amount of expected downtime for a certain cluster of servers, or generate a score that indicates whether segments of customers are likely to respond to an advertising campaign. To do all these things, you would create a prediction query.

Functionally, there are different types of prediction queries supported in SQL Server, depending on the type of inputs to the query:

QUERY TYPE	QUERY OPTIONS
Singleton prediction queries	Use a singleton query when you want to predict outcomes for a single new case, or multiple new cases. You provide the input values directly in the query, and the query is executed as a single session.
Batch predictions	Use batch predictions when you have external data that you want to feed into the model, to use as the basis for predictions. To make predictions for an entire set of data, you map the data in the external source to the columns in the model, and then specify the type of predictive data you want to output. The query for the entire dataset is executed in a single session, making this option much more efficient than sending multiple repeated queries.
Time Series predictions	Use a time series query when you want to predict a value over some number of future steps. SQL Server Data Mining also provides the following functionality in time series queries: You can extend an existing model by adding new data as part of the query, and make predictions based on the composite series. You can apply an existing model to a new data series by using the REPLACE_MODEL_CASES option. You can perform cross-prediction.

The following sections describe the general syntax of prediction queries, the different types of prediction queries, and how to work with the results of prediction queries.

Basic Prediction Query Design

- Adding Prediction Functions
- Singleton Queries
- Batch Prediction Queries
- Time Series Queries

Basic Prediction Query Design

When you create a prediction, you typically provide some piece of new data and ask the model to generate a prediction based on the new data.

- In a batch prediction query, you map the model to an external source of data by using a *prediction join*.
- In a singleton prediction query, you type one or more values to use as inputs. You can create multiple predictions using a singleton prediction query. However, if you need to create many predictions, performance is better when you use a batch query.

Both singleton and batch prediction queries use the PREDICTION JOIN syntax to define the new data. The difference is in how the input side of the prediction join is specified.

- In a batch prediction query, the data comes from an external data source that is specified by using the OPENQUERY syntax.
- In a singleton prediction query, the data is supplied inline as part of the query.

For time series models, input data is not always required; it is possible to make predictions using just the data already in the model. However, if you do specify new input data, you must decide whether you will use the new data to update and extend the model, or to replace the original series of data that was used in the model. For more information about these options, see Time Series Model Query Examples.

Adding Prediction Functions

In addition to predicting a value, you can customize a prediction query to return various kinds of information that are related to the prediction. For example, if the prediction creates a list of products to recommend to a customer, you might also want to return the probability for each prediction, so that you can rank them and present only the top recommendations to the user.

To do this, you add *prediction functions* to the query. Each model or query type supports specific functions. For example, clustering models support special prediction functions that provide extra detail about the clusters created by the model, whereas time series models have functions that calculate differences over time. There are also general prediction functions that work with almost all model types. For a list of the prediction functions supported in different types of queries, see this topic the DMX reference: General Prediction Functions (DMX).

Creating Singleton Prediction Queries

A singleton prediction query is useful when you want to create quick predictions in real time. A common scenario might be that you have obtained information from a customer, perhaps by using a form on a Web site, and you want to submit that data as the input to a singleton prediction query. For example, when a customer chooses a product from a list, you could use that selection as the input to a query that predicts the best products to recommend.

Singleton prediction queries do not require a separate table that contains input. Instead, you provide one or multiple rows of values as input to the model, and the prediction or predictions are returned in real time.

> **WARNING**
> Despite the name, singleton prediction queries do not just make single predictions-you can generate multiple predictions for each set of inputs. You provide multiple input cases by creating a SELECT statement for each input case and combining them with the UNION operator.

When you create a singleton prediction query, you must provide the new data to the model in the form of a PREDICTION JOIN. This means that even though you are not mapping to an actual table, you must make sure

that the new data matches the existing columns in the mining model. If the new data columns and the new data match exactly, Analysis Services will map the columns for you. This is called a *NATURAL PREDICTION JOIN*. However, if the columns do not match, or if the new data does not contain the same kind and amount of data that is in the model, you must specify which columns in the model map to the new data, or specify the missing values.

Batch Prediction Queries

A batch prediction query is useful when you have external data that you want to use in making predictions. For example, you might have built a model that categorizes customers by their online activity and purchasing history. You could apply that model to a list of newly acquired leads, to create projections for sales, or to identify targets for proposed campaigns.

When you perform a prediction join, you must map the columns the model to the columns in the new data source. Therefore, the data source that you choose for an input must data that is somewhat similar to the data in the model. The new information does not have to match exactly, and can be incomplete. For example, suppose the model was trained using information about income and age, but the customer list you are using for predictions has age but nothing about income. In this scenario, you could still map the new data to the model and create a prediction for each customer. However, if income was an important predictor for the model, the lack of complete information would affect the quality of predictions.

To get the best results, you should join as many of the matching columns as possible between the new data and the model. However, the query will succeed even if there are no matches. If no columns are joined, the query will return the marginal prediction, which is equivalent to the statement `SELECT <predictable-column> FROM <model>` without a PREDICTION JOIN clause.

After you have successfully mapped all relevant columns, you run the query, and Analysis Services makes predictions for each row in the new data based on patterns in the model. You can save the results back to a new table in the data source view that contains the external data, or you can copy and paste the data is you are using SQL Server Data Tools (SSDT) or SQL Server Management Studio.

> **WARNING**
> If you use the designer in SQL Server Data Tools (SSDT), the external data source must first be defined as a data source view.

If you use DMX to create a prediction join, you can specify the external data source by using the OPENQUERY, OPENROWSET, or SHAPE commands. The default data access method in the DMX templates is OPENQUERY. For information about these methods, see <source data query>.

Predictions in Time Series Mining Models

Time series models are different from other models types; you can either use the model as it is to create predictions, or you can provide new data to the model to update the model and create predictions based on recent trends. If you add new data, you can specify the way the new data should be used.

- *Extending the model cases* means that you add the new data onto the existing series of data in the time series model. Henceforth, predictions are based on the new, combined series. This option is good when you want to simply add a few data points to an existing model.

  For example, suppose that you have an existing time series model that has been trained on the sales data from the previous year. After you have collected several months of new sales data, you decide to update your sales forecasts for the current year. You can create a prediction join that updates the model by adding new data and extends the model to make new predictions.

- *Replacing the model cases* means that you keep the trained model, but replace the underlying cases with a new set of case data. This option is useful when you want to keep the trend in the model, but apply it to a different set of data.

For example, your original model might have been trained on a set of data with very high sales volumes; when you replace the underlying data with a new series (perhaps from a store with lower sales volume), you preserve the trend, but the predictions begin from the values in the replacement series.

Regardless of which approach you use, the starting point for predictions is always the end of the original series.

For more information about how to create prediction joins on time series models, see Time Series Model Query Examples or PredictTimeSeries (DMX).

Working with the Results of a Prediction Query

Your options for saving the results of a data mining prediction query are different depending on how you create the query.

- When you build a query using Prediction Query Builder in either SQL Server Management Studio or SQL Server Data Tools (SSDT), you can save the results of a prediction query to an existing Analysis Services data source. For more information, see View and Save the Results of a Prediction Query.
- When you create prediction queries using DMX in the Query pane of SQL Server Management Studio, you can use the query output options to save the results to a file, or to the Query Results pane as text or in a grid. For more information, see Query and Text Editors (SQL Server Management Studio).
- When you run a prediction query using the Integration Services components, the tasks provides the ability to write the results to a database by using an available ADO.NET connection manager or OLEDB connection manager. For more information, see Data Mining Query Task.

It is important to understand that the results of a prediction query are not like the results of a query on a relational database, which always returns a single row of related values. Each DMX prediction function that you add to a query returns its own rowset. Therefore, when you make a prediction on a single case, the result might be a predicted value together with several columns of nested tables containing additional detail.

If you combine multiple functions in one query, the return results are combined as a hierarchical rowset. For example, say you use a time series model to predict future values for sales amount and sales quantity, using a query such as this DMX statement:

```
SELECT
  PredictTimeSeries([Forecasting].[Amount]) as [PredictedAmount]
, PredictTimeSeries([Forecasting].[Quantity]) as [PredictedQty]
FROM
  [Forecasting]
```

The results of this query are two columns, one for each predicted series, where each row contains a nested table with the predicted values:

PredictedAmount

$TIME	AMOUNT
201101	172067.11

$TIME	AMOUNT
201102	363390.68

PredictedQty

$TIME	QUANTITY
201101	77

$TIME	QUANTITY
201102	260

If your provider cannot handle hierarchical rowsets, you can flatten the results by using the FLATTEN keyword in the prediction query. For more information, including examples of flattened rowsets, see SELECT (DMX).

See Also

Content Queries (Data Mining)
Data Definition Queries (Data Mining)

Content Queries (Data Mining)

11/28/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A content query is a way of extracting information about the internal statistics and structure of the mining model. Sometimes a content query can provide details that are not readily available in the viewer. You can also use the results of a content query to extract information programmatically for other uses.

This section provides general information about the types of information that you can retrieve by using a content query, and the general DMX syntax for content queries.

Basic Content Queries

- Queries on Structure and Case Data
- Queries on Model Patterns

Examples

- Content Query on an Association Model
- Content Query on a Decision Trees Model

Working with the Query Results

Basic Content Queries

You can create content queries by using the Prediction Query Builder, use the DMX content query templates that are provided in SQL Server Management Studio, or write queries directly in DMX. Unlike prediction queries you do not need to join external data, so content queries are easy to write.

This section provides an overview of the types of content queries you can create.

- Queries on the mining structure or case data let you view the detailed data that was used for training.
- Queries on the model can return patterns, lists of attributes, formulas, and so forth.

Queries on Structure and Case Data

DMX supports queries on the cached data that is used to build mining structures and models. By default, this cache is created when you define the mining structure, and is populated when you process the structure or model.

> **WARNING**
> This cache cannot be cleared or deleted if you need to separate data into training and testing sets. If the cache is cleared, you cannot query the case data.

The following examples show the common patterns for creating queries on the case data, or queries on the data in the mining structure:

Get all the cases for a model

```
SELECT FROM <model>.CASES
```

Use this statement to retrieve specified columns from the case data used to build a model. You must have drillthrough permissions on the model to run this query.

View all the data that is included in the structure

```
SELECT FROM <structure>.CASES
```

Use this statement to view all the data that is included in the structure, including columns that are not included in a particular mining model. You must have drillthrough permissions on the model, as well as on the structure, to retrieve data from the mining structure.

Get range of values

```
SELECT DISTINCT RangeMin(<column>), RangeMax(<column>) FROM <model>
```

Use this statement to find the minimum value, maximum value, and mean of a continuous column, or of the buckets of a DISCRETIZED column.

Get distinct values

```
SELECT DISTINCT <column>FROM <model>
```

Use this statement to retrieve all the values of a DISCRETE column. Do not use this statement for DISCRETIZED columns; use the **RangeMin** and **RangeMax** functions instead.

Find the cases that were used to train a model or structure

```
SELECT FROM <mining structure.CASES WHERE IsTrainingCase()
```

Use this statement to get the complete set of data that was used in a training a model.

Find the cases that are used for testing a model or structure

```
SELECT FROM <mining structure.CASES WHERE IsTestingCase()
```

Use this statement to get the data that has been set aside for testing mining models related to a specific structure.

Drillthrough from a specific model pattern to underlying case data

```
SELECT FROM <model>.CASESWHERE IsTrainingCase() AND IsInNode(<node>)
```

Use this statement to retrieve detailed case data from a trained model. You must specify a specific node: for example, you must know the node ID of the cluster, the specific branch of the decision tree, etc. Moreover, you must have drillthrough permissions on the model to run this query.

Queries on Model Patterns, Statistics, and Attributes

The content of a data mining model is useful for many purposes. With a model content query, you can:

- Extract formulas or probabilities for making your own calculations.
- For an association model, retrieve the rules that are used to generate a prediction.
- Retrieve the descriptions of specific rules so that you can use the rules in a custom application.
- View the moving averages detected by a time series model.
- Obtain the regression formula for some segment of the trend line.
- Retrieve actionable information about customers identified as being part of a specific cluster.

The following examples show some of the common patterns for creating queries on the model content:

Get patterns from the model

```
SELECT FROM <model>.CONTENT
```

Use this statement to retrieve detailed information about specific nodes in the model. Depending on the algorithm type, the node can contain rules and formulas, support and variance statistics, and so forth.

Retrieve attributes used in a trained model

```
CALL System.GetModelAttributes(<model>)
```

Use this stored procedure to retrieve the list of attributes that were used by a model. This information is useful for determining attributes that were eliminated as a result of feature selection, for example.

Retrieve content stored in a data mining dimension

`SELECT FROM <model>.DIMENSIONCONTENT`

Use this statement to retrieve the data from a data mining dimension.

This query type is principally for internal use. However, not all algorithms support this functionality. Support is indicated by a flag in the MINING_SERVICES schema rowset.

If you develop your own plug-in algorithm, you might use this statement to verify the content of your model for testing.

Get the PMML representation of a model

`SELECT * FROM <model>.PMML`

Gets an XML document that represents the model in PMML format. Not all model types are supported.

Examples

Although some model content is standard across algorithms, some parts of the content vary greatly depending on the algorithm that you used to build the model. Therefore, when you create a content query, you must understand what information in the model is most useful to your specific model.

A few examples are provided in this section to illustrate how the choice of algorithm affects the kind of information that is stored in the model. For more information about mining model content, and the content that is specific to each model type, see Mining Model Content (Analysis Services - Data Mining).

Example 1: Content Query on an Association Model

The statement, `SELECT FROM <model>.CONTENT`, returns different kinds of information, depending on the type of model you are querying. For an association model, a key piece of information is the *node type*. Nodes are like containers for information in the model content. In an association model, nodes that represent rules have a NODE_TYPE value of 8, whereas nodes that represent itemsets have a NODE_TYPE value of 7.

Thus, the following query returns the top 10 itemsets, ranked by support (the default ordering).

```
SELECT TOP 10 NODE_DESCRIPTION, NODE_PROBABILITY, SUPPORT
FROM <model>.CONTENT WHERE NODE_TYPE = 7
```

The following query builds on this information. The query returns three columns: the ID of the node, the complete rule, and the product on the right-hand side of the itemset-that is, the product that is predicted to be associated with some other products as part of an itemset.

```
SELECT FLATTENED NODE_UNIQUE_NAME, NODE_DESCRIPTION,
     (SELECT RIGHT(ATTRIBUTE_NAME, (LEN(ATTRIBUTE_NAME)-LEN('Association model name')))
FROM NODE_DISTRIBUTION
WHERE LEN(ATTRIBUTE_NAME)>2
)
AS RightSideProduct
FROM [<Association model name>].CONTENT
WHERE NODE_TYPE = 8
ORDER BY NODE_SUPPORT DESC
```

The FLATTENED keyword indicates that the nested rowset should be converted into a flattened table. The attribute that represents the product on the right-hand side of the rule is contained in the NODE_DISTRIBUTION table; therefore, we retrieve only the row that contains an attribute name, by adding a requirement that the length

be greater than 2.

A simple string function is used to remove the name of the model from the third column. (Usually the model name is prefixed to the values of nested columns.)

The WHERE clause specifies that the value of NODE_TYPE should be 8, to retrieve only rules.

For more examples, see Association Model Query Examples.

Example 2: Content Query on a Decision Trees Model

A decision tree model can be used for prediction as well as for classification. This example assumes that you are using the model to predict an outcome, but you also want to find out which factors or rules can be used to classify the outcome.

In a decision tree model, nodes are used to represent both trees and leaf nodes. The caption for each node contains the description of the path to the outcome. Therefore, to trace the path for any particular outcome, you need to identify the node that contains it, and get the details for that node.

In your prediction query, you add the prediction function PredictNodeId (DMX), to get the ID of the related node, as shown in the following example:

```
SELECT  Predict([Bike Buyer]), PredictNodeID([Bike Buyer])
FROM [<decision tree model name>]
PREDICTION JOIN
<input rowset>
```

Once you have the ID of the node that contains the outcome, you can retrieve the rule or path that explains the prediction by creating a content query that includes the NODE_CAPTION, as follows:

```
SELECT NODE_CAPTION
FROM [<decision tree model name>]
WHERE NODE_UNIQUE_NAME= '<node id>'
```

For more examples, see Decision Trees Model Query Examples.

Working with the Query Results

As the examples demonstrate, content queries mostly return tabular rowsets, but can also include information from nested columns. You can flatten the rowset that is returned, but this can make working with results more complex. The content of the NODE_DISTRIBUTION node in particular is nested, but contains much interesting information about the model.

For more information about how to work with hierarchical rowsets, see the OLEDB specification on MSDN.

See Also

Understanding the DMX Select Statement
Data Mining Queries

Drillthrough Queries (Data Mining)

11/28/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A *drillthrough query* lets you retrieve details from the underlying cases or structure data, by sending a query to the mining model. Drillthrough is useful if you want to view the cases that were used to train the model, versus the cases that are used to test the model, or if you want to see additional details from the case data.

Analysis Services Data Mining provides two different options for drillthrough:

- Drilling through to the **model cases**

  Drillthrough to model cases is used when you want to go from a specific pattern in the model-such as a cluster or branch of a decision tree-and view details about the individual cases.

- Drilling through to the **structure cases**

  Drillthrough to structure cases is used when the structure contains information that might not be available in the model. For example, you would not use customer contact information in a clustering model, even if the data was included in the structure. However, after you create the model, you might want to retrieve contact information for customers who are grouped into a particular cluster.

This section provides examples of how you can create these queries.

Using Drillthrough in Data Mining Designer

Creating Drillthrough Queries using DMX

Considerations When Using Drillthrough

- Security Issues
- Limitations

Using Drillthrough in Data Mining Designer

If a mining model has been configured to allow drillthrough, and if you have the appropriate permissions, when you browse the model, you can click on a node in the appropriate viewer and retrieve detailed information about the cases in that particular node.

Drill Through to Case Data from a Mining Model.

If the training cases were cached when you processed the mining structure, and you have the necessary permissions, you can return information from the model cases and from the mining structure, including columns that were not included in the mining model.

Creating Drillthrough Queries using DMX

You can drill through to case data by creating a DMX query, if you have permissions on the model or on the structure. For examples of the syntax for creating drillthrough queries in DMX, see the following topic:

Create Drillthrough Queries using DMX

Considerations When Using Drillthrough

- If you use the Data Mining Wizard, the option to enable drillthrough to the model cases is on the final page of the wizard. Drillthrough is disabled by default. For more information, see Completing the Wizard (Data Mining Wizard).
- You can add the ability to drill through on an existing mining model, but if you do, the model must be reprocessed before you can drill through to the data.
- Drillthrough works by retrieving information about the training cases that was cached when you processed the mining structure. Therefore, if you cleared the cached data after processing the structure by changing the MiningStructureCacheMode property to **ClearAfterProcessing**, drillthrough will not work. To enable drillthrough to structure columns, you must change the MiningStructureCacheMode property to **KeepTrainingCases** and then reprocess the structure.
- If the mining structure does not allow drillthrough but the mining model does, you can view information only from the model cases, and not from the mining structure.

Security Issues for Drillthrough

If you want to drill through to structure cases from the model, you must verify that both the mining structure and the mining model have the AllowDrillThrough property set to **True**. Moreover, you must be a member of a role that has drillthrough permissions on both the structure and the model. For information about how to create roles, see Role Designer (Analysis Services - Multidimensional Data). see.

Drillthrough permissions are set separately on the structure and model. The model permission lets you drill through from the model, even if you do not have permissions on the structure. Drillthrough permissions on the structure provide the additional ability to include structure columns in drillthrough queries from the model, by using the StructureColumn (DMX) function.

> **NOTE**
> If you enable drillthrough on both the mining structure and the mining model, any user who is a member of a role that has drillthrough permissions on the mining model can also view columns in the mining structure, even if those columns are not included in the mining model. Therefore, to protect sensitive data, you should set up the data source view to mask personal information, and allow drillthrough access on the mining structure only when necessary.

Limitations on Drillthrough

- The following limitations apply to drillthrough operations on a model, depending on the algorithm that was used to create the model:

ALGORITHM NAME	ISSUE
Microsoft Naïve Bayes algorithm	Not supported. These algorithms do not assign cases to specific nodes in the content.
Microsoft Neural Network algorithm	Not supported. These algorithms do not assign cases to specific nodes in the content.
Microsoft Logistic Regression algorithm	Not supported. These algorithms do not assign cases to specific nodes in the content.
Microsoft Linear Regression algorithm	Supported. However, because the model creates a single node, **All**, drilling through returns all the training cases for the model. If the training set is large, loading the results may take a very long time.

ALGORITHM NAME	ISSUE
Microsoft Time Series algorithm	Supported. However, you cannot drill through to structure or case data by using the **Mining Model Viewer** in Data Mining Designer. You must create a DMX query instead. Also, you cannot drill through to specific nodes, or write a DMX query to retrieve cases in specific nodes of a time series model. You can retrieve case data from either the model or the structure by using other criteria, such as date or attribute values. You can also return the dates from the cases in the model, by using the Lag (DMX) function. If you wish to view details of the ARTXP and ARIMA nodes created by the Microsoft Time Series algorithm, you can use the Microsoft Generic Content Tree Viewer (Data Mining).

Related Tasks

Use the following links to work with drillthrough in specific scenarios.

TASK	LINK
Procedure describing use of drillthrough in the Data Mining Designer	Drill Through to Case Data from a Mining Model
To alter an existing mining model to allow drillthrough	Enable Drillthrough for a Mining Model
Enabling drillthrough on a mining structure by using the DMX WITH DRILLTHROUGH clause	CREATE MINING STRUCTURE (DMX)
For information about assigning permissions that apply to drillthrough on mining structures and mining models	Grant permissions on data mining structures and models (Analysis Services)

See Also

Data Mining Model Viewers
Data Mining Queries

Drill Through to Case Data from a Mining Model

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If a mining model has been configured to let you drill through to model cases, when you browse the model, you can retrieve detailed information about the cases that were used to create the model. Moreover, if the underlying mining structure has been configured to allow drillthrough to structure cases, and you have the appropriate permissions, you can return information from the mining structure. This can include columns that were not included in the mining model.

If the mining structure does not allow you to drill through to the underlying data, but the mining model does, you can view information from the model cases, but not from the mining structure.

> **NOTE**
> You can add the ability to drillthrough on an existing mining model by setting the property **AllowDrillthrough** to **True**. However, after you enable drillthrough, the model must be reprocessed before you can see the case data. For more information, see Enable Drillthrough for a Mining Model.

Depending on the type of viewer you are using, you can select the node for drillthrough in the following ways:

VIEWER NAME	PANE OR TAB NAME	SELECT NODE
Microsoft Tree Viewer	**Decision Tree** tab	Click a tree node. **Note** Avoid using drillthrough on the **All** node, because it may take a very long time to return results.
Microsoft Cluster Viewer	**Cluster Diagram**	Click a cluster node.
Microsoft Cluster Viewer	**Cluster Profiles**	Click anywhere in cluster column.
Microsoft Association Viewer	**Rules** tab	Click a row that contains a set of rules.
Microsoft Association Viewer	**Itemsets** tab	Click a row that contains an itemset.
Microsoft Sequence Clustering Viewer	**Rules** tab	Click a row that contains a set of rules.
Microsoft Sequence Clustering Viewer	**Itemsets** tab	Click a row that contains an itemset.

> **NOTE**
> Some models cannot use drillthrough. The ability to use drillthrough depends on the algorithm that was used to create the model. For a list of the mining model types that support drillthrough, see Drillthrough Queries (Data Mining).

To view drillthrough data from a mining model

1. In SQL Server Data Tools (SSDT), open the mining structure that contains the mining model you want.
2. In Data Mining Designer, click the **Mining Model Viewer** tab.
3. Select the model from the **Mining Model** drop-down list.
4. Select a viewer from the **Viewer** drop-down list, and right-click the specific node.
5. Select **Drill Through**, and then select either **Models Columns Only**, or **Model and Structure Columns** to open the **Drill Through** window.
6. To copy the data to the Clipboard, right-click any row in the table, and select **Copy All**.

See Also

Drillthrough Queries (Data Mining)

Create Drillthrough Queries using DMX

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

For all models that support drillthrough, you can retrieve case data and structure data by creating a DMX query in SQL Server Management Studio or any other client that supports DMX.

> **WARNING**
>
> To view the data, drillthrough must have been enabled, and you must have the necessary permissions.

Specifying Drillthrough Options

The general syntax is for retrieving model cases and structure cases is as follows:

```
SELECT <model column list>, StructureColumn('<structure column name') FROM <modelname>.CASES
```

For additional information about using DMX queries to return case data, see SELECT FROM <model>.CASES (DMX) and SELECT FROM <structure>.CASES.

Examples

The following DMX query returns the case data for a specific product series, from a time series model. The query also returns the column **Amount**, which was not used in the model but is available in the mining structure.

```
SELECT [DateSeries], [Model Region], Quantity, StructureColumn('Amount') AS [M200 Pacific Amount]
FROM Forecasting.CASES
WHERE [Model Region] = 'M200 Pacific'
```

Note that in this example, an alias has been used to rename the structure column. If you do not assign an alias to the structure column, the column is returned with the name 'Expression'. This is the default behavior for all unnamed columns.

See Also

Drillthrough Queries (Data Mining)
Drillthrough on Mining Structures

Data Definition Queries (Data Mining)

10/26/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

For data mining, the category *data definition query* means DMX statements or XMLA commands that do the following:

- Create, alter, or manipulate data mining objects, such as a model.
- Define the source of data to be used in training or for prediction.
- Export or import mining models and mining structures.

Creating Data Definition Queries

- Data Definition Queries in SQL Server Data Tools
- Data Definition Queries in SQL Server Management Studio

Scripting Data Definition Statements

Scripting Data Definition Statements

Creating Data Definition Queries

You can create data definition queries (statements) by using the Prediction Query Builder in SQL Server Data Tools (SSDT) and SQL Server Management Studio, or by using the DMX Query window in SQL Server Management Studio. Data definition statements in DMX are part of the Analysis Services data definition language (DDL).

For information about the syntax of specific data definition statements, see Data Mining Extensions (DMX) Reference.

Data Definition Queries in SQL Server Data Tools

The Data Mining Wizard is the preferred tool in SQL Server Data Tools (SSDT) for creating and modifying mining models and mining structures, and for defining the data sources that are used in prediction queries and for training.

However, if you want to know what statements are being sent to the server by the wizard to create data structures or mining models, you can use SQL Server Profiler to capture the data definition statements. For more information, see Use SQL Server Profiler to Monitor Analysis Services.

To view the statements used for defining data sources used for training or prediction, you can use the **SQL View** in the Prediction Query Builder. Sometimes it can be helpful to build basic queries for training and testing models by using Prediction Query Builder, to establish the correct syntax. You can then switch to **SQL View** and manually edit the query. For more information, see Manually Edit a Prediction Query.

Data Definition Queries in SQL Server Management Studio

For data mining objects, you can use data definition queries to perform the following actions:

- Create specific types of models, such as a clustering model or decision tree model, by using CREATE MINING MODEL (DMX).
- Alter an existing mining structure by adding a model or by changing the columns, by using ALTER

MINING STRUCTURE (DMX). Note that you cannot alter a mining model by using DMX; you only add new models to an existing structure.

- Make a copy of a mining model and then alter it, by using SELECT INTO (DMX).
- Define the data set used for training a model, by using INSERT INTO (DMX) together with a data source query such as OPENROWSET.

SQL Server Management Studio provides query templates that can help you create data definition queries. For more information, see Use Analysis Services Templates in SQL Server Management Studio.

In general, the templates that are provided for Analysis Services in SQL Server Management Studio contain only the general syntax definition, which you must customize, either by typing in the **Query** window, or by using the dialog box provided for entering parameters.

For an example of how to enter parameters using the interface, see Create a Singleton Prediction Query from a Template.

Scripting Data Definition Statements

Analysis Services provides multiple scripting and programming languages that you can use to create or alter data mining objects, or to define data sources. Although DMX is designed for expediting data mining tasks, you can also use both XMLA and AMO to manipulate objects in scripts or in custom code.

The Data Mining Add-in for Excel also includes many query templates, and provides the **Advanced Query Editor**, which helps you compose complex DMX statements. You can build a query interactively and then switch to SQL View to capture the DMX statement.

Exporting and Importing Models

You can use data definition statements in DMX to export the definition of a model and its required structure and data sources, and then import that definition into a different server. Using export and import is the fastest and easiest way to move data mining models and mining structures between instances of Analysis Services. For more information, see Management of Data Mining Solutions and Objects.

> **WARNING**
> If your model is based on data from a cube data srouce, you cannot use DMX to export the model, and should use backup and restore instead.

Related Tasks

The following table provides links to tasks that are related to data definition queries.

Work with templates for DMX queries.	Use Analysis Services Templates in SQL Server Management Studio
Design queries of all kinds, using Prediction Query Builder.	Create a Prediction Query Using the Prediction Query Builder
Capture query definitions by using SQL Server Profiler, and use traces to monitor Analysis Services.	Use SQL Server Profiler to Monitor Analysis Services
Learn more about the scripting languages and programming languages provided for Analysis Services.	XML for Analysis (XMLA) Reference Developing with Analysis Management Objects (AMO)

Learn how to manage models in SQL Server Management Studio and SQL Server Data Tools (SSDT).	Export and Import Data Mining Objects EXPORT (DMX) IMPORT (DMX)
Learn more about OPENROWSET and other ways to query external data.	<source data query>.

See Also

Data Mining Wizard (Analysis Services - Data Mining)

Data Mining Schema Rowsets (SSAs)

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In SQL Server 2017, many of the existing OLE DB data mining schema rowsets are exposed as a set of system tables that you can query by using Data Mining Extensions (DMX) statements. By creating queries against the data mining schema rowset, you can identify the services that are available, get updates on the status of your models and structures, and find out details about the model content or parameters. For a description of the data mining schema rowsets, see Data Mining Schema Rowsets.

> **NOTE**
> You can also query the data mining schema rowsets by using XMLA. For more information about how to do this in SQL Server Management Studio, see Create a Data Mining Query by Using XMLA.

List of Data Mining Schema Rowsets

The following table lists the data mining schema rowsets that may be useful for querying and monitoring.

ROWSET NAME	DESCRIPTION
DMSCHEMA_MINING_MODELS	Lists all mining models in the current context. Includes such information as the date created, parameters used to create the model, and the size of the training set.
DMSCHEMA_MINING_COLUMNS	Lists all columns used in mining models in the current context. Information includes mapping to mining structure source column, data type, precision, and prediction functions that can be used with the column.
DMSCHEMA_MINING_STRUCTURES	Lists all mining structure in the current context. Information includes whether the structure is populated, the date the structure was last processed, and the definition of the holdout data set for the structure, if any.
DMSCHEMA_MINING_STRUCTURE_COLUMNS	Lists all columns used in mining structures in the current context. Information includes content type and data type, nullability, and whether the column contains nested table data.
DMSCHEMA_MINING_SERVICES	Lists all mining services, or algorithms, that are available on the specified server. Information includes supported modeling flags, input types, and supported data source types.

ROWSET NAME	DESCRIPTION
DMSCHEMA_MINING_SERVICE_PARAMETERS	Lists all parameters for the mining services that are available on the current instance. Information includes the data type for each parameter, the default values, and the upper and lower limits.
DMSCHEMA_MODEL_CONTENT	Returns the content of the model if the model has been processed. For more information, see Mining Model Content (Analysis Services - Data Mining).
DBSCHEMA_CATALOGS	Lists all databases (catalogs) in the current instance of Analysis Services.
MDSCHEMA_INPUT_DATASOURCES	Lists all data sources in the current instance of Analysis Services.

> **NOTE**
> The list in the table is not comprehensive; it shows only those rowsets that may be of most interest for troubleshooting.

Examples

The following section provides some examples of queries against the data mining schema rowsets.

Example 1: List Data Mining Services

The following query returns a list of the mining services that are available on the current server, meaning the algorithms that are enabled. The columns provided for each mining service include the modeling flags and content types that can be used by each algorithm, the GUID for each service, and any prediction limits that may have been added for each service.

```
SELECT *
FROM $system.DMSCHEMA_MINING_SERVICES
```

Example 2: List Mining Model Parameters

The following example returns the parameters that were used to create a specific mining model:

```
SELECT MINING_PARAMETERS
FROM $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = 'TM Clustering'
```

Example 3: List All Rowsets

The following example returns a comprehensive list of the rowsets that are available on the current server:

```
SELECT *
FROM $system.DBSCHEMA_TABLES
```

Data Mining Query Tools

10/26/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

All data mining queries use the Data Mining Extensions (DMX) language. DMX can be used to create models for all kinds of machine learning tasks, including classification, risk analysis, generation of recommendations, and linear regression. You can also write DMX queries to get information about the patterns and statistics that were generated when you processed the model.

You can write your own DMX, or you can build basic DMX using a tool such as the **Prediction Query Builder** and then modify it. Both SQL Server Management Studio and SQL Server Data Tools (SSDT) provide tools that help you build DMX prediction queries. This topic describes how to create and execute data mining queries using these tools.

- Prediction Query Builder
- Query Editor
- DMX Templates
- Integration Services
- Application Programming Interfaces

Prediction Query Builder

Prediction Query Builder is included in the **Mining Model Prediction** tab of Data Mining Designer, which is available in both SQL Server Management Studioand SQL Server Data Tools (SSDT).

When you use the query builder, you select a mining model, add new case data, and add prediction functions. You can then switch to the text editor to modify the query manually, or switch to the **Results** pane to view the results of the query.

Query Editor

The Query Editor in SQL Server Management Studio also lets you build and run DMX queries. You can connect to an instance of Analysis Services, and then select a database, mining structure columns, and a mining model. The **Metadata Explorer** contains a list of prediction functions that you can browse.

DMX Templates

SQL Server Management Studio provides interactive DMX query templates that you can use to build DMX queries. If you do not see the list of templates, click **View** on the toolbar, and select **Template Explorer**. To see all Analysis Services templates, including templates for DMX, MDX, and XMLA, click the cube icon.

To build a query using a template, you can drag the template into an open query window, or you can double-click the template to open a new connection and a new query pane.

For an example of how to create a prediction query from a template, see Create a Singleton Prediction Query from a Template.

> **WARNING**
> The Data Mining Add-in for Microsoft Office Excel also contains a number of templates, along with an interactive query builder which can help you compose complex DMX statements. To use the templates, click **Query**, and click **Advanced** in the Data Mining Client.

Integration Services Data Mining Components

You can also include prediction queries as part of a SQL Server Integration Services package. The following tasks and transformations in Integration Services support the creation and execution of DMX prediction queries and DMX statements.

COMPONENT	DESCRIPTION
Data Mining Query task	Executes DMX queries and other DMX statements as part of a control flow. The task editor provides the Prediction Query Builder, and a text box for modifying the DMX query manually. However, the task editor cannot validate the query against objects in an Analysis Services solution. Therefore, it is best to create a query within SQL Server Data Tools or Management Studio and then paste the text of the statement or query into the task editor.
Data Mining Query transformation	Executes a prediction query within a data flow, using data supplied by a data flow source. The task editor provides the Prediction Query Builder, and a text box for modifying the DMX query manually. The transformation can only be used for creating queries that use data in the data flow; that is, queries that use the PREDICTION JOIN syntax. This component cannot be used for executing content queries or other kinds of DMX statements.

Application Programming Interfaces

You can create custom applications that execute queries against data mining models by using a variety of programming languages, in combination with server protocols such as OLE DB or Analysis Services ADOMD client. For more information, see Data Mining Programming.

However, XMLA constitutes the underlying message format for all interactions with an Analysis Service server. Within an XMLA message, queries are represented differently depending on whether you are sending a prediction query based on DMX, a content query, or a query that retrieves model metadata using the data mining schema rowsets.

- The text of **prediction queries** (and all other DMX statements) is sent in XMLA by using the Execute Method (XMLA) method, with the DMX query placed as text within the Statement Element (XMLA) element of the XMLA Command Element (XMLA) element.
- To retrieve **model content** and **model metadata**, such as the number of clusters, the attributes used in decision trees, the date the model was last processed, and the algorithm parameters used when creating the model, you can use the Discover Method (XMLA) method and specify one of the data mining schema rowsets in the RequestType Element (XMLA) header. To narrow the scope of the query, enter criteria as

restrictions within the RestrictionList Element (XMLA) element.

See Also

Data Mining Extensions (DMX) Reference
Data Mining Solutions
Understanding the DMX Select Statement
Structure and Usage of DMX Prediction Queries
Create a Prediction Query Using the Prediction Query Builder
Create a DMX Query in SQL Server Management Studio

Data Mining Query Tasks and How-tos

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The ability to create queries is critical if you are to make use of your data mining models. This section provides links to examples of how to create queries against a data mining model by using the tools provided in SQL Server Management Studio and SQL Server Data Tools (SSDT). If you need more information about what a data mining query is, or the different types of queries you can create, see Data Mining Queries.

Creating Queries with Prediction Query Builder

The Prediction Query Builder is provided in both SQL Server Data Tools (SSDT) and SQL Server Management Studio as a way of graphically building queries against data mining models. The following topics explain how you can select a model, specify a data source, customize the predictions, and save output.

- Create a Prediction Query Using the Prediction Query Builder
- Create a Singleton Query in the Data Mining Designer
- Create a Prediction Query Using the Prediction Query Builder
- View and Save the Results of a Prediction Query
- Manually Edit a Prediction Query
- Apply Prediction Functions to a Model
- Choose and Map Input Data for a Prediction Query

Using Other Data Mining Query Tools

In addition to using the Prediction Query Builder, you can type a query directly into SQL Server Management Studio by using DMX or by using XMLA. You can also build prediction queries programmatically and send them to an Analysis Services server. The following topics provide more information about how to create and work with prediction queries outside of the Prediction Query Builder.

Create a Singleton Prediction Query from a Template
Describes how to use the tools in SQL Server Management Studio to build and run a prediction query.

Create a Singleton Prediction Query from a Template
Describes how to use the templates that are provided in SQL Server Management Studio to add parameters to a prediction query.

Change the Time-out Value for Data Mining Queries
Describes how to set properties on the server that control behavior related to data mining queries.

Create a Content Query on a Mining Model
Describes how to create queries that return detailed information that is stored in the mining model by using the data mining schema rowsets.

Create a Data Mining Query by Using XMLA
Describes how to create a query against mining model content by using the XMLA templates in SQL Server Management Studio.

See Also

Query and Expression Language Reference (Analysis Services)
Data Mining Stored Procedures (Analysis Services - Data Mining)

Create a Prediction Query Using the Prediction Query Builder

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can create prediction queries either while you are building a data mining solution in BI Development Studio, or by right-clicking an existing mining model in SQL Server Management Studio, and then choosing the option, **Build Prediction Query**.

The **Prediction Query Builder** has the following three design modes, which you can switch among by clicking the icons in the upper-left corner.

- **Design**
- **Query**
- **Result**

Design mode lets you build a prediction query by choosing input data, mapping the data to the model, and then adding prediction functions into statements you build by using the grid. The design grid contains these building blocks:

Source
Choose the source of the new column. You can use columns from the mining model, input tables included in the data source view, a prediction function, or a customized expression.

Field
Determines the specific column or function that is associated with the selection in the **Source** column.

Alias
Determines how the column is to be named in the result set.

Show
Determines whether the selection in the **Source** column is displayed in the results.

Group
Works with the **And/Or** column to group expressions together by using parentheses. For example, (expr1 or expr2) and expr3.

And/Or
Creates logic in the query. For example, (expr1 or expr2) and expr3.

Criteria/Argument
Specifies a condition or user expression that applies to the column. You can drag columns from the tables to the cell.

Query mode provides a text editor that gives you direct access to the Data Mining Extensions (DMX) language, along with a view of the input data and model columns. When you select **Query** mode, the grid that you used to define the query is replaced by a basic text editor. You can use this editor to copy and save queries you have composed, or to paste in existing DMX queries and from the Clipboard and run them.

Result view runs the current query and displays the results in a grid. If the underlying data has changed and you want to rerun the query, click the Play button in the status bar

You can design a data mining query by using a combination of the visual tools and the text editor. If you type changes to the query in the text editor and switch back to the **Design** view, all the changes are lost, and the query reverts to the original query created by Prediction Query Builder This topic walks you through use of the graphical query builder.

To create a prediction query

1. Click the **Mining Model Prediction** tab in Data Mining Designer.
2. Click **Select Model** on the **Mining Model** table.

   The **Select Mining Model** dialog box opens to show all the mining structures that exist in the current project.
3. Select the model on which you want to create a prediction, and then click **OK**.
4. On the **Select Input Table(s)** table, click **Select Case Table**.

   The **Select Table** dialog box opens.
5. In the **Data Source** list, select the data source that contains the data on which to create a prediction.
6. In the **Table/View Name** box, select the table that contains the data on which to create a prediction, and then click **OK**.

   After you select the input table, Prediction Query Builder creates a default mapping between the mining model and the input table, based on the names of the columns. To delete a mapping, click to select the line that links the column in the **Mining Model** table to the mapped column in the **Select Input Table(s)** table, and then press DELETE. You can also manually create mappings by clicking a column in the **Select Input Table(s)** table and dragging it onto the corresponding column in the **Mining Model** table.
7. Add any combination of the following three types of information to the Prediction Query Builder grid:
   - Predictable columns from the **Mining Model** box.
   - Any combination of input columns from the **Select Input Table(s)** box.
   - Prediction functions
8. Run the query by clicking the first button on the toolbar of the **Mining Model Prediction** tab, and then selecting **Result**.

See Also

Create a Singleton Query in the Data Mining Designer
Data Mining Queries

Create a DMX Query in SQL Server Management Studio

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

SQL Server provides a set of features to help you create prediction queries, content queries, and data definition queries against mining models and mining structures.

- The graphical Prediction Query Builder is available in both SQL Server Data Tools (SSDT) and SQL Server Management Studio, to simplify the process of writing prediction queries and mapping data sets to a model.
- The query templates provided in the Template Explorer jump-start the creation of many kinds of DMX queries, including many types of prediction queries. Templates are provided for content queries, queries used nested data sets, queries that return cases from the mining structure, and even data definition queries.
- The Metadata Explorer in the MDX and DMX query panes provides a list of available models and structures that you can drag and drop into the query builder, as well as a list of DMX functions. This feature makes it easy to get object names right, without typing.

This topic describes how to build a DMX query by using the Metadata Explorer and the DMX query editor.

DMX Query Templates

Templates for creating basic DMX queries are available in Template Explorer. The **DMX** folder contains data mining templates, which are divided into these categories:

- **Model Content**
- **Model Management**
- **Prediction Queries**
- **Structure Content**

You can also create custom templates, for queries or commands that you run frequently.

XMLA Query Templates

Analysis Services also provides templates for XMLA queries.

There is some overlap between the types of queries that you can perform by using XMLA and DMX. For example, you can create some model content queries by using either DMX or the data mining schema rowsets, but the schema rowsets sometimes contain information that is not exposed in DMX content queries.

There are also some key differences in the way that operations are handled in DMX and in XMLA. For example, you can use XMLA to perform administrative operations such as backup of an entire Analysis Services database, but if you want to back up a single mining model, DMX provides a simple command, EXPORT (DMX), that is better suited to that purpose.

Build and Run a DMX Query

Open a new DMX Query window

1. Click **New Query** in Management Studio, and then select **New Analysis Server DMX Query**.
2. When the **Connect to Server** dialog box appears, select the instance of Analysis Services that contains the mining models you want to work with.

Open Template Explorer

1. In SQL Server Management Studio, on the **View** menu, select **Template Explorer**.
2. Click **Analysis Server** to see a tree view of the templates that apply to Analysis Services.

Apply a template to build a query

- Right-click the appropriate query type and select **Open**.
- Or, drag the template into the query editor.
- You can also fill in the parameters for the query by using the option, **Specify Values for Parameters**, from the **Query** menu.

For examples of how to create specific types of queries from templates, see the following topics:

Create a Singleton Prediction Query from a Template

Create a Content Query on a Mining Model

See Also

Data Mining Query Tools
Data Mining Extensions (DMX) Reference

Create a Singleton Query in the Data Mining Designer

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A singleton query is useful if you want to create a prediction for a single case. For more information about singleton queries, see Data Mining Queries.

In the **Mining Model Prediction** tab of Data Mining Designer, you can create many different types of queries. You can create a query by using the designer, or by typing Data Mining Extensions (DMX) statements. You can also start with the designer and modify the query that it creates by changing the DMX statements, or by adding a WHERE or ORDER BY clause.

To switch between the query design view and the query text view, click the first button on the toolbar. When you are in the query text view, you can view the DMX code that Prediction Query Builder creates. You can also run the query, modify the query, and run the modified query. However, the modified query is not persisted if you switch back to the query design view.

The following code shows an example of a singleton query against the targeted mailing model, TM_Decision_Tree.

```
SELECT [Bike Buyer], PredictProbability([Bike Buyer]) as ProbableBuyer
FROM [TM_Decision_Tree]
NATURAL PREDICTION JOIN
(SELECT '2' AS [Number Children At Home], '45' as [Age])
AS [t]
```

The following steps explain how to create this prediction query.

To create a singleton query by using the Data Mining Designer

1. Click the **Mining Model Prediction** tab in Data Mining Designer.

2. Click **Select Model** on the **Mining Model** table.

   The **Select Mining Model** dialog box opens to show all the mining structures that exist in the current project.

   Select the model that you want to use for creating a prediction.

   For example, to create the sample code shown at the start of this topic, select TM_Decision_Tree, and then click **OK**.

3. Click **Singleton query** on the toolbar of the **Mining Model Prediction** tab.

   The **Singleton Query Input** table appears on the tab, with the columns automatically mapped to the columns in the **Mining Model** table.

4. On the **Singleton Query Input** table, select values in the **Value** column to describe the case for which you want to create a prediction.

   For example, select **2** for **Number Children At Home**, and then type **45** for **Age**.

5. Drag a predictable column from the **Mining Model** table to the **Source** column at the bottom of the tab. Optionally, you can type an alias for the column.

For example, drag **Bike Buyer** to the **Source** column.

6. Add any additional functions to the query by selecting **Prediction Function** or **Custom Expression** from the drop-down list in the **Source** column.

   For example, click **Prediction Function**, and select **PredictProbability**.

7. Click **Criteria/Argument** in the **PredictProbability** row, and type the name of the column to predict, and optionally a specific value to predict.

   For example, type **[Bike Buyer], 1**.

8. Click the **Alias** box in the **PredictProbability** row, and type a name to refer to the new column.

   For example, type **ProbableBuyer**.

9. Click **Switch to query result view** on the toolbar of the **Mining Model Prediction** tab.

   A new screen opens to show the result of the query. To view the DMX statement that you just created, click **SQL**.

See Also

Prediction Queries (Data Mining)

Apply Prediction Functions to a Model

11/28/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

To create a prediction query in SQL Server Data Mining, you must first select the mining model on which the query will be based. You can select any mining model that exists in the current project.

After you have selected a model, add a *prediction function* to the query. A prediction function can be used to get a prediction, but you can also add prediction functions that return related statistics, such as a the probability of the predicted value, or information that was used in generating the prediction.

Prediction functions can return the following types of values:

- The name of the predictable attribute, and the value that is predicted.
- Statistics about the distribution and variance of the predicted values.
- The probability of a specified outcome, or of all possible outcomes.
- The top or bottom scores or values.
- Values associated with a specified node, object, or attribute.

The type of prediction functions that are available depend on the type of model you are working with. For example, prediction functions applied to decision tree models can return rules and node descriptions; prediction functions for time series models can return the lag and other information specific to time series.

For a list of the prediction functions that are supported for almost all model types, see General Prediction Functions (DMX).

For examples of how to query a specific type of mining model, see the algorithm reference topic, in Data Mining Algorithms (Analysis Services - Data Mining).

Choose a mining model to use for prediction

1. From SQL Server Management Studio, right-click the model, and select **Build Prediction Query**.

   --OR --

   In SQL Server Data Tools (SSDT), click the tab, **Mining Model Prediction**, and then click **Select Model** in the **Mining Model** table.

2. In the **Select Mining Model** dialog box, select a mining model, and then click **OK**.

   You can choose any model within the current Analysis Services database. To create a query using a model in a different database, you must either open a new query window in the context of that database, or open the solution file that contains that model.

Add prediction functions to a query

1. In the **Prediction Query Builder**, configure the input data used for prediction, either by providing values in the **Singleton Query Input** dialog box, or by mapping the model to an external data source.

   For more information, see Choose and Map Input Data for a Prediction Query.

> **WARNING**
> It is not required that you provide inputs to generate predictions. When there is no input, the algorithm will typically return the mostly likely predicted value across all possible inputs.

2. Click the **Source** column, and choose a value from the list:

<model name>	Select this option to include values from the mining model in the output. You can only add predictable columns. When you add a column from the model, the result returned is the non-distinct list of values in that column. The columns that you add with this option are included in the SELECT portion of the resulting DMX statement.
Prediction Function	Select this option to browse a list of prediction functions. The values or functions you select are added to the SELECT portion of the resulting DMX statement. The list of prediction functions is not filtered or constrained by the type of model you have selected. Therefore, if you have any doubt about whether the function is supported for the current model type, you can just add the function to the list and see if there is an error. List items that are preceded by $ (such as $AdjustedProbability) represent columns from the nested table that is output when you use the function, **PredictHistogram**. These are shortcuts that you can use to return a single column and not a nested table.
Custom Expression	Select this option to type a custom expression and then assign an alias to the output. The custom expression is added to the SELECT portion of the resulting DMX prediction query. This option is useful if you want to add text for output with each row, to call VB functions, or to call custom stored procedures. For information about using VBA and Excel functions from DMX, see VBA functions in MDX and DAX.

3. After adding each function or expression, switch to DMX view to see how the function is added within the DMX statement.

> **WARNING**
> The Prediction Query Builder does not validate the DMX until you click **Results**. Often, you will find that the expression that is produced by the query builder is not valid DMX. Typical causes are referencing a column that is not related to the predictable column, or trying to predict a column in a nested table, which requires a sub-SELECT statement. At this point, you can switch to DMX view and continue editing the statement.

Example: Create a query on a clustering model

1. If you do not have a clustering model available for building this sample query, create the model, [TM_Clustering], using the Basic Data Mining Tutorial.
2. From SQL Server Management Studio, right-click the model, [TM_Clustering], and select **Build Prediction Query**.
3. From the **Mining Model** menu, select **Singleton Query**.
4. In the **Singleton Query Input** dialog box, set the following values as inputs:
   - Gender = M
   - Commute Distance = 5-10 miles
5. In the query grid, for **Source**, select TM_Clustering mining model, and add the column, [Bike Buyer].
6. For **Source**, select **Prediction Function**, and add the function, **Cluster**.
7. For **Source**, select **Prediction Function**, add the function, **PredictSupport**, and drag the model column [Bike Buyer] into the **Criteria/Argument** box. Type **Support** in the **Alias** column.

   Copy the expression representing the prediction function and column reference from the **Criteria/Argument** box.
8. For **Source**, select **Custom Expression**, type an alias, and then reference the Excel CEILING function by using the following syntax:

   ```
   Excel![CEILING](<arguments) as <return type>
   ```

   Paste the column reference in as the argument to the function.

   For example, the following expression returns the CEILING of the support value:

   ```
   EXCEL!CEILING(PredictSupport([TM_Clustering].[Bike Buyer]),2)
   ```

   Type CEILING in the **Alias** column.
9. Click **Switch to query text view** to review the DMX statement that was generated, and then click **Switch to query result view** to see the columns output by the prediction query.

   The following table shows the expected results:

BIKE BUYER	$CLUSTER	SUPPORT	CEILING
0	Cluster 8	954	953.948638926372

If you want to add other clauses elsewhere in the statement-for example, if you want to add a WHERE clause-you cannot add it by using the grid; you must switch to DMX view first.

See Also

Data Mining Queries

Choose and Map Input Data for a Prediction Query

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you create predictions from a mining model, you generally do this by feeding new data into the model. (The exception is time series models, which can make predictions based on historical data only.) To provide the model with new data, you must make sure that the data is available as part of a data source view. If you know in advance which data you will use for prediction, you can include it in the data source view that you used to create the model. Otherwise, you might have to create a new data source view. For more information, see Data Source Views in Multidimensional Models.

Sometimes the data that you need might be contained within more than one table in a one-to-many join. This is the case with data used for association models or sequence clustering models, which use a case table linked to a nested table that contains product or transaction details. If your model uses a case-nested table structure, the data that you use for prediction must also have a case-nested table structure.

> **WARNING**
>
> You cannot add new columns or map columns that are in a different data source view. The data source view that you select must contain all the columns that you need for the prediction query.

After you have identified the tables that contain the data you will use for predictions, you must map the columns in the external data to the columns in the mining model. For example, if your model predicts customer purchasing behavior based on demographics and survey responses, your input data should contains information that generally corresponds to what is in the model. You do not need to have matching data for every single column, but the more columns you can match, the better. If you try to map columns that have different data types, you might get an error. In that case, you could define a named calculation in the data source view to cast or convert the new column data to the data type required by the model. For more information, see Define Named Calculations in a Data Source View (Analysis Services).

When you choose the data to use for prediction, some columns in the selected data source might be automatically mapped to the mining model columns, based on name similarity and matching data type. You can use the **Modify Mapping** dialog box in the **Mining Model Prediction** to change the columns that are mapped, delete inappropriate mappings, or create new mappings for existing columns. The **Mining Model Prediction** design surface also supports drag-and-drop editing of connections.

- To create a new connection, just select a column in the **Mining Model** table and drag it to the corresponding column in the **SelectInput Table(s)** table.
- To remove a connection, select the connection line and press the DELETE key.

The following procedure describes how you can modify the joins that have been created between the case table and a nested table used as inputs to a prediction query, using the **Specify Nested Join** dialog box.

Select an input table

1. On the **Select Input Table(s)** table of the **Mining Accuracy Chart** tab in Data Mining Designer in SQL Server Data Tools (SSDT), click **Select Case Table**.

   The **Select Table** dialog box opens, in which you can select the table that contains the data on which to base your queries.

2. In the **Select Table** dialog box, select a data source from the **Data Source** list.
3. Under **Table/View Name**, select the table that contains the data you want to use to test the models.
4. Click **OK**.

   The columns in the mining structure are automatically mapped to the columns that have the same name in the input table.

Change the way that input data is mapped to the model

1. In Data Mining Designer in SQL Server Data Tools (SSDT), select the **Mining Model Prediction** tab.
2. On the **Mining Model** menu, select **Modify Connections**.

   The **Modify Mapping** dialog box opens. In this dialog box, the column **Mining Model Column** lists the columns in the selected mining structure. The column **Table Column** lists the columns in the external data source that you chose in the **SelectInput Table(s)** dialog box. The columns in the external data source are mapped to columns in the mining model.
3. Under **Table Column**, select the row that corresponds to the mining model column that you want to map to.
4. Select a new column from the list of available columns in the external data source. Select the blank item in the list to delete the column mapping.
5. Click **OK**.

   The new column mappings are displayed in the designer.

Remove a relationship between input tables

1. On the **Select Input Table(s)** table of the **Mining Model Prediction** tab in Data Mining Designer in SQL Server Data Tools (SSDT), click **Modify Join**.

   The **Specify Nested Join** dialog box opens.
2. Select a relationship.
3. Click **Remove Relationship**.
4. Click **OK**.

   The relationship between the case table and the nested table has been removed.

Create a new relationship between input tables

1. On the **Select Input Table(s)** table of the **Mining Model Prediction** tab in Data Mining Designer, click **Modify Join**.

   The **Specify Nested Join** dialog box opens.
2. Click **Add Relationship**.

   The **Create Relationship** dialog box opens.
3. Select the key of the nested table in **Source Columns**.
4. Select the key of the case table in **Destination Columns**.
5. Click **OK** in the **Create Relationship** dialog box.
6. Click **OK** in the **Specify Nested Join** dialog box.

   A new relationship has been created between the case table and the nested table.

Add a nested table to the input tables of a prediction query

1. On the **Mining Model Prediction** tab in Data Mining Designer, click **Select Case Table** to open the **Select Table** dialog box.

   > **NOTE**
   > You cannot add a nested table to the inputs unless you have specified a case table. Use of a nested table requires that the mining model you are using for prediction also uses a nested table.

2. In the **Select Table** dialog box, select a data source from the **Data Source** list, and select the table in the data source view that contains the case data. Click **OK**.

3. Click **Select Nested Table** to open the **Select Table** dialog box.

4. In the **Select Table** dialog box, select a data source from the **Data Source** list, and select the table in the data source view that contains the nested data. Click **OK**.

   If a relationship already exists, the columns in the mining model are automatically mapped to the columns that have the same name in the input table. You can modify the relationship between the nested table and the case table by clicking **Modify Join**, which opens the **Create Relationship** dialog box.

See Also

Prediction Queries (Data Mining)

Manually Edit a Prediction Query

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you have designed a query by using the Prediction Query Builder, you can modify the query by switching to Query Text view on the **Mining Model Prediction** tab of Data Mining Designer. A text editor appears at the bottom of the screen to display the query that the query builder created.

Switching to Query Text view is useful for making additions to the query. For example, you can add a WHERE clause or ORDER BY clause.

Use the grid in the Prediction Query Builder to insert the names of objects and columns and set up the syntax for individual prediction functions, and then switch to manual editing mode to change parameter values.

> **NOTE**
> If you switch back to **Design** view from **Query Text** view, any changes that you made in **Query Text** view will be lost.

Modify a query

1. On the **Mining Model Prediction** tab in Data Mining Designer in SQL Server Data Tools (SSDT), click **SQL**.

   The grid at the bottom of the screen is replaced by a text editor that contains the query. You can type changes to the query in this editor.

2. To run the query, on the **Mining Model** menu, select **Result**, or click the button to switch to query results.

   > **NOTE**
   > If the query that you have created is invalid, the Results window does not display an error and does not display any results. Click the **Design** button, or select **Design** or **Query** from the **Mining Model** menu to correct the problem and run the query again.

See Also

Data Mining Queries
Prediction Query Builder (Data Mining)
Lesson 6: Creating and Working with Predictions (Basic Data Mining Tutorial)

View and Save the Results of a Prediction Query

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you have defined a query in Microsoft SQL Server Analysis Services by using Prediction Query Builder, you can run the query and view the results by switching to the query result view.

You can save the results of a prediction query to a table in any data source that is defined in a Microsoft SQL Server Analysis Services project. You can either create a new table or save the query results to an existing table. If you save the results to an existing table, you can choose to overwrite the data that is currently stored in the table; otherwise, the query results are appended to the existing data in the table.

Run a query and view the results

1. On the toolbar of the **Mining Model Prediction** tab of Data Mining Designer in SQL Server Data Tools (SSDT), click **Result** .

   The query results view opens and runs the query. The results are displayed in a grid in the viewer.

Save the results of a prediction query to a table

1. On the toolbar of the **Mining Model Prediction** tab in Data Mining Designer, click **Save query result**.

   The **Save Data Mining Query Result** dialog box opens.

2. Select a data source from the **Data Source** list, or click **New** to create a new data source.

3. In the **Table Name** box, enter the name of the table. If the table already exists, select **Overwrite if exists** to replace the contents of the table with the query results. If you do not want to overwrite the contents of the table, do not select this check box. The new query results will be appended to the existing data in the table.

4. Select a data source view from **Add to DSV** if you want to add the table to a data source view.

5. Click **Save**.

> **WARNING**
>
> If the destination does not support hierarchical rowsets, you can add the FALTTENED keyword to the results to save as a flat table.

Create a Content Query on a Mining Model

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can query the mining model content programmatically by using AMO or XML/A, but it is easier to create queries by using DMX. You can also create queries against the data mining schema rowsets by establishing a connection to the Analysis Services instance and creating a query using the DMVs provided by Analysis Services.

The following procedures demonstrate how to create queries against a mining model by using DMX, and how to query the data mining schema rowsets.

For an example of how to create a similar query by using XML/A, see Create a Data Mining Query by Using XMLA.

Querying Data Mining Model Content by Using DMX

To create a DMX model content query

1. In SQL Server Management Studio, on the **View** menu, click **Template Explorer**.
2. In the **Template Explorer** pane, click the cube icon to change the list and display Analysis Services templates.
3. In the list of template categories, expand **DMX**, expand **Model Content**, and double-click **Content Query**.
4. In the **Connect to Analysis Services** dialog box, select the instance that contains the mining model you want to query, and click **Connect**.

   The **Content Query** template opens in the appropriate code editor. The metadata pane lists the models that are available in the current database. To change the database, select a different database from the **Available Databases** list.
5. Enter the name of a mining model in the line, `FROM` [<*mining model, name, MyModel*>] `.CONTENT`. If the mining model name contains spaces, you must enclose the name in brackets.

   If you don't want to type the name, you can select a mining model in **Object Explorer** and drag it into the template.
6. In the line, `SELECT` <*select list, expr list, **>, type the names of columns in the mining model content schema rowset.

   To view a list of columns that you can return in mining model content queries, see Mining Model Content (Analysis Services - Data Mining).
7. Optionally, type a condition in the WHERE clause of the template to restrict the rows returned to specific nodes or values.
8. Click **Execute**.

Querying the Data Mining Schema Rowsets

To create a query against the data mining schema rowset

1. In SQL Server Management Studio, on the **New Query** toolbar, click **Analysis Services DMX Query**, or **Analysis Services MDX query**.

2. In the **Connect to Analysis Services** dialog box, select the instance that contains the objects you want to query, and click **Connect**.

   The **Content Query** template opens in the appropriate code editor. The metadata pane lists the objects that are available in the current database. To change the database, select a different database from the **Available Databases** list.

3. In the query editor, type the following:

   ```
   SELECT *
   FROM $system.DMSCHEMA_MINING_MODEL_CONTENT
   WHERE MODEL_NAME = '<model name>'
   ```

4. Click **Execute**.

   The Results pane displays the contents of the model.

> **NOTE**
>
> To view a list of all the schema rowsets that you can query on the current instance, use this query: `SELECT * FROM $system.` DISCOVER_SCHEMA_ROWSETS. Or, for a list of schema rowsets specific to data mining, see Data Mining Schema Rowsets.

See Also

Mining Model Content (Analysis Services - Data Mining)
Data Mining Schema Rowsets

Query the Parameters Used to Create a Mining Model

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The composition of a mining model is affected not only by the training cases, but by the parameters that were set when the model was created. Therefore, you might find it useful to retrieve the parameter settings of an existing model to better understand the behavior of the model. Retrieving parameters is also useful when documenting a particular version of that model.

To find the parameters that were used when the model was created, you create a query against one of the mining model schema rowsets. These schema rowsets are exposed as a set of system views that you can query easily by using Transact-SQL syntax. This procedure describes how to create a query that returns the parameters that were used to create the specified mining model.

To open a Query window for a schema rowset query

1. In SQL Server Management Studio, open the instance of Analysis Services that contains the model you want to query.
2. Right-click the instance name, select **New Query**, and then select **DMX**.

   > **NOTE**
   > You can also create a query against a data mining model by using the **MDX** template.

3. If the instance contains multiple databases, select the database that contains the model you want to query from the **Available Databases** list in the toolbar.

To return model parameters for an existing mining model

1. In the DMX query pane, type or paste the following text:

   ```
   SELECT MINING_PARAMETERS
   FROM $system.DMSCHEMA_MINING_MODELS
   WHERE MODEL_NAME = ''
   ```

2. In Object Explorer, select the mining model you want, and then drag it into the DMX Query pane, between the single quotation marks.
3. Press F5, or click **Execute**.

Example

The following code returns a list of the parameters that were used to create the mining model that you build in the Basic Data Mining Tutorial. These parameters include the explicit values for any defaults used by the mining services available from providers on the server.

```
SELECT MINING_PARAMETERS
FROM $system.DMSCHEMA_MINING_MODELS
WHERE MODEL_NAME = 'TM Clustering'
```

The code example returns the following parameters for the clustering model:

eExample Results:

MINING_PARAMETERS

CLUSTER_COUNT=10,CLUSTER_SEED=0,CLUSTERING_METHOD=1,MAXIMUM_INPUT_ATTRIBUTES=255,MAXIMUM_STATES=100,MINIMUM_SUPPORT=1,MODELLING_CARDINALITY=10,SAMPLE_SIZE=50000,STOPPING_TOLERANCE=10

See Also

Data Mining Query Tasks and How-tos
Data Mining Queries

Create a Singleton Prediction Query from a Template

11/27/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A singleton query is useful when you have a model that you want to use for prediction, but don't want to map it to an external input data set or make bulk predictions. With a singleton query, you can provide a value or values to the model and instantly see the predicted value.

For example, the following DMX query represents a singleton query against the targeted mailing model, TM_Decision_Tree.

```
SELECT * FROM [TM_Decision_tree] ;
NATURAL PREDICTION JOIN
(SELECT '2' AS [Number Children At Home], '45' as [Age])
AS [t]
```

The procedure that follows describes how to use the Template Explorer in SQL Server Management Studio to quickly create this query.

To open the Analysis Services templates in SQL Server Management Studio

1. In SQL Server Management Studio, on the **View** menu, click **Template Explorer**.
2. Click the cube icon to open the **Analysis Server**templates.

To open a prediction query template

1. In **Template Explorer**, in the list of Analysis Server templates, expand **DMX**, and thenexpand **Prediction Queries**.
2. Double-click **Singleton Prediction**.
3. In the **Connect to Analysis Services** dialog box, type the name of the server that has the instance of Analysis Services that contains the mining model to be queried.
4. Click **Connect**.
5. The template opens in the specified database, together with a mining model Object Browser that contains data mining functions and a list of data mining structures and related models.

To customize the singleton query template

1. In the template, click the **Available Databases** drop-down list, and then select an instance of Analysis Service from the list.
2. In the **Mining Model** list, select the mining model that you want to query.

   The list of columns in the mining model appears in the **Metadata** pane of the object browser.
3. On the **Query** menu, select **Specify Values for Template Parameters**.
4. In the **select list** row, type * to return all columns, or type a comma-delimited list of columns and expressions to return specific columns.

   If you type *, the predictable column is returned, together with any columns for which you provide new

values for in step 6.

For the sample code shown at the start of this topic, the **select list** row was set to *.

5. In the **mining model** row, type the name of the mining model from among the list of mining models that appear in **Object Explorer**.

   For the sample code shown at the start of this topic, the **mining model** row was set to the name, **TM_Decision_Tree**.

6. In the **value** row, type the new data value for which you want to make a prediction.

   For the sample code shown at the start of this topic, the **value** row was set to **2** to predict bike buying behavior based on the number of children at home.

7. In the **column** row, type the name of the column in the mining model to which the new data should be mapped.

   For the sample code shown at the start of this topic, the **column** row was set to **Number Children at Home**.

   > **NOTE**
   > When you use the **Specify Values for Template Parameters** dialog box, you do not have to add square brackets around the column name. The brackets will automatically be added for you.

8. Leave the **input alias** as **t**.

9. Click **OK**.

10. In the query text pane, find the red squiggle under the comma and ellipsis that indicates a syntax error. Delete the ellipsis, and add any additional query condition that you want. If you do not add any other conditions, delete the comma.

    For the sample code shown at the start of this topic, the additional query condition was set to **'45' as [Age]**.

11. Click **Execute**.

See Also

Creating Predictions (Basic Data Mining Tutorial)

Create a Data Mining Query by Using XMLA

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can create a variety of queries against data mining objects by using AMO, DMX, or XML/A.

XML is used for communication between the Analysis Services server and all clients. Therefore, although it is generally much easier to create content queries by using DMX, you can write queries by using the DISCOVER and COMMAND statements in XML/A, either by using a client that supports the SOAP protocol, or by creating an XML/A query in SQL Server Management Studio.

This topic explains how to use the XML/A templates that are available in SQL Server Management Studio to create a model content query against a mining model stored on the current server.

Querying Data Mining Schema Rowsets by Using XML/A

To open an XML/A template

1. In SQL Server Management Studio, on the **View** menu, click **Template Explorer**.
2. Click the cube icon to open the list of Analysis Services templates.
3. In the list of template categories, expand **XMLA**, expand **Schema Rowsets**, and double-click **Discover Schema Rowsets** to open the template in the appropriate code editor.
4. In the **Connect to Analysis Services** dialog box, complete the connection information and then click **Connect**. A new query editor window opens, populated with the **Discover Schema Rowsets** template.

To discover column names from the MINING MODEL CONTENT schema rowset

1. With the **Discover Schema Rowsets** template open, click **Execute**.

   A list of schema rowsets is returned in the **Results** pane that contains the rowset names and rowset columns for all rowsets available on the current instance.

2. In the **Query** pane, place the cursor after **<Restriction List>** and press ENTER to add a new line.
3. Place the cursor on the blank line and type **<SchemaName>DMSCHEMA_MINING_MODEL_CONTENT</SchemaName>**

   The complete section for restrictions should appear as follows:

   ```
   <Restrictions>
   <RestrictionList>
   <SchemaName>DMSCHEMA_MINING_MODEL_CONTENT</SchemaName>
   </RestrictionList>
   </Restrictions>
   ```

4. Click **Execute**.

   The **Results** pane shows a list of column names for the specified schema rowset.

To create a content query using the MINING MODEL CONTENT schema rowset

1. In the **Discover Schema Rowsets** template, change the request type by replacing the text inside the

request type tags.

Replace this line:

```
<RequestType>DISCOVER_SCHEMA_ROWSETS</RequestType>
```

with the following line:

<RequestType>DMSCHEMA_MINING_MODEL_CONTENT</RequestType>

2. Change the restriction list to specify a mining model by name, by adding a new condition to the restriction lists.

3. In the template, place the cursor after `<Restriction List>` and press ENTER to add a new line.

4. Place the cursor on the blank line and type **<MODEL_NAME>My model name</MODEL_NAME>**

   The complete section for restrictions should appear as follows:

   ```
   <Restrictions>
   <RestrictionList>
   <MODEL_NAME>My model name</MODEL_NAME>
   </RestrictionList>
   </Restrictions>
   ```

5. Click **Execute**.

   The Results pane displays the schema definition, together with the values for the specified model.

See Also

Mining Model Content (Analysis Services - Data Mining)
Data Mining Schema Rowsets

Change the Time-out Value for Data Mining Queries

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When you build a lift chart or execute a prediction query, sometimes it can take a long time to generate all the data required for the prediction. To prevent the query from timing out, you can change the value that controls how long the Analysis Services server waits to complete a query.

The default value is 15; however, if your models are complex or the data source is large, this might not be enough. If necessary, you can increase the value significantly, to enable enough time for processing. For example, if you set **Query Timeout** to 600, the query could continue to run for up to 10 minutes.

For more information about prediction queries, see Data Mining Query Tasks and How-tos.

Configure the time-out value for data mining queries

1. In SQL Server Data Tools (SSDT), from the **Tools** menu, selection **Options**.
2. In the **Options** pane, expand **Business Intelligence Designers**.
3. Click the **Query Timeout** text box, and type a value for the number of seconds.

See Also

Data Mining Query Tasks and How-tos
Data Mining Queries

Data Mining Solutions

11/28/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A data mining solution is an Analysis Services solution that contains one or more data mining projects.

The topics in this section provide information about how to design and implement an integrated data mining solution by using SQL Server Analysis Services. For an overview of the data mining design process and related tools, see Data Mining Concepts.

For more information about additional projects types that are useful for data mining, see Related Projects for Data Mining Solutions.

Relational vs. Multidimensional Solutions

Deploying Data Mining Solutions

Solution Walkthroughs

Relational vs. Multidimensional Solutions

A data mining solution can be based either on multidimensional data-that is, an existing cube-or on purely relational data, such as the tables and views in a data warehouse, or on text files, Excel workbooks, or other external data sources.

- You can create data mining objects within an existing multidimensional database solution.

  Typically you would create a solution like this if you have already created a cube and want to perform data mining by using the cube as a data source. When you move and backup models based on a cube, the cube must also be moved or copied.

- You can create a data mining solution that contains only data mining objects, including the supporting data sources and data source views, and that uses relational data source only.

  This is the preferred method for creating data mining models, as processing and querying is generally fastest against relational data sources. You can also easily move and backup models between servers by using the EXPORT and IMPORT commands.

Deploying Data Mining Solutions

The instance of Analysis Services to which you deploy the solution must be running in a mode that supports multidimensional objects and data mining objects; that is, you cannot deploy data mining objects to an instance that hosts tabular models or Power Pivot data.

Therefore, when you create a data mining solution in Visual Studio, be sure to use the template, **Analysis Services Multidimensional and Data Mining Project**.

When you deploy the solution, the objects used for data mining are created in the specified Analysis Services instance, in a database with the same name as the solution file.

For more information about how to deploy both relational and multidimensional solutions, see Deployment of Data Mining Solutions.

Solution Walkthrough

Provides an overview of how to create data mining solutions by using the Data Mining Wizard.

Create a Relational Mining Structure
Create a mining structure from relational data, text files, and other sources that can be combined in a data source view.

Create an OLAP Mining Structure
Create a mining structure based on data in an OLAP cube. Models that you create from OLAP data can be saved as a data mining dimension, or you can save the set of data and your models as a new cube.

In This Section

Data Mining Projects

Processing Data Mining Objects

Related Projects for Data Mining Solutions

Deployment of Data Mining Solutions

Related Tasks and Topics

After you have created a basic data mining solution, including data sources and a mining structure, you can build on the solution by adding new models, testing and comparing models, creating predictions, and experimenting with subsets of data.

For more information, see the following links:

TASKS	TOPICS
Test the models you create, validate the quality of your training data, and create charts that represent the accuracy of data mining models.	Testing and Validation (Data Mining)
Train the model by populating the structure and related models with data. Update and extend models with new data.	Processing Data Mining Objects
Customize a mining model by applying filters to the training data, choosing a different algorithm, or setting advanced algorithm parameters.	Customize Mining Models and Structure
Customize a mining model by applying filters to the data used in training the mode.	Add Mining Models to a Structure (Analysis Services - Data Mining)
Update and manage data mining solutions.	Link TBD

See Also

Data Mining Tutorials (Analysis Services)

Data Mining Projects

11/28/2018 • 14 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A data mining project is part of an Analysis Services solution. During the design process, the objects that you create in this project are available for testing and querying as part of a workspace database. When you want users to be able to query or browse the objects in the project, you must deploy the project to an instance of Analysis Services running in multidimensional mode.

This topic provides you with the basic information needed to understand and create data mining projects.

Creating Data Mining Projects

Objects in Data Mining Projects

- Data sources
- Data source views
- Mining structures
- Mining models

Using the Completed Data Mining Project

- View and explore models
- Test and validate models
- Create predictions

Programmatic Access to Data Mining Projects

Creating Data Mining Projects

In SQL Server Data Tools (SSDT), you build data mining projects using the template, **OLAP and Data Mining Project**. You can also create data mining projects programmatically, by using AMO. Individual data mining objects can be scripted using the Analysis Services Scripting language (ASSL). For more information, see Multidimensional Model Data Access (Analysis Services - Multidimensional Data).

If you create a data mining project within an existing solution, by default the data mining objects will be deployed to an Analysis Services database with the same name as the solution file. You can change this name and the target server by using the **Project Properties** dialog box. For more information, see Configure Analysis Services Project Properties (SSDT).

> **WARNING**
>
> To successfully build and deploy your project, you must have access to an instance of Analysis Services that is running in OLAP/Data Mining mode. You cannot develop or deploy data mining solutions on an instance of Analysis Services that supports tabular models, nor can you use data directly from a Power Pivot workbook or from a tabular model that uses the in-memory data store. To determine whether the instance of Analysis Services that you have can support data mining, see Determine the Server Mode of an Analysis Services Instance.

Within each data mining project that you create, you will follow these steps:

1. Choose a *data source*, such as a cube, database, or even Excel or text files, which contains the raw data you will use for building models.
2. Define a subset of the data in the data source to use for analysis, and save it as a *data source view*.
3. Define a *mining structure* to support modeling.
4. Add *mining models* to the mining structure, by choosing an *algorithm* and specifying how the algorithm will handle the data.
5. Train models by populating them with the selected data, or a filtered subset of the data.
6. Explore, test, and rebuild models.

When the project is complete, you can deploy it for users to browse or query, or provide programmatic access to the mining models in an application, to support predictions and analysis.

Objects in Data Mining Projects

All data mining projects contain the following four types of objects. You can have multiple objects of all types.

- Data sources
- Data source views
- Mining structures
- Mining models

For example, a single data mining project can contain a reference to multiple data sources, with each data source supporting multiple data source views. In turn, each data source view can support multiple mining structures, each with many related mining models.

Additionally, your project might include plug-in algorithms, custom assemblies, or custom stored procedures; however, these objects are not described here. For more information, see Analysis Services Developer Documentation.

Data Sources

The data source defines the connection string and authentication information that the Analysis Services server will use to connect to the data source. The data source can contain multiple tables or views; it can be as simple as a single Excel workbook or text file, or as complex as an Online Analytical Processing (OLAP) database or large relational database.

A single data mining project can reference multiple data sources. Even though a mining model can use only one data source at a time, the project could have multiple models drawing on different data sources.

Analysis Services supports data from many external providers, and SQL Server Data Mining can use both relational and cube data as a data source. However, if you develop both types of projects-models based on relational sources and models based on OLAP cubes-you might wish to develop and manage these in separate projects.

- Typically models that are based on an OLAP cube should be developed within the OLAP design solution. One reason is that models based on a cube must process the cube to update data. Generally, you should use cube data only when that is the principal means of data storage and access, or when you require the aggregations, dimensions, and attributes created by the multidimensional project.
- If your project uses relational data only, you should create the relational models within a separate project, so that you do not unnecessarily reprocess other objects. In many cases, the staging database or the data warehouse used to support cube creation already contains the views that are needed to perform data

mining, and you can use those views for data mining rather than use the aggregations and dimensions in the cube.

- You cannot use in-memory or Power Pivot data directly to build data mining models.

The data source only identifies the server or provider and the general type of data. If you need to change data formatting and aggregations, use the data source view object.

To control the way that data from the data source is handled, you can add derived columns or calculation, modify aggregates, or rename columns in the data in the data source view. (You can also work with data downstream, by modifying mining structure columns, or by using modeling flags and filters at the level of the mining model column.)

If data cleansing is required, or the data in the data warehouse must be modified to create additional variables, change data types, or create alternate aggregation, you might need to create additional project types in support of data mining. For more information about these related projects, see Related Projects for Data Mining Solutions.

Data Source Views

After you have defined this connection to a data source, you create a view that identifies the specific data that is relevant to your model.

The data source view also enables you to customize the way that the data in the data source is supplied to the mining model. You can modify the structure of the data to make it more relevant to your project, or choose only certain kinds of data.

For example, by using the Data Source View editor, you can:

- Create derived columns, such as dateparts, substrings, etc.
- Aggregate values using Transact-SQL statements such as GROUP BY
- Restrict data temporarily, or sample data

For more information about how you can modify data within a data source view, see Data Source Views in Multidimensional Models.

> **WARNING**
> If you want to filter the data, you can do so in the data source view, but you can also create filters on the data at the level of the mining model. Because the filter definition is stored with the mining model, using model filters makes it easier to determine the data that was used for training the model. Moreover, you can create multiple related models, with different filter criteria. For more information, see Filters for Mining Models (Analysis Services - Data Mining).

Note that the data source view that you create can contain additional data that is not directly used for analysis. For example, you might add to your data source view data that is used for testing, predictions, or for drillthrough. For more information about these uses, see Testing and Validation (Data Mining) and Drillthrough.

Mining Structures

Once you have created your data source and data source view, you must select the columns of data that are most relevant to your business problem, by defining *mining structures* within the project. A mining structure tells the project which columns of data from the data source view should actually be used in modeling, training, and testing.

To add a new mining structure, you start the Data Mining Wizard. The wizard automatically defines a mining structure, walks you through the process of choosing the data, and optionally lets you add an initial mining model to the structure. Within the mining structure, you choose tables and columns from the data source view or from an OLAP cube, and define relationships among tables, if your data includes nested tables.

Your choice of data will look very different in the Data Mining Wizard, depending on whether you use relational or

online analytical processing (OLAP) data sources.

- When you choose data from a relational data source, setting up a mining structure is easy: you choose columns from the data in the data source view, and set additional customizations such as aliases, or define how values in the column should be grouped or binned. For more information, see Create a Relational Mining Structure.
- When you use data from an OLAP cube, the mining structure must be in the same database as the OLAP solution. To create a mining structure, you select attributes from the dimensions and related measures in your OLAP solution. Numeric values are typically found in measures, and categorical variables in dimensions. For more information, see Create an OLAP Mining Structure.
- You can also define mining structures by using DMX. For more information, see Data Mining Extensions (DMX) Data Definition Statements.

After you have created the initial mining structure, you can copy, modify, and alias the structure columns.

Each mining structure can contain multiple mining models. Therefore, after you are done, you can open the mining structure again, and use Data Mining Designer to add more mining models to the structure.

You also have the option to separate your data into a training data set, used for building models, and a holdout data set to use in testing or validating your mining models.

> **WARNING**
> Some model types, such as time series models, do not support the creation of holdout data sets because they require a continuous series of data for training. For more information, see Training and Testing Data Sets.

Mining Models

The mining model defines the algorithm, or the method of analysis that you will use on the data. To each mining structure, you add one or more mining models.

Depending on your needs, you can combine many models in a single project, or create separate projects for each type of model or analytical task.

After you have created a structure and model, you *process* each model by running the data from the data source view through the algorithm, which generates a mathematical model of the data. This process is also known as *training the model*. For more information, see Processing Requirements and Considerations (Data Mining).

After the model has been processed, you can then visually explore the mining model and create prediction queries against it. If the data from the training process has been cached, you can use *drillthrough* queries to return detailed information about the cases used in the model.

When you want to use a model for production (for example, for use in making predictions, or for exploration by general users) you can deploy the model to a different server. If you need to reprocess the model in future, you must also export the definition of the underlying mining structure (and, necessarily, the definition of the data source and data source view) at the same time.

When you deploy a model, you must also ensure that the correct processing options are set on the structure and model, and that potential users have the permissions they need to perform queries, view models, or drillthrough to structure o model data. For more information, see Security Overview (Data Mining).

Using the Completed Data Mining Project

This section summarizes the ways that you can use the completed data mining project. You can create accuracy charts, explore and validate the data, and make the data mining patterns available to users.

> **WARNING**
> The charts, queries, and visualizations that you use with data mining models are not saved as part of the data mining project, and cannot be deployed. If you need to persist these objects, you must either save the content that is presented or script it as described for each object.

View and Explore Models

After you have created a model, you can use visual tools and queries to explore the patterns in the model and learn more about the underlying patterns and statistics. On the **Mining Model Viewer** tab in Data Mining Designer, Analysis Services provides viewers for each mining model type, which you can use to explore the mining models.

These visualizations are temporary, and are closed without saving when you exit the session with Analysis Services. Therefore, if you need to export these visualizations to another application for presentation or further analysis, use the **Copy** commands provided in each tab or pane of the viewer interface.

The Data Mining Add-ins for Excel also provides a Visio template that you can use to represent your models in a Visio diagram and annotate and modify the diagram using Visio tools. For more information, see Microsoft SQL Server 2008 SP2 Data Mining Add-ins for Microsoft Office 2007.

Test and Validate Models

After you have created a model, you can investigate the results and make decisions about which models perform the best.

Analysis Services provides several charts that you can use to provides tools that you can use to directly compare mining models and choose the most accurate or useful mining model. These tools include a lift chart, profit chart, and a classification matrix. You can generate these charts by using the **Mining Accuracy Chart** tab of Data Mining Designer.

You can also use the cross-validation report to perform iterative subsampling of your data to determine whether the model is biased to a particular set of data. The statistics that the report provides can be used to objectively compare models and assess the quality of your training data.

Note that these reports and charts are not stored with the project or in the ssASnoversion database, so if you need to preserve or duplicate the results, you should either save the results, or script the objects by using DMX or AMO. You can also use stored procedures for cross-validation.

For more information, see Testing and Validation (Data Mining).

Create Predictions

Analysis Services provides a query language called Data Mining Extensions (DMX) that is the basis for creating predictions and is easily scriptable. To help you build DMX prediction queries, SQL Server provides a query builder, available in SQL Server Management Studio. There are also many DMX templates for the query editor in SQL Server Management Studio.If you are new to prediction queries, we recommend that you use the query builder that is provided in both Data Mining Designer and SQL Server Management Studio. For more information, see Data Mining Tools.

The predictions that you create in either SQL Server Data Tools (SSDT) or SQL Server Management Studio are not persisted, so if your queries are complex, or you need to reproduce the results, we recommend that you save your prediction queries to DMX query files, script them, or embed the queries as part of an Integration Services package.

Programmatic Access to Data Mining Objects

Analysis Services provides several tools that you can use to programmatically work with data mining projects and the objects in them. The DMX language provides statements that you can use to create data sources and data

source views, and to create, train, and use data mining structure and models. For more information, see Data Mining Extensions (DMX) Reference.

You can also perform these tasks by using the Analysis Services Scripting Language (ASSL), or by using Analysis Management Objects (AMO). For more information, see Developing with XMLA in Analysis Services.

Related Tasks

The following topics describe use of the Data Mining Wizard to create a data mining project and associated objects.

TASKS	TOPICS
Describes how to work with mining structure columns	Create a Relational Mining Structure
Provides more information about how to add new mining models, and process a structure and models	Add Mining Models to a Structure (Analysis Services - Data Mining)
Provides links to resources that help you customize the algorithms that build mining models	Customize Mining Models and Structure
Provides links to information about each of the mining model viewers	Data Mining Model Viewers
Learn how to create a lift chart, profit chart, or classification matrix, or test a mining structure	Testing and Validation (Data Mining)
Learn about processing options and permissions	Processing Data Mining Objects
Provides more information about Analysis Services	Multidimensional Model Databases

See Also

Data Mining Designer
Creating Multidimensional Models Using SQL Server Data Tools (SSDT)
Workspace Database

Import a Data Mining Project using the Analysis Services Import Wizard

11/28/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes how to create a new data mining project by importing the metadata from an existing data mining project on another server, using the template, **Import from Server (Multidimensional and Data Mining) Project**, in SQL Server Data Tools (SSDT).

Import data sources, mining structures, and mining models from an existing data mining project

When you use the template, **Import from Server (Multidimensional and Data Mining) Project**, SQL Server Data Tools (SSDT) creates a new data mining project, and then copies the metadata from the specified data mining project. The new project contains the same data sources, data source views, mining structures, and mining models as the ssASnoversion database that you imported from. However, the project cannot be used until you have updated certain properties and processed the objects as described:

- The data itself is not copied from the source server to the new data mining project-only the definitions of the data sources and data source views are imported. Therefore, after the import process has completed, and the objects have been created, you must populate the objects with data by training the mining structures and dependent models. You can use the command **Process All** in Data Mining Designer to train the models and structures.

- If you are importing a project that was created in a previous version of Analysis Services, the data source might use providers that are not installed on the server to which you are importing the project. If you encounter errors when processing the imported mining structures, right-click each data source and select **Open Designer** to edit the connection string and review the provider properties.

  At this time, you might also need to verify that the account you are using to process the data mining objects or query data mining models has the necessary permissions on the data source.

- By default, when you import a project, the workspace database is set to localhost, or whatever default instance is configured as the **Default Target Server** in SQL Server Data Tools (SSDT). To set this property, from the **Options** menu, select **Business Intelligence Designers**, select **Analysis Services**, and then select **General**.

  Note that, in SQL Server 2017, there is another, separate option that you can set to configure the default deployment server for Analysis Services tabular model projects. The setting, **Default Deployment Server**, determines the default workspace database for tabular model projects. You cannot use instances that support tabular models for data mining projects

  If you cannot change the default deployment database to use an instance of Analysis Services running in multidimensional or data mining mode, you can always specify the deployment database by using the **Project Properties** dialog box.

To create a new data mining project by importing an existing data mining project

1. In SQL Server Data Tools, on the **File** menu, click **New**, and then click **Project**.

2. In the **New Project** dialog box, under **Installed Templates**, click **Business Intelligence**, click **Analysis**

Services, and then click **Import from Server (Multidimensional/Data Mining)**.

3. For **Name**, type a name for the project, then specify a location and solution name, and then click **OK**.

   The **Import Analysis Services Database wizard** starts. Click OK on the Welcome page to proceed.

4. On the page, **Select Source Database**, for **Server**, specify the Analysis Services instance that contains the solution you want to import.

   For **Database**, choose the Analysis Services database that contains the data mining objects you want to import.

   > **WARNING**
   > You cannot specify the objects you want to import; when you choose an existing Analysis Services database, all multidimensional and data mining objects are imported.

   Click **Next**.

5. The page, **Completing the Wizard**, displays the progress of the import operation. You cannot cancel the operation or change the objects that are being imported. Click **Finish** when done.

   The new project is automatically opened using SQL Server Data Tools (SSDT).

See Also

Project Properties

Processing Data Mining Objects

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A data mining object is only an empty container until it has been processed. *Processing* a data mining model is also called *training*.

Processing mining structures: A mining structure gets data from an external data source, as defined by the column bindings and usage metadata, and reads the data. This data is read in full and then analyzed to extract various statistics. Analysis Services stores a compact representation of the data, which is suitable for analysis by data mining algorithms, in a local cache. You can either keep this cache or delete it after your models have been processed. By default, the cache is stored. For more information, see Process a Mining Structure.

Processing mining models: A mining model is empty, containing definitions only, until it is processed. To process a mining model, the mining structure that it is based on must have been processed. The mining model gets the data from the mining structure cache, applies any filters that may have been created on the model, and then passes the data set through the algorithm to detect patterns. After the model is processed, the model stores only the results of processing, not the data itself. For more information, see Process a Mining Model.

The following diagram illustrates the flow of data when a mining structure is processed, and when a mining model is processed.

Viewing the Results of Processing

After a mining structure has been processed, it contains a compact representation of the data for use in statistical analysis. If the cache has not been cleared, you can access the data in this cache in the following ways:

- Creating a Data Mining Extensions (DMX) query on the model and drilling through to the structure. For more information, see SELECT FROM <model>.CASES (DMX).
- Browsing a model based on the structure, and using one of the options in the user interface to drill through to structure cases. For more information, see Data Mining Model Viewers, or Drill Through to Case Data from a Mining Model.
- Creating a DMX query on the structure cases. For more information, see SELECT FROM <structure>.CASES.

After a mining model has been processed, it contains only the patterns that were derived from analysis, and mappings from the model results to the cached training data. You can browse or query the model results, called *model content*, or you can query the model and structure cases, if they have been cached.

The model content for each mining model depends on the algorithm that was used to create it. For example, if one model is a clustering model and another is a decision trees model, the model content is very different even though the models use exactly the same data. For more information, see Mining Model Content (Analysis Services - Data Mining).

Processing Requirements

Processing requirements may differ depending on whether your mining models are based solely on relational data, or on multidimensional data source.

For relational data source, processing requires only that you create training data and run mining algorithms on that data. However, mining models that are based on OLAP objects, such as dimensions and measures, require that the underlying data be in a processed state. This may requires that the multidimensional objects be processed to populate the mining model.

For more information, see Processing Requirements and Considerations (Data Mining).

See Also

Drillthrough Queries (Data Mining)
Mining Structures (Analysis Services - Data Mining)
Mining Models (Analysis Services - Data Mining)
Logical Architecture (Analysis Services - Data Mining)

Related Projects for Data Mining Solutions

11/27/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The minimum that is required for a data mining solution is the data mining project, which defines data sources, data source views, mining structures and mining models. However, when data mining models are used in daily decision making, it is important that data mining be integrated with other part of a predictive analytics solution, which can include these processes and components:

- Preparation and selection of data and of variables. Includes data cleansing, metadata management and integration of multiple data sources, and the conversion, merging, and uploading of data into a data warehouse.
- Reporting of analysis, presentation of predictions, and auditing/tracking of data mining activities.
- Using multidimensional models or tabular models to explore findings.
- Refinement of the data mining solution to support new data, or changes in the support infrastructure driven by current analysis.

This topic describes the other features of SQL Server 2017 that are often part of a predictive analytics solution, either to support the processes of data preparation and data mining, or to support users by providing tools for analysis and action.

Integration Services

Reporting Services

Data Quality Service

Full-Text Search

Semantic Indexing

SQL Server Integration Services

Integration Services provides components and features that are required for the data preparation and training phases of a data mining project. Although you can perform many data cleansing or preparation tasks by using other tools, such as scripts, Integration Services has numerous advantages for data mining:

- Represents tasks as part of a workflow, which can be repeated, automated, branched, and extended.
- Provides extensive support for auditing and multiple ways of capturing errors and logging events.

  In addition to capturing data lineage, you can monitor changes to the data throughout the data transformation pipeline.

  You can also integrate your SSIS workflows with the features that support Change Data Capture functionality in SQL Server.
- Data mining can be incorporated in the Integration Services workflow, to intelligently separate incoming data into multiple tables. For example, you could use a prediction query to split new customers into different groups for targeting in a mailing campaign.

The following lists provide links to the Integration Services components that are most widely used in support of

data mining.

Control Flow Components

- Analysis Services Execute DDL Task
- Analysis Services Processing Task
- CDC Control Task
- Data Cleansing
- Data Mining Query Task
- Data Profiling Task

Data Flow Components

- CDC Flow Components
- Conditional Split Transformation
- Data Conversion Transformation
- Data Mining Model Training Destination
- Data Mining Query Transformation
- Derived Column Transformation
- Percentage Sampling Transformation
- Term Extraction Transformation
- Term Lookup Transformation

SQL Server Reporting Services

Although Reporting Services is typically not seen as a critical component of data mining solutions, it provides the following features that are useful for presentation of data mining solutions.

- Integration of data from multiple sources in complex reports. Create queries against the model content for analysts, and reports that show predictions and trends for end users.
- The ability to create a report that lets users directly query against an existing mining model.
- Integration with Analysis Services, to support drillthrough and exploration of data mining dimensions and data mining cubes created from OLAP models.
- parameterization and formatting features that are available in Reporting Services.

For more information about how to use Reporting Services with DMX queries as a data source, see these links:

Retrieve Data from a Data Mining Model (DMX) (SSRS)

Analysis Services DMX Query Designer User Interface

Analysis Services Connection Type for DMX (SSRS)

However, it is not necessary to use DMX as the data source. The Integration Services components for data mining also support saving the results of a prediction query to a relational database. If you have an established workflow for updating models using Integration Services, persisting predictions and other data mining query results to SQL Server enable you to use Power View for reporting, as well as other tools that do not interface with DMX.

For more information about using Reporting Services as the presentation layer for data sources, see Integrating Reporting Services into Applications.

Data Quality Services

Data Quality Services (DQS) is new in SQL Server 2017. Because data problems can make data mining impossible, data miners who perform repeated analysis or who work in large organizations with complex data sources are expected to find that a well-planned data project using DQS is a more reliable solution for support of data mining than ad hoc cleansing of data using Transact-SQL or other scripts.

The following features of DQS should be considered for data preparation and data integrity in a data mining solution.

A computer-assisted data cleansing process that analyzes source data and proposes changes.
DQS can compare source data with cloud-based reference data maintained and guaranteed by data quality providers.

DQS can also analyze raw source data and create a knowledge base from user data. The processed data is categorized and then displayed to the user for further processing. The cleansing process is interactive, meaning the data steward can approve, reject, or modify the data proposed by the computer-assisted data cleansing process.

The outcome of the process is a knowledge base that you can continuously improve, or reuse in multiple data-enhancement phases.

For more information, see Data Cleansing.

A computer-assisted matching process that analyzes source data and proposes changes.
To prevent data duplication, you can perform addition cleansing of the data source, to identify exact and approximate matches. These components let you specify the matching rules, and the thresholds at which to apply them.

By finding data matches, you can remove duplicates, which can be a problem for data mining. Data de-duplication is not automatic; the data steward or IT professional must verify both the knowledge in the knowledge base and the changes to be made to the data.

After you have created the initial DQS project, you can automate many of the tasks by using Integration Services components.

For more information, see Data Matching.

While performing cleansing and matching activities in a data quality project, you can get real-time statistics and information about the data that is being processed by DQS. Data profiling helps you assess the extent to which data cleansing or matching helped improve the data quality, and understand the changes that were made. For information about data profiling and notifications, see Data Profiling and Notifications in DQS.

A knowledge base that represents three types of knowledge: out-of-the-box knowledge, knowledge generated by the DQS server, and knowledge generated by the user.
Once you have created a knowledge base, you can use it iteratively to cleanse and verify other data.

You can import new data into the knowledge base data from multiple sources, either known clean data from reference providers, or raw data that is matched to existing data in the knowledge base.

For detailed information about the cleansing activity in a data quality project, see Data Cleansing (DQS).

You can also apply the knowledge in the knowledge base to other sources, to perform data cleansing within other processes. Such data cleansing can help identify user entry errors, corruption in transmission or storage, or mismatched data dictionary definitions.

For more information, see DQS Knowledge Bases and Domains.

Full-Text Search

Full-Text Search in SQL Server lets applications and users run full-text queries against character-based data in SQL Server tables. When full-text search is enabled, you can perform searches against text data that are enhanced by language-specific rules about the multiple forms of a word or phrase. You can also configure search conditions, such as the distance between multiple terms, and use functions to constrain the results that are returned in order of likelihood.

Because full-text queries are a feature provided by the SQL Server engine, you can create parameterized queries, generate custom data sets or term vectors by using full-text search features on a text data source, and use these sources in data mining.

For more information about how full-text queries interact with the full-text index, see Query with Full-Text Search.

An advantage of using the full-text search features of SQL Server is that you can leverage the linguistic intelligence that is contained in the word breakers and stemmers shipped for all SQL Server languages. By using the supplied word breakers and stemmers, you can ensure that words are separated using the characters appropriate for each language, and that synonyms based on diacritics or orthographic variations (such as the multiple number formats in Japanese) are not overlooked.

In addition to linguistic intelligence that governs word boundaries, the stemmers for each language can reduce variants of a word to a single term, based on knowledge of the rules for conjugation and orthographic variation in that language. The rules for linguistic analysis differ for each language and are developed based on extensive research on real-life corpora.

For more information, see Configure and Manage Word Breakers and Stemmers for Search.

The version of a word that is stored after full-text indexing is a token in compressed form. Subsequent queries to the full-text index generate multiple inflectional forms of a particular word based on the rules of that language, to ensure that all probable matches are made. For example, although the token that is stored might be "run", the query engine also looks for the terms "running", "ran", and "runner," because these are regularly derived morphological variations of the root word "run".

You can also create and build a user thesaurus to store synonyms and enable better search results, or categorization of terms. By developing a thesaurus tailored to your full-text data, you can effectively broaden the scope of full-text queries on that data. For more information, see Configure and Manage Thesaurus Files for Full-Text Search.

Requirements for using full-text search include the following:

- The database administrator must create a full-text index on the table.
- Only one full-text index is allowed per table.
- Each column that you index must have a unique key.
- Full-text indexing is supported only for columns with these data types: char, varchar, nchar, nvarchar, text, ntext, image, xml, varbinary, and varbinary(max). If the column is varbinary, varbinary(max), image, or xml, you must specify the file extension of the indexable document (.doc, .pdf, .xls, and so forth), in a separate type column.

Semantic Indexing

Semantic search builds upon the existing full-text search features in SQL Server, but uses additional capabilities and statistics to enable scenarios such as automatic keyword extraction and discovery of related documents. For example, you might use semantic search to build a base taxonomy for an organization, or classify a corpus of

documents. Or, you could use the combination of extracted terms and document similarity scores in clustering or decision tree models.

After you have enabled semantic search successfully, and indexed your data columns, you can use the functions that are provided natively with semantic indexing to do the following:

- Return single-word key phrases with their score.
- Return documents that contain a specified key phrase.
- Return similarity scores, and the terms that contribute to the score.

For more information, see Find Key Phrases in Documents with Semantic Search and Find Similar and Related Documents with Semantic Search.

For more information about the database objects that support semantic indexing, see Enable Semantic Search on Tables and Columns.

Requirements for using semantic search include the following:

- Full-text search also be enabled.
- Installation of the semantic search components also creates a special system database, which cannot be renamed, altered, or replaced.
- Documents that you index using the service must be stored in SQL Server, in any of the database objects that are supported for full-text indexing, including tables and indexed views.
- Not all of the full-text languages support semantic indexing. For a list of supported languages, see sys.fulltext_semantic_languages (Transact-SQL).

See Also

Multidimensional Model Solutions
Tabular Model Solutions

Deployment of Data Mining Solutions

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The last step in the data mining process is to deploy the models to a production environment. Deployment is important because it makes the models available to users so that you can perform any of the following tasks:

- Use the models to create predictions and make business decisions. For information about the tools you can use to create queries, see Data Mining Query Tools.
- Embed data mining functionality directly into an application. You can include Analysis Management Objects (AMO) or an assembly that contains a set of objects that your application can use to create, alter, process, and delete mining structures and mining models.
- Create reports that let users request predictions, view trends, or compare models.

This section provides detailed information about deployment options.

Requirements for Deployment of Data Mining Solutions

Deploying a Relational Solution

Deploying a Multidimensional Solution

Related Resources

In This Section

Deploy a Data Mining Solution to Previous Versions of SQL Server

Export and Import Data Mining Objects

Requirements for Deployment of Data Mining Solutions

The instance of Analysis Services to which you deploy the solution must be running in a mode that supports multidimensional objects and data mining objects; that is, you cannot deploy data mining objects to an instance that hosts tabular models or Power Pivot data.

Therefore, when you create a data mining solution in Visual Studio, be sure to use the template, **Analysis Services Multidimensional and Data Mining Project**.

When you deploy the solution, the objects used for data mining are created in the specified Analysis Services instance, in a database with the same name as the solution file.

Deploying a Relational Solution

When you deploy a relational data mining solution, the required data mining objects are created within a new Analysis Services database, and the objects are processed by default. You can change processing options by using the configuration property, **Processing Option**. For more information, see Configure Analysis Services Project Properties (SSDT).

By default, only incremental changes are deployed each time. In other words, you can modify a mining model, and when you re-deploy the project, only that mining model would be updated. However, if you have multiple clients editing the Analysis Services database, this can lead to errors. To change the default deployment mode so that the entire database is refreshed when you deploy the solution, change the **Deployment Mode** property

In a relational data mining solution, the only objects that must be deployed are the data source definition, any data source views that were used, the mining structures, and all dependent mining models.

Deploying a Multidimensional Solution

When you deploy a multidimensional data mining solution, this solution creates your data mining objects within the same database as the source cube.

When you process the mining structure or mining model, you must process the source cube as well. For this reason, deploying a solution that uses OLAP mining models can take longer than relational data mining solutions.

Typically data mining objects also use the same data sources and data source views that are used for the cube. However, you can add data sources and data source views that are targeted specifically to data mining. For example, typically a cube would not contain data about prospective clients, or external data not used in the multidimensional objects.

Related Resources

Moving Data Mining Objects

If your model is based on relational data only, exporting and importing objects using DMX is the easiest way to move models.

Move an Analysis Services Database

When models use a cube as a data source, refer to this topic for more information about how to move models and their supporting cube data.

Deploy Analysis Services Projects (SSDT)

Provides general information about deployment of Analysis Services projects, and describes the properties that you can set as part of the project configuration.

See Also

Processing a multidimensional model (Analysis Services)
Data Mining Query Tools
Processing Requirements and Considerations (Data Mining)

Deploy a Data Mining Solution to Previous Versions of SQL Server

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This section describes known compatibility issues that may arise when you attempt to deploy a data mining model or data mining structure that was created in an instance of SQL Server 2017 Analysis Services (SSAS) to a database that uses SQL Server 2005 Analysis Services, or when you deploy models created in SQL Server 2005 to an instance of SQL Server 2017.

Deployment to an instance of SQL Server 2000 Analysis Services is not supported.

Deploying Time Series Models

Deploying Models with Holdout

Deploying Models with Filters

Restoring from Database Backups

Using Database Synchronization

Deploying Times Series Models

The Microsoft Time Series algorithm was enhanced in SQL Server 2008 by the addition of a second, complementary algorithm, ARIMA. For more information about the changes in the time series algorithm, see Microsoft Time Series Algorithm.

Therefore, time series mining models that use the new ARIMA algorithm may behave differently when deployed to an instance of SQL Server 2005 Analysis Services.

If you have explicitly set the parameter PREDICTION_SMOOTHING to control the mixture of ARTXP and ARIMA models in prediction, when you deploy this model to an instance of SQL Server 2005, Analysis Services will raise an error stating that the parameter is not valid. To prevent this error, you must delete the PREDICTION_SMOOTHING parameter and convert the models to a pure ARTXP model.

Conversely, if you deploy a time series model that was created using SQL Server 2005 Analysis Services to an instance of SQL Server 2017, when you open the mining model in SQL Server Data Tools (SSDT), the definition files are first converted to the new format, and two new parameters are added by default to all time series models. The parameter FORECAST_METHOD is added with the default value of MIXED, and the parameter PREDICTION_SMOOTHING is added with the default value of 0.5. However, the model will continue to use only ARTXP for forecasting until you reprocess the model. As soon as you reprocess the model, prediction changes to use both ARIMA and ARTXP.

Therefore, if you wish to avoid changing the model, you should only browse the model and never process it. Alternatively, you could explicitly set the FORECAST_METHOD or PREDICTION_SMOOTHING parameters.

For detailed information about configuring mixed models, see Microsoft Time Series Algorithm Technical Reference.

If the provider that is used for the model's data source is SQL Client Data Provider 10, you must also modify the data source definition to specify the previous version of the SQL Server Native Client. Otherwise, SQL Server Data Tools generates an error stating that the provider is not registered.

Deploying Models with Holdout

If you create a mining structure that contains a holdout partition used for testing data mining models, the mining structure can be deployed to an instance of SQL Server 2005, but the partition information will be lost.

When you open the mining structure in SQL Server 2005 Analysis Services, SQL Server Data Tools raises an error, and then regenerates the structure to remove the holdout partition.

After the structure has been rebuilt, the size of the holdout partition is no longer available in the Properties window; however, the value <ddl100_100:HoldoutMaxPercent>30</ddl100_100:HoldoutMaxPercent>) may still be present in the ASSL script file.

Deploying Models with Filters

If you apply a filter to a mining model, the model can be deployed to an instance of SQL Server 2005, but the filter will not be applied.

When you open the mining model, SQL Server Data Tools (SSDT) raises an error, and then regenerates the model to remove the filter.

Restoring from Database Backups

You cannot restore a database backup that was created in SQL Server 2017 to an instance of SQL Server 2005. If you do so, SQL Server Management Studio generates an error.

If you create a backup of a SQL Server 2005 Analysis Services database and restore this backup on an instance of SQL Server 2017, all time series models are modified as described in the previous section.

Using Database Synchronization

Database synchronization is not supported from SQL Server 2017 to SQL Server 2005.

If you attempt to synchronize a SQL Server 2017 database, the server returns an error and database synchronization fails.

See Also

Analysis Services Backward Compatibility

Export and Import Data Mining Objects

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In addition to the functionality provided in Analysis Services for backing up, restoring, and migrating solutions, SQL Server Data Mining provides the ability to quickly transfer data mining structures and models between different servers by using Data Mining Extensions (DMX).

If your data mining solution uses relational data instead of a multidimensional database, transferring models by using **EXPORT** and **IMPORT** is much faster and easier than either using database restore or deploying an entire solution.

This section provides an overview of how to transfer data mining structures and models by using DMX statements. For details of the syntax, together with examples, see EXPORT (DMX) and IMPORT (DMX).

> **NOTE**
> You must be a database or server administrator to export or import objects from a Microsoft SQL Server Analysis Services database.

Exporting Data Mining Structures

When you export a mining structure, the EXPORT statement automatically exports all associated models. If you want to control the objects that are exported, you must specify each object by name.

If the mining structure has been processed and the results have been cached, which is the default behavior, when you export the mining structure, the definition contains a summary of the data on which the structure is based. To remove this summary, you must clear the cache associated with the mining structure by performing a **Process Clear Structure** operation. For more information, see Process a Mining Structure.

Exporting Data Mining Models

You can use the **WITH DEPENDENCIES** keyword to export the data source and data source view definition together with the mining model and its structure.

When you export a mining model without exporting its dependencies, the EXPORT statement will export the definition of the mining model and its mining structure, but does not export the definition of the data sources. As a consequence, after you import the model you will be able to browse the model immediately, but if you want to reprocess the mining model on the target server, or run queries against the underlying data, you must create a corresponding data source on the destination server.

Importing Data Mining Structures and Models

When you import a data mining object, the object is imported to the server and database to which you are connected when you execute the IMPORT statement. If the import file includes a database that does not exist on the server, the database will be created.

You can also import a mining structure or mining model by using the **Restore** command. Your models or structures will be restored into the database that has the same name as the database from which they were exported. For more information, see Restore Options.

Remarks

You cannot import a model or structure to a server if a model or structure of the same name already exists on that server. Also, you cannot export a data mining object and then modify the name of that object in the export file. Therefore, if you anticipate naming conflicts, you should either delete the data mining object on the target server, or rename the data mining object before you export the definition.

See Also

Management of Data Mining Solutions and Objects

Data Mining Architecture

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This section describes the architecture of data mining solutions that are hosted in an instance of Analysis Services. The topics in this section describe the logical and physical architecture of an Analysis Services instance that supports data mining, and also provide information about the clients, providers, and protocols that can be used to communicate with data mining servers, and to work with data mining objects either locally or remotely.

In general, SQL Server Data Mining operates as a service that is provided as part of an Analysis Services instance running in multidimensional mode; therefore, we recommend that you also review the following sections of Books Online that describe the operation, maintenance, and configuration of Analysis Services multidimensional solutions.

Processing a multidimensional model (Analysis Services)

Connect to Analysis Services

Database Storage Location

Switch an Analysis Services database between ReadOnly and ReadWrite modes

For more information about how you can implement data mining in your business intelligence solution, see the Solution Guides section of the MSDN Library.

In This Section

Logical Architecture (Analysis Services - Data Mining)

Physical Architecture (Analysis Services - Data Mining)

Data Mining Services and Data Sources

Management of Data Mining Solutions and Objects

Security Overview (Data Mining)

See Also

Multidimensional Model Programming
Data Mining Programming

Logical Architecture (Analysis Services - Data Mining)

10/26/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data mining is a process that involves the interaction of multiple components.

