PUBLIC

SAP BusinessObjects Business Intelligence platform

Document Version: 4.3 – 2020-06-12

Information Design Tool User Guide

© 2020 SAP SE or an SAP affiliate company. All rights reserved.

THE BEST RUN SAP

Content

1 What's new? . . . 11

2 Getting started. . . . 12

2.1 What is the information design tool? . . . 14

Who uses the information design tool? . . . 15

2.2 Starting the information design tool. . . . 15

Setting the information design tool to start in Set Editing Mode only . . . 16

Starting the information design tool in set editing mode. . . . 17

2.3 Working with the interface. . . . 17

Welcome page. . . . 19

Resetting the user interface display. . . . 20

Preferences. . . . 20

Help. . . . 22

2.4 Using the sample universe. . . . 23

2.5 Using extensions in the tool. . . . 24

3 Converting universe design tool universes. . . . 25

3.1 Benefits of converting .unv universes. . . . 25

In a repository. . . . 28

Locally. . . . 29

3.2 Supported features. . . . 31

3.3 Resolving conversion errors. . . . 35

4 Working with universes. . . . 38

4.1 Relational universe. . . . 38

4.2 Multi-dimensional (OLAP) universe. . . . 39

4.3 Multi-source universe. . . . 40

4.4 Specific data sources. . . . 40

SAP BW. . . . 40

SAP HANA. . . . 46

SAP ERP. . . . 52

Microsoft Analysis Services (MSAS). . . . 53

Essbase. . . . 55

SAS. . . . 57

4.5 Working with the New Universe wizard. . . . 57

Select or create a project in the New Universe wizard. . . . 58

Select the data source type in the New Universe wizard. . . . 58

Select or create a relational connection in the New Universe wizard. . . . 59

Select or create an OLAP connection in the New Universe wizard. 59
Select or create a data foundation in the New Universe wizard. 60
4.6 Creating universes. 60

5 Managing your projects and resources. 64
5.1 Working with projects. 64
Working with local projects and resources. 64
Recovering universe resources in local projects after workspace corruption. 68
Working on shared projects. 69
What is project synchronization?. 71
Saving resources as PDF, HTML, or Text files. 77
5.2 Working with repository resources. 78
Managing repository resources. 78
Managing a session. 80
Running a query on a universe published in a repository. 81
Locking universes. 82

6 Accessing databases through connections. 83
6.1 What are connections?. 83
Local connections. 84
Secured connections. 85
Connection shortcuts. 86
6.2 Working with the Connection Editor. 87
6.3 Creating and editing connections. 87
Relational connection. 87
OLAP connection. 109
Local and secured connections. 120
Connection shortcuts. 121
Editing connection shortcuts. 122

7 Designing a data foundation. 123
7.1 Getting started with data foundations. 123
Single-source data foundations. 124
Multisource-enabled data foundations. 124
Building a data foundation. 126
Data foundation properties. 129
7.2 The Data Foundation Editor. 133
Arranging tables in the Data Foundation Editor. 134
Changing the display of objects in the Data Foundation Editor. 135
Views. 138
7.3 Connections. 142
Adding connections to a data foundation. 144

Changing a connection in a data foundation. . . . 145
Selecting delimitation overrides to keep. . . . 145
Searching for tables in the Connection pane. . . . 146
Filtering the tables in the connection by table type. . . . 147
Filtering tables in an SAP HANA connection by information model. . . . 148
7.4 Table elements. . . . 149
Tables. . . . 149
Table row counts. . . . 158
Inserting a calculated column. . . . 159
Inserting a time column. . . . 160
Input columns. . . . 160
Showing table values. . . . 162
Showing column values. . . . 163
Profiling column values. . . . 164
Alias tables and derived tables. . . . 164
7.5 Managing tables. . . . 170
Refreshing a data foundation. . . . 171
Filtering the data foundation. . . . 173
Searching for tables and columns in the data foundation. . . . 173
Showing local dependencies in the data foundation. . . . 174
7.6 Joins. . . . 174
Joins. . . . 175
Cardinality. . . . 179
Contexts. . . . 181
Using parameters, derived tables, and lists of values in the data foundation. . . . 184
7.7 Optimizing. . . . 185
Loops. . . . 186
Chasm traps. . . . 188
Fan traps. . . . 190
7.8 Stored procedures in the data foundation. . . . 191
Inserting a Stored Procedure. . . . 193
Editing values for Stored Procedures Input Parameters. . . . 193
Refreshing the structure of a data foundation based on stored procedures tables. . . . 193
Stored procedure security right. . . . 195

8 Working with the federation layer. . . . 196
8.1 What is the federation layer?. . . . 196
8.2 Building the federation data flow. . . . 196
8.3 Federated tables. . . . 197
Adding a federated table manually. . . . 199
Adding a federated table from a data source. . . . 199
Editing a federated table. . . . 200

8.4 Input tables and joins. . . . 200
Adding input tables to a mapping. . . . 201
Joining input tables. . . . 202
Configuring meanings of input table joins using core tables. . . . 202
8.5 Mappings in the federation layer. . . . 204
Mapping columns from the input table to columns of the federated table. . . . 204
Editing a mapping formula. . . . 205
About the SQL Expression Editor. . . . 205
Adding a mapping. . . . 206
Activating and deactivating mappings. . . . 206
8.6 Distinct rows on input tables. . . . 207
Activating and deactivating distinct rows. . . . 207
8.7 Pre-filters and post-filters. . . . 207
Adding and editing pre-filters. . . . 208
Editing post-filters. . . . 208
8.8 Showing values in a federated table. . . . 209
8.9 Checking integrity of the federation layer. . . . 209
8.10 Inserting a federated table into the data foundation. . . . 210
8.11 Refreshing the structure of the federation layer. . . . 210

9 Working with the business layer. . . . 212
9.1 Getting started with business layers. . . . 212
Business layers. . . . 212
Business layer views. . . . 213
Business layer objects. . . . 215
Properties. . . . 218
9.2 Building a business layer. . . . 225
Working with the Business Layer Editor. . . . 225
Building a relational business layer. . . . 227
Building an OLAP business layer. . . . 231
9.3 Managing a business layer. . . . 238
Parameters 238
Queries in a business layer. . . . 242
Computing statistics for optimized query execution 243
Stored procedures in the business layer. . . . 243
Linking universes. . . . 244
Refreshing business layers. . . . 254
Index awareness. . . . 256
Analytic functions. . . . 257
Aggregate awareness. . . . 262
9.4 Working with business layer objects. . . . 264
Inserting a folder. . . . 265

Duplicating folders in the business layer. 266
Inserting and editing dimensions. 266
Inserting dimensions directly from the data foundation. 268
Defining keys for dimensions and dimension attributes. 269
Turning an attribute or measure into a dimension. 269
Inserting and editing measures. 270
Turning a dimension or attribute into a measure. 273
Inserting and editing attributes. 274
Turning a dimension or measure into an attribute. 276
Inserting and editing filters. 276
Inserting and editing analysis dimensions. 278
Inserting and editing hierarchies. 279
Inserting and editing hierarchy levels. 280
Inserting and editing named sets. 282
Inserting and editing calculated members. 283
Defining the SQL expression for an object. 284
Defining the MDX expression for an object. 285
Associating extra tables. 285
Changing the state of an object: Active, Hidden, or Deprecated. 286
Setting object access levels. 286
Setting where objects can be used. 287
Setting options for the default list of values. 287
Creating and editing display formats for business layer objects. 288
Source information for business layer objects 295
Inserting and editing custom properties. 295
Showing associated objects. 295
Showing business layer object values. 296
Filtering the business layer. 296
Searching for business layer objects. 297
Navigation paths for objects. 297
Duplicating business objects. 298
Reordering objects in the Business Layer Editor. 299

10 Publishing resources and retrieving published universes. 300
10.1 Publishing resources. 300
Publishing a local connection to the repository. 301
Publishing a local universe to the repository. 302
Publishing a universe. 302
10.2 Retrieving a published universe. 304
Retrieving a published universe from the local file system. 305
Retrieving a published universe from a repository. 305

11 Using the Query Panel. . . . 307
11.1 Building a query. . . . 307
11.2 Setting query properties. . . . 308
11.3 Viewing and editing the query script. . . . 310
11.4 Filtering data in the Query Panel. . . . 310
How to build a business filter. . . . 310
Filtering data using prompts. . . . 312
11.5 Refreshing a query containing BEx variables. . . . 315
11.6 Profiling column values in the query panel. . . . 315
11.7 Working with the Member Selector. . . . 316
About selecting hierarchy members. . . . 316
Opening the Member Selector in the Query Panel . . . 317
Selecting hierarchy members. . . . 318
Selecting members by hierarchy relationship . . . 318
Selecting hierarchy members by level. . . . 319
Selecting named sets. . . . 320
Selecting calculated members. . . . 320
Searching for hierarchy members. . . . 321
Excluding hierarchy members. . . . 321
Defining a prompt to select members. . . . 322
Showing selected members in the Member Selector. . . . 322
Sorting hierarchy members. . . . 323
Setting display options. . . . 323
Showing estimated child count. . . . 323

12 Securing resources. . . . 324
12.1 Security Editor. . . . 325
Securing resources in the information design tool. . . . 325
Opening the Security Editor. . . . 326
Working with the Security Editor. . . . 327
12.2 Security Profiles. . . . 328
Securing a universe using Security Profiles. . . . 328
Managing Data Security Profiles. . . . 329
Changing Security Profile priority. . . . 335
Managing Business Security Profiles. . . . 335
Security Profile aggregation. . . . 341
Assigning Security Profiles to users. . . . 349
Displaying profiles assigned to a user and previewing net profiles. . . . 349
12.3 Locking universes. . . . 350
12.4 CMC rights for information design tool users. . . . 350

13 Segmenting your data with Sets. . . . 353

13.1 Introduction to sets. . . . 353
Overview of a set to filter life cycle. . . . 354
Prerequisites for working with sets. . . . 356
Collation enhancement for SQL Server. . . . 358
Restrictions when working with sets. . . . 358
13.2 Setting up the sets services on the CMC. . . . 359
13.3 Managing set containers. . . . 360
Creating a set container from a universe. . . . 361
Retrieving a set container to build a set. . . . 363
Advanced subjects creation. . . . 364
13.4 Designing sets. . . . 364
Overview of how to create a set. . . . 365
Designing and managing sets using the sets editor. . . . 367
Working with sets in set editing mode. . . . 369
Editing set display options. . . . 369
Using scoring in your sets. . . . 370
Specifying how set filters are used (Consumption tab). . . . 370
Previewing a set as a filter in the query panel. . . . 372
Filtering objects in the set editor. . . . 372
Building a set. . . . 372
13.5 Publishing a set container. . . . 391

14 Tools and Commodities. . . . 392
14.1 Finding and replacing text. . . . 392
Finding and replacing text . . . 392
Where you can search in the data foundation. . . . 392
Where you can search in the business layer. . . . 393
Finding and replacing object names in the data foundation and the business layer. . . . 394
14.2 Showing dependencies between resources. . . . 395
Resource dependencies. . . . 395
Showing local dependencies. . . . 397
Showing repository dependencies. . . . 398
14.3 Checking the integrity. . . . 398
14.4 Reviewing check integrity problems. . . . 400
14.5 Multilingual universes. . . . 400
Translating universe metadata. . . . 402

15 Preferences. . . . 404
15.1 Business Layer Editor preferences. . . . 404
15.2 Check integrity settings. . . . 405
Check integrity default settings. . . . 406
15.3 Universe conversion data format. . . . 409

15.4 Data Foundation Editor preferences. . . . 409
Setting connection display preferences for the Data Foundation Editor. . . . 409
Setting Summary tab preferences for data foundation views. . . . 410
Setting table and join detection options. . . . 412
Setting default join states for contexts. . . . 413
Setting performance-related options for the data foundation view. . . . 413
Setting the upper case and lower case naming. . . . 414
Preferences for inserting tables and joins. . . . 414
15.5 Languages used by the information design tool. . . . 415
15.6 Linking to online tutorials. . . . 415
15.7 Publish/retrieve options for the business layer. . . . 416
15.8 Middleware driver for secured relational connections. . . . 416
15.9 Preferences for the Show Values command. . . . 417
15.10 Sets Editor default options. . . . 417
15.11 Default font for scripts and SQL expression editors. . . . 418
15.12 Display preferences for the data foundation view. . . . 418
15.13 Low memory warning parameters. . . . 419
15.14 Preferences for inserting tables and joins. . . . 420

16 SQL and MDX reference. . . . 421
16.1 The SQL/MDX expression editor. . . . 421
16.2 SAP BusinessObjects SQL function reference for multisource-enabled universes . . . 423
Aggregation functions. . . . 424
`Cast`. . . . 428
`If Else`. . . . 428
`If Null (nvl)`. . . . 429
`Timestamp Diff`. . . . 430
Database functions. . . . 431
16.3 @Functions. . . . 489
@Aggregate_Aware. . . . 490
@DerivedTable. . . . 490
@Execute . . . 491
@Prompt. . . . 492
@Select. . . . 497
@Variable. . . . 498
@Where. . . . 499
16.4 SQL generation parameters. . . . 500
SQL generation parameters reference. . . . 500
SQL generation parameters set in the extended PRM. . . . 514
16.5 Formula language for prompts. . . . 522
Setting dynamic default values for prompts by editing a parameter definition. . . . 522

Setting dynamic default values directly in a universe @prompt 523
Basic Elements for Prompt Formula Language. 524
Operator Expressions for Prompt Formula Language. 527
Function Expressions for Prompt Formula Language. 530
System Functions for Prompt Formula Language. 534
If then Else Expression for Prompt Formula Language. 535

17 Reporting and running queries in the BI repository. 536
17.1 Overview of CMS reporting. 536
CMS reporting sample kit. 537
17.2 CMS database connection. 539
17.3 Creating a CMS report. 540
17.4 CMS architecture and structure. 540

1 What's new?

Links to information about the new features and documentation changes for the information design tool for version 4.3 of SAP BusinessObjects BI Platform.

Information Design Tool Enhancements

What's new	Link to more information
64-bit operating system support.	Architecture tiers
Calculated columns can now be used in lists of values in the *SQL Expression Editor* of the data foundation and business layer.	Using parameters, derived tables, and lists of values in the data foundation [page 184]
You can now use mapped SAP HANA variables.	Using mapped SAP HANA variables [page 52]
SNC Encryption support on SAP BW OLAP connections.	OLAP connection [page 109]
Multidimensional universes on SAP BW InfoProviders and on S/4HANA CDS views set as cubes.	SAP BW [page 40]
You can now enable SQL access on SAP HANA HTTP connections. This gives an increase in performance when refreshing queries.	Configuring SQL access on SAP HANA HTTP [page 115]
More functionality for extensions.	Using extensions in the tool [page 24]

2 Getting started

The information design tool provides the following design resources to extract metadata and build universes.

Resource	Description
Project	A project is a named local workspace that contains the resources used to build one or more universes. A project can be shared so that multiple designers can work on the same resources. A project can contain any number of independent resources, for example data foundations, business layers, and connections. All resources contained within a project can be used interchangeably, for example a connection can be used by several data foundations within the same project. Projects and their resources are displayed in the Local Projects View. To open a resource in the editor, double-click the resource in the Local Projects View.
Connection OLAP Relational	A connection is a named set of parameters that define how a universe can access a relational or OLAP data source. A universe is always associated with at least one connection. A connection is an independent resource and can be used by several universes. You can build a multisource-enabled universe that references one or more relational connections. Connections can be local (stored in a local file) or secured (an object in a shared repository that is referenced by a connection shortcut). Local connections are stored in the local project as .cnx files.
Connection shortcut	A connection shortcut is an object in the local project that references a secured connection in a repository. You use a connection shortcut to refer to secure connections when creating data foundations and business layers based on secure connections. Connection shortcuts are stored in the local project as .cns files.
Data foundation	A data foundation is a schema that defines the relevant tables and joins from one or more relational databases. You enhance the data foundation by adding federated tables, derived tables, alias tables, calculated columns, additional joins, contexts, prompts, lists of values, and other SQL definitions. The data foundation becomes the basis of one or more business layers. Data foundations are stored in the local project as .dfx files.

Resource	Description
Business layer	A business layer is a collection of metadata objects that provides an abstraction of relational database entities or OLAP cubes, understandable by a business user. Objects map via SQL expressions to an underlying data foundation, or via MDX expressions to an underlying OLAP cube. These objects include dimensions, hierarchies, measures, attributes, and predefined conditions. You can add dimensions, hierarchies, measures, attributes, and other objects as the universe design requires. You can validate the SQL or the MDX at any time. You can create queries, lists of values, parameters (also called prompts), and navigation path objects. The business layer is the universe under construction, and when the business layer is complete, it is compiled with the connections or connection shortcuts and data foundation, published, and deployed as a universe. Business layers are stored in the local project as .blx files.
Query	A query is a set of objects that define a request to the database for data. A query can be defined and saved in the business layer as a metadata object to be used to test objects in the business layer.
Parameter List of values	A parameter is a variable in the universe that requires a value at query time. Parameters are often defined to prompt the user to supply a value, and in this case are referred to as prompts. A list of values is a collection of data values that can be associated with an object in the universe, allowing the user to choose values for a prompt. Parameters and lists of values can be defined in the data foundation. They are inherited by all business layers based on that data foundation. Parameters and lists of values can also be defined in the business layer.
Universe	A universe is a compiled file that includes all resources used in the definition of the metadata objects built in the design of the business layer. The universe is used by SAP BusinessObjects data analysis and reporting applications, where the business layer objects are visible for analysis and reporting. Universes are stored either locally or in a repository as .unx files.
Security Profiles Data Business	A security profile is a group of security settings that controls the data and metadata that are displayed to users and modifies the parameters defined in the data foundation and/or business layer. Security profiles are defined on published universes and stored in the repository.

Related Information

Creating universes [page 60]
Working with local projects and resources [page 64]

What are connections? [page 83]
Getting started with data foundations [page 123]
Business layers [page 212]
Queries in a business layer [page 242]
Parameters [page 238]
Securing resources [page 324]

2.1 What is the information design tool?

The information design tool an SAP BusinessObjects metadata design environment that enables a designer to extract, define, and manipulate metadata from relational and OLAP sources to create and deploy SAP BusinessObjects universes.

A universe is an organized collection of metadata objects that enable business users to analyze and report on corporate data in a non-technical language. These objects include dimensions, measures, hierarchies, attributes, pre-defined calculations, functions, and queries. The metadata object layer, called the business layer, is built on a relational database schema or an OLAP cube, so the objects map directly to the database structures via SQL or MDX expressions. A universe includes connections identifying the data sources so queries can be run on the data.

The role of the universe is to provide the business user with semantically understandable business objects. The user is free to analyze data and create reports using relevant business language regardless of the underlying data sources and structures.

Universes created using the information design tool can be used by the following SAP data analysis and reporting applications starting with version BI 4:

- SAP BusinessObjects Web Intelligence
- SAP Crystal Reports for Enterprise
- SAP Lumira
- SAP Predictive Analysis
- SAP Design Studio

> **i Note**
>
> Check the documentation for an application for any restrictions on accessing universes.

To enable the designer to create universes, the information design tool provides the resources necessary to do the following:

- Create connections to data sources.
- Extract a complete OLAP cube schema.
- Extract tables and joins to build a relational schema called a data foundation.
- Create metadata objects from the cube or the data foundation. These objects are contained and organized in a business layer. The SQL and MDX expressions within objects can be validated and queries run against the target databases to test the business layer.
- Share resources to allow multiple designers to work on the same resources concurrently.
- Publish a universe, which compiles the business layer, the data foundation, and the connections into a single universe file (.unx):

  - Publish a universe to a repository to be implemented in deployments of SAP BusinessObjects data analysis and reporting applications.
  - Publish a universe locally, to be implemented by client applications (for example Web Intelligence Rich Client).
- Create security profiles to define user access to universe data and metadata.

Related Information

Who uses the information design tool? [page 15]
Getting started [page 12]
Starting the information design tool [page 15]
Creating universes [page 60]

2.1.1 Who uses the information design tool?

The universe designer may be a database administrator, an applications manager or developer, a project manager, or a report creator who has acquired enough technical skills to create universes for other users. A security administrator also uses the information design tool to define universe security profiles.

There can be more than one universe designer in a company. The number of universe designers depends on the company's data requirements. For example, one universe designer could be appointed for each application, project, department or functional area.

Related Information

Getting started [page 12]
Starting the information design tool [page 15]
Creating universes [page 60]

2.2 Starting the information design tool

The information design tool is installed with the SAP BusinessObjects Business Intelligence Platform Client Tools. For more information on installing the BI Platform Client Tools, see the *SAP BusinessObjects Business Intelligence Suite Master Guide*, or the *SAP Crystal Reports 2020 User Guide*.

Once the client tools are installed on your machine, for example in a Windows installation of the BI Platform, you can start the information design tool with the command: *Start* > *All programs* > *SAP Business Intelligence* > *SAP BusinessObjects BI Platform 4 Client Tools* > *Information Design Tool*.

No authentication is required to use the information design tool in offline mode (not connected to a repository). You can begin creating and editing local resources.

> **i Note**
>
> When you login to the CMC, you can now render your password text visible by clicking the eye icon at the side of the login password field.

> **i Note**
>
> When you double-click a resource file in the local file system (for example a .blx, .dfx, or .cnx file), the information design tool opens without opening the specific resource editor. You must open the editor from the Local Projects View.

Related Information

Working with the interface [page 17]
Getting started [page 12]
Creating universes [page 60]
Securing resources in the information design tool [page 325]

2.2.1 Setting the information design tool to start in Set Editing Mode only

The information design tool can be configured to operate in set editing mode.

You can set the client machines of the information design tool to operate in Set editing mode only. This configuration allows users to access the CMC repository and perform Sets and Groups editing operations in the Set container that you created:

- Create, edit, delete, and build sets
- Create, edit, delete, and build groups
- Save and Publish sets and groups

To do this, you modify the `.ini` file of the information design tool so that the tool opens up in the Set editor mode.

To set the information design tool client machines to open in simplified mode, open, modify, and distribute the `InformationDesign.ini` file. Set command line option: `-Dsets.simplified.perspective=true`, then distribute the file.

> **i Note**
>
> You cannot open a repository session with another user ID.

Related Information

Starting the information design tool in set editing mode [page 17]

2.2.2 Starting the information design tool in set editing mode

You use the information design tool to operate in simplified Set editing mode only. This configuration allows the set editor to do the following:

- Create a set
- Edit a set
- Delete a set
- Build a set
- Create, Edit, and Delete groups of sets.
- Publish a set

To do this, you modify the `.ini` file of the information design tool so that the tool opens up in the Set editor mode.

1. Start the information design tool with the command: *Start* > *All programs* > *SAP Business Intelligence* > *SAP BusinessObjects BI platform 4 Client Tools* > *Information Design Tool*.

   The set authoring login displays.
2. Enter your credentials. If you have previously worked in a set container, the login will automatically connect you to the set container in the CMC, otherwise, browse to the Set Container. Click *Finish*.

The information design tool set editor displays. The local or remote resources are not displayed. You can start editing sets.

Related Information

Introduction to sets [page 353]

2.3 Working with the interface

Welcome page

The first time you start the information design tool, you see the *Welcome* page. From the *Welcome* page you can access all the resource creation wizards, open existing resources, and link to help and training materials.

You can close the *Welcome* page to display the information design tool interface. To re-open the *Welcome* page, select *Help* > *Welcome*.

Views and editors

The information design tool interface is composed of views and editors that let you navigate and work on different resources. For more information about each view, see the Related Topics.

View	Description	How to open the view
Local Projects View	Use this view to create and navigate local projects, and to open and validate resources.	Select *Window* > *Local Projects*.
Repository Resources View	Use this view to navigate repository resources and create secured connections.	Select *Window* > *Repository Resources*.
Data Foundation Editor	Use this editor to define and maintain the data foundation structure and its connections, and to access the data federation layer.	Double-click a data foundation in the Local Projects View.
Business Layer Editor	Use this editor to define and maintain the business layer and its data source.	Double-click a business layer in the Local Projects View.
Connection Editor	Use this editor to edit connection and connection shortcut parameters.	Double-click a connection or connection shortcut in the Local Projects View. To open the editor for a secured connection, double-click the connection in the Repository Resources View.
Project Synchronization View	Use this view to manage shared project resources in the repository and synchronize local resources with the repository.	Select *Window* > *Project Synchronization*.
Check Integrity Problems	Use this view to review the results of last integrity check.	Select *Window* > *Check Integrity Problems*.
Query Panel	Use this view to run queries on business layers and published universes.	To run a query on a business layer, in the Business Layer Editor Query pane, edit an existing query, or create a query. To run a query on a published universe, in the Repository Resources View, right-click a universe and select *Run Query*. You can also run a query on a published universe in the Security Editor.
Security Editor	Use this editor to define security profiles and assign profiles to users.	Select *Window* > *Security Editor*.

Related Information

Resetting the user interface display [page 20]
Working with local projects and resources [page 64]
Managing repository resources [page 78]
The Data Foundation Editor [page 133]
Working with the Business Layer Editor [page 225]
Working with the Connection Editor [page 87]
What is project synchronization? [page 71]
Checking the integrity [page 398]
Using the Query Panel [page 307]
Working with the Security Editor [page 327]

2.3.1 Welcome page

The Welcome page consists of three areas designed to help you use the tool:

Welcome page resources

Area	Description
New resource wizards	You can use the wizards listed in the *Welcome* page to help you create local resources in the information design tool. Watch this video to see how. To get help on a particular page of any wizard, click the help icon in the wizard dialog box. The wizards are also available from the *New* menu in the main tool bar. To see a list of all wizards, select *New* > *Other*. To start a wizard, select it in the list and click *Next*.
Open resource	Open Project allows you to import an existing project from another source. Universe to Convert allows you to open a session and convert a .unv universe to .unx format. Watch these videos to see how, both for local universes and repository universes . Recent Resources allows you to open a recently edited local resource.

Area	Description
Help	Use this area to access the different resources designed to help you use the tool: • Online tutorials: • Help: content from the information design tool in-build help, or your own help. You can configure it in the *Window* > *Preferences* > *Help*. • About: gives information about this version of the information design tool • Cheat Sheets: gives you access to guided processes for creating resources. Watch this video to see how.

Related Information

Getting started [page 12]
Creating universes [page 60]

2.3.2 Resetting the user interface display

The information design tool user interface can be customized by dragging and dropping editor tabs and views, minimizing views, and hiding and splitting panels within the views.

To reset the user interface to the default configuration, select *Window* > *Reset to Default Display*.

2.3.3 Preferences

To set preferences, from the information design tool main menu, select *Window* > *Preferences*. The following table gives a brief description of the types of Preferences [page 404] you can set. For more information, see the detailed section dedicated to preferences and the related topics.

Preference type	Description
General	These settings and customizations require a good understanding of the Eclipse development environment. To learn more about Eclipse, search for the Eclipse Foundation Web site.
Help	Lets you select how you would like the help topics to display when you click the help icon. *>Content*: You can use these settings to include custom help files. These settings require an understanding of the Eclipse help system. To learn more about Eclipse, search for the Eclipse Foundation Web site.

Preference type	Description	
Information Design Tool	*Expression fields and memory management*	The main Information Design Tool Preferences panel lets you set the font style and size for expression editors, and also define low memory parameters.
	Business Layer Editor	Lets you change how object names are generated in relational business layers.
	Check Integrity	Lets you set the integrity rules to run automatically when saving resources. You can also set the severity level of the rules.
	Convert Universe	Lets you set the data type format for the date when you convert a UNV universe to UNX.
	Data Foundation Editor	Lets you set display options for connections in the *Data Foundation Editor*. > *Appearance*: Set display options for columns, tables, and joins in the *Data Foundation Editor*. > *Detections*: Set whether or not to automatically detect tables, joins, and cardinalities when inserting tables into the data foundation. Set the default join state for contexts when adding joins and contexts to the data foundation. > *Insertions*: Set tables and joins are shown when you insert them in the data foundation. > *Performance*: Set options that impact how graphics are displayed in the *Data Foundation Editor*. > *Set Upper and Lower Case Action*: Set if the *Set to Upper* and *Set to Lower* actions also apply to owner and qualifier names.
	Languages	Lets you change the language of the user interface and the Preferred Viewing Locale.
	Online Tutorials	Lets you update the link to the online tutorials.
	Publish/Retrieve	Lets you customize the way you publish the BLX business layer, and gives you options to prefix a folder name that you retrieve..
	Secured Connections	Lets you set whether to use the server or the local middleware driver for secured relational connections. **i Note** This preference applies only if the *Download connection locally* right is granted in the Central Management Console for the connection.

Preference type	Description	
	Sets Editor	Lets you set the following options: • *Warn user when rebuilding or purging a published Set*. • *Automatically apply a Yes/No condition ('fx') when adding numeric or date objects*. • Choose to *Use classic Venn Diagram (ellipses)* for Sets on Sets steps
	Show Values	Lets you set how to display table and column values for the Show Values commands.

Related Information

Preferences [page 404]
Languages used by the information design tool [page 415]
Low memory warning parameters [page 419]
Business Layer Editor preferences [page 404]
Publish/retrieve options for the business layer [page 416]
Check integrity settings [page 405]
Universe conversion data format [page 409]
Setting connection display preferences for the Data Foundation Editor [page 409]
Display preferences for the data foundation view [page 418]
Setting Summary tab preferences for data foundation views [page 410]
Preferences for inserting tables and joins [page 414]
Setting table and join detection options [page 412]
Setting default join states for contexts [page 413]
Setting performance-related options for the data foundation view [page 413]
Setting the upper case and lower case naming [page 414]
Linking to online tutorials [page 415]
Middleware driver for secured relational connections [page 416]
Sets Editor default options [page 417]
Preferences for the Show Values command [page 417]

2.3.4 Help

The *Help* menu in the information design tool lets you link to different types of user assistance for the application.

Help menu command	Description
Welcome	From the *Welcome* page you can access all the resource creation wizards, open existing resources, and link to help and training materials.
Cheat Sheets	A cheat sheet shows how to complete a complex task, for example, creating a relational universe. The *Cheat Sheets* command displays a list of available cheat sheets. Double-click a cheat sheet to start. It opens in the information design tool Help View. For some steps, you can click *Click to perform* and the application will start the appropriate wizard to help you complete that step. To get more help on a step, click the help icon .
Online Tutorials	The *Online Tutorials* command links to the list of official product tutorials for the information design tool on the SAP Community Network. If the URL address of the online tutorials changes, you can enter the new address in the information design tool preferences. See Linking to online tutorials [page 415] to learn more.
Help Contents	The *Help Contents* command opens the *Information Design Tool User Guide* in a help window. To display help topics, you can navigate the Table of Contents, search the text, or look up topics in the index. **i Note** See the SAP Help Portal for the most updated version of the guide at http://help.sap.com/.
Search	The *Search* command opens the Help View on the search function. To search the contents of the *Information Design Tool User Guide*, enter text in the *Search expression* field.

Related Information

Linking to online tutorials [page 415]

2.4 Using the sample universe

You can use the `SPL_Warehouse.unx` sample universe in the information design tool if it has been installed along with the BI Platform server. The universe is stored in the `/Universes/Samples/` folder of the CMS repository.

Do the following:

1. Go to `<bip-install-dir>\Samples\splwarehouse` folder on the BI Platform server machine.
2. The name of the instance database is SPL_Warehouse. Only one SQL Anywhere database with this name can run on the network. In case of name conflict, open the `SPL_Warehouse.conf` configuration file for editing and modify the `SPL_Warehouse` name in the `-n` option of the file.
3. Start the SQL Anywhere database by running the `launch_splwarehouse_database` script.
4. If you have modified the database name, connect to the CMS repository in the information design tool and modify the database parameter in the SPL_Warehouse connection.

See the *Business Intelligence Platform Installation Guide* for more information about the sample installation.

2.5 Using extensions in the tool

You can create your own extensions to use with the information design tool. The extensions must conform to the model as described in the *Information Design Tool Extensions Document*. When you create the extension and its associated icon, you drop the extension into the `plugins` folder. You can access the extension in the following ways:

- For extensions created for the *Local Projects* view, right-click on the object for which the extension has been developed.
- For extensions created for the *Repository Resources* view, right-click on the object for which the extension has been developed.
- Also, in the *Repository Resources* view, the extension icon is visible in the menu bar of the view.

To see which extensions are currently in use by the information design tool, select *Help* > *Extensions*. The *Extensions list* dialog displays a list of extensions that are currently in use by the information design tool.

3 Converting universe design tool universes

The .unv universes are universes that were created with the universe design tool. You cannot work with them directly in the information design tool, you must convert them to .unx format.

Related Information

Benefits of converting .unv universes [page 25]
Supported features [page 31]
Resolving conversion errors [page 35]
In a repository [page 28]
Locally [page 29]

3.1 Benefits of converting .unv universes

You cannot work with the .unv universes created by the universe design tool, must convert them. You can use the *Convert .unv Universe* command in the information design tool to convert a universe created with other SAP BusinessObjects universe design tools, and universes created in previous versions. You can then work on the converted universe in a local project as you would a universe created with the information design tool.

What is a .unv universe?

A .unv universe refers to a universe created with any SAP Business Objects XI 3 design tool, for example Universe Designer.

The following SAP Business Objects BI 4 design tools create .unv universes:

- The universe design tool (new name for Universe Designer)
- The universe design tool desktop edition (new name for Universe Designer Personal)

The universe is stored with a file name of `<universe name>.unv` in a local folder, or in a repository.

What is a .unx universe?

When you publish a universe using the information design tool, the universe is stored with a file name of `<universe name>.unx`. This is called a .unx universe. The *Convert .unv Universe* command converts a .unv universe to the .unx universe format.

Which .unv universes can be converted?

You can convert the following types of .unv universes:

- Relational universes that were created using the SAP BusinessObjects BI 4 tools universe design tool or universe design tool desktop edition.
- Relational universes that were created using SAP BusinessObjects Enterprise XI 3 design tools.

  > **i Note**
  >
  > Before you can convert universes created in version XI 3 that are saved in a repository, you must upgrade the universes using the upgrade management tool. For more information, see the *SAP BusinessObjects Business Intelligence Platform Upgrade Guide*.

- You can convert .unv stored procedure universes to .unx. Any list of values objects are replaced by prompt parameters.
- You can convert linked universes. The core universe (.unx) is generated only in the same folder as the original core universe (.unv). When derived universes are stored in different folders, but share the same core universe, the core universe is no longer duplicated in each folder, but only duplicated in the same folder as the corresponding source core universe (.unv).

  > **i Note**
  >
  > Other derived, converted universes (.unx) must reference this same core converted universe (.unv).

You cannot convert the following types of .unv universes:

- OLAP universes
- Universes based on a Data Federator data source
- Javabean universes

> **i Note**
>
> It is not possible to convert Business Views created with Business View Manager XI 3 to a format compatible with version BI 4 reporting tools.

How to convert .unv universes

How you convert .unv files depends on the software version of the tool that was used to create the universe, and whether the universe is stored locally or in a repository. The following table describes the steps to follow for different conversion scenarios. For details of the conversion procedures, see the Related Topics.

Universe to convert	Workflow
A .unv universe that has been saved in a repository using XI 3 design tools.	First, upgrade the universe in the repository to the latest version using the upgrade management tool. In the information design tool, follow the procedure for converting a .unv universe in a repository. The conversion creates an equivalent .unx universe in the repository with the associated universe and connection rights.
A .unv universe that has been saved in a repository using the universe design tool release BI 4.0 or later.	In the information design tool, follow the procedure for converting a .unv universe in a repository. The conversion creates an equivalent .unx universe in the repository with the associated universe and connection rights.
Any locally-stored .unv universe that has been created using version XI 3 or later design tools. **i Note** A locally-stored universe refers to a non-secured universe that was saved for all users.	In the information design tool, follow the procedure for converting a locally-stored .unv universe. The conversion creates the equivalent universe resources (data foundation, business layer, and local connection) in a local project. Publish the business layer to create the .unx universe.

Converting date to date or to dateTime:

- The `Date` data type format for .unv is converted to the `Date` format for .unx. (This is the default setting.)
- The `Date` data type format for .unv is converted to the `DateTime` format for .unx

See Universe conversion data format [page 409]

to learn more.

After converting .unv universes

When converting a .unv universe, the .unv universe is preserved. Documents in SAP BusinessObjects query and reporting tools based on the universe are still linked to the .unv universe. This gives you the opportunity to check and test the converted universe before changing the documents that depend on it.

Some features of .unv universes are implemented differently in the .unx universe. Once you have converted a universe, you can edit the universe resources in a local project in the information design tool to check and correct inconsistencies, and to take advantage of new universe features. For a description of the supported features and how they are implemented in .unx universes, see the related topic.

After converting the universe, it is recommended to refresh the structure of the data foundation and run a check integrity on the universe. For tips on resolving check integrity errors on converted universes, see the related topic.

Related Information

In a repository [page 28]
Locally [page 29]
Supported features [page 31]
Resolving conversion errors [page 35]
Universe conversion data format [page 409]

3.1.1 In a repository

You can convert a single or a group of multiple .unv universes to .unx universe format from different repository folders. If the .unv universe was created with a design tool version earlier than SAP BusinessObjects BI 4.0, you must first upgrade the universe using the upgrade management tool. For more information about upgrading universes, see the *SAP BusinessObjects Upgrade Guide*.

1. In the information design tool, select *File* > *Convert .unv Universe* > *From a repository*.

   > **i Note**
   >
   > You can also convert a single .unv universe by directly right-clicking it and selecting the *Convert .unv Universe* command.

2. Open a session on the repository where the .unv universe is saved.

   The *Convert .unv Universe* dialog box appears.

3. In the left side pane, browse to the repository folder that contains one or multiple .unv universes to convert.
4. Select one or more universes. Click the right arrow to list the universes in the right side pane, then click *OK*.

   A conversion options box appears. The selected universes are listed with the repository file paths. You have the following conversion options:

Conversion option	When selected
Keep the same destination folder	The converted .unx universes are created in the same folder as the .unv source universes. **i Note** If you have selected the option *Overwrite existing files*, all universes with the same name will be replaced by their latest version. If you have not selected this option, then the new .unx universe is renamed automatically to keep the existing version.
Change the destination folder	You select a different repository folder to receive the converted .unx universes.

Conversion option	When selected
Automatically convert @PROMPT expressions into universe named parameters	The current version of the information design tool uses named parameters, so @PROMPT expressions will be converted. For more information about named parameters, see the related topic below.
Include core universes with converted universe	If the .unv universe is a derived universe, then one single .unx universe is created. It includes all of the objects from the linked core universe in a single business layer. All of the tables and joins from the core universes in a single data foundation. **i Note** Don't select this option if you want to convert core and derived universes to .unx format and preserve the core and derived links. This option is not selected by default.
Overwrite existing file(s)	Any .unx universes in the target repository folder with the same name as the new .unx universe are overwritten.

5. Select your preferred conversion options.
6. Click *OK* to start the conversion.
7. When the conversion is over, double-click on a converted universe to retrieve it to your local project. In the *Select a Local Project* dialog, navigate to the project where you want to retrieve the converted universe(s), and select the options *Save for all users*, and *Lock Universe*, as required.

After conversion, it is recommended to refresh the structure of the data foundation, then run a check integrity on the universe to detect problems in the conversion. For tips on resolving check integrity errors, see the related topic.

Related Information

Supported features [page 31]
Opening a session [page 80]
Creating a local project [page 65]
Parameters [page 238]
@Prompt [page 492]
Refreshing a data foundation [page 171]
Checking the integrity [page 398]
Resolving conversion errors [page 35]

3.1.2 Locally

You need to define a local project folder in the Local Projects View in which to save the resources of the converted universe.

1. In the information design tool, select *File* > *Convert .unv Universe*.

2. In the *Convert a .unv Universe* dialog box, click the *Select .unv universe from the local file system* icon , select the universe you want to convert, and click *OK*.

   A conversion options box appears. The selected universes are listed with the repository file paths. You have the following conversion options:

Conversion option	When selected
Keep the same destination folder	The converted .unx universes are created in the same folder as the .unv source universes. **i Note** If the folder contains a .unx universe with the same name as the newly converted universe it will be replaced by the new .unx universe if you have selected the option *Overwrite existing files* described later in the table. If you have not selected this option, then the new .unx universe is renamed automatically to keep the existing version.
Change the destination folder	You select a different local folder to receive the converted .unx universes.
Automatically convert @PROMPT expressions into universe named parameters	The current version of the information design tool uses named parameters, so @PROMPT expressions will be converted. For more information about named parameters, see the related topic.
Include core universes with converted universe	If the .unv universe is a derived universe, then one single .unx universe is created. It includes all of the objects from the linked core universe in a single business layer. All of the tables and joins from the core universes in a single data foundation. **i Note** Don't select this option if you want to convert core and derived universes to .unx format and preserve the core and derived links. This option is not selected by default.
Overwrite existing files	Any .unx universes in the target folder with the same name as the new .unx universe are overwritten.

3. Select your preferred conversion options.
4. Click *OK* to start the conversion.

   The conversion creates the equivalent universe resources (data foundation, business layer, and local connection) in the specified local project folder.

   At this point, it is recommended to refresh the structure of the data foundation.

   You can now publish the business layer to create the .unx universe file. This creates a local universe. To publish the universe to the repository, continue with the next step.
5. Publish the local connection to a repository.

6. Edit the data foundation and change the connection to use the secured connection published in the last step.
7. Publish the business layer to the repository.

The Publish Wizard allows you to run a check integrity on the universe (recommended). For tips on resolving check integrity errors, see the related topic.

Related Information

Supported features [page 31]
Creating a local project [page 65]
Parameters [page 238]
Refreshing a data foundation [page 171]
Publishing a local connection to the repository [page 301]
Changing a connection in a data foundation [page 145]
Publishing a universe [page 302]
Checking the integrity [page 398]
Resolving conversion errors [page 35]

3.2 Supported features

When converting a .unv universe with the information design tool, the conversion process creates equivalent features in the converted universe. The table below indicates which .unv universe features are supported and how they are implemented in the .unx universe. For certain features, tips are described for obtaining best conversion results.

Feature in original .unv universe	Feature in converted .unx universe
Universe schema	The objects in the universe schema are created in the data foundation: • Tables • Alias tables • Derived tables (including nested derived tables) • Joins (including shortcut joins) • Self-joins (converted to column filters). `@Prompt` functions in self-join expressions may require manual intervention after conversion. See the related topic on resolving check integrity errors. • Contexts. Contexts are converted with all joins explicitly included or excluded. In the *Data Foundation Editor*, you can take advantage of the simplified contexts feature. You can manually restrict the context definition to the ambiguous parts of the schema using neutral joins. For more information about contexts, see the related topic below **! Restriction** When converting a .unv universe, SQL is generated in the data foundation for certain object definitions in the .unv universe (for example, table names). If the .unv object definition contains a reference to a business object in an @Prompt, an SQL list of values is generated in the data foundation. The list of values has the following restrictions: • If the .unv object is index-aware, the index awareness is not applied. • The Security Access Level of the .unv object is not applied. • Any table mapping restrictions on the .unv object are not applied in the Data Security Profile.
Universe outline	The objects in the universe outline are created in the business layer with all their properties: • Classes and sub classes (converted to folders) • Dimensions. For time hierarchies, a dimension is created for each active level in the hierarchy. • measures (including aggregation function) • details (converted to attributes) • conditions (converted to filters, including properties for mandatory filters)
Multilingual universe	All translated strings, language settings, and locale settings are converted.

Feature in original .unv universe	Feature in converted .unx universe
Linked universe (core and derived)	A core universe is converted like any .unv universe. The conversion of a core universe doesn't trigger the conversion of the derived universes that depend on it. A derived universe contains a link to a core universe. When you convert a derived universe, you have the following choices: • Convert all core universes that are linked to the derived universe. The links to the core universes are preserved, so that your .unv derived and core universe relationship is unchanged when converted to .unx format. This is the current default behaviour of the information design tool. • Convert only the selected derived universe and include all of the objects and tables from the linked core universes into one universe. This was the default behavior for previous versions of the information design tool. The data foundation of the converted universe contains all the tables and joins from all core universes and any tables, joins, or contexts that were defined in the derived universe. The business layer contains all classes, objects, and conditions from all core universes including the classes, objects and conditions that were defined in the derived universe.
Universe Controls Parameters: Query Limits	Query limits are converted. You can edit them in the business layer, if required.
Universe SQL Parameters: SQL restrictions	Query, multiple SQL statements, and Cartesian product controls are converted. You can edit the *Allow Cartesian products* and *Multiple SQL statements for each context* restrictions in the data foundation. Edit all other SQL restrictions in the business layer.
Strategies	.unx universes don't support custom strategies.
Universe Parameters: SQL generation parameters	The process does not convert the following customizations: • Customizations to SQL generation parameter settings in the PRM file • Customizations in the universe parameters. You must add customized the values to the converted PRM file, and customize universe parameter settings in the converted universe using the information design tool. Check and reset custom settings to SQL generation parameters in the data foundation properties and business layer properties. For information on setting SQL parameters, see the related topic.

Feature in original .unv universe	Feature in converted .unx universe
@Functions	The following @functions are converted: • `@Aggregate_Aware` • `@Prompt` • `@DerivedTable` • `@Select` • `@Variable` • `@Where` The syntax for all functions is supported. The `@Prompt` function has a new alternative syntax to take advantage of named parameters. For more information, see the related topic.
Prompts	`@Prompt` functions in dimension and measure expressions in the business layer, and in SQL expressions in the data foundation are supported. When converting, you have the option to automatically create a named parameter for the prompt in the business layer. `@Prompt` expressions in the data foundation are not converted. `@Prompt` functions in self-join expressions may require manual intervention after conversion. See the related topic on resolving check integrity errors. Parameters and lists of values in the information design tool can be defined independently from the objects that they reference. You can therefore reference a named parameter or list of values in more than one business layer object.
Lists of values	Named lists of values are created in the business layer for dimension and measure objects that specify a list of values.
Access Restrictions	When converting a universe in a repository, universe Access Restrictions are converted into Data Security Profiles that can be edited using the Security Editor: Special characters in the names of Access Restrictions are handled, with the exception of the colon ":" and the question mark "?". These characters are removed from the profile name. If the new name already exists, the modified name is appended with the lowest available integer (1, or higher if a same name already exists that ends with 1, and so on). • Access Restrictions (except object restrictions) are converted to settings in a Data Security Profile. • Object Access Restrictions are converted to Create Query and Display Data settings in a Business Security Profile. Business Security Profiles allow you to secure the metadata separately from the data. For example, you can allow a user to create a query although the user is not allowed to see the corresponding data. For more information about universe security in the information design tool, see the related topic.

Feature in original .unv universe	Feature in converted .unx universe
Security assignments and priority	When converting a universe in a repository, user and group assignments are converted. In the Security Editor, you can take advantage of the feature allowing you to assign more than one security profile to a user or group. Group priority for access restrictions is converted. For .unv universes, if a user belongs to different groups and has no access restriction assigned, the priority assigned to the groups determines what access restriction the user inherits. For .unx universes, the priority is assigned to Data Security Profiles instead of groups. If the priority for the profile assigned to the group is higher than the priority of the profile assigned to the user, the group profile is used.
Connections	Both .unv and .unx use the same secured relational connection when converting a universe in a repository. If you retrieve the converted universe into a local project, a connection shortcut is created that references the secured connection in the repository. Relational connections can be created and shared by the universe design tool and the information design tool. The connections are published in the same Connections folder in the repository. When converting a locally-stored (non-secured) universe, personal and shared connections are converted to local connections.

Related Information

Resolving conversion errors [page 35]
The Data Foundation Editor [page 133]
Contexts [page 181]
SQL generation parameters [page 500]
@Functions [page 489]
Working with the Business Layer Editor [page 225]
Securing resources [page 324]

3.3 Resolving conversion errors

After converting a .unv universe, it is recommended to run a check integrity in the information design tool on the converted universe. Certain errors in the check integrity results can be resolved by following the best practices described below.

Errors on column data types

Refresh the structure in the data foundation just after the conversion. This avoids data-type errors in the integrity check.

Errors on self-joins with `@Prompt`

If a join expression in the .unv universe contains an `@Prompt` with a list of values referring to an object, the converted join needs to be reworked in the data foundation. The steps to follow for two possible solutions are described. The descriptions use the following example:

The .unv universe contains a self-join on the **`dimProductStrings`** table with a prompt called **`Language`**. The self-join expression is:

```
dimProductStrings.LanguageID= @Prompt('Language','N','Language\Language
Id',mono,constrained)
```

After converting the universe, the data foundation contains a column filter on the **`dimProductStrings`** table. The join expression for the filter contains the `@Prompt`.

The first solution consists of creating a prompted parameter and list of values in the data foundation:

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View
2. Click the *Parameters and Lists of Values* tab.
3. In the data foundation *Lists of Values* pane, define a list of values based on custom SQL for Language. For example:
   `SELECT "LANGUAGES"."LANGUAGEID", "LANGUAGES"."LANGUAGECODE" FROM "LANGUAGES"`
4. In the data foundation *Parameters* pane, define a parameter for Language. Select the option *Prompt to users* and associate the Language list of values to it.
5. In the data foundation, edit the column filter in the **`dimProductStrings`** table. Change the join expression to reference the new prompted parameter, for example:
   `dimProductStrings.LanguageID` = `@Prompt(`**`Language`**`)`
6. Save and close the data foundation.

The second solution consists of using a mandatory filter in the business layer:

1. Open the data foundation in the editor and delete the column filter on the **`dimProductStrings`** table that contains the `@Prompt`.
2. Save and close the data foundation.
3. Open the business layer in the editor.
4. In the business layer *Lists of values* pane, define a list of values based on custom SQL for Language. For example:
   SELECT `"LANGUAGES"."LANGUAGEID", "LANGUAGES"."LANGUAGECODE" FROM "LANGUAGES"`
5. In the business layer *Parameters* pane, define a parameter for **`Language`**. Keep the default option to *Prompt to users* and associate the **`Language`** list of values to it.
6. In the business layer, in the folder associated with **`Product`**, create a filter with an expression that refers to the **`Language`** prompted parameter, for example:
   `dimProductStrings.LanguageID` = `@Prompt(`**`Language`**`)`

7. In the *Properties* tab of the filter definition, select the *Use filter as mandatory in the query* option. Select the *Filter scope* of *Apply on Folder*.
8. Save and close the business layer.

Related Information

Refreshing a data foundation [page 171]
The Data Foundation Editor [page 133]
Using parameters, derived tables, and lists of values in the data foundation [page 184]
Inserting a column filter [page 179]
Working with the Business Layer Editor [page 225]
Parameters [page 238]
Inserting and editing filters [page 276]

4 Working with universes

A universe is an abstraction of a data source that presents data to users in non-technical terms. Refer to the links below to see the different types of universes and how you can create a universe using the New Universe wizard, or by using the complete workflow.

You can use the *New Universe* wizard to create the resources you need to publish a local universe: either a single-source relational or OLAP universe. If you want to base your universe on secured connections, the connection shortcuts must exist in a local project. The wizard lets you create only local connections. For more information, see the related topic. To start the wizard, select *File* > *New Universe*.

Before you create a universe:

- Make sure the middleware drivers are configured for the data sources to which you want to connect.
  For more information about middleware configuration, see the *Data Access Guide*.
  For information about supported data sources, see the SAP BusinessObjects BI Platform 4.3 Supported Platforms (PAM) at https://www.sap.com.
- Make sure you have the appropriate rights defined in the Central Management Console (CMC). See the related topic about CMC rights for information design tool users.
- Decide if the data foundation type should be single-source or multisource-enabled. The type and number of connections available, as well as the SQL syntax that is used to define SQL structures depends on the data foundation type. For more information, see the related topic about data foundation types.
  Connections for multisource-enabled data foundations must be secured, relational connections, and are managed by the data federation service. Information about tuning the data federation service can be found in the *Data Federation Administration Tool Guide*.

Related Information

Relational universe [page 38]
Multi-dimensional (OLAP) universe [page 39]
Multi-source universe [page 40]
Specific data sources [page 40]
Working with the New Universe wizard [page 57]
Creating universes [page 60]

4.1 Relational universe

A relational universe is a graphical representation of dimensions and objects that allow business users to analyze business data. A relational universe is created on top of relational databases like SQL Server, Oracle and so on.

Linked universes

Linked universes are universes that share common components such as parameters, classes, objects, or joins. You can dynamically link one or more universes. When you link two universes, one universe has the role of a core universe, the other a derived universe. When changes are made in the core universe, they are automatically propagated to the derived universes.

The core universe is a universe to which other universes are linked. It contains components that are common to the other universes linking to it. These universes are called derived universes. The core universe represents a re-usable library of components.

A core universe can be a kernel or master universe depending on the way the core universe components are used in the derived universes. Kernel and master universes are described in the section Creating a link between two universes.

A derived universe is a universe that contains a link to a core universe. The link allows the derived universe to share common components of the core universe:

- If the linked core universe is a kernel universe, then components can be added to the derived universe.
- If the linked core universe is a master universe, then the derived universe contains all the core universe components. Classes and objects are not added to the derived universe. They can be hidden in the derived universe depending on the user needs of the target audience.

Stored procedure universes

You can create a universe on one or more stored procedures. A stored procedure is an encapsulated set of sql statements that is stored and run on a database. A typical universe is usually a read-only object, whereas a stored procedure universe allows you to manipulate the data before it is used in the universe and then in the reporting tool. This allows the SQL statements such as CREATE, DROP, INSERT, and UPDATE statements that are not permitted in a typical universe. The SQL statements are not stored or visible from the universe or from the reporting tool.

4.2 Multi-dimensional (OLAP) universe

A multi-dimensional universe, also named OLAP universe, mainly consists of dimension objects organized in meaningful hierarchies.

Multi-dimensional analysis allows users to observe data from various viewpoints. This enables the users to spot trends or exceptions in the data. A hierarchy is an ordered series of related dimensions. An example of a hierarchy is Geography, which may group dimensions such as Country, Region, and City.

You can drill up or down to perform multi-dimensional analysis.

You can now create an OLAP universe on the top of HANA HTTP Info Access (InA) connections. The universe is based on a specific HANA view.

> **i Note**
>
> In HANA HTTP InA universes, only the TEXT and KEY display modes are available for dimension presentation options.

4.3 Multi-source universe

A multi-source universe uses multiple data sources in a single universe. Note the following:

- This type of universe is a .unx universe (.unv universes do not support multiple data sources).
- The universe must be declared as multi-source when it is first created. A single source universe cannot be converted to a multi-source universe at a later date.
- The data sources must be relational data sources. These relational data sources do not have to be the same type of relational database management system (RDBMS). For instance, one could be Oracle and the other could be Microsoft SQL Server.
- A multi-source universe is primarily processed by an Adaptive Processing Server which is running the Data Federation Service, whereas a single source universe is processed by the Web Intelligence Processing Server.
- The connections must be secure connections and must be published to the BI platform.

4.4 Specific data sources

This section describes how to access data from specific data sources.

Related Information

SAP BW [page 40]
SAP HANA [page 46]
SAP ERP [page 52]
Microsoft Analysis Services (MSAS) [page 53]
Essbase [page 55]
SAS [page 57]

4.4.1 SAP BW

When working with SAP BW data sources you have the following possibilities to access your data depending on the type of connection:

Direct access to BW Info Providers and BEx Queries

SAP BusinessObjects query and reporting applications can use direct access to access data in a single BW Info Provider or BEx Query. You do not need to build a universe. Define an OLAP connection to SAP BW that uses the *SAP BICS Client* middleware driver. When defining the connection, select the option to specify a cube in the connection and select the BEx Query.

Multidimensional Business layers on BW Info Providers and BEx Queries

You can create a business layer directly on a BW Info Provider or BEx Query using*SAP BICS Client* and publish the business layer as an OLAP universe available to Web Intelligence.Creating an OLAP universe based on a BW Info Provider or BEx query has a major advantage over using direct access to a BW Info Provider or BEx query as you can organize and customize the query dimensions, measures, and hierarchies in the business layer.

Relational universes on SAP BW

To build a universe on SAP BW, you must create a multisource-enabled data foundation based on a secured relational connection to SAP BW. You then build the business layer on this data foundation. For detailed steps on how to build a universe, see the related topic.

> **i Note**
>
> For information on the authorizations needed to allow users of query and reporting applications to access multisource-enabled universes on SAP BW, see SAP Note #1465871.

Relational connections to SAP BW are managed by the data federation service. For information about optimizing queries, see the *Data Federation Administration Tool Guide*.

To see the InfoProviders supported for relational connections to SAP BW, see the Data Access for the Semantic Layer section of the SAP Business Objects BI Platform 4.3 Supported Platforms (PAM) at https://support.sap.com/pam.

When you add an SAP BW connection to a data foundation, by default tables and joins are automatically inserted. When you create a business layer on the data foundation, by default objects are automatically inserted into the business layer.

To turn off automatic insertion, unselect the *Detect tables* option in the advanced properties of the connection when you are adding the connection into the data foundation. To turn off automatic insertion of business layer objects, unselect the *Automatically create folders and objects* option when selecting the data foundation in the New Business Layer wizard.

See the related topic for information on how the objects in an InfoProvider map to the objects that are automatically inserted into the data foundation and business layer in the information design tool.

Refreshing relational universes based on SAP BW

When objects are added to the underlying InfoProvider, several commands exist to help you update the data foundation and business layer with the changes. The recommended procedure is described in the related topic

Working with SAP BW OLAP universes through connections based on SNC with encryption

When your connection is correctly configured, you can create or modify SAP BW OLAP universes.

Creating an OLAP universe on an SAP BW InfoProvider

You can create an authored BICS OLAP universe on the top of a BW InfoProvider with the information design tool. You create an OLAP BICS connection on a BW InfoProvider, then you can create an authored BICS OLAP universe on the top of a BW InfoProvider. The creation rules and workflows for universes on the top of BW InfoProviders are same to those done for Authored BICS Universes for BEx Queries. You must select a cube in order to create the universe.

> **i Note**
>
> BEx variables are authored in and for BEx queries. They are not available for BW InfoProviders, so the Set variables button is disabled in BICS universes on BW InfoProviders.

> **i Note**
>
> The information design tool does not support the BICS connections based on the prompt authentication mode.

Creating a universe using S4HANA CDS views as Info Providers

Create an OLAP BICS connection on a S4HANA InfoProvider, then create an OLAP BICS universe based on the S4HANA InfoProvider.

> **i Note**
>
> The information design tool does not support the BICS connections based on the prompt authentication mode.

Related Information

OLAP connection [page 109]
Building an OLAP business layer [page 231]
Creating universes [page 60]
How InfoProvider objects are mapped in a universe [page 43]
Refreshing universes based on SAP BW [page 46]
Creating a business layer directly on a BEx query [page 233]

4.4.1.1 How InfoProvider objects are mapped in a universe

When you add an SAP BW connection to a data foundation, by default tables and joins are automatically inserted. When you create a business layer on the data foundation, by default objects are automatically inserted into the business layer.

The following table describes how the objects in an InfoProvider map to the objects that are automatically inserted into the data foundation and business layer in the information design tool. The naming convention for the automatically generated data foundation tables are as follows:

- I-Table: A table with a name prefixed by I (InfoCube) that maps to the InfoProvider fact table.
- D-Table: A table with a name prefixed by D (Dimension) that maps to an InfoProvider master data table.
- T-Table: A table with a name prefixed by T (Text) that maps to an InfoProvider text table.

InfoProvider object	Data foundation	Business layer
Fact table	• Inserts an I-Table mapped to the fact table. • Inserts D-Tables mapped to each master data table. • Inserts T-Tables mapped to each text table.	
Dimension		Inserts a folder in the business layer for each InfoProvider dimension (except for the Data Package and Unit dimensions). Business layer objects for the characteristics in the Unit dimension are inserted into the folder for the related key figure. The Data Package dimension is not mapped.
Characteristic	Inserts a column in the data foundation fact table that maps to the master data value.	Inserts a dimension object in the dimension folder associated with the I-Table column.

InfoProvider object	Data foundation	Business layer
	Creates an alias of the associated T-Table and links it to the column in the I-table. The T-Table contains columns for the description of the master data in short, medium, or long format. The table also has a column CAPTION which contains the longest description available for the characteristic. **i Note** A characteristic of data type DATS or TIMS has no associated text table because date and time characteristics do not have descriptions.	Inserts attribute objects under the dimension for each column in the T-Table.
Display attribute*	If a characteristic contains at least one display attribute, inserts an alias of the D-Table. Inserts a column in this table mapped to the display attribute. The D-Table contains a column for each display attribute of the characteristic.	Inserts a folder under the dimension named for the parent characteristic. Inserts dimension objects in this folder for each column in the D-Table.
	Inserts an alias of the T-Table for each display attribute.	Inserts attribute objects under the dimension object for each column in the T-Table.
Navigation attribute*	Inserts a column in the I-Table and a column in the D-table mapped to the navigation attribute. T-Tables are linked directly to I-Table for both the parent characteristic and its navigation attributes.	Inserts a dimension object in the dimension folder associated with the I-Table column. The dimension object for the navigation attribute is at the same level as the dimension for the parent characteristic but not necessarily adjacent. **→ Tip** Filters on the object for the navigation attribute are more efficient than filters on the display attribute object. When filtering a navigation attribute, the fact table is filtered directly.

InfoProvider object	Data foundation	Business layer
Key figure	Inserts a column in the I-Table with the technical name of the key figure.	For a key figure without unit or currency, inserts a measure in the Measures folder. For a key figure with unit or currency, inserts a sub-folder in the Measures folder. Inserts a measure in the sub-folder for the key figure. Inserts a dimension in the sub-folder for each unit or currency characteristic.
Time-dependent data	If the InfoProvider contains time-dependent data, creates input columns in the appropriate data foundation tables to handle time-dependent data. Creates a parameter in the data foundation called key date for each input column. By default, at query time, the key date parameter is not prompted. It is automatically assigned the current date. You can change this behavior by editing the key date parameter. For more information on input columns and editing parameters, see the related topics.	The business layer inherits the key date parameter.

* If the flag "Attribute only" is checked for a display or navigation attribute in the InfoProvider, it is not exposed in the data foundation.

Related Information

SAP BW [page 40]
Business layer objects [page 215]
Input columns [page 160]
Inserting and editing a parameter [page 239]

4.4.1.2 Refreshing universes based on SAP BW

Use this procedure when objects change in the InfoProvider of an SAP BW data source and you want to reflect the changes in the universe. You can link to more information about each step in the Related Topics.

1. Refresh the structure of the data foundation.

   Refreshing the data foundation compares the existing tables in the data foundation with those in data source and proposes updates to the data foundation tables: deletes obsolete tables and columns, inserts missing columns, and updates changed columns.
2. Synchronize the tables in the data foundation.

   Synchronizing tables searches the data source for new tables (using the SAP BW strategy), and inserts the new tables and joins into the data foundation.
3. Save the data foundation.
4. Refresh the business layer by inserting candidate objects.

   Inserting candidate objects searches the data source for new objects (using the SAP BW strategy) and updates the business layer.

   Inserting candidate objects does not detect obsolete objects in the business layer. You must find and delete obsolete objects manually.

Inserting candidate objects updates the business layer independently of the data foundation. If you do not also refresh the structure and synchronize tables in the data foundation, you can introduce inconsistencies between the data foundation and business layer.

Related Information

Refreshing a data foundation [page 171]
Synchronizing tables [page 172]
Inserting candidate objects [page 255]
SAP BW [page 40]

4.4.2 SAP HANA

Direct access to SAP HANA information models

SAP Crystal Reports for Enterprise can use direct access to access data in a single information model (such as an Analytic View or Calculation View). You do not need to create a universe. Define an OLAP connection to SAP HANA that uses the *SAP HANA Client* middleware driver. In the connection, you can choose to not specify a cube. In this case, the user is prompted to select an information model at query run time. You can also specify an information model in the connection so that all queries are associated with that information model.

> **i Note**
>
> The information design tool now supports SAP HANA 2 multi-tenant data sources.

> **i Note**
>
> Input parameters that modify tables structures at runtime aren't supported.

Building universes on SAP HANA

Universes on SAP HANA are based on relational connections. There are two ways to create resources to build a universe on SAP HANA:

- Use the New Data Foundation and New Business Layer wizards to create a data foundation and business layer separately.
  This method allows you to include tables in the data foundation. You can also include information models.

  > **i Note**
  >
  > If your data foundation will include only tables, the connection to SAP HANA can use an ODBC middleware driver. As soon as you include an information model in the data foundation, the connection to SAP HANA must use a JDBC middleware driver.