- You access sources of data in a SQL Server database or any other data source to use for training, testing, or prediction.
- You define data mining structures and models by using SQL Server Data Tools (SSDT) or Visual Studio.
- You manage data mining objects and create predictions and queries by using SQL Server Management Studio.
- When the solution is complete, you deploy it to an instance of Analysis Services.

The process of creating these solution objects has already been described elsewhere. For more information, see Data Mining Solutions.

The following sections describe the logical architecture of the objects in a data mining solution.

Data Mining Source Data

Mining Structures

Mining Models

Custom Data Mining Objects

Data Mining Source Data

The data that you use in data mining is not stored in the data mining solution; only the bindings are stored. The data might reside in a database created in a previous version of SQL Server, a CRM system, or even a flat file. When you train the structure or model by processing, a statistical summary of the data is created and stored in a cache that can be persisted for use in later operations, or deleted after processing. For more information, see Mining Structures (Analysis Services - Data Mining).

You combine disparate data within the Analysis Services data source view (DSV) object, which provides an abstraction layer on top of your data source. You can specify joins between tables, or add tables that have a many-to-one relationship to create nested table columns. The definition of these objects, the data source and the data source view, are stored within the solution with the file name extensions, *.ds and *.dsv. For more information about creating and using Analysis Services data sources and data source views, see Supported Data Sources (SSAS - Multidimensional).

You can also define and alter data sources and data source views by using AMO or XMLA. For more information about working with these objects programmatically, see Logical Architecture Overview (Analysis Services - Multidimensional Data).

Mining Structures

A data mining structure is a logical data container that defines the data domain from which mining models are built. A single mining structure can support multiple mining models.

When you need to use the data in the data mining solution, Analysis Services reads the data from the source and

generates a cache of aggregates and other information. By default this cache is persisted so that training data can be reused to support additional models. If you need to delete the cache, change the **CacheMode** property on the mining structure object to the value, **ClearAfterProcessing**. For more information, see AMO Data Mining Classes.

Analysis Services also provides the ability to separate your data into training and testing data sets, so that you can test your mining models on a representative, randomly selected set of data. The data is not actually stored separately; rather, case data in the structure cache is marked with a property that indicates whether that particular case is used for training or for testing. If the cache is deleted, that information cannot be retrieved.

For more information, see Mining Structures (Analysis Services - Data Mining).

A data mining structure can contain nested tables. A nested table provides additional detail about the case that is modeled in the primary data table. For more information, see Nested Tables (Analysis Services - Data Mining)

Mining Models

Before processing, a data mining model is only a combination of metadata properties. These properties specify a mining structure, specify a data mining algorithm, and a define collection of parameter and filter settings that affect how the data is processed. For more information, see Mining Models (Analysis Services - Data Mining).

When you process the model, the training data that was stored in the mining structure cache is used to generate patterns, based both on statistical properties of the data and on heuristics defined by the algorithm and its parameters. This is known as *training* the model.

The result of training is a set of summary data, contained within the *model content*, which describes the patterns that were found and provides rules by which to generate predictions. For more information, see Mining Model Content (Analysis Services - Data Mining).

In limited cases, the logical structure of the model can also be exported into a file that represents model formulas and data bindings according to a standard format, the Predictive Modeling Markup Language (PMML). This logical structure can be imported into other systems that utilize PMML and the model so described can then be used for prediction. For more information, see Understanding the DMX Select Statement.

Custom Data Mining Objects

Other objects that you use in the context of a data mining project, such as accuracy charts or prediction queries, are not persisted within the solution, but can be scripted using ASSL or built using AMO.

Additionally, you can extend the services and features available on an instance of Analysis Services by adding these custom objects:

Custom assemblies
.NET assemblies can be defined by using any CLR-or COM-complaint language, and then registered with an instance of SQL Server. Assembly files are loaded from the location defined by the application, and a copy is saved in the server along with the data. The copy of the assembly file is used to load the assembly every time the service is started.

For more information, see Multidimensional Model Assemblies Management.

Custom stored procedures
Analysis Services data mining supports the use of stored procedures to work with data mining objects. You can create your own stored procedures to extend functionality and more easily work with data returned by prediction queries and content queries.

Defining Stored Procedures

The following stored procedures are supported for use in performing cross-validation.

Data Mining Stored Procedures (Analysis Services - Data Mining)

Additionally, Analysis Services contains many system stored procedures that are used internally for data mining. Although the system stored procedures are for internal use, you might find them useful shortcuts. Microsoft reserves the right to change these stored procedures as needed; therefore, for production use, we recommend that you create queries by using DMX, AMO, or XMLA.

Custom plug-in algorithms

Analysis Services provides a mechanism for creating your own algorithms, and then adding the algorithms as a new data mining service to the server instance.

Analysis Services uses COM interfaces to communicate with plugin algorithms. To learn more about how to implement new algorithms, see Plugin Algorithms.

You must register each new algorithm before you can use it. To register an algorithm, you add the required metadata for the algorithms in the .ini file of the instance of Analysis Services. You must add the information to each instance where you plan to use the new algorithm. After you have added the algorithm, you can restart the instance, and use the MINING_SERVICES schema rowset to view the new algorithm, including the options and providers that the algorithm supports.

See Also

Processing a multidimensional model (Analysis Services)
Data Mining Extensions (DMX) Reference

Physical Architecture (Analysis Services - Data Mining)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Microsoft Analysis Services uses both server and client components to supply data mining functionality for business intelligence applications:

- The server component is implemented as a Microsoft Windows service. You can have multiple instances on the same computer, with each instance of Analysis Services implemented as a separate instance of the Windows service.
- Clients communicate with Analysis Services using the public standard XML for Analysis (XMLA), a SOAP-based protocol for issuing commands and receiving responses, exposed as a Web service. Client object models are also provided over XMLA, and can be accessed either by using a managed provider, such as ADOMD.NET, or a native OLE DB provider.
- Query commands can be issued using Data Mining Extensions (DMX), an industry standard query language oriented toward data mining. Analysis Services Scripting Language (ASSL) can also be used to manage Analysis Services database objects.

Architectural Diagram

An Analysis Services instance runs as a stand-alone service and communication with the service occurs through XML for Analysis (XMLA), by using either HTTP or TCP.

AMO is a layer between the user application and the Analysis Services instance that provides access to Analysis Services administrative objects. AMO is a class library that takes commands from a client application and converts those commands into XMLA messages for the Analysis Services instance. AMO presents Analysis Services instance objects as classes to the end user application, with method members that run commands and property members that hold the data for the Analysis Services objects.

The following illustration shows the Analysis Services components architecture, including services within the Analysis Services instance and user components that interact with the instance.

The illustration shows that the only way to access the instance is by using the XML for Analysis (XMLA) Listener, either by using HTTP or TCP.

> **WARNING**
>
> DSO has been deprecated. You should not use DSO to develop solutions.

Server Configuration

One server instance can support multiple Analysis Services databases, each with its own instance of the Analysis Services service that responds to client requests and processes objects.

Separate instances must be installed if you want to work with both tabular models and data mining and/or multidimensional models. Analysis Services supports side-by-side installation of instances running in tabular mode (which uses the xVelocity in-memory analytics engine (VertiPaq) storage engine) and instances running in one of the conventional OLAP, MOLAP, or ROLAP configurations. For more information, see Determine the Server Mode of an Analysis Services Instance.

All communications between a client and the Analysis Services server use XMLA, which is a platform-independent and language-independent protocol. When a request is received from a client, Analysis Services determines whether the request relates to OLAP or data mining, and routes the request appropriately. For more information, see OLAP Engine Server Components.

See Also

Logical Architecture (Analysis Services - Data Mining)

Data Mining Services and Data Sources

10/26/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data mining requires a connection to an instance of SQL Server Analysis Services. Data from a cube is not required for data mining and the use of relational sources is recommended; however, data mining uses components provided by the Analysis Services engine.

This topic provides information that you need to know when connecting to an instance of SQL Server Analysis Services to create, process, deploy, or query data mining models.

Data Mining Services

The server component of Microsoft SQL Server Analysis Services is the msmdsrv.exe application, which ordinarily runs as a Windows service. This application consists of security components, an XML for Analysis (XMLA) listener component, a query processor component and numerous other internal components that perform the following functions:

- Parsing statements received from clients
- Managing metadata
- Handling transactions
- Processing calculations
- Storing dimension and cell data
- Creating aggregations
- Scheduling queries
- Caching objects
- Managing server resources

XMLA Listener

The XMLA listener component handles all XMLA communications between Analysis Services and its clients. The Analysis Services **Port** configuration setting in the msmdsrv.ini file can be used to specify a port on which an Analysis Services instance listens. A value of 0 in this file indicates that Analysis Services listen on the default port. Unless otherwise specified, Analysis Services uses the following default TCP ports:

PORT	DESCRIPTION
2383	Default instance of SQL Server Analysis Services.
2382	Redirector for other instances of SQL Server Analysis Services.
Dynamically assigned at server startup	Named instance of SQL Server Analysis Services.

For more information about controlling the ports used by this service, see Configure the Windows Firewall to Allow Analysis Services Access.

Connecting to Data Sources

Whenever you create or update a data mining structure or model, you use data that is defined by a data source. The data source does not contain the data, which might include Excel workbooks, text files, and SQL Server databases; it only defines the connection information. A data source view (DSV) serves as an abstraction layer on top of that source, modifying or mapping the data that is obtained from the source.

It is beyond the scope of this topic to describe the connection requirements for each of these sources. For more information, see the documentation for the provider. However, in general, you should be aware of the following requirements of Analysis Services, when interacting with providers:

- Because data mining is a service provided by a server, the Analysis Services instance must be given access to the data source. There are two aspects to access: location and identity.

  Location means that, if you build a model using data that is stored only on your computer and then deploy the model to a server, the model would fail to process because the data source cannot be found. To resolve this problem, you might need to transfer data into the same SQL Server instance where Analysis Services is running, or move files to a shared location.

  Identity means that the services on Analysis Services must be able to open the data file or data source with the appropriate credentials. For example, when you built the model, you might have had unlimited permissions to view the data, but the user who is processing and updating the models on the server might have limited or no access to the data, which can cause failure to process or affect the contents of a model. At minimum, the account used for connecting to the remote data source must have read permissions to the data.

- When you move a model, the same requirements apply: you must set up appropriate access to the location of the old data source, copy the data sources, or configure a new data source. Also, you must transfer logins and roles, or set up permissions to allow data mining objects to be processed and updated in the new location.

Configuring Permissions and Server Properties

Data mining requires additional permissions on an Analysis Services database. Most data mining properties can be set by using the Analysis Server Properties Dialog Box (Analysis Services).

For more information about the properties that you can configure, see Server Properties in Analysis Services.

The following server properties are of particular relevance to data mining:

- **AllowAdHocOpenRowsetQueries** Controls ad hoc access to OLE DB providers, which are loaded directly into server memory space.

  > **IMPORTANT**
  >
  > To improve security, we recommend that you set this property to **false**. The default value is **false**. However, even if this property is set to **false**, users can continue to create singleton queries, and can use OPENQUERY on permitted data sources.

- **AllowedProvidersInOpenRowset** Specifies the provider, if ad hoc access is enabled. You can specify multiple providers, by entering a comma-separated list of ProgIDs.

- **MaxConcurrentPredictionQueries** Controls the load on the server caused by predictions. The default value of 0 allows unlimited queries for SQL Server Enterprise, and a maximum of five concurrent queries for SQL Server Standard. Queries above the limit are serialized and may time out.

The server provides additional properties that control which data mining algorithms are available, including any

restrictions on the algorithms, and the defaults for all data mining services. However, there are no settings that allow you to control access to data mining stored procedures specifically. For more information, see Data Mining Properties.

You can also set properties that let you tune the server and control security for client usage. For more information, see Feature Properties.

> **NOTE**
> For more information about Support for plug-in algorithms by the editions of SQL Server, see Features Supported by the Editions of SQL Server 2012 (http://go.microsoft.com/fwlink/?linkid=232473).

Programmatic Access to Data Mining Objects

You can use the following object models to create a connection to an Analysis Services database and work with data mining objects:

ADO Uses OLE DB to connect to an Analysis Services server. When you use ADO, the client is limited to schema rowset queries and DMX statements.

ADO.NET Interacts with SQL Server providers better than other providers. Uses data adaptors to store dynamic rowsets. Uses the dataset object, which is a cache of the server data stored as data tables that can be updated or saved as XML.

ADOMD.NET A managed data provider that is optimized for working with data mining and OLAP. ADOMD.NET is faster and more memory-efficient than ADO.NET. ADOMD.NET also lets you retrieve metadata about server objects. Recommended for client applications except when .NET is not available.

Server ADOMD Object model for accessing Analysis Services objects directly on the server. Used by Analysis Services stored procedures; not for client use.

AMO Management interface for Analysis Services that replaces Decision Support Objects (DSO). Operations such as iterating objects require higher permissions when using AMO than when using other interfaces. That is because AMO accesses metadata directly, whereas ADOMD.NET and other interfaces access only the database schemas.

Browse and Query Access to Servers

You can perform all kinds of predictions by using a instance of Analysis Services in OLAP/Data Mining mode, with the following restrictions:

- If you use server ADOMD, you can use DMX to access the server without making a connection. You can then copy the results directly into a data table. However, you cannot use Server ADOMD with remote instances; you can query only the local server.
- ADO.NET does not support named parameters for data mining. You must use ADOMD.NET.
- ADOMD.NET lets you pass an entire table to use as the parameter; therefore, you can use data on the client, or data that is unavailable to the server. You can also use shaped tables as prediction input.

Using Data Mining Stored Procedures

A common use of stored procedures is to encapsulate queries for reuse. The client can use CALL to run stored procedures, including Analysis Services system stored procedures.

If the procedure returns a dataset, the client will receive a dataset or datatable with a nested table containing the rows. For example, if you create a query against the model content, the query returns the entire model. To avoid bringing back too many rows, you can write stored procedures by using the ADOMD+ object model.

To write a server stored procedure, you must reference the Microsoft.AnalysisServices.AdomdServer namespace.

For more information about how to create and use stored procedures, see User Defined Functions and Stored Procedures.

> **NOTE**
> Stored procedures cannot be used to change security on data server objects. When you execute a stored procedure, the user's current context is used to determine access to all server objects. Therefore, users must have appropriate permissions on any database objects that they access.

See Also

Physical Architecture (Analysis Services - Multidimensional Data)
Physical Architecture (Analysis Services - Data Mining)
Management of Data Mining Solutions and Objects

Management of Data Mining Solutions and Objects

10/26/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

SQL Server 2017 provides client tools that you can use to manage existing mining structures and mining models. This section describes the management operations that you can perform using each environment.

In addition to these tools, you can manage data mining objects programmatically, by using AMO, or use other clients that connect to an Analysis Services database, such as the Data Mining Add-ins for Microsoft Excel 2007.

In this Section

Moving Data Mining Objects

Processing Requirements and Considerations (Data Mining)

Using SQL Server Profiler to Monitor Data Mining (Analysis Services - Data Mining)

Location of Data Mining Objects

Mining structures and models that have been processed are stored in an instance of Analysis Services.

If you create a connection to an Analysis Services database in **Immediate** mode when developing your data mining objects, any objects that you create are immediately added to the server as you work. However, if you design data mining objects in **Offline** mode, which is the default when you work in SQL Server Data Tools (SSDT), the mining objects that you create are only metadata containers until you deploy them to an instance of Analysis Services. Therefore, any time that you make a change to an object, you must redeploy the object to the Analysis Services server. For more information about data mining architecture, see Physical Architecture (Analysis Services - Data Mining).

> **NOTE**
>
> Some clients, such as the Data Mining Add-ins for Microsoft Excel 2007, also let you create session mining models and mining structures, which use a connection to an instance but store the mining structure and models on the server only for the duration of the session. You can still manage these models through the client, the same as you would structures and models stored in an Analysis Services database, but the objects are not persisted after you disconnect from the instance of Analysis Services.

Managing Data Mining Objects in SQL Server Data Tools

SQL Server Data Tools (SSDT) offers features that make it easy to create, browse, and edit data mining objects.

The following links provide information on how you can modify data mining objects by using SQL Server Data Tools (SSDT):

- Edit the Data Source View used for a Mining Structure
- Change the Properties of a Mining Structure
- Change the Properties of a Mining Model
- View or Change Modeling Flags (Data Mining)

- View or Change Algorithm Parameters

Typically you will use SQL Server Data Tools (SSDT) as a tool for developing new projects and adding to existing projects, and then manage projects and objects that have been deployed by using tools such as SQL Server Management Studio.

However, you can directly modify objects that are already deployed to an instance of ssASnoversion by using the **Immediate** option and connecting to the server in Online mode. For more information, see Connect in Online Mode to an Analysis Services Database.

> **WARNING**
> All changes to a mining structure or mining model, including changes to metadata such as a name or description, require that the structure or model be reprocessed.

If you do not have the solution file that was used to create the data mining project or objects, you can import the existing project from the server using the Analysis Services Import wizard, make modifications to the objects, and then redeploy using the **Incremental** option. For more information, see Import a Data Mining Project using the Analysis Services Import Wizard.

Managing Data Mining Objects in SQL Server Management Studio

In SQL Server Management Studio, you can script, process, or delete mining structures and mining models. You can view only a limited set of properties by using Object Explorer; however, you can view additional metadata about mining models by opening a **DMX Query** window and selecting a mining structure.

- Create a DMX Query in SQL Server Management Studio

Managing Data Mining Objects Programmatically

You can create, alter, process, and delete data mining objects by using the following programming languages. Each language is designed for different tasks and as a result, there might be restrictions on the type of operations that you can perform. For example, some properties of data mining objects cannot be changed by using Data Mining Extensions (DMX); you must use XMLA or AMO.

Analysis Management Objects (AMO)

Analysis Management Objects (AMO) is an object model built on top of XMLA that gives you full control over data mining objects. By using AMO, you can create, deploy, and monitor mining structures and mining models

- AMO Concepts and Object Model
- Microsoft.AnalysisServices

Restrictions: None.

Data Mining Extensions (DMX)

Data Mining Extensions (DMX) can be used with other command interfaces such as ADO.NET or ADOMD.Net to create, delete, and query mining structures and mining models.

- Data Mining Extensions (DMX) Data Definition Statements

Restrictions: Some properties cannot be changed by using DMX.

XML for Analysis (XMLA)

XML for Analysis (XMLA) is the data definition language for all of Analysis Services. XMLA gives you control over most data mining objects and server operations. All management operations between the client and the

server can be performed by using XMLA. For convenience, you can use the Analysis Services Scripting Language (ASSL) to wrap the XML.

Restrictions: SQL Server Data Tools (SSDT) generates some XMLA statements that are supported for internal use only, and cannot be used in XML DDL scripts.

See Also

Analysis Services Developer Documentation

Moving Data Mining Objects

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The most common scenarios for moving data mining objects are to deploy a model from a testing or analysis environment to a production environment, or to share models with other users.

This topic describes how you can use the tools and scripting languages provided by Analysis Services, for moving data mining objects.

Moving Data Mining Objects between Databases or Servers

You can move data mining objects between Analysis Services databases or between instances of Analysis Services in the following ways:

- Re-deploying the solution to a different database.
- Scripting individual objects.
- Backing up and then restoring a copy of the database.
- Exporting and importing structures and models.

The following section explains these options in more detail.

Deploying

Deploying the solution to a different server or database requires that you have the solution file that was created by using SQL Server Data Tools (SSDT).

For more information about deploying Analysis Services solutions, see Deploy Analysis Services Projects (SSDT).

Scripting

Analysis Services provides several languages that you can use to script objects.

- **XMLA**: You can script objects using XMLA by right-clicking objects in SQL Server Management Studio. To execute the script, open it in an **XMLA Query** window on the target server.
- **DMX**: You can create scripts by using templates or one of the query builders provided in SQL Server Data Tools and SQL Server Management Studio.

Note, however, that there are differences in the tasks that you can perform with each scripting language:

- Properties such as the object description and data bindings can only by created or changed by using Analysis Services DDL languages, not by using DMX.
- Only DMX supports the import and export of mining objects.
- Only DMX supports generating PMML or importing model definitions from PMML.
- Only DMX supports training a model with application data. Moreover, the DMX INSERT INTO statement supports training a model without providing values for a key column.

For more information, see Developing with Analysis Services Scripting Language (ASSL).

Backup and Restore

Backup and restore of an entire Analysis Services database is the method of choice if your data mining solution relies on OLAP objects. SQL Server 2017 provides backup and restore functionality that makes database backups faster and easier.

For more information about backup, see Backup and Restore of Analysis Services Databases.

Exporting and Importing

Exporting and then re-importing mining models and structures by using DMX statements is the easiest way to move or back up individual relational data mining objects. For more information about the DMX syntax for these operations, see the following topics:

- EXPORT (DMX)
- IMPORT (DMX)

If you specify the INCLUDE DEPENDENCIES option, Analysis Services will also export the definition of any required data source views, and when you import the model or structure, it will re-create the data source view on the target server. After you have finished importing the model, make sure to set the necessary mining permissions on the object.

> **NOTE**
> You cannot export and import OLAP models by using DMX. If your mining model is based on an OLAP cube, you must use the functionality provided by Analysis Services for backing up and restoring an entire database, or redeploy the cube and its models.

See Also

Management of Data Mining Solutions and Objects

Processing Requirements and Considerations (Data Mining)

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes some technical considerations to keep in mind when processing data mining objects. For a general explanation of what processing is, and how it applies to data mining, see Processing Data Mining Objects.

Queries on Relational Store

Processing Mining Structures

Processing Mining Models

Queries on the Relational Store during Processing

For data mining, there are three phases to processing: querying the source data, determining raw statistics, and using the model definition and algorithm to train the mining model.

The Analysis Services server issues queries to the database that provides the raw data. This database might be an instance of SQL Server 2017 or an earlier version of the SQL Server database engine. When you process a data mining structure, the data in the source is transferred to the mining structure and persisted on disk in a new, compressed format. Not every column in the data source is processed: only the columns that are included in the mining structure, as defined by the bindings.

Using this data, Analysis Services builds an index of all data and discretized columns, and creates a separate index for continuous columns. One query is issued for each nested table to create the index, and an additional query per nested table is generated to process relationships between each pair of a nested table and case table. The reason for creating multiple queries is to process a special internal multidimensional data store. You can limit the number of queries that Analysis Services sends to the relational store by setting the server property, **DatabaseConnectionPoolMax**. For more information, see OLAP Properties.

When you process the model, the model does not reread the data from the data source, but instead gets the summary of the data from the mining structure. Using the cube that was created, together with the cached index and case data has been cached, the server creates independent threads to train the models.

For more information about the editions of SQL Server that support Parallel Model Processing, see Features Supported by the Editions of SQL Server 2012 (http://go.microsoft.com/fwlink/?linkid=232473).

Processing Mining Structures

A mining structure can be processed together with all dependent models, or separately. Processing a mining structure separately from models can be useful when some models are expected to take a long time to process and you want to defer that operation.

For more information, see Process a Mining Structure.

If you are concerned about conserving hard disk space, note that Analysis Services retains mining structure caches locally. That is, it writes out all the training data to your local hard disk. If you do not want the data cached, you can change the default by setting the MiningStructureCacheMode property on the mining structure

to **ClearAfterProcessing**. This will destroy the cache after models are processed; however, it will also disable drillthrough on the mining structure. For more information, see Drillthrough Queries (Data Mining).

Also, if you clear the cache, you will not be able to use the holdout test set, if you defined one, and the definition of the test set partition will be lost. For more information about holdout test sets, see Training and Testing Data Sets.

Processing Mining Models

You can process a mining model separately from its associated mining structure, or you can process all models that are based on the structure, together with the structure.

For more information, see Process a Mining Model.

However, in SQL Server Data Tools (SSDT) and SQL Server Management Studio, you cannot multiselect mining models to process with the structure. If you need to control which models are processed, you must select them individually, or use XMLA or DMX to process models serially.

When Reprocessing is Required

You must process the Analysis Services models that you define before you can start to work with them. You must also reprocess the mining models whenever you change the mining model structure, update the training data, change an existing mining model, or add a new mining model to the structure.

Mining models are also processed in these scenarios:

Deployment of a project: Depending on the project settings and the current state of the project, the mining models in the project are typically processed in full when the project is deployed.

When you initiate deployment, processing starts automatically, unless there is a previously processed version on the Analysis Services server and there have been no structural changes. You can deploy a project by selecting **Deploy solution** from the drop-down list or by pressing the F5 key. You can

For more information about how to set Analysis Services deployment properties that control how mining models are deployed, see Deployment of Data Mining Solutions.

Moving a mining model: When you move a mining model by using the EXPORT command, only the definition of the model is exported, which includes the name of the mining structure that is expected to provide data to the model.

Reprocessing requirements for the following scenarios using the EXPORT and IMPORT commands:

- The mining structure exists on the target instance and the mining structure is in an unprocessed state.

  Both the structure and model must be reprocessed.

- The mining structure exists on the target instance and the mining structure has been processed. Only the mining model was exported.

  The model can be used without processing.

- The mining structure definition was also exported by using the WITH DEENDENCIES keyword.

  Both the structure and model must be reprocessed.

For more information, see Export and Import Data Mining Objects.

See Also

Mining Structures (Analysis Services - Data Mining)

Mining Structures (Analysis Services - Data Mining)
Processing a multidimensional model (Analysis Services)

Using SQL Server Profiler to Monitor Data Mining (Analysis Services - Data Mining)

11/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If you have the necessary permissions, you can use SQL Server Profiler to monitor data mining activities that are issued as requests sent to an instance of SQL Server Analysis Services. Data mining activity can include the processing of models or structures, prediction queries or content queries, or the creation of new models or structures.

SQL Server Profiler uses a **trace** to monitor requests sent from multiple clients, including SQL Server Data Tools (SSDT), SQL Server Management Studio, Web services, or the Data Mining Add-ins for Excel, so long as the activities all use the same instance of SQL Server Analysis Services. You must create a separate trace for each instance of SQL Server Analysis Services that you want to monitor. For general information about traces, and how to use SQL Server Profiler, see Use SQL Server Profiler to Monitor Analysis Services.

For specific guidance about the types of events to capture, see Create Profiler Traces for Replay (Analysis Services).

Using Traces to Monitor Data Mining

When you capture information in a trace, you can specify whether the information is saved in a file or in a table on an instance of SQL Server. Regardless of the method you use to store the data, you can use SQL Server Profiler to view the trace and filter by events. The following table lists some of the events and subclasses in the default Analysis Services trace that are of interest for data mining.

EVENTCLASS	EVENTSUBCLASS	DESCRIPTION
Query Begin **Query End**	**0 - MDXQuery**	Contains the text of all calls to Analysis Services stored procedures.
Query Begin **Query End**	**1 - DMXQuery**	Contains the text and results of Data Mining Extensions (DMX) statements.
Progress Report Begin **Progress Report End**	**34 - DataMiningProgress**	Provides information about the progress of the data mining algorithm: for example, if you are building a clustering model, the progress message tells you which candidate cluster is being built
Query Begin **Query End**	EXECUTESQL	Contains the text of the Transact-SQL query that is being executed
Query Begin **Query End**	**2- SQLQuery**	Contains the text of any queries against the schema rowsets in the form of system tables.

EVENTCLASS	EVENTSUBCLASS	DESCRIPTION
DISCOVER Begin **DISCOVER End**	Multiple	Contains the text of DMX function calls or DISCOVER statements, encapsulated in XMLA.
Error	(none)	Contains the text of errors sent by the server to the client. Error messages prefaced with **Error (Data Mining):** or **Informational (Data Mining):** are generated specifically in response to DMX requests. However, it not sufficient to view only these error messages. Other errors, such as those generated by the parser, may be related to data mining but do not have this prefix.

By viewing the command statements in the trace log, you can also see the syntax of complex statements sent by the client to the Analysis Services server, including calls to system stored procedures. This information can be useful for debugging, or you can use valid statements as a template for creating new prediction queries or models. For some examples of stored procedure calls that you can capture via a trace, see Clustering Model Query Examples.

See Also

Monitor Analysis Services with SQL Server Extended Events

Security Overview (Data Mining)

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The process of securing Microsoft SQL Server Analysis Services occurs at multiple levels. You must secure each instance of Analysis Services and its data sources to make sure that only authorized users have read or read/write permissions to selected dimensions, mining models, and data sources. You must also secure underlying data sources to prevent unauthorized users from maliciously compromising sensitive business information. The process of securing an instance of Analysis Services is described in the following topics.

Security Architecture

See the following resources to learn about the basic security architecture of an instance of Analysis Services, including how Analysis Services uses Microsoft Windows Authentication to authenticate user access.

- Security Roles (Analysis Services - Multidimensional Data)
- Security Properties
- Configure Service Accounts (Analysis Services)
- Authorizing access to objects and operations (Analysis Services)

Configuring the Logon Account for Analysis Services

You must select an appropriate logon account for Analysis Services and specify the permissions for this account. You must make sure that the Analysis Services logon account has only those permissions that are necessary to perform necessary tasks, including appropriate permissions to the underlying data sources.

For data mining, you need a different set of permissions to build and process models than you need to view or query the models. Making predictions against a model is a kind of query and does not require administrative permissions.

Securing an Analysis Services Instance

Next you must secure the Analysis Services computer, the Windows operating system on the Analysis Services computer, Analysis Services itself, and the data sources that Analysis Services uses.

Configuring Access to Analysis Services

When you set up and define authorized users for an instance of Analysis Services, you need to determine which users should also have permission to administer specific database objects, which users can view the definition of objects or browse the models, and which users are able to access data sources directly.

Special Considerations for Data Mining

To enable an analyst or developer to create and test data mining models, you must give that analyst or developer administrative permissions on the database where the mining models are stored. As a consequence, the data mining analyst or developer can potentially create or delete other objects that are not related to data mining, including data mining objects that were created and are being used by other analysts or developers, or OLAP objects that are not included in the data mining solution.

Accordingly, when you create a solution for data mining, you must balance the needs of the analyst or developer to develop, test and tune models, against the needs of other users, and take measures to protect existing database objects. One possible approach is to create a separate database dedicated to data mining, or to create separate databases for each analyst.

Although the creation of models requires the highest level of permissions, you can control the user's access to data mining models for other operations, such as processing, browsing, or querying, by using role-based security. When you create a role, you set permissions that are specific to data mining objects. Any user who is a member of a role automatically has all permissions associated with that role.

Additionally, data mining models often reference data sources that contain sensitive information. If the mining structure and mining model has been configured to allow users to drill through from the model to the data in the structure, you must take precautions to mask sensitive information, or to limit the users who have access to the underlying data.

If you use Integration Services packages to clean data, to update mining models, or to make predictions, you must ensure that the Integration Services service has the appropriate permissions on the database where the model is stored, and appropriate permissions on the source data.

See Also

Roles and Permissions (Analysis Services)

Data Mining Tools

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Microsoft SQL Server Analysis Services provides the following tools that you can use to create data mining solutions:

- The **Data Mining Wizard** in SQL Server Data Tools (SSDT) makes it easy to create mining structures and mining models, using either relational data sources or multidimensional data in cubes.

  In the wizard, you choose data to use, and then apply specific data mining techniques, such as clustering, neural networks, or time series modeling.

- **Model viewers** are provided in both SQL Server Management Studio and SQL Server Data Tools (SSDT), for exploring your mining models after they are created. You can browse models using viewers tailored to each algorithm, or go deeper into analysis by using the model content viewer.

- The **Prediction Query Builder** is provided in both SQL Server Management Studio and SQL Server Data Tools (SSDT) to help you create prediction queries. You can also test the accuracy of models against a holdout data set or external data, or use cross-validation to assess the quality of your data set.

- SQL Server Management Studio is the interface where you manage existing data mining solutions that have been deployed to an instance of Analysis Services. You can reprocess structures and models to update the data in them.

- SQL Server Integration Services contains tools that you can use to clean data, to automate tasks such as creating predictions and updating models, and to create text mining solutions.

The following sections provide more information about the data mining tools in SQL Server.

Data Mining Wizard

Use the Data Mining Wizard to get started creating data mining solutions. The wizard is quick and easy, and guides you through the process of creating a data mining structure and an initial related mining model, and includes the tasks of selecting an algorithm type and a data source, and defining the case data used for analysis.

For More Information: Data Mining Wizard (Analysis Services - Data Mining)

Data Mining Designer

After you have created a mining structure and mining model by using the Data Mining Wizard, you can use the Data Mining Designer from either SQL Server Data Tools (SSDT) or SQL Server Management Studio to work with existing models and structures.

The designer includes tools for these tasks:

- Modify the properties of mining structures, add columns and create column aliases, change the binning method or expected distribution of values.

- Add new models to an existing structure; copy models, change model properties or metadata, or define filters on a mining model.

- Browse the patterns and rules within the model; explore associations or decision trees. Get detailed

statistics about

Custom viewers are provided for each different time of model, to help you analyze your data and explore the patterns revealed by data mining.

- Validate models by creating lift charts, or analyzing the profit curve for models. Compare models using classification matrices, or validate a data set and its models by using cross-validation.
- Create predictions and content queries against existing mining models. Build one-off queries, or set up queries to generate predictions for entire tables of external data.

SQL Server Management Studio

After you create and deploy mining models to a server, you can use SQL Server Management Studio to manage the Analysis Services database that hosts the data mining objects. You can also continue to perform tasks that use the model, such as exploring the models, processing new data, and creating predictions.

Management Studio also contains query editors that you can use to design and execute Data Mining Extensions (DMX) queries, or ot work with data mining objects by using XMLA.

Integration Services Data Mining Tasks and Transformations

SQL Server Integration Services provides many components that support data mining.

Some tools in Integration Services are designed to help automate common data mining tasks, including prediction, model building, and processing. For example:

- Create an Integration Services package that automatically updates the model every time the dataset is updated with new customers
- Perform custom segmentation or custom sampling of case records.
- Automatically generate models passed on parameters.

However, you can also use data mining in a package workflow, as an input to other processes. For example:

- Use probability values generated by the model to weight scores for text mining or other classification tasks.
- Automatically generate predictions based on prior data and use those values to assess the validity of new data.
- Using logistic regression to segment incoming customers by risk.

For More Information: Related Projects for Data Mining Solutions

See Also

Data Mining Extensions (DMX) Reference
Mining Model Tasks and How-tos
Mining Model Viewer Tasks and How-tos
Data Mining Solutions

Data Mining Wizard (Analysis Services - Data Mining)

11/28/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Data Mining Wizard in Microsoft SQL Server Analysis Services starts every time that you add a new mining structure to a data mining project. The wizard helps you choose a data source and set up a data source view that defines the data to be used for analysis, and then helps you create an initial model.

In the final phase of the wizard, you can optionally divide your data into training and testing sets, and enable features such as drillthrough.

What to Know Before You Start

Here are the things you need to know before you start the wizard.

- Will you build the data mining structure and models from a relational database or from an existing cube in an OLAP database?
- Which columns contain the keys that uniquely identify a case record?
- Which columns or attributes do you want to use for prediction? Which columns or attributes are good to use as input for analysis?
- Which algorithm should you use? The algorithms provided in SQL Server Analysis Services all have different characteristics and produce different results. Fortunately you are not limited to one model for each set of data, so feel free to experiment by adding different models.
- Do you need to be able to test your models on a unified data set? If so, consider using the option to set some data aside for testing. You can choose a percentage, and cap that by a specified number of rows if desired.

Starting the Data Mining Wizard

To use the Data Mining Wizard, you must have opened a solution in SQL Server Data Tools (SSDT) that contains at least one data mining or OLAP project.

- If your solution is ready for data mining, you can simply right-click the **Mining Structures** node in Solution Explorer and select **New Mining Structure** to start the wizard.
- If your solution does not contain any existing projects, you can add a new data mining project. From the **File** menu, select **New**, and then select **Project**. Be sure to choose the template, **Analysis Services Multidimensional and Data Mining Project**.
- You can also use the Analysis Services Import Wizard to obtain metadata from an existing data mining solution. However, you cannot select the individual objects to import; the entire database is imported, including any cubes, data source views, etc. Also note that the new solution that is created via import is automatically configured to use the local default database. You might need to change this to another instance before you can process or browse the objects, and if you are importing from a previous version of Analysis Services, you might need to update references to providers.

Next, you will create the mining structure and one associated data mining model. You can also create just the

mining structure and add models later, but it is generally easiest to create a test model first.

Relational vs. OLAP Mining Models

The next important option that you have is whether to use a relational data source, or to base your model on multidimensional (OLAP) data.

The Data Mining Wizard branches into two paths at this point, depending on whether your data source is relational or in a cube. Everything else except the data selection process is the same-the choice of algorithm, the ability to add a holdout data set, etc.-but selecting cube data is a bit more complex than using relational data. (You also get some additional options at the end if you create a model based on a cube.)

See the following topics for a walkthrough of each option in more detail:

Create a Relational Mining Structure
Walks you through the decisions you make when building a relational data mining model.

Create an OLAP Mining Structure
Describes the additional options and selections to make when choosing data from an OLAP cube.

> **NOTE**
> You do not need to have a cube or an OLAP database to do data mining. Unless your data is already stored in a cube, or you want to mine OLAP dimensions or the results of OLAP aggregations or calculations, we recommend that you use a relational table or data source for data mining.

Choosing an Algorithm

Next, you must decide on which algorithm to use in processing your data. This decision can be difficult to make. Each algorithm provided in Analysis Services has different features and produces different results, so you can experiment and try several different models before determining which is most appropriate for your data and your business problem. See the following topic for an explanation of the tasks to which each algorithm is best suited:

Data Mining Algorithms (Analysis Services - Data Mining)

Again, you can create multiple models using different algorithms, or change parameters for the algorithms to create different models. You are not locked into your choice of algorithm, and it is good practice to create several different models on the same data.

Define the Data Used for Modeling

In addition to choosing the data from a source, you must specify which of the table in the data source view contains the *case data*. The case table will be used to train the data mining model, and as such should contain the entities that you want to analyze: for example, customers and their demographic information. Each case must be unique, and must be identifiable by a *case key*.

In addition to specifying the case table, you can include *nested tables* in your data. A nested table usually contains additional information about the entities in the case table, such as transactions conducted by the customer, or attributes that have a many-to-one relationship with the entity. For example, a nested table joined to the **Customers** case table might include a list of products purchased by each customer. In a model that analyzes traffic to a Web site, the nested table might include the sequences of pages that the user visited. For more information, see Nested Tables (Analysis Services - Data Mining)

Additional Features

To assist you in choosing the right data, and configuring the data sources correctly, the Data Mining Wizard provides these additional features:

- **Auto -detection of data types**: The wizard will examine the uniqueness and distribution of column values and then recommend the best data type, and suggest a usage type for the data. You can override these

suggestions by selecting values from a list.

- **Suggestions for variables**: You can click on a dialog box and start an analyzer that calculates correlations across the columns included in the model, and determines whether any columns are likely predictors of the outcome attribute, given the configuration of the model so far. You can override these suggestions by typing different values.
- **Feature selection**: Most algorithms will automatically detect columns that are good predictors and use those preferentially. In columns that contain too many values, *feature selection* will be applied, to reduce the cardinality of the data and improve the chances for finding a meaningful pattern. You can affect feature selection behavior by using model parameters.
- **Automatic cube slicing**: If your mining model is based on an OLAP data source, the ability to slice the model by using cube attributes is automatically provided. This is handy for crating models based on subsets of cube data.

Completing the Wizard

The last step in the wizard is to name the mining structure and the associated mining model. Depending on the type of model you created, you might also have the following important options:

- If you select **Allow drill through**, the ability to *drill through* is enabled in the model. With drillthrough, users who have the appropriate permissions can explore the source data that is used to build the model.
- If you are building an OLAP model, you can select the options, **Create a new data mining cube**, **or Create a data mining dimension**. Both these options make it easier to browse the completed model and drill through to the underlying data.

After you complete the Data Mining Wizard, you use Data Mining Designer to modify the mining structure and models, to view the accuracy of the model, view characteristics of the structure and models, or make predictions by using the models.

Back to Top

Related Content

To learn more about the decisions you need to make when creating a data mining model, see the following links:

Data Mining Algorithms (Analysis Services - Data Mining)

Content Types (Data Mining)

Data Types (Data Mining)

Feature Selection (Data Mining)

Missing Values (Analysis Services - Data Mining)

Drillthrough on Mining Models

See Also

Data Mining Tools
Data Mining Solutions

Create a Relational Mining Structure

11/28/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Most data mining models are based on relational data sources. The advantages of creating a relational data mining model are that you can assemble ad hoc data and train and update a model without the complexity of creating a cube.

A relational mining structure can draw data from disparate sources. The raw data can be stored in tables, files, or relational database systems, so long as the data can be defined as part of data source view. For example, you should use a relational mining structure if your data is in Excel, a SQL Server data warehouse or SQL Server reporting database, or in external sources that are accessed via the OLE DB or ODBC providers.

This topic provides an overview of how to use the Data Mining Wizard to create a relational mining structure.

Requirements

Process for Creating a Relational Mining Structure

How to Choose Data Sources

How to Specify Content Type and Data Type

Why and How to Create a Holdout Data Set

Why and How to Enable Drillthrough

Requirements

First, you must have an existing data source. You can use the Data Source designer to set up a data source, if one does not already exist. For more information, see Create a Data Source (SSAS Multidimensional).

Next, use the Data Source View Wizard to assemble required data into a single data source view. For more information about how you can select, transform, filter, or manage data with data source views, see Data Source Views in Multidimensional Models.

Overview of Process

Start the Data Mining Wizard, by right-clicking the **Mining Structures** node in Solution Explorer, and selecting **Add New Mining Structure**. The wizard guides you through the following steps to create the structure for a new relational mining model:

1. **Select the Definition Method**: Here you select a data source type, and choose **From relational database or data warehouse**.

2. **Create the Data Mining Structure**: Determine whether you will build just a structure, or a structure with a mining model.

   You also choose an appropriate algorithm for your initial model. For guidance on which algorithm is best for certain tasks, see Data Mining Algorithms (Analysis Services - Data Mining).

3. **Select Data Source View**: Choose a data sources view to use in training your model. The data source view can also contain data used for testing, or unrelated data. You get to pick and choose which data is actually used in the structure and in the model. You can also apply filters to the data later on.

4. **Specify Table Types**: Select the table that contains the cases used for analysis. For some data sets, especially ones used for building market basket models, you might also include a related table, to use as a nested table.

   For each table, you must specify the key, so that the algorithm knows how to identify a unique record, and related records if you have added a nested table.

   For more information, see Mining Structure Columns.

5. **Specify the Training Data**: On this page, you choose as the *case table*, which is the table that contains the most important data for analysis.

   For some data sets, especially ones used for building market basket models, you might also include a related table. The values in that nested table will be handled as multiple values that are all related to a single row (or case) in the main table.

6. **Specify Columns Content and Data Types**: For each column that you use in the structure, you must choose both a *data type* and a *content type*.

   The wizard will automatically detect possible data types, but you don't need to use the data type recommended by the wizard. For example, even if your data contains numbers, they might be representative of categorical data. Columns that you specify as keys are automatically assigned the correct data type for that particular model type. For more information, see Mining Model Columns and Data Types (Data Mining).

   The *content type* that you choose for each column that you use in the model tells the algorithm how the data should be processed.

   For example, you might decide to discretize numbers, rather than use continuous values. You can also ask the algorithm to automatically detect the best content type for the column. For more information, see Content Types (Data Mining).

7. **Create Testing Set**: On this page, you can tell the wizard how much data should be set aside for use in testing the model. If your data will support multiple models, it is a good idea to create a holdout data set, so that all models can be tested on the same data.

   For more information, see Testing and Validation (Data Mining).

8. **Completing the Wizard**: On this page, you give a name to the new mining structure and the associated mining model, and save the structure and model.

   You also can set some important options, depending on the model type. For example, you can enable drillthrough on the structure.

   At this point the mining structure and its model are just metadata; you will need to process them both to get results.

How to Choose Relational Data

Relational mining structures can be based on any data that is available through an OLE DB data source. If the source data is contained within multiple tables, you use a data source view to assemble the tables and columns that you need in one place.

If the tables include any one-to-many relationships-for example, you have multiple purchase records for each customer that you want to analyze-you can add both tables, and then use one table as the case table, linking data on the many side of the relationship as a nested table.

The data in a mining structure is derived from whatever is in the existing data source view. You can modify data as you need within the data source view, adding relationships or derived columns that might not be present in the

underlying relational data. You can also create named calculations or aggregations within the data source view. These features are very handy if you do not have control over the arrangement of data in the data source, or if you want to experiment with different aggregations of data for your data mining models.

You do not have to use all the data that is available; you can pick and choose which columns to include in the mining structure. All models that are based on that structure then can use those columns, or you can flag certain columns as **Ignore** for a particular model. You can enable users of a data mining model to drill down from the results of the mining model to see additional mining structure columns that were not included in the mining model itself.

How to Specify Content Type and Data Type

The data type is pretty much the same as the data types you specify in SQL Server or other application interfaces: dates and times, numbers of different sizes, Boolean values, text and other discrete data.

However, content types are important for data mining and affect the outcome of analysis. The content type tells the algorithm what it should do with the data: should numbers be treated on a continuous scale, or binned? How many potential values are there? Is each value distinct? If the value is a key, what kind of key is it - does it indicate a date/time value, a sequence, or some other kind of key?

Note that the choice of data type can limit your choice of content types. For example, you cannot discretize values that are not numeric. If you cannot see the content type that you want, you can click **Back** to return to the data type page and try a different data type.

You need not worry too much about getting the content type wrong. It is very easy to create a new model and change the content type within the model, as long as the new content type is supported by the data type set in the mining structure. It is also very common to create multiple models using different content types, either as an experiment, or to meet the requirements of a different algorithm.

For example, if your data contains an income column, you could create two different models when using the Microsoft Decision Trees algorithm, and configure the column alternately as either continuous numbers or discrete ranges. However, if you added a model using the Microsoft Naïve Bayes algorithm, you would be forced to change the column to discretized values only, because that algorithm does not support continuous numbers.

Why and How to Split Data into Training and Testing Sets

Near the end of the wizard, you must decide whether to partition your data into training and testing sets. The ability to provision a randomly sampled portion of the data for testing is very convenient, as it ensures that a consistent set of test data is available for use with all mining models associated with the new mining structure.

> **WARNING**
>
> Note that this option is not available for all model types. For example, if you create a forecasting model, you won't be able to use holdout, because the time series algorithm requires that there be no gaps in data. For a list of the model types that support holdout data sets, see Training and Testing Data Sets.

To create this holdout data set, you specify the percentage of the data you want to use for testing. All remaining data will be used for training. Optionally, you can set a maximum number of cases to use for testing, or set a seed value to use in starting the random selection process.

The definition of the holdout test set is stored with the mining structure, so that whenever you create a new model based on the structure, the testing data set will be available for assessing the accuracy of the model. If you delete the cache of the mining structure, the information about which cases were used for training and which were used for testing will be deleted as well.

Why and How to Enable Drillthrough

Almost at the very end of the wizard, you have the option to enable *drillthrough*. It is easy to miss this option, but it's an important one. Drillthrough lets you view source data in the mining structure by querying the mining model.

Why is this useful? Suppose you are viewing the results of a clustering model, and want to see the customers who were put into a specific cluster. By using drillthrough, you can view details such as contact information.

> **WARNING**
>
> To use drillthrough, you must enable it when you create the mining structure. You can enable drillthrough on models later, by setting a property on the model, but mining structures require that this option be set at the beginning. For more information, see Drillthrough Queries (Data Mining).

See Also

Data Mining Designer
Data Mining Wizard (Analysis Services - Data Mining)
Mining Model Properties
Properties for Mining Structure and Structure Columns
Mining Structure Tasks and How-tos

Create an OLAP Mining Structure

11/27/2018 • 11 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

There are many advantages to creating a data mining model based on an OLAP cube or other multidimensional data store. An OLAP solution already contains huge amounts of data that is well organized, cleaned and properly formatted; however, the complexity of the data is such that users are unlikely to find meaningful patterns by ad hoc exploration. Data mining offers the ability to discover new correlations and provide actionable insight.

This topic describes how to create an OLAP mining structure, based on a dimension and related measures in an existing multidimensional solution.

Requirements

Overview of OLAP Data Mining Process

Scenarios for Using Data Mining in OLAP Solutions

Filters

Using Nested Tables

Data Mining Dimensions

Requirements for OLAP Mining Structure and Models

If you are designing an OLAP mining model, your data source already exists, in the database that was used to build the cube. You cannot connect to a remote cube and build data mining objects; the cube objects must be available within the same solution as the database as the mining structure that you will build.

If you do not have the original project files, or do not wish to alter them, you can use the option in Visual Studio, **Import from Server (Multidimensional or Data Mining)**, to get a copy of the metadata and solution objects. You can then modify the deployment target, edit data sources, and work with the cube objects without affecting the existing objects.

For more information, see Import a Data Mining Project using the Analysis Services Import Wizard.

Overview of OLAP Data Mining Process

Start the Data Mining Wizard by right-clicking the **Mining Structures** node in Solution Explorer, and selecting **New Mining Structure**. The wizard guides you through the following steps to create the structure for a new structure and model:

1. **Select the Definition Method**: Here you select a data source type, and choose **From existing cube**.

   > **NOTE**
   >
   > The OLAP cube that you use as a source must exist within the same database as the mining structure, as described above. Also, you cannot use a cube created by the Power Pivot for Excel add-in as a source for data mining.

2. **Create the Data Mining Structure**: Determine whether you will build just a structure, or a structure with a mining model.

You must also choose an appropriate algorithm for analyzing your data. For guidance on which algorithm is best for certain tasks, see HYPERLINK "ms-help://SQL111033/as_1devconc/html/ed1fc83b-b98c-437e-bf53-4ff001b92d64.htm" Data Mining Algorithms (Analysis Services - Data Mining).

3. **Select the Source Cube Dimension**: This step is the same as selecting a data source. You need to choose the single dimension that contains the most important data used for in training your model. You can add data from other dimensions later, or filter the dimension.

4. **Select the Case Key**: Within the dimension that you just selected, choose an attribute (column) to serve as the unique identifier for your case data.

   Typically a column will be pre-selected for you, but you can change the column if in fact there are multiple keys.

5. **Selecting Case Level Columns**: Here you choose the attributes from the selected dimension, and the related measures, that are relevant to your analysis. This step is equivalent to selecting columns from a table.

   The wizard automatically includes for your review and selection any measures that were created using attributes from the selected dimension.

   For example, if your cube contains a measure that calculates freight cost based on the customer's geographical location, and you chose the Customer dimension as your main data source for modeling, the measure will be proposed as a candidate for adding to the model. Beware of adding too many measures that are already directly based on attributes, as there is already one implicit relationship between the columns, as defined in the measure formula, and the strength of this (expected) correlation can obscure other relationships that you might otherwise discover.

6. **Specify Mining Model Column Usage**: For each attribute or measure that you added to the structure, you must specify whether the attribute should be used for prediction, or used as input. If you do not select either of these options, the data will be processed but will not be used for analysis; however, it will be available as background data in case you later enable drillthrough.

7. **Add nested tables**: Click to add related tables. In the **Select a Measure Group Dimension** dialog box, you can choose a single dimension from among the dimensions that are related to the current dimension.

   Next, you use the **Select a Nested Table Key** dialog box to define how the new dimension is related to the dimension that contains the case data.

   Use the **Select Nested Table Columns** dialog box to choose the attributes and measures from the new dimension that you want to use in analysis. You must also specify whether the nested attribute will be used for prediction.

   After you have added all the nested attributes you might need, return to the page, **Specify Mining Model Column Usage**, and click **Next**.

8. **Specify Columns Content and Data Type**: By this point, you have added all the data that will be used for analysis, and must specify the *data type* and *content type* for each attribute.

   In an OLAP model, you do not have the option to automatically detect data types, because the data type is already defined by the multidimensional solution and cannot be changed. Keys are also automatically identified. For more information, see Data Types (Data Mining).

   The *content type* that you choose for each column that you use in the model tells the algorithm how the data should be processed. For more information, see Content Types (Data Mining).

9. **Slicing the source cube**: Here you can define filters in a cube to select just a subset of data and train models that are more targeted.

You filter a cube by choosing the dimension to filter on, selecting the level of the hierarchy that contains the criteria you want to use, and then typing a condition to use as the filter.

10. **Create Testing Set**: On this page, you can tell the wizard how much data should be set aside for use in testing the model. If your data will support multiple models, it is a good idea to create a holdout data set, so that all models can be tested on the same data.

    For more information, see Testing and Validation (Data Mining).

11. **Completing the Wizard**: On this page, you give a name to the new mining structure and the associated mining model, and save the structure and model.

    On this page, you can also can set the following options:

    - **Allow drillthrough**
    - **Create mining model dimension**
    - **Create cube using mining model dimension**

    To learn more about these options, see the section later in this topic, Understanding Data Mining Dimensions and Drillthrough.

At this point the mining structure and its model are just metadata; you will need to process them both to get results.

Scenarios for Use of Data Mining with OLAP Data

OLAP cubes frequently contain so many members and dimensions that it can be difficult to know where to begin with data mining. To help identify the patterns that the cubes contain, typically you identify a single dimension of interest, and then begin to explore patterns related to that dimension. The following table lists several common OLAP data mining tasks, describes sample scenarios in which you might apply each task, and identifies the data mining algorithm to use for each task.

TASK	SAMPLE SCENARIO	ALGORITHM
Group members into clusters	Segment a customer dimension based on customer member properties, the products that the customers buy, and the amount of money that the customers spend.	Microsoft Clustering Algorithm
Find interesting or abnormal members	Identify interesting or abnormal stores in a store dimension based on sales, profit, store location, and store size.	Microsoft Decision Trees Algorithm
Find interesting or abnormal cells	Identify store sales that go against typical trends over time.	Microsoft Time Series Algorithm
Find correlations	Identify factors that are related to server downtime, including region, machine type, OS, or purchase date.	Microsoft Naïve Bayes algorithm

Slicing a Cube vs. Filtering Models

Slicing the cube while you are building a model is like creating a filter on a relational mining model. In a relational model, the filter on the data source is defined as a WHERE clause on a SQL statement; in a cube, you use the editor to create filter statements using MDX.

For example, a cube might contains information about purchases of products worldwide, but for your marketing campaign, you want to create a model based on analysis of female customers over 30 who live in the United Kingdom.

In this scenario, you would create two filters:

- For the first filter, you would choose the Geography dimension, choose the hierarchy for Region, and then use the **Filter Expression** list to choose "United Kingdom" from the possible values.
- For the second filter, you would choose the Customer dimension, select the Gender attribute, and select "Female" from the list of attribute values.

After the mining structure has been created, you can modify both the definition of the cube data and the filter criteria. For more information, see Filters for Mining Models.

Both the **Mining Structure** tab and the **Mining Model** tab provide an option to add a filter to an existing mining structure, by clicking **Define a Cube Slice**. The **Slice Cube** dialog box helps you build a valid MDX filter expression by choosing value from dropdown lists.

> **WARNING**
> Note that the interface for designing and browsing cubes has been changed in SQL Server 2017. For more information, see Browse data and metadata in Cube.

You can add as many filters on the cube as are required to return the data that you need for the mining model. You can also define slices on individual cube slices. For example, if your structure contains two nested tables that are based on products, you could slice one table on March 2004 and the other table on April 2004. The resulting model could then be used to predict purchases made in April based on the purchases that were made in March.

Using Nested Tables in an OLAP Mining Model

When you use the Data Mining Wizard to build a model based on cube data, you can add nested tables by specifying the names of related dimensions and then choosing the attributes or measures to add to the model

For example, if the main dimension used for case data is Customer, you might add as a related dimension the Products dimension, because you expect that a customer might have ordered multiple products over time, and the cube already links each customer to the many products via the order fact tables.

You add nested tables in the **Specify Mining Model Column Usage** page of the wizard, by clicking **Add Nested Tables**. A dialog box opens that guides you through process of choosing a related dimension, as well as any measures. The case and nested dimensions must be related by a foreign key, and measures must use one of the attributes that are already included in the case or nested tables. Unfortunately, these restrictions really don't do much to narrow the scope, so you must be careful to select only those attributes that are useful for modeling.

For each attribute or measure that you add to the nested table, you must specify whether the nested attribute will be used for prediction or not, by selecting **Predictable** or **Input** in the **Select Nested Table Columns** dialog box. If you do not select either of these options, the data will be added to the mining structure but not used for analysis.

For each attribute and measure, you must also specify whether the attribute is discrete, discretized, or continuous. The wizard will preselect a default based on the data type of the attribute, but you might need to change these, depending on the algorithm requirements. If you choose a content type that is not compatible with the algorithm you have chosen (for example, you use a continuous numeric type with a Naïve Bayes model), you won't get an error message till you try to process the model.

When you are done setting these options, the wizard adds the nested table to the case table. The default name for

the nested table is the nested dimension name, but you can rename the nested table and its columns. You can repeat this process to add multiple nested tables to the mining structure.

The ability to use nested table data like this is a feature of SQL Server data mining that is particularly powerful, and in a cube, there are almost limitless possibilities for using related subsets of data.

Understanding Data Mining Dimensions and Drillthrough

The option, **Allow drillthrough**, lets you run queries against the underlying cube data while you are browsing the model. The data is not contained in the new data mining dimension, but the Analysis Services database can use the data bindings to retrieve the information from the source cube.

The option, **Create mining model dimension**, lets you generate a new dimension within the existing cube that contains the patterns discovered by the algorithm. The hierarchy within the new dimension is determined largely by the model type. For example, the representation of a clustering model is fairly flat, with the (All) node at the top of the hierarchy and each cluster in the next level. In contrast, the dimension that is created for a decision tree model might have a very deep hierarchy, representing the branching of the tree.

The option, **Create cube using mining model dimension**, lets you export the new data mining dimension into a new cube. Any objects that are required for drillthrough on the data mining dimension will be included automatically.

> **WARNING**
>
> Only these model types support the creation of data mining dimensions: models based on the Microsoft Clustering algorithm, the Microsoft Decision Trees algorithm, or the Microsoft Association algorithm.

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Mining Structure Columns
Mining Model Columns
Mining Model Properties
Properties for Mining Structure and Structure Columns

Add Mining Models to a Structure (Analysis Services - Data Mining)

11/28/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A mining structure is intended to support multiple mining models. Therefore, after you finish the wizard, you can open the structure and add new mining models. Each time that you create a model, you can use a different algorithm, change the parameters, or apply filters to use a different subset of the data.

Adding New Mining Models

When you use the Data Mining Wizard to create a new mining model, by default you must always create a mining structure first. The wizard then gives you the option to add an initial mining model to the structure. However, you don't need to create a model right away. If you create the structure only, you do not need to make a decision about which column to use as the predictable attribute, or how to use the data in a particular model. Instead, you just set up the general data structure that you want to use in future, and later you can use Data Mining Designer to add new mining models that are based on the structure.

> **NOTE**
> In DMX, the CREATE MINING MODEL statement begins with the mining model. That is, you define your choice of mining model, and Analysis Services automatically generates the underlying structure. Later you can continue to add new mining models to that structure, by using the ALTER STRUCTURE... ADD MODEL statement.

Choosing an Algorithm

When you add a new model to an existing structure, the first thing you should do is select a data mining algorithm to use in that model. Choosing the algorithm is important because each algorithm performs a different type of analysis and has different requirements.

When you select an algorithm that is incompatible with your data, you will get a warning. In some cases, you might need to ignore columns that cannot be processed by the algorithm. In other cases, the algorithm will automatically make the adjustments for you. For example, if your structure contains numeric data, and the algorithm can only work with discrete values, it will group the numeric values into discrete ranges for you. In some cases, you might need to manually fix the data first, by choosing a key or choosing a predictable attribute.

You do not need to change the algorithm when you create a new model. Often you can get very different results by using the same algorithm, but filtering the data, or changing a parameter such as the clustering method or the minimum itemset size. We recommend that you experiment with multiple models to see which parameters produce the best results.

Note that all new models need to be processed before you can use them.

Specifying the Usage of Columns in a New Mining Model

When you add new mining models to an existing mining structure, you must specify how each column of data should be used by the model. Depending on the type of algorithm you choose for the model, some of these choices may be made by default. If you do not specify a usage type for a column, the column will not be included in the mining structure. However, the data in the column can still be available for drillthrough, if the model

supports it.

Columns from the mining structure that are used by the model (if not set to Ignore) must be a key, an input column, a predictable column, or a predictable column the values of which are also used as inputs to the model.

- Key columns contain a unique identifier for each row in a table. Some mining models, such as those based on the sequence clustering or time series algorithms, can contain multiple key columns. However, these multiple keys are not compound keys in the relational sense, but instead must be selected so as to provide support for time series and sequence clustering analysis.
- Input columns provide the information from which predictions are made. The Data Mining Wizard provides the **Suggest** feature, which is enabled when you select a predictable column. If you click this button, the wizard will sample the predictable values and determine which of the other columns in the structure make good variables. It will reject key columns or other columns with many unique values, and suggest columns that appear to be correlated with the outcome.

  This feature is particularly handy when datasets contain more columns than you really need to build a mining model. The **Suggest** feature calculates a numeric score, from 0 to 1, that describes the relationship between each column in the dataset and the predictable column. Based on this score, the feature suggests columns to use as input for the mining model. If you use the **Suggest** feature, you can use the suggested columns, modify the selections to fit your needs, or ignore the suggestions.
- Predictable columns contain the information that you try to predict in the mining model. You can select multiple columns as the predictable attributes. Clustering models are the exception in that a predictable attribute is optional.

  Depending on the model type, the predictable column might need to be a specific data type: for example, a linear regression model requires a numeric column as the predicted value; Naïve Bayes algorithm requires a discrete value (and all the inputs must be discrete as well).

Specifying Column Content

For some columns, you might also need to specify the *column content*. In SQL Server data mining, the Content Type property of each data columns tells the algorithm how it should process the data in that column. For example, if your data has an Income column, you must specify that the column contains continuous numbers by setting the content type to Continuous. However, you could also specify that the numbers in the Income column be grouped into buckets by setting the content type to Discretized and optionally specifying the exact number of buckets. You can create different models that handle columns differently: for example, you might try one model that buckets customers into three age groups, and another model that buckets customers into 10 age groups.

See Also

Mining Structures (Analysis Services - Data Mining)
Create a Relational Mining Structure
Mining Model Properties
Mining Model Columns

Customize Mining Models and Structure

11/27/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you have selected an algorithm that meets your business needs, you can customize the mining model in the following ways to potentially improve results.

- Use different columns of data in the model, or change the usage, content type, or discretization method for the columns.
- Create filters on the mining model to restrict the data used in training the model.
- Change the algorithm that was used to analyze data.
- Set algorithm parameters to control thresholds, tree splits, and other important conditions.

This topic describes these options.

Changing Data Used by the Model

The decisions that you make about which columns of data to use in the model, and how to use and process that data, greatly affect the results of analysis. The following topics provide information to help you understand these choices.

Using Feature Selection

Most data mining algorithms in Analysis Services use a process called *feature selection* to select only the most useful attributes for addition to a model. Reducing the number of columns and attributes can improve performance and the quality of the model. The feature selection methods that are available differ depending on the algorithm that you choose.

Feature Selection (Data Mining).

Changing Usage

You can change which columns are included in a mining model and how each column is used. If you do not get the results you expect, you should example the columns you used as input, and ask yourself whether the columns are a good choice, and whether there is anything you can do to improve the handling of data, including:

- Identifying categorical variables that have mistakenly labeled as numbers.
- Adding categories to collapse the number of attributes and make it easier t find correlations.
- Changing the way that numbers are binned, or discretized.
- Removing columns that have a lot of unique values, or columns that are really reference data and not useful for analysis, such as addresses or middle names.

You don't need to physically remove columns from the mining structure; you can just flag the column as **Ignore**. The column is removed from the mining model, but can still be used by other mining models in the structure, ore referenced in a drillthrough query.

Creating Aliases for Model Columns

When Analysis Services creates the mining model, it uses the same column names that are in the mining structure. You can add an alias to any column in the mining model. This might make it easier to understand the

column contents or usage, or make the name shorter for convenience in creating queries. Aliases are also helpful when you want to create a copy of a column and name it something descriptive.

You create an alias by editing the **Name** property of the mining model column. Analysis Services continues to use the original name as the ID of the column, and the new value that you type for **Name** becomes the column alias, and appears in the grid in parentheses next to the column usage.

The graphic shows related models that have multiple copies of a mining structure column, all related to Income. Each copy of the structure column has been discretized in a different way. The models in the diagram each use a different column from the mining structure; however, for convenience in comparing the columns across the models, the column in each model has been renamed to [**Income**].

Adding Filters

You can add a filter to a mining model. A filter is a set of WHERE conditions that restrict the data in the model cases to some subset. The filter is used when training the model, and can optionally be used when you test the model or create accuracy charts.

By adding filters, you can reuse mining structures but create models based on very different subsets of the data. Or, you can simply use filters to eliminate certain rows and improve the quality of analysis.

For more information, see Filters for Mining Models (Analysis Services - Data Mining).

Changing the Algorithm

Although new models that you add to a mining structure share the same data set, you can get different results by using a different algorithm (if the data supports it), or by changing the parameters for the algorithm. You can also set modeling flags.

The choice of algorithm determines what kind of results you will get. For general information about how a specific algorithm works, or the business scenarios where you would benefit from using a particular algorithm, see Data Mining Algorithms (Analysis Services - Data Mining).

See the technical reference topic for each algorithm for a description of the requirements and restrictions, as well as detailed information about the customizations that each algorithm supports.

Microsoft Decision Trees Algorithm	Microsoft Time Series Algorithm
Microsoft Clustering Algorithm	Microsoft Neural Network Algorithm
Microsoft Naive Bayes Algorithm	Microsoft Logistic Regression Algorithm
Microsoft Association Algorithm	Microsoft Linear Regression Algorithm

Microsoft Sequence Clustering Algorithm	

Customizing Algorithm Parameters

Each algorithm supports parameters that you can use to customize the behavior of the algorithm and fine-tune the results of your model. For a description of how to use each parameter, see the following topics:

The topic for each algorithm type also lists the prediction functions that can be used with models based on that algorithm.

PROPERTY NAME	APPLIES TO
AUTO_DETECT_PERIODICITY	Microsoft Time Series Algorithm Technical Reference
CLUSTER_COUNT	Microsoft Clustering Algorithm Technical Reference Microsoft Sequence Clustering Algorithm Technical Reference
CLUSTER_SEED	Microsoft Clustering Algorithm Technical Reference
CLUSTERING_METHOD	Microsoft Clustering Algorithm Technical Reference
COMPLEXITY_PENALTY	Microsoft Decision Trees Algorithm Technical Reference Microsoft Time Series Algorithm Technical Reference
FORCE_REGRESSOR	Microsoft Decision Trees Algorithm Technical Reference Microsoft Linear Regression Algorithm Technical Reference Modeling Flags (Data Mining)
FORECAST_METHOD	Microsoft Time Series Algorithm Technical Reference
HIDDEN_NODE_RATIO	Microsoft Neural Network Algorithm Technical Reference
HISTORIC_MODEL_COUNT	Microsoft Time Series Algorithm Technical Reference
HISTORICAL_MODEL_GAP	Microsoft Time Series Algorithm Technical Reference
HOLDOUT_PERCENTAGE	Microsoft Logistic Regression Algorithm Technical Reference Microsoft Neural Network Algorithm Technical Reference Note: This parameter is different from the holdout percentage value that applies to a mining structure.
HOLDOUT_SEED	Microsoft Logistic Regression Algorithm Technical Reference Microsoft Neural Network Algorithm Technical Reference Note: This parameter is different from the holdout seed value that applies to a mining structure.

PROPERTY NAME	APPLIES TO
INSTABILITY_SENSITIVITY	Microsoft Time Series Algorithm Technical Reference
MAXIMUM_INPUT_ATTRIBUTES	Microsoft Clustering Algorithm Technical Reference Microsoft Decision Trees Algorithm Technical Reference Microsoft Linear Regression Algorithm Technical Reference Microsoft Naive Bayes Algorithm Technical Reference Microsoft Neural Network Algorithm Technical Reference Microsoft Logistic Regression Algorithm Technical Reference
MAXIMUM_ITEMSET_COUNT	Microsoft Association Algorithm Technical Reference
MAXIMUM_ITEMSET_SIZE	Microsoft Association Algorithm Technical Reference
MAXIMUM_OUTPUT_ATTRIBUTES	Microsoft Decision Trees Algorithm Technical Reference Microsoft Linear Regression Algorithm Technical Reference Microsoft Logistic Regression Algorithm Technical Reference Microsoft Naive Bayes Algorithm Technical Reference Microsoft Neural Network Algorithm Technical Reference
MAXIMUM_SEQUENCE_STATES	Microsoft Sequence Clustering Algorithm Technical Reference
MAXIMUM_SERIES_VALUE	Microsoft Time Series Algorithm Technical Reference
MAXIMUM_STATES	Microsoft Clustering Algorithm Technical Reference Microsoft Neural Network Algorithm Technical Reference Microsoft Sequence Clustering Algorithm Technical Reference
MAXIMUM_SUPPORT	Microsoft Association Algorithm Technical Reference
MINIMUM_IMPORTANCE	Microsoft Association Algorithm Technical Reference
MINIMUM_ITEMSET_SIZE	Microsoft Association Algorithm Technical Reference
MINIMUM_DEPENDENCY_PROBABILITY	Microsoft Naive Bayes Algorithm Technical Reference
MINIMUM_PROBABILITY	Microsoft Association Algorithm Technical Reference
MINIMUM_SERIES_VALUE	Microsoft Time Series Algorithm Technical Reference

PROPERTY NAME	APPLIES TO
MINIMUM_SUPPORT	Microsoft Association Algorithm Technical Reference Microsoft Clustering Algorithm Technical Reference Microsoft Decision Trees Algorithm Technical Reference Microsoft Sequence Clustering Algorithm Technical Reference Microsoft Time Series Algorithm Technical Reference
MISSING_VALUE_SUBSTITUTION	Microsoft Time Series Algorithm Technical Reference
MODELLING_CARDINALITY	Microsoft Clustering Algorithm Technical Reference
PERIODICITY_HINT	Microsoft Time Series Algorithm Technical Reference
PREDICTION_SMOOTHING	Microsoft Time Series Algorithm Technical Reference
SAMPLE_SIZE	Microsoft Clustering Algorithm Technical Reference Microsoft Logistic Regression Algorithm Technical Reference Microsoft Neural Network Algorithm Technical Reference
SCORE_METHOD	Microsoft Decision Trees Algorithm Technical Reference
SPLIT_METHOD	Microsoft Decision Trees Algorithm Technical Reference
STOPPING_TOLERANCE	Microsoft Clustering Algorithm Technical Reference

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Physical Architecture (Analysis Services - Data Mining)

Data Mining Designer

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data Mining Designer is the primary environment in which you work with mining models in Microsoft SQL Server Analysis Services. You can access the designer either by selecting an existing mining structure, or by using the Data Mining Wizard to create a new mining structure and mining model. You can use Data Mining Designer to perform the following tasks:

- Modify the mining structure and the mining model that were initially created by the Data Mining Wizard.
- Create new models based on an existing mining structure.
- Train and browse mining models.
- Compare models by using accuracy charts.
- Create prediction queries based on mining models.

Mining Structure Tab

Use the **Mining Structure** tab to add columns and modify the properties of an existing mining structure. The following tasks and topics provide more information about working with mining structures:

Mining Structures (Analysis Services - Data Mining)

Mining Structure Tasks and How-tos

Mining Models Tab

Use the **Mining Models** tab to manage existing mining models and to create new models. Mining models are always based on an existing mining structure .

In the **Mining Models** tab, you can change the algorithm type, add or remove columns that are associated with the model structure, adjust algorithm-specific column properties, specify the mining model column usage, and adjust algorithm parameters that are associated with the mining model. You can also process the mining structure together with selected models or with all the associated models.

See the following topics for more information about working with mining models:

Mining Models (Analysis Services - Data Mining)

Mining Model Tasks and How-tos

Mining Model Viewer Tab

Use the **Mining Model Viewer** tab to visually explore your mining models. Each mining model is associated with a custom viewer that displays content that is specific to that model. You can also view mining model content by using the content viewer.

See the following topics for more information about exploring mining models with the data mining viewers:

Data Mining Model Viewers

Mining Model Viewer Tasks and How-tos

Mining Accuracy Chart Tab

Use the **Mining Accuracy Chart** tab to test the predictive accuracy of a single mining model, or to compare the effectiveness of multiple mining models contained within a mining structure. The tab contains tools for filtering the data, selecting mining models, and displaying the results in a lift chart, profit chart, or classification matrix.

See the following topics for more information about testing and validating mining models:

Testing and Validation (Data Mining)

Testing and Validation Tasks and How-tos (Data Mining)

Mining Model Prediction Tab

The **Mining Model Prediction** tab includes Prediction Query Builder, which you can use to create a Data Mining Extensions (DMX) prediction query. The tab contains tools for specifying mining models and input tables, mapping the columns in the mining model to columns in the input table, adding functions to a query, and specifying criteria for each column.

After you design a query, you can use different views in the tab to display the results of the query and to modify the query manually. You can also save the results of the query to a table in a database.

See the following topics for more information about creating data mining queries:

Data Mining Queries

Data Mining Query Tasks and How-tos

See Also

Data Mining Solutions

Data Mining Model Viewers

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you train a data mining model in Microsoft SQL Server Analysis Services, you can explore the model to look for interesting trends. Because the results of mining models are complex and can be difficult to understand in a raw format, visually investigating the data is often the easiest way to understand the rules and relationships that algorithms discover within the data.

Each algorithm that you use to build a model returns a different type of results. Therefore, Analysis Services provides a separate viewer for each algorithm. When you browse a mining model in SQL Server Data Tools (SSDT), the model is displayed on the **Mining Model Viewer** tab of Data Mining Designer, using the appropriate viewer for the model.

How to Use the Model Viewers

First you select the mining model and then you select a viewer. Each model always has two viewers available: a custom viewer, which can include multiple tabs, and the generic viewer.

Depending on the type of the model that you selected, you will see very different options for exploring the model. The custom viewers associated with each model type are tailored to the algorithm that you used to create the selected data mining model. Each custom viewer has a variety of tools and dialog boxes for helping you explore the statistics and patterns in the model, view charts, or interactively work with probability thresholds or filter out items by name.

The following diagram illustrates the difference when you choose a custom viewer and the generic viewer for the same model.

1. First, you see the custom viewer that is displayed when you select a mining model based on the Microsoft Time Series algorithm.

   This particular custom viewer automatically creates a graph of the time series and provides five predictions.

2. Next, you see the same model, displayed using the **Microsoft Generic Content Tree Viewer**.

   On the left, the generic viewer displays a list of the nodes in the model. You can click a node to view its contents in the right-hand pane.

More about the Microsoft Generic Content Tree Viewer

Each model can also be viewed using the Microsoft Generic Content Tree Viewer (Data Mining). This viewer presents the contents of the mining model according to a standard HTML table format. However, the arrangement of the nodes and the content of each node will differ greatly depending on the algorithm used to generate the results.

Whereas the custom viewers are designed for exploring and understanding the model, the generic viewer is more useful when you already understand the model and want to extract statistics or rules from a specific node. For example, you would use the generic viewer when you want to view detailed information about the patterns and statistics that Analysis Services captured during analysis, such as the probability of a node, or a regression formula.

You can also write *content queries* using DMX to get all of the information that is presented in this viewer. For more information, see Content Queries (Data Mining).

In This Section

The following topics describe in more detail each of the viewers, and how to interpret the information in them.

Browse a Model Using the Microsoft Tree Viewer

Describes the Microsoft Tree Viewer. This viewer displays mining models that are built with the Microsoft Decision Trees algorithm and the Microsoft Linear Regression algorithm.

Browse a Model Using the Microsoft Cluster Viewer
Describes the Microsoft Cluster Viewer. This viewer displays mining models that are built with the Microsoft Clustering algorithm.

Browse a Model Using the Microsoft Time Series Viewer
Describes the Microsoft Time Series Viewer. This viewer displays mining models that are built with the Microsoft Time Series algorithm.

Browse a Model Using the Microsoft Naive Bayes Viewer
Describes the Microsoft Naive Bayes viewer. This viewer displays mining models that are built with the Microsoft Naive Bayes algorithm.

Browse a Model Using the Microsoft Sequence Cluster Viewer
Describes the Microsoft Sequence Cluster Viewer. This viewer displays mining models that are built with the Microsoft Sequence Clustering algorithm.

Browse a Model Using the Microsoft Association Rules Viewer
Describes the Microsoft Association Rules Viewer. This viewer displays mining models that are built with the Microsoft Association algorithm.

Browse a Model Using the Microsoft Neural Network Viewer
Describes the Microsoft Neural Network Viewer. This viewer displays mining models that are built with the Microsoft Neural Network algorithm, including models that use the Microsoft Logistic Regression algorithm.

Browse a Model Using the Microsoft Generic Content Tree Viewer
Describes the detailed information that is available in the generic viewer for all data mining models and provides examples of how to interpret the information for each algorithm.

See Also

Data Mining Algorithms (Analysis Services - Data Mining)
Data Mining Designer

Browse a Model Using the Microsoft Tree Viewer

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Tree Viewer in Microsoft SQL Server Analysis Services displays decision trees that are built with the Microsoft Decision Trees algorithm. The Microsoft Decision Trees algorithm is a hybrid decision tree algorithm that supports both classification and regression. Therefore, you can also use this viewer to view models based on the Microsoft Linear Regression algorithm. The Microsoft Decision Trees algorithm is used for predictive modeling of both discrete and continuous attributes. For more information about this algorithm, see Microsoft Decision Trees Algorithm.

> **NOTE**
> To view detailed information about the equations used in the model and the patterns that were discovered, use the Microsoft Generic Content Tree viewer. For more information, see Browse a Model Using the Microsoft Generic Content Tree Viewer or Microsoft Generic Content Tree Viewer (Data Mining).

Viewer Tabs

When you browse a mining model in Analysis Services, the model is displayed on the **Mining Model Viewer** tab of Data Mining Designer in the appropriate viewer for the model. The Microsoft Tree Viewer includes the following tabs and panes:

- Decision Tree
- Dependency Network
- Mining Legend

Decision Tree

When you build a decision tree model, Analysis Services builds a separate tree for each predictable attribute. You can view an individual tree by selecting it from the **Tree** list on the **Decision Tree** tab of the viewer.

A decision tree is composed of a series of splits, with the most important split, as determined by the algorithm, at the left of the viewer in the **All** node. Additional splits occur to the right. The split in the **All** node is most important because it contains the strongest split-causing conditional in the dataset, and therefore it caused the first split.

You can expand or collapse individual nodes in the tree to show or hide the splits that occur after each node. You can also use the options on the **Decision Tree** tab to affect how the tree is displayed. Use the **Show Level** slider to adjust the number of levels that are shown in the tree. Use **Default Expansion** to set the default number of levels that are displayed for all trees in the model.

Predicting Discrete Attributes

When a tree is built with a discrete predictable attribute, the viewer displays the following on each node in the tree:

- The condition that caused the split.
- A histogram that represents the distribution of the states of the predictable attribute, ordered by popularity.

You can use the **Histogram** option to change the number of states that appear in the histograms in the tree. This is useful if the predictable attribute has many states. The states appear in a histogram in order of popularity from left to right; if the number of states that you choose to display is fewer than the total number of states in the attribute, the least popular states are displayed collectively in gray. To see the exact count for each state for a node, pause the pointer over the node to view an InfoTip, or select the node to view its details in the **Mining Legend**.

The background color of each node represents the concentration of cases of the particular attribute state that you select by using the **Background** option. You can use this option to highlight nodes that contain a particular target in which you are interested.

Predicting Continuous Attributes

When a tree is built with a continuous predictable attribute, the viewer displays a diamond chart, instead of a histogram, for each node in the tree. The diamond chart has a line that represents the range of the attribute. The diamond is located at the mean for the node, and the width of the diamond represents the variance of the attribute at that node. A thinner diamond indicates that the node can create a more accurate prediction. The viewer also displays the regression equation, which is used to determine the split in the node.

Additional Decision Tree Display Options

When drill through is enabled for a decision tree model, you can access the training cases that support a node by right-clicking the node in the tree and selecting **Drill Through**. You can enable drill through within the Data Mining Wizard, or by adjusting the drill through property on the mining model in the **Mining Models** tab.

You can use the zoom options on the **Decision Tree** tab to zoom in or out of a tree, or use **Size to Fit** to fit the whole model in the viewer screen. If a tree is too large to be sized to fit the screen, you can use the **Navigation**option to navigate through the tree. Clicking **Navigation** opens a separate navigation window that you can use to select sections of the model to display.

You can also copy the tree view image to the Clipboard, so that you can paste it into documents or into image manipulation software. Use **Copy Graph View** to copy only the section of the tree that is visible in the viewer, or use **Copy Entire Graph** to copy all the expanded nodes in the tree.

Back to Top

Dependency Network

The **Dependency Network** displays the dependencies between the input attributes and the predictable attributes in the model. The slider at the left of the viewer acts as a filter that is tied to the strengths of the dependencies. If you lower the slider, only the strongest links are shown in the viewer.

When you select a node, the viewer highlights the dependencies that are specific to the node. For example, if you choose a predictable node, the viewer also highlights each node that helps predict the predictable node.

If the viewer contains numerous nodes, you can search for specific nodes by using the **Find Node** button. Clicking **Find Node** opens the **Find Node** dialog box, in which you can use a filter to search for and select specific nodes.

The legend at the bottom of the viewer links color codes to the type of dependency in the graph. For example, when you select a predictable node, the predictable node is shaded turquoise, and the nodes that predict the selected node are shaded orange.

Back to Top

Mining Legend

The **Mining Legend** displays the following information when you select a node in the decision tree model:

- The number of cases in the node, broken down by the states of the predictable attribute.
- The probability of each case of the predictable attribute for the node.

- A histogram that includes a count for each state of the predictable attribute.
- The conditions that are required to reach a specific node, also known as the *node path*.
- For linear regression models, the regression formula.

You can dock and work with the **Mining Legend** in a similar manner as with Solution Explorer.

Back to Top

See Also

Microsoft Decision Trees Algorithm
Mining Model Viewers (Data Mining Model Designer)
Mining Model Viewer Tasks and How-tos
Data Mining Tools
Data Mining Model Viewers

Browse a Model Using the Microsoft Cluster Viewer

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Cluster Viewer in Microsoft SQL Server Analysis Services displays mining models that are built with the Microsoft Clustering algorithm. The Microsoft Clustering algorithm is a segmentation algorithm for use in exploring data to identify anomalies in the data and to create predictions. For more information about this algorithm, see Microsoft Clustering Algorithm.

> **NOTE**
>
> To view detailed information about the equations used in the model and the patterns that were discovered, use the Microsoft Generic Content Tree viewer. For more information, see Browse a Model Using the Microsoft Generic Content Tree Viewer or Microsoft Generic Content Tree Viewer (Data Mining).

Viewer Tabs

When you browse a mining model in Analysis Services, the model is displayed on the **Mining Model Viewer** tab of Data Mining Designer in the appropriate viewer for the model. The Microsoft Cluster Viewer provides the following tabs for use in exploring clustering mining models:

- Cluster Diagram
- Cluster Profiles
- Cluster Characteristics
- Cluster Discrimination

Cluster Diagram

The **Cluster Diagram** tab of the Microsoft Cluster Viewer displays all the clusters that are in a mining model. The shading of the line that connects one cluster to another represents the strength of the similarity of the clusters. If the shading is light or nonexistent, the clusters are not very similar. As the line becomes darker, the similarity of the links becomes stronger. You can adjust how many lines the viewer shows by adjusting the slider to the right of the clusters. Lowering the slider shows only the strongest links.

By default, the shade represents the population of the cluster. By using the **Shading****Variable** and **State** options, you can select which attribute and state pair the shading represents. The darker the shading, the greater the attribute distribution is for a specific state. The distribution decreases as the shading gets lighter.

To rename a cluster, right-click its node and select **Rename Cluster**. The new name is persisted to the server.

To copy the visible section of the diagram to the Clipboard, click **Copy Graph View**. To copy the complete diagram, click **Copy Entire Graph**. You can also zoom in and out by using **Zoom In** and **Zoom Out**, or you can fit the diagram to the screen by using **Scale Diagram to Fit in Window**.

Back to Top

Cluster Profiles

The **Cluster Profiles** tab provides an overall view of the clusters that the algorithm in your model creates. This view displays each attribute, together with the distribution of the attribute in each cluster. An InfoTip for each cell displays the distribution statistics, and an InfoTip for each column heading displays the cluster population.

Discrete attributes are shown as colored bars, and continuous attributes are shown as a diamond chart that represents the mean and standard deviation in each cluster. The **Histogram bars** option controls the number of bars that are visible in the histogram. If more bars exist than you choose to display, the bars of highest importance are retained, and the remaining bars are grouped together into a gray bucket.

You can change the default names of the clusters, to make the names more descriptive. Rename a cluster by right-clicking its column heading and selecting **Rename cluster**. You can also hide clusters by selecting **Hide column**.

To open a window that provides a larger, more detailed view of the clusters, double-click either a cell in the **States** column or a histogram in the viewer.

Click a column heading to sort the attributes in order of importance for that cluster. You can also drag columns to reorder them in the viewer.

Back to Top

Cluster Characteristics

To use the **Cluster Characteristics** tab, select a cluster from the **Cluster** list. After you select a cluster, you can examine the characteristics that make up that specific cluster. The attributes that the cluster contains are listed in the **Variables** columns, and the state of the listed attribute is listed in the **Values** column. Attribute states are listed in order of importance, described by the probability that they will appear in the cluster. The probability is shown in the **Probability** column.

Back to Top

Cluster Discrimination

You can use the **Cluster Discrimination** tab to compare attributes between two clusters. Use the **Cluster 1** and **Cluster 2** lists to select the clusters to compare. The viewer determines the most important differences between the clusters, and displays the attribute states that are associated with the differences, in order of importance. A bar to the right of the attribute shows which cluster the state favors, and the size of the bar shows how strongly the state favors the cluster.

Back to Top

See Also

Microsoft Clustering Algorithm
Mining Model Viewer Tasks and How-tos
Mining Model Viewer Tasks and How-tos
Data Mining Tools
Data Mining Model Viewers

Browse a Model Using the Microsoft Time Series Viewer

5/16/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Time Series Viewer in Microsoft SQL Server Analysis Services displays mining models that are built with the Microsoft Time Series algorithm. The Microsoft Time Series algorithm is a regression algorithm that creates data mining models for prediction of continuous columns, such as product sales, in a forecasting scenario. These time series models can include information based on different algorithms:

- The ARTxp algorithm, which is optimized for short-term prediction.
- The ARIMA algorithm, which is optimized for long-term prediction.
- A blend of the ARTxp and ARIMA algorithms.

For more information about these algorithms, see Microsoft Time Series Algorithm and Microsoft Time Series Algorithm Technical Reference.

> **NOTE**
> To view detailed information about the equations used in the model and the patterns that were discovered, use the Microsoft Generic Content Tree viewer. For more information, see Browse a Model Using the Microsoft Generic Content Tree Viewer or Microsoft Generic Content Tree Viewer (Data Mining).

Viewer Tabs

When you browse a mining model in Analysis Services, the model is displayed on the **Mining Model Viewer** tab of Data Mining Designer in the appropriate viewer for the model. The Microsoft Time Series Viewer provides the following tabs:

- Model
- Charts

Note The information shown for the model content and in the Mining Legend depends on the algorithm that the model uses. However, the **Model** and **Charts** tabs are the same regardless of the algorithm mix.

Model

When you build a time series model, Analysis Services presents the completed model as a tree. If your data contains multiple case series, Analysis Services builds a separate tree for each series. For example, you are predicting sales for the Pacific, North America, and Europe regions. The predictions for each of these regions are case series. Analysis Services builds a separate tree for each of these series. To view a particular series, select the series from the **Tree** list.

For each tree, the time series model contains an **All** node, and then splits into a series of nodes that represent periodic structures discovered by the algorithm. You can click each node to display statistics such as the number of cases and the equation.

If you created the model using ARTxp only, the **Mining Legend** for the root node contains only the total number of cases. For each non-root node, the **Mining Legend** contains more detailed information about the tree split:

for example, it might show the equation for the node and the number of cases. The *rule* in the legend contains information that identifies the series, and the time slice to which the rule applies. For example, the legend text `M200 Europe Amount -2` indicates that the node represents the model for the M200 Europe series, at a period two time slices ago.

If you created the model using ARIMA only, the **Model** tab contains a single node with the caption, **All**. The **Mining Legend** for the root node contains the ARIMA equation.

If you created a mixed model, the root node contains the number of cases and the ARIMA equation only. After the root node, the tree splits into separate nodes for each periodic structure. For each non-root node, the Mining Legend contains both the ARTxp and ARIMA algorithms, the equation for the node, and the number of cases in the node. The ARTxp equation is listed first, and is labeled as the tree node equation. This is followed by the ARIMA equation. For more information about how to interpret this information, see Microsoft Time Series Algorithm Technical Reference.

In general, the decision tree graph shows the most important split, the **All** node, at the left of the viewer. In decision trees, the split after the **All** node is the most important because it contains the condition that most strongly separates the cases in the training data. In a time series model, the main branching indicates the most likely seasonal cycle. Splits after the **All** node appear to the right of the branch.

You can expand or collapse individual nodes in the tree to show or hide the splits that occur after each node. You can also use the options on the **Decision Tree** tab to affect how the tree is displayed. Use the **Show Level** slider to adjust the number of levels that are shown in the tree. Use **Default Expansion** to set the default number of levels that are displayed for all trees in the model.

The shading of the background color for each node signifies the number of cases that are in the node. To find the exact number of cases in a node, pause the pointer over the node to view an InfoTip for the node.

Back to Top

Charts

The **Charts** tab displays a graph that shows the behavior of the predicted attribute over time, together with five predicted future values. The vertical axis of the chart represents the value of the series, and the horizontal axis represents time.

> **NOTE**
> The time slices used on the time axis depend on the units used in your data: they could represent days, months, or even seconds.

Use the **Abs** button to toggle between absolute and relative curves. If your chart contains multiple models, the scale of the data for each model might be very different. If you use an absolute curve, one model might appear as a flat line, whereas another model shows significant changes. This occurs because the scale of one model is greater than the scale of the other model. By switching to a relative curve, you change the scale to show the percentage of change instead of absolute values. This makes it easier to compare models that are based on different scales.

If the mining model contains multiple time series, you can select one or multiple series to display in the chart. Just click the list at the right of the viewer and select the series you want from the list. If the chart becomes too complex, you can filter the series that are displayed by selecting or clearing the series check boxes in the legend.

The chart displays both historical and future data. Future data appears shaded, to differentiate it from historical data. The data values appear as solid lines for historical data and as dotted lines for predictions. You can change the color of the lines that are used for each series by setting properties in SQL Server Data Tools (SSDT) or SQL Server Management Studio. For more information, see Change the Colors Used in the Data Mining Viewer.

You can adjust the range of time that is displayed by using the zoom options. You can also view a specific time range by clicking the chart, dragging a time selection across the chart, and then clicking again to zoom in on the selected range.

You can select how many future time **steps** that you want to see in the model by using **Prediction Steps**. If you select the **Show Deviations** check box, the viewer provides error bars so that you can see how accurate the predicted value is.

Back to Top

See Also

Mining Model Viewer Tasks and How-tos
Microsoft Time Series Algorithm
Time Series Model Query Examples
Data Mining Model Viewers

Browse a Model Using the Microsoft Naive Bayes Viewer

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Naive Bayes Viewer in Microsoft SQL Server Analysis Services displays mining models that are built with the Microsoft Naive Bayes algorithm. The Microsoft Naive Bayes algorithm is a classification algorithm that is highly adaptable to predictive modeling tasks. For more information about this algorithm, see Microsoft Naive Bayes Algorithm.

Because one of the main purposes of a naive Bayes model is to provide a way to quickly explore the data in a dataset, the Microsoft Naive Bayes Viewer provides several methods for displaying the interaction between predictable attributes and input attributes.

> **NOTE**
> If you want to view detailed information about the equations used in the model and the patterns that were discovered, you can switch to the Microsoft Generic Content Tree viewer. For more information, see Browse a Model Using the Microsoft Generic Content Tree Viewer or Microsoft Generic Content Tree Viewer (Data Mining).

Viewer Tabs

When you browse a mining model in Analysis Services, the model is displayed on the **Mining Model Viewer** tab of Data Mining Designer in the appropriate viewer for the model. The Microsoft Naive Bayes Viewer provides the following tabs for exploring data:

- Dependency Network
- Attribute Profiles
- Attribute Characteristics
- Attribute Discrimination

Dependency Network

The **Dependency Network** tab displays the dependencies between the input attributes and the predictable attributes in a model. The slider at the left of the viewer acts as a filter that is tied to the strengths of the dependencies. Lowering the slider shows only the strongest links.

When you select a node, the viewer highlights the dependencies that are specific to the node. For example, if you choose a predictable node, the viewer also highlights each node that helps predict the predictable node.

The legend at the bottom of the viewer links color codes to the type of dependency in the graph. For example, when you select a predictable node, the predictable node is shaded turquoise, and the nodes that predict the selected node are shaded orange.

Back to Top

Attribute Profiles

The **Attribute Profiles** tab displays histograms in a grid. You can use this grid to compare the predictable attribute that you select in the **Predictable** box to all other attributes that are in the model. Each column in the tab represents a state of the predictable attribute. If the predictable attribute has many states, you can change the number of states that appear in the histogram by adjusting the **Histogram bars**. If the number you choose is less than the total number of states in the attribute, the states are listed in order of support, with the remaining states collected into a single gray bucket.

To display a Mining Legend that relates the colors of the histogram to the states of an attribute, click the **Show Legend** check box. The Mining Legend also displays the distribution of cases for each attribute-value pair that you select.

To copy the contents of the grid to the Clipboard as an HTML table, right-click the **Attribute Profiles** tab and select **Copy**.

Back to Top

Attribute Characteristics

To use the **Attribute Characteristics** tab, select a predictable attribute from the **Attribute** list and select a state of the selected attribute from the **Value** list. When you set these variables, the **Attribute Characteristics** tab displays the states of the attributes that are associated with the selected case of the selected attribute. The attributes are sorted by importance.

Back to Top

Attribute Discrimination

To use the **Attribute Discrimination** tab, select a predictable attribute and two of its states from the **Attribute**, **Value 1**, and **Value 2** lists. The grid on the **Attribute Discrimination** tab then displays the following information in columns:

Attribute
Lists other attributes in the dataset that contain a state that highly favors one state of the predictable attribute.

Values
Shows the value of the attribute in the **Attribute** column.

Favors <value 1>
Shows a colored bar that indicates how strongly the attribute value favors the predictable attribute value shown in **Value 1**.

Favors <value 2>
Shows a colored bar that indicates how strongly the attribute value favors the predictable attribute value shown in **Value 2**.

Back to Top

See Also

Microsoft Naive Bayes Algorithm
Mining Model Viewer Tasks and How-tos
Data Mining Tools
Data Mining Model Viewers

Browse a Model Using the Microsoft Sequence Cluster Viewer

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Sequence Cluster Viewer in Microsoft SQL Server Analysis Services displays mining models that are built with the Microsoft Sequence Clustering algorithm. The Microsoft Sequence Clustering algorithm is a sequence analysis algorithm for use in exploring data that contains events that can be linked by following paths, or *sequences*. For more information about this algorithm, see Microsoft Sequence Clustering Algorithm.

> **NOTE**
> To view detailed information about the equations used in the model and the patterns that were discovered, use the Microsoft Generic Content Tree viewer. For more information, see Browse a Model Using the Microsoft Generic Content Tree Viewer or Microsoft Generic Content Tree Viewer (Data Mining).

> **NOTE**
> The Microsoft Sequence Cluster Viewer provides functionality and options that are similar to the Microsoft Cluster Viewer. For more information, see Browse a Model Using the Microsoft Cluster Viewer.

Viewer Tabs

When you browse a mining model in Analysis Services, the model is displayed on the **Mining Model Viewer** tab of Data Mining Designer in the appropriate viewer for the model. The Microsoft Sequence Cluster Viewer provides the following tabs for use in exploring sequence clustering mining models:

- Cluster Diagram
- Cluster Profiles
- Cluster Characteristics
- Cluster Discrimination
- Cluster Transitions

Cluster Diagram

The **Cluster Diagram** tab of the Microsoft Sequence Cluster Viewer displays all the clusters that are in a mining model. The shading of the line that connects one cluster to another represents the strength of the similarity of the clusters. If the shading is light or nonexistent, the clusters are not very similar. As the line becomes darker, the similarity of the links becomes stronger. You can adjust how many lines the viewer shows by adjusting the slider to the right of the clusters. Lowering the slider shows only the strongest links.

By default, the shade represents the population of the cluster. By using the **Shading****Variable** and **State** options, you can select which attribute and state pair the shading represents. The darker the shading, the greater the attribute distribution is for a specific state. The distribution decreases as the shading gets lighter.

To rename a cluster, right-click its node and select **Rename Cluster**. The new name is persisted to the server.

To copy the visible section of the diagram to the Clipboard, click **Copy Graph View**. To copy the complete diagram, click **Copy Entire Graph**. You can also zoom in and out by using **Zoom In** and **Zoom Out**, or you can fit the diagram to the screen by using **Scale Diagram to Fit in Window**.

Back to Top

Cluster Profiles

The **Cluster Profile** tab provides an overall view of the clusters that the algorithm in your model creates. Each column that follows the **Population** column in the grid represents a cluster that the model discovered. The <attribute>.samples row represents different sequences of data that exist in the cluster, and the <attribute> row describes all the items that the cluster contains and their overall distribution.

The **Histogram bars** option controls the number of bars that are visible in the histogram. If more bars exist than you choose to display, the bars of highest importance are retained, and the remaining bars are grouped together into a gray bucket.

You can change the default names of the clusters, to make the names more descriptive. Rename a cluster by right-clicking its column heading and selecting **Rename cluster**. You can hide clusters by selecting **Hide column**, and you can also drag columns to reorder them in the viewer.

To open a window that provides a larger, more detailed view of the clusters, double-click either a cell in the **States** column or a histogram in the viewer.

Back to Top

Cluster Characteristics

To use the **Cluster Characteristics** tab, select a cluster from the **Cluster** list. After you select a cluster, you can examine the characteristics that make up that specific cluster. The attributes that the cluster contains are listed in the **Variables** columns, and the state of the listed attribute is listed in the **Values** column. Attribute states are listed in order of importance, described by the probability that they will appear in the cluster. The probability is shown in the **Probability** column.

Back to Top

Cluster Discrimination

You can use the **Cluster Discrimination** tab to compare attributes between two clusters, to determine how items in a sequence favor one cluster over another. Use the **Cluster 1** and **Cluster 2** lists to select the clusters to compare. The viewer determines the most important differences between the clusters, and displays the attribute states that are associated with the differences, in order of importance. A bar to the right of the attribute shows which cluster the state favors, and the size of the bar shows how strongly the state favors the cluster.

Back to Top

Cluster Transitions

By selecting a cluster on the **Cluster Transitions** tab, you can browse the transitions between sequence states in the selected cluster. Each node in the viewer represents a state of the sequence column. An arrow represents a transition between two states and the probability that is associated with the transition. If a transition returns back to the originating node, an arrow can point back to the originating node.

An arrow that originates from a dot represents the probability that the node is the start of a sequence. An ending edge that leads to a null represents the probability that the node is the end of the sequence.

You can filter the edge of the nodes by using the slider at the left of the tab.

Back to Top

See Also

Mining Model Viewer Tasks and How-tos
Mining Model Viewer Tasks and How-tos
Microsoft Sequence Clustering Algorithm
Data Mining Tools
Data Mining Model Viewers
Browse a Model Using the Microsoft Cluster Viewer

Browse a Model Using the Microsoft Association Rules Viewer

11/28/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Association Rules Viewer in Microsoft SQL Server Analysis Services displays mining models that are built with the Microsoft Association algorithm. The Microsoft Association algorithm is an association algorithm for use in creating data mining models that you can use for market basket analysis. For more information about this algorithm, see Microsoft Association Algorithm.

Following are the primary reasons for using the Microsoft Association algorithm:

- To find itemsets that describe items that are typically found together in a transaction.
- To discover rules that predict the presence of other items in a transaction based on existing items.

> **NOTE**
> To view detailed information about the equations used in the model and the patterns that were discovered, use the Microsoft Generic Content Tree viewer. For more information, see Browse a Model Using the Microsoft Generic Content Tree Viewer or Microsoft Generic Content Tree Viewer (Data Mining).

For a walkthrough of how to create, explore, and use an association mining model, see Lesson 3: Building a Market Basket Scenario (Intermediate Data Mining Tutorial).

Viewer Tabs

When you browse a mining model in Analysis Services, the model is displayed on the **Mining Model Viewer** tab of Data Mining Designer in the appropriate viewer for the model. The Microsoft Association Rules Viewer includes the following tabs:

- Itemsets
- Rules
- Dependency Net

Each tab contains the **Show long name** check box, which you can use to show or hide the table from which the itemset originates in the rule or itemset.

Itemsets

The **Itemsets** tab displays the list of itemsets that the model identified as frequently found together. The tab displays a grid with the following columns: **Support**, **Size**, and **Itemset**. For more information about support, see Microsoft Association Algorithm. The **Size** column displays the number of items in the itemset. The **Itemset** column displays the actual itemset that the model discovered. You can control the format of the itemset by using the **Show** list, which you can set to the following options:

- **Show attribute name and value**
- **Show attribute value only**
- **Show attribute name only**

You can filter the number of itemsets that are displayed in the tab by using **Minimum support** and **Minimum itemset size**. You can limit the number of displayed itemsets even more by using **Filter Itemset** and entering an itemset characteristic that must exist. For example, if you type "Water Bottle = existing", you can limit the itemsets to only those that contain a water bottle. The **Filter Itemset** option also displays a list of the filters that you have used previously.

You can sort the rows in the grid by clicking a column heading.

Back to Top

Rules

The **Rules** tab displays the rules that the association algorithm discovered. The **Rules** tab includes a grid that contains the following columns: **Probability**, **Importance**, and **Rule**. The probability describes how likely the result of a rule is to occur. The importance is designed to measure the usefulness of a rule. Although the probability that a rule will occur may be high, the usefulness of the rule may in itself be unimportant. The importance column addresses this. For example, if every itemset contains a specific state of an attribute, a rule that predicts state is trivial, even though the probability is very high. The greater the importance, the more important the rule is.

You can use **Minimum probability** and **Minimum importance** to filter the rules, similar to the filtering you can do on the **Itemsets** tab. You can also use **Filter Rule** to filter a rule based on the states of the attributes that it contains.

You can sort the rows in the grid by clicking a column heading.

Back to Top

Dependency Net

The **Dependency Net** tab includes a dependency network viewer. Each node in the viewer represents an item, such as "state = WA". The arrow between nodes represents the association between items. The direction of the arrow dictates the association between the items according to the rules that the algorithm discovered. For example, if the viewer contains three items, A, B, and C, and C is predicted by A and B, if you select node C, two arrows point toward node C - A to C and B to C.

The slider at the left of the viewer acts as a filter that is tied to the probability of the rules. Lowering the slider shows only the strongest links.

Back to Top

See Also

Microsoft Association Algorithm
Mining Model Viewer Tasks and How-tos
Mining Model Viewer Tasks and How-tos
Data Mining Tools
Data Mining Model Viewers

Browse a Model Using the Microsoft Neural Network Viewer

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Neural Network Viewer in Microsoft SQL Server Analysis Services displays mining models that are built with the Microsoft Neural Network algorithm. The Microsoft Neural Network algorithm creates classification and regression mining models that can analyze multiple inputs and outputs, and is very useful for open-ended analyses and exploration. For more information about this algorithm, see Microsoft Neural Network Algorithm.

When you explore a model using the Microsoft Neural Network Viewer, you typically pick some target attribute and state, and then use the viewer to see how input attributes affect the outcome

For example, suppose you know these facts about a class of potential customers:

- Middle aged (40 to 50 years old).
- Owns a home.
- Has two children who still live at home.

How can you correlate these attributes with the likelihood that the customer will make a purchase?

By building a neural network model using purchasing behavior as the target outcome, you can explore multiple combinations on customer attributes, such as high income, and discover which combination of attributes is most likely to influence purchasing behavior. For example, you might discover that the determining factor is the distance that they commute to work.

If you need to more view detailed information, such as the equations that represent each pattern that was discovered, you can switch views and use the Microsoft Generic Content Tree viewer. For more information, see Browse a Model Using the Microsoft Generic Content Tree Viewer or Microsoft Generic Content Tree Viewer (Data Mining).

Viewer Tabs

When you browse a mining model in Analysis Services, the model is displayed on the **Mining Model Viewer** tab of Data Mining Designer in the appropriate viewer for the model. The Microsoft Neural Network Viewer provides the following tabs for use in exploring neural network mining models:

- Inputs
- Outputs
- Variables

Inputs

Use the **Inputs** tab to choose the attributes and values that the model used as inputs. By default, the viewer opens with all attributes included. In this default view, the model chooses which attribute values are the most important to display.

To select an input attribute, click inside the **Attribute** column of the **Input** grid, and select an attribute from the drop-down list. (Only attributes that are included in the model are included in the list.)

The first distinct value appears under the **Value** column. Clicking the default value reveals a list that contains all the possible states of the associated attribute. You can select the state that you want to investigate. You can select as many attributes as you want.

Back to Top

Outputs

Use the **Outputs** tab to choose the outcome attribute to investigate. You can choose any two outcome states to compare, assuming the columns were defined as predictable attributes when the model was created.

Use the **OutputAttribute** list to select an attribute. You can then select two states that are associated with the attribute from the **Value 1** and **Value 2** lists. These two states of the output attribute will be compared in the **Variables** pane.

Back to Top

Variables

The grid in the **Variables** tab contains the following columns: **Attribute**, **Value**, **Favors [value 1]**, and **Favors [value 2]**. By default, the columns are sorted by the strength of **Favors [value 1]**. Clicking a column heading changes the sort order to the selected column.

A bar to the right of the attribute shows which state of the output attribute the specified input attribute state favors. The size of the bar shows how strongly the output state favors the input state.

Back to Top

See Also

Microsoft Neural Network Algorithm
Mining Model Viewer Tasks and How-tos
Mining Model Viewer Tasks and How-tos
Data Mining Tools
Data Mining Model Viewers

Browse a Model Using the Microsoft Generic Content Tree Viewer

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The Microsoft Generic Mining Model Content Viewer provides detailed information about the patterns found by the mining algorithm, and also provides access to various statistics generated during the analysis process. The amount and type of information depends on the algorithm that was used, but can include the following categories:

- Segments of data, and their characteristics.
- Descriptive statistics about each group or about the whole set of data.
- The number of branches or child nodes in a tree.
- Calculations, such as variance and mean, for a cluster or a whole set of data.

Viewing this information can help you better understand the results of your analysis. You can also identify ways to fine-tune, and then retrain, your model. Or, you might decide to retrain by using a different algorithm.

Viewing Mining Model Content

The Microsoft Generic Content Viewer displays the columns, rules, properties, attributes, nodes, and other content from the *content schema rowset* of the mining model. The content schema rowset is a generic framework for presenting detailed information about the content of a data mining model.

This detailed information is contained in an HTML table that represents the patterns, clusters, or trees in the model as nodes. You can click on each node and expand it to see more detail, such as the formulas or the count of distinct values for a numeric attribute. You can also explore the child-parent relationships among the nodes.

For more information about the general meaning of the terms used in the mining model content, see Mining Model Content (Analysis Services - Data Mining). The topic also contains links to information about mining model content for specific types of models. Each type of mining model contains information that is highly specific to the algorithm and the patterns found in the data, so we recommend that you consult the technical reference topic for each model type in order to fully understand each model type.

Querying Mining Model Content

The same information that is provided by the Microsoft Generic Content Tree Viewer is also available by querying the mining model. You can create queries against mining model content by using Data Mining Extensions (DMX) statements. For example, in SQL Server Management Studio, you can run a content query by executing the following DMX statement:

```
SELECT * FROM [<mining model name>].CONTENT
```

For more information, see Data Mining Queries.

See Also

Microsoft Generic Content Tree Viewer (Data Mining)
Data Mining Algorithms (Analysis Services - Data Mining)

Mining Model Viewer Tasks and How-tos

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data Mining Designer in SQL Server Data Tools (SSDT) contains several tools that you can use to explore mining models. The following topics provide step-by-step instructions on how to complete tasks that are specific to using the **Mining Model Viewer** tab in the designer.

In This Section

- Select a Mining Model and a Data Mining Viewer
- Copy a View of a Mining Model
- Find a Specific Node in a Dependency Network
- Filter a Rule in an Association Rules Model
- Filter an Itemset in an Association Rules Model
- Drill Through to Case Data from a Mining Model
- View the Formula for a Time Series Model (Data Mining)
- Change the Colors Used in the Data Mining Viewer

See Also

Basic Data Mining Tutorial
Mining Model Viewers (Data Mining Model Designer)

Select a Mining Model and a Data Mining Viewer

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can explore a mining model by using one of the viewers on the **Mining Model Viewer** tab of Data Mining Designer. You can easily switch between models, or change the viewer that is used.

- The **Mining Model** drop-down list box on the **Mining Model Viewer** tab of Data Mining Designer in SQL Server Data Tools (SSDT) contains a list of all the mining models that are in the current mining structure.
- Custom viewers are provided for each type of model. For an overview of all the custom viewers, see Mining Model Viewers (Data Mining Model Designer). For a walkthrough of how to use the custom viewers to understand a model, see Lesson 4: Exploring the Targeted Mailing Models (Basic Data Mining Tutorial).
- The Microsoft Generic Content Viewer shows the patterns discovered by the algorithm in a standard representation of nodes in a tree. Although the generic tree view shows all the content for the model in rich detail, it is more difficult to interpret. For more information, see Browse a Model Using the Microsoft Generic Content Tree Viewer.

To select a viewer type

- Select the viewer that is provided for the specific type of model that you created, using the **Viewer** list at the top of the **Mining Model Viewer** tab.
- Or, use the **Microsoft Generic Mining Content Viewer**

To select a mining model to view

- On the **Mining Model Viewer** tab in Data Mining Designer, select the mining model that you want to view from the **Mining Model** list.

The selected mining model opens in the viewer that is provided for that model type.

See Also

Mining Model Viewer Tasks and How-tos

Copy a View of a Mining Model

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The **Mining Model Viewer** tab of Data Mining Designer in SQL Server Data Tools (SSDT) uses a separate viewer for each type of mining model. Several of the viewers have components from which you can copy the contents to the Clipboard, and from there paste the contents into a document or into image manipulation software. The following components make this functionality available:

- Cluster Diagram in the Microsoft Cluster Viewer and the Microsoft Sequence Cluster Viewer
- Decision Tree in the Microsoft Tree Viewer and the Microsoft Time Series Viewer
- State Transitions in the Microsoft Sequence Cluster Viewer
- Dependency Network in the Microsoft Association Rules Viewer, the Microsoft Naive Bayes Viewer, and the Microsoft Tree Viewer
- Mining model content, from the Node Details pane of the Microsoft Generic Content Tree Viewer

You can copy the complete representation of the mining model, or just the part that is visible in the viewer.

> **WARNING**
> When you copy a model using the viewer, it does not create a new model object. To create a new model, you must use either the wizard, or the Data Mining Designer,. For more information, see Make a Copy of a Mining Model.

To copy the complete model to the Clipboard

1. From the **Mining Model** list on the **Mining Model Viewer** tab, select the mining model that you want to view.
2. Select the appropriate tab, such as the **Dependency Network** tab, and then click **Copy Entire Graph** on the toolbar of that tab.

To copy the visible piece of the model to the Clipboard

1. From the **Mining Model** list on the **Mining Model Viewer** tab, select the mining model that you want to view.
2. Select the appropriate tab, such as the **Dependency Network** tab, and then zoom in or out to view the model at the level that you want.
3. Click **Copy Graph View** on the toolbar of the selected tab.

To copy the mining model content to the Clipboard

1. From the **Mining Model** list on the **Mining Model Viewer** tab, select the mining model that you want to view.
2. From the **Viewer** drop-down list, select **Microsoft Generic Content Tree Viewer**.
3. In the **Node Caption (Unique ID)** pane, click a node.
4. Right-click the **Node Details** pane and then select **Select All**.
5. Right-click the **Node Details** pane again and select **Copy**.

See Also

Mining Model Viewer Tasks and How-tos

Find a Specific Node in a Dependency Network

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A dependency network in a Microsoft SQL Server Analysis Services mining model can contain many nodes, making it difficult to locate the data you are interested in. To solve this problem, you can use the **Find Node** dialog box on the **Dependency Network** tab of Data Mining Designer to search for a specific node.

To find a specific node in a dependency network

1. On the **Mining Model Viewer** tab of **Data Mining Designer** in SQL Server Data Tools (SSDT), click **Find Node** on the toolbar of the **Dependency Network** tab of the mining model viewer.

   The **Find Node** dialog box opens.

2. In the **Node name contains** box, enter part of the name of the node for which you want to search.

   The list of nodes is filtered to display only those nodes that contain part of the search path.

3. Select the correct node from the list, and then click **OK**.

See Also

Mining Model Viewer Tasks and How-tos

Filter a Rule in an Association Rules Model

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can use filtering with association models to restrict the results to just the associations that interest you. For example, you might filter the rules to show only those that include a specific product.

In Data Mining Designer, you use the controls on the **Rules** tab of the Microsoft Association Rules Viewer to filter the rules that are displayed. You can also create a query on the model to see only itemset that contains a particular value.

> **NOTE**
> This option is available only for mining models that have been created by using the Microsoft Association Algorithm.

Filter a rule in an association model

1. Open the mining model by using the **Association Rules Viewer**. To do this in SQL Server Management Studio, right click the model name and select **Browse**. To do this in SQL Server Data Tools (SSDT), double-click the mining structure that contains the model, and then click the **Mining Model Viewer** tab of **Data Mining Designer**.

2. Click the **Rules** tab of the **Association Rules Viewer**.

3. Type a rule condition into the **Filter Rule** box. For example, a rule condition might be "Bike Stand", which also returns "Bike Stands".

   The **Filter Rule** text box supports regular expressions as defined by the .NET language. Therefore, you can use expressions such as the following: `((.Helmets.*Fenders.*)|(.*Fenders.*Helmets.*))`. This expression would return any itemsets that include attributes with the words Helmets and Fenders, in any order.

4. For **Minimum probability**, increase the value of probability to see fewer rules, or decrease the value to see more rules.

5. For **Minimum importance**, increase the value of importance to see fewer rules, or decrease the value to see more rules.

6. For **Show**, select one of the following options: **Show attribute name and value**, **Show attribute name only**, or **Show attribute value only**.

7. For **Maximum rows**, increase the value to increase the total number of rules that meet the specified conditions, or decrease the value to limit the number of rules returned. Rules are ordered by probability, so you might eliminate additional rules that meet the specified conditions for probability or importance.

8. Select or deselect the **Show long name** check box to toggle the way that the rules names are displayed.

   The rules are now filtered to only display rules that contain the indicated item. The filter condition applies to attribute values either before or after the rule delimiter, "->".

> **NOTE**
> The viewer caches the initial list of rules, based on a query to the mining model, and does not refresh the list of rules unless you change the conditions of the query by setting the maximum rows, the probability, importance, or the display of long names. Therefore, if you type a condition and the display does not immediately refresh, you can force the viewer to refresh the data by selecting and then deselecting the **Show long names** check box.

Create a query on the itemsets in an association model

- Association Model Query Examples

See Also

Mining Model Viewer Tasks and How-tos
Browse a Model Using the Microsoft Association Rules Viewer
Lesson 3: Building a Market Basket Scenario (Intermediate Data Mining Tutorial)

Filter an Itemset in an Association Rules Model

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Microsoft SQL Server Analysis Services, you can filter the itemsets that are displayed in the **Itemsets** tab of the Microsoft Association Rules Viewer.

To filter an itemset

1. On the **Mining Model Viewer** tab of Data Mining Designer in SQL Server Data Tools (SSDT), click the **Itemsets** tab of the **Association Rules Viewer**.
2. Enter a rule condition in the **Filter itemset** box. For example, a rule condition might be "Touring-1000 = existing"
3. Click **Enter**.

The itemsets are now filtered to display only those itemsets that contain the selected items. The box is not case-sensitive. Filters are stored in memory so that you can select an old filter from the list.

See Also

Mining Model Viewer Tasks and How-tos

Use Drillthrough from the Model Viewers

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Depending on the model type, you can use drillthrough from the browse viewers on the **Mining Model Viewer** tab of Data Mining Designer to explore the cases used in the mining model or to see additional columns in the mining structure. Although many model types do not support drillthrough because the patterns in the model cannot be directly linked to specific cases, the following model types do support drillthrough.

Note that drillthrough must have been enabled on the model, and you must have the appropriate permissions. The drillthrough option might also be disabled if the model is in an unprocessed state, regardless of whether the model was previously processed and has content. To retrieve model case data by using drillthrough, the cache of the structure and model must be current.

Use drillthrough in the Microsoft Tree Viewer

1. In Data Mining Designer, select a decision trees model, and select **Browse Model** to open the model in the **Microsoft Tree Viewer**. In SQL Server Management Studio, right-click the model, and select **Browse**
2. Right-click any node in the tree graph, and select **Drill through**.
3. Select one of the following options: **Model Columns Only** or **Model and Structure Columns**. If you do not have permissions, an option might not be available.
4. The **Drill Through** dialog box opens, displaying the case data and/or structure data. The title bar of the dialog box also contains a description that identifies the node from which the drillthrough query was executed.
5. Right-click anywhere in the results and select **Copy All** to save the results to the Clipboard.

Use drillthrough in the Microsoft Cluster Viewer

1. In Data Mining Designer, select a clustering model, and select **Browse Model** to open the model in the **Microsoft Cluster Viewer**. In SQL Server Management Studio, right-click the model, and select **Browse**.
2. On the **Cluster** tab, right-click any node.
3. Select **Drill through**, and then select one of the following options: **Model Columns Only** or **Model and Structure Columns**. If you do not have permissions, an option might not be available.
4. The **Drill Through** dialog box opens, displaying the case data and/or structure data. The title bar of the dialog box also contains a description that identifies the cluster for the cases.
5. Right-click anywhere in the results and select **Copy All** to save the results to the Clipboard.

Use drillthrough in the Microsoft Association Rules Viewer

1. In Data Mining Designer, select an association model, and select **Browse Model** to open the model in the **Microsoft Association Rules Viewer**. In SQL Server Management Studio, right-click the model, and select **Browse**
2. On the **Rules** tab, right-click any row that represents a rule. On the **Itemsets** tab, click on any row that contains an itemset.
3. Select **Drill through**, and then select one of the following options: **Model Columns Only** or **Model and Structure Columns**. If you do not have permissions, an option might not be available.

4. The **Drill Through** dialog box opens, displaying the case data and/or structure data. The title bar of the dialog box also contains a description that identifies the rule name.

5. Right-click anywhere in the results and select **Copy All** to save the complete case results to the Clipboard. You can also select **Copy** to copy just the selected case. If the model contains a nested table column, only the name of the nested table column is pasted; to retrieve the data values inside the nested table column for each case you must create a query on the model content.

Use drillthrough in the Microsoft Sequence Cluster Viewer

1. In Data Mining Designer, select a clustering model, and select **Browse Model** to open the model in the **Microsoft Cluster Viewer**. In SQL Server Management Studio, right-click the model, and select **Browse**.

2. On the **Cluster Diagram tab**, right-click any node that represents a cluster. From the **Cluster Profiles** tab, click anywhere in a cluster profile or in the cluster representing the total model population.

3. Select **Drill through**, and then select one of the following options: **Model Columns Only** or **Model and Structure Columns**. If you do not have permissions, an option might not be available.

4. The **Drill Through** dialog box opens, displaying the case data and/or structure data. The title bar of the dialog box also contains a description that identifies the cluster for the cases.

5. Right-click anywhere in the results and select **Copy All** to save the results to the Clipboard. If the model contains a nested table column, only the name of the nested table column is pasted; to retrieve the data values inside the nested table column for each case you must create a query on the model content.

See Also

Mining Model Viewer Tasks and How-tos
Drillthrough on Mining Models
Drillthrough on Mining Structures

View the Formula for a Time Series Model (Data Mining)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If you created a time series model using SQL Server Data Mining, the easiest way to see the regression equation for the model is to use the **Mining Legend** of the Microsoft Time Series Viewer, which presents all the constants in a readable format.

To view the ARTXP regression formula for a time series model

1. In SQL Server Management Studio, select the time series model that you want to view, and click **Browse**.

   -- or --

   In SQL Server Data Tools (SSDT), select the time series model, and then click the **Mining Model Viewer** tab.

2. Click the **Model** tab.

3. If the model contains multiple trees, select a single tree from the **Tree** drop-down list.

   > **NOTE**
   >
   > A model will always have multiple trees if you have more than one data series. However, you will not see as many trees in the **Time Series viewer** as you will see in the Microsoft Generic Content Tree Viewer. That is because the Time Series viewer combines the ARIMA and ARTXP information for each data series into a single representation.

4. Click any leaf node in the tree.

   Nodes that are labeled as **Data Series** are always leaf nodes and can contain an equation. If an **(All)** node has no child nodes, it can also contain an equation.

5. If the **Mining Legend** is not available, right-click the node, and select **Show Legend**.

   The ARTXP formula is displayed in the first half of the **Mining Legend**, as the **Tree node equation**.

To view the ARIMA formula for a time series model

1. In SQL Server Management Studio, select the time series model that you want to view, and click **Browse**.

   -- or --

In SQL Server Data Tools (SSDT), select the time series model, and then click the **Mining Model Viewer** tab.

2. Click the **Model** tab.

3. If the model contains multiple trees, select a single tree from the **Tree** drop-down list.

> **NOTE**
> The model will always have multiple trees if you include more than one data series.

4. Click any node in the tree.

   The ARIMA formula is displayed in the second half of the **Mining Legend**, as the **ARIMA equation**.

5. If the **Mining Legend** is not available, right-click the node, and select **Show Legend**.

To get the coefficients and terms for the equation

1. You can also get the terms and coefficients of the regression formula for a time series model by creating a **content query** on the model content.

   For more information, see Time Series Model Query Examples

2. You can also browse the time series models and find the terms and coefficients by using the Microsoft Generic Content Tree Viewer.

   For more information, see Mining Model Content for Time Series Models (Analysis Services - Data Mining).

> **NOTE**
> If you browse the content of a mixed model that uses both the ARIMA and ARTXP models, the two models are in separate trees, joined at the root node representing the model. Even though the ARIMA and ARTXP models are presented in one viewer for convenience, the structures are very different, as are the equations, which cannot be combined or compared. The ARTXP tree is more like a decision tree, whereas the ARIMA tree represents a series of moving averages.

See Also

Mining Model Viewer Tasks and How-tos
Browse a Model Using the Microsoft Time Series Viewer

Change the Colors Used in the Data Mining Viewer

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can change the colors that are used in the data mining viewers to display data series, nodes, or clusters. You do this by setting options in SQL Server Data Tools (SSDT) or SQL Server Management Studio. After you have changed the settings, the color selections apply to all models that you view by using SQL Server Data Tools (SSDT); however, you must close SQL Server Data Tools (SSDT) and reopen the model in the viewer to see the new colors.

To change the colors used in the data mining viewers

1. On the **Tools** menu of SQL Server Data Tools (SSDT) or SQL Server Management Studio, select **Options**.
2. In the **Options** dialog box, expand **Designers**, and then expand **Data Mining Viewers**.
3. Find the viewer property that you want to change, and click the current color selection.
4. Select a new color from the dropdown list. Click the **Custom** tab to select a color from the palette of custom colors available on the current computer, or click **System** to select a color from among the colors specified in the Windows color scheme.

> **NOTE**
>
> If you use a custom color, the color is represented by a color icon and the RBG values of the color. If you use a System color or Web color, the color is represented by a color icon and the name of the color.

SQL Server Data Mining Add-Ins for Office

11/28/2018 • 2 minutes to read • Edit Online

SQL Server 2012 (11.x) Data Mining Add-ins for Office is a lightweight set of tools for predictive analytics that lets you use data in Excel to build analytical models for prediction, recommendation, or exploration.

> **IMPORTANT**
>
> The data mining add-in for Office is not supported in Office 2016 or later.

The wizards and data management tools in the add-ins provide step-by-step instruction for these common data mining tasks:

- **Organize and clean your data prior to modeling.** Use data stored in Excel or any Excel data source. You can create and save connections to re-use data sources, repeat experiments, or re-train models.
- **Profile, sample, and prepare.** Many experienced data miners say that as much as 70-90 percent of a data mining project is spent on data preparation. The add-ins can make this task go faster, by providing visualizations in Excel and wizards that help you with these common tasks:
  - Profile data and understand its distribution and characteristics.
  - Create training and testing sets through random sampling or oversampling.
  - Find outliers and remove or replace them.
  - Re-label data to improve the quality of analysis.
- **Analyze patterns through supervised or unsupervised learning.** Click through friendly wizards to perform some of the most popular data mining tasks, including clustering analysis, market basket analysis, and forecasting.

  Among the well-known machine learning algorithms included in the add-ins are Naïve Bayes, logistic regression, clustering, time series, and neural networks.

  If you are new to data mining, get help building prediction queries from the **Query** wizard.

  Advanced users can build custom DMX queries with the drag-and-drop **Advanced Query Editor**, or automate predictions using Excel VBA.
- **Document and manage.** After you've created a data set and built some models, document your work and your insights by generating a statistical summary of the data and model parameters.
- **Explore and visualize.** Data mining is not an activity that can be fully automated - you need to explore and understand your results to take meaningful action. The add-ins help you with exploration by providing interactive viewers in Excel, Visio templates that let you customize model diagrams, and the ability to export charts and tables to Excel for additional filtering or modification.
- **Deploy and integrate.** When you've created a useful model, put your model into production, by using the management tools to export the model from your experimental server to another instance of Analysis Services.

  You can also leave the model on the server where you created it, but refresh the training data and run predictions using Integration Services or DMX scripts.

Power users will appreciate the **Trace** functionality, that lets you see the XMLA and DMX statements sent to the server.

Getting Started

For more information, see What's Included in the Data Mining Add-Ins for Office

Support and Requirements

The SQL Server Data Mining Add-Ins for Office is a free download. You must have one of the following versions of Office already installed to use these tools:

- Office 2010, 32-bit or 64-bit version
- Office 2013, 32-bit or 64-bit version

> **WARNING**
> Be sure to download the version of the add-ins that matches your version of Excel.

The Data Mining Add-ins requires a connection to one of the following editions of SQL Server Analysis Services:

- Enterprise
- Business Intelligence
- Standard

Depending on the edition of SQL Server Analysis Services that you connect to, some of the advanced algorithms might not be available. For information, see Features Supported by the Editions of SQL Server 2016.

For additional help with installation, see this page on the Download Center:
http://www.microsoft.com/download/details.aspx?id=29061

Data Mining Stored Procedures (Analysis Services - Data Mining)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Beginning in SQL Server 2005 (9.x), Analysis Services supports stored procedures that can be written in any managed language. The managed languages that are supported include Visual Basic .NET, C#, and managed C++. In SQL Server Management Studio, you can call the stored procedures directly by using the **CALL** statement, or as part of a Data Mining Extensions (DMX) query.

For more information about calling Analysis Services stored procedures, see Calling Stored Procedures.

For general information about programmability, see Data Mining Programming.

For additional information about how to program data mining objects, see the article, "SQL Server Data Mining Programmability", in the MSDN library.

> **NOTE**
> When you query mining models, especially when you test new data mining solutions, you might find it convenient to call the system stored procedures that are used internally by the data mining engine. You can view the names of these system stored procedures by using SQL Server Profiler to create a trace on the Analysis Services server, and then creating, browsing, and querying the data mining models. However, Microsoft does not guarantee the compatibility of system stored procedures between versions, and you should never use calls to the system stored procedures in a production system. Instead, for compatibility, you should create your own queries by using DMX or XML/A.

In This Section

- SystemGetCrossValidationResults (Analysis Services - Data Mining)
- SystemGetClusterCrossValidationResults (Analysis Services - Data Mining)
- SystemGetAccuracyResults (Analysis Services - Data Mining)
- SystemGetClusterAccuracyResults (Analysis Services - Data Mining)

See Also

Cross-Validation (Analysis Services - Data Mining)
Cross-Validation Tab (Mining Accuracy Chart View)
Calling a Stored Procedure

SystemGetCrossValidationResults (Analysis Services - Data Mining)

5/16/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Partitions the mining structure into the specified number of cross-sections, trains a model for each partition, and then returns accuracy metrics for each partition.

> **NOTE**
> This stored procedure cannot be used to cross-validate clustering models, or models that are built by using the Microsoft Time Series algorithm or the Microsoft Sequence Clustering algorithm. To cross-validate clustering models, you can use the separate stored procedure, SystemGetClusterCrossValidationResults (Analysis Services - Data Mining).

Syntax

```
SystemGetCrossValidationResults(
<mining structure>
[, <mining model list>]
,<fold count>
,<max cases>
,<target attribute>
[,<target state>]
[,<target threshold>]
[,<test list>])
```

Arguments

mining structure
Name of a mining structure in the current database.

(required)

mining model list
Comma-separated list of mining models to validate.

If a model name contains any characters that are not valid in the name of an identifier, the name must be enclosed in brackets.

If a list of mining models is not specified, cross-validation is performed against all models that are associated with the specified structure and that contain a predictable attribute.

> **NOTE**
> To cross-validate clustering models, you must use a separate stored procedure, SystemGetClusterCrossValidationResults (Analysis Services - Data Mining).

(optional)

fold count
Integer that specifies the number of partitions into which to separate the data set. The minimum value is 2. The maximum number of folds is **maximum integer** or the number of cases, whichever is lower.

Each partition will contain roughly this number of cases: *max cases/fold count*.

There is no default value.

> **NOTE**
> The number of folds greatly affects the time that is required to perform cross-validation. If you select a number that is too high, the query might run for a very long time, and in some cases the server can become unresponsive or time out.

(required)

max cases
Integer that specifies the maximum number of cases that can be tested across all folds.

A value of 0 indicates that all the cases in the data source will be used.

If you specify a value that is greater than the actual number of cases in the data set, all cases in the data source will be used.

There is no default value.

(required)

target attribute
String that contains the name of the predictable attribute. A predictable attribute can be a column, nested table column, or nested table key column of a mining model.

> **NOTE**
> The existence of the target attribute is validated only at run time.

(required)

target state
Formula that specifies the value to predict. If a target value is specified, metrics are collected for the specified value only.

If a value is not specified or is **null**, the metrics are computed for the most probable state for each prediction.

The default is **null**.

An error is raised during validation if the specified value is not valid for the specified attribute, or if the formula is not the correct type for the specified attribute.

(optional)

target threshold
Double greater than 0 and less than 1. Indicates the minimum probability score that must be obtained for the prediction of the specified target state to be counted as correct.

A prediction that has a probability less than or equal to this value is considered incorrect.

If no value is specified or is **null**, the most probable state is used, regardless of its probability score.

The default is **null**.

> **NOTE**
> Analysis Services will not raise an error if you set *state threshold* to 0.0, but you should never use this value. In effect, a threshold of 0.0 means that predictions with a 0 percent probability are counted as correct.

(optional)

test list
A string that specifies testing options.

Note This parameter is reserved for future use.

(optional)

Return Type

The rowset that is returned contains scores for each partition in each model.

The following table describes the columns in the rowset.

COLUMN NAME	DESCRIPTION
ModelName	The name of the model that was tested.
AttributeName	The name of the predictable column.
AttributeState	A specified target value in the predictable column. If this value is **null**, the most probable prediction was used. If this column contains a value, the accuracy of the model is assessed against this value only.
PartitionIndex	An 1-based index that identifies to which partition the results apply.
PartitionSize	An integer that indicates how many cases were included in each partition.
Test	Category of the test that was performed. For a description of the categories and the tests that are included in each category, see Measures in the Cross-Validation Report.
Measure	The name of the measure returned by the test. Measures for each model depend on the type of the predictable value. For a definition of each measure, see Cross-Validation (Analysis Services - Data Mining). For a list of measures returned for each predictable type, see Measures in the Cross-Validation Report.
Value	The value of the specified test measure.

Remarks

To return accuracy metrics for the complete data set, use SystemGetAccuracyResults (Analysis Services - Data Mining).

If the mining model has already been partitioned into folds, you can bypass processing and return only the results of cross-validation by using SystemGetAccuracyResults (Analysis Services - Data Mining).

Examples

The following example demonstrates how to partition a mining structure for cross-validation into two folds, and then test two mining models that are associated with the mining structure, `[v Target Mail]`.

Line three of the code lists the mining models that you want to test. If you do not specify the list, all non-clustering models associated with the structure are used. Line four of the code specifies the number of partitions. Because no value is specified for *max cases*, all cases in the mining structure are used and distributed evenly across the partitions.

Line five specifies the predictable attribute, Bike Buyer, and line six specifies the value to predict, 1 (meaning "yes, will buy").

The NULL value in line seven indicates that there is no minimum probability bar that must be met. Therefore, the first prediction that has a non-zero probability will be used in assessing accuracy.

```
CALL SystemGetCrossValidationResults(
[v Target Mail],
[Target Mail DT], [Target Mail NB],
2,
'Bike Buyer',
1,
NULL
)
```

Sample results:

MODELNAME	ATTRIBUTENAME	ATTRIBUTESTATE	PARTITIONINDEX	PARTITIONSIZE	TEST	MEASURE	VALUE
Target Mail DT	Bike Buyer	1	1	500	Classification	True Positive	144
Target Mail DT	Bike Buyer	1	1	500	Classification	False Positive	105
Target Mail DT	Bike Buyer	1	1	500	Classification	True Negative	186
Target Mail DT	Bike Buyer	1	1	500	Classification	False Negative	65
Target Mail DT	Bike Buyer	1	1	500	Likelihood	Log Score	-0.619042807138345
Target Mail DT	Bike Buyer	1	1	500	Likelihood	Lift	0.0740963734002671
Target Mail DT	Bike Buyer	1	1	500	Likelihood	Root Mean Square Error	0.346946279977653

MODELNAME	ATTRIBUTENAME	ATTRIBUTESTATE	PARTITIONINDEX	PARTITIONSIZE	TEST	MEASURE	VALUE
Target Mail DT	Bike Buyer	1	2	500	Classification	True Positive	162
Target Mail DT	Bike Buyer	1	2	500	Classification	False Positive	86
Target Mail DT	Bike Buyer	1	2	500	Classification	True Negative	165
Target Mail DT	Bike Buyer	1	2	500	Classification	False Negative	87
Target Mail DT	Bike Buyer	1	2	500	Likelihood	Log Score	-0.654117781086519
Target Mail DT	Bike Buyer	1	2	500	Likelihood	Lift	0.038997399132084
Target Mail DT	Bike Buyer	1	2	500	Likelihood	Root Mean Square Error	0.342721344892651

Requirements

Cross-validation is available only in SQL Server Enterprise beginning with SQL Server 2008.

See Also

SystemGetCrossValidationResults
SystemGetAccuracyResults (Analysis Services - Data Mining)
SystemGetClusterCrossValidationResults (Analysis Services - Data Mining)
SystemGetClusterAccuracyResults (Analysis Services - Data Mining)

SystemGetClusterCrossValidationResults (Analysis Services - Data Mining)

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Partitions the mining structure into the specified number of cross-sections, trains a model for each partition, and then returns accuracy metrics for each partition.

Note This stored procedure can be used only with a mining structure that contains at least one clustering model. To cross-validate non-clustering models, you must use SystemGetCrossValidationResults (Analysis Services - Data Mining).

Syntax

```
SystemGetClusterCrossValidationResults(
<structure name>,
[,<mining model list>]
,<fold count>}
,<max cases>
<test list>])
```

Arguments

mining structure
Name of a mining structure in the current database.

(required)

mining model list
Comma-separated list of mining models to validate.

If a list of mining models is not specified, cross-validation is performed against all clustering models that are associated with the specified structure.

> **NOTE**
> To cross-validate models that are not clustering models, you must use a separate stored procedure, SystemGetCrossValidationResults (Analysis Services - Data Mining).

(optional)

fold count
Integer that specifies the number of partitions into which to separate the data set. The minimum value is 2. The maximum number of folds is **maximum integer** or the number of cases, whichever is lower.

Each partition will contain roughly this number of cases: *max cases/fold count*.

There is no default value.

> **NOTE**
> The number of folds greatly affects the time required to perform cross-validation. If you select a number that is too high, the query may run for a very long time, and in some cases the server may become unresponsive or time out.

(required)

max cases
Integer that specifies the maximum number of cases that can be tested.

A value of 0 indicates that all the cases in the data source will be used.

If you specify a number that is higher than the actual number of cases in the data set, all cases in the data source will be used.

(required)

test list
A string that specifies testing options.

Note This parameter is reserved for future use.

(optional)

Return Type

The Return Type table contains scores for each individual partition and aggregates for all models.

The following table describes the columns returned.

COLUMN NAME	DESCRIPTION
ModelName	The name of the model that was tested.
AttributeName	The name of the predictable column. For cluster models, always **null**.
AttributeState	A specified target value in the predictable column. For cluster models, always **null.**
PartitionIndex	An 1-based index that identifies which partition the results apply to.
PartitionSize	An integer that indicates how many cases were included in each partition.
Test	The type of test that was performed.
Measure	The name of the measure returned by the test. Measures for each model depend on the type of the predictable value. For a definition of each measure, see Cross-Validation (Analysis Services - Data Mining). For a list of measures returned for each predictable type, see Measures in the Cross-Validation Report.
Value	The value of the specified test measure.

Remarks

To return accuracy metrics for the entire data set, use SystemGetClusterAccuracyResults (Analysis Services - Data Mining).

Also, if the mining model has already been partitioned into folds, you can bypass processing and return only the results of cross-validation by using SystemGetClusterAccuracyResults (Analysis Services - Data Mining).

Examples

The following example demonstrates how to partition a mining structure into three folds, and then test two clustering models that are associated with the mining structure.

Line three of the code lists the specific mining models that you want to test. If you do not specify the list, all clustering models associated with the structure are used.

Line four of the code specifies the number of folds, and line five specifies the maximum number of cases to use.

Because these are clustering models, you do not need to specify a predictable attribute or value.

```
CALL SystemGetClusterCrossValidationResults(
[v Target Mail],
[Cluster 1], [Cluster 2],
3,
10000
)
```

Sample results:

MODELNAME	ATTRIBUTENAME	ATTRIBUTESTATE	PARTITIONINDEX	PARTITIONSIZE	TEST	MEASURE	VALUE
Cluster 1			1	3025	Clustering	Case Likelihood	0.930524511864121
Cluster 1			2	3025	Clustering	Case Likelihood	0.919184178430778
Cluster 1			3	3024	Clustering	Case Likelihood	0.929651120490248
Cluster 2			1	1289	Clustering	Case Likelihood	0.922789726933607
Cluster 2			2	1288	Clustering	Case Likelihood	0.934865535691068
Cluster 2			3	1288	Clustering	Case Likelihood	0.924724595688798

Requirements

Cross-validation is available only in SQL Server Enterprise beginning in SQL Server 2008.

See Also

SystemGetCrossValidationResults (Analysis Services - Data Mining)

SystemGetAccuracyResults (Analysis Services - Data Mining)
SystemGetClusterCrossValidationResults
SystemGetClusterAccuracyResults (Analysis Services - Data Mining)

SystemGetAccuracyResults (Analysis Services - Data Mining)

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Returns cross-validation accuracy metrics for a mining structure and all related models, excluding clustering models.

This stored procedure returns metrics for the whole data set as a single partition. To partition the dataset into cross-sections and return metrics for each partition, use SystemGetCrossValidationResults (Analysis Services - Data Mining).

> **NOTE**
> This stored procedure is not supported for models that are built by using the Microsoft Time Series algorithm or the Microsoft Sequence Clustering algorithm. Also, for clustering models, use the separate stored procedure, SystemGetClusterAccuracyResults (Analysis Services - Data Mining).

Syntax

```
SystemGetAccuracyResults(<mining structure>,
[,<mining model list>]
,<data set>
,<target attribute>
[,<target state>]
[,<target threshold>]
[,<test list>])
```

Arguments

mining structure
Name of a mining structure in the current database.

(Required)

model list
Comma-separated list of models to validate.

The default is **null**. This means that all applicable models are used. When the default is used, clustering models are automatically excluded from the list of candidates for processing.

(Optional)

data set
A integer value that indicates which partition in the mining structure is used for testing. The value is derived from a bitmask that represents the sum of the following values, where any single value is optional:

Training cases	0x0001
Test cases	0x0002
Model filter	0x0004

For a complete list of possible values, see the Remarks section of this topic.

(required)

target attribute
String that contains the name of a predictable object. A predictable object can be a column, nested table column, or nested table key column of a mining model.

(required)

target state
String that contains a specific value to predict.

If a value is specified, the metrics are collected for that specific state.

If no value is specified, or if null is specified, the metrics are computed for the most probable state for each prediction.

The default is **null**.

(optional)

target threshold
Number between 0.0 and 1 that specifies the minimum probability in which the prediction value is counted as correct.

The default is **null**, which means that all predictions are counted as correct.

(optional)

test list
A string that specifies testing options. This parameter is reserved for future use.

(optional)

Return Type

The rowset that is returned contains scores for each partition and aggregates for all models.

The following table lists the columns returned by **GetValidationResults**.

COLUMN NAME	DESCRIPTION
Model	The name of the model that was tested. **All** indicates that the result is an aggregate for all models.
AttributeName	The name of the predictable column.

COLUMN NAME	DESCRIPTION
AttributeState	A target value in the predictable column. If this column contains a value, metrics are collected for the specified state only. If this value is not specified, or is null, the metrics are computed for the most probable state for each prediction.
PartitionIndex	Denotes the partition to which the result applies. For this procedure, always 0.
PartitionCases	An integer that indicates the number of rows in the case set, based on the <*data set*> parameter.
Test	The type of test that was performed.
Measure	The name of the measure returned by the test. Measures for each model depend on the model type, and the type of the predictable value. For a list of measures returned for each predictable type, see Measures in the Cross-Validation Report. For a definition of each measure, see Cross-Validation (Analysis Services - Data Mining).
Value	The value for the specified measure.

Remarks

The following table provides examples of the values that you can use to specify the data in the mining structure that is used for cross-validation. If you want to use test cases for cross-validation, the mining structure must already contain a testing data set. For information about how to define a testing data set when you create a mining structure, see Training and Testing Data Sets.

INTEGER VALUE	DESCRIPTION
1	Only training cases are used.
2	Only test cases are used.
3	Both the training cases and testing cases are used.
4	Invalid combination.
5	Only training cases are used, and the model filter is applied.
6	Only test cases are used, and the model filter is applied.
7	Both the training and testing cases are used, and the model filter is applied.

For more information about the scenarios in which you would use cross-validation, see Testing and Validation

(Data Mining).

Examples

This example returns accuracy measures for a single decision tree model, `v Target Mail DT`, that is associated with the `vTargetMail` mining structure. The code on line four indicates that the results should be based on the testing cases, filtered for each model by the filter specific to that model. `[Bike Buyer]` specifies the column that is to be predicted, and the 1 on the following line indicates that the model is to be evaluated only for the specific value 1, meaning "Yes, will buy".

The final line of the code specifies that the state threshold value is 0.5. This means that predictions that have a probability greater than 50 percent should be counted as "good" predictions when calculating accuracy.

```
CALL SystemGetAccuracyResults (
[vTargetMail],
[vTargetMail DT],
6,
'Bike Buyer',
1,
0.5
)
```

Sample Results:

MODELNAME	ATTRIBUTENAME	ATTRIBUTESTATE	PARTITIONINDEX	PARTITIONSIZE	TEST	MEASURE	VALUE
v Target Mail DT	Bike Buyer	1	0	1638	Classification	True Positive	605
v Target Mail DT	Bike Buyer	1	0	1638	Classification	False Positive	177
v Target Mail DT	Bike Buyer	1	0	1638	Classification	True Negative	501
v Target Mail DT	Bike Buyer	1	0	1638	Classification	False Negative	355
v Target Mail DT	Bike Buyer	1	0	1638	Likelihood	Log Score	-0.598454638753028
v Target Mail DT	Bike Buyer	1	0	1638	Likelihood	Lift	0.0936717116894395
v Target Mail DT	Bike Buyer	1	0	1638	Likelihood	Root Mean Square Error	0.361630800104946

Requirements

Cross-validation is available only in SQL Server Enterprise beginning with SQL Server 2008.

See Also

SystemGetCrossValidationResults (Analysis Services - Data Mining)
SystemGetAccuracyResults
SystemGetClusterCrossValidationResults (Analysis Services - Data Mining)
SystemGetClusterAccuracyResults (Analysis Services - Data Mining)

SystemGetClusterAccuracyResults (Analysis Services - Data Mining)

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Returns cross-validation accuracy metrics for a mining structure and related clustering models.

This stored procedure returns metrics for the entire data set as a single partition. To partition the dataset into cross-sections and return metrics for each partition, use SystemGetClusterCrossValidationResults (Analysis Services - Data Mining).

> **NOTE**
> This stored procedure works only for clustering models. For non-clustering models, use SystemGetAccuracyResults (Analysis Services - Data Mining).

Syntax

```
SystemGetClusterAccuracyResults(
<mining structure>
[,<mining model list>]
,<data set>
,<test list>])
```

Arguments

mining structure
Name of a mining structure in the current database.

(Required)

mining model list
Comma-separated list of models to validate.

The default is **null**, meaning that all applicable models are used. When the default is used, non-clustering models are automatically excluded from the list of candidates for processing.

(Optional)

data set
An integer value that indicates which partition in the mining structure is to be used for testing. The value is derived from a bitmask that represents the sum of the following values, where any single value is optional:

Training cases	0x0001
Test cases	0x0002

Model filter	0x0004

For a complete list of possible values, see the Remarks section of this topic.

(Required)

test list
A string that specifies testing options. This parameter is reserved for future use.

(optional)

Return Type

A table that contains scores for each individual partition and aggregates for all models.

The following table lists the columns returned by **SystemGetClusterAccuracyResults**. To learn more about how to interpret the information returned by the stored procedure, see Measures in the Cross-Validation Report.

COLUMN NAME	DESCRIPTION
ModelName	The name of the model that was tested. **All** indicates that the result is an aggregate for all models.
AttributeName	Not applicable to clustering models.
AttributeState	Not applicable to clustering models.
PartitionIndex	A number that indicates the partition. For this stored procedure, the number is always 0.
PartitionCases	An integer that indicates how many cases have been tested.
Test	The type of test that was performed.
Measure	The name of the measure returned by the test. Measures for each model depend on the model type, and the type of the predictable value. For a list of measures returned for each predictable type, see Measures in the Cross-Validation Report. For a definition of each measure, see Cross-Validation (Analysis Services - Data Mining).
Value	A probability score that indicates the cluster case likelihood.

Remarks

The following table provides examples of the values that you can use to specify the data in the mining structure that is used for cross-validation. If you want to use test cases for cross-validation, the mining structure must already contain a testing data set. For information about how to define a testing data set when you create a mining structure, see Training and Testing Data Sets.

INTEGER VALUE	DESCRIPTION
1	Only training cases are used.
2	Only test cases are used.
3	Both the training cases and testing cases are used.
4	Invalid combination.
5	Only training cases are used, and the model filter is applied.
6	Only test cases are used, and the model filter is applied.
7	Both the training and testing cases are used, and the model filter is applied.

For more information about the scenarios in which you would use cross-validation, see Testing and Validation (Data Mining).

Examples

This example returns accuracy measures for two clustering models, named `Cluster 1` and `Cluster 2`, that are associated with the vTargetMail mining structure. The code on line four indicates that the results should be based on the testing cases alone, without using any filters that might be associated with each model.

```
CALL SystemGetClusterAccuracyResults (
[vTargetMail],
[Cluster 1], [Cluster 2],
2
)
```

Sample Results:

MODELNAME	ATTRIBUTENAME	ATTRIBUTESTATE	PARTITIONINDEX	PARTITIONSIZE	TEST	MEASURE	VALUE
Cluster 1			0	5545	Clustering	Case Likelihood	0.796514342249313
Cluster 2			0	5545	Clustering	Case Likelihood	0.732122471228572

Requirements

Cross-validation is available only in SQL Server Enterprise beginning in SQL Server 2008.

See Also

SystemGetCrossValidationResults (Analysis Services - Data Mining)
SystemGetAccuracyResults (Analysis Services - Data Mining)
SystemGetClusterCrossValidationResults (Analysis Services - Data Mining)
SystemClusterGetAccuracyResults

Data Mining Extensions (DMX) Reference

6/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data Mining Extensions (DMX) is a language that you can use to create and work with data mining models in Microsoft SQL Server Analysis Services. You can use DMX to create the structure of new data mining models, to train these models, and to browse, manage, and predict against them. DMX is composed of data definition language (DDL) statements, data manipulation language (DML) statements, and functions and operators.

Microsoft OLE DB for Data Mining Specification

The data mining features in Analysis Services are built to comply with the Microsoft OLE DB for Data Mining specification.

The Microsoft OLE DB for Data Mining specification defines the following:

- A structure to hold the information that defines a data mining model.
- A language for creating and working with data mining models.

The specification defines the basis of data mining as the data mining model virtual object. The data mining model object encapsulates all that is known about a particular mining model. The data mining model object is structured like an SQL table, with columns, data types, and meta information that describe the model. This structure lets you use the DMX language, which is an extension of SQL, to create and work with models.

For More Information: Mining Structures (Analysis Services - Data Mining)

DMX Statements

You can use DMX statements to create, process, delete, copy, browse, and predict against data mining models. There are two types of statements in DMX: data definition statements and data manipulation statements. You can use each type of statement to perform different kinds of tasks.

The following sections provide more information about working with DMX statements:

- Data Definition Statements
- Data Manipulation Statements
- Query Fundamentals

Data Definition Statements

Use data definition statements in DMX to create and define new mining structures and models, to import and export mining models and mining structures, and to drop existing models from a database. Data definition statements in DMX are part of the data definition language (DDL).

You can perform the following tasks with the data definition statements in DMX:

- Create a mining structure by using the CREATE MINING STRUCTURE statement, and add a mining model to the mining structure by using the ALTER MINING STRUCTURE statement.
- Create a mining model and associated mining structure simultaneously by using the CREATE MINING MODEL statement to build an empty data mining model object.

- Export a mining model and associated mining structure to a file by using the EXPORT statement. Import a mining model and associated mining structure from a file that is created by the EXPORT statement by using the IMPORT statement.
- Copy the structure of an existing mining model into a new model, and train it with the same data, by using the SELECT INTO statement.
- Completely remove a mining model from a database by using the DROP MINING MODEL statement. Completely remove a mining structure and all its associated mining models from the database by using the DROP MINING STRUCTURE statement.

To learn more about the data mining tasks that you can perform by using DMX statements, see Data Mining Extensions (DMX) Statement Reference.

Back to DMX Statements

Data Manipulation Statements

Use data manipulation statements in DMX to work with existing mining models, to browse the models and to create predictions against them. Data manipulation statements in DMX are part of the data manipulation language (DML).

You can perform the following tasks with the data manipulation statements in DMX:

- Train a mining model by using the INSERT INTO statement. This does not insert the actual source data into a data mining model object, but instead creates an abstraction that describes the mining model that the algorithm creates. The source query for an INSERT INTO statement is described in <source data query>.
- Extend the SELECT statement to browse the information that is calculated during model training and stored in the data mining model, such as statistics of the source data. Following are the clauses that you can include to extend the power of the SELECT statement:
  - SELECT DISTINCT FROM <model > (DMX)
  - SELECT FROM <model>.CONTENT (DMX)
  - SELECT FROM <model>.CASES (DMX)
  - SELECT FROM <model>.SAMPLE_CASES (DMX)
  - SELECT FROM <model>.DIMENSION_CONTENT (DMX)
- Create predictions that are based on an existing mining model by using the PREDICTION JOIN clause of the SELECT statement. The source query for a PREDICTION JOIN statement is described in <source data query>.
- Remove all the trained data from a model or a structure by using the DELETE (DMX) statement.

To learn more about the data mining tasks that you can perform by using DMX statements, see Data Mining Extensions (DMX) Statement Reference.

Back to DMX Statements

DMX Query Fundamentals

The SELECT statement is the basis for most DMX queries. Depending on the clauses that you use with such statements, you can browse, copy, or predict against mining models. The prediction query uses a form of SELECT to create predictions based on existing mining models. Functions extend your ability to browse and query the mining models beyond the intrinsic capabilities of the data mining model.

You can use DMX functions to obtain information that is discovered during the training of your models, and to calculate new information. You can use these functions for many purposes, including to return statistics that

describe the underlying data or the accuracy of a prediction, or to return an expanded explanation of a prediction.

For More Information: Understanding the DMX Select Statement, General Prediction Functions (DMX), Structure and Usage of DMX Prediction Queries, Data Mining Extensions (DMX) Function Reference

Back to DMX Statements

See Also

Data Mining Extensions (DMX) Function Reference
Data Mining Extensions (DMX) Operator Reference
Data Mining Extensions (DMX) Statement Reference
Data Mining Extensions (DMX) Syntax Conventions
Data Mining Extensions (DMX) Syntax Elements
General Prediction Functions (DMX)
Structure and Usage of DMX Prediction Queries
Understanding the DMX Select Statement

Power Pivot for SharePoint (SSAS)

8/13/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

An Analysis Services server in Power Pivot mode provides server hosting of Power Pivot data in a SharePoint farm. Power Pivot data is an analytical data model that you build using one of the following:

- The Power Pivot for Excel 2010 add-in
- Excel 2013
- Excel 2016

Applies to: Power Pivot for SharePoint 2016 | Power Pivot for SharePoint 2013

Server hosting of that data requires SharePoint, Excel Services, and an installation of Power Pivot for SharePoint. Data is loaded on Power Pivot for SharePoint instances where it can be refreshed at scheduled intervals using the Power Pivot data refresh capability that the server provides for Excel 2010 workbooks, that the Power Pivot provides for Excel 2013 and 2016 workbooks.

Power Pivot for SharePoint 2016

SQL Server 2017 Power Pivot for SharePoint supports Microsoft SharePoint 2016 and Office Online Server usage of Excel workbooks containing data models and Reporting Services Power View reports.

Excel, within Office Online Server includes data model functionality to enable interaction with a Power Pivot workbook in the browser. You do not need to deploy the Power Pivot for SharePoint 2016 add-in into the farm. You only need to install an Analysis Services server in Power Pivot mode and register the server with Office Online Server.

Deploying the Power Pivot for SharePoint 2016 add-in enables additional functionality and features in your SharePoint farm. The additional features include Power Pivot Gallery and Schedule Data Refresh.

Power Pivot for SharePoint 2013

SQL Server 2017 Power Pivot for SharePoint supports Microsoft SharePoint 2013 Excel Services usage of Excel workbooks containing data models and Reporting Services Power View reports.

Excel Services in SharePoint 2013 includes data model functionality to enable interaction with a Power Pivot workbook in the browser. You do not need to deploy the Power Pivot for SharePoint 2013 add-in into the farm. You only need to install an Analysis Services server in SharePoint mode and register the server within the Excel Services **Data Model** settings.

Deploying the Power Pivot for SharePoint 2013 add-in enables additional functionality and features in your SharePoint farm. The additional features include Power Pivot Gallery, Schedule Data Refresh, and the Power Pivot Management Dashboard.

In This Section

Power Pivot Server Administration and Configuration in Central Administration

Power Pivot Configuration using Windows PowerShell

Power Pivot Configuration Tools

Power Pivot Authentication and Authorization

Configure Power Pivot Health Rules

Power Pivot Management Dashboard and Usage Data

Power Pivot Gallery

Power Pivot Data Access

Power Pivot Data Refresh

Power Pivot Data Feeds

Power Pivot BI Semantic Model Connection (.bism)

In other sections

Additional topics

Upgrade Power Pivot for SharePoint

Install Analysis Services in Power Pivot Mode

PowerShell Reference for Power Pivot for SharePoint

Example License Topologies and Costs for SQL Server 2016 Business Intelligence

See Also

Power Pivot Planning and Deployment
Disaster Recovery for Power Pivot for SharePoint

Power Pivot Server Administration and Configuration in Central Administration

7/11/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Power Pivot server administration and configuration is performed by SharePoint service application administrators, using SharePoint Central Administration.

Power Pivot for SharePoint must be configured before it can be used. After you install Power Pivot for SharePoint using SQL Server Setup, you can configure it using any of the following approaches:

- Power Pivot Configuration Tool or Power Pivot for SharePoint 2013 Configuration tool
- SharePoint Central Administration
- PowerShell cmdlets

All three approaches deliver a fully configured server.

This section includes tasks for configuring the software using Central Administration. At a minimum, you must perform all three of the required configuration tasks noted in the list below.

> **IMPORTANT**
>
> For SharePoint 2010, SharePoint 2010 Service Pack 1 (SP1) must be installed before you can configure either Power Pivot for SharePoint, or a SharePoint farm that uses a SQL Server 2012 (11.x) database server. If you have not yet installed the service pack, do so now before you begin configuring the server.

Benefits of Configuring Power Pivot for SharePoint Using Central Administration

SharePoint Central Administration is the administrative application of a SharePoint farm. If you are farm administrator, you might prefer to use a familiar tool when adding a Power Pivot for SharePoint instance to your farm.

In contrast with the Power Pivot Configuration Tools or PowerShell cmdlets, Central Administration provides pages that fully specify all of the options you might set when configuring an application or server. Other approaches either condense the configuration workflow into a fewer number of steps, or require prior knowledge of how to configure a SharePoint server using PowerShell.

Related Content

Power Pivot Configuration using Windows PowerShell

Power Pivot Configuration Tools

Related Tasks

LINK	TYPE	TASK DESCRIPTION
Deploy Power Pivot Solutions to SharePoint	Required	This step installs the solution files that add program files and application pages to the farm and to site collections.
Create and Configure a Power Pivot Service Application in Central Administration	Required	This step provisions the Power Pivot System Service.
Activate Power Pivot Feature Integration for Site Collections in Central Administration	Required	This step turns on Power Pivot features at the site collection level.
Add MSOLAP.5 as a Trusted Data Provider in Excel Services	Required	This step adds the Analysis Services OLE DB provider as a trusted provider in Excel Services.
Power Pivot Data Refresh with SharePoint 2010	Recommended	Data refresh is optional but recommended. It allows you to schedule unattended updates to the Power Pivot data in published Excel workbooks.
Configure the Power Pivot Unattended Data Refresh Account (Power Pivot for SharePoint)	Recommended	This step provisions a special-purpose account that can be used to run data refresh jobs on the server.
Configure Usage Data Collection for (Power Pivot for SharePoint	Optional	Usage data collection is configured by default. You can use these steps to modify the default settings.
Configure Dedicated Data Refresh or Query-Only Processing (Power Pivot for SharePoint)	Optional	A Power Pivot instance can be dedicated to just data refresh jobs or queries. In addition, you can modify default settings for parallel data refresh jobs.
Configure Power Pivot Service Accounts	Optional	Explains how to update passwords or change service accounts.
Connect a Power Pivot Service Application to a SharePoint Web Application in Central Administration	Optional	Explains how to modify service associations.
Create a trusted location for Power Pivot sites in Central Administration	Optional	Explains how to add the Power Pivot Gallery as a trusted location.
Configure and View SharePoint Log Files and Diagnostic Logging (Power Pivot for SharePoint)	Optional	Event logging is configured by default. You can use these steps to modify the default settings.
Configure Power Pivot Health Rules	Optional	Server health rules are configured by default. You can use these steps to modify some of the default settings.

LINK	TYPE	TASK DESCRIPTION
Create and Customize Power Pivot Gallery	Optional	For installations that you are configuring manually, this procedure explains how to create a Power Pivot Gallery library that shows image thumbnails of the Power Pivot workbooks it contains.
Add a BI Semantic Model Connection Content Type to a Library (Power Pivot for SharePoint)	Optional	Explains how to extend a document library to support the creation of BI semantic model connection files.

See Also

Power Pivot for SharePoint 2010 Installation
Configuration Setting Reference (Power Pivot for SharePoint)
Disaster Recovery for Power Pivot for SharePoint

Deploy Power Pivot Solutions to SharePoint

11/28/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use the following instructions to manually deploy two solution packages that add Power Pivot features to a SharePoint Server 2010 environment. Deploying the solutions is a required step for configuring Power Pivot for SharePoint on a SharePoint 2010 server. To view the complete list of required steps, see Power Pivot Server Administration and Configuration in Central Administration.

Alternatively, you can use the Power Pivot Configuration Tool to deploy the solutions. Using the configuration tool is easier and more efficient for a single server installation, but you might want to use Central Administration and PowerShell if you prefer using a familiar tool or if you are configuring multiple features at the same time. For more information about using the configuration tool, see Power Pivot Configuration Tools.

Before deploying the solutions, you must first install Power Pivot for SharePoint using the SQL Server 2012 installation media. SQL Server Setup installs the solution packages that you are about to deploy.

This topic contains the following sections:

Prerequisite: Verify the Web Application uses Classic Mode Authentication

Step 1: Deploy the Farm Solution

Step 2: Deploy the Power Pivot Web Application Solution to Central Administration

Step 3: Deploy the Power Pivot Web Application Solution to Other Web Applications

Redeploy or retract the Solution

About the Power Pivot Solutions

Prerequisite: Verify the Web Application uses Classic Mode Authentication

Power Pivot for SharePoint is only supported for web applications that use Windows-classic mode authentication. To check whether the application uses Classic mode, run the following PowerShell cmdlet from the **SharePoint 2010 Management Shell**, replacing **http://<top-level site name>** with the name of your SharePoint site:

```
Get-spwebapplication http://<top-level site name> | format-list UseClaimsAuthentication
```

The return value should be **false**. If it is **true**, you cannot access Power Pivot data with this web application.

Step 1: Deploy the Farm Solution

This section shows you how to deploy solutions using PowerShell, but you can also use the Power Pivot Configuration Tool to complete this task. For more information, see Configure or Repair Power Pivot for SharePoint 2010 (Power Pivot Configuration Tool).

This task only needs to be performed once, after you install Power Pivot for SharePoint.

1. On a server that has an installation of Power Pivot for SharePoint, open a SharePoint 2010 Management Shell using the **Run as Administrator** option.

2. Run the following cmdlet to add the farm solution.

```
Add-SPSolution -LiteralPath "C:\Program Files\Microsoft SQL
Server\110\Tools\PowerPivotTools\ConfigurationTool\Resources\PowerPivotFarm.wsp"
```

   The cmdlet returns the name of the solution, its solution ID, and Deployed=False. In the next step, you will deploy the solution.

3. Run the next cmdlet to deploy the farm solution:

```
Install-SPSolution -Identity PowerPivotFarm.wsp -GACDeployment -Force
```

Step 2: Deploy the Power Pivot Web Application Solution to Central Administration

After you deploy the farm solution, you must deploy the Web application solution to Central Administration. This step adds the Power Pivot Management Dashboard to Central Administration.

1. Open a SharePoint 2010 Management Shell using the **Run as Administrator** option.

2. Run the following cmdlet to create a reference to Central Administration:

```
$centralAdmin = $(Get-SPWebApplication -IncludeCentralAdministration | Where {
$_.IsAdministrationWebApplication -eq $TRUE})
```

3. Run the following cmdlet to add the farm solution.

```
Add-SPSolution -LiteralPath "C:\Program Files\Microsoft SQL
Server\110\Tools\PowerPivotTools\ConfigurationTool\Resources\PowerPivotWebApp.wsp"
```

   The cmdlet returns the name of the solution, its solution ID, and Deployed=False. In the next step, you will deploy the solution.

4. Run the next cmdlet to install the web application solution to Central Administration.

```
Install-SPSolution -Identity PowerPivotWebApp.wsp -GACDeployment -Force -WebApplication $centralAdmin
```

Now that the web application solution is deployed to Central Administration, you can use Central Administration to complete all remaining configuration steps.

Step 3: Deploy the Power Pivot Web Application Solution to Other Web Applications

In the previous task, you deployed Powerpivotwebapp.wsp to Central Administration. In this section, you deploy the powerpivotwebapp.wsp on each existing web application that supports Power Pivot data access. If you add more web applications later, be sure to repeat this step for those additional web applications.

1. In Central Administration, in System Settings, click **Manage farm solutions**.

2. Click **powerpivotwebapp.wsp**.

3. Click **Deploy Solution**.

4. In **Deploy To?**, select the SharePoint web application for which you want to add Power Pivot feature support.
5. Click **OK**.
6. Repeat for other SharePoint web applications that will also support Power Pivot data access.

Redeploy or retract the Solution

Although SharePoint Central Administration provides solution retraction, you do not need to retract the powerpivotwebapp.wsp file unless you are systematically troubleshooting an installation or patch deployment problem.

1. In SharePoint 2010 Central Administration, in System Settings, click **Manage farm solutions**.
2. Click **Powerpivotwebapp.wsp**.
3. Click **Retract Solution**.

If you encounter server deployment issues that you trace back to the farm solution, you can redeploy it by running the **Repair** option in the Power Pivot Configuration Tool. Repair operations via the tool is preferred because it requires fewer steps on your part. For more information, see Configure or Repair Power Pivot for SharePoint 2010 (Power Pivot Configuration Tool).

If you still want to re-deploy all solutions, be sure to do so in this order:

1. Retract the Power Pivot Web application solution from all SharePoint web applications that use it.
2. Retract the Power Pivot farm solution.
3. Redeploy the Power Pivot farm solution.
4. Redeploy the Power Pivot Web application solution to all SharePoint web applications.

About the Power Pivot Solutions

Power Pivot for SharePoint uses two solution packages to deploy its application pages and program files to the farm and to individual web applications.

- The farm solution is global. It is deployed once and then becomes automatically available to any new Power Pivot for SharePoint servers that you add to the farm later.
- The web application solution is application-specific and must be deployed to each web application in the farm, including the Central Administration web application.

Each solution is deployed differently. The farm solution is deployed once, after the first Power Pivot for SharePoint instance is installed. To deploy the farm solution, you use the Power Pivot Configuration Tool or PowerShell cmdlets.

The web application solution is initially deployed to Central Administration, followed by subsequent solution deployments to any additional web applications that will support requests for Power Pivot data. To deploy the web application solution to Central Administration, you must use the Power Pivot Configuration Tool or PowerShell cmdlet. For all other web applications, you can deploy the web application solution manually using Central Administration or PowerShell.

SOLUTION	DESCRIPTION

SOLUTION	DESCRIPTION
Powerpivotfarm.wsp	Adds Microsoft.AnalysisServices.SharePoint.Integration.dll to the global assembly. Adds Microsoft.AnalysisServices.ChannelTransport.dll to the global assembly. Installs features and resources files, and registers content types. Adds library templates for Power Pivot Gallery and Data Feed libraries. Adds application pages for service application configuration, Power Pivot Management Dashboard, data refresh, and Power Pivot Gallery.
Powerpivotwebapp.wsp	Adds Microsoft.AnalysisServices.SharePoint.Integration.dll resources files to the web server extensions folder on the Web front-end. Adds Power Pivot Web service to the Web-front end. Adds thumbnail image generation for Power Pivot Gallery.

See Also

Upgrade Power Pivot for SharePoint
Power Pivot Server Administration and Configuration in Central Administration
Power Pivot Configuration using Windows PowerShell

Create and Configure Power Pivot Service Application in CA

7/11/2018 • 11 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A Power Pivot service application is a shared service instance of the Power Pivot System Service. Each service application has its own application identity, configuration settings, properties, and internal data storage.

This topic contains the following sections:

Determine whether to create a new Power Pivot Service Application

Create a Power Pivot Service Application

Configure the Power Pivot Service Application

Assign a Power Pivot Service Application to a Web Application

Edit Service Application Properties

Determine whether to create a new Power Pivot Service Application

A Power Pivot for SharePoint installation must have at least one Power Pivot service application in the farm. A service application is created automatically if you used the Power Pivot Configuration Tool to configure the server. Otherwise, you must create a Power Pivot service application manually in Central Administration.

Creating a service application makes the service available and generates the service application database. Depending on options you select when creating the service application, a Power Pivot service connection is added to the default service connection group. All SharePoint Web applications that subscribe to the default service connection group will get immediate access to the Power Pivot service application automatically.

You can create multiple Power Pivot service applications. Although one service application is sufficient for most deployment scenarios, you might consider creating an additional Power Pivot service application if your business requirements include the following:

- Using a different unattended Power Pivot data refresh account for each application.
- Using different timeouts, usage history, and thresholds for query response reporting.
- Delegating service administration to different people. An administrator will see data refresh history, usage data, and other properties only for the application he or she is administering. If you are required to isolate SharePoint web applications (for example, if your company is a hosting service that must guarantee data isolation for the SharePoint Web applications that belong to different customers), creating separate Power Pivot service applications can help meet isolation requirements by ensuring each service administrator sees only the configuration settings and properties for the application he or she manages.

Creating additional service application introduces new requirements for managing service associations. Namely, it will require that you create and use custom service association lists for each additional service application that you create.

If you do not have a specific reason for creating additional Power Pivot service application, you should use a single service application for all of the Web applications in the farm.

Create a Power Pivot Service Application

1. In Central Administration, in Application Management, click **Manage service applications**.
2. In the **Service Applications** ribbon, click **New**.
3. Select **SQL Server Power Pivot Service Application**. If it does not appear in the list, Power Pivot for SharePoint is not installed or configured correctly.
4. In the **Create New Power Pivot Service Application** page, enter a name for the application. The default is PowerPivotServiceApplication<number>. If you are creating multiple Power Pivot service applications, a descriptive name will help other administrators understand how the application is used.
5. In Application Pool, create a new application pool for the application (recommended). Select or create a managed account for the application pool. Be sure to specify a domain user account. A domain user account enables the use of SharePoint's managed account feature, which lets you update passwords and account information in one place. Domain accounts are also required if you plan to scale out the deployment to include additional service instances that will run under the same identity.
6. In **Database Server**, the default value is the SQL Server Database Engine instance that hosts the farm configuration databases. You can use this server or choose a different SQL Server.
7. In **Database Name**, the default value is PowerPivotServiceApplication1_<guid>. You must create a unique database for each Power Pivot service application. The default database name corresponds to the default name of the service application. If you entered a unique service application name, follow a similar naming convention for your database name so that you can manage them together.
8. In **Database Authentication**, the default is Windows Authentication. If you choose **SQL Authentication**, refer to the SharePoint administrator guide for best practices on how to use this authentication type in a SharePoint deployment.
9. Optionally, select the checkbox for **Add the proxy for this Power Pivot Service Application to the farm's default proxy group.** This adds the service application connection to the default service connection group.

   You must select this checkbox if you are creating your first Power Pivot service application. There must be one Power Pivot service application in the default connection group in order for Power Pivot Management Dashboard to work properly.

   Do not add the Power Pivot service application to the default connection group if one already exists. Adding multiple entries of the same service application type is not a supported configuration and might cause errors. If you are creating additional service applications, leave them out of the default connection group and add them to custom lists instead.

   For more information about service associations, see Connect a Power Pivot Service Application to a SharePoint Web Application in Central Administration.
10. Click **OK.** The service will appear alongside other managed services in the farm's service application list.

Configure Power Pivot Service Application

A Power Pivot service application is created using a default configuration. The default settings are recommended for most scenarios. Change them only if you encounter slow response time or dropped connections, or if you are varying Power Pivot service configuration for specific SharePoint Web applications.

1. In Central Administration, in Application Management, click **Manage service applications**.

   In the list of service applications, you should see the service application you just created and named. The

default is **PowerPivotServiceApplication1**.

2. Click the Power Pivot service application. This opens the Power Pivot Management Dashboard.

3. In the **Actions** list on the top right corner of the dashboard, click **Configure service application settings**.

4. In **Database Load Timeout**, increase or decrease the value to change how long the Power Pivot service waits for a response from the SQL Server Analysis Services (Power Pivot) instance to which it forwarded a load data request. Because very large datasets take time to move over the wire, you must allow sufficient time for the Power Pivot service instance to retrieve the Excel workbook and move the Power Pivot data to an Analysis Services instance for query processing. Because Power Pivot data can be unusually large, the default value is 30 minutes.

5. In **Connection Pool Timeout**, increase or decrease the value to change how many minutes an idle data connection will remain open. The default value is 30 minutes. During this period, the Power Pivot service will reuse an idle data connection for read-only requests from the same SharePoint user for the same Power Pivot data. If no further requests are received for that data during the period specified, the connection is removed from the pool. Valid values are 1 to 3600 seconds. For more information about connection pools, see Configuration Setting Reference (Power Pivot for SharePoint).

6. In **Maximum User Connection Pool Size**, increase or decrease the value to change the maximum number of idle connections the Power Pivot service will create in individual connection pools for each SharePoint user, Power Pivot dataset, and version combinations.

   The default value is 1000 idle connections. Valid values are -1 (unlimited), 0 (disables user connection pooling), or 1 to 10000.

   These connection pools enable the service to more efficiently support ongoing connections to the same read-only data by the same user. If you disable connection pooling, every connection will be created anew.

   Note that changing the limit on connection pool size, including setting it to 0, will not result in dropped connections. Connection pools exist to reduce wait times when connecting to data. The Power Pivot service will never deny a connection based on connection pool settings.

7. In **Maximum Administrative Connection Pool Size**, increase or decrease the value to change the number of open connections in a connection pool created for a Power Pivot service connection to Analysis Services. Each Power Pivot service instance opens a separate administrative connection to the Analysis Services instance on the same computer. Power Pivot service creates a separate pool to reuse administrative connections for the purpose of checking for idle connections and monitoring server health. The default value is 200 connections. Valid values are -1 (unlimited), 0 (disables administrative connection pooling), or 1 to 10000. If you select 0, every connection will be created anew.

8. In **Allocation Method**, you can specify the load balancing schema that the Power Pivot System Service uses to select a specific Power Pivot service application for load balancing an initial request. The default is **Health Based**, which allocates requests based on server state, as measured by available memory and processor utilization. Alternatively, you can choose **Round Robin** to allocate requests to servers in the same repeating order, regardless of whether a server is busy or idle.

9. In Data Refresh, in **Business Hours**, you can specify a range of hours that defines a business day. Data refresh schedules can run after the close of a business day to pick up transactional data that was generated during normal business hours.

10. In **Power Pivot Unattended Data Refresh Account**, you can specify a predefined Secure Store Service target application that stores a predefined account for running Power Pivot data refresh jobs. Be sure to specify the target application name, and not the ID. The target application for unattended data refresh is created automatically if you used the New Server option in SQL Server Setup to install Power Pivot for SharePoint. Otherwise, you must create the target application manually. For instructions on how to

configure the account, see Configure the Power Pivot Unattended Data Refresh Account (Power Pivot for SharePoint).

11. In **Allow users to enter custom Windows credentials**, you can select or clear the checkbox to specify whether schedule owners can enter arbitrary Windows credentials to run a data refresh schedule. If you select this checkbox, Power Pivot service application will create and manage a target application each set of stored credentials. For more information, see Configure Stored Credentials for Power Pivot Data Refresh (Power Pivot for SharePoint).

12. In **Maximum Processing History Length**, you can specify how long to retain a historical record of data refresh processing. This information appears in data refresh history pages that are kept for each workbook that uses data refresh. It also appears in the Power Pivot Management Dashboard.

13. In Usage Data Collection, in **Query Reporting Interval**, specify an interval of time for reporting query statistics. Query statistics are reported as a single event to minimize server-to-server communication.

14. In Usage Data History, specify how long to keep a historical record of usage data. Usage information appears in the Power Pivot Management Dashboard. The reports will be less effective if you specify too low a value for usage data history.

15. In Usage Data Collection, in each query response threshold, specify an upper limit that determines where one category stops and another begins. These categories establish a baseline against which query behavior is measured. You can use these categories to monitor trends in query response times for your system. This information appears in the Power Pivot Management Dashboard.

16. Click **OK** to save your changes.

    Changes to the load timeout or allocation method are only applied to new incoming requests. Requests that are already in progress are subject to the values that were in effect when the request was received.

Assign a Power Pivot Service Application to a Web Application

After you configure a Power Pivot service application, you can assign it to a Web application by adding it to the service application connection list for that Web application. There are two ways to do this:

- Add it to the **Default** connection group. The *default connection group* is a collection of service application connections that are available to any Web application that references it. You must add a Power Pivot service application to this list.
- Create a **Custom** connection list for a specific Web application. If you created multiple Power Pivot service applications, you can choose which one to use by selecting it in a custom list.

The default connection group will accept more than one service application of the same type. Be aware, however, that adding more than one Power Pivot service applications to this list is not a supported configuration.

1. In Central Administration, in **Application Management**, click **Manage web applications**.
2. Select the application for which you want to assign a connection (for example, SharePoint -80).
3. Click **Service Connections**.
4. In **Edit the following group of associations**, select **default** or **[custom]**.
5. For **[custom]**, select the checkbox next to each service application connection you want to use. If you have multiple Power Pivot service applications (indicated by Type set to **Power Pivot Service Application Proxy**), be sure to choose just one.
6. Click **OK**.

Edit Service Application Properties

Use the following instructions to re-open the property page that specifies the service application name, application pool, database settings, and service associations.

1. In Central Administration, in Application Management, click **Manage service applications**.
2. Select, but do not click, the Power Pivot service application. You can click the type name to select the entire row.
3. Click **Properties** on the ribbon.

See Also

Power Pivot Server Administration and Configuration in Central Administration

Connect Power Pivot Service App to SharePoint Web App in CA

10/25/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A Power Pivot service application can be used by any number of SharePoint Web applications in the farm. To make a Power Pivot service application available, you add it to a service association list.

> **IMPORTANT**
> You must have one Power Pivot service application in the default group to ensure that Power Pivot Management Dashboard works properly. Do not add more than one Power Pivot service application to the default group. Adding multiple entries of the same service application type is not a supported configuration and might cause errors. If you are creating additional service applications, add them to custom lists.

This topic contains the following sections:

Add Power Pivot Services Application to the Default Group

Add Power Pivot Services Application a Custom Service Association List

Add Services Application to the Default Group

A service association list is a list of shared services that provide resources to other SharePoint Web applications in the farm. There is one default group of service associations for the farm.

To be on the list, a Power Pivot service application can either be added when you create the application or afterwards by using the following steps.

1. In Central Administration, in **Application Management**, click **Configure service application associations**.
2. In Application Proxy Group, click **default**.
3. Select the checkbox next to the Power Pivot service application (indicated by type name **Power Pivot Service Application Proxy**). If you have more than one Power Pivot service application, choose just one.
4. Click **OK**.

Add Power Pivot Services Application a Custom Service Association List

The default group can be replaced by a custom list. A custom list is created specifically for a single SharePoint Web application. It overrides the default group and replaces it with only those service associations that a farm or service administrator specifies. If you created multiple Power Pivot service applications, you must use a custom list to specify which one to use. A custom list cannot be reused by other Web applications. It applies only to the Web application for which it was created.

1. In Central Administration, in **Application Management**, click **Manage web applications**.
2. Select the application (for example, SharePoint -80).

3. In Web Applications, in Manage, click **Service Connections**.
4. In **Edit the following group of connections**, select **[custom]**.
5. Select the checkbox next to each service application connection you want to use. If you have multiple Power Pivot service applications (indicated by Type set to **Power Pivot Service Application Proxy**), be sure to choose only one.
6. Click **OK**.

See Also

Create and Configure a Power Pivot Service Application in Central Administration
Initial Configuration (Power Pivot for SharePoint)

Activate Power Pivot Integration for Site Collections in CA

10/25/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Activating Power Pivot feature integration for specific site collections is required if you used the Existing Farm installation option to install SQL Server Power Pivot for SharePoint. If you installed Power Pivot for SharePoint using the New Server option, you can skip this task because SQL Server Setup already activated Power Pivot feature integration for the root site collection when it configured your deployment.

Feature activation at the site collection level is necessary to make application pages and templates available to your sites, including configuration pages for scheduled data refresh and application pages for Power Pivot Gallery and Data Feed libraries.

You must activate Power Pivot integration for each site collection that supports Power Pivot query processing.

Prerequisites

You must be a site collection administrator.

Activate Power Pivot Features

1. On a SharePoint site, click **Site Actions**.

   By default, SharePoint web applications are accessed through port 80. This means that you can often access a SharePoint site by entering http://<computer name> to open the root site collection.

2. Click **Site Settings**.

3. In Site Collection Administration, click **Site collection features**.

4. Scroll down the page until you find **Power Pivot Integration Site Collection Feature**.

5. Click **Activate**.

6. Repeat for additional site collections by opening each site and clicking **Site Actions**.

See Also

Power Pivot Server Administration and Configuration in Central Administration
Initial Configuration (Power Pivot for SharePoint)
Power Pivot for SharePoint 2010 Installation

Add MSOLAP.5 as a Trusted Data Provider in Excel Services

7/11/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

MSOLAP.5 refers to the Analysis Services OLE DB provider for SQL Server 2012. Excel Services must trust this provider before it will make the connection request that results in the availability of Power Pivot data on a server.

If you configured Power Pivot for SharePoint using the Power Pivot Configuration Tool, MSOLAP.5 might already be a trusted provider because the tool includes an action that satisfies this requirement. However, if you are using PowerShell, Central Administration, or if you excluded the trusted provider action in the configuration tool, the provider might be missing, which case you should add it now as part of configuring the farm for Power Pivot data access.

You only need to perform this step once for each Excel Services service application.

Each physical server that handles a Power Pivot data request, such as a Power Pivot for SharePoint server or an Excel Services server, must have the OLE DB provider installed on the computer. A Power Pivot for SharePoint installation always includes the OLE DB provider, but if Excel Services is running on a computer that does not have Power Pivot for SharePoint, you must install the provider manually. For more information, see Install the Analysis Services OLE DB Provider on SharePoint Servers.

Add a trusted provider to Excel Services

1. In Central Administration, click **Manage service applications**, and then click the Excel Services service application.
2. Click **Trusted Data Providers**.
3. Verify that MSOLAP.5 appears in the list. Depending on how you configured Power Pivot for SharePoint, MSOLAP.5 might already be trusted.
4. If it is not listed, click **Add Trusted Data Provider**.
5. In Provider ID, type **MSOLAP.5**.
6. For Provider Type, ensure that OLE DB is selected.
7. In Provider Description, type **Microsoft OLE DB Provider for OLAP Services 11.0**.

Create a trusted location for Power Pivot sites in Central Administration

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Excel Services lets you specify which locations are valid repositories for workbooks that you open on a SharePoint server. These locations are called 'trusted locations', and you can use different configuration settings for each trusted location you create. For a deployment of Power Pivot for SharePoint, you might consider creating a trusted location for sites that contain Power Pivot workbooks so that you can apply the settings that work best for Power Pivot data access, while preserving default settings for the rest of the farm.

Prerequisites

You must be a farm or service administrator to designate a URL as a trusted location.

You must know the URL address of the SharePoint site that contains the Power Pivot Gallery or other library that stores your workbooks. To get the address, open the site that contains the library, right-click **Power Pivot Gallery**, select **Properties**, and then copy the first part of the Address (URL) that contains the server name and site path.

Overview

An initial installation of Excel Services specifies 'http://' as its trusted location, which means that workbooks from any site in the farm can be opened on the server. If you require tighter control over which locations are considered trustworthy, you can create new trusted locations that map to specific sites in your farm, and then vary the settings and permissions for each one.

Creating a new trusted location for sites that host Power Pivot workbooks is especially useful if you want to preserve default values for the rest of the farm, while applying different settings that work best for Power Pivot data access. For example, a trusted location that is optimized for Power Pivot workbooks could have a maximum workbook size of 50 MB, while the rest of the farm uses the default value of 10MB.

Creating a trusted location is recommended if you are using Power Pivot Gallery libraries to preview published workbooks and you encounter data refresh warnings instead of the preview image you expect. Power Pivot Gallery renders thumbnail images of reports and workbooks using data and presentation information within the document. If Warn on Data Refresh is enabled for a trusted location, Power Pivot Gallery might not have sufficient permissions to perform the refresh, causing an error to appear instead of the thumbnail image. Adding a site that contains Power Pivot Gallery as a new trusted location can eliminate this problem.

Create a Trusted Location for Power Pivot Data Access

1. In Central Administration, in Application Management, click **Manage service applications**.
2. Click the Excel Services Service Application.
3. Click **Trusted File Locations**.
4. Click **Add Trusted File Location**.
5. Enter the URL of a site that contains a Power Pivot Gallery library.

6. In Location Type, select **Microsoft SharePoint Foundation**.

> **IMPORTANT**
>
> UNC and HTTP location types are not supported for Power Pivot data access.

7. Accept all of the default settings for properties in Session Management, Workbook Properties, and Calculation Behavior.

8. In Workbook Properties, set **Maximum Workbook Size** to **50**. This aligns the upper limit for workbook file size to the upper limit for file uploads to the parent web application. If your workbooks are larger than 50 megabytes, you must further increase the file size limit. For more information, see Configure Maximum File Upload Size (Power Pivot for SharePoint).

9. In External Data, verify that Allow External Data is set to **Trusted data connection libraries and embedded**. This setting is required for Power Pivot data access in a workbook.

10. Also in External Data, for Warn on Refresh, clear the checkbox for **Refresh warning enabled**. Clearing the checkbox allows Power Pivot Gallery to bypass routine warning messages and show preview images of a workbook instead.

11. Click **OK**.

See Also

Power Pivot Gallery
Create and Customize Power Pivot Gallery
Use Power Pivot Gallery

Configure Power Pivot Service Accounts

11/28/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A Power Pivot for SharePointinstallation includes two services that support server operations. The **SQL Server Analysis Services (Power Pivot)** service is a Windows service that provides Power Pivot data processing and query support on an application server. The login account for this service is always specified during SQL Server Setup when you install Analysis Services in SharePoint integrated mode.

A second account must be specified for the Power Pivot service application, a shared web service that runs under an application pool identity in a SharePoint farm. This account is specified when you configure a Power Pivot for SharePointinstallation using either the Power Pivot Configuration tool or PowerShell.

Each service should run under a dedicated account so that you can audit connections or enable Kerberos authentication protocol in the farm.

Once the service accounts are set, any changes to either account must be made through SharePoint Central Administration. If you use alternative tools (such as the Services console application, IIS Manager, or SQL Server Configuration Manager), permissions will not be updated for database access in the farm or for local file access on the physical server.

This topic contains the following sections:

Update an expired password for SQL Server Analysis Services (Power Pivot) instance

Update an expired password for the Power Pivot Service Application

Change the account under which each service runs

Create or change the application pool for a Power Pivot service application

Account Requirements and Permissions

Troubleshooting: Grant Administrative Permissions Manually

Troubleshooting: Resolve HTTP 503 Errors Due to Expired Passwords for Central Administration or the SharePoint Foundation Web Application Service

Update an expired password for SQL Server Analysis Services (Power Pivot) instance

1. Point to Start, click **Administrative Tools**, and then click **Services**. Double-click **SQL Server Analysis Services (Power Pivot)**. Click **Log On**, and then enter the new password for the account.
2. In Central Administration, in the Security section, click **Configure managed accounts**.
3. Click **Edit** to change a specific account.
4. Select **Change password now**.
5. Select **Set account password to new value**. All services that run under the managed account will use the updated credentials.

Update an expired password for the Power Pivot Service Application

1. In Central Administration, in the Security section, click **Configure managed accounts**.
2. Click **Edit** to change a specific account.
3. Select **Change password now**.
4. Select **Set account password to new value**. All services that run under the managed account will use the updated credentials.

Change the account under which each service runs

1. In Central Administration, in the Security section, click **Configure service accounts**.
2. Select **Windows Service - SQL Server Analysis Services** to change the Analysis Services service account.
3. In **Select an account for this service**, choose an existing managed account or create a new one. The account must be a domain user account.
4. Select **Service Application Pool - SharePoint Web Services System** to change the application pool identity of the default Power Pivot service application. Depending on how your installation was configured, the service might be running under an existing service application pool created for SharePoint services. By default, the Power Pivot Configuration Tool registers the service as **Default Power Pivot Service Application (Power Pivot Service Application)**.

   If the service was configured manually by a SharePoint administrator, the service most likely has its own service application pool.
5. In **Select an account for this service**, choose an existing managed account or create a new one. The account must be a domain user account.
6. Click **OK**.

Create or change the application pool for a Power Pivot service application

1. In Central Administration, in Application Management, click **Manage service applications**.
2. Select, but do not click, the Power Pivot service application. Clicking on the application name opens the Power Pivot Management Dashboard, which does not provide a link to the properties page that specifies the application pool. You can click the empty white space in the row, or click on type name, to select the Power Pivot service application.
3. Click **Properties** on the ribbon.
4. Select **Create a new application pool**. Provide a name for the application pool and specify a managed account for its identity.

Account Requirements and Permissions

When planning a Power Pivot for SharePoint deployment, you must plan for the following service accounts.

- Analysis Services service account. Analysis Services processes Power Pivot queries and data refresh jobs in the farm. This account is always specified during SQL Server Setup when you install Power Pivot for SharePoint.
- Power Pivot Service Application Pool. A Power Pivot service application is associated with a Power Pivot System Service that provides SharePoint integration and infrastructure for Power Pivot query processing

in a farm. The application pool that you specify for a Power Pivot service application is the service identity of the Power Pivot System Service. You can have multiple Power Pivot service applications in a farm. Each one that you create should run in its own application pool.

Analysis Services Service Account

REQUIREMENT	DESCRIPTION
Provisioning requirement	This account must be specified during SQL Server Setup using the **Analysis Services - Configuration page** in the installation wizard (or the **ASSVCACCOUNT** installation parameter in a command line setup). You can modify the user name or password using Central Administration, PowerShell, or the Power Pivot Configuration Tool. Using other tools to change accounts and passwords is not supported.
Domain user account requirement	This account must be a Windows domain user account. Built-in machine accounts (such as Network Service or Local Service) are prohibited. SQL Server Setup enforces the domain user account requirement by blocking installation whenever a machine account is specified.
Permission requirements	This account must be a member of the SQLServerMSASUser$<server>$PowerPivot security group and the WSS_WPG security groups on the local computer. These permissions should be granted automatically. For more information on how to check or grant permissions, see Grant the Power Pivot Service Account Administrative Permissions Manually in this topic and Initial Configuration (Power Pivot for SharePoint).
Scale-out requirements	If you install multiple Power Pivot for SharePoint server instances in a farm, all of the Analysis Services server instances must run under the same domain user account. For example, if you configure the first Analysis Services service instance to run as Contoso\ssas-srv01, then all additional Analysis Services service instances that you deploy thereafter in the same farm must also run as Contoso\ssas-srv01 (or whatever the current account happens to be). Configuring all service instances to run under the same account allows the Power Pivot System service to allocate query processing or data refresh jobs to any Analysis Services service instance in the farm. In addition, it enables the use of the Managed Account feature in Central Administration for Analysis Services server instances. By using the same account for all Analysis Services service instances, you can change account or password once, and all service instances that use those credentials are updated automatically. SQL Server Setup enforces the same-account requirement. In a scale-out deployment where a SharePoint farm already has an instance of Power Pivot for SharePoint installed, Setup will block the new installation if the Analysis Services service account you specified is different from the one already in use in the farm.

Power Pivot Service Application Pool

REQUIREMENT	DESCRIPTION
Provisioning requirement	Power Pivot System Service is a shared resource on the farm that becomes available when you create a service application. The service application pool must be specified when the service application is created. It can be specified two ways: using the Power Pivot Configuration Tool, or through PowerShell commands. You might have configured the application pool identity to run under a unique account. But if you didn't, consider changing it now to run under a different account.
Domain user account requirement	The application pool identity must be a Windows domain user account. Built-in machine accounts (such as Network Service or Local Service) are prohibited.
Permission requirements	This account does not need local system Administrator permissions on the computer. However, this account must have Analysis Services system administrator permissions on the local Analysis Services service that is installed on the same computer. These permissions are granted automatically by SQL Server Setup or when you set or change the application pool identity in Central Administration. Administrative permissions are required for forwarding queries to the Analysis Services service. They are also required for monitoring health, closing inactive sessions, and listening for trace events. The account must have connect, read, and write permissions to the Power Pivot service application database. These permissions are granted automatically when the application is created, and updated automatically when you change accounts or passwords in Central Administration. The Power Pivot service application will check that a SharePoint user is authorized to view data before retrieving the file, but it does not impersonate the user. There are no permission requirements for impersonation.
Scale-out requirements	None.

Troubleshooting: Grant Administrative Permissions Manually

Administrative permissions will fail to update if the person updating the credentials is not a local administrator on the computer. If this occurs, you can grant administrative permissions manually. The easiest way to do this is to run the Power Pivot Configuration timer job in Central Administration. With this approach, you can reset permissions for all Power Pivot servers in the farm. Note that this approach will only work if the SharePoint timer job is running as both farm administrator and as local administrator on the computer.

1. In Monitoring, click **Review job definitions**.
2. Select **Power Pivot Configuration Timer Job**.
3. Click **Run Now**.

As a last resort, you can ensure correct permissions by granting Analysis Services System Administration permissions to the Power Pivot service application, and then specifically add the service application identity to the SQLServerMSASUser$<servername>$PowerPivot Windows security group. You must repeat these steps for

every Analysis Services instance that is integrated with the SharePoint farm.

You must be a local administrator to update Windows security groups.

1. In SQL Server Management Studio, connect to the Analysis Services instance as <server name>\POWERPIVOT.
2. Right-click the server name and select **Properties**.
3. Click **Security**.
4. Click **Add**.
5. Type the name of the account that is used for Power Pivot service application pool, and then click **OK**.
6. In Administrative Tools, click **Computer Management**.
7. Open **Local Users and Groups**.
8. Open **Groups**.
9. Double-click SQLServerMSASUser$<servername>$PowerPivot.
10. Click **Add**.
11. Type the name of the account that is used for Power Pivot service application pool, and then click **OK**.

Troubleshooting: Resolve HTTP 503 Errors Due to Expired Passwords for Central Administration or the SharePoint Foundation Web Application Service

If Central Administration service or the SharePoint Foundation Web Application service stops working due to an account reset or password expiration, you will encounter HTTP 503 "Service unavailable" error messages when attempting to open SharePoint Central Administration or a SharePoint site. Follow these steps to bring your server back online. Once Central Administration is available, you can proceed to update expired account information.

1. In Administrative tools, click **Internet Information Services Manager**.
2. For the identity of the site or Central Administration application pool is a domain user account that has an expired password, do the following:
   a. Right-click the application pool name and select **Advanced Settings**.
   b. Select **Identity** and click the ... button to open the Application Pool Identity dialog box.
   c. Click **Set**.
   d. Type the username and password.
3. Run IISRESET. To do this, open an Administrator command prompt and type **iisreset** at the command.
4. In SharePoint Central Administration, in Security, select **Configure managed accounts**.
5. Click **Edit** to update the information of the managed account that has the expired password.
6. Select **Change password now**.
7. Click **Use existing password**.
8. Type the password, and then click **OK**.

If Reporting Services is installed, use the Reporting Services Configuration Manager to update passwords for the report server and the connection to the report server database. For more information, see Configuration and Administration of a Report Server (Reporting Services SharePoint Mode).

See Also

Start or Stop a Power Pivot for SharePoint Server
Configure the Power Pivot Unattended Data Refresh Account (Power Pivot for SharePoint)

Configure and View SharePoint and Diagnostic Logging

11/27/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Power Pivot server operations, events, and messages are recorded in SharePoint log files. Use the information in this topic to configure logging levels and view log file information. You can control which Power Pivot server events are logged to the file. You can also control the severity of messages that are logged. For more information, see Configure Usage Data Collection for (Power Pivot for SharePoint.

In this topic:

- Log File Location
- Modify diagnostic logging levels for individual event categories
- How to View SharePoint Log Files

Log File Location

By default, SharePoint log files are saved to the following location:

```
C:\Program files\Common Files\Microsoft Shared\Web Server Extensions\14\LOGS
```

The LOGS folder contains log files (`.log`), data files (`.txt`), and usage files (`.usage`). The file naming convention for a SharePoint trace log is the server name followed by a date and time stamp. SharePoint trace logs are created at regular intervals and whenever there is an IISRESET. It is common to have many trace logs within a 24 hour period.

Modify diagnostic logging levels for individual event categories

By default, ULS logging of Power Pivot events is set at *Medium*. This setting is new for SQL Server 2012. If you upgraded a server from the prior release, the logging level might still be set at *Verbose*, the default level in SQL Server 2008 R2. If you are accustomed to reviewing the ULS logs for Power Pivot server usage information, you will notice that as a result of this change, there is less information about Power Pivot server operations.

Except for exceptions, which are of type *High*, all Power Pivot messages fall into the Verbose category. If you want log entries for routine server operations such as connections, requests, or query reporting, you must change the logging level to Verbose.

To modify diagnostic logging levels for individual event categories:

1. In SharePoint Central Administration, click **Monitoring**.
2. Click **Configure diagnostic logging**.
3. Scroll to **Power Pivot Service**.
4. Expand the category and select individual categories.

   Application Page Request specifies events triggered by the service application when locating a Analysis Services service for loading a Power Pivot data source and communicating with other servers in the farm.

   Request Processing specifies events triggered by query requests against a Power Pivot database that is

loaded on a server in the farm.

Usage specifies an event related to Power Pivot usage data collection.

5. In Least critical event to report to the event log, select **None** to disable event logging for the category, or **Error** to limit logging for just errors.
6. Select **Verbose** to log all events to the event log.
7. In Least critical event to report to the trace log, select **None** to disable tracing for the category, or **Error** to limit tracing for just errors.
8. Select **Verbose** to log all events to the trace log.
9. Click **OK**.

How to View SharePoint Log Files

Log files are text files. You can open them in any text editor. You can also use third-party log viewer applications.

Use a Text Editor

If you are using a text editor to troubleshoot a Power Pivot server error, the following tips can help you locate relevant information in the file:

- For errors related to publishing, viewing, or refreshing a Power Pivot workbook, search the log file for the name of the workbook.
- For errors that provide a correlation ID, copy the ID and use it as a search term in the log file.
- Search for error status "High" or "Exception". Search for "Power Pivot Service".
- If you know when the error occurred, use the date and time information to narrow the scope of entries you must scroll through.

Use a Log Viewer Application

Although you can use a text editor to view the trace logs individually, a log viewer application that enables you to view several log files together can be far more useful. Fortunately, there are several third-party log viewer applications available for download on the Codeplex site that can help you view multiple trace logs in a single workspace.

The following instructions include links to popular SharePoint ULS Log Viewers that you can download from Codeplex.

1. Go to SharePoint LogViewer or SharePoint ULS Log Viewer on the Codeplex site.
2. Click the **Downloads** tab.
3. Click the executable file. It will either be **SharePointLogViewer.exe** or **ULS Viewer 2.0 Binary**.
4. Read and accept the licensing terms.
5. Click **Save** to download the software to your local system.

   If you are downloading SharePoint ULS Log Viewer, save ULSViewer.zip to the Downloads folder.
6. In the Downloads folder, right-click ULSViewer.zip and select **Extract All**.
7. Specify a destination folder, and then click **Extract**.
8. In the folder that contains the extracted program files, double-click **ULSViewer** and run the application.
9. Click the folder icon at the top left corner of the application window.

10. Browse to \Program files\Common Files\Microsoft Shared\Web Server Extensions\14\Logs.
11. Select one or more trace logs. By default, trace log file names consist of the computer name plus date and time information. The file type is Text Document.
12. Click **Open** and wait for the files to load.
13. Use the built-in filters to select records based on severity, process, category, or a user-defined text file. You can also click column headings to change the sort order.

Entries for Power Pivot Services

The following table describes entries for Power Pivot server operations most likely to be found in a SharePoint log file.

PROCESS	AREA	CATEGORY	LEVEL	MESSAGE	DETAILS
w3wp.exe	Power Pivot Service	Usage	Verbose	There are no current request statistics, nothing to log.	At predefined intervals, the service reports query response statistics as a usage event to the usage data collection system. This message indicates there were no query statistics to report.
w3wp.exe	Power Pivot Service	Web front end	Verbose	Starting to locate an application server for data source=<*path*>	When it receives a connection request, the Power Pivot service identifies an available Analysis Services service to handle the request. If there is only one server in the farm, the local server accepts the request in all cases.
w3wp.exe	Power Pivot Service	Web front end	Verbose	Locating the application server succeeded.	The request was allocated to a Power Pivot service application.
w3wp.exe	Power Pivot Service	Web front end	Verbose	Redirecting request for the <*PowerPivotdata source*> to the Analysis Services service.	The request was forwarded to the Analysis Services service.

PROCESS	AREA	CATEGORY	LEVEL	MESSAGE	DETAILS
w3wp.exe	Power Pivot Service	Request Processing	Verbose	Redirecting request for UserName<*SharePoint user*> to the database	An impersonated connection to the Power Pivot data source was created on behalf of the SharePoint user.

See Also

Power Pivot Usage Data Collection
View and Read SQL Server Setup Log Files
Configure Usage Data Collection for (Power Pivot for SharePoint

Configure Disk Space Usage (Power Pivot for SharePoint)

11/27/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A Power Pivot for SharePoint deployment uses the disk space of the host computer to cache Power Pivot databases for faster reloads. Every Power Pivot database that is loaded in memory is first cached to disk so that it can be quickly reloaded later to service new requests. By default, Power Pivot for SharePoint uses all available disk space to cache its databases, but can modify this behavior by setting properties that limit how much disk space is used.

This topic explains how to set the limits on disk space usage.

This topic does not provide guidance for disk space management of Power Pivot databases (embedded in Excel workbooks) that are stored in content databases. Power Pivot databases can be large, thereby placing new demands on the storage capacity of the farm. In addition, if versioning is enabled, you might easily have multiple copies of the data in the same content database, further increasing the amount of disk space required for content storage. Although Power Pivot databases are an important consideration for disk management, they are not something that can be managed independently of other content that you store in a SharePoint farm. You will need to monitor disk space more closely as your business increases its use of Power Pivot workbooks. You can also track Power Pivot workbook activity in the Power Pivot Management Dashboard and remove workbooks that are no longer used.

How Power Pivot for SharePoint manages cached databases

To manage its cache, the Power Pivot System Service runs a background job at regular intervals to clean up unused or outdated databases that have newer versions in a content library. The purpose of the cleanup job is to unload inactive databases from memory and to delete unused, cached databases from the file system. The cleanup job is for long-term maintenance, ensuring that databases do not remain on the system indefinitely. On an active server, databases might be removed more often due to memory pressure on the server, database deletion in SharePoint, or newer versions of the database in a content library.

Although you cannot schedule the cleanup job, you can customize cache file management by setting server configuration properties that do the following:

- Set limits on the amount of disk space used by the cache.
- Specify how much data to delete when maximum disk space is reached.

How to check disk space usage

Power Pivot for SharePoint is installed on application servers in a SharePoint farm. Each installation has a data directory that includes a Backup folder. The Backup folder contains all of the data files that are cached by the Analysis Services instance on the computer. By default, the Backup folder can be found at the following path:

```
%drive%:\Program Files\Microsoft SQL Server\MSAS10_50.PowerPivot\OLAP\Backup\Sandboxes\<serviceApplicationName>
```

To check how much total disk space is used by the cache, you have to check the size of the Backup folder. There is no property in Central Administration that reports on current cache size.

The Backup folder provides common cache storage for any Power Pivot database that is loaded in memory on the

local computer. If you have multiple Power Pivot service applications defined in your farm, any one of them can use the local server to load and subsequently cache Power Pivot data. Both data loading and caching are Analysis Services server operations. As such, total disk space usage is managed at the Analysis Services instance level, on the Backup folder. Configuration settings that limit disk space usage are therefore set on the single SQL Server Analysis Services instance that runs on a SharePoint application server.

The cache contains only Power Pivot databases. Power Pivot databases are stored in multiple files under a single parent folder (the Backup folder). Because Power Pivot databases are meant to be used as internal data to an Excel workbook, database names are GUID based rather than descriptive. A GUID folder under **<serviceApplicationName>** is the parent folder of a Power Pivot database. As Power Pivot databases are loaded on the server, additional folders are created for each one.

Because Power Pivot data might be loaded on any Analysis Services instance in a farm, the same data might also be cached on multiple computers in the farm. This practice favors performance over disk space utilization, but the tradeoff is that users get faster access to data if it is already available on disk.

To immediately reduce disk space consumption, you can shut down the service and then delete a Power Pivot database from the Backup folder. Manually deleting files is a temporary measure, as a newer copy of the database will be cached again the next time the Power Pivot data is queried. Permanent solutions include limiting disk space used by the cache.

At the system level, you can create email alerts that notify you when disk space is low. Microsoft System Center includes an email alert feature. You can also use File Server Resource Manager, Task Scheduler, or PowerShell script to set up alerts. The following links provide useful information for setting up notifications about low disk space:

- What's New in File Server Resource Manager (http://technet.microsoft.com/library/hh831746.aspx).
- File Server Resource Manager Step-by-Step Guide for Windows Server 2008 R2 (http://go.microsoft.com/fwlink/?LinkID=204875).
- Setting low disk space alerts on Windows Server 2008 (http://go.microsoft.com/fwlink/?LinkID=204870).

How to limit the amount of disk space used for storing cached files

1. In Central Administration, in Application Management, click **Manage Services on Server**.
2. Click **SQL Server Analysis Services**.

   Notice that limits are set on the Analysis Services instance that runs on the physical server, and not at the service application level. All service applications that use the local Analysis Services instance are subject to the single maximum disk space limit that is set for that instance.
3. In Disk Usage, set a value (in gigabytes) for **Total disk space** to set an upper limit on the amount of space used for caching purposes. The default is 0, which allows Analysis Services to use all available disk space.
4. In Disk Usage, in the **Delete cached databases in last 'n' hours** setting, specify last-used criteria for emptying the cache when disk space is at the maximum limit.

   The default is 4 hours, meaning that all databases that have been inactive for 4 hours or more are deleted from the file system. Databases that are inactive but still in memory are unloaded and then deleted from the file system.

How to limit how long a database is kept in the cache

1. In Central Administration, in Application Management, click **Manage Service Applications**.
2. Click **Default Power Pivot Service Application** to open the management dashboard.

3. In Actions, click **Configure service application settings**.

4. In the Disk Cache section, you can specify how long an inactive database stays in memory to service new requests (by default, 48 hours) and how long it stays in the cache (by default, 120 hours).

   Keep Inactive Database in Memory specifies how long an inactive database stays in memory to service new requests for that data. An active database is always kept in memory as long as you are querying it, but after it is no longer active, the system will keep the database in memory for an additional time period in case there are more requests for that data.

   Because Power Pivot databases are cached first and then loaded in memory, database files consume disk space immediately. However, while the database is active (and for 48 hours after that), all requests are directed to the in-memory database first, ignoring the cached database. After 48 hours of inactivity, the file is unloaded from memory, but stays in the cache where it can be quickly reloaded if a new connection request for that data is intercepted by the local Power Pivot server instance. Connection requests to an inactive database are served from the cache rather than the content library, minimizing impact on the content databases.

   It is important to note that the content library is the only permanent location for Power Pivot databases. Cached copies are used only if the database in the library is the same as the copy on disk.

   Keep Inactive Database in Cache specifies how long an inactive database remains on the file system after it has been unloaded from memory. The cleanup job uses this setting to determine which files to delete. All Power Pivot databases that are inactive for 168 hours (48 hours in memory and 120 hours in the cache) are deleted from disk by the cleanup job.

5. Click **OK** to save your changes.

Next Steps

A Power Pivot for SharePoint installation provides health rules so that you can take corrective action when problems are detected in server health, configuration, or availability. Some of these rules use configuration settings to determine the conditions under which health rules are triggered. If you are actively tuning server performance, you might also want to review these settings to ensure the defaults are the best choice for your system. For more information, see Configure Power Pivot Health Rules.

See Also

Power Pivot Server Administration and Configuration in Central Administration

Configure Maximum File Upload Size (Power Pivot for SharePoint)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Power Pivot workbooks often contain large amounts of data that result in files that exceed the maximum file size allowed for SharePoint uploads. When you try to upload a file that exceeds the upper limit, you will get the following error on SharePoint:

- "The specified file is larger than the maximum supported file size."

To increase the file size, start by adjusting the Maximum Workbook Size for Excel Services to 50 megabytes. This aligns the maximum file size limits for Excel to those of SharePoint web applications (set to 50 megabytes by default in SharePoint 2010 and 200 in SharePoint 2013). If your files exceed 50 megabytes in size, increase the file size limits for both Excel Services and your web application.

You must be a SharePoint administrator to change the maximum file upload size.

Configure maximum file size for Excel Services

1. In Central Administration, in Application Management, click **Manage service applications**.
2. Click the name of your Excel Services Application.
3. Click **Trusted File Locations**.
4. Click the location to edit the properties. By default, Excel Services considers the default web application a trusted site. If you are using the default web application, click **http://** to open the configuration page for this location.
5. Scroll to **Workbook Properties**.
6. In Maximum Workbook Size, increase the file size from 10 (the default value) to 50 or a larger size that accommodates the files you are working with.

   By default, 50 megabytes is the maximum file upload size for SharePoint web applications. If you set Maximum Workbook Size to a number larger than 50 megabytes, follow the steps in the next procedure to increase the Maximum Upload Size for your SharePoint web application to the same value.

   The maximum value that you can specify is 2 gigabytes (or 2047 megabytes as specified in Central Administration).
7. Click **OK**.

Configure maximum file size for a SharePoint web application

1. In Central Administration, in Application Management, click **Manage web applications**.

   > **NOTE**
   > Perform the following steps only if you increased the Maximum Workbook Size in Excel Services.

2. Select the application (for example, **SharePoint - 80**).
3. On the Web Applications ribbon, click the down arrow on the General Settings button.

4. Click **General Settings**.
5. Scroll to **Maximum Upload Size**.
6. Set the property to the same number, or larger as the Maximum Workbook Size in Excel Services.
7. Click **OK**.

Configuration Setting Reference (Power Pivot for SharePoint)

7/11/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic provides reference documentation for configuration settings used by Power Pivot service applications in a SharePoint farm. If you are using PowerShell script to configure a server, or if you want to look up information for a specific setting, the information in this topic provides detailed descriptions.

Configuration settings are set for each Power Pivot service application. Within a farm, you can create multiple service applications as a way to configure independent logical instances of the same physical service instance. Configuration settings are stored in the Power Pivot application database created for each service application that you configure.

If you change configuration settings, the changes are picked up immediately and used for subsequent requests and connections. Operations that are in progress are governed by the settings that were in effect when the operation began.

Click the following links to read about specific configuration areas:

Data Load Timeout

Connection Pools

Load Balancing

Data Refresh

Usage Data Collection

For instructions on how to create a Power Pivot service application, see Create and Configure a Power Pivot Service Application in Central Administration.

Data Load Timeout

Power Pivot data is retrieved and loaded by Analysis Services server instances in the farm. Depending on how and when the data was last accessed, it will either be loaded from a content library or from a local file cache. Data is loaded into memory whenever a query or processing request is received. To maximize overall server availability, you can set a timeout value that instructs the server to stop a load data request if it cannot be completed within the allotted time.

NAME	DEFAULT	VALID VALUES	DESCRIPTION

NAME	DEFAULT	VALID VALUES	DESCRIPTION
Data Load Timeout	1800 (in seconds)	1 to 3600	Specifies the amount of time a Power Pivot service application will wait for a response from a specific Analysis Services server instance. By default, the service application will wait 30 minutes for a data payload from the Engine service instance to which it forwarded a specific request. If the Power Pivot data source cannot be loaded within this period of time, the thread will be stopped and a new one will be started.

Connection Pools

The Power Pivot service application creates and manages connection pools to enable connection reuse. There are two types of connection pools: one for data connections to read-only data, and a second for server connections.

Data connection pools contain cached connections for Power Pivot data source. Each connection pool is based on the context that was set when the database was loaded. This context includes the identity of the physical service instance, the database ID, and the identity of the SharePoint user who is requesting data. A separate connection pool is created for each combination. For example, requests from different users of the same database running on the same server will consume connections from different pools.

The purpose of a connection pool is to use cached connections for read-only requests for the same Analysis Services database by the same SharePoint user. The Power Pivot service instance is the server that has the data loaded in memory. The database ID is an internal identifier for the in-memory data structures of the data model (a model is instantiated as an Analysis Services cube database). Version information is implicitly incorporated in the identifier.

Server connection pools contain cached connections from a Power Pivot service application instance to an Analysis Services server instance, where the service application is connecting with Analysis Services Sysadmin permissions on the Analysis Services server. These connections are used to issue a load database request and monitor system health.

Each type of connection pool has upper limits that you can set to ensure best use of system memory for connection management.

NAME	DEFAULT	VALID VALUES	DESCRIPTION

NAME	DEFAULT	VALID VALUES	DESCRIPTION
Connection Pool Timeout	1800 (in seconds)	1 to 3600.	This setting applies to data connection pools. It specifies how much time an idle connection can remain in a connection pool before it is removed. By default, the service application will remove a connection if it is inactive for more than five minutes.
Maximum User Connection Pool Size	1000	-1, 0, or 1 to 10000. -1 specifies an unlimited number of idle connections. 0 means no idle connections are kept. New connections to a Power Pivot data source must be created each time.	This setting applies to the number of idle connections in all data connection pools created for a specific Power Pivot service application instance. Individual connection pools are created for unique combinations of a SharePoint user, Power Pivot data, and service instance. If you have many users accessing a variety of Power Pivot data sources, server performance might benefit from an increase in connection pool size. If there are more than 100 idle connections to a Power Pivot service instance, newly idle connections are disconnected rather than returned to the pool.
Maximum Administrative Connection Pool Size	200	-1, 0, or 1 to 10000. -1 specifies an unlimited number of idle connections.	The maximum number of idle server connections in all administrative connection pools created for Power Pivot service application connections to an Analysis Services server instance. Server connections are used for requests to load databases and to save changes back to the SharePoint database.

Load Balancing

One of the functions that the Power Pivot service performs is to determine where Analysis Services data will be loaded among the available Power Pivot service instances. The **AllocationMethod** setting specifies the criteria against which a service instance is selected.

NAME	DEFAULT	VALID VALUES	DESCRIPTION
Allocation Method	RoundRobin	Round Robin Health Based	A scheme for allocating load requests among two or more Analysis Services server instances. By default, the Power Pivot service will alternate requests based on server health. Health based allocates requests to the server that has the most system resources available based on available memory and CPU utilization. Round robin rotates requests amongst the available servers in sequential order, regardless of current load or server health.

Data Refresh

Specify the range of hours that defines a normal or typical business day for your organization. These configuration settings determine when after-hours data processing occurs for data refresh operations. After-hours processing can begin at the end time of the business day. After-hours processing is a schedule option for document owners who want to refresh a Power Pivot data source with transactional data that was generated during normal business hours.

NAME	DEFAULT	VALID VALUES	DESCRIPTION
Start time	04:00 a.m.	1 to 12 hours, where the value is a valid integer within that range. Type is Time.	Sets the lower limit of a business hour range.
End time	08:00 p.m.	1 to 12 hours, where the value is a valid integer within that range. Type is Time.	Sets the upper limit of a business hour range.

NAME	DEFAULT	VALID VALUES	DESCRIPTION
Power Pivot Unattended Data Refresh Account	None	A target application ID	This account is used to run data refresh jobs on behalf of a schedule owner. The unattended data refresh account must be defined in advance before it can be referenced in the service application configuration page. For more information, see Configure the Power Pivot Unattended Data Refresh Account (Power Pivot for SharePoint).
Allow users to enter custom Windows credentials	Enabled	Boolean	Determines whether the scheduled data refresh configuration page shows an option that allows a schedule owner to specify Windows user account and password to run a data refresh job. Secure Store Service must be enabled in order for this option to work. For more information, see Configure Stored Credentials for Power Pivot Data Refresh (Power Pivot for SharePoint).
Maximum Processing History Length	365	1 to 5000 days	Determines how long data refresh history is retained in the Power Pivot service application database. For more information, see Power Pivot Usage Data Collection.

Usage Data Collection

Usage reports that appear in the Power Pivot Management Dashboard can provide important information about how Power Pivot-enabled workbooks are used. The following configuration settings control aspects of usage data collection for Power Pivot server events that are subsequently presented in usage or activity reports.

NAME	DEFAULT	VALID VALUES	DESCRIPTION

NAME	DEFAULT	VALID VALUES	DESCRIPTION
Query Reporting Interval	300 (in seconds)	1 to n seconds, where n is any valid integer.	To ensure that usage data collection does not consume too much of the data transfer capacity of the farm, query statistics are collected on each connection and reported as a single event. The Query Reporting Interval determines how often an event is reported. By default, query statistics are reported every 5 minutes. Because connections are immediately closed as soon as a request is sent, the system generates a very large number of connections for even a single user accessing a single Power Pivot data source. For this reason, connection pools are created for each user and Power Pivot data source combination so that once a connection is created it can be reused by the same user for the same data. Periodically, at intervals specified through this configuration setting, the Power Pivot service application reports the usage data for each connection in the connection pool. Increasing the time-to-report value will cause fewer events to be logged. However, if you set it too high, you risk losing event data if the server restarts or a connection is closed. Lowering the value will cause more events to be logged with greater frequency, adding more Power Pivot-related usage data to the data collection system in the SharePoint usage database. Generally, do not change this configuration setting unless you are trying to resolve a specific problem (for example, if the usage database is growing too quickly as a result of Power Pivot usage data)

NAME	DEFAULT	VALID VALUES	DESCRIPTION
			Pivot usage data).
Usage Data History	365 (in days)	0, or 1 to n days, where n is any valid integer. 0 means that history is always retained and never deleted.	By default, usage data is kept for one year in the Power Pivot service application database. Records that are older than one year are dropped from the database. A check for expired historical data occurs daily, when the Microsoft SharePoint Foundation Usage Data Processing job runs. The timer job will read this setting and trigger a data deletion command for expired history in the Power Pivot service application database.
Trivial Response Upper Limit	500 (in milliseconds)	1 to n milliseconds, where n is any valid integer.	By default, the threshold for trivial requests is half a second. Trivial requests include server pings, requests for metadata, and starting sessions.
Quick Response Upper Limit	1000 (in milliseconds)	1 to n milliseconds, where n is any valid integer.	By default, the threshold for quick requests is one second. Quick requests are those that have an extremely small dataset, or requests for metadata that span large member sets.
Expected Response Upper Limit	3000 (in milliseconds)	1 to n milliseconds, where n is any valid integer.	By default, the threshold for expected requests is three seconds. This threshold sets the upper limit of an expected query time.
Long Response Upper Limit	10000 (in milliseconds)	1 to n milliseconds, where n is any valid integer.	By default, the threshold for long requests is ten seconds. These are requests that run longer than expected, but still fall within an acceptable range.

See Also

Create and Configure a Power Pivot Service Application in Central Administration
Power Pivot Data Refresh with SharePoint 2010

Configure Usage Data Collection for (Power Pivot for SharePoint
Configure Power Pivot Service Accounts
Power Pivot Management Dashboard and Usage Data

Start or Stop a Power Pivot for SharePoint Server

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Power Pivot System Service and an Analysis Services service instance operate together on the same local application server to support coordinated request and data processing in a SharePoint farm.

This topic contains the following sections:

Service Dependencies

Start or Stop the Services

Effects of Stopping a Power Pivot Server

Service Dependencies

The Power Pivot System Service has a dependency on the local Analysis Services server instance that is installed with it on the same physical server. If you stop the Power Pivot System Service, you must also manually stop the local Analysis Services server instance. If one service is running without the other, you will encounter request allocation errors for Power Pivot data processing.

The Analysis Services server should only be run by itself if you are diagnosing or troubleshooting a problem. In all other cases, the server requires the Power Pivot System Service that runs locally on the same server.