  Create a single-source data foundation unless you need to federate data from multiple sources. A multisource-enabled data foundation is necessary if you need to access data in more than one SAP HANA server or instance.
  Select the tables or information models that you want to include in the data foundation. Any columns hidden in an SAP HANA view are also hidden in the data foundation table. It is not recommended to create joins between tables representing SAP HANA views as this can impact performance.
  The New Business Layer wizard automatically creates the dimensions and attributes in each information model in a business layer folder. For single-source data foundations, the wizard uses metadata from the SAP HANA view to create measures in the business layer with the appropriate aggregation function.
  For more information, see the related topic on how to build a universe.
- Use the New SAP HANA Business Layer wizard to automatically create a single-source data foundation and business layer based on selected SAP HANA information models.

  > **i Note**
  >
  > The connection to SAP HANA must use a JDBC middleware driver.

  This wizard creates the data foundation tables for the selected SAP HANA views. Any columns hidden in an SAP HANA view are also hidden in the data foundation table. The wizard then creates the dimensions and measures (with appropriate aggregation functions) defined in the SAP HANA views.
  The advantage of this method is that for any dimensions and attributes that are common to different views, the wizard creates a single business layer object.
  The wizard also creates contexts in the data foundation and aggregate awareness in the business layer so that when you run a query on the universe, the experience is the same as accessing a standard relational universe:
  - For all measures and dimensions common to the SAP HANA views accessed in the query, results display in a single block.
  - For dimensions that are not common to the SAP HANA views accessed in the query, results display in separate blocks.

For more information, see the related topic on creating relational resources on SAP HANA information models.

Browsing SAP HANA connections

When browsing the connection in the data foundation, different views are identified by table types each with its own icon. You can filter the tables in the connection by table type. There is also a filter, set by default, to show only tables representing information models. For more information, see the related topics about filtering tables in the connection.

> **i Note**
>
> An Analytic View sometimes appears as a Calculated View table type in the connection. This happens when the Analytic View contains a calculated measure in the SAP HANA model.

Showing values in a data foundation based on SAP HANA

In single-source data foundations, when showing table and column values for a table corresponding to an Analytic View, the information design tool aggregates the values in columns that represent measures by using the aggregation function defined in the model. The measure values are grouped by the selected columns that represent attributes. For example, if you show values for `Product` and `Sales` in an Analytic View where `Sales` is aggregated with the sum function, the results show `Sales` by `Product`.

SAP HANA variables in the data foundation

> **i Note**
>
> SAP HANA variables and input parameters are not supported in multi-source data foundations.

In single-source data foundations, variables and input parameters in SAP HANA information models are associated with the corresponding tables in the data foundation.

When showing values in the data foundation or running a query in the Query Panel, you are prompted to enter values for the variables and parameters.

You can get information about variables and parameters in the data foundation properties *Variables* tab. Also, when refreshing the structure of the data foundation, any variables in the view that are added, deleted, or modified are taken into account.

You may want to enhance the data foundation by creating derived objects: derived tables, calculated columns, or custom SQL lists of values. Since the SAP HANA variables are hidden, you need to manage the variables directly in the SQL expressions for the objects that you create. For recommendations and limitations on enhancing data foundations that contain SAP HANA variables, see SAP Note 1913504.

> **i Note**
>
> SAP HANA optional input parameters are not supported on derived tables. You receive an error message when you run Check Integrity to validate a data foundation that includes a derived table containing SAP HANA optional input variables.

If any SAP HANA variables or input parameters change in the underlying SAP HANA information model, you must run a refresh structure on the data foundation in the information design tool.

Related Information

OLAP connection [page 109]
Creating universes [page 60]
Creating relational resources on SAP HANA information models [page 49]
Filtering the tables in the connection by table type [page 147]
Filtering tables in an SAP HANA connection by information model [page 148]
Showing SAP HANA variable information [page 132]
Refreshing a data foundation [page 171]
Creating relational resources on SAP HANA information models [page 49]
Inserting an SAP HANA View in the data foundation [page 50]
SAP HANA variables in the data foundation [page 51]
Using mapped SAP HANA variables [page 52]

4.4.2.1 Creating relational resources on SAP HANA information models

The New HANA Business Layer wizard automatically creates a data foundation and business layer based on selected SAP HANA information models.

The wizard creates a single-source data foundation referring to the local connection to SAP HANA that you provide. The data foundation contains a table for each view. The tables are not joined.

The wizard creates a business layer that contains the dimensions and measures defined in the SAP HANA views. Dimensions and attributes that are common to different views are bound to a single business layer object.

Before you begin, you need the following resources:

- A local project
- In the local project, a local connection to SAP HANA.

> **i Note**
>
> The connection must use a JDBC middleware driver.

1. In the Local Projects View, right-click the project and select *New* > *SAP HANA Business Layer*.
2. Enter a name for the business layer and data foundation.

   By default, the data foundation has the same name. You can change the data foundation name.
3. Optionally enter a description for the business layer and click *Next*.
4. Select a connection and click *Next*.

   Only local, relational connections to SAP HANA in the local project are listed.
5. Select one or more activated views to be the basis of the business layer and click *Finish*.
   A data foundation and business layer are created in the local project. The business layer opens in the editor.

You can publish the business layer to a local folder. If you want to publish the universe to a repository, see the related topic.

If any SAP HANA variables or input parameters change in the underlying SAP HANA information model, you must run a refresh structure on the data foundation in the information design tool.

Related Information

Publishing a local connection to the repository [page 301]
Changing a connection in a data foundation [page 145]
Publishing a universe [page 302]
Publishing a local universe to the repository [page 302]
SAP HANA [page 46]
Refreshing a data foundation [page 171]

4.4.2.2 Inserting an SAP HANA View in the data foundation

You are creating a data foundation and business layer based on selected SAP HANA information models.

You can now create .unx universes based on the following SAP HANA views.

- Native HANA Analytical View
- Native HANA Calculation View
- HANA HDI Calculation View

You can do the following:

- Insert the views through the data foundation.
- Insert views with HANA variables.
- Browse the views through the data foundation.
- Generate business layers on the views.
- Run *Refresh Structure* on the views.
- Run *Check Integrity* on the views.
- Insert and browse views in the data foundation.

You have created a single-source data foundation referring to the local connection to SAP HANA that you provide. The data foundation contains a table for each view. The tables are not joined.

You create a business layer that contains the dimensions and measures defined in the SAP HANA views. Dimensions and attributes that are common to different views are bound to a single business layer object.

Before you begin, you need the following resources:

- A local project
- In the local project, a local connection to SAP HANA.

> **i Note**
>
> The connection must use a JDBC middleware driver.

1. In the Local Projects View, right-click the project and select *New* > *SAP HANA Business Layer*.
2. Enter a name for the business layer and data foundation.

   By default, the data foundation has the same name. You can change the data foundation name.
3. Optionally enter a description for the business layer and click *Next*.
4. Select a connection and click *Next*.

   Only local, relational connections to SAP HANA in the local project are listed.
5. Select one or more activated views to be the basis of the business layer and click *Finish*.
   A data foundation and business layer are created in the local project. The business layer opens in the editor.

You can publish the business layer to a local folder. If you want to publish the universe to a repository, see the related topic.

If any SAP HANA variables or input parameters change in the underlying SAP HANA information model, you must run a refresh structure on the data foundation in the information design tool.

Related Information

Publishing a local connection to the repository [page 301]
Changing a connection in a data foundation [page 145]
Publishing a universe [page 302]
Publishing a local universe to the repository [page 302]
SAP HANA [page 46]
Refreshing a data foundation [page 171]

4.4.2.3 SAP HANA variables in the data foundation

When a data foundation is based on a HANA view that has HANA variables, you can define the order of SAP HANA variables in the universe for your default queries.

1. In the *Variables* tab for the data foundation, click *HANA Variables General Order*.

2. In the HANA Variable Order dialog, select the *Custom order* option and select a variable, contiguous variables (shift-click), or several variables that are not contiguous (Ctrl-click).
3. Use the up/down arrows to move the selected variables.

You can return to the order determined by the SAP HANA view by selecting the *Apply default order* option.

4.4.2.4 Using mapped SAP HANA variables

You can now use mapped SAP HANA variables in your queries. Variable mapping is a powerful and useful feature that allows reuse of parameterized SAP HANA views. SAP HANA cascading variables (or nested variables) can be consumed in reports or stories.

This feature is available for:

- SAP HANA relational (UNX) universes based on SAP HANA relational connections
- SAP HANA OLAP (UNX) universes based on SAP HANA HTTP connections
- Native SAP HANA views and SAP HANA HDI views

When variables mappings are set, the Semantic Layer will run the SAP HANA variable list of values according to information provided in the variable mapping. At query time, you provide values for the variables according to your business case.

When you have a published SAP HANA calculation view containing attributes, measures, and SAP HANA variables, and one of your variables has an input parameter mapped to it, the tool now will run the SAP HANA variable list of values according to information provided in the variable mapping.

> **i Note**
>
> SAP HANA variable mapping is not supported for SAP HANA UNV universes or SAP HANA multisource universes.

4.4.3 SAP ERP

To build a universe on SAP ERP, you must create a data foundation on a relational ERP connection. Then build the business layer on this data foundation. For detailed steps on building a universe, see the related topic.

When you create a relational connection to the SAP ERP data source, the InfoSets, SAP Queries, and ABAP functions in the data source are exposed as tables in the connection. For more information about how ERP data sources are mapped in the connection, see the *Data Access Guide*.

The data foundation can be single-source to support local connections. Single-source data foundations support joins between tables, with the following restrictions:

- You can detect joins only based on database keys in the ERP data source. You must first detect database keys in the data foundation.
- You cannot insert joins manually, insert calculated columns, or insert column filters.

For support of calculated columns, filters, and manual joins, create a multisource-enabled data foundation on a secured connection.

When you insert a table into the data foundation, the table type of InfoSet, SAP Query, or ABAP Function is saved as a table property in the data foundation.

When you insert an ABAP Function table, one data foundation table is created to map the main function. The table contains input columns for the input parameters of the function. These parameters can be mandatory or optional. To assign a value to mandatory parameters, you need to edit the input columns. To do this, see the related topic.

When creating the business layer, the object names are automatically generated from the column descriptions in the data foundation, rather than from the column names. For reference, the column names are saved as the description of the business layer object.

> **! Restriction**
>
> Measures containing aggregate functions cannot be used as filters in the Query Panel. This limitation is due to the fact that the resulting SQL expression contains the `HAVING` clause, which is not supported by the SAP ERP connection. If you add a measure containing an aggregation function as a filter, an error occurs when you refresh the query.

Related Information

Creating universes [page 60]
Editing input columns [page 161]
Using parameters, derived tables, and lists of values in the data foundation [page 184]
Multisource-enabled data foundations [page 124]

4.4.4 Microsoft Analysis Services (MSAS)

When you create a business layer on an MSAS data source, the business layer objects are generated automatically.

The business layer can be refreshed to reflect changes in the underlying cube using the *Refresh Structure* command in the Business Layer Editor *Actions* menu.

The table below gives details about how certain objects in the MSAS cube are mapped in the business layer.

MSAS object	Business layer mapping
Perspective	When you create a business layer, in the *Select OLAP Connection* page of the New Business Layer wizard, the base cube in the MSAS data source is listed first in the list of connection cubes. Additional cubes and perspectives in the data source are mapped as cubes and are listed in alphabetical order. The cube you select in the list of connection cubes becomes the basis for the objects in the business layer.

MSAS object	Business layer mapping
Dimension	Analysis dimensions are created in the business layer for each dimension in the cube.
Display folder	Folders are created in the analysis dimension to contain the hierarchies in the display folder.
Hierarchy	For value-based (parent-child) hierarchies, a value-based hierarchy is created in the analysis dimension. The attributes are created in the *Attributes* folder in the hierarchy. **i Note** Unbalanced hierarchies are supported. For level-based hierarchies, a business layer dimension is created in the analysis dimension. A hierarchy is created in the business layer dimension with the levels and their properties (as level attributes) in the *Levels* folder.
Attribute hierarchy	Attribute hierarchies in the cube are created as level-based hierarchies in the analysis dimension.
Named set	Named sets are created in the related analysis dimension, in the folder *Named sets*.
Measure group	Folders are created to contain the measures in measure groups and subgroups.
Measure Calculated measure	Measures and calculated measures are created as measures in the appropriate measure group folder. A measure attribute is created for the formatted value.

MSAS object	Business layer mapping
KPI	KPIs are not exposed in connection metadata, but you can make the KPI values available to queries by creating measures in the business layer with the MDX functions `KPIValue` and `KPIGoal`. For example, if the cube contains a KPI with the name Operating Profit, you can create measures in the business layer with the following MDX expressions. In this example, the measures are created in the measure group folder\subfolder **`Performance\Profit`**. The name of the KPI value in the cube is surrounded by double quotes in the MDX function. **Measure name** / **MDX Expression** **`Actual Profit`** — `KPIValue("Operating Profit")` **`Profit Target`** — `KPIGoal("Operating Profit")` **`Profit Variance`** — `(@Select(Performance\Profit\Actual Profit) - @Select(Performance\Profit\Profit Target)) / abs(@Select(Performance\Profit\Profit Target))` **`Profit Pct Achieved`** — `IIF(ISEMPTY(@Select(Performance\Profit\Profit Target)), null, @Select(Performance\Profit\Profit Variance) +1)`
Visual totals	You can use the MSAS-specific function `VisualTotals` in the MDX expressions for business layer objects.

Related Information

Creating universes [page 60]
OLAP data source properties [page 220]
Business layer objects [page 215]
Inserting and editing measures [page 270]
Refreshing an OLAP business layer [page 255]

4.4.5 Essbase

When you create a business layer on an Essbase data source, the business layer objects are generated automatically.

The business layer can be refreshed to reflect changes in the underlying cube using the *Refresh Structure* command in the Business Layer Editor *Actions* menu.

> **! Restriction**
>
> When you insert an MDX object into an Essbase business layer (for example, a named set, calculated member, or measure), be sure that the object name is not the same as data in the cube. For example, if Region is the name of a hierarchical level in the cube, you cannot name a new MDX object Region. If the new object is given the same name as data in the cube, the object is not usable in a query.

The table below gives details about how certain objects in the Essbase cube are mapped in the business layer.

Essbase object	Business layer mapping
Dimension	Analysis dimensions are created in the business layer for each dimension in the cube.
Accounts dimension	In the New Business Layer wizard, you select which dimension to use to create the measures in the business layer. By default this is the dimension flagged as the Accounts dimension. Measures are created in the business layer for each object in the dimension. The organization of the measures in the Essbase outline is maintained in the business layer. **→ Tip** For certain applications you may want to specify a dimension other than the Accounts-type dimension for the measures. In this case, the measures are created as an analysis dimension in the business layer and the analysis capabilities of hierarchies, such as member selection, are available.
Hierarchy	For each hierarchy in the cube, a hierarchy is created in the analysis dimension. All hierarchies are generated as value-based. **i Note** When opening the business layer in the Query Panel, hierarchy levels are determined spontaneously and can be selected in the query. Also, you can insert levels into the business layer.
User Defined Attribute (UDA)	UDAs are created as named sets defined on the associated hierarchy and appear in the analysis dimension.
Attribute	Attributes are created in an Attributes folder in the hierarchy.
Attribute hierarchy	If the attributes are designed as a hierarchy in the cube, an attribute hierarchy is also created in the analysis dimension.
Dynamic Time Series (DTS)	A DTS is not generated automatically in the business layer, but you can use MDX functions, such as HTD, QTD (history-to-date, quarter-to-date) in the object definitions.

Essbase object	Business layer mapping

Substitution variable	Substitution variables are not exposed in the business layer, but you can invoke a substitution variable in an MDX expression. The substitution variable name must be prefixed with the ampersand (&) character.
	
	For example, if the cube contains a variable named **CurrentMonth**, you can use the variable in the definition of a named set:
	
	`WITH SET [Current Month] AS '{[Time].[&CurrentMonth ]}'`
	
	Example of the substitution variable in the definition of a calculated member:
	
	`WITH MEMBER [Measures].[Current Month Quantity] AS '([Measures].[Quantity Sold], [Time].[&CurrentMonth])'`

Related Information

Creating universes [page 60]
Business layer objects [page 215]
Refreshing an OLAP business layer [page 255]

4.4.6 SAS

To build a universe on SAS, you must create a multisource-enabled data foundation on a secured connection. Then build the business layer on this data foundation. For detailed steps on building a universe, see the related topic.

Connections to SAS are managed by the data federation service. For information about optimizing queries to SAS data sources, see the *Data Federation Administration Tool Guide*.

Related Information

Creating universes [page 60]

4.5 Working with the New Universe wizard

You can use the *New Universe* wizard to create the resources you need to publish a local universe: either a single-source relational or OLAP universe.

You can also select existing resources. At each step of the process, you can choose to create a resource or select an existing resource.

For more information on each step of the wizard, click the help icon.

To start the wizard, from the information design tool main menu, select *File* > *New Universe*.

Once you have completed the wizard, publish the universe to a local folder or repository.

Related Information

Publishing a universe [page 302]
Publishing a local universe to the repository [page 302]

4.5.1 Select or create a project in the New Universe wizard

Create a project if you want to build all the resources for your universe from scratch. All the resources used to build the universe must be located in the same local project.

If you want to base the universe on existing resources, choose the project that contains these resources.

Related Information

Working with local projects and resources [page 64]

4.5.2 Select the data source type in the New Universe wizard

A universe is based on a relational or an OLAP data source.

- Relational universes are based on a data foundation that defines the relevant tables and joins from one or more relational databases. The objects in the business layer map to the database structures via SQL expressions. In the next step, you will select or create a relational connection on which to base the data foundation.
- OLAP universes are based on a connection to an OLAP cube. The objects in the business layer map directly to the cube via MDX expressions. In the next step, you will select or create an OLAP connection on which to base the business layer.

> **i Note**
>
> HANA OLAP connections:
>
> - The New Universe creation wizard gives the option of choosing an OLAP Connection and creating a HANA OLAP universe. You can only do this when there is an associated cube. When there is no cube, this connection cannot be used by Web Intelligence and cannot be used by the information design tool to create a unx universe.

- When you use a HANA OLAP connection in Web Intelligence, Web Intelligence will create a transient (temporary) universe. This universe cannot be authored in information design tool except in Debug Mode for debugging purposes.

4.5.3 Select or create a relational connection in the New Universe wizard

The wizard lets you to create only local connections. If you want to base your data foundation on a secured connection, do one of the following:

- Select an existing secured connection.
- Cancel the wizard and use the New Relational Connection wizard to create a secured connection.
- Continue with the wizard and create a local connection. You can publish the connection and re-publish the business layer to a repository later.

Related Information

Relational connection [page 87]
Publishing a local universe to the repository [page 302]
Local connections [page 84]

4.5.4 Select or create an OLAP connection in the New Universe wizard

The wizard lets you to create only local connections. If you want to base your universe on a secured connection, do one of the following:

- Select an existing secured connection.
- Cancel the wizard and use the New OLAP Connection wizard to create a secured connection.
- Continue with the wizard and create a local connection. You can publish the connection and re-publish the business layer to a repository later.

Related Information

OLAP connection [page 109]
Publishing a local universe to the repository [page 302]
Local connections [page 84]

4.5.5 Select or create a data foundation in the New Universe wizard

The wizard lets you to create only single-source data foundations. If you want to create a multisource-enabled data foundation on a secured connection, do one of the following:

- Select an existing multisource-enabled data foundation. In the previous step you must select one of the secured connections referenced in the data foundation.
- Cancel the wizard and use the New Data Foundation wizard to create a multisource-enabled data foundation.

To learn more about single-source and multisource-enabled data foundations, see the related topics.

Related Information

Single-source data foundations [page 124]
Multisource-enabled data foundations [page 124]
Building a data foundation [page 126]

4.6 Creating universes

- Refer to the additional information available if you are creating a universe on one of the following data sources:
  - SAP BW [page 40]
  - SAP HANA [page 46]
  - SAP ERP [page 52]
  - Microsoft Analysis Services (MSAS) [page 53]
  - Essbase [page 55]
  - SAS [page 57]
  - Multilingual universes [page 400]

> **i Note**
>
> For OLAP universes, you do not need to create a data foundation. The business layer is built directly from the objects you select in the source cube.

The following procedure describes how to make any kind of universe from scratch. Links to more information on each step in the procedure can be found in the Related Topics.

> **i Note**
>
> Don't use the following characters in the universe name: `\ / : * ? " < > |`.

1. Create a local project. In the Local Projects View select *File* > *New* > *Project*.

The resources you use to build the universe are created and stored in the local project folder.

2. Define the connections. Connections can be local or secured:
   - Create a secured connection if you want to create a multisource-enabled universe, or if you want to publish the universe in a repository without first publishing locally. For secured connections, you must create connection shortcuts in the local project to reference the secured connections in the repository. Use a local connection if you want to publish the universe on the local file system. Later, you can publish the business layer to a repository.

   Use a local connection if

Option	Command
To create a local relational connection	In the Local Projects View, right-click the project folder and select *New* > *Relational Connection*.
To create a local OLAP connection	In the Local Projects View, right-click the project folder and select *New* > *OLAP Connection*.
To create a secured relational connection	In the Repository Resources View, start a repository session. Right-click the Connections folder or sub-folder and select *Insert Relational Connection*. To create a connection shortcut, select the connection in the Connections folder of the repository and select *Create Relational Connection Shortcut*.
To create a secured OLAP connection	In the Repository Resources View, start a repository session. Right-click the Connections folder or sub-folder and select *Insert OLAP Connection*. To create a connection shortcut, select the connection in the Connections folder of the repository and select *Create OLAP Connection Shortcut*.

3. Create the data foundation (relational data sources only). In the Local Projects View, right-click the project folder and select *New* > *Data Foundation*.
   - For a single-source universe, select a single connection identifying the database source.
   - To build a data foundation with multiple relational connections, create a multisource-enabled data foundation.

   The data foundation opens in the editor. To build the structure of the data foundation, see the related topic.

4. Create the business layer. In the Local Projects View, right-click the project folder and select *New* > *Business Layer*.
   - For relational business layers, select the data foundation to be the basis for the business layer. You can choose to automatically generate objects in the business layer for all data foundation structures, or select columns to be mapped as objects.

> **i Note**
>
> To build a universe on multiple data sources (relational data sources only), the business layer must be based on a multisource-enabled data foundation.

- For OLAP business layers, select the connection to the OLAP cube. Objects are created automatically for all structures in the cube.

The business layer opens in the editor. To build the business layer, see the related topic.

5. In the business layer, you can create and run queries to validate and test the universe.
6. Publish the business layer:
   - Business layers based on local connections must be published to a folder on the local file system. Later, you can publish the resulting local universe to a repository. See the related topic about publishing a local universe to the repository.
   - Business layers based on one or more secured connections must be published to the repository on the same Central Management System where the secured connections are stored.

Option	Command
To publish the universe locally	In the Local Projects View, right-click the business layer and select *Publish* > *To a Local Folder*.
To publish the universe to a repository	In the Local Projects View, right-click the business layer and select *Publish* > *To a Repository*.

7. Define universe security. To open the Security Editor, on the information design tool main menu, select *Window* > *Security Editor*. Open a session on the repository where the universe is published.

   Use the Security Editor to define security profiles on the published universe. You also use the Security Editor to assign profiles to users and groups.

Related Information

Working with the interface [page 17]
Getting started [page 12]
CMC rights for information design tool users [page 350]
Creating a local project [page 65]
Relational connection [page 87]
OLAP connection [page 109]
Connection shortcuts [page 121]
Building a data foundation [page 126]
Building a relational business layer [page 227]
Building an OLAP business layer [page 231]
Queries in a business layer [page 242]
Publishing a universe [page 302]
Publishing a local universe to the repository [page 302]

Securing a universe using Security Profiles [page 328]

5 Managing your projects and resources

This chapter describes how to work with local projects and resources, how to share them, and how to work with repository resources.

Related Information

Working with projects [page 64]
Working with repository resources [page 78]

5.1 Working with projects

You create and edit all resources (except secured connections and security profiles) in a local project in the local file system. To create or work with resources in the information design tool, you work in a local project folder in the *Local Projects* view. The local resources can be shared or published to a repository and shared in the repository.

5.1.1 Working with local projects and resources

The first step in creating resources in the information design tool is to create a local project in the Local Projects View. You create and edit all resources (except secured connections and security profiles) in a local project.

The resources and folders in a local project are stored as physical files and folders in the local file system. The Local Projects View lets you to navigate local projects and open resources in the information design tool.

Once you have created a local project, there are several ways you can populate it with resources:

- Create universe resources using the wizards available on the *New* menu.
- Convert a .unv universe that was created with the universe design tool, or migrated from an earlier version.
- Retrieve a published universe.
- Create folders to organize resources within the project.
- Create file resources by entering a file name and extension.

You edit the resources using the information design tool editors by double-clicking the resource name in the local project. To open a resource from a list of resources recently opened, select *File* > *Recent Resources* >.

You can get information on the resources you create by right-clicking the resource name and selecting *Properties*. The properties displayed include the path to the resource in the local file system and the date the resource was last modified.

You can also perform the following tasks on resources from the Local Projects View:

- Create a shared project so that you can share resources with other designers.
- Check integrity of data foundations and business layers.
- Edit and test local connections.
- Change and test the connection referenced by a connection shortcut.
- Publish a business layer as a universe to the local file system or a repository.
- Publish a connection to a repository.
- Show dependent resources.
- Save a resource as a report.

When you copy resources, it is best to copy the entire folder so that all the references between the resources are maintained. This is because the paths to referenced resources are relative, not absolute. The information design tool assumes that all resources that reference each other are in the same folder. If you copy a single resource to a location outside of the folder without copying the resources it references, the references are broken.

You can delete a project from the Local Projects View. The project files remain in the local file system until you explicitly delete them. Open the project to make it available again in the Local Projects View.

Related Information

Creating a local project [page 65]
Deleting a local project [page 67]
Opening a local project [page 67]
Getting started [page 12]
Finding universe resources in the local file system [page 66]
Benefits of converting .unv universes [page 25]
Retrieving a published universe from a repository [page 305]
Working on shared projects [page 69]
Checking the integrity [page 398]
Publishing resources [page 300]
Resource dependencies [page 395]
Saving resources as PDF, HTML, or Text files [page 77]

5.1.1.1 Creating a local project

1. From the information design tool main menu, select *File* > *New* > *Project*.
2. Give the project a unique name.

3. In *Project Location*, the file path to the default root directory for all projects (workspace) displays. To select a different local folder to contain the project, click the browse button .
4. Click *Finish*.

The project is created in the local file system and displayed in the Local Projects View.

Related Information

Working with local projects and resources [page 64]

5.1.1.2 What are resource names?

Resource names identify the connections, data foundations, and business layers in the local project. You give a name to the resource when you create it. The name must be unique within the local project.

> **i Note**
>
> If you use the same name for resources in different projects, when publishing the resources to the same repository you may have naming conflicts because the names are not unique.

You can optionally enter a description of the resource.

Related Information

Getting started [page 12]
Resource dependencies [page 395]

5.1.1.3 Finding universe resources in the local file system

1. In the Local Projects View, open the project that contains the universe resources.
2. Right-click a resource (for example, a data foundation, connection, or business layer) and select *Properties*.

   The properties displayed include the path to the resource in the local file system and the date the resource was last modified.

Related Information

Working with local projects and resources [page 64]

5.1.1.4 Opening a local project

Information design tool projects saved on the file system can be opened in the Local Projects View.

1. From the information design tool main menu, select *File* > *Open Project*.
2. Select the *Select root directory* option and click *Browse*.

   The *Browse for Folder* dialog box opens with the default root directory for all projects (workspace) already selected.
3. Click *OK* to select the default directory, or browse to the folder containing the project you want to open.

   All projects not already open in the Local Projects View are listed in *Projects* and are selected by default.
4. Clear the checkbox for any projects you do not want to open and click *Finish*.

Related Information

Working with local projects and resources [page 64]

5.1.1.5 Deleting a local project

1. Right-click the project in the Local Projects View and select *Delete*.
2. If you want to permanently delete the project from the Local Projects View and the local file system, select the *Delete project contents on disk* option in the *Confirm Project Delete* dialog box.

   > **i Note**
   >
   > If you select this option, the deletion is permanent and cannot be undone.
3. Click *Yes* to confirm the deletion.

If you did not delete the project contents permanently, you can open the project to make it available again in the Local Projects View.

Related Information

Opening a local project [page 67]

5.1.1.6 Searching for and filtering resources in the Local Projects View

1. To filter the types of resources displayed in the Local Projects View, click the filter icon in the icon bar of the view. Select the types to include or exclude.
   The Local Projects View displays only resources of the types selected.
2. To search the list, click the *Show/Hide search bar* icon .
3. In the search text box, enter the text and press the `Enter` key to start the search.
   The first resource name that contains the search text is highlighted in the view. The total number of resource names containing the search text displays in the search text box.

   > **i Note**
   >
   > The search does not highlight resource types that are unselected in the filter.

4. To highlight the next resource found, press the `Enter` key again. Use the `Enter` key to browse through all resource names that match the search text.

5.1.2 Recovering universe resources in local projects after workspace corruption

Sometimes when the information design tool ends unexpectedly, the local workspace becomes corrupted and you are unable to restart the information design tool. Use this procedure to recreate the workspace and recover your local projects.

1. On the local file system, navigate to the folder `%USERPROFILE%\.businessobjects\bimodeler_14\`.
   For example:

   `C:\Documents and Settings\Administrator\.businessobjects\bimodeler_14\`
2. Rename the `workspace` folder to `workspace.bak`.
3. Start the information design tool.

   A new workspace folder is created automatically.

   Somtimes you can recover project files by reopening them from the backup workspace (see step 4). Otherwise, projects can be manually recovered (see step 5).
4. To recover projects using the *Open Project* command:
   a. Select *File* > *Open Project* from the main menu.
   b. In the *Import Existing Projects* dialog box, select the root directory and browse to the path of the `workspace.bak` folder created in step 2.
   c. Select the projects you want to restore.
   d. Select the *Copy projects into workspace* option and click *Finish*.
5. To recover projects manually:
   a. Open a system file and folder manager (for example, Windows Explorer) and navigate to the path of the `workspace.bak` folder created in step 2.

b. Copy and paste project folders from the `workspace.bak` folder to the new `workspace` folder created in step 3.

5.1.3 Working on shared projects

A shared project is a project in a repository whose resources are available to other designers. You create a shared project in a repository from an existing local project in the Local Projects View.

To start working on shared resources, you use the following tasks in the Project Synchronization View:

- Synchronize the project to copy resources between the local and shared projects.
- Lock and unlock resources in the shared project to inform other designers when you are working on them.
- Synchronize a shared project created by another designer. This creates a local project associated with the shared project so that you can start working on the shared resources.

Related Information

Creating a shared project from a local project [page 69]
Working in a shared project [page 70]
Synchronizing a project [page 74]
Locking a resource [page 75]
Unlocking a resource [page 76]
Merging changes to shared resources [page 77]

5.1.3.1 Creating a shared project from a local project

The shared project will automatically be given the same name as the local project. A project with this name cannot already exist in the repository. If you need to rename an existing shared project, use the rename command in the Project Synchronization View. To do this, see the related topic.

1. In the Local Projects View, right-click the project you want to share and select *New Shared Project*.
2. In the *Open Session* dialog box, select the repository system session you want to open and enter your system authentication.
   The Project Synchronization View opens showing a shared project with the same name as the local project. At this point, the shared project is empty.
3. In the Project Synchronization View, synchronize the resources you want to save in the shared project.

Related Information

Renaming a shared project [page 70]

Opening a session [page 80]
Synchronizing a project [page 74]

5.1.3.2 Working in a shared project

Use this procedure to work on resources in an existing shared project.

1. Open the Project Synchronization View with a session on the repository system where the shared project is saved.
2. Select the shared project from the *Shared Project* list.
3. In the selected shared project, lock the resources you want to work on.

   Locks are available as a communication tool between designers. When other designers open the Project Synchronization View, your lock informs them that you are making changes. It also prevents other designers from updating these resources in the shared project while you have them locked. However, any designer can unlock the resource if necessary.
4. Synchronize the project to update the resources in the local project with the latest changes saved on the server.

   If you do not already have a local version of the project, one is created in the Local Projects View.

   You may want to review the changes made on the server before updating them in the local project. For more information, see the related topic on merging changes in shared resources.
5. Once you have made your changes, in the Project Synchronization View, synchronize the project to save your changes on the server.
6. Unlock the resources.

Related Information

Opening the Project Synchronization View [page 74]
Locking a resource [page 75]
Synchronizing a project [page 74]
Merging changes to shared resources [page 77]
Unlocking a resource [page 76]

5.1.3.3 Renaming a shared project

Use this procedure to rename a shared project that exists in the repository.

1. Open the Project Synchronization View with a session on the repository system where the shared project is saved.
2. Select the shared project from the *Shared Project* list.

3. Click the *Rename Shared Project* icon .
4. Enter a new name that is unique in the repository.
5. To create a local project with the new name in the Local Projects View, synchronize the project.
   Local projects with the original name are no longer associated with the shared project with the new name. The resources in those local projects can no longer be synchronized with the newly named shared project.

Related Information

Opening the Project Synchronization View [page 74]
Synchronizing a project [page 74]

5.1.3.4 Deleting a shared project

1. Open the Project Synchronization View with a session on the repository system where the shared project is saved.
2. Select the shared project from the *Shared Project* list.
3. Click the *Delete Shared Project* icon .

> **i Note**
>
> Deleting a shared project cannot be undone.

The shared project is deleted from the repository. Resources in the local projects associated with the deleted shared project are not affected, however synchronization statuses of the local project are lost.

5.1.4 What is project synchronization?

Synchronizing a project starts with comparing the resources in a project in the Local Projects View with an associated shared project on the repository server. Synchronization detects added resources, deleted resources, and differences between the resources. Based on the differences detected, you can update the local and shared resources.

Use the Project Synchronization View to synchronize a project. The view displays synchronization information in two panes:

- The *Shared Project* pane lists the resources in the shared project on the server. A lock icon appears next to the resource if it is locked. Other information about the resources on the server is given: the user who last modified the resource and on what date, the user who locked the resource and on what date.
- The *Synchronization Status* pane lists the status of each resource. The status is determined by comparing the resources in the local and shared projects.

The different synchronization statuses and what they mean are listed in the table.

Status	Description
Added Locally	The resource was added in the local project, but not in the shared project.
Changed Locally	The resource was changed in the local project, but not in the shared project since the last synchronization.
Deleted Locally	The resource was deleted in the local project but still exists in the shared project.
Added on Server	The resource is not in the local project but exists in the shared project.
Changed on Server	The resource was changed in the shared project, but not in the local project since the last synchronization.
Deleted on Server	The resource exists in the local project, but was deleted in the shared project.
Conflicting	Any of the following situations creates a conflicting status: • The resource was changed in both the local and shared projects with different changes since the last synchronization. • A resource with the same name was added both in the local and shared projects since the last synchronization. • The resource was changed in the local project, but deleted from the shared project. • The resource was changed in the shared project, but deleted from the local project.
Synchronized	The resources are identical.

Three commands allow you to synchronize resources. When you select the resources to be synchronized, you can select individual resources or folders. The following table summarizes the possible synchronization actions.

Icon	Command	Synchronization Action
	Get Changes from Server	For the selected resources: If the status is *Added on Server*, the resource is added to the local project. If the status is *Changed on Server*, the resource is updated in the local project. If the status is *Deleted on Server*, the resource is deleted from the local project. If the status is *Conflicting*, the resource on the server (whether it is changed, added, or deleted) is copied to the local project, regardless of the change made in the local project. For all other statuses, no action is taken. **i Note** You may want to review the changes made on the server before updating them in the local project. For more information, see the related topic on merging changes in shared resources.

Icon	Command	Synchronization Action
	Save Changes on Server	For the selected resources: • If the status is *Added Locally*, the resource is added to the shared project on the server. • If the status is *Changed Locally*, the resource is updated in the shared project on the server. • If the status is *Deleted Locally*, the resource is deleted from the shared project on the server. • If the status is *Conflicting*, the resource in the local project (whether it is changed, added, or deleted) is copied to the shared project, regardless of the change made in the shared project. **i Note** If a resource is locked by another user, an error message displays and the changes and deletions are not made on the server. For all other statuses, no action is taken.
	Revert Changes	For the selected resources, the local project is updated with the shared project on the server, regardless of the status. **i Note** *Revert Changes* updates the local project in the same way as *Get Changes from Server* with the exception that if a resource has been created in the local project and has not yet been saved on the server, *Revert Changes* deletes the new local resource, whereas *Get Changes from Server* preserves the new local resource.

The list of resources in the *Synchronization Status* pane can be filtered on status using the icons in the pane's tool bar:

	Shows all resources. This clears the filters and lists all resources regardless of their status.
	Shows/Hides the resources with a status of *Synchronized*.
	Shows/Hides the resources in the local project that have changed with respect to the server.
	Shows/Hides the resources with a status of *Conflicting*.
	Shows/Hides the resources on the server that have changed with respect to the local project.

Related Information

Synchronizing a project [page 74]
Locking a resource [page 75]
Unlocking a resource [page 76]
Merging changes to shared resources [page 77]

5.1.4.1 Opening the Project Synchronization View

1. To open the Project Synchronization View, from the main menu, select *Window* > *Project Synchronization* >.
2. Click the *Change Session* icon in the Project Synchronization View to open a session on the repository system where the shared projects are saved.

After entering your authentication information, you can manage shared projects, and select a project to synchronize in the *Shared Project* list.

Related Information

Opening a session [page 80]
What is project synchronization? [page 71]

5.1.4.2 Synchronizing a project

To synchronize a project, the project must be shared.

Synchronize a project when you want do the following tasks:

- Update local resources with changes stored in the shared project.
- Save in the shared project the changes you made to local resources.
- Revert local resources to the copy stored in the shared project.
- Create a local copy of a shared project.

1. Open the Project Synchronization View with a session on the repository system where the shared project is saved.
2. Select the shared project from the *Shared Project* list.
3. To see the latest synchronization status of the resources in the project, in the pane labeled *Synchronization status (Local Project Compared to Shared Project)*, expand the project and click the refresh icon .

   For more information about synchronization status and possible actions, see the related topic about project synchronization.

4. Synchronize the project:
   - To update the local project with resources that have changed in the shared project, select the resources in the list and click the *Get Changes from Server* icon .
     If no project exists in the Local Projects View with the name of the shared project, a local project is created.

     > **i Note**
     >
     > You may want to review the changes made on the server before updating them in the local project. For more information, see the related topic on merging changes in shared resources.

   - To update the shared project with changes made locally, select the resources in the list and click the *Save Changes on Server* icon. .

     > **i Note**
     >
     > Resources that are locked by another user cannot be updated on the server. However, if needed, any user can unlock the resource.
     >
     > If you are updating the server with resources that you have locked, synchronizing updates the resources, but does not unlock them. You must explicitly unlock the resources on the server.

   - To revert resources in the local project to the copy stored on the server, select the resources and click the *Revert Changes* icon .

Related Information

What is project synchronization? [page 71]
Opening the Project Synchronization View [page 74]
Locking a resource [page 75]
Unlocking a resource [page 76]
Merging changes to shared resources [page 77]

5.1.4.3 Locking a resource

To lock a resource, the resource must be in a shared project.

Lock a resource when you want to inform other designers that you are working on the resource when they open the Project Synchronization View.

1. Open the Project Synchronization View with a session on the repository system where the shared project is saved.
2. Select the shared project from the *Shared Project* list.
3. In the *Shared Project* pane, expand the project.

4. Right-click the resource and select *Lock*.

> **i Note**
>
> The lock action does not update the resource contents in either the local or shared project. To save any changes, synchronize the resource.

Related Information

Opening the Project Synchronization View [page 74]
What is project synchronization? [page 71]

5.1.4.4 Unlocking a resource

Unlock a resource once you have updated your changes on the server and you want to inform other designers that you have finished. Once you have unlocked the resource, other designers can lock it and/or update the server version with changes.

> **i Note**
>
> If necessary, you can unlock a resource locked by another user.

1. Open the Project Synchronization View with a session on the repository system where the shared project is saved.
2. Select the shared project from the *Shared Project* list.
3. In the *Shared Project* pane, expand the project.
4. Right-click the resource and select *Unlock*.

> **i Note**
>
> The unlock action does not update the resource on the server with any changes made in the local project. To save any changes, synchronize the resource.

Related Information

Opening the Project Synchronization View [page 74]
What is project synchronization? [page 71]

5.1.4.5 Merging changes to shared resources

When synchronizing a shared resource, before getting changes from the server, you may want to review the changes and decide which changes to apply to the local resource. This procedure presents a way to manually merge changes between differing resources.

For example, you are working on a resource called **`NewDatafoundation`** in a local project. This project is shared in a repository. When you synchronize the project, **`NewDatafoundation`** has a synchronization status of *Changed on Server*, or *Conflicting*.

To review and manually merge the changes:

1. In the local project, right-click **`NewDatafoundation`** and select *Copy*.
2. Right-click again (in the local project) and select *Paste*.
   A copy of **`NewDatafoundation`** is saved in the local project.
3. In the Project Synchronization View, select **`NewDatafoundation`** and synchronize by selecting *Get Changes from Server*.
4. Open both **`NewDatafoundation`** and **`Copy of NewDatafoundation`** in the Data Foundation Editor by double-clicking each resource name in the local project.

   Each copy will open in a separate tab of the editor.
5. Compare the changes from the server in **`NewDatafoundation`** to your local changes in **`Copy of NewDatafoundation`**.
6. In the editor tab with **`NewDatafoundation`** open, delete any server changes you do not want to keep, and add any changes that you made locally that you want to keep.
7. Save the changes to **`NewDatafoundation`** in the editor.
8. In the Project Synchronization View, refresh the synchronization. Update the server by selecting **`NewDatafoundation`** and then selecting *Save Changes on Server*.

Eventually, once you have verified the merged changes, you can delete **`Copy of NewDatafoundation`** from the local project.

5.1.5 Saving resources as PDF, HTML, or Text files

You can save any resource in a local project as a report in a local file. The PDF, HTML, or Text files is intended for documentation purposes.

1. Right-click the resource name in the Local Projects View, and select *Save As*.
2. In the *Report Location* box, enter a file path, file name, and file type for the report. The file type can be .pdf, .html, or .txt.

   To browse the local file system to find a file path, click the browse button [...].
3. For larger resources (data foundations and business layers), you can select which metadata elements to include in the report in the *Metadata Elements* box.
4. Click *Finish* to generate the report.

5.2 Working with repository resources

Repository resources are the universes, sets, and connections that have been secured in a repository on a Central Management System (CMS). The Repository Resources View lets you navigate and interact with the folders and resources in repositories. You can retrieve repository resources to your local projects folder and work on the resources locally.

5.2.1 Managing repository resources

Repository resources are the universes and connections that have been secured in a repository on a Central Management System (CMS). The Repository Resources View lets you navigate and interact with the folders and resources in repositories.

The Connections folder contains the secured connections created using the information design tool and universe design tool.

> **i Note**
>
> A CommonConnections sub-folder sometimes appears in the Connections folder. The CommonConnections folder contains OLAP connections created in the Central Management Console for use in SAP BusinessObjects Advanced Analysis.

The Universe folder contains universes published with the information design tool (.unx universes), as well as universes created and exported with the universe design tool or migrated from earlier versions (.unv universes).

To navigate a repository, open a session on the CMS where the repository is stored. For more information about sessions, see the related topic.

The following sections summarize the tasks you can do from the Repository Resources View.

To copy a system name, right-click on the system name and click *Copy System Name*. You can paste the system name into another document.

To copy an object path, right-click on the object and click *Copy Object Path*. You can paste the object path into another document.

Folder management

With appropriate rights, you can insert, rename, and delete sub-folders in the Connections and Universes folders.

Secured Connection management

- Edit an existing connection.
- Insert a new secured relational or OLAP connection in the repository.
- Create a connection shortcut in a local project from an existing secured connection.
- Delete a secured connection from the repository.

Universe management

You can do the following tasks on .unx universes (published using the information design tool):

- Run a check integrity.
- Run a query. This command opens the Query Panel. The security settings defined in the security profiles for the universe are applied according to the username in the session.
- Retrieve a universe. This command saves the business layer and its referenced resources in a local project so that you can edit them.
- Rename a universe. This command renames only the universe, and not the underlying business layer.
- Delete a universe from the repository.

You can do the following tasks on .unv universes (created using the universe design tool or migrated from earlier versions):

- Convert a universe. You can save the converted resources in a local project, or publish the converted .unx universe in the repository.
- Delete a universe from the repository.

Related Information

Managing a session [page 80]
Working with the Connection Editor [page 87]
Relational connection [page 87]
OLAP connection [page 109]
Connection shortcuts [page 86]
Checking the integrity [page 398]
Running a query on a universe published in a repository [page 81]
Retrieving a published universe from a repository [page 305]
Benefits of converting .unv universes [page 25]

5.2.2 Managing a session

A session contains the Central Management Server (CMS) system name and authentication information needed to access resources stored in a repository. You need at least one session defined to connect to a repository. You can define additional sessions that connect to the same repository as a different user.

Workflows in the information design tool that require access to secured resources prompt you with the *Open Session* dialog box. If you have not already defined a session for the repository you want to access, you can select *New Session* from the *Sessions* list. You can also define a session with the *Insert Session* command in the Repository Resources View.

Once the session is defined, it is retained in the Repository Resources View, and also on the *Sessions* list. The next time you open the session, you will only need to enter the password.

Once a session is opened it stays open until you exit the information design tool. To explicitly close a session, you must do so from the Repository Resources View.

Several sessions can be open at the same time, as long as the sessions are on different CMS systems. If you need to open a session with a different username and password on a CMS that has another session open, you must first close the open session.

If you no longer need a session and want to delete it from the list, use the *Delete Session* command in the Repository Resources View.

Related Information

CMC rights for information design tool users [page 350]
Opening a session [page 80]
Closing a session [page 81]

5.2.2.1 Opening a session

Different workflows require you to open a session. If you are prompted to open a session, a list of predefined sessions is available. The *Sessions* list is organized in the following order:

- Open sessions in alphabetical order
- Closed sessions in alphabetical order
- *New Session*

To open a session on a repository already defined in the information design tool:

1. Do one of the following:
   - In the Repository Resources View, right-click the repository name and select *Open Session*.
   - Select the session in the *Sessions* list.
2. The authentication information for the CMS is filled in for you. If the session is not already open, enter your *Password*.
3. Depending on the workflow, click *OK*, *Next*, or *Connect*.

To open a session on a repository not already defined in the information design tool:

1. Do one of the following:
   - In the Repository Resources View, from the *Insert* menu, select *Insert Session*.
   - Select *New Session* from the *Sessions* list.
2. In the *System* box, enter the name of the Central Management System where the repository is located.

   > **i Note**
   >
   > To insert a session for a repository that is hosted on a machine in a different domain than the client hosting the application, you need to provide the host information in a hosts file on the client. Update the hosts file is in the following location:
   >
   > `C:\WINDOWS\system32\drivers\etc\hosts`

3. Enter your *User Name* and *Password*.

   > **i Note**
   >
   > For *Authentication* type *Windows AD*, specify the complete domain name in the *User Name*. For example, enter **`myuser@domain.com`** instead of **`myuser@domain`**.

4. In the *Authentication* list, select the authentication method to use.
5. In the *Description* field, type a description for your session. When you hover your mouse over the session, this description appears in a pop-up pane.
6. Depending on the workflow, click *OK*, *Next*, or *Connect*.

The session remains open until you explicitly close it in the Repository Resources View, or exit the information design tool.

Related Information

Closing a session [page 81]

5.2.2.2 Closing a session

All open sessions close when you exit the information design tool. To explicitly close a session:

1. In the Repository Resources View, select the session you want to close.
2. Right-click and select *Close Session*.

5.2.3 Running a query on a universe published in a repository

When you run a query on a universe published in a repository, the Query Panel applies the settings defined in the security profiles for the universe according to the user name defined in the session.

1. In the Repository Resources View, select the universe. Select .unx universes only.
2. Right-click the universe name and select *Run Query*.

The Query Panel opens with a list of views and objects granted for your user name.

> **i Note**
>
> When you run a query on a translated universe, the language that the objects are displayed in depends on your connection:
>
> - If you run a query connected to the CMS (based on a shortcut connection), the localization is set on the server side (SAP BusinesObjects Web Intelligence) settings.
> - If you run a query using a local connection, the localization uses the information design tool preferences.

Related Information

Building a query [page 307]

5.2.4 Locking universes

To prevent other users working on a universe that you are modifying, you can lock the universe. To lock the universe, right-click on the universe in the *Repository Resources* view and select *Lock Universe*. When a universe is locked, a padlock symbol is displayed next to the universe name in the *Repository Resources* view. The universe can only be unlocked by the person who locked it, or the Administrator. To unlock a universe, right-click it and select *Unlock Universe*.

6 Accessing databases through connections

This chapter describes how to create and edit connections for working with different databases.

Related Information

What are connections? [page 83]
Working with the Connection Editor [page 87]
CMS database connection [page 539]
Relational connection [page 87]
OLAP connection [page 109]
Connection shortcuts [page 121]
Local and secured connections [page 120]
Editing connection shortcuts [page 122]
Showing values in a relational connection [page 108]
Showing values in an OLAP connection [page 120]

6.1 What are connections?

A connection is a named set of parameters that define how one or more SAP BusinessObjects applications can access relational or OLAP data sources. The connection can be a local file, or a remote object in a repository that is referenced by a local shortcut in the information design tool.

> **i Note**
>
> From version 4.3, the information design tool only supports 64-bit drivers and operating systems.

You use connections for the following purposes:

Purpose	Description
Relational data source for a data foundation	You associate one or more relational connections to a data foundation, and build a business layer on the data foundation. When you publish the business layer as a universe, the connections and data foundation are integrated in the universe and provide the data for queries run against the universe.

Purpose	Description
OLAP data source for a business layer	For an OLAP data source, you associate a business layer directly to a connection. The business layer is published as a universe, but the connection provides direct access to the cube.
Direct access to an SAP BW BEx Query	You define SAP BW connections that use the *SAP BICS Client* middleware driver to provide access to a BEx Query. SAP BusinessObjects query and reporting applications connect directly to the BEx Query. You cannot use these connections as a source for business layers or universes. For information on how to build a universe on SAP BW, see the related topic on using SAP BW data sources.
Direct access to an SAP HANA information model	You define SAP HANA connections that use the *SAP HANA Client* middleware driver to provide direct access to a single information model (such as an Analytic View or Calculation View). SAP BusinessObjects query and reporting applications connect directly to the cube representing the information model. You cannot use these connections as a source for business layers or universes. For information on how to build a universe on SAP HANA, see the related topic on using SAP HANA data sources.

Connections can be either local or secured.

Related Information

Local connections [page 84]
Secured connections [page 85]
SAP BW [page 40]
SAP HANA [page 46]
Relational connection [page 87]
OLAP connection [page 109]
Working with the Connection Editor [page 87]

6.1.1 Local connections

You create local connections in the information design tool local project. Local connections are saved as independent objects on the local file system as .cnx files.

Local connections are used for the following purposes:

- To access relational data sources when authoring a data foundation and relational business layer.

  > **i Note**
  >
  > To create a multisource-enabled data foundation, you must reference secured connections.

- To access an OLAP cube when authoring an OLAP business layer.
- To run queries on a target database to test modifications in the business layer, or to build lists of values.

Local connections have limited or no security as they can be used by any user with access to the machine running the information design tool.

To secure a local connection, you publish the connection to a repository.

Related Information

Publishing a local connection to the repository [page 301]
Secured connections [page 85]
Relational connection [page 87]
OLAP connection [page 109]
Local and secured connections [page 120]

6.1.2 Secured connections

A secured connection is a connection that has been created in, or published to, a repository. It is stored in a dedicated Connections folder in the repository. You can create subfolders in the Connections folder to organize the storage of connections in the repository.

When a connection is published, a connection object containing the same parameters as the local connection is created in the Connections folder or subfolder in the repository.

You can also create secured connections directly in the repository using the *Insert Relational Connection* and *Insert OLAP Connection* commands from the Repository Resources View.

Secured connections cannot be copied to the local file system, but are made available in the Local Projects View as connection shortcuts. The shortcut can be used in the same way as a local connection. However, the connection properties can only be modified by connecting to the repository system.

Secured connections and connection shortcuts are used for the following purposes:

- To retrieve data for universes published to a repository.
- To retrieve data for SAP BusinessObjects reporting products directly accessing database middleware.
- As a data source when creating a data foundation or OLAP business layer.

A secured connection is subject to the following general security constraints in the repository:

- Users must be authenticated.
- User rights can be defined at the user level to grant or deny access to connections or connection properties.
- Connections can only be shared and used by authenticated users.

Downloading relational connections locally

To maintain confidentiality, some sensitive secured connection parameters, for example username and password, remain stored in the repository.

In order to edit the connection in the information design tool, the *Download connection locally* right must be granted in the Central Management Console (in addition to the *Create, modify, or delete connections* application right and the *Edit objects* connection right).

If the *Download connection locally* right is granted, you can choose to run queries on the server using the server middleware driver, or locally using the local middleware driver. To use the local middleware, select the local middleware option in the information design tool preferences. If this right is denied, the information design tool uses the server middleware.

Related Information

Connection shortcuts [page 86]
Publishing a local connection to the repository [page 301]
CMC rights for information design tool users [page 350]
Middleware driver for secured relational connections [page 416]
Local and secured connections [page 120]

6.1.3 Connection shortcuts

A connection shortcut is an object that references a secured connection in a repository. The shortcut is saved as a .cns file on the local file system. The shortcut contains the repository address and port number, the type of connection (OLAP or relational), and an ID that identifies the connection on the server.

You use a connection shortcut when authoring, or modifying any data foundation or business layer that uses a connection stored in the repository.

You can create a connection shortcut in two ways:

- Publish a local connection to the repository.
- Create a shortcut from an existing secured connection in the Repository Resources View.

Related Information

Publishing a local connection to the repository [page 301]
Connection shortcuts [page 121]
Editing connection shortcuts [page 122]

6.2 Working with the Connection Editor

Use the Connection Editor to do the following tasks. For more information, see the related topics.

- Edit connection properties and parameters, and change the middleware driver.
- Edit the properties of connection shortcuts, and change the referenced connection.
- Browse the values in the tables referenced by a relational connection.
- Browse the objects in an OLAP cube and run an MDX query on the cube.

Related Information

Local and secured connections [page 120]
Editing connection shortcuts [page 122]
Showing values in a relational connection [page 108]
Showing values in an OLAP connection [page 120]
Middleware driver for secured relational connections [page 416]

6.3 Creating and editing connections

6.3.1 Relational connection

Use the New Relational Connection wizard to create local and secured connections to a relational data source.

Make sure the middleware driver is configured for the data source to which you want to create a connection. For more information about middleware configuration, see the *Data Access Guide*. For information about supported data sources, see the SAP Business Objects BI Platform 4.3 Supported Platforms (PAM) at https://www.sap.com.

Before you create a local connection, you must have a project available in the Local Projects View. For more information on creating local projects, see the related topic.

> **i Note**
>
> You must create relational connections to SAP BW and SAS sources directly in the repository as secured connections.

1. Do one of the following:
   - To create a local connection, select the project folder in the Local Projects View. Select *File* > *New* > *Relational Connection*.

- To create a secured connection, in the Repository Resources View, open a session on the repository where you want to create the secured connection. Right-click the Connections folder or subfolder in the repository, and select *Insert Relational Connection*.

2. Follow the steps in the New Relational Connection wizard to enter the following information:
   - Name for the connection
   - Middleware for the target database
   - Login parameters to connect to the relational data source
   - Configuration and custom parameters to optimize the connection

   If you need help on a particular step, click the help icon in the wizard dialog box.

Related Information

Creating a local project [page 65]
Managing repository resources [page 78]
Name a connection [page 88]
What are connections? [page 83]
Middleware driver for secured relational connections [page 416]

6.3.1.1 Name a connection

This section describes the Resource name page of the New Relational Connection wizard.

You name a connection and can enter a description of the data source. The name and description are available as properties of the connection and can be edited at any time.

Properties	Description
Resource name	Connection name. This field is mandatory.
Description	Information describing data source. This information can be useful when the connection is used for multiple data foundations. This is optional information.

When you have entered name information, click *Next* to continue the wizard.

Related Information

Select a middleware driver [page 89]

6.3.1.2 Select a middleware driver

You select a connection driver to connect to the correct middleware version for the target database. The connection driver is the SAP BusinessObjects driver that maps information in the middleware to the user interface of the SAP BusinessObjects application.

Expand the database and middleware node for the target database and select the connection driver. Click *Next* to continue the wizard.

> **i Note**
>
> SAP BW and SAS connections are only listed if you are creating the connection directly in the repository.

> **i Note**
>
> If you are using the information design tool from a Crystal Server 2011 installation, SAP middleware drivers are not available.

Related Information

Set the connection parameters [page 89]
Middleware driver for secured relational connections [page 416]

6.3.1.3 Set the connection parameters

The connection parameters vary depending on the type of data source for which you are defining the connection. Select from the related topics the link to more information about the connection parameters.

Related Information

Login parameters for relational connections [page 90]
Login parameters for SAP BW and ERP connections [page 91]
Login parameters for SAS connections [page 95]
Login parameters for Oracle EBS connections [page 97]
Login and schema parameters for CSV file connections [page 104]
Parameters for OData connections [page 97]
Parameters for XML and Web Services connections [page 99]
Configuration parameters for relational connections [page 101]
Custom parameters for relational connections [page 104]

6.3.1.3.1 Login parameters for relational connections

The following login parameters apply to most relational connections.

Parameter	Description
Authentication Mode	The method used to authenticate the user's login credentials when accessing the data source: • *Use specified user name and password*: Uses the *User Name* and *Password* parameters defined for the connection. • *Use BusinessObjects credential mapping*: Uses the database credentials associated with the user's account defined on the Central Management Server (CMS) to connect to the data source. The database credentials are set in the User's Properties in the Central Management Console. For more information, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*. • *Use Single Sign On*: This authentication mode is used to support end-to-end single sign on defined in the Central Management Server (CMS). If you use an external authentication source (for example LDAP), the CMS and the data source must be configured to use this external authentication source. For more information on single sign on, see the *SAP BusinessObjects Business Intelligence platform Administrator Guide*. • *Use single sign-on when refreshing reports at view time*: If you have opened a session using Windows Active Directory single sign-on (SSO), and if some connections have been defined to use SSO, choose this mode of authentication when connecting to the CMS repository. For more information, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.
User Name	The user name to access the data source if *Authentication Mode* is *Use specified user name and password*.
Password	The password to access the data source if *Authentication Mode* is *Use specified user name and password*.
Use SSL	This parameter applies only to SAP HANA connections. If selected, uses SSL protocol to connect to the server.
Single Server	This parameter applies only to SAP HANA connections. Check this option if you are connecting to only one SAP HANA database server. • *Host Name*: the name of the server hosting the data source. Do not include the port number. • *Instance Number*: the SAP HANA instance number, which represents the second and third digits of the port number. Must be set between **00** and **99**. For example, if the port number is `30215`, the instance number is **02**.

Parameter	Description
Multiple Servers	This parameter applies only to SAP HANA connections. Check this option to take advantage of the SAP HANA failover mechanism. *Server (host:port{;host:port})*: a list of servers separated by a semi-colon, for example: **`(host1:30015;host2:30015;host3:30015)`**. The JDBC driver will choose one of these hosts for connecting. If a host is not available, the driver chooses the next host from the list. You can also enter the host and port of only one server in the field.
Server (<host>:<port>)	The name and port of the server hosting the data source. For connections to Oracle, you can enter a list of servers separated by commas: (<host>:<port>, <host>:<port>).
Server	The name of the server hosting the data source.
Database	The database name.
Data Source Name	For ODBC connections, the name of the data source that you have defined using your operating system's data source manager.
Alias	For DB2 connections, this is the alias of the database that you have created in the DB2 configuration assistant.
Net Service	For Oracle connections using JDBC middleware, the Oracle Net Service name.
Service	For Oracle connections, the alias that contains the server IP and Net Service information.
JDBC_URL *JDBC_CLASS*	For generic JDBC connections, the JDBC URL and class used to connect to the database.
Informix Server	For Informix connections, the name of the Informix server that you have defined.
OLE DB Provider Name	For generic OLE DB providers, the provider name.

6.3.1.3.2 Login parameters for SAP BW and ERP connections

The following parameters apply to connections to SAP BW (relational and BICS Client connections), and SAP ERP.

> **i Note**
>
> When you create a connection, you declare the catalog name. The catalog name is used in the data foundation and business layer references to the tables. The references in the information design tool are managed (updated) dynamically, so if you change the catalog name, the table references remain valid in

> the information design tool. However, for a published document used by SAP BusinessObjects Web Intelligence, when the catalog name has been changed, this information is not updated in SAP BusinessObjects Web Intelligence, and a refresh of an existing report will be broken. To get the report to work again, edit the references to the connection in the data foundation in the information design tool and republish so that the new catalog name is taken into account in the report.

To set ABAP function and InfoSet parameters for SAP ERP connections, after entering the login parameters, click *Next*.

Parameter	Description
Authentication Mode	The method used to authenticate the user's login credentials when accessing the data source: • *Use specified user name and password*: Uses the *User Name* and *Password* parameters defined for the connection. • *Use BusinessObjects credential mapping*: Uses the database credentials associated with the user's account defined on the Central Management Server (CMS) to connect to the data source. The database credentials are set in the User's Properties in the Central Management Console. For more information, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*. • *Use Single Sign On*: This authentication mode is used to support end-to-end single sign on defined in the Central Management Server (CMS). If you use an external authentication source (for example LDAP), the CMS and the data source must be configured to use this external authentication source. For more information on single sign on, see the *SAP BusinessObjects Business Intelligence platform Administrator Guide*
Client Number	The number used to identify the client on the SAP system.
User Name	The user name to access the data source if *Authentication Mode* is *Use specified user name and password*.
Password	The password to access the data source if *Authentication Mode* is *Use specified user name and password*.
Language	The 2-character ISO language code for the language to be used for the connection to the data source. For example, **EN** for English. **i Note** In some cases, select the language from the list.
Save Language	Specifies which language to use for the connection: • If you select the *Save Language* option, the value from the *Language* parameter is used. • If you clear *Save Language*, the value from the user's session (Preferred Viewing Locale) is used.

Parameter	Description
System ID	The 3-character SAP system ID. **i Note** Required for both application and message server types. **i Note** For a successful connection to the message server, you need to add the message server system ID to the following file on the machine hosting the application: C:\WINDOWS\system32\drivers\etc\services At the end of the existing file, add the line: `sapmsXXX <tab> 3601/tcp` where `sapms` means SAP message server, `xxx` is the is the System ID of the server that is used, and `3601/tcp` is the default TCP port used for communication.
Server type	• Select *Application Server* to connect directly to the SAP server without using load balancing. • Select *Message Server* to benefit from SAP load balancing capabilities.
Server Name for *Application Server*	The name of the SAP application server.
System Number for *Application Server*	The system number of the SAP application server. This is a two-digit integer between 00 and 99.
Server Name for *Message Server*	The name or IP address of the SAP message server used for load balancing.
Group Name for *Message Server*	Name of the Logon group; a set of dedicated application servers used to log on.

The following parameters apply only to SAP BW connections:

Parameter	Description
Use Custom ID Program Mapping	An optional parameter for SAP BW relational connections only. The *Program ID Mapping* defines the program IDs for the callback that SAP BW uses to contact the data federation server. Enter *Program ID Mapping* as a list of one or more server name=program ID pairs separated by the semi-colon character (;). For example: `<MySIA.DF_Server1>`=RFC1;`<MySIA.DF_Server2>`=RFC2 Each program ID must match the name of an RFC destination created on SAP BW. If this parameter is not defined, the data federation server automatically creates an RFC destination. For more detailed information, see the description of the `programIDMapping` connector property in the Data Federation Administration Tool Guide.
Use Custom Gateway	An optional parameter for SAP BW relational connections only. In *Gateway Host Name*, enter the name of the server hosting the SAP BW gateway. In *Gateway Service Name*, enter the name or port number of the SAP BW gateway service. If this option is not selected, SAP BW provides the gateway host name and service name via an RCF.
InfoProvider	For SAP BW relational connections, the name of the InfoCube or MultiProvider to be used as the fact table in the center of the snowflake schema in the data foundation.
Catalog	For SAP BW relational connections, the name used to identify the connection to the query server. **i Note** A default catalog name is registered automatically with the query server the first time the connection is added to any multisource-enabled data foundation.