Start or Stop the Services

Always use Central Administration to start or stop the Power Pivot System Service or Analysis Services server instance. Central Administration allows you to start or stop the services together from the same page. In addition, Central Administration uses a timer job called **One or more services have started or stopped** to restart services that it thinks should be running. If you stop the Power Pivot System Service or Analysis Services using a non-SharePoint tool, the services will be restarted when the timer job runs.

Starting and stopping services is an action that applies to a physical service instance. If you have additional Power Pivot for SharePoint servers in the farm, the other servers within the farm will continue to accept requests for Power Pivot data.

You cannot start or stop all physical services simultaneously across the farm. You must select each server and then start or stop a particular service.

You cannot start, pause, or stop a Power Pivot System Service for a specific Web application, but you can remove a service from the default connection list to make it unavailable. For more information, see Connect a Power Pivot Service Application to a SharePoint Web Application in Central Administration.

1. In Central Administration, in **System Settings**, click **Manage services on server**.
2. At the top of the page, in Server, click the down arrow, and then click **Change Server**.
3. Select the SharePoint server that has the Power Pivot System Service or Analysis Services service instance that you want to start or stop.
4. Select the service, and then click the action. Remember to start or stop the services as a pair. If you start or stop the Power Pivot System Service, be sure to also start or stop the Analysis Services server instance that runs on the same computer.

Effects of Stopping a Power Pivot Server

The following table describes the effects of stopping the Power Pivot System Service and Analysis Services service on a SharePoint server.

EFFECT ON	DESCRIPTION
Existing queries	Queries that are in progress on an Analysis Services server will stop immediately. The user will get a data not found or data source connection not found error.
Existing data refresh jobs that are currently processing	Jobs that are in progress on the current Analysis Services server will stop immediately. Data refresh will fail and an error will be logged to data refresh history. You can view the status of current jobs before you stop the service by using the Check job status page in SharePoint Central Administration. While it is possible to know which jobs are currently processing, there is no way to view the queue itself to see whether other jobs are about to start.
Existing data refresh requests in the queue	Scheduled data refresh requests will remain in the processing queue for a complete cycle of the schedule (that is, it stays in the queue until the next start time). If the Power Pivot System Service is not restarted by then, the data refresh request will be dropped and an error will be logged.
New requests for queries or data refresh	If you are stopping the only Power Pivot for SharePoint server in the farm, new requests for Power Pivot data will not be handled and a request for data will result in a data not found error. If you have additional Power Pivot for SharePoint servers, the request will go to one of the available servers.
Usage data	Usage data will not be collected while the services are stopped.

See Also

Configure Power Pivot Service Accounts

Power Pivot Configuration using Windows PowerShell

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

SQL Server 2017 includes Windows PowerShell cmdlets that you can use to configure an installation of Power Pivot for SharePoint. To fully configure an installation with PowerShell requires the use of both SharePoint cmdlets and Power Pivot for SharePoint cmdlets. A majority of configuration can be completed using one of the Power Pivot tools. For more information on the tools, see Power Pivot Configuration Tools.

> **IMPORTANT**
>
> For a SharePoint 2010 farm, SharePoint 2010 SP1 must be installed before you can configure either Power Pivot for SharePoint, or a SharePoint farm that uses a SQL Server 2012 (11.x) database server. If you have not yet installed the service pack, install it before you begin configuring the server.

Benefits of Configuring Power Pivot for SharePoint Using PowerShell

You can build Windows PowerShell script (.ps1) files to automate configuration tasks. This approach is recommended if you require scripted installation and configuration steps that you can run on any server. You might require such a script as part of a disaster recovery plan for rebuilding a server in the event of a hardware failure.

View a list of the Power Pivot Cmdlets on a Server

To see content and examples of the Power Pivot cmdlets, see PowerShell Reference for Power Pivot for SharePoint.

To use PowerShell to view a list of the Power Pivot cmdlets:

1. Open SharePoint Management Shell using the **Run as Administrator** option.
2. Enter the following command:

   ```
   Get-help *powerpivot*
   ```

   You should see a list of cmdlets that include Power Pivot in the cmdlet name. For example `Get-PowerPivotServiceApplication`. The number of cmdlets available depends on the version of Analysis Services you are using.

   - 10 cmdlets with SQL Server 2012 SP1 Analysis Services server configured in SharePoint mode, and SharePoint 2013. The 2012 SP1 version utilizes a new architecture that allows the Analysis Server to run outside the SharePoint farm and requires fewer management Windows PowerShell cmdlets.
   - 17 cmdlets with SQL Server 2012 Analysis Services server configured in SharePoint mode, and SharePoint 2010.

   If no commands are returned in the list or you see an error message similar to "`get-help could not find *powerpivot* in a help file in this session.`", see the next section in this topic

for instructions on how to enable the Power Pivot cmdlets on the server.

All cmdlets have online help. The following example shows how to view the online help for the **New-PowerPivotServiceApplication** cmdlet:

```
Get-help new-powerpivotserviceapplication -full
```

Alternatively, to view just the examples, use the following syntax:

```
Get-help new-powerpivotserviceapplication -example
```

Enable Power Pivot Cmdlets on a Server

Power Pivot cmdlets are available after you install Power Pivot for SharePoint and deploy the farm solution. The solutions are deployed when you ran the Power Pivot Configuration tool. Follow these steps to enable the use of cmdlets:

1. Open SharePoint Management Shell using the **Run as Administrator** option.
2. Run the first cmdlet:

   ```
   Add-SPSolution -LiteralPath "C:\Program Files\Microsoft SQL
   Server\110\Tools\PowerPivotTools\ConfigurationTool\Resources\PowerPivotFarm.wsp"
   ```

   The cmdlet returns the name of the solution, its solution ID, and Deployed=False. In the next step, you deploy the solution.
3. Run the second cmdlet to deploy the solution:

   ```
   Install-SPSolution -Identity PowerPivotFarm.wsp -GACDeployment -Force
   ```
4. Close the window. Reopen it, again using the **Run as Administrator** option.

Related Content

Power Pivot Server Administration and Configuration in Central Administration

Power Pivot Configuration Tools

PowerShell Reference for Power Pivot for SharePoint

Power Pivot Configuration Tools

12/10/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Configure, repair, or remove a SQL Server 2017 Power Pivot for SharePoint with the Power Pivot Configuration Tools.

The SQL Server 2017 Setup Wizard does not install Power Pivot configuration tools for SharePoint. The configuration tools will be available for download in a future release of SQL Server 2017.

This topic describes the general use of the Power Pivot tools and the differences between them.

Applies to: SharePoint 2013

In this Topic:

- Requirements for Using the Configuration Tools
- Two Versions of the Configuration Tool
- Overview of Using a Power Pivot Configuration Tool
- Start One of the Power Pivot Configuration Tools

Requirements for Using the Configuration Tools

- You must be a farm administrator.
- You must be a server administrator on the Analysis Services instance (SharePoint 2010 only).
- You must be db_owner on the farm's configuration database.
- There are no TCP/IP port requirements for using the configuration tools and therefore you should not have to configure your firewall to accommodate the configuration tools. The configuration tool expects the web applications and shared services are available as part of the SharePoint platform. You may need to configure your firewall for the Analysis Services server. For more information, see Configure the Windows Firewall to Allow Analysis Services Access.

Two Versions of the Configuration Tool

The SQL Server 2017 Setup Wizard installs the Power Pivot Configuration Tool for SharePoint 2010 as well as a Power Pivot Configuration Tool for SharePoint 2013.

The tools can only be used with a SQL Server 2017 or SQL Server 2012 SP1 (11.0.3x) instance of Power Pivot for SharePoint. Do not use them with SQL Server 2008 R2 installations.

NAME	SUPPORTED VERSION OF SHAREPOINT	DETAILED CONFIGURATION
Power Pivot for SharePoint 2013 Configuration	SharePoint 2013	Configure or Repair Power Pivot for SharePoint 2013 (Power Pivot Configuration Tool)

NAME	SUPPORTED VERSION OF SHAREPOINT	DETAILED CONFIGURATION
Power Pivot Configuration Tool	SharePoint 2010 with SharePoint 2010	Configure or Repair Power Pivot for SharePoint 2010 (Power Pivot Configuration Tool)

How the Two Configuration Tools are Different

The two versions of the Configuration tool are similar but there are differences in the configuration steps the two tools run. The differences are due to changes between SharePoint 2010 and SharePoint 2013 as well as the architecture differences between the SQL Server 2012 SP1 version of Power Pivot for SharePoint and the previous versions of Power Pivot for SharePoint.

The following table describes new and changed features in the **Power Pivot for SharePoint 2013 Configuration** tool. The table also describes features in the **Power Pivot Configuration Tool** that are not in the Power Pivot for SharePoint 2013 Configuration tool. The rows in the table are in the same order as the tabs in the configuration tools.

POWER PIVOT FOR SHAREPOINT 2013 CONFIGURATION	POWER PIVOT CONFIGURATION TOOL
The main page has a new option for **Power Pivot Server for Excel Services**. The option supports the new architecture with Analysis Services running outside of the SharePoint farm. You configure Excel Services to use one or more Analysis Services servers that are running in SharePoint mode. 	
	The 2010 tool includes the page **Register SQL Server Analysis Services (Power Pivot) on Local Server** to configure a local instance of Analysis Services. This page is not part of the 2013 tool because there is no local instance of Analysis Services.
	The page **Create Power Pivot Service Application** has an extra option of **Upgrade Workbooks to enable data refresh**. This option is not available in the 2013 tool.
The 2013 tool has a new page **Configure Power Pivot Servers**. This page supports the new architecture of Analysis Services running outside the SharePoint farm. By default, the server name that was typed on the main page in the text box **Power Pivot Server for Excel Services**, is also be listed on the **Configure Power Pivot Servers**. 	

POWER PIVOT FOR SHAREPOINT 2013 CONFIGURATION	POWER PIVOT CONFIGURATION TOOL
The 2013 tool has a new page **Register Power Pivot addin as Excel Services Usage Tracker**. SharePoint 2010 Excel Services does not track usage data for Power Pivot.	
	The 2010 tool includes the page **Add MSOLAP.5 as a trusted Provider** to register MSOLAP so that Excel Services in SharePoint 2010 can load Power Pivot models. This page is not part of the 2013 tool. SharePoint 2013 Excel Services does not use the MSOLAP provider to load models.

Overview of Using a Power Pivot Configuration Tool

When you start one of the Power Pivot Configuration Tools, the tool evaluates the existing installation to determine which operations are applicable. On a new installation, only the configuration task is available. After the server is configured, the remove task appears. If you started with a SQL Server 2008 R2 instance, upgrade will also be enabled in the list of available tasks.

If you are not familiar with Central Administration or Windows PowerShell, you can run the configuration tool as an alternative to complete a Power Pivot for SharePoint installation.

Additionally, the tool can detect whether the farm is configured or required features are missing. If the SharePoint program files are installed but the farm is not configured, the tool provides actions for configuring both the farm and the Power Pivot for SharePoint installation.

You can review the **Script** tab to learn and understand how to configure Power Pivot and SharePoint using Windows PowerShell. For more information, see the following:

- Power Pivot Configuration using Windows PowerShell
- PowerShell Reference for Power Pivot for SharePoint

> **NOTE**
>
> The tool does not configure Reporting Services. If you are adding Reporting Services to your SharePoint environment, you need to install and configure Reporting Services separately. For more information, see the following:
>
> - Install The First Report Server in SharePoint Mode.
> - Install Reporting Services SharePoint Mode for SharePoint 2010.

Start One of the Power Pivot Configuration Tools

1. On the **Start** screen, type **powerpivot**

   On the **Start** screen, type **powerpivot** or on the **Start** menu, click **All Programs**, click **Microsoft SQL Server 2017**, click **Configuration Tools**, and then click one of the following:

   - **Power Pivot Configuration Tool**.
   - **OR**
   - **Power Pivot for SharePoint 2013 Configuration**.

Note: The tools are available only when Power Pivot for SharePoint is installed on the local server.

2. At startup, the configuration tools check the status of your installation and provide tasks that are valid for your installation.

3. Depending on the current state of your installation, one or more of the following tasks can be performed:

   a. Click **Configure or Repair Power Pivot for SharePoint** to complete post-installation tasks, or to repair an installation.

   b. Click **Remove Features, Services, Applications and Solutions** to remove features and solutions from the farm.

   c. Click **Upgrade Features, Services, Applications and Solutions** to upgrade features and solutions that were installed using a previous version of Power Pivot for SharePoint.

   For example, the image shows the startup page of the Power Pivot Configuration for SharePoint 2013 tool.

Each task is composed of individual actions that address some aspect of server configuration. For example, the configuration task includes actions for deploying solutions, creating a Power Pivot service application, activating features, and configuring data refresh. The list of actions will vary based on the current state of your installation. If an action is not needed, the tool excludes it from the task list.

When you click Run , the tool processes all actions in batch mode. Although each action appears as a separate item in the task list, all actions that are included in the task process together. Only actions that pass a validation check are processed. You might need to add or change some of the input values to pass the validation check.

Related Content

Upgrade Power Pivot for SharePoint Describes the workflow that upgrades an existing installation that is already in a farm.

Uninstall Power Pivot for SharePoint Describes the workflow that removes Power Pivot for SharePoint services, solutions, and application pages from a farm.

Power Pivot Configuration using Windows PowerShell

Power Pivot Server Administration and Configuration in Central Administration

Configure or Repair Power Pivot for SharePoint 2013

12/10/2018 • 15 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

To configure or repair an installation of SQL Server 2017Power Pivot for SharePoint 2013, use the Power Pivot for SharePoint Configuration tool. The configuration tool starts by scanning the system and returns a list of actions needed to complete or repair an installation. The SQL Server 2017 setup wizard installs the Power Pivot Configuration Tool for SharePoint 2010 as well as a Power Pivot Configuration Tool for SharePoint 2013. This topic describes the Power Pivot Configuration Tool for SharePoint 2013. For more information on SharePoint 2010, see Configure or Repair Power Pivot for SharePoint 2010 (Power Pivot Configuration Tool).

Applies to: SharePoint 2013

In this topic:

Before you start

To Use the Power Pivot for SharePoint 2013 Configuration Tool

Configuration steps

Input values used to configure the server

Next steps

Before you start

The Power Pivot for SharePoint 2013 Configuration tool scans for program files, registry settings, and available ports. To get the most use out of the tools, review the following.

- General requirements to run the configuration tool, Power Pivot Configuration Tools.
- Power Pivot for SharePoint 2013 prefers web application configured for claims based authentication. If the Power Pivot for SharePoint 2013 Configuration tool creates the application for you, it configures the application to use claims-based Windows authentication. For more information about authentication requirements, see Power Pivot Authentication and Authorization.
- Port 80 must be available for the PPower Pivot for SharePoint 2013 Configuration tool to create a web application.

To Use the Power Pivot for SharePoint 2013 Configuration Tool

The first page of the tool provides a summary of the input values used to configure the SharePoint farm. In addition to the input values that you provide, default values are used to configure the system. Default names are used for service applications, service application databases, and service application properties.

> **TIP**
>
> If the configuration tool scans the computer and returns a blank task list in the left pane, then no features or settings were detected that need configuration. To modify the SharePoint or Power Pivot configuration, use Windows PowerShell or the management pages in SharePoint Central Administration. For more information, see Power Pivot Server Administration and Configuration in Central Administration.

Values for service accounts are used for multiple services. For example, the configuration tool uses the default account on the first page to set all application pool identities. You can change these accounts later by modifying the service application properties in Central Administration.

The tool provides a tabbed interface that includes parameter inputs, Windows PowerShell script, and status messages.

The tool uses Windows PowerShell to configure the server. You can click the **Script** tab to review the Windows PowerShell script that the tool uses to configure the server.

	DESCRIPTION
(1)	Task list window.
(2)	Individual actions.
(3)	The Windows PowerShell scripts created by the configuration tool.
(4)	Log messages created when you start validation or run actions.
(5)	Description of the page.
(6)	Input parameters
(7)	The **Run** button is enabled after you validate the actions.

Configuration steps

The link to the configuration tool is only visible when Power Pivot for SharePoint 2013 is installed on the local

server.

1. On the **Start** menu, point to **All Programs**, click **Microsoft SQL Server 2017**, click **Configuration Tools**, and then click **Power Pivot for SharePoint 2013 Configuration**.

2. Click **Configure or Repair Power Pivot for SharePoint**.

3. Expand the window to full size. You should see a button bar at the bottom of the window that includes **Validate**, **Run**, and **Exit** commands.

4. **Default Account:** On the Parameters tab, type a domain user account for the **Default Account Username**. This account is used to provision essential services, including the Power Pivot service application pool. Do not specify a built-in account such as Network Service or Local System. The tool blocks configurations that specify built-in accounts.

   Passphrase: type a passphrase. If the SharePoint farm is new, the passphrase is used whenever new servers or applications are added to the SharePoint farm. If the farm exists, enter the passphrase that allows you to add a server application to the farm.

5. **Port:** Optionally, type a port number to connect to the Central Administration web application or use the provided randomly generated number. The configuration tool checks that the number is available before offering it as an option.

6. On the main page, type the name of a SQL Server 2017Power Pivot Server running in SharePoint mode.

7. Optionally, review the remaining input values used to complete each action. For more information about each one, see Input values used to configure the server in this topic.

8. Optionally, remove any actions that you do not want to process. For example, if you want to configure Secure Store Service later, click **Configure Secure Store Service**, and then clear the checkbox **Include this action in the task list**.

9. Click **Validate** to check whether the tool has sufficient information to process the actions in the list.

10. Click **Run** to process all of the actions in the task list. The **Run** button becomes available after you validate the actions. If **Run** is not enabled, click **Validate** first.

    If you see an error message similar to the following, verify that the SQL Server database instance is started.

    ```
    Cannot connect to the database server instance
    ```

11. Verify a Power Pivot for SharePoint Installation.

Input values used to configure the server

The Power Pivot Configuration Tool uses a combination of input values that you type in and default values that it detects or uses automatically.

The list of actions the configuration tool displays, depends on the SharePoint farms current configuration. For example, if the SharePoint farm is already configured, you do not see actions that configure the farm or create a web application. You can run the tool at any time to configure, repair, or detect configuration errors. If required services such as Excel Services or Secure Store Service are not running in the farm, the tool detects the missing services and provide options to enable them. If no actions are required, the task list is empty.

The following table describes the values that are used to configure the server.

PAGE	INPUT VALUE	SOURCE	DESCRIPTION
Configure or Repair Power Pivot for SharePoint	Default Account	Current user	The default account is a domain Windows user account used to provision shared services in the farm. It is used to provision the following:
			- Power Pivot service application
			-Secure Store Service
			-Excel Services
			-The web application pool identity
			-The site collection administrator
			-The Power Pivot unattended data refresh account.
			By default, the domain account of the current user is used. Note: It is recommended you replace the default value unless you are configuring a server for evaluation and non-production purposes. You can change service identities after configuration or repair, using Central Administration. Optionally, in the Power Pivot Configuration tool, specify dedicated accounts for the following:
			-Web application, using the **Create Default Web Application** page (assuming that the tool is creating a web application for the farm).

PAGE	INPUT VALUE	SOURCE	DESCRIPTION
			- Power Pivot unattended data refresh account, using the **Create Unattended Account for Data Refresh** page in this tool.
	Database Server	Local Power Pivot named instance, if available	If a database engine instance is installed as a Power Pivot named instance, the tool populates the database server field with this instance name. If you did not install the database engine, this field is empty. **Database Server** is a required parameter. It can be any version or edition of SQL Server that is supported for SharePoint farms.
	Passphrase	User input	If you are creating a new farm, the passphrase you enter is used as the passphrase for the farm. If you are adding Power Pivot for SharePoint to an existing farm, type the existing farm passphrase.
	SharePoint Central Administration port	Default, if needed	If the farm is not configured, the tool provides options to create the farm, including creating an HTTP endpoint to Central Administration. It picks a randomly generated port number that is not in use.
	Power Pivot for Excel Services ([ServerName]\ Power Pivot)	User input	The Power Pivot server is required for Excel Services to Enable the core Power Pivot capabilities. The server name you type on this page is also added to the list on the **Configure Power Pivot Servers** page.
Configure New Farm	Database Server Farm Account PassPhrase SharePoint Central Administration port	Default, if needed	The settings default to what you entered on the main page.

PAGE	INPUT VALUE	SOURCE	DESCRIPTION
Create Power Pivot Service Application	Service Application Name	Default	Power Pivot Service Application Name The default name is **Default Power Pivot Service Application**. You can substitute a different value in the tool.
	Database Server	Default	The database server to host the Power Pivot service application database. The default server name is the same database server used for the farm. You can substitute a different value than the default server name.
	Database Name	Default	The name of the database to create for the Power Pivot Service Application Database. The default database name is based on the service application name, followed by a GUID to ensure a unique name. You can substitute a different value in the tool.
Create Default Web Application	Web Application Name	Default, if needed	If no web applications exist, the tool creates one. The web application is configured for classic mode authentication, and listens on port 80. The maximum file upload size is set to 2047, the maximum allowed by SharePoint. The larger file upload size is to accommodate the large Power Pivot files that will be uploaded to the server.
	URL	Default, if needed	The tool creates a URL based on the server name, using the same file naming conventions as SharePoint.
	Application Pool	Default, if needed	The tool creates a default application pool in IIS.
	Application Pool Account and Password	Default, if needed	The application pool account is based on the default account, but you can override it in the tool.

PAGE	INPUT VALUE	SOURCE	DESCRIPTION
	Database Server	Default, if needed	The default database instance is pre-selected to store the application content database, but you can specify a different SQL Server instance in the tool.
	Database Name	Default, if needed	The name of the application database. The database name is based on the file naming conventions of SharePoint, but you can choose a different name.
Deploy Web Application Solution	URL	Default, if needed	The Default URL is from the default web application.
	Maximum File Size (in MB)	Default, if needed	Default setting is 2047. SharePoint document libraries also have a maximum size and the Power Pivot setting should not exceed the document library setting. For more information, see Configure Maximum File Upload Size (Power Pivot for SharePoint).
Create Site Collection	Site Administrator	Default, if needed	The tool uses the default account. You can override it in the **Create Site Collection** page.
	Contact E-mail	Default, if needed	If Microsoft Outlook is configured on the server, the tool will use the email address of the current user. Otherwise, a placeholder value is used.
	Site URL	Default, if needed	The tool creates the site URL, using the same URL naming conventions as SharePoint.
	Site Title	Default, if needed	The tool adds **Power Pivot Site** as the default title.
Activate Power Pivot Feature in a Site Collection	Site URL		URL of the site collection for which you are activating Power Pivot features.
	Enable premium feature for this site		Enable the SharePoint site feature "PremiumSite".

PAGE	INPUT VALUE	SOURCE	DESCRIPTION
Create Secure Store Service Application	Service Application Name	Default, if needed	Type the name for the Secure Store service application.
	Database Server	User input	Type the name of the database server to use for the Secure Store service application.
Create Secure Store Service Application Proxy	Service Application Name	Default, if needed	Type the name for the Secure Store service application you typed in the previous page.
	Service Application Proxy	Default, if needed	Type the name for the Secure Store service application proxy. The name will appear in the default connection group that associates applications with SharePoint content web applications.
Update Secure Store Service Master Key	Service Application Proxy	Default, if needed	Type the name of the Secure Store service application proxy you typed in the previous page.
	Passphrase	User input	The master key used for data encryption. By default, the passphrase used to generate the key is the same passphrase that is used to provision new servers in the farm. You can replace the default passphrase with a unique passphrase.
Create Unattended Account for DataRefresh	Target Application ID	Default, if needed	Create a target application to stores credentials for unattended Power Pivot data refresh. The application ID can be descriptive text.
	Friendly Name for Target Application	Default, if needed	

PAGE	INPUT VALUE	SOURCE	DESCRIPTION
	Unattended Account User Name and Password	Default, if needed	Type credentials of a Windows user account that is used by the target application to run unattended data refresh. For more information, see Configure Excel Services data refresh by using the unattended service account in SharePoint Server 2013 (http://technet.microsoft.com/library/hh525344(office.15).aspx).
	Site URL	Default, if needed	Type the site URL of the site collection associated wit the target application. To associate with additional site collections, use SharePoint Central administration.
Create Excel Services Service Application	Service Application Name	Default, if needed	Type a service application name. A service application database with the same name is created on the SharePoint farm's database server.
Configure Power Pivot Servers	Service Application Name	Default, if needed	Service application name you typed on the previous page.
	Power Pivot Server Name		List of Registered Power Pivot Servers. The server name typed on the main page is automatically be added to this page.
Register Power Pivot Add-in as Excel Services Usage Tracker	Service Application Name		Service application name you typed on the previous page.

If the Power Pivot for SharePoint 2013 Configuration tool creates the farm, it creates the required databases on the database server, using the same file naming conventions as SharePoint. You cannot change the farm database name.

If the tool creates a site collection, it creates a content database on the database server, using the same file naming conventions as SharePoint. You cannot change the content database name.

Verify the Configuration

See the "Verify Power Pivot Configuration" section of Configure Power Pivot and Deploy Solutions (SharePoint 2013).

Next steps

After you complete server installation, there are several post-installation tasks you should perform:

- Grant SharePoint permissions to individuals and groups. This task is necessary to enable access to sites and content.
- Change the service application pool identities to run under a different account. Specifying different identities for services and applications is a SharePoint best practice recommendation for secure deployment.
- Create additional trusted sites in Excel Services so that you can vary permissions and configuration settings that work best for Power Pivot data access.
- Install commonly used data providers to enable server-side data refresh.

Grant SharePoint Permissions to Workbook Users

Users will need SharePoint permissions before they can publish or view workbooks. Grant **View** permissions to users who need to view published workbooks and **Contribute** permissions to users who publish or manage workbooks. You must be a site collection administrator to grant permissions.

1. In a SharePoint 2013 site, click the settings icon ⚙ and then click **Site Settings**.
2. Click **Site Permissions** in the **Users and Permissions** group.
3. Create groups as needed if you want a set of users with **Contribute** permissions and another group for a set of users with only **View** permissions.
4. Enter the Windows domain user or group accounts who should have membership in the groups. As before, do not use e-mail addresses or distribution group if the application is configured for classic authentication.

Install Data Providers Used in Data Refresh and Check User Permissions

Server-side data refresh allows users to re-import updated data to their workbooks in unattended mode. In order for data refresh to succeed, the server runnin Analysis Services in SharePoint mode must have the same data providers that were used to originally import the data. In addition, the user account under which data refresh runs often requires read permissions on the external data sources. Be sure to check the requirements for enabling and configuring data refresh to ensure a successful outcome. For more information, see Power Pivot Data Refresh with SharePoint 2010.

> **NOTE**
>
> For SQL Server 2017Power Pivot for SharePoint 2013, the data providers are installed when you run the **spPowerPivot.msi** installer and Power Pivot for SharePoint 2013 Configuration tool. For more information, see Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2013).

Change Application Pool and Service Identities in SharePoint

The Power Pivot Configuration tool provisions farm features, applications, and services to run under a single account. This simplifies installation, but it does not result in a deployment that meets the security requirements of a SharePoint farm. To create a more robust deployment, change the application pools and service identities to run under different accounts after setup is complete. For more information, see Configure Power Pivot Service Accounts.

Create Additional Trusted Sites in Excel Services

You can add trusted sites in Excel Services to vary permissions and configuration settings on sites that provide Excel workbooks and Power Pivot data. For more information, see Create a trusted location for Power Pivot sites in Central Administration.

Build a Power Pivot Workbook

After you have the server components installed in a farm, you can create your first Excel 2013 workbook that uses embedded Power Pivot data, and then publish it to a SharePoint library. Alternatively, you can upload or publish a sample Power Pivot workbook to verify Power Pivot data access in SharePoint. For more information, see the following:

- Power Pivot Help (https://support.office.com/en-us/article/Power-Pivot-Help-241aac41-92e3-4e46-ae58-2f2cd7dbcf4f).
- Start Power Pivot in Excel 2013 add-in (http://office.microsoft.com/excel-help/start-powerpivot-in-excel-2013-add-in-HA102837097.aspx?CTT=5&origin=HA102837110).

Add additional Analysis Services servers in SharePoint mode

Over time, if you determine that additional data storage and processing capability is needed, you can add additional servers running Analysis Services in SharePoint mode to the farm. For SQL Server 2017Power Pivot for SharePoint 2013, you install new Analysis Services servers in SharePoint mode and then configure Excel Services. For more information, see the "Beyond the single-server installation" section of Install Analysis Services in Power Pivot Mode.

Additional Resources

Submit feedback and contact information through SQL Server Feedback.

See Also

Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2013)
Power Pivot Configuration Tools
Power Pivot Server Administration and Configuration in Central Administration
Upgrade Workbooks and Scheduled Data Refresh (SharePoint 2013)

Power Pivot Availability and Disaster Recovery

12/6/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Availability and disaster recovery plans for Power Pivot for SharePoint depend primarily on the design of your SharePoint farm, the amount of downtime acceptable for different components, and the tools and best practices you implement for SharePoint availability. This topic summarizes technologies and includes example topology diagrams to consider when planning availability and disaster recovery for a SQL Server 2014 (12.x) Power Pivot for SharePoint deployment.

Applies to: SharePoint 2013 | SharePoint 2010

In this topic:

- Example SharePoint 2013 topology for Power Pivot high availability
- Example SharePoint 2010 topology for Power Pivot high availability
- PowerPivot service application database and SQL Server availability and recovery technologies
- Links to more information

Example SharePoint 2013 topology for Power Pivot high availability

In a Power Pivot for SharePoint 2013 deployment there is more flexibility in how you design Power Pivot availability. In SharePoint 2013, the Analysis Services instance deployed in SharePoint mode, also referred to as the Power Pivot for SharePoint server, runs outside the SharePoint farm and can be installed on separate servers. Each instance of Analysis Services in SharePoint mode is registered with Excel Services. The Power Pivot shared service and service application run on SharePoint application servers.

The following diagram illustrates an example Power Pivot for SharePoint 2013 deployment. This example supports good availability of the Power Pivot for SharePoint services and assumes the databases are backed up on a regular basis.

- **(1)** The Web front-end servers. Use the Power Pivot for SharePoint 2013 Add-in to install the data providers on each server. For more information, see Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2013).
- **(2)** The Power Pivot shared service runs **on each** application server and allows the service application to run **across** application servers. Therefore if a single application server goes offline, the Power Pivot for SharePoint application will still be available.
- **(3)** Excel Calculation Services runs one each application server and allows the service application to run across application servers. Therefore if a single application server goes offline, Excel Calculation Services will still be available.
- **(4)** and **(6)** Instances of Analysis Services ins SharePoint mode run on servers outside the SharePoint farm, this includes the Windows Service **SQL Server Analysis Services (Power Pivot)**. Each of these instances is registered with Excel Services **(3)**. Excel Services manages load balancing of requests to the Power Pivot for SharePoint servers. The Power Pivot for SharePoint 2013 architecture enables you to have multiple servers for Power Pivot for SharePoint so you can easily add more instances as needed. For more information, see Manage Excel Services data model settings (SharePoint Server 2013).

- **(5)** The SQL Server databases used for content, configuration, and application databases. This includes the Power Pivot service application database. Your DR plan should include the database layer. In this design the databases run on the same server as **(4)** one of the Power Pivot for SharePoint instances. **(4)** and **(5)** could also be on different servers.
- **(7)** Some form of SQL Server database backup or redundancy.

Example SharePoint 2010 topology for Power Pivot high availability

The Power Pivot for SharePoint 2010 architecture requires all Power Pivot components run on the same SharePoint application servers. This includes the Analysis Services instance deployed in SharePoint mode and two shared services compared to only one in a SharePoint 2013 deployment.

The following diagram illustrates an example Power Pivot for SharePoint 2010 deployment. This example supports good availability of the Power Pivot for SharePoint services and assumes the databases are backed up on a regular basis.

- **(1)** The Web front-end servers. Install the data providers on each server. For more information, see Install the Analysis Services OLE DB Provider on SharePoint Servers.

- **(2)** The two Power Pivot shared services and **(4)** the Windows Service **SQL Server Analysis Services (Power Pivot)** are installed on the SharePoint application servers.

  The Power Pivot System Service runs **on each** application server and allows the service application to run **across** application servers. If a single application server goes offline, the Power Pivot for SharePoint service application will still be available.

  To increase Power Pivot capacity in a SharePoint 2010, deploy more SharePoint application servers running the Power Pivot System Service.

- **(3)** Excel Calculation Services runs on each application server and allows the service application to run across application servers. Therefore if a single application server goes offline, Excel Calculation Services will still be available.

- **(5)** The SQL Server databases used for content, configuration, and application databases. This includes the Power Pivot service application database. Your DR plan should include the database layer.

- **(6)** Some form of SQL Server database backup or redundancy.

PowerPivot service application database and SQL Server availability and recovery technologies

Include the Power Pivot for SharePoint service application database in you SharePoint high availability planning. The default name of the database is `DefaultPowerPivotServiceApplicationDB-<GUID>`. The following is a summary of SQL Server availability technologies and recommendations when used with the Power Pivot for SharePoint database. For more information, see Supported high availability and disaster recovery options for SharePoint databases (SharePoint 2013).

	COMMENTS
SQL Server 2008 R2 and SQL Server 2012 (11.x) synchronous mirroring in a farm for availability.	Supported but not recommended. The recommendation is to use Always On in Synchronous - commit mode.
SQL Server Always On availability groups in Synchronous-Commit mode	Supported and recommended.
SQL Server 2008 R2 asynchronous mirroring or log-shipping to another farm for disaster recovery.	Supported.
SQL Server Always On availability groups with asynchronous-commit for disaster recovery	Supported

- Always On availability groups
- SQL Server Log Shipping

For more information on how to plan a cold standby scenario with Power Pivot for SharePoint, see PowerPivot Disaster Recovery.

Verification

For guidance and scripts to help you verify a Power Pivot for SharePoint deployment before and after a disaster recovery cycle, see CheckList: Use PowerShell to Verify Power Pivot for SharePoint.

Links to more information

- Supported high availability and disaster recovery options for SharePoint databases (SharePoint 2013)
- Plan for disaster recovery (SharePoint Server 2010)
- Microsoft® SQL Server Backup to Microsoft Windows® Azure® Tool

Community content

- Managing Service Instances on SharePoint 2013

Power Pivot Authentication and Authorization

11/27/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A Power Pivot for SharePoint deployment that runs within a SharePoint 2010 farm uses the authentication subsystem and authorization model provided by SharePoint servers. SharePoint security infrastructure extends to Power Pivot content and operations because all Power Pivot-related content is stored in SharePoint content databases and all Power Pivot-related operations are performed by Power Pivot shared services in the farm. Users who request a workbook that contains Power Pivot data are authenticated using a SharePoint user identity that is based on their Windows user identity. View permissions on the workbook determine whether the request is granted or denied.

Because integration with Excel Services is required for self-service data analytics, securing a Power Pivot server requires that you also understand Excel Services security. When a user queries a PivotTable that has a data connection to Power Pivot data, Excel Services forwards a data connection request to a Power Pivot server in the farm to load the data. This interaction between the servers requires that you understand how to configure security settings for both servers.

Click the following links to read specific sections in this topic:

Windows Authentication Using Classic Mode Sign-in Requirement

Power Pivot Operations Requiring User Authorization

SharePoint Permissions for Power Pivot Data Access

Excel Services security considerations for Power Pivot workbooks

Windows Authentication Using Classic Mode Sign-in Requirement

Power Pivot for SharePoint supports a reduced set of the authentication options that are available in SharePoint. Of the available authentication options, only Windows authentication is supported for a Power Pivot for SharePoint deployment. In addition, the web application through which sign-in occurs must be configured for classic mode.

Windows authentication is required because the Analysis Services data engine in a Power Pivot for SharePoint deployment supports only Windows authentication. Excel Services establishes connections to Analysis Services via the MSOLAP OLE DB provider using a Windows user identity that was authenticated via NTLM or the Kerberos protocol.

The second requirement, classic mode authentication on the web application, is required to ensure the operability of the Power Pivot web service. The web service is a component that runs on a Web front-end and provides HTTP redirection to a Power Pivot for SharePoint server in the farm. While the web service is claims aware for service-to-service communications, it is not claims aware for the data connection requests that it routes to a Power Pivot shared service in the farm. Requests to load Power Pivot data are only supported for authenticated connections that come from IIS using a Windows identity. Classic mode sign-in on the web application is what enables a successful connection from the Power Pivot web service to Power Pivot shared services in the farm.

Although classic mode sign-in is not required for the more common data access scenario (where Power Pivot data is extracted from the same Excel workbook that renders it) do not attempt to use Power Pivot for SharePoint with SharePoint web applications that are configured to use other authentication providers. Doing so will result in a connection failure whenever users try to connect to Power Pivot workbooks as an external data source.

Without classic mode sign-in, the following types of requests that are handled by Power Pivot Web service will fail:

- Any request for Power Pivot data that originates from outside the farm (for example, creating a report in Report Designer or Report Builder, where the data source is a SharePoint URL to a Power Pivot workbook)
- In-farm requests from a client application or report that uses the Power Pivot workbook as an external data source (for example, creating a workbook in the Excel desktop application, using as your data source a second published Excel workbook containing Power Pivot data)

How to check the authentication provider for your application

When creating new web applications, be sure to select the **Classic mode authentication** option in the Create New Web Application page.

For existing web applications, use the following instructions to verify the web application is configured to use Windows authentication.

1. In Central Administration, in Application Management, click **Manage web applications**.
2. Select the web application.
3. Click **Authentication Providers**.
4. Verify that you have one provider for each zone, and the Default zone is set to Windows.

Power Pivot Operations Requiring User Authorization

SharePoint authorization is used exclusively for all levels of access to Power Pivot query and data processing.

The Analysis Services role-based authorization model is not supported. There is no role-based authorization for Power Pivot data at the cell, row, or table level. You cannot secure different parts of the workbook to grant or deny access to sensitive data within it for specific users. Embedded Power Pivot data is wholly available to users who have View permissions on the Excel workbook in a SharePoint library.

Power Pivot for SharePoint will impersonate a SharePoint user in the following cases:

- Queries to PivotTables or PivotCharts that have data connections to a Power Pivot database, where a Power Pivot service application establishes connections on behalf of a user to a specific Power Pivot shared service instance that processes the data.
- Loading Power Pivot data from cache or a library if the data is not otherwise available. If a data connection request is made for Power Pivot data that is not already loaded in the system, the Analysis Services service instance will use the identity of the SharePoint user to retrieve the data source from a content library and load it into memory.
- Data refresh operations that save an updated copy of the data source to the workbook in a content library. In this case, an actual log on operation is performed using the user name and password that is retrieved from a target application in Secure Store Service. Credentials can be the Power Pivot unattended data refresh account, or credentials that were stored with the data refresh schedule when it was created. For more information, see Configure Stored Credentials for Power Pivot Data Refresh (Power Pivot for SharePoint) and Configure the Power Pivot Unattended Data Refresh Account (Power Pivot for SharePoint).

SharePoint Permissions for Power Pivot Data Access

Publishing, managing, and securing a Power Pivot workbook is supported solely through SharePoint integration. SharePoint servers provide authentication and authorization subsystems that ensure legitimate access to data. There are no supported scenarios for securely deploying a Power Pivot workbook outside of a SharePoint farm.

User access to Power Pivot data is read-only on the server through View permissions or higher. Contribute permissions allow for adding and editing the file. Changes to Power Pivot data require that you download the workbook to an Excel desktop application that has Power Pivot for Excel installed. Contribute permissions on the file will determine whether the user can download the file locally and then save changes back to SharePoint.

As such, Contribute and View Only levels of permission define the effective set of permissions for user access to Power Pivot data. Other permission levels work to the extent that they have the same permissions as Contribute and View Only (For example, because Read includes View Only permissions, a user who is assigned to Read will have same level of access as View Only).

The following table summarizes the permission levels that determine access to Power Pivot data and server operations:

PERMISSION LEVEL	ALLOWS THESE TASKS
Farm or service administrator	Install, enable, and configure services and applications. Use Power Pivot Management Dashboard and view administrative reports.
Full control	Activate Power Pivot feature integration at the site collection level. Create a Power Pivot Gallery library. Create a data feed library.
Contribute	Add, edit, delete, and download Power Pivot workbooks. Configure data refresh. Create new workbooks and reports based on Power Pivot workbooks on a SharePoint site. Create data service documents in a data feed library
Read	Access Power Pivot workbooks as an external data source, where the workbook URL is explicitly entered in a connection dialog box (for example, in Excel's Data Connection Wizard).
View Only	View Power Pivot workbooks. View data refresh history. Connect a local workbook to a Power Pivot workbook on a SharePoint site, to repurpose its data in other ways. Download a snapshot of the workbook. The snapshot is a static copy of the data, without slicers, filters, formulas, or data connections. The contents of the snapshot are similar to copying cell values from the browser window.

Excel Services security considerations for Power Pivot workbooks

Power Pivot server-side query processing is tightly coupled with Excel services. Product integration begins at the document level, in that Power Pivot workbooks are Excel workbook (.xlsx) files that either contain or reference Power Pivot data. There is no separate file extension for a Power Pivot workbook.

When a Power Pivot workbook is opened on a SharePoint site, Excel Services reads the embedded Power Pivot

data connection string and forwards the request to the local SQL Server Analysis Services OLE DB provider. The provider then passes the connection information to a Power Pivot server in the farm. In order for requests to flow seamlessly between the two servers, Excel Services must be configured to use settings that are required by Power Pivot for SharePoint.

In Excel Services, security-related configuration settings are specified on trusted locations, trusted data providers, and trusted data connection libraries. The following table describes the settings that enable or enhance Power Pivot data access. If a setting is not listed here, it has no effect on Power Pivot server connections. For instructions on how to specify these settings step by step, see the section "Enable Excel Services" in Initial Configuration (Power Pivot for SharePoint).

> **NOTE**
> Most security-related settings apply to trusted locations. If you want to preserve default values or use different values for different sites, you can create an additional trusted location for sites that contain Power Pivot data, and then configure the following settings for just that site. For more information, see Create a trusted location for Power Pivot sites in Central Administration.

AREA	SETTING	DESCRIPTION
Web application	Windows authentication provider	Power Pivot converts a claims token that it gets from Excel Services to a Windows user identity. Any web application that uses Excel Services as a resource must be configured to use the Windows authentication provider.
Trusted location	Location Type	This value must be set to **Microsoft SharePoint Foundation**. Power Pivot servers retrieve a copy of the .xlsx file and load it on an Analysis Services server in the farm. The server can only retrieve .xlsx files from a content library.
	Allow External Data	This value must be set to **Trusted data connection libraries and embedded**. Power Pivot data connections are embedded in the workbook. If you disallow embedded connections, users can view the Power Pivot cache, but they will not be able to interact with the Power Pivot data.
	Warn on Refresh	This value should be disabled if you are using Power Pivot Gallery to store workbooks and reports. Power Pivot Gallery includes a document preview feature that works best if both refresh on open and Warn on Refresh are turned off.

AREA	SETTING	DESCRIPTION
Trusted data providers	MSOLAP.4 MSOLAP.5	MSOLAP.4 is included by default, but Power Pivot data access requires that the MSOLAP.4 provider be the SQL Server 2008 R2 version. MSOLAP.5 is installed with the SQL Server 2017 version of Power Pivot for SharePoint. Do not remove these providers from the trusted data provider list. In some cases, you might need to install additional copies of this provider on other SharePoint servers in your farm. For more information, see Install the Analysis Services OLE DB Provider on SharePoint Servers.
Trusted data connection libraries	Optional.	You can use Office Data Connection (.odc) files in Power Pivot workbooks. If you use .odc files to provide connection information to local Power Pivot workbooks, you can add the same .odc files to this library.
User defined function assembly	Not applicable.	Power Pivot for SharePoint ignores user-defined function assemblies that you build deploy for Excel Services. If you rely on user-defined assemblies for a specific behavior, be aware that Power Pivot query processing will not use the user-defined functions you created.

See Also

Configure Power Pivot Service Accounts
Configure the Power Pivot Unattended Data Refresh Account (Power Pivot for SharePoint)
Create a trusted location for Power Pivot sites in Central Administration
Power Pivot Securtiy Architecture

Configure Power Pivot Health Rules

11/27/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Power Pivot for SharePoint includes SharePoint health rules that help you monitor and remedy server availability and configuration problems. Health rules that apply to Power Pivot for SharePoint appear in the Review rule definitions page.

Health rules provide early detection of server problems that could eventually result in service disruptions. Power Pivot for SharePoint provides a number of rules to help you identify and fix problems before they impact your users. You can customize many of these rules to fit the unique characteristics of your deployment. For example, if you want more time to address warnings about disk space, you could raise the available disk space percentage from 5% to 10% so that you get the warning earlier.

Rules that can be customized are those that report on resource consumption or server availability. Customization is helpful in these areas because underlying system capacity varies widely across different servers and deployment topologies. In contrast, no customization is available for rules that identify server configuration or security issues. Those rules are intended to be applied uniformly across all installations.

Applies to: SharePoint 2013 | SharePoint 2010

Note: Health rule settings are configured separately for the SQL Server Analysis Services instance and the Power Pivot service application. Use the instructions in this topic to configure health rules for each service. For a SharePoint 2013 deployment, Power Pivot for SharePoint only uses the service application. Therefore Power Pivot for SharePoint installs different sets of health rules for different versions of SharePoint. See the "version" column in the topic Health Rules Reference (Power Pivot for SharePoint), or you can run the following Windows PowerShell command to see the installed rules.

```
Get-SPHealthAnalysisRule | select name, enabled, summary | where {$_.summary -like "*power*"}  | format-table
-property * -autosize | out-default
```

In this topic:

View Power Pivot health rules

Configure health rules used to evaluate server stability (SQL Server Analysis Services)

Configure health rules used to evaluate application stability (Power Pivot Service Application)

Prerequisites

You must be a service application administrator to change configuration properties of the Analysis Services instance and of the Power Pivot service application.

View Power Pivot health rules

1. In SharePoint Central Administration, click **Monitoring**, and in the **Health Analyzer** section, click **Review rule definitions**.
2. In the Configuration section, find the rules that have the **Power Pivot:** prefix. All Power Pivot-related

health rules have this prefix to help you distinguish them from the built-in SharePoint rules.

These rules will appear in the **Review problems and solutions** page when problems are detected.

If you suspect a problem that you want to investigate immediately, you can run a rule check manually to find out if there is an issue.

To do this, click the rule to open its rule definition, and then click **Run Now** in the ribbon. Click **Close** to go back to the **Review problems and solutions** page to view the report. If the rule detected a problem, a warning or error will be reported on the page. In some cases, it can take a few minutes before the error or warning will appear.

Configure health rules used to evaluate server stability (SQL Server Analysis Services)

The Analysis Services instance includes health rules that detect problems at the system level (CPU, memory, and disk space used for caching purposes). Use the following instructions to modify thresholds that trigger specific health rules.

1. In SharePoint Central Administration, in the **System Settings** section, click **Manage services on server**.
2. At the top of the page, select the server in your SharePoint farm that has an instance of Analysis Services (in the following illustration, the server name is AW-SRV033). **SQL Server Analysis Services** will appear in the list of services.

3. Click **SQL Server Analysis Services**.
4. In the service property pages, in Health Rule Settings, modify the following settings:

   Insufficient CPU Resource Allocation (default is 80%)
   This health rule is triggered if the CPU resources used by Analysis Services server process (msmdsrv.exe) remains at or above 80% over a 4 hour period (as specified through the Data Collection Interval setting).

   This configuration setting corresponds to the following rule definition on the **Review problems and solutions** page: **Power Pivot: Analysis Services does not have sufficient CPU resources to perform requested operations.**

   Insufficient CPU Resources on the System (default is 90%)
   This health rule is triggered if CPU resources for the server remain at or above 90% over a 4 hour period (as specified through the Data Collection Interval setting). Overall CPU utilization is measured as part of

the health-based load balancing algorithm that monitors CPU usage as a measure of server health.

This configuration setting corresponds to the following rule definition on the **Review problems and solutions** page: **Power Pivot: Overall CPU usage is too high.**

Insufficient Memory Threshold (default is 5%)
On a SharePoint application server, a SQL Server Analysis Services instance should always have a small amount of memory in reserve that is always unused. Because the server is memory-bound for the majority of its operations, the server runs best if it does not run all the way to the upper limit. The 5% of unused memory is calculated as a percentage of memory allocated to Analysis Services. For example, if you have 200 GB of total memory, and Analysis Services is allocated 80% of that (or 160 GB), then the 5% of unused memory is 5% of 160 GB (or 8 GB).

This configuration setting corresponds to the following rule definition on the **Review problems and solutions** page: **Power Pivot: Analysis Services does not have sufficient memory to perform requested operations.**

Maximum Number of Connections (default is 100)
This health rule is triggered if the number of connections to the Analysis Services instance remains at or above 100 connections over a 4 hour period (as specified through the Data Collection Interval setting). This default value is arbitrary (it is not based on the hardware specifications of your server or on user activity) so you might raise or lower the value depending on the server capacity and user activity in your environment.

This configuration setting corresponds to the following rule definition on the **Review problems and solutions** page: **Power Pivot: The high number of connections indicates that more servers should be deployed to handle the current load.**

Insufficient Disk Space (default is 5%)
Disk space is used to cache Power Pivot data each time a database is requested. This rule lets you know when disk space is running low. By default, this health rule is triggered when disk space is less than 5% on the disk drive where the backup folder is located. For more information about disk usage, see Configure Disk Space Usage (Power Pivot for SharePoint).

This configuration setting corresponds to the following rule definition on the **Review problems and solutions** page: **Power Pivot: Disk space is running low on the drive where Power Pivot data is cached.**

Data Collection Interval (in hours)
You can specify the data collection period used for calculating the numbers used for triggering health rules. Although the system is monitored constantly, the thresholds used to trigger health rule warnings are calculated using data that was generated over a predefined interval. The default interval is 4 hours. The server retrieves system and usage data collected over the previous 4 hours to evaluate the number of user connections, disk space usage, and CPU and memory utilization rates.

Configure health rules used to evaluate application stability (Power Pivot Service Application)

1. In Central Administration, in Application Management, click **Manage service applications**.
2. In the Service Applications page, click **Default Power Pivot Service Application**.

3. The Power Pivot Management Dashboard appears. Click the **Configure service application settings** in the **Actions** list to open the service application settings page.

4. In Health Rule Settings, modify the following settings:

   Load to Connection Ratio (default is 20%)
   This health rule is triggered if the number of load events is high relative to the number of connection events, signaling that the server might be unloading databases too quickly, or that cache reduction settings are too aggressive.

   This configuration setting corresponds to the following rule definition on the **Review problems and solutions** page: **Power Pivot: The ratio of load events to connections is too high.**

   Data Collection Interval (default is 4 hours)
   You can specify the data collection period used for calculating the numbers used for triggering health rules. Although the system is monitored constantly, the thresholds used to trigger health rule warnings are calculated using data that was generated over a predefined interval. The default interval is 4 hours. The server retrieves system and usage data collected over the previous 4 hours to evaluate the load to collection ratio.

   Check for Updates to Power Pivot Management Dashboard.xlsx (default is 5 days)
   The Power Pivot Management Dashboard.xlsx file is a data source used by reports in Power Pivot Management Dashboard. In a default server configuration, the .xlsx file is refreshed daily, using usage data collected by SharePoint and the Power Pivot System Service. In event the file is not updated, a health rule reports it as a problem. By default, the rule is triggered if the timestamp of the file has not changed for 5 days.

   For more information about usage data collection, see Configure Usage Data Collection for (Power Pivot

for SharePoint.

This configuration setting corresponds to the following rule definition on the **Review problems and solutions** page: **Power Pivot: Usage data is not getting updated at the expected frequency.**

See Also

Configure Disk Space Usage (Power Pivot for SharePoint)
Power Pivot Management Dashboard and Usage Data

Health Rules Reference (Power Pivot for SharePoint)

11/27/2018 • 12 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This reference topic describes the SharePoint health rules that are added by a Power Pivot for SharePoint installation. These rules are used to report problems with server health, availability, or configuration of a Power Pivot for SharePoint service application or its associated Analysis Services instance.

The following table lists the rules in the order in which they appear on the Health Analyzer Rule Definitions page in SharePoint Central Administration. Configurable rules are those for which you can change the thresholds at which the rule is triggered. For more information, see Configure Power Pivot Health Rules. Auto Repair indicates that there is a built-in remedy that you can click from within the Problem reports page to resolve the issue.

Applies to: SharePoint 2013 | SharePoint 2010

Note: Power Pivot for SharePoint installs different sets of health rules for different versions of SharePoint. See the "version" column in the table below, or you can run the following Windows PowerShell command to see the installed rules.

```
Get-SPHealthAnalysisRule | select name, enabled, summary | where {$_.summary -like "*power*"}  | format-table
-property * -autosize | out-default
```

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
Power Pivot: The Analysis Services OLE DB provider is not installed on this computer.	No	No	SharePoint 2010	The Analysis Services OLE DB provider is either not installed on the server or it is the wrong version. This rule appears when your SharePoint farm includes instances of Excel Services on application servers that do not have Power Pivot for SharePoint. The rule warns you that the Analysis Services OLE DB provider used by Excel Services to connect to Power Pivot data is not installed. To resolve this issue, install the OLE DB provider on each Excel Services server that does not have the Analysis Services OLE DB provider. You can download and install the Analysis Services OLE DB provider from the Microsoft Download center. For more information, see Install the Analysis Services OLE DB Provider on SharePoint Servers.

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
Power Pivot: Registry settings for Microsoft.AnalysisServices.ChannelTransport.dll are not valid after you install the SQL Server 2008 R2 version of the MSOLAP provider on this computer.	No	Yes	SharePoint 2010	This is a server configuration issue. Most likely, the ChannelTransport.dll is not registered in the global assembly. Run the automatic repair for this rule to register the .dll on each server that has an installation of Power Pivot for SharePoint. Alternatively, you can run regasm.exe manually to register the file. If the SharePoint timer service is not running as local administrator, manual registration might be required. Failure to update the registry settings results in slow server communication between Excel Services and Power Pivot System Service, and can result in connection failures in certain security configurations.
Power Pivot: Power Pivot service application does not have permission to complete operation.	No	No	SharePoint 2010	This rule checks whether the Power Pivot service application identity is database owner of the Power Pivot server application database and has administrative permissions on the local SQL Server Analysis Services instance. These permissions are granted automatically during installation and deployment, but if this step failed to complete, this health rule will occur.

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
Power Pivot: The Power Pivot service application identity should not be a member of the local Administrators group.	No	No	SharePoint 2010	This is a best practice that improves the overall security of your deployment. If you configured the Power Pivot service application to run under an account that belongs to the local Administrator group, you should change the service account to one that does not belong to that group. The recommendation is to use a least-privileged, dedicated account for each service. Doing so provides service isolation and makes it easier to audit logins. For more information about changing the service account, see Configure Power Pivot Service Accounts.
Power Pivot: The Analysis Services instance runs in Tabular mode, but the configuration setting that specifies this mode is turned off.	No	No	SharePoint 2010	This rule checks whether the SQL Server Analysis Services instance in a Power Pivot for SharePoint installation has the **DeploymentMode** server property set to 1. If the property is set to another value, or if the SharePoint Timer service that runs the rule checker does not have permission to open the file, this rule will fail. For more information about the deployment mode property, see Determine the Server Mode of an Analysis Services Instance.

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
Power Pivot: The Power Pivot Data Refresh Timer Job is disabled.	No	No	SharePoint 2013 SharePoint 2010	Check the timer job settings to verify the timer job is enabled. If you are not using the Power Pivot data refresh feature, you can ignore this rule. For more information, see Power Pivot Data Refresh with SharePoint 2010.
Power Pivot: The SQL Server Analysis Services (Power Pivot) service account information that is managed by the SQL Server Configuration Manager is different from the account information that is managed by Central Administration.	No	No	SharePoint 2010	This rule checks whether the service account information in SQL Server Configuration Manager is identical to the managed account information in Central Administration for the same Analysis Services instance. If the accounts are different, an entry is added to the Problem and Resolution report so that you can change the service account information in SQL Server Configuration Manager back to the account specified in Central Administration. SQL Server Configuration Manager is not a supported tool for changing a service account username or password in a Power Pivot for SharePoint installation. Using Central Administration enables the use of the managed accounts feature in SharePoint. More importantly, if your farm includes multiple Power Pivot for SharePoint servers, having inconsistent service account settings can disrupt processing and query operations on the server that has incorrect service information.

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
				On a single server, Power Pivot workbooks will function temporarily when this rule is triggered, but it is advised that you fix the problem as soon as possible. Database and file system permissions are updated using the account information specified in Central Administration.
Power Pivot: The deployed farm solution is not up-to-date.	No	Yes	SharePoint 2010	A Power Pivot for SharePoint installation uses a farm level solution and a web application level solution to install its features. This rule indicates that the farm solution is not current relative to the version or the server or possibly the web solution. Most likely, this is server deployment problem. To remedy this problem, consider running SQL Server Setup to repair one of the Power Pivot for SharePoint installations in your farm. For more information about solutions in a Power Pivot for SharePoint installation, see Deploy Power Pivot Solutions to SharePoint.

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
Power Pivot: Overall CPU usage is too high.	Yes	No	SharePoint 2010	This rule reports on CPU consumption at the system level. Overall CPU usage is monitored because the Power Pivot System Service uses it as a measure of server health, for health-based load balancing among multiple Power Pivot for SharePoint servers in a farm. Consider adding another application server to the farm, and moving CPU intensive applications to that server.
Power Pivot: Analysis Services does not have sufficient CPU resources to perform requested operations.	Yes	No	SharePoint 2010	The amount of CPU resources available to the Analysis Services process (msmdsrv.exe) is not sufficient for the level of activity on this server. Consider adding another Power Pivot for SharePoint server to the farm. For more information, see Deployment Checklist: Scale-out by adding Power Pivot Servers to a SharePoint 2010 farm.

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
Power Pivot: Analysis Services does not have sufficient memory to perform requested operations.	No	No	SharePoint 2010	This rule is triggered when there is only 5% available memory left to Analysis Services. On a SharePoint application server, a SQL Server Analysis Services instance should always have a small amount of memory in reserve that is always unused. Because the server is memory-bound for the majority of its operations, the server runs best if it does not run all the way to the upper limit. By default, insufficient memory warnings occur when available memory is down to 5%. You can change this value to be higher or lower by adjusting settings on the Analysis Services instance. For more information, see Configure Power Pivot Health Rules. The 5% of unused memory is calculated as a percentage of memory allocated to Analysis Services. For example, if you have 200 GB of total memory, and Analysis Services is allocated 80% of that (or 160 GB), then the 5% of unused memory is 5% of 160 GB (or 8 GB).

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
Power Pivot: The high number of connections indicates that more servers should be deployed to handle the current load.	Yes	No	SharePoint 2010	By default, this health rule is triggered when the number of distinct user connections exceeds 100. This default value is arbitrary (it is not based on the hardware specifications of your server or on user activity) so you might raise or lower the value depending on the server capacity and user activity in your environment. For more information, see Configure Power Pivot Health Rules.
Power Pivot: The ratio of load events to connections is too high.	Yes	No	SharePoint 2013 SharePoint 2010	By default, this health rule is triggered when the percentage of load events to connection events exceeds 50% over the entire data collection period (by default, 4 hours). A ratio this high indicates a very high number of connections to unique workbooks, or cache reduction settings that are too aggressive (where workbooks are quickly unloaded and removed from the system, while requests for that data are still active). To avoid counting false positives, there must be at least 20 connections per 4 hour period before the ratio can be calculated. You can base this health rule on a different ratio. For more information, see Configure Power Pivot Health Rules. For more information about configuring the cache, see Configure Disk Space Usage (Power Pivot for SharePoint).

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
Power Pivot: One or more minidump files were found in the Logs directory, indicating a program crash.	No	No	SharePoint 2013 SharePoint 2010	Minidump files are generated during a program crash to capture information about Power Pivot service application state just prior to the crash. This information can be sent to Microsoft and used for troubleshooting. This rule is triggered when .dmp files are detected on the server. The rule provides a link to the file, which can be found in the \OLAP\Log folder of the Power Pivot for SharePoint instance. Note that you cannot use a text editor to view the contents of the file. Viewing a minidump file requires that you download and install a separate debugging tool. For more information, see Debugging Tools for Windows.
Power Pivot: Disk space is running low on the drive where Power Pivot data is cached.	Yes	No	SharePoint 2010	By default, this health rule is triggered when disk space is less than 5% on the disk drive where the backup folder is located. For more information about setting this percentage, see Configure Power Pivot Health Rules. For more information about disk usage, see Configure Disk Space Usage (Power Pivot for SharePoint).

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
Power Pivot: Usage data is not getting updated at the expected frequency.	Yes	No	SharePoint 2013 SharePoint 2010	Power Pivot for SharePoint uses the built-in usage data collection system to gather metrics about connections, data refresh, and query response times. It stores this usage data in the Power Pivot service application database, which in turn updates a Power Pivot workbook (Power Pivot Management Data.xlsx) that provides data to reports in the Power Pivot Management Dashboard. This rule indicates that usage data is not getting moved to the Power Pivot Management Data.xlsx file with sufficient frequency. The rule uses the timestamp on the .xlsx file as proof that the file is updated. If there are other problems in the usage data collection system that undermines the accuracy of the data, this rule will not detect it. To troubleshoot this error, check the timer jobs to verify they are running. For more information about usage data collection, see Configure Usage Data Collection for (Power Pivot for SharePoint.
Power Pivot: Midtier process account should have 'Full Read' permission on all associated SPWebApplications.	No	Yes	SharePoint 2013 SharePoint 2010	The Power Pivot service application identity must have **Full Read** permissions in order to access the SharePoint content databases on behalf of users who have View Only permissions on a document.

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
				To determine which account is used as the Power Pivot service application identity, open the **Configure service accounts** page in Central Administration. Most likely, the service application runs in either the **SharePoint Web Services System** service application pool or in a dedicated application pool. Although this rule provides a **Repair Automatically** option, you will get better results if you grant the permissions manually: 1) In Central Administration, click **Manage web applications**. 2) Select a web site, and then click **User Policy**. 3) Click **Add Users**. 4) Select (All zones) and click **Next**. 5) In Users, enter the Power Pivot service application identity, and then click the **Full Read** checkbox. Click **6) Finish**. 6) Verify the repair. In Monitoring, click **Review rule definitions**. Find and then open the Power Pivot rule. Click **Run Now**. Go back to **Review problems and solutions** to verify the rule no longer appears.

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
Power Pivot: Secondary Logon service (seclogon) is disabled	No	No	SharePoint 2013 SharePoint 2010	The Secondary Logon service is used to generate thumbnail images of Power Pivot workbooks in the Power Pivot Gallery. By default, the Secondary Logon service is set to manual startup. If the service is disabled, thumbnail generation will fail. Additionally, the ULS logs will contain the following error: "The error 1058 can have as a root cause the fact the Windows service "Secondary Logon" is disabled." To check service configuration, use the Services console application to find Secondary Logon and change its **Startup Type** to **Manual**. If you cannot enable the service, your organization might have a group policy that disables it. Check with an administrator to determine whether this is the case. After you enable the service, thumbnail or snapshot images will refresh over time. Optionally, you can force a refresh by restarting the service and opening and then resaving the property pages of a specific report. For more information, see How to Use Power Pivot Gallery.

RULE	CONFIGURABLE	AUTO REPAIR	VERSION	DESCRIPTION
Power Pivot: ADOMD.NET is not installed on a standalone WFE that is configured for central admin	No	No	SharePoint 2013 SharePoint 2010	ADOMD.NET is an Analysis Services client library that supports connections to an Analysis Services database. In a deployment of Power Pivot for SharePoint, ADOMD.NET provides access to the built-in reports in the Power Pivot management dashboard in Central Administration. Built-in reports are actually Power Pivot workbooks that contain embedded Analysis Services data. The management dashboard uses ADOMD.NET to send a connection request to the server that loads data contained in the workbook. On topologies that include Central Administration running on a standalone web front end server, you must install ADOMD.NET manually if you want to view these reports in the management dashboard. For more information, see Install ADOMD.NET on Web Front-End Servers Running Central Administration.

Power Pivot Management Dashboard and Usage Data

1/28/2019 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Power Pivot Management Dashboard is a collection of predefined reports and web parts in SharePoint Central Administration that help you administer a SQL Server Power Pivot for SharePoint deployment. The Management Dashboard provides information related to server health, workbook activity, and data refresh. The dashboard uses data from the SharePoint usage data collection.

Prerequisites

You must be a service administrator to open Power Pivot Management Dashboard for a Power Pivot service application that you manage.

Overview of the sections of the Dashboard

Power Pivot Management Dashboard contains Web Parts and embedded reports that drill down into specific information categories. The following list describes each part of the dashboard:

DASHBOARD	DESCRIPTION
Infrastructure - Server Health	Shows trends in CPU usage, memory consumption, and query response times over time so that you can assess whether system resources are nearing maximum capacity or are under utilized.
Actions	Contains links to other pages in Central Administration including the current service application, a list of service applications, and usage logging.
Workbook Activity - Chart	Reports on frequency of data access. You can learn how often connections to Power Pivot data sources occur on a daily or weekly basis.
Workbook Activity - Lists	Reports on frequency of data access. You can learn how often connections to Power Pivot data sources occur on a daily or weekly basis.
Data Refresh - Recent Activity	Reports on the status of data refresh jobs, including jobs that failed to run. This report provides a composite view into data refresh operations at the application level. Administrators can see at a glance the number of data refresh jobs that are defined for the entire Power Pivot service application.
Data Refresh - Recent Failures	Lists the Power Pivot workbooks that did not complete data refresh successfully.
Reports	Contains links to reports that you can open in Excel.

Open Power Pivot Management Dashboard

The dashboard shows information for one Power Pivot service application at a time. You can open the management dashboard from two different locations.

Open the dashboard from General Application Settings

1. In Central Administration, in the **General Application Settings** group, click **Power Pivot Management Dashboard**.
2. On the main page, select the Power Pivot service application for which you want to view operations data.

Open the dashboard from a Power Pivot service application

1. In Central Administration, in **Application Management**, click **Manage service applications**.
2. Click the name of the Power Pivot service application. The Power Pivot Management Dashboard displays operational data for the current service application.

Change the current service application.

To change current Power Pivot service application in the management dashboard:

1. At the top of the Power Pivot management dashboard, note the name of the current service application, for example **Default Power Pivot Service Application**.
2. In the **Actions** dashboard, click **List Service Applications**.
3. Click the name of the Power Pivot Service application for which you want to see management dashboard reports.

Source Data in Dashboards

Dashboards, reports and web parts show data from an internal data model that pulls data from the system and Power Pivot application databases. The internal data model is embedded in a Power Pivot workbook hosted in the Central Administration site. The structure of the data model is fixed. Although you can use the PowertPivot workbook as a data source to create new reports, you must not modify the structure in a way that breaks the predefined reports that use it.

For more information about how data is collected, see the following:

- Power Pivot Usage Data Collection
- Configure Usage Data Collection for (Power Pivot for SharePoint

To capture data about the Power Pivot server system, verify event messaging, data refresh history, and other usage history is enabled for each Power Pivot service application. The server and usage data gathered during normal server operations is the source data that ends up in the internal data model. **Note:** If you turn off events or usage history, the composite reports will be incomplete or erroneous.

Edit Power Pivot Dashboard

If you have expertise in dashboard development or customization, you can edit the dashboard to include new web parts. You can also edit the web part properties that are included in the dashboard.

Create Custom Reports for Power Pivot Management Dashboard

For reporting purposes, Power Pivot usage data and history is kept in an internal Power Pivot workbook that is created and configured along with the dashboard. If the default reports do not provide the information you require, you can create custom reports in Excel based on the workbook. Both the workbook and any custom

reports that you create will be preserved if you upgrade or uninstall the Power Pivot solution files later. The workbook and reports are stored in the Power Pivot Management library within the Central Administration site. This library is not visible by default, but you can view the library using the View All Site Content action under Site Actions.

To help you get started with custom reporting, Power Pivot Management Dashboard provides an Office Data Connection (.odc) file to connect to the source workbook. Dor example, you can use the .odc file in Excel to create additional reports.

> **NOTE**
> Edit the file to avoid the following error when attempting to use the .odc file in Excel: "Initialization of the data source failed". The auto-generated .odc file includes a parameter that are not supported by the MSOLAP OLE DB provider. The following instructions provide the workaround for removing the parameters.

You must be a farm or service administrator to build reports that are based on the Power Pivot workbook in Central Administration.

1. Open the Power Pivot Management Dashboard.
2. Scroll to the **Reports** section at the bottom of the page.
3. Click **Power Pivot Management Data**.
4. Save the .odc file to a local folder.
5. Open the .odc file in a text editor.
6. In the **<odc:ConnectionString>** element, scroll to the end of the line and remove **Embedded Data=False**, and then remove **Edit Mode=0**. If the last character in the string is a semicolon, remove it now.
7. Save the File. The remaining steps depend on which version of Power Pivot and Excel you are using.
8. a. Start Excel 2013
   b. In the **Power Pivot** ribbon, click **Manage**.
   c. Click **Get External Data** and then click **Existing connections**.
   d. Click the .ODC file if you see it. If you do not see the .ODC file, click **Browse for More** and then in the file path, specify the .odc file.
   e. Click **Open**
   f. Click **Test Connection** to verify the connection succeeds.
   g. Click type a name for the connection and then click **Next**.
   h. In Specify MDX Query, click **Design** to open the MDX query designer to assemble the data you want to work with **If you see the error message** "The Edit Mode property name is not formatted correctly.", verify you edits the .ODC file.
   i. Click **OK** and then click **Finish**.
   j. Create PivotTable or PivotChart reports to visualize the data in Excel.
9. a. Start Excel 2010.
   b. On the Power Pivot ribbon, click **Launch Power Pivot Window**.

c. On the Design ribbon in the Power Pivot window, click **Existing Connections**.

d. Click **Browse for More**.

e. In the file path, specify the .odc file.

f. Click **Open**. The Table Import Wizard starts, using the connection string to the Power Pivot workbook that contains usage data.

g. Click **Test Connection** to verify that you have access.

h. Enter a friendly name for the connection, and then click **Next**.

i. In Specify MDX Query, click **Design** to open the MDX query designer to assemble the data you want to work with, and then create PivotTable or PivotChart reports to visualize the data in Excel.

See Also

Power Pivot Data Refresh with SharePoint 2010
Configure Usage Data Collection for (Power Pivot for SharePoint

Power Pivot Usage Data Collection

5/16/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Usage data collection is a farm-level SharePoint feature. Power Pivot for SharePoint uses and extends this system to provide reports in the Power Pivot Management Dashboard that show how Power Pivot data and services are used. Depending on how you install SharePoint, usage data collection might be turned off for the farm. A farm administrator must enable usage logging to create the usage data that appears in the Power Pivot Management Dashboard.

For information on usage data in the Power Pivot Management Dashboard, see Power Pivot Management Dashboard and Usage Data.

Usage Data Collection and Reporting Architecture

Power Pivot usage data is collected, stored, and managed using a combination of features from the SharePoint infrastructure and Power Pivot server components. SharePoint infrastructure provides a centralized usage service and built-in timer jobs. Power Pivot for SharePoint adds longer term storage for Power Pivot usage data and reports that you view in SharePoint Central Administration.

In the usage data collection system, event information enters the usage collection system on the application server or Web front end. Usage data moves through the system in response to timer jobs that cause data to move from temporary data files on the physical server to persistent storage on a database server. The following diagram illustrates the components and processes that move usage data through the data collection and reporting system.

Note: Verify usage data collection is enabled. To verify, go to **Monitoring** in SharePoint Central Administration. For more information, see Configure Usage Data Collection for (Power Pivot for SharePoint.

PHASE	DESCRIPTION

PHASE	DESCRIPTION
1	Usage data collection is triggered by events that are generated by Power Pivot components and Analysis Services data providers in SharePoint deployments. Configurable events that can be turned on or off include connection requests, load and unload requests, and query response timing events that are monitored by the Power Pivot service on the application server. Other events that are managed solely by the server and cannot be turned off. These include data refresh and server health events. Initially, usage data is collected and stored in local log files using the data collection features of the SharePoint system. The files and their location are part of the standard usage data collection system in SharePoint. The location of the files is the same on every server in the farm. To view or change the location of the logging directory, go to **Monitoring** in SharePoint Central Administration, and then click **Configure usage and health data collection**.
2	At scheduled intervals (every hour by default) the Microsoft SharePoint Foundation Usage Data Import timer job moves usage data from the local files to the Power Pivot service application database. If you have multiple Power Pivot service applications in a farm, each one will have its own database. Events include internal information that identifies which Power Pivot service application produced the event. The application identifiers ensure that usage data is bound to the application that created it.
3	Data is copied to an internal reporting database that is available to the Power Pivot Management Dashboard in Central Administration.
4	The data source is a Power Pivot workbook that you can access to create custom reports in Excel. There is only one instance of the source workbook. The localized reports are all based on the same source workbook.
5	Usage data is presented in reports for Power Pivot service applications for administrators who manage server performance and availability. Localized instances of the workbooks are created for the supported SharePoint languages. For more information, see Reporting on Usage Data in this topic.

Sources of Usage Data

When usage data collection is enabled, data is generated for the following server events.

EVENT	DESCRIPTION	CONFIGURABLE

EVENT	DESCRIPTION	CONFIGURABLE
Connections	Server connections made on behalf of a user who is querying Power Pivot data in an Excel workbook. Connection events identify who opened a connection to a Power Pivot workbook. In reports, this information is used to identify the most frequent users, the Power Pivot data sources that are accessed by the same users, and trends in connections over time.	You can enable and disable Configure Usage Data Collection for (Power Pivot for SharePoint.
Query response times	Query response statistics that categorize queries based on how long they take to complete. Query response statistics show patterns in how long it takes the server to respond to query requests.	You can enable and disable Configure Usage Data Collection for (Power Pivot for SharePoint.
Data load	Data load operations by the Analysis Services service. Data load events identify which data sources are used most often.	You can enable and disable Configure Usage Data Collection for (Power Pivot for SharePoint.
Data unload	Data unload operations by Power Pivot service applications. An Analysis Services service unloads inactive Power Pivot data sources if it is not being used, or when the server is under memory pressure or needs additional memory to run data refresh jobs.	You can enable and disable Configure Usage Data Collection for (Power Pivot for SharePoint.
Server health	Server operations that indicate server health, measured in CPU and memory utilization. This data is historical. It does not provide real time information about the current processing load on the server.	No. Usage data is always collected for this event.
Data refresh	Data refresh operations initiated by the Power Pivot service for scheduled data updates. Usage history for data refresh is collected at the application level for operational reports, and is reflected in the Manage Data Refresh pages for individual workbooks. **Note:** For SQL Server 2012 SP1 (11.0.3x) and SharePoint 2013 deployments, data refresh is managed by Excel Services and not the Analysis Services Server.	No. Data refresh usage data is always collected if you enable data refresh for the Power Pivot service application.

Services and Timer Jobs

The following table describes the services and data collection stores in the usage data collection system. For instructions on how to override the timer job schedules to force a data refresh of server health and usage data in Power Pivot Management Dashboard reports, see enter link description here. You can see the timer jobs in SharePoint central Administration. Go to **Monitoring**, then click **Check Job Status**. Click **Review Job**

Definitions.

COMPONENT	DEFAULT SCHEDULE	DESCRIPTION
SharePoint Timer Service (SPTimerV4)		This Windows service runs locally on every member computer in the farm and processes all timer jobs that are defined at the farm level.
Microsoft SharePoint Foundation Usage Data Import	Every 30 minutes in SharePoint 2010. Every 5 minutes in SharePoint 2013.	This timer job is configured globally at the farm level. It moves usage data from local usage log files to the central usage data collection database. You can run this timer job manually to force a data import operation.
Microsoft SharePoint Foundation Usage Data Processing timer job	Daily at 3:00 A.M.	Starting with SQL Server 2012 Power Pivot for SharePoint, this Time job is supported for upgrade or migration scenarios where you may have older usage data still in the SharePoint usage databases. Starting with SQL Server 2012 Power Pivot for SharePoint, the SharePoint usage database is not used for Power Pivot usage collection and Management dashboard workflow. The time job can be manually run to move any remaining Power Pivot related data in the SharePoint usage database to the Power Pivot service application databases. This timer job is configured globally at the farm level. It checks for expired usage data in the central usage data collection database (that is, any records that are older than 30 days). For Power Pivot servers in the farm, this timer job performs an additional check for Power Pivot usage data. When Power Pivot usage data is detected, the timer job moves the data to a service application database using an application identifier to find the correct database. You can run this timer job manually to force a check on expired data, or to force a data import of Power Pivot usage data to a Power Pivot service application database.
Power Pivot Management Dashboard Processing timer job	Daily at 3:00 A.M.	This timer job updates the internal Power Pivot workbook that provides administrative data to the Power Pivot Management Dashboard. It gets updated information that is managed by SharePoint, including server names, user names, application names, and file names that appear in dashboard reports or web parts.

Reporting on Usage Data

To view usage data on Power Pivot data, you can view built-in reports in the Power Pivot Management Dashboard. The built-in reports consolidate usage data that is retrieved from reporting data structures in the service application database. Because the underlying report data is updated on a daily basis, the built-in usage reports will show updated information only after the Microsoft SharePoint Foundation Usage Data Processing timer job copies data to a Power Pivot service application database. By default, this occurs once a day.

For more information about how to view reports, see Power Pivot Management Dashboard and Usage Data.

See Also

Power Pivot Management Dashboard and Usage Data
Configuration Setting Reference (Power Pivot for SharePoint)
Configure Usage Data Collection for (Power Pivot for SharePoint

Configure Usage Data Collection for (Power Pivot for SharePoint

11/27/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Usage data collection is a farm-level SharePoint feature. Power Pivot for SharePoint uses and extends this system to provide reports in the Power Pivot Management Dashboard that show how Power Pivot data and services are used. Depending on how you install SharePoint, usage data collection might be turned off for the farm. A farm administrator must enable usage logging to create the usage data that appears in the Power Pivot Management Dashboard.

For information on usage data in the Power Pivot Management Dashboard, see Power Pivot Management Dashboard and Usage Data.

In this topic:

Enable usage data collection and choose events that trigger data collection

Set log file location

Configure the timer jobs used in usage data collection

Limit how long usage data history is stored

Define fast, medium and slow query response categories for reporting purposes

Specify how often query statistics are reported to the usage data collection system

Open the Power Pivot Service Application page to access configuration settings

The default configuration for Power Pivot usage data collection

> **IMPORTANT**
>
> Usage data provides insight into how users are accessing data and resources, but it does not guarantee reliable, persistent data about server operations and user access. For example, if there is a server restart, event usage data will be lost and will not be recoverable. Similarly, if the temporary log files reach maximum size, no new data will be added until the files are cleared. If you require audit capability, consider using the workflow and content type features that SharePoint provides to build out an auditing subsystem for your farm. For more information, look for product and community documentation on the web.

Enable usage data collection and choose events that trigger data collection

Configure usage data collection in SharePoint Central Administration.

1. In Central Administration, click **Monitoring**.
2. In the **Reporting section**, click **Configure usage and health data collection.**
3. Select the **Enable usage data collection**.
4. In the **Events to log** section, select or clear the checkboxes to either enable or disable the following

Analysis Services events:

EVENT	DESCRIPTION
Power Pivot Connections	Power Pivot Connection event is used to monitor Power Pivot server connections that are made on behalf of a user.
Power Pivot Load Data Usage	Power Pivot Load Data Usage is used to monitor requests that load Power Pivot data into server memory. A load event is generated for Power Pivot data files loaded from a content database or from cache.
Power Pivot Unload Data Usage	Power Pivot Unload Data Usage is used to monitor requests for unloading a Power Pivot data source after a period of inactivity. Caching a Power Pivot data source to disk will be reported as an unload event.
Power Pivot Query Usage	Power Pivot Query Usage is used to monitor query processing times for data that is loaded in an Analysis Services service instance.

> **NOTE**
> Server health and data refresh operations also generate usage data, but there is no event associated with these processes.

5. You can also update the log file location. For more information, see the next section.
6. Click **OK** to save your changes.
7. Optionally, you can specify whether all messages or just errors are logged. For more information on how to throttle event messages, see Configure and View SharePoint Log Files and Diagnostic Logging (Power Pivot for SharePoint).

Set log file location

Power Pivot usage data is initially stored in usage log files on the local server, and then moved at regular intervals to the Power Pivot service application databases. The log file location is set in Central Administration. The default location is:

```
C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\15\logs
```

To view or change these properties, use the **Usage and health data collection** page.

1. On the Home page in Central Administration, click **Monitoring**.
2. In the Monitoring section, click **Configure usage and health data collection.**
3. In Usage Data Collection Settings, view or modify the file location, name, or maximum file size. If you specify a file size that is too low, the file size will reach the maximum limit and no new entries will be added to it until its contents are moved to the central usage data collection database.

Configure the timer jobs used in usage data collection

Power Pivot server health and usage data is moved to different locations in the usage data collection system through two timer jobs.:

- The "Microsoft SharePoint Foundation Usage Data Import" timer job moves Power Pivot usage to the Power Pivot service application database.
- The "Power Pivot Management Dashboard Processing timer job" the data to Power Pivot workbook that is the source of data for built-in administrative reports.

If you need to refresh the administrative reports that appear in the Power Pivot Management Dashboard more frequently, follow these steps.

1. In Central Administration, click **Monitoring**.
2. Click **Review job definitions.** In the **Timer jobs** section.
3. Click **Microsoft SharePoint Foundation Usage Data Import**.
4. Click **Run Now**. If the **Run Now** button is disabled, click **Enable** and then click **Run Now**.
5. In the Job Definitions list, click **Power Pivot Data Management Dashboard Processing Timer Job**.
6. Click **Run Now**.
7. Check the reports to view the refresh data. For more information, see Power Pivot Management Dashboard and Usage Data.

Limit how long usage data history is stored

Usage data history is stored for events (connections, load, unload, and on-demand query processing) and data refresh (scheduled data processing). Although usage data is collected through the SharePoint usage data collection system, the reporting data is moved to a Power Pivot application database and a reporting database for longer term storage. The usage data history setting controls how long usage data is retained in the Power Pivot application databases. The same limit is applied equally to all types of stored usage data in the same Power Pivot service application database.

1. Open the Power Pivot Service Application Page.
2. In the **Usage Data Collection** section, in **Usage Data History**, enter the number of days for which you want to keep a record of data refresh activity for each workbook.

   - The default is 365 days.
   - 0 specifies unlimited storage where usage data is kept indefinitely.
   - Alternatively, you can also specify a range between 1 and 5000.

   Decreasing the retention period to a smaller number of days will delete any data that exceeds the new limit. For example, changing the value from 365 to 30 will result in usage data deletion for all historical information that occurred more than 30 days ago. Only data from the last 30 days is retained.

   Data is actually deleted when the next event occurs. The limit on usage data history is checked only when the system processes an event.

3. Click **OK**.

For more information about how usage data is collected and stored, see Power Pivot Usage Data Collection.

Define fast, medium and slow query response categories for reporting purposes

Query processing performance is measured against predefined categories that define a request-response cycle by how long it takes to complete. Predefined categories include: Trivial, Quick, Expected, Long-running, and

Exceeded. Every request to a Power Pivot server will fall into one of the categories based on time to completion.

Query response information is used in activity reports. Within the reports, each category is used differently to better reveal the performance trends of the Power Pivot system. For example, trivial requests are excluded completely because doing so removes noise in the data and shows more meaningful trends using the remaining categories. In contrast, Long-running or Exceeded request statistics are prominent in the report so that administrators or workbook owners can take corrective action immediately.

Although you cannot add or delete categories, you can define the upper and lower limits that determine where one category stops and the next one begins. If your organization uses Service Level Agreements (SLA) to define acceptable levels of server availability and performance, you can tune these categories to reflect the SLA you create.

1. Open the Power Pivot Service Application Page.
2. In the **Usage Data Collection** section, in **Trivial Response upper limit** , enter a value (in milliseconds) that sets the upper boundary for completing a trivial response. Requests that fall into this category typically include server pings, session initiation, and metadata query. The default is 500 milliseconds (or half a second).
3. In Quick Requests Upper Limit, enter a value (in milliseconds) that sets the upper boundary for completing a quick response. Requests that fall into this category include queries of very small datasets or metadata servers of large datasets. The default is 1000 milliseconds (or 1 second).
4. In **Expected Response Upper Limit**, enter a value (in milliseconds) that sets the upper boundary for completing a response in an expected or average time frame. Requests that fall into this category include loading data into a viewer. The default is 3000 milliseconds (or 3 seconds).
5. In **Long Response Upper Limit**, enter a value (in milliseconds) that sets the upper boundary for completing long running response. Requests that fall into this category run longer than expected, but within range that is still acceptable. The default is 10000 milliseconds (or 10 seconds).

   Any requests that exceed this limit are categorized as *Exceeded*. There is no configurable threshold for *Exceeded*. It is inferred from the upper limit you specify on Long Requests Upper Limit. Requests that fall into the Exceeded category run longer than is allowed by an SLA you have defined.
6. Click **OK**.

Specify how often query statistics are reported to the usage data collection system

The time-to-report interval specifies how often query statistics are reported to the usage data collection system. Query statistics accumulate in a process and are reported as a single event at regular intervals. You can adjust the interval to write to the log file more or less often.

1. Open the Power Pivot Service Application Page.
2. In the **Usage Data Collection** section, in **Query Reporting Interval**, enter the number of seconds after which the server will report the query statistics for all categories (trivial, quick, expected, long running, and exceed) as a single event to the usage data collection system.

   - The range is 1 to any positive integer.
   - The default is 300 seconds (or 5 minutes). This value is recommended for dynamic farm environments that run a variety of applications and services.

   If you raise this value to a much larger number, you might lose statistical data before it can be reported.

For example, a service restart will cause query statistics to be lost. Conversely, if your built-in activity reports show insufficient data, consider decreasing the interval to get time-to-report events more frequently.

3. Click **OK**.

Open the Power Pivot Service Application page to access configuration settings

You must be a farm or service administrator to modify service application settings. If you defined multiple Power Pivot service applications in the farm, you must modify each one individually.

1. In SharePoint Central Administration, in **Application Management**, click **Manage service applications**.
2. Find the Power Pivot Service application. You can identify a service application by its type. A Power Pivot service application type is **Power Pivot Service Application**.
3. Click the Power Pivot service application name. The Power Pivot Management Dashboard opens.
4. In **Actions**, click **Configure service application settings**. The Power Pivot Service Application Settings page will open.

The default configuration for Power Pivot usage data collection

Usage data collection for Power Pivot service operations can be enabled with default settings to make it immediately available in applications that support the Analysis Services integration feature. The default settings include events that trigger usage data collection, limits on how long usage data is stored, and thresholds for categorizing query response times.

The following table shows the default values for usage data collection configuration.

SETTING	DEFAULT VALUE	TYPE	VALID RANGE
Analysis Services usage events (Connection, Load, Unload, Requests)	<enabled>	Boolean	These values are either enabled or disabled.
Query Reporting interval	300 (in seconds)	Integer	1 to any positive integer. The default is 5 minutes.
Usage data history	365 (in days)	Integer	0 specifies unlimited, but you can also set an upper limit to expire historical data and have it deleted automatically. Valid values for a limited retention period are 1 to 5000 (in days).
Trivial Response Upper Limit	500 (in milliseconds)	Integer	Sets an upper boundary that defines a trivial request-response exchange. Any request that completes between 0 to 500 milliseconds is a trivial request, and ignored for reporting purposes.

SETTING	DEFAULT VALUE	TYPE	VALID RANGE
Quick Response Upper Limit	1000 (in milliseconds)	Integer	Sets an upper boundary that defines a quick request-response exchange.
Expected Response Upper Limit	3000 (in milliseconds)	Integer	Sets an upper boundary that defines an expected request-response exchange.
Long Running Response Upper Limit	10000 (in milliseconds)	Integer	Sets an upper boundary that defines a long running request-response exchange. Any requests that exceed this upper limit fall into the Exceeded category, which has no upper threshold.

See Also

Configuration Setting Reference (Power Pivot for SharePoint)
Power Pivot Usage Data Collection

Create and Customize Power Pivot Gallery

11/27/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Power Pivot Gallery is a special type of SharePoint document library that provides rich preview and document management for published Excel workbooks and Reporting Services reports that contain Power Pivot data.

In this topic

- Prerequisites
- Overview
- Create the Power Pivot Gallery
- Customize a Power Pivot Gallery Library
- Disable or Hide the Refresh Button
- Switch to Theater View or Gallery View

Prerequisites

- You must have Silverlight. Silverlight can be downloaded and installed through Microsoft Update. If you view a Power Pivot Gallery library using a browser that does not have Silverlight, click the link on the page to install it. You must close and reopen the browser after you install it.

> **NOTE**
>
> The Power Pivot Gallery requires Microsoft Silverlight. The Microsoft Edge browser does not support Silverlight.
> To view the library contents in Edge, click the **Library** tab in Power Pivot gallery, and then change the document library view to **All Documents**.
> To change the default view , click the **Library** tab and then click Modify View. Click "Make this the default view" and then click OK to save the default view.
> For more information on what Edge supports, see the Windows blog, A break from the past, part 2: Saying goodbye to ActiveX, VBScript...

- You must be a site owner to create a library.
- You must have Contribute permissions or above to publish or upload a file.
- Power Pivot Gallery cannot be in a restricted site. The parent site that contains Power Pivot Gallery must be added to either the trusted site or Local intranet zone.
- The Power Pivot web application solution must have been deployed for your application, and the Power Pivot feature must have been activated for the site collection. For more information, see Deploy Power Pivot Solutions to SharePoint andActivate Power Pivot Feature Integration for Site Collections in Central Administration.
- To view or create a Reporting Services report that is based on a Power Pivot workbook, both the workbook and the report must be in the same Power Pivot Gallery. The report must either use a Power Pivot workbook that contains embedded data, or the workbook must contain at most one external data source that is a Power Pivot workbook.

Overview

Power Pivot Gallery is a library template that is available when you install Microsoft SQL Server 2016 Power Pivot for SharePoint on a SharePoint server. The Power Pivot Gallery combines an accurate preview of the file contents with facts about document origin. You can see immediately who created the document and when it was last modified. To create preview images, Power Pivot Gallery uses a snapshot service that can read Power Pivot workbooks and Reporting Services reports that contain Power Pivot data. If you publish a file that the snapshot service cannot read, no preview image will be available for that file.

Preview images are based on how the workbook is rendered by Excel Services. The representation in Power Pivot Gallery should be identical to what you see when you view a Power Pivot workbook in a browser. However, preview has a limited surface area. Portions of a workbook or report might be trimmed to fit the available space. You might need to open a workbook or report to view the document in its entirety.

Refreshing Power Pivot workbook data from external data sources is fully supported in Power Pivot Gallery, but it requires additional configuration. A farm or service administrator must add the Power Pivot Gallery as an Excel Services trusted location. For more information, see Create a trusted location for Power Pivot sites in Central Administration.

Create the Power Pivot Gallery

Power Pivot Gallery is created for you when you install Microsoft SQL Server 2016 Power Pivot for SharePoint using the New Server installation option. If you added Power Pivot for SharePoint to an existing farm or if you want an additional library, you can create a new one for your application or site.

1. a. **SharePoint 2010**: Click **Site Actions** at the top left corner of your site's home page.

   b. Click **More Options**.

   c. Under Libraries, click **Power Pivot Gallery**.

   d. **SharePoint 2013**: Click the settings icon ⚙. Click **Site Contents**

   e. Click **add an app**.

   f. Click **Power Pivot Gallery**.

2. Type a name for the library. Be sure to include descriptive information that helps users identify this library as rich preview for Power Pivot workbooks and Reporting Services reports.

3. Click **Create**.

4. Ask a farm or service administrator to add the Power Pivot Gallery as a trusted location for Excel Services. This step is necessary to avoid errors if a user configures a workbook for Power Pivot data refresh. For more information about this task, see Create a trusted location for Power Pivot sites in Central Administration.

A link to the Power Pivot Gallery library will appear in the navigation Quick Launch pane for the current site.

You can create additional Power Pivot Gallery libraries if you are enforcing different permissions for different site collections or individual sites.

Customize a Power Pivot Gallery Library

Power Pivot Gallery is a SharePoint document library. Therefore, you can use standard library tools in SharePoint to change library settings or work with individual documents in the library. Each library that you create can be independently customized to use a different view or library settings.

Sort order and filters can be modified to change where workbooks appear in the list. By default, documents are listed in the order in which they were added, where the last document published appears at the bottom of the list. Once a document is published, it retains its place in the list. Updating and republishing the document updates it in place in the list.

You cannot enable or disable preview for specific documents. The snapshot service will generate preview images for all Power Pivot workbooks and for Reporting Services reports that are based on Power Pivot workbooks that are stored in the same library. These images can be viewed by all users who have View permissions on the document.

You cannot extend Power Pivot Gallery to provide preview for other document types. Preview is supported only for Excel 2010 workbooks or SQL Server 2008 R2 Reporting Services reports that contain Power Pivot data.

You cannot change the settings that control document origin information. Facts that appear about individual documents, such as who added or last modified the workbook, are determined by a fixed set of columns that cannot be modified.

Change sort order, add filters, or limit the number of documents

Power Pivot Gallery always shows 'Last modified' and 'Created by' values. You cannot disable these columns. You cannot enable other columns for the library.Use the following instructions to change the sort order, add a filter, or limit the number of documents that are visible.

1. In a SharePoint site, open Power Pivot Gallery.
2. In the ribbon, click **Library**.
3. **SharePoint 2010:** In Custom Views, click **Modify this View**.

   SharePoint 2013: In **Manage Views**, click **Modify View**.
4. In Sort, specify the criteria that will be used to determine how workbooks appear in the list. By default, documents are listed in the order in which they were added.
5. In Filter, specify criteria that will be used to show or hide workbooks based on conditional values set on columns. For example, you might want to hide all workbooks created prior to a certain date.
6. In Item Limit, specify options that are useful for Power Pivot Gallery libraries that contain a very large number of documents. You can limit the actual number of items that appear in the list, or display items in batches.
7. Click **OK** to save your changes.

Disable or Hide the Refresh Button

You cannot hide the **Manage Data Refresh** button. However the button is disabled if the user does not sufficient permissions.

Workbook owners or authors must have **Contribute** permission to schedule data refresh on a workbook. Users with Contribute permissions can open and edit the workbook's data refresh configuration page to specify the credentials and schedule information used to refresh the data.

So, users that only have **View** or **Read** permission levels, will not be able to access the refresh button. The refresh button is visible but disabled. For more information, see User permissions and permission levels in SharePoint 2013.

Switch to Theater View or Gallery View

Preview varies depending on how you configure the view for the library. In Gallery view, you can hover the mouse

pointer over individual worksheets in the workbook to bring a sheet into focus in the preview area.

The following table describes the different layouts for presenting thumbnail sketches of each previewed page:

VIEW	DESCRIPTION
Gallery View (default)	Gallery is the default view for a Power Pivot Gallery. The preview appears to the left. Smaller thumbnails of each worksheet appear next to it in a sequential left-to-right order.
All Documents	This is the standard layout for document libraries. You can choose this view to manage individual documents or view library contents in a list format. Use this view to edit properties, delete, or move individual documents. If you enabled versioning, you must use this view to check documents in or out of the library.
Theatre View and Carousel View	These are specialized views that work best if you are showcasing a small number of related documents. The full rotation of thumbnails includes all pages in all documents in the library. If you have a large number of documents, these views might be impractical for users who want to find or open a specific Power Pivot workbook. Theatre view: The preview area is centered. Smaller thumbnails of each worksheet appear lower on the page, on either side. Carousel view: The preview area is centered. The thumbnails that immediately precede and follow the current thumbnail are adjacent to the preview area.

Switch to a different view

1. In a SharePoint site, open Power Pivot Gallery.
2. In the ribbon, click **Library**.
3. In Manage Views, in Current View, select the view you want to use from the list. Predesigned views include Gallery, Theater, and Carousel. Alternatively, you can choose All Documents if you want to move, delete, or manage documents in the library.

See Also

Troubleshoot a Power Pivot for SharePoint Installation
Use Power Pivot Gallery
Create a trusted location for Power Pivot sites in Central Administration
Delete Power Pivot Gallery

Use Power Pivot Gallery

11/27/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Power Pivot Gallery is a special SharePoint document library that provides rich preview and document management for published Excel workbooks and Reporting Services reports that contain Power Pivot data.

> **NOTE**
> Depending on how your server is configured, you might see warning or error messages in the preview area for specific documents. Messages can appear when an Excel workbook is set to automatically refresh its data whenever it is opened. Data refresh warning messages will appear as the preview image if Excel Services is configured to show Warn on data refresh error messages. Farm or service administrators can modify configuration settings to allow a preview of the actual worksheet to appear. For more information, see Create a trusted location for Power Pivot sites in Central Administration.

In this topic

Prerequisites

Icons in Power Pivot Gallery

Save an Excel Workbook to Power Pivot Gallery

Create new reports or workbooks based on a published Power Pivot workbook

Open a workbook or report in full page mode

Schedule data refresh for Power Pivot workbooks in Power Pivot Gallery

Delete a workbook or report in Power Pivot Gallery

Refresh a thumbnail image

Known Issues

Prerequisites

> **NOTE**
> The Power Pivot Gallery requires Microsoft Silverlight. The Microsoft Edge browser does not support Silverlight.
> To view the library contents in Edge, click the **Library** tab in Power Pivot gallery, and then change the document library view to **All Documents**.
> To change the default view , click the **Library** tab and then click Modify View. Click "Make this the default view" and then click OK to save the default view.
> For more information on what Edge supports, see the Windows blog, A break from the past, part 2: Saying goodbye to ActiveX, VBScript...

For a complete list of prerequisites, see Create and Customize Power Pivot Gallery.

Icons in Power Pivot Gallery

Icons provide a visual indicator on content availability and status.

ICON	DESCRIPTION
	The hourglass icon appears when a thumbnail image of each page in the document is being generated. Refresh the page to display the update the image.
	The pages icon appears when a workbook or report has more pages than can be displayed in Power Pivot Gallery. To view all the pages, you must use a client application.
	The error icon appears when a thumbnail image could not be rendered for the document. The document is published to the library, but it cannot be rendered in the custom Power Pivot Gallery views. You should be able to view the document in a client application, such as the Power Pivot for Excel add-in.
	The unavailable content icon appears when the document you uploaded cannot be rendered in Power Pivot Gallery. Supported document types include Power Pivot workbooks and reports created in SQL Server 2008 R2 Reporting Services Report Builder. This icon also appears if you recycle a document from Recycle Bin. If you are getting this icon for a document that previously presented a valid preview image, you can refresh the image by editing a document property and then saving your changes.
	The locked content icon appears when thumbnail images are purposely disabled for this document. Power Pivot Gallery does not generate thumbnail images for Excel workbooks that contain no Power Pivot data, or for Power Pivot workbooks or Reporting Services reports that do not meet the requirements for snapshot generation. For more information, see the Prerequisites section in this topic.

Save an Excel Workbook to Power Pivot Gallery

You can publish Power Pivot workbooks to the library using all of the sharing techniques that Excel 2010 provides. For example, in Excel 2010, you can use Save As to specify all or part of a SharePoint path to a library.

1. Save the file.
2. a. **Excel 2010:** In the File menu, click **Save & Send**.
   b. Click **Save to SharePoint**.
   c. Click **Publish Options** if you want to use Excel Services Options to select individual sheets or parameters that you want to publish. For example, the Parameters tab in Excel Services Options lets you choose which slicers appear in the published workbook.
   d. **Excel 2013:** In the File menu, click **Save**.
   e. Click **Browser View Options** if you want to use Excel Services Options to select individual sheets or parameters that you want to publish. For example, the Parameters tab in Excel Services Options lets you choose which slicers appear in the published workbook.
3. In the Save As dialog box, in File name, enter a full or partial URL to Power Pivot Gallery. If you enter a portion of the URL address, such as the server name, you can browse the site to find the Power Pivot

Gallery. To do this, click **Save** to open a connection to the server you specified.

4. Using the Save As dialog box, select Power Pivot Gallery on your site.
5. Click **Open** to open the library.
6. Click **Save** to publish the workbook to the library.

In a browser window, verify that the document appears in Power Pivot Gallery. Newly published documents will appear in the list. Library settings determine where the document appears (for example, sorted in ascending order by date, or alphabetically by name). You might need to refresh the browser window to view the most recent additions.

Upload a workbook into Power Pivot Gallery

You can also upload a workbook if you want to start from SharePoint and select from your computer which file to publish.

1. In a SharePoint site, open Power Pivot Gallery.
2. In Library ribbon, click **Documents**.
3. In **Upload Document**, select an upload option and then enter the name and location of the file you want to upload. Library settings determine where the document appears. You might need to refresh the browser window to view the latest addition.

Create new reports or workbooks based on a published Power Pivot workbook

For Power Pivot workbooks that you publish to Power Pivot Gallery, you can create additional workbooks or Reporting Services reports that use the published workbook as a connected data source.

	Click the downward arrow portion of the New Report button to launch Report Builder or Excel 2010. Power Pivot Gallery must use one of the predesigned views (Theater, Gallery, or Carousel) in order for the New Report button to be available.

Create Report Builder Report

Creating a new report based on an existing Power Pivot workbook in the library requires that Reporting Services is configured for SharePoint integration for the same sites that contain Power Pivot Gallery. When you select the Create Report Builder Report option, Report Builder is downloaded from the report server and installed on the local workstation on first use. A placeholder report file is created for the new report and saved to Power Pivot Gallery. Connection information to the Power Pivot workbook is created for you as a new data source in the report. As a next step, you can build the datasets and report layout in the design workspace. As you use Report Builder to assemble your report, you can save your changes and the final result to the report document in the

gallery. To avoid data disconnections later, be sure to keep the report and workbook files together in the same library.

Open New Excel Workbook

To create a new Excel workbook from an existing workbook, you must already have Excel and Power Pivot for Excel on the local computer. Choosing Open New Excel Workbook starts Excel, opens a blank workbook (.xlsx) file, and loads Power Pivot data in the background as a connected data source. Only the data from the Power Pivot window in the original workbook is used in the new workbook. PivotTables or PivotCharts from the original workbook are excluded. The new workbook links to data in the original workbook. The data is not copied to the new workbook itself.

Open a workbook or report in full page mode

Click on any visible thumbnail image of the previewed document to open it in full page mode independent of Power Pivot Gallery preview. Power Pivot workbooks will open in a browser. Reporting Services reports will open in the ReportViewer web part that is part of Reporting Services deployment on a SharePoint server.

An alternative approach to viewing the workbook in a browser is to open it in Excel on a client workstation. You must have Excel 2013 or Excel 2010 and the Power Pivot for Excel add-in to view the file. You can use Excel 2007 to open the file but you cannot use it to pivot on the data. For this reason, Excel 2013 or Excel 2010 are recommended for both viewing and creating Power Pivot data. If you do not have the required applications, you must use a browser to view the workbook from SharePoint.

Schedule data refresh for Power Pivot workbooks in Power Pivot Gallery

Power Pivot data in a published Excel workbook can be refreshed at scheduled intervals.

	Click the Manage Data refresh button to create or view a schedule that retrieves updated data from connected data sources. For instructions on how to create a schedule, see Schedule a Data Refresh (Power Pivot for SharePoint).

Delete a workbook or report in Power Pivot Gallery

To delete a document from the library, switch to the All Documents view first.

1. In a SharePoint site, open Power Pivot Gallery.
2. In the ribbon, click **Library**.
3. In Manage Views, in the Current View list, click the down arrow and select All Documents.
4. Select the workbook or report you want to delete.
5. In Documents (Files), in Manage, click the **Delete Document** button.

Refresh a thumbnail image

Use the following steps to regenerate a thumbnail image for a document in Power Pivot Gallery.

1. Switch Power Pivot Gallery to All Documents view. To do this, click **Library** in the ribbon and change the **Current View** to **All Documents**.
2. Select the workbook or report for which you want to refresh the thumbnail image..

3. Click the down arrow to the right, and then select **Edit Properties**.

4. Click **Save**. Saving the document forces the snapshot service to regenerate the preview image.

Known Issues

Document Type is Not Supported

The **Power Pivot Gallery Document** content type is not supported. If you enable the **Power Pivot Gallery Document** content type for a document library, and attempt to create a new document of that type, you will see an error message similar to one of the following:

- 'New Document' requires a Microsoft Sharepoint Foundation-compatible application and web browser. To add a document to this document library, click the 'Upload Document' button.
- "The Internet address 'http://[server name]/testSite/PowerPivot Gallery/ReportGallery/Forms/Template.xlsx' is not valid.""Microsoft Excel cannot access the file 'http://[server name]/testSite/PowerPivot Gallery/ReportGallery/Forms/Template.xlsx'. There are several possible reasons:

The **Power Pivot Gallery Document** content type is not automatically added to document libraries, You will not encounter this issue unless you manually enable the unsupported content type.

See Also

Create a trusted location for Power Pivot sites in Central Administration
Delete Power Pivot Gallery
Create and Customize Power Pivot Gallery
Schedule a Data Refresh (Power Pivot for SharePoint)

Delete Power Pivot Gallery

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Deleting a Power Pivot Gallery also deletes all of its items, including the data refresh history for Power Pivot workbooks that are configured for scheduled data refresh.

> **NOTE**
>
> This topic provides instructions on how to delete a SharePoint library. For instructions on how to delete individual documents in the library, see Use Power Pivot Gallery.

You must be a site owner to delete a library.

Delete Power Pivot Gallery SharePoint 2013

1. Click Settings, then click **Site settings**.
2. Click **Site Settings**.
3. Under Site Administration, click **Site libraries and lists**.
4. Click **Customize "Power Pivot Gallery"**.
5. Under Permissions and Management, click **Delete this document library**.

Delete Power Pivot Gallery SharePoint 2010

1. Click **Site Actions**.
2. Click **Site Settings**.
3. Under Site Administration, click **Sites libraries and lists**.
4. Click **Customize "Power Pivot Gallery"**.
5. Under Permissions and Management, click **Delete this library**.

See Also

Power Pivot Gallery

Power Pivot Data Access

12/6/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes the ways in which data is retrieved from a Power Pivot workbook that is published to a SharePoint library.

Power Pivot data is stored inside an Excel workbook. The connection string is a URL to a workbook on a SharePoint site.

Power Pivot data is most often used by the workbook that contains it, as the data behind PivotTables and PivotCharts. Alternatively, Power Pivot data can also be used as an external data source, where a workbook, dashboard, or report connects to a separate Excel (.xlsx) file in SharePoint and retrieves the data for subsequent use. Client tools that typically use Power Pivot data are Excel, Power View, other Reporting Services reports, and PerformancePoint.

On the desktop, the Power Pivot add-in uses AMO and ADOMD.NET to create, process, and query the Power Pivot data in the client workspace.

On a SharePoint farm, Excel Services uses the local MSOLAP OLE DB provider to connect to Power Pivot data. The provider sends the connection request to a Power Pivot for SharePoint server in the farm. That server loads the data, runs the query, and returns the result set.

Querying Power Pivot Data in SharePoint

When you view a Power Pivot workbook from a SharePoint library, the Power Pivot data that is inside the workbook is detected, extracted, and processed separately on Analysis Services server instances within the farm, while Excel Services renders the presentation layer. You can view the fully-processed workbook in a browser window or in an Excel 2010 desktop application that has the Power Pivot add-in.

The following diagram shows how a request for query processing moves through the farm. Because Power Pivot data is part of an Excel 2010 workbook, a request for query processing occurs when a user opens an Excel workbook from a SharePoint library and interacts with a PivotTable or PivotChart that contains Power Pivot data.

Excel Services and Power Pivot for SharePoint components process different parts of the same workbook (.xlsx) file. Excel Services detects Power Pivot data and requests processing from a Power Pivot server in the farm. The Power Pivot server allocates the request to an Analysis Services service instance, which extracts the data from the workbook in the content library and loads the data. Data that is stored in memory is merged back into the rendered workbook, and passed back to Excel Web Access for presentation in a browser window.

Not all data in a Power Pivot workbook is handled by Power Pivot for SharePoint. Excel Services processes tables and cell data in a worksheet. Only PivotTables, PivotCharts, and Slicers that go against Power Pivot data are handled by the Power Pivot for SharePoint.

See Also

Connect to Analysis Services

Power Pivot Data Refresh

12/10/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

After you create a workbook that contains Power Pivot data, you might want to periodically refresh the data by rerunning a query or command to get updated information from the sources you used originally to create the workbook. This process is called **data refresh**, and you can refresh data on demand in Power Pivot for Excel, or as a scheduled operation that runs as an Analysis Services process on an application server in a SharePoint farm. For more information, see:

- Power Pivot Data Refresh with SharePoint 2010
- Configure the Power Pivot Unattended Data Refresh Account (Power Pivot for SharePoint)
- Configure Stored Credentials for Power Pivot Data Refresh (Power Pivot for SharePoint)
- Schedule a Data Refresh (Power Pivot for SharePoint)
- View Data Refresh History (Power Pivot for SharePoint)

NOTE

SQL Server 2017 and SharePoint Server 2013 Excel Services use a different architecture for data refresh of Power Pivot data models. The SharePoint 2013 supported architecture utilizes Excel Services as the primary component to load Power Pivot data models. The previous data refresh architecture used relied on a server running Power Pivot System Service and Analysis Services in SharePoint mode to load data models. For more information, see the following:

- Power Pivot Data Refresh with SharePoint 2013
- Upgrade Workbooks and Scheduled Data Refresh (SharePoint 2013)

View Data Refresh History (Power Pivot for SharePoint)

10/25/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data refresh history is a record of all data refresh activity for Power Pivot data in an Excel workbook. Data refresh operations are performed on an Analysis Services server instance in a SharePoint farm on a schedule that you provide. By default, data refresh history is retained for one year. However, a farm administrator can specify a different retention policy for usage and event history that determines how long data refresh records are kept.

Applies to: SharePoint 2013 | SharePoint 2010

In this topic:

Prerequisites

View Data Refresh History for an Individual Workbook

View Data Refresh History for All Workbooks

Use History Information

Prerequisites

You must have Contribute permissions or greater to view the data refresh history.