6.3.1.3.2.1 InfoProvider fact table selection

When you create an SAP BW connection, the *Select InfoProvider Fact Table* dialog box lets you choose a fact table that will become the center of the snowflake schema in your data foundation.

The *filter* button lets you filter by types of InfoProvider.

6.3.1.3.2.2 ABAP function and InfoSet parameters for ERP connections

The following parameters apply to SAP ERP connections. For more information about SAP ERP connections, see the *Data Access Guide*.

Parameter	Description
Function Name Wildcard	Wildcards are filters that reduce the number of tables exposed in the connection. The wildcard character is *, and it matches 0 to any number of characters. The wildcard character can be used with keywords. For example: `*keyword_one*keyword_two*` The above wildcard will expose only tables containing keyword_one followed by keyword_two.
Map Table Parameters into Input Columns	If selected, the table parameters are considered as both input and output parameters of the ABAP function. If unselected, the table parameters are considered as output only.
Map Selection Fields into Table Columns	If selected, any selection field in the SAP Query is mapped into a table column and considered as an optional input column: • A query can contain only the EQUAL filter in this column • If the column is only in the projection, NULL is returned If unselected, the selection fields are ignored. No filtering is possible on these fields.

6.3.1.3.3 Login parameters for SAS connections

The following parameters apply to connections to SAS data sources.

To include access to multiple data sets that are not pre-defined to the SAS/SHARE server, after entering the login parameters, click *Next*.

Parameter	Description
Authentication Mode	The method used to authenticate the user's login credentials when accessing the data source: • *Use specified user name and password*: Uses the *User Name* and *Password* parameters defined for the connection. • *Use BusinessObjects credential mapping*: Uses the database credentials associated with the user's account defined on the Central Management Server (CMS) to connect to the data source. The database credentials are set in the User's Properties in the Central Management Console. For more information, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*. • *Use Single Sign On*: This authentication mode is used to support end-to-end single sign on defined in the Central Management Server (CMS). If you use an external authentication source (for example LDAP), the CMS and the data source must be configured to use this external authentication source. For more information on single sign on, see the *SAP BusinessObjects Business Intelligence platform Administrator Guide*.
User Name	The user name to access the data source if *Authentication Mode* is *Use specified user name and password*.
Password	The password to access the data source if *Authentication Mode* is *Use specified user name and password*.
Host Name	The host name of the server where the SAS/SHARE is running.
Port	The port to which to connect.
Catalog	The name used to identify the connection to the query server. **i Note** A default catalog name is registered automatically with the query server the first time the connection is added to any multisource-enabled data foundation.

6.3.1.3.3.1 Setting SAS data sets

When you create an SAS connection, the *Set SAS Data Sets* dialog box lets you configure the data foundation to access multiple data sets that are not pre-defined to the SAS/SHARE server. These are data sets that are not included in the current SAS configuration.

1. Select the *Use data sets that are non pre-defined to the SAS/SHARE server* option.
2. Click *Add* and, in the *Location* field, enter the path to the data set in the format required for the operating system that you are using.
3. In the *Library Name* field, enter a name to use to refer to the data set.
4. Click *Add* to add other data sets if needed.
5. Click *Finish*.

6.3.1.3.4 Login parameters for Oracle EBS connections

The following login parameters apply to Oracle EBS connections.

Parameter	Description
User Name	The user name to access the Oracle database server.
Password	The password to access the Oracle database server.
Service	The Oracle service name.
Authentication Mode	The method used to authenticate the user's login credentials when accessing the EBS application: • *Use specified user name and password*: Uses the *Oracle EBS User* and *Oracle EBS Password* parameters defined for the connection. • *Use Single Sign On*: Uses the credentials associated with the user's account defined on the Central Management Server (CMS) when users log into the SAP BusinessObjects BI platform using Oracle EBS user name and password. For more information, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.
Oracle EBS User	The user name to access the application if *Authentication Mode* is *Use specified user name and password*.
Oracle EBS Password	The password to access the application if *Authentication Mode* is *Use specified user name and password*.
Language	The application language.
Application	The application name.
Security Group	The Oracle security group.

6.3.1.3.5 Parameters for OData connections

The following parameters apply to connections to OData data sources.

Login parameters for OData connections

Parameter	Description
Service Root URI	The URI string of the OData service. For example: `http://services.odata.org/OData/OData.svc`

Parameter	Description
Authentication Mode	The method used to authenticate the user's login credentials when accessing the data source: • *Use specified user name and password*: Uses the *User Name* and *Password* parameters defined for the connection.
User Name	The optional user name for HTTP authentication.
Password	The optional password for HTTP authentication.
Proxy Address	The path to the HTTP proxy server (`<host:port>`). For example: `myproxy.com:8080`
Proxy User Name	The user name used to access the proxy server.
Proxy Password	The password used to access the proxy server.

Extended parameters for OData connections

Parameter	Description
Custom Authentication Parameters	Custom parameters used for authentication. They are attached to the URI but not traced to avoid exposing secured information. For example: `apikey=1234&authinfo=1234`
Column Selection *Supported Filter Conditions* *Sorting*	If selected, the Odata service provider executes the corresponding operation of the SQL query. If unselected, the OData driver executes the operation. **i Note** SAP recommends you do not use the data access driver to execute these operations, because it may affect the connection performance. Use it only if the service provider does not support or partially supports the operation.

Configuration parameters for Odata connections

Parameter	Description
Connection Pool Mode	If using a connection pool, the method to use to keep the connection active.

Parameter	Description
Pool Timeout	If the *Connection Pool Mode* is set to *Keep the connection active for*, the length of time in minutes to keep the connection open.
Connection Timeout	**! Restriction** Specific to HTTP connections to OData and Web Service data sources. The time in seconds a connection remains active in case of no response from the data source. Default value is `10`. The connection remains active indefinitely if *Connection Timeout* is set to **0**.
Cache Metamodel	If selected, caches the metamodel in the connection so that the model does not need to be parsed and recreated for each connection call. **i Note** SAP recommends you do not use the data access driver to execute this operation, because it may affect the connection performance.

6.3.1.3.6 Parameters for XML and Web Services connections

The following parameters apply to connections to XML data sources and web services.

Parameter	Description
Location Type *Protocol*	If *Location Type* is set to *Local*, the protocol and login credential parameters are grayed. If *Location Type* is set to *Remote*, select a *Protocol* to make the appropriate login parameters available. **i Note** Schema files can be local even if *Location Type* is set to *Remote*. If the schema file is remote, the *Protocol* applies to both the data source and schema files.

Parameter	Description

Filepath or Pattern	The path to a single XML file or the path to a folder containing multiple XML files. Files can be local or remote (HTTP, FTP, and SMB). If remote, the datasource is the location URL. Paths with Microsoft Windows or UNIX styles are valid. Wildcards can be used. Blank characters must be replaced with `%20`.
	
	For example:
	• `C:\report.xml` for a single file
	• `C:\XMLFiles\` or `C:\XMLFiles\*.xml` for multiple files
	• `/home/user/xmlfiles/report.xml` for a single file located on a UNIX machine
	• Remote locations:
	  ○ `http://host:port/path/file`
	  ○ `ftp://host:port/path/file`
	  ○ `smb://server:port/path/file`

Choose the XML schema	If set to *Explicitly indicate the XML Schema (XSD)*, the data access driver uses the XML schema that you enter in *Schema File*. If set to *The XML schema is included in the XML files*, the data access driver uses the XML schema included in the XML files.
Schema File	The path to the XML schema. Required if *Choose the XML schema* is set to *Explicitly indicate the XML schema (XSD)*.
User Name	The user name to access the XML files in remote connection.
Password	The password to access the XML files in remote connection.
SMB Domain	The domain used for SMB connections.
Proxy Address	The path to the HTTP proxy server (`<host:port>`). For example: `myproxy.com:8080`
Proxy User Name	The user name used to access the proxy server.
Proxy Password	The password used to access the proxy server.
Merge Files into One	This Boolean value indicates if tables are automatically concatenated when a pattern is given as data source. For example, if the data source pattern is `report_*.xml`, then the driver concatenates all the tables from XML files that match the pattern. ⚠ Caution XML files must have the same structure.

Parameter	Description
Web Service URL	The path to the web service over HTTP or HTTPS.

6.3.1.3.7 Configuration parameters for relational connections

The *Configuration Parameters* dialog box contains parameters that you can set to override default configuration options.

The following configuration parameters apply to most relational connections.

Parameter	Description
Connection Pool Mode	If using a connection pool, the method to use to keep the connection active.
Pool Timeout	If the *Connection Pool Mode* is set to *Keep the connection active for*, the length of time in minutes to keep the connection open.
Connection Timeout	**! Restriction** Specific to HTTP connections to OData and Web Service data sources. The time in seconds a connection remains active in case of no response from the data source. Default value is `10`. The connection remains active indefinitely if *Connection Timeout* is set to **0**.
Array Fetch Size	The maximum number of rows authorized with each fetch from the database. For example, if you enter 20, and your query returns 100 rows, the connection retrieves the data in 5 fetches of 20 rows each. To deactivate array fetch, enter an *Array Fetch Size* of 1. The Data is retrieved row by row. **i Note** Deactivating array fetch size can increase the efficiency of retrieving your data, but slows server performance. The greater the value in the *Array Fetch Size*, the faster your rows are retrieved. You must, however, ensure you have adequate client system memory.
Array Bind Size	This parameter is not used for universes created using the information design tool.
Login Timeout	The number of seconds before a connection attempt times out and an error message is displayed.

Parameter	Description
Maximum Parallel Queries	Indicates the maximum number of queries that can be executed for this connection concurrently with other connections when a document with multiple data providors is refreshed. For this connection you can set a value between 1 and 64. This applies to the capability of a `Web Intelligence`document to perform parallel data refresh jobs in reports based on multiple data providers. `Web Intelligence` cannot run in parallel more than the maximal value set on this connection when refreshing data. For example, if you set the value to 1 then `Web Intelligence` can execute one query on this connection in parallel with other queries using other connections. If you set the value to 64, then `Web Intelligence` can execute 64 queries on this connection concurrently with other connections. If you set the value to 0, then the refresh on this connection will be performed in serial, one query at a time. **i Note** For a Web Intelligence Server instance, the administrator can set the maximal number of parallel data refresh jobs per document from the Central Management Console. If the setting for a connection is less than the number of parallel queries authorized for the document, then the connection setting limits the number of parallel queries possible over all of the connections accessed by the Refresh job. This parameter is applicable per document, which means that all paralleled queries might be over the limit set here.
Query Timeout	**! Restriction** Specific to connections to SAP HANA and Oracle data sources using JDBC middleware. The number of seconds before a query running on the database times out and is forced to end.
Max Rows	**! Restriction** Specific to connections to SAP HANA and Oracle data sources using JDBC middleware. Maximum number of rows of data that are displayed when the query runs.

Parameter	Description
Add File(s)	**! Restriction** Specific to connections to Apache Hadoop HIVE data sources. The paths to external resources to be added to the Hadoop Distributed Cache of the cluster. Typically, resources can be Python transform script files that you make available at query execution time. This parameter corresponds to the `add FILE` HIVE command line. You can define the paths to more than one file, separated by semi-colons. For example: **`/tmp/foo.py;/tmp/bar.py`** Only UNIX style paths are valid.
Add Jar(s)	**! Restriction** Specific to connections to Apache Hadoop HIVE data sources. The paths to external JAR files to be added to the Java classpath. This parameter corresponds to the `add JAR` HIVE command line. You can define the paths to more than one JAR file, separated by semi-colons. For example: **`/usr/lib/hive/hive-contrib-1.jar;/usr/lib/hive/hive-contrib-2.jar`** Only UNIX style paths are valid.
JDBC Driver Properties (key=value,key=value)	Values for JDBC driver properties. You can define the value of more than one property, separated by commas. For example, the following value for *JDBC Driver Properties* sets the **`oracle.jdbc.defaultNChar`** and **`defaultNChar`** driver properties: **`oracle.jdbc.defaultNChar=true,defaultNChar=true`** **i Note** If a property is defined in the <driver>.sbo file, the value defined in this parameter is used. For more information about SBO files, see the *Data Access Guide*.
Owner Name	For DB2 connections, this parameter adds the name of the owner of the table as a prefix on the table name, to match the DB2 convention for naming tables.
Table Suffix	For DB2 connections, this parameter adds a suffix on the table name, to match the DB2 convention for naming tables.

6.3.1.3.8 Custom parameters for relational connections

The Custom Parameters dialog box lets you override the value of certain parameters. You can also add parameters and their values.

Parameter	Description
ConnectInit	The value is added to the SQL and is run once when a user connects to the database.
Hint	For Oracle connections, the value is used by the Oracle query optimizer to choose an execution plan. Consult your Oracle documentation for full information on the Hints that can be used, and how they can be used to optimize queries.

6.3.1.3.9 Login and schema parameters for CSV file connections

The following parameters apply to connections to Comma-Separated Value (CSV) files. See the related topic for more information about file format and regional settings for CSV file connections.

Parameter	Description
Location Type *Protocol*	The *Location Type* and *Protocol* parameters guide you through setting the necessary parameters for your data source. If *Location Type* is set to *Local*, the protocol and login credential parameters are grayed. If *Location Type* is set to *Remote*, select a *Protocol* to make the appropriate login parameters available. **i Note** Schema files can be local even if *Location Type* is set to *Remote*. If the schema file is remote, the *Protocol* applies to both the data source and schema files.

Parameter	Description

Datasource	*Filepath or Pattern*
	
	The path to a single CSV file or the path to a folder containing multiple CSV files. Files can be local or remote (HTTP, FTP, and SMB). If remote, the datasource is the location URL. Paths with MS Windows or UNIX styles are valid. Wildcards can be used. Blank characters must be replaced with `%20`.
	
	For example:
	• `C:\report.csv` for a single file
	• `C:\CSVFiles\` or `C:\CSVFiles\*.csv` for multiple files
	• `/home/user/csvfiles/report.csv` for a single file located on a UNIX machine
	• Remote locations:
	  ○ `http://host:port/path/file`
	  ○ `ftp://host:port/path/file`
	  ○ `smb://server:port/path/file`

Schema Detection	The method used to detect the schema of the CSV file. Possible values are: • *auto* The data access driver finds the schema automatically. *File Type* must be set to *delimited*. • *no detection* The data access driver skips comment lines, analyzes the first line, and determines the number of columns, but not the column types. *File Type* must be set to *delimited*. • *ddl* The data access driver uses a Data Definition Language (DDL) file to detect the schema. • *sqlddl* The data access driver uses a DDL file corresponding to the Standard SQL to detect the schema.
Schema File	The path to one DDL or SQLDDL schema file. If you want to define schemas for several tables, use an SQLDDL file. Required if *Schema detection* is set to *ddl* or *sqlddl*. If the schema file is remote, the protocol of the file must match the *Protocol* parameter setting.

Parameter	Description
Probe Rows	The method to parse lines for checking column information (name, type, size, and nullability). Required if *Schema Detection* is set to *auto*. Possible values are: • *auto* The driver parses the file until a type is detected for each column. If there is no NULL value in the first row, the parsing terminates after the first row. ⚠ Caution This method may lead to type conversion conflicts in the case where only the first rows are parsed and other types are used in the next rows. • *int* Parses a specific number of rows. This setting can be used as a trade-off between scalability for big CSV files and a low data quality of CSV files. ⚠ Caution This method may lead to conflicts if other types are used in the rows that are not parsed. • *all* Parses the whole file. This method allows you to find the longest string value, which corresponds to the column size for non-numeric values. ℹ Note The recommended method is *all* to allow for correct detection of column information. Since this is the slowest detection method, use DDL files if CSV files are large.
Number of Probe Rows	The number of CSV file lines that are parsed to check the column types. Required if *Probe Rows* is set to *int*.
Credentials	The *User Name* and *Password* to access the CSV files in a remote connection. If *Protocol* is set to *SMB (Windows Share)*, enter the *SMB Domain* for the connection.
HTTP Proxy	If *Protocol* is set to *HTTP*, enter the proxy parameters for the connection. *Proxy Address*: The path to the HTTP or FTP proxy server (`<host:port>`). For example: `myproxy.com:8080` *Proxy User Name*: The user name used to access the proxy server. *Proxy Password*: The password used to access the proxy server.

Related Information

File format and regional settings for CSV file connections [page 107]

6.3.1.3.10 File format and regional settings for CSV file connections

The following parameters apply to connections to Comma-Separated Value (CSV) files.

File Settings

Parameter	Description
File Charset	The character set used in the CSV files. **ℹ Note** All files must have the same character set.
File Type	The file type can be one of the following: • *delimited* The CSV file entries are delimited by a separator. • *fixed* The CSV file entries have fixed width.
Lenient Mode	If selected, invalid rows are skipped automatically (not enough or too many columns).
Column Names in First Line	This Boolean value tells if the first row of the CSV file contains column names. **ℹ Note** If the file does not contain any column names and if schema detection is enabled, then the data access driver names columns as `col1`, `col2`, ...,`col<n>`.
Merge Files into One	This Boolean value indicates if tables are automatically concatenated when a pattern is given as data source. For example, if the data source pattern is `report_*.csv`, then the driver concatenates all the tables from CSV files that match the pattern. **⚠ Caution** CSV files must have the same structure.
Number of Comment Lines in the Beginning	The number of rows that contain comments at the beginning of the CSV file. Maximum is `1000`.

Parameter	Description
Separator	A character that is used to separate CSV file entries. It must be different from text qualifier and escape character. **i Note** If the tab key is used to separate entries, the word *TAB* can be set as separator.
Text Qualifier	A character that surrounds a file entry, for example quote (`'`) or double quote (`"`). If you do not want to use a text qualifier, use a character that is not used in the CSV file to prevent the data access driver from using the default value.
Escape Character	A character that allows the text qualifier to be treated as literal text. **i Note** Text qualifier and escape character must be different.

Regional Settings

Parameter	Description
Decimal Separator	Default value is period (`.`). For example: 100.20.
Thousands Grouping Character	Default value is comma (`,`). For example: 1,000.20.
Date Format	Date and time formats used in the CSV files. They must match the format in the CSV files to enable the driver to recognize the date and time formats and to parse them. Default values are: • `yyyy-MM-dd` for dates • `yyyy-MM-dd HH:mm:ss` for timestamps • `HH:mm:ss` for times
Timestamp Format	
Time Format	

6.3.1.4 Showing values in a relational connection

1. Open the connection in the editor:

Option	Description
To open a local connection	Double-click the connection name in the Local Projects View.

Option	Description
To open a secured connection	In the Repository Resources View, open a session on the repository where the connection is published. In the Connections folder or subfolder, double-click the connection name.

2. Click the *Show Values* tab.
3. In the *Catalog* pane, double-click the table name (to show all columns), or double-click a column name.

   To see what you can do in the pane where values are displayed, see the related topic.

Related Information

Showing values in a data source [page 162]

6.3.2 OLAP connection

You use the New OLAP Connection wizard to create local and secured connections to an OLAP data source.

You can also connect to HANA through HANA HTTP Info Access (InA) services with BI client tools including the information design tool.

Before you create a local connection in the information design tool, you must have a project available in the Local Projects View. For more information on creating local projects, see the related topic.

> **i Note**
>
> The OLAP connections you create in the information design tool are not supported by the universe design tool. Also, OLAP connections created in the universe design tool are not available for building universes in the information design tool.

> **i Note**
>
> The information desin tool supports SNC encryption for BOE systems based on SAP BW OLAP connections. The SNC configuration with SAP BusinessObjects Enterprise on SAP BW is described in SAP Note 2183696.

1. Do one of the following:
   - To create a local connection, select the project folder in the Local Projects View. Select *File* > *New* > *OLAP Connection*.
   - To create a secured connection, in the Repository Resources View, open a session on the repository where you want to create the secured connection. Right-click the Connections folder or subfolder in the repository, and select *Insert OLAP Connection*.
2. Follow the steps in the New OLAP Connection wizard to enter the following information:
   - Name for the connection

- Middleware driver for the target database
- Authentication parameters to connect to the OLAP data source
- OLAP cube to connect to

If you need help on a particular step, click the help icon in the wizard dialog box.

3. The following option must be present in the `InformationDesignTool.ini` INI file: `Djco.client.snc_lib=${BOE_INSTALL_DIR}\sapcrypto.dll` where the `jco.client.snc_lib` option indicates the path for the SAP Crypto library in the client machine and `${BOE_INSTALL_DIR}` indicates the path where the BOE binaries are installed: `<DRIVE>:\Program Files (x86)\SAP BusinessObjects\SAP BusinessObjects Enterprise XI 4.0\win64_x64`

Related Information

Creating a local project [page 65]
Select an OLAP middleware driver [page 110]
What are connections? [page 83]

6.3.2.1 Select an OLAP middleware driver

This section describes the OLAP driver selection page of the New OLAP Connection wizard.

You select an OLAP driver to connect to the OLAP server. The OLAP driver maps information from the OLAP server middleware to the user interface of the SAP BusinessObjects application.

Depending on your target OLAP server, expand the middleware node, and select the target driver.

> **i Note**
>
> If you are using the information design tool from a Crystal Server 2011 installation, SAP middleware drivers are not available.

6.3.2.2 Set login parameters for OLAP data sources

The connection parameters vary depending on the type of data source for which you are defining the connection. Select from the related topics the link to more information about the connection parameters.

Related Information

Login parameters for OLAP connections [page 111]
Login parameters for SAP BW and ERP connections [page 91]

6.3.2.2.1 Login parameters for OLAP connections

The following parameters apply to most OLAP connections.

For a description of login parameters for SAP BW (BICS Client), see the related topic.

Login parameter	Description
Authentication mode	The method used to authenticate the user's login credentials when accessing the data source: • *Use specified user name and password*: Uses the *User Name* and *Password* parameters defined for the connection. • *Use BusinessObjects credential mapping*: Uses the database credentials associated with the user's account defined on the Central Management Server (CMS) to connect to the data source. The database credentials are set in the User's Properties in the Central Management Console. For more information, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*. • *Use Single Sign On*: This authentication mode is used to support end-to-end single sign on defined in the Central Management Server (CMS). If you use an external authentication source (for example LDAP), the CMS and the data source must be configured to use this external authentication source. For more information on single sign on, see the *SAP BusinessObjects Business Intelligence platform Administrator Guide*.
Data Source	This parameter applies only to SAP HANA connections. It is made up of the name of the server hosting the data base and the port number, as follows: `<Hostname>:<Port number>`. On a multi-tenant system, the port number specifies the tenant. On a platform hosting multiple servers, you can declare additional data sources as follows: `<Hostname1>:<Portnumber1>;<Hostname2>:<Portnumber2>`.
Server	For MSAS connections, the URL path, for example: **`http://<server_name>/olap_2005/msmdpump.dll`** For Essbase connections, the server name for the data source.
User Name	The user name to use to access the OLAP server if the *Authentication mode* is *Use specified user name and password*.
Password	The password to use to access the OLAP server if the *Authentication mode* is *Use specified user name and password*.
Language	The language that will be used for the connection.
Auto Reconnect	This parameter applies only to SAP HANA connections. If selected, the application automatically reconnects to the host server if the connection fails.

Login parameter	Description
Use SSL	This parameter applies only to SAP HANA connections. If selected, uses SSL protocol to connect to the host server.
Fetch Size	This parameter applies only to SAP HANA connections. The maximum number of rows authorized with each fetch from the database. The recommended *Fetch Size* for OLAP connections to SAP HANA is 7000.
Array Fetch Size	The maximum number of rows authorized with each fetch from the database. For example, if you enter 20, and your query returns 100 rows, the connection retrieves the data in 5 fetches of 20 rows each. To deactivate array fetch, enter an *Array Fetch Size* of 1. The Data is retrieved row by row. **i Note** Deactivating array fetch size can increase the efficiency of retrieving your data, but slows server performance. The greater the value in the *Array Fetch Size*, the faster your rows are retrieved. You must, however, ensure you have adequate client system memory.
Query Timeout	**! Restriction** Specific to connections to Oracle data sources using JDBC middleware. The number of seconds before a query running on the database times out and is forced to end.
Maximum Parallel Queries	Indicates the maximum number of queries that `Web Intelligence` can execute concurrently for a connection. You can set a value between 1 and 64. For example, if you set the value to 1 then `Web Intelligence` can execute one query in parallel with one other query on the same connection. If you set the value to 64, then `Web Intelligence` can execute 64 queries in parallel on the same connection. **i Note** You can also run these queries in parallel with other connections.

Related Information

Login parameters for SAP BW and ERP connections [page 91]

6.3.2.2.2 Login parameters for SAP BW and ERP connections

The following parameters apply to connections to SAP BW (relational and BICS Client connections), and SAP ERP.

> **i Note**
>
> When you create a connection, you declare the catalog name. The catalog name is used in the data foundation and business layer references to the tables. The references in the information design tool are managed (updated) dynamically, so if you change the catalog name, the table references remain valid in the information design tool. However, for a published document used by SAP BusinessObjects Web Intelligence, when the catalog name has been changed, this information is not updated in SAP BusinessObjects Web Intelligence, and a refresh of an existing report will be broken. To get the report to work again, edit the references to the connection in the data foundation in the information design tool and republish so that the new catalog name is taken into account in the report.

To set ABAP function and InfoSet parameters for SAP ERP connections, after entering the login parameters, click *Next*.

Parameter	Description
Authentication Mode	The method used to authenticate the user's login credentials when accessing the data source: • *Use specified user name and password*: Uses the *User Name* and *Password* parameters defined for the connection. • *Use BusinessObjects credential mapping*: Uses the database credentials associated with the user's account defined on the Central Management Server (CMS) to connect to the data source. The database credentials are set in the User's Properties in the Central Management Console. For more information, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*. • *Use Single Sign On*: This authentication mode is used to support end-to-end single sign on defined in the Central Management Server (CMS). If you use an external authentication source (for example LDAP), the CMS and the data source must be configured to use this external authentication source. For more information on single sign on, see the *SAP BusinessObjects Business Intelligence platform Administrator Guide*
Client Number	The number used to identify the client on the SAP system.
User Name	The user name to access the data source if *Authentication Mode* is *Use specified user name and password*.
Password	The password to access the data source if *Authentication Mode* is *Use specified user name and password*.

Parameter	Description
Language	The 2-character ISO language code for the language to be used for the connection to the data source. For example, **EN** for English. **i Note** In some cases, select the language from the list.
Save Language	Specifies which language to use for the connection: • If you select the *Save Language* option, the value from the *Language* parameter is used. • If you clear *Save Language*, the value from the user's session (Preferred Viewing Locale) is used.
System ID	The 3-character SAP system ID. **i Note** Required for both application and message server types. **i Note** For a successful connection to the message server, you need to add the message server system ID to the following file on the machine hosting the application: C:\WINDOWS\system32\drivers\etc\services At the end of the existing file, add the line: **sapmsXXX <tab> 3601/tcp** where **sapms** means SAP message server, **xxx** is the is the System ID of the server that is used, and **3601/tcp** is the default TCP port used for communication.
Server type	• Select *Application Server* to connect directly to the SAP server without using load balancing. • Select *Message Server* to benefit from SAP load balancing capabilities.
Server Name for *Application Server*	The name of the SAP application server.
System Number for *Application Server*	The system number of the SAP application server. This is a two-digit integer between 00 and 99.
Server Name for *Message Server*	The name or IP address of the SAP message server used for load balancing.
Group Name for *Message Server*	Name of the Logon group; a set of dedicated application servers used to log on.

The following parameters apply only to SAP BW connections:

Parameter	Description
Use Custom ID Program Mapping	An optional parameter for SAP BW relational connections only. The *Program ID Mapping* defines the program IDs for the callback that SAP BW uses to contact the data federation server. Enter *Program ID Mapping* as a list of one or more server name=program ID pairs separated by the semi-colon character (;). For example: `<MySIA.DF_Server1>`=RFC1;`<MySIA.DF_Server2>`=RFC2 Each program ID must match the name of an RFC destination created on SAP BW. If this parameter is not defined, the data federation server automatically creates an RFC destination. For more detailed information, see the description of the `programIDMapping` connector property in the Data Federation Administration Tool Guide.
Use Custom Gateway	An optional parameter for SAP BW relational connections only. In *Gateway Host Name*, enter the name of the server hosting the SAP BW gateway. In *Gateway Service Name*, enter the name or port number of the SAP BW gateway service. If this option is not selected, SAP BW provides the gateway host name and service name via an RCF.
InfoProvider	For SAP BW relational connections, the name of the InfoCube or MultiProvider to be used as the fact table in the center of the snowflake schema in the data foundation.
Catalog	For SAP BW relational connections, the name used to identify the connection to the query server. **i Note** A default catalog name is registered automatically with the query server the first time the connection is added to any multisource-enabled data foundation.

6.3.2.3 Configuring SQL access on SAP HANA HTTP

You can configure SQL access on HANA HTTP connections to directly connect a remote HANA database instance when you refresh a document. This increase the performance at runtime on this type of connection.

When executing queries on HANA, Web Intelligence and the semantic layer use the best HANA access (between HTTP and SQL access) according to the query specifications.

To do this, you must configure SQL access on HANA HTTP connections to directly connect a remote HANA database instance at refresh time. The SQL access option does not impact the HANA variable support level on HANA HTTP connections, and the query panel capabilities are unchanged.

The SQL access option does not impact the list of values retrieval when querying on HANA HTTP connections: in the query panel, the lists of values for query filters are provided by the HTTP access on HANA, in the prompt

are of the query panel. The lists of values for query filter prompts and for HANA variables/Input Parameters are provided by the HTTP access on HANA.

> **i Note**
>
> This connection configuration does not work for UNX universes on HANA HTTP connections. It only works for the direct access to HANA from SAP BusinessObjects Web Intelligence.

To use this feature:

- You must have access to HANA HTTP.
- SQL access to your remote HANA database is granted for users who use the HANA HTTP connections in Web Intelligence and the semantic layer.
- If your SAP HANA XS server is configured for HTTPS/SSL with a signed certificate, then the SQL access to the remote HANA database should be also configured for SSL connections (see HANA documentation on how to configure SSL on a HANA server).
- If your HANA http connection is using SAML SSO connection, then the SQL access to the remote HANA database should be also configured for SSO connections.

Related Information

Connection parameters for SAP HANA HTTP [page 116]
Remote database configuration parameters [page 117]

6.3.2.3.1 Connection parameters for SAP HANA HTTP

You are creating an HTTP connection to SAP HANA in order to enable SQL access.

1. In the Edit OLAP connection wizard pane, enter the following information:

Login parameter	Description
Authentication Mode	The method used to authenticate the user's login credentials when accessing the data source: *Use specified user name and password*: Uses the *User Name* and *Password* parameters defined for the connection.
User Name	The user name to use to access the OLAP server if the *Authentication mode* is *Use specified user name and password*.
Password	The password to use to access the OLAP server if the *Authentication Mode* is *Use specified user name and password*.
DataSource Reference	

Login parameter	Description
Language	The language that will be used for the connection.
Data Source	This parameter applies only to SAP HANA connections. Provide the URL of the SAP HANA HTTP data source.
Allow SQL access to the remote database	The HANA JDBC driver can be used to connect to the remote HANA database. By default, this option is not set.
Remote database connections	Define the format type to use to set the database server definition. There are two options: ○ *Connect to port or failover*. Default setting. ○ *Connect to tenant*.
Server (host:port[;host:port])	This property is used when you select the *Connect to port or failover* option. The remote HANA database server. Its syntax is `<server>:<port> [<server2>:<port2>]`
Database Name	Specify the database name of the remote HANA server instance.
Maximum Parallel Queries	Indicates the maximum number of queries that `Web Intelligence` can execute concurrently for a connection. You can set a value between 1 and 64. For example, if you set the value to 1 then `Web Intelligence` can execute one query in parallel with one other query on the same connection. If you set the value to 64, then `Web Intelligence` can execute 64 queries in parallel on the same connection. **i Note** You can also run these queries in parallel with other connections.

2. Test the connection.

6.3.2.3.2 Remote database configuration parameters

When you are configuring SQL access on an SAP HANA HTTP connection, you must define the configuration parameters for the remote database.

1. Define the following parameters:

Database parameter	
Connection Pool Mode	If using a connection pool, the method to use to keep the connection active.
Pool Timeout	If the *Connection Pool Mode* is set to *Keep the connection active for*, this is the length of time in minutes to keep the connection open.
Array Fetch Size	The maximum number of rows authorized with each fetch from the database. For example, if you enter 20, and your query returns 100 rows, the connection retrieves the data in 5 fetches of 20 rows each. To deactivate array fetch, enter an *Array Fetch Size* of 1. The Data is retrieved row by row. **i Note** Deactivating array fetch size can increase the efficiency of retrieving your data, but slows server performance. The greater the value in the *Array Fetch Size*, the faster your rows are retrieved. You must, however, ensure you have adequate client system memory.
Array Bind Size	
Login Timeout	The number of seconds of inactivity before the connection times out and the user mustlogin again.
Query Timeout	**! Restriction** Specific to connections to Oracle data sources using JDBC middleware. The number of seconds before a query running on the database times out and is forced to end.
Max Rows	The maximum number of rows that will be returned for a query.
JDBC Driver Properties	Values for JDBC driver properties. You can define the value of more than one property, separated by commas. For example, the following value for JDBC Driver Properties sets the oracle.jdbc.defaultNChar and defaultNChar driver properties: `oracle.jdbc.defaultNChar=true,defaultNChar=true`

Database parameter	
	i Note If a property is defined in the `<driver>.sbo` file, the value defined in this parameter is used. For more information about SBO files, see the *Data Access Guide*.

2. Click *Finish*.

6.3.2.4 Select an OLAP cube

The following options apply to associating a cube with the OLAP connection.

> **i Note**
> For connections using the *SAP BICS Client*, select the *Specify a cube in the connection* option. Open a catalog, the select the BEx Query for the connection. You can seach BW InfoProviders from an OLAP BICS connection on SAP BW

Option	Description
Do not specify a cube in the connection	Select this option to create the connection without specifying a cube. In this case, each time you access the connection, either when building a business layer or in a query and reporting tool, you will be prompted to select a cube.
Specify a cube in the connection	Select this option to always associate a cube with the connection. The cube selection page lists the cubes available for the target database. You can enter a search string in the search text box. Select the cube in the list.

6.3.2.5 Configuring SNC connection for SAP BW connections

The information design tool supports SNC encryption for BOE systems based on SAP BW OLAP connections. The SNC configuration with SAP BusinessObjects Enterprise on SAP BW is described in SAP Note 2183696.

1. Do one of the following:
   - To create a local connection, select the project folder in the *Local Projects View*. Select *File* > *NewOLAP Connection*.
   - To create a secured connection, in the *Repository Resources View*, open a session on the repository where you want to create the secured connection. Right-click the *Connections* folder or subfolder in the repository, and select *Insert OLAP Connection*.

Follow the steps in the New OLAP Connection wizard to enter the following information:

2. Enter the name for the connection.
3. Select the middleware driver for the target database.
4. Select the authentication parameters to connect to the OLAP data source.
5. Select the OLAP cube to connect to.
6. Ensure the following option is present in the InformationDesignTool.ini INI file:

   `Djco.client.snc_lib=${BOE_INSTALL_DIR}\sapcrypto.dll` where the `jco.client.snc_lib` option indicates the path for the SAP Crypto library in the client machine and `${BOE_INSTALL_DIR}` indicates the path where the BOE binaries are installed: `<DRIVE>:\Program Files (x86)\SAP BusinessObjects\SAP BusinessObjects Enterprise XI 4.0\win64_x64`.

6.3.2.6 Showing values in an OLAP connection

1. Open the connection in the editor:

Option	Description
To open a local connection	Double-click the connection name in the Local Projects View.
To open a secured connection	In the Repository Resources View, open a session on the repository where the connection is published. In the Connections folder or subfolder, double-click the connection name.

2. To browse the objects in the cube and their properties, click the *Browse Metadata* tab.

   Select an object to display its properties in the properties pane.
3. To run an MDX query on the cube, select the *Query* tab.

   > **i Note**
   >
   > MDX querying is not available for OLAP connections used for direct access (such as direct access to a BEx Query or SAP HANA information model).

4. Build an MDX query in the *MDX Query* pane by dragging and dropping objects from the *OLAP Metadata* pane and entering MDX statements.
5. To validate the MDX, click *Parse*.
6. To run the query, click *Run*.

6.3.3 Local and secured connections

1. To open the connection in the editor, do one of the following:

Option	Description
To open a local connection	Double-click the connection name in the Local Projects View.
To open a secured connection	In the Repository Resources View, open a session on the repository where the connection is published. In the Connections folder or subfolder, double-click the connection name.

2. To edit the connection name or description, click the *General Information* tab.
3. To edit the connection parameters, click *Edit*.

   For local connections, you can also right-click the connection name in the Local Projects View and select *Edit Connection*.
4. To change the middleware driver for relational connections, select *Change Driver*. Select a new driver and enter the new connection parameters.
5. To test the availability of the database server, click *Test Connection*.

   You can also right-click the connection or shortcut name in the Local Projects View and select *Test Connection*.
6. Save the connection information by clicking the Save icon in the main tool bar.

Related Information

Showing values in a relational connection [page 108]
Showing values in an OLAP connection [page 120]
What are connections? [page 83]
Middleware driver for secured relational connections [page 416]

6.3.4 Connection shortcuts

When you publish a connection, you have the option of creating a connection shortcut in the Local Projects View. You can also create a shortcut for an existing published connection. Use the following procedure to create a connection shortcut for an existing secured connection.

You must have a local project in the *Local Projects* view.

1. In the *Repository Resources* view, open a session on the repository where the secured connection is stored.
2. In the *Connections* folder or subfolder, right-click the connection name.
   - For OLAP connections, select *Create OLAP Connection Shortcut*.
   - For relational connections, select *Create Relational Connection Shortcut*
3. In the *Select a Local Project* dialog box, select the project in which you want to create the shortcut.

The connection shortcut is created in your selected local project.

Related Information

Creating a local project [page 65]
Connection shortcuts [page 86]

6.3.5 Editing connection shortcuts

You can edit the name and description of a connection shortcut. You can also change the shortcut to reference a different connection in the same repository where the existing connection is published.

1. Open the connection shortcut in the editor by double-clicking the shortcut name in the *Local Projects* view.
2. You can enter or change text in *Shortcut Name* and *Description*.
3. To change the connection that the shortcut references, click *Change Connection*.

   You can also right-click the shortcut name in the *Local Projects* view and select *Change Connection*.
4. To test the referenced connection, click *Test Connection*.

   You can also right-click the shortcut name in the *Local Projects* view and select *Test Connection*.
5. Save the shortcut by clicking the *Save* icon in the main tool bar.

Related Information

Connection shortcuts [page 86]

7 Designing a data foundation

In order to create a universe based on a relational database, you must use the *Data Foundation Editor* to define and maintain the data foundation structure and its connections, and to access the data federation layer, if necessary.

You then build one or more business layers based on the data foundation. The connection, data foundation, and business layer make up the universe.

Related Information

Getting started with data foundations [page 123]
The Data Foundation Editor [page 133]
Building a data foundation [page 126]

7.1 Getting started with data foundations

A data foundation contains a schema of relevant tables and joins from one or more relational databases that are used as a basis for one or more business layers. Single-source and multisource-enabled are two types of data foundations that allow you to take advantage of different data foundation features.

You reference relational connections in the data foundation. You insert tables and joins from the databases referenced in the connections.

Using the *Data Foundation Editor*, you can enhance the data foundation by adding federated tables (designed in the federation layer), derived tables, alias tables, calculated columns, additional joins, contexts, prompts, and lists of values. The availability of some features depends on the data foundation type. See the related topic for more information about data foundation types.

You can build any number of business layers on the same data foundation. In this case the data foundation becomes the basis for multiple universes.

Related Information

Single-source data foundations [page 124]
Multisource-enabled data foundations [page 124]
What is the federation layer? [page 196]

7.1.1 Single-source data foundations

Single-source data foundations support a single connection. The connection can be local or secured, which means you can publish universes based on the data foundation either locally or to a repository.

Single-source data foundations support database-specific SQL syntax for derived tables, calculated columns, and join expressions. Database-specific SQL syntax allows functions or operators that are offered by a specific database and not by standard SQL-92 (for example, Oracle analytical functions). You must select single-source if you want to publish to a local folder the universes that are based on this data foundation.

Single-source data foundations are recommended for the following situations:

- You want to work exclusively with database-specific SQL syntax.
- You want to publish the universe locally and work outside of a repository.

Related Information

Multisource-enabled data foundations [page 124]
Building a data foundation [page 126]
Changing a connection in a data foundation [page 145]

7.1.2 Multisource-enabled data foundations

Multisource-enabled data foundations support one or more connections. You can add connections when you create the data foundation and anytime later. Multisource-enabled data foundations only support secured connections, and universes based on this type of data foundation can only be published to a repository.

Multisource-enabled data foundations support most relational connections supported in single-source data foundations. In addition, multisource-enabled data foundations support the following relational connections that are not supported in single-source data foundations:

- SAP BW connections
- SAS connections

Connections for multisource-enabled data foundations are managed by the data federation service. For information about tuning the data federation service, see the *Data Federation Administration Tool Guide*.

The federation layer is available in multisource-enabled data foundations. It allows you to create federated tables that you can then include in the data foundation.

SQL-92 standard syntax is the default for calculated columns, derived tables, and join expressions. In addition, the SAP BusinessObjects SQL database functions are available. You can use database-specific SQL syntax in a multisource-enabled data foundation by defining a database-specific derived table or calculated column. Database-specific SQL syntax allows functions or operators that are offered by a specific database and not by standard SQL-92 (for example, Oracle analytical functions).

> **i Note**
>
> For database functions, the SAP BusinessObjects syntax can be different from the syntax of the same function provided by database-specific SQL.

Multisource-enabled data foundations are required in the following situations:

- You want to insert tables and joins from more than one relational data source, or create federated tables.
- You want to use SAP BW or SAS connections.
- You want to use SQL-92 standard syntax and SAP BusinessObjects SQL functions.

For more information on these situations, see the related topics.

Related Information

Data foundations with multiple connections [page 125]
What is the federation layer? [page 196]
SQL expressions in multisource-enabled data foundations [page 126]
Single-source data foundations [page 124]
Building a data foundation [page 126]
SAP BW [page 40]

7.1.2.1 Data foundations with multiple connections

To be able to add multiple connections to a data foundation, you must select the multisource-enabled type when you create the data foundation.

You can select multiple connections when you create the data foundation. You can also add connections to an existing multisource-enabled data foundation. Connections must be secured, and therefore available in a repository. The connections are represented by a connection shortcut in the local project.

The connections in a multisource-enabled data foundation have the following additional properties:

- A short name used to identify the connection in the data foundation and to modify the table name in SQL expressions. You specify the short name when adding the connection. This name must be unique within the data foundation and is limited to forty characters. If you change the short name for the connection, the SQL expressions are automatically updated with the new name.
- A color for the connection. This color is used in the table header in data foundation views. You select the color when adding the connection. You can change the color for a connection at any time.
- A catalog used to identify the connection to the query server. A default catalog name is registered automatically with the query server the first time the connection is added to any multisource-enabled data foundation.
- For SAP BW connections, properties related to the automatic insertion of tables and joins. For more information on these properties, see the related topic.

In a multisource-enabled data foundation, the table name as it appears in SQL expressions has the format:

`@catalog`(short name)."database _qualifier.database_owner"."table_name"

A multi-source join can be created between tables from different connections. You can use the *Detect Joins* command to detect joins between tables referenced in different connections, or explicitly define them with the *Insert Join* command.

Related Information

SAP BW [page 40]
Connections [page 142]
Changing a connection in a data foundation [page 145]

7.1.2.2 SQL expressions in multisource-enabled data foundations

SQL expressions that define joins, calculated columns, and derived tables in a multisource-enabled data foundation use SQL-92 ANSI standard syntax.

In SQL-92 expressions, you can include SAP BusinessObjects database functions. The SQL syntax can be different from the syntax of the same function provided by database-specific SQL. For more information, see the related topic.

In SQL-92 expressions, you can include @functions. Which @functions you can include depends on the type of expression. For more information, see the related topic.

In order to use functions or operators that are offered by the database and not by SQL-92 (for example, Oracle analytical functions), you define database-specific calculated columns and derived tables. An option in the SQL Expression Editor allows you to use database-specific SQL.

Database-specific calculated columns and derived tables support the SQL syntax of the associated connection. The following rules apply to database-specific SQL expressions:

- You can reference only standard tables and database-specific derived tables in a single connection.
- You cannot reference tables in SAS or SAP BW connections.
- You can include @functions with certain restrictions. For more information, see the related topic.

Related Information

SAP BusinessObjects SQL function reference for multisource-enabled universes [page 423]
@Functions [page 489]

7.1.3 Building a data foundation

Before you begin:

- You need a local project in which to create the data foundation.
- In the local project, you need the relational connection or connection shortcuts to secured relational connections. Multisource-enabled data foundations require connection shortcuts.

For links to more detailed information on each step, see the Related Topics.

1. To start the *New Data Foundation* wizard, do one of the following:
   - Right-click a relational connection or connection shortcut in the Local Projects View and select *New* > *Data Foundation*.
   - Right-click the project folder in the Local Projects View and select *New* > *Data Foundation*.

   The data foundation is created in a .dfx file in the local project. It opens automatically in the *Data Foundation Editor*.
2. If you want use federated tables in your data foundation (multisource-enabled data foundations only), create the federated tables in the *Federation Layer*.
3. In the *Data Foundation Editor*, insert tables into the data foundation:

Option	Command
To insert tables from the connection	In the *Connection* pane, open and browse the tables in the connection. You have tools to search for and filter the tables in the *Connection* pane. Drag and drop the tables you want into the data foundation Master view.
To insert tables using a wizard	In the data foundation view tool bar, select *Insert* > *Insert Tables*. You have the option to detect and insert keys, joins, cardinalities, and row counts automatically. **i Note** To detect joins between tables referenced by different connections, you need to use the *Detect Joins* command.
To insert federated tables (multisource-enabled data foundations only)	In the data foundation view tool bar, select *Insert* > *Federated Table*.

4. Insert joins:

Option	Command
To insert joins manually	In the data foundation view, click the column name in the first table and drag it to the column in the second table. A join path appears between the two tables. You can also insert joins by opening the *Edit Join* dialog box. In the data foundation view tool bar, select *Insert* > *Insert Join*.

Option	Command
To detect joins	In the data foundation view tool bar, select *Detect* > *Detect Joins*.

To edit a join, double-click the join path. For more information about editing and detecting joins, see the related topics.

5. Check the cardinality of the joins in the data foundation. In the data foundation view tool bar, select *Detect* > *Detect Cardinalities*.

   In the *Detect Cardinalities* dialog box, you can set or detect cardinality for any or all joins.
6. You can enhance the function of the data foundation in several ways, for example:
   - Insert calculated columns
   - Insert derived tables
   - Insert alias tables
   - Insert parameters with optional prompts
   - Insert lists of values to be associated with a prompt
   - Set SQL options and SQL generation parameters in the data foundation properties
7. Verify the join paths and resolve any loops. Use the commands in the *Aliases and Contexts* pane to detect aliases and contexts automatically.
8. Run an integrity check to validate the tables, columns, and joins in the data foundation. Right-click the data foundation name in the *Data Foundation* pane, and select *Check Integrity*.
9. Save the data foundation by clicking the *Save* icon in the main tool bar.

Some commands to help you maintain the data foundation are listed below.

- If you change tables or columns, use *Show Local Dependencies* to find the business layers and objects that might be impacted by the changes.
- Refresh the structure to update the data foundation with changes that have occurred to the databases referenced in the connections.
- You can change a connection, or for multisource-enabled data foundations, add or remove a connection.
- You can enhance the ease of maintenance by creating custom views, grouping tables using families, arranging the table display, and inserting comments.

Related Information

Creating a local project [page 65]
Relational connection [page 87]
Connection shortcuts [page 121]
The Data Foundation Editor [page 133]
Inserting tables into the data foundation [page 151]
Searching for tables in the Connection pane [page 146]
Filtering the tables in the connection by table type [page 147]
What is the federation layer? [page 196]
Inserting and editing a join [page 176]

Detecting and setting cardinalities [page 180]
Inserting a calculated column [page 159]
Derived tables [page 168]
Alias tables [page 165]
Using parameters, derived tables, and lists of values in the data foundation [page 184]
Data foundation properties [page 129]
Resolving loops [page 187]
Checking the integrity [page 398]
Showing local dependencies in the data foundation [page 174]
Refreshing a data foundation [page 171]
Changing a connection in a data foundation [page 145]
Adding connections to a data foundation [page 144]
Managing data foundation views [page 139]
Grouping tables using families [page 137]
Auto-arranging tables in the data foundation view [page 136]
Inserting a comment into the data foundation view [page 138]

7.1.4 Data foundation properties

The following properties apply to the entire data foundation:

Property	Description
Description	Describes the data foundation. The description can be entered when you create the data foundation in the New Data Foundation wizard, and edited at any time in the data foundation properties.
Allow Cartesian products	When selected, if the SQL expression that defines an object in the data foundation could result in a Cartesian product, the SQL is allowed. **i Note** A Cartesian product is a result set which contains all the possible combinations of each row in each table included in a query. A Cartesian product is almost always an incorrect result.
Multiple SQL statements for each context	When selected, allows the user to select the query path when the query involves contexts. This option should be selected if the data foundation contains contexts.
SQL Parameters	Specifies custom values for SQL generation parameters that override the default values.
Comments	Contains comments about the data foundation.

Property	Description
Summary	Displays a summary of the number of each type of object defined in the data foundation.

Related Information

Editing SQL options in the data foundation [page 130]
Setting SQL generation parameters in the data foundation [page 130]
Showing a data foundation summary [page 131]
Contexts [page 181]

7.1.4.1 Editing SQL options in the data foundation

1. Open the data foundation in the editor by double-clicking the data foundation in the Local Projects View.
2. Make sure the top level of the data foundation is selected in the tree view in the *Data Foundation* pane.
3. Click the *SQL Options* tab in the properties pane.
4. Select or unselect options as required. For a description of the options, see the related topic.
5. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Data foundation properties [page 129]

7.1.4.2 Setting SQL generation parameters in the data foundation

The custom values for SQL generation parameters in the data foundation override the default values.

1. Open the data foundation in the editor by double-clicking the data foundation in the Local Projects View.
2. Make sure the top level of the data foundation is selected in the tree view in the *Data Foundation* pane.
3. Make sure the *Properties* tab is selected in the properties pane.
4. Click the *Parameters* button.
5. In the *Query Script Parameters* dialog box, edit the parameters:

   The currently defined SQL generation parameters are listed. Non-default parameters and parameters with non-default values are in bold type.

Option	Command
To change the value of an existing parameter	Click the *Value* column and select or enter the new value.
To add a predefined parameter	Click the arrow in the list box next to the *Add* button to display the list of predefined parameters. Select the parameter from the list and click *Add*.
To add a custom parameter	Make sure no predefined parameter is listed in the box next to the *Add* button, then click *Add*. A parameter with a default name is added to the table. To edit the parameter name, click the *Name* column. Click the *Value* column to enter a value.

To see a description of all the predefined SQL generation parameters and their values, click the help button.

6. To return to the default list of parameters and the default values, click *Default Values*. This removes any added parameters from the list and sets all values to the default.

Related Information

SQL generation parameters [page 500]

7.1.4.3 Showing a data foundation summary

Use this command to show the number of each type of object defined in the data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation in the Local Projects View.
2. Make sure the top level of the data foundation is selected in the tree view in the *Data Foundation* pane.
3. Make sure the *Properties* tab is selected in the properties pane.
4. Click the *Summary* button.

The data foundation summary displays in a new dialog box.

7.1.4.4 Editing the data foundation description and comments

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Make sure the top level of the data foundation is selected in the tree view in the *Data Foundation* pane.
3. To enter or edit a description, click the *Properties* tab in the properties pane.
4. To enter or edit comments, click the *Comments* tab in the properties pane.

   The comments apply to the entire data foundation. You can also enter comments in the data foundation display. For more information, see the related link.

5. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Data foundation properties [page 129]
Inserting a comment into the data foundation view [page 138]

7.1.4.5 Showing SAP HANA variable information

Variables and input parameters in SAP HANA information models are automatically associated with the corresponding tables in the data foundation. You can see variable information in the properties pane. In the variable information tab, you can also set some options as described in the table at the end of this task.

1. Open the data foundation in the editor by double-clicking the data foundation in the Local Projects View.
2. Make sure the *Data Foundation* pane is selected.
3. Select the table in the data foundation that corresponds to the information model for which you want to see the variables.
4. Select the *Variables* tab in the properties pane.

The properties for each variable are displayed in rows. You can edit the following properties:

Editable Properties

Property	Description
Description	Type in the description field to update the description of a variable.
Default Value	Type in the Default Value field to edit set a new default value.
Prompt to User	Click the ellipsis to open the Prompt dialog and edit the prompt that the user will see.
Keep last values	You can choose to keep the last values entered by the user for this variable.

Related Information

SAP HANA [page 46]

7.2 The Data Foundation Editor

This topic describes how to navigate the *Data Foundation Editor*. For steps to help you build the structure of your data foundation, see Building a data foundation [page 126].

The *Data Foundation Editor* is divided into a data foundation view pane, a properties pane, and browsing panes.

The data foundation view is a graphical representation of the tables and joins. The *Master* view contains all tables and joins and cannot be deleted. You can define custom views that contain subsets of the tables. Access the views by the tabs at the bottom of the view pane. For more information about custom views, see the related topic.

You can create a data foundation based on a stored procedure You can only use stored procedures in that data foundation, you cannot have a mix of stored procedure tables and standard database tables.

The properties pane displays the properties of the data foundation object that is currently selected (the entire data foundation, a table, a column, or a join). To edit properties that apply to the entire data foundation, see the Data Foundation Editor preferences [page 409].

In the data foundation view, you can work on tables and joins using commands in the *Insert* and *Detect* menus, or by clicking objects directly in the view.

The browsing panes allow you to work with different elements of the data foundation. Access the panes by clicking the corresponding tab:

- *Connections* (access stored procedures from this pane to create a specific business object.)
- *Data Foundation* (displays a tree view of the tables and joins)
- *Aliases and Contexts*
- *Parameters and Lists of Values*
- *Federation Layer*

For more information about what you can do in each of the browsing panes, see the related topic.

Navigating the data foundation view

To access a menu of commands that are available on tables, right-click the table header in the data foundation view. To select multiple tables, click the table headers while holding down the `CTRL` key.

To access commands that are available on columns, right-click the column name in the table in the data foundation view.

Several commands in the table right-click menu are available to help you locate related tables in the data foundation:

- *Select Related Tables* automatically selects all tables linked by joins to the selected table.
- *Arrange Tables* allows you to organize the positioning of selected tables and set the width of selected tables.
- *Change Table Display* allows you to change and toggle the way tables are displayed (all tables or selected tables).

- *Highlight Related Tables* grays any table that is not linked to the selected table by a join.
- *Highlight Aliases* grays all tables except the selected original table and its alias tables.
- *Highlight Original Table* grays all tables except the selected alias table and the original table it is based on.
- *Center on Selection* lets you temporarily change the zoom on the data foundation display so that all the tables in a selection are visible in the display window.

You can use the search panel to perform advanced searches on the data foundation. To open the search panel, click .

For information on commands you can use to change the display of objects in the data foundation view, see the related topic.

Related Information

Stored procedures in the data foundation [page 191]
Changing the display of objects in the Data Foundation Editor [page 135]
Arranging tables in the Data Foundation Editor [page 134]
Data Foundation Editor preferences [page 409]
Centering the view on a selection [page 139]
Data foundation properties [page 129]
Connections [page 142]
Contexts [page 181]
Using parameters, derived tables, and lists of values in the data foundation [page 184]

7.2.1 Arranging tables in the Data Foundation Editor

How to arrange the table positioning in the *Data Foundation Editor* to align them, distribute them, and set widths.

In the *Data Foundation Editor*, select some tables. Then, click the Arrange Tables icon ( ) in the toolbar, or right click them, and in the menu, select the Arrange Tables menu. Then, select one of the following commands:

Table alignment options

Option	Description
Align Left	Align the left edges of the selected tables with the left edge of the leftmost selected table.
Align Center	Align the centers of the selected tables with the center point of the selected tables.
Align Right	Align the right edges of the selected tables with the right edge of the rightmost selected table.
Align Top	Align the top edges of the selected tables with the top edge of the uppermost selected table.

Option	Description
Align Middle	Align the middles of the selected tables with the middle point of the uppermost selected table and the lowest selected table.
Align Bottom	Align the bottom edges of the selected tables with the bottom edge of the lowest selected table.
Distribute Horizontally	Space out the tables equally, horizontally.
Distribute Vertically	Space out the tables equally, vertically.
Set Width to Maximum	Set the table's column width to the width of the widest column of the selected tables.
Set Width to Minimum	Set the table's column width to the width of the narrowest column of the selected tables.

> **i Note**
>
> Use *Ctrl+Z* to undo actions if you find the resulting tables arrangement is not what you wanted. You can use *Ctrl+Z* repeatedly to undo the last actions.

See the Data Foundation Editor preferences [page 409] page to learn more about the Editor's settings.

Related Information

The Data Foundation Editor [page 133]
Changing the display of objects in the Data Foundation Editor [page 135]
Stored procedures in the data foundation [page 191]
Data Foundation Editor preferences [page 409]

7.2.2 Changing the display of objects in the Data Foundation Editor

The following commands let you change the display of objects (tables, columns, and joins) in the data foundation view.

In the *Data Foundation Editor*, you can display the tables in the following ways:

Table Display Options

Option	How the tables are displayed
Toggle	Toggle between the following view options: Collapsed, Joins Only, Expanded.

Option	How the tables are displayed
Collapsed	Collapse the tables and show just the table header.
Joins Only	Collapse the tables and show the joins between the tables.
Expanded	Expand the tables and show the full tables.

You can change the display for all the tables in the *Data Foundation Editor*, or you can select multiple tables and change the appearance of the selected tables. You can also use *Ctrl+T* to toggle the (selected) tables between the different display options.

> **i Note**
>
> When you select several tables and select Toggle for the first time, the status of the tables is aligned with the status of the first selected table.

You can also set application preferences that affect the display of data foundation objects. For more information, see the The Data Foundation Editor [page 133] and the Data Foundation Editor preferences [page 409] pages.

Related Information

Data Foundation Editor preferences [page 409]
Arranging tables in the Data Foundation Editor [page 134]
Changing the display of objects in the Data Foundation Editor [page 135]
Stored procedures in the data foundation [page 191]
Auto-arranging tables in the data foundation view [page 136]
Changing table display [page 137]
Display preferences for the data foundation view [page 418]
Setting performance-related options for the data foundation view [page 413]

7.2.2.1 Auto-arranging tables in the data foundation view

Once you have inserted tables and joins into the data foundation view, you can automatically arrange the tables according to the flow of joins from one to many.

To change the display of individual tables, see the related links.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Click the *Auto Arrange Tables* icon in the data foundation view.
   The tables are arranged in the view.

3. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Changing table display [page 137]
Hiding and unhiding table columns [page 154]
Grouping tables using families [page 137]

7.2.2.2 Changing table display

For each table in the data foundation, you can select how much table information to display in the view.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Select one or several tables.
3. Right-click one of the selected tables, and select the *Display* menu.
4. Select a display mode:

Option	Description
Collapsed	Displays only the table header.
Joins only	Displays the table header and any columns that are part of a join.
Expanded	Displays the table header and all columns.

You can also toggle through the different display modes by clicking the arrow icon on the right side of the table header, use the *Ctrl+T* shortcut, or in the right-click menu, select the *DisplayToggle Display* commands.

5. Save the data foundation by clicking the *Save* icon in the main tool bar.

7.2.2.3 Grouping tables using families

A family is a set of display parameters that can be used to visually group tables of the same type. For example, you might define different families for fact and dimension tables.

The display parameters include background color, text color and font.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Click the *Edit Families* icon in the data foundation view.
3. In the *Edit Families* dialog box, create a family for each table group:
   a. Click *Add* to define a new family.

b. Enter a name for the family in *Name*.
c. Edit the table color, background, and font for the family.
d. Click *Apply* to save the family definition.

You can export and import family definitions. Export creates a file in a local folder that can be shared between different users of the information design tool.

4. When you have added all families, click *OK*.
5. Assign tables to families. For each family:
   a. Select the tables to be assigned to one family. Click the table headers while holding down the `CTRL` key.
   b. In the *Families* list in the data foundation view tool bar, select the family.

> **i Note**
>
> When you assign a table to a family, the table acquires the display attributes of the family in the current view and all data foundation views where the table is present.

6. To remove a table from a family, select the table, and in the *Families* list, select *No family*.
7. Save the data foundation by clicking the *Save* icon in the main tool bar.

7.2.3 Views

A custom data foundation view is a subset of the data foundation *Master* view. Views are displayed as tabs at the bottom of the data foundation. Click on them to navigate from one view to another. Use views when you are editing a data foundation that contains many tables, and you are interested in working with a subset of tables. You can define multiple custom views for the data foundation.

Related Information

Display preferences for the data foundation view [page 418]
Inserting a comment into the data foundation view [page 138]
Centering the view on a selection [page 139]
Managing data foundation views [page 139]

7.2.3.1 Inserting a comment into the data foundation view

A comment is a note that you can place anywhere in a data foundation view.

1. Open the data foundation in the editor by double-clicking the data foundation in the Local Projects View.
2. Select *Insert Comment* from the *Insert* menu.

3. In the *Edit Comment* dialog box, define the display parameters of the note, and enter the comment text.
   The comment is inserted into top left corner of current view.
4. Drag the comment to the location in the view where you want it to appear.
5. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Display preferences for the data foundation view [page 418]

7.2.3.2 Centering the view on a selection

The *Center on Selection* command lets you temporarily change the zoom on the data foundation view display so that all the tables in a selection are visible in the display window.

1. In the *Data Foundation Editor*, make a selection of tables.
   For example, using the Search Panel, select all tables in a particular family.
2. Right-click the table header of one of the selected tables and select *Center on Selection*.

The data foundation display zooms so that all the selected tables appear in the display window. To reset the display, close the Search Panel if it is open, or click the *Reset Zoom* icon in the lower toolbar of the data foundation view.

> → Tip
>
> You can also center the display on a table or join by selecting the table or join name in the tree view in the *Data Foundation* panel to the left of the display.

Related Information

Searching for tables and columns in the data foundation [page 173]
Display preferences for the data foundation view [page 418]

7.2.3.3 Managing data foundation views

Table operations are allowed from all views. Any changes to a table, for example assigning the table to a family, are propagated to all views in the data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. Do one of the following:

Option	Command
To insert an empty view	Click the*Insert View* tab at the bottom of the data foundation, or right-click a view and select the *Insert View* command.
To insert a view based on a selection of tables	Select one or several tables (click the table headers while holding down the `CTRL` key). Right-click the selection and select *Insert* > *View from Selection*.
To insert a view based on search results	Use the Search Panel to search for the tables to include in the view. For more information on how to search, see the related topic. Right-click the selection in the search results view and select *Insert* > *View from Selection*.

3. Enter a name for the view and click *OK*.

   A new tab appears at the bottom of the view pane and the new view is displayed.
4. To add a table to a view:
   a. Click the *Master* view tab, or any other view that contains the table you want to add.
   b. Select the table or tables you want to add.
   c. Right-click the table header and select *Add to View*.
   d. Select the view from the list (only views that do not already contain the table are listed).
5. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Searching for tables and columns in the data foundation [page 173]
Display preferences for the data foundation view [page 418]

7.2.3.3.1 Copying a data foundation view core into a linked data foundation

You can insert into a linked universe a copied version of a data foundation view based on a core view.

> **i Note**
>
> This is only available for linked universes.

1. Select *Insert View* from the *Insert* menu. and select *Insert View from Core*.

The *Import a view from core* dialog box appears.

2. Choose a core universe.
3. Choose a view.
4. Click *OK*.

The Core View is copied in the inked universe.

Related Information

Display preferences for the data foundation view [page 418]

7.2.3.4 Organizing data foundation views

The right-click contextual menu in the data foundation views contains the following commands:

Option	Description
Go to	This menu opens a sub-menu contains the list of data foundation views. • If the summary has been enabled, then the Summary choice is on top of the list, otherwise it is not displayed • It is followed by the Master choice, and then the ordered list of views • If the list is too large, then arrows on top and on the bottom on the list allow the user to scroll in this list of views • The selected view is identified by a bullet in front of it
Rename	To rename the selected view.
Copy	To copy the selected view.
Cut	To cut the selected view.
Paste	To paste the copied view after the selected view.
Duplicate	To copy and paste the selected view after the selected view.
Delete	To delete the selected view.
Insert View	To insert a new view after the selected view.
Reorder Views	This opens a dialog box where you can reorder the views.