You must have enabled and scheduled data refresh on a workbook that contains Power Pivot data. If you have not scheduled data refresh, you will see the schedule definition page instead of history information.

View Data Refresh History for an Individual Workbook

1. On a SharePoint site, open the library that contains an Excel workbook that contains Power Pivot data.

   There is no visual indicator that identifies which workbooks in a SharePoint library contain Power Pivot data. You must know in advance which workbook contains refreshable Power Pivot data.

2. Select the workbook, and then click the down arrow that appears to the right.

3. Select **Manage Power Pivot Data Refresh**.

The history page appears, showing a complete record for all refresh activity for Power Pivot data in the current Excel workbook.

View Data Refresh History for All Workbooks

Using the Power Pivot Management Dashboard in Central administration, farm administrators and service application administrators can get a comprehensive view of data refresh history and status for all Power Pivot workbooks. For more information, see Power Pivot Management Dashboard and Usage Data.

Use History Information

The data refresh history page provides detailed information about each refresh operation. You can use the information on this page to confirm whether refresh occurred or determine why it failed.

ITEM	DESCRIPTION
Name	Specifies the file name of the Excel workbook that contains Power Pivot data.
Current status	Values include **Scheduled**, **Refreshing**, **Succeeded**, or **Failed**. **Scheduled** appears when you first create the schedule. After data refresh runs the first time, this status message no longer appears. **Refreshing** indicates that data refresh is in progress. A request is either in the process queue or actively running on the server. **Succeeded** indicates that the last data refresh operation completed and the updated workbook is checked back into the SharePoint library. **Failed** indicates that the last data refresh operation did not succeed. The refreshed data was not saved. The workbook contains the same data it had before data refresh began.
Last successful refresh	Specifies the date on which the last data refresh completed successfully.
Next schedule refresh	Specifies the date on which the next data refresh is scheduled to occur. The **Configure schedule** link takes you to the schedule definition page. If you have Contribute permissions on the workbook, you can click the link to view and modify the schedule information that controls unattended data refresh for Power Pivot data in the workbook.
Started	Within the history details section, **Started** indicates the actual processing time. The actual processing time might be different from what you scheduled. Processing will begin when sufficient memory is available on the server. If the server is very busy, the processing might begin several hours after the start time you specified.
Completed	Within the history details section, **Completed** indicates when the data refresh operation finished. The date and time indicates when the workbook was checked back into the library. If data refresh fails, one or more error messages explain the cause of the failure. You can expand each record to view detailed status. Each data source is listed individually, alongside success or failure messages that explain why data refresh did not complete.
Time	Provides the cumulative time from when data refresh started to when it was completed.
Status	Provides a historical record of whether a refresh operation succeeded or failed.

See Also

Configure Usage Data Collection for (Power Pivot for SharePoint
Schedule a Data Refresh (Power Pivot for SharePoint)
Power Pivot Data Refresh

Power Pivot Data Refresh with SharePoint 2013

12/10/2018 • 11 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The design for refresh of Power Pivot data models in SharePoint 2013 utilizes Excel Services as the primary component to load and refresh data models on an instance of SQL Server 2017 Analysis Services running in SharePoint mode. The Analysis Services server runs external to the SharePoint farm. The architecture in SharePoint 2013 Excel Services supports both **interactive data refresh** and **scheduled data refresh**.

Applies to: SharePoint 2013

In this topic:

- Interactive Data Refresh
- Windows Authentication with Workbook Data Connections and Interactive Data Refresh
- Scheduled Data Refresh
- Scheduled Data Refresh Architecture in SharePoint 2013
- Additional Authentication Considerations
- More Information

Background

SharePoint Server 2013 Excel Services manages data refresh for Excel 2013 workbooks and triggers the data model processing on a SQL Server 2017 Analysis Services server that is running in SharePoint mode. For Excel 2010 workbooks, Excel Services also manages the loading and saving of workbooks and data models. However, Excel Services relies on the Power Pivot System Service to send the processing commands to the data model. The following table summarizes the components that send processing commands for data refresh depending on the version of the workbook. The assumed environment is a SharePoint 2013 farm configured to use a SQL Server 2017 Analysis Server running in SharePoint mode.

	Excel 2013 Workbooks	Excel 2010 Workbooks
Trigger Data Refresh	**Interactive:** Authenticated User **Scheduled:** Power Pivot System Service	Power Pivot System Service
Load Workbook from Content Databases	SharePoint 2013 Excel Services	SharePoint 2013 Excel Services
Load Data model on Analysis Services instance	SharePoint 2013 Excel Services	SharePoint 2013 Excel Services
Send Processing Commands to Analysis Services instance	SharePoint 2013 Excel Services	Power Pivot System Service
Update Workbook Data	SharePoint 2013 Excel Services	SharePoint 2013 Excel Services

Save Workbook and Data model to Content Database	**Interactive:** N/A **Scheduled:** SharePoint 2013 Excel Services	SharePoint 2013 Excel Services

The following table summarizes the supported refresh features in a SharePoint 2013 farm configured to use a SQL Server 2017 Analysis Server running in SharePoint mode:

WORKBOOK CREATED IN	SCHEDULED DATA REFRESH	INTERACTIVE REFRESH
2008 R2 Power Pivot for Excel	Not supported. Upgrade the workbook **(*)**	Not supported. Upgrade the workbook **(*)**
2012 Power Pivot for Excel	Supported	Not supported. Upgrade the workbook **(*)**
Excel 2013	Supported	Supported

(*) For more information on workbook upgrades, see Upgrade Workbooks and Scheduled Data Refresh (SharePoint 2013).

Interactive Data Refresh

Interactive, or manual data refresh in SharePoint Server 2013 Excel Services can refresh data models with data from the original data source. Interactive data refresh is available after you configure an Excel Services application by registering an Analysis Services server, running in SharePoint mode. For more information, see Manage Excel Services data model settings (SharePoint Server 2013) (http://technet.microsoft.com/library/jj219780.aspx).

> **NOTE**
> Interactive data refresh is only available for workbooks that created in Excel 2013. If you try to refresh an Excel 2010 workbook, Excel Services displays an error message similar to " Power Pivot Operation Failed: The Workbook was created in an older version of Excel and Power Pivot cannot be refreshed until the file is upgraded". For more information on upgrading workbooks, see Upgrade Workbooks and Scheduled Data Refresh (SharePoint 2013).

Interactive refresh key point of interest:

- Interactive data refresh only refreshes the data in the current user session. The data is not automatically saved back to the workbook item in the SharePoint content database.
- **Credentials:** Interactive data refresh can use the identity of the currently logged-on user as credentials or stored credentials to connect to the data source. The credentials used depend on the Excel Services Authentication Settings defined for the workbook connection to the external data source.
- **Supported Workbooks:** Workbooks created in Excel 2013.

To refresh data:

- See the illustration that follows the steps.

1. In a SharePoint document library, open a Power Pivot workbook in the browser.
2. In the browser window, click the **Data** menu and then click **Refresh Selected Connection** or **Refresh All Connections**.

3. Excel Services loads the Power Pivot database, processes it and then queries it to refresh the Excel workbook cache.

4. **Note:** The updated workbook is not automatically saved back to the document library.

Windows Authentication with Workbook Data Connections and Interactive Data Refresh

Excel Services sends the Analysis Services server a process command that instructs the server to impersonate a user account. To obtain system rights sufficient to perform the user impersonation-delegation process, the Analysis Services service account, requires **Act as part of the operating system** privilege on the local server. The Analysis Services server also needs to be able to delegate the user's credentials to data sources. The query result is sent to Excel Services.

Typical user experience: When a customer selects "Refresh All Connections" in an Excel 2013 workbook that contains a Power Pivot model, they see an error message similar to the following:

- **External Data Refresh Failed:** An error occurred while working on the Data Model in the workbook. Please try again. We are unable to refresh one or more data connections in this workbook.

Depending on the data provider you are using, you see messages similar to the following in the ULS log.

With the SQL Native Client:

- Failed to create an external connection or execute a query. Provider message: Out of line object 'DataSource', referring to ID(s) '20102481-39c8-4d21-bf63-68f583ad22bb', has been specified but has not been used. OLE DB or ODBC error: A network-related or instance-specific error has occurred while establishing a connection to SQL Server. Server is not found or not accessible. Check if instance name is correct and if SQL Server is configured to allow remote connections. For more information see SQL Server Books Online.; 08001; SSL Provider: The requested security package does not exist ; 08001; Client unable to establish connection; 08001; Encryption not supported on the client.; 08001. , ConnectionName: ThisWorkbookDataModel, Workbook: book1.xlsx.

With the Microsoft OLE DB Provider for SQL Server:

- Failed to create an external connection or execute a query. Provider message: Out of line object 'DataSource', referring to ID(s) '6e711bfa-b62f-4879-a177-c5dd61d9c242', has been specified but has not been used. OLE DB or ODBC error. , ConnectionName: ThisWorkbookDataModel, Workbook: OLEDB Provider.xlsx.

With the .NET Framework Data Provider for SQL Server:

- Failed to create an external connection or execute a query. Provider message: Out of line object 'DataSource', referring to ID(s) 'f5fb916c-3eac-4d07-a542-531524c0d44a', has been specified but has not been used. Errors in the high-level relational engine. The following exception occurred while the managed IDbConnection interface was being used: Could not load file or assembly 'System.Transactions, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089' or one of its dependencies. Either a required impersonation level was not provided, or the provided impersonation level is invalid. (Exception from HRESULT: 0x80070542). , ConnectionName: ThisWorkbookDataModel, Workbook: NETProvider.xlsx.

Summary of Configuration Steps To configure **Act as part of the operating system** privilege on the local

server:

1. On the Analysis Services Server running in SharePoint mode, Add the Analysis Services service account to the "Act as part of the operating system" privilege:

   a. Run "`secpol.msc`"

   b. Click **Local Security Policy**, then click **Local policies**, and then click **User rights assignment**.

   c. Add the service account.

2. Restart Excel Services and reboot the Analysis Services server.

3. Delegation from the Excel Services service account or from Claims to Windows token service (C2WTS) to the Analysis services instance is not required. Therefore no configuration for KCD from Excel Services or C2WTS to Power Pivot AS service is necessary. If the backend data source is on the same server as the Analysis Services instance, Kerberos Constrained Delegation is not required. However, the Analysis Services service account requires the right to Act As Part Of The Operating System.

For more information, see Act as part of the operating system (http://technet.microsoft.com/library/cc784323(WS.10).aspx).

Scheduled Data Refresh

Scheduled data refresh key points of interest:

- Requires the deployment of the Power Pivot for SharePoint add-in. For more information, see Install or Uninstall the Power Pivot for SharePoint Add-in (SharePoint 2013).
- A user configures a refresh schedule for a workbook. At the scheduled time, the Power Pivot System Service sends a request to Excel Services to:
  - Load and process the Power Pivot database.
  - Refresh the workbook.
  - Save the workbook back to the content database.
- **Credentials:** Uses stored credentials. Does not use the identity of the current user.
- **Supported Workbooks:** Workbooks created using the SQL Server 2012 (11.x)Power Pivot add-in for Excel 2010 or using Excel 2013. Workbooks created in Excel 2010 with the SQL Server 2008 R2Power Pivot add-in are not supported. Upgrade the workbook to at least the SQL Server 2012 (11.x)Power Pivot format. For more information on workbook upgrades, see Upgrade Workbooks and Scheduled Data Refresh (SharePoint 2013).

To display the **Manage Data Refresh** page:

- See the illustration that follows the steps.

1. In a SharePoint document library, click the **Open menu** (**...**) for a Power Pivot workbook.
2. Click the second **Open menu** and then click **Manage Power Pivot Data Refresh**.
3. On the **Manage Data Refresh** page, click **Enable** and then configure the refresh schedule.
4. At the specified time, the Power Pivot System Service sends a request to Excel Services to:
   - Load and process the Power Pivot data model.
   - Refresh the workbook.
   - Save the workbook back to the content database.

TIP

For information on refreshing workbooks from SharePoint online, see Refreshing Excel workbooks with embedded Power Pivot models from SharePoint Online (white paper) (http://technet.microsoft.com/library/jj992650.aspx).

Scheduled Data Refresh Architecture in SharePoint 2013

The following illustration summarizes the data refresh architecture in SharePoint 2013 and SQL Server 2012 SP1.

	DESCRIPTION	
(1)	Analysis Services Engine	An SQL Server 2017Analysis Services server that is running in SharePoint mode. The server runs outside of the SharePoint farm.
(2)	User Interface	The user interface is comprised of two pages. One to define the schedule and the second to view the refresh history. The pages do not directly access the Power Pivot Service application databases but use the Power Pivot system service to access the databases.
(3)	Power Pivot System Service	The service is installed when you deploy the Power Pivot for SharePoint add-in. The service is used for the following:
		This service hosts the refresh scheduling engine, which calls Excel Services APIs for data refresh of Excel 2013 workbooks. For Excel 2010 workbooks, the service directly performs the data model processing but continues to reply on Excel Services for loading the data model and updating the workbook.
		This service provides methods for components such as the user interface pages, to communicate with the system service.
		Manages requests for external access to workbooks as a data source, received through the Power Pivot Web Service.

	DESCRIPTION	
		Scheduled data refresh request management for timer jobs and configuration pages. The service manages requests to read data in and out of the service application database and trigger data refresh with Excel Services.
		Usage processing and related timer job.
(4)	Excel Calculation Services	Responsible for loading the data models.
(5)	Secure Store Service	If the authentication settings in the workbook are configured to **Use the authenticated user's account** or **None**, the credentials stored in the Secure Store target application ID are used for data refresh. For more information, see the Additional Authentication Considerations section in this topic.
(6)	Power Pivot data refresh timer Job	Instructs the Power Pivot system service to connect with Excel Services for refreshing data models.

Analysis Services requires appropriate data providers and client libraries so that the Analysis Services server in SharePoint mode can access data sources.

> **NOTE**
> Because the Power Pivot system service no longer loads or saves Power Pivot models, most of the settings for caching models on an application server do not apply to a SharePoint 2013 farm.

Data refresh log data

Usage Data: You can view data refresh usage data in the Power Pivot Management Dashboard. To see the usage data:

1. In SharePoint Central Administration, click **Power Pivot Management Dashboard** in the **General application settings** group.
2. At the bottom of the dashboard, see the **Data Refresh - Recent Activity** and **Data Refresh - Recent Failures**.
3. For more information on usage data and how to enable it, see Power Pivot Management Dashboard and Usage Data.

Diagnostic log data: You can view SharePoint diagnostic log data related to data refresh. First, verify the configuration of diagnostic logging for the **Power Pivot Service** in SharePoint Central Administration **Monitoring** page. You may need to increase the level of logging for the "least critical event" to log. For example, temporarily set the value to **Verbose** and then rerun data refresh operations.

The log entries contain:

- The **Area** of **Power Pivot Service**.
- The category of **Data Refresh**.

Review the **configure diagnostic logging**. For more information, see Configure and View SharePoint Log Files and Diagnostic Logging (Power Pivot for SharePoint).

Additional Authentication Considerations

The settings in the **Excel Services Authentication Settings** dialog in Excel 2013, determine the Windows identity that Excel Services and Analysis Services use for data refresh.

- **Use the authenticated user's account**: Excel Services performs data refresh under the identity of the currently logged-in user.
- **Use a stored account**: Assumes a SharePoint Secure Store Service application ID, which Excel Services uses to retrieve the user name and password to authenticate data refresh authentication.
- **None**: The Excel Services **Unattended Service Account** is used. The service account is associated with a Secure Store Proxy. Configure the settings on the **Excel Services Application Settings** page, in the **External Data** section.

To open the authentication settings dialog:

1. Click the **Data** tab in Excel 2013.
2. Click **Connections** in the ribbon.
3. In the **Workbook connections dialog**, select the connection and click **Properties**.
4. In the **Connection properties** dialog, click **Definition**, and then click the **Authentication Settings...** button.

For more information on data refresh authentication and usage of credentials, see the blog post Refreshing Power Pivot Data in SharePoint 2013.

More Information

Troubleshooting Power Pivot Data Refresh.

Excel Services in SharePoint 2013 (http://msdn.microsoft.com/library/sharepoint/jj164076(v=office.15).

See Also

Install Analysis Services in Power Pivot Mode

Schedule Data Refresh and Data Sources - No Windows Authentication

11/27/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes a workflow of Power Pivot for SharePoint schedule data fresh that can use data sources that do **NOT** support Windows Authentication. For example Oracle or IDM DB2 data sources. The illustrations and steps in this topic reference Oracle data sources but the same workflow applies to other data sources.

Applies to: SharePoint 2010 | SharePoint 2013.

Overview: Create two Secure Store Target Applications. Configure the first target application (PowerPivotDataRefresh) to use Windows credentials. Configure the second target application with the credentials for a data source that does not support windows authentication, for example, an Oracle database. The second target application also uses the first target application for the unattended data refresh account.

- **(1) PowerPivotDatarefresh:** A Secure Store Target Application ID that is set with windows authentication.
- **(2) OracleAuthentication:** A Secure Store Target Application ID that is set with Oracle credentials.
- **(3)** The Power Pivot Service application is configure to use the target application "PowerPivotDataRefresh" for the **Unattended Data Refresh Account**.
- **(4)** PowerePivot Workbook uses Oracle data. The workbook refresh settings specify the data source connection to use the target application **(2)** for credentials.

Prerequisites

- A Power Pivot Service Application exists.
- A Secure Store Service Application exists.
- An Excel workbook with a Power Pivot data model exists.

To Create a Target Application ID that uses Windows Authentication

1. In SharePoint Central administration, click **Manage Service Applications**.
2. Click the name of your secure store service application.
3. On the **Manage** page, click **New**.
4. On the **Create New Secure Store Target Application** page, configure the following values:
   - **Target Application ID:** PowerPivotDataRefresh.
   - **Display Name:** PowerPivotDataRefresh.
   - **Contact E-mail:** ?
   - **Target Application Type:** Group.
   - **Target Application Page URL:** None.
5. Click **Next**.
6. On the Credentials page, leave the two default field names and field types for **Windows User Name** and **Windows Password**.
7. Click **Next**.
8. On the **Membership Settings** page, add at least one **Target Application Administrator** and then add members that need access to the target application.
9. Click **OK**.
10. The new Target Application ID is added to the list. Select the Target application ID and click **Set Credentials**.
11. Type the Windows User Name and Windows Password and then click **OK**.

To Create a Target Application ID that uses Oracle credentials

1. In SharePoint Central administration, click **Manage Service Applications**.
2. Click the name of your Secure Store Service application.
3. On the **Manage** page, click **New**.
4. On the **Create New Secure Store Target Application** page, configure the following values:
   - **Target Application ID:** OracleAuthentication.
   - **Display Name:** OracleAuthentication.
   - **Contact E-mail:** ?
   - **Target Application Type:** Group.

- **Target Application Page URL:** None.

5. Click **Next**.
6. On the **Credentials** page, change the first field name to **Oracle User ID** and change the **Field Type** to **User Name**.

   Change the second field name to **Oracle Password** and the **Field Type** to **Password**.
7. Click **Next**.
8. On the **Membership Settings** page, add at least one **Target Application Administrator** and then add members that need access to the target application.
9. Click **OK**.
10. The new Target Application ID is added to the list. Select the Target application ID and click **Set Credentials** .
11. Type the Oracle User ID and Oracle Password and then click **OK**.

For more information, see section "To Create a target application for SQL Server Authentication" in Use Secure Store with SQL Server Authentication (SharePoint Server 2013) (http://technet.microsoft.com/library/gg298949.aspx).

To Configure the Power Pivot Service Application

1. In SharePoint central administration, click Manage Service Applications.
2. Click the name of your Power Pivot Service Application, for example "Default Power Pivot Service Application".
3. Click **Configure service application settings** in the Actions section.
4. In the **Data Refresh** section, set the **Power Pivot Unattended Data Refresh Account**to**PowerPivotDataRefresh** and then click **OK**.

To configure the workbook

1. Browse to your workbook in the Power Pivot Gallery and click **Manage Data Refresh**.
2. If you see the **Data Refresh History** page, click **Configure Schedule**.
3. Click **Enable**.
4. Click **Also Refresh as soon as possible**.
5. In the **Credentials** section, click **Use the Data refresh account configured by the administrator**.
6. Clear **All data sources**.
7. Select **Refresh** for the data source that uses the Oracle data. The name of the data source name can be changed in Microsoft Excel in the **Data**, **Connections**, **Properties** menu.
8. Under your data source, Select **Use Default Schedule**.

9. Select **Connect using the credentials saved in Secure Store Service (SSS) to log on to the data source. Enter the ID used to look up the credentials in the SSS ID box**.

10. In the **ID:** box, type **OracleAuthentication**.

11. Click **OK**.

    If you see an error message similar to:
    `The provided Secure Store target application is either incorrectly configured or does not exist`.

    The two common solutions are:

    - Verify that the Target Application ID is correct.
    - Verify that you set credentials for the Target Application.

To Verify Data Refresh with the new authentication

When you click **ok**, you see the **Refresh History** page. Within a few minutes, you should see a new item in the refresh history because in the previous steps you selected **Also Refresh as soon as possible**. The default value for the timer job **Power Pivot Data Refresh Timer Job** is 1 minute. If you do not see a new item in the refresh history, wait a few minutes and refresh your browser. If you still do not see the new item, verify the current value of the timer job.

More Information

- Configure the Secure Store Service in SharePoint 2013.
- See the "Scheduled Data Refresh" section of Power Pivot Data Refresh with SharePoint 2013 and SQL Server 2012 SP1 (Analysis Services).

Power Pivot Data Feeds

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A data feed provides a stream of data on demand from an online data source to a client application. In Analysis Services, you can use data feeds to add data from an Atom-aware Web service or application to a Power Pivot data source in an Excel workbook.

This section provides information to help you make best use of data feeds to get existing corporate or workgroup data into Power Pivot workbooks. It also provides information on how share data feeds with others in your organization, and how data feeds are constructed and used in case you need to troubleshoot any problems that arise.

Related Tasks

Create or Customize a Data Feed Library (Power Pivot for SharePoint)

Delete a Power Pivot Data Feed Library

Use Data Feeds (Power Pivot for SharePoint)

Share Data Feeds Using a Data Feed Library (Power Pivot for SharePoint)

Create or Customize a Data Feed Library (Power Pivot for SharePoint)

11/28/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A *data feed library* is a special-purpose SharePoint library that enables you to register and share Atom data service documents (.atomsvc). These documents provide XML data feeds to Power Pivot workbooks or other client applications that support the Atom data feed format. A data feed library is different from other SharePoint libraries because it gives you the ability to:

- Create or edit a *data service document*, used to specify an HTTP connection to a specific feed.
- Share and manage data service documents in a central location.
- Visually identify data service documents by an icon, so that you can easily distinguish service documents from other documents stored in the same library:

A data feed library always contains data service document (.atomsvc) files, and never the data feed itself. Unlike a data feed, which consists of static XML data, the data service document specifies a URL to a service or application that generates a feed upon request, providing reusable connection information for repeatable import operations.

This topic contains the following sections:

Prerequisites

Create a New Data Feed Library

Add the Data Feed Content Type to Any Library

Prerequisites

Power Pivot feature integration must be activated for the sites for which you are creating the data feed library. If the data feed library template type is not available, the most likely cause is that this prerequisite has not been met. For more information, see Activate Power Pivot Feature Integration for Site Collections in Central Administration.

You must be a site owner to create the library.

Create a New Data Feed Library

Creating a data feed library is the first step to enabling data feeds for Power Pivot workbooks. Because a data feed library provides application and management pages for data service documents, you must have this library in place before you can create a new document.

A data feed library is based on a built-in template and a preconfigured *data service document content type* that defines properties and behaviors for a data service document.

1. Click **Site Actions** at the top left corner of the page.
2. Click **More Options**...
3. Under Libraries, click **Data Feed Library**.
4. Type the name, description, launch, and version preferences. Include descriptive information to help users identify this library as storage for data service documents.

5. Click **Create**.

A link to the data feed library will appear in the navigation Quick Launch pane for the current site.

After you create a library, you can use it to create data service documents. For more information, see Use Data Feeds (Power Pivot for SharePoint).

Add the Data Feed Content Type to Any Library

If you do not want to create a dedicated data feed library, but you still want to create and manage data service documents from a SharePoint site, you can manually add and configure the data service document content type for any library that you will use to share data service document (.atomsvc) files.

You must have at least the Manage Lists permission to add and configure a content type. This permission is built into the Design permission level and above.

The following steps must be repeated for each library in which you want to create or edit data feed registration documents.

Step 1: Enable Content Type Management

1. Open the document library for which you want to enable multiple content types.
2. On the SharePoint ribbon, in Library Tools, click **Library**.
3. Click **Settings**.
4. Click **Library Settings**.
5. In General Settings, click **Advanced settings**.
6. In Content Types, in the "Allow management of content types?" section, click **Yes**.
7. Click **OK**.

Step 2: Add the Data Service Document Content Type

1. In the Content Types section, click **Add from existing site content types**. If you do not see this page, go back to the site, click **Library** in Library Tools, and then click **Library Settings**.
2. In Content Types, click **Add from existing site content types**.
3. In Select site content types from:, select **Business Intelligence**.
4. In Available Site Content Types, click **Data Service Document**, and then click **Add** to move the selected content type to the Content types to add list.
5. Click **OK**.

Step 3: Verify Data Service Document Configuration

1. Open the site home page.
2. Open the library.
3. On the SharePoint ribbon, click **Documents**.
4. Click the down arrow on New Document, and select **Data Service Document**. The New Data Service Document page should appear.

See Also

Use Data Feeds (Power Pivot for SharePoint)
Delete a Power Pivot Data Feed Library

Power Pivot Server Administration and Configuration in Central Administration
Power Pivot Data Feeds

Use Data Feeds (Power Pivot for SharePoint)

7/11/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data feeds are one or more data streams that are generated from an online data source and streamed to a destination document or application. If you are using Power Pivot for Excel, data feeds can help you get existing corporate or business data from arbitrary data sources into the Power Pivot window in your Excel 2010 workbook. After you import a data feed to a workbook, you can reference it later in any data refresh operations that you schedule on a SharePoint server.

How you use a data feed depends on whether you are using built-in export features in applications that support Atom data feeds, or creating and using custom data services. Applications that are able to publish and read Atom XML data provide seamless data transfer that hides the mechanics of data feeds and data services from users. To a user, he or she is simply moving data from one application to another.

SQL Server 2008 R2 Reporting Services and Microsoft SharePoint 2010 provide data feeds that can be used in Power Pivot workbooks. You can use the information in this topic to learn how to access data feeds from reports and lists that you already have.

This topic contains the following sections:

Prerequisites

Create a Data Feed from a SharePoint List

Create a Data Feed from a Reporting Services Report

Create a Data Feed from a Data Service Document

Prerequisites

You must have the Power Pivot for Excel to import a data feed into Excel 2010.

You must have a Web service or a data service that provides data in the Atom 1.0 format. Both SQL Server 2008 R2 Reporting Services and SharePoint 2010 can provide data in this format.

Before you can export a SharePoint list as a data feed, you must install ADO.NET Data Services on the SharePoint server. For more information, see Install ADO.NET Data Services to support data feed exports of SharePoint lists.

Create a Data Feed from a SharePoint List

In a SharePoint 2010 farm, a SharePoint list has an Export as Data Feed button on the List ribbon. You can click this button to export the list as a feed. For best results, you should have Excel 2010 with the Power Pivot client application on your workstation. The Power Pivot client application will launch in response to the data feed export, creating a new Power Pivot table that contains the list.

1. Open the list on your SharePoint site.
2. In List Tools, click **List**.
3. In Connect and Export, click **Export as Data Feed**.

> **NOTE**
> The **Export as Data Feed** button is added to SharePoint by way of Power Pivot. If you do not have Power Pivot for SharePoint installed or you did not activate the Power Pivot feature, this button will not be available.

4. Click **Open** if Power Pivot for Excel is installed locally, or click **Save** to save the .atomsvc document to your hard drive for import operations at a later time.

5. If you chose **Open**, use the Table Import Wizard to import the data feed to a worksheet. The data feed will be added as a new table in the Power Pivot window.

An error will occur if ADO.NET Data Services 3.5.1 is not installed on the SharePoint server. For more information about the error and how to resolve it, see Install ADO.NET Data Services to support data feed exports of SharePoint lists.

Create a Data Feed from a Reporting Services Report

If you have a deployment of SQL Server 2008 R2 Reporting Services, you can use the new Atom rendering extension to generate a data feed from an existing report. For best results, you should have Excel 2010 with the Power Pivot for Excel on your workstation. The Power Pivot client application will launch in response to the data feed export, automatically adding and relating the tables and columns as they are streamed in.

For instructions on how to export a data feed from a report, see Generate Data Feeds from a Report (Report Builder and SSRS) in the Report Builder help file.

> **NOTE**
> To set up a recurring data refresh schedule that re-imports report data into a Power Pivot workbook that is published to a SharePoint library, the report server must be configured for SharePoint integration. For more information about using Power Pivot for SharePoint and Reporting Services together, see Configuration and Administration of a Report Server (Reporting Services SharePoint Mode).

Create a Data Feed from a Data Service Document

If you have a custom data service that generates Atom feeds, you can set up a data service document as a way to make the data available to users and applications. A *data service document* (.atomsvc) file specifies one or more connections to online sources that publish data in the Atom wire format. Data service documents can be created in a *data feed library*, which is a special-purpose library that provides a common access point for browsing data service documents that have been published to a SharePoint server. Information workers who have permission to access data service documents in the data feed library can reference the document's SharePoint URL to import the data feeds to their workbooks and applications.

1. Open a data feed library that was created by your site administrator. For more information, see Create or Customize a Data Feed Library (Power Pivot for SharePoint).

2. In Library Tools, click **Documents**.

3. Click **New Document**.

4. Provide a file name and description.

5. Specify one or more URLs that provide the feed:

   a. **Base URL** is optional. You should specify it if a data service document provides multiple feeds. Base URL should specify the portion of the URL that is common to all the feeds (for example, the server

name and site). If you are creating a data service document to a Reporting Services report, the base URL would be the report server URL and report.

b. **Web Service URL** is required. Without the Base URL, this value must include `http://` or `https://` in the address. If you specified a Base URL, the Web service URL is the portion that follows the Base URL. For example, if the full URL is `http://adventure-works/inventory/today.aspx`, the Base URL would be `http://adventure-works/inventory`, and the Web service URL would be /today.aspx.

   The Web service URL can include parameters that filter or select a subset of data. The application or service that provides the feed must support the parameters that you specify in the URL.

6. Enter a **Table Name**, one table for each feed. This value is required. The table name is used by a client application that consumes the data feed. In Power Pivot for Excel, the table name is used to name tables in the Power Pivot window that will contain the imported data.

See Also

Activate Power Pivot Feature Integration for Site Collections in Central Administration
Share Data Feeds Using a Data Feed Library (Power Pivot for SharePoint)

Delete a Power Pivot Data Feed Library

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You must be the site owner to delete a data feed library. If the library contains items, they will be deleted along with the library.

You must know the name of the data feed library you want to delete.

Delete a Data Feed Library

1. Click **Site Actions**.
2. Click **Site Settings**.
3. Under Site Administration, click **Sites libraries and lists**.
4. Click **Customize "<your library name>"**
5. Under Permissions and Management, click **Delete this library**.

See Also

Power Pivot Data Feeds

Share Data Feeds Using a Data Feed Library (Power Pivot for SharePoint)

5/16/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A data feed is an XML data stream that is generated from a service or application that exposes data in the Atom wire format. Increasingly, it is used to transport data between applications and to client-side viewers. In a Power Pivot for SharePoint deployment, data feeds are used to populate a Power Pivot data source with data from an Atom-aware application or service.

If you already use a combination of Atom-aware applications, you might never need to know how feeds are generated and consumed because the data transfer is seamless between the applications. However, organizations that use custom solutions to publish Atom feeds often need a way to make feeds available to information workers. One way to do that is to create and share data service document (.atomsvc) files that provide connections to the online sources that produce the feeds. A special-purpose library, called a data feed library, supports creating and sharing data service documents in a SharePoint web application.

This topic contains the following sections:

Prerequisites

Create a Data Service Document

Secure a Data Service Document

Modify a Data Service Document

Next Step: Use a Data Service Document

> **NOTE**
> Although data feeds are used to add Web data to a Power Pivot data source that you create in a Power Pivot for Excel, any client application that can read an Atom feed can process a data service document.

Prerequisites

You must have a deployment of SQL Server 2008 R2 Power Pivot for SharePoint that adds Power Pivot query processing to a SharePoint farm. Data Feed support is deployed through the Power Pivot solution package.

You must have a SharePoint library that supports the data service document content type. A default Data Feed library is recommended for this purpose, but you can manually add the content type to any library. For more information, see Create or Customize a Data Feed Library (Power Pivot for SharePoint).

You must have a data service or online data source that provides XML tabular data in the Atom 1.0 format.

You must have Contribute permissions on a SharePoint site to create or manage a data service document in a SharePoint library.

Create a Data Service Document

A data service document is a standing request to stream data upon request from an online data source or

application that provides data in a feed format. When you create a data service document, you specify a pointer to one or more URL addressable data services that provide XML tabular in Atom syndicated format.

A single document can specify multiple data feeds. This is useful if you want to retrieve a set of data payloads from the same service, or even from different services, in a single import operation.

1. On a SharePoint site, open the Data Feed library or another document library to which you have added and configured the data service content type. To find a Data Feed library that was previously created, click **View All** on the Quick Launch.
2. On the ribbon at the top of the page, in Document Tools, click **Documents**.
3. Click **New Document,** and then select **Data Service Document**.
4. In the New Data Service Document page, enter the following information:
   a. A name and description for the data service document. Be sure to provide sufficient detail so that users can determine whether to use the feed.
   b. In Data Feed, enter a URL to a data service or Web application that provides data in the Atom 1.0 format.

      The URL must resolve to a service that returns structured or semi-structured data in rows and columns. The service should return data anonymously or via the security credentials of the current user.

      The URL must resolve to a service that supports Windows Authentication, Basic authentication, or anonymous access. The user who imports the feed specifies which scheme to use. Integrated security is selected by default.

      A data feed URL can include parameters. Different types of data service technologies support advanced URL addressing schemes that allow you to precisely select the data you want to use. For example, an ADO.NET data service provides URL parameters for specifying entities, associations, and navigation paths in the underlying data. By specifying a complex URL as a source of a data feed, you can precisely specify the dataset you want to use.
   c. For the same data feed, enter a table name that subsequently identifies the dataset in a client application. In the Power Pivot for Excel, each data feed that you import is placed in its own table control in an Power Pivot data source. You must specify the name of the table that receives the imported data when you set up the data feed.
5. Click "Add another data feed" to repeat the previous steps for specifying additional feeds from the same service or a different service.

   Each data service document is processed as a single operation. All of the data feeds in the document will be generated asynchronously and returned to a client application in the same operation. For this reason, only specify the URL-table pairs for data feeds that you want to use together.

   Because authentication schemes are set at the data service document level, each additional data feed must originate from service or application that supports the same authentication scheme as the first feed. All feeds within the same data service document will be authenticated under the same method at run time.
6. Save the document. The data service document is stored as a physical file (.atomsvc) in a content library that is configured for this content type.

To use the data service document, you can open an Power Pivot workbook in the Power Pivot for Excel and choose the **From Data Feed** option in the Import Data wizard. When prompted, a user will specify the SharePoint URL of the data service document to start a data import operation. For more information, see Use Data Feeds (Power Pivot for SharePoint).

Secure a Data Service Document

A data service document inherits the permissions of the library that contains it. Permissions that you set on the item will determine whether a user can open, modify or delete the data service document.

To use a data service document as a data feed import in the Power Pivot client application, a user only needs view permissions on the document. View permissions are sufficient to resolve the URL in the Import Wizard.

View permissions on a data service document are checked only when a data feed import operation begins. Post-import, permissions on the document are not checked on an ongoing basis; feeds that were added to a Power Pivot data source exist as static data, disconnected from the data service document that provided the original connection information.

Similarly, any data refresh operations that you subsequently schedule also exclude the data service document. At the time of import, connection information for each feed is copied into the Power Pivot data source for refresh purposes. As such, permissions on a data service document are not checked for data refresh, because the document itself is never referenced in a refresh operation.

TASK	SHAREPOINT PERMISSION REQUIREMENTS
Import data feeds to a Power Pivot-enabled workbook.	View permissions to the data service document in a library.
In the Power Pivot client application, refresh data that was previously retrieved via a feed.	Not applicable. The Power Pivot client application uses embedded HTTP connection information to connect directly to the data services and applications that provide the feed. The Power Pivot client application does not use the data service document.
In a SharePoint farm, refresh data as a scheduled task, with no user input required.	Not applicable. The Power Pivot service uses embedded HTTP connection information to connect directly to the data services and applications that provide the feed. The Power Pivot service does not use the data service document.
Delete a data service document in a library	Contribute permissions on the library.

Modify a Data Service Document

You can add, edit, or remove individual URL-table entries in a data service document. After you save your changes, users who select the service document in a new import operation will get the data feeds you specified.

Power Pivot workbooks that used a previous version of the document are unaffected by any changes you make. This is because a data service document is read only once during the initial import operation. During the import, service URL and table names are copied and stored internally in the workbook. These internal values are then used in subsequent refresh operations to get updated data.

Because there is no persistent link between a data service document on a SharePoint site and the Power Pivot workbook that contains the imported feed, modifying any part of a data service document has no effect on existing Power Pivot workbooks.

> **IMPORTANT**
>
> Although the data service document is read only once, data services that provide the actual data can be accessed at regular intervals to get newer feeds. For more information about how to refresh data, see Power Pivot Data Refresh.

Next Step: Use a Data Service Document

To use a data service document that you created in a SharePoint library, you use the **From Data Feeds** import option in a Power Pivot data source. For instructions, see Use Data Feeds (Power Pivot for SharePoint).

See Also

Power Pivot Data Feeds

Power Pivot BI Semantic Model Connection (.bism)

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A BI semantic model connection (.bism) is a portable connection that connects Excel or Power View reports to an Analysis Services tabular model database or an Analysis Services instance in multidimensional mode. If you are familiar with office data connection (.odc) files, you will notice a similarity in how a .bism connection file is defined and used.

A BI semantic model connection is created and accessed via SharePoint. Creating BI semantic model connections enables quick launch commands on a BI semantic model connection in a library. Quick launch commands open a new Excel workbook or options for editing the connection file. If Reporting Services is installed, you will also see a command to create a Power View report.

Supported databases

A BI semantic model connection points to tabular model data. There are three sources for this data:

- A tabular model database running on a standalone Analysis Services instance in tabular server mode. A deployment of a standalone Analysis Services instance is external to the farm. Accessing data sources off the farm requires additional permissions, which you can read about in this topic: Create a BI Semantic Model Connection to a Tabular Model Database.
- Power Pivot workbooks saved to SharePoint. Embedded Power Pivot databases inside Excel workbooks are equivalent to tabular model databases that run on a standalone Analysis Services tabular mode server. If you already use Power Pivot for Excel and Power Pivot for SharePoint, you can define a BI semantic model connection that points to Power Pivot workbooks in a SharePoint library and build Power View reports using existing Power Pivot data. You can use workbooks created in either SQL Server 2008 R2 or SQL Server 2012 (11.x) versions of Power Pivot for Excel.
- A multidimensional data model on an Analysis Services instance.

For a comparison of the data sources, see the community content Understanding the SQL Server 2012 BI Semantic Model (BISM).

Understanding the Connection Sequence for BI Semantic Connections

This section explains the connection behavior between various client applications, such as the Excel desktop application or the Power View reporting client on SharePoint, and a tabular model database inside or outside the SharePoint farm.

All connections to a tabular model database are made using the credentials of the user who is requesting the data.

However, the mechanics of that connection will vary depending on whether the connection is an in-farm connection, a single or double-hop connection, and whether Kerberos is enabled. For more information about authenticated connections between SharePoint and backend data sources, see Double-hop authentication: Why NTLM fails and Kerberos works.

Connecting from Excel to tabular data on a network

When an Excel user specifies a BI semantic model connection as a data source, the connection information inside the .bism file is downloaded to the client application, which then issues its own direct request to the tabular model database on Analysis Services. To access the .bism connection, the Excel user must be a SharePoint user with read permissions on the .bism connection file. Once the connection information is downloaded, all subsequent connections bypass SharePoint, flowing directly from Excel to the backend tabular model database.

The following illustration shows this connection sequence. It starts with a request for the .bism connection, followed by the download of connection information to the client, and finally the single-hop connection to the database. The connection is made using the Windows credentials of the Excel user, who has read permissions on the Analysis Services database. It is a single hop, so even if Kerberos is enabled, it is not required for this scenario.

Connecting from Power View to tabular data on a network

When a SharePoint user clicks on a BI semantic connection in a document library, Power View (if it is installed), starts immediately and opens a connection to the tabular model database.

Connections between Power View and a tabular model database follow a double-hop authentication sequence where the user identity is flowed from the client to SharePoint, and then from SharePoint to a back-end Analysis Services tabular model database that runs outside of the farm. The ADOMD.NET client library that handles the connection request always tries Kerberos on the first attempt. If Kerberos is configured, the user identity is impersonated on the connection to the tabular model database, and the connection succeeds.

If Kerberos is not configured and the request fails, Reporting Services makes a second attempt. Under this scenario, the client library connects to Analysis Services using the Reporting Services service identity and NTLM authentication. The identity of the Power View user is passed on the connection string using the **effectiveusername** parameter.

Only a member of the system administrator role on the Analysis Services instance has permission to make a connection using the **effectiveusername** parameter and impersonate another user on the server instance. For this reason, the execution account of the Reporting Services shared service must have administrative rights on the Analysis Services instance. Instructions for granting administrative permissions to the service account is provided in this topic, Create a BI Semantic Model Connection to a Tabular Model Database.

The following illustration shows a connection sequence that uses the same Windows user identity for each connection. On the last connection to Analysis Services, the connection is made by the Reporting Services service application identity, passing the Windows user identity using **effectiveusername**.

Connecting from Power View to Power Pivot data in SharePoint

When a SharePoint user clicks on a BI semantic connection that resolves to a Power Pivot workbook in the same

farm, the connections occur within the context of the SharePoint environment. A Power Pivot service application handles the connection request, which it forwards to the Analysis Services instance on the same computer. The Analysis Services instance extracts the Power Pivot data from the workbook and loads it. All subsequent connections are managed by Power Pivot service applications in the farm.

In this scenario, all connections occur within the same farm, so there is no requirement for Kerberos or constrained delegation.

Related Tasks

Add a BI Semantic Model Connection Content Type to a Library (Power Pivot for SharePoint)

Create a BI Semantic Model Connection to a Power Pivot Workbook

Create a BI Semantic Model Connection to a Tabular Model Database

Use a BI Semantic Model Connection in Excel or Reporting Services

See Also

Determine the Server Mode of an Analysis Services Instance
Connect to Analysis Services

Add BI Semantic Model Connection Content Type to Library

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A BI semantic model connection is created in SharePoint and provides redirection to business intelligence semantic model data in a Power Pivot workbook or Analysis Services tabular model database on a network server. Before you can create a BI semantic model connection in SharePoint, you must extend a document library to allow the creation of a .bism file. This step only needs to be performed once for each library, but you will need to repeat it for any library from which you want to create .bism files. Best practices recommend that you create a centralized library for storing .bism files so that you can manage permissions in one place.

> **NOTE**
> If you already use SharePoint Data Connection Libraries, the BI Semantic Model Connection content type is automatically added to that library template. You can skip the steps in this section if you use a data connection library that already lets you create new BI semantic model connection documents.

Add the content type to a document library

You must have at least the Manage Lists permission to add and configure a content type. This permission is built into the Design permission level and above.

The site that contains the document library must have feature activation for Power Pivot for SharePoint. For more information, see Activate Power Pivot Feature Integration for Site Collections in Central Administration.

1. Open the document library for which you want to enable the BI Semantic Model Connection content type.
2. On the SharePoint ribbon, in Library Tools, click **Library**.
3. Click **Library Settings**.
4. In General Settings, click **Advanced settings**.
5. In Content Types, in the "Allow management of content types?" section, click **Yes**.
6. Click **OK**.
7. In the Content Types section, click **Add from existing site content types**. If you do not see this page, go back to the site, click **Library** in Library Tools, and then click **Library Settings**.
8. In Content Types, click **Add from existing site content types**.
9. In Select site content types from:, select **Business Intelligence**.
10. In Available Site Content Types, click **BI Semantic Model Connection File**, and then click **Add** to move the selected content type to the Content types to add list.
11. Click **OK**.
12. To verify you added the content type, go back to the library and click **New Document** on the Documents area of the library ribbon. You should see **BI Semantic Model Connection File** in the New Documents list.

After you enable the BI semantic model connection content type for a library, you can create a connection that provides redirection to business semantic model data that can be used by Excel or Power View reports. Choose from the following links to learn more about this next step:

Create a BI Semantic Model Connection to a Power Pivot Workbook

Create a BI Semantic Model Connection to a Tabular Model Database

See Also

Power Pivot BI Semantic Model Connection (.bism)
Use a BI Semantic Model Connection in Excel or Reporting Services

Create a BI Semantic Model Connection to a Power Pivot Workbook

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use the information in this topic to set up a BI semantic model connection that redirects to a Power Pivot workbook in the same farm.

After you create a BI semantic model connection and configure SharePoint permissions, you can use it as a data source for Excel or Power View reports.

This topic includes the following sections. Perform each task in the order given.

Review Prerequisites

Create a Connection

Configure SharePoint Permissions on the BI Semantic Model Connection

Configure SharePoint Permissions on the Workbook

Next Steps

Review Prerequisites

You must have Contribute permissions or above to create a BI semantic model connection file.

You must have a library that supports the BI semantic model connection content type. For more information, see Add a BI Semantic Model Connection Content Type to a Library (Power Pivot for SharePoint).

You must know the URL of the Power Pivot workbook for which you are setting up a BI semantic model connection (for example, `http://adventure-works/shared documents/myworkbook.xlsx`). The workbook must be in the same farm.

All computers and users that participate in the connection sequence must be in the same domain or trusted domain (two-way trust).

Create a Connection

1. In the library that will contain the BI semantic model connection, click **Documents** on the SharePoint ribbon. Click the down arrow on New Document, and select **BISM Connection File** to open the New BI Semantic Model Connection page.

2. Set the **Server** property to the SharePoint URL of the Power Pivot workbook (for example, `http://mysharepoint/shared documents/myWorkbook.xlsx`. In a Power Pivot for SharePoint deployment, data

can be loaded on any server in the farm. For this reason, data source connections to Power Pivot data specify just the path to the workbook. The Power Pivot System Service determines which server loads the data.

Do not use the **Database** property; it is not used when specifying the location of a Power Pivot workbook.

Your page should look similar to the following illustration.

Optionally, if you have SharePoint permissions to the workbook, an extra validation step is performed, ensuring that the location is valid. If you do not have permission to access the data, you are given the option of saving the BI semantic model connection without the validation response.

Configure SharePoint Permissions on the BI Semantic Model Connection

Ability to use a BI semantic model connection as a data source for an Excel workbook or Reporting Services report requires **Read** permissions on the BI semantic model connection item in a SharePoint library. The Read permission level includes the **Open Items** permission that enables downloading BI semantic model connection information to an Excel desktop application.

There are several ways to grant permissions in SharePoint. The following instructions explain how to create a new group called **BISM Users** that have the **Read** permission level.

You must be a site owner to change permissions.

1. In Site Actions, click **Site Permissions**.
2. Click **Create Group** and name the new group **BISM Users**.
3. Choose the **Read** permission level and click **Create**.
4. Select **BISM Users** in People and Groups.
5. Point to New, click **Add Users**, and then add user or group accounts.

   These users and groups will now have Read permissions throughout the site, including all libraries and lists that inherit permissions from the site level. If these permissions are too high, you can selectively remove this group from specific libraries, lists, or items.

To selectively remove permissions at the item level, do the following:

1. In a library, select a document. Click the right down arrow and then click **Manage Permissions**.
2. By default, an item inherits permissions. To change the permissions of individual documents in this library, click **Stop Inheriting Permissions**.
3. Select the checkbox next to **BISM Users**.
4. Click **Remove User Permissions**.

Configure SharePoint Permissions on the Workbook

If you are using a Power Pivot database inside an Excel workbook, the SharePoint permissions on the Excel workbook determine data access via the BI semantic model connection. All users who access the workbook must have Read permissions on the workbook in order to use it as an external data source.

If you created a **BISM Users** group using the instructions in the previous section, user and group accounts that are members of **BISM Users** will have sufficient permission on the workbook as well as the BI semantic model connection file, assuming the workbook uses inherited permissions.

Next Steps

After you create and secure a BI semantic model connection, you can specify it as a data source. For more information, see Use a BI Semantic Model Connection in Excel or Reporting Services.

See Also

Power Pivot BI Semantic Model Connection (.bism)
Use a BI Semantic Model Connection in Excel or Reporting Services
Create a BI Semantic Model Connection to a Tabular Model Database

Create a BI Semantic Model Connection to a Tabular Model Database

5/16/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use the information in this topic to set up a BI semantic model connection that redirects to a tabular model database running on an Analysis Services instance outside the SharePoint farm.

After you create a BI semantic model connection and configure SharePoint and Analysis Services permissions, people can use it as a data source for Excel or Power View reports.

This topic includes the following sections. Perform each task in the order given.

Review Prerequisites

Grant Analysis Services Administrative Permissions to Shared Service Applications

Grant Read Permissions on the Tabular Model Database

Create a BI Semantic Model Connection to a Tabular Model Database

Configure SharePoint permissions on the BI Semantic Model Connection

Next Steps

Review Prerequisites

You must have Contribute permissions or above to create a BI semantic model connection file.

You must have a library that supports the BI semantic model connection content type. For more information, see Add a BI Semantic Model Connection Content Type to a Library (Power Pivot for SharePoint).

You must know the server and database name for which you are setting up a BI semantic model connection. Analysis Services must be configured for tabular mode. Databases running on the server must be tabular model databases. For instructions on how to check for server mode, see Determine the Server Mode of an Analysis Services Instance.

In certain scenarios, the shared services in a SharePoint environment must have administrative permissions on the Analysis Services instance. These services include Power Pivot service applications, Reporting Services service applications, and PerformancePoint service applications. Before you can grant administrative permissions, you must know the identity of these service applications. You can use Central Administration to determine the identity.

You must be a SharePoint service administrator to view security information in Central Administration.

You must be an Analysis Services system administrator to grant administrative rights in Management Studio.

Power Pivot for SharePoint must be accessed via web applications that use classic authentication mode. BI semantic model connections to external data sources have a dependency on classic mode sign-in. For more information, see Power Pivot Authentication and Authorization.

All computers and users that participate in the connection sequence must be in the same domain or trusted domain (two-way trust).

Grant Analysis Services Administrative Permissions to Shared Service Applications

Connections that originate from SharePoint to a tabular model database on an Analysis Services server are sometimes made by a shared service on behalf of the user requesting the data. The service making the request might be a Power Pivot service application, a Reporting Services service application, or a PerformancePoint service application. In order for the connection to succeed, the service must have administrative permissions on the Analysis Services server. In Analysis Services, only an administrator is allowed to make an impersonated connection on behalf of another user.

Administrative permissions are necessary when the connection is used under these conditions:

- When verifying the connection information during the configuration of a BI semantic model connection file.
- When starting a Power View report using a BI semantic model connection.
- When populating a PerformancePoint web part using a BI semantic model connection.

To ensure these behaviors perform as expected, grant to each service identity administrative permissions on the Analysis Services instance. Use the following instructions to grant the necessary permission.

Add service identities to the Server Administrator role

1. In SQL Server Management Studio, connect to the Analysis Services instance.
2. Right-click the server name and select **Properties**.
3. Click **Security**, and then click **Add**. Enter the Windows user account that is used to run the service application.

   You can use Central Administration to determine the identity. In the Security section, open the **Configure service accounts** to view which Windows account is associated with the service application pool used for each application, then follow the instructions provided in this topic to grant the account administrative permissions.

Grant Read Permissions on the Tabular Model Database

Because the database is running on a server that is external to the farm, part of setting up your connections will include granting database user permissions on the backend Analysis Services server. Analysis Services uses a role-based permission model. Users who connect to model databases must do so with Read permissions or higher, through a role that grants read access to its members.

Roles, and sometimes role membership, are defined when the model is created in SQL Server Data Tools (SSDT). You cannot use SQL Server Management Studio to create roles, but you can use it to add members to a role that is already defined. For more information about creating roles, see Create and Manage Roles.

Assign role membership

1. In SQL Server Management Studio, connect to the instance of Analysis Services, expand the database in Object Explorer, and then expand **Roles**. You should see a role that is already defined. If a role does not exist, contact the author of the model and request the addition or a role. The model must be redeployed before the role is visible in Management Studio.
2. Right-click the role, and select **Properties**.
3. In the Membership page, add the Windows group and user accounts that require access.

Create a BI Semantic Model Connection to a Tabular Model Database

After you set permissions in Analysis Services, you can return to SharePoint and create a BI semantic model connection.

1. In the library that will contain the BI semantic model connection, click **Documents** on the SharePoint ribbon.

2. Click the down arrow on New Document, and select **BI Semantic Model Connection File** to open the New BI Semantic Model Connection page.

3. Set both **Server** and **Database** properties. If you are unsure of the database name, use SQL Server Management Studio to view a list of the databases that are deployed on the server.

   Server name is either the network name of the server, the IP address, or the fully qualified domain name (for example, myserver.mydomain.corp.adventure-works.com). If the server is installed as a named instance, enter the server name in this format: computername\instancename.

   Database must be a tabular database that is currently available on the server. Do not specify another BI semantic model connection file, an Office Data Connection (.odc) file, an Analysis Services OLAP database, or a Power Pivot workbook. To get the database name, you can use Management Studio to connect to the server and view the list of available databases. Use the property page of the database to ensure you have the correct name.

4. Click **OK** to save the page. At this point, the Power Pivot service application will verify the connection.

   Verification succeeds if the connection information is correct, and you have granted administrative permissions to the Power Pivot service application so that it can connect to Analysis Services as the current user.

   Verification fails if the connection information is wrong, or the service application lacks permissions. A validation message will appear on the page asking whether you want to save the file. If you know that the connection is valid, you should save the file anyway, because the error is the result of missing permissions rather than invalid connection information.

   You can verify the connection by using it in Excel or Power View to connect to tabular model database. If the data source connection succeeds, the connection is valid despite the verification warning.

Configure SharePoint permissions on the BI Semantic Model Connection

Ability to use a BI semantic model connection as a data source for an Excel workbook or Reporting Services report requires **Read** permissions on the BI semantic model connection item in a SharePoint library. The Read permission level includes the **Open Items** permission that enables downloading BI semantic model connection information to an Excel desktop application.

There are several ways to grant permissions in SharePoint. The following instructions explain how to create a new group called **BISM Users** that have the **Read** permission level.

You must be a site owner to change permissions.

1. In Site Actions, click **Site Permissions**.

2. Click **Create Group** and name the new group **BISM Users**.

3. Choose the **Read** permission level and click **Create**.

4. Select **BISM Users** in People and Groups.

5. Point to New, click **Add Users**, and then add user or group accounts.

These users and groups will now have Read permissions throughout the site, including all libraries and lists that inherit permissions from the site level. If these permissions are too high, you can selectively remove this group from specific libraries, lists, or items.

To selectively remove permissions at the item level, do the following:

1. In a library, select a document. Click the right down arrow and then click **Manage Permissions**.
2. By default, an item inherits permissions. To change the permissions of individual documents in this library, click **Stop Inheriting Permissions**.
3. Select the checkbox next to **BISM Users**.
4. Click **Remove User Permissions**.

Next Steps

After you create and secure a BI semantic model connection, you can specify it as a data source. For more information, see Use a BI Semantic Model Connection in Excel or Reporting Services.

See Also

Power Pivot BI Semantic Model Connection (.bism)
Create a BI Semantic Model Connection to a Power Pivot Workbook

Use a BI Semantic Model Connection in Excel or Reporting Services

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic explains how to use the BI semantic model connections you created using instructions in other topics. If you have not yet created a BI semantic model, see Create a BI Semantic Model Connection to a Power Pivot Workbook and Create a BI Semantic Model Connection to a Tabular Model Database.

Connect from Excel

You can specify a BI semantic model connection as a data source in Excel or any other business application that uses Analysis Services tabular model data. This section explains the two approaches for connecting to BI semantic model data using Excel.

BI semantic model connections from Excel require that you have Excel 2010 and the MSOLAP.5 OLE DB provider installed on your workstation. Additional information about connection requirements is provided further on in this section.

Starting from SharePoint

- Right-click a BI semantic model connection in a library and select **Launch Excel**.

Click **Enable** when prompted to enable data connections. Excel opens a workbook that contains a PivotTable field list populated with fields from the underlying data source.

Starting from Excel

1. Start Excel and open a workbook. On the Data tab, in Get External Data, click **From Other Sources**.
2. Click **From Analysis Services** and use the Data Connection Wizard to import the data.
3. Enter the SharePoint URL of the BI semantic model connection file (for example, `http://mysharepoint/shared documents/myData.bism`). Accept the default log on credentials option, **Use Windows Authentication**. Click **Next**.
4. On the next page, click **Next** again. Although you are prompted to select a database, you can only use the one database that is specified in the BI semantic model connection.
5. On the last page, you can provide a friendly name and description. Click **Finish**, and then click **OK** on the Import Data dialog box to import the data.

For connections to succeed you must have Excel 2010 and the MSOLAP.5.dll installed on the client computer. You

can get the provider by installing the version of Power Pivot for Excel that is current for this release or you can download just the Analysis Services OLE DB provider from the Feature Pack download page.

To confirm that MSOLAP.5.dll is the current version, check **HKEY_CLASSES_ROOT\MSOLAP** in the registry. **CurVer** should be set to MSOLAP.5.

You must also have Read permissions on the BI semantic model file in SharePoint. Read permissions include download rights. Excel downloads the BI semantic model connection information from SharePoint and opens a direct connection to the database via **HTTP Get**. Connection requests do not flow through SharePoint once BI semantic model connection information is stored locally.

If you are connecting to a tabular model database that runs on an Analysis Services server, SharePoint permissions are not sufficient. You must also have database read permissions on the server. This step should have been performed when you created the BI semantic model connection. For more information, see Create a BI Semantic Model Connection to a Tabular Model Database.

Connect from Reporting Services in SharePoint

You can use a BI semantic model connection the same way you use most data sources, by specifying the file as a data source in the document or tool that uses the data. Although a BI semantic model connection points to a physical database on another server, you use the connection file as if it were the data source. The SharePoint URL of the BI semantic model connection is a valid data source location for Power View reports that use BI semantic model data.

For ad hoc report design in SharePoint, the user who creates the report must have SharePoint permissions on the BI semantic model connection (.bism) file and on the business intelligence semantic model database. The security context of the connection is the interactive user who is creating the report.

Errors and Events Reference (Power Pivot for SharePoint)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This section provides information about errors and events for Power Pivot for SharePoint. Errors in this section are identified by the error message text that appears in a log or error window.

To view the list of error messages documented in this section, use the contents pane in Books Online.

See Also

Configure Usage Data Collection for (Power Pivot for SharePoint
Configure and View SharePoint Log Files and Diagnostic Logging (Power Pivot for SharePoint)

An error occurred during an attempt to establish a connection

12/10/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This error occurs if you query Power Pivot data on a server that does not have Power Pivot for SharePoint installed. It also occurs if the SQL Server Analysis Services (Power Pivot) service is stopped, or if you are attempting to view Power Pivot data from an earlier version.

Details

Applies to	Power Pivot for SharePoint
Product Version	SQL Server 2008 R2, SQL Server 2012 (11.x), SQL Server 2014 (12.x)
Cause	The data connection failed.
Message Text	An error occurred during an attempt to establish a connection to the external data source. The following connections failed to refresh: Power Pivot Data

Explanation

Excel Services returns this error when you query Power Pivot data in an Excel workbook that is published to SharePoint, and the SharePoint environment does not have a Power Pivot for SharePoint server, or the SQL Server Analysis Services (Power Pivot) service is stopped.

The error occurs when you slice or filter Power Pivot data when the query engine is not available.

User Action

Install Power Pivot for SharePoint or move the Power Pivot workbook to a SharePoint environment that has Power Pivot for SharePoint installed. For more information, see Power Pivot for SharePoint 2010 Installation.

If the software is installed, verify that the SQL Server Analysis Services (Power Pivot) instance is running. Check **Manage services on server** in Central Administration. Also check the Services console application in Administrative Tools.

For Power Pivot workbooks that were created in a SQL Server 2008 R2 version of Power Pivot for Excel, you must install the SQL Server 2008 R2 version of the Analysis Services OLE DB provider. This error will occur if you installed the provider, but did not register the Microsoft.AnalysisServices.ChannelTransport.dll file. For more information about file registration, see Install the Analysis Services OLE DB Provider on SharePoint Servers.

See Also

The data connection uses Windows Authentication and user credentials could not be delegated. The following connections failed to refresh: Power Pivot Data

Could not load file or assembly Microsoft-Data-Services

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In SharePoint 2010 environments that have Power Pivot for SharePoint, this error will occur if you attempt a data feed export and the system is missing the required version of Microsoft ADO.NET Data Services.

Details

Applies to	Power Pivot for SharePoint
Product Version	SQL Server 2008 R2, SQL Server 2012 (11.x), SQL Server 2014 (12.x)
Cause	ADO.NET Data Services 3.5 SP1 was not found.
Message Text	Could not load file or assembly 'Microsoft.Data.Services, Version=3.5.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089' or one of its dependencies. The system cannot find the file specified

Explanation

SharePoint 2010 includes an Export as Data Feed button that you can use to export a SharePoint list as an XML data feed. In addition, both Reporting Services in SharePoint mode and Power Pivot for SharePoint support data feed features that require ADO.NET Data Services.

If ADO.NET Data Services is not installed, this error will occur when you request a data feed.

User Action

1. Go to the hardware and software requirements documentation for SharePoint 2010, Determine Hardware and Software Requirements (SharePoint 2010) (http://go.microsoft.com/fwlink/?LinkId=169734).
2. In **Installing software prerequisites**, find the link for ADO.NET Data Services 3.5 that corresponds to the operating system you are using.
3. Click the link and run the setup program that installs the service.

See Also

Deploy Power Pivot Solutions to SharePoint

Could not load Microsoft.AnalysisServices.SharePoint.Integration

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In SharePoint 2010 environments that have Power Pivot for SharePoint, this error will occur if the application-level solution for Power Pivot did not deploy correctly.

Details

Applies to	Power Pivot for SharePoint
Product Version	SQL Server 2008 R2, SQL Server 2012 (11.x), SQL Server 2014 (12.x)
Cause	Powerpivotwebapp solution is not deployed or did not deploy correctly.
Message Text	Could not load file or assembly 'Microsoft.AnalysisServices.SharePoint.Integration'

Explanation

Power Pivot for SharePoint uses solution packages to deploy its features on a SharePoint server. One of the solutions is not deployed correctly. As a result, this error appears whenever you try to open Power Pivot Gallery or other Power Pivot application pages on a SharePoint site.

User Action

Deploy the solution package.

1. In Central Administration, in System Settings, click **Manage farm solutions**.
2. Click **Powerpivotwebapp**.
3. Click **Deploy Solution**.
4. Choose the web application for which this error is occurring. If there is more than one web application, redeploy the solution for all of hem.
5. Click **OK**.

See Also

Deploy Power Pivot Solutions to SharePoint

The data connection path is invalid

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

For Excel workbooks that contain Power Pivot data, Excel Services returns this error if it cannot connect to embedded data sources.

Details

Applies to	Power Pivot for SharePoint
Product Version	SQL Server 2008 R2, SQL Server 2012 (11.x), SQL Server 2014 (12.x)
Cause	Excel Services is configured to only allow data connections from .odc files that are in a trusted data connection library.
Message Text	The data connection path in the workbook points to a file on the local drive or is an invalid URI. The following connections failed to refresh: Power Pivot Data

Explanation

Power Pivot workbooks contain embedded data connections. To support workbook interaction through slicers and filters, Excel Services must be configured to allow external data access through embedded connection information. External data access is required for retrieving Power Pivot data that is loaded on Power Pivot servers in the farm.

User Action

Change the configuration settings to allow embedded data source connections.

1. In Central Administration, in Application Management, click **Manage service applications**.
2. Click **Excel Services Application**.
3. Click **Trusted File Location**.
4. Click **http://** or the location you want to configure.
5. In External Data, in Allow External Data, click **Trusted data connection libraries and embedded**.
6. Click **OK**.

Alternatively, you can create a new trusted location for sites that contain Power Pivot workbooks, and then modify the configuration settings for just that site. For more information, see Create a trusted location for Power Pivot sites in Central Administration.

The data connection user could not be delegated

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

For Excel workbooks that contain Power Pivot data, Excel Services returns this error if it cannot connect to a Power Pivot server instance in SharePoint.

Details

Applies to	Power Pivot for SharePoint
Product Version	SQL Server 2008 R2, SQL Server 2012 (11.x), SQL Server 2014 (12.x)
Cause	Connection failed when attempting to use a Power Pivot data provider.
Message Text	The data connection uses Windows Authentication and user credentials could not be delegated. The following connections failed to refresh: Power Pivot Data

Explanation

There are multiple causes for this error message. The common factor behind all of them is that Excel Services cannot get a valid Windows user identity from a claims token in SharePoint. In the case of Excel workbooks that contain Power Pivot data, this error occurs when any of the following conditions exist:

- The Claims to Windows Token Service is not running. You can confirm the cause of this error by viewing the SharePoint log file. If the SharePoint logs include the message "The pipe endpoint 'net.pipe://localhost/s4u/022694f3-9fbd-422b-b4b2-312e25dae2a2' could not be found on your local machine", the Claims to Windows Token Service is not running. To start it, use Central Administration and then verify the service is running in the Services console application.
- A domain controller is not available to validate the user identity. The Claims to Windows Token Service does not use cached credentials. It validates the user identity for each connection. You can confirm the cause of this error by viewing the SharePoint log file. If the SharePoint logs include the message "Failed to get WindowsIdentity from IClaimsIdentity", the user identity could not be authenticated.
- The computers must be members of the same domain or in domains that have a two-way trust relationship.
- You must use Windows domain user accounts. The accounts must have a Universal Principal Name (UPN).
- The Excel Services service account must have Active Directory permissions to query the object.

User Action

Use the following instructions to check the status of the Claims to Windows Token Service.

For all other scenarios, check with your network administrator.

Enable Claims to Windows Token Service

1. In Central Administration, in System Settings, click **Manage services on server**.
2. Select **Claims to Windows Token Service**, and then click **Start**.
3. Verify the service is also running in the Services console:
   a. In Administrative Tools, click Services.
   b. Start the Claims to Windows Token Service if it is not running.

See Also

Configure Power Pivot Service Accounts

Comments, Shapes, other objects not supported by Excel Services

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This error occurs when you add Slicers to a Power Pivot workbook from a Power Pivot Field List.

Details

Applies to	Power Pivot for SharePoint
Product Version	SQL Server 2008 R2, SQL Server 2012 (11.x), SQL Server 2014 (12.x)
Cause	Excel Web Access cannot render the Shape object used to control positioning and formatting of Slicers added to a workbook from the Power Pivot Field List.
Message Text	The following features are not supported by Excel Services and may not display or may display only partially: Comments, Shapes, or other objects Some features, such as external data queries, display cached data which can only be refreshed in Microsoft Excel.

Explanation

Excel Web Access shows this error when you open a Power Pivot workbook in a browser and you click the **Details** button for the message, **Unsupported Features This workbook may not display as intended**.

This error occurs because the Power Pivot workbook contains Slicers whose layout is controlled by a hidden Shape object in Excel. The Shape object controls the formatting and positioning of the Slicers in horizontal and vertical placements.

Excel Services cannot render a Shape object, but because the object is hidden, the fact that it does not render is not an issue.

User Action

This error can be ignored. Click **OK** to close the error message and proceed to use the workbook and Slicers with no problem.

Trusted location does not allow external data connections

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

For Excel workbooks that contain Power Pivot data, Excel Services returns this error if it cannot connect to embedded data sources.

Details

Applies to	Power Pivot for SharePoint
Product Version	SQL Server 2008 R2, SQL Server 2012 (11.x), SQL Server 2014 (12.x)
Cause	Excel Services is configured to deny external data access.
Message Text	The trusted location where the workbook is stored does not allow external data connections. The following connections failed to refresh: Power Pivot Data

Explanation

Power Pivot workbooks contain embedded data connections. To support workbook interaction through slicers and filters, Excel Services must be configured to allow external data access through embedded connection information. External data access is required for retrieving Power Pivot data that is loaded on Power Pivot servers in the farm.

User Action

Change the configuration settings to allow embedded data sources.

1. In Central Administration, in Application Management, click **Manage service applications**.
2. Click **Excel Services Application**.
3. Click **Trusted File Location**.
4. Click **http://** or the location you want to configure.
5. In External Data, in Allow External Data, click **Trusted data connection libraries and embedded**.
6. Click **OK**.

Alternatively, you can create a new trusted location for sites that contain Power Pivot workbooks, and then modify the configuration settings for just that site. For more information, see Create a trusted location for Power Pivot sites in Central Administration.

This workbook contains one or more queries that refresh external data

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

For Excel workbooks that contain Power Pivot data, Excel Services shows this warning when it detects connection information and prompts you to enable or disable queries for this workbook.

Details

Product Name	Power Pivot for SharePoint
Product Version	SQL Server 2008 R2, SQL Server 2012 (11.x), SQL Server 2014 (12.x)
Cause	Excel Services is configured to warn on data refresh.
Message Text	This workbook contains one or more queries that refresh external data. A malicious user can design a query to access confidential information and distribute it to other users or perform other harmful actions. If you trust the source of this workbook, click Yes to enable queries to external data in this workbook. If you are not sure, click No so that changes are not applied to your workbook. Do you want to enable queries to external data in this workbook?

Explanation

Power Pivot workbooks contain embedded data connection strings that are used by Excel to communicate with an external Power Pivot server that loads and calculates the data. When warn on data refresh is enabled, Excel Services detects this connection string and warns the user accordingly.

To filter and slice Power Pivot data in the workbook, queries must be enabled. Be sure that you enable queries for only those Power Pivot workbooks that you trust.

User Action

Click **Yes** to enable queries.

You can also change the configuration settings so that warn on refresh no longer occurs:

1. In Central Administration, in Application Management, click **Manage service applications**.
2. Click **Excel Services Application**.
3. Click **Trusted File Location**.
4. Click **http://** or the location you want to configure.

5. In External Data, clear the checkbox for **Warn on data refresh**.

6. Click **OK**.

Alternatively, you can create a new trusted location for sites that contain Power Pivot workbooks, and then modify the configuration settings for just that site. For more information, see Create a trusted location for Power Pivot sites in Central Administration.

Unable to refresh data for a data connection in the workbook

12/10/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

For Excel workbooks that contain Power Pivot data, Excel Services returns this error if it submits a connection request to a Power Pivot server and the request fails.

Details

Applies to:	Power Pivot for SharePoint installation
Product Version	SQL Server 2008 R2, SQL Server 2012 (11.x), SQL Server 2014 (12.x)
Cause	See below.
Message Text	Unable to refresh data for a data connection in the workbook. Try again or contact your system administrator. The following connections failed to refresh: Power Pivot Data

Explanation and Resolution

Excel Services cannot connect to or load the Power Pivot data. Conditions that cause this error to occur include the following:

Scenario 1: Service is not started

The SQL Server Analysis Services (Power Pivot) instance is not started. An expired password will cause the service to stop running. For more information about changing the password, see Configure Power Pivot Service Accounts and Start or Stop a Power Pivot for SharePoint Server.

Scenario 2a: Opening an earlier version workbook n the server

The workbook you are attempting to open might have been created in the SQL Server 2008 R2 version of Power Pivot for Excel. Most likely, the Analysis Services data provider that is specified in the data connection string is not present on the computer that is handling the request.

If this is the case, you will find this message in the ULS log: "Refresh failed for ' Power Pivott Data' in the workbook '<URL to workbook>'", followed by "Unable to get a connection".

To determine the version of the workbook, open it in Excel and check which data provider is specified in the connection string. A SQL Server 2008 R2 workbook uses MSOLAP.4 as its data provider.

To work around this issue, you can upgrade the workbook. Alternatively, you can install client libraries from the SQL Server 2008 R2 version of Analysis Services on the physical computers running Power Pivot for SharePoint or Excel Services, Install the Analysis Services OLE DB Provider on SharePoint Servers.

Scenario 2b: Excel Services is running on an application server that has the wrong version of the client libraries

By default, SharePoint Server 2010 installs the SQL Server 2008 version of the Analysis Services OLE DB provider on application servers that run Excel Services. In a farm that supports Power Pivot data access, all physical servers running applications that request Power Pivot data, such as Excel Services and Power Pivot for SharePoint, must use a later version of the data provider.

Servers that run Power Pivot for SharePoint get the updated OLE DB data provider automatically. Other servers, such as those running a standalone instance Excel Services without Power Pivot for SharePoint on the same computer, must be patched to use the newer client libraries. For more information, see Install the Analysis Services OLE DB Provider on SharePoint Servers.

Scenario 3: Domain controller is unavailable

The cause might be that a domain controller is not available to validate the user identity. A domain controller is required by the Claims to Windows Token Service to authenticate the SharePoint user on each connection. The Claims to Windows Token Service does not use cached credentials. It validates the user identity for each connection.

You can confirm the cause of this error by viewing the SharePoint log file. If the SharePoint logs include the message "Failed to get WindowsIdentity from IClaimsIdentity", the user identity could not be authenticated.

To work around this problem, join the computer to the same domain as the Power Pivot server, or install a domain controller on your local computer. The second solution, installing the domain controller, will require you to create local domain accounts for all services and users. You will need to configure service accounts and SharePoint permissions for the accounts you define.

Installing a domain controller on your computer is useful if your objective is to use Power Pivot for SharePoint in an offline state. For detailed instructions on how to use Power Pivot offline, see the blog entry for "Taking your Power Pivot server off the network" on http://www.powerpivotgeek.com.

Scenario 4: Unstable server

One or more services might be in an inconsistent state. In some cases, running IISRESET will resolve the problem.

SQL Server Analysis Services server management

11/15/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

For Azure Analysis Services, see Manage Azure Analysis Services.

A server instance of Analysis Services is a copy of the **msmdsrv.exe** executable that runs as an operating system service. Each instance is fully independent of other instances on the same server, having its own configuration settings, permissions, ports, startup accounts, file storage, and server mode properties.

Each instance runs as Windows service, Msmdsrv.exe, in the security context of a defined logon account.

- The service name of default instance is MSSQLServerOLAPService.
- The service name of each named instance of is MSOLAP$InstanceName.

> **NOTE**
> If multiple instances are installed, Setup also installs a redirector service, which is integrated with the SQL Server Browser service. The redirector service is responsible for directing clients to the appropriate named instance of Analysis Services. The SQL Server Browser service always runs in the security context of the Local Service account, a limited user account used by Windows for services that do not access resources outside the local computer.

Multi-instance means that you can scale-up by installing multiple server instances on the same hardware. For Analysis Services in particular, it also means that you can support different server modes by having multiple instances on the same server, each one configured to run in a specific mode.

Server mode is a server property that determines which storage and memory architecture is used for that instance. A server that runs in Multidimensional mode uses the resource management layer that was built for multidimensional cube databases and data mining models. In contrast, Tabular server mode uses the VertiPaq in-memory analytics engine and data compression to aggregate data as it is requested.

Differences in storage and memory architecture mean that a single instance of Analysis Services will run either tabular databases or multidimensional databases, but not both. The server mode property determines which type of database runs on the instance.

Server mode is set during installation when you specify the type of database that will run on the server. To support all available modes, you can install multiple instances of Analysis Services, each deployed in a server mode that corresponds to the projects you are building.

As a general rule, most of the administrative tasks you must perform do not vary by mode. As an Analysis Services system administrator, you can use the same procedures and scripts to manage any Analysis Services instance on your network regardless of how it was installed.

> **NOTE**
> The exception is Power Pivot for SharePoint. Server administration of a Power Pivot deployment is always within the context of a SharePoint farm. Power Pivot differs from other server modes in that it is always single-instance, and always managed through SharePoint Central Administration or the Power Pivot Configuration Tool. Although it is possible to connect to Power Pivot for SharePoint in SQL Server Management Studio or SQL Server Data Tools (SSDT), it is not desirable. A SharePoint farm includes infrastructure that synchronizes server state and oversees server availability. Using other tools can interfere with these operations. For more information about Power Pivot server administration, see Power Pivot for SharePoint .

Common server management topics

LINK	TASK DESCRIPTION
Post-install Configuration	Describes the required and optional tasks that complete or modify an installation of Analysis.
Connect to Analysis Services	Describes connection string properties, client libraries, authentication methodologies, and steps for establishing or clearing connections.
Monitor an Analysis Services Instance	Describes tools and techniques for monitoring a server instance, including how to use Performance Monitor and SQL Server Profiler.
High availability and Scalability	Describes the most commonly used techniques for making Analysis Services databases high available and scalable.
Globalization scenarios for Analysis Services	Explains language and collation support, steps for changing both properties, and tips for setting and testing language and collation behaviors.
Log operations in Analysis Services	Describes the logs and explains how to configure them.

See also

Comparing Tabular and Multidimensional Solutions
Determine the Server Mode of an Analysis Services Instance

Server Properties in Analysis Services

12/26/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

An Analysis Services administrator can modify default server configuration properties of an Analysis Services instance. Each instance has its own configuration properties, which are set independently of other instances on the same server.

To configure the server, use SQL Server Management Studio or edit the msmdsrv.ini file of a specific SQL Server Analysis Services instance.

Property pages in SQL Server Management Studio show a subset of the properties most likely to be modified. The full list of properties is found in the msmdsrv.ini file.

> **NOTE**
>
> In a default SQL Server Analysis Services installation, msmdsrv.ini can be found in the \Program Files\Microsoft SQL Server\MSAS13.MSSQLSERVER\OLAP\Config folder.
>
> Other properties affecting server configuration include deployment configuration properties in SQL Server Data Tools (SSDT). For more information about those properties, see Specifying Configuration Settings for Solution Deployment.

Configure properties in Management Studio

1. In SQL Server Management Studio, connect to an Analysis Services instance.
2. In Object Explorer, right-click the Analysis Services instance, and then click **Properties**. The General page appears, displaying the more commonly used properties.
3. To view additional properties, click the **Show Advanced (All) Properties** checkbox at the bottom of the page.

   Modifying server properties is supported only for tabular mode and multidimensional mode servers. If you installed Power Pivot for SharePoint, always use the default values unless otherwise directed otherwise by Microsoft Support.

Configure properties in msmdsrv.ini

Some properties can only be set in the msmdrsrv.ini file. These properties do not apply to Azure Analysis Services. If the property you want to set is not visible even after you show advanced properties, you might need to edit the msmdsrv.ini file directly.

1. Check the **DataDir** property in the General property page in Management Studio to verify the location of the Analysis Services program files, including the msmdsrv.ini file.

   On a server that has multiple instances, checking the program file location ensures you're modifying the correct file.
2. Navigate to the **config** folder of the program files folder location.
3. Create a backup of the file in case you need to revert to the original file.
4. Use a text editor to view or edit the msmdsrv.ini file.

5. Save the file and restart the service.

Server Property Reference

The following topics describe the various Analysis Services configuration properties:

TOPIC	DESCRIPTION
General Properties	The general properties are both basic and advanced properties, and include properties that define the data directory, backup directory, and other server behaviors.
Data Mining Properties	The data mining properties control which data mining algorithms are enabled and which are disabled. By default, all of the algorithms are enabled.
DAX Properties	Defines properties related to DAX queries.
DSO	DSO is no longer supported. DSO properties are ignored.
Feature Properties	The feature properties pertain to product features, most of them advanced, including properties that control links between server instances.
Filestore Properties	The file store properties are for advanced use only. They include advanced memory management settings.
Lock Manager Properties	The lock manager properties define server behaviors pertaining to locking and timeouts. Most of these properties are for advanced use only.
Log Properties	The log properties controls if, where, and how events are logged on the server. This includes error logging, exception logging, flight recorder, query logging, and traces.
Memory Properties	The memory properties control how the server uses memory. They are primarily for advanced use.
Network Properties	The network properties control server behavior pertaining to networking, including properties that control compression and binary XML. Most of these properties are for advanced use only.
OLAP Properties	The OLAP properties control cube and dimension processing, lazy processing, data caching, and query behavior. These include both basic and advanced properties.
Security Properties	The security section contains both basic and advanced properties that define access permissions. This includes settings pertaining to administrators and users.
Thread Pool Properties	The thread pool properties control how many threads the server creates. These are primarily advanced properties.

See Also

Analysis Services Instance Management
Specifying Configuration Settings for Solution Deployment

Data Mining Properties

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services supports the data mining server properties listed in the following tables. For more information about additional server properties and how to set them, see Server Properties in Analysis Services.

Applies to: Multidimensional server mode only

Non-Specific Category

AllowSessionMiningModels
A Boolean property that indicates whether session mining models can be created.

The default value for this property is false, which indicates that session mining models cannot be created.

AllowAdHocOpenRowsetQueries
A Boolean property that indicates whether adhoc open rowset queries are allowed.

The default value for this property is false, which indicates that open rowset queries are not allowed during a session.

AllowedProvidersInOpenRowset
A string property that identifies which providers are allowed in an open rowset, consisting of a comma/semi-colon separated list of provider ProgIDs, or else [All].

MaxConcurrentPredictionQueries
A signed 32-bit integer property that defines the maximum number of concurrent prediction queries.

Algorithms Category

Microsoft_Association_Rules\ Enabled
A Boolean property that indicates whether the Microsoft_Association_Rules algorithm is enabled.

Microsoft_Clustering\ Enabled
A Boolean property that indicates whether the Microsoft_Clustering algorithm is enabled.

Microsoft_Decision_Trees\ Enabled
A Boolean property that indicates whether the Microsoft_DecisionTrees algorithm is enabled.

Microsoft_Naive_Bayes\ Enabled
A Boolean property that indicates whether the Microsoft_ Naive_Bayes algorithm is enabled.

Microsoft_Neural_Network\ Enabled
A Boolean property that indicates whether the Microsoft_Neural_Network algorithm is enabled.

Microsoft_Sequence_Clustering\ Enabled
A Boolean property that indicates whether the Microsoft_Sequence_Clustering algorithm is enabled.

Microsoft_Time_Series\ Enabled
A Boolean property that indicates whether the Microsoft_Time_Series algorithm is enabled.

Microsoft_Linear_Regression\ Enabled
A Boolean property that indicates whether the Microsoft_Linear_Regression algorithm is enabled.

Microsoft_Logistic_Regression\ Enabled

A Boolean property that indicates whether the Microsoft_Logistic_Regression algorithm is enabled.

> **NOTE**
>
> In addition to properties that define the data mining services available on the server, there are data mining properties that define the behavior of specific algorithms. You configure these properties when you create an individual data mining model, not at the server level. For more information, see Data Mining Algorithms (Analysis Services - Data Mining).

See Also

Physical Architecture (Analysis Services - Data Mining)
Server Properties in Analysis Services
Determine the Server Mode of an Analysis Services Instance

DAX Properties

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The DAX section of msmdsrv.ini contains settings used to control certain query behaviors in Analysis Services, such as the upper limit on the number of rows returned in a DAX query result set.

For very large rowsets, such as those returned in DirectQuery models, the default of one million rows could be insufficient. You'll know whether the limit needs adjusting if you get this error: "The result set of a query to external data source has exceeded the maximum allowed size of '1000000' rows."

To increase the upper limit, specify the **MaxIntermediateRowSize** configuration setting. You will need to manually add the entire element to the DAX section of the configuration file. The setting is not present in the file until you add it.

Configuration snippet

```
<ConfigurationSettings>
. . .
<DAX>
  <PredicateCheckSpoolCardinalityThreshold>5000
  </PredicateCheckSpoolCardinalityThreshold>
  <DQ>
     <MaxIntermediateRowsetSize>1000000
     </MaxIntermediateRowsetSize>
  </DQ>
</DAX>
. . .
```

Property descriptions

SETTING	VALUE	DESCRIPTION
MaxIntermediateRowsetSize	1000000	Maximum number of rows returned in a DAX query. Manually add this entry to the msmdsrv.ini file and increase the value if the default is too low.
PredicateCheckSpoolCardinalityThreshold	5000	An advanced property that you should not change, except under the guidance of Microsoft support.

For more information about additional server properties and how to set them, see Server Properties in Analysis Services.

Feature Properties

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Feature properties pertain to product features, most of them advanced, including properties that control links between server instances.

Analysis Services supports the server properties listed in the following table. For more information about additional server properties and how to set them, see Server Properties in Analysis Services.

Applies to: Multidimensional server mode only

Properties

PROPERTY	DEFAULT	DESCRIPTION
ManagedCodeEnabled	1	A Boolean property that indicates whether CLR storage procedures are enabled.
LinkInsideInstanceEnabled	1	A Boolean property that indicates whether a linked object can be created inside the same server instance.
LinkToOtherInstanceEnabled	0	A Boolean property that indicates whether objects on remote servers can be linked to.
LinkFromOtherInstanceEnabled	0	A Boolean property that indicates whether objects can be linked to from other server instances.
ConnStringEncryptionEnabled	1	A Boolean property that indicates whether the connection string is encrypted in the server configuration file.
UseCachedPageAllocators	0	A Boolean property that indicates whether cached page allocators are enabled.
ComUdfEnabled	0	A Boolean property that indicates whether user-defined functions defined as COM objects are enabled.
SQMSupportEnabled	1	A Boolean property that indicates whether error and feature usage reports are sent to Microsoft automatically.

PROPERTY	DEFAULT	DESCRIPTION
ResourceMonitoringEnabled	1	A Boolean property that indicates whether internal resource monitoring counters are enabled. This property is on by default. When enabled, this property allows counters to collect usage data about CPU, memory, and I/O activity. Internal resource monitoring counters are used by Dynamic Management Views (DMV) that report on resource utilization. If you disable this property, the DMV queries will still run, but the result set will be invalid. DMVs that depend on this property include the following: **DISCOVER_OBJECT_ACTIVITY** **DISCOVER_COMMAND_OBJECTS** **DISCOVER_SESSIONS** (for SESSION_READS, SESSION_WRITES, SESSION_CPU_TIME_MS) Note: On a multi-core system that uses NUMA architecture, disabling this property can improve query performance, particularly for high multi-user workloads. You will need to run comparison tests to determine whether query performance is improved as the result of changing this property. For best practices on running comparison tests, including clearing the cache and avoiding common mistakes, see the SQL Server 2008 R2 Analysis Services Operations Guide.

See Also

Server Properties in Analysis Services
Determine the Server Mode of an Analysis Services Instance
Use Dynamic Management Views (DMVs) to Monitor Analysis Services

Filestore Properties

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services supports the `filestore` server properties listed in the following tables. These are all advanced properties that you should not change, except under the guidance of Microsoft support. For more information about additional server properties and how to set them, see Server Properties in Analysis Services.

Applies to: Multidimensional and Tabular server mode

Properties

MemoryLimit
An advanced property that you should not change, except under the guidance of Microsoft support.

MemoryLimitMin
An advanced property that you should not change, except under the guidance of Microsoft support.

PercentScanPerPrice
An advanced property that you should not change, except under the guidance of Microsoft support.

PerformanceTrace
An advanced property that you should not change, except under the guidance of Microsoft support.

RandomFileAccessMode
A Boolean property that indicates whether database files and cached files are accessed in random file access mode. This property is off by default. By default, the server does not set the random file access flag when opening partition data files for read access.

On high-end systems, particularly those with large memory resources and multiple NUMA nodes, it can be advantageous to use random file access. In random access mode, Windows bypasses page-mapping operations that read data from disk into the system file cache, thereby lowering contention on the cache.

You will need to run comparison tests to determine whether query performance is improved as the result of changing this property. For best practices on running comparison tests, including clearing the cache and avoiding common mistakes, see the SQL Server 2008 R2 Analysis Services Operations Guide. For additional information on the tradeoffs of using this property, see http://support.microsoft.com/kb/2549369.

To view or modify this property in Management Studio, enable the advanced properties list in the server properties page. You can also change the property in the msmdsrv.ini file. Restarting the server is recommended after setting this property; otherwise files that are already open will continue to be accessed in the previous mode.

UnbufferedThreshold
An advanced property that you should not change, except under the guidance of Microsoft support.

Memory Model Category

MemoryModel\Tax
An advanced property that you should not change, except under the guidance of Microsoft support.

MemoryModel\Income
An advanced property that you should not change, except under the guidance of Microsoft support.

MemoryModel\MaximumBalance

An advanced property that you should not change, except under the guidance of Microsoft support.

MemoryModel\MinimumBalance

An advanced property that you should not change, except under the guidance of Microsoft support.

MemoryModel\InitialBonus

An advanced property that you should not change, except under the guidance of Microsoft support.

See Also

Server Properties in Analysis Services

Determine the Server Mode of an Analysis Services Instance

General Properties

12/7/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services supports the server properties listed in the following tables. This topic documents those server properties in the msmdsrv.ini file that are not otherwise included in a specific section, such as Security, Network, or ThreadPool. For more information about additional server properties and how to set them, see Server Properties in Analysis Services.

Applies to: Multidimensional and Tabular server mode, unless noted otherwise

Non-Specific Category

AdminTimeout
A signed 32-bit integer property that defines the administrator timeout in seconds. This is an advanced property that you should not change, except under the guidance of Microsoft support.

The default value for this property is zero (0), which indicates there is no timeout.

AllowedBrowsingFolders
A string property that specifies in a delimited list the folders that can be browsed when saving, opening, and finding files in Analysis Services dialog boxes. The Analysis Services service account must have read and write permissions to any folders that you add to the list.

BackupDir
A string property that identifies the name of the directory where backup files are stored by default, in the event a path is not specified as part of the Backup command.

CollationName
A string property that identifies the server collation. For more information, see Languages and Collations (Analysis Services).

CommitTimeout
An integer property that specifies how long (in milliseconds) the server will wait to acquire a write lock for the purpose of committing a transaction. A waiting period is often required because the server must wait for other locks to be released before it can take a write lock that commits the transaction.

The default value for this property is zero (0), which indicates that the server will wait indefinitely. For more information about lock-related properties, see SQL Server 2008 R2 Analysis Services Operations Guide.

CoordinatorBuildMaxThreads
A signed 32-bit integer property that defines the maximum number of threads allocated to building partition indexes. Increase this value in order to speed-up partition indexing, at the cost of memory usage. For more information about this property, see SQL Server 2008 R2 Analysis Services Operations Guide.

CoordinatorCancelCount
A signed 32-bit integer property that defines how frequently the server should check whether a Cancel event occurred (based on internal iteration count). Decrease this number in order to check for Cancel more frequently, at the expense of general performance. This property is ignored in tabular server mode.

CoordinatorExecutionMode
A signed 32-bit integer property that defines the maximum number of parallel operations the server will attempt,

including processing and querying operations. Zero (0) indicates that the server will decide, based on an internal algorithm. A positive number indicates the maximum number of operations in total. A negative number, with the sign reversed, indicates the maximum number of operations per processor.

The default value for this property is -4, which indicates the server is limited to 4 parallel operations per processor. For more information about this property, see SQL Server 2008 R2 Analysis Services Operations Guide.

CoordinatorQueryMaxThreads
A signed 32-bit integer property that defines the maximum number of threads per partition segment during query resolution. The fewer the number of concurrent users, the higher this value can be, at the cost of memory. Conversely, it may need to be lowered if there are a large number of concurrent users.

CoordinatorShutdownMode
A Boolean property that defines coordinator shutdown mode. This is an advanced property that you should not change, except under the guidance of Microsoft support.

DataDir
A string property that identifies the name of the directory where data is stored.

DeploymentMode
Determines the operational context of an Analysis Services server instance. This property is referred to as 'server mode' in dialog boxes, messages, and documentation. This property is configured by SQL Server Setup based on the server mode you selected when installing Analysis Services. This property should be considered internal only, always using the value specified by Setup.

Valid values for this property include the following:

VALUE	DESCRIPTION
0	This is the default value. It specifies multidimensional mode, used to service multidimensional databases that use MOLAP, HOLAP, and ROLAP storage, as well as data mining models.
1	Specifies Analysis Services instances that were installed as part of a Power Pivot for SharePoint deployment. Do not change the deployment mode property of Analysis Services instance that is part of a Power Pivot for SharePoint installation. Power Pivot data will no longer run on the server if you change the mode.
2	Specifies Tabular mode used for hosting tabular model databases that use in-memory storage or DirectQuery storage.

Each mode is exclusive of the other. A server that is configured for tabular mode cannot run Analysis Services databases that contain cubes and dimensions. If the underlying computer hardware can support it, you can install multiple instances of Analysis Services on the same computer and configure each instance to use a different deployment mode. Remember that Analysis Services is a resource intensive application. Deploying multiple instances on the same system is recommended only for high-end servers.

EnableFast1033Locale
An advanced property that you should not change, except under the guidance of Microsoft support.

ExternalCommandTimeout
An integer property that defines the timeout, in seconds, for commands issued to external servers, including relational data sources and external Analysis Services servers.

The default value for this property is 3600 (seconds).

ExternalConnectionTimeout
An integer property that defines the timeout, in seconds, for creating connections to external servers, including relational data sources and external Analysis Services servers. This property is ignored if a connection timeout is specified on the connection string.

The default value for this property is 60 (seconds).

ForceCommitTimeout
An integer property that specifies how long, in milliseconds, a write commit operation should wait before canceling other commands that preceded the current command, including queries in progress. This allows the commit transaction to proceed by canceling lower priority operations, such as queries.

The default value for this property is 30 seconds (30000 milliseconds), which indicates that other commands will not be forced to timeout until the commit transaction has been waiting for 30 seconds.

> **NOTE**
> Queries and processes cancelled by this event will report the following error message: "`Server: The operation has been cancelled`"

For more information about this property, see SQL Server 2008 R2 Analysis Services Operations Guide.

> **IMPORTANT**
> **ForceCommitTimeout** applies to cube processing commands and to writeback operations.

IdleConnectionTimeout
An integer property that specifies a timeout, in seconds, for connections that are inactive.

The default value for this property is zero (0), which indicates that idle connections will not be timed out.

IdleOrphanSessionTimeout
An integer property that defines how long, in seconds, orphaned sessions will be retained in server memory. An orphaned session is one that no longer has an associated connection. The default is 120 seconds.

InstanceVisible
A Boolean property that indicates whether the server instance is visible to discover instance requests from SQL Server Browser service. The default is True. If you set it to false, the instance is not visible to SQL Server Browser.

Language
A string property that defines the language, including error messages and number formatting. This property overrides the CollationName property.

The default value for this property is blank, which indicates that the CollationName property defines the language.

LogDir
A string property that identifies the name of the directory that contains server logs. This property only applies when disk files are used for logging, as opposed to database tables (the default behavior).

MaxIdleSessionTimeout
An integer property that defines the maximum idle session timeout, in seconds. The default is zero (0), indicating that sessions are never timed out. However, idle sessions will still be removed if the server is under resource constraints.

MinIdleSessionTimeout
An integer property that defines the minimum time, in seconds, that idle sessions will timeout. The default is 2700

seconds. After this time expires, the server is permitted to end the idle session, but will only do so as memory is needed.

Port
An integer property that defines the port number on which server will listen for client connections. If not set, server dynamically finds first unused port.

The default value for this property is zero (0), which in turn defaults to port 2383. For more information about port configuration, see Configure the Windows Firewall to Allow Analysis Services Access.

ServerTimeout
An integer that defines the timeout, in seconds, for queries. The default is 3600 seconds (or 60 minutes). Zero (0) specifies that no queries will timeout.

TempDir
A string property that specifies the location for storing temporary files used during processing, restoring, and other operations. The default value for this property is determined by setup. If not specified, the default is the Data directory.

RequestPrioritization Category

Enabled
An advanced property that you should not change, except under the guidance of Microsoft support.

StatisticsStoreSize
An advanced property that you should not change, except under the guidance of Microsoft support.

See Also

Server Properties in Analysis Services
Determine the Server Mode of an Analysis Services Instance

Lock Manager Properties

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services supports the lock manager server properties listed in the following table. For more information about additional server properties and how to set them, see Server Properties in Analysis Services.

Applies to: Multidimensional and Tabular server mode

Properties

DefaultLockTimeoutMS
A signed 32-bit integer property that defines default lock timeout in milliseconds for internal lock requests.

The default value for this property is -1, which indicates there is no timeout for internal lock requests.

LockWaitGranularityMS
An advanced property that you should not change, except under the guidance of Microsoft support.

DeadlockDetectionGranularityMS
An advanced property that you should not change, except under the guidance of Microsoft support.

See Also

Server Properties in Analysis Services
Determine the Server Mode of an Analysis Services Instance

Log Properties

12/7/2018 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services supports the log server properties listed in the following tables. For more information about additional server properties and how to set them, see Server Properties in Analysis Services.

> **NOTE**
> Not all properties apply to Azure Analysis Services.

General

File
A string property that identifies the name of the server log file. This property only applies when a disk file is used for logging, as opposed to a database table (the default behavior).

The default value for this property is msmdsrv.log.

FileBufferSize
An advanced property that you should not change, except under the guidance of Microsoft support.

MessageLogs
An advanced property that you should not change, except under the guidance of Microsoft support.

Error Log

You can set these properties at the server instance level to modify the default values for Error Configuration that appear in other tools and designers. See Error Configuration for Cube, Partition, and Dimension Processing (SSAS - Multidimensional) and ErrorConfiguration for more information.

ErrorLog\ErrorLogFileName
A property used as a default during processing operation performed by the server.

ErrorLog\ErrorLogFileSize
A property used as a default during processing operation performed by the server.

ErrorLog\KeyErrorAction
Specifies the action taken by the server when a **KeyNotFound** error occurs. Valid responses to this error include:

- **ConvertToUnknown** tells the server to allocate the error key value to the unknown member.
- **DiscardRecord** tells the server to exclude the record.

ErrorLog\KeyErrorLogFile
This is a user-defined filename that must have a .log file extension, located in a folder on which the service account has read-write permissions. This log file will only contain errors generated during processing. Use the Flight Recorder if you require more detailed information.

ErrorLog\KeyErrorLimit
This is the maximum number of data integrity errors that the server will allow before failing the processing. A value of -1 indicates that there is no limit. The default is 0, which means processing stops after the first error

occurs. You can also set it to a whole number.

ErrorLog\KeyErrorLimitAction
Specifies the action taken by the server when the number of key errors has reached the upper limit. Valid responses to this action include:

- **StopProcessing** tells the server to stop processing when the error limit is reached.
- **StopLogging** tells the server to stop logging errors when the error limit is reached, but allow processing to continue.

ErrorLog\ LogErrorTypes\KeyNotFound
Specifies the action taken by the server when a **KeyNotFound** error occurs. Valid responses to this error include:

- **IgnoreError** tells the server to continue processing without logging the error or counting it towards the key error limit. By ignoring the error, you simply allow processing to continue without adding to the error count or logging it to the screen or log file. The record in question has a data integrity problem and cannot be added to the database. The record will either be discarded or aggregated to Unknown Member, as determined by the **KeyErrorAction** property.
- **ReportAndContinue** tells the server to log the error, count it towards the key error limit, and continue processing. The record triggering the error is discarded or converted to Unknown Member.
- **ReportAndStop** tells the server to log the error and stop processing immediately, regardless of the key error limit. The record triggering the error is discarded or converted to Unknown Member.

ErrorLog\ LogErrorTypes\KeyDuplicate
Specifies the action taken by the server when a duplicate key is found. Valid values include **IgnoreError** to continue processing as if the error did not occur, **ReportAndContinue** to log the error and continue processing, and **ReportAndStop** to log the error and stop processing immediately, even if the error count is below the error limit.

ErrorLog\ LogErrorTypes\NullKeyConvertedToUnknown
Specifies the action taken by the server when a null key has been converted to Unknown Member. Valid values include **IgnoreError** to continue processing as if the error did not occur, **ReportAndContinue** to log the error and continue processing, and **ReportAndStop** to log the error and stop processing immediately, even if the error count is below the error limit.

ErrorLog\ LogErrorTypes\NullKeyNotAllowed
Specifies the action taken by the server when **NullProcessing** is set to **Error** for a dimension attribute. An error is generated when a null value is not allowed in a given attribute. This error configuration property informs the next step, which is to report the error and continue processing until the error limit is reached. Valid values include **IgnoreError** to continue processing as if the error did not occur, **ReportAndContinue** to log the error and continue processing, and **ReportAndStop** to log the error and stop processing immediately, even if the error count is below the error limit.

ErrorLog\ LogErrorTypes\CalculationError
A property used as a default during processing operation performed by the server.

ErrorLog\IgnoreDataTruncation
A property used as a default during processing operation performed by the server.

Exception

Exception\CreateAndSendCrashReports
An advanced property that you should not change, except under the guidance of Microsoft support.

Exception\CrashReportsFolder

An advanced property that you should not change, except under the guidance of Microsoft support.

Exception\SQLDumperFlagsOn
An advanced property that you should not change, except under the guidance of Microsoft support.

Exception\SQLDumperFlagsOff
An advanced property that you should not change, except under the guidance of Microsoft support.

Exception\MiniDumpFlagsOn
An advanced property that you should not change, except under the guidance of Microsoft support.

Exception\MinidumpErrorList
An advanced property that you should not change, except under the guidance of Microsoft support.

Flight Recorder

FlightRecorder\Enabled
A Boolean property that indicates whether the flight recorder feature is enabled.

FlightRecorder\FileSizeMB
A signed 32-bit integer property that defines the size of the flight recorder disk file, in megabytes.

FlightRecorder\LogDurationSec
A signed 32-bit integer property that defines the frequency that the flight recorder is rolled over, in seconds.

FlightRecorder\SnapshotDefinitionFile
A string property that defines the name of the snapshot definition file, containing discover commands that are issued to the server when a snapshot is taken.

The default value for this property is blank, which in turn defaults to file name FlightRecorderSnapshotDef.xml.

FlightRecorder\SnapshotFrequencySec
A signed 32-bit integer property that defines the snapshot frequency, in seconds.

FlightRecorder\TraceDefinitionFile
A string property that specifies the name of the flight recorder trace definition file.

The default value for this property is blank, which in turn defaults to FlightRecorderTraceDef.xml.

Query Log

Applies to: Multidimensional server mode only

QueryLog\QueryLogFileName
A string property that specifies the name of the query log file. This property only applies when a disk file is used for logging, as opposed to a database table (the default behavior).

QueryLog\QueryLogSampling
A signed 32-bit integer property that specifies the query log sampling rate.

The default value for this property is 10, meaning that 1 out of every 10 server queries is logged.

QueryLog\QueryLogFileSize
An advanced property that you should not change, except under the guidance of Microsoft support.

QueryLog\QueryLogConnectionString
A string property that specifies the connection to the query log database.

QueryLog\QueryLogTableName
A string property that specifies the name of the query log table.

The default value for this property is OlapQueryLog.

QueryLog\CreateQueryLogTable
A Boolean property that specifies whether to create the query log table.

The default value for this property is false, which indicates the server will not automatically create the log table and will not log query events.

Trace

Trace\TraceBackgroundDistributionPeriod
An advanced property that you should not change, except under the guidance of Microsoft support.

Trace\TraceBackgroundFlushPeriod
An advanced property that you should not change, except under the guidance of Microsoft support.

Trace\TraceFileBufferSize
An advanced property that you should not change, except under the guidance of Microsoft support.

Trace\TraceFileWriteTrailerPeriod
An advanced property that you should not change, except under the guidance of Microsoft support.

Trace\TraceMaxRowsetSize
An advanced property that you should not change, except under the guidance of Microsoft support.

Trace\TraceProtocolTraffic
An advanced property that you should not change, except under the guidance of Microsoft support.

Trace\TraceQueryResponseTextChunkSize
An advanced property that you should not change, except under the guidance of Microsoft support.

Trace\TraceReportFQDN
An advanced property that you should not change, except under the guidance of Microsoft support.

Trace\TraceRequestParameters
An advanced property that you should not change, except under the guidance of Microsoft support.

Trace\TraceRowsetBackgroundFlushPeriod
An advanced property that you should not change, except under the guidance of Microsoft support.

See Also

Server Properties in Analysis Services
Determine the Server Mode of an Analysis Services Instance

Memory Properties

1/15/2019 • 6 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services pre-allocates a modest amount of memory at startup so requests can be handled immediately. Additional memory is allocated as query and processing workloads increase.

By specifying configuration settings, you can control the thresholds at which memory is released. For example, the **HardMemoryLimit** setting specifies a self-imposed out-of-memory condition (by default, this threshold is not enabled), where new requests are rejected outright until more resources become available.

To learn more about maximum memory utilized per SQL Server Analysis Services instance by edition, see Editions and supported features of SQL Server.

The following settings apply to both tabular and multidimensional servers unless noted otherwise.

Default memory configuration

Under the default configuration, each Analysis Services instance allocates a small amount of RAM (40 MB to 50 MB) at startup, even if the instance is idle.

Recall that configuration settings are per instance. If you are running multiple instances of Analysis Services, such as a tabular and multidimensional instance on the same hardware, each instance will allocate its own memory independently of other instances.

The table below briefly describes the more commonly used memory settings (with more detail in the reference section). Configure these settings only if Analysis Services is competing for memory with other applications on the same server:

SETTING	DESCRIPTION
LowMemoryLimit	For multidimensional instances, a lower threshold at which the server first begins releasing memory allocated to infrequently used objects.
VertiPaqMemoryLimit	For tabular instances, a lower threshold at which the server first begins releasing memory allocated to infrequently used objects.
TotalMemoryLimit	An upper threshold at which Analysis Services begins releasing memory more aggressively to make room for requests that are in execution as well as new high priority requests.
HardMemoryLimit	Another threshold at which Analysis Services begins rejecting requests outright due to memory pressure.

Property reference

The following properties apply to both tabular and multidimensional modes unless specified otherwise.

Values between 1 and 100 represent percentages of **Total Physical Memory** or **Virtual Address Space**,

whichever is less. Values over 100 represent memory limits in bytes.

LowMemoryLimit
A signed 64-bit double-precision floating-point number property that defines the first threshold at which Analysis Services begins releasing memory for low-priority objects, such as an infrequently used cache. Once the memory is allocated, the server does not release memory below this limit. The default value is 65; which indicates the low memory limit is 65% of physical memory or the virtual address space, whichever is less.

TotalMemoryLimit
Defines a threshold that when reached, causes the server to deallocate memory to make room for other requests. When this limit is reached, the instance will start to slowly clear memory out of caches by closing expired sessions and unloading unused calculations. The default value 80% of physical memory or the virtual address space, whichever is less. **TotalMemoryLimit** must always be less than **HardMemoryLimit**

HardMemoryLimit
Specifies a memory threshold after which the instance aggressively terminates active user sessions to reduce memory usage. All terminated sessions will receive an error about being canceled by memory pressure. The default value, zero (0), means the **HardMemoryLimit** will be set to a midway value between **TotalMemoryLimit** and the total physical memory of the system; if the physical memory of the system is larger than the virtual address space of the process, then virtual address space will be used instead to calculate **HardMemoryLimit**.

QueryMemoryLimit
Azure Analysis Services only. An advanced property to control how much memory can be used by temporary results during a query. Applies only to DAX measures and queries. It does not account for general memory allocations used by the query. Specified in percentage, the default value is determined by your plan.

PLAN	DEFAULT
D1	80
All other plans	20

This property can be changed. Setting a value of 0 means no limit is specified.

VirtualMemoryLimit
An advanced property that you should not change, except under the guidance of Microsoft support.

VertiPaqPagingPolicy
For tabular instances only, specifies the paging behavior in the event the server runs low on memory. Valid values are as follows:

SETTING	DESCRIPTION
0	(default for Azure Analysis Services) Disables paging. If memory is insufficient, processing fails with an out-of-memory error. If you disable paging, you must grant Windows privileges to the service account. See Configure Service Accounts (Analysis Services) for instructions.
1	(default for SQL Server Analysis Services) This property enables paging to disk using the operating system page file (pagefile.sys).

When set to 1, processing is less likely to fail due to memory constraints because the server will try to page to disk using the method that you specified. Setting the **VertiPaqPagingPolicy** property does not guarantee that memory errors will never happen. Out of memory errors can still occur under the following conditions:

- There is not enough memory for all dictionaries. During processing, the server locks the dictionaries for each column in memory, and all of these together cannot be more than the value specified for **VertiPaqMemoryLimit**.
- There is insufficient virtual address space to accommodate the process.

To resolve persistent out of memory errors, you can either try to redesign the model to reduce the amount of data that needs processing, or you can add more physical memory to the computer.

VertiPaqMemoryLimit
For tabular instances only, if paging to disk is allowed, this property specifies the level of memory consumption (as a percentage of total memory) at which paging starts. The default is 60. If memory consumption is less than 60 percent, the server will not page to disk. This property depends on the **VertiPaqPagingPolicyProperty**, which must be set to 1 in order for paging to occur.

HighMemoryPrice
An advanced property that you should not change, except under the guidance of Microsoft support.

MemoryHeapType
An advanced property that you should not change, except under the guidance of Microsoft support. Valid values in SQL Server 2016 SP1 and later Analysis Services are as follows:

SETTING	DESCRIPTION
-1	(default) Automatic. The engine will decide which one to use.
1	Analysis Services HEAP.
2	Windows LFH.
5	Hybrid allocator. This allocator will use Windows LFH for <= 16 KB allocations and the AS Heap for >16 KB allocations.
6	Intel TBB allocator. Available in SQL Server 2016 SP1 (and later) Analysis Services.

HeapTypeForObjects
An advanced property that you should not change, except under the guidance of Microsoft support. Valid values are as follows:

SETTING	DESCRIPTION
-1	(default) Automatic. The engine will decide which one to use.
0	Windows LFH heap.
1	Analysis Services slot allocator.
3	Each object has its own Analysis Services Heap.

DefaultPagesCountToReuse
An advanced property that you should not change, except under the guidance of Microsoft support.