Related Information

Display preferences for the data foundation view [page 418]

7.2.3.5 Copying a data foundation view core into a linked data foundation

You can insert into a linked universe a copied version of a data foundation view based on a core view.

> **i Note**
>
> This is only available for linked universes.

1. Select *Insert View* from the *Insert* menu. and select *Insert View from Core*.
   The *Import a view from core* dialog box appears.
2. Choose a core universe.
3. Choose a view.
4. Click *OK*.

The Core View is copied in the inked universe.

Related Information

Display preferences for the data foundation view [page 418]

7.3 Connections

Connections in the data foundation are listed in the *Connection* pane of the *Data Foundation Editor*. Some connections allow for multiple databases (called qualifiers), with different owners:

- Some data sources provide both qualifiers and owners (for example, Microsoft SQL Server)
- Some data sources only provide qualifiers (for example, MySQL and text files)
- Some data sources only provide owners (for example, Oracle, SAP HANA, DB2, and Teradata)

For single-source data foundations, the *Connection* pane display is as follows:

- Qualifiers, if available, are listed under the connection
  - Owners, if available, are listed under each qualifier
    - Tables are listed under each owner

    - Columns are listed under each table

For multisource-enabled data foundations, the *Connections* pane display is as follows:

- Qualifier.Owners, known as the schema, are listed under the connection (or owners only, if the data source does not provide qualifiers).
    - Tables are listed under each schema
        - Columns are listed under each table

The list of tables is sorted alphabetically.

By default, tables are listed for all qualifiers and owners. To list only tables for the currently used qualifier/owner, click the *Show Qualifiers and Owners* icon to deselect it.

When the connection contains more than 1000 qualifier/owners or tables, the qualifier/owners or tables are grouped in packages of 1000. The packages are listed with the first several letters of the first and last qualifier/owner or table names in the package surrounded by brackets, for example:

[AAAA....] - [MMMM]

[NNNN...] - [ZZZZZ]

For tables that are already inserted into the data foundation, the table icon displays a green check mark: . You can insert a table into the data foundation by double-clicking the table name in the *Connections* pane.

Navigating the list of tables in the *Connection* pane

The *Show/Hide Table Search* icon, lets you search for tables in a connection to obtain a filtered list of tables.

Some connections have different table types (for example, in an SAP HANA connection, you can have several table types, including *Analytic View* and *Calculation View*). The *Filter by Table Type* icon lets you select table types to filter the list of tables displayed in the connection.

For SAP HANA connections, the *Filter Information Models* icon by default filters the list to show only tables representing information models.

The table search, filter by table type, and filter on information models can be used in combination. For more information on searching and filtering, see the related topics.

Operations on connections

You can do the following tasks on connections from the *Connection* pane:

- *Change* lets you change the connection and its associated properties. For more information about this task, see the related topic.

- *Open* opens the connection or connection shortcut properties in the Connection Editor.
- *Test* lets you test if the database referenced by the connection is available.

In addition, for multisource-enabled data foundations, you can do the following tasks from the *Connections* pane:

- *Add Connections* lets you add connections to the data foundation. For more information about this task, see the related topic.
- *Remove* lets you remove a connection from the data foundation. The connection itself remains in the repository and registered with the catalog name.

Related Information

Searching for tables in the Connection pane [page 146]
Tables [page 149]
Filtering the tables in the connection by table type [page 147]
Filtering tables in an SAP HANA connection by information model [page 148]
Changing a connection in a data foundation [page 145]
Adding connections to a data foundation [page 144]
Working with the Connection Editor [page 87]
Setting connection display preferences for the Data Foundation Editor [page 409]

7.3.1 Adding connections to a data foundation

To add connections to the data foundation, the following conditions are required:

- The data foundation type must be multisource-enabled.
- The connections to add must be relational, secured connections.
- For each connection to add, you must create a connection shortcut in the local project where the data foundation is stored.

1. Double-click the data foundation name in the local project to open the *Data Foundation Editor*.
2. In the *Data Foundation Editor*, click the *Connections* tab.
3. In the *Connections* pane, click the *Add Connections* icon .
   The *Add Connections* dialog box lists the available connections, including the connections currently defined in the data foundation.
4. Select the connection shortcut name of each connection you want to add, and click *Next*.
5. In the *Connections Properties* dialog box, you can define additional connection properties. A dialog box opens for every connection added.

   For more information about multisource-enabled connection properties, see the related topic.
6. When you have finished defining the properties for additional connections, click *Finish*.
7. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Connection shortcuts [page 121]
Data foundations with multiple connections [page 125]

7.3.2 Changing a connection in a data foundation

The connection you change to must be a relational connection. For multisource-enabled data foundations, the connection must also be secured.

Before you can change to a connection, you must create the local connection or connection shortcut in the local project where the data foundation is stored.

1. Double-click the data foundation name in the local project to open the *Data Foundation Editor*.
2. In the *Data Foundation Editor*, click the *Connections* tab.
3. In the *Connections* pane, right-click the connection and select *Change...*.
4. Select a new connection. How to do this depends on the data foundation type:
   - If the data foundation is single-source, the *Change Connection* dialog box lists the available connections, including the currently defined connection. Select the connection you want to change to, and click *OK*.
     If you have delimitation overrides set, you will be prompted for the delimitations you want to use in the new connection. For more information, see the related topic.
   - If the data foundation is multisource-enabled, the *Change Connection* dialog box displays the connection properties for the currently defined connection. Click the browse button in the *Connection* text box. A dialog box lists the available connections. Select the connection you want to change to, and click *Finish*.
5. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Selecting delimitation overrides to keep [page 145]

7.3.3 Selecting delimitation overrides to keep

When you change the connection in a data foundation, the delimitation requirements for the new connection are automatically detected. Any delimitation overrides made for the previous connection may or may not apply. The *Select Delimitation Overrides to Keep* dialog box displays so that you can indicate which overrides to keep.

Only tables with delimitation overrides for the table name, qualifier, or owner are listed. The override value is shown in the column under *Name*, *Qualifier*, or *Owner*.

1. For each override:

- To keep the delimitation value set for the previous connection, leave the check box selected.
- To remove the delimitation override and use the newly detected delimitation value, unselect the check box.

> **i Note**
>
> A tool tip for each override gives you the following information:

- The table name, qualifier, or owner in the previous connection with the previous delimitation.
- The new table name, qualifier, or owner in the new connection with the previous delimitation.
- The new table name, qualifier, or owner in the new connection with the newly detected delimitation.

2. When you are finished selecting overrides to keep, click *Finish* to complete the connection change.

For all other tables in the data foundation (without previous delimitation overrides) the automatically detected delimitation for the new connection applies.

Related Information

Tables [page 149]

7.3.4 Searching for tables in the Connection pane

You can create a filtered list of tables in the *Connection* pane using a search string. You can insert tables into the data foundation from the filtered list.

1. In the *Connection* pane of the *Data Foundation Editor*, click the *Show/Hide Table Search* icon, .
   The *Connection* pane splits. The top pane continues to display all tables in all connections. The lower pane (Search pane) displays only the tables matching the search string in the selected connection.
2. In the *Connection* pane, select the part of the connection catalog that you want to search. You can select:
   - The entire connection
   - A qualifier (if available)
   - An owner (if available)

   > **i Note**
   >
   > For data foundations with multiple connections you can search only one connection at a time. You must select a connection. You can select the entire connection or a schema.

3. In the Search pane, enter the string you want to search for.

   > **i Note**
   >
   > - The wildcard character is allowed. In your search string, you must enter the wildcard character that is defined for the data source.
   > - Many data sources use the percent sign (%) as a wildcard, where the percent sign matches one or more characters. To enable the automatic insertion of wildcard characters (%), click the *Enable*

Automatic Wildcards icon . When automatic wildcards are enabled, if, for example, you enter the search text **2012**, the application searches for `%2012%`.

- The search is case-sensitive when searching connections in multisource-enabled data foundations.

4. Click the search icon in the Search pane.
   The tables with names matching the search string are listed in the Search pane. You can double-click a table in the Search pane to insert it into the data foundation.
5. To start a new search, change the search string, or select a different part of the catalog in the *Connections* pane to search, and click the search icon in the Search pane again.

You can also filter the list of tables by table type using the icon, or, in SAP HANA connections, by information model using the . If the list of tables in the *Connection* pane is filtered, the filter applies in both the *Connection* pane and the Search pane. For more information about filtering, see the related topics.

Related Information

Connections [page 142]
Filtering the tables in the connection by table type [page 147]
Filtering tables in an SAP HANA connection by information model [page 148]

7.3.5 Filtering the tables in the connection by table type

In the *Connection* pane of the *Data Foundation Editor*, you can filter the list of tables in the connection by table type. The filter mechanism is case sensitive.

1. Select the connection you want to filter, and open the qualifier and/or owner until tables are listed.
   The application discovers table types in the connection as tables of each type are listed in the *Connection* pane. Once the first table type is discovered, the *Filter by Table Type* icon is available.
2. Click the down arrow next to the *Filter by Table Type* icon .
   All table types discovered so far are listed. When a table type is discovered, it is automatically selected to be displayed in the connection.
3. To stop displaying a table type, unselect it in the list.
   The tables are filtered in both the *Connection* pane and the table search results pane. For more information on table search, see the related topic.
4. To redisplay a table type, open the *Filter by Table Type* list and select the table type in the list.

The table filter only affects the display. When you close the *Data Foundation Editor*, or when you change the connection, the filter selection is reset.

You can also filter the list of tables in the connection with a search string using the icon to open the Search pane. The table type filter applies in both the *Connection* pane and the Search pane. For more information about searching for tables, see the related topic.

Related Information

Connections [page 142]
Searching for tables in the Connection pane [page 146]
Filtering tables in an SAP HANA connection by information model [page 148]

7.3.6 Filtering tables in an SAP HANA connection by information model

In the *Connection* pane of the *Data Foundation Editor*, the list of tables in an SAP HANA connection (owner _SYS_BIC) is filtered by default to only tables representing information models (such as an Analytic View or Calculation View). You can toggle the filter off or on.

You can also set an application preference to change the default state for the information model filter. For more information, see the related topic.

1. Select the connection you want to filter and open the node for the _SYS_BIC owner.
2. To show only information models, make sure the *Filter Information Models* icon is selected. The filter mechanism is case sensitive.
3. To show all tables in the _SYS_BIC owner, unselect the *Filter Information Models* icon.

You can also filter the list of tables by table type using the icon.

You can search the list of tables in the connection with a search string using the icon to open the Search pane. The information model and table type filters apply in both the *Connection* pane and the Search pane.

Related Information

Setting connection display preferences for the Data Foundation Editor [page 409]
Filtering the tables in the connection by table type [page 147]
Searching for tables in the Connection pane [page 146]

7.4 Table elements

A table is a graphical representation in the data foundation of a physical database table. When you insert a table in the data foundation, you can edit the table properties, hide columns, show values, and so on.

Related Information

Tables [page 149]
Table keys [page 157]
Table row counts [page 158]
Inserting a calculated column [page 159]
Inserting a time column [page 160]
Derived tables [page 168]
Alias tables [page 165]
Input columns [page 160]
Showing table values [page 162]
Showing column values [page 163]
Profiling column values [page 164]

7.4.1 Tables

A standard table is a graphical representation in the data foundation of a physical database table. You create standard tables when you insert database tables into the data foundation. The table and column names are inherited from the data source.

> **! Restriction**
>
> The information design tool does not support any table or column names that are the same as an SQL reserved word. Rename these objects in the data source before inserting them into the data foundation.

> **! Restriction**
>
> A data foundation table cannot have the same name as the qualifier or owner (see the section about qualifiers and owners). If a database table has the same name as the qualifier or owner, create an alias table with a different name.

Some data sources have tables of different types. The table type is inherited from the data source and stored as a table property in the data foundation.

Once you have inserted a standard table, you can modify it in the following ways:

- Edit table properties (name and description)
- Set the case of the table name

- Hide and unhide columns
- Change the data type of columns
- Set columns as primary and foreign keys

Tables in the data foundation can also be federated, derived, or alias tables. For more information, see the related topics.

About hidden columns

Some connections specify that certain columns in the data source are not for querying and are therefore hidden when the table is inserted into the data foundation. You can hide and unhide columns in standard tables.

Hidden columns are ignored in most workflows. For example, they do not appear in the data foundation table display, when showing table values, or when dragging and dropping the table into the business layer pane to create the related business layer objects.

If a table is joined on a hidden column, in the data foundation display, the join line points to the table header. The column is displayed when editing the join.

Hidden columns are taken into account during a refresh structure on the data foundation.

About qualifiers and owners

Some connections allow for multiple databases (called qualifiers), with different owners. Standard tables and their columns inherit the current qualifier and owner from the database. The syntax for a standard table name is as follows:

- A single-source standard table name, if the table is inserted from the current qualifier and owner, has the syntax:
  "table_name"
- A single-source standard table name, if the table is inserted from a different qualifier or owner, has the syntax:
  "database_qualifier"."database_owner"."table_name"
- A multisource-enabled standard table name has the syntax:
  `@catalog`('short_name')."database_qualifier.database_owner"."table_name"

> **i Note**
>
> Qualifier and owner are not relevant for some connections, in which case they do not appear in the data foundation connection pane and are not inherited by tables.

About delimitation

When you insert a table into the data foundation, the database requirements for delimited names is determined and the information is stored with the data foundation table properties. If the table or column

names need to be delimited, the names are surrounded by double quotes in the display of the table in the data foundation.

The table, column, qualifier, and owner names that need to be delimited are surrounded by double quotes when used in an SQL expression.

In single-source data foundations, you can override the default delimitation requirements as follows:

- For standard tables you can override the delimitation on table names, column names, qualifiers, and owners.
- For alias tables you can override the delimitation on table names only. The overrides on the columns are inherited from the original table.
- For derived tables you can override the delimitation on table names only.

To override the default delimitation for tables and columns, use the *Delimit* command. For qualifiers and owners, use the *Change Qualifier/Owner* command.

When you change the connection in the data foundation, if you have delimitation overrides set, you will be prompted for the delimitations you want to use in the new connection.

Related Information

Inserting tables into the data foundation [page 151]
Editing table properties [page 153]
Setting case of table names [page 154]
Hiding and unhiding table columns [page 154]
Changing column data types [page 155]
Table keys [page 157]
Federated tables [page 197]
Derived tables [page 168]
Alias tables [page 165]
Changing qualifiers and owners [page 155]
Changing table and column delimitation [page 156]
Selecting delimitation overrides to keep [page 145]
Changing the display of objects in the Data Foundation Editor [page 135]

7.4.1.1 Inserting tables into the data foundation

Before you begin, verify that table and column names in the data source are not the same as an SQL reserved word. If so, rename these objects in the data source before inserting them in the data foundation.

This procedure describes inserting tables from the data source. To insert a federated table, see the related topic.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. Select *Insert Tables* from the *Insert* menu in the data foundation view.

   The *Insert Tables* dialog box lists the connections defined in the data foundation.

3. Expand the connection to see the database tables referenced in the connection.

   By default, tables are listed for all qualifiers and owners. To list only tables for the currently used qualifier/owners, click the *Show Qualifiers and Owners* icon .

   You can filter the list of tables in the connection by table type using the *Filter by Table Type* icon . The application discovers table types in the connection as tables of each type are listed in the pane. Once the first table type is discovered, the *Filter by Table Type* icon is available. Click the down arrow next to the *Filter by Table Type* icon to select a table type.

   For SAP HANA connections, the list of tables (owner _SYS_BIC) is filtered by default to only tables representing information models (such as an Analytic View or Calculation View). Click the *Filter Information Models* icon to toggle the filter off and on.

4. Select a table name to insert it and all its columns into the data foundation.

   Tables that are already inserted into the data foundation have an icon with a green check mark. If you insert an existing table, an alias table is inserted and you are prompted to enter a name for the alias table.

   To show the values in a table, right-click the table name and select *Show Table Values*. To show the values in one column, expand the table, right-click the column name and select *Show Column Values*.

5. Select the objects you want to detect and insert automatically into the data foundation when inserting the selected tables:

Option	Description
Detect keys	Sets the key columns in the data foundation tables as they are in the database tables.
Detect row counts	Saves the number of rows in each table in the data foundation.
Detect joins	Inserts the joins between the tables being inserted. For multisource-enabled data foundations, only joins between tables referenced by the same connection are detected. To detect joins between tables referenced by different connections, use the *Detect Joins* command after inserting the tables.
Detect cardinalities	Saves the cardinalities of the joins as they are in the database joins.

   The recommended detection options are selected by default. You can change the defaults in the application preferences. See the related topic on setting table and join detection options.

6. Click *Finish* to insert the selected tables.
7. Save the data foundation by clicking the *Save* icon in the main tool bar.

You can also insert tables into the data foundation by dragging them from the *Connections* pane and dropping them into the data foundation view.

Related Information

Tables [page 149]
Inserting a federated table into the data foundation [page 210]
Alias tables [page 165]
Detecting joins [page 178]
Setting table and join detection options [page 412]
Preferences for inserting tables and joins [page 414]

7.4.1.2 Editing table properties

For alias and standard tables you can edit the table name and description.

For standard tables, you can also remove columns from the table display, edit the column data types, and set or unset primary and foreign keys. The column changes you make are also made to any related alias tables.

Edit federated tables in the *Federation Layer*.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header in the data foundation view and select *Edit*.

   If you are editing a derived table, the *Edit Derived Table* dialog box appears. For information on editing derived tables, see the related topic.
3. To change the table name, enter a new *Name*.

   > **i Note**
   >
   > When you change the name of a standard table, you break the link with the database table. For information about renaming tables using aliases, see the related link about alias tables.
4. To remove columns from the table display, unselect the columns that you want to hide and click *OK*.

   This only affects the display in the data foundation view. The columns remain visible when showing table values or inserting the table into the business layer.
5. To change the data type of a column, select a data type in the list in the *Data Type* column.

   The next time you refresh the data foundation structure, the original data type of the column in the database is proposed.
6. To set or unset keys, select *None*, *Primary*, or *Foreign* from the list in the `Keys` column.

   The next time you use the *Detect Keys* command, the keys defined in the database tables override the keys set manually for a table. You can set an application preference so that if no keys are detected, the keys that you set manually in a data foundation table are kept.
7. Optionally, enter or edit the table *Description*.
8. Click *OK* to save the changes.
9. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Derived tables [page 168]
Alias tables [page 165]
Editing a federated table [page 200]
Table keys [page 157]
Setting table and join detection options [page 412]
Tables [page 149]
Refreshing a data foundation [page 171]

7.4.1.3 Setting case of table names

Some databases require that table names be all upper case or all lower case. Use the *Set case to* command to change the case of table names.

You cannot set the case of a federated table in the data foundation. You need to edit the table name in the *Federation Layer*.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header in the data foundation view and select *Set case to*. Then select *Upper case* or *Lower case*.

   To select multiple tables, click the table headers while holding down the `CTRL` key.
3. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Editing a federated table [page 200]

7.4.1.4 Hiding and unhiding table columns

Hiding columns applies only to standard tables. For more information on the impact of hiding columns, see the related topic.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header in the data foundation view and select *Edit*.
3. Unselect the columns that you want to hide, select columns that you want to unhide, and click *OK*.

> **i Note**
>
> Hidden columns are ignored in most workflows. For example, they do not appear in the display, when showing table values, or when dragging and dropping the table into the business layer pane to create

> the related business layer objects. You can hide a column that is involved in a join. In this case, the column appears when editing the join.

4. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Tables [page 149]

7.4.1.5 Changing column data types

You can change column data types for standard tables in the data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header in the data foundation view and select *Edit*.
3. In the list of columns, select a data type in the list in the *Data Type* column.

> **i Note**
>
> The next time you refresh the data foundation structure, the original data type of the column in the database is proposed.

4. Click *OK* to save the changes.
5. Save the data foundation by clicking the *Save* icon in the main tool bar.

7.4.1.6 Changing qualifiers and owners

You can change the qualifiers and owners for standard tables in the data foundation. For tables in single-source data foundations, you can also change whether to delimit the qualifier and owner names.

1. Open the data foundation in the editor by selecting it in the Local Projects View.
2. Right-click the table header in the data foundation view and select *Change Qualifier/Owner*.

   To select multiple tables, click the table headers while holding down the `CTRL` key.
3. In the *Change Qualifier/Owner* dialog box, click the browse button in the *Qualifier* field and select a new qualifier.

   If the qualifier name is delimited by default, the *Delimit* option is selected. To override the default delimitation, select or unselect *Delimit*.
4. To change the owner, click the browse button in the *Owner* field and select a new owner.

   If the owner name is delimited by default, the *Delimit* option is selected. To override the default delimitation, select or unselect *Delimit*.

5. When you are finished changing the qualifier and owner information, click *OK*.
6. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Tables [page 149]

7.4.1.7 Changing table and column delimitation

You can override the default delimitation of table and column names for tables in a single-source data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. To override the delimitation of table names and/or all column names in a table, right-click the table header in the data foundation view and select *Delimit*.

   To select multiple tables, click the table headers while holding down the `CTRL` key.

   Select one of the options:

Option	Description
Yes (Tables and Columns)	Delimits the table names and all column names.
Yes (Tables only)	Delimits the table names.
Yes (Columns only)	Delimits all column names.
No (Tables and Columns)	Stops delimiting the table names and all column names.
No (Tables only)	Stops delimiting the table names.
No (Columns only)	Stops delimiting all column names.

   > **i Note**
   >
   > For alias and derived tables, you can override only table names. The columns in alias tables inherit overrides from the original table.

3. To override the delimitation of individual columns, right-click the column name and select *Delimit*.

   To select multiple columns, click the columns while holding down the `CTRL` key.
4. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Tables [page 149]

7.4.1.8 Table keys

Tables in the data foundation can have two types of keys:

Key	Description
Primary	Single or combination of columns in a table whose values identify each row in the table. The primary key guarantees row uniqueness in a table. Each table has only one primary key.
Foreign	Column or combination of columns whose values are required to match a primary or another unique key in another table. Foreign keys implement constraints, for example, not allowing a sale to be added in the **`Sales`** table for a customer that does not exist in the **`Customer`** table. Each table can have multiple foreign keys.

Keys are indicated with icons next to column in the data foundation view.

You can set keys in data foundation tables manually, or by detecting the keys in the database tables. Detecting keys does not apply to federated tables.

Related Information

Setting and detecting table keys [page 157]

7.4.1.8.1 Setting and detecting table keys

You can set keys in data foundation tables manually, or by detecting the keys in the database tables. Detecting keys does not apply to federated tables.

1. Open the data foundation in the *Data Foundation Editor* by double-clicking the data foundation name in the Local Projects View.
2. To set keys according to the keys detected in the database, in the data foundation view, select *Detect Keys* from the *Detect* menu.

   You can set an application preference to automatically detect keys when inserting a table into the data foundation. See the related topic on setting preferences for the *Data Foundation Editor*.
3. To set or unset keys manually, right-click the column in the table and select *Set as Key* and select *Primary*, *Foreign*, or *None*.
4. Save the data foundation by clicking the *Save* icon in the main tool bar.

You cannot set keys for alias tables. Alias tables inherit keys from the original table.

The next time you use the *Detect Keys* command, the keys defined in the database tables override the keys set manually for a table. You can set an application preference so that if no keys are detected, the keys that you set manually in a data foundation table are kept.

Related Information

Table keys [page 157]
Setting table and join detection options [page 412]

7.4.2 Table row counts

Detect row counts

The number of rows in database tables can be detected and stored in the data foundation. Row counts are used to detect cardinalities in the absence of table keys.

When you detect rows counts, the number of rows for the selected tables is counted and stored.

> **i Note**
>
> Column filters are not applied when detecting row counts.

You can also set estimated row counts for tables. This can be useful if you are working with a reduced sample of data, but want queries to be optimized for the size of the production data. The row count that you set is replaced by the detected row count when you do a detect row count for that table.

The *Detect Row Count* command on the *Detect* menu lists the current row counts for all tables in the data foundation. From this list, you can set row counts and detect row counts for a selection of tables.

To detect row count for one table, right-click the table header in the data foundation view and select *Detect* > *Row Count*. The row count for the selected table is updated. To select multiple tables, click the table headers while holding down the `CTRL` key.

You can set an application preference to detect row counts automatically every time a table is inserted into the data foundation. See the related topic on setting table and join detection options.

Count rows

Use the *Count Rows* command on multiple tables linked by joins to see the number of rows returned by the resulting query. Column filters are applied.

To count rows returned in a query, select tables in the data foundation view in one of the following ways:

- Right-click a table and select *Select Related Tables*.
- Click the table headers while holding down the `CTRL` key.

Then right-click a table in the selection and select *Count Rows*.

Related Information

Setting table and join detection options [page 412]

7.4.3 Inserting a calculated column

A calculated column is a new column in a data foundation table that is the result of a calculation based on one or more columns of the same table.

> **i Note**
>
> Inserting a calculated time column based on a column with a time-related data type is a special case of a calculated column. For the procedure to insert a time column, see the related topic.

The following rules apply to calculated columns:

- You can insert calculated columns into standard tables only.
- You can include only columns from the same table in the SELECT statement.
- Sub-queries are not allowed.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header in the data foundation view, or right-click in a column, and select *Insert Calculated Column*.
3. If the data foundation is multisource-enabled and you want to use database-specific SQL to define the calculated column, select the *Database-specific* option.

   > **i Note**
   >
   > Some data sources do not support database-specific SQL for defining calculated columns. In this case, the *Database-specific* option is unavailable.

   For more information about SQL expressions in multisource-enabled data foundations, see the related topic.
4. Build the SQL SELECT statement that defines the column by dragging and dropping columns and functions into the *SELECT* pane.

   For more information about using the SQL expression editor, see the related topic.
5. Click *Validate* to check the validity of the SQL expression.
6. Click *OK*.

   The column is inserted into the table and appears in the data foundation view with a special icon. A tool tip displays the SQL expression of the calculated column when you pass the cursor over the column name.
7. To check the results of the calculated column, right-click the column and select *Show Column Values*.
8. Save the data foundation by clicking the *Save* icon in the main tool bar.

To edit the definition, right-click the column name in the table in the data foundation view and select *Edit Calculated Column*.

Related Information

Inserting a time column [page 160]
The SQL/MDX expression editor [page 421]
SQL expressions in multisource-enabled data foundations [page 126]

7.4.4 Inserting a time column

A time column is a calculated column that contains a date part (for example, month, quarter, or year) based on a column with a time-related data type.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click a column with a time-related data type and select *Insert Time Column*.

   Columns with time-related data types have a special icon that resembles a calendar.
3. Select a date part from the list.

   A calculated column is inserted into the table and appears in the data foundation view with a special icon. A tool tip displays the SQL expression of the calculated column when you pass the cursor over the column name.
4. To check the results of the calculated column, right-click the column and select *Show Column Values*.
5. Save the data foundation by clicking the *Save* icon in the main tool bar.

To edit the definition, right-click the column name in the table in the data foundation view and select *Edit Calculated Column*.

Related Information

Inserting a calculated column [page 159]

7.4.5 Input columns

An input column is a parameter in the data source that expects a value. The parameter is represented in a table column in the data foundation.

For each input column, you can provide a static value, or a parameter defined in the data foundation. The parameter can prompt the user for a value and be associated with a list of values. For some input columns, providing a value is optional. An SAP HANA input parameter can also accept one or multiple entries if the option Multiple Entries is activated in SAP HANA Studio.

Following are examples of input columns in the data foundation:

- SAP BW key date variables. An input column is inserted into each table in the data foundation that handles time-dependent data. To resolve these input columns at query time, a parameter is inserted into the data

foundation called key date. Because SAP BW key date variables are mandatory, by default, at query time, the key date parameter is not prompted. It is automatically assigned the current date. You can edit the prompt parameters in the data foundation.

- SAP ERP ABAP function input parameters. One data foundation table is created to map the main function. It contains input columns for the input parameters of the function. These parameters can be mandatory or optional, single or multiple. For mandatory parameters, you enter either a static value or a data foundation parameter for the associated input column.

Related Information

Editing input columns [page 161]
SAP BW [page 40]

7.4.5.1 Editing input columns

To assign a parameter to an input column, you must first define the parameter in the data foundation. For SAP BW key date variables, a data foundation parameter is automatically inserted. For more information on parameters, see the related topic.

1. You can list input columns to edit in three ways:

Option	Command
To list input columns for a table	Right-click the table header in the data foundation view and select *Edit Input Columns*.
To list input columns for all tables	Right-click anywhere in the data foundation view and select *Edit Input Columns*.
To edit a single input column	Right-click the column name in the data foundation view and select *Edit Input Column*.

   If no input columns exist in the table or data foundation, the *Edit Input Column* command is not available.

2. To assign a value to an input column, in the *Edit Input Columns* dialog box, select the column in the list.
   - *No assignment* is selected by default and means that no value or parameter is assigned to the column.
   - To assign a static value, select *Value* and enter a value in the text box. You can enter blanks or leave the text box empty to assign a blank value to a column with character data type.
   - To assign a parameter, select *Parameter*. Click the icon to select from a list of parameters defined in the data foundation.

   > **i Note**
   >
   > The *Values* column indicates mandatory input columns as *[mandatory]*, and optional input columns as *[optional]*.

3. To assign the values you entered, click *OK*.
4. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Parameters [page 238]

7.4.6 Showing table values

You can show values for one or more tables in a data foundation. If filters are defined on any of the columns, the filters are applied when showing values. To see values in the database for a table (no data foundation filters are applied), show the values on a table in the Connections pane.

The show values command by default opens a tab in the editor to display the values. You can set a preference to have the values open in a dedicated view or a dialog box. For more information, see the related topic.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

   You can also show table values from the data foundation view in the *Business Layer Editor*. In this case, open the business layer.
2. Do one of the following:

Option	Command
To show values for one or more tables with filters applied	Right-click the table header in the data foundation view. To select multiple tables, click while holding down the `CTRL` key.
To show values for one table without filters applied	From the *Connections* pane in the *Data Foundation Editor*, expand the connection and right-click the table name.

3. Select *Show Table Values*.

   The show values window appears. To see what you can do in this window, see the related topic on showing values in a data source.

Related Information

Showing values in a data source [page 162]
Preferences for the Show Values command [page 417]

7.4.6.1 Showing values in a data source

You can show values in the underlying data source for data foundation tables and columns, business layer objects, and connections. This topic describes what you can do when showing values.

> **i Note**
>
> Unless you are showing values from the connection, any column filters defined in the data foundation are applied when retrieving values.

To limit the number of rows returned from the data source, enter a number in *Max Rows*.

To see the query script, click *View Log*.

What you can do when showing values in the *Raw Data* tab:

- Re-order the columns in the display: Drag and drop column headers to the new location in the table.
- Sort rows by column: Click the column header to sort the rows by the column value in ascending or descending order.
- Filter rows by column: Click *Add Filter* and construct a filter for one or multiple columns using the filter value selector.
- Filter results to rows that contain a character or group of characters in any of the columns: Enter the characters to filter on in the *Enter your filter* text box. You can use the * character as a wildcard, for example:
  - If you enter B in the filter text box, only rows with a column value that contains the character B display.
  - If you enter B*, only rows with a column that contains a value starting with B display.
  - If you enter *B, only rows with a column that contains a value ending with B display.
- Export the results to a local file (.csv or .xml format): Click *Save as File*.

To see distinct values for a selected column, click the *Distinct Values* tab and select a column.

To build and format a chart, click the *Analysis* tab.To save the chart as an image, click *Save As Image*.

Related Information

Showing table values [page 162]
Showing column values [page 163]
Showing business layer object values [page 296]

7.4.7 Showing column values

You can show values for one or more columns in a data foundation table. If a filter is defined on the column, the filter is applied when showing values. To see values in the database for a column (no data foundation filters are applied), show the values on a column in the Connections pane.

The show values command by default opens a tab in the editor to display the values. You can set a preference to have the values open in a dedicated view or a dialog box. For more information, see the related topic.

1. Open the data foundation in the *Data Foundation Editor* by double-clicking the data foundation name in the Local Projects View.

   You can also show column values from the data foundation view in the *Business Layer Editor*. In this case, open the business layer.

2. Do one of the following:

Option	Command
To show values for one or more columns with filters applied	Right-click the column name in the data foundation view. To select multiple columns, click while holding down the `CTRL` key.
To show values for one column without filters applied	From the *Connections* pane in the *Data Foundation Editor*, expand the connection and right-click the column name.

3. Select *Show Column Values*.

   The show values window appears. To see what you can do in this window, see the related topic on showing and profiling values in a data source.

Related Information

Showing values in a data source [page 162]
Preferences for the Show Values command [page 417]

7.4.8 Profiling column values

You can profile the values for a column in a data foundation table. Profiling shows graphically (in a pie or bar chart) the number of occurrences of each value of a column. If the column has a filter defined, the filter is applied.

1. You can profile column values from the data foundation view in either the *Business Layer Editor* or *Data Foundation Editor*. Open the editor by double-clicking the resource in the Local Projects View.
2. In the data foundation view, right-click the column name in the table display and select *Profile Column Values*.
   The profiled data is shown in a table.
3. To see the profiled data as a chart, select the *Pie Chart* or *Bar Chart* option.

7.4.9 Alias tables and derived tables

The following sections describe how to create alias tables and derived tables.

- An alias table is basically a copy of an existing table. This way you can use the same table more than once in a query. Do this to break loops and resolve fan traps.
- A derived table contains results of calculations and functions based on other tables. These operations are performed before the result set is returned to a document, which saves time and reduces the need for complex analysis of large amounts of data at the report level. A derived table can also be used to group columns from different tables, and to combine several tables.

Related Information

Alias tables [page 165]
Inserting alias tables [page 166]
Detecting alias tables [page 166]
Highlighting aliases [page 167]
Highlighting the original table of an alias [page 167]
Derived tables [page 168]
Inserting a derived table based on a data foundation table [page 168]
Merging tables [page 169]
Inserting and editing a derived table [page 169]

7.4.9.1 Alias tables

An alias table is a reference to a standard, derived, or federated table in the data foundation. It is an identical duplicate of the original table (except for column filters), but has a different name. The data in the table is exactly the same as the original table, but the different name "tricks" the SQL of a query to accept that you are using two different tables.

You use alias tables to break loops in the join paths and fan traps in the data foundation. The *Detect Aliases* command analyzes the join paths and proposes alias tables to break any loops detected in the data foundation. For more information about resolving loops, see the related topic.

You also use alias tables to rename a table. The link between the data foundation and database is based on the table name. If you create an alias to give the table a new name, the link to the database table is preserved, but the alias table name is used in the data foundation.

To find alias tables already inserted into the data foundation, you can do a data foundation search. There are also commands to highlight the alias tables of an original table, and highlight the original for an alias table. For more information, see the related topics.

Related Information

Detecting alias tables [page 166]
Inserting alias tables [page 166]
Resolving loops [page 187]
Searching for tables and columns in the data foundation [page 173]
Highlighting aliases [page 167]
Highlighting the original table of an alias [page 167]

7.4.9.1.1 Inserting alias tables

You use alias tables to break loops in the join paths in the data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects view.
2. In the data foundation view, select the table to be the basis of the alias.

   You can create aliases for more than one table at a time. Click the table headers while holding down the `CTRL` key.
3. Right-click the selection and select *Insert* > *Alias Table*.
4. Select *Duplicate joins* to join the generated alias tables with the same tables as the aliased tables.
5. Edit the names for the alias tables in the *Alias Name* column, and click *OK*.

The selected alias tables are inserted into the data foundation. The original table name is listed in parentheses in the table header.

To edit the name and description of an alias table, click the table header in the data foundation view and select *Edit*.

Related Information

Alias tables [page 165]

7.4.9.1.2 Detecting alias tables

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Select *Detect Aliases* from the *Detect* menu in the data foundation view.

   You can also detect aliases from the *Aliases and Contexts* pane in the *Data Foundation Editor*. Click the *Detect Aliases* icon .

   The command analyzes the join paths and proposes alias tables to break any loops detected in the data foundation.
3. If alias tables are proposed, select which aliases you want to insert automatically.

The selected alias tables are inserted into the data foundation. The original table name is listed in parentheses in the table header.

To edit the name and description of an alias table, click the table header in the data foundation view and select *Edit*.

Related Information

Alias tables [page 165]

7.4.9.1.3 Highlighting aliases

Use this command to highlight the alias tables associated with a standard or derived table in the data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header of the original table and select *Highlight Aliases*.

   The original and any associated alias tables are highlighted. All other tables are grayed.

   > **i Note**
   >
   > Some alias tables may be off the visible area of the data foundation view. You can quickly check for hidden highlighted tables by clicking the *Fit to Window* icon at the bottom of the data foundation view . To undo the *Fit to Window*, click the *Reset Zoom* icon .

3. To return to the normal data foundation view display, click anywhere in the view.

7.4.9.1.4 Highlighting the original table of an alias

Use this command to highlight the original table in the data foundation of an alias table.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header of the alias table and select *Highlight Original Table*.

   The original and alias tables are highlighted. All other tables are grayed.

   > **i Note**
   >
   > The original table may be off the visible area of the data foundation view. You can quickly check for a hidden table by clicking the *Fit to Window* icon at the bottom of the data foundation view . To undo the *Fit to Window*, click the *Reset Zoom* icon .

3. To return to the normal data foundation view display, click anywhere in the view.

7.4.9.2 Derived tables

A derived table is a virtual table in the data foundation that combines other tables using calculations and functions. You can create objects in the business layer on a derived table in the same way that you do for a standard table. Use derived tables in the following situations:

- To create a table with columns from other tables. The column definitions can include complex calculations and functions.
- To create a single table that combines two or more tables (called merging tables).
- To create a table that contains a selection of columns from different tables.

> **i Note**
>
> Derived tables for SAP HANA views that contain an SAP HANA optional input parameter are not supported. During table creation, you receive an error message that advises you to edit the table expression so that it doesn't contain an optional prompt.

Related Information

Inserting a derived table based on a data foundation table [page 168]
Merging tables [page 169]
Inserting and editing a derived table [page 169]

7.4.9.2.1 Inserting a derived table based on a data foundation table

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header of the table to be the basis for the derived table, and select *Insert* > *Derived Table*.
3. Enter a name for the derived table that is unique in the data foundation, and click *OK*.

A derived table with the new name and all the columns from the original table is inserted into the data foundation.

Edit the derived table to make the desired modifications.

Related Information

Inserting and editing a derived table [page 169]
Derived tables [page 168]

7.4.9.2.2 Merging tables

Merging tables inserts a derived table into the data foundation that consists of the combined columns from two or more tables linked by joins. Federated tables cannot be merged.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. In the data foundation view select the tables you want to merge:

Option	Command
To select a table and all tables related to it by joins	Right-click the table and select *Select Related Tables*.
To select tables manually	Click the table headers while holding down the `CTRL` key.

3. Right-click the selection of tables and select *Merge*.
4. Enter a name for the table that is unique within the data foundation and click *OK*.

   The merged table is inserted as a derived table. The new table is joined to any tables that the original tables were joined to.
5. Select if you want to delete the original tables.

   The original tables become obsolete and you have the choice of deleting them. If you choose to keep the original tables, the joins linking those tables are deleted, however the tables stay in the data foundation.

In a multisource-enabled data foundation, a derived table that is the result of a merge creates expressions using SQL-92 standard syntax. To use database-specific SQL, you must edit the derived table and explicitly select database-specific syntax.

To edit the merged table, right-click the table header and select *Edit*.

Related Information

Inserting and editing a derived table [page 169]
Derived tables [page 168]
SQL expressions in multisource-enabled data foundations [page 126]

7.4.9.2.3 Inserting and editing a derived table

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Do one of the following:

Option	Command
To edit an existing derived table	Right-click the table header in the data foundation view and select *Edit*.

Option	Command
To insert and edit a derived table	Select the *Insert Derived Table* command from the *Insert* menu in the data foundation view.

3. Give the derived table a unique name within the data foundation.
4. In a multisource-enabled data foundation, if you want to include database-specific functions in the definition of the derived table, select the *Database-specific* option.

   For more information about SQL syntax in multisource-enabled data foundations, see the related topic.
5. Enter or edit the SQL expression for the derived table in *Expression*.

   > **i Note**
   >
   > To build an expression for the first time, you can use the *SQL Builder*. The SQL Builder works like the query panel. You drag and drop tables and columns to be included in the derived table. The SQL expression is generated automatically.
   >
   > For more information on using the SQL Expression Editor, see the related topic.
6. Click *Validate* to check the validity of the SQL expression.
7. Click *OK*.
8. Link the derived table to other tables in the data foundation by inserting the appropriate joins.
9. To check the results of the derived table, right-click the table and select *Show Table Values*.
10. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

The SQL/MDX expression editor [page 421]
Inserting and editing a join [page 176]
Using analytical functions in a derived table definition [page 262]

7.5 Managing tables

This section describes how to manage the tables that you have inserted in the data foundation.

Related Information

Refreshing a data foundation [page 171]
Filtering the data foundation [page 173]
Searching for tables and columns in the data foundation [page 173]

Showing local dependencies in the data foundation [page 174]

7.5.1 Refreshing a data foundation

Refreshing the structure of a data foundation compares the existing tables in the data foundation with those in the data source and proposes updates to the data foundation tables.

The refresh is done through a wizard that allows you to select the tables to refresh. The wizard detects the following changes and lists them each in their own dialog box (in each case, you select which of the proposed changes to make in the data foundation):

- Tables in the data foundation that were deleted in the database. The wizard proposes to delete these tables and any related joins from the data foundation.
- Columns added in the database. The wizard proposes to update each corresponding table in the data foundation to add these columns.
- Columns in the data foundation tables that are not present in the database. The wizard proposes the best matching columns from the database table. You can choose to delete your data foundation table column, or replace the column with one from the proposed list of database columns.
- Column data types changed in the database. The wizard proposes to update the data type of each column in the data foundation that is different from the database column type.
- For data foundations based on SAP HANA, variables in the data source that are added, deleted, or modified.

The wizard lists your selected changes in a summary dialog box and asks for confirmation before continuing with the refresh.

After refreshing the structure, save the data foundation by clicking the *Save* icon in the main tool bar.

> **i Note**
>
> For data foundations based on SAP BW connections, you can detect new tables and new joins that have been created in the data source, and insert them into the data foundation using the *Synchronize Tables* command.

Related Information

Refreshing selected tables in the data foundation [page 171]
Synchronizing tables [page 172]

7.5.1.1 Refreshing selected tables in the data foundation

Rather than refreshing the whole data foundation, you can now select the tables that you want to refresh. This reduces the time required for the refresh process.

1. To start the *Refresh Structure* wizard: - Select *Refresh Structure* in the *Actions* main menu bar item or the *Detect* menu bar item in the *Data Foundation* view. Or select the tables in the data foundation that you want to refresh. Right-click a selected table and select *Refresh Structure* in the contextual menu
2. Right-click a selected table, or use the *Detect* menu bar item and select *Refresh Structure*.
3. In the *Refresh Data Foundation Structure* panel, select the tables that you want to refresh, then click *Next*.
4. If there are any missing tables, use the *Missing Tables* pane to select the tables that you want to use in place of the missing tables, then click *Next*.
5. If there are any missing columns, use the *Missing Columns* pane to select the columns that you want to use in place of the missing columns, then click *Next*.
6. If any columns have been added to the database, use the *Added Columns* pane to select the columns that you want to add to your data foundation, then click *Next*.
7. If any columns have been modified in the database, use the *Modified Columns* pane to select the columns that you want to update in your data foundation, then click *Next*.
8. The *Summary of Changes* pane provides a summary of the changes that will be applied when you refresh the data foundation. Click *Finish* to apply the changes and refresh the data foundation based.

7.5.1.2 Synchronizing tables

Before synchronizing tables, refresh the data foundation structure to make sure all existing data foundation tables are updated with any new columns in the data source.

Synchronizing tables applies only to multisource-enabled data foundations based on SAP BW data sources.

Synchronizing tables searches the data source for new tables (using the SAP BW strategy), and inserts the new tables and joins into the data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Click the *Connections* pane.
3. Right-click the connection in the *Connections* pane and select *Synchronize Tables*.
4. You are prompted to optionally detect new joins.
5. Save the data foundation by clicking the *Save* icon in the main tool bar.

   The insertion can be undone using the Edit > Undo command on the main menu.

Refresh the business layer with new objects in the data source using the *Insert Candidate Objects* command.

Related Information

Refreshing a data foundation [page 171]
Inserting candidate objects [page 255]
Refreshing universes based on SAP BW [page 46]

7.5.2 Filtering the data foundation

To filter the list of tables displayed in the Data Foundation tree list, click the Filtering Options button in the Data Foundation side panel. The filter mechanism is case sensitive.

This opens a box where you can select to filter the tables by their name or by other parameters such as the tables type, if they have auto-joins, filters, calculated columns or input columns.

7.5.3 Searching for tables and columns in the data foundation

1. To open the search panel, in the data foundation view, click the *Show/Hide Search Panel* icon .
2. By default, the search looks for tables. To look for columns, click the icon in the filter text box.
3. You can limit your search in several ways:
   - Enter text to search for in the filter text box.
   - Select connections, table types, column types, families, and contexts in the respective lists.

   The tables that match the search criteria are highlighted in the data foundation view.
4. To modify the view to display only the matching tables, click the *Search Options* icon at the top of the search panel, and select *Auto arrange search results*.

   > → Tip
   >
   > You can also use the *Center on Selection* command to change the zoom on the data foundation display so that all the tables in a selection are visible in the display window.
5. Click *Reset* to clear the search criteria and start a new search.

Certain operations on tables are not possible when the Search Panel is active, for example inserting alias and derived tables, detecting joins, or checking integrity. The data foundation commands that are not available when using the Search Panel are grayed. To use these commands, close the Search Panel by clicking the *Show/Hide Search Panel* icon .

Related Information

Centering the view on a selection [page 139]

7.5.4 Showing local dependencies in the data foundation

Use the *Show Local Dependencies* command when you plan to change tables and columns in the data foundation. The command will find the business layers and their objects that depend on the table or column.

1. Right-click the table header or column name in the data foundation view and select *Show Local Dependencies*.

   You can select several tables and/or columns by holding down the `CTRL` key.

   The business layers that depend on the selected tables and columns are listed.
2. Select the business layer for which you want to see the dependent objects.
   A dialog box lists the data foundation tables and columns, and the business layer objects that depend on them.
3. To edit a business object, double-click the object name in the *Business layers and objects* box. The Business Layer opens with the focus on the selected object.

Related Information

Resource dependencies [page 395]

7.6 Joins

A join is a condition that links tables in the data foundation, and restricts the data returned when the two tables are queried.

A join is a condition that links the data in separate but related tables. The tables usually have a parent-child relationship. If a query does not contain a join, the database returns a result set that contains all possible combinations of the rows in the query tables. Such a result set is known as a Cartesian product and is rarely useful.

For example, the Cartesian product of a query referencing two tables with 100 and 50 rows respectively has 5000 rows. In large databases or queries involving many tables, Cartesian products quickly become unmanageable. In the information design tool, joins are represented as lines linking tables in a schema.

You can modify a join by editing its properties. Use cardinality to describe how many rows in one table match rows in another table, or contexts to collect joins to provide a valid query path. You can also optimize joins by resolving join path problems related to loops, chasm traps, and fan traps.

Related Information

Using parameters, derived tables, and lists of values in the data foundation [page 184]
Optimizing [page 185]
Contexts [page 181]

Cardinality [page 179]
Joins [page 175]

7.6.1 Joins

A join is a condition that links tables in the data foundation. A join restricts the data returned when the two tables are queried.

Tables that are joined usually have a parent-child relationship. If tables are not joined, then a query run on the two tables can return a result set that contains all possible row combinations. Such a result set is known as a Cartesian product and is rarely useful.

Joins are defined by linking a column in one table to a column in a second table. You can insert joins into the data foundation, or detect joins automatically.

The following sections describe the types of joins you can create.

Equi-joins

An equi-join is the join type created by default between two tables. An equi-join links tables based on the equality between the values in the column of one table and the values in the column of a second table. In a normalized database, the columns used in an equi-join are often the primary key from one table and the foreign key in the other.

Self-restricting joins

A self-restricting join is when the two tables are the same. Self-restricting joins are used to define column filters. For more information on column filters, see the related topic.

Theta joins

When there is no obvious direct column to column relationship between two tables, you can use a theta join. A theta join links tables based on a relationship other than equality between two columns. It is used to link a value to a range of values. For example, an order date in one table is joined to a date between start date and end date in a second table.

Outer joins

An outer join can be used to link tables when one table contains rows that do not have a match in the common column of the other table. Unlike an equi-join, an outer join returns all rows, regardless of whether or not there is a matching value in the joined table.

A left outer join returns all rows in the first (or left-hand side) table, even if they have no match in the second table.

A right outer join returns all rows in the second (or right-hand side) table, even if they have no match in the first table.

A full outer join returns all rows from both tables, with null values when there is no match.

Shortcut joins

A shortcut join is a join that provides an alternative path between two tables. Shortcut joins improve the performance of a query by not taking into account intermediate tables, and so shortening a normally longer join path.

Shortcut joins are not taken into account to define contexts, but only to decrease the number of joins whenever possible.

Related Information

Inserting and editing a join [page 176]
Detecting joins [page 178]
Inserting a column filter [page 179]
Contexts [page 181]

7.6.1.1 Inserting and editing a join

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Do one of the following:

Option	Command
To edit an existing join	Right-click the join line in the data foundation view and select *Edit Join*.
To insert and edit a join	Select the *Insert Join* command from the *Insert* menu in the data foundation view.

3. To define the first side of the join, select the table from the list in *Table 1*, and then select the column name.

   You can enter a filter pattern to filter the list of columns in Table 1. Only the column names that contain the pattern are listed.
4. To define the second side of the join, select the table from the list in *Table 2*, and then select the column name.

   You can enter a filter pattern to filter the list of columns in Table 2. Only the column names that contain the pattern are listed.
5. Select the join operator:

   Between Table 1 and Table 2, a list of join operators lets you select how to compare the values of the columns in the join.

   The default operator creates an equi-join (=). The other operators are for joins that are not based on equality between column values (>, >=, <, <=, !=).

   To create a theta join using the `BETWEEN` operator, select the = operator. While holding down the `CTRL` key, select a second column in *Table 2*.

   For more information about the types of joins possible, see the related topic about joins.
6. To create a shortcut join, select the *Shortcut join* option.

   A shortcut join is a join that provides an alternative path between two tables. Shortcut joins improve the performance of a query by not taking into account intermediate tables, and so shortening a normally longer join path.
7. To create an outer join, select the *Outer join* options.

   An outer join allows rows to be returned even when there is no matching row in the joined table. Select the options as follows:

   To create a left outer join, select the *Outer join* option below Table 1. This join will return all rows in Table 1, even if they have no match in Table 2.

   To create a right outer join, select the *Outer join* option below Table 2. This join will return all rows in Table 2, even if they have no match in Table 1.

   To create a full outer join, select the *Outer join* option below both tables. This join returns all rows from both tables, with null values when there is no match.
8. Select the cardinality for the join from the *Cardinality* list. You can also click the *Detect* button to automatically detect the cardinality defined for the join in the database.

   For more information about cardinality, see the related topic.
9. Optionally, edit and validate the join expression.

   Based on the columns and operators you select, an SQL expression is automatically generated to define the join. You can type a custom expression for the join. To get help editing the join expression, click the *SQL Assistant* icon .

   > **i Note**
   >
   > If you change the table or column names when editing the join expression, the changes are not immediately reflected in the *Table 1* and *Table 2* lists. The changes are reflected in the lists when you save and re-edit the join.
10. Click *OK* to save the join.

11. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Joins [page 175]
Cardinality [page 179]
Inserting a column filter [page 179]
Preferences for inserting tables and joins [page 414]

7.6.1.2 Detecting joins

Join detection looks at the data foundation tables and proposes appropriate joins. The following methods are used:

- Join detection based on column name. This method looks for identical column names in different tables. It also checks that the data type of the two columns is the same. If more than one column matches between two tables, joins are proposed for each column. Joins between a table and its alias are not proposed.
- Join detection based on database keys. This method looks for relationships defined in the database between primary keys and foreign keys.
- For data foundations with an SAP BW connection, join detection is based on the joins in the database schema referenced in the connection.

Before you begin, set or detect keys in the data foundation if you want to use join detection based on database keys.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Select *Detect Joins* from the *Detect* menu in the data foundation view.
3. Select the join detection method.

   For a multisource-enabled data foundation, select a method for each connection. This method is used to detect joins between tables referenced by the connection. You can also detect joins between tables from different connections. In this case, the method used is by column name.
4. From the detected joins proposed in the dialog box, select the joins to be inserted into the data foundation.

   To automatically detect cardinalities for the selected joins, select the *Detect cardinalities* option.
5. Save the data foundation by clicking the *Save* icon in the main tool bar.

You can set an application preference to detect and insert joins automatically every time a table is inserted into the data foundation. See the related topic on setting table and join detection options.

Related Information

Setting and detecting table keys [page 157]

Joins [page 175]
Cardinality [page 179]
Setting table and join detection options [page 412]

7.6.1.3 Inserting a column filter

A column filter, also called a self-restricting join, lets you restrict the values returned whenever the table is used in a query.

The following rules apply to column filters:

- Only one filter per column is allowed.
- You can insert a filter on a calculated column.
- The expression can contain sub-queries.
- The following @Functions are allowed in the expression: `@Prompt` and `@Variable`.
- If you insert a filter into a standard table, and then create an alias from the table, the filter is not inserted into the alias table.
- If you insert a filter into an alias tables, the filter is not automatically inserted into the original standard table.
- When you merge tables that include filters, the filters are not included in the resulting derived table.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the column name in the table in the data foundation view and select *Insert Filter*.

   A self-join on the column is proposed in the *Edit Join* dialog box. For example, if you insert a filter on the **`age`** column in the **`Customer`** table, the following self-join is proposed:

   `"Customer"."age"="Customer"."age"`
3. Edit the second part of the self join (the expression to the right of the equal sign) to filter the column values. To get help editing the join expression, click the *SQL Assistant* icon .

Related Information

Joins [page 175]

7.6.2 Cardinality

Cardinality further describes how tables are joined by stating how many rows in one table match rows in another table. Cardinalities are needed when detecting aliases and contexts to resolve loops in the data foundation.

Cardinality of a table is expressed as a pair of numbers: the number of rows in one table that match the number of rows in the joined table. The number of rows that match can be none (0), one (1), or many (n) for each table.

For example, the tables **Customer** and **Reservations** are linked by a join.

- For each customer, there can be one or more reservations, so the cardinality of the **Customer** table is one-to-many, or 1,n.
- For each reservation, there can be one and only one customer, so the cardinality of the **Reservations** table is one-to-one, or 1,1.

Cardinality of the join is also expressed as a pair of numbers: the maximum number of rows in the second table that match one row in the first table, and the maximum number of rows in the first table that match one row in the second table.

In the example, the cardinality of the **Customer-Reservations** join is n,1, because the maximum rows that can match a row in **Customer** is n, and the maximum rows that can match a row in **Reservations** is 1.

Cardinalities can be detected for joins automatically and stored in the data foundation. The detection method first detects primary and foreign keys. Cardinalities are set according to the key status of the column in the two tables as follows:

First table column	Second table column	Cardinality
Primary key	Foreign key	1, n
Foreign key	Primary key	n,1

If no keys are detected, the cardinality is set using table row counts.

Related Information

Detecting and setting cardinalities [page 180]
Joins [page 175]
Table keys [page 157]
Table row counts [page 158]

7.6.2.1 Detecting and setting cardinalities

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Select *Detect Cardinalities* in the *Detect* menu.

   The *Detect Cardinalities* dialog box lists the current cardinalities for all joins in the data foundation.
3. Select the joins for which you want to detect cardinalities and click *Detect Cardinalities*.
4. To set the cardinality of a join manually, select the cardinality from the list in the *Cardinality* column for the join.
5. Click *Finish* to save the changes.

You can set an application preference to detect and insert cardinality automatically every time a join is inserted into the data foundation. See the related topic on setting table and join detection options.

Related Information

Cardinality [page 179]
Setting table and join detection options [page 412]

7.6.3 Contexts

A context is a collection of joins which provide a valid query path. The most common use of contexts is to resolve loops in the data foundation when the loop cannot be resolved by creating an alias table. Another use of contexts is when multiple fact tables share a dimension table. In this case, a context is created for each fact table.

In the information design tool, a context resolves a loop by identifying a set of joins that define one specific join path through the tables in the loop. The user is prompted for the context to use at query time. The context ensures that joins are not included from different paths within the same SQL query.

A context is defined by setting states for the joins involved in the ambiguity. In a context, a join has one of three states:

- Included joins: In a part of the schema that is ambiguous, the context solves the loop by defining a path with the included joins.
- Excluded joins: In a part of the schema that is ambiguous, the excluded joins define the path that context will never take.
- Neutral joins are in a part of the schema that is not ambiguous, and are always included in the query path of the context. Any join that is not explicitly included or excluded is neutral.

When a new join or table is inserted into the data foundation, it is neutral by default. Contexts will not need to be updated unless the new table or join is explicitly involved. You can change the default so that added joins are automatically excluded or included. You change this default behavior in the application preferences for the *Data Foundation Editor*. You can also choose to use the new default behavior when adding contexts.

You can insert contexts into the data foundation manually, or by detecting contexts. The detect command analyzes the join paths and proposes contexts to resolve any loops that cannot be resolved by alias tables.

Related Information

Detecting contexts [page 182]
Inserting and editing contexts [page 182]
Selecting contexts (joins) [page 183]
Resolving loops [page 187]
Setting default join states for contexts [page 413]

7.6.3.1 Detecting contexts

Before detecting contexts, you need to set cardinalities and detect aliases. See the related topic about resolving loops for the prerequisite tasks.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. From the *Aliases and Contexts* pane, select the *Detect Contexts* icon .

   The command analyzes the join paths and proposes contexts to resolve any loops that cannot be resolved by alias tables.

   > **i Note**
   >
   > You may get a message that the loop can be resolved using aliases. See the related topic on resolving loops.

3. In the *Detect Contexts* dialog box, select the contexts you want to insert.

   To see the context highlighted in the data foundation view, click the proposed context name. A join that is included in the context is shown by an included icon . A join that is excluded is shown by the excluded icon .
4. Click *OK* to insert the selected contexts into the data foundation.

   The new contexts are listed in the *Aliases and Contexts* pane in the *Contexts* folder.
5. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Contexts [page 181]
Resolving loops [page 187]

7.6.3.2 Inserting and editing contexts

The table listing joins has been improved: it is by default in edit mode, the status of the join is indicated by an icon in the first column (Included, Excluded, Ignored), and you can filter joins using a text filter field. You can sort and filter the joins by status.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Select the *Aliases and Contexts* pane.
3. Do one of the following:

Option	Command
To edit an existing context	Select the context in the *Contexts* folder.
To insert and edit a context	Select the *Insert Context* icon .

The properties of the context are displayed in the *Context Properties* pane:

- Context name
- All joins in the data foundation
- The state of the join in this context: whether the join is included, excluded, or ignored

4. Edit the context name in *Name*.
5. To include or exclude a join, or to ignore it, click the join expression in *Join Expression* list. The state toggles each time you double-click.

   You can also toggle the state by double-clicking the join line in the data foundation view.

   A join that is included in the context is shown by an included icon. A join that is ignored is shown by the ignored icon.
6. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Contexts [page 181]

7.6.3.3 Selecting contexts (joins)

How you select contexts in the information design tool has been enhanced.

You can perform the following actions on contexts in both the data foundation schema and the joins table:

Selecting joins/contexts in the schema or the joins table

To	Do this
Select a join	Click the join
Add a join to a multiple selection of joins	Ctrl-click the join
Remove a join from a multiple selection	Ctrl-click the join
Switch the status of a single join (Included-Excluded-Ignored)	Double-click the join

To	Do this
Handle the way the selected join(s) is/are handled in the current context	Right-click the join and select from the contextual menu: • *Include in this context* (Include the selected join(s) in the context) • *Exclude from this context* (Exclude the selected join(s) from the context) • *Ignore in this context* (Ignore the selected join(s) in this context) • *Include in all contexts* (Include the selected join(s) in all contexts) • *Exclude from all contexts* (Exclude the selected join(s) from all contexts) • *Ignore in all contexts* (Ignore the selected join(s) in all contexts) • *Include in a specific context* (Include the selected join(s) in a selected context) • *Exclude from a specific context* (Exclude the selected join(s) from a selected context) • *Ignore in a specific context* (Ignore the selected join(s) in a selected context)
Select several joins in the joins table	Use shift-click for continuous multi-selection

7.6.4 Using parameters, derived tables, and lists of values in the data foundation

A parameter is a variable in the universe that requires a value at query time. Parameters are often defined to prompt the user to supply a value, and in this case are referred to as prompts.

A list of values is a collection of data values that can be associated with an object in the universe, allowing the user to choose values for a prompt. You can include calculated columns in a list of values.

You can insert parameters and lists of values into the data foundation. The parameters can also include a constant, or a formula that sets a dynamic default value to a prompt at run time. All parameters and lists of values are inherited by any business layer built on the data foundation, but cannot be modified in the business layer.

See Related Links for more information.

Inserting a parameter or list of values in the data foundation

To insert a parameter or list of values in a data foundation, go to the *Parameters and Lists of Values* tab in the *Data Foundation Editor*. From there the procedure is the same as inserting parameters and lists of values into a

business layer. To create a parameter that uses a formula to set a dynamic default value for a prompt, you use a formula editor accesible from the *Default values* section of the *Data Foundation Editor*.

Inserting a derived table in a list of values

To use a derived table in the definition of a list of values based on custom SQL, you must use the @DerivedTable function.

To insert it, go to the *Parameters and Lists of Values* tab in the *Data Foundation Editor*. Select or create a list of values based on custom SQL, and in the *List of Values* editor, click the *Edit SQL* button. Click in the appropriate place in the *SQL Expression Editor* where you want to add the derived table, then double-click on the drived table in the list of available tables. The name of the derived table, preceded by @DerivedTable is added to the SQL expression.

Inserting a calculated column in a list of values

To use a calculated column in the definition of a list of values based on custom SQL, you must use the @CalculatedColumn function. In the *SQL Expression Editor*, either drag and drop the column in the appropriate place or double-click the calculated column. The name of the table and column, preceded by @CalculatedColumn is added to the SQL expression.

Sample Code

```
SELECT @CalculatedColumn ( product_promotion_facts.CorrectedDuration )
FROM product_promotion_facts
```

Related Information

Inserting and editing a parameter [page 239]
Parameters [page 238]

7.7 Optimizing

This chapter describes the types of problems that can arise as you create joins between the tables in your schema.

This chapter explains how you can detect and resolve join problems to ensure that the join paths taken by queries run on the universe return correct results. You must resolve the join problems before building the universe.

A join path is a series of joins that a query can use to access data in the tables linked by the joins.

Join path problems can arise from the limited way that lookup and fact tables are related in a relational database. The three major join path problems that you encounter when designing a schema are the following:

- loops
- chasm traps
- fan traps

You can solve all these problems by creating aliases (a copy of a base table), contexts (a defined join path), and using features available in the universe design tool to separate queries on measures or contexts.

This section briefly defines lookup and fact tables, and describes the types of join path problems that you can encounter using these tables. It explains how you can use aliases, contexts, and other universe design tool features to resolve join path problems in your universe schema.

In the universe design tool, you typically create joins between lookup tables and fact tables.

7.7.1 Loops

Loops occur when multiple paths join tables. The rows that are returned from the query are the intersection of the results for each path, so fewer rows are returned than expected.

An alias table breaks a loop by using the same table twice in the query, once for each path. That way, the rows returned in the query are the union of the results for each path.

A cycle is a loop that occurs when the tables joined by the loop all have a cardinality of (1,n). In this case, the *Detect Aliases* command cannot determine which table to create an alias for.

When loops cannot be resolved with an alias table, contexts are used. You use contexts to resolve the ambiguity by explicitly directing the query as to which join path to use.

Related Information

Using parameters, derived tables, and lists of values in the data foundation [page 184]
Contexts [page 181]
Cardinality [page 179]
Joins [page 174]
Resolving loops [page 187]
Resolving chasm traps [page 189]
Optimizing [page 185]
Resolving fan traps [page 190]

7.7.1.1 Effects on queries

A loop is a join path problem which causes the query to return fewer records than expected. The loop exists when joins between tables form a closed path.

Related Information

Visually identifying loops [page 187]
Resolving loops [page 187]

7.7.1.2 Visually identifying loops

Use the following guidelines to help you analyze your schema to determine whether an alias or context is appropriate for resolving loops.