HandleIA64AlignmentFaults
An advanced property that you should not change, except under the guidance of Microsoft support.

MidMemoryPrice

An advanced property that you should not change, except under the guidance of Microsoft support.

MinimumAllocatedMemory

An advanced property that you should not change, except under the guidance of Microsoft support.

PreAllocate

An advanced property that you should not change, except under the guidance of Microsoft support.

SessionMemoryLimit

An advanced property that you should not change, except under the guidance of Microsoft support.

WaitCountIfHighMemory

An advanced property that you should not change, except under the guidance of Microsoft support.

See Also

Server Properties in Analysis Services

Determine the Server Mode of an Analysis Services Instance

Network Properties

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services supports the server properties listed in the following tables. For more information about additional server properties and how to set them, see Server Properties in Analysis Services.

Applies to: Multidimensional and Tabular server mode

General

ListenOnlyOnLocalConnections

A Boolean property that identifies whether to listen only on local connections, for example localhost.

Listener

IPV4Support

A signed 32-bit integer property that defines support for IPv4 protocol. This property has one of the values listed in the following table:

VALUE	DESCRIPTION
0	IPv4 disabled; clients can't connect.
1	(Default) IPv4 is required; server won't start if it cannot listen to IPv4.
2	IPv4 is optional; server tries to listen to IPv4 but starts even if unable to.

IPV6Support

A signed 32-bit integer property that defines support for IPv6 protocol. This property has one of the values listed in the following table:

VALUE	DESCRIPTION
0	IPv6 disabled; clients can't connect.
1	(Default) IPv6 is required; server won't start if it cannot listen to IPv6.
2	IPv6 is optional; server tries to listen to IPv6 but starts even if unable to.

MaxAllowedRequestSize

An advanced property that you should not change, except under the guidance of Microsoft support.

RequestSizeThreshold

An advanced property that you should not change, except under the guidance of Microsoft support.

ServerReceiveTimeout

An advanced property that you should not change, except under the guidance of Microsoft support.

ServerSendTimeout
An advanced property that you should not change, except under the guidance of Microsoft support.

Requests

EnableBinaryXML
A Boolean property that specifies whether the server will recognize requests binary xml format.

EnableCompression
A Boolean property that specifies whether compression is enabled for requests.

Responses

CompressionLevel
An advanced property that you should not change, except under the guidance of Microsoft support.

EnableBinaryXML
A Boolean property that specifies whether the server is enabled for binary xml responses.

EnableCompression
A Boolean property that specifies whether compression is enabled for responses to client requests.

TCP

InitialConnectTimeout
An advanced property that you should not change, except under the guidance of Microsoft support.

MaxCompletedReceiveCount
An advanced property that you should not change, except under the guidance of Microsoft support.

MaxPendingAcceptExCount
An advanced property that you should not change, except under the guidance of Microsoft support.

MaxPendingReceiveCount
An advanced property that you should not change, except under the guidance of Microsoft support.

MaxPendingSendCount
An advanced property that you should not change, except under the guidance of Microsoft support.

MinPendingAcceptExCount
An advanced property that you should not change, except under the guidance of Microsoft support.

MinPendingReceiveCount
An advanced property that you should not change, except under the guidance of Microsoft support.

ScatterReceiveMultiplier
An advanced property that you should not change, except under the guidance of Microsoft support.

SocketOptions\ DisableNonblockingMode
An advanced property that you should not change, except under the guidance of Microsoft support.

SocketOptions\ EnableLingerOnClose
An advanced property that you should not change, except under the guidance of Microsoft support.

SocketOptions\EnableNagleAlgorithm
An advanced property that you should not change, except under the guidance of Microsoft support.

SocketOptions\ LingerTimeout

An advanced property that you should not change, except under the guidance of Microsoft support.

SocketOptions\ ReceiveBufferSize

An advanced property that you should not change, except under the guidance of Microsoft support.

SocketOptions\ SendBufferSize

An advanced property that you should not change, except under the guidance of Microsoft support.

See Also

Server Properties in Analysis Services

Determine the Server Mode of an Analysis Services Instance

OLAP Properties

12/7/2018 • 12 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services supports the OLAP server properties listed in the following tables. For more information about additional server properties and how to set them, see Server Properties in Analysis Services.

Applies to: Multidimensional server mode only

Memory

DefaultPageSizeForData
An advanced property that you should not change, except under the guidance of Microsoft support.

DefaultPageSizeForDataHeader
An advanced property that you should not change, except under the guidance of Microsoft support.

DefaultPageSizeForIndex
An advanced property that you should not change, except under the guidance of Microsoft support.

DefaultPageSizeForIndexHeader
An advanced property that you should not change, except under the guidance of Microsoft support.

DefaultPageSizeForString
An advanced property that you should not change, except under the guidance of Microsoft support.

DefaultPageSizeForHash
An advanced property that you should not change, except under the guidance of Microsoft support.

DefaultPageSizeForProp
An advanced property that you should not change, except under the guidance of Microsoft support.

LazyProcessing

Enabled
A Boolean property that specifies whether lazy aggregation processing is enabled.

SleepIntervalSecs
A signed 32-bit integer property that defines the interval, in seconds, that the server checks whether there are lazy processing jobs pending.

MaxCPUUsage
A signed 64-bit double-precision floating-point number property that defines maximum CPU usage for lazy processing, expressed as a percentage. The server monitors average CPU use based on snapshots. It is normal behavior for the CPU to spike above this threshold.

The default value for this property is 0.5, indicating a maximum of 50% of the CPU will be devoted to lazy processing.

MaxObjectsInParallel
A signed 32-bit integer property that specifies the maximum number of partitions that can be lazily processed in parallel.

MaxRetries

A signed 32-bit integer property that defines the number of retries in the event that lazy processing fails before an error is raised.

ProcessPlan

CacheRowsetRows
An advanced property that you should not change, except under the guidance of Microsoft support.

CacheRowsetToDisk
An advanced property that you should not change, except under the guidance of Microsoft support.

DistinctBuffer
A signed 32-bit integer property that defines the size of an internal buffer used for distinct counts. Increase this value to speed up distinct count processing at the cost of memory use.

EnableRolapDimQueryTableGrouping
A Boolean property that specifies whether table grouping is enabled for ROLAP dimensions. If True, when querying ROLAP dimensions at runtime, entire ROLAP dimension tables are queried at once, as opposed to separate queries for each attribute.

EnableTableGrouping
A Boolean property that specifies whether table grouping is enabled. If True, when processing dimensions, entire dimension tables are queried at once, as opposed to separate queries for each attribute.

ForceMultiPass
An advanced property that you should not change, except under the guidance of Microsoft support.

MaxTableDepth
An advanced property that you should not change, except under the guidance of Microsoft support.

MemoryAdjustConst
An advanced property that you should not change, except under the guidance of Microsoft support.

MemoryAdjustFactor
An advanced property that you should not change, except under the guidance of Microsoft support.

MemoryLimit
A signed 64-bit double-precision floating-point number property that defines the maximum amount of memory to be devoted to processing, expressed as a percentage of physical memory.

The default value for this property is 65, indicating that 65% of physical memory may be devoted to cube and dimension processing.

MemoryLimitErrorEnabled
An advanced property that you should not change, except under the guidance of Microsoft support.

OptimizeSchema
An advanced property that you should not change, except under the guidance of Microsoft support.

ProactiveCaching

DefaultRefreshInterval
An advanced property that you should not change, except under the guidance of Microsoft support.

DimensionLatencyAccuracy
An advanced property that you should not change, except under the guidance of Microsoft support.

PartitionLatencyAccuracy

An advanced property that you should not change, except under the guidance of Microsoft support.

Process

AggregationMemoryLimitMax
A signed 64-bit double-precision floating-point number property that defines the maximum amount of memory that can be devoted to aggregation processing, expressed as a percentage of physical memory.

The default value for this property is 80, indicating that 80% of physical memory may be devoted to aggregation processing.

AggregationMemoryLimitMin
A signed 64-bit double-precision floating-point number property that defines the minimum amount of memory that can be devoted to aggregation processing, expressed as a percentage of physical memory. A larger value may speed up aggregation processing at the cost of memory usage.

The default value for this property is 10, indicating that minimally 10% of physical memory will be devoted to aggregation processing.

AggregationNewAlgo
An advanced property that you should not change, except under the guidance of Microsoft support.

AggregationPerfLog2
An advanced property that you should not change, except under the guidance of Microsoft support.

AggregationsBuildEnabled
A Boolean property that specifies whether aggregation building is enabled. This is a mechanism to benchmark aggregation building without changing aggregation design.

BufferMemoryLimit
A signed 64-bit double-precision floating-point number property that defines the processing buffer memory limit, expressed as a percent of physical memory.

The default value for this property is 60, which indicates that up to 60% of physical memory can be used for buffer memory.

BufferRecordLimit
A signed 32-bit integer property that defines the number of records that can be buffered during processing.

The default value for this property is 1048576 (records).

CacheRecordLimit
An advanced property that you should not change, except under the guidance of Microsoft support.

CheckDistinctRecordSortOrder
A Boolean property that defines if the sort order for the results of a distinct count query are meaningful when processing partitions. True indicates the sort order is not meaningful and must be "checked" by the server. When processing partitions with distinct count measure, query sent to SQL with order-by. Set to false to speed up processing.

The default value for this property is True, which indicates that the sort order is not meaningful and must be checked.

DatabaseConnectionPoolConnectTimeout
A signed 32-bit integer property that specifies timeout when opening a new connection in seconds.

DatabaseConnectionPoolGeneralTimeout
A signed 32-bit integer property that specifies database connection timeout for use with external OLEDB connections in seconds.

DatabaseConnectionPoolMax
A signed 32-bit integer property that specifies the maximum number of pooled database connections.

The default value for this property is 50 (connections).

DatabaseConnectionPoolTimeout
An advanced property that you should not change, except under the guidance of Microsoft support.

DataFileInitEnabled
An advanced property that you should not change, except under the guidance of Microsoft support.

DataPlacementOptimization
An advanced property that you should not change, except under the guidance of Microsoft support.

DataSliceInitEnabled
An advanced property that you should not change, except under the guidance of Microsoft support.

DeepCompressValue
A Boolean property applying to measures with Double data type that specifies whether numbers can be compressed, causing a loss in numeric precision. A value of False indicates no compression and no precision loss.

The default value for this property is True, which indicates that compression is enabled and precision will be lost.

DimensionPropertyKeyCache
A Boolean property that specifies whether dimension property keys are cached. Must be set to True if keys are non-unique.

IndexBuildEnabled
A Boolean property that specifies whether indexes are built upon processing. This property is for benchmarking and informational purposes.

IndexBuildThreshold
A signed 32-bit integer property that specifies a row count threshold below which indexes will not be built for partitions.

The default value for this property is 4096 (rows).

IndexFileInitEnabled
An advanced property that you should not change, except under the guidance of Microsoft support.

MapFormatMask
An advanced property that you should not change, except under the guidance of Microsoft support.

RecordsReportGranularity
A signed 32-bit integer property that specifies how often the server records Trace events during processing, in rows.

The default value for this property is 1000, which indicates that a Trace event is logged once every 1000 rows.

ROLAPDimensionProcessingEffort
An advanced property that you should not change, except under the guidance of Microsoft support.

Query

AggregationsUseEnabled
A Boolean property that defines whether stored aggregations are used at runtime. This property allows aggregations to be disabled without changing the aggregation design or re-processing, for informational and benchmarking purposes.

The default value for this property is True, indicating that aggregations are enabled.

AllowSEFiltering
An advanced property that you should not change, except under the guidance of Microsoft support.

CalculationCacheRegistryMaxIterations
An advanced property that you should not change, except under the guidance of Microsoft support.

CalculationEvaluationPolicy
An advanced property that you should not change, except under the guidance of Microsoft support.

ConvertDeletedToUnknown
A Boolean property that specifies whether deleted dimension members are converted to Unknown member.

CopyLinkedDataCacheAndRegistry
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCacheRegistryMaxIterations
An advanced property that you should not change, except under the guidance of Microsoft support.

DefaultDrillthroughMaxRows
A signed 32-bit integer property that specifies the maximum number of rows that will return from a drill-through query.

The default value for this property is 10000 (rows).

DimensionPropertyCacheSize
A signed 32-bit integer property that specifies the amount of memory (in bytes) used to cache dimension members used in a query.

The default is 4,000,000 bytes (or 4 MB) per attribute hierarchy, per active query. The default value provides a well-balanced cache size for solutions having typical hierarchies. However, dimensions with a very large number of members (in the millions) or deep hierarchies perform better if you increase this value.

Implications of increasing cache size:

- Memory utilization costs increase when you allow more memory to be used by the dimension cache. Actual usage depends on query execution. Not all queries will use the allowable maximum.

  Note that the memory used by these caches is considered nonshrinkable and will be included when accounting against the **TotalMemoryLimit**.

- Affects all databases on the server. **DimensionPropertyCachesize** is a server-wide property. Changing this property affects all databases running on the current instance.

Approach for estimating dimension cache requirements:

1. Start by increasing the size by a large number to determine whether there is a benefit to increasing the dimension cache size. For example, you might want to double the default value as an initial step.

2. If a performance improvement is evident, incrementally reduce the value until you reach a balance between performance and memory utilization.

ExpressNonEmptyUseEnabled
An advanced property that you should not change, except under the guidance of Microsoft support.

IgnoreNullRolapRows
An advanced property that you should not change, except under the guidance of Microsoft support.

IndexUseEnabled
A Boolean property that defines whether indexes are used at runtime. This property is for informational and benchmarking purposes.

MapHandleAlgorithm
An advanced property that you should not change, except under the guidance of Microsoft support.

MaxRolapOrConditions
An advanced property that you should not change, except under the guidance of Microsoft support.

UseCalculationCacheRegistry
An advanced property that you should not change, except under the guidance of Microsoft support.

UseDataCacheFreeLastPageMemory
An advanced property that you should not change, except under the guidance of Microsoft support.

UseDataCacheRegistry
A Boolean property that specifies whether to enable the data cache registry, where query results are cached (though not calculated results).

UseDataCacheRegistryHashTable
An advanced property that you should not change, except under the guidance of Microsoft support.

UseDataCacheRegistryMultiplyKey
An advanced property that you should not change, except under the guidance of Microsoft support.

UseDataSlice
A Boolean property that defines whether to use partition data slices at runtime for query optimization. This property is for benchmarking and informational purposes.

UseMaterializedIterators
An advanced property that you should not change, except under the guidance of Microsoft support.

UseSinglePassForDimSecurityAutoExist
An advanced property that you should not change, except under the guidance of Microsoft support.

UseVBANet
A Boolean property that defines whether to use the VBA .net assembly for user-defined functions.

CalculationPrefetchLocality\ ApplyIntersect
An advanced property that you should not change, except under the guidance of Microsoft support.

CalculationPrefetchLocality\ ApplySubtract
An advanced property that you should not change, except under the guidance of Microsoft support.

CalculationPrefetchLocality\ PrefetchLowerGranularities
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\ CachedPageAlloc\ Income
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\ CachedPageAlloc\ InitialBonus
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\ CachedPageAlloc\ MaximumBalance
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\ CachedPageAlloc\ MinimumBalance
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\ CachedPageAlloc\ Tax
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\CellStore\ Income

An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\CellStore\ InitialBonus
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\CellStore\ MaximumBalance
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\CellStore\ MinimumBalance
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\CellStore\ Tax
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\ MemoryModel \ Income
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\ MemoryModel \ InitialBonus
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\ MemoryModel \ MaximumBalance
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\ MemoryModel \ MinimumBalance
An advanced property that you should not change, except under the guidance of Microsoft support.

DataCache\ MemoryModel\ Tax
An advanced property that you should not change, except under the guidance of Microsoft support.

Jobs

ProcessAggregation\ MemoryModel\ Income
An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ MemoryModel\ InitialBonus
An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ MemoryModel\ MaximumBalance
An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ MemoryModel\ MinimumBalance
An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ MemoryModel\ Tax
An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ ProcessPartition\ Income
An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ ProcessPartition \ InitialBonus
An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ ProcessPartition \ MaximumBalance
An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ ProcessPartition \ MinimumBalance
An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ ProcessPartition \ Tax
An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ ProcessProperty\ Income

An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ ProcessProperty\ InitialBonus

An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ ProcessProperty\ MaximumBalance

An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ ProcessProperty\ MinimumBalance

An advanced property that you should not change, except under the guidance of Microsoft support.

ProcessAggregation\ ProcessProperty\ Tax

An advanced property that you should not change, except under the guidance of Microsoft support.

See Also

Server Properties in Analysis Services

Determine the Server Mode of an Analysis Services Instance

Security Properties

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services supports the security server properties listed in the following table. For more information about additional server properties and how to set them, see Server Properties in Analysis Services.

Applies to: Multidimensional and Tabular server mode

Properties

RequireClientAuthentication
A Boolean property that indicates whether client authentication is required.

The default value for this property is True, which indicates that client authentication is required.

SecurityPackageList
A string property that contains a comma-separated list of SSPI packages used by server for client authentication.

DisableClientImpersonation
A Boolean property that indicates whether client impersonation (for example, from stored procedures) is disabled.

The default value for this property is False, which indicates that client impersonation is enabled.

BuiltinAdminsAreServerAdmins
A Boolean property that indicates whether members of the local machine administrators group are Analysis Services administrators.

ServiceAccountIsServerAdmin
A Boolean property that indicates whether the service account is a server administrator.

The default value for this property is True, which indicates that the service account is a server administrator.

ErrorMessageMode
An advanced property that you should not change, except under the guidance of Microsoft support.

DataProtection\ RequiredProtectionLevel
A signed 32-bit integer property that defines the required protection level for all client requests. This property has one of the values listed in the following table.

VALUE	DESCRIPTION
0	None, clear text allowed.
1	(Default) Encryption required, no clear-text logging.
2	Clear-text requests allowed but only with signatures (weaker protection than encryption).

AdministrativeDataProtection\ RequiredProtectionLevel
An advanced property that you should not change, except under the guidance of Microsoft support.

See Also

Server Properties in Analysis Services
Determine the Server Mode of an Analysis Services Instance

Thread Pool Properties

1/28/2019 • 29 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services uses multi-threading for many operations, improving overall server performance by running multiple jobs in parallel. To manage threads more efficiently, Analysis Services uses thread pools to preallocate threads and facilitate thread availability for the next job.

Each instance of Analysis Services maintains its own set of thread pools. There are differences in how tabular and multidimensional instances use thread pools. For example, only multidimensional instances use the **IOProcess** thread pool. As such, the **PerNumaNode** property, described in this article, is not meaningful for Tabular instances. In the Property Reference section below, mode requirements are called out for each property.

> **NOTE**
> Tabular deployment on NUMA systems is out of scope for this topic. Although tabular solutions can be successfully deployed on NUMA systems, the performance characteristics of the in-memory database technology used by tabular models may show limited benefits on a highly scaled up architectures. For more information, see Analysis Services Case Study: Using Tabular Models in Large-scale Commercial Solutions and Hardware Sizing a Tabular Solution.

Thread Management in Analysis Services

Analysis Services uses multi-threading to take advantage of the available CPU resources by increasing the number of tasks executing in parallel. The storage engine is multi-threaded. Examples of multi-threaded jobs that execute within the storage engine include processing objects in parallel or handling discrete queries that have been pushed to the storage engine, or returning data values requested by a query. The formula engine, due to the serial nature of the calculations it evaluates, is single threaded. Each query executes primarily on a single thread, requesting and often waiting for data returned by the storage engine. Query threads have longer executions, and are released only after the entire query is completed.

By default, on versions SQL Server 2012 (11.x) and later, Analysis Services will use all available logical processors, up to 640 on systems running higher editions of Windows and SQL Server. Upon start up, the msmdsrv.exe process will be assigned to a specific processor group, but over time threads can be scheduled on any logical processor, in any processor group.

One side-effect of using a large number of processors is that you can sometimes experience performance degradation as query and processing loads are spread out across a large number of processors and contention for shared data structures increase. This can occur particularly on high-end systems that use NUMA architecture, but also on non-NUMA systems running multiple data intensive applications on the same hardware.

To alleviate this problem, you can set affinity between types of Analysis Services operations and a specific set of logical processors. The **GroupAffinity** property lets you create custom affinity masks that specify which system resource to use for each of the thread pool types managed by Analysis Services.

We recommend SQL Server 2016 Cumulative Update 1 (CU1) or later for setting **GroupAffinity** in tabular instances.

GroupAffinity is a property that can be set on any of the thread pools used for various Analysis Services workloads:

- **ThreadPool \ Parsing \ Short** is a parsing pool for short requests. Requests that fit within a single

network message are considered short.

- **ThreadPool \ Parsing \ Long** is a parsing pool for all other requests that do not fit within a single network message.

> **NOTE**
> A thread from either parsing pool can be used to execute a query. Queries that execute quickly, such as a fast Discover or Cancel request, are sometimes executed immediately rather than queued to the Query thread pool.

- **ThreadPool \ Query** is the thread pool that executes all requests that are not handled by the parsing thread pool. Threads in this thread pool will execute all types of operations, such as Discovers, MDX, DAX, DMX, and DDL commands. A
- **ThreadPool \ IOProcess** is used for IO jobs associated with storage engine queries in the multidimensional engine. The work done by these threads is expected to not have dependencies on other threads. These threads will typically be scanning a single segment of a partition and performing filtering and aggregation on the segment data. **IOProcess** threads are particularly sensitive to NUMA hardware configurations. As such, this thread pool has the **PerNumaNode** configuration property which can be used to tune the performance if needed.
- **ThreadPool \ Process** is for longer duration storage engine jobs, including aggregations, indexing, and commit operations. ROLAP storage mode also uses threads from the Processing thread pool.
- **VertiPaq \ ThreadPool** is the thread pool for executing table scans in a tabular model.

To service requests, Analysis Services may exceed the maximum thread pool limit, requesting additional threads if they are necessary to perform the work. However, when a thread finishes executing its task, if the current count of threads is greater than the maximum limit, the thread is simply ended, rather than returned to the thread pool.

> **NOTE**
> Exceeding the maximum thread pool count is a protection invoked only when certain deadlock conditions arise. To prevent runaway thread creation beyond the maximum, threads are created gradually (after a short delay) after the maximum limit has been reached. Exceeding maximum thread count can lead to a slowdown in task execution. If the performance counters show the thread counts are regularly beyond the thread pool maximum size, you might consider that as an indicator that thread pool sizes are too small for the degree of concurrency being requested from the system.

By default, thread pool size is determined by Analysis Services, and is based on the number of cores. You can observe the selected default values by examining the msmdsrv.log file after server startup. As a performance tuning exercise, you might choose to increase the size of the thread pool, as well as other properties, to improve query or processing performance.

Thread Pool Property Reference

This section describes the thread pool properties found in the msmdsrv.ini file of each Analysis Services instance. A subset of these properties also appears in SQL Server Management Studio.

Properties are listed in alphabetical order.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
IOProcess \ Concurrency	double	A double-precision floating point value that determines the algorithm for setting a target on the number of threads that can be queued at one time.	2.0	An advanced property that you should not change, except under the guidance of Microsoft support. Concurrency is used to initialize thread pools, which are implemented using IO Completion Ports in Windows. See I/O Completion Ports for details. Applies to multidimensional models only.
IOProcess \ GroupAffinity	string	An array of hexadecimal values that correspond to processor groups on the system, used to set affinity of threads in the IOProcess thread pool to logical processors in each processor group.	none	You can use this property to create custom affinities. The property is empty by default. See Set GroupAffinity to affinitize threads to processors in a processor group for details. Applies to multidimensional models only.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
IOProcess \ MaxThreads	int	A signed 32-bit integer that specifies the maximum number of threads to include in the thread pool.	0	0 indicates the server determines the defaults. By default, the server either sets this value to 64, or to 10 times the number of logical processors, whichever is higher. For example, on a 4-core system with hyperthreading, the thread pool maximum is 80 threads. If you set this value to a negative value, the server multiples that value by the number of logical processors. For example, when set to -10 on a server having 32 logical processors, the maximum is 320 threads. The maximum value is subject to available processors per any custom affinity masks that you previously defined. For example, if you already set thread pool affinity to use 8 out of 32 processors, and you now set MaxThreads to -10, then the upper bound on the thread pool would 10 times 8, or 80 threads. The actual values used for this thread pool property are written to the msmdsrv log file upon service start up. More information about tuning the thread pool settings can be found in the Analysis Services Operations Guide. Applies to multidimensional models only.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
IOProcess \ MinThreads	int	A signed 32-bit integer that specifies the minimum number of threads to preallocate for the thread pool.	0	0 indicates the server determines the defaults. By default, the minimum is 1. If you set this value to a negative value, the server multiples that value by the number of logical processors. The actual values used for this thread pool property are written to the msmdsrv log file upon service start up. More information about tuning the thread pool settings can be found in the Analysis Services Operations Guide. Applies to multidimensional models only.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
IOProcess \ PerNumaNode	int	A signed 32-bit integer that determines the number of thread pools created for the msmdsrv process.	-1	Valid values are -1, 0, 1, 2 -1 = The server selects a different IO Thread Pool strategy based on the number of NUMA nodes. On systems having fewer than 4 NUMA nodes, the server behavior is the same as 0 (one IOProcess thread pool is created for the system). On systems having 4 or more nodes, the behavior is the same as 1 (IOProcess thread pools are created for each node). 0 = Disables per NUMA node thread pools so that there is only one IOProcess thread pool used by the msmdsrv.exe process. 1 = Enables one IOProcess thread pool per NUMA node. 2 = One IOProcess thread pool per logical processor. Threads in each thread pool are affinitized to the NUMA node of the logical processor, with the ideal processor set to the logical processor. See Set PerNumaNode to affinitize IO threads to processors in a NUMA node for details. Applies to multidimensional models only.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
IOProcess \ PriorityRatio	int	A signed 32-bit integer that can be used to ensure that lower priority threads are sometimes executed even when a higher priority queue is not empty.	2	An advanced property that you should not change, except under the guidance of Microsoft support. Applies to multidimensional models only.
IOProcess \ StackSizeKB	int	A signed 32-bit integer that can be used to adjust memory allocation during thread execution.	0	An advanced property that you should not change, except under the guidance of Microsoft support. Applies to multidimensional models only.
Parsing \ Long \ Concurrency	double	A double-precision floating point value that determines the algorithm for setting a target on the number of threads that can be queued at one time.	2.0	An advanced property that you should not change, except under the guidance of Microsoft support. Concurrency is used to initialize thread pools, which are implemented using IO Completion Ports in Windows. See I/O Completion Ports for details.
Parsing \ Long \ GroupAffinity	string	An array of hexadecimal values that correspond to processor groups on the system, used to set affinity of parsing threads to logical processors in each processor group.	none	You can use this property to create custom affinities. The property is empty by default. See Set GroupAffinity to affinitize threads to processors in a processor group for details.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
Parsing \ Long \ NumThreads	int	A signed 32-bit integer property that defines the number of threads that can be created for long commands.	0	0 indicates that the server determines the defaults. The default behavior is to set **NumThreads** to an absolute value of 4, or 2 times the number of logical processors, whichever is higher. If you set this value to a negative value, the server multiples that value by the number of logical processors. For example, when set to -10 on a server having 32 logical processors, the maximum is 320 threads. The maximum value is subject to available processors per any custom affinity masks that you previously defined. For example, if you already set thread pool affinity to use 8 out of 32 processors, and you now set NumThreads to -10, then the upper bound on the thread pool would 10 times 8, or 80 threads. The actual values used for this thread pool property are written to the msmdsrv log file upon service start up.
Parsing \ Long \ PriorityRatio	int	A signed 32-bit integer that can be used to ensure that lower priority threads are sometimes executed even when a higher priority queue is not empty.	0	An advanced property that you should not change, except under the guidance of Microsoft support.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
Parsing \ Long \ StackSizeKB	int	A signed 32-bit integer that can be used to adjust memory allocation during thread execution.	0	An advanced property that you should not change, except under the guidance of Microsoft support.
Parsing \ Short \ Concurrency	double	A double-precision floating point value that determines the algorithm for setting a target on the number of threads that can be queued at one time.	2.0	An advanced property that you should not change, except under the guidance of Microsoft support. Concurrency is used to initialize thread pools, which are implemented using IO Completion Ports in Windows. See I/O Completion Ports for details.
Parsing \ Short \ GroupAffinity	string	An array of hexadecimal values that correspond to processor groups on the system, used to set affinity of parsing threads to logical processors in each processor group.	none	You can use this property to create custom affinities. The property is empty by default. See Set GroupAffinity to affinitize threads to processors in a processor group for details.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
Parsing \ Short \ NumThreads	int	A signed 32-bit integer property that defines the number of threads that can be created for short commands.	0	0 indicates that the server determines the defaults. The default behavior is to set **NumThreads** to an absolute value of 4, or 2 times the number of logical processors, whichever is higher. If you set this value to a negative value, the server multiples that value by the number of logical processors. For example, when set to -10 on a server having 32 logical processors, the maximum is 320 threads. The maximum value is subject to available processors per any custom affinity masks that you previously defined. For example, if you already set thread pool affinity to use 8 out of 32 processors, and you now set NumThreads to -10, then the upper bound on the thread pool would 10 times 8, or 80 threads. The actual values used for this thread pool property are written to the msmdsrv log file upon service start up.
Parsing \ Short \ PriorityRatio	int	A signed 32-bit integer that can be used to ensure that lower priority threads are sometimes executed even when a higher priority queue is not empty.	0	An advanced property that you should not change, except under the guidance of Microsoft support.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
Parsing \ Short \ StackSizeKB	int	A signed 32-bit integer that can be used to adjust memory allocation during thread execution.	64 * logical processors	An advanced property that you should not change, except under the guidance of Microsoft support.
Process \ Concurrency	double	A double-precision floating point value that determines the algorithm for setting a target on the number of threads that can be queued at one time.	2.0	An advanced property that you should not change, except under the guidance of Microsoft support. Concurrency is used to initialize thread pools, which are implemented using IO Completion Ports in Windows. See I/O Completion Ports for details.
Process \ GroupAffinity	string	An array of hexadecimal values that correspond to processor groups on the system, used to set affinity of processing threads to logical processors in each processor group.	none	You can use this property to create custom affinities. The property is empty by default. See Set GroupAffinity to affinitize threads to processors in a processor group for details.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
Process \ MaxThreads	int	A signed 32-bit integer that specifies the maximum number of threads to include in the thread pool.	0	0 indicates the server determines the defaults. By default, the server either sets this value to an absolute value of 64, or the number of logical processors, whichever is higher. For example, on a 64-core system with hyperthreading enabled (resulting in 128 logical processors), the thread pool maximum is 128 threads. If you set this value to a negative value, the server multiples that value by the number of logical processors. For example, when set to -10 on a server having 32 logical processors, the maximum is 320 threads. The maximum value is subject to available processors per any custom affinity masks that you previously defined. For example, if you already set thread pool affinity to use 8 out of 32 processors, and you now set MaxThreads to -10, then the upper bound on the thread pool would 10 times 8, or 80 threads. The actual values used for this thread pool property are written to the msmdsrv log file upon service start up. More information about tuning the thread pool settings can be found in the Analysis Services Operations Guide.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
Process \ MinThreads	int	A signed 32-bit integer that specifies the minimum number of threads to preallocate for the thread pool.	0	0 indicates the server determines the defaults. By default, the minimum is 1. If you set this value to a negative value, the server multiples that value by the number of logical processors. The actual values used for this thread pool property are written to the msmdsrv log file upon service start up. More information about tuning the thread pool settings can be found in the Analysis Services Operations Guide.
Process \ PriorityRatio	int	A signed 32-bit integer that can be used to ensure that lower priority threads are sometimes executed even when a higher priority queue is not empty.	2	An advanced property that you should not change, except under the guidance of Microsoft support.
Process \ StackSizeKB	int	A signed 32-bit integer that can be used to adjust memory allocation during thread execution.	0	An advanced property that you should not change, except under the guidance of Microsoft support.
Query \ Concurrency	double	A double-precision floating point value that determines the algorithm for setting a target on the number of threads that can be queued at one time.	2.0	An advanced property that you should not change, except under the guidance of Microsoft support. Concurrency is used to initialize thread pools, which are implemented using IO Completion Ports in Windows. See I/O Completion Ports for details.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
Query \ GroupAffinity	string	An array of hexadecimal values that correspond to processor groups on the system, used to set affinity of processing threads to logical processors in each processor group.	none	You can use this property to create custom affinities. The property is empty by default. See Set GroupAffinity to affinitize threads to processors in a processor group for details.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
Query \ MaxThreads	int	A signed 32-bit integer that specifies the maximum number of threads to include in the thread pool.	0	0 indicates the server determines the defaults. By default, the server either sets this value to an absolute value of 10, or 2 times the number of logical processors, whichever is higher. For example, on a 4-core system with hyperthreading, the maximum thread count is 16. If you set this value to a negative value, the server multiples that value by the number of logical processors. For example, when set to -10 on a server having 32 logical processors, the maximum is 320 threads. The maximum value is subject to available processors per any custom affinity masks that you previously defined. For example, if you already set thread pool affinity to use 8 out of 32 processors, and you now set MaxThreads to -10, then the upper bound on the thread pool would 10 times 8, or 80 threads. The actual values used for this thread pool property are written to the msmdsrv log file upon service start up. More information about tuning the thread pool settings can be found in the Analysis Services Operations Guide.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
Query \ MinThreads	int	A signed 32-bit integer that specifies the minimum number of threads to preallocate for the thread pool.	0	0 indicates the server determines the defaults. By default, the minimum is 1. If you set this value to a negative value, the server multiples that value by the number of logical processors. The actual values used for this thread pool property are written to the msmdsrv log file upon service start up. More information about tuning the thread pool settings can be found in the Analysis Services Operations Guide.
Query \ PriorityRatio	int	A signed 32-bit integer that can be used to ensure that lower priority threads are sometimes executed even when a higher priority queue is not empty.	2	An advanced property that you should not change, except under the guidance of Microsoft support.
Query \ StackSizeKB	int	A signed 32-bit integer that can be used to adjust memory allocation during thread execution.	0	An advanced property that you should not change, except under the guidance of Microsoft support.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
VertiPaq \ CPUs	int	A signed 32-bit integer that specifies the maximum number of processors to use for tabular queries.	0	0 indicates the server determines the defaults. By default, the server either sets this value to an absolute value of 10, or 2 times the number of logical processors, whichever is higher. For example, on a 4-core system with hyperthreading, the maximum thread count is 16. If you set this value to a negative value, the server multiples that value by the number of logical processors. For example, when set to -10 on a server having 32 logical processors, the maximum is 320 threads. The maximum value is subject to available processors per any custom affinity masks that you previously defined. For example, if you already set thread pool affinity to use 8 out of 32 processors, and you now set MaxThreads to -10, then the upper bound on the thread pool would 10 times 8, or 80 threads. The actual values used for this thread pool property are written to the msmdsrv log file upon service start up.

NAME	TYPE	DESCRIPTION	DEFAULT	GUIDANCE
VertiPaq \ GroupAffinity	string	An array of hexadecimal values that correspond to processor groups on the system, used to set affinity of processing threads to logical processors in each processor group.	none	You can use this property to create custom affinities. The property is empty by default. See Set GroupAffinity to affinitize threads to processors in a processor group for details. Applies to Tabular only.

Set GroupAffinity to affinitize threads to processors in a processor group

GroupAffinity is provided for advanced tuning purposes. You can use the **GroupAffinity** property to set affinity between Analysis Services thread pools and specific processors; however, for most installations, Analysis Services performs best when it can use all available logical processors. Accordingly, group affinity is unspecified by default.

Should performance testing indicate a need for CPU optimization, you might consider a higher level approach, such as using Windows Server Resource Manager to set affinity between logical processors and a server process. Such an approach can be simpler to implement and manage than defining custom affinities for individual thread pools.

If that approach is insufficient, you can achieve greater precision by defining custom affinities for thread pools. Customizing affinity settings is more likely to be recommended on large multi-core systems (either NUMA or non-NUMA) experiencing performance degradation due to thread pools spread out over too-wide a range of processors. Although you can set **GroupAffinity** on systems having fewer than 64 logical processors, the benefit is negligible and might even degrade performance.

> **NOTE**
>
> **GroupAffinity** is constrained by editions that limit the number of cores used by Analysis Services. At startup, Analysis Services uses edition information and the **GroupAffinity** properties to compute affinity masks for each thread pool managed by Analysis Services. Standard edition can use a maximum of 24 cores. If you install Analysis Services standard edition on a large multi-core system that has more than 24 cores, Analysis Services will only use 24 of them. For more information about processor maximums, see the cross-box scale limits in Features by editions in SQL Server.

Syntax

The value is hexadecimal for each processor group, with the hexadecimal representing the logical processors that Analysis Services attempts to use first when allocating threads for a given thread pool.

Bitmask for Logical Processors

You can have up to 64 logical processors within a single processor group. The bitmask is 1 (or 0) for each logical processor in the group that is used (or not used) by a thread pool. Once you compute the bitmask, you then calculate the hexadecimal value as the value for **GroupAffinity**.

Multiple processor groups

Processor groups are determined on system startup. **GroupAffinity** accepts hexadecimal values for each processor group in a comma delimited list. Given multiple processor groups (up to 10 on higher end systems), you can bypass individual groups by specifying 0x0. For example, on a system with four processor groups (0, 1, 2, 3),

you could exclude groups 0 and 2 by entering 0x0 for the first and third values.

```
<GroupAffinity>0x0, 0xFF, 0x0, 0xFF</GroupAffinity>
```

Steps for computing the processor affinity mask

You can set **GroupAffinity** in msmdsrv.ini or in server property pages in SQL Server Management Studio.

1. **Determine the number of processors and processor groups**

   You can download Coreinfo utility from winsysinternals.

   Run **coreinfo** to get this information from the Logical Processor to Group Map section. A separate line is generated for each logical processor.

2. Sequence the processors, from right to left: `7654 3210`

   The example shows only 8 processors (0 through 7), but a processor group can have a maximum of 64 logical processors, and there can be up to 10 processor groups in an enterprise-class Windows server.

3. **Compute the bitmask for the processor groups you want to use**

   ```
   7654 3210
   ```

   Replace the number with a 0 or 1, depending on whether you want to exclude or include the logical processor. On a system having eight processors, your calculation might look like this if you wanted to use processors 7, 6, 5, 4, and 1 for Analysis Services:

   ```
   1111 0010
   ```

4. **Convert the binary number to a Hex value**

   Using a calculator or conversion tool, convert the binary number to its hexadecimal equivalent. In our example, `1111 0010` converts to `0xF2`.

5. **Enter the hex value in the GroupAffinity property**

   In msmdsrv.ini or in the server property page in Management Studio, set **GroupAffinity** to the value calculated in step 4.

> **IMPORTANT**
>
> Setting **GroupAffinity** is a manual task encompassing multiple steps. When computing **GroupAffinity**, check your calculations carefully. Although Analysis Services will return an error if the entire mask is invalid, a combination of valid and invalid settings results in Analysis Services ignoring the property. For example, if the bitmask includes extra values, Analysis Services ignores the setting, using all processors on the system. There is no error or warning to alert you when this action occurs, but you can check the msmdsrv.log file to learn how the affinities are actually set.

Set PerNumaNode to affinitize IO threads to processors in a NUMA node

For multidimensional Analysis Services instances, you can set **PerNumaNode** on the **IOProcess** thread pool to further optimize thread scheduling and execution. Whereas **GroupAffinity** identifies which set of logical processors to use for a given thread pool, **PerNumaNode** goes one step further by specifying whether to create multiple thread pools, further affinitized to some subset of the allowed logical processors.

> **NOTE**
>
> On Windows Server 2012, use Task Manager to view the number of NUMA nodes on the computer. In Task Manager, on the Performance tab, select **CPU** and then right-click the graph area to view NUMA nodes. Alternatively, download the Coreinfo utility from Windows Sysinternals and run `coreinfo -n` to return NUMA nodes and logical processors in each node.

Valid values for **PerNumaNode** are -1, 0, 1, 2, as described in the Thread Pool Property Reference section in this topic.

Default (Recommended)

On systems having NUMA nodes, we recommend using the default setting of PerNumaNode=-1, allowing Analysis Services to adjust the number of thread pools and their thread affinity based on node count. If the system has fewer than 4 nodes, Analysis Services implements the behaviors described by **PerNumaNode**=0, whereas **PerNumaNode**=1 is used on systems having 4 or more nodes.

Choosing a value

You can also override the default to use another valid value.

Setting PerNumaNode=0

NUMA nodes are ignored. There will be just one IOProcess thread pool, and all threads in that thread pool will be affinitized to all logical processors. By default (where PerNumaNode=-1), this is the operative setting if the computer has fewer than 4 NUMA nodes.

NUMA Nodes	0								1								2								3							
Logical Processors	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
IO Thread Pools	0																															

Setting PerNumaNode=1

IOProcess thread pools are created for each NUMA node. Having separate thread pools improves coordinated access to local resources, such as local cache on a NUMA node.

NUMA Nodes	0								1								2								3							
Logical Processors	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
IO Thread Pools	0								1								2								3							

Setting PerNumaNode=2

This setting is for very high-end systems running intensive Analysis Services workloads. This property sets IOProcess thread pool affinity at its most granular level, creating and affinitizing separate thread pools at the logical processor level.

In the following example, on a system having 4 NUMA nodes and 32 logical processors, setting **PerNumaNode** to 2 would result in 32 IOProcess thread pools. The threads in the first 8 thread pools would be affinitized to all the logical processors in the NUMA node 0, but with the ideal processor set to 0, 1, 2, up to 7. The next 8 thread pools would be affinitized to all the logical processors in NUMA node 1, with the ideal processor set to 8, 9, 10, up to 15, and so on.

NUMA Nodes	0								1								2								3							
Logical Processors	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
IO Thread Pools	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

At this level of affinity, the scheduler always attempts to use the ideal logical processor first, within the preferred NUMA node. If the logical processor is unavailable, the scheduler chooses another processor within the same node, or within the same processor group if no other threads are available. For more information and examples, see Analysis Services 2012 Configuration settings (Wordpress Blog).

Work distribution among IOProcess threads

As you consider whether to set the **PerNumaNode** property, knowing how **IOProcess** threads are used can help you make a more informed decision.

Recall that **IOProcess** is used for IO jobs associated with storage engine queries in the multidimensional engine.

When a segment is scanned, the engine identifies the partition to which the segment belongs, and attempts to queue the segment job to the thread pool used by the partition. In general, all segments belonging to a partition will queue their tasks to the same thread pool. On NUMA systems, this behavior is particularly advantageous because all scans for a partition will use memory in the file system cache that is allocated locally to that NUMA node.

The following scenarios suggest adjustments that can sometimes improve query performance on NUMA systems:

- For measure groups that are under-partitioned (for example, having a single partition), increase the number of partitions. Using just one partition will cause the engine to always queue tasks to one thread pool (thread pool 0). Adding more partitions allows the engine to use additional thread pools.

  Alternatively, if you cannot create additional partitions, try setting **PerNumaNode**=0 as a way to increase the number of threads available to thread pool 0.

- For databases in which segment scans are evenly distributed across multiple partitions, setting **PerNumaNode** to 1 or 2 can improve query performance because it increases the overall number of **IOProcess** thread pools used by the system.

- For solutions that have multiple partitions, but only one is heavily scanned, try setting **PerNumaNode**=0 to see if it improves performance.

Although both partition and dimension scans use the **IOProcess** thread pool, dimension scans only use thread pool 0. This can result in a slightly uneven load on that thread pool, but the imbalance should be temporary, as dimension scans tend to be very fast and infrequent.

> **NOTE**
>
> When changing a server property, remember that the configuration option applies to all databases running on the current instance. Choose settings that benefit the most important databases, or the greatest number of databases. You cannot set processor affinity at the database level, nor can you set affinity between individual partitions and specific processors.

For more information about job architecture, see section 2.2 in SQL Server Analysis Services Performance Guide.

Dependent or Related Properties

As explained in section 2.4 of the Analysis Services Operations Guide, if you increase the processing thread pool, you should make sure that the **CoordinatorExecutionMode** settings, as well as the **CoordinatorQueryMaxThreads** settings, have values that enable you to make full use of the increased thread pool size.

Analysis Services uses a coordinator thread for gathering the data needed to complete a processing or query request. The coordinator first queues up one job for each partition that must be touched. Each of those jobs then continues to queue up more jobs, depending on the total number of segments that must be scanned in the partition.

The default value for **CoordinatorExecutionMode** is -4, meaning a limit of 4 jobs in parallel per core, which constrains the total number of coordinator jobs that can be executed in parallel by a subcube request in the storage engine.

The default value for **CoordinatorQueryMaxThreads** is 16, which limits the number of segment jobs that can be

executed in parallel for each partition.

Determine current thread pool settings

At each service startup, Analysis Services outputs the current thread pool settings into the msmdsrv.log file, including minimum and maximum threads, processor affinity mask, and concurrency.

The following example is an excerpt from the log file, showing the default settings for the Query thread pool (MinThread=0, MaxThread=0, Concurrency=2), on a 4-core system with hyper-threading enabled. The affinity mask is 0xFF, indicating 8 logical processors. Notice that leading zeros are prepended to the mask. You can ignore the leading zeros.

```
"10/28/2013 9:20:52 AM) Message: The Query thread pool now has 1 minimum threads, 16 maximum threads, and a
concurrency of 16. Its thread pool affinity mask is 0x00000000000000ff. (Source: \\?\C:\Program Files\Microsoft
SQL Server\MSAS11.MSSQLSERVER\OLAP\Log\msmdsrv.log, Type: 1, Category: 289, Event ID: 0x4121000A)"
```

Recall that the algorithm for setting **MinThread** and **MaxThread** incorporates system configuration, particularly the number of processors. The following blog post offers insights into how the values are calculated: Analysis Services 2012 Configuration settings (Wordpress Blog). Note that these settings and behaviors are subject to adjustment in subsequent releases.

The following list shows examples of other affinity mask settings, for different combinations of processors:

- Affinity for processors 3-2-1-0 on an 8 core system results in this bitmask: 00001111, and a hexadecimal value: 0xF
- Affinity for processors 7-6-5-4 on an 8 core system results in this bitmask: 11110000, and a hexadecimal value: 0xF0
- Affinity for processors 5-4-3-2 on an 8 core system results in this bitmask: 00111100, and a hexadecimal value: 0x3C
- Affinity for processors 7-6-1-0 on an 8 core system results in this bitmask: 11000011, and a hexadecimal value: 0xC3

Recall that on systems having multiple processor groups, a separate affinity mask is generated for each group, in a comma separated list.

About MSMDSRV.INI

The msmdsrv.ini file contains configuration settings for an Analysis Services instance, affecting all databases running on that instance. You cannot use server configuration properties to optimize performance of just one database to the exclusion of all others. However, you can install multiple instances of Analysis Services and configure each instance to use properties that benefit databases sharing similar characteristics or workloads.

All server configuration properties are included in the msmdsrv.ini file. Subsets of the properties more likely to be modified also appear in administration tools, such as SSMS.

The contents of msmdsrv.ini are identical for both Tabular and Multidimensional instances of Analysis Services. However, some settings apply to one mode only. Differences in behavior based on server mode are noted in property reference documentation.

> **NOTE**
>
> For instructions on how to set properties, see Server Properties in Analysis Services.

See Also

About Processes and Threads
Multiple Processors
Processor Groups
Analysis Services Thread Pool Changes in SQL Server 2012
Analysis Services 2012 Configuration settings (Wordpress Blog)
Supporting Systems That Have More Than 64 Processors
SQL Server Analysis Services Operations Guide

Monitoring overview

1/28/2019 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services has several different tools to help you monitor and tune the performance of your servers. The choice of tool depends on the type of monitoring or tuning to be done and the particular events to be monitored.

For more information about monitoring SQL Server Analysis Services, see the SQL Server 2008 R2 Operations Guide.

Monitoring tools

TOOL	DESCRIPTION
SQL Server Profiler	Tracks engine process events. It also captures data about those events, enabling you to monitor server and database activity.
Extended Events	A light-weight tracing and performance monitoring system that uses very few system resources, making it an ideal tool for diagnosing problems on both production and test servers.
Dynamic Management Views (DMVs)	Queries that return information about model objects, server operations, and server health. The query, based on SQL, is an interface to schema rowsets.
Trace Events	Follow the activity of an instance by capturing and then analyzing the trace events generated by the instance. Trace events are grouped so that you can more easily find related trace events.
Performance counters*	Using Performance Monitor, you can monitor the performance of a Microsoft SQL Server Analysis Services (SSAS) instance by using performance counters.
Log operations*	A SQL Server Analysis Services instance will log server notifications, errors, and warnings to the msmdsrv.log file - one for each instance you install.

* Applies to SQL Server Analysis Services only.

See also

Monitor Azure Analysis Services server metrics
Setup diagnostic logging for Azure Analysis Services

Use SQL Server Profiler to Monitor Analysis Services

11/15/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

SQL Server Profiler tracks engine process events, such as the start of a batch or a transaction, and it captures data about those events, thus enabling you to monitor server and database activity (for example, user queries or login activity). You can capture SQL Server Profiler data to a SQL Server table or a file for later analysis, and you can also replay the events captured on the same or another Analysis Services instance to see exactly what happened. You can replay events in real time or on a step-by-step basis. It is also very useful to run the trace events along with the Performance counters on the same machine. The profiler can correlate these two based on time and display them together along a single timeline. Trace events will give you more details while Performance counters give you an aggregate view. For information about how to create and run traces, see Create Profiler Traces for Replay (Analysis Services).

The following table describes the topics in this section.

In This Section

TOPIC	DESCRIPTION
Introduction to Monitoring Analysis Services with SQL Server Profiler	Describes how to use SQL Server Profiler to monitor an Analysis Services instance.
Create Profiler Traces for Replay (Analysis Services)	Describes the requirements for creating a trace for replay by using SQL Server Profiler.
Analysis Services Trace Events	Describes Analysis Services event classes. These event classes map to actions generated by Analysis Services and are used for trace replays.

Introduction to Monitoring Analysis Services with SQL Server Profiler

1/8/2019 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

You can use SQL Server Profiler to monitor events generated by an instance of Microsoft SQL Server Analysis Services. By using SQL Server Profiler, you can do the following:

- Monitor the performance of an instance of Analysis Services.
- Debug Multidimensional Expressions (MDX) statements.
- Identify MDX statements that run slowly.
- Test MDX statements in the development phase of a project by stepping through statements to confirm that the code works as expected.
- Troubleshoot problems in Analysis Services by capturing events on a production system and replaying them on a test system. This approach is useful for testing or debugging purposes and lets users continue to use the production system without interference.
- Audit and review activity that occurred on an instance of Analysis Services. A security administrator can review any one of the audited events. This includes the success or failure of a login try and the success or failure of permissions in accessing statements and objects.
- Display data about the captured events to the screen, or capture and save data about each event to a file or SQL Server table for future analysis or playback. When you replay data, you can rerun the saved events as they originally occurred, either in real time or step by step.

Using SQL Server Profiler

To use SQL Server Profiler to create or replay traces, you must be a member of the Analysis Services server role. If you are a member of the Analysis Services server role, you can start SQL Server Profiler from the Microsoft SQL Server program group on the **Start** menu.

When you use SQL Server Profiler, note the following:

- Trace definitions are stored with the Analysis Services database by using the CREATE statement.
- Multiple traces can be running at the same time.
- Multiple connections can receive events from the same trace.
- A trace can continue when Analysis Services stops and restarts.

> **NOTE**
>
> Passwords are not revealed in trace events, but are replaced by ****** in the event.

For optimal performance, use SQL Server Profiler to monitor only those events in which you are most interested. Monitoring too many events adds overhead and can cause the trace file or table to grow very large, especially when you monitor over a long period of time. In addition, use filtering to limit the amount of data that is collected

and to prevent traces from becoming too large.

See Also

Analysis Services Trace Events
Create Profiler Traces for Replay (Analysis Services)

Create Profiler Traces for Replay (Analysis Services)

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

To replay queries, discovers, and commands that are submitted by users to Microsoft SQL Server Analysis Services, SQL Server Profiler must gather the required events. In order to initiate collection of these events, appropriate event classes must be selected in the **Event Selection** tab of the **Trace Properties** dialog box. For example if the Query Begin event class is selected, events that contain queries are collected and used for replay. Also, the trace file contains sufficient information to support replaying server transactions in a distributed environment in the original sequence of transactions.

Replay for Queries

To replay queries, SQL Server Profiler must capture the following events:

- Audit Login event class with all its data columns. This event class provides information about which user logged in and about the session settings. The server process ID (SPID) provides the reference to the user session. For more information, see Security Audit Data Columns.
- Query Begin event class with all its data columns. This event class provides information about the query that was submitted to Analysis Services. The Event Subclass column provides information about the type of query. The TextData column provides the actual text of the query. The RequestParameters column provides the parameters for parameterized queries, and the RequestProperties column provides the properties of an XML for Analysis (XMLA) request. For more information, see Queries Events Data Columns.
- Query End event class with all its data columns. This event class verifies the status of the query execution. For more information, see Queries Events Data Columns.

Replay for Discovers

To replay discovers, SQL Server Profiler must capture the following events:

- Audit Login event class with all its data columns. This event class provides information about which user logged in and about the session settings. The SPID provides the reference to the user session. For more information, see Security Audit Data Columns.
- Discover Begin event class with all its data columns. The TextData column provides the <RequestType> portion of the discover request, and the RequestProperties column provides the <Properties> portion of the discover request. The EventSubclass column provides the discover type. For more information, see Discover Events Data Columns.
- Discover End event class with all its data columns. This event class verifies the status of the discover request. For more information, see Discover Events Data Columns.

Replay for Commands

To replay commands, SQL Server Profiler must capture the following events:

- Command Begin event class with all its data columns. The TextData column provides the command details, such as the process type, database ID, and cube ID. The RequestParameters column provides the parameters for parameterized command, and the RequestProperties column provides the properties of an

XMLA request. For more information, see Command Events Data Columns.

- Command End event class with all its data columns. This event class verifies the status of the command. For more information, see Command Events Data Columns.

See Also

Analysis Services Trace Events
Introduction to Monitoring Analysis Services with SQL Server Profiler

Monitor Analysis Services with SQL Server Extended Events

11/28/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Extended Events (*xEvents*) is a light-weight tracing and performance monitoring system that uses very few system resources, making it an ideal tool for diagnosing problems on both production and test servers. It's also highly scalable, configurable, and in SQL Server 2016 , easier to use through new built-in tool support. In SQL Server Management Studio, on connections to Analysis Services instances, you can configure, run, and monitor a live trace, similar to using SQL Server Profiler. The addition of better tooling should make xEvents a more reasonable replacement for SQL Server Profiler and creates more symmetry in how you diagnose issues in your database engine and Analysis Services workloads.

Besides SQL Server Management Studio, you can also configure Analysis Services Extended Event sessions the old way, through XMLA scripting, as was supported in previous releases.

All Analysis Services events can be captured and targeted to specific consumers, as defined in Extended Events.

> **NOTE**
>
> Watch this quick video introduction or read the supporting blog post to learn more about xEvents for Analysis Services in SQL Server 2016.

Use Management Studio to Configure Analysis Services

For both tabular and multidimensional instances, Management Studio provides a new Management folder that contains user-initiated xEvent sessions. You can run multiple sessions at once. However, in the current implementation, the Analysis Services Extended Events user interface does not support updating or replaying an existing session.

Choose Events

If you already know which events you want to capture, searching for them is the easiest way to add them to the trace. Otherwise, the following events are commonly used for monitoring operations:

- **CommandBegin** and **CommandEnd**.
- **QueryBegin**, **QueryEnd**, and **QuerySubcubeVerbose** (shows the entire MDX or DAX query sent to the server), plus **ResourceUsage** for stats on resources consumed by the query and how many rows are returned.

- **ProgressReportBegin** and **ProgressReportEnd** (for processing operations).
- **AuditLogin** and **AuditLogout** (captures the user identity under which a client application connects to Analysis Services).

Choose Data Storage

A session can be streamed live to a window in Management Studio or persisted to a file for subsequent analysis using Power Query or Excel.

- **event_file** stores session data in an .xel file.
- **event_stream** enables the **Watch Live Data** option in Management Studio.
- **ring_buffer** stores session data in memory for as long as the server is running. On a server restart, the session data is thrown out

Add Event Fields

Be sure to configure the session to include event fields so that you can easily see information of interest.

Configure is an option on the far side of the dialog box.

In configuration, on the Event Fields tab, select **TextData** so that this field appears adjacent to the event, showing return values, including queries that are executing on the server.

After you configure a session for the desired events and data storage, you can click the script button to send your configuration to one of supported destinations including a file, a new query in SQL Server Management Studio, and the clipboard.

Refresh Sessions

Once you create the session, be sure to refresh the Sessions folder in Management Studio to see the session you just created. If you configured an event_stream, you can right-click the session name and choose **Watch Live Data** to monitor server activity in real time.

XMLA Script to Start Extended Events in Analysis Services

Extended Event tracing is enabled using a similar XMLA create object script command as shown below:

```
<Execute ...>
   <Command>
      <Batch ...>
         <Create ...>
            <ObjectDefinition>
               <Trace>
                  <ID>trace_id</ID>
                  <Name>trace_name</Name>
                  <ddl300_300:XEvent>
                     <event_session ...>
                        <event package="AS" name="AS_event">
                           <action package="PACKAGE0" .../>
                        </event>
                        <target package="PACKAGE0" name="asynchronous_file_target">
                           <parameter name="filename" value="data_filename.xel"/>
                           <parameter name="metadatafile" value="metadata_filename.xem"/>
                        </target>
                     </event_session>
                  </ddl300_300:XEvent>
               </Trace>
            </ObjectDefinition>
         </Create>
      </Batch>
   </Command>
   <Properties></Properties>
</Execute>
```

Where the following elements are to be defined by the user, depending on the tracing needs:

trace_id
Defines the unique identifier for this trace.

trace_name
The name given to this trace; usually a human readable definition of the trace. It is a common practice to use the *trace_id* value as the name.

AS_event
The Analysis Services event to be exposed. See Analysis Services Trace Events for names of the events.

data_filename
The name of the file that contains the events data. This name is suffixed with a time stamp to avoid data overwriting if the trace is sent over and over.

metadata_filename
The name of the file that contains the events metadata. This name is suffixed with a time stamp to avoid data overwriting if the trace is sent over and over.

XMLA Script to Stop Extended Events in Analysis Services

To stop the Extended Events tracing object you need to delete that object using a similar XMLA delete object script command as shown below:

```
<Execute xmlns="urn:schemas-microsoft-com:xml-analysis">
   <Command>
      <Batch ...>
         <Delete ...>
            <Object>
               <TraceID>trace_id</TraceID>
            </Object>
         </Delete>
      </Batch>
   </Command>
   <Properties></Properties>
</Execute>
```

Where the following elements are to be defined by the user, depending on the tracing needs:

trace_id
Defines the unique identifier for the trace to be deleted.

See Also

Extended Events

Dynamic Management Views (DMVs)

12/6/2018 • 9 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services Dynamic Management Views (DMV) are queries that return information about model objects, server operations, and server health. The query, based on SQL, is an interface to *schema rowsets*. Schema rowsets are predescribed tables that contain information about Analysis Services objects and server state, including database schema, active sessions, connections, commands, and jobs that are executing on the server.

DMV queries are an alternative to running XML/A Discover commands. For most administrators, writing a DMV query is simpler because the syntax is based on SQL. In addition, the result is returned in a table format that is easier to read and copy.

Most DMV queries use a **SELECT** statement and the **$System** schema with an XML/A schema rowset, for example:

```
SELECT * FROM $System.<schemaRowset>
```

DMV queries return information about server and object state at the time the query is run. To monitor operations in real-time, use tracing instead. To learn more about real-time monitoring using traces, see Use SQL Server Profiler to Monitor Analysis Services.

Query syntax

The query engine for DMVs is the Data Mining parser. The DMV query syntax is based on the SELECT (DMX) statement. Although DMV query syntax is based on a SQL SELECT statement, it does not support the full syntax of a SELECT statement. Notably, JOIN, GROUP BY, LIKE, CAST, and CONVERT are not supported.

```
SELECT [DISTINCT] [TOP <n>] <select list>
FROM $System.<schemaRowset>
[WHERE <condition expression>]
[ORDER BY <expression>[DESC|ASC]]
```

The following example for DISCOVER_CALC_DEPENDENCY illustrates the use of the WHERE clause for supplying a parameter to the query:

```
SELECT * FROM $System.DISCOVER_CALC_DEPENDENCY
WHERE OBJECT_TYPE = 'ACTIVE_RELATIONSHIP'
```

For schema rowsets that have restrictions, the query must include the SYSTEMRESTRICTSCHEMA function. The following example returns CSDL metadata about 1103 compatibility level tabular models. Note that CATALOG_NAME is case-sensitive:

```
Select * from SYSTEMRESTRICTSCHEMA ($System.Discover_csdl_metadata, [CATALOG_NAME] = 'Adventure Works DW')
```

Examples and scenarios

A DMV query can help you answer questions about active sessions and connections, and which objects are

consuming the most CPU or memory at a specific point in time. For example:

`Select * from $System.discover_object_activity`

This query reports on object activity since the service last started.

`Select * from $System.discover_object_memory_usage`

This query reports on memory consumption by object.

`Select * from $System.discover_sessions`

This query reports on active sessions, including session user and duration.

`Select * from $System.discover_locks`

This query returns a snapshot of the locks used at a specific point in time.

Tools and permissions

You can use any client application that supports MDX or DMX queries. In most cases, it's best to use SQL Server Management Studio. You must have server administrator permissions on the instance to query a DMV.

To run a DMV query from SQL Server Management Studio

1. Connect to the server and model object you want to query.
2. Right-click the server or database object > **New Query** > **MDX**.
3. Type your query, and then click **Execute**, or press F5.

Schema rowsets

Not all schema rowsets have a DMV interface. To return a list of all the schema rowsets that can be queried using DMV, run the following query.

```
SELECT * FROM $System.DBSchema_Tables
WHERE TABLE_TYPE = 'SCHEMA'
ORDER BY TABLE_NAME ASC
```

If a DMV is not available for a given rowset, the server returns error: `The <schemarowset> request type was not recognized by the server.` All other errors indicate problems with the syntax.

Schema rowsets are described in two SQL Server Analysis Services protocols:

[MS-SSAS-T]: SQL Server Analysis Services Tabular Protocol - Describes schema rowsets for tabular models at the 1200 and higher compatibility levels.

[MS-SSAS]: SQL Server Analysis Services Protocol - Describes schema rowsets for multidimensional models and tabular models at the 1100 and 1103 compatibility levels.

Rowsets described in the [MS-SSAS-T]: SQL Server Analysis Services Tabular Protocol

ROWSET	DESCRIPTION
TMSCHEMA_ANNOTATIONS	Provides information about the Annotation objects in the model.
TMSCHEMA_ATTRIBUTE_HIERARCHIES	Provides information about the AttributeHierarchy objects for a column.
TMSCHEMA_COLUMNS	Provides information about the Column objects in each table.

ROWSET	DESCRIPTION
TMSCHEMA_COLUMN_PERMISSIONS	Provides information about the ColumnPermission objects in each table-permission.
TMSCHEMA_CULTURES	Provides information about the Culture objects in the model.
TMSCHEMA_DATA_SOURCES	Provides information about the DataSource objects in the model.
TMSCHEMA_DETAIL_ROWS_DEFINITIONS	Provides information about the DetailRowsDefinition objects in the model.
TMSCHEMA_EXPRESSIONS	Provides information about the Expression objects in the model.
TMSCHEMA_EXTENDED_PROPERTIES	Provides information about the ExtendedProperty objects in the model.
TMSCHEMA_HIERARCHIES	Provides information about the Hierarchy objects in each table.
TMSCHEMA_KPIS	Provides information about the KPI objects in the model.
TMSCHEMA_LEVELS	Provides information about the Level objects in each hierarchy.
TMSCHEMA_LINGUISTIC_METADATA	Provides information about the synonyms for objects in the model for a particular culture
TMSCHEMA_MEASURES	Provides information about the Measure objects in each table.
TMSCHEMA_MODEL	Specifies a Model object in the database.
TMSCHEMA_OBJECT_TRANSLATIONS	Provides information about the translations of different objects for a culture.
TMSCHEMA_PARTITIONS	Provides information about the Partition objects in each table.
TMSCHEMA_PERSPECTIVE_COLUMNS	Provides information about the PerspectiveColumn objects in each PerspectiveTable object.
TMSCHEMA_PERSPECTIVE_HIERARCHIES	Provides information about the PerspectiveHierarchy objects in each PerspectiveTable object.
TMSCHEMA_PERSPECTIVE_MEASURES	Provides information about the PerspectiveMeasure objects in each PerspectiveTable object.
TMSCHEMA_PERSPECTIVE_TABLES	Provides information about the Table objects in a perspective.
TMSCHEMA_PERSPECTIVES	Provides information about the Perspective objects in the model.
TMSCHEMA_RELATIONSHIPS	Provides information about the Relationship objects in the model.

ROWSET	DESCRIPTION
TMSCHEMA_ROLE_MEMBERSHIPS	Provides information about the RoleMembership objects in each role.
TMSCHEMA_ROLES	Provides information about the Role objects in the model.
TMSCHEMA_TABLE_PERMISSIONS	Provides information about the TablePermission objects in each role.
TMSCHEMA_TABLES	Provides information about the Table objects in the model.
TMSCHEMA_VARIATIONS	Provides information about the Variation objects in each column.

Rowsets described in the [MS-SSAS]: SQL Server Analysis Services Protocol

ROWSET	DESCRIPTION
DBSCHEMA_CATALOGS	Describes the catalogs that are accessible on the server.
DBSCHEMA_COLUMNS	Returns a row for each measure, each cube dimension attribute, and each schema rowset column, exposed as a column.
DBSCHEMA_PROVIDER_TYPES	Identifies the (base) data types supported by the server.
DBSCHEMA_TABLES	Returns dimensions, measure groups, or schema rowsets exposed as tables.
DISCOVER_CALC_DEPENDENCY	Returns information about the calculation dependency for an object that is specified in a Tabular database or in a DAX query that is executed against a Tabular database.
DISCOVER_COMMAND_OBJECTS	Provides resource usage and activity information about the objects in use by the referenced command.
DISCOVER_COMMANDS	Provides resource usage and activity information about the currently executing or last executed commands in the opened connections on the server.
DISCOVER_CONNECTIONS	Provides resource usage and activity information about the currently opened connections on the server.
DISCOVER_CSDL_METADATA	Returns information about database metadata for in-memory databases.
DISCOVER_DATASOURCES	Returns a list of the data sources that are available on the server.
DISCOVER_DB_CONNECTIONS	Provides resource usage and activity information about the currently opened connections from the server to a database.
DISCOVER_DIMENSION_STAT	returns statistics on the specified dimension.

ROWSET	DESCRIPTION
DISCOVER_ENUMERATORS	Returns a list of names, data types, and enumeration values of enumerators supported by the XMLA Provider for a specific data source.
DISCOVER_INSTANCES	Describes the instances on the server.
DISCOVER_JOBS	Provides information about the active jobs executing on the server. A job is a part of a command that executes a specific task on behalf of the command.
DISCOVER_KEYWORDS (XMLA)	Returns information about keywords that are reserved by the XMLA server.
DISCOVER_LITERALS	Returns information about literals supported by the server.
DISCOVER_LOCATIONS	Returns information about the contents of a backup file.
DISCOVER_LOCKS	Provides information about the current standing locks on the server.
DISCOVER_MASTER_KEY	Returns the server's master encryption key.
DISCOVER_MEMORYGRANT	Returns a list of internal memory quota grants that are taken by jobs that are currently running on the server.
DISCOVER_MEMORYUSAGE	Returns the DISCOVER_MEMORYUSAGE statistics for various objects allocated by the server.
DISCOVER_OBJECT_ACTIVITY	Provides resource usage per object since the start of the service.
DISCOVER_OBJECT_MEMORY_USAGE	Returns the DISCOVER_MEMORYUSAGE statistics for various objects allocated by the server.
DISCOVER_PARTITION_DIMENSION_STAT	Returns statistics on the dimension that is associated with a partition.
DISCOVER_PARTITION_STAT	Returns statistics on aggregations in a particular partition.
DISCOVER_PERFORMANCE_COUNTERS	Returns the value of one or more specified performance counters.
DISCOVER_PROPERTIES	Returns a list of information and values about the properties that are supported by the server for the specified data source.
DISCOVER_RING_BUFFERS	Returns information about the current XEvent ring buffers on the server.
DISCOVER_SCHEMA_ROWSETS	Returns the names, restrictions, description, and other information for all Discover requests.
DISCOVER_SESSIONS	Provides resource usage and activity information about the currently opened sessions on the server.

ROWSET	DESCRIPTION
DISCOVER_STORAGE_TABLE_COLUMN_SEGMENTS	Returns information about the column segments used for storing data for in-memory tables.
DISCOVER_STORAGE_TABLE_COLUMNS	Contains information about the columns used for representing the columns of an in-memory table.
DISCOVER_STORAGE_TABLES	Returns statistics about in-memory tables available to the server.
DISCOVER_TRACE_COLUMNS	
DISCOVER_TRACE_DEFINITION_PROVIDERINFO	Contains the DISCOVER_TRACE_COLUMNS schema rowset.
DISCOVER_TRACE_EVENT_CATEGORIES	Contains the DISCOVER_TRACE_EVENT_CATEGORIES schema rowset.
DISCOVER_TRACES	Contains the DISCOVER_TRACES schema rowset.
DISCOVER_TRANSACTIONS	Returns the current set of pending transactions on the system.
DISCOVER_XEVENT_TRACE_DEFINITION	Provides information about the XEvent traces that are currently active on the server.
DISCOVER_XEVENT_PACKAGES	Provides information about the XEvent packages that are described on the server.
DISCOVER_XEVENT_OBJECTS	Provides information about the XEvent objects that are described on the server.
DISCOVER_XEVENT_OBJECT_COLUMNS	Provides information about the schema of XEvent objects that are described on the server.
DISCOVER_XEVENT_SESSIONS	Provides information about the current XEvent sessions on the server.
DISCOVER_XEVENT_SESSION_TARGETS	Provides information about the current XEvent session targets on the server.
DISCOVER_XML_METADATA	Returns a rowset with one row and one column.
DMSCHEMA_MINING_COLUMNS	Describes the individual columns of all described data mining models that are deployed on the server.
DMSCHEMA_MINING_FUNCTIONS	Describes the data mining functions that are supported by the data mining algorithms that are available on a server that is running Analysis Services.
DMSCHEMA_MINING_MODEL_CONTENT	Enables the client application to browse the content of a trained data mining model.
DMSCHEMA_MINING_MODEL_CONTENT_PMML	Returns the XML structure of the mining model. The format of the XML string follows the PMML 2.1 standard.

ROWSET	DESCRIPTION
DMSCHEMA_MINING_MODEL_XML	Returns the XML structure of the mining model. The format of the XML string follows the PMML 2.1 standard.
DMSCHEMA_MINING_MODELS	Enumerates the data mining models that are deployed on the server.
DMSCHEMA_MINING_SERVICE_PARAMETERS	Provides a list of parameters that can be used to configure the behavior of each data mining algorithm that is installed on the server.
DMSCHEMA_MINING_SERVICES	Provides information about each data mining algorithm that the server supports.
DMSCHEMA_MINING_STRUCTURE_COLUMNS	Describes the individual columns of all mining structures that are deployed on the server.
DMSCHEMA_MINING_STRUCTURES	Enumerates information about the mining structures in the current catalog.
MDSCHEMA_ACTIONS	Describes the actions that can be available to the client application.
MDSCHEMA_CUBES	Describes the structure of cubes within a database. Perspectives are also returned in this schema.
MDSCHEMA_DIMENSIONS	Describes the dimensions within a database.
MDSCHEMA_FUNCTIONS	Returns information about the functions that are currently available for use in the DAX and MDX languages.
MDSCHEMA_HIERARCHIES	Describes each hierarchy within a particular dimension.
MDSCHEMA_INPUT_DATASOURCES	Describes the data source objects described within the database.
MDSCHEMA_KPIS	Describes the KPIs within a database.
MDSCHEMA_LEVELS	Describes each level within a particular hierarchy.
MDSCHEMA_MEASUREGROUP_DIMENSIONS	Enumerates the dimensions of measure groups.
MDSCHEMA_MEASUREGROUPS	Describes the measure groups within a database.
MDSCHEMA_MEASURES	Describes each measure.
MDSCHEMA_MEMBERS	Describes the members within a database.
MDSCHEMA_PROPERTIES	Describes the properties of members and cell properties.
MDSCHEMA_SETS	Describes any sets that are currently described in a database, including session-scoped sets.

NOTE

STORAGES DMVs do not have a schema rowset described in the protocol.

Performance Counters (SSAS)

5/16/2018 • 17 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Using Performance Monitor, you can monitor the performance of a Microsoft SQL Server Analysis Services (SSAS) instance by using performance counters.

Performance Monitor is a Microsoft Management Control (MMC) snap-in that tracks resource usage. You can start this MMC snap-in by typing in **PerfMon** at the command prompt or from Control Panel by clicking **Administrative Tools**, then **Performance Monitor**. Performance Monitor lets you track server and process performance and activity by using predefined objects and counters, and monitor events by using user-defined counters. Performance Monitor collects counts instead of data about the events, for example, memory usage, number of active transactions, or CPU activity. You can also set thresholds on specific counters to generate alerts that notify operators.

Performance Monitor can monitor remote and local instances of Analysis Services or SQL Server. For more information, see Using Performance Monitor.

To see the description of any counter that can be used with SQL Server Analysis Services, in Performance Monitor, open the **Add Counters** dialog box, select a performance object, and then click **Show Description**. The most important counters are CPU usage, memory usage, disk IO rate. It is recommended you start with these important counters, and move to more detailed counters when you have a better idea of what else could be improved through monitoring. For more information about which counters to include, see the SQL Server 2008 R2 Operations Guide.

Counters are grouped so you can more easily find related counters.

Counters by Groups

GROUP	DESCRIPTION
Cache	Statistics related to the Analysis Services aggregation cache.
Connection	Statistics related to Microsoft Analysis Services connections.
Data Mining Prediction	Statistics related to processing data mining models processing.
Data Mining Model Processing	Statistics related to creating predictions from data mining models.
Locks	Statistics related to Microsoft Analysis Services internal server locks.
MDX	Statistics related to Microsoft Analysis Services MDX calculations.
Memory	Statistics related to Microsoft Analysis Services internal server memory.

GROUP	DESCRIPTION
Proactive Caching	Statistics related to Microsoft Analysis Services Proactive Caching.
Processing Aggregations	Statistics related to processing of aggregations in MOLAP data files.
Processing Indexes	Statistics related to processing of indexes for MOLAP data files.
Processing	Statistics related to processing of data.
Storage Engine Query	Statistics related to Microsoft Analysis Services storage engine queries.
Threads	Statistics related to Microsoft Analysis Services threads.

Cache

Statistics related to the Microsoft Analysis Services aggregation cache.

COUNTER	DESCRIPTION
Current KB	Current memory used by the aggregation cache, in KB.
KB added/sec	Rate of memory added to the cache, KB/sec.
Current entries	Current number of cache entries.
Inserts/sec	Rate of insertions into the cache. The rate is tracked per partition per cube per database.
Evictions/sec	Rate of evictions from the cache. This is per partition per cube per database. Evictions are typically due to background cleaner.
Total inserts	Insertions into the cache. The rate is tracked per partition per cube per database.
Total evictions	Evictions from the cache. Evictions are tracked per partition per cube per database. Evictions are typically due to background cleaner.
Direct hits/sec	Rate of cache direct hits. A cache hit indicates that queries were answered from an existing cache entry.
Misses/sec	Rate of cache misses.
Lookups/sec	Rate of cache lookups.
Total direct hits	Total count of direct cache hits. A direct cache hit indicates that queries were answered from existing cache entries.
Total misses	Total count of cache misses.

COUNTER	DESCRIPTION
Total lookups	Total number of lookups into the cache.
Direct hit ratio	Ratio of cache direct hits to cache lookups, for the period between counter values.
Total filtered iterator cache hits	Total number of cache hits that returned an indexed iterator over the filtered results.
Total filtered iterator cache misses	Total number of cache hits that were unable to build an indexed iterator over the filtered results and had to build a new cache with the filtered results.

Connection

Statistics related to Microsoft Analysis Services connections.

COUNTER	DESCRIPTION
Current connections	Current number of client connections established.
Requests/sec	Rate of connection requests. These are arrivals.
Total requests	Total connection requests. These are arrivals.
Successes/sec	Rate of successful connection completions.
Total successes	Total successful connections.
Failures/sec	Rate of connection failures.
Total failures	Total failed connection attempts.
Current user sessions	Current number of user sessions established.

Data Mining Model Processing

Statistics related to Microsoft Analysis Services Data Mining model processing.

COUNTER	DESCRIPTION
Cases/sec	Rate at which cases are processed.
Current models processing	Current number of models being processed.

Data Mining Prediction

Statistics related to Microsoft Analysis Services Data Mining prediction.

COUNTER	DESCRIPTION
Concurrent DM queries	Current number of data mining queries being actively worked on.

COUNTER	DESCRIPTION
Predictions/sec	Number of predictions generated in data mining queries
Rows/sec	Number of rows handled during a data mining prediction query.
Queries/sec	Number of data mining queries that were handled.
Total Queries	Total data mining queries received by the server.
Total Rows	Total rows returned by data mining queries.
Total Predictions	Total data mining prediction queries received by the server.

Locks

Statistics related to Microsoft Analysis Services internal server locks.

COUNTER	DESCRIPTION
Current latch waits	Current number of threads waiting for a latch. These are latch requests that could not be given immediate grants and are in a wait state.
Latch waits/sec	Rate of latch requests that could not be granted immediately and had to wait before being granted.
Current locks	Current number of locked objects.
Current lock waits	Current number of clients waiting for a lock.
Lock requests/sec	Number of lock requests per second.
Lock grants/sec	Number of lock grants per second.
Lock waits/sec	Number of lock waits per second. These are lock requests that could not be given immediate lock grants and were put in a wait state.
Lock denials/sec	Rate of lock denials.
Unlock requests/sec	Number of unlock requests per second.
Total deadlocks detected	Total number of deadlocks detected.

MDX

Statistics related to Microsoft Analysis Services MDX Calculations.

COUNTER	DESCRIPTION
Number of calculation covers	Total number of evaluation nodes built by MDX execution plans, including active and cached.

COUNTER	DESCRIPTION
Current number of evaluation nodes	Current (approximate) number of evaluation nodes built by MDX execution plans, including active and cached.
Number of Storage Engine evaluation nodes	Total number of Storage Engine evaluation nodes built by MDX execution plans.
Number of cell-by-cell evaluation nodes	Total number of cell-by-cell evaluation nodes built by MDX execution plans.
Number of bulk-mode evaluation nodes	Total number of bulk-mode evaluation nodes built by MDX execution plans.
Number of evaluation nodes that covered a single cell	Total number of evaluation nodes built by MDX execution plans that covered only one cell.
Number of evaluation nodes with calculations at the same granularity	Total number of evaluation nodes built by MDX execution plans for which the calculations were at the same granularity as the evaluation node.
Current number of cached evaluation nodes	Current (approximate) number of cached evaluation nodes built by MDX execution plans.
Number of cached Storage Engine evaluation nodes	Total number of cached Storage Engine evaluation nodes built by MDX execution plans
Number of cached bulk-mode evaluation nodes	Total number of cached bulk-mode evaluation nodes built by MDX execution plans.
Number of cached 'other' evaluation nodes	Total number of cached evaluation nodes built by MDX execution plans that are neither Storage Engine nor Bulk-mode.
Number of evictions of evaluation nodes	Total number of cache evictions of evaluation nodes due to collisions.
Number of hash index hits in the cache of evaluation nodes	Total number of hits in the cache of evaluation nodes that were satisfied by the hash index.
Number of cell-by-cell hits in the cache of evaluation nodes	Total number of cell-by-cell hits in the cache of evaluation nodes.
Number of cell-by-cell misses in the cache of evaluation nodes	Total number of cell-by-cell misses in the cache of evaluation nodes.
Number of subcube hits in the cache of evaluation nodes	Total number of subcube hits in the cache of evaluation nodes.
Number of subcube misses in the cache of evaluation nodes	Total number of subcube misses in the cache of evaluation nodes.
Total Sonar subcubes	Total number of subcubes that the query optimizer generated.
Total cells calculated	Total number of cell properties calculated.

COUNTER	DESCRIPTION
Total recomputes	Total number of cells recomputed due to error.
Total flat cache inserts	Total number of cell values inserted into flat calculation cache.
Total calculation cache registered	Total number of calculation caches registered.
Total NON EMPTY	Total number of times a NON EMPTY algorithm was used.
Total NON EMPTY unoptimized	Total number of times an unoptimized NON EMPTY algorithm was used.
Total NON EMPTY for calculated members	Total number of times a NON EMPTY algorithm looped over calculated members.
Total Autoexist	Total number of times Autoexist was performed.
Total EXISTING	Total number of times an EXISTING set operation was performed.

Memory

Statistics related to Microsoft Analysis Services internal server memory.

COUNTER	DESCRIPTION
Page Pool 64 Alloc KB	Memory borrowed from system, in KB. This memory is given away to other parts of the server.
Page Pool 64 Lookaside KB	Current memory in 64KB lookaside list, in KB. (Memory pages ready to be used.)
Page Pool 8 Alloc KB	Memory borrowed from 64KB page pool, in KB. This memory is given away to other parts of the server.
Page Pool 8 Lookaside KB	Current memory in 8KB lookaside list, in KB. (Memory pages ready to be used.)
Page Pool 1 Alloc KB	Memory borrowed from 64KB page pool, in KB. This memory is given away to other parts of the server.
Page Pool 1 Lookaside KB	Current memory in 8KB lookaside list, in KB. (Memory pages ready to be used.)
Cleaner Current Price	Current price of memory, $/byte/time, normalized to 1000.
Cleaner Balance/sec	Rate of balance+shrink operations.
Cleaner Memory shrunk KB/sec	Rate of shrinking, in KB/sec.
Cleaner Memory shrinkable KB	Amount of memory, in KB, subject to purging by the background cleaner.

COUNTER	DESCRIPTION
Cleaner Memory nonshrinkable KB	Amount of memory, in KB, not subject to purging by the background cleaner.
Cleaner Memory KB	Amount of memory, in KB, known to the background cleaner. (Cleaner memory shrinkable + Cleaner memory nonshrinkable.)
Memory Usage KB	Memory usage of the server process as used in calculating cleaner memory price. Equal to counter Process\PrivateBytes plus the size of memory-mapped data, ignoring any memory which was mapped or allocated by the xVelocity in-memory analytics engine (VertiPaq) in excess of the xVelocity engine Memory Limit.
Memory Limit Low KB	Low memory limit, from configuration file.
Memory Limit High KB	High memory limit, from configuration file.
AggCacheKB	Current memory allocated to aggregation cache, in KB.
Quota KB	Current memory quota, in KB. Memory quota is also known as a memory grant or memory reservation.
Quota Blocked	Current number of quota requests that are blocked until other memory quotas are freed.
Filestore KB	Current memory allocated to filestore (file cache), in KB.
Filestore Page Faults/sec	Filestore page fault rate.
Filestore Reads/sec	Filestore pages read/sec.
Filestore KB Reads/sec	Filestore KB read/sec.
Filestore Writes/sec	Filestore pages written/sec. The writes are asynchronous.
Filestore KB Write/sec	Filestore KB written/sec. The writes are asynchronous.
Filestore IO Errors/sec	Filestore IO Error rate.
Filestore IO Errors	Filestore IO Errors total.
Filestore Clock Pages Examined/sec	Rate of background cleaner examining pages for eviction consideration.
Filestore Clock Pages HaveRef/sec	Rate of background cleaner examining pages that have a current reference count (are currently in use).
Filestore Clock Pages Valid/sec	Rate of background cleaner examining pages that are valid candidates for eviction.
Filestore Memory Pinned KB	Current filestore memory pinned, in KB.

COUNTER	DESCRIPTION
In-memory Dimension Property File KB	Current size of in-memory dimension property file, in KB.
In-memory Dimension Property File KB/sec	Rate of writes to in-memory dimension property file, in KB.
Potential In-memory Dimension Property File KB	Potential size of in-memory dimension property file, in KB.
Dimension Property Files	Number of dimension property files.
In-memory Dimension Index (Hash) File KB	Size of current in-memory dimension index (hash) file, in KB.
In-memory Dimension Index (Hash) File KB/sec	Rate of writes to in-memory dimension index (hash) file, in KB.
Potential In-memory Dimension Index (Hash) File KB	Potential size of in-memory dimension index (hash) file, in KB.
Dimension Index (Hash) Files	Number of dimension index (hash) files.
In-memory Dimension String File KB	Current size of in-memory dimension string file, in KB.
In-memory Dimension String File KB/sec	Rate of writes to in-memory dimension string file, in KB.
Potential In-memory Dimension String File KB	Potential size of in-memory dimension string file, in KB.
Dimension String Files	Number of dimension string files.
In-memory Map File KB	Current size of in-memory map file, in KB.
In-memory Map File KB/sec	Rate of writes to in-memory map file, in KB.
Potential In-memory Map File KB	Potential size of in-memory map file, in KB.
Map Files	Number of map files.
In-memory Aggregation Map File KB	Current size of in-memory aggregation map file, in KB.
In-memory Aggregation Map File KB/sec	Rate of writes to in-memory aggregation map file, in KB.
Potential In-memory Aggregation Map File KB	Size of potential in-memory aggregation map file, in KB.
Aggregation Map Files	Number of aggregation map files.
In-memory Fact Data File KB	Size of current in-memory fact data file, in KB.
In-memory Fact Data File KB/sec	Rates of writes to in-memory fact data file KB rate.
Potential In-memory Fact Data File KB	Size of potential in-memory fact data file, in KB.
Fact Data Files	Number of fact data files.
In-memory Fact String File KB	Size of current in-memory fact string file, in KB.

COUNTER	DESCRIPTION
In-memory Fact String File KB/sec	Rate of writes to in-memory fact string file, in KB.
Potential In-memory Fact String File KB	Size of potential in-memory fact string file, in KB.
Fact String Files	Number of fact string files.
In-memory Fact Aggregation File KB	Current size of in-memory fact aggregation file, in KB.
In-memory Fact Aggregation File KB/sec	Rate of writes to in-memory fact aggregation file, in KB.
Potential In-memory Fact Aggregation File KB	Size of potential in-memory fact aggregation file, in KB.
Fact Aggregation Files	Number of fact aggregation files.
In-memory Other File KB	Size of current in-memory other file, in KB.
In-memory Other File KB/sec	Rate of writes to in-memory other file, in KB.
Potential In-memory Other File KB	Size of potential in-memory other file, in KB.
Other Files	Number of other files.
VertiPaq Paged KB	Kilobytes of paged memory in use for in-memory data.
VertiPaq Nonpaged KB	Kilobytes of memory locked in the working set for use by the in-memory engine.
VertiPaq Memory-Mapped KB	Kilobytes of pageable memory in use for in-memory data.
Memory Limit Hard KB	Hard memory limit, from configuration file.
Memory Limit VertiPaq KB	In-memory limit, from configuration file.

Proactive Caching

Statistics related to Microsoft Analysis Services Proactive Caching.

COUNTER	DESCRIPTION
Notifications/sec	Rate of notifications from relational database.
Processing Cancellations/sec	Rate of processing cancellations caused by notifications.
Proactive Caching Begin/sec	Rate of proactive caching begin.
Proactive Caching Completion/sec	Rate of proactive caching completion.

Processing Aggregations

Statistics related to Microsoft Analysis Services processing of aggregations in MOLAP data files.

COUNTER	DESCRIPTION
Current partitions	Current number of partitions being processed.
Total partitions	Total number of partitions processed (successfully or otherwise).
Memory size rows	Size of current aggregations in memory. This count is an estimate.
Memory size bytes	Size of current aggregations in memory. This count is an estimate.
Rows merged/sec	Rate of rows merged or inserted into an aggregation.
Rows created/sec	Rate of aggregation rows created.
Temp file rows written/sec	Rate of writing rows to a temporary file. Temporary files are written when aggregations exceed memory limits.
Temp file bytes written/sec	Rate of writing bytes to a temporary file. Temporary files are written when aggregations exceed memory limits.

Processing Indexes

Statistics related to Microsoft Analysis Services processing of indexes for MOLAP data files.

COUNTER	DESCRIPTION
Current partitions	Current number of partitions being processed.
Total partitions	Total number of partitions processed (successfully or otherwise).
Rows/sec	Rate of rows from MOLAP files used to create indexes.
Total rows	Total rows from MOLAP files used to create indexes.

Processing

Statistics related to Microsoft Analysis Services processing of data.

COUNTER	DESCRIPTION
Rows read/sec	Rate of rows read from all relational databases.
Total rows read	Count of rows read from all relational databases.
Rows converted/sec	Rate of rows converted during processing.
Total rows converted	Count of rows converted during processing.
Rows written/sec	Rate of rows written during processing.
Total rows written	Count of rows written during processing.

Storage Engine Query

Statistics related to Microsoft Analysis Services storage engine queries.

COUNTER	DESCRIPTION
Current measure group queries	Current number of measure group queries being actively worked on.
Measure group queries/sec	Rate of measure group queries
Total measure group queries	Total number of queries to measure group.
Current dimension queries	Current number of dimension queries being actively worked on.
Dimension queries/sec	Rate of dimension queries
Total dimension queries.	Total number of dimension queries.
Queries answered/sec	Rate of queries answered.
Total queries answered	Total number of queries answered.
Bytes sent/sec	Rate of bytes sent by server to clients, in response to queries.
Total bytes sent	Total bytes sent by server to clients, in response to queries.
Rows sent/sec	Rate of rows sent by server to clients.
Total rows sent	Total rows sent by server to clients.
Queries from cache direct/sec	Rate of queries answered from cache directly.
Queries from cache filtered/sec	Rate of queries answered by filtering existing cache entry.
Queries from file/sec	Rate of queries answered from files.
Total queries from cache direct	Total number of queries derived directly from cache. Note that this is per partition.
Total queries from cache filtered	Total queries answered by filtering existing cache entries.
Total queries from file	Total number of queries answered from files.
Map reads/sec	Number of logical read operations using the Map file.
Map bytes/sec	Bytes read from the Map file.
Data reads/sec	Number of logical read operations using the Data file.
Data bytes/sec	Bytes read from the Data file.

COUNTER	DESCRIPTION
Avg time/query	Average time per query, in milliseconds. Response time based on queries answered since the last counter measurement.
Network round trips/sec	Rate of network round trips. This includes all client/server communication.
Total network round trips	Total network round trips. This includes all client/server communication.
Flat cache lookups/sec	Rate of flat cache lookups. This includes global, session, and query scope flat caches.
Flat cache hits/sec	Rate of flat cache hits. This includes global, session, and query scope flat caches.
Calculation cache lookups/sec	Rate of calculation cache lookups. This includes global, session, and query scope calculation caches.
Calculation cache hits/sec	Rate of calculation cache hits. This includes global, session, and query scope calculation caches.
Persisted cache lookups/sec	Rate of persisted cache lookups. Persisted caches are created by the MDX script CACHE statement.
Persisted cache hits/sec	Rate of persisted cache hits. Persisted caches are created by the MDX script CACHE statement.
Dimension cache lookups/sec	Rate of dimension cache lookups.
Dimension cache hits/sec	Rate of dimension cache hits.
Measure group cache lookups/sec	Rate of measure group cache lookups.
Measure group cache hits/sec	Rate of measure group cache hits.
Aggregation lookups/sec	Rate of aggregation lookups.
Aggregation hits/sec	Rate of aggregation hits.

Threads

Statistics related to Microsoft Analysis Services threads.

COUNTER	DESCRIPTION
Short parsing idle threads	Number of idle threads in the short parsing thread pool.
Short parsing busy threads	Number of busy threads in the short parsing thread pool.
Short parsing job queue length	Number of jobs in the queue of the short parsing thread pool.
Short parsing job rate	Rate of jobs through the short parsing thread pool.

COUNTER	DESCRIPTION
Long parsing idle threads	Number of idle threads in the long parsing thread pool.
Long parsing busy threads	Number of busy threads in the long parsing thread pool.
Long parsing job queue length	Number of jobs in the queue of the long parsing thread pool.
Long parsing job rate	Rate of jobs through the long parsing thread pool.
Query pool idle threads	Number of idle threads in the query thread pool.
Query pool busy threads	Number of busy threads in the query thread pool.
Query pool job queue length	Number of jobs in the queue of the query thread pool.
Query pool job rate	Rate of jobs through the query thread pool.
Processing pool idle non-I/O threads	Number of idle threads in the processing thread pool dedicated to non-I/O jobs.
Processing pool busy non-I/O threads	Number of threads running non-I/O jobs in the processing thread pool.
Processing pool job queue length	Number of non-I/O jobs in the queue of the processing thread pool.
Processing pool job rate	Rate of non-I/O jobs through the processing thread pool.
Processing pool idle I/O job threads	Number of idle threads for I/O jobs in the processing thread pool.
Processing pool busy I/O job threads	Number of threads running I/O jobs in the processing thread pool.
Processing pool I/O job queue length	Number of I/O jobs in the queue of the processing thread pool.
Processing pool I/O job completion rate	Rate of I/O jobs through the processing thread pool.

Log operations in Analysis Services

11/28/2018 • 7 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

An Analysis Services instance will log server notifications, errors, and warnings to the msmdsrv.log file - one for each instance you install. Administrators refer to this log for insights into routine and extraordinary events alike. In recent releases, logging has been enhanced to include more information. Log records now include product version and edition information, as well as processor, memory, connectivity, and blocking events. You can review the entire change list at Logging improvements.

Besides the built-in logging feature, many administrators and developers also use tools provided by the Analysis Services community to collect data about server operations, such as **ASTrace**. See Microsoft SQL Server Community Samples: Analysis Services for the download links.

This topic contains the following sections:

- Location and types of logs
- General information on log file configuration settings
- MSMDSRV service log file
- Query logs
- Mini dump (.mdmp) files
- Tips and best practices

Location and types of logs

Analysis Services provides the logs described below.

FILE NAME OR LOCATION	TYPE	USED FOR	ON BY DEFAULT
Msmdsrv.log	Error log	Routine monitoring and basic troubleshooting	Yes
OlapQueryLog table in a relational database	Query log	Collect inputs for the Usage Optimization Wizard	No
SQLDmp<guid>.mdmp files	Crashes and exceptions	Deep troubleshooting	No

We highly recommend the following link for additional information resources not covered in this topic: Initial data collection tips from Microsoft Support.

General information on log file configuration settings

You can find sections for each log in the msmdsrv.ini server configuration file, located in the \Program Files\Microsoft SQL Server\MSAS13.MSSQLSERVER\OLAP\Config folder. See Server Properties in Analysis Services for instructions on editing the file.

Where possible, we suggest that you set logging properties in the server properties page of Management Studio. Although in some cases, you must edit the msmdsrv.ini file directly to configure settings that are not visible in the

administrative tools.

```
- <Log>
    <File>msmdsrv.log</File>
    <FileBufferSize>0</FileBufferSize>
    <MessageLogs>File;Console;System</MessageLogs>
  + <ErrorLog>
  + <QueryLog>
  + <Exception>
  + <Trace>
  + <FlightRecorder>
</Log>
```

MSMDSRV service log file

Analysis Services logs server operations to the msmdsrv.log file, one per instance, located at \program files\Microsoft SQL Server\<instance>\Olap\Log.

This log file is emptied at each service restart. In previous releases, administrators would sometimes restart the service for the sole purpose of flushing the log file before it could grow so large as to become unusable. This is no longer necessary. Configuration settings, introduced in SQL Server 2012 SP2 and later, give you control over the size of the log file and its history:

- **MaxFileSizeMB** specifies a maximum log file size in megabytes. The default is 256. A valid replacement value must be a positive integer. When **MaxFileSizeMB** is reached, Analysis Services renames the current file as msmdsrv{current timestamp}.log file, and starts a new msmdsrv.log file.
- **MaxNumberFiles** specifies retention of older log files. The default is 0 (disabled). You can change it to a positive integer to keep versions of the log file. When **MaxNumberFiles** is reached, Analysis Services deletes the file with the oldest timestamp in its name.

To use these settings, do the following:

1. Open msmdsrv.ini in NotePad.
2. Copy the following two lines:

   ```
   <MaxFileSizeMB>256</MaxFileSizeMB>
   <MaxNumberOfLogFiles>5</MaxNumberOfLogFiles>
   ```

3. Paste the two lines into the Log section of msmdsrv.ini, below the filename for msmdsrv.log. Both settings must be added manually. There are no placeholders for them in the msmdsrv.ini file.

   The changed configuration file should look like the following:

   ```
   <Log>
   <File>msmdsrv.log</File>
   <MaxFileSizeMB>256</MaxFileSizeMB>
   <MaxNumberOfLogFiles>5</MaxNumberOfLogFiles>
   <FileBufferSize>0</FileBufferSize>
   ```

4. Edit the values if those provided differ from what you want.
5. Save the file.
6. Restart the service.

Query logs

The query log is a bit of a misnomer in that it does not log the MDX or DAX query activity of your users. Instead, it collects data about queries generated by Analysis Services, which is subsequently used as data input in the Usage Based Optimization Wizard. The data collected in the query log is not intended for direct analysis. Specifically, the datasets are described in bit arrays, with a zero or a one indicating the parts of dataset is included in the query. Again, this data is meant for the wizard.

For query monitoring and troubleshooting, many developers and administrators use a community tool, **ASTrace**, to monitor queries. You can also use SQL Server Profiler, xEvents, or an Analysis Services trace.

When should you use the query log? We recommend enabling the query log as part of a query performance tuning exercise that includes the Usage Based Optimization Wizard. The query log does not exist until you enable the feature, create the data structures to support it, and set properties used by Analysis Services to locate and populate the log.

To enable the query log, follow these steps:

1. Create a SQL Server relational database to store the query log.
2. Grant the Analysis Services service account sufficient permissions on the database. The account needs permission to create a table, write to the table, and read from the table.
3. In SQL Server Management Studio, right-click **Analysis Services** | **Properties** | **General**, set **CreateQueryLogTable** to true.
4. Optionally, change **QueryLogSampling** or **QueryLogTableName** if you want to sample queries at a different rate, or use a different name for the table.

The query log table will not be created until you have run enough MDX queries to meet the sampling requirements. For example, if you keep the default value of 10, you must run at least 10 queries before the table will be created.

Query log settings are server wide. The settings you specify will be used by all databases running on this server.

Name	Value	Current Value	Default Value	Restart
DataMining \ MaxConcurrentPredictionQueries	0	0	0	
Feature \ ComUdfEnabled	false	false	false	
Feature \ LinkFromOtherInstanceEnabled	false	false	false	
Feature \ LinkInsideInstanceEnabled	true	true	true	
Feature \ LinkToOtherInstanceEnabled	false	false	false	
ForceCommitTimeout	30000	30000	30000	
Log \ FlightRecorder \ Enabled	true	true	true	
Log \ QueryLog \ CreateQueryLogTable	true	true	false	
Log \ QueryLog \ QueryLogConnectionString	Provider=SQLNCLI11.1;Data	Provider=SQLN...		
Log \ QueryLog \ QueryLogSampling	10	10	10	
Log \ QueryLog \ QueryLogTableName	OlapQueryLog	OlapQueryLog	OlapQueryLog	
LogDir	C:\Program Files\Microsoft...	C:\Program File...		yes

After the configuration settings are specified, run an MDX query multiple times. If sampling is set to 10, run the query 11 times.Verify the table is created. In Management Studio, connect to the relational database engine, open the database folder, open the **Tables** folder, and verify that **OlapQueryLog** exists. If you do not immediately see the table, refresh the folder to pick up any changes to its contents.

Allow the query log to accumulate sufficient data for the Usage Based Optimization Wizard. If query volumes are cyclical, capture enough traffic to have a representative set of data. See Usage Based Optimization Wizard for instructions on how to run the wizard.

See Configuring the Analysis Services Query Log to learn more about query log configuration. Although the paper is quite old, query log configuration has not changed in recent releases and the information it contains still applies.

Mini dump (.mdmp) files

Dump files capture data used for analyzing extraordinary events. Analysis Services automatically generates mini dumps (.mdmp) in response to a server crash, exception, and some configuration errors. The feature is enabled, but does not send crash reports automatically.

Crash reports are configured through the Exception section in the Msmdsrv.ini file. These settings control memory dump file generation. The following snippet shows the default values:

```
<Exception>
<CreateAndSendCrashReports>1</CreateAndSendCrashReports>
<CrashReportsFolder/>
<SQLDumperFlagsOn>0x0</SQLDumperFlagsOn>
<SQLDumperFlagsOff>0x0</SQLDumperFlagsOff>
<MiniDumpFlagsOn>0x0</MiniDumpFlagsOn>
<MiniDumpFlagsOff>0x0</MiniDumpFlagsOff>
<MinidumpErrorList>0xC1000000, 0xC1000001, 0xC102003F, 0xC1360054, 0xC1360055</MinidumpErrorList>
<ExceptionHandlingMode>0</ExceptionHandlingMode>
<CriticalErrorHandling>1</CriticalErrorHandling>
<MaxExceptions>500</MaxExceptions>
<MaxDuplicateDumps>1</MaxDuplicateDumps>
</Exception>
```

Configure Crash Reports

Unless otherwise directed by Microsoft Support, most administrators use the default settings. This older KB article is still used to provide instruction on how to configure dump files: How to configure Analysis Services to generate memory dump files.

The configuration setting most likely to be modified is the **CreateAndSendCrashReports** setting used to determine whether a memory dump file will be generated.

VALUE	DESCRIPTION
0	Turns off the memory dump file. All other settings under the Exception section are ignored.
1	(Default) Enables, but does not send, the memory dump file.
2	Enables and automatically sends an error report to Microsoft.

CrashReportsFolder is the location of the dump files. By default, an .mdmp file and associated log records can be found in the \Olap\Log folder.

SQLDumperFlagsOn is used to generate a full dump. By default, full dumps are not enabled. You can set this property to **0x34**.

The following links provide more background:

- Looking Deeper into SQL Server using Minidumps
- How to create a user mode dump file
- How to use the Sqldumper.exe utility to generate a dump file in SQL Server

Tips and best practices

This section is a recap of the tips mentioned throughout this article.

- Configure the msmdsrv.log file to control the size and number of msmdsrv log file. The settings are not enabled by default, so be sure to add them as post-installation step. See MSMDSRV service log file in this

topic.

- Review this blog post from Microsoft Customer Support to learn what resources they use to get information about server operations: Initial Data Collection
- Use ASTrace2012, rather than a query log, to find out who is querying cubes. The query log is typically used to provide input into the Usage Based Optimization Wizard, and the data it captures is not easy to read or interpret. ASTrace2012 is a community tool, widely used, that captures query operations. See Microsoft SQL Server Community Samples: Analysis Services.

See Also

Analysis Services Instance Management
Introduction to Monitoring Analysis Services with SQL Server Profiler
Server Properties in Analysis Services

Clear the Analysis Services Caches

11/15/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services caches data to boost query performance. This topic provides recommendations for using the XMLA ClearCache command to clear caches that were created in response to an MDX query. The effects of running ClearCache vary depending on whether you are using a tabular or multidimensional model.

When to clear the cache for multidimensional models

For multidimensional databases, Analysis Services builds caches in the formula engine when evaluating a calculation, and in the storage engine for the results of dimension queries and measure group queries. Measure group queries occur when the formula engine needs measure data for a cell coordinate or subcube. Dimension queries occur when querying unnatural hierarchies and when applying autoexists.

Clearing the cache is recommended when conducting performance testing. By clearing the cache between test runs, you ensure that caching does not skew any test results that measure the impact of a query design change.

When to clear the cache for tabular models

Tabular models are generally stored in memory, where aggregations and other calculations are performed at the time a query is executed. As such, the ClearCache command has a limited effect on tabular models. For a tabular model, data may be added to the Analysis Services caches if MDX queries are run against it. In particular, DAX measures referenced by MDX and autoexists operations can cache results in the formula cache and dimension cache respectively. Note however, that unnatural hierarchies and measure group queries do not cache results in the storage engine. Additionally, it is important to recognize that DAX queries do not cache results in the formula and storage engine. To the extent that caches exist as a result of MDX queries, running ClearCache against a tabular model will invalidate any cached data from the system.

Running ClearCache will also clear in-memory caches in the xVelocity in-memory analytics engine (VertiPaq). The xVelocity engine maintains a small set of cached results. Running ClearCache will invalidate these caches in the xVelocity engine.

Finally, running ClearCache will also remove residual data that is left in memory when a tabular model is reconfigured for **DirectQuery** mode. This is particularly important if the model contains sensitive data that is subject to tight controls. In this case, running ClearCache is a precautionary action that you can take to ensure that sensitive data exists only where you expect it to be. Clearing the cache manually is necessary if you are using Management Studio to deploy the model and change the query mode. In contrast, using SQL Server Data Tools to specify **DirectQuery** on the model and partitions will automatically clear the cache when you switch the model to use that query mode.

Compared with recommendations for clearing multidimensional model caches during performance testing, there is no broad recommendation for clearing tabular model caches. If you are not managing the deployment of a tabular model that contains sensitive data, there is no specific administrative task that calls for clearing the cache.

Clear the cache for Analysis Services models

To clear the cache, use XMLA and SQL Server Management Studio. You can clear the cache at the database, cube, dimension or table, or measure group level. The following steps for clearing the cache at the database level apply to both multidimensional models and tabular models.

> **NOTE**
> Rigorous performance testing might require a more comprehensive approach to clearing the cache. For instructions on how to flush Analysis Services and file system caches, see the section on clearing caches in the SQL Server 2008 R2 Analysis Services Operations Guide.

For both multidimensional and tabular models, clearing some of these caches can be a two-step process that consists of invalidating the cache when ClearCache executes, followed by emptying the cache when the next query is received. A reduction in memory consumption will be evident only after the cache is actually emptied.

Clearing the cache requires that you provide an object identifier to the **ClearCache** statement in an XMLA query. The first step in this topic explains how to get an object identifier.

Step 1: Get the object identifier

1. In Management Studio, right-click an object, select **Properties**, and copy the value from the ID property in the **Properties** pane. This approach works for the database, cube, dimension, or table.

2. To get the measure group ID, right-click the measure group and select **Script Measure Group As**. Choose either **Create** or **Alter**, and send the query to a window. The ID of the measure group will be visible in the object definition. Copy the ID of the object definition.

Step 2: Run the query

1. In Management Studio, right-click a database, point to **New Query**, and select **XMLA**.

2. Copy the following code example into the XMLA query window. Change **DatabaseID** to the ID of the database on the current connection.

```
<ClearCache xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
  <Object>
    <DatabaseID> Adventure Works DW Multidimensional</DatabaseID>
  </Object>
</ClearCache>
```

   Alternatively, you can specify a path of a child object, such as a measure group, to clear the cache for just that object.

```
<ClearCache xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
  <Object>
    <DatabaseID>Adventure Works DW Multidimensional</DatabaseID>
        <CubeID>Adventure Works</CubeID>
        <MeasureGroupID>Fact Currency Rate</MeasureGroupID>
  </Object>
</ClearCache>
```

3. Press F5 to execute the query. You should see the following result:

```
<return xmlns="urn:schemas-microsoft-com:xml-analysis">
  <root xmlns="urn:schemas-microsoft-com:xml-analysis:empty" />
</return>
```

Database Consistency Checker (DBCC) for Analysis Services

11/27/2018 • 19 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

DBCC provides on-demand database validation for Multidimensional and Tabular databases on an Analysis Services instance. You can execute DBCC in an MDX or XMLA query window in SQL Server Management Studio (SSMS) and trace the DBCC output in either SQL Server Profiler or xEvent sessions in SSMS.
The command takes an object definition and returns either an empty result set or detailed error information if the object is corrupted. In this article, you'll learn how to run the command, interpret results, and address any problems that arise.

For Tabular databases, consistency checks performed by DBCC are equivalent to the built-in validation that occurs automatically every time you reload, synchronize, or restore a database. In contrast, consistency checks for Multidimensional databases happen only when you run DBCC on demand.

The range of validation checks will vary by mode, with Tabular databases subject to a broader range of checks. Characteristics of a DBCC workload also varies by server mode. Check operations on Multidimensional databases involve reading data from disk, constructing temporary indexes for comparison against actual indexes -- all of which takes significantly longer to complete.

Command syntax for DBCC uses the object metadata specific to the type of database you are checking:

- Multidimensional + pre-SQL Server 2016 Tabular 1100 or 1103 compatibility level databases are described in Multidimensional modeling constructs like **cubeID**, **measuregroupID**, and **partitionID**.
- Metadata for new Tabular model databases at compatibility level 1200 and higher consist of descriptors like **TableName** and **PartitionName**.

DBCC for Analysis Services will execute on any Analysis Services database at any compatibility level, as long as the database is running on a SQL Server 2016 instance. Just make sure you're using the right command syntax for each database type.

> **NOTE**
> If you're familiar with DBCC (Transact-SQL), you'll quickly notice that the DBCC in Analysis Services has a much narrower scope. DBCC in Analysis Services is a single command that reports exclusively on data corruption across the database or on individual objects. If you have other tasks in mind, such as collecting information, try using AMO PowerShell or XMLA scripts instead.

Permission requirements

You must be an Analysis Services database or server administrator (a member of the server role) to run the command. See Grant database permissions (Analysis Services) or Grant server admin rights to an Analysis Services instance for instructions.

Command syntax

Tabular databases at the 1200 and higher compatibility levels use tabular metadata for object definitions. The complete DBCC syntax for a tabular database created at a SQL Server 2016 functional level is illustrated in the

following example.

Key differences between the two syntaxes include a newer XMLA namespace, no <Object> element, and no <Model> element (there is still only one model per database).

```
<DBCC xmlns="http://schemas.microsoft.com/analysisservices/2014/engine">
     <DatabaseID>MyTabular1200DB_7811b5d8-c407-4203-8793-12e16c3d1b9b</DatabaseID>
     <TableName>FactSales</TableName>
     <PartitionName>FactSales 4</PartitionName>
</DBCC>
```

You can omit lower-level objects, such as table or partition names, to check the entire schema.