Visually identifying loops can be useful to understand your schema, but you should use *Detect Aliases* and *Detect Contexts* to formally identify and resolve loops.

Loop contains	Can be resolved by
Only one lookup table	Alias
A look up table that receives only "one" ends of joins	Alias
Two or more fact tables	Context

Related Information

Loops [page 186]
Resolving loops [page 187]
Contexts [page 181]

7.7.1.3 Resolving loops

In this section, you see how to solve a loop in a data foundation design and avoid incorrect query results.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Detect and insert all the joins in the data foundation.
3. Detect or set the cardinalities for the joins.
4. Make sure there are no cycles in the data foundation. You can do this by examining the cardinalities of any loops, or by refreshing the loop resolution status (see the following steps in this procedure).
5. Make sure none of the joins have a cardinality of (n,n). Aliases and contexts cannot be detected. Set the cardinality manually for (n,n) joins.
6. From the *Aliases and Contexts* pane, detect aliases.

7. Detect contexts. It is recommended to insert all proposed contexts.
8. In the *Loops* box, click the *Visualize Loops* icon to check if all the loops have been resolved.

   Possible loops are listed in *Loops*. To check if the loops are resolved, click the *Refresh Loop Resolution Status* icon .

   A message appears suggesting what to do about unresolved loops.

   A loop is resolved when you see a green check mark next to the loop name.

Related Information

Detecting joins [page 178]
Detecting and setting cardinalities [page 180]
Detecting alias tables [page 166]
Detecting contexts [page 182]

7.7.2 Chasm traps

A chasm trap is a common problem in relational database schemas in which a join path returns more data than expected.

A chasm trap is a type of join between at least three tables when too "many-to-one" joins converge to the same table. The inclusion of such joins can cause incorrect results to be returned, especially when a query includes measures based on both leaf tables. The incorrect results are referred to as Cartesian products.

A typical example of a chasm trap occurs when you're using a dimension with two different facts tables: sales and reservations.

Related Information

Using parameters, derived tables, and lists of values in the data foundation [page 184]
Contexts [page 181]
Cardinality [page 179]
Joins [page 174]
Resolving loops [page 187]
Resolving chasm traps [page 189]
Optimizing [page 185]
Resolving fan traps [page 190]

7.7.2.1 Resolving chasm traps

In this section, you see how to solve a chasm trap in a data foundation design and avoid incorrect query results.

You can find chasm traps by using *Detect Contexts* to detect and propose candidate contexts, and then examining the table where any two contexts diverge. This point where two contexts intersect is the source of a chasm trap. If you have two fact tables with many to one joins converging to a single lookup table, then you have a potential chasm trap.

The query engine solves the chasm trap by splitting a query into at least two queries. For measures defined in the business layer having SQL aggregation functions, the query split into two queries, and the results of the two queries are merged in a single result.

> **i Note**
>
> To allow the split of a query into two or more queries, check that the business layer *Multiple SQL Statements for each measure* option is activated.

1. Select *Tools*. Click on *Detect Contexts*.
   The Candidate Contexts box appears.
2. Select a proposed context in the Candidate Contexts list box. Click on *Add* to add it to the Accept Contexts list box.
3. Repeat for other contexts listed in the Contexts pane of the List View bar.
4. Click on *File* and select *Parameters*.
   The Universe Parameters dialog box appears.
5. Click the *SQL* tab.
   The SQL page appears.
6. Check *Multiple SQL statements for each context*.
7. Click *OK*.

When you run queries on the tables in the chasm trap, the query is separated for measures and dimensions defined on the affected tables.

Related Information

Resolving loops [page 187]
Resolving fan traps [page 190]
Joins [page 175]
Cardinality [page 179]
Contexts [page 181]
Using parameters, derived tables, and lists of values in the data foundation [page 184]

7.7.3 Fan traps

A fan trap is a less common problem than chasm traps in a relational database schema. It has the same effect of returning more data than expected.

A fan trap is a type of join between at least three tables when a "one-to-many" join links a table which is also linked to another table by another "one-to-many" join. The fanning out effect of "one-to-many" joins can cause incorrect results to be returned when a query includes objects based on both tables. This type of schema is frequently used to define "one-to-many" joins.

The inclusion of such fanned-out "one-to-many" joins can cause incorrect results to be returned, especially when a query includes objects based on both tables. The incorrect results are referred to as Cartesian products.

You cannot automatically detect fan traps. You need to visually examine the direction of the cardinalities displayed in the table schema.

Related Information

Using parameters, derived tables, and lists of values in the data foundation [page 184]
Contexts [page 181]
Cardinality [page 179]
Joins [page 174]
Resolving loops [page 187]
Resolving chasm traps [page 189]
Optimizing [page 185]

7.7.3.1 Resolving fan traps

In this section, you see how to solve a fan trap in a data foundation design and avoid incorrect query results.

You cannot automatically detect fan traps. You need to visually examine the direction of the cardinalities displayed in the table schema.

If you have two tables that are referenced by measure objects and are joined in a series of many to one joins, then you may have a potential fan trap.

The query engine solves the fan trap by splitting a query into at least two queries. For the measures defined in the business layer having SQL aggregation functions, the query is split into two queries, and the result of the two queries are merged in a single result set. To allow the split of a query into two or more queries, check that the business layer *Multiple SQL Statements for each measure* option is activated.

Another solution for solving a fan trap is to follow these steps:

1. Identify the potential fan trap by analyzing the "one-to-many-to-one-to-many" join path relations in the schema.
2. Create an alias for the table that is producing the multiplied aggregation.

3. Create a one-to-one join between the original table and the alias table.
4. Build the object that is causing the aggregation on the alias tables.
5. Select *Tools* and click on *Detect Contexts*.
   The Candidate Contexts box appears. It proposes the candidate contexts for the join path for the base table and the new join path for the alias table. If you have used a one-to-one join between the alias and the base table, then you need to create the context manually.
6. Select a candidate context and click on *Add*.
7. Repeat for other candidate contexts.
8. Click *OK*.
   The contexts are created in the schema. You can view them in the Contexts pane when List Mode is active (to activate it, click on *View*, and on *List Mode*).
9. Click on *File*, and then on *Parameters*.
   The Parameters dialog appears.
10. Click the *SQL tab.SQL page*.
    The SQL page appears.
11. Check *Multiple SQL statements for each context*.
12. Click *OK*.
13. Run queries to test the fan trap solution.

Related Information

Resolving chasm traps [page 189]
Resolving loops [page 187]
Joins [page 175]
Cardinality [page 179]
Contexts [page 181]
Using parameters, derived tables, and lists of values in the data foundation [page 184]

7.8 Stored procedures in the data foundation

A stored procedure is a function that is stored in the database and is exposed as a table in the Connections pane of the *Data Foundation Editor*. Use the stored procedures to create a specific business object for your requirements.

To view the available database stored procedures in your connection, in the filter icon of the *Connection* pane, select *Stored Procedure*. This option is only available if the target driver supports stored procedures.

Note the following:

- When you create a data foundation based on a stored procedure, you can only use stored procedures in that data foundation. You cannot have a mix of stored procedure tables and standard database tables.

- When inserted in the data foundation, the stored procedure can be represented by one or more tables (depending on the stored procedure).
- In the Query Panel, you can't view or edit scripts.
- For business objects built on stored procedures, you can't use the business objects in filters or sorts.
- You can't join a stored procedure table with another stored procedure table.
- You can't create alias tables or derived tables based on stored procedures tables.
- The table(s) representing the stored procedure can contain variables that must be filled with static values or prompts when the stored procedure is inserted in the data foundation.

When you have inserted a stored procedures table in the data foundation, you can right-click on the table to edit it. You have the following options available:

Option	Description
Edit	Change the name of a stored procedure and its columns.
Edit Input Parameter	Edit the input parameter(s) for a stored procedure. This option is not available if the stored procedure does not have any input parameters.
Change qualifier/owner	You can change the qualifier or owner of a stored procedure.
Show procedure values	Execute and show the results of a stored procedure.
Select related procedures	When a stored procedure returns multiple results, show the related procedures.
Highlight related procedures	When a stored procedure returns multiple results, you can highlight the related procedures.
Center on selection	Lets you temporarily change the zoom on the data foundation display so that all the tables in a selection are visible in the display window.
Show in connection	This works the same way as for the Show Procedure table in the connection panel.
Show local dependencies	Show the business layers and objects that might be impacted by changes..
Delete	Delete the selected stored procedure.

> **i Note**
>
> Stored procedures are not supported for multi-source universes or linked universes.

Related Information

Inserting a Stored Procedure [page 193]

Data Foundation Editor preferences [page 409]

7.8.1 Inserting a Stored Procedure

Stored procedures are inserted in the *Connections* pane as a table as follows:

1. In the editor, open the *Connections* tab.
2. Drag and drop a stored procedure table into the editor or click the *Insert* icon and select the *Insert Stored Procedures* option, select a stored procedure table from the list of available tables.

   The stored procedures table is represented in the editor pane as one or more tables. If the stored procedure contains variables, you must fill in the values with static values or prompts.
3. Drag and drop other stored procedures tables into the editor as required. You cannot mix table types. All the tables must be stored procedures tables.

Related Information

Editing values for Stored Procedures Input Parameters [page 193]

7.8.2 Editing values for Stored Procedures Input Parameters

When a stored procedure contains a variable or input parameter (IN or INOUT), you must enter a value (a static value, or a prompt) for the parameter.

1. To enter a value for a stored procedure variable, double-click on the stored procedure table.

   The *Edit Input parameters* dialog displays.
2. Set values for the input parameters

7.8.3 Refreshing the structure of a data foundation based on stored procedures tables

You can run Refresh Structure on a data foundation that contains stored procedures tables. Select the stored procedure tables that you want to refresh. Note that:

- If the tool detects that new stored procedures parameters are present, you must provide values for the parameters and they will be added to the data foundation.
- If the tool detects changes in the names of columns, the new columns are added.
- When stored procedures are missing, the tool detects similar columns and proposes a list of suitable columns. You can delete the missing stored procedures from the data foundation or select a column to

replace your original column. The list of proposed columns is updated accordingly, and you cannot reselect the same column for other missing columns.

- When the tool detects changes in multiple result sets, the structure of the result set is updated according to its index position in the stored procedure table in the data foundation.

7.8.3.1 Missing stored procedures

Select the stored procedures that you want to refresh. The wizard will verify that the stored procedures tables are not missing from the database. You can select all of the tables, or individual tables. The selected tables are highlighted in the data foundation.

7.8.3.2 Enter the input parameters

When refreshing the structure of stored procedures, use this page to enter input parameters that are required.

1. Select a stored procedure input parameter.
2. Enter a value for the input parameter.
3. Select the next input parameter(s) and enter the relevant value(s).
4. Click *Next*.

7.8.3.3 Delete or replace missing columns

When columns are missing in the database, the wizard automatically proposes corresponding columns. You can choose a column or you can delete the column from the data foundation table.

1. Choose a clumn that is identified as missing.
2. Select Delete to remove the column from your data foundation table, or choose the proposed column.
3. Click *Next*.

7.8.3.4 Including added columns

When the wizard detects that new colulns have been added to the stored procedures table in the database, you can add them to your data foundation table.

1. To include a new column in your data foundation view of the stored procedure, select the column.
2. Click *Next*.

7.8.3.5 Selecting modified columns

This page shows columns that have been modified in the database for your data foundation stored procedure tables. By default, all modified columns are selected.

1. Select or unselect the columns for your stored procedures table.
2. Click *Next*.

7.8.3.6 Summary of changes

The wizard shows a summary of the changes that to be made to your data foundation stored procedures table. Click *Finish* to apply the changes.

7.8.3.7 Removed tables

These stored procedure tables that you selected for the refresh are missing from the database. You can select the stored procedures tables that you want to delete from the data foundation. By default all the missing tables are selected.

7.8.3.8 Variables have changed

The following stored procedures parameters were added, modified, or are missing from the database. Select the stored procedures parameters that you want to add, delete, or update in the data foundation. By default, all stored procedures parameters are selected.

7.8.4 Stored procedure security right

The connection's *Use connection for Stored Procedures* security right is now enforced at universe authoring time.

- This right is used to prevent users from creating or editing new stored procedure, or editing data foundation based on stored procedure.
- This right is not verified when a client tool runs the query on top of a universe based on stored procedure.

If you don't have this security right granted, the stored procedures of this connection are not visible to you in data foundation created on top of this connection. You can set this security right in the Central Management Console (CMC). See Admin guide or ask your admin

8 Working with the federation layer

8.1 What is the federation layer?

The federation layer is available only in multisource-enabled data foundations. It lets you create federated tables that can include data from any of the data source connections defined in the data foundation. Federated tables can be inserted into the data foundation and used to define the schema on which the universe is built.

At design time, you use the federation layer to define a data flow composed of data source tables and federated tables. You define the data flow graphically without needing to write a lot of detailed SQL statements. You can specify complex transformations of data within the flow, and build multi-level data flows by using a federated table as input to another federated table.

The federation layer lets you maintain a coherent set of federated tables. From this set, you selectively insert federated tables into the data foundation.

Related Information

Building the federation data flow [page 196]

8.2 Building the federation data flow

Before you can build the federation data flow, you must have a multisource-enabled data foundation based on at least one valid connection.

Building the federation data flow is a matter of designing the input flow into a coherent set of federated tables that your applications will query.

In a top-down approach to the design, you start with the final schema of federated tables. You define these federated tables, then define the inputs and mapping for each column.

In a bottom-up approach, you start with the data source tables. You add federated tables from a data source table, and then modify the mappings.

The following procedure describes the steps to build a federation data flow. Links to more information on each step in the procedure can be found in the Related Topics.

1. You build the federation data flow using the *Data Foundation Editor*. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. To open the *Federation Data Flow* view, click *Federation Layer*.
3. Add a federated table either manually, or from a data source table.
4. Define the input to the federated table. An input table can be a data source table, or a federated table.
   - If you added a federated table manually, add one or more input tables and join them.
   - If you added a federated table from a data source, you can add other input tables and join them.
5. Map the columns from the input tables to the federated table.
6. You can further refine the mapping by editing the mapping formulas, adding pre-filters and post-filters, and specifying distinct rows for input tables.
7. Optionally, you can define additional mappings for the federated table.

   Mappings can be activated and deactivated. When more than one mapping is activated, the effective mapping is a union of all activated mappings.
8. Repeat the steps to add other federated tables to your data flow.
9. To save the data flow in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

Check the integrity of the federation layer often during the design of federated tables. After you have created and validated a federated table, you can insert it into the data foundation.

Related Information

The Data Foundation Editor [page 133]
Adding a federated table manually [page 199]
Adding a federated table from a data source [page 199]
Input tables and joins [page 200]
Mapping columns from the input table to columns of the federated table [page 204]
Editing a mapping formula [page 205]
Pre-filters and post-filters [page 207]
Adding a mapping [page 206]
Activating and deactivating mappings [page 206]
Checking integrity of the federation layer [page 209]
Inserting a federated table into the data foundation [page 210]
Distinct rows on input tables [page 207]

8.3 Federated tables

Federated tables are the tables that you create to present data in the correct format for your data foundation. A federated table may be the end result, or a table that contributes to a federated table at a higher level.

A federated table can be added in two ways:

- A federated table added manually is empty. You add the columns and define their properties.

- A federated table added from a data source by default contains the same columns as the data source table. The columns inherit the properties from the data source.

You edit a federated table to add and delete columns and change the column properties.

Federated table columns have the following properties:

Property	Description
Name	The default column name can be edited.
Data Type	The data type of the column can be selected from a list.
Input	Whether or not the column expects input. The input can be optional or mandatory. Input columns can be resolved in the federation layer by a join or filter. Input columns that are not resolved in the federation layer, are resolved in the data foundation. When showing table values for a federated table in the data foundation or federation layer, you are prompted to enter values for the input columns. A message indicates if a value is mandatory or optional. Select the input column and enter a value in *Assignment*.
Description	An optional description for the column.

You can also describe additional logic by adding distinct rows on input tables, pre-filters, joins between input tables and post-filters.

The logic you build into a mapping is applied in the following order:

1. Distinct rows
2. Pre-filters
3. Input table joins
4. Post-filters
5. Mapping formulas

A federated table can have more than one mapping. All mappings are activated by default. When more than one mapping is activated, the effective mapping is a union of all activated mappings.

Related Information

Input columns [page 160]
Adding a federated table manually [page 199]
Adding a federated table from a data source [page 199]
Editing a federated table [page 200]
Input tables and joins [page 200]
Distinct rows on input tables [page 207]
Pre-filters and post-filters [page 207]
Mappings in the federation layer [page 204]

8.3.1 Adding a federated table manually

Before you begin, you must have a multisource-enabled data foundation based on at least one valid connection.

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. From the *Federation Layer* pane, click *Add Federated Table*.
3. In the *Add Federated Table* dialog box, click the *Add Row* icon to add columns to the table.
4. For each column, edit the name, select a data type, and select whether or not input is needed.

   Optionally, you can enter a description of the column.
5. Optionally, you can add a description for the federated table.
6. To save the federated table, click *OK*.
7. To save the data flow in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

Next you need to define a mapping for the federated table just added.

Related Information

Building a data foundation [page 126]
Federated tables [page 197]
Adding a mapping [page 206]

8.3.2 Adding a federated table from a data source

Before you begin, you must have a multisource-enabled data foundation based on at least one valid connection.

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. Click *Federation Layer* to open the *Federation Layer* pane, then click *Connections*.
3. From the *Connections* pane, select the table from the data source and drag it to the *Federation Data Flow* pane.
   A federated table is automatically added with the same name and columns as the data souce table. A default mapping is added that maps one-to-one the data source columns to the federated table columns.
4. To save the data flow in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

Next you need to further define the input to the federated table by changing the mappings.

Related Information

Building a data foundation [page 126]

Federated tables [page 197]

8.3.3 Editing a federated table

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. In the *Federation Layer* pane, right-click the federated table that you want to edit and select *Edit*.
3. In the *Edit Federated Table* dialog box, you can perform the following operations:
   - Edit the table name.
   - Add or delete columns.
   - Change the order of columns.
   - Edit column names and descriptions.
   - Change the data type of a column.
   - Change whether or not a column needs input.
   - Edit the table description.
4. To save the updates to the table, click *OK*.
5. To save the data flow in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Federated tables [page 197]

8.4 Input tables and joins

Input tables define the input to a federated table. They can be data source tables or other federated tables.

The input tables are mapped to the federated table using mappings. In a mapping, a column in an input table is mapped to a column in the federated table. You can define the mapping formula so that one federated table column depends on one or more input table columns.

You can add multiple input tables to a mapping. In this case, you need to join the input tables.

To join input tables, first distinguish between core tables and non-core tables:

- Use a core table to choose the set of rows that will populate your federated table (the result set).When you set two or more tables as core, the result set is defined by the join of all the core tables. The core tables are joined using an inner join.
- Use non-core tables to extend the attributes of each row in the result set. A non-core table is joined using an outer join with core tables. If a row exists in the core table that doesn't match a non-core table row, it will return a row with nulls for the non-core columns.

The following restrictions apply to input tables and joins:

Direct joins between two non-core input tables are not allowed.

Cycles are not allowed (for example, if input table A is joined with B is joined with C, C cannot join to A.)

> **i Note**
>
> When a table is a core table, the table name in the input tables pane in the mapping is in bold.

Related Information

Adding input tables to a mapping [page 201]
Joining input tables [page 202]

8.4.1 Adding input tables to a mapping

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. In the *Federation Layer* pane, select the federated table.
   The mapping tabs appear in the Properties pane.
3. If the federated table has more than one mapping, select the appropriate mapping tab.
4. Do one of the following:

Option	Description
To add a data source table as an input table	Click *Connections*. Drag the data source table from the *Connections* pane to *Input Tables* in the mapping tab.
To add a federated table as an input table	Click *Federation Layer*. In the Properties tool bar, click the *Add* menu and select *Add Input Table*. You can also drag the federated table from the *Federation Layer* pane to *Input Tables* in the mapping tab.

5. To save the data flow in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

You can now map the columns, edit mapping formulas, and add filters for the new mapping.

Related Information

Mapping columns from the input table to columns of the federated table [page 204]
Editing a mapping formula [page 205]
Pre-filters and post-filters [page 207]

8.4.2 Joining input tables

Joining input tables applies when a mapping contains more than one input table.

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. In the *Federation Layer* pane, select the federated table.

   The mapping tabs appear in the *Properties* pane.
3. If the federated table has more than one mapping, select the appropriate mapping tab.
4. In the *Properties* pane, do one of the following:

Option	Description
Add a join visually	This option lets you add a single-column join. In the input tables pane, click the column in the first input table and drag it to a column in the second input table. The two input tables are now joined by a default inner join on the selected columns.
Add join with the join editor	This option lets you add single-column and multi-column joins and simple formulas. For more information on the restrictions for join expressions, see the related topic. In the Properties tool bar, click the *Add* menu and select *Add Join*. In the *Add Join* dialog box, select a column in the left table and a column in the right table. You can edit the SQL for the join expression and click *Validate* to validate the SQL expression. To save the join definition, click *OK*.

5. To select or unselect a table as a core table, right-click the table name in the input table pane and select *Core Table*.

   When a table is a core table, the table name is bolded. For more information about core tables, see the related topic about input tables.
6. To save the data flow in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

To edit a join, right-click the join line and select *Edit Join*.

Related Information

Input tables and joins [page 200]

8.4.3 Configuring meanings of input table joins using core tables

When you map multiple input tables to a federated table, you must distinguish between core tables and non-core tables.

- Use a core table to choose the set of rows that will populate your federated table (the result set). When you set two or more input tables as core, the result set is defined by the join of all the core tables.
- Use non-core tables to extend the attributes of each row in the result set.

Example: The effect of setting an input table as core or non-core

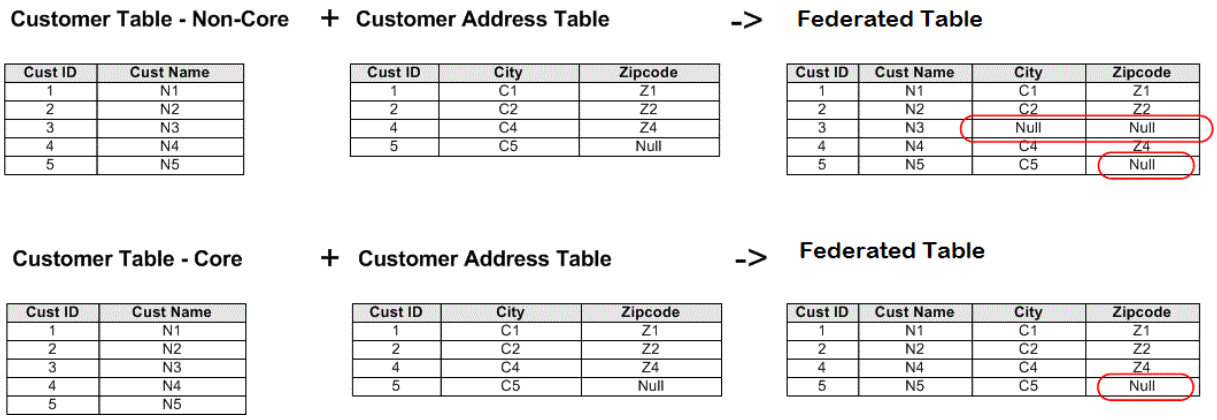
Suppose that you have two input tables: `Customers` and `Orders`.

Setting on the `Customers` table	Setting on the `Orders` table	Result of a join between the two tables
core	non-core	all customers, including those who did not purchase anything (a left outer join)
core	core	only those customers who purchased something (an inner join)

The table below describes how you use core tables to configure meanings of input table joins:

Number and type of your input tables	Desired join result	Action
One input table	want to map some columns to the federated table	make sure the input table is a core table
Two input tables	want to display all values in all rows, including null values	make sure only one input table is a core table
Two input tables	want to display rows that contain null values	make sure both input tables are core tables
Three input tables	have a non-core table between two core tables	make sure you change the non-core table to a core table or one of the outer core tables to a non-core table

The effects on the federated table, of assigning an input table as a core table are represented in the following diagram (example in English):



8.5 Mappings in the federation layer

Mappings define transfomations of values in your input tables and values in your federated tables.

When mapping columns, you need to know the data types of the columns that you are mapping. An icon showing the data type of the column is displayed in front of the column name. For example, `AB` indicates a string data type, and `12` indicates numeric. To see the data types of federated tables, you can also edit the table.

After you have mapped a column, you can edit the mapping formula to transform the value. For example, you can use formulas to construct new values in the federated table column, combine multiple values, or calculate results.

Related Information

Mapping columns from the input table to columns of the federated table [page 204]
Editing a mapping formula [page 205]
Pre-filters and post-filters [page 207]
Input tables and joins [page 200]
Adding a mapping [page 206]
Activating and deactivating mappings [page 206]

8.5.1 Mapping columns from the input table to columns of the federated table

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. In the *Federation Layer* pane, select the federated table.
   The mapping tabs appear in the *Properties* pane.
3. If the federated table has more than one mapping, select the appropriate mapping tab.
4. Select a column in an input table and drag it to a column in the federated table.
   A mapping line appears between the columns.

Edit the mapping formula for the column.

Related Information

Editing a mapping formula [page 205]
Mappings in the federation layer [page 204]

8.5.2 Editing a mapping formula

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. In the *Federation Layer* pane, select the federated table.

   The mapping tabs appear in the *Properties* pane.
3. If the federated table has more than one mapping, select the appropriate mapping tab.
4. Right-click the column in the federated table and select *Edit Mapping Formula*.
5. Edit and validate the SQL expression for the mapping formula in the SQL Expression Editor, and when you are finished, click *OK*.
6. To save the data flow in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

About the SQL Expression Editor [page 205]

8.5.3 About the SQL Expression Editor

The SQL Expression Editor helps you write valid SQL expressions.

You can expand the SQL Expression Editor to full screen using the expand/collapse button.

You can type SQL directly into the *Expression* box, drag and drop table or column names from the *Sources* pane, and drag and drop operators and database functions from the *Functions* pane. These panes are described in the following table. To display a resource pane, click the icon in the tool bar of the *Expression* pane.

Click the *Validate* icon in the tool bar of the *Expression* pane to check if the expression you have defined is valid SQL.

To save the expression, click *OK*.

Icon	Description
Sources	The list of tables and columns in the data foundation. To see a list of values for a column, click the icon next to column name.

Icon	Description
Functions	The list of functions that can be used in the expression. The functions are grouped by type: • *Operators*: Common database operators, for example *, SUM, IS NOT NULL. • *Database Functions*: The SQL functions that are valid for multisource-enabled data foundations. See the related topic about SAP BusinessObjects SQL functions.

Related Information

SAP BusinessObjects SQL function reference for multisource-enabled universes [page 423]
Default font for scripts and SQL expression editors [page 418]

8.5.4 Adding a mapping

If you have not yet defined a default mapping for the federated table, see the related topic about mappings. This task describes adding mappings in addition to the default mapping.

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. In the *Federation Layer* pane, select the federated table for which you want to add a mapping.
3. In the *Properties* pane, click the *Add Mapping* tab.
4. Enter a name for the mapping and click *OK*.
5. To save the data flow in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

You can now add input tables, map the columns, edit mapping formulas, and add filters for the new mapping.

Related Information

Mappings in the federation layer [page 204]
Adding input tables to a mapping [page 201]
Mapping columns from the input table to columns of the federated table [page 204]
Editing a mapping formula [page 205]
Pre-filters and post-filters [page 207]

8.5.5 Activating and deactivating mappings

The effective mapping for a federated table is the implicit union of all activated mappings.

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.

2. In the *Federation Layer* pane, select the federated table in the *Federation Data Flow* pane.
3. Right-click the mapping that you want to activate or deactivate and select *Activate*.

   When a mapping is deactivated, the mapping name is crossed-out in the table view.
4. To save the data flow in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

8.6 Distinct rows on input tables

The distinct rows feature lets you specify whether the rows coming from an input table should be distinct. You can turn on the distinct rows feature on each input table.

8.6.1 Activating and deactivating distinct rows

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. In the *Federation Layer* pane, select the federated table.

   The mapping tabs appear in the Properties pane.
3. If the federated table has more than one mapping, select the appropriate mapping tab.
4. Right-click the input table name and select *Distinct Row*.

   > **i Note**
   >
   > If there is a checkmark before the Distinct Rows menu item, it indicates that the feature is active and if there is not one, it means the feature is not active.
5. To save the data flow in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

8.7 Pre-filters and post-filters

Filters let you transform data in mappings in two ways:

- Pre-filters let you limit the source data that is queried in the mapping. For example, you can use a filter to limit customer data to those customers who are born after a certain date.
  You can use a pre-filter on each input table that is used in a mapping.
- Post-filters let you limit the data after it has been treated by table joins. Use post-filters when the filter definition depends on columns from more than one input table. For example, to limit the orders to customers who were 18 years old or older on the order date.
  You can use one post-filter per mapping.

Pre-filters are applied before the table joins. Post-filters are applied after the table joins. Mapping formulas are applied after the post-filters.

8.7.1 Adding and editing pre-filters

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. In the *Federation Layer* pane, select the federated table.

   The mapping tabs appear in the *Properties* pane.
3. If the federated table has more than one mapping, select the appropriate mapping tab.
4. Select the input table and do one of the following:

Option	Description
To add a pre-filter	Right-click the input table name and select *Add Pre-filter*.
To edit and existing pre-filter	Right-click the input table name and select *Edit Pre-filter*.

> **i Note**
>
> One pre-filter is allowed per input table.

5. Edit and validate the SQL expression for the pre-filter in the SQL Expression Editor, and when you are finished, click *OK*.
6. To save the data flow in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

About the SQL Expression Editor [page 205]
Pre-filters and post-filters [page 207]

8.7.2 Editing post-filters

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. In the *Federation Layer* pane, select the federated table for which you want to add a mapping.
3. If the federated table has more than one mapping, select the appropriate mapping tab.
4. Click the *Edit Post-Filter* icon .

   > **i Note**
   >
   > One post-filter is allowed per mapping.
5. Edit and validate the SQL expression for the post-filter in the SQL Expression Editor, and when you are finished, click *OK*.
6. To save the data flow in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

About the SQL Expression Editor [page 205]
Pre-filters and post-filters [page 207]

8.8 Showing values in a federated table

The show values command applies the pre-filters, joins, post-filters, and mapping formulas. If the federated table contains an input column, you are prompted for a value.

By default, show values opens a tab in the editor to display the values. You can set a preference to have the values open in a dedicated view or a dialog box. For more information, see the related topic.

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. In the *Federation Layer* pane, right-click the federated table and select *Show Table Values*.

   The show values window appears. To see what you can do in this window, see the related topic on showing values in a data source.

Related Information

Showing values in a data source [page 162]
Preferences for the Show Values command [page 417]

8.9 Checking integrity of the federation layer

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. Do one of the following:

Option	Description
To check integrity of the federation layer only	Click *Federation Layer*
To check integrity of the data foundation, including the federation layer	Click *Data Foundation* and select the top level of the data foundation in the tree view.

3. From the main tool bar, select the *Check Integrity* icon .

For more information about check integrity rules and the results of an integrity check, see the Checking the integrity [page 398] page. To learn how to select rules to be run automatically whenever you save a resource or to change the severity of the messages returned by each rule see the Check integrity settings [page 405] page.

Related Information

Checking the integrity [page 398]
Check integrity settings [page 405]
Reviewing check integrity problems [page 400]

8.10 Inserting a federated table into the data foundation

Before you can insert a federated table into the data foundation, you must define the federated table and data flow in the federation layer.

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. Select *Insert Federated Table* from the *Insert* menu in the Data Foundation view.
3. In *Insert Federated Tables*, select the federated tables you want to insert, and click *OK*.
4. Save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Building the federation data flow [page 196]

8.11 Refreshing the structure of the federation layer

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. Do one of the following:

Option	Description
To refresh the structure of the federation layer only	Click *Federation Layer*. In the *Federation Layer* pane, click the *Refresh Structure* icon.
To refresh the structure of the data foundation, including the federation layer	Click *Data Foundation* and then select *Actions* > *Refresh Structure*.

For the Federation Layer, the wizard detects the following changes and lists them each in their own dialog box. In each case, you select which of the proposed changes to make in the federation layer.

- Tables in the federation layer that were deleted in the database. The wizard proposes to delete these tables and any related joins from the federation data flow.

- Columns in federation layer tables that were deleted in the database tables. The wizard proposes to update each corresponding table in the federation layer to delete these columns and the joins that use these columns.
- Columns added in the database. The wizard proposes to update each corresponding table in the federation layer to add these columns.
- Column data types changed in the database. The wizard proposes to update the data type of each column in the data foundation that is different from the database column type.

3. Select changes in a summary dialog box and click *Finish* to continue with the refresh.
4. To save the changes in the federation layer, save the data foundation by clicking the *Save* icon in the main tool bar.

Check the integrity of the federation layer to detect definitions in your data flow that need to be updated because of the changes made by the refresh structure.

Related Information

Checking integrity of the federation layer [page 209]

9 Working with the business layer

Create the business layer that contains the objects for your queries and sets. You create and edit your business layer with the *Business Layer Editor*.

Related Information

Getting started with business layers [page 212]
Business layers [page 212]
Building a relational business layer [page 227]
Building an OLAP business layer [page 231]
Working with the Business Layer Editor [page 225]

9.1 Getting started with business layers

A business layer is a collection of metadata objects that map to SQL or MDX definitions in a database. The objects correspond to a unit of business information that can be manipulated in a query to return data. The principle role of the business layer is to define and organize metadata before it is published as a universe. Use the *Business Layer* editor to insert and manage objects. The business layer is saved as a .blx format file

Related Information

Business layers [page 212]
Business layer views [page 213]
Business layer objects [page 215]
Properties [page 218]

9.1.1 Business layers

A business layer is a collection of metadata objects that map to SQL or MDX definitions in a database, for example, columns, views, database functions, or pre-aggregated calculations. The metadata objects include dimensions, hierarchies, measures, attributes, and predefined conditions. Each object corresponds to a unit of business information that can be manipulated in a query to return data. Business layers can be created directly on an OLAP cube, or on a data foundation that is built on a relational database.

When a business layer is complete, it is published to a repository or a local folder as a universe. A universe is a published .unx file that includes a business layer and its connection to an OLAP cube, or a business layer and its corresponding data foundation. The universe is available in the repository to SAP BusinessObjects data analysis and report creation applications.

The principle role of the business layer is to define and organize metadata before it is published as a universe. An alternative way to understand the business layer is to think of it as a metadata workbench that a designer uses to assemble and modify a metadata set before publishing as a universe for data analysis and report creation applications.

Related Information

Business layer objects [page 215]
Properties [page 218]
Building a relational business layer [page 227]
Building an OLAP business layer [page 231]
Business Layer Editor preferences [page 404]

9.1.2 Business layer views

You can modify the display of business layer objects by using business layer views to restrict the number of objects displayed in the *Business Layer* pane. Use business layer views to group objects that share a business relationship.

You can also copy a business layer view that you have already built in a core universe into a linked business layer, so that you can reuse everything that has been defined in the core universe. You can select business layer views and order them using drag and drop in the Query Panel. You can use business layer views to define security to grant or deny the use of business layer objects to certain users or groups. For more information on defining security using business layer views, see the related topic on Business Security Profile Create Query settings.

You can also filter the *Business Layer* pane in the editor by business layer view.

Related Information

Creating and editing a business layer view [page 214]
Filtering by business layer view [page 214]
Business Security Profile Create Query settings [page 338]

9.1.2.1 Creating and editing a business layer view

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Click the *Manage Business Layer Views* icon at the top of the *Business Layer* pane.
   The *Edit Business Layer View* dialog box opens.
3. Do one of the following:
   - To add a view, click *New*.
   - To edit an existing view, select the view in the list.

   > **i Note**
   >
   > You cannot edit the *Master* view.
4. Edit the view name in the *Name* text box.
5. In the *Objects in view* box, select or clear the check boxes next to objects in the business layer to include or exclude them from the view.

   To work with only the objects already included in the view, select *Show selected objects only*.
6. Enter or edit a description for the view in the *Description* text box.
7. Click *OK* to save the changes.

Related Information

Business layer views [page 213]

9.1.2.2 Filtering by business layer view

By default all the folders and objects in the business layer are displayed in the *Business Layer* pane of the editor. You can filter what you see in the Business Layer pane using a business layer view.

You must have at least one business layer view defined.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Select the business layer view in the list at the top of the Business Layer pane.

   To return to the display of all objects in the business layer, select *Master* from the list.

Related Information

Creating and editing a business layer view [page 214]
Business layer views [page 213]

9.1.2.3 Copying a business layer view core into a linked business layer

You can copy a Business Layer view that you have already built in a core universe into a Linked Business Layer.

> **i Note**
>
> You can edit the business layer view copied into the linked Business layer.

1. Open the business layer in the editor by double-clicking the business layer name in the *Local Projects* pane.
2. Click the *Manage Business Layer Views* icon at the top of the *Business Layer* pane.
   The *Edit Business Layer View* dialog box opens.
3. Click the *Import from core* tab.
4. Select a core business layer.
5. Select a view and click *Copy the View*.
   The view is copied in the *View* tab.
6. Click *OK* to select it and edit it.

9.1.3 Business layer objects

The *Business Layer* objects pane contains the metadata objects that make up the business layer.

Every object in the business layer has a name that can be edited. When naming objects, use business vocabulary familiar to the users who will be using the universe for querying, analysis, and reporting.

Objects can have three states:

- *Active*: Object is visible in the Query Panel. This is the default state.
- *Hidden*: Object is valid but not available in the Query Panel (used by other objects as a hidden object).
- *Deprecated*: Object is hidden and not valid. One possible use for this state is when the target database field no longer exists, but you want to keep the object for possible future use.

Each object in the business layer has properties that are applied in the published universe. You can define properties when you insert an object, and modify object properties at any time. See the related topics about inserting and editing objects.

Depending on the type of data source, you can create and edit the following types of objects in the business layer:

Object	Description
Folder	A folder is a container that holds a group of related objects. You create folders to group objects that have a common purpose in the business layer. The folder has no role in a query. It is only used to organize objects.

Object	Description
Dimension	A dimension is an object that maps to one or more table columns or a function in a database and represents an axis of analysis in a query. For example, Product, Geography, Time, and Employee are common dimensions. Each dimension classifies an aspect of an activity in a business environment. In a business layer, dimensions represent contextual information (the axes of analysis).
Measure	Measures are objects that represent calculations and aggregate functions that map to statistical and analytic data in the database. In a business layer, measures represent the factual information (data). Numeric data is usually, but not always, the source of a measure. Aggregating the information must make sense for the object to be a measure. For example, summing up sales revenue makes sense, so Sales Revenue is a measure. Summing product list prices is not necessarily useful, so List Price is a dimension, or perhaps an attribute of the Product Dimension. You can create measures from non-numeric objects by counting things. This can result in measures like Number of Orders.
Attribute Measure attribute	An attribute is an object attached to a parent object that provides additional descriptive information about the parent. Attributes can be defined for dimensions, measures, hierarchies, and levels. In an OLAP business layer, a measure attribute provides information for the formatted value.
Filter	A filter is a condition object that limits the data returned in a query. Filters can be inserted into the Query Filters pane in the Query Panel to be applied to the query. Native filters are defined by an SQL WHERE clause on data foundation tables. Native filters apply to business layers based on data foundations. Business filters are defined by creating and combining conditions on dimensions and measures in the business layer.
Analysis dimension (OLAP only)	An analysis dimension allows you to logically group dimensions and hierarchies that share the same axis of analysis. Analysis dimensions are often used for hierarchical analysis. You define a default hierarchy for the analysis dimension. This is the hierarchy used when the entire analysis dimension is included as a result object in a query. Default hierarchies have the following icon:

Object	Description
Hierarchy (OLAP only)	A hierarchy is the representation in the business layer of the hierarchy in the OLAP cube. If the hierarchy in the cube is level-based, level objects in the business layer represent the levels. If the hierarchy in the cube is value-based (parent-child), the levels are not represented in the business layer. The levels are visible when previewing members and in the Member Selector. Value-based hierarchies generated automatically in the business layer have the following icon: .
Level (OLAP only)	A hierarchy level in a level-based hierarchy.
Named set (OLAP only)	A named set is a collection of members of a hierarchy in the business layer. A native named set is defined using an MDX expression. For some connections, native named sets are created automatically to represent named sets in the cube. A business named set is defined by selecting members.
Calculated member (OLAP only)	A calculated member is a member of a hierarchy that is calculated using an explicitly defined MDX expression that can include data from the OLAP cube, mathematical operators, numbers, and functions. Calculated members are available in the *Member Selector* when creating queries.

i Note

For date objects in the business layer, in the *Advanced* properties tab, there is a *Database Format* property with text Box :- "The format below determines the default regional settings. You can specify another format". This format (if it's not empty) will be used for SQL generation. By default, the date format for the object is defined in the Regional Settings Properties dialog box of the MS-Windows Control Panel. You can modify this to use the target database format for storing dates. For example, the date format could be US format, or European format.

Related Information

Working with business layer objects [page 264]
Inserting a folder [page 265]
Inserting and editing dimensions [page 266]
Inserting and editing measures [page 270]
Inserting and editing attributes [page 274]
Inserting and editing filters [page 276]
Inserting and editing analysis dimensions [page 278]
Inserting and editing hierarchies [page 279]
Inserting and editing hierarchy levels [page 280]
Inserting and editing named sets [page 282]

Inserting and editing calculated members [page 283]

9.1.3.1 Broken objects

When a table that is used by an object has been deleted, the object is broken.

An object is considered as broken when an element is missing, usually because a table has been deleted, and one of the table columns was used by the object. You can create a new table to replace the deleted table and the information design tool will offer the option to perform the necessary binding operations to repair the object. You can use the menu bar item *Actions* > *Repair Broken References*.

9.1.4 Properties

The following properties and options are defined for the entire business layer. The restrictions are applied in the published universe.

Property		Description
Name		Identifies the business layer and also the universe when the business layer is published.
Description		Describes the universe purpose and content. This description is available to display in the query and reporting tools that use the published universe.
Query Limits	*Limit result set size to*	Specifies the number of rows that are returned in a query. This limits the number of rows returned, but does not restrict the RDBMS from processing all rows in the query. It only limits the number once the RDBMS has started to send rows.
	Limit execution time to	Specifies the number of minutes to limit the time passed for query execution, but does not stop the process on the database.
	Warn if estimate exceeds	When selected, displays a message when an estimation of the execution time exceeds the specified number of minutes.
Query Options (apply to business layers based on data foundations)	*Allow use of subqueries*	When selected, allows subqueries in a query.
	Allow use of union, intersect, and minus operators	When selected, lets you combine queries using the data set operators union, intersect, and minus, to obtain one set of results.
	Allow complex operands in Query Panel	When selected, displays complex operands in the list of operands available when defining a filter in the Query Panel.

Property		Description
	Multiple SQL statements for each measure	When selected, one SQL query is generated for each measure or group of measures belonging to a different fact table, or for measures that have a WHERE clause (filtered measure). If the measure objects are based on columns in the same table, then the separate SQL queries are not generated, even if this option is selected.
	Allow query stripping	When selected, a report user can allow query stripping to be enabled for relational universes. (Query stripping is enabled by default in OLAP universes). Query stripping is used only by SAP BusinessObjects Web Intelligence. For a description of query stripping and how it can impact reporting in relational universes, see the related topic.
Data source		Specifies the data source of the business layer: either a data foundation or an OLAP connection. The *Change Data Foundation* button lets you change the underlying data foundation. The *Change OLAP Connection* button lets you change to OLAP connection and edit data source properties.
SQL Parameters (apply to business layers based on data foundations)		Specifies custom values for SQL generation parameters that override the default values or any customized value in the data foundation properties.
Comments		Contains comments about the business layer.
Summary		Displays a summary of the number of each type of object defined in the business layer. For business layers based on a data foundation, the type and number of data foundation objects are also displayed.

Related Information

Editing the business layer name, description, and comments [page 222]
Editing query limits and options in the business layer [page 222]
Changing the data source of a business layer [page 222]
Query stripping [page 220]
Setting SQL generation parameters in the business layer [page 223]
Displaying a business layer summary [page 224]

9.1.4.1 OLAP data source properties

The following properties apply to the OLAP data source for the business layer:

Property	Description
OLAP Connection	The connection or connection shortcut that provides the access to the OLAP data source. To change the connection, click the browse icon at the end of the field to open a list of available connections.
Cube	The cube selected for the current connection. You can select a different cube only if a cube was not specified when the connection was defined. To change the cube, click the browse icon at the end of the field to open a list of available cubes.
Accounts Dimension	For connections to Essbase data sources, the dimension in the data source to use as the Accounts dimension. Select a dimension from the list. When refreshing the structure of the business layer, measures are created in the business layer from the objects in the specified Accounts dimension.
END_MDX value	The value of the END_MDX parameter. The END_MDX parameter is equivalent to the END_SQL parameter available for universes based on data foundations. The value of the END_MDX is added to the end of every MDX statement. For example, you can use the END_MDX parameter to track the database server activity by tracing who is running queries. The solution consists of adding a comment at the end of every MDX query with information about the user and the universe. For example: `//User: @Variable('BOUSER') Universe: @Variable('UNVNAME')`

9.1.4.2 Query stripping

Query stripping is a reporting feature that can be used to optimize performance. Query stripping is used only by SAP BusinessObjects Web Intelligence.

For relational universes, query stripping is only enabled if the following parameters are set:

- The *Allow query stripping* option is selected in the business layer properties in the information design tool (unselected by default).
- The *Enable query stripping* option is selected for the data provider in the Query Properties in Web Intelligence.
- The *Enable query stripping* option is selected in the Document Properties in Web Intelligence (selected by default if query stripping is enabled for the data provider).

For OLAP universes, query stripping is enabled by default.

When query stripping is enabled, the query is rewritten to reference only objects that are used in the report. Let's take for example a query that contains three result objects: **`Country`**, **`City`**, and **`Revenue`**. A report based on this query may contain only **`City`** and **`Revenue`**. If query stripping is enabled, when the report is refreshed, in most cases the query will only retrieve the data for **`City`** and **`Revenue`**.

In relational universes, a report with query stripping enabled may return different data than when query stripping is disabled, depending on the schema of the data foundation. Let's look again at the example of a query that contains **`Country`**, **`City`**, and **`Sales Revenue`**. In the data foundation, there is a self-restricting join on the **`Country`** table that restricts the country to the US. With query stripping disabled, the report on **`City`** and **`Revenue`** returns revenue for only cities in the US. With query stripping enabled, the report returns revenue for cities in all countries, because the **`Country`** table was stripped out of the query.

Enhanced Query Stripping

The USE_ENHANCED_QUERY_STRIPPING parameter allows you to take advantage of enhancements to the query stripping method. Normal query stripping rewrites the query to contain only the objects referenced in the report and any joins concerned by those objects. Enhanced query stripping only optimizes the SELECT and GROUP BY clauses to avoid fetching unused data, but it doesn't modify the other clauses or the joins. It is recommended to use enhanced query stripping in the following situations:

- The data foundation contains outer joins.
- The data foundation contains self-restricting joins (column filters).
- The data foundation contains shortcut joins.

If aggregate awareness is defined in the business layer (using the @Aggregate_aware function in the definition of business layer objects), enhanced query stripping is used in every case, even if the USE_ENHANCED_QUERY_STRIPPING parameter is not set.

The USE_ENHANCED_QUERY_STRIPPING parameter is not set by default. It can be set in the data foundation or business layer. For more information, see the related topics.

Related Information

Editing query limits and options in the business layer [page 222]
USE_ENHANCED_QUERY_STRIPPING [page 514]
Setting SQL generation parameters in the data foundation [page 130]
Setting SQL generation parameters in the business layer [page 223]

9.1.4.3 Editing the business layer name, description, and comments

1. Open the business layer in the editor by double-clicking the business layer in the Local Projects View.
2. Make sure the top level of the business layer is selected in the tree view in the *Business Layer* pane.
3. Change the business layer properties in the editing pane:
   - To change the business layer name, edit *Name*.
   - To enter or edit the business layer description, click the *Properties* tab.
   - To enter or edit comments for the business layer, click the *Comments* tab.
4. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Properties [page 218]

9.1.4.4 Editing query limits and options in the business layer

1. Open the business layer in the editor by double-clicking the business layer in the Local Projects View.
2. Make sure the top level of the business layer is selected in the tree view in the *Business Layer* pane.
3. Click the *Query Options* tab in the editing pane.
4. Select or unselect options and edit limit values as required. For a description of the options, see the related topic.
5. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Properties [page 218]

9.1.4.5 Changing the data source of a business layer

To change the data source for a business layer, the new data source (data foundation, OLAP connection or connection shortcut) must be saved in the same local project folder as the business layer.

> **Note**
>
> When you change the connection to an existing cube, the original connection information is still stored in cache, and the *Change OLAP Connection* button is not activated. You must reset the connection by closing and reopening the universe editor. The *Change OLAP Connection* action is now available.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Make sure the top level of the business layer is selected in the tree view in the *Business Layer* pane.
3. Make sure the *Properties* tab is selected in the editing pane.
4. Do one of the following depending on the type of data source for the business layer:

Option	Command
For data foundation sources	Click *Change Data Foundation*. Select the new data foundation from the list and click *OK*.
For OLAP sources	Click *Change OLAP Connection*. In the *Edit OLAP Data Source Properties* dialog box, click the browse icon ... at the end of the *OLAP Connection* text box. Select the new OLAP connection or connection shortcut and click *OK*. **Note** For more information on advanced OLAP properties, see the related topic.

5. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

OLAP data source properties [page 220]

9.1.4.6 Setting SQL generation parameters in the business layer

The custom values for SQL generation parameters in the business layer override the default values or any customized value set in the data foundation properties.

> **Example**
>
> In SAP HANA you want to be able to identify the user name of universe users executing queries in Web Intelligence or the information design tool. In the *Query Script Parameters* dialog box, you can set the SQL generation parameter END_SQL to use the BOUSER variable to pass the user name to SAP HANA as follows: `END_SQL=-- @Variable('BOUSER')`.

1. Open the business layer in the editor by double-clicking the business layer in the Local Projects View.
2. Make sure the top level of the business layer is selected in the tree view in the *Business Layer* pane.
3. Make sure the *Properties* tab is selected in the editing pane.
4. Click the *Parameters* button.
5. In the *Query Script Parameters* dialog box, edit the parameters:

   The currently defined SQL generation parameters are listed. Non-default parameters and parameters with non-default values are in bold type.

Option	Action
Change the value of an existing parameter.	Click the *Value* column and select or enter the new value.
Add a predefined parameter.	To display the list of predefined parameters, click the arrow in the list box next to the *Add* button. Select the parameter from the list and click *Add*.
Add a custom parameter.	Make sure no predefined parameter is listed in the box next to the *Add* button, then click *Add*. A parameter with a default name is added to the table. To edit the parameter name, click the *Name* column. Click the *Value* column to enter a value.

   To see a description of all the predefined SQL generation parameters and their values, click the help button.

6. To return to the default list of parameters and the default values, click *Default Values*. This removes any added parameters from the list and sets all values to the default.
7. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

SQL generation parameters [page 500]

9.1.4.7 Displaying a business layer summary

Use this command to display the number of each type of object defined in the business layer. For business layers based on a data foundation, the type and number of data foundation objects are also displayed.

1. Open the business layer in the editor by double-clicking the business layer in the Local Projects View.
2. Make sure the top level of the business layer is selected in the tree view in the *Business Layer* pane.
3. Make sure the *Properties* tab is selected in the editing pane.
4. Click the *Summary* button.

The business layer summary displays in a new dialog box.

9.2 Building a business layer

Exactly how you build your business layer depends on the type of data source. Refer to the following sections to build your business layer.

Related Information

Working with the Business Layer Editor [page 225]
Building a relational business layer [page 227]
Building an OLAP business layer [page 231]

9.2.1 Working with the Business Layer Editor

You use the Business Layer Editor to create and edit business layer objects and properties. This topic describes how to navigate the editor. For steps to help you build the structure of your business layer, see Building a relational business layer [page 227] or Building an OLAP business layer [page 231].

The Business Layer Editor is divided into browsing panes on the left, an editing pane on the upper right, and a data source pane on the lower right.

The browsing panes allow you to work with different elements of the business layer. Access the panes by clicking the corresponding tab:

- *Business Layer*
- *Queries*
- *Parameters and Lists of Values*
- *Navigation Paths*

For more information about what you can do in each of the browsing panes, see the related topic.

The *Business Layer* is the default browsing pane. It displays a tree view of the objects in the business layer. The following options are available for displaying and navigating the business layer tree view:

- Filter by business layer view
- Search for an object
- Change display options: Show or hide objects, show unique names
- Toggle the table display between Collapsed, Joins Only or Expanded

The editing pane lets you edit the properties of the object or element selected in the browsing pane. To change how object names are generated in relational business layers, see the Business Layer Editor preferences [page 404] page.

The data source pane displays the data foundation or the OLAP connection information:

- The data foundation master view containing all tables and joins displays by default. Tabs for other data foundation views, if defined, appear at the bottom of the data source pane. To change views, click the tab.

- The OLAP metadata in the connection displays in the left side of the data source pane. Select a metadata object to display its properties in the right side of the pane.

Related Information

Business Layer Editor preferences [page 404]
Properties [page 218]
Business layer objects [page 215]
Queries in a business layer [page 242]
Parameters [page 238]
Navigation paths for objects [page 297]
Business layer views [page 213]
Filtering by business layer view [page 214]
Searching for business layer objects [page 297]
Changing display options of the business layer tree view [page 226]
Changing the table display in the Business Layer Editor [page 227]

9.2.1.1 Changing display options of the business layer tree view

When editing a business layer, the *Business Layer* browsing pane displays a tree view of the objects in the business layer. Use this procedure to change the display mode of the business layer objects. To change how object names are generated in relational business layers, see the Business Layer Editor preferences [page 404] page.

1. Click the *Display options* icon at the top of the *Business Layer* browsing pane.
2. For business layers based on an OLAP connection, select one of three options:
   - *Display caption* to display the object names.
   - *Display unique name* to display the unique object name from the cube.
   - *Display caption and unique name*
3. To show only active objects in the business layer tree view, select *Hide non-active objects*.

The display options remain in effect until you close the editor.

Related Information

Changing the state of an object: Active, Hidden, or Deprecated [page 286]
Business Layer Editor preferences [page 404]

9.2.1.2 Changing the table display in the Business Layer Editor

In the *Business Layer Editor*, you can display tables in the following ways: Collapsed, Joins Only or Expanded, and you can toggle between the views.

You can display the tables in the following ways:

Table Display Options

Option	How the tables are displayed
Toggle	Toggle between the following view options: Collapsed, Joins Only, Expanded.
Collapsed	Collapse the tables and show just the table header.
Joins Only	Collapse the tables and show the joins between the tables.
Expanded	Expand the tables and show the full tables.

You can change the display for all the tables in the *Data Foundation Editor*, or you can select multiple tables and change the appearance of the selected tables. You can also use *Ctrl+T* to toggle the (selected) tables between the different display options.

> **i Note**
>
> When you select several tables and select Toggle for the first time, the status of the tables is aligned with the status of the first selected table.

To change how object names are generated in relational business layers, see the Business Layer Editor preferences [page 404] page.

Related Information

Business Layer Editor preferences [page 404]

9.2.2 Building a relational business layer

Before you begin:

- You need a project in the Local Projects View.
- You need a data foundation saved in the same local project.

For links to more detailed information on each step, see the Related Topics.

1. To start the *New Business Layer* wizard, do one of the following:

- Right-click a data foundation in the Local Projects View and select *New* > *Business Layer*.
- Right-click the project folder in the Local Projects View and select *New* > *Business Layer* and select *Relational data foundation* as the data source.

The business layer is created in a .blx file in the local project folder. It opens automatically in the Business Layer Editor.

2. Build the business layer:

   If you created the business layer with the *Automatically create folders and objects* option selected (default), all objects are created in the business layer as dimensions. You need to specify the measures explicitly using the *Turn into Measure with Aggregation Function* command.

   Otherwise, insert the business layer objects:

   a. Insert folders and subfolders to organize the business layer.

   > **i Note**
   >
   > When you drag a table into the business layer from the data foundation, a folder is automatically inserted.

   b. Drag and drop tables and columns into the desired folders and rename the objects if needed.
   c. Specify the measures using the *Turn into Measure with Aggregation Function* command.
3. You can enhance the function of the business layer in several ways, for example:
   - Insert attributes to provide descriptive information for dimensions and measures
   - Insert additional measures
   - Insert pre-defined filters (mandatory or optional) that can limit data returned in queries
   - Insert parameters with optional prompts
   - Insert custom lists of values to be associated with a prompt
   - Insert navigation paths to define drill paths
   - Create business layer views to restrict the objects seen in the Query Panel
   - Set SQL options and SQL generation parameters in the business layer properties
   - Set up aggregate awareness to improve query performance
4. Run an integrity check to validate the dependencies, object expressions, and parameters and lists of values. Right-click the business layer name in the *Business Layer* pane, and select *Check Integrity*.
5. Save the business layer by clicking the *Save* icon in the main tool bar.

Some commands to help you maintain the business layer are listed below.

- If you change object definitions, use *Show Local Dependencies* to find the other business layer objects and data foundation objects that might be impacted by the changes.
- If the related data foundation is modified, you must refresh the business layer manually. The related topic describes commands to help you do this.
- Use *Change Data Foundation* to change the source data foundation for the business layer.
- For multisource-enabled business layers, use *Compute Statistics* to improve query performance.

Related Information

Creating a local project [page 65]

Building a data foundation [page 126]
Working with the Business Layer Editor [page 225]
Business layer objects [page 215]
Turning a dimension or attribute into a measure [page 273]
Inserting a folder [page 265]
Inserting and editing dimensions [page 266]
Inserting dimensions directly from the data foundation [page 268]
Inserting and editing measures [page 270]
Inserting and editing filters [page 276]
Inserting and editing a parameter [page 239]
Business layer views [page 213]
Properties [page 218]
Aggregate awareness [page 262]
Resource dependencies [page 395]
Checking the integrity [page 398]
Refreshing business layers [page 254]
Changing the data source of a business layer [page 222]
Computing statistics for optimized query execution [page 243]

9.2.2.1 Specifying the type of data source for a business layer

This section describes the *Select the type of data source for the business layer* page of the New Business Layer wizard.

You choose to create a business layer from either a relational or an OLAP data source.

Data Source Type	Description
Relational	The business layer is based on a data foundation. You can select any data foundation in the current project folder.
OLAP	The business layer is based on an OLAP cube. You can select any OLAP connection or connection shortcut in the current project folder.

1. Click one of the data source types in the list.
2. Click *Next*.

9.2.2.2 Naming a business layer

This section describes the *Resource Name* page of the New Business Layer wizard.

You enter a name and description of the business layer. This is the name of the universe that is published from the business layer.

> **i Note**
>
> Don't use the following characters in the business layer name: `\ / : * ? " < > |`.

Related Information

Selecting a data foundation for a business layer [page 230]
Selecting an OLAP connection for a business layer [page 232]

9.2.2.3 Selecting a data foundation for a business layer

This section describes the *Select Data Foundation* page of the New Business Layer wizard.

Select a data foundation as the data source for the new business layer. You can choose to do one of the following:

- Automatically create the business layer objects from the tables and columns in the data foundation.
- Create an empty business layer. You must manually add the objects from the data foundation after creation.

1. Click the browse button at the end of the Data Foundation text field.
   A list of available data foundations appears.
2. Select a data foundation in the list and click *OK*.

   The data foundation name appears in the name field. The *Automatically create folders and objects* option is selected by default.

   Business layer object names are generated based on table and column names (except for data foundations based on SAP ERP and SAP BW, which use a dedicated strategy for naming objects). You can set an application preference to determine how names are generated. For more information, see the Business Layer Editor preferences [page 404] page.
3. Do one of the following:
   - If you want to automatically populate the business layer with objects and folders, click *Finish*.
   - If you do not want to automatically populate the business layer, unselect the option and click *Finish*.

The new business layer opens in an editing tab. You can now insert and edit the business layer objects.

The *Automatically create folders and objects* option creates all objects in the business layer as dimensions. Specify the measures explicitly using the *Turn into Measure with Aggregation Function* command. For more information, see the related topic.

Related Information

Building a relational business layer [page 227]
Turning a dimension or attribute into a measure [page 273]

Business layer objects [page 215]
Business Layer Editor preferences [page 404]

9.2.3 Building an OLAP business layer

Before you begin:

- You need a project in the Local Projects View.
- You need an OLAP connection or connection shortcut saved in the same local project.

> **i Note**
>
> You can now create a business layer directly on a BEx Query using SAP BICS Client. For BEx query specific business layer options and settings, see the section Working with a BEx query business layer [page 234].

For links to more detailed information on each step, see the Related Topics.

1. To start the *New Business Layer* wizard, do one of the following:
   - Right-click an OLAP connection or connection shortcut in the Local Projects View and select *New* > *Business Layer*.
   - Right-click the project folder in the Local Projects View and select *New* > *Business Layer* and select *OLAP connection* as the data source.

   The business layer is created in a .blx file in the local project folder. It opens automatically in the *Business Layer Editor*.
2. Objects in the business layer are inserted automatically based on the cube. You can enhance the function of the business layer in several ways, for example:
   - Insert analytical dimensions, hierarchies, and attributes
   - Insert named sets
   - Insert calculated members
   - Insert measures and their formatted value attributes
   - Insert pre-defined filters (mandatory or optional) to limit data returned in queries
   - Insert parameters with optional prompts
   - Insert custom lists of values to be associated with a prompt
   - Create business layer views to restrict the objects seen in the Query Panel
   - Set values for SQL generation parameters that affect the generation of query scripts.
3. Run an integrity check to validate the dependencies, object expressions, and parameters and lists of values. Right-click the business layer name in the *Business Layer* pane, and select *Check Integrity*.
4. Save the business layer by clicking the *Save* icon in the main tool bar.

Some commands to help you maintain the business layer are listed below.

- If you change object definitions, use *Show Local Dependencies* to find the other business layer objects that might be impacted by the changes.
- If the underlying data source is modified, use *Refresh Structure* to remove deprecated objects, add created objects, or update modified objects.

- Use *Change OLAP Connection* to change the connection for the business layer and to edit the OLAP data source properties.

Related Information

Creating a local project [page 65]
OLAP connection [page 109]
Business layer objects [page 215]
Inserting and editing analysis dimensions [page 278]
Inserting and editing hierarchies [page 279]
Inserting and editing hierarchy levels [page 280]
Inserting and editing attributes [page 274]
Inserting and editing named sets [page 282]
Inserting and editing calculated members [page 283]
Inserting and editing measures [page 270]
Inserting and editing filters [page 276]
Inserting and editing a parameter [page 239]
Business layer views [page 213]
Checking the integrity [page 398]
Refreshing an OLAP business layer [page 255]
Changing the data source of a business layer [page 222]
Creating a business layer directly on a BEx query [page 233]

9.2.3.1 Selecting an OLAP connection for a business layer

You select an OLAP connection and then either an OLAP cube or a BEx query as a data source for your business layer. You have the following options when creating the data source:

OLAP connection options	Description
OLAP connection	Click the browse button at the end of the text field to select an OLAP connection or connection shortcut defined in the project.
Search	Enter a search string for a cube and click the search icon.
List of connection cubes	List of available cubes to the connection. If there are multiple cubes, you browse to and select the target cube.

If a BEx query contains variables, these will appear as prompts in the prompt pane. You set how these prompts are resolved in the *Set Variables* prompt manager that appears after the BEx query is selected. You have the following choices to determine how prompts are resolved when a document is refreshed:

- *Set as prompt*: When selected, you set the value for a static prompt and it's visibility. When hidden, the value is passed silently.
- *Use BEx query defined default values at runtime*: When selected, the default values defined in the query are used as the prompt response.

Related Information

Selecting objects from an OLAP cube for a business layer [page 237]
About projection functions [page 272]
SAP BW [page 40]
Creating a business layer directly on a BEx query [page 233]

9.2.3.2 Creating a business layer directly on a BEx query

You can create a business layer directly on a BEx Query using BICS access and publish the business layer as a universe available to Web Intelligence. Creating a universe based on a BEx query has a major advantage over using direct access to a BEx query as you can organize and customize the query dimensions, measures, and hierarchies in the business layer. Although you can edit certain components of the business layer, the data foundation is built automatically on the query and is read only.

> **i Note**
>
> In this release, navigation paths, parameters, and lists of values are not supported.

There are a number of constraints that apply to business layers built on BEx queries. These are described in Related Topics.