You can get object names and DatabaseID from Management Studio, through the property page of each object.

Command syntax for Multidimensional and Tabular 110x databases

DBCC uses identical syntax for multidimensional as well as tabular 1100 and 1103 databases. You can run DBCC against specific database objects, including the entire database. See Object Element (XMLA) for more information about the object definition.

```
<DBCC xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
     <Object>
          <DatabaseID>AdventureWorksDW2014Multidimensional-EE</DatabaseID>
          <CubeID>Adventure Works</CubeID>
          <MeasureGroupID>Fact Internet Sales 1</MeasureGroupID>
          <PartitionID>Internet_Sales_2006</PartitionID>
     </Object>
</DBCC>
```

To run DBCC on objects higher up the object chain, delete any lower-level object ID elements you don't need:

```
<DBCC xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
     <Object>
          <DatabaseID>AdventureWorksDW2014Multidimensional-EE</DatabaseID>
          <CubeID>Adventure Works</CubeID>
     </Object>
</DBCC>
```

For tabular 110x databases, the object definition syntax is modeled after the syntax of Process command (specifically, in how tables are mapped to dimensions and measure groups).

- **CubeID** maps to the model ID, which is **Model**.
- **MeasureGroupID** maps to a table ID.
- **PartitionID** maps to a partition ID.

Usage

In SQL Server Management Studio, you can invoke DBCC using either an MDX or XMLA query window. Additionally, you can use either SQL Server 2017 Profiler or Analysis Services xEvents to view DBCC output. Note that SSAS DBCC messages are not reported to the Windows application event log or the msmdsrv.log file.

DBCC checks for physical data corruption, as well as logical data corruption that occur when orphaned members exist in a segment. A database must be processed before you can run DBCC. It skips remote, empty, or

unprocessed partitions.

The command runs in a read transaction, and can thus be kicked out by force commit timeout. Partition checks are run in parallel.

A service restart might be required to pick up any corruption errors that have occurred since the last service restart. Reconnecting to the server is not enough to pick up the changes.

Run DBCC commands in Management Studio

For ad hoc queries, open an MDX or XMLA query window in SQL Server Management Studio. To do this, right-click the database | **New Query** | **XMLA**) to run the command and read the output.

The Results tab will indicate an empty result set (as shown in the screenshot) if no problems were detected.

The Messages tab provides detail information but is not always reliable for smaller databases. Status messages are sometimes trimmed, indicating the command completed, but without the status check messages on each object. A typical message report might look similar to the one shown below.

Messages reported from DBCC for the cube validation check

```
Executing the query ...
READS, 0
READ_KB, 0
WRITES, 0
WRITE_KB, 0
CPU_TIME_MS, 0
ROWS_SCANNED, 0
ROWS_RETURNED, 0

<DBCC xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
<Object>
<DatabaseID>AdventureWorksDW2014Multidimensional-EE</DatabaseID>
<CubeID>Adventure Works</CubeID>
</Object>
</DBCC>
Started checking segment indexes for the 'Internet_Sales_2011' partition.
Started checking segment indexes for the 'Internet_Sales_2012' partition.
Finished checking segment indexes for the 'Internet_Sales_2011' partition.
Started checking segment indexes for the 'Internet_Sales_2013' partition.
Finished checking segment indexes for the 'Internet_Sales_2012' partition.
Started checking segment indexes for the 'Internet_Sales_2014' partition.
Started checking segment indexes for the 'Internet_Orders_2011' partition.
Finished checking segment indexes for the 'Internet_Sales_2014' partition.
Started checking segment indexes for the 'Internet_Orders_2012' partition.
Started checking segment indexes for the 'Internet_Orders_2013' partition.
Finished checking segment indexes for the 'Internet_Orders_2012' partition.
...
Run complete
```

Output when running DBCC against an earlier version of Analysis Services

DBCC is only supported on databases running on a SQL Server 2017 instance. Running the command on older systems will return this error.

```
Executing the query ...
The DBCC element at line 7, column 87 (namespace http://schemas.microsoft.com/analysisservices/2003/engine)
cannot appear under Envelope/Body/Execute/Command.
Execution complete
```

Trace DBCC output in SQL Server Profiler 2016

You can view DBCC output in a Profiler trace that includes Progress Reports events (Progress Report Begin, Progress Report Current, Progress Report End, and Progress Report Error).

1. Start a trace. See Use SQL Server Profiler to Monitor Analysis Services for help on how to use SQL Server Profiler with Analysis Services.
2. Choose **Command Begin** and **Command End** plus any or all of the **Progress Report** Events.
3. Run the DBCC command in Management Studio in either an XMLA or MDX query window, using the syntax provided in a previous section.
4. In SQL Server Profiler, DBCC activity is indicated through **Command** events having an event subclass of DBCC:

EventClass	EventSubclass	ActivityID	ApplicationName
Command Begin	12 - Batch		
Command Begin	32 - DBCC	374D0361...	Microsoft SQL Server Management Studio - Query
Command End	32 - DBCC	374D0361...	Microsoft SQL Server Management Studio - Query

Event code 32 is DBCC execution.

Event code 64 is a DBCC progress report on individual objects.

Event code 63 is a segment check for multidimensional objects.

For both event subclases, review **TextData** values for messages returned by DBCC.

Status messages start with "Checking consistency of <object>" , "Started checking <object>", or "Finished checking <object>".

> **NOTE**
> In CTP 3.0, objects are identified by internal names. For example, a Categories hierarchy is articulated as H$Categories-<objectID>. Internal names should be replaced by user friendly names in an upcoming CTP.

Error messages are listed below.

Trace DBCC output in an xEvent session in SSMS

Extended events sessions can use both profiler events or xEvents. Refer to the previous section for guidance on adding **Command** and **Progress Report** events.

1. Start a session by right-clicking a database > **Management** >**Extended Events** > **Sessions** > **New Session**. See Monitor Analysis Services with SQL Server Extended Events for more information.
2. Choose any or all of the **Progress Report** Events for the Profiler event category or **RequestProgress** events for the PureXevent category.
3. Run the DBCC command in Management Studio in either an XMLA or MDX query window, using the syntax provided in a previous section.
4. In SSMS, refresh the Sessions folder. Right-click the session name > **Watch Live Data**.
5. Review TextData values for messages returned by DBCC. TextData is a property of an event field and shows status and error messages returned by the event.

   Status messages start with "Checking consistency of <object>" , "Started checking <object>", or "Finished checking <object>".

   Error messages are listed below.

Reference: Consistency checks and errors for Multidimensional databases

For multidimensional databases, only partition indexes are validated. During execution, DBCC builds a temporary index for each partition and compares it with the persisted index on disk. Building a temporary index requires reading all data from the partition data on disk and then holding the temporary index in memory for comparison. Given the additional workload, your server might experience significant disk IO and memory consumption while running a DBCC execution.

Detection of Multidimensional index corruption includes the following checks. Errors in this table appear in xEvent or Profiler traces for failures at the object level.

Object	**DBCC check description**	**Error on failure**

Partition Index	Check segment statistics and indexes. Compares the ID of each member in the temporary partition index against the partition statistics stored on disk. If a member is found in the temporary index with a data ID value outside the range stored for the partition index statistics on disk, then the statistics for the index are considered corrupt.	The partition segment statistics are corrupted.
Partition Index	Validates metadata. Verifies that each member in the temporary index can be found in the index header file for the segment on disk.	The partition segment is corrupted.
Partition Index	Scan segments to look for physical corruptions. Reads the index file on disk for each member in the temporary index and verifies that the size of the index records match, and that the same data pages are flagged as having records for the current member.	The partition segment is corrupted.

Reference: Consistency checks and errors for Tabular databases

The following table is list of all consistency checks performed on tabular objects, alongside errors that are raised if the check indicates corruption. Errors in this table appear in xEvent or Profiler traces for failures at the object level.

Object	**DBCC check description**	**Error on failure**
Database	Checks count of tables in the database. A value less than zero indicates corruption.	There is corruption in the storage layer. The collection of tables in the '%{parent/}' database is corrupt.
Database	Checks internal structure used to track Referential Integrity and throw s an error if the size is incorrect.	Database files failed to pass consistency checks.
Table	Checks internal value used to determine if table is a Dimension or Fact table. A value that falls outside the known range indicates corruption.	Database consistency checks (DBCC) failed while checking the table statistics.
Table	Checks that the number of partitions in the segment map for the table matches the number of partitions defined for the table.	There is corruption in the storage layer. The collection of partitions in the '%{parent/}' table is corrupt.

Table	If a tabular database was created or imported from PowerPivot for Excel 2010 and has a partition count greater than one, an error will be raised, as partition support was added in later versions and this would indicate corruption.	Database consistency checks (DBCC) failed while checking the segment map.
Partition	Verifies for each partition that the number segments of data and the record count for each segment of data in the segment matches the values stored in the index for the segment.	Database consistency checks (DBCC) failed while checking the segment map.
Partition	Raise an error if the number of total records, segments, or records per segment is not valid (less than zero), or the number of segments doesn't match the calculated number of segments needed based on total record count.	Database consistency checks (DBCC) failed while checking the segment map.
Relationship	Raise an error if the structure used to store data about the relationship contains no records or if the name of the table used in the relationship is empty.	Database consistency checks (DBCC) failed while checking relationships.
Relationship	Verify that the name of the primary table, primary column, foreign table, and foreign column are set and that the columns and tables involved in the relationship can be accessed. Verify that the column types involved are valid and that the index of Primary Key-Foreign Key values results in a valid lookup structure.	Database consistency checks (DBCC) failed while checking relationships.
Hierarchy	Raise an error if the sort order for the hierarchy isn't a recognized value.	Database consistency checks (DBCC) failed while checking the '%{hier/}' hierarchy.
Hierarchy	The checks performed on the hierarchy depend on the internal type of hierarchy mapping scheme used. All hierarchies are checked for correct processed state, that the hierarchy store exists, and that where applicable, data structures used for a data-ID-to-hierarchy-position conversion exists. Assuming all these checks pass, the hierarchy structure is walked to verify that each position in the hierarchy points to the correct member. If any of these tests fail, an error is raised.	Database consistency checks (DBCC) failed while checking the '%{hier/}' hierarchy.

User defined Hierarchy	Checks that the hierarchy level names are set. If the hierarchy has been processed, check that the internal hierarchy data store has the correct format. Verify that the internal hierarchy store doesn't contain any invalid data values. If the hierarchy is marked as unprocessed, confirm that this state applies to old data structures and that all levels of the hierarchy are marked as empty.	Database consistency checks (DBCC) failed while checking the '%{hier/}' hierarchy.
Column	Raise an error if the encoding used for the column is not set to a known value.	Database consistency checks (DBCC) failed while checking the column statistics.
Column	Check whether the column was compressed by the in-memory engine or not.	Database consistency checks (DBCC) failed while checking the column statistics.
Column	Check the compression type on the column for known values.	Database consistency checks (DBCC) failed while checking the column statistics.
Column	When the column "tokenization" is not set to a known value, raise an error.	Database consistency checks (DBCC) failed while checking the column statistics.
Column	If the id range stored for a columns data dictionary does not match the number of values in the data dictionary or is outside the allowed range, raise an error.	Database consistency checks (DBCC) failed while checking the data dictionary.
Column	Check that the number of data segments for a column matches the number of data segments for the table to which it belongs.	There is corruption in the storage layer. The collection of segments in the '%{parent/}' column is corrupt.
Column	Check that the number of Partitions for a data column matches the number of partitions for the data segment map for the column.	Database consistency checks (DBCC) failed while checking the segment map.
Column	Verify that the number of records in a column segment matches the record count stored in the index for that column segment.	There is corruption in the storage layer. The collection of segments in the '%{parent/}' column is corrupt.
Column	If a column has no segment statistics, raise an error.	Database consistency checks (DBCC) failed while checking the segment statistics.

Column	If a column has no compression information or segment storage, raise an error.	Database files failed to pass consistency checks.
Column	Report an error if segment statistics for a column don't match the actual column values for Minimum Data ID, Maximum Data ID, number of Distinct values, number of rows, or presence of NULL values.	Database consistency checks (DBCC) failed while checking the segment statistics.
ColumnSegment	If the minimum data ID or maximum data ID is less than the system reserved value for NULL, mark the column segment information as corrupt.	Database consistency checks (DBCC) failed while checking the segment statistics.
ColumnSegment	If there are no rows for this segment, the minimum and maximum data values for the column should be set to the system reserved value for NULL. If the value is not null, raise an error.	Database consistency checks (DBCC) failed while checking the segment statistics.
ColumnSegment	If the column has rows and at least one non-null value, checkthat the minimum and maximum data id for the column is greater than the system reserved value for NULL.	Database consistency checks (DBCC) failed while checking the segment statistics.
Internal	Verify that the store tokenization hint is set and that if the store is processed, there are valid pointers for internal tables. If the store is not processed, verify all the pointers are null. If not, return a generic DBCC error.	Database files failed to pass consistency checks.
DBCC Database	Raise an error if the database schema has no tables or one or more tables cannot be accessed.	There is corruption in the storage layer. The collection of tables in the '%{parent/}' database is corrupt.
DBCC Database	Raise an error if a table is marked as temporary or has an unknown type.	A bad table type was encountered.
DBCC Database	Raise an error if the number of relationships for a table has a negative value, or if any table has a relationship defined and a corresponding relationship structure cannot be found.	There is corruption in the storage layer. The collection of relationships in the '%{parent/}' table is corrupt.
DBCC Database	If the compatibility level for the database is 1050 (SQL Server 2008 R2/PowerPivot v1.0) and the number of relationships exceeds the number of tables in the model, mark the database as corrupted.	Database files failed to pass consistency checks.

DBCC Table	For the table under validation, check if the number of columns is less than zero and raise an error if true. An error also occurs if the column store for a column in the table is NULL.	There is corruption in the storage layer. The collection of columns in the '%{parent/}' table is corrupt.
DBCC Partition	Checks the table that the partition being validated belongs to, and if the number of columns for the table is less than zero, it indicates the Columns collection is corrupt for the table. An error will also occur if the column store for a column in the table is NULL.	There is corruption in the storage layer. The collection of columns in the '%{parent/}' table is corrupt.
DBCC Partition	Loops through each column for the selected partition and checks that each segment for the partition has a valid link to a column segment structure. If any segment has a NULL link, the partition is considered corrupt.	There is corruption in the storage layer. The collection of segments in the '%{parent/}' column is corrupt.
Column	Returns an error if the column type is not valid.	A bad segment type was encountered.
Column	Returns an error if any column has a negative count for the number of segments in a column, or if the pointer to the column segment structure for a segment has a NULL link.	There is corruption in the storage layer. The collection of segments in the '%{parent/}' column is corrupt.
DBCC Command	The DBCC Command will report multiple status messages as it proceeds through the DBCC operation. It will report a status message before starting that includes the database, table, or column name of the object, and again after finishing each object check.	Checking consistency of the <objectname> <objecttype>. Phase: pre-check. Checking consistency of the <objectname> <objecttype>. Phase: post-check.

Common resolutions for error conditions

The following errors appear in SQL Server Management Studio or in msmdsrv.log files. These errors appear when one or more checks fail to pass. Depending on the error, the recommended resolution is to either reprocess an object, delete and redeploy a solution, or restore the database.

ERROR	ISSUE	RESOLUTION
Errors in the metadata manager The object reference '<objectID>' is not valid. It does not match the structure of the metadata class hierarchy.	malformed command	Check the command syntax. Most likely, you included a lower level object without specifying one or more of its parent objects.

ERROR	ISSUE	RESOLUTION
Errors in the metadata manager Either the <object> with the ID of '<objectID>' does not exist in the <parentobject> with the ID of '<parentobjectID>', or the user does not have permissions to access the object.	Index corruption (multidimensional)	Reprocess the object and any dependent objects.
Error occurred during consistency check of the partition An error occurred while checking consistency of the <partition-name> partition of the <measure-group-name> measure group for the <cube-name> cube from the <database-name> database. Please re-process the partition or indexes in order to fix the corruption.	Index corruption (multidimensional)	Reprocess the object and any dependent objects.
Partition segment statistics corrupted	Index corruption (multidimensional)	Reprocess the object and any dependent objects.
Partition segment is corrupted	Metadata corruption (multidimensional or tabular)	Delete and redeploy the project, or restore from a backup and reprocess. See How to handle corruption in Analysis Services databases (blog) for instructions.
Table metadata corruption Table <table-name> metadata file is corrupt. The main table is not found under DataFileList node.	Metadata corruption (tabular only)	Delete and redeploy the project, or restore from a backup and reprocess. See How to handle corruption in Analysis Services databases (blog) for instructions.
Corruption in storage layer Corruption in storage layer: collection of <type-name> in <parent-name> <parent-type> is corrupt.	Metadata corruption (tabular only)	Delete and redeploy the project, or restore from a backup and reprocess. See How to handle corruption in Analysis Services databases (blog) for instructions.
System table is missing System table <table-name> is missing.	Object corruption (tabular only)	Reprocess the object and any dependent objects
Table statistics are corrupt Statistics of table System table <table-name> is missing.	Metadata corruption (tabular only)	Delete and redeploy the project, or restore from a backup and reprocess. See How to handle corruption in Analysis Services databases (blog) for instructions.

Disable automatic consistency checks on database load operations

through the msmdsrv.ini configuration file

Although its not recommended, you can disable the built-in database consistency checks that occur automatically on database load events (on tabular databases only). To do this, you will need to modify a configuration setting in the msmdsrv.ini file:

```
<ConfigurationSettings>
     <Vertipaq />
          <DisableConsistencyChecks />
```

This setting is not present in the configuration file and must be added manually.

Valid values are as follows:

- **-2** (default) DBCC is enabled. If the server can logically resolve the error with a high degree of certainty, a fix will be applied automatically. Otherwise, an error will be logged.
- **-1** DBCC is partially enabled. It is enabled for RESTORE and on pre-commit validations that check database state at the end of a transaction.
- **0** DBCC is partially enabled. Database consistency checks are performed during RESTORE, IMAGELOAD, LOCALCUBELOAD, and ATTACH
  operations.
- **1** DBCC is disabled. Data integrity checks are disabled, however deserialization checks will still occur.

> **NOTE**
>
> This setting has no impact on DBCC when running the command on demand.

See Also

Process Database, Table, or Partition (Analysis Services)
Processing a multidimensional model (Analysis Services)
Compatibility Level for Tabular models in Analysis Services
Server Properties in Analysis Services

Connect to Analysis Services

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use the information in this section to learn about connection string properties, client libraries used for connections, which authentication methods are supported by Analysis Services, and how to set up or clear connections before taking a server offline.

To learn about connecting to Azure Analysis Services, see Connect to a server.

Analysis Services connections

Analysis Services uses TCP as the network protocol and XML for Analysis (XMLA) as a communication protocol. At the lowest level, all of the client libraries provided with Analysis Services implementing XMLA-over-TCP. Although it is possible to build applications based on raw XMLA, most applications and application developers use client libraries to take advantage of the object models and coding efficiencies that they provide. For client connections to Analysis Services, you can use IIS as an intermediary connection if you cannot use TCP across the stack. One advantage of using HTTP access via IIS is the ability to connect from applications that pass credentials on the connection string.

Any discussion involving connectivity typically includes authentication. In contrast with other SQL Server features, Analysis Services uses Windows credentials exclusively. You cannot use SQL Server database authentication, claims authentication, forms-based authentication, or digest on the backend connection to Analysis Services. More about authentication is provided in this section.

Connection Tasks

LINK	TASK DESCRIPTION
Connect from client applications (Analysis Services)	If you are new to Analysis Services, read this topic to get started with the tools and applications most often used with Analysis Services.
Connection String Properties (Analysis Services)	Analysis Services includes numerous server and database properties, allowing you to customize a connection for a specific application, independent of how the instance or database is configured.
Authentication methodologies supported by Analysis Services	This topic is a brief introduction to the authentication methods used by Analysis Services.
Configure Analysis Services for Kerberos constrained delegation	Many business intelligence solutions require impersonation to ensure that only authorized data is returned to each user. In this topic, learn the requirements for using impersonation. This topic also explains the steps for configuring Analysis Services for Kerberos constrained delegation.

LINK	TASK DESCRIPTION
SPN registration for an Analysis Services instance	Kerberos authentication requires a valid Service Principle Name (SPN) for services that impersonate or delegate user identities in multi-server solutions. Use the information in this topic to learn the construction and steps for SPN registration for Analysis Services.
Configure HTTP Access to Analysis Services on Internet Information Services (IIS) 8.0	Basic authentication or cross-domain boundaries are two important reasons for configuring Analysis Services for HTTP access.
Data providers used for Analysis Services connections	Analysis Services provides three client libraries for accessing server operations or Analysis Services data. This topic offers a brief introduction to ADOMD.NET, Analysis Services Management Objects (AMO), and the Analysis Services OLE DB provider (MSOLAP).
Disconnect Users and Sessions on Analysis Services Server	Clear existing connections and sessions before taking a server offline or conducting baseline performance tests.

See Also

Post-install Configuration (Analysis Services)
Server Properties in Analysis Services

Connect from client applications (Analysis Services)

5/16/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If you are new to Analysis Services, use the information in this topic to connect to an existing instance of Analysis Services using common tools and applications. This topic also explains how to connect under different user identities for testing purposes.

- Connect using SQL Server Management Studio (SSMS)
- Connect using Excel
- Connect using SQL Server Data Tools
- Test connections

Connection string reference documentation is provided separately. For more information, see Connection String Properties (Analysis Services).

Successful connections depend on a valid port configuration and appropriate user permissions. Click the following links to learn more about each requirement.

- Configure the Windows Firewall to Allow Analysis Services Access
- Authorizing access to objects and operations (Analysis Services)

Connect using SQL Server Management Studio (SSMS)

Connect to Analysis Services in SSMS to manage server instances and databases interactively. You can also run XMLA or MDX queries to perform administrative tasks or retrieve data. In contrast with other tools and applications that only load databases when a query is sent, SSMS loads all databases when you connect to the server, assuming you have permission to view the database. This means that if you have numerous tabular databases on the server, all are loaded into system memory when you connect using SSMS.

You can test permissions by running SSMS under a specific user identity and then connect to Analysis Services as that user.

Hold-down the Shift key and right-click the **SQL Server Management Studio** shortcut to access the **Run as different user** option.

1. Start SQL Server Management Studio. In the **Connect to Server** dialog box, select the Analysis Services server type.

2. In the Login tab, enter the server name by typing the name of the computer on which the server is running. You can specify the server using its network name or a fully-qualified domain name.

   For a named instance, the server name must be specified in this format: servername\instance name. An example of this naming convention might be ADV-SRV062\Finance for a server that has a network name of ADV-SRV062, where Analysis Services was installed as a named instance entitled Finance.

   For servers deployed in a failover cluster, connect using the network name of the SSAS cluster. This name is specified during SQL Server setup, as **SQL Server Network Name**. Note that if you installed SSAS as a named instance onto a Windows Server Failover Cluster (WSFC), you never add the instance name on the connection. This practice is unique to SSAS; in contrast, a named instance of a clustered relational database

engine does include the instance name. For example, if you installed both SSAS and the database engine as named instance (Contoso-Accounting) with a SQL Server Network Name of SQL-CLU, you would connect to SSAS using "SQL-CLU" and to the database engine as "SQL-CLU\Contoso-Accounting". See How to Cluster SQL Server Analysis Services for more information and examples.

For servers deployed in a network load balanced cluster, connect using the virtual server name of the NLB.

3. Authentication is always Windows authentication, and the user identity is always the Windows user who is connecting via Management Studio.

   In order for the connection to succeed, you must have permission to access the server or a database on the server. Most tasks that you want to perform in Management Studio require administrative permissions. Ensure that the account you are connecting with is a member of the Server Administrator role. For more information, see Grant server admin rights to an Analysis Services instance.

4. Click **Connection Properties** to specify a particular database, set timeout values, or encryption options. Optional connection information includes connection properties used for the current connection only.

5. Click **Additional Connection Parameters** tab to set connection properties not available in the Connect to Server dialog box. For example, you might type `Roles=Reader` in the text box.

   Connecting through a role with less permission lets you test database behaviors when that role is in effect.

   ```
   Provider=MSOLAP; Data Source=SERVERNAME; Initial Catalog=AdventureWorks2012; Roles=READER
   ```

Connect using Excel

Microsoft Excel is often used for analyzing business data. As part of an Excel installation, Office installs the Analysis Services OLE DB provider (MSOLAP DLL), ADOMD.NET, and other data providers so that you can more readily use the data on your network servers. If you are using a newer version of Analysis Services with an older version of Excel, you most likely need to install newer data providers on each workstation that connects to Analysis Services. See Data providers used for Analysis Services connections for more information.

When you set up a connection to an Analysis Services cube or tabular model database, Excel saves the connection information in .odc file for future use. The connection is made in security context of the current Windows user. The user account must have read permissions on the database in order for the connection to succeed.

When using Analysis Services data in an Excel workbook, connections are held for the duration of a query request. This is why you are likely to see lots of connections for each session, held for very short periods of time, when monitoring a query workload from Excel.

You can test permissions by starting Excel under a specific user identity.

Hold-down the Shift key and right-click the **Excel** shortcut to access the **Run as different user** option.

1. On the Data tab in Excel, click **From Other Sources**, and then click **From Analysis Services**. Enter the server name, and then select a cube or perspective to query.

   For servers deployed in a load-balanced cluster, use the virtual server name assigned to the cluster.

2. When setting up a connection in Excel, on the last page of the Data Connection Wizard, you can specify authentication settings for Excel Services. These settings are used to set properties on the workbook should you upload it to a SharePoint server that has Excel Services. The settings are used in data refresh operations. Options include **Windows Authentication**, **Secure Store Service** (SSS), and **None**.

   Avoid using **None**. Analysis Services does not let you specify a user name and password on the connection string unless you are connecting to a server that has been configured for HTTP access. Similarly, do not

use SSS unless you already know that the SSS target application ID is mapped to a set of Windows user credentials that have user access to the Analysis Services databases. For most scenarios, using the default option of Windows authentication is the best choice for an Analysis Services connection from Excel.

For more information, see Connect to or import data from SQL Server Analysis Services.

Connect using SQL Server Data Tools

SQL Server Data Tools is used for building BI solutions, including Analysis Services models, Reporting Services reports, and SSIS packages. When building reports or packages, you might need to specify a connection to Analysis Services.

The following links explain how to connect to Analysis Services from a Report Server project or an Integration Services project:

- Analysis Services Connection Type for MDX (SSRS)
- Analysis Services Connection Manager

> **NOTE**
>
> When using SQL Server Data Tools to work on an existing Analysis Services project, remember that you can connect offline using a local or version controlled project, or connect in online mode to update Analysis Services objects while the database is running. For more information, see Connect in Online Mode to an Analysis Services Database. More commonly, connections from SQL Server Data Tools are in project mode, where changes are deployed to the database only when you explicitly deploy the project.

Test connections

You can use SQL Server Profiler to monitor connections to Analysis Services. The Audit Login and Audit Logout events provide evidence of a connection. The identity column indicates the security context under which the connection is made.

1. Start **SQL Server Profiler** on the Analysis Services instance and then start a new trace.
2. In Events Selection, verify that **Audit Login** and **Audit Logout** are checked in the Security Audit section.
3. Connect to Analysis Services via an application service (such as SharePoint or Reporting Services) from a remote client computer. The Audit Login event will show the identity of the user connecting to Analysis Services.

Connection errors are often traced to an incomplete or invalid server configuration. Always check server configuration first:

- Ping the server from a remote computer to ensure it allows remote connections.
- **Firewall rules on the server allow inbound connections from clients in the same domain**

  With the exception of Power Pivot for SharePoint, all connections to a remote server require that you have configured the firewall to allow access to the port that Analysis Services is listening on. If you are getting connection errors, verify that the port is accessible and that user permissions are granted to the appropriate databases.

  To test, use Excel or SSMS to on a remote computer, specifying the IP address and port used by the Analysis Services instance. If you can connection, the firewall rules are valid for the instance and the instance allows remote connections.

  Also, when using TCP/IP for the connection protocol, remember that Analysis Services requires client

connections originate from the same domain or a trusted domain. If connections flow across security boundaries, you will most likely need to configure HTTP access. For more information, see Configure HTTP Access to Analysis Services on Internet Information Services (IIS) 8.0.

- Can you connect using some tools but not others? The problem might be the wrong version of a client library. You can get client libraries from the SQL Server Feature Pack download page.

Resources that can help you resolve connection failures include the following:

Resolving Common Connectivity Issues in SQL Server 2005 Analysis Services Connectivity Scenarios. This document is a few years old, but the information and methodologies still apply.

See Also

Connect to Analysis Services
Authentication methodologies supported by Analysis Services
Impersonation
Create a Data Source (SSAS Multidimensional)

Connection String Properties (Analysis Services)

12/6/2018 • 17 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This topic describes connection string properties you might set in one of the designer or administration tools, or see in connection strings built by client applications that connect to and query Analysis Services data. As such, it covers just a subset of the available properties. The complete list includes numerous server and database properties, allowing you to customize a connection for a specific application, independent of how the instance or database is configured on the server.

Developers who build custom connection strings in application code should review the API documentation for ADOMD.NET client to view a more detailed list: ConnectionString

The properties described in this topic are used by the Analysis Services client libraries, ADOMD.NET, AMO, and the OLE DB provider for Analysis Services. The majority of connection string properties can be used with all three client libraries. Exceptions are called out in the description.

> **NOTE**
> When setting properties, if you inadvertently set the same property twice, the last one in the connection string is used.

For more information about how to specify an Analysis Services connection in existing Microsoft applications, see Connect from client applications (Analysis Services).

Connection parameters in common use

The following table describes those properties most often used when building a connection string.

PROPERTY	DESCRIPTION	EXAMPLE
Data Source or **DataSource**	Specifies the server instance. This property is required for all connections. Valid values include the network name or IP address of the server, local or localhost for local connections, a URL if the server is configured for HTTP or HTTPS access, or the name of a local cube (.cub) file. Valid value for Azure Analysis Services, `<protocol>://<region>/<servername>` where protocol is string asazure, region is the Uri where the server was created (for example, westus.asazure.windows.net) and servername is the name of your unique server within the region.	`Data source=asazure://westus.asazure.windows.net/my` `Data source=AW-SRV01` for the default instance and port (TCP 2383). `Data source=AW-SRV01$Finance:8081` for a named instance ($Finance) and fixed port. `Data source=AW-SRV01.corp.Adventure-Works.com` for a fully qualified domain name, assuming the default instance and port. `Data source=172.16.254.1` for an IP address of the server, bypassing DNS server lookup, useful for troubleshooting connection problems.
Initial Catalog or **Catalog**	Specifies the name of the Analysis Services database to connect to. The database must be deployed on Analysis Services, and you must have permission to connect to it. This property is optional for AMO connections, but required for ADOMD.NET.	`Initial catalog=AdventureWorks2016`

PROPERTY	DESCRIPTION	EXAMPLE
Provider	Valid values include MSOLAP.<version>, where <version> is either 4, 5, 6 or 7. - MSOLAP.4 released in SQL Server 2008 and again SQL Server 2008 R2 (filename is msolap100.dll for SQL Server 2008 and 2008 R2) - MSOLAP.5 released in SQL Server 2012 (filename is msolap110.dll) - MSOLAP.6 released in SQL Server 2014 (filename is msolap1200.dll) - MSOLAP.7 released in SQL Server 2016 (filename is msolap130.dll) This property is optional. By default, the client libraries read the current version of the OLE DB provider from the registry. You only need to set this property if you require a specific version of the data provider, for example to connect to a SQL Server 2012 instance. MSOLAP.4 was released in both SQL Server 2008 and SQL Server 2008 R2. The 2008 R2 version supports Power Pivot workbooks and sometimes needs to be installed manually on SharePoint servers. To distinguish between these versions, you must check the build number in the file properties of the provider: Go to Program files\Microsoft Analysis Services\AS OLEDB\10. Right-click msolap110.dll and select **Properties**. Click **Details**. View the file version information. The version should include 10.50.<buildnumber> for SQL Server 2008 R2. For more information, see Install the Analysis Services OLE DB Provider on SharePoint Servers and Data providers used for Analysis Services connections.	`Provider=MSOLAP.7` is used for connections that require the SQL Server 2016 version of the OLE DB provider for Analysis Services.
Cube	Cube name or perspective name. A database can contain multiple cubes and perspectives. When multiple targets are possible, include the cube or perspective name on the connection string.	`Cube=SalesPerspective` shows that you can use the Cube connection string property to specify either the name of a cube or the name of a perspective.

Authentication and Security

This section includes connection string properties related to authentication and encryption. Analysis Services uses Windows Authentication only, but you can set properties on the connection string to pass in a specific user name and password.

Properties are listed in alphabetical order.

PROPERTY	DESCRIPTION

PROPERTY	DESCRIPTION
EffectiveUserName	Use when an end user identity must be impersonated on the server. Specify the account in a domain\user format. To use this property, the caller must have administrative permissions in Analysis Services. For more information about using this property in an Excel workbook from SharePoint, see Use Analysis Services EffectiveUserName in SharePoint Server 2013. **EffectiveUserName** is used in a Power Pivot for SharePoint installation to capture usage information. The user identity is provided to the server so that events or errors that include user identity can be recorded in the log files. In the case of Power Pivot, it is not used for authorization purposes.
Encrypt Password	Specifies whether a local password is to be used to encrypt local cubes. Valid values are True or False. The default is False.
Encryption Password	The password used to decrypt an encrypted local cube. Default value is empty. This value must be explicitly set by the user.
Impersonation Level	Indicates the level of impersonation that the server is allowed to use when impersonating the client. Valid values include: - Anonymous. The client is anonymous to the server. The server process cannot obtain information about the client, nor can the client be impersonated. - Identify. The server process can get the client identity. The server can impersonate the client identity for authorization purposes but cannot access system objects as the client. - Impersonate. This is the default value. The client identity can be impersonated, but only when the connection is established, and not on every call. - Delegate. The server process can impersonate the client security context while acting on behalf of the client. The server process can also make outgoing calls to other servers while acting on behalf of the client.
Integrated Security	The Windows identity of the caller is used to connect to Analysis Services. Valid values are blank, SSPI, and BASIC. **Integrated Security**=**SSPI** is the default value for TCP connections, allowing NTLM, Kerberos, or Anonymous authentication. Blank is the default value for HTTP connections. When using **SSPI**, **ProtectionLevel** must be set to one of the following: **Connect**, **PktIntegrity**, **PktPrivacy**.
Persist Encrypted	Set this property when the client application requires the data source object to persist sensitive authentication information, such as a password, in encrypted form. By default, authentication information is not persisted.
Persist Security Info	Valid values are True and False. When set to True, security information, such as the user identity or password previously specified on the connection string, can be obtained from the connection after the connection is made. The default value is False.

PROPERTY	DESCRIPTION
Protection Level	Determines the security level used on the connection. Valid values are: - **None**. Unauthenticated or anonymous connections. Performs no authentication on data sent to the server. - **Connect**. Authenticated connections. Authenticates only when the client establishes a relationship with a server. - **Pkt Integrity**. Encrypted connections. Verifies that all data is received from the client and that it has not been changed in transit. - **Pkt Privacy**. Signed encryption, supported only for XMLA. Verifies that all data is received from the client, that it has not been changed in transit, and protects the privacy of the data by encrypting it. For more information, see Establishing Secure Connections in ADOMD.NET
Roles	Specify a comma-delimited list of predefined roles to connect to a server or database using permissions conveyed by that role. If this property is omitted, all roles are used, and the effective permissions are the combination of all roles. Setting the property to an empty value (for example, Roles=' ') the client connection has no role membership. An administrator using this property connects using the permissions conveyed by the role. Some commands might fail if the role does not provide sufficient permission.
SSPI	Explicitly specifies which security package to use for client authentication when **Integrated Security** is set to **SSPI**. SSPI supports multiple packages, but you can use this property to specify a particular package. Valid values are Negotiate, Kerberos, NTLM, and Anonymous User. If this property is not set, all packages will be available to the connection.
Use Encryption for Data	Encrypts data transmissions. Value values are True and False.
User ID=...; **Password**=	**User ID** and **Password** are used together. Analysis Services impersonates the user identity specified through these credentials. On an Analysis Services connection, putting credentials on the command line is used only when the server is configured for HTTP access, and you specified Basic authentication instead of integrated security on the IIS virtual directory. When connecting directly to the server, **UserID** and **Password** connection string params are ignored and the connection is made using the context of the logged on user. The user name and password must be the credentials of a Windows identity, either a local or a domain user account. Notice that **User ID** has an embedded space. Other aliases for this property include **UserName** (no space), and **UID**. Alias for **Password** is **PWD**.

Special-purpose parameters

This section describes the remainder of the connection string parameters. These are used to ensure specific connection behaviors required by an application.

Properties are listed in alphabetical order.

PROPERTY	DESCRIPTION

PROPERTY	DESCRIPTION
Application Name	Sets the name of the application associated with the connection. This value can be useful when monitoring tracing events, especially when you have several applications accessing the same databases. For example, adding Application Name='test' to a connection string causes 'test' to appear in a SQL Server Profiler trace, as shown in the following screenshot: Aliases for this property include **sspropinitAppName**, **AppName**. For more information, see Use Application Name parameter when connecting to SQL Server.
AutoSyncPeriod	Sets the frequency (in milliseconds) of client and server cache synchronization. ADOMD.NET provides client caching for frequently used objects that have minimal memory overhead. This helps reduce the number of round trips to the server. The default is 10000 milliseconds (or 10 seconds). When set to null or 0, automatic synchronization is turned off.
Character Encoding	Defines how characters are encoded on the request. Valid values are Default or UTF-8 (these are equivalent), and UTF-16
CommitTimeout	An XMLA property. Determines how long, in milliseconds, the commit phase of a currently running command waits before rolling back. When greater than 0, overrides the value of the corresponding CommitTimeout property in the server configuration.
CompareCaseSensitiveStringFlags	Adjusts case-sensitive string comparisons for a specified locale. For more information about setting this property, see CompareCaseSensitiveStringFlags Property.
Compression Level	If **TransportCompression** is XPRESS, you can set the compression level to control how much compression is used. Valid values are 0 through 9, with 0 having least compression, and 9 having the most compression. Increased compression slows performance. The default value is 0.
Connect Timeout	Determines the maximum amount of time (in seconds) the client attempts a connection before timing out. If a connection does not succeed within this period, the client quits trying to connect and generates an error.
DbpropMsmdRequestMemoryLimit	This property overrides the Memory\QueryMemoryLimit server property value for a connection. Specified in kilobytes.
MDX Compatibility	The purpose of this property is to ensure a consistent set of MDX behaviors for applications that issue MDX queries. Excel, which uses MDX queries to populate and calculate a PivotTable connected to Analysis Services, sets this property to 1, to ensure that placeholder members in ragged hierarchies are visible in a PivotTable. Valid values include 0, 1, 2. 0 and 1 expose placeholder members; 2 does not. If this is empty, 0 is assumed.
MDX Missing Member Mode=Error	Indicates whether missing members are ignored in MDX statements. Valid values are Default, Error, and Ignore. Default uses a server-defined value. Error generates an error when a member does not exist. Ignore specifies that missing values should be ignored.

PROPERTY	DESCRIPTION
Optimize Response	A bitmask indicating which of the following query response optimizations are enabled. - 0x01 Use the NormalTupleSet (this is the default) - 0x02 Use when slicers are empty
Packet Size	A network packet size (in bytes) between 512 and 32,767. The default network packet size is 4096.
Protocol Format	Sets the format of the XML sent to the server. Valid values are Default, XML, or Binary. The protocol is XMLA. You can specify that the XML be sent in compressed form (this is the default), as raw XML, or in a binary format. Binary format encodes XML elements and attributes, making them smaller. Compression is a proprietary format that further reduces the size of requests and responses. Compression and binary formats are used to speed up data transfer requests and responses. You must use a client library on the connection if using binary or compressed format. OLE DB provider can format requests and responses in binary or compressed format. AMO and ADOMD.NET format the requests as Text, but accept responses in binary or compressed format. This connection string property is equivalent to the **EnableBinaryXML** and **EnableCompression** server configuration settings.
Real Time Olap	Set this property to bypass caching, causing all partitions to actively listen for query notifications. By default, this property is not set.
Safety Options	Sets the safety level for user-defined functions and actions. Valid values are 0, 1, 2. In an Excel connection this property is Safety Options=2. Details about this option can be found in ConnectionString.
SQLQueryMode	Specifies whether SQL queries include calculations. Valid values are Data, Calculated, IncludeEmpty. Data means that no calculations are allowed. Calculated allows calculations. IncludeEmpty allows calculations and empty rows to be returned in the query result.
Timeout	Specifies how long (in seconds) the client library waits for a command to complete before generating an error.
Transport Compression	Defines how client and server communications are compressed, when compression is specified via the **Protocol Format** property. Valid values are Default, None, Compressed and **gzip**. Default is no compression for TCP, or **gzip** for HTTP. None indicates that no compression is used. Compressed uses XPRESS compression (SQL Server 2008 and later). **gzip** is only valid for HTTP connections, where the HTTP request includes Accept-Encoding=gzip.
UseExistingFile	Used when connecting to a local cube. This property specifies whether the local cube is overwritten. Valid values are True or False. If set to True, the cube file must exist. The existing file will be the target of the connection. If set to False, the cube file is overwritten.

PROPERTY	DESCRIPTION
VisualMode	Set this property to control how members are aggregated when dimension security is applied. For cube data that everyone is allowed to see, aggregating all of the members makes sense because all of the values that contribute to the total are visible. However, if you filter or restrict dimensions based on user identity, showing a total based on all the members (combining both restricted and allowed values into a single total) might be confusing or show more information than should be revealed. To specify how members are aggregated when dimension security is applied, you can set this property to True to use only allowed values in the aggregation, or False to exclude restricted values from the total. When set on the connection string, this value applies to the cube or perspective level. Within a model, you can control visual totals at a more granular level. Valid values are 0, 1, and 2. - 0 is the default. Currently, the default behavior is equivalent to 2, where aggregations include values that are hidden from the user. - 1 excludes hidden values from the total. This is the default for Excel. - 2 includes hidden values in the total. This is the default value on the server. Aliases for this property include **Visual Total** or **Default MDX Visual Mode**.

Reserved for future use

The following properties are allowed on a connection string, but are not operational in current releases of Analysis Services.

- Authenticated User
- Cache Authentication
- Cache Mode (Use of this property was investigated in earlier releases. Although you might find blog posts recommending its usage, you should avoid setting this property unless instructed by Microsoft Support).
- Cache Policy
- Cache Ratio
- Cache Ratio2
- Dynamic Debug Limit
- Debug Mode
- Mode
- SQLCompatibility
- Use Formula Cache

Example connection strings

This section shows the connection string that you'll most likely use when setting up an Analysis Services connection in commonly used applications.

Generic connection string

You might use a connection string like this one if you are configuring a connection from Reporting Services.

```
Data source=<servername>; initial catalog=<databasename>
```

Connection string in Excel

The default ADOMD.NET connection string in Excel specifies the data provider, server, database name, Windows integrated security. The MDX Compatibility level is always set to 1. Although you can change the value for the current session, Excel will reset MDX Compatibility to1 when the file is next opened.

```
Provider=MSOLAP.5;Integrated Security=SSPI;Persist Security Info=True;Initial Catalog=Adventure Works DW 2008R2;Data
Source=AW-SRV01;MDX Compatibility=1;Safety Options=2;MDX Missing Member Mode=Error
```

For more information, see Data Connections, Data Sources, and Connection Strings (Report Builder and SSRS) and Data Authentication for Excel Services in SharePoint Server 2013.

Connection string formats used in Analysis Services

This section lists all of the connection string formats supported by Analysis Services. With the exception of connections to Power Pivot databases, you can specify these connections strings in applications that connect to Analysis Services.

Native (or direct) connections to the server

`Data Source=server[:port][\instance]` where "port" and "\instance" are optional. For example, specifying "Data Source=server1" opens a connection to the default instance (and default port 2383) on a server named "server1".

"Data Source=server1:port1" will open a connection to an Analysis Services instance running on port "port1" on "server1".

"Data Source=server1\instance1" will open a connection to SQL Browser (on its default port 2382), resolve the port for the named instance "instance1", then open the connection to that Analysis Services port.

"Data Source=server1:port1\instance1" will open a connection to SQL Browser on "port1", resolve the port for the "instance1" named instance, then open the connection to that Analysis Services port.

Local cube connections (.cub files)

`Data Source=<path>`, for example "Data Source=c:\temp\a.cub"

Http(s) connections to msmdpump.dll

`Data Source=<URL>`, where the URL is the HTTP or HTTPS address to the virtual IIS folder that contains the msmdpump.dll. For more information, see Configure HTTP Access to Analysis Services on Internet Information Services (IIS) 8.0.

Http(s) connections to Power Pivot workbooks (.xlsx, .xlsb or .xlsm files)

`Data Source=<URL>`, where the URL is the SharePoint path to a Power Pivot workbook that has been published to a SharePoint library. For example, `Data Source=http://localhost/Shared Documents/Sales.xlsx`.

Http(s) connections to BI Semantic Model Connection files

`Data Source=<URL>` where the URL is the SharePoint path to the .bism file. For example, `Data Source=http://localhost/Shared Documents/Sales.bism`.

Embedded Power Pivot connections

`Data Source=$Embedded$` where $embedded$ is a moniker that refers to an embedded Power Pivot data model inside the workbook. This connection string is created and managed internally. Do not modify it. Embedded connection strings are resolved by the Power Pivot for Excel add-in on client workstations, or by Power Pivot for SharePoint instances in a SharePoint farm.

Local server context in Analysis Services stored procedures

`Data Source=*`, where * resolves to the local instance.

Encrypting Connection Strings

Analysis Services uses its own encryption keys to encrypt connection strings. It does not generate a self-signed certificate.

Analysis Services encrypts and stores the connection strings it uses to connect to each of its data sources. If the connection to a data source requires a user name and password, you can have Analysis Services store the name and password with the

connection string, or prompt you for the name and password each time a connection to the data source is required. Having Analysis Services prompt you for user information means that this information does not have to be stored and encrypted. However, if you store this information in the connection string, this information does need to be encrypted and secured.

To encrypt and secure the connection string information, Analysis Services uses the Data Protection API.

Analysis Services uses a separate encryption key to encrypt connection string information for each Analysis Services database. Analysis Services creates this key when you create a database, and encrypts connection string information based on the Analysis Services startup account. When Analysis Services starts, the encrypted key for each database is read, decrypted, and stored. Analysis Services then uses the appropriate decrypted key to decrypt the data source connection string information when Analysis Services needs to connect to a data source.

See Also

Configure HTTP Access to Analysis Services on Internet Information Services (IIS) 8.0
Configure Analysis Services for Kerberos constrained delegation
Data providers used for Analysis Services connections
Connect to Analysis Services

Authentication methodologies supported by Analysis Services

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Connections from a client application to an Analysis Services instance require Windows authentication (integrated). You can provide a Windows user identity using any of the following methods:

- NTLM
- Kerberos (see Configure Analysis Services for Kerberos constrained delegation)
- EffectiveUserName on the connection string
- Basic or Anonymous (requires configuration for HTTP access)
- Stored credentials

Notice that Claims authentication is not supported. You cannot use a Windows Claims token to access Analysis Services. The Analysis Services client libraries only work with Windows security principles. If your BI solution includes claims identities, you will need Windows identity shadow accounts for each user, or use stored credentials to access Analysis Services data.

For more information about BI and Analysis Services authentication flows, see Microsoft BI Authentication and Identity Delegation.

Understanding your authentication alternatives

Connecting to an Analysis Services database requires a Windows user or group identity and associated permissions. The identity might be a general purpose login used by anyone who needs to view a report, but a more likely scenario includes the identity of individual users.

Often, a tabular or multidimensional model will have different levels of data access, by object or within the data itself, based on who is making the request. To meet this requirement, you can use NTLM, Kerberos, EffectiveUserName, or Basic authentication. All of these techniques offer an approach for passing in different user identities with each connection. However, most of these choices are subject to a single hop limitation. Only Kerberos with delegation allows the original user identity to flow across multiple computer connections to a backend data store on a remote server.

NTLM

For connections that specify `SSPI=Negotiate`, NTLM is the backup authentication subsystem used when a Kerberos domain controller is not available. Under NTLM, any user or client application can access a server resource as long as the request is a direct connection from a client to the server, the person requesting the connection has permission to the resource, and the client and server computers are in the same domain.

In multi-tier solutions, the single hop restriction of NLTM can be a major constraint. The user identity making the request can be impersonated on exactly one remote server, but travels no further. If the current operation requires services running on multiple computers, you will need to configure Kerberos constrained delegation to reuse the security token on backend servers. Alternatively, you can use stored credentials or Basic authentication to pass in new identity information over a single hop connection.

Kerberos Authentication and Kerberos Constrained Delegation

Kerberos authentication is the basis of Windows integrated security in Active Directory domains. As with NTLM, impersonation under Kerberos is limited to a single hop unless you enable delegation.

To support multi-hop connections, Kerberos provides both constrained and unconstrained delegation, but for most scenarios, constrained delegation is considered a security best practice. Constrained delegation allows a service to pass the security token of the user identity to a designated down-level service on a remote computer. For multi-tier applications, delegating a user identity from a middle tier application server to a backend database such as Analysis Services is a common requirement. For example, a tabular or multidimensional model that returns different data based on user identity would require identity delegation from a middle tier service to avoid having the user re-enter credentials, or getting security credentials some other way.

Constrained delegation requires additional configuration in Active Directory, where services on both the sending and receiving end of the request are explicitly authorized for delegation. Although there are configuration costs up front, once the service is configured, password updates are managed independently in Active Directory. You do not need to update stored account information in applications, as you would if using the stored credentials option described further on.

For more information about configuring Analysis Services for constrained delegation, see Configure Analysis Services for Kerberos constrained delegation.

> **NOTE**
>
> Windows Server 2012 supports constrained delegation across domains. In contrast, configuring Kerberos constrained delegation in domains at lower functional levels, such as Windows Server 2008 or 2008 R2, require both client and server computers to be members of the same domain.

EffectiveUserName

EffectiveUserName is a connection string property used for passing identity information to Analysis Services. Power Pivot for SharePoint uses it to record user activity in the usage logs. Excel Services and PerformancePoint Services can use it to retrieve data used by workbooks or dashboards in SharePoint. It can also be used in custom applications or scripts that perform operations on an Analysis Services instance.

For more information about using EffectiveUserName in SharePoint, see Use Analysis Services EffectiveUserName in SharePoint Server 2010.

Basic Authentication and Anonymous User

Basic authentication provides yet a fourth alternative for connecting to a backend server as a specific user. Using Basic authentication, the Windows user name and password are passed on the connection string, introducing additional wire encryption requirements to ensure sensitive information is protected while in transit. An important advantage to using Basic authentication is that the authentication request can cross domain boundaries.

For Anonymous authentication, you can set the anonymous user identity to a specific Windows user account (IUSR_GUEST by default) or an application pool identity. The anonymous user account will be used on the Analysis Services connection, and must have data access permissions on the Analysis Services instance. When you use this approach, only the user identity associated with the Anonymous account is used on the connection. If your application requires additional identity management, you will need to choose one of the other approaches, or supplement with an identity management solution that you provide.

Basic and Anonymous are available only when you configure Analysis Services for HTTP access, using IIS and the msmdpump.dll to establish the connection. For more information, see Configure HTTP Access to Analysis Services on Internet Information Services (IIS) 8.0.

Stored Credentials

Most middle tier application services include functionality for storing a user name and password subsequently used to retrieve data from a down-level data store, such as Analysis Services or the SQL Server relational engine. As such, stored credentials provide a fifth alternative for retrieving data. Limitations with this approach include maintenance overhead associated with keeping user names and passwords up to date, and the use of a single identity on the connection. If your solution requires the identity of the original caller, then stored credentials would not be a viable alternative.

For more information about stored credentials, see Create, Modify, and Delete Shared Data Sources (SSRS) and Use Excel Services with Secure Store Service in SharePoint Server 2013.

See Also

Using Impersonation with Transport Security
Configure HTTP Access to Analysis Services on Internet Information Services (IIS) 8.0
Configure Analysis Services for Kerberos constrained delegation
SPN registration for an Analysis Services instance
Connect to Analysis Services

Configure Analysis Services for Kerberos constrained delegation

11/27/2018 • 8 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

When configuring Analysis Services for Kerberos authentication, you are most likely interested in achieving one or both of the following outcomes: having Analysis Services impersonate a user identity when querying data; or having Analysis Services delegate a user identity to a down-level service. Each scenario calls for slightly different configuration requirements. Both scenarios require verification to ensure configuration was done properly.

> **TIP**
>
> **Microsoft Kerberos Configuration Manager for SQL Server** is a diagnostic tool that helps troubleshoot Kerberos related connectivity issues with SQL Server. For more information, see Microsoft Kerberos Configuration Manager for SQL Server.

This topic contains the following sections:

- Allow Analysis Services to impersonate a user identity
- Configure Analysis Services for trusted delegation
- Test for impersonated or delegated identity

> **NOTE**
>
> Delegation is not required if the connection to Analysis Services is a single hop, or your solution uses stored credentials provided by SharePoint Secure Store Service or Reporting Services. If all connections are direct connections from Excel to an Analysis Services database, or based on stored credentials, you can use Kerberos (or NTLM) without having to configure constrained delegation.
>
> Kerberos constrained delegation is required if the user identity has to flow over multiple computer connections (known as "double-hop"). When Analysis Services data access is contingent upon user identity, and the connection request is from a delegating service, use the checklist in the next section to ensure that Analysis Services is able to impersonate the original caller. For more information about Analysis Services authentication flows, see Microsoft BI Authentication and Identity Delegation.
>
> As a security best practice, Microsoft always recommends constrained delegation over unconstrained delegation. Unconstrained delegation is a major security risk because it allows the service identity to impersonate another user on *any* downstream computer, service, or application (as opposed to just those services explicitly defined via constrained delegation).

Allow Analysis Services to impersonate a user identity

To allow up-level services such as Reporting Services, IIS, or SharePoint to impersonate a user identity on Analysis Services, you must configure Kerberos constrained delegation for those services. In this scenario, Analysis Services impersonates the current user using the identity provided by the delegating service, returning results based on role membership of that user identity.

TASK	DESCRIPTION
Step 1: Verify that accounts are suitable for delegation	Ensure that the accounts used to run the services have the correct properties in Active Directory. Service accounts in Active Directory must not be marked as sensitive accounts, or specifically excluded from delegation scenarios. For more information, see Understanding User Accounts. Note: Generally, all accounts and servers must belong to the same Active Directory domain or to trusted domains in the same forest. However, because Windows Server 2012 supports delegation across domain boundaries, you can configure Kerberos constrained delegation across a domain boundary if the domain functional level is Windows Server 2012. Another alternative is to configure Analysis Services for HTTP access and use IIS authentication methods on the client connection. For more information, see Configure HTTP Access to Analysis Services on Internet Information Services (IIS) 8.0.
Step 2: Register the SPN	Before setting up constrained delegation, you must register a Service Principle Name (SPN) for the Analysis Services instance. You will need the Analysis Services SPN when configuring Kerberos constrained delegation for middle tier services. See SPN registration for an Analysis Services instance for instructions. A Service Principle Name (SPN) specifies the unique identity of a service in a domain configured for Kerberos authentication. Client connections using integrated security often request an SPN as part of SSPI authentication. The request is forwarded to an Active Directory Domain Controller (DC), with the KDC granting a ticket if the SPN presented by the client has a matching SPN registration in Active Directory.

TASK	DESCRIPTION
Step 3: Configure constrained delegation	After validating the accounts you want to use and registering SPNs for those accounts, your next step is to configure up-level services, such as IIS, Reporting Services, or SharePoint web services for constrained delegation, specifying the Analysis Services SPN as the specific service for which delegation is allowed. Services that run in SharePoint, such as Excel Services or Reporting Services in SharePoint mode, often host workbooks and reports that consume Analysis Services multidimensional or tabular data. Configuring constrained delegation for these services is a common configuration task, and necessary for supporting data refresh from Excel Services. The following links provide instructions for SharePoint services, as well as other services likely to present a downstream data connection request for Analysis Services data: Identity delegation for Excel Services (SharePoint Server 2010) or How to configure Excel Services in SharePoint Server 2010 for Kerberos authentication Identity delegation for PerformancePoint Services (SharePoint Server 2010) Identity delegation for SQL Server Reporting Services (SharePoint Server 2010) For IIS 7.0 see Configure Windows Authentication (IIS 7.0) or How to configure SQL Server 2008 Analysis Services and SQL Server 2005 Analysis Services to use Kerberos authentication.
Step 4: Test connections	When testing, connect from remote computers, under different identities, and query Analysis Services using the same applications as business users. You can use SQL Server Profiler to monitor the connection. You should see the user identity on the request. For more information, see Test for impersonated or delegated identity in this section.

Configure Analysis Services for trusted delegation

Configuring Analysis Services for Kerberos constrained delegation allows the service to impersonate a client identity on a down-level service, such as the relational database engine, so that data can be queried as if the client was connected directly.

Delegation scenarios for Analysis Services are limited to tabular models configured for **DirectQuery** mode. This is the only scenario in which Analysis Services can pass delegated credentials to another service. In all other scenarios, such as SharePoint scenarios mentioned in the previous section, Analysis Services is on the receiving end of the delegation chain. For more information about DirectQuery, see DirectQuery Mode.

> **NOTE**
> A common misconception is that ROLAP storage, processing operations, or access to remote partitions somehow introduce requirements for constrained delegation. This is not the case. All of these operations are executed directly by the service account (also referred to as the processing account), on its own behalf. Delegation is not required for these operations in Analysis Services, given that permissions for such operations are granted directly to the service account (for example, granting db_datareader permissions on the relational database so that the service can process data). For more information about server operations and permissions, see Configure Service Accounts (Analysis Services).

This section explains how to set up Analysis Services for trusted delegation. After you complete this task, Analysis Services will be able to pass delegated credentials to SQL Server, in support of DirectQuery mode used in Tabular solutions.

Before you start:

- Verify that Analysis Services is started.
- Verify the SPN registered for Analysis Services is valid. For instructions, see SPN registration for an Analysis Services instance

When both prerequisites are satisfied, continue with the following steps. Note that you must be a domain administrator to set up constrained delegation.

1. In Active Directory Users and Computers, find the service account under which Analysis Services runs. Right-click the service account and choose **Properties**.

   For illustrative purposes, the following screenshots use OlapSvc and SQLSvc to represent Analysis Services and SQL Server, respectively.

   OlapSvc is the account that will be configured for constrained delegation to SQLSvc. When you complete this task, OlapSvc will have permission to pass delegated credentials on a service ticket to SQLSvc, impersonating the original caller when requesting data.

2. On the Delegation tab, select **Trust this user for delegation to specified services only**, followed by **Use Kerberos only**. Click **Add** to specify which service Analysis Services is permitted to delegate credentials.

   Delegation tab appears only when the user account (OlapSvc) is assigned to a service (Analysis Services), and the service has an SPN registered for it. SPN registration requires that the service is running.

3. On the Add Services page, click **Users or Computers**.

4. On the Select Users or Computer page, enter the account used to run the SQL Server instance providing data to Analysis Services tabular model databases. Click **OK** to accept the service account.

   If you cannot select the account you want, verify that SQL Server is running and has an SPN registered for that account. For more information about SPNs for the database engine, see Register a Service Principal Name for Kerberos Connections.

5. The SQL Server instance should now appear in Add Services. Any service also using that account will also appear in the list. Choose the SQL Server instance you want to use. Click **OK** to accept the instance.

6. The properties page of the Analysis Services service account should now look similar to the following screenshot. Click **OK** to save your changes.

7. Test for successful delegation by connecting from a remote client computer, under a different identity, and query the tabular model. You should see the user identity on the request in SQL Server Profiler.

Test for impersonated or delegated identity

Use SQL Server Profiler to monitor the identity of the user who is querying data.

1. Start **SQL Server Profiler** on the Analysis Services instance and then start a new trace.
2. In Events Selection, verify that **Audit Login** and **Audit Logout** are checked in the Security Audit section.
3. Connect to Analysis Services via an application service (such as SharePoint or Reporting Services) from a remote client computer. The Audit Login event will show the identity of the user connecting to Analysis Services.

Thorough testing will require the use of network monitoring tools that can capture Kerberos requests and responses on the network. The Network Monitor utility (netmon.exe), filtered for Kerberos, can be used for this task. For more information about using Netmon 3.4 and other tools for testing Kerberos authentication, see Configuring Kerberos authentication: Core configuration (SharePoint Server 2010).

Additionally, see The Most Confusing Dialog Box in Active Directory for a thorough description of each option in the Delegation tab of the Active Directory object's property dialog box. This article also explains how to use LDP to test and interpret test results.

See Also

Microsoft BI Authentication and Identity Delegation
Mutual Authentication Using Kerberos
Connect to Analysis Services
SPN registration for an Analysis Services instance
Connection String Properties (Analysis Services)

SPN registration for an Analysis Services instance

11/28/2018 • 10 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A Service Principle Name (SPN) uniquely identifies a service instance in an Active Directory domain when Kerberos is used to mutually authenticate client and service identities. An SPN is associated with the logon account under which the service instance runs.

For client applications connecting to Analysis Services via Kerberos authentication, the Analysis Services client libraries construct an SPN using the host name from the connection string and other well-known variables, such as the service class, that are fixed in any given release of Analysis Services.

For mutual authentication to occur, the SPNs constructed by the client must match a corresponding SPN object on an Active Directory Domain Controller (DC). This means that you might need to register multiple SPNs for a single Analysis Services instance to cover all of the ways in which a user might specify the host name on a connection string. For example, you probably need two SPNs to handle both the fully-qualified domain name (FQDN) of a server, as well as the short computer name. Correctly registering the Analysis Services SPN is essential for a successful connection. If the SPN is non-existent, malformed, or duplicated, the connection will fail.

SPN registration is a manual task performed by the Analysis Services administrator. Unlike the SQL Server database engine, Analysis Services never auto-registers its SPN at service startup. Manual registration is required when Analysis Services runs under the default virtual account, a domain user account, or a built-in account, including a per-service SID.

SPN registration is not required if the service runs under a predefined managed service account created by a domain administrator. Note that depending on the functional level of your domain, registering an SPN can require domain administrator permissions.

> **TIP**
>
> **Microsoft Kerberos Configuration Manager for SQL Server** is a diagnostic tool that helps troubleshoot Kerberos related connectivity issues with SQL Server. For more information, see Microsoft Kerberos Configuration Manager for SQL Server.

This topic contains the following sections:

When SPN registration is required

SPN format for Analysis Services

SPN registration for a virtual account

SPN registration for a domain account

SPN registration for a built-in account

SPN registration for a named instance

SPN registration for an SSAS cluster

SPN registration for SSAS instances configured for HTTP access

SPN registration for SSAS instances listening on fixed ports

When SPN registration is required

Any client connection that specifies "SSPI=Kerberos" on the connection string will introduce SPN registration requirements for an Analysis Services instance.

SPN registration is required under the following circumstances. For more detailed information, see Configure Analysis Services for Kerberos constrained delegation.

- Identity delegation is necessary to flow the user identity from the client application or middle-tier service to Analysis Services. Identity delegation is typically used when per-user permissions or filters are defined on specific objects.

  A common scenario involving identity delegation is configuring middle-tier services, such as Excel Services or Reporting Services, for constrained delegation for the purpose of impersonating a user identity when retrieving data in Analysis Services. To support this behavior, you must provide an Analysis Services SPN as the destination service when configuring Excel Services or Reporting Services for constrained delegation.

- Analysis Services delegates a user identity when retrieving data from a SQL Server relational database for tabular databases using DirectQuery mode. This is the only scenario in which Analysis Services will delegate the user identity to another service.

SPN format for Analysis Services

Use **setspn** to register an SPN. On newer operating systems, **setspn** is installed as a system utility. For more information, see SetSPN.

The following table describes each part of an Analysis Services SPN.

ELEMENT	DESCRIPTION
Service class	MSOLAPSvc.3 identifies the service as an Analysis Services instance. The .3 is a reference to the version of the XMLA-over-TCP/IP protocol used in Analysis Services transmissions. It is unrelated to product release. As such, MSOLAPSvc.3 is the correct service class for SQL Server 2005, 2008, 2008 R2, 2012 and any future release of Analysis Services until the protocol itself is revised.
Host-name	Identifies the computer on which the service is running. This can be a fully-qualified domain name or a NetBIOS name. You should register an SPN for both. When registering an SPN for the NetBIOS name of a server, be sure to use `SetupSPN -S` to check for duplicate registration. NetBIOS names are not guaranteed to be unique in a forest, and having a duplicate SPN registration will cause the connection to fail. For Analysis Services load balanced clusters, the host name should be the virtual name assigned to the cluster. Never create an SPN using the IP address. Kerberos uses the DNS resolution capabilities of the domain. Specifying an IP address bypasses that capability.

ELEMENT	DESCRIPTION
Port-number	Although the port number is part of SPN syntax, you never specify a port number when registering an Analysis Services SPN. The colon (:) character, typically used to provide a port number in standard SPN syntax, is used by Analysis Services to specify the instance name. For an Analysis Services instance, the port is assumed to be the default port (TCP 2383) or a port assigned by the SQL Server Browser service (TCP 2382).
Instance-name	Analysis Services is a replicable service that can be installed multiple times on the same computer. Each instance is identified through its instance name. The instance name is prefixed with a colon (:) character. For example, given a host computer named SRV01 and a named instance of SSAS-Tabular, the SPN should be SRV01:SSAS-Tabular. Note that the syntax for specifying a named Analysis Services instance differs from that used by other SQL Server instances. Other services uses a backslash (\ ) to append the instance name in an SPN.
Service-account	This is the startup account of the **MSSQLServerOLAPService** Windows service. It can be a Windows domain user account, virtual account, managed service account (MSA) or a built-in account such as a per-service SID, NetworkService, or LocalSystem. A Windows domain user account can be formatted as domain\user or user@domain.

SPN registration for a virtual account

Virtual accounts are the default account type for SQL Server services. The virtual account is **NT Service\MSOLAPService** for a default instance and **NT Service\MSOLAP$**<instance-name> for a named instance.

As the name implies, these accounts do not exist in Active Directory. A virtual account exists only on the local computer. When connecting to external services, applications, or devices, the connection is made using the local machine account. For this reason, an SPN registration for Analysis Services running under a virtual account is actually an SPN registration for the machine account.

Example syntax for a default instance running as NT Service\MSOLAPService

This example shows **setspn** syntax for Analysis Services default instance running under the default virtual account. In this example, the computer host name is **AW-SRV01**. As noted, SPN registration must specify the *machine account* instead of the virtual account, **NT Service\MSOLAPService**.

```
Setspn -s MSOLAPSvc.3/AW-SRV01.AdventureWorks.com AW-SRV01
```

> **NOTE**
> Remember to create two SPN registrations, one for the NetBIOS host name and a second for a fully-qualified domain name of the host. Different client applications use different host name conventions when connecting to Analysis Services. Having two SPN registrations ensures that both versions of the host name are accounted for.

Example syntax for a named instance running as NT Service\MSOLAP$<instance-name>

This example shows **setspn** syntax for a named instance running under the default virtual account. In this example, the computer host name is **AW-SRV02**, and the instance name is **AW-FINANCE**. Again, it is the machine account that is specified for the SPN, rather than the virtual account **NT Service\MSOLAP$**<instance-name>.

```
Setspn -s MSOLAPSvc.3/AW-SRV02.AdventureWorks.com:AW-FINANCE AW-SRV02
```

SPN registration for a domain account

Using a domain account to run an Analysis Services instance is a common practice.

For Analysis Services instances that run in a network or hardware load balanced cluster, a domain account is required, with each instance in the cluster running under the same domain account.

Example syntax for a default instance running as a domain user

This example shows **setspn** syntax for Analysis Services default instance running under a domain user account, **SSAS-Service**, in the AdventureWorks domain.

```
Setspn -s msolapsvc.3/AW-SRV01.Adventureworks.com AdventureWorks\SSAS-Service
```

> **TIP**
> Verify whether the SPN was created for the Analysis Services server by running `Setspn -L <domain account>` or `Setspn -L <machinename>`, depending on how the SPN was registered. You should see MSOLAPSVC.3/<hostname> in the list.

SPN registration for a built-in account

Although this practice is not recommended, older Analysis Services installations are sometimes configured to run under built-in accounts like Network Service, Local Service, or Local System.

Example syntax for a default instance running under a built-in account

SPN registration for a service running under a built-in account or per-service SID is equivalent to the SPN syntax used for the virtual account. Instead of the account name, use the machine account:

```
Setspn -s MSOLAPSvc.3/AW-SRV01.AdventureWorks.com AW-SRV01
```

SPN registration for a named instance

Named instances of Analysis Services use dynamic port assignments that are detected by the SQL Server Browser service. When using a named instance, register an SPN for both the SQL Server Browser Service and the Analysis Services named instance. For more information, see An SPN for the SQL Server Browser service is required when you establish a connection to a named instance of SQL Server Analysis Services or of SQL Server.

Example of SPN syntax for the SQL Browser Service running as LocalService

The service class is **MSOLAPDisco.3**. By default, this service runs as NT AUTHORITY\LocalService, which means SPN registration is set for the machine account. In this example, the machine account is **AW-SRV01**, corresponding to the computer name.

```
Setspn -S MSOLAPDisco.3/AW-SRV01.AdventureWorks.com AW-SRV01
```

SPN registration for an SSAS cluster

For Analysis Services failover clusters, the host name should be the virtual name assigned to the cluster. This is the SQL Server network name, specified during SQL Server Setup when you installed Analysis Services on top of an existing WSFC. You can find this name in Active Directory. You can also find it in **Failover Cluster Manager** | **Role** | **Resources** tab. The server name on the Resources tab is what should be used as the 'virtual name' in the SPN command.

SPN syntax for an Analysis Services cluster

```
Setspn -s msolapsvc.3/<virtualname.FQDN > <domain user account>
```

Recall that nodes in an Analysis Services cluster are required to use the default port (TCP 2383) and run under the same domain user account so that each node has the same SID. See How to Cluster SQL Server Analysis Services for more information.

SPN registration for SSAS instances configured for HTTP access

Depending on solution requirements, you might have configured Analysis Services for HTTP access. If your solution includes IIS as a middle tier component, and Kerberos authentication is a solution requirement, you might need to manually register an SPN for IIS. For more information, see "Configure the settings on the computer running IIS" in How to configure SQL Server 2008 Analysis Services and SQL Server 2005 Analysis Services to use Kerberos authentication.

In terms of SPN registration for the Analysis Services instance, there is no difference between an instance configured for TCP or HTTP. The connection to Analysis Services from IIS, using the MSMDPUMP ISAPI extension, is always TCP.

This means that you can use the instructions from previous sections for default or named instance to register the SPN. When specifying the host name, be sure to use the host name you specified in the msmdpump.ini file when you configured the service for HTTP access.

For more information about HTTP access, see Configure HTTP Access to Analysis Services on Internet Information Services (IIS) 8.0.

SPN registration for SSAS instances listening on fixed ports

You cannot specify a port number on an Analysis Services SPN registration. If you installed Analysis Services as the default instance and configured it to listen on a fixed port, you must now configure it to listen on the default port (TCP 2383). For named instances, you need to use SQL Server Browser service and dynamic port assignments.

An Analysis Services instance can only listen on a single port. Using multiple ports is not supported. For more information about port configuration, see Configure the Windows Firewall to Allow Analysis Services Access.

See Also

Microsoft BI Authentication and Identity Delegation
Mutual Authentication Using Kerberos
How to configure SQL Server 2008 Analysis Services and SQL Server 2005 Analysis Services to use Kerberos authentication

Service Principal Names (SPNs) SetSPN Syntax (Setspn.exe)
What SPN do I use and how does it get there?
SetSPN
Service Accounts Step-by-Step Guide
Configure Windows Service Accounts and Permissions
How to use SPNs when you configure Web applications that are hosted on Internet Information Services
what's new in service accounts
Configure Kerberos authentication for SharePoint 2010 Products (white paper)

Configure HTTP Access to Analysis Services on IIS 8.0

2/11/2019 • 18 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This article explains how to set up an HTTP endpoint for accessing an Analysis Services instance. You can enable HTTP access by configuring MSMDPUMP.dll, an ISAPI extension that runs in Internet Information Services (IIS) and pumps data to and from client applications and an Analysis Services server. This approach provides an alternative means for connecting to Analysis Services when your BI solution calls for the following capabilities:

- Client access is over Internet or extranet connections, with restrictions on which ports can be enabled.
- Client connections are from non-trusted domains in the same network.
- Client application runs in a network environment that allows HTTP but not TCP/IP connections.
- Client applications cannot use the Analysis Services client libraries (for example, a Java application running on a UNIX server). If you cannot use the Analysis Services client libraries for data access, you can use SOAP and XML/A over a direct HTTP connection to an Analysis Services instance.
- Authentication methods other than Windows integrated security are required. Specifically, you can use Anonymous connections and Basic authentication when configuring Analysis Services for HTTP access. Digest, Forms, and ASP.NET authentication are not supported. A requirement of Basic authentication is one of the primary reasons for enabling HTTP access. To learn more, see Microsoft BI Authentication and Identity Delegation.

You can configure HTTP access for any supported version or edition of Analysis Services, running either tabular mode or multidimensional mode. Local cubes are an exception. You cannot connect to a local cube via an HTTP endpoint.

Setting up HTTP access is a post-installation task. Analysis Services must be installed before you can configure it for HTTP access. As the Analysis Services administrator, you will need to grant permissions to Windows accounts before HTTP access is possible. Additionally, it is a best practice to validate your installation first, ensuring that it is fully operational before configuring the server any further. After HTTP access is configured, you can use both the HTTP endpoint and the regular network name of the server over TCP/IP. Setting up HTTP access does not invalidate other approaches for data access.

As you move forward with MSMDPUMP configuration, remember there are two connections to consider: client-to-IIS, IIS-to-SSAS. The instructions in this article are about IIS-to-SSAS. Your client application might require additional configuration before it can connect to IIS. Decisions such as whether to use SSL, or how to configure bindings, are out of scope for this article. See Web Server (IIS) for more information about IIS.

Overview

MSMDPUMP is an ISAPI extension that loads into IIS and provides redirection to a local or remote Analysis Services instance. By configuring this ISAPI extension, you create an HTTP endpoint to an Analysis Services instance.

You must create and configure one virtual directory for each HTTP endpoint. Each endpoint will need its own set of MSMDPUMP files, for each Analysis Services instance you want to connect to. A configuration file in this file set specifies the name of the Analysis Services instance used for each HTTP endpoint.

On IIS, MSMDPUMP connects to Analysis Services using the Analysis Services OLE DB provider over TCP/IP. Although client requests can originate outside of domain trust, both Analysis Services and IIS must be in the same domain or in trusted domains in order for the native connection to succeed.

When MSMDPUMP connects to Analysis Services, it does so under a Windows user identity. This account will either be the Anonymous account if you configured the virtual directory for anonymous connections, or a Windows user account. The account must have the appropriate data access rights on the Analysis Services server and database.

The following table lists additional considerations when you enable HTTP access for different scenarios.

SCENARIO	CONFIGURATION
IIS and Analysis Services on the same computer	This is the simplest configuration because it allows you to use the default configuration (where the server name is localhost), the local Analysis Services OLE DB provider, and Windows integrated security with NTLM. Assuming that the client is also in the same domain, authentication is transparent to the user, with no additional work on your part.
IIS and Analysis Services on different computers	For this topology, you must install the Analysis Services OLE DB provider on the web server. You must also edit the msmdpump.ini file to specify the location of Analysis Services instance on the remote computer. This topology adds a double-hop authentication step, where credentials must flow from the client to the web server, and on to the backend Analysis Services server. If you are using Windows credentials and NTLM, you will get an error because NTLM does not allow delegation of client credentials to a second server. The most common solution is to use Basic authentication with Secure Sockets Layer (SSL), but this will require users to provide a user name and password when accessing the MSMDPUMP virtual directory. A more straightforward approach might be to enable Kerberos and configure Analysis Services constrained delegation so that users can access Analysis Services in a transparent manner. See Configure Analysis Services for Kerberos constrained delegation for details. Consider which ports to unblock in Windows Firewall. You will need to unblock ports on both servers to allow access to the web application on IIS, and to Analysis Services on a remote server.

SCENARIO	CONFIGURATION
Client connections are from a non-trusted domain or an extranet connection	Client connections from a non-trusted domain introduce further restrictions on authentication. By default, Analysis Services uses Windows integrated authentication, which requires users to be on the same domain as the server. If you have Extranet users who connect to IIS from outside the domain, those users will get a connection error if the server is configured to use the default settings. Workarounds include having Extranet users connect through a VPN using domain credentials. However, a better approach might be to enable Basic authentication and SSL on your IIS web site.

Prerequisites

The instructions in this article assume IIS is already configured and that Analysis Services is already installed. Windows Server 2012 ships with IIS 8.x as a server role that you can enable on the system.

Extra configuration in IIS 8.0

The default configuration of IIS 8.0 is missing components that are necessary for HTTP access to Analysis Services. These components, found in the **Security** and **Application Development** feature areas of the **Web Server (IIS)** role, include the following:

- **Security** | **Windows Authentication**, or **Basic Authentication**, and any other security features required for your data access scenario.
- **Application Development** | **CGI**
- **Application Development** | **ISAPI Extensions**

To verify or add these components, use **Server Manager** | **Manage** | **Add Roles and Features**. Step through the wizard until you get to **Server Roles**. Scroll down to find **Web Server (IIS)**.

1. Open **Web Server** | **Security** and choose the authentication methods.
2. Open **Web Server** | **Application Development** and choose **CGI** and **ISAPI Extensions**.

When IIS is on a remote server

A remote connection between IIS and Analysis Services requires that you install the Analysis Services OLE DB provider (MSOLAP) on the Windows server running IIS.

1. Go to the download page for SQL Server 2014 Feature Pack
2. Click the red Download button.
3. Scroll down to find ENU\x64\SQL_AS_OLEDB.msi
4. Follow the instructions in the wizard to complete the installation.

> **NOTE**
> Remember to unblock the ports in Windows Firewall to allow client connections to a remote Analysis Services server. For more information, see Configure the Windows Firewall to Allow Analysis Services Access.

Step 1: Copy the MSMDPUMP files to a folder on the Web server

Each HTTP endpoint that you create must have its own set of MSMDPUMP files. In this step, you copy the MSMDPUMP executable, configuration file, and resource folder from the Analysis Services program folders to a new virtual directory folder that you will create on the file system of the computer running IIS.

The drive must be formatted for the NTFS file system. The path to the folder that you create must not contain any spaces.

1. Copy the following files, found at <drive>:\Program Files\Microsoft SQL Server\<instance>\OLAP\bin\isapi: MSMDPUMP.DLL, MSMDPUMP.INI, and a Resources folder.

2. On the web server, create a new folder: <drive>:\inetpub\wwwroot**OLAP**
3. Paste the files that you previously copied into this new folder.
4. Verify that the \inetpub\wwwroot\OLAP folder on your web server contains the following: MSMDPUMP.DLL, MSMDPUMP.INI, and a Resources folder. Your folder structure should look like this:
   - <drive>:\inetpub\wwwroot\OLAP\MSMDPUMP.dll
   - <drive>:\inetpub\wwwroot\OLAP\MSMDPUMP.ini
   - <drive>:\inetpub\wwwroot\OLAP\Resources

> **NOTE**
> The IIS manager might not be able to connect to the Analysis Services in the current version if the Database is a backup from a previous one. This is caused by changes in the MSMDPUMP and should be solved by copying the msmdpump.dll file from the previous working version.

Step 2: Create an application pool and virtual directory in IIS

Next, create an application pool and an endpoint to the pump.

Create an application pool

1. Start IIS Manager.

2. Open the server folder, right-click **Application Pools** and then click **Add Application Pool**. Create an application pool named **OLAP**, using the .NET Framework, with Managed pipeline mode set to **Classic**.

3. By default, IIS creates application pools using **ApplicationPoolIdentity** as the security identity, which is a valid choice for HTTP access to Analysis Services. If you have specific reasons for changing the identity, right-click **OLAP**, and then select **Advanced Settings**. Select **ApplicationPoolIdentity**. Click the **Change** button for this property to replace the built-in account with the custom account you want to use.

4. By default, on a 64-bit operating system, IIS sets the **Enable 32-bit Applications** property to **false**. If you copied msmdpump.dll from a 64-bit installation of Analysis Services, this is the correct setting for the MSMDPUMP extension on a 64-bit IIS server. If you copied the MSMDPUMP binaries from a 32-bit installation, set it to **true**. Check this property now in **Advanced Settings** to ensure it is set correctly.

Create an application

1. In IIS Manager, open **Sites**, open **Default Web Site**. You should see a folder named **Olap**. This is a reference to the OLAP folder you created under \inetpub\wwwroot.

2. Right-click the folder and choose **Convert to Application**.

3. In Add Application, enter **OLAP** for the alias. Click **Select** to choose the OLAP application pool. Physical Path should be set to C:\inetpub\wwwroot\OLAP

4. Click **OK**. Refresh the web site and notice that the OLAP folder is now an application under the default web site. The virtual path to the MSMDPUMP file is now established.

> **NOTE**
>
> Previous versions of these instructions included steps for creating a virtual directory. That step is no longer necessary.

Step 3: Configure IIS authentication and add the extension

In this step, you further configure the SSAS virtual directory you just created. You will specify an authentication method and then add a script map. Supported authentication methods for Analysis Services over HTTP include:

- Windows authentication (Kerberos or NTLM)
- Basic authentication
- Anonymous authentication

Windows authentication is considered the most secure, and leverages existing infrastructure for networks that use Active Directory. To use Windows authentication effectively, all browsers, client applications, and server applications must support it. This is the most secure and recommended mode, but it requires that IIS be able to access a Windows domain controller that can authenticate the identity of the user requesting a connection.

For topologies that put Analysis Services and IIS on different computers, you will need to address double-hop issues that arise when a user identity needs to be delegated to a second service on a remote machine, typically by enabling Analysis Services for Kerberos constrained delegation. For more information, see Configure Analysis Services for Kerberos constrained delegation.

Basic authentication is used when you have Windows identities, but user connections are from non-trusted domains, prohibiting the use of delegated or impersonated connections. Basic authentication lets you specify a user identity and password on a connection string. Instead of using the security context of the current user, credentials on the connection string are used to connect to Analysis Services. Because Analysis Services supports only Windows authentication, any credentials passed to it must be a Windows user or group that is a member of the domain in which Analysis Services is hosted.

Anonymous authentication is often used during initial testing because its ease of configuration helps you to quickly validate HTTP connectivity to Analysis Services. In just a few steps, you can assign a unique user account as the identity, grant that account permissions in Analysis Services, use the account to verify data access in a client application, and then disable Anonymous authentication when testing is complete.

You can also use Anonymous authentication in a production environment if your users do not have Windows user accounts, but follow best practices by locking down permissions on the host system, as called out in this article: Enable Anonymous Authentication (IIS 7). Be sure that authentication is set on the virtual directory, and not on the parent web site, to further reduce the level of account access.

When Anonymous is enabled, any user connection to the HTTP endpoint is allowed to connect as the anonymous user. You won't be able to audit individual user connections, nor use the user identity to select data from a model. As you can see, using Anonymous impacts everything from model design, to data refresh and access. However, if users do not have a Windows user login to start with, using the Anonymous account might be your only option.

Set the authentication type and add a script map

1. In IIS Manager, open **Sites**, open **Default Web Site**, and then select the **OLAP** virtual directory.
2. Double-click **Authentication** in the IIS section of the main page.

3. Enable **Windows Authentication** if you are using Windows integrated security.

4. Alternatively, enable **Basic Authentication** if your client and server applications are in different domains. This mode requires the user to enter a user name and password. The user name and password are transmitted over the HTTP connection to IIS. IIS will try to impersonate the user using the provided credentials when connecting to MSMDPUMP, but the credentials will not be delegated to Analysis

Services. Instead, you will need to pass a valid user name and password on a connection, as described in Step 6 in this document.

> **IMPORTANT**
>
> Please note that it is imperative for anyone building a system where the password is transmitted to have ways of securing the communication channel. IIS provides a set of tools that help you secure the channel. For more information, see How to Set Up SSL on IIS 7.

5. Disable **Anonymous Authentication** if you are using Windows or Basic authentication. When Anonymous authentication is enabled, IIS will always use it first, even if other authentication methods are enabled.

   Under Anonymous authentication, the pump (msmdpump.dll) runs as the user account you established for anonymous user. There is no distinction between the user connecting to IIS and the user connecting to Analysis Services. By default, IIS uses the IUSR account, but you can change it to a domain user account that has network permissions. You'll need this capability if IIS and Analysis Services are on different computers.

   For instructions on how to configure credentials for Anonymous authentication, see Anonymous Authentication.

   > **IMPORTANT**
   >
   > Anonymous authentication is most likely found in an extremely controlled environment, where users are given or denied access by way of access control lists in the file system. For best practices, see Enable Anonymous Authentication (IIS 7).

6. Click the **OLAP** virtual directory to open the main page. Double-click **Handler Mappings**.

7. Right-click anywhere on the page and then select **Add Script Map**. In the Add Script Map dialog box, specify ***.dll** as the request path, specify c:\inetpub\wwwroot\OLAP\msmdpump.dll as the executable, and type **OLAP** as the name. Keep all of the default restrictions associated with this script map.

8. When prompted to allow the ISAPI extension, click **Yes**.

Step 4: Edit the MSMDPUMP.INI file to set the target server

The MSMDPUMP.INI file specifies the Analysis Services instance that MSMDPUMP.DLL connects to. This instance can be local or remote, installed as the default or as a named instance.

Open the msmdpump.ini file located in folder C:\inetpub\wwwroot\OLAP and take a look at the contents of this file. It should look like the following:

```
<ConfigurationSettings>
<ServerName>localhost</ServerName>
<SessionTimeout>3600</SessionTimeout>
<ConnectionPoolSize>100</ConnectionPoolSize>
</ConfigurationSettings>
```

If the Analysis Services instance for which you are configuring HTTP access is located on the local computer and installed as a default instance, there is no reason to change this setting. Otherwise, you must specify the server name (for example, <ServerName>ADWRKS-SRV01</ServerName>). For a server that is installed as a named instance, be sure to append the instance name (for example, <ServerName>ADWRKS-SRV01\Tabular</ServerName>).

By default, Analysis Services listens on TCP/IP port 2383. If you installed Analysis Services as the default instance, you do not need to specify any port in <ServerName> because Analysis Services knows how to listen on port 2383 automatically. However, you do need to allow inbound connections to that port in Windows Firewall. For more information, see Configure the Windows Firewall to Allow Analysis Services Access.

If you configured a named or default instance of Analysis Services to listen on a fixed port, you must add the

port number to the server name (for example, <ServerName>AW-SRV01:55555</ServerName>) and you must allow inbound connections in Windows Firewall to that port.

Step 5: Grant data access permissions

As previously noted, you will need to grant permissions on the Analysis Services instance. Each database object will have roles that provide a given level of permissions (read or read/write), and each role will have members consisting of Windows user identities.

To set permissions, you can use SQL Server Management Studio. Under the **Database** | **Roles** folder, you can create roles, specify database permissions, assign membership to Windows user or group accounts, and then grant read or write permissions on specific objects. Typically, **Read** permissions on a cube are sufficient for client connections that use, but do not update, model data.

Role assignment varies depending on how you configured authentication.

Anonymous	Add to the Membership list the account specified in **Edit Anonymous Authentication Credentials** in IIS. For more information, see Anonymous Authentication,
Windows authentication	Add to the Membership list the Windows user or group accounts requesting Analysis Services data via impersonation or delegation. Assuming Kerberos constrained delegation is used, the only accounts that need permissions are the Windows user and group accounts requesting access. No permissions are necessary for the application pool identity.
Basic authentication	Add to the Membership list the Windows user or group accounts that will be passed on the connection string. In addition, if you are passing credentials via **EffectiveUserName** on the connection string, then the application pool identity must have administrator rights on the Analysis Services instance. In SSMS, right-click the instance	**Properties**	**Security**	**Add**. Enter the application pool identity. If you used the built-in default identity, the account is specified as **IIS AppPool\DefaultAppPool**.

For more information about setting permissions, see Authorizing access to objects and operations (Analysis Services).

Step 6: Test your configuration

The connection string syntax for MSMDPUMP is the URL to the MSMDPUMP.dll file.

If the web application is listening on a fixed port, append the port number to the server name or IP address (for example, `http://my-web-srv01:8080/OLAP/msmdpump.dll` or `http://123.456.789.012:8080/OLAP/msmdpump.dll`.

To quickly test the connection, you can open a connection using Internet Explorer, Microsoft Excel, or SQL Server Management Studio.

Troubleshoot connections using Internet Explorer

A connection request that terminates with this error might not give you much to go on: "Either a connection cannot be made to '<server name>', or Analysis Service is not running on the server".

To get a more informative error, do the following:

1. In **Internet Explorer** > **Internet Options** > **Advanced**, clear the checkbox for **Show Friendly HTTP messages**.

2. Retry the connection (for example, `http://my-web-srv01:8080/OLAP/msmdpump.dll`)

If you see an ERROR XML displayed in the browser window, you can eliminate MSMDPUMP as the potential cause and shift your focus to the certificate.

Test connections using SQL Server Management Studio

1. In Management Studio, in the Connect to Server dialog box, select **Analysis Services** as the server type. In Server name, enter the HTTP address of the msmdpump extension: `http://my-web-srv01/OLAP/msmdpump.dll`.

   Object Explorer displays the HTTP connection:

2. Authentication must be Windows authentication, and the person using Management Studio must be an Analysis Services administrator. An administrator can grant further permissions to enable access by other users.

Test connections using Excel

1. On the Data tab in Excel, in Get External Data, click **From Other Sources**, and then choose **From Analysis Services** to start the Data Connection wizard.

2. In Server name, enter the HTTP address of the msmdpump extension: `http://my-web-srv01/OLAP/msmdpump.dll`.

3. For Log on credentials, choose **Use Windows Authentication** if you are using Windows integrated security or NTLM, or Anonymous user.

   For Basic authentication, choose **Use the following User Name and Password**, and then specify the credentials used to sign on. The credentials you provide will be passed on the connection string to Analysis Services.

Test connections using AMO

You can test HTTP access programmatically using AMO, substituting the URL of the endpoint for the server name. For details, see Forum Post (How to sync SSAS 2008 R2 databases via HTTPS across domain/forest and

firewall boundaries).

An example connection string illustrating the syntax for HTTP(S) access using Basic authentication:

```
Data Source=https://<servername>/olap/msmdpump.dll; Initial Catalog=AdventureWorksDW2012; Integrated
Security=Basic; User ID=XXXX; Password=XXXXX;
```

For more information about setting up the connection programmatically, see Establishing Secure Connections in ADOMD.NET.

As a final step, be sure to follow-up with more rigorous testing by using a client computer that runs in the network environment from which the connections will originate.

See Also

Forum post (http access using msmdpump and basic authentication)
Configure the Windows Firewall to Allow Analysis Services Access
Authorizing access to objects and operations (Analysis Services)
IIS Authentication Methods
How to Set Up SSL on IIS 7

Client libraries (data providers) used for Analysis Services connections

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services provides three client libraries, also known as **data providers**, for server and data access from tools and client applications. Tools like SSMS and SSDT, and applications like Power BI Desktop and Excel connect to Analysis Services by using these libraries. Two of the client libraries, ADOMD.NET and Analysis Services Management Objects (AMO), are managed client libraries. The Analysis Services OLE DB provider (MSOLAP DLL) is a native client library. Client libraries are the same for both SQL Server Analysis Services and Azure Analysis Services.

Where to get newer versions

The version installed on a client computer should match the major version of the server providing the data. If the server installation is newer than the data providers installed on the workstations in your network, you might need to install newer libraries.

Client libraries included in earlier SQL Server Feature Packs correspond to that SQL version; however, they may not be the latest. Connecting to Azure Analysis Services may require later versions. All versions are backward compatible.

To get the latest, see Client libraries for connecting to Azure Analysis Services.

See also

Connect to Analysis Services

Disconnect Users and Sessions on Analysis Services Server

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

An administrator of Analysis Services may want to end user activity as part of workload management. You do this by canceling sessions and connections. Sessions can be formed automatically when a query is run (implicit), or named at the time of creation by the administrator (explicit). Connections are open conduits over which queries can be run. Both sessions and connections can be ended while they are active. For example, an administrator may want to end processing for a session if the processing is taking too long or if some doubt has arisen as to whether the command being executed was written correctly.

Ending Sessions and Connections

To manage sessions and connections, you can use Dynamic Management Views (DMVs) and XMLA:

1. In SQL Server Management Studio, connect to an Analysis Services instance.
2. Paste any one of the following DMV queries in an MDX query window to get a list of all sessions, connections, and commands that are currently executing:

   ```
   Select * from $System.Discover_Sessions
   ```

   ```
   Select * from $System.Discover_Connections
   ```

   ```
   Select * from $System.Discover_Commands
   ```

3. Press F5 to execute the query.

   The DMV query returns session and connection information in a tabular result set that is easier read and copy from.

Keep the query window open. In the next step, you will want to return to this page to copy the SPIDs of the session you want to disconnect.

To end a session, open a second XMLA query window.

1. Paste the following syntax into an MDX query window, replacing the ConnectionID, SessionID, or SPID placeholder with a valid value copied from the previous step.

   ```
   <Cancel xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">

      <ConnectionID>111</ConnectionID>
      <SessionID>222</SessionID>
      <SPID>333</SPID>

   <CancelAssociated>1</CancelAssociated>
   </Cancel>
   ```

2. Press F5 to execute the cancel command.

Ending a connection cancels all sessions and SPIDs, closing the host session.

Ending a session stops all commands (SPIDs) that are running as part of that session.

Ending a SPID cancels a particular commend.

In rare cases, Analysis Services will not close a connection if it cannot track all the sessions and SPIDs associated with the connection (for example, when multiple sessions are open in an HTTP scenario).

For more information about the XMLA referenced in this topic, see Execute Method (XMLA) and Cancel Element (XMLA).

See Also

Managing Connections and Sessions (XMLA)
BeginSession Element (XMLA)
EndSession Element (XMLA)
Session Element (XMLA)

Analysis Services Developer Documentation

10/26/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Analysis Services, almost every object and workload is programmable, and often there is more than one approach to choose from. Options include writing managed code, script, or using open standards like XMLA and MSOLAP if your solution requirements preclude using the .NET framework.

What you can accomplish in code

Typical programming scenarios include server and database deployment, administration, model and database creation, and data access from your custom applications and reports that consume Analysis Services data. Common to all these scenarios is a fixed architecture and object definition hierarchy, with well-understood operations that span data definition, processing, and query workloads.

Although objects and workloads are programmable, they are not extensible. Specifically, you cannot create custom data cartridges that retrieve data from unsupported data sources, customize or replace formula or storage engine behaviors, nor can you create new types of object metadata on a server, database, or model.

To further elaborate on the last point about creating new object types: while you cannot create a new type of object, you can create calculated objects built from expressions or code at run time. Not everything in your model needs to be predefined and mapped to an existing data structure. Additionally, you can extend the schema via Annotations in AMO to pass object-specific information to your client application.

Choose a platform or approach to development

Analysis Services provides many ways to customize a solution through code, but most developers use the managed APIs or script.

- Managed APIs include AMO and TOM for data definition and administrative tasks, and ADOMD.NET for query support from client code. In SQL Server 2016, AMO is updated to use the new Tabular metadata for models created or upgraded to compatibility level 1200 and higher.
- Script can often achieve the same results as a program executable, with possibly less work.
  - You can write PowerShell script using Analysis Services PowerShell components that call AMO types directly. Within PowerShell, you can also create and execute ASSL/XMLA or TMSL (in JSON) script.
  - ASSL and TMSL are script languages that provide objects used in discover and execute operations. Which type of script you use depends on the underlying server, database, or model.
  - Tabular models or databases at compatibility level 1200 and higher use the Tabular Model Scripting Language (TMSL), which is in JSON.
  - Multidimensional models and Tabular models at compatibility levels 1050-1103 use Analysis Services Scripting Language (ASSL), which is the Analysis Services extension of the XMLA open standard.
  - You can generate ASSL or TMSL script in Management Studio. You can also use **View Code** in SQL Server Data Tools to view the model definition in ASSL or TMSL.
- Although it is possible to build a solution based on the open standards of XMLA and MDX, it's quite rare to

do so. There is no documentation other than XMLA and MDX reference to help you, and most community and forum support draws from experiences with .NET or native (MSOLAP) technologies.

Programming in Analysis Services

Data Mining Programming Describes the approaches building solutions that include data mining objects.

Multidimensional Model Programming Describes the development tasks and approaches for integrating multidimensional model objects in a custom solution.

Tabular Model Programming for Compatibility Level 1200 and higher **New in SQL Server 2016**. Summarizes the interfaces and script languages used for working with Tabular 1200 and higher models programmatically.

Tabular Model Programming for Compatibility Levels 1050 through 1103 This documentation is intended for developers who support tabular models at earlier compatibility levels. It describes the CSDL extensions that define a tabular model in XML syntax. It also includes information about tabular object model definitions and syntax.

Analysis Services Management Objects (AMO) Developer reference documentation for the managed provider, Analysis Services Management Objects (AMO), for data definition and administration, including processing.

ADOMD.NET Developer reference documentation for the managed provider, ADOMD.NET, used for programmatic data access and query workloads.

Analysis Services Schema Rowsets Describes the schema rowsets that provide information about server state, server operations, and database objects.

XML for Analysis (XMLA) Reference Describes XMLA concepts that can help you understand how XMLA contributes to your custom solution. It also describes the level of compliance with the XMLA 1.1 specification.

Analysis Services Scripting Language (ASSL for XMLA) Describes the ASSL extensions to XMLA. ASSL provides a data definition and manipulation language for Analysis Services multidimensional models that supplements the XMLA specification.

Tabular Model Scripting Language (TMSL) Reference TMSL is a JSON representation of Tabular models at compatibility level 1200 and higher. Object definitions are based on tabular metadata constructs like table, column, and relationship rather than multidimensional metadata that might be unfamiliar if you are new to Analysis Services data modeling in Tabular mode.

Analysis Services PowerShell Reference Documents the cmdlets used for administrative functions, plus the general-purpose **Invoke-ASCmd** cmdlet that accepts any script or query as input.

See Also

Technical Reference Query and Expression Language Reference (Analysis Services)

Tabular Model Programming for Compatibility Level 1200 and higher

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Beginning with compatibility level 1200, Tabular metadata is used to describe model constructs, replacing historical Multidimensional metadata as descriptors for Tabular model objects. Metadata for tables, columns, and relationships are table, column, and relationship, rather than the Multidimensional equivalents (dimension and attribute).

You can create new models at compatibility level 1200 or higher by using the Microsoft.AnalysisServices.Tabular APIs, the latest version of SQL Server Data Tools (SSDT), or by changing the **CompatibilityLevel** of an existing Tabular model to upgrade it (also done in SSDT). Doing so binds the model to newer versions of the server, tools, and programming interfaces.

Upgrading an existing Tabular solution is recommended but not required. Existing script and custom solutions that access or manage Tabular models or databases can be used as-is. Earlier compatibility levels are fully supported in SQL Server 2016 using the features available at that level. Azure Analysis Services supports compatibility level 1200 and higher only.

New Tabular models will require different code and script, summarized below.

Object Model Definitions as Tabular Metadata Constructs

The Tabular Object Model for 1200 or higher models is exposed in JSON through the Tabular Model Scripting Language and through the AMO data definition language through a new namespace, Microsoft.AnalysisServices.Tabular

Script for Tabular Models and Databases

TMSL is a JSON scripting language for Tabular models, with support for create, read, update, an delete operations. You can refresh data via TMSL and invoke database operations for attach, detatch, backup, restore, and synchronize.

AMO PowerShell accepts TMSL script as input.

See Tabular Model Scripting Language (TMSL) Reference and Analysis Services PowerShell Reference for more information.

Query Languages

DAX and MDX are supported for all Tabular models.

Expression Language

Filters and expressions used to create calculated objects, including measures and KPIs, are formulated in DAX. See Understanding DAX in Tabular Models and Data Analysis Expressions (DAX) in Analysis Services.

Managed Code for Tabular Models and Databases

AMO includes a new namespace, Microsoft.AnalysisServices.Tabular, for working with models programmatically. See Microsoft.AnalysisServices Namespace for more information.

> **NOTE**
>
> Analysis Services Management Objects (AMO), ADOMD.NET, and Tabular Object Model (TOM) client libraries now target the .NET 4.0 runtime.

See Also

Analysis Services Developer Documentation
Tabular Model Programming for Compatibility Levels 1050 through 1103
Technical Reference Upgrade Analysis Services
Compatibility levels of Tabular models and databases

Tabular Model Programming for Compatibility Levels 1050 through 1103

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Tabular models use relational constructs to model the Analysis Services data used by analytical and reporting applications. These models run on an Analysis Service instance that is configured for tabular mode, using an in-memory analytics engine for storage and fast table scans that aggregate and calculate data as it is requested.

If the requirements of your custom BI solution are best met by a tabular model database, you can use any of the Analysis Services client libraries and programming interfaces to integrate your application with tabular models on an Analysis Services instance. To query and calculate tabular model data, you can use either embedded MDX or DAX in your code.

For Tabular models created in earlier versions of Analysis Services, in particular models at compatibility levels 1050 through 1103, the objects you work with programmatically in AMO, ADOMD.NET, XMLA, or OLE DB are fundamentally the same for both tabular and multidimensional solutions. Specifically, the object metadata defined for multidimensional models is also used for tabular model compatibility levels 1050-1103.

Beginning with SQL Server 2016, Tabular models can be built or upgraded to the 1200 or higher compatibility level, which uses tabular metadata to define the model. Metadata and programmability are fundamentally different at this level. See Tabular Model Programming for Compatibility Level 1200 and higher and Upgrade Analysis Services for more information.

In This Section

CSDL Annotations for Business Intelligence (CSDLBI)

Understanding the Tabular Object Model at Compatibility Levels 1050 through 1103

Technical Reference for BI Annotations to CSDL

IMDEmbeddedData Interface

ASSL Objects and Object Characteristics

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Objects in Analysis Services Scripting Language (ASSL) follow specific guidelines in regards to object groups, inheritance, naming, expansion, and processing.

Object Groups

All Microsoft SQL Server Analysis Services objects have an XML representation. The objects are divided into two groups:

Major objects
Major objects can be independently created, altered, and deleted. Major objects include:

- Servers
- Databases
- Dimensions
- Cubes
- Measure groups
- Partitions
- Perspectives
- Mining models
- Roles
- Commands associated with a server or database
- Data sources

Major objects have the following properties to track their history and status.

- **CreatedTimestamp**
- **LastSchemaUpdate**
- **LastProcessed** (where appropriate)

> **NOTE**
> The classification of an object as a major object affects how an instance of Analysis Services treats that object and how that object is handled in the object definition language. However, this classification does not guarantee that Analysis Services management and development tools will allow the independent creation, modification, or deletion of these objects.

Minor objects
Minor objects can only be created, altered, or deleted as part of creating, altering, or deleting the parent major object. Minor objects include:

- Hierarchies and levels
- Attributes
- Measures
- Mining model columns
- Commands associated with a cube
- Aggregations

Object Expansion

The **ObjectExpansion** restriction can be used to control the degree of expansion of ASSL XML returned by the server. This restriction has the options listed in the following table.

ENUMERATION VALUE	ALLOWED FOR <ALTER>	DESCRIPTION
ReferenceOnly	no	Returns only the name, ID, and timestamp for the requested object and for all contained major objects recursively.
ObjectProperties	yes	Expands the requested object and minor contained objects, but does not return major contained objects.
ExpandObject	no	Same as *ObjectProperties*, but also returns the name, ID, and timestamp for contained major objects.
ExpandFull	yes	Fully expands the requested object and all contained objects recursively.

This ASSL reference section describes the *ExpandFull* representation. All other **ObjectExpansion** levels are derived from this level.

Object Processing

ASSL includes read-only elements or properties (for example, **LastProcessed**) that can be read from the Analysis Services instance, but which are omitted when command scripts are submitted to the instance. Analysis Services ignores modified values for read-only elements without warning or error.

Analysis Services also ignores inappropriate or irrelevant properties without raising validation errors. For example, the X element should only be present when the Y element has a particular value. The Analysis Services instance ignores the X element instead of validating that element against the value of the Y element.

IMDEmbeddedData Interface

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The IMDEmbeddedData interface is a public interface used to manage an embedded Power Pivot database or a tabular model database. The interface inherits from the **IPersistStream** interface. The interface allows for the following operations:

- Get an identifier to the embedded stream in the container document.
- Set the URL of the containing document.
- Set a flag to indicate if the embedding application is in a hosted environment.
- Set the path to temporary files used by the embedding application.
- Cancel the current embedded operation.
- Get the estimated size (in bytes) of the stream to save the embedded object. Inherited from **IPersistStream**.
- Verify if the embedded database has changed since it was last saved. Inherited from **IPersistStream**.
- Load the embedded database to the local or in-process engine. Inherited from **IPersistStream**.
- Save the local or in-process database to the embedded stream in the container document. Inherited from **IPersistStream**.

Reference

The following reference documents the **IMDEmbeddedData** interface as presented in **msmd.h** header file.

Source file: PXOEmbeddedData.idl

```
[
  local,
  object,
  uuid(6B6691CF-5453-41c2-ADD9-4F320B7FD421),
  pointer_default(unique)
]
interface IMDEmbeddedData : IPersistStream
{
 [id(1), helpstring("Set flag indicating if the application is in a hosted environment")]
 HRESULT SetHosted(
  [in] BOOL in_fIsHosted);

 [id(2), helpstring("Set the URL for the document containing the embedded stream")]
 HRESULT SetContainerURL(
  [in] BSTR in_bstrURL);

 [id(3), helpstring("Get identifier used to look up embedded stream in container document")]
 HRESULT GetStreamIdentifier(
  [out, retval] BSTR* out_pbstrStreamId);

 [id(4), helpstring("Set the path used by the embedding application for temporary files")]
 HRESULT SetTempDirPath(
  [in]  BSTR in_bstrPath);

 [id(5), helpstring("Cancel the current operation")]
 HRESULT Cancel();
};
```

IMDEmbeddedData::GetStreamIdentifier

```
HRESULT GetStreamIdentifier (
    [out, retval] BSTR * out_pbstrStreamId
    )
```

Description

Gets the identifier used by the host application to the embedded stream in the container document.

Parameters

out_pbstrStreamId

Specifies the location of the stream identifier.

Return Value

S_OK

The stream identifier was successfully returned.

S_FALSE

There is no stream identifier.

E_FAIL

An error occurred accessing the stream identifier.

Remarks

To verify if the current connection contains an embedded database, the user should check the value of the DBPROP_MSMD_EMBEDDED_DATA property from the OLE DB connection properties.

The possible values for DBPROP_MSMD_EMBEDDED_DATA are:

NAME	VALUE	DEFINITION
DBPROPVAL_EMBED_NONE	0x00	No embedded database available

NAME	VALUE	DEFINITION
DBPROPVAL_EMBED_EMBEDDED	0x01	The current application contains the embedded database
DBPROPVAL_EMBED_LINKED	0x02	The embedded database is hosted in a remote application (i.e. SharePoint Server)

Source

```
[id(1), helpstring("Get identifier used to look up embedded stream in container document")]
 HRESULT GetStreamIdentifier(
  [out, retval] BSTR* out_pbstrStreamId);
```

IMDEmbeddedData::SetContainerURL

```
HRESULT SetContainerURL (
    [in] BSTR in_bstrURL
    )
```

Description

Sets the URL for the file containing the embedded stream.

Parameters

in_bstrURL

Specifies the URL for the containing document.

Return Value

S_OK

The container URL was successfully set.

E_FAIL

An error occurred while setting the container URL.

Source

```
[id(2), helpstring("Set the URL for the document containing the embedded stream")]
 HRESULT SetContainerURL(
  [in] BSTR in_bstrURL);
```

IMDEmbeddedData::SetHosted

```
HRESULT SetHosted (
    [in] BOOL in_fIsHosted
    )
```

Description

Set a flag to indicate if the embedding application is in a hosted environment.

Parameters

in_ftHosted

TRUE if caller is in a hosted in a service application (like IIS).

Return Value

S_OK

The flag was successfully set.

E_FAIL

An error occurred while setting the flag.

Source

```
[id(5), helpstring("Set flag indicating if the application is in a hosted environment")]
 HRESULT SetHosted(
  [in]  BOOL in_fIsHosted);
```

IMDEmbeddedData::SetTempDirPath

```
HRESULT SetTempDirPath (
    [in] BSTR in_bstrPath
    )
```

Description

Set the path to temporary files used by the embedding application.

Parameters

in_bstrPath

The path used by the host application for temporary files.

Return Value

S_OK

The temporary file directory was successfully set.

E_FAIL

An error occurred while setting the path.

Source

```
[id(4), helpstring("Set the path used by the host application for temporary files")]
 HRESULT SetTempDirPath(
  [in]  BSTR in_bstrPath);
```

IMDEmbeddedData::Cancel

```
HRESULT Cancel ( void )
```

Description

Cancels the current embedded database operation

Parameters

None.

Return Value

S_OK

The operation was successfully cancelled.

DB_E_CANTCANCEL

No cancellable operation is currently in progress.

E_FAIL

An error occurred while cancelling the embedded operation.

Source

```
[id(5), helpstring("Cancel the current operation")]
 HRESULT Cancel();
```

IMDEmbeddedData::GetSizeMax (IPersistStream::GetSizeMax)

```
HRESULT GetSizeMax (
    [out] ULARGE_INTEGER * out_pcbSize
    )
```

Description

Gets the estimated size (in bytes) of the stream to save the embedded object. Inherited from **IPersistStream**.

Parameters

in_bstrPath

The estimated size (in bytes) of the embedded database image.

Return Value

S_OK

The size was successfully obtained.

E_FAIL

An error occurred while obtaining the size.

IMDEmbeddedData::IsDirty (IPersistStream::IsDirty)

```
HRESULT IsDirty ( void )
```

Description

Verifies if the embedded database has changed since it was last saved. Inherited from **IPersistStream**.

Parameters

none

Return Value(s)

S_OK

The database has changed since it was last saved.

S_FALSE

The database has not changed since it was last saved.

E_FAIL

An error occurred while obtaining the database state.

IMDEmbeddedData::Load (IPersistStream::Load)

```
HRESULT Load (
    [in] IStream * in_pStm
    )
```

Description

Loads the embedded database to the local or in-process engine. Inherited from **IPersistStream**.

Parameters

in_pStm

A pointer to a stream interface from where to load the embedded database.

Return Value(s)

S_OK

The database was successfully loaded.

E_OUTOFMEMORY
Not enough memory to load the database.

E_FAIL
An error occurred while loading the database, different than **E_OUTOFMEMORY**.

IMDEmbeddedData::Save (IPersistStream::Save)

```
HRESULT Save (
    [in] IStream * in_pStm,
    [in] BOOL in_fClearDirty
    )
```

Description

Saves the local or in-process database to the embedded stream in the container document. Inherited from **IPersistStream**.

Parameters

in_pStm
A pointer to a stream interface to where to save the embedded database.

in_fClearDirty
A flag that indicates whether the dirty flag should be cleared after this operation.

Return Value(s)

S_OK
The database was successfully saved.

STG_E_CANTSAVE
An error occurred while saving the database, different than **STG_E_MEDIUMFULL**.

STG_E_MEDIUMFULL
The database could not be saved because there is no space left on the storage device.

Connection Representation (Tabular)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The connection object defines the source of the data that populates the tabular model.

Connection Representation

The specification for the connection object follows the rules of OLE DB providers.

Connection in AMO

When using AMO to manage a tabular model database, the DataSource object in AMO matches one-to-one to the connection logical object.

The following code snippet shows how to create an AMO data source, or Connection object in a tabular model.

```
//Create an OLEDB connection string
StringBuilder SqlCnxStr = new StringBuilder();
SqlCnxStr.Append(String.Format("Data Source={0};" ,SQLServer.Text));
SqlCnxStr.Append(String.Format("Initial Catalog={0};", SQLDatabase.Text));
SqlCnxStr.Append(String.Format("Persist Security Info={0};", false));
SqlCnxStr.Append(String.Format("Integrated Security={0};", "SSPI"));
SqlCnxStr.Append(String.Format("Provider={0}", "SQLNCLI11"));
String DatasourceCnxString = SqlCnxStr.ToString();

//Verify connection string and connectivity
System.Data.OleDb.OleDbConnection OleDbCnx = new System.Data.OleDb.OleDbConnection(DatasourceCnxString);
try
{
    OleDbCnx.Open();
}
catch ()
{
    throw;
}
String newDataSourceName = (string.IsNullOrEmpty(DataSourceName.Text) ||
string.IsNullOrWhiteSpace(DataSourceName.Text)) ? SQLDatabase.Text : DataSourceName.Text;
AMO.DataSource newDatasource = newDatabase.DataSources.Add(newDataSourceName, newDataSourceName);
newDatasource.ConnectionString = DatasourceCnxString.Text;
if (impersonateServiceAccount.Checked)
    newDatasource.ImpersonationInfo = new
AMO.ImpersonationInfo(AMO.ImpersonationMode.ImpersonateServiceAccount);
else
{
    if (String.IsNullOrEmpty(impersonateAccountUserName.Text))
    {
        MessageBox.Show(String.Format("Account User Name cannot be blank when using 'ImpersonateAccount'
mode"), "AMO to Tabular message", MessageBoxButtons.OK, MessageBoxIcon.Information);
        return;
    }
    newDatasource.ImpersonationInfo = new AMO.ImpersonationInfo(AMO.ImpersonationMode.ImpersonateAccount,
impersonateAccountUserName.Text, impersonateAccountPassword.Text);
}
newDatasource.Update();
```

Tabular AMO 2012 sample

To have a better understanding on how to use AMO to create and manipulate connection representations, see source code in the Tabular AMO 2012 sample; specifically check in the following source file: Database.cs. The sample is available at Codeplex. Sample code is provided only as a support to the logical concepts explained here, and should not be used in a production environment.

Database Representation(Tabular)

11/28/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In Tabular mode, the database is the container for all objects in the tabular model.

Database Representation

The database is the place where all objects that form a tabular model reside. Contained by the database, the developer finds objects like connections, tables, roles and many more.

Database in AMO

When using AMO to manage a tabular model database, the Database object in AMO matches one-to-one the database logical object in a tabular model.

> **NOTE**
>
> In order to gain access to a database object, in AMO, the user needs to have access to a server object and connect to it.

Database in ADOMD.Net

When using ADOMD to consult and query a tabular model database, connection to a specific database is obtained through the AdomdConnection object.

You can connect directly to a certain database using the following code snippet:

```
using ADOMD = Microsoft.AnalysisServices.AdomdClient;
...
   ADOMD.AdomdConnection currrentCnx = new ADOMD.AdomdConnection("Data Source=<<server\instance>>;Catalog=
<<database>>");
   currrentCnx.Open();
...
```

Also, over an existing connection object (that hasn't been closed), you can change the current database to another as shown in the following code snippet:

```
currentCnx.ChangeDatabase("myOtherDatabase");
```

Database in AMO

When using AMO to manage a database object, start with a Server object. Then search for your database in the databases collection or create a new database by adding one to the collection.

The following code snippet shows the steps to connect to a server and create an empty database, after checking the database doesn't exist:

```
AMO.Server CurrentServer = new AMO.Server();
try
{
    CurrentServer.Connect(currentServerName);
}
catch (Exception cnxException)
{
    MessageBox.Show(string.Format("Error while trying to connect to server: [{0}]\nError message: {1}",
currentServerName, cnxException.Message), "AMO to Tabular message", MessageBoxButtons.OK,
MessageBoxIcon.Error);
    return;
}
newDatabaseName = DatabaseName.Text;
if (CurrentServer.Databases.Contains(newDatabaseName))
{
    return;
}
try
{
    AMO.Database newDatabase = CurrentServer.Databases.Add(newDatabaseName);

    CurrentServer.Update();
}
catch (Exception createDBxc)
{
    MessageBox.Show(String.Format("Database [{0}] couldn't be created.\n{1}", newDatabaseName,
createDBxc.Message), "AMO to Tabular message", MessageBoxButtons.OK, MessageBoxIcon.Error);
    newDatabaseAvailable = false;
}
```

For a practical understanding on how to use AMO to create and manipulate database representations, see source code in the Tabular AMO 2012 sample; specifically check in the following source file: Database.cs. The sample is available at Codeplex. Sample code is provided only as a support to the logical concepts explained here, and should not be used in a production environment.

Perspective Representation (Tabular)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A perspective is a mechanism to simplify or focus the model into a smaller portion of it for the client application.

See Perspective Representation (Tabular) for a detailed explanation on how to create and manipulate the perspective representation.

> **WARNING**
>
> Perspectives are not a security mechanism; objects outside the perspective can still be accessed by the user through other interfaces.

Perspective Representation

In terms of AMO objects a perspective representation has a one to one mapping relationship with Perspective and no other main AMO objects are required.

Perspective in AMO

The following code snippet shows how to create a perspective in a Tabular model. The key element in this piece of code is the perspectiveElements; this object is a graphical representation of all the objects in the tabular model that are exposed to the user. *perspectiveElements* contains 4 columns and for this scenario only columns 1, 2 and 3 are relevant. Column 1 contains the type of element displayed -elementTypeValue-; column 2 contains the complete name of the element --, that probably will need to parsed to enter the element in the perspective; column 3 contains a checkbox item -checkedElement- that tells if the element is part of the perspective or not.

```
private void updatePerspective_Click(
                     AMO.Cube modelCube
                  ,  DataGridView perspectiveElements
                  ,  string updatedPerspectiveID
             )
{
    //Update is done by delete first, create new and insert after
    //if perspective doesn't exist then create first and insert after
    updatedPerspectiveID = updatedPerspectiveID.Trim();
    if (modelCube.Perspectives.Contains(updatedPerspectiveID))
    {
        modelCube.Perspectives.Remove(updatedPerspectiveID, true);
        newDatabase.Update(AMO.UpdateOptions.ExpandFull, AMO.UpdateMode.UpdateOrCreate);
    }
    AMO.Perspective currentPerspective = modelCube.Perspectives.Add(updatedPerspectiveID,
updatedPerspectiveID);

    foreach (DataGridViewRow currentDGVRow in perspectiveElements.Rows)
    {
        bool checkedElement = (bool)currentDGVRow.Cells[3].Value;
        if (checkedElement)
        {
            string elementTypeValue = currentDGVRow.Cells[1].Value.ToString();
            string elementNameValue = currentDGVRow.Cells[2].Value.ToString();
            switch (elementTypeValue)
            {
                case "Column: Attribute":
```

```
                case "Column: Calculated Column":
                    {
                        string perspectiveDimensionName = Regex.Match(elementNameValue, @"(?<=')(.*?)(?
=')").ToString();
                        string perspectiveDimensionAttributeName = Regex.Match(elementNameValue, @"(?<=\[)
(.*?)(?=\])").ToString();
                        AMO.PerspectiveDimension currentPerspectiveDimension;
                        if (!currentPerspective.Dimensions.Contains(perspectiveDimensionName))
                        {
                            currentPerspectiveDimension =
currentPerspective.Dimensions.Add(perspectiveDimensionName);
                        }
                        else
                        {
                            currentPerspectiveDimension =
currentPerspective.Dimensions[perspectiveDimensionName];
                        }
                        if
(!currentPerspectiveDimension.Attributes.Contains(perspectiveDimensionAttributeName))
                        {
                            currentPerspectiveDimension.Attributes.Add(perspectiveDimensionAttributeName);
                        }
                    }
                    break;
                case "Hierarchy":
                    {
                        string perspectiveDimensionName = Regex.Match(elementNameValue, @"(?<=')(.*?)(?
=')").ToString();
                        string perspectiveDimensionHierarchyName = Regex.Match(elementNameValue, @"(?<=\[)
(.*?)(?=\])").ToString();
                        AMO.PerspectiveDimension currentPerspectiveDimension;
                        if (!currentPerspective.Dimensions.Contains(perspectiveDimensionName))
                        {
                            currentPerspectiveDimension =
currentPerspective.Dimensions.Add(perspectiveDimensionName);
                        }
                        else
                        {
                            currentPerspectiveDimension =
currentPerspective.Dimensions[perspectiveDimensionName];
                        }
                        if
(!currentPerspectiveDimension.Hierarchies.Contains(perspectiveDimensionHierarchyName))
                        {
                            currentPerspectiveDimension.Hierarchies.Add(perspectiveDimensionHierarchyName);
                        }
                    }
                    break;
                case "Measure":
                    {
                        string perspectiveMeasureGroupName = Regex.Match(elementNameValue, @"(?<=')(.*?)(?
=')").ToString();
                        string measureName = Regex.Match(elementNameValue, @"(?<=\[)(.*?)(?=\])").ToString();
                        string perspectiveMeasureName = string.Format("[Measures].[{0}]", measureName);
                        AMO.PerspectiveCalculation currentPerspectiveCalculation = new
AMO.PerspectiveCalculation(perspectiveMeasureName, AMO.PerspectiveCalculationType.Member);
                        if (!currentPerspective.Calculations.Contains(perspectiveMeasureName))
                        {
                            currentPerspective.Calculations.Add(currentPerspectiveCalculation);
                        }
                        if
(modelCube.MdxScripts["MdxScript"].CalculationProperties.Contains(string.Format("KPIs.[{0}]", measureName)))
                        {//Current Measure is also a KPI ==> will be added to the KPIs collection of the
perspective
                            AMO.PerspectiveKpi currentPerspectiveKpi = new
AMO.PerspectiveKpi(perspectiveMeasureName);
                            if (!currentPerspective.Kpis.Contains(perspectiveMeasureName))
                            {
                                currentPerspective.Kpis.Add(currentPerspectiveKpi);
```

```
                            }
                        }
                    }
                    break;
                default:
                    break;
            }
        }
    }

    newDatabase.Update(AMO.UpdateOptions.ExpandFull, AMO.UpdateMode.CreateOrReplace);
    MessageBox.Show(String.Format("Perpective successfully updated."), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Information);
}
```

AMO2Tabular sample

However, to have an understanding on how to use AMO to create and manipulate perspective representations see the source code of the AMO to Tabular sample. The sample is available at Codeplex. An important note about the code: the code is provided only as a support to the logical concepts explained here and should not be used in a production environment; nor should it be used for other purpose other than the pedagogical one.

Querying a Tabular Model

5/16/2018 • 3 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

As a developer querying a tabular model means to retrieve data from the tabular database; to accomplish this goal you have two options: use table queries in DAX, or use MDX and retrieve the data as it were coming from a cube. However, depending on the underlying mode of your tabular model you might be restricted to use only DAX table queries; DirectQuery mode requires the usage of DAX table queries.

Querying with ADOMD.Net

Using ADOMD.Net to query a tabular model is simple and flexible; you can either send MDX statements or tabular query expressions from DAX to the server to get your results.

The following code shows how to pass your query statements to a tabular model and receive your results

```
//Function: tabularQueryExecute(string qry, ADOMD.AdomdConnection cnx)
//   - arg: qry, the tabular query or MDX expression to be evaluated
//   - arg: cnx, an active and opened ADOMD connection to the database where 'qry' is to be evaluated
//
// This function takes a query or mdx expression -qry-, a connection -cnx-
// and runs the query it against a Tabular Model using ADOMD.net
//
// Important note:
//    If the MDX query contains more than 2 axes (ON COLUMNS, ON ROWS), each axis will come as a new column
//    If the (All) value of the members in any axis have not been defined, a blank cell is returned. This might
be misleading
//    if the model also has missing values... which are also represented with blank cells.
private DataTable tabularQueryExecute(string qry, ADOMD.AdomdConnection cnx)
{
    ADOMD.AdomdDataAdapter currentDataAdapter = new ADOMD.AdomdDataAdapter(qry, cnx);
    DataTable tabularResults = new DataTable();
    currentDataAdapter.Fill(tabularResults);
    return tabularResults;
}
```

Example

If the above code is used with the following MDX expression:

```
SELECT { [Measures].[Internet Total Sales], [Measures].[Reseller Total Sales], [Measures].[Total Sales] } ON
COLUMNS
     , NON EMPTY [Product Category].[Product Category Name].MEMBERS ON ROWS "
     , NON EMPTY [Date].[Calendar Year].members ON 2 \n"
FROM [Model]
```

Against the sample model 'Adventure Works DW Tabular Denali CTP3' you should receive the following values as the resulting table:

CALENDAR YEAR	PRODUCT CATEGORY NAME	INTERNET TOTAL SALES	RESELLER TOTAL SALES	TOTAL SALES

CALENDAR YEAR	PRODUCT CATEGORY NAME	INTERNET TOTAL SALES	RESELLER TOTAL SALES	TOTAL SALES
		$ 29,358,677.22	$ 80,450,596.98	$ 109,809,274.20
	Accessories	$ 700,759.96	$ 571,297.93	$ 1,272,057.89
	Bikes	$ 28,318,144.65	$ 66,302,381.56	$ 94,620,526.21
	Clothing	$ 339,772.61	$ 1,777,840.84	$ 2,117,613.45
	Components		$ 11,799,076.66	$ 11,799,076.66
2001		$ 3,266,373.66	$ 8,065,435.31	$ 11,331,808.96
2001	Accessories		$ 20,235.36	$ 20,235.36
2001	Bikes	$ 3,266,373.66	$ 7,395,348.63	$ 10,661,722.28
2001	Clothing		$ 34,376.34	$ 34,376.34
2001	Components		$ 615,474.98	$ 615,474.98
2002		$ 6,530,343.53	$ 24,144,429.65	$ 30,674,773.18
2002	Accessories		$ 92,735.35	$ 92,735.35
2002	Bikes	$ 6,530,343.53	$ 19,956,014.67	$ 26,486,358.20
2002	Clothing		$ 485,587.15	$ 485,587.15
2002	Components		$ 3,610,092.47	$ 3,610,092.47
2003		$ 9,791,060.30	$ 32,202,669.43	$ 41,993,729.72
2003	Accessories	$ 293,709.71	$ 296,532.88	$ 590,242.59
2003	Bikes	$ 9,359,102.62	$ 25,551,775.07	$ 34,910,877.69
2003	Clothing	$ 138,247.97	$ 871,864.19	$ 1,010,112.16
2003	Components		$ 5,482,497.29	$ 5,482,497.29
2004		$ 9,770,899.74	$ 16,038,062.60	$ 25,808,962.34
2004	Accessories	$ 407,050.25	$ 161,794.33	$ 568,844.58
2004	Bikes	$ 9,162,324.85	$ 13,399,243.18	$ 22,561,568.03
2004	Clothing	$ 201,524.64	$ 386,013.16	$ 587,537.80
2004	Components		$ 2,091,011.92	$ 2,091,011.92

If the MDX expression is replaced with this DAX table query expression:

```
DEFINE
   MEASURE 'Product Category'[Internet Sales] = SUM( 'Internet Sales'[Sales Amount])
   MEASURE 'Product Category'[Reseller Sales] = SUM('Reseller Sales'[Sales Amount]) \n"
   EVALUATE ADDCOLUMNS('Product Category', \"Internet Sales - All Years\", 'Product Category'[Internet Sales],
\"Reseller Sales - All Years\", 'Product Category'[Reseller Sales])
```

And sent to the server using the above sample code, against the sample model 'Adventure Works DW Tabular Denali CTP3' you should receive the following values as the resulting table:

PRODUCT CATEGORY ID	PRODUCT CATEGORY ALTERNATE ID	PRODUCT CATEGORY NAME	INTERNET SALES	RESELLER SALES
4	4	Accessories	$ 700,759.96	$ 571,297.93
1	1	Bikes	$ 28,318,144.65	$ 66,302,381.56
3	3	Clothing	$ 339,772.61	$ 1,777,840.84
2	2	Components		$ 11,799,076.66

Relationship Representation (Tabular)

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A relationship is a connection between two tables of data. The relationship establishes how the data in the two tables should be correlated.

See Relationship Representation (Tabular) for a detailed explanation on how to create and manipulate the relationship representation.

Relationship Representation

In tabular models, multiple relationships can be defined between two tables. When multiple relationships, between two tables, are defined only one can be defined as the default relationship for the model and it is named as the Active relationship; all other relationships are named as Inactive.

Relationship in AMO

In terms of AMO objects all inactive relationships have a representation of a one to one mapping relationship with Relationship and no other main AMO objects are required; for the active relationship other requirements exist and a mapping to the ReferenceMeasureGroupDimension is also required.

The following code snippets show how to create a relationship in Tabular models, how to activate a relationship and how to define a primary key in a table (other than "RowNumber"). To create an active relationship a primary key need to be defined in the primary key table - PKTableName - of the relationship (the one side of the relationship), the sample shown here creates the primary key on the PKColumnName if no primary key is defined in this column. Inactive relationships can be created without the need of having a primary key on the primary key column.

```
private Boolean createRelationship(string PKTableName, string PKColumnName, string MVTableName, string
MVColumnName, AMO.Database tabularDb, string cubeName, Boolean forceActive)
{
    //verify input parameters
    if(     string.IsNullOrEmpty(PKTableName) || string.IsNullOrWhiteSpace(PKTableName)
        ||  string.IsNullOrEmpty(PKColumnName) || string.IsNullOrWhiteSpace(PKColumnName)
        ||  string.IsNullOrEmpty(MVTableName) || string.IsNullOrWhiteSpace(MVTableName)
        ||  string.IsNullOrEmpty(MVColumnName) || string.IsNullOrWhiteSpace(MVColumnName)
        ||  (tabularDb == null)
        ) return false;
    if(!tabularDb.Dimensions.ContainsName(PKTableName) || !tabularDb.Dimensions.ContainsName(MVTableName))
return false;
    if(!tabularDb.Dimensions[PKTableName].Attributes.ContainsName(PKColumnName) ||
!tabularDb.Dimensions[MVTableName].Attributes.ContainsName(MVColumnName)) return false;

    //Verify underlying cube structure
    if (!tabularDb.Cubes[cubeName].Dimensions.ContainsName(PKTableName) ||
!tabularDb.Cubes[cubeName].Dimensions.ContainsName(MVTableName)) return false; //Should return an exception!!
    if (!tabularDb.Cubes[cubeName].MeasureGroups.ContainsName(PKTableName) ||
!tabularDb.Cubes[cubeName].MeasureGroups.ContainsName(MVTableName)) return false; //Should return an
exception!!

    //Make sure PKTableName.PKColumnName  is set as PK ==> <attribute>.usage == AMO.AttributeUsage.Key
    if (tabularDb.Dimensions[PKTableName].Attributes[PKColumnName].Usage != AMO.AttributeUsage.Key)
    {
        //... here we are 'fixing', if there is an issue with PKTableName.PKColumnName not beeing the PK of
the table
```

```
        setPKColumn(tabularDb, PKTableName, PKColumnName);
    }

    //Terminology note:
    // -   the many side of the relationship is named the From end in AMO
    // -   the PK side of the relationship in named the To end in AMO
    //
    //It seems relationships flow FROM the many side of the relationship in TO the primary key side of the
relationship in AMO
    //
    //Verify the relationship we are creating does not exist, regardless of name.
    //if it exists, return true (no need to recreate it)
    //if it doesn't exists it will be created after this validation

    //
    foreach (AMO.Relationship currentRelationship in tabularDb.Dimensions[MVTableName].Relationships)
    {
        if ((currentRelationship.FromRelationshipEnd.Attributes[0].AttributeID == MVColumnName)
            && (currentRelationship.ToRelationshipEnd.DimensionID == PKTableName)
            && (currentRelationship.ToRelationshipEnd.Attributes[0].AttributeID == PKColumnName))
        {
            if (forceActive)
            {
                //Activate the relationship
                setActiveRelationship(tabularDb.Cubes[cubeName], MVTableName, MVColumnName, PKTableName,
currentRelationship.ID);
                //Update server with changes made here
                tabularDb.Update(AMO.UpdateOptions.ExpandFull, AMO.UpdateMode.CreateOrReplace);
                tabularDb.Cubes[cubeName].MeasureGroups[MVTableName].Process(AMO.ProcessType.ProcessFull);
            }
            return true;
        }
    }

    //A relationship like the one to be created does not exist; ergo, let's create it:

    //First, create the INACTIVE relationship definitions in the MultipleValues end of the relationship
    #region define unique name for relationship
    string newRelationshipID = string.Format("Relationship _{0}_{1}_ to _{2}_{3}_", MVTableName, MVColumnName,
PKTableName, PKColumnName);
    int rootLen = newRelationshipID.Length;
    for (int i = 0; tabularDb.Dimensions[MVTableName].Relationships.Contains(newRelationshipID); )
    {
        newRelationshipID = string.Format("{0}_{1,8:0}", newRelationshipID.Substring(0, rootLen), i);
    }
    #endregion
    AMO.Relationship newRelationship = tabularDb.Dimensions[MVTableName].Relationships.Add(newRelationshipID);

    newRelationship.FromRelationshipEnd.DimensionID = MVTableName;
    newRelationship.FromRelationshipEnd.Attributes.Add(MVColumnName);
    newRelationship.FromRelationshipEnd.Multiplicity = AMO.Multiplicity.Many;
    newRelationship.FromRelationshipEnd.Role = string.Empty;
    newRelationship.ToRelationshipEnd.DimensionID = PKTableName;
    newRelationship.ToRelationshipEnd.Attributes.Add(PKColumnName);
    newRelationship.ToRelationshipEnd.Multiplicity = AMO.Multiplicity.One;
    newRelationship.ToRelationshipEnd.Role = string.Empty;

    //Update server to create relationship
    tabularDb.Update(AMO.UpdateOptions.ExpandFull, AMO.UpdateMode.UpdateOrCreate);
    tabularDb.Dimensions[MVTableName].Process(AMO.ProcessType.ProcessDefault);
    tabularDb.Dimensions[PKTableName].Process(AMO.ProcessType.ProcessDefault);

    //Second, activate the relationship if relationship is to be set as the active relationship:
'forceActive==true'
    //... an inactive relationship needs only to be created on the dimensions object
    if (forceActive)
    {
        //Activate the relationship
        setActiveRelationship(tabularDb.Cubes[cubeName], MVTableName, MVColumnName, PKTableName,
```

```
newRelationshipID);
    }
    return true;
}

private void setActiveRelationship(AMO.Cube currentCube, string MVTableName, string MVColumnName, string
PKTableName, string relationshipID)
{
    if (!currentCube.MeasureGroups.Contains(MVTableName))
    {
        throw new AMO.AmoException(string.Format("Cube [{0}] does not contain Measure Group [{1}]\nError
activating relationship [{2}]: [{4}] <--- [{1}].[{3}]"
                                                , currentCube.Name, MVTableName, relationshipID, MVColumnName,
PKTableName));
    }
    AMO.MeasureGroup currentMG = currentCube.MeasureGroups[MVTableName];

    if (!currentMG.Dimensions.Contains(PKTableName))
    {
        AMO.ReferenceMeasureGroupDimension newReferenceMGDim = new AMO.ReferenceMeasureGroupDimension();
        newReferenceMGDim.CubeDimensionID = PKTableName;
        newReferenceMGDim.IntermediateCubeDimensionID = MVTableName;
        newReferenceMGDim.IntermediateGranularityAttributeID = MVColumnName;
        newReferenceMGDim.Materialization = AMO.ReferenceDimensionMaterialization.Regular;
        newReferenceMGDim.RelationshipID = relationshipID;
        foreach (AMO.CubeAttribute PKAttribute in currentCube.Dimensions[PKTableName].Attributes)
        {
            AMO.MeasureGroupAttribute PKMGAttribute =
newReferenceMGDim.Attributes.Add(PKAttribute.AttributeID);
            OleDbType PKMGAttributeType = PKAttribute.Attribute.KeyColumns[0].DataType;
            PKMGAttribute.KeyColumns.Add(new AMO.DataItem(PKTableName, PKAttribute.AttributeID,
PKMGAttributeType));
            PKMGAttribute.KeyColumns[0].Source = new AMO.ColumnBinding(PKTableName, PKAttribute.AttributeID);
        }
        currentMG.Dimensions.Add(newReferenceMGDim);

        AMO.ValidationErrorCollection errors = new AMO.ValidationErrorCollection();

        newReferenceMGDim.Validate(errors, true);
        if (errors.Count > 0)
        {
            StringBuilder errorMessages = new StringBuilder();
            foreach (AMO.ValidationError err in errors)
            {
                errorMessages.AppendLine(string.Format("At {2}: # {0} : {1}", err.ErrorCode,
err.FullErrorText, err.Source));
            }
            throw new AMO.AmoException(errorMessages.ToString());
        }
        //Update changes in the server
        currentMG.Update(AMO.UpdateOptions.ExpandFull, AMO.UpdateMode.CreateOrReplace);
    }
    else
    {
        AMO.ReferenceMeasureGroupDimension currentReferenceMGDim =
(AMO.ReferenceMeasureGroupDimension)currentMG.Dimensions[PKTableName];
        currentReferenceMGDim.RelationshipID = relationshipID;
        currentReferenceMGDim.IntermediateGranularityAttributeID = MVColumnName;
        //Update changes in the server
        currentMG.Update(AMO.UpdateOptions.ExpandFull, AMO.UpdateMode.UpdateOrCreate);
    }
    //process MG to activate relationship
    currentMG.Process(AMO.ProcessType.ProcessFull);

}

private void setPKColumn(AMO.Database tabularDb, string PKTableName, string PKColumnName)
{
        //Find all 'unwanted' Key attributes, remove their Key definitions and include the attributes in the
```

```
        //Find all "unwanted" Key attributes, remove their Key definitions and include the attributes in the
["RowNumber"].AttributeRelationships
        foreach (AMO.DimensionAttribute currentDimAttribute in tabularDb.Dimensions[PKTableName].Attributes)
        {
            if ((currentDimAttribute.Usage == AMO.AttributeUsage.Key) && (currentDimAttribute.ID !=
PKColumnName))
            {
                currentDimAttribute.Usage = AMO.AttributeUsage.Regular;
                if (currentDimAttribute.ID != "RowNumber")
                {
                    currentDimAttribute.KeyColumns[0].NullProcessing = AMO.NullProcessing.Preserve;
                    currentDimAttribute.AttributeRelationships.Clear();
                    if
(!tabularDb.Dimensions[PKTableName].Attributes["RowNumber"].AttributeRelationships.ContainsName(currentDimAttr
ibute.ID))
                    {
                        AMO.DimensionAttribute currentAttribute =
tabularDb.Dimensions[PKTableName].Attributes[currentDimAttribute.ID];
                        AMO.AttributeRelationship currentAttributeRelationship =
tabularDb.Dimensions[PKTableName].Attributes["RowNumber"].AttributeRelationships.Add(currentAttribute.ID);
                        currentAttributeRelationship.OverrideBehavior = AMO.OverrideBehavior.None;
                    }

tabularDb.Dimensions[PKTableName].Attributes["RowNumber"].AttributeRelationships[currentDimAttribute.ID].Cardi
nality = AMO.Cardinality.Many;
                }
            }
        }

        //Remove PKColumnName from ["RowNumber"].AttributeRelationships
        int PKAtribRelationshipPosition =
tabularDb.Dimensions[PKTableName].Attributes["RowNumber"].AttributeRelationships.IndexOf(PKColumnName);
        if (PKAtribRelationshipPosition != -1)
tabularDb.Dimensions[PKTableName].Attributes["RowNumber"].AttributeRelationships.RemoveAt(PKAtribRelationshipP
osition, true);

        //Define PKColumnName as Key and add ["RowNumber"] to PKColumnName.AttributeRelationships with
cardinality of One
        tabularDb.Dimensions[PKTableName].Attributes[PKColumnName].Usage = AMO.AttributeUsage.Key;
        tabularDb.Dimensions[PKTableName].Attributes[PKColumnName].KeyColumns[0].NullProcessing =
AMO.NullProcessing.Error;
        if
(!tabularDb.Dimensions[PKTableName].Attributes[PKColumnName].AttributeRelationships.ContainsName("RowNumber"))

        {
            AMO.DimensionAttribute currentAttribute =
tabularDb.Dimensions[PKTableName].Attributes["RowNumber"];
            AMO.AttributeRelationship currentAttributeRelationship =
tabularDb.Dimensions[PKTableName].Attributes[PKColumnName].AttributeRelationships.Add(currentAttribute.ID);
            currentAttributeRelationship.OverrideBehavior = AMO.OverrideBehavior.None;
        }

tabularDb.Dimensions[PKTableName].Attributes[PKColumnName].AttributeRelationships["RowNumber"].Cardinality =
AMO.Cardinality.One;

        //Update Table before going creating the relationship
        tabularDb.Update(AMO.UpdateOptions.ExpandFull, AMO.UpdateMode.UpdateOrCreate);
        tabularDb.Dimensions[PKTableName].Process(AMO.ProcessType.ProcessDefault);

}
```

AMO2Tabular sample

However, to have an understanding on how to use AMO to create and manipulate relationship representations see the source code of the AMO to Tabular sample. The sample is available at Codeplex. An important note about the

code: the code is provided only as a support to the logical concepts explained here and should not be used in a production environment; nor should it be used for other purpose other than the pedagogical one.

Tables - Calculated Column Representation

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A calculated column is DAX expression that creates a new column in a table and the obtained values are stored in the table. The calculated column expression is evaluated every time the table is processed.

Calculated Column Representation

Calculated Columns in AMO

When using AMO to manage a tabular model table, there is no one-to-one object match for a calculated column in AMO. A calculated column is represented by an attribute in Dimension and an attribute in MeasureGroup.

The following code snippet shows how to add a calculated column to an existing tabular model. The code assumes you have an AMO database object, newDatabase, and an AMO cube object, modelCube.

```
private void addCalculatedColumn(
                   AMO.Database newDatabase
                 , AMO.Cube modelCube
                 , String ccTableName
                 , String ccName
                 , String newCalculatedColumnExpression
             )
{
    if (string.IsNullOrEmpty(ccName) || string.IsNullOrWhiteSpace(ccName))
    {
        MessageBox.Show(String.Format("Calculated Column name is not defined."), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    if (string.IsNullOrEmpty(newCalculatedColumnExpression) ||
string.IsNullOrWhiteSpace(newCalculatedColumnExpression))
    {
        MessageBox.Show(String.Format("Calculated Column expression is not defined."), "AMO to Tabular
message", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    if (newDatabase.Dimensions[ccTableName].Attributes.Contains(ccName))
    {
        MessageBox.Show(String.Format("Calculated Column name already defined."), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    //Add CC attribute to the Dimension
    AMO.Dimension dim = newDatabase.Dimensions[ccTableName];
    AMO.DimensionAttribute currentAttribute = dim.Attributes.Add(ccName, ccName);
    currentAttribute.Usage = AMO.AttributeUsage.Regular;
    currentAttribute.KeyUniquenessGuarantee = false;

    currentAttribute.KeyColumns.Add(new AMO.DataItem(ccTableName, ccName, OleDbType.Empty));
    currentAttribute.KeyColumns[0].Source = new AMO.ExpressionBinding(newCalculatedColumnExpression);
    currentAttribute.KeyColumns[0].NullProcessing = AMO.NullProcessing.Preserve;
    currentAttribute.NameColumn = new AMO.DataItem(ccTableName, ccName, System.Data.OleDb.OleDbType.WChar);
    currentAttribute.NameColumn.Source = new AMO.ExpressionBinding(newCalculatedColumnExpression);
    currentAttribute.NameColumn.NullProcessing = AMO.NullProcessing.ZeroOrBlank;

    currentAttribute.OrderBy = AMO.OrderBy.Key;
```

```
    currentAttribute.OrderBy = AMO.OrderBy.Key;
    AMO.AttributeRelationship currentAttributeRelationship =
dim.Attributes["RowNumber"].AttributeRelationships.Add(currentAttribute.ID);
    currentAttributeRelationship.Cardinality = AMO.Cardinality.Many;
    currentAttributeRelationship.OverrideBehavior = AMO.OverrideBehavior.None;

    //Add CC as attribute to the MG
    AMO.MeasureGroup mg = modelCube.MeasureGroups[ccTableName];
    AMO.DegenerateMeasureGroupDimension currentMGDim =
(AMO.DegenerateMeasureGroupDimension)mg.Dimensions[ccTableName];
    AMO.MeasureGroupAttribute mga = new AMO.MeasureGroupAttribute(ccName);

    mga.KeyColumns.Add(new AMO.DataItem(ccTableName, ccName, OleDbType.Empty));
    mga.KeyColumns[0].Source = new AMO.ExpressionBinding(newCalculatedColumnExpression);

    currentMGDim.Attributes.Add(mga);

    try
    {
        //Update Dimension, CubeDimension and MG in server
        newDatabase.Update(AMO.UpdateOptions.ExpandFull, AMO.UpdateMode.UpdateOrCreate);
    }
    catch (AMO.OperationException amoOpXp)
    {
        MessageBox.Show(String.Format("Calculated Column expression contains syntax errors, or references
undefined or missspelled elements.\nError message: {0}", amoOpXp.Message), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }
    catch (AMO.AmoException amoXp)
    {
        MessageBox.Show(String.Format("AMO exception accessing the server.\nError message: {0}",
amoXp.Message), "AMO to Tabular message", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }
    catch (Exception)
    {
        throw;
    }
}
```

Tables - Calculated Measure Representation

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A calculated measure is a named DAX expression evaluated every time it is used.

Calculated Measure Representation

Calculated Measure in AMO

When using AMO to manage a tabular model calculated measure, there is a one-to-one match between the logical Calculated Measure object and a measure defined in a Command object of the MdxScript object. Each **Calculated Measure** is defined as a **CREATE MEASURE** expression inside a Command object and separated by a semicolon. All calculated measures in a tabular model correspond to the collection **CREATE MEASURE** string in one command object in a MdxScript object. For each calculated measure, there is one-to-one mapping with a CalculationProperty.

The following code snippet shows how to create a calculated measure.

```
private void addCalculatedMeasure(
                   AMO.Cube modelCube
                 , string cmTableName
                 , string cmName
                 , string newCalculatedMeasureExpression
             )
{
    //Verify input requirements
    if (string.IsNullOrEmpty(cmName) || string.IsNullOrWhiteSpace(cmName))
    {
        MessageBox.Show(String.Format("Calculated Measure name is not defined."), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    if (string.IsNullOrEmpty(newCalculatedMeasureExpression) ||
string.IsNullOrWhiteSpace(newCalculatedMeasureExpression))
    {
        MessageBox.Show(String.Format("Calculated Measure expression is not defined."), "AMO to Tabular
message", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    StringBuilder measuresCommand = new StringBuilder();

    AMO.MdxScript mdxScript = modelCube.MdxScripts["MdxScript"];

    //ToDo: Verify if measure already exits and ask user what wants to do next

    if (mdxScript.Commands.Count == 1)
    {
        measuresCommand.AppendLine("------------------------------------------------------------");
        measuresCommand.AppendLine("-- Tabular Model measures command (do not modify manually) --");
        measuresCommand.AppendLine("------------------------------------------------------------");
        measuresCommand.AppendLine();
        measuresCommand.AppendLine();
        mdxScript.Commands.Add(new AMO.Command(measuresCommand.ToString()));

    }
    else
```

```
    {
        measuresCommand.Append(mdxScript.Commands[1].Text);
    }
    measuresCommand.AppendLine(string.Format("CREATE MEASURE '{0}'[{1}]={2};", cmTableName, cmName,
newCalculatedMeasureExpression));

    mdxScript.Commands[1].Text = measuresCommand.ToString();

    if (!mdxScript.CalculationProperties.Contains(cmName))
    {
        AMO.CalculationProperty cp = new AMO.CalculationProperty(cmName, AMO.CalculationType.Member);
        cp.FormatString = ""; // ToDo: Get formatting attributes for the member
        cp.Visible = true;
        mdxScript.CalculationProperties.Add(cp);
    }

    try
    {
        modelCube.Update(AMO.UpdateOptions.ExpandFull, AMO.UpdateMode.UpdateOrCreate);
        MessageBox.Show(String.Format("Calculated Measure successfully defined."), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    catch (AMO.OperationException amoOpXp)
    {
        MessageBox.Show(String.Format("Calculated Measure expression contains syntax errors, or references
undefined or missspelled elements.\nError message: {0}", amoOpXp.Message), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }
    catch (AMO.AmoException amoXp)
    {
        MessageBox.Show(String.Format("AMO exception accessing the server.\nError message: {0}",
amoXp.Message), "AMO to Tabular message", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }
    catch (Exception)
    {
        throw;
    }
}
```

Tables - Hierarchy Representation

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In tabular models a hierarchy is navigation path from one attribute to another based on values selected by the user.

Hierarchy Representation

Hierarchy in AMO

When using AMO to manage a tabular model table, there is a one-to-one object match for a hierarchy in AMO. A hierarchy is represented by the Hierarchy object.

The following code snippet shows how to add a hierarchy to an existing tabular model. The code assumes you have an AMO database object, newDatabase, and an AMO cube object, modelCube.

```
private void addHierarchy(
                    AMO.Database newDatabase
                 ,  AMO.Cube modelCube
                 ,  string tableName
                 ,  string hierarchyName
                 ,  string levelsText
             )
{
    //Validate input
    if (string.IsNullOrEmpty(hierarchyName) || string.IsNullOrEmpty(levelsText))
    {
        MessageBox.Show(String.Format("Hierarchy Name or Layout must be provided."), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    if (!overwriteHierarchy.Checked && newDatabase.Dimensions[tableName].Hierarchies.Contains(hierarchyName))
    {
        MessageBox.Show(String.Format("Hierarchy already exists.\nGive a new name or enable overwriting"), "AMO
to Tabular message", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    try
    {
        if (newDatabase.Dimensions[tableName].Hierarchies.Contains(hierarchyName))
        {
            //Hierarchy exists... deleting it to write it later
            newDatabase.Dimensions[tableName].Hierarchies.Remove(hierarchyName, true);
            newDatabase.Dimensions[tableName].Update(AMO.UpdateOptions.AlterDependents);
        }
        AMO.Hierarchy currentHierarchy = newDatabase.Dimensions[tableName].Hierarchies.Add(hierarchyName,
hierarchyName);
        currentHierarchy.AllMemberName = string.Format("(All of {0})", hierarchyName);
        //Parse hierarchyLayout
        using (StringReader levels = new StringReader(levelsText))
        {
            //Each line:
            //  must come with: The columnId of the attribute in the dimension --> this represents the
SourceAttributeID
            //  optional: A display name for the Level (if this argument doesn't come the default is the
SourceAttributeID)
            string line;
```

```
            while ((line = levels.ReadLine()) != null)
            {
                if (string.IsNullOrEmpty(line) || string.IsNullOrWhiteSpace(line)) continue;
                line = line.Trim();
                string[] hierarchyData = line.Split(',');
                if (string.IsNullOrEmpty(hierarchyData[0])) continue; //first argument cannot be empty or
blank,
                                                                      //assume is a blank line and ignore it
                string levelSourceAttributeID = hierarchyData[0].Trim();
                string levelID = (hierarchyData.Length > 1 && !string.IsNullOrEmpty(hierarchyData[1])) ?
hierarchyData[1].Trim() : levelSourceAttributeID;
                currentHierarchy.Levels.Add(levelID).SourceAttributeID = levelSourceAttributeID;
            }
        }
        newDatabase.Dimensions[tableName].Update(AMO.UpdateOptions.AlterDependents);

    }
    catch (Exception ex)
    {
        MessageBox.Show(String.Format("Error creating hierarchy [{0}].\nError message: {1}",
newHierarchyName.Text, ex.Message), "AMO to Tabular message", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        if (newDatabase.Dimensions[tableName].Hierarchies.Contains(hierarchyName))
        {
            //Hierarchy was created but exception prevented complete hierarchy to be written... deleting
incomplete hierarchy
            newDatabase.Dimensions[tableName].Hierarchies.Remove(hierarchyName, true);
            newDatabase.Dimensions[tableName].Update(AMO.UpdateOptions.AlterDependents);
        }

    }
    newDatabase.Dimensions[tableName].Process(AMO.ProcessType.ProcessFull);
    modelCube.MeasureGroups[tableName].Process(AMO.ProcessType.ProcessFull);
}
```

Tables - Key Performance Indicator Representation

11/28/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A KPI is used to gauge performance of a value, defined by a Base measure, against a Target value

Key Performance Indicator Representation

In tabular object models a key performance indicator -kpi- is a measure with additional information for the client application to display it graphically. A kpi usually has information about a goal to be obtained, the status of the measure compared to a the goal, and information for the client tool on how to display graphically the status.

Key Performance Indicator in AMO

When using AMO to manage a Tabular Model kpi there is no one-to-one object match for a kpi in AMO, the AMO Kpiobject is not used for this purpose; in AMO, for tabular models, a kpi is represented the by a series of objects created in one of the elements in the Commands collection and in the CalculationProperties.

The following code snippet shows how to create one of many possible kpi definitions.

```
private void addStaticKPI(object sender, EventArgs e)
{
    double KPIGoal = 0, status1ThresholdValue = 0, status2ThresholdValue = 0
        , redAreaValue = 0, yellowAreaValue = 0, greenAreaValue = 0;
    string clientStatusGraphicImageName = "Three Circles Colored";

    //Verify input requirements
    //Goal values
    if (staticTargetKPI.Checked)
    {//Static KPI Goal selected
        if (string.IsNullOrEmpty(staticTargetKPIGoal.Text)
            || string.IsNullOrWhiteSpace(staticTargetKPIGoal.Text)
            || !double.TryParse(staticTargetKPIGoal.Text, out KPIGoal))
        {
            MessageBox.Show(String.Format("Static Goal is not defined or is not a valid number."), "AMO to
Tabular message", MessageBoxButtons.OK, MessageBoxIcon.Error);
            return;
        }
    }
    else
    {//Measure KPI Goal selected
        if (!TargetMeasureForKPISelected)
        {
            MessageBox.Show(String.Format("Measure Goal is not selected."), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            return;
        }
    }

    //Status
    if (string.IsNullOrEmpty(firstStatusThresholdValue.Text)
        || string.IsNullOrWhiteSpace(firstStatusThresholdValue.Text)
        || !double.TryParse(firstStatusThresholdValue.Text, out status1ThresholdValue)
        || string.IsNullOrEmpty(secondStatusThresholdValue.Text)
        || string.IsNullOrWhiteSpace(secondStatusThresholdValue.Text)
        || !double.TryParse(secondStatusThresholdValue.Text, out status2ThresholdValue))
    {
        MessageBox.Show(String.Format("Status Threshold are not defined or they are not a valid number."), "AMO
```

```
to Tabular message", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    if (string.IsNullOrEmpty(statusValueRedArea.Text)
        || string.IsNullOrWhiteSpace(statusValueRedArea.Text)
        || !double.TryParse(statusValueRedArea.Text, out redAreaValue)
        || string.IsNullOrEmpty(statusValueYellowArea.Text)
        || string.IsNullOrWhiteSpace(statusValueYellowArea.Text)
        || !double.TryParse(statusValueYellowArea.Text, out yellowAreaValue)
        || string.IsNullOrEmpty(statusValueGreenArea.Text)
        || string.IsNullOrWhiteSpace(statusValueGreenArea.Text)
        || !double.TryParse(statusValueGreenArea.Text, out greenAreaValue))
    {
        MessageBox.Show(String.Format("Status Area values are not defined or they are not a valid number."),
"AMO to Tabular message", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    //Set working variables
    string kpiTableName = ((DataRowView)MeasuresInModelList.CheckedItems[0])[0].ToString();
    string kpiMeasureName = ((DataRowView)MeasuresInModelList.CheckedItems[0])[1].ToString();

    //Verify if KPI is already defined
    if (modelCube.MdxScripts["MdxScript"].CalculationProperties.Contains(string.Format("KPIs.[{0}]",
kpiMeasureName)))
    {
        //ToDo: Verify with the user if wants to update KPI or exit
        //If user wants to update then remove KPI from mdxScripts and continue with the creating the KPI
        //
        // Until the code to remove KPI is finished we'll have to exit with a message of no duplicated KPIs are
allowed
        MessageBox.Show(String.Format("Another KPI exists for the same measeure."), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    StringBuilder kpiCommand = new StringBuilder();

    AMO.MdxScript mdxScript = modelCube.MdxScripts["MdxScript"];
    kpiCommand.Append(mdxScript.Commands[1].Text);

    string goalExpression;
    if (staticTargetKPI.Checked)
    {//Static KPI Goal selected
        goalExpression = KPIGoal.ToString();
    }
    else
    {//Measure KPI Goal selected
        string measureGoalMeasureName = ((DataRowView)KpiTargetMeasures.CheckedItems[0])[1].ToString();
        goalExpression = string.Format("[Measures].[{0}]", measureGoalMeasureName);
    }

    kpiCommand.AppendLine(string.Format("CREATE MEMBER CURRENTCUBE.Measures.[_{1} Goal] AS '{2}',
ASSOCIATED_MEASURE_GROUP = '{0}';"
            , kpiTableName, kpiMeasureName, goalExpression));

    string statusExpression;
    if (staticTargetKPI.Checked)
    {//Static KPI Goal selected
        statusExpression = string.Format("KpiValue(\"{0}\")", kpiMeasureName).Trim();
    }
    else
    {//Measure KPI Goal selected
        string measureGoalMeasureName = ((DataRowView)KpiTargetMeasures.CheckedItems[0])[1].ToString().Trim();

        string M = string.Format("[Measures].[{0}]", kpiMeasureName);
        string T = string.Format("[Measures].[{0}]", measureGoalMeasureName);
```

```
        if (KpiRelationDifference.Checked)
        {
            statusExpression = string.Format("{1} - {0}", M, T);
        }
        else
        {
            if (KpiRelationRatioMT.Checked)
            {
                statusExpression = string.Format("{0} / {1}", M, T);
            }
            else
            {
                statusExpression = string.Format("{1} / {0}", M, T);
            }
        }
    }
    kpiCommand.AppendLine(string.Format("CREATE MEMBER CURRENTCUBE.Measures.[_{1} Status] "
                                            + " AS 'Case When IsEmpty({9}) Then Null "
                                                     + " When ({9}) {2} {3} Then {4} "
                                                     + " When ({9}) {5} {6} Then {7} "
                                                     + " Else {8} End'"
                                            + ", ASSOCIATED_MEASURE_GROUP = '{0}';"
        , kpiTableName, kpiMeasureName // 0, 1
        , statusThreshold1ComparisonOperator.Text, status1ThresholdValue, redAreaValue // 2, 3, 4
        , statusThreshold2ComparisonOperator.Text, status2ThresholdValue, yellowAreaValue, greenAreaValue // 5,
6, 7, 8
        , statusExpression // 9
        ));

    kpiCommand.AppendLine(string.Format("CREATE MEMBER CURRENTCUBE.Measures.[_{1} Trend] AS '0',
ASSOCIATED_MEASURE_GROUP = '{0}';"
        , kpiTableName, kpiMeasureName));

    kpiCommand.AppendLine(string.Format("CREATE KPI CURRENTCUBE.[{1}] AS Measures.[{1}]"
                                             + ", ASSOCIATED_MEASURE_GROUP = '{0}'"
                                             + ", GOAL = Measures.[_{1} Goal]"
                                             + ", STATUS = Measures.[_{1} Status]"
                                             + ", TREND = Measures.[_{1} Trend]"
                                             + ", STATUS_GRAPHIC = '{2}'"
                                             + ", TREND_GRAPHIC = '{2}';"
        , kpiTableName, kpiMeasureName, clientStatusGraphicImageName));

    {//Adding Calculation Reference for the Measure itself
        if (!mdxScript.CalculationProperties.Contains(kpiMeasureName))
        {
            AMO.CalculationProperty cp = new AMO.CalculationProperty(kpiMeasureName,
AMO.CalculationType.Member);
            cp.FormatString = ""; // ToDo: Get formatting attributes for the member
            cp.Visible = true;
            mdxScript.CalculationProperties.Add(cp);
        }
    }
    {//Adding Calculation Reference for the Goal measure
        AMO.CalculationProperty cp = new AMO.CalculationProperty(string.Format("Measures.[_{0} Goal]",
kpiMeasureName), AMO.CalculationType.Member);
        cp.FormatString = ""; // ToDo: Get formatting attributes for the member
        cp.Visible = false;
        mdxScript.CalculationProperties.Add(cp);
    }

    {//Adding Calculation Reference for the Status measure
        AMO.CalculationProperty cp = new AMO.CalculationProperty(string.Format("Measures.[_{0} Status]",
kpiMeasureName), AMO.CalculationType.Member);
        cp.FormatString = ""; // ToDo: Get formatting attributes for the member
        cp.Visible = false;
        mdxScript.CalculationProperties.Add(cp);
    }

    {//Adding Calculation Reference for the Status measure
```

```
        AMO.CalculationProperty cp = new AMO.CalculationProperty(string.Format("Measures.[_{0} Trend]",
kpiMeasureName), AMO.CalculationType.Member);
        cp.FormatString = ""; // ToDo: Get formatting attributes for the member
        cp.Visible = false;
        mdxScript.CalculationProperties.Add(cp);
    }

    {//Adding Calculation Reference for the KPI
        AMO.CalculationProperty cp = new AMO.CalculationProperty(string.Format("KPIs.[{0}]", kpiMeasureName),
AMO.CalculationType.Member);
        cp.FormatString = ""; // ToDo: Get formatting attributes for the member
        cp.Visible = true;
        mdxScript.CalculationProperties.Add(cp);
    }
    try
    {
        newDatabase.Update(AMO.UpdateOptions.ExpandFull, AMO.UpdateMode.UpdateOrCreate);
        MessageBox.Show(String.Format("KPI successfully defined."), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    catch (AMO.OperationException amoOperationException)
    {
        //ToDo: remove anything left in mdxScript up to the point where the exception was thrown
        MessageBox.Show(String.Format("Error creating KPI for Measure '{0}'[{1}]\nError message: {2}",
kpiTableName, kpiMeasureName, amoOperationException.Message), "AMO to Tabular message", MessageBoxButtons.OK,
MessageBoxIcon.Information);
    }

}
```

AMO2Tabular sample

To have an understanding on how to use AMO to create and manipulate Key Performance Indicator representations see the source code of the AMO to Tabular sample; specifically check in the following source file: AddKPIs.cs. The sample is available at Codeplex. An important note about the code: the code is provided only as a support to the logical concepts explained here and should not be used in a production environment; nor should it be used for other purpose other than the pedagogical one.

Tables - Partition Representation

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

For operational purposes, a table can be divided in different subsets of rows that when combined together form the table; each of those subsets is a partition of the table.

Partition Representation

In terms of AMO objects a partition representation has a one to one mapping relationship with Partition and no other main AMO objects are required

Partition in AMO

When using AMO to manage a partition in a you need to find the measure group that represents the Tabular Model table and work from there.

The following code snippet shows how to add a partition to an existing tabular model table.

```
private void AddPartition(
                      AMO.Cube modelCube
                  ,  AMO.DataSource newDatasource
                  ,  string mgName
                  ,  string newPartitionName
                  , string partitionSelectStatement
                  , Boolean processPartition
             )
{
    mgName = mgName.Trim();
    newPartitionName = newPartitionName.Trim();
    partitionSelectStatement = partitionSelectStatement.Trim();
    //Validate Input
    if (string.IsNullOrEmpty(newPartitionName) || string.IsNullOrWhiteSpace(newPartitionName))
    {
        MessageBox.Show(String.Format("Partition Name cannot be empty or blank"), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    if (modelCube.MeasureGroups[mgName].Partitions.ContainsName(newPartitionName))
    {
        MessageBox.Show(String.Format("Partition Name already defined. Duplicated names are not allowed"), "AMO
to Tabular message", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    if (string.IsNullOrEmpty(partitionSelectStatement) || string.IsNullOrWhiteSpace(partitionSelectStatement))
    {
        MessageBox.Show(String.Format("Select statement cannot be empty or blank"), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    //Input validated
    string partitionID = newPartitionName; //Using partition name as the ID
    AMO.Partition currentPartition = new AMO.Partition(partitionID, partitionID);
    currentPartition.StorageMode = AMO.StorageMode.InMemory;
    currentPartition.ProcessingMode = AMO.ProcessingMode.Regular;
    currentPartition.Source = new AMO.QueryBinding(newDatasource.ID, partitionSelectStatement);
    modelCube.MeasureGroups[mgName].Partitions.Add(currentPartition);
    try
    {
        modelCube.Update(AMO.UpdateOptions.ExpandFull, AMO.UpdateMode.UpdateOrCreate);
        if (processPartition)
        {
            modelCube.MeasureGroups[mgName].Partitions[partitionID].Process(AMO.ProcessType.ProcessFull);
            MessageBox.Show(String.Format("Partition successfully processed."), "AMO to Tabular message",
MessageBoxButtons.OK, MessageBoxIcon.Information);
        }

    }
    catch (AMO.AmoException amoXp)
    {
        MessageBox.Show(String.Format("AMO exception accessing the server.\nError message: {0}",
amoXp.Message), "AMO to Tabular message", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }
    catch (Exception)
    {
        throw;
    }
}
```

AMO2Tabular sample

However, to have an understanding on how to use AMO to create and manipulate partition representations see the source code of the AMO to Tabular sample. The sample is available at Codeplex. An important note about the code: the code is provided only as a support to the logical concepts explained here and should not be used in a production environment; nor should it be used for other purpose other than the pedagogical one.

Tables Representation (Tabular)

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

In tabular models, a table is the base representation of the data.

Tables Representation

Tables in AMO

When using AMO to manage a table, there is no one-to-one object match. In AMO, a table is represented by a Dimension and MeasureGroup. For a measure group to exist, a Cube must be defined to host the measure group. For a dimension, measure group, and cube to exist, a Data Source View object must be defined to hold the binding definitions to the data source.

From a procedural perspective, a Data Source View needs to be created before any other object is defined. The data source view object contains the mapping for all relevant objects in the data source. The mapping of the relational model is embedded in the data source view as a .Net DataSet object and stored in the Schema property of the DSV.

The following code snippet assumes you have a SQL client connection string, a dictionary of Select statements that map to all tables in the relational model you intend to represent in your tabular model, and a variable newDataSourceViewName with the name of the data source view (usually the name of your relational database).

```
DataSet newDataSourceViewDataSet = new DataSet(newDataSourceViewName);

Foreach( String tableName in listOfSqlStatements.Keys)
{
    String sqlStmt = listOfSqlStatements[tableName];

    DataTable newTable = new DataTable(tableName);

    using (SqlConnection SqlCnx = new SqlConnection(SqlCnxStr))
    {
        SqlDataAdapter dataAdapter = new SqlDataAdapter(sqlStmt, SqlCnx);
        dataAdapter.FillSchema(newTable, SchemaType.Source);
    }
    newDataSourceViewDataSet.Tables.Add(newTable);
}

AMO.DataSourceView newDatasourceView = newDatabase.DataSourceViews.AddNew(newDataSourceViewName,
newDataSourceViewName);
newDatasourceView.DataSourceID = newDatasource.ID; //This is the ID of the DataSource object
newDatasourceView.Schema = newDataSourceViewDataSet; //Here you are storing all the relational schema in the
DSV
newDatasourceView.Update();
```

Once the Data Source View has been created and updated, the cube object needs to be created, but not updated in the server until the first table is created. A cube object cannot be created empty. The following code snippet shows how to create a cube; the snippet assumes you have a non-empty string newCubeName with the name of the cube already validated for duplicates too.

```
modelCube = newDatabase.Cubes.Add(newCubeName, newCubeName);
modelCube.Source = new AMO.DataSourceViewBinding(newDatasourceView.ID);
modelCube.StorageMode = AMO.StorageMode.InMemory;
modelCube.Language = newDatabase.Language;
modelCube.Collation = newDatabase.Collation;
//Create initial MdxScript
AMO.MdxScript mdxScript = modelCube.MdxScripts.Add("MdxScript", "MdxScript");
StringBuilder initialCommand = new StringBuilder();
initialCommand.AppendLine("CALCULATE;");
initialCommand.AppendLine("CREATE MEMBER CURRENTCUBE.Measures.[__No measures defined] AS 1;");
initialCommand.AppendLine("ALTER CUBE CURRENTCUBE UPDATE DIMENSION Measures, Default_Member = [__No measures
defined];");
mdxScript.Commands.Add(new AMO.Command(initialCommand.ToString()));
```

Once the cube is defined locally, tables can be created and updated. The following procedure outlines the necessary steps to create a table:

1. Create the dimension that represents the table and don't update the server yet.
2. Create the RowNumber attribute and define it as the key attribute of the dimension.
3. Create dimension attributes and mark them with a one-to-many relationship to RowNumber.
4. Add dimension to cube dimensions.
5. Create the measure group that also represents the table.
6. Add dimension to measure group.
7. Create AMO default measure object to the measure group. Note, this is the only time an AMO measure object is used; calculated measures, in tabular models, are defined in the AMO MdxScripts["MdxScript"] object.
8. Create default partition.
9. Update database.

The following code snippet shows how to create a table:

```
private Boolean CreateTable(
                      AMO.Database db      //the AMO database object where dimension are created
                    , AMO.Cube cb          //the AMO cube where measure group is created
                    , DataTable dataTable  //the schema of the table to be created
                    )
{
    String tableID = dataTable.TableName;

    if (db.Dimensions.Contains(tableID))
    {
        if (cb.MeasureGroups.Contains(tableID))
        {
            cb.MeasureGroups[tableID].Measures.Clear();
            cb.MeasureGroups[tableID].Partitions.Clear();
            cb.MeasureGroups.Remove(tableID, true);
        }
        if (cb.Dimensions.Contains(tableID))
        {
            cb.Dimensions.Remove(tableID, true);
        }
        db.Dimensions.Remove(tableID);
    }
```

```
    #region Create Dimension
    //Define Dimension general properties
    AMO.Dimension currentDimension = db.Dimensions.AddNew(tableID, tableID);
    currentDimension.Source = new AMO.DataSourceViewBinding(newDatasourceView.ID);
    currentDimension.StorageMode = AMO.DimensionStorageMode.InMemory;
    currentDimension.UnknownMember = AMO.UnknownMemberBehavior.AutomaticNull;
    currentDimension.UnknownMemberName = "Unknown";
    currentDimension.ErrorConfiguration = new AMO.ErrorConfiguration();
    currentDimension.ErrorConfiguration.KeyNotFound = AMO.ErrorOption.IgnoreError;
    currentDimension.ErrorConfiguration.KeyDuplicate = AMO.ErrorOption.ReportAndStop;
    currentDimension.ErrorConfiguration.NullKeyNotAllowed = AMO.ErrorOption.ReportAndStop;
    currentDimension.Language = db.Language;
    currentDimension.Collation = db.Collation;
    currentDimension.ProactiveCaching = new AMO.ProactiveCaching();
    TimeSpan defaultProactiveChachingTimeSpan = new TimeSpan(0, 0, -1);
    currentDimension.ProactiveCaching.SilenceInterval = defaultProactiveChachingTimeSpan;
    currentDimension.ProactiveCaching.Latency = defaultProactiveChachingTimeSpan;
    currentDimension.ProactiveCaching.SilenceOverrideInterval = defaultProactiveChachingTimeSpan;
    currentDimension.ProactiveCaching.ForceRebuildInterval = defaultProactiveChachingTimeSpan;
    currentDimension.ProactiveCaching.Source = new AMO.ProactiveCachingInheritedBinding();

    //Manualy add a "RowNumber" attribute as the key attribute of the dimension, until a primary key is defined
    //"RowNumber" a required column for a tabular model and has to be of type AMO.AttributeType.RowNumber and
binding AMO.RowNumberBinding.
    //The name of the "RowNumber" attribute can be any name, as long as type and binding are correctly set
    //By default, the MS client tools set the column name and column ID of the RowNumber attribute to
"RowNumber"
    //In this sample, to avoid problems, on any customer table that contains a column named 'RowNumber'
    //the Id value of the column (in the dimension object) will be renamed to 'RowNumber_in_<TableName>' and
the Name of the column will remain "RowNumber"

    AMO.DimensionAttribute currentAttribute = currentDimension.Attributes.Add("RowNumber", "RowNumber");
    currentAttribute.Type = AMO.AttributeType.RowNumber;
    currentAttribute.KeyUniquenessGuarantee = true;
    currentAttribute.Usage = AMO.AttributeUsage.Key;
    currentAttribute.KeyColumns.Add(new AMO.DataItem());
    currentAttribute.KeyColumns[0].DataType = System.Data.OleDb.OleDbType.Integer;
    currentAttribute.KeyColumns[0].DataSize = 4;
    currentAttribute.KeyColumns[0].NullProcessing = AMO.NullProcessing.Error;
    currentAttribute.KeyColumns[0].Source = new AMO.RowNumberBinding();
    currentAttribute.NameColumn = new AMO.DataItem();
    currentAttribute.NameColumn.DataType = System.Data.OleDb.OleDbType.WChar;
    currentAttribute.NameColumn.DataSize = 4;
    currentAttribute.NameColumn.NullProcessing = AMO.NullProcessing.ZeroOrBlank;
    currentAttribute.NameColumn.Source = new AMO.RowNumberBinding();
    currentAttribute.OrderBy = AMO.OrderBy.Key;
    currentAttribute.AttributeHierarchyVisible = false;
    //Deferring AttributeRelationships until after adding each other attribute
    //Add each column in the table as an attribute in the dimension
    foreach (DataColumn dataColumn in dataTable.Columns)
    {
        string attributeID, attributeName;
        if (dataColumn.ColumnName != "RowNumber")
        {
            attributeID = dataColumn.ColumnName;
        }
        else
        {
            attributeID = string.Format("RowNumber_in_{0}", dataTable.TableName);
        }
        attributeName = dataColumn.ColumnName;
        currentAttribute = currentDimension.Attributes.Add(attributeName, attributeID);
        currentAttribute.Usage = AMO.AttributeUsage.Regular;
        currentAttribute.KeyUniquenessGuarantee = false;
        currentAttribute.KeyColumns.Add(new AMO.DataItem(dataTable.TableName, dataColumn.ColumnName,
AMO.OleDbTypeConverter.GetRestrictedOleDbType(dataColumn.DataType)));
        currentAttribute.KeyColumns[0].Source = new AMO.ColumnBinding(dataTable.TableName,
dataColumn.ColumnName);
```

```
        currentAttribute.KeyColumns[0].NullProcessing = AMO.NullProcessing.Preserve;
        currentAttribute.NameColumn = new AMO.DataItem(dataTable.TableName, dataColumn.ColumnName,
System.Data.OleDb.OleDbType.WChar);
        currentAttribute.NameColumn.Source = new AMO.ColumnBinding(dataTable.TableName, dataColumn.ColumnName);
        currentAttribute.NameColumn.NullProcessing = AMO.NullProcessing.ZeroOrBlank;
        currentAttribute.OrderBy = AMO.OrderBy.Key;
        AMO.AttributeRelationship currentAttributeRelationship =
currentDimension.Attributes["RowNumber"].AttributeRelationships.Add(currentAttribute.ID);
        currentAttributeRelationship.Cardinality = AMO.Cardinality.Many;
        currentAttributeRelationship.OverrideBehavior = AMO.OverrideBehavior.None;
    }
    #endregion

    #region Add Dimension to Model cube
    cb.Dimensions.Add(tableID, tableID, tableID);
    #endregion

    #region Add MeasureGroup to Model cube
    AMO.MeasureGroup currentMeasureGroup = cb.MeasureGroups.Add(tableID, tableID);
    currentMeasureGroup.StorageMode = AMO.StorageMode.InMemory;
    currentMeasureGroup.ProcessingMode = AMO.ProcessingMode.Regular;

    //Adding Dimension
    AMO.DegenerateMeasureGroupDimension currentMGDim = new AMO.DegenerateMeasureGroupDimension(tableID);
    currentMeasureGroup.Dimensions.Add(currentMGDim);
    currentMGDim.ShareDimensionStorage = AMO.StorageSharingMode.Shared;
    currentMGDim.CubeDimensionID = tableID;
    foreach (AMO.CubeAttribute ca in cb.Dimensions[tableID].Attributes)
    {
        AMO.MeasureGroupAttribute mga = new AMO.MeasureGroupAttribute(ca.AttributeID);
        if (mga.AttributeID == "RowNumber")
        {
            mga.Type = AMO.MeasureGroupAttributeType.Granularity;
            AMO.DataItem rowNumberKeyColumn = new AMO.DataItem(new AMO.ColumnBinding(tableID, "RowNumber"));
            rowNumberKeyColumn.DataType = System.Data.OleDb.OleDbType.Integer;
            mga.KeyColumns.Add(rowNumberKeyColumn);
        }
        else
        {
            foreach (AMO.DataItem di in ca.Attribute.KeyColumns)
            {
                AMO.DataItem keyColumn = new AMO.DataItem(new AMO.ColumnBinding(tableID,
((AMO.ColumnBinding)di.Source).ColumnID));
                keyColumn.DataType = di.DataType;
                keyColumn.NullProcessing = AMO.NullProcessing.Preserve;
                keyColumn.InvalidXmlCharacters = AMO.InvalidXmlCharacters.Remove;
                mga.KeyColumns.Add(keyColumn);
            }
        }
        currentMGDim.Attributes.Add(mga);
    }

    //Adding default Measure
    String defaultMeasureID = string.Concat("_Count ", tableID);
    AMO.Measure currentMeasure = currentMeasureGroup.Measures.Add(defaultMeasureID, defaultMeasureID);
    currentMeasure.AggregateFunction = AMO.AggregationFunction.Count;
    currentMeasure.DataType = AMO.MeasureDataType.BigInt;
    AMO.DataItem currentMeasureSource = new AMO.DataItem(new AMO.RowBinding(tableID));
    currentMeasureSource.DataType = System.Data.OleDb.OleDbType.BigInt;
    currentMeasure.Source = currentMeasureSource;

    //Partitions
    AMO.Partition currentPartition = new AMO.Partition(tableID, tableID);
    currentPartition.StorageMode = AMO.StorageMode.InMemory;
    currentPartition.ProcessingMode = AMO.ProcessingMode.Regular;
    currentPartition.Source = new AMO.QueryBinding(newDatasource.ID,
(String)dataTable.ExtendedProperties["sqlStmt"]);
    currentMeasureGroup.Partitions.Add(currentPartition);
```

```
    #endregion

    #region Update new objects in database
    db.Update(AMO.UpdateOptions.ExpandFull, AMO.UpdateMode.UpdateOrCreate);
    #endregion

    return true;
}
```

Caution

The above code snippet has no error checking or clean up procedures in case of failure.

Understanding Tabular Object Model at Levels 1050 through 1103

11/28/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A tabular model is a logical representation of tables, relationships, hierarchies, perspectives, measures, and Key Performance. This section introduces the internal implementation using AMO. See Developing with Analysis Management Objects (AMO) if you haven't used AMO before.

The approach here is top-down, all relevant objects in the tabular model are logically mapped to AMO objects, and the required interaction or workflow explained. A source code sample to create a tabular model using AMO, AMO to Tabular sample is available from Codeplex. An important note about the code in the sample: it is provided only to support to the logical concepts explained here and should not be used in a production environment. The sample is provided without support or warranty.

Database Representation

A database provides the container object for the tabular model. All objects in a tabular model are contained in the database. In terms of AMO objects, a database representation has a one-to-one mapping relationship with `xref:Microsoft.AnalysisServices.Database`, and no other main AMO objects are required. It is important to note that this doesn't mean that all contained objects in the AMO database object can be used when modeling.

See Database Representation(Tabular) for a detailed explanation on how to create and manipulate the database representation.

Connection Representation

A connection establishes the relationship between the data to be included in a tabular model solution and the model itself. In terms of AMO objects, a connection has a one-to-one mapping relationship with `xref:Microsoft.AnalysisServices.DataSource`, and no other main AMO objects are required. It is important to note that this doesn't mean that all contained objects in the AMO datasource object can be used when modeling.

See Connection Representation (Tabular) for a detailed explanation on how to create and manipulate the data source representation.

Table Representation

Tables are database objects that contain the data in the database. In terms of AMO objects, a table has a one-to-many mapping relationship. A table is represented by the usage of the following AMO objects: `xref:Microsoft.AnalysisServices.DataSourceView`, `xref:Microsoft.AnalysisServices.Dimension`, `xref:Microsoft.AnalysisServices.Cube`, `xref:Microsoft.AnalysisServices.CubeDimension`, `xref:Microsoft.AnalysisServices.MeasureGroup` and `xref:Microsoft.AnalysisServices.Partition` are the main required objects. However, it is important to note that this doesn't mean that all contained objects in the previously mentioned AMO objects can be used when modeling.

See Tables Representation (Tabular) for a detailed explanation on how to create and manipulate the table representation.

Calculated Column Representation

Calculated columns are evaluated expressions that generate a column in a table, where a new value is calculated and stored for each row in the table. In terms of AMO objects, a calculated column has a one-to-many mapping relationship. A calculated column is represented by the usage of the following AMO objects: `xref:Microsoft.AnalysisServices.Dimension` and `xref:Microsoft.AnalysisServices.MeasureGroup` are the main required objects. It is important to note that this doesn't mean that all contained objects in the previously mentioned AMO objects can be used when modeling.

See Calculated Column Representation (Tabular) for a detailed explanation on how to create and manipulate the calculated column representation.

Calculated Measure Representation

Calculated Measures are stored expressions that are evaluated upon request once the model is deployed. In terms of AMO objects, a calculated measure has a one-to-many mapping relationship. A calculated column is represented by the usage of the following AMO objects: `xref:Microsoft.AnalysisServices.MdxScript.Commands%2A` and `xref:Microsoft.AnalysisServices.MdxScript.CalculationProperties%2A` are the main required objects. It is important to note that this doesn't mean that all contained objects in the previously mentioned AMO objects can be used when modeling.

> **NOTE**
> The `xref:Microsoft.AnalysisServices.Measure` objects have no relationship with the calculated measures in tabular models, and are not supported in tabular models.

See Calculated Measure Representation (Tabular) for a detailed explanation on how to create and manipulate the calculated measure representation.

Hierarchy Representation

Hierarchies are a mechanism to provide a richer drill-up and drill-down experience to the end user. In terms of AMO objects, a hierarchy representation has a one-to-one mapping relationship with `xref:Microsoft.AnalysisServices.Hierarchy`, and no other main AMO objects are required. It is important to note that this doesn't mean that all contained objects in the AMO database object can be used when doing tabular modeling.

See Hierarchy Representation (Tabular) for a detailed explanation on how to create and manipulate the hierarchy representation.

Key Performance Indicator -KPI- Representation

A KPI is used to gauge performance of a value, defined by a Base measure, against a Target value. In terms of AMO objects, a KPI representation has a one-to-many mapping relationship. A KPI is represented by the usage of the following AMO objects: `xref:Microsoft.AnalysisServices.MdxScript.Commands%2A` and `xref:Microsoft.AnalysisServices.MdxScript.CalculationProperties%2A` are the main required objects. It is important to note that this doesn't mean that all contained objects in the previously mentioned AMO objects can be used when modeling.

> **NOTE**
> Also, important distinction, the `xref:Microsoft.AnalysisServices.Kpi` objects have no relationship with the KPIs in tabular models. And, they are not supported in tabular models.

See Key Performance Indicator Representation (Tabular) for a detailed explanation on how to create and manipulate the KPI representation.

Partition Representation

For operational purposes, a table can be divided in different subsets of rows that, when combined together, form the table. Each of those subsets is a partition of the table. In terms of AMO objects, a partition representation has a one-to-one mapping relationship with `xref:Microsoft.AnalysisServices.Partition` and no other main AMO objects are required. It is important to note that this doesn't mean that all contained objects in the AMO database object can be used when modeling.

See Partition Representation (Tabular) for a detailed explanation on how to create and manipulate the partition representation.

Relationship Representation

A relationship is a connection between two tables of data. The relationship establishes how the data in the two tables should be correlated.

In tabular models, multiple relationships can be defined between two tables. When multiple relationships between two tables are defined, only one can be defined as the default Active relationship. All other relationships are Inactive.

In terms of AMO objects, all inactive relationships have a representation of a one-to-one mapping relationship with `xref:Microsoft.AnalysisServices.Relationship`, and no other main AMO objects are required. For the active relationship, other requirements exist and a mapping to the `xref:Microsoft.AnalysisServices.ReferenceMeasureGroupDimension` is also required. It is important to note that this doesn't mean that all contained objects in the AMO relationship or referenceMeasureGroupDimension object can be used when modeling.

See Relationship Representation (Tabular) for a detailed explanation on how to create and manipulate the relationship representation.

Perspective Representation

A perspective is a mechanism to simplify or focus the mode. In terms of AMO objects, a relationship representation has a one-to-one mapping relationship with `xref:Microsoft.AnalysisServices.Perspective` and no other main AMO objects are required. It is important to note that this doesn't mean that all contained objects in the AMO perspective object can be used when doing tabular modeling.

See Perspective Representation (Tabular) for a detailed explanation on how to create and manipulate the perspective representation.

> **WARNING**
>
> Perspectives are not a security mechanism; objects outside the perspective can still be accessed by the user through other interfaces.

Multidimensional Model Programming

10/26/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services provides several APIs that you can use to program against an Analysis Services instance and the multidimensional databases that it makes available. This section describes the approaches available to developers who want to create custom applications using Analysis Services multidimensional solutions. You can use this information to choose the programming interface that best meets the requirements of a particular project. Analysis Services development projects can be based on managed or non-managed code that runs on a Windows platform, or other platforms that support HTTP access.

In This Section

Understanding Microsoft OLAP Architecture

Developing with ADOMD.NET

Developing with Analysis Management Objects (AMO)

Developing with XMLA in Analysis Services

Developing with Analysis Services Scripting Language (ASSL)

Extending OLAP functionality

Analysis Services OLE DB Provider (Analysis Services - Multidimensional Data)

See Also

Tabular Model Programming for Compatibility Levels 1050 through 1103
Data Mining Programming

ASSL Objects and Object Characteristics

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Objects in Analysis Services Scripting Language (ASSL) follow specific guidelines in regards to object groups, inheritance, naming, expansion, and processing.

Object Groups

All Microsoft SQL Server Analysis Services objects have an XML representation. The objects are divided into two groups:

Major objects
Major objects can be independently created, altered, and deleted. Major objects include:

- Servers
- Databases
- Dimensions
- Cubes
- Measure groups
- Partitions
- Perspectives
- Mining models
- Roles
- Commands associated with a server or database
- Data sources

Major objects have the following properties to track their history and status.

- **CreatedTimestamp**
- **LastSchemaUpdate**
- **LastProcessed** (where appropriate)

> **NOTE**
> The classification of an object as a major object affects how an instance of Analysis Services treats that object and how that object is handled in the object definition language. However, this classification does not guarantee that Analysis Services management and development tools will allow the independent creation, modification, or deletion of these objects.

Minor objects
Minor objects can only be created, altered, or deleted as part of creating, altering, or deleting the parent major object. Minor objects include:

- Hierarchies and levels
- Attributes
- Measures
- Mining model columns
- Commands associated with a cube
- Aggregations

Object Expansion

The **ObjectExpansion** restriction can be used to control the degree of expansion of ASSL XML returned by the server. This restriction has the options listed in the following table.

ENUMERATION VALUE	ALLOWED FOR <ALTER>	DESCRIPTION
ReferenceOnly	no	Returns only the name, ID, and timestamp for the requested object and for all contained major objects recursively.
ObjectProperties	yes	Expands the requested object and minor contained objects, but does not return major contained objects.
ExpandObject	no	Same as *ObjectProperties*, but also returns the name, ID, and timestamp for contained major objects.
ExpandFull	yes	Fully expands the requested object and all contained objects recursively.

This ASSL reference section describes the *ExpandFull* representation. All other **ObjectExpansion** levels are derived from this level.

Object Processing

ASSL includes read-only elements or properties (for example, **LastProcessed**) that can be read from the Analysis Services instance, but which are omitted when command scripts are submitted to the instance. Analysis Services ignores modified values for read-only elements without warning or error.

Analysis Services also ignores inappropriate or irrelevant properties without raising validation errors. For example, the X element should only be present when the Y element has a particular value. The Analysis Services instance ignores the X element instead of validating that element against the value of the Y element.

Database Objects (Analysis Services - Multidimensional Data)

5/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

A Microsoft SQL Server Analysis Services instance contains database objects and assemblies for use with online analytical processing (OLAP) and data mining.

- Databases contain OLAP and data mining objects, such as data sources, data source views, cubes, measures, measure groups, dimensions, attributes, hierarchies, mining structures, mining models and roles.
- Assemblies contain user-defined functions that extend the functionality of the intrinsic functions provided with the Multidimensional Expressions (MDX) and Data Mining Extensions (DMX) languages.

The Database object is the container for all data objects that are needed for a business intelligence project (such as OLAP cubes, dimensions, and data mining structures), and their supporting objects (such as DataSource, Account, and Role).

A Database object provides access to objects and attributes that include the following:

- All cubes that you can access, as a collection.
- All dimensions that you can access, as a collection.
- All mining structures that you can access, as a collection.
- All data sources and data source views, as two collections.
- All roles and database permissions, as two collections.
- The collation value for the database.
- The estimated size of the database.
- The language value of the database.
- The visible setting for the database.

In This Section

The following topics describe objects shared by both OLAP and data mining features in Analysis Services.

TOPIC	DESCRIPTION
Data Sources in Multidimensional Models	Describes a data source in Analysis Services.
Data Source Views in Multidimensional Models	Describes a logical data model based on one or more data sources, in Analysis Services.
Cubes in Multidimensional Models	Describes cubes and cube objects, including measures, measure groups, dimension usage relationships, calculations, key performance indicators, actions, translations, partitions, and perspectives.

TOPIC	DESCRIPTION
Dimensions (Analysis Services - Multidimensional Data)	Describes dimensions and dimension objects, including attributes, attribute relationships, hierarchies, levels, and members.
Mining Structures (Analysis Services - Data Mining)	Describes mining structures and mining objects, including mining models.
Security Roles (Analysis Services - Multidimensional Data)	Describes a role, the security mechanism used to control access to objects in Analysis Services.
Multidimensional Model Assemblies Management	Describes an assembly, a collection of user-defined functions used to extend the MDX and DMX languages, in Analysis Services.

See Also

Supported Data Sources (SSAS - Multidimensional)
Multidimensional Model Solutions
Data Mining Solutions

Client Architecture Requirements for Analysis Services Development

5/16/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Microsoft SQL Server Analysis Services supports a thin-client architecture. The Analysis Services calculation engine is entirely server-based, so all queries are resolved on the server. As a result, only a single round trip between the client and the server is required for each query, resulting in scalable performance as queries increase in complexity.

The native protocol for Analysis Services is XML for Analysis (XML/A). Analysis Services provides several data access interfaces for client applications, but all of these components communicate with an instance of Analysis Services using XML for Analysis.

Several different providers are provided with Analysis Services to support different programming languages. A provider communicates with an Analysis Services server by sending and receiving XML for Analysis in SOAP packets over TCP/IP or over HTTP through Internet Information Services (IIS). An HTTP connection uses a COM object instantiated by IIS, called a data pump, which acts as a conduit for Analysis Services data. The data pump does not examine the underlying data contained in the HTTP stream in any way, nor are any of the underlying data structures available to any of the code in the data library itself.

Win32 client applications can connect to an Analysis Services server using OLE DB for OLAP interfaces or the Microsoft® ActiveX® Data Objects (ADO) object model for Component Object Model (COM) automation languages, such as Microsoft Visual Basic®. Applications coded with .NET languages can connect to an Analysis Services server using ADOMD.NET.

Existing applications can communicate with Analysis Services without modification simply by using one of the Analysis Services providers.

PROGRAMMING LANGUAGE	DATA ACCESS INTERFACE
C++	OLE DB for OLAP
Visual Basic 6	ADO MD
.NET languages	ADO MD.NET
Any language that supports SOAP	XML for Analysis

Analysis Services has a Web architecture with a fully scalable middle tier for deployment by both small and large organizations. Analysis Services provides broad middle tier support for Web services. ASP applications are supported by OLE DB for OLAP and ADO MD, ASP.NET applications are supported by ADOMD.NET. The middle tier, illustrated in the following figure, is scalable to many concurrent users.

Both client and middle tier applications can communicate directly with Analysis Services without using a provider. Client and middle tier applications may send XML for Analysis in SOAP packets over TCP/IP, HTTP, or HTTPS. The client may be coded using any language that supports SOAP. Communication in this case is most easily managed by Internet Information Services (IIS) using HTTP, although a direct connection to the server using TCP/IP may also be coded. This is the thinnest possible client solution for Analysis Services.

Analysis Services in Tabular or SharePoint Mode

In SQL Server 2017, the server can be started in xVelocity in-memory analytics engine (VertiPaq) mode for tabular

databases and for Power Pivot workbooks that have been published to a SharePoint site.

Power Pivot for Excel and SQL Server Data Tools (SSDT) are the only client environments that are supported for creating and querying in-memory databases that use SharePoint or Tabular mode, respectively. The embedded Power Pivot database that you create by using the Excel and Power Pivot tools is contained within the Excel workbook, and is saved as part of the Excel .xlsx file.

However, a Power Pivot workbook can use data that is stored in a traditional cube if you import the cube data into the workbook. You can also import data from another Power Pivot workbook if it has been published to a SharePoint site.

> **NOTE**
>
> When you use a cube as a data source for a Power Pivot workbook, the data that you get from the cube is defined as an MDX query; however, the data is imported as a flattened snapshot. You cannot interactively work with the data or refresh the data from the cube.

For more information about using an SSAS cube as a data source, see the Power Pivot for Excel.

Interfaces for Power Pivot Client

Power Pivot interacts with the xVelocity in-memory analytics engine (VertiPaq) storage engine within the workbook by using the established interfaces and languages for Analysis Services: AMO and ADOMD.NET, and MDX and XMLA. Within the add-in, measures are defined by using a formula language similar to Excel, Data Analysis Expressions (DAX). DAX expressions are embedded within the XMLA messages that are sent to the in-process server.

Providers

Communications between Power Pivot and Excel use the MSOLAP OLEDB provider (version 11.0). Within the MSOLAP provider, there are four different modules, or transports, that can be used for sending messages between the client and server.

TCP/IP Used for normal client-server connections.

HTTP Used for HTTP connections via the SSAS data pump service, or by a call to the SharePoint Power Pivot Web Service (WS) component.

INPROC Used for connections to the in-process engine.

CHANNEL Reserved for communications with the Power Pivot System Service in the SharePoint farm.

See Also

OLAP Engine Server Components

Analysis Services Personalization Extensions

5/16/2018 • 5 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

SQL Server Analysis Services personalization extensions are the foundation of the idea of implementing a plug-in architecture. In a plug-in architecture, you can develop new cube objects and functionality dynamically and share them easily with other developers. As such, Analysis Services personalization extensions provide the functionality that makes it possible to achieve the following:

- **Dynamic design and deployment** Immediately after you design and deploy Analysis Services personalization extensions, users have access to the objects and functionality at the start of the next user session.
- **Interface independence** Regardless of the interface that you use to create the Analysis Services personalization extensions, users can use any interface to access the objects and functionality.
- **Session context** Analysis Services personalization extensions are not permanent objects in the existing infrastructure and do not require the cube to be reprocessed. They become exposed and created for the user at the time that the user connects to the database, and remain available for the length of that user session.
- **Rapid distribution** Share Analysis Services personalization extensions with other software developers without having to go into detailed specifications about where or how to find this extended functionality.

Analysis Services personalization extensions have many uses. For example, your company has sales that involve different currencies. You create a calculated member that returns the consolidated sales in the local currency of the person who is accessing the cube. You create this member as a personalization extension. You then share this calculated member to a group of users. Once shared, those users have immediate access to the calculated member as soon as they connect to the server. They have access even if they are not using the same interface as the one that was used to create the calculated member.

Analysis Services personalization extensions are a simple and elegant modification to the existing managed assembly architecture and are exposed throughout the Analysis Services Microsoft.AnalysisServices.AdomdServer object model, Multidimensional Expressions (MDX) syntax, and schema rowsets.

Logical Architecture

The architecture for Analysis Services personalization extensions is based on the managed assembly architecture and the following four basic elements:

The [PlugInAttribute] custom attribute
When starting the service, Analysis Services loads the required assemblies and determines which classes have the Microsoft.AnalysisServices.AdomdServer.PlugInAttribute custom attribute.

> **NOTE**
> The .NET Framework defines custom attributes as a way to describe your code and affect run-time behavior. For more information, see the topic, "Attributes Overview," in the .NET Framework Developer's Guide on MSDN.

For all classes with the Microsoft.AnalysisServices.AdomdServer.PlugInAttribute custom attribute, Analysis Services invokes their default constructors. Invoking all the constructors at startup provides a common location from which to build new objects and that is independent of any user activity.

In addition to building a small cache of information about authoring and managing personalization extensions, the class constructor typically subscribes to the Microsoft.AnalysisServices.AdomdServer.Server.SessionOpened and Microsoft.AnalysisServices.AdomdServer.Server.SessionClosing events. Failing to subscribe to these events may cause the class to be inappropriately marked for cleanup by the common language runtime (CLR) garbage collector.

Session context
For those objects that are based on personalization extensions, Analysis Services creates an execution environment during the client session and dynamically builds most of those objects in this environment. Like any other CLR assembly, this execution environment also has access to other functions and stored procedures. When the user session ends, Analysis Services removes the dynamically created objects and closes the execution environment.

Events
Object creation is triggered by the session events **On-Cube-OpenedCubeOpened** and **On-Cube-ClosingCubeClosing**.

Communication between the client and the server occurs through specific events. These events make the client aware of the situations that lead to the client's objects being built. The client's environment is dynamically created using two sets of events: session events and cube events.

Session events are associated with the server object. When a client logs on to a server, Analysis Services creates a session and triggers the Microsoft.AnalysisServices.AdomdServer.Server.SessionOpened event. When a client ends the session on the server, Analysis Services triggers the Microsoft.AnalysisServices.AdomdServer.Server.SessionClosing event.

Cube events are associated with the connection object. Connecting to a cube triggers the Microsoft.AnalysisServices.AdomdServer.AdomdConnection.CubeOpened event. Closing the connection to a cube, by either closing the cube or by changing to a different cube, triggers a Microsoft.AnalysisServices.AdomdServer.AdomdConnection.CubeClosing event.

Traceability and error handling
All activity is traceable by using SQL Server Profiler. Unhandled errors are reported to the Windows event log.

All object authoring and management is independent of this architecture and is the sole responsibility of the developers of the objects.

Infrastructure Foundations

Analysis Services personalization extensions are based on existing components. The following is a summary of enhancements and improvements that provide the personalization extensions functionality.

Assemblies

The custom attribute, Microsoft.AnalysisServices.AdomdServer.PlugInAttribute, can be added to your custom assemblies to identify Analysis Services personalization extensions classes.

Changes to the AdomdServer Object Model

The following objects in the Microsoft.AnalysisServices.AdomdServer object model have been enhanced or added to the model.

New AdomdConnection Class

The Microsoft.AnalysisServices.AdomdServer.AdomdConnection class is new and exposes several personalization extensions through both properties and events.

Properties

- Microsoft.AnalysisServices.AdomdServer.AdomdConnection.SessionID*, a read-only string value representing the session Id of the current connection.

- Microsoft.AnalysisServices.AdomdServer.AdomdConnection.ClientCulture*, a read-only reference to the client culture associated with current session.
- Microsoft.AnalysisServices.AdomdServer.AdomdConnection.User*, a read-only reference to the identity interface representing the current user.

Events

- Microsoft.AnalysisServices.AdomdServer.AdomdConnection.CubeOpened
- Microsoft.AnalysisServices.AdomdServer.AdomdConnection.CubeClosing

New Properties in the Context class

The Microsoft.AnalysisServices.AdomdServer.Context class has two new properties:

- Microsoft.AnalysisServices.AdomdServer.Context.Server*, a read-only reference to the new server object.
- Microsoft.AnalysisServices.AdomdServer.Context.CurrentConnection*, a read-only reference to the new Microsoft.AnalysisServices.AdomdServer.AdomdConnection object.

New Server class

The Microsoft.AnalysisServices.AdomdServer.Server class is new and exposes several personalization extensions through both class properties and events.

Properties

- Microsoft.AnalysisServices.AdomdServer.Server.Name*, a read-only string value representing the server name.
- Microsoft.AnalysisServices.AdomdServer.Server.Culture*, A read-only reference to the global culture associated with the server.

Events

- Microsoft.AnalysisServices.AdomdServer.Server.SessionOpened
- Microsoft.AnalysisServices.AdomdServer.Server.SessionClosing

AdomdCommand class

The Microsoft.AnalysisServices.AdomdServer.AdomdCommand class now supports of the following MDX commands:

- CREATE MEMBER Statement (MDX)
- UPDATE MEMBER Statement (MDX)
- DROP MEMBER Statement (MDX)
- CREATE SET Statement (MDX)
- DROP SET Statement (MDX)
- CREATE KPI Statement (MDX)
- DROP KPI Statement (MDX)

MDX extensions and enhancements

The CREATE MEMBER command is enhanced with the **caption** property, the **display_folder** property, and the **associated_measure_group** property.

The UPDATE MEMBER command is added to avoid member re-creation when an update is needed with the consequent loss of precedence in solving calculations. Updates cannot change the scope of the calculated member,

move the calculated member to a different parent, or define a different **solveorder**.

The CREATE SET command is enhanced with the **caption** property, the **display_folder** property, and the new **STATIC | DYNAMIC** keyword. *Static* means that set is evaluated only at creation time. *Dynamic* means that the set is evaluated every time that the set is used in a query. The default value is **STATIC** if a keyword is omitted.

CREATE KPI and DROP KPI commands are added to the MDX syntax. KPIs can be created dynamically from any MDX script.

Schema Rowsets extensions

On MDSCHEMA_MEMBERS *scope* column is added. Scope values are as follows: MDMEMBER_SCOPE_GLOBAL=1, MDMEMBER_SCOPE_SESSION=2.

On MDSCHEMA_SETS *set_evaluation_context* column is added. Set evaluation context values are as follows: MDSET_RESOLUTION_STATIC = 1, MDSET_RESOLUTION_DYNAMIC = 2.

On MDSCHEMA_KPIS scope column is added. Scope values are as follows: MDKPI_SCOPE_GLOBAL=1, MDKPI_SCOPE_SESSION=2.

Data Mining Programming

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

If you find that the built-in tools and viewers in Analysis Services do not meet your requirements, you can extend the power of Analysis Services by coding your own extensions. In this approach, you have two options:

- **XMLA**

  Analysis Services supports XML for Analysis (XMLA) as a protocol for communication with client applications. Additional commands are supported by Analysis Services that extend the XML for Analysis specification.

  Because Analysis Services uses XMLA for data definition, data manipulation, and data control support, you can create mining structures and mining models by using the visual tools provided by SQL Server Data Tools (SSDT), and then extend the data mining objects that you have created by using Data Mining Extensions (DMX) and Analysis Services Scripting Language (ASSL) scripts.

  You can create and modify data mining objects entirely in XMLA scripts, and run prediction queries against the models programmatically from your own applications.

- **Analysis Management Objects (AMO)**

  Analysis Services also provides a complete framework that enables third-party data mining providers to integrate the data mining objects into Analysis Services.

  You can create mining structures and mining models by using AMO. See the following samples in CodePlex:

  - AMO Browser

    Connects to the SSAS instance you specify and lists all server objects and their properties, including mining structure and mining models.

  - AMO Simple Sample

    The AS Simple Sample covers programmatic access to most major objects, and demonstrates metadata browsing, and access to the values in objects.

    The sample also demonstrates how to create and process a data mining structure and model, as well as browse an existing data mining model.

- **DMX**

  You can use DMX to encapsulate command statements, prediction queries, and metadata queries and return results in a tabular format, assuming you have created a connection to an Analysis Services server.

In This Section

OLE DB for Data Mining

Describes additions to the specification to support data mining and multidimensional data: new schema rowsets and columns, Data Mining Extensions (DMX) language for creating and managing mining structures.

Related Reference

Developing with ADOMD.NET
Introduces ADOMD.NET client and server programming objects.

Developing with Analysis Management Objects (AMO)
Introduces the AMO programming library.

Developing with Analysis Services Scripting Language (ASSL)
Introduces XML for Analysis (XMLA) and its extensions.

See Also

Analysis Services Developer Documentation
Data Mining Extensions (DMX) Reference

Data Mining Extensions (DMX) Reference

6/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data Mining Extensions (DMX) is a language that you can use to create and work with data mining models in Microsoft SQL Server Analysis Services. You can use DMX to create the structure of new data mining models, to train these models, and to browse, manage, and predict against them. DMX is composed of data definition language (DDL) statements, data manipulation language (DML) statements, and functions and operators.

Microsoft OLE DB for Data Mining Specification

The data mining features in Analysis Services are built to comply with the Microsoft OLE DB for Data Mining specification.

The Microsoft OLE DB for Data Mining specification defines the following:

- A structure to hold the information that defines a data mining model.
- A language for creating and working with data mining models.

The specification defines the basis of data mining as the data mining model virtual object. The data mining model object encapsulates all that is known about a particular mining model. The data mining model object is structured like an SQL table, with columns, data types, and meta information that describe the model. This structure lets you use the DMX language, which is an extension of SQL, to create and work with models.

For More Information: Mining Structures (Analysis Services - Data Mining)

DMX Statements

You can use DMX statements to create, process, delete, copy, browse, and predict against data mining models. There are two types of statements in DMX: data definition statements and data manipulation statements. You can use each type of statement to perform different kinds of tasks.

The following sections provide more information about working with DMX statements:

- Data Definition Statements
- Data Manipulation Statements
- Query Fundamentals

Data Definition Statements

Use data definition statements in DMX to create and define new mining structures and models, to import and export mining models and mining structures, and to drop existing models from a database. Data definition statements in DMX are part of the data definition language (DDL).

You can perform the following tasks with the data definition statements in DMX:

- Create a mining structure by using the CREATE MINING STRUCTURE statement, and add a mining model to the mining structure by using the ALTER MINING STRUCTURE statement.
- Create a mining model and associated mining structure simultaneously by using the CREATE MINING MODEL statement to build an empty data mining model object.

- Export a mining model and associated mining structure to a file by using the EXPORT statement. Import a mining model and associated mining structure from a file that is created by the EXPORT statement by using the IMPORT statement.
- Copy the structure of an existing mining model into a new model, and train it with the same data, by using the SELECT INTO statement.
- Completely remove a mining model from a database by using the DROP MINING MODEL statement. Completely remove a mining structure and all its associated mining models from the database by using the DROP MINING STRUCTURE statement.

To learn more about the data mining tasks that you can perform by using DMX statements, see Data Mining Extensions (DMX) Statement Reference.

Back to DMX Statements

Data Manipulation Statements

Use data manipulation statements in DMX to work with existing mining models, to browse the models and to create predictions against them. Data manipulation statements in DMX are part of the data manipulation language (DML).

You can perform the following tasks with the data manipulation statements in DMX:

- Train a mining model by using the INSERT INTO statement. This does not insert the actual source data into a data mining model object, but instead creates an abstraction that describes the mining model that the algorithm creates. The source query for an INSERT INTO statement is described in <source data query>.
- Extend the SELECT statement to browse the information that is calculated during model training and stored in the data mining model, such as statistics of the source data. Following are the clauses that you can include to extend the power of the SELECT statement:
  - SELECT DISTINCT FROM <model > (DMX)
  - SELECT FROM <model>.CONTENT (DMX)
  - SELECT FROM <model>.CASES (DMX)
  - SELECT FROM <model>.SAMPLE_CASES (DMX)
  - SELECT FROM <model>.DIMENSION_CONTENT (DMX)
- Create predictions that are based on an existing mining model by using the PREDICTION JOIN clause of the SELECT statement. The source query for a PREDICTION JOIN statement is described in <source data query>.
- Remove all the trained data from a model or a structure by using the DELETE (DMX) statement.

To learn more about the data mining tasks that you can perform by using DMX statements, see Data Mining Extensions (DMX) Statement Reference.

Back to DMX Statements

DMX Query Fundamentals

The SELECT statement is the basis for most DMX queries. Depending on the clauses that you use with such statements, you can browse, copy, or predict against mining models. The prediction query uses a form of SELECT to create predictions based on existing mining models. Functions extend your ability to browse and query the mining models beyond the intrinsic capabilities of the data mining model.

You can use DMX functions to obtain information that is discovered during the training of your models, and to calculate new information. You can use these functions for many purposes, including to return statistics that

describe the underlying data or the accuracy of a prediction, or to return an expanded explanation of a prediction.

For More Information: Understanding the DMX Select Statement, General Prediction Functions (DMX), Structure and Usage of DMX Prediction Queries, Data Mining Extensions (DMX) Function Reference

Back to DMX Statements

See Also

Data Mining Extensions (DMX) Function Reference
Data Mining Extensions (DMX) Operator Reference
Data Mining Extensions (DMX) Statement Reference
Data Mining Extensions (DMX) Syntax Conventions
Data Mining Extensions (DMX) Syntax Elements
General Prediction Functions (DMX)
Structure and Usage of DMX Prediction Queries
Understanding the DMX Select Statement

Data Mining Programming - OLE DB

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

The data mining features in Microsoft SQL Server Analysis Services comply with the Microsoft OLE DB for Data Mining 1.0 specification released in June 2000.

Analysis Services has extended the specification by adding new schema rowsets, adding columns to existing schema rowsets, and adding syntax to the Data Mining Extensions (DMX) language for creating and managing mining structures.

For More Information: Data Mining Schema Rowsets, CREATE MINING STRUCTURE (DMX), ALTER MINING STRUCTURE (DMX), DROP MINING STRUCTURE (DMX)

See Also

Analysis Services Schema Rowsets
Data Mining Extensions (DMX) Reference

Analysis Services samples

8/16/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Use the information here to find sample databases and projects to help you learn about and test your Analysis Services solutions.

Samples on GitHub

Git repo for Analysis Services includes code samples and community projects.

Sample solutions and databases

While no longer officially supported or updated, Adventure Works remains one of the most inclusive and robust sample datasets for learning about and testing Analysis Services. Analysis Services sample projects and databases, as well as examples in documenation, blog posts, and presentations use the Adventure Works 2014 sample data.

Download Adventure Works sample projects and databases on Codeplex.

A new collection of sample data, Wide World Importers, was introduced for SQL Server 2016. You can use Wide World Importers sample data to learn about and test Analysis Services; however, no tutorials, examples, or documentation are provided. The Wide World Importers dataset and schema do not support many of the features in Analysis Services. To learn more, see Wide World Importers.

See also

- Analysis Services tutorials

Analysis Services PowerShell Reference

6/25/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Analysis Services PowerShell cmdlets are included in the SqlServer module.

> **NOTE**
> Azure Analysis Services database operations use the same SqlServer module as SQL Server Analysis Services. However, not all cmdlets are supported for Azure Analysis Services. To learn more, see Manage Azure Analysis Services with PowerShell.

Analysis Services Cmdlets

Analysis Services provides cmdlets correspond to methods in the **Microsoft.AnalysisServices** namespace. The following table describes each cmdlet and provides a link to the corresponding AMO method.

If you want to use PowerShell to perform a task that is not represented in the following list (for example to create or synchronize a database), you can write TMSL or XMLA script for that action, and then execute it using the **Invoke-ASCmd** cmdlet.

CMDLET	DESCRIPTION	EQUIVALENT AMO METHODS
Add-RoleMember cmdlet	Add a member to a database role.	Add
Backup-ASDatabase cmdlet	Backup an Analysis Services database.	Database.Backup
Invoke-ASCmd cmdlet	Execute a query or script in XMLA or TSML (JSON) format.	Execute
Invoke-ProcessASDatabase	Process a database.	Process
Invoke-ProcessCube cmdlet	Process a cube.	Process
Invoke-ProcessDimension cmdlet	Process a dimension.	Process
Invoke-ProcessPartition cmdlet	Process a partition.	Process
Invoke-ProcessTable cmdlet	Process a table in a Tabular model, compatibility model 1200 or higher.	Process
Merge-Partition cmdlet	Merge a partition.	Merge
New-RestoreFolder cmdlet	Create a folder to contain a database backup.	RestoreFolder
New-RestoreLocation cmdlet	Specify one or more remote servers on which to restore the database.	RestoreLocation

CMDLET	DESCRIPTION	EQUIVALENT AMO METHODS
Remove-RoleMember cmdlet	Remove a member from a database role.	Remove
Restore-ASDatabase cmdlet	Restore a database on a server instance.	Restore

PowerShell Reference for Power Pivot for SharePoint

12/7/2018 • 2 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

This section lists the PowerShell cmdlets used to configure or administer a Power Pivot for SharePoint installation. For more information about enabling the cmdlets and viewing built-in help, see Power Pivot Configuration using Windows PowerShell.

> **NOTE**
> This article may contain outdated information and examples. Use the Get-Help cmdlet for the latest.

Cmdlet List

To see a list of cmdlets from the PowerShell prompt:

1. Open SharePoint Management Shell using the **Run as Administrator** option.
2. Enter the following command: `Get-help *powerpivot*`

CMDLET	SUPPORTED VERSIONS
Get-PowerPivotServiceApplication cmdlet	SharePoint 2013 | SharePoint 2016
Get-PowerPivotSystemService cmdlet	SharePoint 2013 | SharePoint 2016
Get-PowerPivotSystemServiceInstance cmdlet	SharePoint 2013 | SharePoint 2016
New-PowerPivotServiceApplication cmdlet	SharePoint 2013 | SharePoint 2016
New-PowerPivotSystemServiceInstance cmdlet	SharePoint 2013 | SharePoint 2016
Remove-PowerPivotServiceApplication cmdlet	SharePoint 2013 | SharePoint 2016
Remove-PowerPivotSystemServiceInstance cmdlet	SharePoint 2013 | SharePoint 2016
Set-PowerPivotServiceApplication cmdlet	SharePoint 2013 | SharePoint 2016
Set-PowerPivotSystemService cmdlet	SharePoint 2013 | SharePoint 2016
Update-PowerPivotSystemService cmdlet	SharePoint 2013 | SharePoint 2016

Note: The following Cmdlets only worked with SharePoint 2010, which SQL Server 2016 (13.x) does not support.

- Get-PowerPivotEngineService
- Get-PowerPivotEngineServiceInstance
- New-PowerPivotEngineServiceInstance
- Remove-PowerPivotEngineServiceInstance

- Set-PowerPivotEngineService
- Set-PowerPivotEngineServiceInstance
- Update-PowerPivotEngineService

Data Mining Extensions (DMX) Reference

6/7/2018 • 4 minutes to read • Edit Online

APPLIES TO: SQL Server Analysis Services Azure Analysis Services

Data Mining Extensions (DMX) is a language that you can use to create and work with data mining models in Microsoft SQL Server Analysis Services. You can use DMX to create the structure of new data mining models, to train these models, and to browse, manage, and predict against them. DMX is composed of data definition language (DDL) statements, data manipulation language (DML) statements, and functions and operators.

Microsoft OLE DB for Data Mining Specification

The data mining features in Analysis Services are built to comply with the Microsoft OLE DB for Data Mining specification.

The Microsoft OLE DB for Data Mining specification defines the following:

- A structure to hold the information that defines a data mining model.
- A language for creating and working with data mining models.

The specification defines the basis of data mining as the data mining model virtual object. The data mining model object encapsulates all that is known about a particular mining model. The data mining model object is structured like an SQL table, with columns, data types, and meta information that describe the model. This structure lets you use the DMX language, which is an extension of SQL, to create and work with models.

For More Information: Mining Structures (Analysis Services - Data Mining)

DMX Statements

You can use DMX statements to create, process, delete, copy, browse, and predict against data mining models. There are two types of statements in DMX: data definition statements and data manipulation statements. You can use each type of statement to perform different kinds of tasks.

The following sections provide more information about working with DMX statements:

- Data Definition Statements
- Data Manipulation Statements
- Query Fundamentals

Data Definition Statements

Use data definition statements in DMX to create and define new mining structures and models, to import and export mining models and mining structures, and to drop existing models from a database. Data definition statements in DMX are part of the data definition language (DDL).

You can perform the following tasks with the data definition statements in DMX:

- Create a mining structure by using the CREATE MINING STRUCTURE statement, and add a mining model to the mining structure by using the ALTER MINING STRUCTURE statement.
- Create a mining model and associated mining structure simultaneously by using the CREATE MINING MODEL statement to build an empty data mining model object.

- Export a mining model and associated mining structure to a file by using the EXPORT statement. Import a mining model and associated mining structure from a file that is created by the EXPORT statement by using the IMPORT statement.
- Copy the structure of an existing mining model into a new model, and train it with the same data, by using the SELECT INTO statement.
- Completely remove a mining model from a database by using the DROP MINING MODEL statement. Completely remove a mining structure and all its associated mining models from the database by using the DROP MINING STRUCTURE statement.

To learn more about the data mining tasks that you can perform by using DMX statements, see Data Mining Extensions (DMX) Statement Reference.

Back to DMX Statements

Data Manipulation Statements

Use data manipulation statements in DMX to work with existing mining models, to browse the models and to create predictions against them. Data manipulation statements in DMX are part of the data manipulation language (DML).

You can perform the following tasks with the data manipulation statements in DMX:

- Train a mining model by using the INSERT INTO statement. This does not insert the actual source data into a data mining model object, but instead creates an abstraction that describes the mining model that the algorithm creates. The source query for an INSERT INTO statement is described in <source data query>.
- Extend the SELECT statement to browse the information that is calculated during model training and stored in the data mining model, such as statistics of the source data. Following are the clauses that you can include to extend the power of the SELECT statement:
  - SELECT DISTINCT FROM <model > (DMX)
  - SELECT FROM <model>.CONTENT (DMX)
  - SELECT FROM <model>.CASES (DMX)
  - SELECT FROM <model>.SAMPLE_CASES (DMX)
  - SELECT FROM <model>.DIMENSION_CONTENT (DMX)
- Create predictions that are based on an existing mining model by using the PREDICTION JOIN clause of the SELECT statement. The source query for a PREDICTION JOIN statement is described in <source data query>.
- Remove all the trained data from a model or a structure by using the DELETE (DMX) statement.

To learn more about the data mining tasks that you can perform by using DMX statements, see Data Mining Extensions (DMX) Statement Reference.

Back to DMX Statements

DMX Query Fundamentals

The SELECT statement is the basis for most DMX queries. Depending on the clauses that you use with such statements, you can browse, copy, or predict against mining models. The prediction query uses a form of SELECT to create predictions based on existing mining models. Functions extend your ability to browse and query the mining models beyond the intrinsic capabilities of the data mining model.

You can use DMX functions to obtain information that is discovered during the training of your models, and to

calculate new information. You can use these functions for many purposes, including to return statistics that describe the underlying data or the accuracy of a prediction, or to return an expanded explanation of a prediction.

For More Information: Understanding the DMX Select Statement, General Prediction Functions (DMX), Structure and Usage of DMX Prediction Queries, Data Mining Extensions (DMX) Function Reference

Back to DMX Statements

See Also

Data Mining Extensions (DMX) Function Reference
Data Mining Extensions (DMX) Operator Reference
Data Mining Extensions (DMX) Statement Reference
Data Mining Extensions (DMX) Syntax Conventions
Data Mining Extensions (DMX) Syntax Elements
General Prediction Functions (DMX)
Structure and Usage of DMX Prediction Queries
Understanding the DMX Select Statement

Multidimensional Expressions (MDX) Reference

10/26/2018 • 2 minutes to read • Edit Online

In This Section

TOPIC	DESCRIPTION
MDX Syntax Elements (MDX)	Describes the various syntax elements available in the MDX language for MDX expressions, statements, and scripts.
MDX Language Reference (MDX)	Describes the statements, operators, and functions that define the MDX language.

See Also

Querying Multidimensional Data with MDX
Analysis Services Scripting Language (ASSL for XMLA)
Retrieving Data from an Analytical Data Source
Define Assignments and Other Script Commands