Related Information

Managing objects in a BEx query business layer [page 236]
Creating a BEx query business layer [page 233]
Building an OLAP business layer [page 231]
Selecting an OLAP connection for a business layer [page 232]

9.2.3.2.1 Creating a BEx query business layer

You can create a business layer on a BEx Query using BICS access.

1. Right-click a BW OLAP connection in a project.
2. Select *New Business Layer* from the contextual menu.
3. Type a name for the business layer in the *Resource* name box and click *Next*.
4. Browse to and select a BEx query then click *Next*.
5. If the query contains a variable the *Set Variables* dialog box appears. You can select prompt values from available members, or select the default BEx query run time value. Once you have defined how the prompt is managed, click OK.

For information on setting options in the *Set Variables* dialog box, refer to the section Working with a BEx query business layer [page 234].

6. Expand the BEx query and select the objects that you want to be included in the business layer. Click *Finish*.

   The new business layer appears in the Business Layer tab.

Related Information

Working with a BEx query business layer [page 234]
Managing objects in a BEx query business layer [page 236]
Building an OLAP business layer [page 231]
Selecting an OLAP connection for a business layer [page 232]

9.2.3.2.2 Working with a BEx query business layer

You work with business layer built on a BEx query in much the same way as for other OLAP data sources, however, there are a number of differences in the properties that you can set for an object, and available options compared to other types of business layers.

The following properties apply to all objects in a BEx query business layer:

- You cannot edit the OLAP query expression of objects.
- You cannot associate a list of values nor specify that the object can be used in a Sort.
- You can select a display format for the list of values.
  The following properties or options are dependant on the selected object:

Option	What does it do?	How do you set it?
Zero Suppression on Rows	Removes rows containing only measures with zero values. This setting is applied at the query level. The Zero suppression settings in BEx are not taken into account because they are not always applicable to the Web Intelligence display. The filtering applied by BICS allows consistency across the different SAP Business Objects applications in the BI suite.	1. Click the business layer root node in the Business Layer browser. 2. Click the *Query Options* tab and select *Zero Suppression on Rows*.
Show values	Shows a list of characteristic or key figure data returned from a query run on the fact table.	○ Click an object then click *Show Values* in the general properties editor. Or ○ Right-click an object and select *Show Values* from list.
Preview list of values	Shows a preview list of BW characteristic values.	Right-click an object and select *Preview list of values* from list.

Option	What does it do?	How do you set it?
Relative Depth	You can use this setting to specify how members are structured in the memory cache. This allows you to control the volume of data returned to Web Intelligence. It is available only for BEx queries with Hierarchy Node variables. As Web Intelligence fetches data in a single operation, the hierarchies are stored expanded in the memory cache. As large hierarchies can have millions of members, only a subset of descendants can be fetched. *Relative Depth* allows you to optimize the data fetched.	1. Click a hierarchy node and click the *Advanced* tab. 2. In the Relative Depth section, set one of the following: ○ *Hierarchy node descendants*: Depends on the BEx default setting (Expand to Level). ○ *Hierarchy members based on relative depth*: You chose the number of levels relative to the parent node.
Use BW presentation	You can choose the most accurate text presentation for a dimension. A BW characteristic can have multiple text presentations; short text, long text, or key. For example, a member `France` of the BW characteristic `Country` can be represented as: ○ Short text: FR ○ Medium text: FRA ○ Long text: FRANCE However in a business layer, the dimension has only a single text presentation. You can set *Use BW presentation* to choose the most accurate text presentation. **i Note** *Use BW presentation* only applies to the result set in Web Intelligence. It does not apply to the list of values related to the BEx variable as this uses a default BW/BEx presentation, so the presentation definition set on a characteristic is not kept when variables are viewed in the set variable list of values (LOV).	1. Click a dimension node and click the *Advanced* tab. 2. In the *Display* section, select a BW characteristic text presentation from the *Use BW presentation* drop down list.
Projection Function	You can configure the ability of Web Intelligence to compute grouping set by extending aggregation provided by BW. You have the following options: ○ *Delegated*: All aggregations performed by BW. ○ *MIN, MAX, SUM, AVERAGE*: Aggregation is computed by Web Intelligence. ○ *NONE*: Do not aggregate measure.	1. Click a measure. 2. Select an aggregation option from the *Projection Function* drop down box in the general properties editor.

Option	What does it do?	How do you set it?
Set Variables	If a BEx query contains variables, these will appear as prompts in the prompt pane. You set how these prompts are resolved in the *Set Variables* prompt manager. You have the following choices to determine how prompts are resolved when a document is refreshed: ○ *Set as prompt*: When selected, you set the value for a static prompt and it's visibility. When hidden, the value is passed silently. ○ *Use BEx query defined default values at runtime*: When selected, the default values defined in the query are used as the prompt response.	1. Click the business layer root node in the Business Layer browser. 2. Click the *Set Variables* button at the bottom of the *Properties* tab. 3. Select an option to determine how the prompt is resolved.

Related Information

Managing objects in a BEx query business layer [page 236]

9.2.3.2.2.1 Managing objects in a BEx query business layer

Business layers created on BEx queries have certain limitations compared to business layers built on standard relational and OLAP data sources. The following possible actions are available to objects in a BEx business layer:

> **i Note**
>
> You cannot copy objects from one node or hierarchy to another. Where permitted, you can only move objects to change position.

Objects	Possible actions	Comments
Folder	Delete, rename, create, move.	You can only create a folder at the root level or inside an existing folder.
• Dimension • Dimension attributes • Measures • Measure attributes	Delete, rename, move.	Delete action also deletes child objects. You cannot change the parent of a dimension attribute or measure attribute.

Objects	Possible actions	Comments
• Default hierarchy • Hierarchy	Delete, rename, move.	You cannot change the parent of a hierarchy.

9.2.3.2.3 Refreshing the business layer

When you run *Refresh Structure* on the business layer, you recreate the query view. If new mandatory variables are present, the Set Variables dialog box appears. You can select how you want to manage these values. Refer to Related Links for information on setting variable options.

In some instances, running a *Refresh Structure* may be not sufficient to detect certain types of modifications to a BEx query or related InfoObjects. If this happens, for example if you added a hierarchy to a Characteristic, you need to restart the information design tool for the modification to be taken into account.

> **i Note**
>
> When you run the *Refresh Structure* wizard on a business layer, one or more levels may display under a hierarchy, which would not have appeared when the business layer was created from scratch. These levels are not errors, but are required by the application for future use in the business layer.

Related Information

Working with a BEx query business layer [page 234]

9.2.3.3 Selecting an Essbase Accounts dimension

This section describes the *Select Accounts Dimension* page of the New Business Layer wizard.

For connections to Essbase data sources, the New Business Layer wizard creates measures in the business layer from the objects in the specified Accounts dimension in the data source.

Select one dimension in the list you want to use as the Accounts dimension and click *Next*.

9.2.3.4 Selecting objects from an OLAP cube for a business layer

This section describes the *Select Objects* page of the New Business Layer wizard.

Expand the object nodes under the selected cube and select the objects to be included in the new business layer. Click *Finish* when you have completed the selection.

The new business layer appears in the Business Layer pane.

9.3 Managing a business layer

This section describes how to manage you business layer.

Related Information

Parameters [page 238]
Queries in a business layer [page 242]
Computing statistics for optimized query execution [page 243]
Stored procedures in the business layer [page 243]
Linking universes [page 244]
Refreshing business layers [page 254]
Index awareness [page 256]
Analytic functions [page 257]
Aggregate awareness [page 262]

9.3.1 Parameters

A parameter is a variable in the business layer or data foundation that requires a value at run time. A parameter can have two input types:

- User input as a response to a prompt. The prompt is a question or directive that requires a user to set one or more values to restrict a result set.
- Pre-defined input that specifies a fixed value for the parameter at run time.

Parameters are defined as individual components in a business layer or data foundation, and are available to all objects in the business layer. You use parameter objects in the SQL or MDX definition of an object to prompt a user response or to implement a fixed value response to a query.

> **i Note**
>
> Parameters inserted in the data foundation are inherited by any business layer based on the data foundation. These parameters cannot be edited in the business layer. You must edit them in the data foundation.

The following properties are available for parameters:

Property	Description
Prompt to users	If selected, the user is prompted to enter a value at run time. If cleared, a pre-defined value is entered at runtime for the parameter.
Prompt Text	The text for the prompt question or directive if *Prompt to users* is selected.
Set values	Available when the *Prompt to users* option is unselected. Lets you enter one or more values to be used for the parameter at the run time.
Data Type	The data type required for the answer to the prompt.
Allow multiple values	If selected, lets the user select multiple values from the list of values.
Keep last values	If selected, the last value chosen by the user is kept when the prompt is re-run.
Index aware prompt	If selected, the key column is included in the prompt to restrict the values in a list. The key column is not visible to the user.
Associated list of values	A list of values to provide values for the prompt.
Select only from list	If selected, the user is forced to select a member in the list.
Set default values	Lets you select values to be used as default propisition when a prompt is run. **i Note** The *Index aware prompt* checkbox must be cleared before you can select *Set default values*.
Constant	If selected, you can enter a constant to be the default value displayed when the prompt is run.
Formula	If selected, you can define a formula to set a dynamic default value for the prompt, for example the current date for a date entry prompt. **i Note** The *Formula* property is available when the *Prompt to users* property is either selected or cleared.

Related Information

Inserting and editing a parameter [page 239]
Reordering objects in the Business Layer Editor [page 299]
Business Layer Editor preferences [page 404]

9.3.1.1 Inserting and editing a parameter

The parameter editor can be started from the *Business Layer Editor* or *Data Foundation Editor* tabs. See the Business Layer Editor preferences [page 404] page and the Data Foundation Editor preferences [page 409] page to learn more.

> **i Note**
>
> Parameters inserted in the data foundation are inherited by any business layer based on the data foundation. These parameters cannot be edited in the business layer. You must edit them in the data foundation.

1. Click the *Parameters and Lists of Values* tab in the browsing pane of the editor.
2. Do one of the following:
   - To insert a parameter, click the *Insert Parameter* icon at the top of the *Parameters* pane.
   - To edit a parameter, click the parameter name in the list.

   The properties for the parameter appear in the editor to the right of the *Parameters* pane.
3. Do one of the following:
   - Edit properties as required. Parameter properties are described in Related Links.
   - To define default values, select *Set default values* then do one of the following:

     > **i Note**
     >
     > The *Index aware prompt* checkbox must be cleared before you can select *Set default values*.

Default value defined by	Do this
Constant	○ Click *Constant*. ○ Enter a value in the left pane of the *Prompts* list box, then click the right arrow head to add it to the list in the right pane, or if values have already been defined, select one or more values in the right pane, then click OK.
Formula	○ Click *Formula*. ○ Enter a formula expression in the Expression box of the Formula Editor. You can expand and select operators and functions to insert in the expression from the *Functions*pane. ○ Click *Validate* to check the expression syntax, then click OK.

Refer to Related Links for information on@Prompt syntax and the SQL function reference for formula expressions.

Related Information

@Prompt syntax [page 493]
SAP BusinessObjects SQL function reference for multisource-enabled universes [page 423]
Parameters [page 238]
Business Layer Editor preferences [page 404]

9.3.1.2 Creating an index-aware prompt

An index-aware prompt takes advantage of indexes on key columns in tables when accessing lists of values. You can define the prompt so that when running the query, you see and select the user-friendly name for the object. When retrieving values, the query uses the key column for better performance.

To create an index-aware prompt, you create a list of values and a parameter in the data foundation or the business layer. For more detailed information on each step, see the related topics.

1. Create a list of values based on custom SQL.
   a. Include both the key column and name column in the `SELECT` statement, for example:

      `SELECT reservations.Airline_ID, reservations.Airline_Name FROM reservations`

   b. In the *Properties* tab in the list-of-values definition, select the row for the name column. Open the drop-down list in *Key Column*. Select the key column.

      For example, the **`Key Column`** for **`Airline_Name`** is set to **`Airline_ID`**.

   c. In the row for the key column, select the *Hidden* check box.

      For example, *Hidden* is selected for **`Airline_ID`**.
2. Create a parameter.
   a. In the *Options* tab in the parameter definition, select *Prompt to users* and enter a *Prompt Text*.
   b. In *Associated List of Values*, select the list of values that you created in step 1.
   c. Select the *Select only from list* check box.
   d. Make sure that the *Index aware prompt* check box is selected.
3. If you want to use the parameter in a query filter in the Query Panel, you need to make the corresponding dimension index aware in the business layer by defining a key on the dimension.

Example

Following is an example of how you might use the index-aware prompt in a `WHERE` clause (for example in the SQL expression for a derived table, calculated column, or an object in the business layer).

```
WHERE reservations.Airline_ID= @Prompt (<parameter name>)
```

Related Information

Inserting and editing a parameter [page 239]
Defining keys for dimensions and dimension attributes [page 269]
Index awareness [page 256]
Business Layer Editor preferences [page 404]

9.3.2 Queries in a business layer

A query object is a query that is saved and associated with the business layer. You use the Query Panel to create queries. Queries are cataloged in the *Query* pane of the editor.

> **i Note**
>
> Queries can be used within the information design tool to test the business layer and to preview queries. Query objects are not available to reporting and analysis products using the published universe.

Related Information

Inserting and editing a query in the business layer [page 242]
Reordering objects in the Business Layer Editor [page 299]

9.3.2.1 Inserting and editing a query in the business layer

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Click the *Queries* tab under the *Business Layer* pane.
3. In the *Queries* pane, do one of the following:

Option	Command
To edit an existing query	Select the query. Edit the *Name* and *Description* in the *Query Properties* pane. To edit the query results and filters in the Query Panel, click *Edit Query*.
To insert a query	Click the *Insert Query* icon .

4. In the Query Panel, build or edit the query and click *OK*.
   The new query is available in the *Queries* pane.

Related Information

Building a query [page 307]

9.3.3 Computing statistics for optimized query execution

For queries on multisource-enabled universes, you can obtain the best performance if accurate table and columns statistics are available for the data federation service. The cost-based optimizer of the data federation service uses these statistics to determine the optimal join method and order.

The *Compute statistics* command optimizes query execution because it allows you to compute and store statistics in the repository for the universe.

You should compute statistics periodically for tables that might change in volume or for which column values are changing frequently.

The following statistics are generated for the optimization process:

- The table row count
- The number of distinct values for the columns

You set the following options:

- Select all tables and columns that were computed before a certain date
- Select all tables and columns that were never computed
- Select every table and column
- Unselect every table and column

Related Information

9.3.4 Stored procedures in the business layer

A stored procedure is a function that is stored in the database. Use the stored procedures to create a specific business object for your requirements. Note the following:

- When you create a business layer based on a stored procedure, you can only use stored procedures in that data foundation. You cannot have a mix of stored procedure tables and standard database tables.
- The stored procedure can contain variables that must be filled with static values or prompts when the stored procedure is inserted in the business layer.
- In the query panel, you can't view or edit scripts.
- In the autogenerated business layer, you can't edit the SQL of the business layer.
- You can't use the projection function of a measure based on a stored procedure column.
- The projection function of a measure based on a stored procedure column can't be set to delegated. The default behavior is SUM.
- You can't edit the Data type of a business object that is based on stored procedure.
- For business objects built on stored procedures, you can't use the business objects in filters or sorts.

> **i Note**
>
> Stored procedures are not supported for multi-source universes or linked universes.

Related Information

Inserting a stored procedure in a business layer [page 244]

9.3.4.1 Inserting a stored procedure in a business layer

1. In the *Business Layer Editor*, click *Insert* and select the *Stored Procedure* option, or drag and drop stored procedure tables or columns in the business layer outline.
2. Select a stored procedure table from the list of available tables

9.3.5 Linking universes

9.3.5.1 What are linked universes?

You can create a link between two universes so one inherits the data foundation and business layer from the other.

A linked universe is a .UNX universe that contains a link to a designated core universe in the CMS. It inherits the core universe's data foundation and business layer as read only resources. You can create views and objects from the core universe's components, as well as add new tables and objects that are independent of the core universe. When changes are made in the core universe, they are automatically propagated to the shared components in its linked universes, however, when you want to publish the universe, you must manually select Synchronize core first before publishing the universe in order for the changes to be visible in the published universe.

A core universe acts as a reusable and dynamic library of data foundation and business layer components for universes that link to it, so using linked universes allows you to take advantage of components in predefined and tested universes as a starting point to quickly build new universes.

A linked universe can reference one or more core universes so it can use components in each core universe's data foundation and business layer.

Related Information

What are the advantages of using linked universes? [page 245]
Working with linked universes [page 245]

9.3.5.2 What are the advantages of using linked universes?

You have the following advantages when linking universes:

Advantages of linked universes	Description
Reduce development and maintenance time.	When you modify a component in the core universe, the changes are propagated to the same component in all of its linked universes when each universe is refreshed.
Centralization of components.	Often used components can be centralized in a core universe and then included in all linked universes. You do not have to recreate components each time you create a new universe.
Facilitate specialization of universe design.	Universe development can be split between database administrators who can set up a core universe, and then the more specialized designers who create more functional business layers based on their specific field.

9.3.5.3 Working with linked universes

You create a linked universe in one of two ways:

Ways to create a linked universe	Description
Create a new linked universe directly from a core universe in the repository.	The core business layer and data foundation become the foundation for the new universe. Creating a new linked universe [page 246]
Add a link to a core universe from an existing universe.	The core business layer and data foundation are added to the current business layer and data foundation. Adding a core universe to an existing universe [page 247]

> **i Note**
>
> A linked universe contains a predefined data foundation and business layer inherited from one or more core universes, however, there are some components that are not inherited from the core universe. These must be recreated in the linked universe. Refer to Related Links for information on non-inherited components.

Related Information

Core universe components not inherited by a linked universe [page 248]

9.3.5.3.1 Creating a new linked universe

You create a linked universe from scratch by starting with an existing universe in the CMS. You want to use this universe's data foundation and business layer as pre-fabricated building blocks for your new universe. Once you have created the linked universe, you can save its data foundation and business layer inherited from the core universe as new files so they have a life cycle independent of the core universe.

You can use the same connection as the core universe or change the connection to another that is compatible with the core universe data source. You can add tables, create joins linking core data foundation tables with the new ones, and add new components to the business layer in the same way that you do for any other universe. Any changes in the core components are automatically propagated to the linked universe when it is checked into the CMS.

> **i Note**
>
> You cannot create a linked universe from a core universe that is itself a linked universe. However, it is possible that a core universe can be updated to reference another core universe after it has been linked.

For a universe to be used as a core universe, the following requirements and restrictions must be met:

The core universe is:

- a .unx file version 4.2 format
- single data source only
- not already a linked universe
- not an OLAP universe

1. Right-click a universe that is qualified to be a core universe in the *Repository Resources* panel.
2. Select *New Linked Universe* from the menu.
3. Select a project for the new universe then click *Next*.

   The *Select Connection* appears. It lists the available secured connections. The *Use same connection as the core universe* check box is selected by default. If you accept to use the same connection as the core universe, a shortcut for the core universe connection is created in the *Local Projects* pane.
4. In the *Select Connection* box, do one of the following then click *Next*.
   - Leave the *Use same connection as the core universe* check box selected if you want the new universe to use the same connection.
   - Clear the *Use same connection as the core universe* check box and select a connection in the list. The connection needs to access data that is compatible with the core universe's original connection.
5. In the *Resource Names* box, enter a name for the business layer and the data foundation then click *Finish*.

   The business layer and data foundation appear in their respective panes. Business layer objects appear in folders that include the core universe name. The core business layer and data foundation components are read only. In general, you can work with the business layer and data foundation as you would with any other universe, you can insert new tables, join new tables with core tables, create new objects, and views based on both the core and new tables, however, there are some limitations and behavior differences that are documented in Related Links.

   In a linked universe, the core universe business layer and data foundation components are displayed differently from the linked universe components. Refer to Related Information for the description of how core components are displayed in a linked universe.

Related Information

How are core universe components displayed in a linked universe? [page 248]
Working with the data foundation of a linked universe [page 252]

9.3.5.3.2 Adding a core universe to an existing universe

You can link to one or more core universes from the *Core Business Layers* tab of the *Business Layer Editor*.

> **i Note**
>
> When you link to a core universe, a unique identifier is attributed to each core universe component. Any objects in the linked business layer or data foundation that you associate with the core components, for example joins, views, or dimensions are based on these identifiers. If you remove the core universe and then re-link it, a new set of identifiers will be attributed to the same objects, so you will have to reassociate all of the previous components with the core universe.

1. Open a business layer that you want to link to a core universe.

   > **i Note**
   >
   > You cannot create a linked universe from a core universe that is itself a linked universe. However, it is possible that a core universe can be updated to reference another core universe after it has been linked.

2. If the data foundation is open, save and close it.

   The linked universe data foundation is updated with the core universe components when it is reopened after the universes have been linked. If you leave the data foundation open, you won't see the added core components until you close and reopen the linked universe data foundation.

3. In the *Business Layer Editor*, click the *Core Business Layers* tab.
4. Click the *+ Add Core* button.

   A list of repository folders and files appears.

5. In the left pane, browse to and select a universe folder, then click a universe name in the right pane. This is the core universe that you want to link to your active universe.
6. Click *OK*.

   The name of the core universe appears in the list of core universes in the *Core Business Layers* tab. The business layer components from the core universe appear in the business layer browsing pane to the left of the *Business Layer Editor*.

7. Open the data foundation for the newly linked universe from the *Local Project* browser.

   The core universe data foundation components now appear in the linked universe data foundation.

   In a linked universe, the core universe business layer and data foundation components are displayed differently from the linked universe components. Refer to Related Information for the description of how core components are displayed in a linked universe.

Related Information

How are core universe components displayed in a linked universe? [page 248]
Managing core universes from the Business Layer Editor [page 249]

9.3.5.3.3 Core universe components not inherited by a linked universe

A linked universe does not inherit the following core universe components:

Components not inherited from a core universe	Description
Rights	You must ensure that the linked universe rights are compatible with the core universe rights.
Connection	When creating a new linked universe or adding a core universe, you have a choice to reuse the core universe connection or use another compatible connection that allows you to access components outside of the core universe. It is recommended that you use another connection.
Contexts	You must create new contexts. Copy the context of the linked universe. The copied context is no longer synchonous with the core universe context. Any modification to the core universe context will not be reflected in the copied context.
Views	You must create new data foundation and business layer views.
Security	You must create new security rights for the linked universe.
Customs Navigation Path	You must create new navigation paths..

9.3.5.3.4 How are core universe components displayed in a linked universe?

In a linked universe, the core universe business layer and data foundation components are distinguished from the linked universe components as follows:

Core universe	Component	Displays as
Business layer	Folder names	Gray font followed by the name of the core universe in brackets. The name of the core universe is displayed in light blue font.

Core universe	Component	Displays as
	Object names	Gray
Data foundation	Table names	Gray followed by the name of the core universe in brackets. The name of the core universe is displayed in light blue font.
	Column names	Gray italics
	Joins	No change in the appearance, however the joins can't be edited.

> **i Note**
>
> A table that is common to multiple core universes is displayed following rules of priority to ensure component uniqueness in a data foundation. The display prioritization of common tables is described in Related Information.

Related Information

Prioritizing the display of tables common to multiple core data foundations [page 253]

9.3.5.3.5 Managing core universes from the Business Layer Editor

You can add or remove links to core universes and perform other management tasks for core universes linked to the active universe from the *Business Layer Editor*.

You can do the following management tasks for linked universes from the *Core Business Layer* tab of the *Business Layer Editor*:

Option	Click to:	More information
+ Add core	Select a core universe in the CMS to be linked to the active universe.	Adding a core universe to an existing universe [page 247]
Synchronize core	Synchronize all core universe structures linked in the active local universe.	Synchronizing core universe components in a linked universe [page 250]
x Remove core	Remove a link to a selected core universe.	Removing a core universe from a linked universe [page 250]

Option	Click to:	More information
Include core	Integrate the core universe components into the local universe. The link to the core universe is removed, so the core business layers and data foundation are now fully functional components of the active universe.	Integrating a core universe into a linked universe [page 251]

To learn more about the *Business Layer Editor,* see the Business Layer Editor preferences [page 404] page.

Related Information

Business Layer Editor preferences [page 404]

9.3.5.3.6 Synchronizing core universe components in a linked universe

You can synchronize the core universe structure and components in a linked universe with the core universe in the repository. This ensures that a linked universe consistantly contains the most recent version of its core universes. When you synchronize the core universe components, all core objects are updated, deleted core objects are removed, and new core objects are added.

> **i Note**
>
> You must do a Synchronize Core action before you publish a new version of your current universe.

1. In the *Business Layer Editor*, click the *Core Business Layer* tab.
2. Click a core universe name in the list.
3. Click the *Synchronize core* button.

   A message box appears when the synchronization has finished. Depending on the differences between the repository and current universe version of the core universe, components may have been modified, added, or deleted in the updated core business layer or data foundation.

9.3.5.3.7 Removing a core universe from a linked universe

When you remove a core universe from its linked universe the link is deleted between the core universe in the CMS and the current universe. You can not undo the removal. Once a core universe is removed, if you re-link, you create a new link and you must re-associate all previously referenced components such as views, calculated columns, and business layer objects.

1. In the *Business Layer Editor*, click the *Core Business Layer* tab.
2. Click a core universe name in the list.
3. Click the *Remove core* button.

   The core universe is removed from the list. All core universe objects from the business layer and its data foundations are removed.

9.3.5.3.8 Integrating a core universe into a linked universe

When you include a core universe into a linked universe, you permenantly integrate all objects from the selected core into the current universe. The link with the core universe is broken, and its objects in the current universe now exist independently from the core universe. All objects are now managed within the current universe, and ncluding core universe objects into the current universe is definitive, so you cannot return to any previous state. Once the link is broken, all previous core objects now have an independent lifecycle from the core universe.

1. In the *Business Layer Editor*, click the *Core Business Layer* tab.
2. Click a core universe name in the list.
3. Click the *Include core* button.

   The core universe is removed from the list. When you save the current universe, the previous core business layer and data foundation components are displayed exactly the same as the original components of the universe. The included core components are now writable. The core universe object and table naming font is replaced with the same labelling as the components specific to the current universe.

9.3.5.3.9 Repairing a linked universe when its core universe is missing

If the information design tool cannot find the original core universe, the indication *??* appears in the *Revision* tab. You can choose to repair and choose a compatible universe.

1. Right-click the core universe in the *Local Projects* pane and select *Publish* > *To a repository*.
2. Click *Next* in the *Publish Universe* dialog box.
3. Click the *Refresh* icon in the *Business Layer* pane.
   The *Info* dialog box appears. It informs you that the original core file has disappeared and that an equivalent core has been detected.
4. Select *Repair* to replace the missing core with the one proposed by the system.

9.3.5.4 Working with the data foundation of a linked universe

Generally you can work with a data foundation in a linked universe in the same way as you would any data foundation that is not part of a linked universe. The following components are inherited or recreated in the linked universe data foundation with no change:

- Calculated columns
- Parameters (prompts)
- SAP HANA variables

> **i Note**
>
> If you add multiple core universes to a linked universe, and a parameter with the same name exists in two core universe data foundations, then the parameter name in the first added core data foundation is kept, and the second has a number added as suffix. For example, if a parameter `<Prefered Country>` is common to two core universes, then the first linked core universe parameter retains the original name, and in the second core the parameter becomes `<Prefered Country(1)>`.

There are certain differences in the available user interface options of the linked universe and the behaviour related to the use of core universe components:

Data foundation differences in linked universes	Go here for information
Core universe components are read only	How are core universe components displayed in a linked universe? [page 248]
Core universe components are displayed differently from components created in the linked universe.	
Alternative ways of viewing the different components in the data foundation object browser when a linked universe contains multiple core universes.	Different ways to view core universe data foundation components [page 253]
Prioritized treatment for a table that is common to multiple core universe data foundations to ensure the uniqueness of a table identifier.	Prioritizing the display of tables common to multiple core data foundations [page 253]
Aliases and contexts in the core data foundation are not inherited in the linked universe. They need to be recreated in the linked universe.	Inserting alias tables [page 166] Inserting and editing contexts [page 182]
Views are not inherited in the linked universe. They need to be recreated in the linked universe.	Creating and editing a business layer view [page 214]
Joins can be created between linked and core universe data foundations, however, any join between a core universe table can not be edited.	

9.3.5.4.1 Different ways to view core universe data foundation components

A linked universe can inherit data foundation components from multiple core universes. To facilitate the display of data foundation objects in the data foundation browser pane, you can chose the following ways to display components of core universe data foundations:

Data foundation browser display option	Description
Consolidated view	Joins and tables under their respective nodes are not separated by core universe but listed together and distinguised by the name of the parent core universe in each join and table label.
View by data foundation	Within the *Joins*and *Tables* nodes, the joins and tables are organized in sub-groups for each respective core universe.

1. In the data foundation browser pane, click the *Display Options* spanner icon at the top of the pane.
2. Select either *Consolidated view* or *View by data foundation* from the context menu.

9.3.5.4.2 Prioritizing the display of tables common to multiple core data foundations

A linked universe can be linked to multiple core universes. If data foundations in two or more core unverses contain a table with the same name, only the table from the first core universe to be linked is displayed to ensure component uniqueness in the linked universe. The table from the first core linked always takes priority. The table is considered globally so that any differences in common tables are taken into account as follows:

Table1 (Core1) is added to the linked universe before Table1 (Core2).

Relationship between common core universe tables	What is displayed in the refreshed core universe table?
Table1 (Core1) and Table1 (Core2) are identical.	Table1 (Core1)
Table1 (Core2) has extra columns not included in Table1 (Core1), for example calculated or new columns.	Table1 (Core1). Table1(Core2) is ignored and the extra columns also.
Table1 (Core1) has columns removed	Table1 (Core1). The common columns in Table1 (Core2) are not taken into account if they have been removed from Table1 (Core1).

9.3.5.4.3 Joins between core and linked universe tables

You can create, edit, or remove joins between columns in a linked universe and its core universe as follows:

- Join two columns between linked universe tables only.
- Join two columns between tables where each is in a different core universe.
- Join a column of a linked universe table and a core universe table.
- Join two columns between tables in the same core universe.

9.3.6 Refreshing business layers

Refreshing an OLAP business layer

For business layers based on an OLAP cube, the *Refresh Business Layer* wizard detects changes in the OLAP cube and applies the changes to the business layer.

On the *Select Options* page, you can select which types of changes the wizard should detect in the cube.

Based on the detections, the wizard lists possible update actions in the *Select Actions* page. You can select which update actions you want to apply to the business layer.

Before applying the changes, the wizard displays a summary of update actions on the *Refresh Summary* page. You can save the summary to a file. You can go back and modify your selection before finishing the wizard.

A summary list is shown with the changes that are proposed in the business layer based on the changes in the cube structure. You can clear and select proposed changes before applying the update.

> **i Note**
>
> A refresh can be undone using the undo action. Undo will recover the business layer to its state before the refresh. To undo, from the information design tool main menu, select *Edit* > *Undo*.

Refreshing an SAP BW multisource-enabled business layer

When objects are added to the InfoProvider of an SAP BW data source, the procedure to update the universe involves several steps. First, you refresh the structure and synchronize tables in the data foundation. You refresh a business layer with new objects from the data source using the *Insert Candidate Objects* command.

Refreshing relational business layers

To update the business layer with changes to the underlying data foundation, you must manually delete and insert objects. To insert objects for new tables, in the *Business Layer Editor*, you can drag and drop tables from the data foundation view to the business layer browsing pane.

To identify objects based on tables that have been deleted from the data foundation, follow these steps:

1. In the Local Projects View, right-click the business layer and select *Refresh*.
2. Open the business layer in the editor. If an object in the business layer is based on a table that was deleted from or changed in the data foundation, in the *SQL Definition* tab of the object properties, the *Tables* field label is in red and the field contains the message *[Unresolved table]*.

Related Information

Refreshing an OLAP business layer [page 255]
Refreshing universes based on SAP BW [page 46]
Inserting candidate objects [page 255]
Inserting dimensions directly from the data foundation [page 268]

9.3.6.1 Refreshing an OLAP business layer

Use the *Refresh Business Layer* wizard to update a business layer based on changes in the OLAP cube since the business layer was created, or since the last refresh.

1. Open the business layer by clicking the business layer name in the Local Project View.
2. From the information design tool main menu, select *Actions* > *Refresh Structure*.
3. Follow the instructions on the wizard pages. For more information about what to do on a particular page, click the help icon.

Related Information

Refreshing business layers [page 254]

9.3.6.2 Inserting candidate objects

Before inserting candidate objects, first run a refresh structure and then synchronize tables in the data foundation.

Inserting candidate objects applies only to business layers that are based on multisource-enabled data foundations on SAP BW connections. The *Insert Candidate Objects* command detects objects that have been added to the data source since the business layer was created, or since the business layer was last updated with the *Insert Candidate Objects* command. The command detects new objects using the SAP BW strategy.

Inserting candidate objects updates the business layer independently of the data foundation. If you do not also refresh the structure and synchronize tables in the data foundation, you can introduce inconsistencies between the data foundation and business layer.

1. Open the business layer by double-clicking the business layer name in the Local Projects View.
2. Right-click the business layer name in the Business Layer pane and select *Insert Candidate Objects*.

   A list of business layer objects displays. The objects are grouped in folders based on the source tables. Candidate objects are highlighted and pre-selected.
3. Select the objects in the list to insert into the business layer.

   You can select objects that exist in the business layer. In this case, the definitions from the data source will overwrite the existing business object definitions.
4. To insert the selected objects into the business layer, click *Finish*, and save the business layer.

   The insertion can be undone using the Edit > Undo command on the main menu.

Inserting candidate objects does not detect obsolete objects in the business layer. You must find and delete obsolete objects manually.

Related Information

Refreshing a data foundation [page 171]
Synchronizing tables [page 172]
Refreshing universes based on SAP BW [page 46]

9.3.7 Index awareness

In a relational business layer, index awareness is the ability to take advantage of the indexes on key columns to improve query performance.

The objects in the business layer are based on database columns that are meaningful for querying data. For example, a Customer object retrieves the value in the customer name column of the customer table. In many databases, the customer table has a primary key (for example an integer) to uniquely identify each customer. The key value is not meaningful for reporting, but it is important for database performance.

When you set up index awareness, you define which database columns are primary and foreign keys for the dimensions and attributes in the business layer. The benefits of defining index awareness include the following:

- Joining and filtering on key columns are faster than on non-key columns.
- Fewer joins are needed in a query, therefore fewer tables are requested. For example, in a star schema database, if you build a query that involves filtering on a value in a dimension table, the query can apply the filter directly on the fact table by using the dimension table foreign key.
- Uniqueness in filters and lists of values is taken into account. For example, if two customers have the same name, the application retrieves only one customer unless it is aware that each customer has a separate primary key.

For details on how to define primary and foreign keys on business layer objects, see the related topic.

Related Information

Defining keys for dimensions and dimension attributes [page 269]
Creating an index-aware prompt [page 241]

9.3.8 Analytic functions

You use the information design tool to define analytic functions for objects in a universe in order to calculate, for example, rankings, moving sums or averages, and relative calculations.

An analytic function is a function provided by the relational database that performs an analytical task on a result set. An analytic function in a query returns, with each row in the result set, a calculation from a group of rows. The groups of rows can be ordered and partitioned.

For example, you can use analytic functions to retrieve the following results:

- The rank of a record, for example the rank of retail stores by the sales amount for last month.
- Moving sum or average, for example the average volume of sales in a three-month period.
- Displaying the same information in different contexts, for example this quarter's sales and last quarter's sales.
- Relative calculations, for example the difference between the sales of this quarter and the highest sales amount ever.

Following are some examples of analytic functions:

- Aggregation-like functions: SUM, COUNT, AVG, STDDEV, MEDIAN, VARIANCE
- Order-based functions: RANK, PERCENT_RANK, DENSE_RANK, LEAD, LAG, FIRST_VALUE, ROW_NUMBER

For a full description of the analytic functions available in your database, refer to the database documentation.

In the information design tool you can use analytic functions in the SELECT statement for measures and dimensions in the business layer, and for derived tables in the data foundation. A universe object defined with an analytic function can perform data analysis that would normally require the use of extended syntax at the report level. You might also see improved query performance because calculations are done on the server.

Related Information

Analytic functions: syntax and examples [page 258]
Analytic functions: rules, restrictions, and best practices [page 260]
Using analytical functions in a business layer object definition [page 261]
Using analytical functions in a derived table definition [page 262]

9.3.8.1 Analytic functions: syntax and examples

The generic syntax and examples of analytic functions are given to help you understand how analytic functions can be used.

The exact syntax of analytic functions varies depending on the database. Many analytic functions have the following syntax:

Function (arguments) `OVER` ([<`PARTITION BY` clause>] [<`ORDER BY` clause>] [<`ROW` or `RANGE` clause>])

Part of analytic function statement	Description
Function (arguments)	The name and arguments of the function that define the calculation.
`OVER (`	`OVER` signals that this is an analytic function. The `OVER` clause defines the data over which you want to do the calculation. It has three optional clauses.
`PARTITION BY` clause	The grouping over which the calculation is applied.
`ORDER BY` clause	The order of the results to be used in the calculation.
`ROW` or `RANGE` clause)	The interval of records used for calculation.

The `PARTITION BY` clause lets you define the groups of data over which the function will be calculated. For example:

`SELECT` employee_id, department, `COUNT`(employee_id) `OVER` (`PARTITION BY` department) `FROM` employee_table

This query returns, for each employee, the employee's department and the count of the number of employees in each department. The count is returned with every row (employee) in the result set.

employee_id	department	count
1	Marketing	2
2	Marketing	2
3	Sales	3
4	Sales	3
5	Sales	3

The `ORDER BY` clause lets you define the order in which the rows are used when applying the calculation. For example:

`SELECT` employee_id, salary, `RANK () OVER` (`ORDER BY` salary)

This query returns, for each employee, the employee's salary and the employee's overall rank by salary.

employee_id	salary	rank
3	3000	1
2	5000	2

employee_id	salary	rank
5	6000	3
4	7000	4
1	7200	5

The `ROW` or `RANGE` clause lets you define a window or interval of ordered rows to take into account when calculating the function on a given row. For example:

`SELECT` employee_id, salary, `SUM`(salary) `OVER` (`ORDER BY` salary `ROWS between unbounded preceding and current row`)

This query returns, for each employee, the employee's salary and the sum of salaries starting from the lowest salary up to and including the current employee's salary. The results are ordered by salary. The sum for the last row represents the sum of salaries for all employees.

employee_id	salary	sum
3	3000	3000
2	5000	8000
5	6000	14000
4	7000	21000
1	7200	28200

The following example uses both the `PARTITION BY` and `ORDER BY` clauses:

`SELECT` employee_id, department, salary, `RANK() OVER` (`PARTITION BY` department `ORDER BY` salary)

This query returns, for each employee, the employee's department, salary, and rank within the department, ordered by salary within the department.

employee_id	department	salary	rank
2	Marketing	5000	1
1	Marketing	7200	2
3	Sales	3000	1
5	Sales	5000	2
4	Sales	7000	3

The following example uses all three clauses:

`SELECT` employee_id, department, salary, `SUM` (salary) `OVER` (`PARTITION BY` department `ORDER BY` salary `ROWS between unbounded preceding and current row`)

This query returns, for each employee, the employee's department and salary, and the sum of salaries within the department starting from the lowest salary in the department up to and including the current employee's salary. The rows are ordered by salary within each department.

employee_id	department	salary	sum
2	Marketing	5000	5000
1	Marketing	7200	12200
3	Sales	3000	3000
5	Sales	5000	8000
4	Sales	7000	15000

Related Information

Analytic functions [page 257]

9.3.8.2 Analytic functions: rules, restrictions, and best practices

Rules, restrictions, and best practices for using analytic functions are given to help you design the universe.

The following are some rules and restrictions that apply when using analytic functions in the universe:

- Analytic functions are calculated after the joins are applied, and after the `WHERE`, `HAVING`, and `GROUP BY` clauses are applied. So, for example, the following query returns the value 1:
  `COUNT` (*) `OVER` () `FROM` employee_table `WHERE` employee_id=312
- Analytic functions cannot be used in universe conditions and sorts. Disable this usage in the Advanced tab in the object properties. You can, however, use analytic functions in conditions in derived tables.
- Analytic functions cannot appear in a `GROUP BY` clause. The query expects aggregate functions in the `GROUP BY` clause.
- Some analytic functions will not work in the same query that contains a `GROUP BY` clause. Before using analytic and aggregate functions in the same query make sure they are compatible.

The following are some best practices to observe when using analytic functions in the universe:

- In the business layer, keep measures on analytic functions separated from aggregated measures (for example, put them in a different folder).
- Label business layer objects as analytic. Put them in a separate folder or business layer view.
- In the description of the business layer object or derived table, specify any restrictions. For example, that the object cannot be used in a query with the `GROUP BY` clause (with aggregate functions), or that when using the object you should not apply filters to the query.
- If you anticipate a lot of ad-hoc queries on the universe, consider defining the analytic functionality in derived tables in the data foundation. In the business layer, expose only objects which will always work together.

Related Information

Analytic functions [page 257]
Setting where objects can be used [page 287]

9.3.8.3 Using analytical functions in a business layer object definition

To use analytic functions in the business layer, you define the analytic function in the SELECT statement for a measure or dimension.

Many analytic functions are listed in the *Database Functions* folder in the *Functions* pane of the SQL Expression Editor. If the function you want to use is not listed, you can add it by updating the extended PRM file.

> **i Note**
>
> If a function has both an aggregate and analytic version (for example SUM and SUM OVER), you must define the analytic function in the PRM file if it is not already defined.

To add an analytic function, see the procedure to verify and add analytic function support to PRM files in the *Data Access Guide*. You will need to restart the information design tool after updating the PRM file.

See the rules, restrictions, and best practices for analytic functions in the related topic.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, select the object.
3. In the object properties pane, select the *SQL Definition* tab.
4. Click the *SQL Assistant* button next to the *SELECT* statement to use the SQL editor to build the *SELECT* statement.

   Open the *Database Functions* folder in the *Functions* pane and select the desired analytic function. For help on syntax and examples, see the related topic.
5. When you have completed building the SELECT and WHERE statements for the object, save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Analytic functions: rules, restrictions, and best practices [page 260]
Analytic functions: syntax and examples [page 258]
Analytic functions [page 257]

9.3.8.4 Using analytical functions in a derived table definition

To use analytic functions in the data foundation, you define the analytic function in the SELECT statement for a derived table.

Many analytic functions are listed in the *Database Functions* folder in the *Functions* pane of the SQL Expression Editor. If the function you want to use is not listed, you can add it by updating the extended PRM file. To do this, see the procedure to verify and add analytic function support to PRM files in the *Data Access Guide*. You will need to restart the information design tool after updating the PRM file.

> **i Note**
>
> To use an analytic function in a derived table definition, updating the extended PRM file is optional, not required.

See the rules, restrictions, and best practices for analytic functions in the related topic.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Insert or edit an existing derived table based on the table that contains the object.
3. Edit the expression for the derived table to build the `SELECT` statement.

   Open the *Database Functions* folder in the *Functions* pane and select the desired analytic function. For help on syntax and examples, see the related topic.
4. When you have completed building the SELECT statement for the table, click *OK* and save the data foundation by clicking the *Save* icon in the main tool bar.

Related Information

Analytic functions: rules, restrictions, and best practices [page 260]
Analytic functions: rules, restrictions, and best practices [page 260]
Inserting and editing a derived table [page 169]
Analytic functions [page 257]

9.3.9 Aggregate awareness

Aggregate awareness is the ability of a relational universe to take advantage of database tables that contain pre-aggregated data (aggregate tables). Setting up aggregate awareness accelerates queries by processing fewer facts and aggregating fewer rows.

If an aggregate aware object is included in a query, at run time the query generator retrieves the data from the table with the highest aggregation level that matches the level of detail in the query.

For example, in a data foundation there is a fact table for sales with detail on the transaction level, and an aggregate table with sales summed by day. If a query asks for sales details, then the transaction table is used. If a query asks for sales per day, then the aggregate table is used. Which table is used is transparent to the user.

Setting up aggregate awareness in the universe has several steps. See the related topic for more information.

Related Information

Setting up aggregate awareness [page 263]

9.3.9.1 Setting up aggregate awareness

This topic outlines the steps to set up aggregate awareness in a relational universe. For links to more detailed information on each step, see the Related Topics.

1. The first step is done at the database level. The database administrator must define and load the aggregate tables into the database.

   The reliability and usefulness of aggregate awareness in a universe depends on the accuracy of the aggregate tables. They must be refreshed at the same time as all the fact tables.
2. Insert the aggregate tables into the data foundation.
3. Define aggregate aware objects. These are objects in the business layer for which you want queries to use the aggregate tables when possible instead of performing aggregation using non-aggregate tables.

   In the SQL expression for the object, define the SELECT statement to use the `@Aggregate_Aware` function:

   `@Aggregate_Aware(sum(<Aggregate table 1>), ... sum(<Aggregate table n>))`

   In the `@Aggregate_Aware` function, `<Aggregate table 1>` is the aggregate table with the highest level of aggregation, and `<Aggregate table n>` is the aggregate table with the lowest level (the detailed fact table).
4. Specify the incompatible objects for each aggregate table in the universe. In the business layer, use the *Set Aggregate Navigation* command.
5. Resolve any loops in the data foundation using contexts if needed.

Related Information

Aggregate awareness [page 262]
Inserting tables into the data foundation [page 151]
Defining the SQL expression for an object [page 284]
@Aggregate_Aware [page 490]
Setting aggregate navigation [page 264]
Resolving loops [page 187]

9.3.9.2 Setting aggregate navigation

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. From the information design tool main menu, select *Actions* > *Set Aggregate Navigation*.

   In the *Aggregate Navigation* dialog box, you specify which tables contain objects that are not compatible with aggregate tables containing objects optimized for aggregate awareness:
   - If the object is at the same level of aggregation or higher, it is compatible with the aggregate table.
   - If the object is at a lower level of aggregation, it is incompatible.
   - If the object is unrelated to the aggregate table, it is incompatible.

   > **i Note**
   >
   > A measure summed by year is at a higher level of aggregation than a measure summed by quarter.

3. Click an aggregate table in the *Data Foundation Tables* pane. You can also click the filter icon and filter on tables and their characteristics before detecting incompatibilities.
4. In the *Associated Incompatible Objects* pane, select all incompatible objects. You can also click the filter icon and filter on objects, their type, compatibility, and other characteristics before detecting incompatibilities.
5. Repeat the above steps for each aggregate table in the data foundation.

   > **i Note**
   >
   > The dialog box also features a *Detect Incompatibility* button that can guide you in the process of specifying incompatible objects. When you click a table and then click this button, objects considered as incompatible are automatically selected. Consider the incompatible objects proposed by *Detect Incompatibility* as suggestions, not final choices.

6. When all incompatible objects for all the tables are specified, click *OK*.
7. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Aggregate awareness [page 262]

9.4 Working with business layer objects

This section describes how to insert, edit, display, and search for business layer objects. You can organize the folders and objects in alphabetical order. For relational universes, you can sort objects alphabetically. In the *Contents* tab of the *Business Layer Editor*, there are two sort buttons: you can sort the objects alphabetically in the selected folder, and you can sort objects and sub-folders alphabetically in the current folder.There are two buttons to order folder children, in alphabetical order, and in reverse.

Business Layer Editor preferences [page 404]

> **i Note**
>
> When a new object is added, it is not automatically sorted.

9.4.1 Inserting a folder

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, select the folder or analysis dimension in which you want to insert the folder.

   To insert a folder at the top level, select the top node (business layer name) in the tree.
3. Click the *Insert object* icon at the top of the *Business Layer* pane and select *Folder*.
4. Edit the folder properties in the *Folder Properties* pane. See the related topics for more information about specific properties.

Property	Description
Name	The name of the folder as it appears in the Query Panel.
Description	An optional description of the folder.
State	The state of the folder, whether it is *Active*, *Hidden*, or *Deprecated*. **i Note** If the state is set to *Hidden* or *Deprecated*, the state of the objects in the folder are unchanged, however they do not appear in the Query Panel.
Content	A list of objects in the folder that lets you define properties that describe what the object is used for in the query (*for result*, *for filter*, *for sort*). You can change the order of the objects in the folder using the up and down arrow keys to the right of the list.
Custom Properties	Optional custom properties and their values.

5. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Business layer objects [page 215]
Changing the state of an object: Active, Hidden, or Deprecated [page 286]
Inserting and editing custom properties [page 295]

9.4.2 Duplicating folders in the business layer

You can now duplicate folders in the business layer. you select the folder(s) in the following ways:

- Select the folder(s), then right-click a folder and select *Duplicate*.
- Select the folder(s), then in the *Edit* menu, select *Duplicate*.
- In the main editor toolbar, click the *Duplicate* icon.

The selected folders and their contents are duplicated.

9.4.3 Inserting and editing dimensions

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, do one of the following:

Option	Command
To edit an existing dimension	Select the dimension.
To insert a dimension	Select the folder or analysis dimension in which you want to insert the dimension. To insert a dimension at the top level, select the top node (business layer name) in the tree. Click the *Insert object* icon at the top of the *Business Layer* pane and select *Dimension*. **i Note** Because of an MDX limitation, dimensions that are inserted or copied into an OLAP business layer cannot be used in conditions or sorts. The options *Object can be used in Condition* and *Object can be used in Sort* are unavailable.

3. Edit the dimension properties in the *Dimension Properties* pane. See the related topics for more information about specific properties.

Property	Description
Name	The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel. **i Note** Avoid using parentheses in dimension names. They can cause Check Integrity errors when used with the @Select function.

Property	Description
Description	An optional description of the object.
State	The state of the object, whether it is *Active*, *Hidden*, or *Deprecated*.
Data Type	The data type of the dimension.
SQL Definition or *MDX Definition*	The SQL or MDX query expression that defines the object.
Tables (relational only)	The tables associated in the query (by a join in the SQL expression) when returning values for the business layer object. To associate extra tables, click the browse button in the *Tables* field.
Hierarchy (OLAP only)	The hierarchy that is associated with the dimension. **i Note** A hierarchy must be specified if you want to insert dimension attributes.
Keys tab (relational only)	The database columns used as primary and foreign keys. Keys allow queries to take advantage of indexes on key columns. Defining keys speeds up data retrieval by optimizing the SQL that is generated for the query. For example in a star schema database if you build a query that filters on a value in a dimension table, the filter can be applied directly on the fact table by using the dimension table foreign key. This avoids inefficient joins to dimension tables.
Advanced tab	Properties that include settings for: ○ Access Levels ○ Where in query expressions the object can be used. **i Note** Because of an MDX limitation, dimensions that are inserted or copied into an OLAP business layer cannot be used in conditions or sorts. The options *Object can be used in Condition* and *Object can be used in Sort* are unavailable. ○ List of values ○ Display options
Source Information	Descriptive fields that apply to objects used by Data Integrator.
Custom Properties	Optional custom properties and their values.

4. To see the SQL query script for the dimension definition, click *Show Script*.
5. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Business layer objects [page 215]
Inserting dimensions directly from the data foundation [page 268]
Changing the state of an object: Active, Hidden, or Deprecated [page 286]
Defining the SQL expression for an object [page 284]
Defining the MDX expression for an object [page 285]
Associating extra tables [page 285]
Defining keys for dimensions and dimension attributes [page 269]
Setting object access levels [page 286]
Setting where objects can be used [page 287]
Defining custom display formats [page 289]
Source information for business layer objects [page 295]
Inserting and editing custom properties [page 295]

9.4.4 Inserting dimensions directly from the data foundation

For business layers based on a data foundation, you can drag and drop objects from the data foundation into the business layer.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View. The data foundation on which the business layer is based displays in the data source pane on the lower right side of the editing tab.
2. Select the objects in the data foundation view that you want to insert:
   - To select a table, click the table header.
   - To select multiple tables, click the table headers while holding down the `CTRL` key.
   - To select a column, click the column name in the table.
   - To select multiple columns, click the column names while holding down the `CTRL` key.
3. Drag the selection to the *Business Layer* pane and drop the selection into the desired folder in the business layer. To insert the dimensions at the top level, drop the selection into the top node (business layer name) of the tree.

   When you drag and drop a table, a folder is automatically inserted into the business layer. The folder contains a dimension for every column.

   The SQL expression for each dimension is automatically defined.
4. Save the business layer by clicking the *Save* icon in the main tool bar.

If necessary, turn any of the inserted dimensions into measures using the *Turn into Measure with Aggregation Function* command. For more information, see the related topic.

Related Information

Turning a dimension or attribute into a measure [page 273]

Inserting and editing dimensions [page 266]

9.4.5 Defining keys for dimensions and dimension attributes

Defining keys is available for dimensions and dimension attributes built on a data foundation.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Select the dimension or dimension attribute in the *Business Layer* pane.
3. In the *Dimension Properties* pane, click the *Keys* tab.
4. Add keys to the table:

Option	Description
Click *Add key*.	Adds a key row to the table.
Drag a table column from the data foundation display into the keys table.	Adds a key row to the table and a SELECT statement for the selected column.
Click *Detect*.	Detects existing key columns in the database and inserts the keys into the table.

   You can define one primary and multiple foreign keys for an object. The first key added is the primary key.

5. To edit the SELECT statement, click the *SELECT* column.

   Enter the *SELECT* statement directly and click to validate the statement, or click the SQL icon to use the SQL editor to build the statement.
6. To enter or edit the WHERE statement, click the *WHERE* column.

   Enter the *WHERE* statement directly and click to validate the statement, or click the SQL icon to use the SQL editor to build the statement.
7. Click the *Active* column to enable or disable the key.
8. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Inserting and editing dimensions [page 266]
The SQL/MDX expression editor [page 421]

9.4.6 Turning an attribute or measure into a dimension

This task applies to relational business layers.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View. The business layer objects appear in the *Business Layer* pane and the properties in the editing pane on the right.
2. Select the attribute or measure in the *Business Layer* pane.

   You can select several attributes or several measures. Click the objects while holding down the `CTRL` key. The command applies to all objects selected.
3. Right-click the selection and select *Turn into Dimension*.
   Each dimension is created in the folder of the original object.
4. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Inserting and editing dimensions [page 266]

9.4.7 Inserting and editing measures

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, do one of the following:

Option	Command
To edit an existing measure	Select the measure.
To insert a measure	Select the folder or analysis dimension in which you want to insert the measure. To insert a measure at the top level, select the top node (business layer name) in the tree. Click the *Insert object* icon at the top of the *Business Layer* pane and select *Measure*.

3. Edit the measure properties in the *Measure Properties* pane. See the related topics for more information about specific properties.

Property	Description
Name	The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.
Description	An optional description of the object.
State	The state of the object, whether it is *Active*, *Hidden*, or *Deprecated*.

Property	Description
Data Type	The data type of the object. **i Note** When the Data Type is *Numeric*, selecting the *High Precision* check box, applies the decimal number type, which implements the IEEE 754-2008 Decimal data format for big numbers.
Projection Function	Defines how further aggregation is done if needed for the report. Projection aggregation is different from, and is done after the aggregation defined for the measure in the SQL or MDX definition. For more information, see the related topic.
SQL Definition or *MDX Definition*	The SQL or MDX query expression that defines the object.
Tables (relational only)	The tables associated in the query (by a join in the SQL expression) when returning values for the business layer object. To associate extra tables, click the browse button in the *Tables* field.
MDX Properties (OLAP only)	You can enter values for the following MDX calculation and format properties to be included in the MDX query: ○ *Solve order* ○ *Format string* ○ *Scope Isolation* ○ *Language*
Advanced tab	Properties that include settings for: ○ Access Levels ○ Where in query expressions the object can be used ○ List of values ○ Display options
Source Information	Descriptive fields that apply to objects used by Data Integrator.
Custom Properties	Optional custom properties and their values.

4. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Business layer objects [page 215]
Changing the state of an object: Active, Hidden, or Deprecated [page 286]
About projection functions [page 272]
Defining the SQL expression for an object [page 284]
Defining the MDX expression for an object [page 285]
Associating extra tables [page 285]
Setting object access levels [page 286]

Setting where objects can be used [page 287]
Defining custom display formats [page 289]
Source information for business layer objects [page 295]
Inserting and editing custom properties [page 295]

9.4.7.1 Setting high precision Numeric type

For the applications supporting high precision number formats, such as the IEEE 754-2008 Decimal Floating Point format supported in SAP BusinessObjects Web Intelligence 4.2, you can check the *High Precision* property associated to the *Numeric* Data Type.

On measure objects where this property is checked, the application consuming the universe will then be able to automatically use its high precision number format, in order to improve its calculations precision.

> **i Note**
>
> The *High Precision* property is only available on measure objects of *Numeric* Data Type and only in .UNX universes.

> **→ Tip**
>
> Calculations in high precision format consume more memory and take longer to execute. You should only check this property for the measures of the business layer where high precision calculations are necessary.

Procedure

In the information design tool, you apply the *High Precision* property for a numeric measure as follows:

1. In the business layer, select a measure which Data Type is *Numeric*.
2. Check the *High Precision* check box to the right of the Data Type.
3. In the main tool bar, click the *Save* icon to save the business layer.

9.4.7.2 About projection functions

The projection function defines how to re-aggregate a measure locally for a report. Projection functions apply only to SAP BusinessObjects Web Intelligence reports.

Measures can be aggregated at two different times during the query process:

- First, when the query retrieves the data from the data source, the measure is aggregated according to the SQL or MDX definition of the measure.
- Once the data is retrieved, it is possible to change the level of aggregation in the report. For example, the query retrieves Sales aggregated for Country and City. Then in the Web Intelligence report, Sales is

reported only by Country. The projection function defines how to do the local aggregation that is needed to project the data onto the report.

If the projection function is *Sum*, the measure will be summed locally in the report. If the projection function is *Delegated*, the projection function requests that the aggregation be done in the database instead of aggregating locally.

> **i Note**
>
> You can define the count projection function's behavior in the .UNX Universes. Depending on what is defined in the .UNX Universes, you can change which count projection function is used, by using *Count with empty* values or *Count without empty* values. Select *Count without empty* to ensure compatibility with .UNV Universes that use this projection function.

Related Information

Inserting and editing measures [page 270]

9.4.8 Turning a dimension or attribute into a measure

This task applies to relational business layers.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View. The business layer objects appear in the *Business Layer* pane and the properties in the editing pane on the right.
2. Select the dimension or attribute in the *Business Layer* pane.

   You can select several dimensions or several attributes. Click the objects while holding down the `CTRL` key. The command applies to all objects selected.
3. Right-click the selection and select *Turn into Measure with Aggregation Function*.

   The aggregation functions that are valid for the data type of the dimension or attribute are listed in a submenu.

   > **i Note**
   >
   > If you selected multiple objects with different data types, the aggregation functions for all data types are available, but may not be valid for all objects in the selection.
4. Select the aggregation function for the measure, or *None*.

   The SELECT statement in the SQL definition is updated to aggregate the values using the selected function. The data type of the resulting measure is automatically changed, if appropriate. For example, if the original object is of type *DateTime*, and you turn this into a measure with the aggregation function *Count*, the resulting measure has a data type of *Numeric*.

   The projection function is automatically set according to the aggregation function selected:

Aggregation Function	Projection Function
Sum	*Sum*
Count	*Sum*
Max	*Max*
Min	*Min*
Average	*Delegated*
None	*Delegated*

For more information about projection functions, see the related topic.

You receive a message listing any objects for which the selected aggregation function is invalid. In this case, the object is turned into a measure, but the new aggregation function is ignored and the projection function is set to *Delegated*.

> **i Note**
>
> If any dimension you select is included in a navigation path, you receive a warning that the navigation path will be impacted by the change. If you proceed to turn the dimension into a measure, the dimension is automatically removed from the navigation path.

5. Save the business layer by clicking the *Save* icon in the main tool bar.

You can change the aggregation function at any time by editing the SELECT statement for the measure. If necessary, select another projection function directly in the *Projection Function* dropdown list.

Related Information

Inserting and editing measures [page 270]
Navigation paths for objects [page 297]
About projection functions [page 272]

9.4.9 Inserting and editing attributes

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, do one of the following:

Option	Command
To edit an existing attribute	Select the attribute.
To insert an attribute	Select the dimension, measure, hierarchy, or level in which you want to insert the attribute.

Option	Command
	i Note In an OLAP business layer, when inserting an attribute under a measure, select a measure that was inserted manually into the business layer. The attribute is called *Formatted Value Attribute*. Click the *Insert object* icon at the top of the *Business Layer* pane and select *Attribute* or *Formatted Value Attribute*.

3. Edit the attribute properties in the *Attribute Properties* pane. See the related topics for more information about specific properties.

Property	Description
Name	The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.
Description	An optional description of the object.
State	The state of the object, whether it is *Active*, *Hidden*, or *Deprecated*.
Data Type	The data type of the object.
SQL Definition or *MDX Definition*	The SQL or MDX query expression that defines the object.
Tables (relational only)	The tables associated in the query (by a join in the SQL expression) when returning values for the business layer object. To associate extra tables, click the browse button in the *Tables* field.
Keys tab (relational dimension attributes only)	The database columns used as primary and foreign keys. Keys allow queries to take advantage of indexes on key columns. Defining keys speeds up data retrieval by optimizing the SQL that is generated for the query.
Advanced tab	Properties that include settings for: ○ Access Levels ○ Where in query expressions the object can be used ○ List of values ○ Display options
Source Information	Descriptive fields that apply to objects used by Data Integrator.
Custom Properties	Optional custom properties and their values.

4. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Business layer objects [page 215]
Changing the state of an object: Active, Hidden, or Deprecated [page 286]
Defining the SQL expression for an object [page 284]
Defining the MDX expression for an object [page 285]
Associating extra tables [page 285]
Defining keys for dimensions and dimension attributes [page 269]
Setting object access levels [page 286]
Setting where objects can be used [page 287]
Defining custom display formats [page 289]
Source information for business layer objects [page 295]
Inserting and editing custom properties [page 295]

9.4.10 Turning a dimension or measure into an attribute

This task applies to relational business layers.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View. The business layer objects appear in the *Business Layer* pane and the properties in the editing pane on the right.
2. Select the dimension or measure in the *Business Layer* pane.

   You can select several dimensions or several measures. Click the objects while holding down the `CTRL` key. The command applies to all objects selected.
3. Right-click the selection and select *Turn into Attribute*.
4. In the *Select a Parent Dimension or Measure* dialog box, select the dimension or measure to which the attributes belong.
5. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Inserting and editing attributes [page 274]

9.4.11 Inserting and editing filters

You can insert and edit filters. You can define filters as mandatory. When a filter is mandatory, it is identified by one of three possible icons:

- Mandatory filter applied to a universe
- Mandatory filter applied on a folder

- Mandatory filter applied on a set applied on a universe

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, do one of the following:

Option	Command
To edit an existing filter	Select the filter.
To insert a filter	Select the folder or analysis dimension in which you want to insert the filter. To insert a filter at the top level, select the top node (business layer name) in the tree. Click the *Insert object* icon at the top of the *Business Layer* pane and select *Filter*.

3. Edit the filter properties in the *Filter Properties* pane. See the related topics for more information about specific properties.

Property	Description
Name	The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.
Description	An optional description of the object.
State	The state of the object, whether it is *Active*, *Hidden*, or *Deprecated*.
Filter Type	Either *Native* (relational business layers only), or *Business*.
SQL Definition	For native filters, the SQL WHERE expression that defines the object.
Tables (relational only)	The tables associated in the query (by a join in the SQL expression) when returning values for the business layer object. To associate extra tables, click the browse button in the *Tables* field.
Filter Definition	For business filters, click *Edit Filter* to define the filter based on objects in the business layer. See the related topic about building a business filter.
Properties tab	When the *Use filter as mandatory in query* option is selected, the filter is applied to every query using any object in either the universe or the folder, depending on the selected scope (*Apply on universe* or *Apply on folder*). When the *Apply on list of values* option is selected, the filter is applied on list-of-value queries. When *Use filter as mandatory in query* is unselected, the filter is applied only when explicitly added to the query.
Custom Properties	Optional custom properties and their values.

4. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Business layer objects [page 215]
Changing the state of an object: Active, Hidden, or Deprecated [page 286]
Defining the SQL expression for an object [page 284]
Associating extra tables [page 285]
How to build a business filter [page 310]
Inserting and editing custom properties [page 295]

9.4.12 Inserting and editing analysis dimensions

Analysis dimensions can be inserted into OLAP business layers only.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, do one of the following:

Option	Command
To edit an existing analysis dimension	Select the analysis dimension.
To insert an analysis dimension	Select the business layer name or folder in which you want to insert the analysis dimension. Click the *Insert object* icon at the top of the *Business Layer* pane and select *Analysis Dimension*.

3. Edit the analysis dimension properties in the *Analysis Dimension Properties* pane. See the related topics for more information about specific properties.

Property	Description
Name	The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.
Description	An optional description of the object.
State	The state of the object, whether it is *Active*, *Hidden*, or *Deprecated*. **i Note** If the state is set to *Hidden* or *Deprecated*, the state of the objects in the analysis dimension are unchanged, however they do not appear in the Query Panel.
Type	This property is not currently used.

Property	Description
Default hierarchy	The hierarchy that is taken as the default when the entire analysis dimension is added as a result object in the Query Panel.
Key attribute	This property is not currently used.
Custom Properties	Optional custom properties and their values.

4. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Business layer objects [page 215]
Changing the state of an object: Active, Hidden, or Deprecated [page 286]
Inserting and editing custom properties [page 295]

9.4.13 Inserting and editing hierarchies

Hierarchies can be inserted into OLAP business layers only.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, do one of the following:

Option	Command
To edit an existing hierarchy	Select the hierarchy.
To insert a hierarchy	Select the folder or analysis dimension in which you want to insert the hierarchy. To insert a hierarchy at the top level, select the top node (business layer name) in the tree. Click the *Insert object* icon at the top of the *Business Layer* pane and select *Hierarchy*.

3. Edit the hierarchy properties in the *Hierarchy Properties* pane. See the related topics for more information about specific properties.

Property	Description
Name	The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.
Description	An optional description of the object.

Property	Description
State	The state of the object, whether it is *Active*, *Hidden*, or *Deprecated*.
MDX Definition	The MDX query expression that defines the object.
Advanced tab	Properties that include settings for: ○ Access Levels ○ Where in query expressions the object can be used ○ List of values ○ Display options
Source Information	Descriptive fields that apply to objects used by Data Integrator.
Custom Properties	Optional custom properties and their values.

4. Save the business layer by clicking the *Save* icon in the main tool bar.

The following objects can be added to the hierarchy:

- Levels
- Attributes
- Named sets
- Calculated members

Related Information

Business layer objects [page 215]
Changing the state of an object: Active, Hidden, or Deprecated [page 286]
Defining the MDX expression for an object [page 285]
Setting object access levels [page 286]
Setting where objects can be used [page 287]
Defining custom display formats [page 289]
Source information for business layer objects [page 295]
Inserting and editing custom properties [page 295]
Inserting and editing hierarchy levels [page 280]
Inserting and editing attributes [page 274]
Inserting and editing named sets [page 282]
Inserting and editing calculated members [page 283]

9.4.14 Inserting and editing hierarchy levels

Levels can be inserted into hierarchies in OLAP business layers only.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.

2. In the *Business Layer* pane, do one of the following:

Option	Command
To edit an existing level	Select the level.
To insert a level	Select the hierarchy in which you want to insert the level. Click the *Insert object* icon at the top of the *Business Layer* pane and select *Level*.

3. Edit the level properties in the *Level Properties* pane. See the related topics for more information about specific properties.

Property	Description
Name	The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.
Description	An optional description of the object.
State	The state of the object, when it is *Active*, *Hidden*, or *Deprecated*.
Business Type	This property is not currently used.
MDX Definition	The MDX query expression that defines the object.
Advanced tab	Properties that include settings for: ○ Access Levels ○ Where in query expressions the object can be used ○ List of values ○ Display options
Source Information	Descriptive fields that apply to objects used by Data Integrator.
Custom Properties	Optional custom properties and their values.

Related Information

Business layer objects [page 215]
Changing the state of an object: Active, Hidden, or Deprecated [page 286]
Defining the MDX expression for an object [page 285]
Setting object access levels [page 286]
Setting where objects can be used [page 287]
Defining custom display formats [page 289]
Source information for business layer objects [page 295]
Inserting and editing custom properties [page 295]

9.4.15 Inserting and editing named sets

Named sets can be inserted into OLAP business layers only.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, do one of the following:

Option	Command
To edit an existing named set	Select the named set.
To insert a named set	Select the folder, analysis dimension, or hierarchy in which you want to insert the named set. To insert a named set at the top level, select the top node (business layer name) in the tree. Click the *Insert object* icon at the top of the *Business Layer* pane and select *Named Set*.

3. Edit the named set properties in the *Named Set Properties* pane. See the related topics for more information about specific properties.

Property	Description
Name	The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.
Description	An optional description of the object.
State	The state of the object, whether it is *Active*, *Hidden*, or *Deprecated*.
Hierarchy	The hierarchy for the named set.
Named Set Type	The named set type: ○ *Native* named sets are defined with an MDX expression. ○ *Business* named sets are defined by selecting members using the *Member Selector*.
MDX Definition	For native named sets, The MDX query expression that defines the set.
Definition tab	For business named sets, the list of members. To select members: 1. Select a hierarchy from the *Hierarchy* list. 2. Click *Edit Members*. 3. In the *Member Selector*, select or unselect members from the given hierarchy to be included or excluded in the named set. For more information on using the *Member Selector*, see the related topic.

4. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Business layer objects [page 215]
Changing the state of an object: Active, Hidden, or Deprecated [page 286]
Defining the MDX expression for an object [page 285]
Working with the Member Selector [page 316]

9.4.16 Inserting and editing calculated members

Calculated members can be inserted into OLAP hierarchies only.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, do one of the following:

Option	Command
To edit an existing calculated member	Select the calculated member.
To insert a calculated member	Select the hierarchy in which you want to insert the calculated member. Click the *Insert object* icon at the top of the *Business Layer* pane and select *Calculated Member*.

3. Edit the calculated member properties in the *Calculated Member Properties* pane. See the related topics for more information about specific properties.

Property	Description
Name	The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.
Description	An optional description of the object.
State	The state of the object, whether it is *Active*, *Hidden*, or *Deprecated*.
Hierarchy	The hierarchy for the calculated member.
Parent member	The level in the hierarchy under which the calculated member is to appear. If not specified, the member appears at the root level.
Expression	The MDX expression that defines the calculated member. For more information, see the related topic.

Property	Description
MDX Properties	You can enter values for the following MDX calculation and format properties to be included in the MDX query: ○ *Solve order* ○ *Format string* ○ *Scope Isolation* ○ *Language*

4. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Business layer objects [page 215]
Changing the state of an object: Active, Hidden, or Deprecated [page 286]
Defining the MDX expression for an object [page 285]

9.4.17 Defining the SQL expression for an object

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, select the object.
3. In the object properties pane, select the *SQL Definition* tab.
4. Enter the *SELECT* statement directly and click to validate the SELECT statement, or click the *SQL Assistant* button to use the SQL editor to build the statement.

   Most measures require an SQL aggregation function to be defined in the SELECT expression, for example: `sum(efashion."Shop_facts"."Amount_sold")`.

   At query run time, the aggregation defined for the measure in the SQL is done before the projection aggregation. You define the projection function separately. For more information, see the related topic.
5. Enter the *WHERE* statement directly and click to validate the SELECT statement, or click the *SQL Assistant* button to use the SQL editor to build the statement.
6. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

The SQL/MDX expression editor [page 421]
About projection functions [page 272]
Using analytical functions in a business layer object definition [page 261]

9.4.18 Defining the MDX expression for an object

You can edit the MDX expression for objects inserted into the business layer. If you want to edit the definition of a native object (an object generated automatically from the cube when the business layer was created), copy the native object and edit the copy.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, select the object.
3. In the object properties pane, select the *MDX Definition* tab.
4. Enter the *Expression* directly and click to validate the expression, or click the *MDX Assistant* button to use the MDX editor to build the statement.

   > **i Note**
   >
   > When inserting a dimension or level, the best practice is to include `.members` in the MDX expression. For example, following is the expression if you insert the dimension Category in the Category hierarchy in the Product analysis dimension:
   >
   > ```
   > [Product].[Category].[Category].members
   > ```

   For some data sources, it is necessary to add `.members` in order to preview members.
5. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

The SQL/MDX expression editor [page 421]

9.4.19 Associating extra tables

The *Associated SQL Tables* dialog box lets you associate tables with an object in the business layer.

All associated tables are included in the query (by a join in the SQL expression) when returning values for the business layer object. For example, if the object is based on City in table City, when you associate Region and Country as extra tables, the City values from Region and Country are included when values are returned for the business object.

Tables that can be associated in the SQL are proposed in the list.

1. To include values from an associated table, check the box next to the table name.
2. To stop including values from a table, uncheck the box next to the table name.

   > **i Note**
   >
   > The table on which the business object is based is displayed in bold and cannot be unchecked.

   When a table is explicitly added as an extra table to an object, if the table is also used in the object's SELECT or WHERE clauses and is implicitly available in the object definition, then the table is kept in the list

of associated tables. When the table is removed from SELECT or WHERE clause, it is also removed from the list of associated tables.

Related Information

Inserting and editing dimensions [page 266]
Inserting and editing measures [page 270]
Inserting and editing attributes [page 274]

9.4.20 Changing the state of an object: Active, Hidden, or Deprecated

Objects in the business layer can have three states:

- *Active*: Object is visible in the Query Panel. This is the default state.
- *Hidden*: Object is valid but not available in the Query Panel (used by other objects as a hidden object).
- *Deprecated*: Object is hidden and not valid. One possible use for this state is when the target database field no longer exists, but you want to keep the object for possible future use.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View. The business layer objects appear in the *Business Layer* pane and the properties in the editing pane on the right.
2. Select the object in the *Business Layer* pane.

   To select several objects, click while holding down the `CTRL` key.
3. Right-click the selection and select *Change State*.
4. Select the new state.

   The new state is applied to all the objects in the selection. If the state is set to *Hidden* or *Deprecated* for a folder or analysis dimension, the state of the objects in the folder are unchanged, however they do not appear in the Query Panel.
5. Save the business layer by clicking the *Save* icon in the main tool bar.

9.4.21 Setting object access levels

The security access level of an object restricts use of the object to users with the appropriate object-access level granted. You can assign the following access levels to an object:

- *Public*
- *Private*
- *Controlled*
- *Restricted*
- *Confidential*

If you assign *Public* then all users can see and use the object. If you assign *Restricted*, then only users that are granted the object-access level of *Restricted* or higher can see and use the object in the Query Panel.

Universe object-access levels are granted to users and groups in the Central Management Console. For more information, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, select the object for which you want to set the access level. To select multiple objects, click while holding down the `CTRL` key.
3. Right-click the selection and select *Change Access Level* and select the new access level from the list.

   You can also set the access level for an object on the *Advanced* tab of the object properties.
4. Save the business layer by clicking the *Save* icon in the main tool bar.

9.4.22 Setting where objects can be used

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, select the object.
3. In the object properties pane, click the *Advanced* tab.
4. Select or deselect where the object can be used:

Option	Description
Results	When selected, the object can be used in a query.
Conditions	When selected, the object can be used to set in a condition. **Note** Because of an MDX limitation, this option is not available for dimensions that are inserted or copied in an OLAP business layer.
Sort	When selected, returned values can be sorted. **Note** Because of an MDX limitation, this option is not available for dimensions that are inserted or copied in an OLAP business layer.

5. Save the business layer by clicking the *Save* icon in the main tool bar.

9.4.23 Setting options for the default list of values

Dimensions, measures, attributes, and hierarchies are associated with a default list of values. You can set options for the default list of values, or associate a custom list of values with the object.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, select the object.

3. In the object properties pane, click the *Advanced* tab.
4. Select or deselect list of values options:

Option	Description
Force users to filter values before use	If selected, the user running a query using this list of values is required to enter search criteria before getting filtered values for the list of values. Only the values that match the search criteria are returned in the list of values. Characters used to define the matching criteria are: ○ * - Matches any number of characters, even zero characters. ○ ? - Matches exactly one character. ○ \ - Escapes the next character allowing you to search for a wildcard character.
Allow users to search values in the database	If selected, the user running a query using this list of values can search for a value in the database. This option is useful when the user performs a search on partial list of values results.

5. To associate a custom list of values, see the related topic. The options defined in the custom list of values apply.
6. Save the business layer by clicking the *Save* icon in the main tool bar.

Related Information

Publish/retrieve options for the business layer [page 416]

9.4.24 Creating and editing display formats for business layer objects

You can customize display formats for business layer objects with data types of DateTime and Numeric. There are pre-defined formats available to choose from, or you can create your own custom format using the Format Editor.

> **i Note**
>
> When you save the business layer, the custom formats you created for objects in that business layer are saved in the Custom category in the Format Editor. The formats are available to other business layers currently open in the information design tool.
>
> When opening a new information design tool session, to make custom formats available to other business layers, open the business layer in which the formats were defined.
>
> Display formats can be created, edited, and deleted for multiple business layer objects at once.

To create a display format

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, select one or more objects, right-click the selection and select *Create Display Format*.
   If a display format was already created for an object, the available command is *Edit Display Format*.
3. Using the *Data* tab of the Format Editor, you can select a pre-defined format or define a custom format:
   - To select a pre-defined format for the objects, select a Format Category (*Date-Time*, *Numeric*, or *Custom*) and select an available format in the list.
   - To define a custom format, follow the task in the related topics.

   > **i Note**
   >
   > The Format Editor also displays tabs: *Alignment*, *Border*, *Shading*, and *Font*. The information on these tabs is not taken into account by the information design tool.

4. Click *OK* to use the selected format for the objects.
5. Save the business layer by clicking the *Save* icon in the main tool bar.

Deleting display formats

The *Delete Display Format* command is available for objects with a display format already created. Use this command to unselect the display format previously selected for the object.

> **i Note**
>
> Deleting the display format in the business layer does not delete a custom format definition. You must delete a custom format from the Format Editor.

Related Information

Defining custom display formats [page 289]

9.4.24.1 Defining custom display formats

1. In the Format Editor, select an available format to use as a basis for the custom format and click *Custom Format*.

   A custom format consists of text and tokens. A token is a pre-formatted part of a number or date. For example, *Day: 1-31* is a token that displays the day part of the date as a number between 1 and 31. For more information about tokens, see the related topics.

   The Custom Format Editor lists the token categories. Open the categories to see the list of tokens.

2. Drag tokens from the Tokens list into the *Format Definition*.

   Tokens appear in the format definition with a rectangle border and a gray background. You can also type text directly into the *Format Definition*.

   When defining a numeric format, you can enter a different format to be displayed when the value is negative or equal to zero. If you do not enter a format, the format defined for positive values is used.
3. In the *Undefined* box, you can enter text to display if no value is returned at reporting time. By default, in the case of an undefined value, no text is displayed.
4. To define a display color for a format, click the color choice box at the end of the format definition.
5. To save the custom format, click *OK*.

Related Information

Date and time format tokens [page 290]
Number format tokens [page 293]
Creating and editing display formats for business layer objects [page 288]

9.4.24.1.1 Date and time format tokens

Example: Date and time format display

This example shows how the date, Wednesday March 5th 2008, is displayed using different formats defined in the Custom Format Editor.

Format defined with tokens:	Preview display:
[Day name] , [Month name] [Day 01-31] [Year 0000-9999]	Wednesday, March 05 2008
[Month 01-12] / [Day 01-31] / [Year 0000-9999]	03/05/2008
[Capitalized short day name] [Day 01-31] [Capitalized short month name]	Wed 05 Mar
[Day name], week [Week of year 01-53]	Wednesday, week 10
The current date is [Day name], [Month name] [Day 01-31] [Year 0000-9999] . Day name is [Upper case day name] . Month name is [Lower case month name]. The year is [Year 00-99] .	The current date is Wednesday, March 05 2008. Day Name is WEDNESDAY. Month name is march. The year is 08.

List of date and time tokens

Category	Token	Description
Day	*Day 01-31*	Day in the month with two digits from 01 to 31.
	Day 1-31	Day in the month with one or two digits from 1 to 31.
	Day name	Day name according to the locale, for example, Monday.
	Short day name	Short day name with capitalization according to the locale, for example, Mon.
	Day of year 001-366	Day in the year with three digits from 001 to 366.
	Day of year 01-366	Day in the year with two or three digits from 01 to 366.
	Day of year 1-366	Day in the year with one, two, or three digits from 1 to 366.
	Day of week in month	Day of the week in the month according to the locale, for example, 3 for the 3rd Monday of June.
	Upper-case day name	Day name in upper case, for example, MONDAY.
	Lower-case day name	Day name in lower case, for example, monday.
	Capitalized day name	Capitalized day name, for example, Monday.
	Upper-case short day name	Short day name in upper case, for example, MON.
	Lower-case short day name	Short day name in lower case, for example, mon.
	Capitalized short day name	Capitalized short day name, for example, Mon.
Month	*Month 01-12*	Month in the year with two digits from 01 to 12.
	Month 1-12	Month in the year with one or two digits from 1 to 12.
	Month name	Month name with capitalization according to the locale, for example, June.
	Short month name	Short month name with capitalization according to the locale, for example, Jun.
	Upper-case month name	Month name in upper case, for example, JUNE.
	Lower-case month name	Month name in lower case, for example, june.
	Capitalized month name	Capitalized month name, for example, June.
	Upper-case short month name	Short month name in upper case, for example JUN.
	Lower-case short month name	Short month name in lower case, for example, jun.
	Capitalized short month name	Capitalized short month name, for example, Jun.
Year and Era	*Year 00-99*	Year with two digits from 00 to 99.
	Year 0000-9999	Year with four digits from 0000 to 9999.
	Japanese Imperial period and year	Japanese Imperial period and year number, for example, 平成 20.
	Japanese Imperial period (English) and year	Japanese Imperial period (English abbreviated) and year number, for example, *H20*.

Category	Token	Description
	Japanese Imperial year number 01-99	Japanese Imperial year number with two digits.
	Japanese Imperial year number 1-99	Japanese Imperial year number with one or two digits.
	Japanese Imperial period	Japanese Imperial period.
	Japanese Imperial year	Deprecated. Returns the same result as *Japanese Imperial year number 0-99* token.
	Era	Era abbreviation, for example, AD or BC.
Week	*Week of month*	Week in the month with one digit from 1 to 6.
	Week of year 01-53	Week in the year (ISO week) with two digits from 01 to 53.
	Week of year 1-53	Week in the year (ISO week) with one or two digits from 1 to 53.
	Year of week of year 0000	ISO year number (consistent with ISO week) with four digits from 0000 to 9999.
	Year of week of year 00	ISO year number (consistent with ISO week) with two digits from 00 to 99.
Quarter and Semester	*Quarter number 1-4*	Quarter number with one digit from 1 to 4.
	Short quarter name	Quarter short name from Q1 to Q4.
	Quarter name	Quarter name from 1st quarter to 4th quarter.
	Semester 1-2	Semester number from 1 to 2.
Hour	*Hour 00-23*	Hour in 24-hour format with two digits from 00 to 23.
	Hour 0-23	Hour in 24-hour format with one or two digits from 0 to 23.
	Hour 01-12	Hour in 12-hour format with two digits from 01 to 12.
	Hour 1-12	Hour in 12-hour format with one or two digits from 1 to 12.
	Hour 01-24	Hour in 24-hour format with two digits from 01 to 24.
	Hour 1-24	Hour in 24-hour format with one or two digits from 1 to 24.
	Hour 00-11	Hour in 12-hour format with two digits from 00 to 11.
	Hour 0-11	Hour in 12-hour format with one or two digits from 0 to 11.
Minute	*Minutes 00-59*	Minutes with two digits from 00 to 59.
	Minutes 0-59	Minutes with one or two digits from 0 to 59.
Second and subsecond	*Seconds 00-59*	Seconds with two digits from 00 to 59.
	Seconds 0-59	Seconds with one or two digits from 0 to 59.
	Milliseconds 000-999	Milliseconds with three digits from 000 to 999.
	Hundredths of a second 000-999	Hundredths of a second with two digits from 00 to 99.
	Tenths of a second 0-9	Tenths of a second with one digit from 1 to 9.

Category	Token	Description
Time Zone	*Time zone*	The offset from Coordinated Universal Time, for example, GMT+00:00.
AM/PM	*AM/PM*	Morning/afternoon abbreviation, capitalized according to locale, for example, AM or PM. Recommended.
	Upper-case AM/PM	Morning/afternoon abbreviation in upper-case, for example, AM or PM.
	Lower-case am/pm	Morning/afternoon abbreviation in lower-case, for example, am or pm.
	Capitalized Am/Pm	Capitalized morning/afternoon abbreviation, for example, Am or Pm. Not recommended.
Separator	*Date separator*	Deprecated. This token was used as a date separator in Desktop Intelligence and is not recommended. Type the character you wish to use as a date separator directly into the format description, or use a default format.
	Time separator	Deprecated. This token was used as a time separator in Desktop Intelligence and is not recommended. Type the character you wish to use as a time separator directly into the format description, or use a default format.

9.4.24.1.2 Number format tokens

Number format definitions

A number format definition is made of sections:

- the sign (optional)
- the integer value before the decimal separator
- a grouping separator, to be added in the integer value
- the decimal separator (optional)
- the decimal value after the decimal separator (optional)
- the exponential symbol followed by the exponential value (optional)

Two tokens are used to define the number of significant digits to display in the integer, decimal, and exponential values. Each token in the format definition represents a digit to display:

- The mandatory digit token, *0*, displays the digit if it is significant, otherwise displays a zero.
- The optional digit token, *#*, only displays the digit if it is significant.

When determining the significant digits, the integer value and exponential value are evaluated from right to left, and the decimal value is evaluated from left to right. The last *0* or *#* token is mapped to the remaining digits, if any.

Example: Number format display

This example shows how the value -1,234 is displayed using different formats defined in the Format Editor.

Format defined with tokens:	Preview display:
[Sign] [#]	-1234
[Neg. start] [0] [0] [0] [0] [0] [0] [Neg. end]	(001234)
[Sign always] [#] [Dec.Sep.] [0] [0]	-1234.00
[Sign] [#] [Decimal separator] [0] [0] [E+] [0] [0] [0]	-1.23E+003
Revenue: [Sign always] [#] [Decimal separator] [0] [0]	Revenue: -1234.00 €
[Boolean]	true

List of number format tokens

Category	Token	Description
Signs	*Sign*	Negative sign if the value is negative. Nothing if the value is positive or zero.
	Sign always	Negative sign if the value is negative. Positive sign if the value is positive or zero.
	Negative start	Open parenthesis if the value is negative. Nothing if the value is positive or zero.
	Negative end	Closing parenthesis if the value is negative. Nothing if the value is positive or zero.
Digits	*#*	Optional digit. Displays digit only if significant.
	0	Mandatory digit. Displays digit if significant, otherwise displays zero.
Separators	*Decimal separator*	The symbol used to separate the integer and decimal parts of the number. The symbol used is determined by the locale. The decimal separator can be used only once in an expression.
	Grouping	By default, digits are grouped using the rule and the separator defined by the locale. The grouping symbol can be used only once in an expression. It must appear before the decimal separator.
Exponents	*E+*	Exponent sign in upper case, always signed. Can be used only once in an expression.
	E-	Exponent sign in upper case, signed only if the value is negative. Can be used only once in an expression.
	e+	Exponent sign in lower case, always signed. Can be used only once in an expression.
	e-	Exponent sign in lower case, signed only if the value is negative. Can be used only once in an expression.
Percent	*Percent*	The value multiplied by 100.

Category	Token	Description
	Percent %	The value multiplied by 100 followed by the percent sign (%). Can be used only once in an expression.
Boolean	*Boolean*	Localized value of true if the numerical value is not zero; localized value of false if the numerical value is zero.
	True	Always displays the localized value of true.
	False	Always displays the localized value of false.

9.4.25 Source information for business layer objects

The *Source Information* tab in the business layer object properties contains information about universes generated from Data Integrator. Technical descriptions and formulas used to calculate target tables are displayed.

Property	Description
Technical Information	Information about a column, for example the original database name of the concerned column for the object.
Mapping	Initial formula information describing how a column has been specified (used in Data Integrator), for example revenue = column calculated from several sources.
Lineage	Source columns for the formula used to calculate the column in the database.

9.4.26 Inserting and editing custom properties

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the *Business Layer* pane, select the object for which you want to insert or edit custom properties
3. In the object properties pane, select the *Custom Properties* tab.
4. To add a custom property, click *Add*.
5. Edit the property object name and value by clicking the column in the list.
6. To delete a property, select it in the list and click *Delete*.
7. Save the business layer by clicking the *Save* icon in the main tool bar.

9.4.27 Showing associated objects

For relational business layers, you can show the objects in the business layer that reference selected data foundation tables and columns.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.

2. In the data foundation view pane, select the tables or columns for which you want to see the associated business layer objects. To select a table, click the table header. To select a column, click the column name. To select multiple objects, click while holding down the CTRL key.
3. Right-click the selection and select *Show Associated Objects*.
   All objects that reference the selected data foundation objects are highlighted in the business layer.

Related Information

Resource dependencies [page 395]

9.4.28 Showing business layer object values

You can show the values in the underlying data source for a business layer object.

Note that for relational business layers:

- If the object references columns in the data foundation for which a filter is defined, the filters are applied.
- You can also show table and column values from the data foundation view in the Business Layer Editor.

The show values command by default opens a tab in the editor to display the values. You can set a preference to have the values open in a dedicated view or a dialog box. For more information, see the related topic.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Right-click an object in the *Business Layer* pane and select *Show Values*.

   The show values window appears. To see what you can do in this window, see the related topic on showing values in a data source.

Related Information

Showing values in a data source [page 162]
Showing table values [page 162]
Showing column values [page 163]
Preferences for the Show Values command [page 417]

9.4.29 Filtering the business layer

To filter the list of objects displayed in the business layer tree list, click the *Filtering Options* button in the *Business Layer* side panel.

This opens a box where you can select to filter the objects by:

- Name
- Object type
- Data type
- State
- Object
- Security

9.4.30 Searching for business layer objects

The search panel in the *Business Layer* pane displays the results of a search. All object-contextual commands available in the *Business Layer* pane are also available in the search panel.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.

   The *Business Layer* browsing pane displays a tree view of the objects in the business layer.

2. Click the *Show/Hide Search Panel* icon at the top of the *Business Layer* browsing pane.
   The *Search Objects* panel opens below the tree view of the business layer.

3. Select the object types to include in the search. Click the filter icon in the *Search Objects* panel. Select the types to include or exclude.
   The *Search Objects* panel displays only objects of the types selected.

4. To search the list, click the *Show/Hide search bar* icon .
5. In the search text box, enter the text and press the `Enter` key to start the search.
   The first object that contains the search text is highlighted in the *Search Objects* and *Business Layer* panels. The total number of objects containing the search text displays in the search text box.
6. To highlight the next object found, press the `Enter` key again. Use the `Enter` key to browse through all objects that match the search text.

7. When you have finished searching, click the *Show/Hide Search Panel* icon again to hide the *Search Objects* panel.

9.4.31 Navigation paths for objects

A Navigation path is an object that defines the drill path used in SAP BusinessObjects reporting tools. A drill path is a list of drillable business objects that allow a report analyst to drill down on a dimension.

A navigation path object can be one of two types:

Navigation path type	Description
Default	The path is defined by the hierarchical organization of the business objects in the business layer. If the business layer contains analysis dimensions, the navigation paths include the dimensions under each analysis dimension. Otherwise, the navigation paths are the dimensions under each folder. You can view the default navigation path in the *Navigation Paths* tab of the *Business Layer Editor*. The default path cannot be edited.
Custom	You define the path based on the available dimensions.

Related Information

Inserting a navigation path object into a business layer [page 298]
Reordering objects in the Business Layer Editor [page 299]

9.4.31.1 Inserting a navigation path object into a business layer

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Click the *Navigation Paths* tab under the *Business Layer* pane.
3. Select *Custom* at the top of the *Navigation Paths* pane.
4. Click the *Insert Navigation Path* icon .
5. Enter a *Name* and optionally a *Description* for the path.

   The name and description are available to display in the query and reporting tools that use the published universe.
6. Click *Add* to select dimensions for the path. Use the up and down arrow keys to change the order of dimensions in the list.
7. Save the business layer.

Related Information

Navigation paths for objects [page 297]

9.4.32 Duplicating business objects

9.4.33 Reordering objects in the Business Layer Editor

You can reorder lists of values, parameters, queries, and custom navigation paths in the *Business Layer Editor*. The order is saved in the business layer and the universe outline in the Query Panel. For example, when selecting a parameter to drag it to the *Filter Objects* pane, the parameter objects are presented in the custom order. The custom order in the business layer does not affect the order of prompts when querying.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Click the tab in the *Business Layer* pane depending on the objects you want to reorder: *Parameters and Lists of Values*, *Queries*, or *Navigation Paths*.

   To reorder navigation paths, select *Custom*.
3. Drag and drop the objects into the order that you want them to be listed.

   For example, in the *Parameters* pane, drag and drop the parameter names into the desired order.

   Inherited lists of values and parameters cannot be reordered in the *Business Layer Editor*. You must reorder them in the *Data Foundation Editor*. The inherited objects are listed after the business layer objects in their custom order.
4. Save the business layer by clicking the *Save* icon in the main tool bar.

> **i Note**
>
> The A-Z sort direction affects the display only in the editor and is lost if you disable sort or close the editor. By contrast, the custom order that you establish by dragging and dropping objects is kept even after you have closed the editor. To restore the custom sort order, click the *Sort direction* icon and select *Disable sort*.

To learn more about the *Business Layer Editor,* see the Business Layer Editor preferences [page 404] page.

Related Information

Business Layer Editor preferences [page 404]

10 Publishing resources and retrieving published universes

This chapter describes how to publish resources, connections and universes. It also describes how to retrieve published universes from the local file system, or from a repository.

Related Information

Publishing resources [page 300]
Publishing a local connection to the repository [page 301]
Publishing a local universe to the repository [page 302]
Publishing a universe [page 302]
Retrieving a published universe from the local file system [page 305]
Retrieving a published universe from a repository [page 305]

10.1 Publishing resources

Publication is the last step in the universe creation process. Using the Publish Universe Wizard, you publish a business layer to either your local file system or a repository.

When you publish a business layer, the wizard exports the business layer and the resources it references (local connection, connection shortcuts, and data foundation), and creates a universe which is then available to users of query, reporting and analysis tools.

> **i Note**
>
> When you make changes to a local connection and republish your resource, the changes made in the local connection are not reflected in the connection editor: You must refresh (close and reopen) the connection editor for the changes to be taken into account.

Publishing locally

Only business layers built on local connections can be published locally. This can be a business layer based on a local OLAP connection, or a business layer based on a single-source data foundation with a local connection.

The published universe is saved in the local file system folder that you specify.

Publishing to a repository

To secure a universe, you must publish it first to a repository on a Central Management Server (CMS). The universe inherits the object-level security and user security rights defined for the CMS. The data and metadata in the universe are secured by defining security profiles in the information design tool Security Editor.

When you create a connection in a local project, it is an unsecured local connection and must be published before you can publish a business layer that references the connection. To secure a connection, publish it to a repository on a CMS. The Publish Connection Wizard creates the secured connection and provides a connection shortcut for the local project.

To browse and manage resources once they are published to a repository, use the Repository Resources View.

Editing published resources

You cannot edit a published universe directly in the information design tool. To work on it, you must retrieve it using the Retrieve Universe Wizard. The wizard retrieves the universe from the local folder or the repository, separates it into the business layer and the resources it references (local connection, connection shortcuts, data foundation), and creates these resources in a local project where they can be edited.

Connections can only be published to a repository. To edit a published connection, you must edit it from the Repository Resources View.

Related Information

Publishing a universe [page 302]
Retrieving a published universe from a repository [page 305]
Publishing a local connection to the repository [page 301]
Publishing a local universe to the repository [page 302]
Securing resources [page 324]
Managing repository resources [page 78]

10.1.1 Publishing a local connection to the repository

1. To start the Publish Connection Wizard, select the connection in the Local Projects View, right-click the connection and select *Publish Connection to a Repository*.
2. Follow the instructions on the wizard pages. For more information about what to do on a particular page, click the help icon in the lower left corner.

The connection is published in the repository. The local connection is deleted from the Local Projects View. You have the choice to create connection shortcut in the local project. To publish a business layer based on this connection, edit the business layer or data foundation to refer to the new shortcut.

Related Information

Opening a session [page 80]
Connection shortcuts [page 86]
Changing a connection in a data foundation [page 145]
Changing the data source of a business layer [page 222]
Synchronizing a project [page 74]

10.1.2 Publishing a local universe to the repository

You can publish a local universe to a repository if you have a secured connection and update the dependent resources to use this connection.

1. Create a connection shortcut in the Local Projects View:

Option	Command
Publish the local connection that the universe is based on.	Follow the procedure to publish a local connection.When asked, create a connection shortcut.
Use an existing secured connection to the data source.	Follow the procedure to create a connection shortcut.

2. Change the dependent resource to reference the connection shortcut:

Option	Command
Relational universes	Edit the data foundation and change the connection to use the connection shortcut.
OLAP universes	Edit the business layer and change the connection to use the connection shortcut.

3. In the Local Projects View, right-click the business layer and select *Publish* > *To a Repository*.

Related Information

Publishing a local connection to the repository [page 301]
Connection shortcuts [page 121]
Changing a connection in a data foundation [page 145]
Changing the data source of a business layer [page 222]

10.1.3 Publishing a universe

To publish a universe to a repository, the business layer must reference one or more secured connection shortcuts. All shortcuts must reference connections defined in the repository where the universe is to be published.

> **i Note**
>
> If the business layer references a local connection and you want to publish to a repository, first publish the connection and change the connection reference in the data foundation (relational), or in the business layer (OLAP) to use the connection shortcut. See the related topics for more information.

To publish a universe locally, the business layer must reference only a local connection that is not secured in any repository.

> **i Note**
>
> You can choose to *Republish* a universe. The information design tool automatically proposes the location where the universe was last published.

Recommended actions before publishing a universe:

- Save the business layer and all the resources it references.
- If the business layer references resources that are shared, synchronize the project to ensure that all changes are taken into account in the published universe.
- Check integrity of the business layer and, if applicable, the data foundation. The Publish Universe Wizard gives you the option to run a check integrity before publishing.

1. Start the Publish Universe Wizard:
   - To publish to a repository, select the business layer in the Local Projects view, right-click the business layer and select *Publish* > *To a Repository*.
   - To publish to a local folder, select the business layer in the Local Projects view , right-click the business layer and select *Publish* > *To a Local Folder*.
2. Follow the instructions on the wizard pages. For more information about what to do on a particular page, click the help icon in the lower left corner.

   In the Local Projects View, select the business layer and select *Publish* > *To a Local Folder*.

The universe is created as a .unx file in the local folder or the repository.

Related Information

Publishing a local connection to the repository [page 301]
Changing a connection in a data foundation [page 145]
Changing the data source of a business layer [page 222]
Checking the integrity [page 398]
Opening a session [page 80]
Publishing resources [page 300]

10.1.3.1 Selecting a repository folder

When publishing or retrieving resources on a repository, the wizard displays the folders in the repository in the left-hand pane. The table in the right-hand pane lists the resources in the folder.

When publishing a resource to a repository, navigate to the repository folder in the navigation tree in the left-hand pane. You can insert a folder.

When retrieving a published universe, navigate to the repository folder in the left-hand pane, and select the universe in the universe list in the right-hand pane.

> **i Note**
>
> By default, the resources are retrieved into the local project and are secured locally by requiring you to enter the CMS authentication when opening a retrieved data foundation or business layer.
>
> To remove the local security requirement, select the *Save for all users* option.

> **i Note**
>
> When you choose to republish a universe, the tool automatically proposes the repository folder where the universe was last published.

10.1.3.2 Selecting a local folder

When publishing or retrieving a resource in a local folder, the wizard prompts you for a local folder.

1. Enter the path to a folder accessible from your local machine.
2. To browse the file system and select a folder, click *Browse*.

10.2 Retrieving a published universe

This section describes how to retrieve a universe that has been published.

Related Information

Retrieving a published universe from the local file system [page 305]
Retrieving a published universe from a repository [page 305]

10.2.1 Retrieving a published universe from the local file system

To retrieve a published universe, you must have a project in the Local Projects View where the business layer and referenced resources are to be saved. You can create the local project on the fly.

1. In the Local Projects View, right-click the project folder and select *Retrieve Universe* > *from Local Folder*.
2. Follow the instructions on the wizard pages. For more information about what to do on a particular page, click the help button.

When the wizard finishes, the business layer and its dependent resources (connections, connection shortcuts, and data foundation) are created in the local project and are ready to be edited.

Related Information

Creating a local project [page 65]

10.2.2 Retrieving a published universe from a repository

To retrieve a published universe, you must have a project in the Local Projects View where the business layer and referenced resources are to be saved. You can create the local project on the fly.

1. You can retrieve a universe from a repository in two ways:

Option	Command
From the Local Projects View	Right-click the project folder in the Local Projects View and select *Retrieve Universe* > *from a Repository*.
From the Repository Resources View	Right-click the universe in the Repository Resources View and select *Retrieve Universe*.

> **i Note**
>
> By default, the resources are retrieved into the local project and are secured locally by requiring you to enter your repository system authentication when opening a retrieved data foundation or business layer.
>
> To remove the local security requirement, select the *Save for all users* option when selecting the universe in the repository.

2. Follow the instructions on the wizard pages. For more information about what to do on a particular page, click the help button.

When the wizard finishes, the business layer and its dependent resources (connections, connection shortcuts, and data foundation) are created in the local project and are ready to be edited.

Related Information

Opening a session [page 80]
Selecting a repository folder [page 304]
Creating a local project [page 65]

11 Using the Query Panel

Use the Query Panel to build, test, and preview the results of queries on a business layer or published universe.

In the information design tool, you can start the Query Panel in the following ways:

- To insert a query into the business layer.
- To open an existing query in the business layer.
- To run a query on a universe published in a repository.

> **i Note**
>
> When you include a set in in the Query Panel step, when you run the query on a .unx universe, the set unintentionally displays in the Query Panel outline.

Related Information

Inserting and editing a query in the business layer [page 242]
Running a query on a universe published in a repository [page 81]
Building a query [page 307]

11.1 Building a query

This procedure assumes that you have opened the Query Panel on a business layer or published universe. See the related topic on Using the Query Panel.

You can use this procedure to run queries on published universes, but to save the query you must start the Query Panel from the *Business Layer Editor Queries* pane.

For links to more detailed information about each step, see the Related Topics.

1. To select the objects you want to include in the query, drag objects from the business layer on the right into the *Result Objects* pane.
2. For hierarchy result objects, select members to include or exclude in the results. To open the Member Selector, click the arrow to the right of the hierarchy object name: .
3. To filter the results of the query, drag objects from the business layer into the *Filter Objects* pane.

   If a mandatory filter is defined on an object, the filter is triggered when you add the object to the *Result Objects* pane. The mandatory filter is visible in the query script, but not in the *Filter Objects* pane.

   Non-mandatory pre-defined filters are listed in the business layer. You can drag these pre-defined filters into the *Filter Objects* pane to limit the results. The filter is visible in the query script.

You can also build business filters, including filters that use prompts. For detailed information, see the related topics.

4. For relational universes, you can build combined queries. To open the *Combined Queries* pane, click the icon.
5. To set query properties, click the icon.
6. To see or edit the query script, click *View Script*.
7. To preview the query results, click the refresh button in the *Data Preview* pane.

   You can profile the values in the result columns. In the *Data Preview* pane, click the *Advanced Preview* icon .

   To change the layout of hierarchical data, click the *Result set display options* icon and select an option from the list:

Option	Description
Flat layout	Displays repeated values for a level in every row.
Hierarchical layout	Displays repeated values once for a level.

8. To save the query, click *OK*.

   The *OK* button is only available when running the Query Panel from the *Business Layer Editor*. The query is saved in the business layer and can be executed or edited from the *Queries* pane.

Related Information

Using the Query Panel [page 307]
Working with the Member Selector [page 316]
How to build a business filter [page 310]
Filtering data using prompts [page 312]
Setting query properties [page 308]
Viewing and editing the query script [page 310]
Profiling column values in the query panel [page 315]

11.2 Setting query properties

1. In the Query Panel, click the *Query properties* toolbar button .
2. Edit the query property settings as required.

Property	Description
Retrieve duplicate rows	When this option is selected, the query returns all related rows, even if there are duplicate rows. If you do not want duplicate rows in the result set, unselect this option.
Retrieve empty rows (only supported in OLAP universes)	An empty row occurs typically in multidimensional queries when the data for the intersection of two or more dimensions does not exist. When this option is selected, the result set includes rows that can contain empty cells. When this option is unselected, the result set contains only rows with non-empty cells.
Max retrieval time	Defines the maximum time (in seconds) that a query can run before the query is stopped. By default, this value is the same as the Limit Execution Time parameter in the universe parameters. When you set this value to 0, this option is disabled. When the Limit Execution Time parameter is lower than this setting, the Limit Execution Time value is used for limiting the query execution time.
Max rows retrieved	Defines the maximum number of rows of data that are displayed when the query runs. The query retrieves all the possible rows, but only displays the first n rows, where n is the maximum number of rows set for this parameter. The administrator can override this setting in the user security profile settings.
Sample result set	This parameter (when supported by the database) samples n database rows, where n is the value set for the sample result set. This method is faster than using the Max rows retrieved parameter.
Reset contexts on refresh	This is only available on relational universes. When this option is selected, when a user refreshes a query that contains contexts, the user must choose the context(s). The user can clear the previously selected contexts by clicking Clear Contexts. When this option is not selected, the query is refreshed using the original contexts. If the contexts have been edited since the last run for the query, the user must choose the contexts again since the query is considered as a new query.
Prompt order	When there are several prompts in a query, use this feature to set the order in which prompts are executed in a query. Click a prompt and use the up or down arrow to change the position of the prompt.

3. Click *OK* to close the *Query Properties* and save the changes.

11.3 Viewing and editing the query script

You can view the query script of a the query you build in the Query Panel. For relational universes, you can also edit the query script.

1. In the Query Panel, click *View script*.
   The query script displays in the *Query Script Viewer*.
2. For OLAP universes, your only option is to click *OK* to close the *Query Script Viewer*.
3. For relational universes, to edit the query script, select the *Use custom query script* option.
   a. In the *Query script* pane, edit the query.
   b. Click *Validate* to check the script syntax.
   c. Click *Undo* to undo the last edit to the script.
   d. Click *OK* to save and use the edited query script.

      The edited query script is used until you unselect the *Use custom query script* option, or close the Query Panel.
4. To use the query script generated by the Query Panel, select the *Use the query script generated by your query* option.
5. Click *OK* to save the changes.

11.4 Filtering data in the Query Panel

11.4.1 How to build a business filter

This procedure assumes you are building a business filter either in the Query Panel or the Edit Business Filter dialog box.

Business filters are filters based on objects in the business layer. They limit the data returned in the query.

Query filters have the following structure: filtered object, operator, operand.

```
[Country] InList (US;France)
```

The [**Country**] dimension is the filtered object, `InList` is the operator, and the list of values (**US**;**France**) is the operand. The filter removes all values of [**Country**] other than **US** and **France** from the query result.

The following table describes the components of a filter:

Filter Component	Description
Filtered object	The filtered object is the business layer object whose values are filtered. Dimensions, attributes, measures, hierarchies, and hierarchy levels can be used as filtered objects.
Operator	The operator is used to compare the filtered object with the operand. For example, the *Equal To* operator retains only those values of the filtered object that correspond exactly to the value of the operand.
Operand	The operand supplies the value or values used to filter the filtered object.

1. Drag an object from the business layer to the *Filter Objects* pane. This is the filtered object.
2. In the *Filter Objects* pane, select an operator from the list.
3. In the *Filter Objects* pane, select an operand from the list.

   Depending on the type of business layer and purpose of the filter, the following types of operands are available:

Operand type	Description
Constant	Use the *Constant* operand to enter values directly into the filter. For example, you can use a constant to enter **`France`** into the filter: `[Country] Equal To France` You can also enter the `@Variable` function to retrieve the value of a system variable or User Attribute. For example, to filter on the current user login name, enter the constant operand as `@Variable('BOUSER')`. For more information about `@Variable`, see the related topic.
List of Values	Use the *List of Values* operand to select values from the list associated with the filtered object. For example, if the filtered object is [**`City`**], you can use the list of values to select one or more of the cities associated with the object.
Object	Use the object operand to specify an object in the business layer. Drag and drop a business layer object to the operand position when defining the filter. **i Note** You cannot select an object as an operand on some OLAP data sources or if the filtered object is a hierarchy.
Prompt	Use the prompt operand when you want to be prompted for a value when the query is refreshed. See the related topic about filtering using prompts. **i Note** Prompt operands are not available if the business filter is defined for a Business Security Profile.

4. If you want to filter data on more than one criteria, add an additional filter by dragging another object to the *Filter Objects* pane.

   By default, the filters are combined with the AND operator. To use the OR operator, double-click the box with the *And* operator.

> **i Note**
>
> The OR operator is not supported for OLAP data sources.

5. If you want to nest query filters, drag another business object and drop it onto an existing query filter in the *Filter Objects* pane.

   Nesting query filters allows you to create more complex filter conditions than is possible when you combine filters at the same level. When you nest filters, you set the order in which they are evaluated. Nesting filters only makes sense once you have defined two filters at the same level.

Related Information

Using the Query Panel [page 307]
Inserting and editing filters [page 276]
@Variable [page 498]
Building a new prompt to filter data [page 313]

11.4.2 Filtering data using prompts

A prompt is a special type of query filter. It is a dynamic filter that displays a question every time you refresh the data in a query. You answer prompts by either typing or selecting the values you want to view before you refresh the data. The query then returns only the values you specified. You can also define a prompt that uses a formula to specify dynamic values as possible responses to the prompt question.

Prompts allow multiple users viewing a single document to specify a different sub-set of the database information and display it in the same report tables and charts. Prompts also reduce the time it takes for the data to be retrieved from the database.

When you define a prompt query filter, you can either build a new prompt, or use an existing prompt defined as a parameter in the business layer.

If you define more than one prompt in a query, you can change the order in which prompts are presented. Change prompt order in the query properties.

Merged Prompts

When querying a business layer or universe, similar prompts are merged. For prompts to be merged, the following rules must be true:

- The prompts have the same prompt text.
- The prompts expect answers having the same data type.
- The prompts expect the same number of answers. (The number of answers to be given depends on the operator used to reference the prompt. For example, *Equal To* expects one answer. *Between* expects multiple answers.)

A single prompt message appears for merged prompts. The list of values displayed by the merged prompt is the list associated with the prompt that has the most display property constraints.

> **i Note**
>
> All prompts in the query are candidates for merging: parameters defined in the business layer or data foundation, prompts defined as query filters, and prompts defined in the query expression of a business layer object with the `@Prompt` function.

Related Information

Building a new prompt to filter data [page 313]
Using an existing prompt to filter data [page 314]
Setting query properties [page 308]
Parameters [page 238]

11.4.2.1 Building a new prompt to filter data

This procedure assumes you are building a business filter either in the Query Panel or the Edit Business Filter dialog box.

1. Drag the object you want to filter with a prompt and drop it onto the *Query Filters* pane.
   The query filter appears in outline in the *Query Filters* pane. The outline shows the filtered object, the operator and the type of filter applied to the object. (By default the filter is a constant.)
2. Select the filter operator from the list.

   > **i Note**
   >
   > The list of available operators depends on the type of filtered object.

3. Click the arrow at the right of the outline query filter and select *Prompt* from the menu to filter the object using a prompt.
   The *Edit Prompt* dialog box appears and the *New Prompt* option is selected by default.
4. Edit the prompt question in the *Prompt Text* box.
5. Select *Prompt with List of Values* to allow the user to select from a list of values when answering the prompt.

   The option is only available if the filtered object has an associated list of values in the universe.
6. Select *Select only from list* to restrict the user choice to values from the list of values.

   You can select this option only if the *Prompt with List of Values* option is selected.
7. Select *Keep last values* if you want the prompt to propose the last value that the user selected on the previous refresh. The first time the query is run, the default value (if set) is proposed.
8. Select *Optional prompt* to make the prompt optional. If the user does not supply a value for an optional prompt, the prompt is ignored.

9. Select *Set default values* if you want the prompt to propose values by default when it displays.
   a. To enter or select the default values, click the button to the right of the text field to display the available lists of values.
   b. If the filter object has an associated list of values, select the default values from the list.
   c. If the filter object has no associated list of values, enter default values.
   d. Click *OK* to save the default values.
10. Click *OK* to save the new prompt definition.

Related Information

How to build a business filter [page 310]

11.4.2.2 Using an existing prompt to filter data

This procedure assumes you are building a business filter either in the Query Panel or the Edit Business Filter dialog box.

1. Drag the object on which you want to apply a prompt and drop it onto the *Query Filters* pane.
   The query filter appears in outline in the *Query Filters* pane.
2. Select the filter operator from the list.

   > **i Note**
   >
   > The list of available operators depends on the type of filtered object.

3. Click the arrow at the right of the Query Filter and select *Prompt* from the menu.
4. In the *Edit Prompt* dialog box, select the *Use Universe Parameter* option.
5. Select an existing parameter.

   The list displays only those universe prompts that are compatible with the object you are filtering. For example, the filtered object and the universe prompt must have the same data type.
6. Click *OK* to save the prompt definition.

Related Information

How to build a business filter [page 310]
Parameters [page 238]

11.5 Refreshing a query containing BEx variables

When you refresh a query containing a BEx variable, the variable is resolved as a prompt. Depending on the type of variable, the prompt can be resolved either by selecting a value in a list of values, or manually entering a value or expression.

You can manually enter values for the following types of BEx variables when refreshing a BEx query in the Query panel:

- Single value
- Multi single value
- Single keydate
- Selection option

> **i Note**
>
> Manual entry is only available for Selection option when it is set as Multivalue, and support is added to the Web intelligence server or the Windows registry depending on your query or reporting application. Refer to the SAP Business Intelligence Platform Administrator Guide for information on setting up multivalue support for support for Selection option. If multivalue is not supported, then Selection option's default property is Interval.

Refer to the SAP BusinessObjects Web Intelligence User's Guide for information on using BEx queries and variables.

1. Open a BEx query in the Query Panel.
2. Click *Refresh*.
   The *Prompts* input box appears.
3. In the *Prompt summary* pane, click the prompt question for the variable .
4. Enter a value or an expression in the *Enter value* box then click the right arrow. You can enter multiple values or expressions.
   The filter values are shown in the Selected values pane.
5. Click *OK*.
   The query is refreshed with the applied variable filters. The values appear in the *Result set* pane.

Related Information

Using the Query Panel [page 307]

11.6 Profiling column values in the query panel

You can profile the values for a columns in the query results. Profiling shows graphically (in a pie or bar chart) the number of occurrences of each value of a column. If the column has a filter defined, the filter is applied.

1. Open a query in the Query Panel and refresh the results.
2. In the *Data Preview* pane, click the *Advanced Preview* icon .
   The Profile Column Values window appears. To see what you can do in this window, see the related topic.

Related Information

Showing values in a data source [page 162]
Using the Query Panel [page 307]

11.7 Working with the Member Selector

The Member Selector lets you visualize and select members in a hierarchy. Use the Member Selector to:

- Select the members you want to appear in the query result set.
- Define members that will be excluded from queries.
- Define prompts to allow the selection of members to appear in the query each time you run the query.
- Select the members for a named set.
- Select the members when defining a Business Security Profile filter.

You open the Member Selector from hierarchy objects that you include in queries in the Query Panel. The Member Selector opens automatically when you edit named sets or filters for a Business Security Profile on a hierarchical business layer.

Related Information

Selecting hierarchy members [page 318]
About selecting hierarchy members [page 316]

11.7.1 About selecting hierarchy members

In the Member Selector, you can select members in several ways:

- Select members explicitly in the hierarchy. For example, explicitly select the [California] and [Los Angeles] members of the [Geography] hierarchy.
- Select members implicitly using hierarchy relationships. For example, to select US states, you can select the child members of the [US] member.
- Select members included in a named set, for example Top Cities by Revenue, to include the cities that generate the most revenue.

- Select all members in a hierarchy level.
- Select all members up to a certain level in the hierarchy.
- Select calculated members.

The Member Selector contains three tabs:

Tab	Description
Members	Displays the members arranged hierarchically. Use this tab to select members explicitly, by hierarchical relationships, and by specifying all members up to a given level.
Metadata	Shows the hierarchy levels (if the hierarchy supports named levels), named sets, and calculated members.
Prompts	Lets you define and modify prompts.

For information on how to select, display, search for, and sort hierarchy members, see the related topics.

Related Information

Selecting hierarchy members [page 318]
Selecting members by hierarchy relationship [page 318]
Selecting hierarchy members by level [page 319]
Selecting named sets [page 320]
Selecting calculated members [page 320]
Searching for hierarchy members [page 321]
Excluding hierarchy members [page 321]
Defining a prompt to select members [page 322]
Showing selected members in the Member Selector [page 322]
Sorting hierarchy members [page 323]
Setting display options [page 323]
Showing estimated child count [page 323]

11.7.2 Opening the Member Selector in the Query Panel

1. In the Query Panel, add the hierarchy object to the *Result Objects* pane.
2. To open the Member Selector, click the arrow to the right of the hierarchy object name: .
3. You can now select members in the hierarchy for inclusion or exclusion in a query. For descriptions of different ways to select members, see the related topic.

Related Information

About selecting hierarchy members [page 316]

11.7.3 Selecting hierarchy members

1. In the Member Selector, click the *Members* tab to display the hierarchy members.
2. Select members in the hierarchy display.
3. To select all members in the hierarchy, click the *Select* icon , and select *Select All*.
4. To select all members up to a specified level in the hierarchy, click the *Select* icon . You can identify the level in two ways:

Option	Description
Select a named level	This option is only available if the hierarchy has named levels. Select *Select All Members until Named Level* and select the level from the submenu.
Select a number of levels below the root	Select *Select All Members until* and select the number of levels from the submenu.

5. When you complete your selection, click *OK*.

The selected members appear below the hierarchy object in the *Result Objects* pane of the Query Panel. When you run the query, only those members are included in the query result.

Related Information

Opening the Member Selector in the Query Panel [page 317]

11.7.4 Selecting members by hierarchy relationship

1. In the Member Selector, click the *Members* tab to display the hierarchy members.
2. In the hierarchy, right-click the member for which you want to define the hierarchy relationship.
3. Select the relationship function from the menu:

> **i Note**
>
> *Children/Descendants* and *Parents/Ancestors* are mutually exclusive pairs. You cannot select both the children and the descendants of a member, and you cannot select both the parents and the ascendants of a member.

Relationship Function	Description
Self	Includes only the selected member. This is the default setting.
Children	Includes members one level below the selected member that have the selected member as their parent. The selected member is not included.
Descendants	Includes all members at all levels below the selected member. The selected member is not included.
Descendants until Named Level...	Includes the members at levels below the selected member until the named level you select. This option is only available if the hierarchy has named levels.
Descendants until...	Includes the members at levels below the selected member until the number of levels you select.
Parent	Includes the member that is one level above the selected member. The selected member is not included.
Ancestors	Includes all members at all levels above the selected member. The selected member is not included.
Siblings	Includes members at the same level that have the same parent as the selected member. The selected member is not included.
Exclude	Excludes members according to the relationship function (Self/Children/Descendants/Parent/Ancestors/Siblings).

Related Information

Opening the Member Selector in the Query Panel [page 317]

11.7.5 Selecting hierarchy members by level

To select members by level, the hierarchy must have named levels.

1. In the Member Selector, click the *Metadata* tab to display the hierarchy levels.

   > **i Note**
   >
   > If the *Levels* folder does not display in the *Metadata* tab, the hierarchy is not level-based and you cannot select members by level.

2. Select levels in the *Levels* folder.
3. Click *OK*.

Related Information

Opening the Member Selector in the Query Panel [page 317]

11.7.6 Selecting named sets

To select members by named set, the hierarchy must have at least one named set defined. Named sets are defined in the business layer of the universe.

1. In the Member Selector, click the *Metadata* tab to display the named sets.

   > **i Note**
   >
   > If the *Named Sets* folder does not display in the *Metadata* tab, the hierarchy has no named sets defined.

2. Select named sets in the *Named Sets* folder.
3. Click *OK*.

Related Information

Opening the Member Selector in the Query Panel [page 317]
Inserting and editing named sets [page 282]

11.7.7 Selecting calculated members

To select calculated members, the hierarchy must have at least one calculated member defined. Calculated members are defined in the business layer of the universe.

1. In the Member Selector, click the *Metadata* tab to display the calculated members.

   > **i Note**
   >
   > If the *Calculated Members* folder does not display in the *Metadata* tab, the hierarchy has no calculated members defined.

2. Select calculated members in the *Calculated Members* folder.
3. Click *OK*.

Related Information

Opening the Member Selector in the Query Panel [page 317]

Inserting and editing calculated members [page 283]

11.7.8 Searching for hierarchy members

Use the Search function in the Member Selector to select hierarchy members from a list of search results.

1. To open the *Member Search* dialog box, in the Member Selector *Members* tab, click the *Search* icon .
2. Enter text to search for in the *Search pattern* box.

   You can use wildcards in the search:

Wildcard	Description
*	Matches any string of characters
?	Matches any one character

3. To search for text in the keys, select the *Search Keys* radio button.
4. Click *Search*.
5. To select members from the search results, select the members in the *Search results* table.
6. Click *OK*.

11.7.9 Excluding hierarchy members

1. In the Member Selector, select the members that you want to exclude.

   You can select members explicitly, by hierarchy relationship, by level, by named set, and calculated members.

   The selected members are listed in the *Summary* pane of the Member Selector.
2. In the *Summary* pane, select the *Exclude* option next to the members or member sets you want to exclude.
3. Click *OK*.

Below the hierarchy object in the *Result Objects* pane of the Query Panel, the excluded members appear with a line drawn through the names to indicate that they are excluded from the query.

Related Information

Opening the Member Selector in the Query Panel [page 317]
Selecting hierarchy members [page 318]
Selecting members by hierarchy relationship [page 318]
Selecting hierarchy members by level [page 319]
Selecting named sets [page 320]
Selecting calculated members [page 320]
Searching for hierarchy members [page 321]

11.7.10 Defining a prompt to select members

You can define a prompt to defer member selection to the time the query is run.

> **i Note**
>
> When selecting members in response to a prompt, you can only select members explicitly. You cannot select members by hierarchy relationship.

1. In the Member Selector, click the *Prompt* tab.
2. Select *Enable Parameter* to defer member selection to when the query is run.

   You cannot access the other tabs in the Member Selector when the *Enable Parameter* option is selected.
3. Enter text for the prompt in the *Prompt Text* box.
4. If you want the prompt to select the previously-selected values by default when it displays, select *Keep last values selected*.
5. To define default values for the prompt, select *Set default values* and click *Edit*. In the *Select Parameter Values* dialog box, select default values for the prompt and click *OK*.
6. Click *OK*.

Related Information

Opening the Member Selector in the Query Panel [page 317]

11.7.11 Showing selected members in the Member Selector

In the Member Selector *Members* tab, you can click the *Expand tree to show selections* icon to show the selected members in the hierarchy display.

The display automatically expands to show the following members:

- Explicitly selected members.
- Members used to select related members. The related members implicitly selected are not necessarily shown. For example, if the member called France was used to select its children, the tree view expands to show France. If the node France contains no explicitly selected members, the node is not expanded the show the implicitly selected children.

> **→ Tip**
>
> The *Expand tree to show selections* command does not collapse nodes that are already expanded. To reduce the length of the display, close all open nodes in the hierarchy display before clicking the icon.

Related Information

Opening the Member Selector in the Query Panel [page 317]

11.7.12 Sorting hierarchy members

By default in the Member Selector, hierarchy members are displayed sorted in the order they are stored in the database. To help find members in the hierarchy, you can sort the display in ascending or descending alphabetical order.

In the Member Selector *Members* tab, click the *Sort order* icon and select the desired sort order.

The members are sorted locally in the Member Selector. Display of members in the query is not affected.

Related Information

Opening the Member Selector in the Query Panel [page 317]

11.7.13 Setting display options

By default, the Member Selector displays hierarchy member captions. You can set the display options to display unique names, or both captions and unique names.

In the Member Selector *Members* tab, click the *Member display options* icon and select the desired display option.

11.7.14 Showing estimated child count

The Member Selector estimates the number of children for each member. By default, the estimates are hidden. You can show the estimated child count in the hierarchy display.

In the Member Selector *Members* tab, click the *Show/Hide estimated child count* icon to toggle the display of child counts.

12 Securing resources

Universe security starts when the universe is published to a repository on a Central Management Server (CMS). Published universes are stored in the Universes folder and secured connections are stored in the Connections folder.

You secure universes based on the users and groups that are defined in the system repository using the Central Management Console (CMC).

As a first level of security, using the CMC, you grant the right to access specific folders, resources, universes, and connections in the repository to specific users and groups. How to define these rights is described in the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.

You define another level of security using the information design tool Security Editor. You can restrict the data that is returned in a query using query limits and controls, filters, and row restrictions. You can also grant or deny access to objects and views in the business layer. To create this level of security, you define security profiles for the universe and assign these profiles to users and groups. The basics of how security profiles work are presented in this topic.

Security profiles

A security profile is a group of security settings that apply to a universe published in the repository. The settings control the data that is displayed and modify the parameters defined in the data foundation and/or business layer. Once the profile is assigned to a user or a group, the settings in the profile determine what objects, data, and connections the user sees when connecting to the universe. There are two types of profiles:

- Data Security Profiles have security settings defined on objects in the data foundation and on relational connections.
- Business Security Profiles have security settings defined on objects in the business layer and on OLAP connections.

Multiple profiles can be defined for each universe. The profiles are saved in the repository.

How profiles work

A user of query and reporting tools who has been granted access to a universe using the CMC, and who has no security profiles assigned or inherited, can see all the objects in the universe and all the data returned by those objects.

When you assign a profile to the user, the security settings defined in the profile are applied whenever the user runs a query on the universe.

In the information design tool, security profiles are applied when you run a query from the Repository Resources View or the *Security Editor*. They are applied according to the user name you used to open the repository session. When you run a query from the *Business Layer Editor*, security profile settings do not apply.

How multiple profiles are handled

You can assign more than one profile to a user or group. A user might be assigned a profile and also inherit profiles from groups. When more than one profile is assigned to a user, the profiles are aggregated to produce a single group of settings called the net profile.

Aggregation follows priority and restriction levels that you can modify in the Security Editor. You can also see which profiles a user or group inherits, and preview net profiles for a user or group.

Profile maintenance

Profiles are stored independently from the universe itself: changes in the data foundation or business layer of the universe do not affect the profiles when the universe is republished. Similarly, changes in a profile are independent of assignments, so you do not have to reassign a profile when it is modified. It remains assigned, including any changes.

If you republish a universe, run a check integrity on the universe to flag any discrepancies between the universe and its security profiles.

Profiles created for a universe are deleted when the universe is deleted.

Related Information

Data Security Profile settings [page 330]
Business Security Profile settings [page 336]
Security Profile aggregation [page 341]
Displaying profiles assigned to a user and previewing net profiles [page 349]
Running a query on a universe published in a repository [page 81]
Working with the Security Editor [page 327]

12.1 Security Editor

12.1.1 Securing resources in the information design tool

No authentication is required to start the information design tool.

A user can create and edit unsecured resources (data foundations, business layers, connections) in the Local Projects view. The resources are saved in a local project.

Resources are secured when a user shares a local project and its resources, or publishes universes or connections to a repository. Shared projects and published resources are stored securely in a repository on the Central Management Server (CMS).

Application rights are granted in the Central Management Console (CMC). How to define these rights is described in the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.

A user with appropriate rights can retrieve a published universe from the repository for editing. Resources can also be retrieved from a shared project during project synchronization. In both cases, the resources are retrieved into the local project and are secured locally by requiring the user to enter the CMS authentication when opening a retrieved data foundation or business layer.

When you start the information design tool, it reopens any resources that were open when you last closed the tool. If secured resources are open, you need to enter your CMS authentication to start the tool.

> **i Note**
>
> To remove the local security requirement, you must have the *Save for all Users* right granted in the CMC. When a resource is saved for all users, any user can open the resource without entering CMS authentication.

Secured connections cannot be retrieved from the repository and stored locally in the information design tool. Instead, a shortcut to the connection in the repository is stored in the local project. Secured connections must be edited directly in the repository from the Repository Resources View. To be able to access data from a secured connection (for example, show table values or run a query), the user must enter the CMS authentication for the repository where the connection is published. The system uses the authentication to determine what rights the user has for that connection.

Related Information

Working with local projects and resources [page 64]
Connection shortcuts [page 86]

12.1.2 Opening the Security Editor

1. On the information design tool toolbar, click the *Security Editor* icon .
2. In the *Open Session* dialog box, select the session you want to open.
3. If you are not already logged into the selected session, enter the required information.

The Security Editor opens in a new tab.

> **i Note**
>
> You can open more than one session of the Security Editor at one time. The sessions must be on different repositories.

Related Information

Opening a session [page 80]
Working with the Security Editor [page 327]

12.1.3 Working with the Security Editor

Use the Security Editor to create and edit security profiles, and assign profiles to users and groups. This topic describes how to navigate the Security Editor. For steps to help you build universe security, see Securing a universe using Security Profiles [page 328].

The session name displays on the tab of the Security Editor. If the session name is prefixed by an asterisk, it means you have made changes to the security profiles or assignments in the Security Editor that have not yet been saved in the repository.

The Security Editor can be viewed in two ways: by universe, or by users/groups. Select the tab on the left-hand side of the Security Editor to display the view you want to work with.

- The *Universes / Profiles* tab lets you to do tasks by first selecting a universe in the repository.
- The *Users / Groups* tab lets you to do tasks by first selecting a user or group. The three icons in the *Users / Groups* panel lets you to display users and groups in three ways:

Icon	Description
	Shows only users.
	Shows all groups and users they contain. A group is shown even if it has no groups or users assigned. Groups are displayed as a flat list. This is the default display.
	Shows all groups and the groups and users they contain. Groups are thus displayed with their different parent groups.

Application rights granted to you in the Central Management Console control what tasks you can perform in the Security Editor. For more information, see the Rights Appendix in the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.

Related Information

Securing resources [page 324]
Securing a universe using Security Profiles [page 328]
Changing Security Profile priority [page 335]
Changing security profile aggregation options [page 342]

Assigning Security Profiles to users [page 349]
Displaying profiles assigned to a user and previewing net profiles [page 349]

12.2 Security Profiles

12.2.1 Securing a universe using Security Profiles

Before you begin:

- The universe for which you want to create security must be published to a repository. In the information design tool, you can create security for .unx universes only.
- Make sure you have the necessary rights defined in the Central Management Console (CMC). See the related topic about CMC rights.

For links to more detailed information on each step, see the Related Topics.

1. Open the Security Editor with a session in the repository where the universe is published.
2. Select the universe in the *Universes / Profiles* pane to define security profiles.
3. For relational universes, insert a Data Security Profile to define the following types of security:
   - Define replacement connections to override the connections defined in the universe.
   - Define overrides for the query options and query limits defined in the universe.
   - Restrict data returned to specific rows using a WHERE clause.
   - Define replacement tables.

   > **i Note**
   >
   > You can create more than one Data Security Profile for a universe.

4. Insert a Business Security Profile to define the following types of security:
   - Define a replacement connection to override the connection defined in the universe.
   - Restrict objects that appear in the Query Panel to create queries.
   - Restrict objects for which data is returned.
   - Filter data returned in queries.

   > **i Note**
   >
   > You can create more than one Business Security Profile for a universe.

5. Save the changes to the security settings in the repository by clicking the save icon in the main tool bar.
6. Select the *Users / Groups* pane to assign the profiles to users and groups.
7. If more than one profile is assigned to a user (either directly or by inheritance), preview the net results of the aggregated profiles.
8. If modifications to the way profiles are aggregated is needed, in the *Universes / Profiles* pane, change profile priority and profile aggregation options.

9. Save the changes to the security settings in the repository by clicking the save icon in the main tool bar.
10. Test the security profiles for a particular user:
    a. Open the Security Editor using the login information for the user who is being assigned the security profiles.
    b. In the *Universes / Profiles* pane, right-click the universe and select *Run Query*.

       The Query Panel opens. The security profiles assigned to the user are applied.

       > **i Note**
       >
       > Because a user needs the "Administer security profiles" application right granted in order to open the Security Editor, this method of testing profiles is limited. The security profile for a user can be tested in a query application, for example Web Intelligence.

If you republish a universe, run a check integrity on the universe to flag any discrepancies between the universe and its security profiles. In the *Universes / Profiles* pane, right-click the universe and select *Check Integrity*.

Related Information

CMC rights for information design tool users [page 350]
Opening the Security Editor [page 326]
Managing Data Security Profiles [page 329]
Managing Business Security Profiles [page 335]
Assigning Security Profiles to users [page 349]
Displaying profiles assigned to a user and previewing net profiles [page 349]
Changing Security Profile priority [page 335]
Changing security profile aggregation options [page 342]
Building a query [page 307]
Checking the integrity [page 398]

12.2.2 Managing Data Security Profiles

> **⚠ Caution**
>
> Changes to security profiles overwrite any previous changes. In the event that more than one user is editing the same universe profiles at the same time, the changes saved last overwrite changes done previously by others.

1. In the Security Editor *Universes / Profiles* pane, select the universe.
2. Do one of the following:

Option	Command
To insert a profile	Right-click the universe name and select *Insert Data Security Profile*.
To edit an existing profile	Double-click the profile name.
To duplicate an existing data security profile	In the *Universes/Profiles* pane, navigate to the profile you want to copy, then select *Duplicate Data Security Profile*.

3. Define security settings in each of the tabs by clicking the tab you want. You can expand the panel to full screen using the expand/collapse button.

   For more information on the Data Security Profile settings, see the related topics.

   > **i Note**
   >
   > Clicking the *Reset* button returns the settings on all tabs to the default values as they are defined in the data foundation and business layer.

4. When you have defined all of the settings, click *OK*.
5. To save the changes to the security settings in the repository, click the save icon in the main tool bar.

Related Information

Opening the Security Editor [page 326]
Data Security Profile Connections settings [page 331]
Data Security Profile Controls settings [page 332]
Data Security Profile SQL settings [page 332]
Data Security Profile Rows settings [page 333]
Data Security Profile Tables settings [page 334]

12.2.2.1 Data Security Profile settings

A Data Security Profile is a group of settings that define security on a published universe using objects in the data foundation and the data connections.

All Data Security Profile settings apply to relational universes only.

Security settings in Data Security Profiles

Security Setting	Description
Connections	Defines replacement relational connections.
Controls	Defines replacement query timeout and size limits.
SQL	Defines replacement query options.
Rows	Defines an SQL WHERE clause to restrict rows returned in the query.

Security Setting	Description
Tables	Defines replacement tables.

Each type of Data Security Profile setting is described in a related topic.

Related Information

Data Security Profile Connections settings [page 331]
Data Security Profile Controls settings [page 332]
Data Security Profile SQL settings [page 332]
Data Security Profile Rows settings [page 333]
Data Security Profile Tables settings [page 334]
Security Profile aggregation [page 341]
Managing Data Security Profiles [page 329]

12.2.2.2 Data Security Profile Connections settings

Connections settings are defined in the Data Security Profile for relational universes only (multisource-enabled and single-source). Define replacement connections for OLAP universes in the Business Security Profile.

Use the Data Security Profile Connections setting to define replacement connections that can override the connections defined in the universe. Once a user is assigned or inherits a profile that contains a replacement connection, when the user runs a query on the universe, the replacement connection is used instead of the connection defined in the universe.

Only secured connections can be defined as replacement connections. Relational connections fall into one of three types, listed below. The replacement connection must be of the same type as the original connection.

- SAP BW relational databases
- SAS relational databases
- Other relational databases

To define a replacement connection, select the original connection in the table and click *Edit*.

Select a connection in the Connections folder and sub-folders for which you have the *View objects* right granted for the repository where you are defining the security profiles.

For multisource universes that rely on multiple connections, you can define a replacement for each connection.

Related Information

Aggregation of Connections settings [page 342]

12.2.2.3 Data Security Profile Controls settings

Controls settings can be defined for multisource-enabled and single-source relational universes.

Use the Data Security Profile Controls settings to define replacement query limits to override default limits when retrieving data from the database. Default query limits are set by the universe designer in the business layer. Once a user is assigned or inherits a profile with replacement Controls settings, when the user runs a query, the replacement limits are used instead of the limits defined in the business layer properties.

In the editor for Data Security Profiles, the limits selected and the limit values defined in the business layer are displayed. When you select or unselect a limit, or enter a new value for a limit, the label appearance changes to bold. This shows that the limit is an override and not the default limit defined in the universe.

Query Limit	Possible values
Limit size of result set to	True and a numerical size between 0 and 2147483647 rows False
Limit execution time to	True and a numerical size between 0 and 2147483647 minutes False
Warn if cost estimate exceeds	True and a numerical size between 0 and 10000 minutes False

For more information on query limits, see the related topic about business layer properties.

Related Information

Properties [page 218]
Aggregation of Controls settings [page 343]

12.2.2.4 Data Security Profile SQL settings

SQL settings can be defined for multisource-enabled and single-source relational universes.

Use the Data Security Profile SQL settings to define replacement query options. The universe designer defines default query options in the business layer and data foundation properties. Once a user is assigned or inherits a profile with SQL settings, when the user uses the query panel, the replacement options are used instead of the query options defined in the universe.

In the editor for Data Security Profiles, the SQL settings selected in the business layer and data foundation are displayed. When you select or unselect an option, the label appearance changes to bold. This shows that the option is an override and not the default defined for the universe.

Query Option	Possible values
Allow use of subqueries	True False
Allow use of union, intersect and minus operators	True False
Allow complex operands in Query Panel	True False
Multiple SQL statements for each context	True False
Multiple SQL statements for each measure	True False
Allow Cartesian products	True False

For more information on query options, see the related topics about business layer and data foundation properties.

Related Information

Properties [page 218]
Data foundation properties [page 129]
Aggregation of SQL settings [page 344]

12.2.2.5 Data Security Profile Rows settings

Rows settings can be defined for multisource-enabled and single-source relational universes.

Use Data Security Profile Rows settings to restrict the rows returned in a query. You restrict the rows by defining an SQL `WHERE` clause for a specified table. Once a user is assigned or inherits a profile with a Rows setting, when the user runs a query on the universe, the defined `WHERE` clause is added to the SQL generated if the table is referenced in the query.

> **i Note**
>
> A user who has the right to edit the generated SQL in the reporting tool can change the `WHERE` clause generated by the Rows setting. Remember to manage the user's rights in the reporting tool to prevent the user from modifying the SQL.

You can define the `WHERE` clause for any standard table in the data foundation. The SQL for the `WHERE` clause can include:

- @Functions such as `@Variable` and `@Prompt`

- For multisource enabled universes, references to other tables in any connection defined for the universe
- For multisource-enabled universes, SAP BusinessObjects SQL functions

The SQL for the `WHERE` clause cannot include:

- Calculated columns
- Derived tables

Related Information

Aggregation of Rows settings [page 344]

12.2.2.6 Data Security Profile Tables settings

Tables settings can be defined for multisource-enabled and single-source relational universes.

Use the Data Security Profile Tables setting to define replacement tables. Once a user is assigned or inherits a profile with a Tables setting, when the user runs a query that references the original table, the replacement table is used instead.

The original table can be a standard table or a federated table in the data foundation. The replacement table can be one of the following types of tables:

- Standard table in the data foundation
- Federated table in the data foundation
- Database table in the connection

Alias and derived tables cannot be defined as either the original or the replacement table.

If you want to specify an owner and qualifier for a replacement table in the database, you must enter these in the fields provided. When you specify a replacement table in this way, the table does not need to exist in the database at design time. So for example, you can specify the table in anticipation of a table that exists at query run time. For more information about data foundation table names, see the related topic.

> **i Note**
>
> A user who has the right to edit the generated SQL in the reporting tool can change the replacement table name. Remember to manage the user's rights in the reporting tool to prevent the user from modifying the SQL.

Related Information

Tables [page 149]
Aggregation of Tables settings [page 345]

12.2.3 Changing Security Profile priority

Priority is used to aggregate certain security settings if more than one Data Security Profile or Business Security Profile is assigned to a user or group. For more information on profile aggregation, see the related topic.

1. In the Security Editor *Universes / Profiles* pane, select the universe.
2. Right-click the universe name and do one of the following:
   - Select *Change Data Security Profile Priority* (this command is only available if the universe has more than one Data Security Profile defined).
   - Select *Change Business Security Profile Priority* (this command is only available if the universe has more than one Business Security Profile defined).
3. In the dialog box listing the Security Profiles, use the arrow buttons to move profiles up and down in the list. The first profile in the list has highest priority.
4. When you have finished prioritizing, click *OK*.
5. To save the changes in the repository, click the save icon in the main tool bar.

Related Information

Security Profile aggregation [page 341]
Opening the Security Editor [page 326]

12.2.4 Managing Business Security Profiles

> ⚠ Caution
>
> Changes to security profiles overwrite any previous changes. In the event that more than one user is editing the same universe profiles at the same time, the changes saved last overwrite changes done previously by others.

1. In the Security Editor *Universes / Profiles* pane, select the universe.
2. Do one of the following:

Option	Command
To edit an existing profile	Double-click the profile name.
To insert a profile	Right-click the universe name and select *Insert Business Security Profile*.
To duplicate a business security profile	In the *Universes/Profiles* pane, navigate to the profile you want to copy, then select *Duplicate Business Security Profile*.

3. Define security settings in each of the tabs by clicking the tab you want.You can expand the panel to full screen using the expand/collapse button.

For more information on the *Business Security Profile* settings, see the related topics.

> **i Note**
>
> Clicking the *Reset* button returns the settings on all tabs to the default values as they are defined in the data foundation and business layer.

4. When you have defined all of the settings, click *OK*.
5. To save the changes to the security settings in the repository, click the save icon in the main tool bar.

Related Information

Business Security Profile Connections settings [page 337]
Business Security Profile Create Query settings [page 338]
Business Security Profile Display Data settings [page 339]
Business Security Profile Filters settings [page 340]

12.2.4.1 Business Security Profile settings

A Business Security Profile is a group of settings that define security on a published universe using objects in the business layer.

Security settings for Business Security Profiles

Security Setting	Description
Connections	Defines a replacement OLAP connection.
Create Query	Defines the universe views and business layer objects available to the user in the query panel. **i Note** Create Query settings secure metadata only.
Display Data	Grants or denies access to the data retrieved by objects in the business layer when the user runs a query.
Filters	Defines filters using objects in the business layer.

Each type of Business Security Profile setting is described in a related topic.

In the business layer, designers can set the status of objects to *Active*, *Hidden*, or *Deprecated*. When defining profile settings, you have access to all active objects in the business layer. Objects that are hidden or deprecated in the business layer never appear in the query panel or on reports.

Related Information

Business Security Profile Connections settings [page 337]
Business Security Profile Create Query settings [page 338]
Business Security Profile Display Data settings [page 339]
Business Security Profile Filters settings [page 340]
Security Profile aggregation [page 341]
Managing Business Security Profiles [page 335]

12.2.4.2 Business Security Profile Connections settings

Connections settings are defined in the Business Security Profile for OLAP universes only. Define replacement connections for relational universes in the Data Security Profile.

Use the Business Security Profile Connections setting to define a replacement connection that can override the connection defined in the universe. Once a user is assigned or inherits a profile that contains a replacement connection, when the user runs a query on the universe, the replacement connection is used instead of the connection defined in the universe.

The replacement connection has the following requirements:

- It must be a secured OLAP connection.
- It must refer to the same database type as the original connection (for example MSAS or Essbase).
- It must specify the catalog and the cube in the connection definition.

! Restriction

- SAP BW OLAP connections (*BICS Client*) cannot be used as replacement connections.
- A replacement connection cannot use prompted authentication.

When applying the security setting, the catalog and the cube defined in the replacement connection are used.

To define a replacement connection, select the original connection in the table and click *Edit*.

Select a connection in the Connections folder and sub-folders for which you have the *View objects* right granted for the repository where you are defining the security profiles.

Related Information

Aggregation of Connections settings [page 342]

12.2.4.3 Business Security Profile Create Query settings

Use the Business Security Profile Create Query setting to grant or deny the use of business layer objects in the query panel.

By default, a user with access to the universe granted in the repository can see all universe objects in the query panel. Once the user is assigned or inherits a profile with a Create Query setting, only the views and objects granted by the setting are displayed and can be selected for a query.

If an object is not granted and not denied explicitly, it is denied by default. Unlike objects that are explicitly denied, objects that are denied by default could be granted by inheritance after aggregating Business Security Profiles to determine the net profile for a user. For more information about aggregating profiles, see the related topic.

There are two ways to grant or deny objects:

- By business layer view: Grants or denies all objects in a view. The *All business layer views* option allows you to grant or deny all views defined for the universe.
- By object: You can grant or deny the objects listed below. The *All objects* option allows you to grant or deny all objects in the business layer.
  - Dimensions
  - Attributes
  - Measures
  - Calculated members
  - Filters
  - Prompts
  - Named sets
  - Folders: Grants or denies all objects in the folder.
  - Analysis dimensions: Grants or denies all objects in the dimension.
  - Hierarchies: Grants or denies all objects in the hierarchy.

    > **i Note**
    >
    > It is not possible to grant or deny a hierarchy level.

> **→ Tip**
>
> If most views are allowed, it is easiest to grant all views, and then deny the ones that are not allowed. Using the *All business layer views* and *All objects* options has the advantage that any new view or object defined in the business layer is automatically included in the Create Query setting when the universe is published.

If the *All business layer views* or *All objects* option is used, the settings are aggregated to determine the net setting for this profile, for example:

- If *All business layer views* are denied and one view is granted, this profile denies all views except the one granted.
- If *All business layer views* are granted and one view is denied, this profile grants all views except the one denied.
- If *All objects* are denied and one object is granted, any parent folders in the path to access the object are granted, but only to access the object. The other objects in the parent folders are denied.
- If *All objects* are granted and one object is denied, the parent folders in the path to access the object are denied, but only to prevent access to this object. The other objects in the parent folders are granted.

The objects in a granted view are granted in that view only. If the same object is contained in another view, it is not automatically granted.

Whether or not a user sees a particular object in the query panel is determined after aggregating the Create Query settings in all profiles assigned to the user, and taking into consideration object access level. For more information about aggregating profiles, see the related topic.

Related Information

Aggregation of Create Query settings [page 346]

12.2.4.4 Business Security Profile Display Data settings

Use the Business Security Profile Display Data settings to grant or deny access to the data retrieved by objects in the business layer.

By default, a user with access to the universe granted in the repository can see the data retrieved by all universe objects. Once the user is assigned or inherits a profile with a Display Data setting, only the data corresponding to the objects granted by the setting is displayed.

If an object is not granted and not denied explicitly, it is denied by default. Unlike objects that are explicitly denied, objects that are denied by default could be granted by inheritance after aggregating Business Security Profiles to determine the net profile for a user. For more information about aggregating profiles, see the related topic.

The following objects can be granted or denied. The *All objects* option allows you to grant or deny all objects in the business layer.

- Dimensions
- Attributes
- Measures
- Calculated Members
- Named Sets
- Folders: Grants or denies all the objects in the folder.
- Hierarchies

Using the *All objects* option has the advantage that any new object defined in the business layer is automatically included in the display data setting when the universe is published.

If the *All objects* option is used, the settings are aggregated to determine the net setting for this profile, for example:

- If *All objects* are denied and one object is granted, any parent folders in the path to access the object are granted, but only to access the object. The other objects in the parent folders are denied.
- If *All objects* are granted and one object is denied, the parent folders in the path to access the object are denied, but only to prevent access to this object. The other objects in the parent folders are granted.

A user who is denied an object by a Display Data setting might refresh a report containing the denied object. You can specify what the refresh should do in this case by setting the SQL generation parameter AUTO_UPDATE_QUERY in the business layer.

- If this parameter is set to No, then refreshing the document generates an error message, and the document is not or modified.
- If this parameter is set to Yes, then the denied objects are removed from the query and from any filters defined in the business layer the next time the document is refreshed. Data for other granted objects is retrieved and displayed to the user. Denied objects are removed from the document where the objets are used: table columns are removed, formulas return errors, charts show errors, and so on.

Whether or not a user sees data for a particular object is determined after aggregating the Display Data settings in all profiles assigned to the user, and taking into consideration object access level. For more information about aggregating profiles, see the related topic.

Related Information

Business Security Profile Display Data settings [page 339]

12.2.4.5 Business Security Profile Filters settings

Use the Business Security Profile Filters setting to define a filter using objects in the business layer, or named member sets. You create and edit filters explicitly for the Business Security Profile using the Security Editor. Filters in the Business Security Profile are not accessible in the business layer. If the Business Security Profile is deleted, the filter or named set is also deleted.

Once the user is assigned or inherits a profile with a Filters setting, the filter is added to the query script (and therefore combined with any filters defined in the business layer) to restrict the data displayed.

Relational universes

For relational universes, you define filters on dimensions and measures in the business layer. You can define compound filters that are linked by the AND and OR operators. You can also define multiple filters to apply to the query.

When a user runs a query, the filters are always applied to the query and to the returned data. This is different from the Data Security Profile Rows setting which only applies if a defined table is referenced in the query.

OLAP universes

For OLAP universes, you define a named set of members. You can include or exclude members for any dimension in the business layer. The excluded members are removed from the query when data is retrieved from the cube.

> **i Note**
>
> The filter does not impact the aggregation of values in the report. Only the display of members is filtered.

You can include or exclude members from multiple dimensions. You can also define multiple named sets to apply to the query.

Related Information

How to build a business filter [page 310]
Working with the Member Selector [page 316]
Aggregation of Filters settings [page 348]

12.2.5 Security Profile aggregation

More than one Data Security Profile or Business Security Profile defined for a universe may be assigned to the same user. Multiple profiles can be directly assigned to a user or group, and be inherited from parent groups. When this happens, the security settings in the different profiles are aggregated to result in one effective Data Security Profile, and one effective Business Security Profile, called net profiles. The settings in the net profiles are applied when the user creates a query or views a report.

Two methods are used for aggregating security settings: priority and restriction level.

Priority is determined by the order that the Security Profiles appear under the universe in the Security Editor. Use the *Change Data Security Profile Priority* and *Change Business Security Profile Priority* commands to set the priority.

Restriction levels (very restrictive, moderately restrictive, less restrictive) define which operators (for example AND, OR) to use to aggregate profiles. You can change these restriction levels in the Security Editor to affect how the profiles are aggregated.

- The less restrictive level is appropriate when security is designed with roles, each role granting new rights to the user.
- The most restrictive level is appropriate when each profile is used to restrict what the user can see.
- The moderately restrictive level uses the most restrictive level for inherited profiles and the less restrictive level for merged profiles.

The rules for inheriting or merging profiles are as follows:

- If the user or group is assigned Profile A and belongs to a group that is assigned Profile B, Profile A and Profile B are inherited.
- If the user or group belongs to a group assigned Profile A and another group assigned Profile B, Profile A and Profile B are merged.
- If the user or group is assigned both Profile A and Profile B, Profile A and Profile B are merged.

The method and operators used to aggregate profile settings vary for the different settings. For detailed information about aggregation for each type of setting, see the related topic.

The Data Security Profile Rows setting and Business Security Profile Filters setting both generate a WHERE clause to filter the query. The Rows setting is applied first. The WHERE clause in the Filters setting is then applied to the results of the first query. In effect, the two WHERE clauses are aggregated with the AND operator.

Related Information

Aggregation of Connections settings [page 342]
Aggregation of Controls settings [page 343]
Aggregation of SQL settings [page 344]
Aggregation of Rows settings [page 344]
Aggregation of Tables settings [page 345]
Aggregation of Create Query settings [page 346]
Aggregation of Display Data settings [page 347]
Aggregation of Filters settings [page 348]
Changing Security Profile priority [page 335]
Changing security profile aggregation options [page 342]

12.2.5.1 Changing security profile aggregation options

1. In the Security Editor *Universes / Profiles* pane, select the universe.
   The current aggregation options for the universe display on the lower, right-hand side of the editor.
2. For each security setting, select a new aggregation option from the list.
   The options apply only to the universe currently selected.
3. To save the changes in the repository, click the save icon in the main tool bar.

Related Information

Security Profile aggregation [page 341]
Opening the Security Editor [page 326]

12.2.5.2 Aggregation of Connections settings

If more than one Security Profile for a universe is assigned to or inherited by the same user, the connection defined in the Security Profile with the highest priority is used.

> **i Note**
>
> Connections settings for relational universes are in Data Security Profiles and for OLAP universes, in Business Security Profiles. Therefore Data and Business Security Profiles are never prioritized together.

If the relational universe has multiple connections, aggregation of the Connections setting is done for each connection independently.

Related Information

Changing Security Profile priority [page 335]
Business Security Profile Connections settings [page 337]
Data Security Profile Connections settings [page 331]

12.2.5.3 Aggregation of Controls settings

If more than one Data Security Profile for a universe is assigned to or inherited by the same user, the following rules are used to aggregate the Controls settings. The rules are applied to each query limit to determine the value to be used when the user runs a query or report.

Restriction level	Aggregation Rule
Very Restrictive	The limit is active only if it is selected in all merged and inherited profiles. The value used is the minimum value for the limit among all the merged and inherited profiles.
Moderately Restrictive	The limit is active only if it is selected in all inherited profiles and selected in at least one merged profile. First, the minimum value is determined for the limit comparing the inherited profiles. This value is compared to the values among the merged profiles. The value used is the maximum among these values.
Less Restrictive	The limit is active if it is selected in any merged or inherited profile. The value used is the maximum value for the limit among all the merged and inherited profiles.
Priority (default)	The activation and value of the limit in the Data Security Profile with the highest priority is used.

> **i Note**
>
> For a definition of inherited and merged profiles, see the related topic on security profile aggregation.

Related Information

Security Profile aggregation [page 341]
Changing security profile aggregation options [page 342]
Changing Security Profile priority [page 335]

Data Security Profile Controls settings [page 332]

12.2.5.4 Aggregation of SQL settings

If more than one Data Security Profile for a universe is assigned to or inherited by the same user, the following rules are used to aggregate the SQL settings. The rules are applied to each query option to determine the value to be used when the user creates a query.

Restriction level	Aggregation Rule
Very Restrictive	The option is active only if it is selected in all merged and inherited profiles.
Moderately Restrictive	The option is active if it is selected in all inherited profiles and selected in at least one assigned profile.
Less Restrictive	The option is active if it is selected in any merged or inherited profile.
Priority (default)	The activation and value of the option in the Data Security Profile with the highest priority is used.

> **i Note**
>
> For a definition of inherited and merged profiles, see the related topic on security profile aggregation.

Related Information

Security Profile aggregation [page 341]
Changing security profile aggregation options [page 342]
Changing Security Profile priority [page 335]
Data Security Profile SQL settings [page 332]

12.2.5.5 Aggregation of Rows settings

If more than one Data Security Profile for a universe is assigned to or inherited by the same user, the following rules are used to aggregate the Rows settings and determine the WHERE clause to be used when the user runs a query or report.

First, the WHERE clauses for each table are aggregated according to the restriction level:

Restriction level	Aggregation Rule
Very Restrictive (default)	The WHERE clauses in all profiles that apply to the same table are combined with the AND operator.
Moderately Restrictive	Inherited WHERE clauses are aggregated using the AND operator. Merged WHERE clauses are aggregated using the OR operator.
Less Restrictive	The WHERE clauses in all profiles that apply to the same table are combined with the OR operator.

After aggregation according to restriction level, the WHERE clauses for each table are aggregated together with the AND operator to produce the final WHERE clause that is applied to the query.

> **i Note**
>
> For a definition of inherited and merged profiles, see the related topic on security profile aggregation.

Related Information

Security Profile aggregation [page 341]
Changing security profile aggregation options [page 342]
Data Security Profile Rows settings [page 333]

12.2.5.6 Aggregation of Tables settings

If more than one Data Security Profile for a universe is assigned to or inherited by the same user, the replacement table defined in the Data Security Profile with the highest priority is used. If settings are defined for multiple tables, the aggregation is done for each table independently.

Related Information

Changing Security Profile priority [page 335]
Data Security Profile Tables settings [page 334]

12.2.5.7 Aggregation of Create Query settings

If more than one Business Security Profile for a universe is assigned to or inherited by the same user, the Create Query settings are aggregated. Object access levels, if they are defined, are applied to determine whether or not a user sees a particular object in the query panel.

First, the list of views that the user can select in the query panel is determined by aggregating the profiles according to the restriction level:

Restriction level	Aggregation Rule
Very Restrictive (default)	The user can select the view in the query panel only if it is granted in all inherited and merged profiles.
Moderately Restrictive	The user can select the view in the query panel only if it is granted in all inherited profiles and granted in at least one merged profile.
Less Restrictive	The user can select the view in the query panel if it is granted in any inherited or merged profile.

Once a view is selected in the query panel, an object appears if it is included in the view, and if it is not explicitly denied after aggregating the profiles according to restriction level:

Restriction level	Aggregation Rule
Very Restrictive (default)	The object is denied if it is explicitly denied in any inherited or merged profile.
Moderately Restrictive	The object is denied if it is explicitly denied in any inherited profile and denied in all merged profiles.
Less Restrictive	The object is denied only if it is explicitly denied in all inherited and merged profiles.

After aggregation, the denied objects are not displayed even if they belong to a granted view. If a folder is denied, then all sub-folders and objects in the folder are denied.

Finally, the access level granted to the user in the Central Management Console determines which of the objects granted by the net Business Security Profile are available in the query panel. The user sees only the objects with an access level lower than or equal to his authorized access level. You assign access levels to objects in the *Business Layer Editor*.

> **i Note**
>
> For a definition of inherited and merged profiles, see the related topic on security profile aggregation.

For more information about object access levels, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.

Related Information

Security Profile aggregation [page 341]
Changing security profile aggregation options [page 342]
Business Security Profile Create Query settings [page 338]

12.2.5.8 Aggregation of Display Data settings

If more than one Business Security Profile for a universe is assigned to or inherited by the same user, the Display Data settings are aggregated. Object access levels, if they are defined, are applied to determine whether or not a user sees the data for an object in the business layer.

First, the list of objects that the user can see data for is determined by aggregating the profiles according to restriction level.

Restriction level	Aggregation Rule
Very Restrictive (default)	The data appears only if it is granted in all inherited and merged profiles.
Moderately Restrictive	The data appears only if the object is granted in all inherited profiles and granted in at least one merged profile.
Less Restrictive	The data appears if the object is granted in any inherited or merged profile.

If a folder is denied, then the data for all objects in the folder and its subfolders is denied.

Finally, the access level granted to the user in the Central Management Console determines which of the objects granted by the net Business Security Profile the user sees data for. The user sees data only for the objects with an access level lower than or equal to his authorized access level. You assign access levels to objects in the *Business Layer Editor*.

> **i Note**
>
> For a definition of inherited and merged profiles, see the related topic on security profile aggregation.

For more information about object access levels, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.

Related Information

Security Profile aggregation [page 341]
Changing security profile aggregation options [page 342]
Business Security Profile Display Data settings [page 339]

12.2.5.9 Aggregation of Filters settings

If more than one Business Security Profile for a universe is assigned to or inherited by the same user, the following rules are used to aggregate the filters settings and determine the filter to be added to the query script when the user runs a query or report.

For relational universes, the filters are aggregated according to the restriction level. The resulting filter is added to the WHERE clause applied to the query.

Restriction level	Aggregation Rule
Very Restrictive (default)	The filters in all profiles are combined using the AND operator.
Moderately Restrictive	Inherited filters are aggregated using the AND operator. Merged filers are aggregated using the OR operator.
Less Restrictive	The filters in all profiles are combined using the OR operator.

For OLAP universes, the named sets are aggregated according to the restriction level.

Restriction level	Aggregation Rule
Very Restrictive (default)	The user sees a member only if it is included in every named set defined in all profiles.
Moderately Restrictive	The user sees a member if it is included in every named set defined in the inherited profiles, and included in at least one named set defined in the merged profiles.
Less Restrictive	The user sees a member if it is included in any named set defined in any profile.

> **i Note**
>
> For a definition of inherited and merged profiles, see the related topic on security profile aggregation.

Related Information

Security Profile aggregation [page 341]
Changing security profile aggregation options [page 342]
Business Security Profile Filters settings [page 340]

12.2.6 Assigning Security Profiles to users

1. In the Security Editor *Universes / Profiles* pane, select the universe.
   Any currently assigned users or groups appear in the *Assigned Users* list.
2. To assign, select the user or group in the list of users on the right-hand side of the editor, and click the arrow that points to the *Assigned Users* list.
3. To unassign, select the user or group in the *Assigned Users* list and click the arrow that points to the list of all users.

   > ⚠ Caution
   >
   > The double-arrow icon >> unassigns all users and groups whether they are selected or not.

4. To save the changes in the repository, click the save icon in the main tool bar.

Related Information

Opening the Security Editor [page 326]

12.2.7 Displaying profiles assigned to a user and previewing net profiles

1. In the Security Editor, click the *Users / Groups* pane on the left-hand side of the editor.
2. In the *Users / Groups* pane, select the user or group.
3. In the *Universes / Profiles* pane in the upper right-hand side of the editor, select the universe.

   > → Tip
   >
   > You can change the display to list only universes that have profiles assigned to the selected user or group by selecting the *Display only universes assigned to the selected user/group* option.

   Once you have selected a user and a universe, the assigned profiles appear in the profiles list on the lower right-hand side of the editor.
4. To preview the net Data Security Profile or net Business Security Profile, click *Preview Net Profile* below the corresponding profile list.

   The Data Security Profile or Business Security Profile editor opens in read-only mode. The settings on each tab represent the settings that will be used after aggregation of all the profiles assigned to the user is taken into account.

Related Information

Security Profile aggregation [page 341]
Opening the Security Editor [page 326]

12.3 Locking universes

To prevent other users working on a universe that you are modifying, you can lock the universe. To lock the universe, right-click on the universe in the *Repository Resources* view and select *Lock Universe*. When a universe is locked, a padlock symbol is displayed next to the universe name in the *Repository Resources* view. The universe can only be unlocked by the person who locked it, or the Administrator. To unlock a universe, right-click it and select *Unlock Universe*.

12.4 CMC rights for information design tool users

The application, universe, and connection rights necessary to do tasks in the information design tool are summarized in this topic.

Rights are granted in the Central Management Console (CMC). How to define these rights is described in the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.

To open a session on the repository system and do all secured tasks in the information design tool:

- You must have a username and password configured by the system administrator in the CMC for the CMS where the repository is stored.
- You must have the right *Connect the CMS with Information design tool and view this object in the CMC* granted in the CMC.

Additional rights needed are listed in the table by task.

Task	Rights required
Publish a connection to a repository	• *Create, modify, or delete connections* application right • *View objects* right on the connections folder • *Add objects to the folder* right on the connections folder
Edit a secured connection from the Repository Resources View	• *Create, modify, or delete connections* application right • *Add objects to the folder* on the connection folder (to create) • *Edit objects* connection right • *Download connection locally* connection right (relational connections only)
Use a the local middleware driver for a secured connection	• *Download connection locally* connection right (relational connections only)

Task	Rights required
Publish a universe to a repository	• *Publish universes* application right • *View objects* right on the universes folder • *Add objects to the folder* right on the universe folder • *Edit objects* universe right (to republish)
Retrieve a published universe from a repository	• *Retrieve universes* application right • *View objects* right on the universe folder • *View Objects* universe right • *Retrieve universe* universe right
Edit secured local resources	• No rights are required, but the user must provide the CMS authentication of the user who saved the resources.
Unsecure local resources	• *Save for all users* application right • *Retrieve universes* application right • *View objects* right on the universes folder • *View Objects* universe right • *Retrieve universe* universe right • *Save for all users* universe right
Open the Security Editor	• *Administer security profiles* application right
Define security profiles	• *View objects* universe right • *Edit security profiles* universe right
Assign security profiles to users and groups	• *View objects* universe right • *Assign security profiles* universe right
Run a query on a published universe	• *View objects* universe right • *Create and edit queries based on this universe* universe right • *Data Access* universe right • *View objects* right on underlying connections • *Data Access* right on underlying connections
Share project resources: • Share a local project • Open the Project Synchronization View • Synchronize project resources • Lock and Unlock resources • Rename or delete a shared project	• *Share projects* application right
Convert a .unv universe stored in the repository	• *View objects* right on the universe folder • *Add objects to the folder* right on the universe folder • *View objects* universe right
Compute statistics for a multisource universe	• *Compute statistics* application right • *View objects* universe right

Task	Rights required
Delete a universe from the repository	• *View objects* universe right • *Delete objects* universe right
Delete a connection from the repository	• *Create, modify, or delete connections* application right • *View objects* connection right • *Delete objects* connection right

Related Information

Managing a session [page 80]

13 Segmenting your data with Sets

A set defines a sequence of related lists of values defined directly on a universe, and combined in the information design tool, to create a complex query filter available to the SAP BusinessObjects Web Intelligence Query panel. You create sets using the Set editor.

Related Information

Introduction to sets [page 353]
Setting up the sets services on the CMC [page 359]
Managing set containers [page 360]
Designing sets [page 364]
Building a set [page 372]
Publishing a set container [page 391]

13.1 Introduction to sets

You build sets in the information design tool by creating, restricting, and combining lists of values from universe objects. When you publish a set to the repository, Web Intelligence users can use the set as a query filter in the SAP BusinessObjects Web Intelligence Query Panel.

Understanding your lists to define sets

When you ask a question about some aspect of your data, most of the time you will end up with one or more lists of values. These could be lists of customers, products, sales, user satisfaction indices, and all of it will be stored in some form of list. If you are looking for something specific, then you would consider only the part of a list that interests you, and then compare it with other lists that may also show a limited number of values based on your search focus.

For example, you may be planning a marketing campaign to boost sales of a line of your products. As you don't want to waste money targeting the wrong customer group, you would probably consult your product line, sales, and customer lists. Then you would look at more limited lists, those customers who: belong to a certain age group, have bought related products, buy predominantly from the online store, are single, are employed in a technology related industry, and so on.

How do you create and manage these lists? What is the best way to limit or restrict the values in a list, then compare the results with other ones? Once you have finished comparing and restricting the values in different

lists, wouldn't it be useful to be able to use the sequence of list combinations for other questions? Perhaps over different calendar periods, or for different regions?

One solution that SAP BusinessObjects provides for handling multiple lists, is to allow the data in these lists to be accessed by objects in a universe built by the information design tool. The universe is available as a datasource to the SAP BusinessObjects Web Intelligence Query Panel, and you can follow a line of questioning about your data by building queries combining these objects.

You can restrict the list of values by creating query filters. However, if you have multiple lists with many filters, a query can quickly become quite complex. You would save a lot of time if all of the lists and filters that you use regularly were combined into one object. This is what you can do with a set; create a single object that combines any combination of lists into a single filter. A subject is the central theme of a set, for example customers, accounts, or products. It is based on a single dimension in the business layer that has a primary key defined in the Keys tab of the dimension properties.

A set allows you to combine multiple lists of values with their restrictions, into a single repository object that is available to SAP BusinessObjects Web Intelligence users in the Query Panel.

Viewing the membership values for a set

In the *Sets* outline, you can right-click on a saved set and select *Show Set Values* to display the associated membership values (if any) for this set.

Set tables and column collation in SQL Server

When the information design tool creates any tables for sets which are related to columns in your SQL Server database, the tool automatically detects collation for the referenced columns and ensures all materialization tables storing data from those columns are created with the same collation.

Related Information

Managing set containers [page 360]
Designing sets [page 364]
Overview of a set to filter life cycle [page 354]

13.1.1 Overview of a set to filter life cycle

You create a set by combining lists of values based on most universe objects, then build and publish the set to the repository. When you build the set, it saves the data to a database column. As a set designer you need database access to write to the set tables. The set is called automatically at run time into the Query Panel when

the universe is used as the datasource. The set itself is not an object in the universe, but is part of an independent object in the repository called a set container associated with its source universe.

Here are the main stages in a set filter's life cycle:

Set Filter Life Cycle

Stage	What you need to do
Enable set creation by creating a set container.	The administrator creates the set container based on a universe. Select a universe in the repository, then create a set container that will hold the subjects, sets, and the calendars used for time based sets. You must define your calendar(s) here if you are going to create temporal sets. You can only have one set container, but the set container can contain many sets. The published new container is just a placeholder for the final sets that will be created and used.
Design and build the set(s).	The person designing the sets retrieves the set container to a local project, then starts designing the set definition using the data available and a subject. A set contains data from a single subject and can be used to filter on most universe objects and calendar periods. If you have not finished developing the set, you can make the set *Hidden* so that the end users cannot see or use the set. Sets can be set to *Active*, *Hidden*, or *Deprecated*. • When you *Build* a set, the set tables are created and filled in the database. Tables are actually created when a first Set is built, either via an ad-hoc build from the information design tool, or via scheduling in the CMC. • When you refresh a set, its membership is updated according to the set definition • When you *Purge* a set, the database tables are emptied.
Define the set as hidden.	When a set is hidden, when it is published to the repository, it cannot be seen in SAP BusinessObjects Web Intelligence and SAP Crystal Reports. This prevents users from using a set filter that is not ready, or is not intended for use. Use this to create a technical set, simplifying maintenance of the business set
Freeze the set	In the set edition pane, click *Frozen* to freeze the set. When you freeze a set, it cannot be edited or manipulated. You can now tag specific Sets as references, so they can't be edited or built/purged any more. Only a change of name or description is allowed. You can use this to create a reference for subsequent analysis of return on investment.

Stage	What you need to do
Publish the set container.	Publish the set container to the repository. The sets are made available at runtime to SAP BusinessObjects Web Intelligence and SAP Crystal Reports as query filters associated with the universe.
Materialize your sets schema to a different database.	In order to avoid saving the set membership to the original source database, you can materialize to a defined location.
Schedule the refresh of the set container.	You can schedule the times to refresh the set membership in the Central Management Console.
Declare a set as deprecated.	When a set is replaced by a different set, or if the set is no longer maintained with respect to the database objects, declare the set filter as deprecated. Deprecated sets cannot be used to create new sets.You can schedule or refresh reports based on deprecated sets or groups.

See Related Information for an end to end procedure that gives a more detailed overview of the creation process, and information on the different roles used to set up and build the set.

13.1.2 Prerequisites for working with sets

Before you can start creating and publishing sets with the information design tool, you need to take note of the following conditions and required pre-requisites.

Before you work with sets you need	Applies to	Description
Database access rights	*Set Designer*	You need to have the following database permissions to ensure that the appropriate statements are executed when building a set in the database: • CREATE TABLE • INSERT • UPDATE • DELETE • SELECT • DROP TABLE Each subject will have its own table for all sets based on this subject. The table is only created when the first set on this subject is built. Each Set container will have 2 additional tables for storing set meta-data. These are created when the first set in the container is built.. **i Note** A second connection can be used by the Set Designer to create set tables. The initial universe connection is usually read only. However, when building sets, the Set Designer must use the connection with write access to create the set tables.
CMS access rights	• *Universe Designer* • *Set Container Administrator* • *Set Designer*	You need to have the administrator rights to create a connection to a data source, and publish files to the CMS repository. This is defined in the CMC.
Universe access rights	• *Universe Designer* • *Set Designer*	You need to have Universe Designer rights to open a business layer and data foundation. You create sets independently of the universe, but you need to access object properties and the data foundation for certain set management tasks.
Primary key defined for dimensions to be used as set subjects.	*Universe Designer*	A subject is based on a single dimension in the business layer that has a primary key defined in the *Keys* tab of the dimension properties. Additionally, dimensions used for a subject must match the following criteria: • is based on one table • does not use @ constructs • is of a simple type • does not involve prompts or contexts

13.1.3 Collation enhancement for SQL Server

On SQL Server there can be multiple Databases used for sets:

- The customer database
- The materialization database (where the set tables are created with data from the database columns)
- TempDB used by SQL Server for temporary tables. Sets also use this database.

If the database or string columns have a different collation configured then when comparing the columns against each other, SQL Server may throw an error. This is because SQL Server can't resolve the collation conflict between the capitalization in the column names when they are compared.

When you create any tables for sets which are related to the customer columns in SQL Server, all materialization tables storing data from those columns are now created with the same collation (this includes temporary tables in the TempDB).

13.1.4 Restrictions when working with sets

The following restrictions apply when working with sets.

Restrictions

Feature	Restriction
Subjects containing Unicode values	Set subjects cannot be based on a table column containing Unicode values.
Set table names	Set table names should only contain the following characters: abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789 underscores The set table names must not start with an underscore as the first character.
Set container	You can only have one set container, but the set container can contain many sets. Manage set containers/When you add a new Calendar/Start Date of first and last Periods, the date is truncated
Manage sets/Calendar editor	In the dialog for creating a calendar, the user interface string incorrectly mentions deletion, when it should refer to selecting the start/end dates. "Start Date of first and last Periods:" was wrongly replaced by "Delete Sets"

Feature	Restriction
Set: Visual Data step: custom value for date type objects is not supported.	Only use the proposed data formats: integer and string.
Set: Visual Data step: custom values for numeric type.	Only use integers for custom numeric type values in the Visual Data step editor.
Set: Visual Data step: Measures must be at the last step position in a set.	Once you have added a measure to your Visual Data query, you can only add measures.
Set: Query Panel step: incompatible objects are not supported.	Ensure your queries consist of objects that are compatible with the other steps in the set.
Sets on Sets editor	You can only add existing sets. Currently, hidden sets do not appear when they are used in Sets on Sets.
CMC: Set container scheduling:	Scheduling is limited to single set or group

13.2 Setting up the sets services on the CMC

You can isolate the sets materialization service inside a new Adaptive Processing Server (APS)

Isolating the sets materialization service can be useful to create a dedicated Adaptive Processing Server for various reasons. You also need to add the Adaptive Connectivity Service inside the same APS. This can be achieved by cloning an existing APS. You must be an admistrator to do perform this task. Here's how to do this:

1. In the CMS, select *Web* > *Servers* > *Server List*
2. Create the server, choose Category - *Core services*
3. Choose *Sets Materialization Service*
4. Click *Next*
5. Add the *Adaptive Connectivity Service* to the right panel
6. Click *Next*
7. Give the server a name and click *Create*
8. Stop the original `MYSIAAaptiveProcessingServer`
9. Right-click the `MYSIAAaptiveProcessingServer` file and choose *Select services*.
10. Remove *Set Materialization* (this forces the new server to be the only one running this service, thus isolating it)
11. Check two things in the properties pane of the new APS:
    - Check that the command line parameters contains the line `"-Dbusinessobjects.connectivity.directory=C:/Program Files (x86)/SAP BusinessObjects/SAP BusinessObjects Enterprise XI 4.0//dataAccess/connectionServer"` (this is an example, but depends on your installation).
    - Check the option *Automatically start this server when the Server Intelligence Agency starts*

12. Right-click on the new server and select *Enable*.
13. Start the original server and the new server.

This service is transparent to the information design tool.

Setting the information design tool to start in simplified sets editing mode

You must configure the information design tool .ini file so that the tool starts up in simplified sets editing mode. To do this:

1. Open the `informationdesigntool.ini` file
2. Add the line: `-Dsets.simplified.perspective=true`
3. Start `informationdesigntool.exe` and the tool will start in sets editing mode only.

13.3 Managing set containers

You need a Set Container before you can start creating sets. You use the Set Containers Management dialog to do the following:

- Select or Remove subjects from the set container you are creating/editing, and add a description for each subject, if required. You can also create multi-key subjects (click *Add Subjects*).
- Create or edit calendars for temporal sets
- Publish your sets schema to a different database to your production environment.
- Create, edit, or remove containers.

When you create the Set Container, A Set Container is a `.sets` file that specifies the subjects and calendars that you can use as the basis for you sets. It is created from a universe (.UNX) and published to the CMS repository. You then retrieve the Set Container to a local project where Subjects are automatically detected by the information design tool and made available to the Set Container. When you design your sets, the set definition is stored in the Set Container. Calendars are available to all set containers. When you have finished creating your sets, you publish the Set Container to the repository. The sets can now be used by SAP BusinessObjects Web Intelligence.

Usage of Set Container is compatible with concurrent set authoring. The system automatically merged existing sets and created sets when publishing local container or retrieving the container from repository.

> **i Note**
>
> You can only create one set container for each universe.

Related Information

Creating a set container from a universe [page 361]

13.3.1 Creating a set container from a universe

You create and manage a set container using the *Set Containers Management* dialog box, accessed by right-clicking a single source relational universe (.UNX) in the repository. Before you can create a set container, ensure that the following pre-requisites are met:

- The universe must be a single source relational universe.
- You have the appropriate permissions required for the *Set Container Administrator* role.
- The universe has been set up to allow the use of sub-queries.
- Dimensions in the business layer that you want available as compatible subjects. *Set Containers Management* automatically detects these dimensions. If the business layer does not have any primary key dimensions, then no subjects will be available for the set container.

You have the following information and options to create and manage a set container available in *Set Containers Management*:

Set Containers Management tabs	Description
Subjects	Subjects detected in the universe are listed. Click *Synchronize with universe* to update the list with the latest version of the universe and potentially add new subject candidates. Existing subjects might no longer meet the requirements which led to their initial creation, because dimensions have disappeared or lost their primary key.
Calendars	Calendars available to all temporal sets are listed. Click *Add a new Calendar* to create a new calendar or click the ellipsis button to modify an existing one in the list.
Database	Useful to provision set data in a dedicated store and manage security accesses.
Deployment	Current set containers associated with the universe are listed. Click *Add a new Container* to create a new container. **i Note** For this release, only one set container is available for each universe.

Do the following to create a new Set Container:

1. Right-click a universe in the *Repository Resources* tab then select *Manage Set Containers* from the drop down list.

   The set container manager appears.
2. Click the *Subjects* tab.

   The subjects that have been detected in the universe are listed. A membership table with a default name is listed for each subject in the *Membership table* column. A membership table contains the data for a subject. It is created in the database when a set based on this subject is built for the first time. You can remove unwanted subjects from the container, they will not be available to the sets you want to create.
3. If you want to change subject or membership table names, double-click a name and type a new one. You can also accept the default names, however, you may prefer to give them a more relevant one, since once the set container is published, you can't modify subject or membership table names.
4. Click the *Calendars* tab.

   This tab lists the calendars available to the set container. When you create a set container, the list is empty.

5. If you want to create temporal sets to evaluate changes in your data over calendar periods, then create a calendar for your Sets Container by doing the following:
   - Click the *Add a new calendar* button.
   - In the *Calendar editor*, type a name for the calendar, select its period duration, the number of periods to apply it to, and pick a start period date, then click *OK*. You can refer to the section on creating calendars in Related Topics for more information.
6. Click the *Deployment* tab.

   This tab lists the set containers available. When you create a set container the list is empty.
7. Click the *Add a new Container* button.

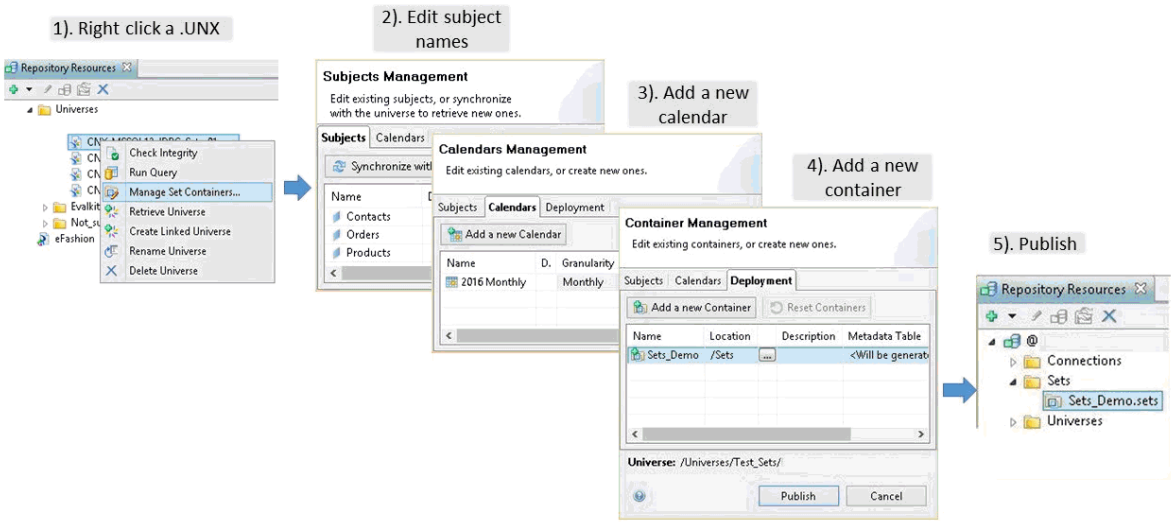
   Default set container name, location, and table names are automatically entered in the list. The Metadata and History tables are created in the database when the set container is built for the first time. These are used to show data changes when sets are built.
8. If you want to change the default values, do the following:
   - Double-click the set container name, the metadata table or history table names and type new ones.
   - Click the ellipsis button in the Location column to select a different repository path indicating where to store the set container.

   You can also accept the default names and location, however, you may prefer to give them relevant names as once the set container is published, you can't modify the set container name, the associated table names, or the project path.
9. Click *Publish*.

   A new set container is added to the *Sets* folder in the *Repository Resources* pane. You will retrieve this Set Container to your Local Project folder and build your sets locally.

Example

The following animated image shows an overview of the procedure and a worked example. The example does not show use of the *Database* tab which was a recent addition to the tool. The animation may not be available in some reading formats that do not support animated gifs.



13.3.2 Retrieving a set container to build a set

A Set Container published to the repository has the file extension .SETS. Before you start authoring sets, you initially need to retrieve the set container from the repository and save it to a local project. You can then open the local set container in the *Sets* editor in the information design tool and start defining and developing sets based on the subjects in the set container. The sets are saved when you save the set container locally. When you publish to the repository it updates the current version.

1. In the *Repository Resources* pane, browse to the `Sets` folder and right-click the Set Container `.sets` file.
2. Select *Retrieve Sets Container* from the context menu.

   The *Select a Local Project* selection box appears. It lists the local projects you have defined in the information design tool.
3. Browse to and select the project to receive the set container, then click *OK*.
4. The set container is copied to the local project and opens automatically in the *Sets* editor.

Related Information

Managing set containers [page 360]

13.3.3 Advanced subjects creation

You can combine two or more subjects to create a composite subject that is made up of multiple primary keys. Select the first subject that you want to combine, the tool automatically shows only you only subjects with a compatible primary key. Select one or more compatible subjects. Click *OK* to create the composite subject or *Cancel*. In the *Containers Management* pane, you can add a name and a description for your composite subject. The composite subject is listed in the available subjects when you create your sets.

> **i Note**
>
> Subjects making up composite subjects are not necessarily public and can be removed from the container, without compromising their composite subjects.

13.4 Designing sets

A set is made up of one or more lists of values that are linked as a series of steps, each list restricting the values available for the following ones. When published to the repository, the set appears as a filter object available to the Query Panel in SAP BusinessObjects Web Intelligence and to SAP Crystal Reports. If you are looking for something in your data, then the step based sequence of lists that make up the set will help you find it, while at the same time hiding the potential complexity behind its construction.

You create a set based on a subject defined in a set container. The subject is typically built on a business layer dimension. The members of the dimension make the data available to build the set. You can create two types of set:

Set type	Description
Static	The set members are fixed at a moment in time. A static set is like a snapshot of your data that captures a state of the population that you are considering for analysis. For example the Gold customers targeted in a marketing campaign, where the campaign is an event and the Gold customers are the facts associated with that event.
Temporal	The set members are dynamic, so can change over multiple calendar periods used to build the set. A temporal set is a collection of members who joined, left or stayed over each calendar period. A calendar is defined by a begin and end date - for example from January 1st 2015 to December 31st 2015 - a certain number of periods, and a duration for each period. In a temporal set, periods do not overlap, and have all the same duration.

A new set is empty. You need lists with members to make it useful. You can add members by using one of the following:

- *Insert Visual Data*: Selecting the set members from a tree view by drilling down on business items, showing their values and counts.
- *Insert Sets on Sets*: Combining any of the sets defined on the same subject? OR ON A COMPATIBLE ONE. You do this graphically by combining set collections that can contain multiple sets to create combinations of sets using intersection, union, or exclusion of sets.

- *Insert Query Panel* Step: Unleash the full power of the Query Panel.

You add lists of members in a series of steps that correspond to a sequence of questions. The logic of the questions is to refine your data, so you create steps of member lists in a sequence that you can add, subtract, or keep based on the previous list of members.

Once you have defined your new set, you can build it (this is optional), then save it to the set container in your local project. If you publish the set container to the repository, the set is made available in the Query Panel for SAP BusinessObjects Web Intelligence and SAP Crystal Reports, where it appears as a set filter.

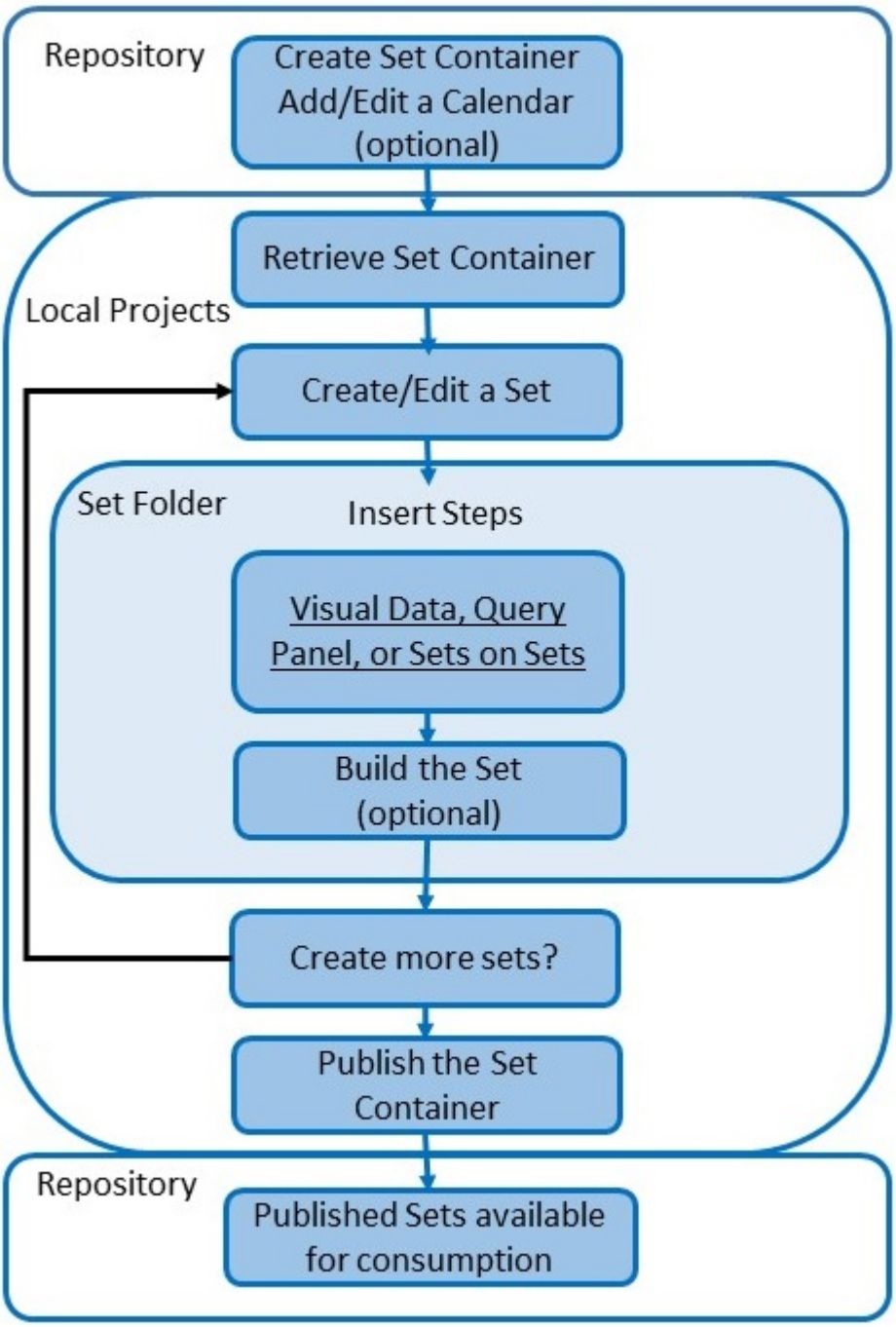
> **i Note**
>
> A set that has been built, but remains non-published, is always available in the preview panel of the Sets editor unless it is marked as hidden or deprecated.

This is an overview of the process you follow to design then build a new set:

What you do to build and design a new set	Go here for information
Retrieve and open the set container.	Retrieving a set container to build a set [page 363]
Choose the type of set. It can be either a static or temporal set.	• Defining a static set [page 373] • Defining a temporal set [page 376]
Choose the members of the set visually by: • selecting the members from a list • combining any of the subject-compatible sets in the container to create a new set. • using the querying capabilities of the *Query Panel*.	• Inserting a visual data step in a set [page 383] • Creating a Sets on Sets Step [page 385] • Inserting a Query Panel step [page 387]
Build your set and publish it to the repository.	• Building a set [page 372] • Publishing a set container [page 391]

13.4.1 Overview of how to create a set

You create your sets in a container. The administrator creates the container and calendars for temporal sets, then the container is retrieved locally. You then create a set and insert steps according to your requirements. You can hide work in progress and declare sets as deprecated when necessary.



Related Information

Designing and managing sets using the sets editor [page 367]
Defining a static set [page 373]
Defining a static set based on a measure (Visual Data step) [page 375]
Defining a temporal set [page 376]

Set steps [page 381]
Building a set [page 372]
Publishing a set container [page 391]
Editing set display options [page 369]
Specifying how set filters are used (Consumption tab) [page 370]
Previewing a set as a filter in the query panel [page 372]
Hiding a set [page 390]

13.4.2 Designing and managing sets using the sets editor

When you retrieve a set container to a project, the container opens in the Sets Editor. The Sets Editor is where you create and edit your sets. You start in the left pane to create an empty set definition, then define the lists of values and properties for your set in the right pane by working your way through a series of tabs.

To help you get started creating your static and temporal sets using the editor, here is a description of its components and an explanation of how you use it. Refer to Related Information for the more detailed procedures to create the different types of sets:

Sets Editor Pane Components	Description
Set Browser: The left side pane with the tree view of your sets.	
Insert Static Set	Click to insert an empty static set. Click directly on the icon to create a set that is based on the current subject. Click the down arrow next to the icon to select another subject, that will become the current one.
Insert Temporal Set	Click to select a calendar to insert a temporal set. Click directly on the icon to create a set that is based on the current subject. Click the down arrow next to the icon to select another subject, that will become the current one.
Preview Outline in Query Panel	Click to view the set as a set filter in the Query Panel.
Edit Display Options	Click to specify which type of sets are shown in the Set browser based on whether the set is Temporal, Hidden, Published, and more.
Set Definition and Properties: The right side pane which contains zones to define properties and tabs to define your sets.	
Name and Description	Enter the name, and a description to add a commentary for the set.

Sets Editor Pane Components	Description
Steps tab	Each list of values in a set is called a Step. Use this tab to select and insert data values for each list of values. *Insert Visual Data* allows you to manually select values from a hierarchy of business objects, *Insert Sets on Sets* gives you the choice of combining sets to create a list based on the union or intersection of the sets. *Insert Query Panel* step enables you to create a complex filter by using the Query Panel. *Edit Step* opens the appropriate step wizard to modify your values choices. *Build* actually creates your set by inserting its values in a database column. *Purge* allows you to purge a built set. *Temporal Management* gives you options to customize the calendar periods actually built by an auto-fill temporal set and to set rollback and auto-fill options.
Time tab	Shows you information on the type of temporal set and its calendar. You can change the Auto-fill nature of a temporal set only if the set has no membership, right after creation time, or after a complete purge.
History tab	Shows you statistical information for a built set. You can see at a glance the members who have joined and left the set over the calendar period. In the graph section you can select a chart that tracks the joiners and leavers for any part of the associated calendar period. You can also click the Save icon at the right of the tab to save the graphic as an image file. For a static set, members should not change over time, unless you modify its definition, or your data changes. *Build* actually creates your set membership by inserting its values in a database column. *Purge* allows you to purge a built set. Use the *Temporal Management* option to customize the calendar periods actually built by an auto-fill temporal set and to use rollback and auto-fill features.
Consumption tab	Shows you various pieces of information about how this set will behave at consumption time, which can be tested beforehand via the dedicated Query Panel: • Its preferred location in the Query Panel's outline (e.g. after this particular business item, as the last child of this folder, etc.) • Its visibility through security access level • Its visibility through state (active, hidden, deprecated) • Its visibility through its mandatory nature (same notion as a standard filter)
Repository tab	Shows you information about the set, set container and universe that you are using, such as: • The repository path of the universe associated with this particular set container • The repository path of the set container • The various revisions of the set, the set container and the universe • The last publisher of the set, set container and universe

Related Information

Defining a static set [page 373]
Defining a temporal set [page 376]
Previewing a set as a filter in the query panel [page 372]
Editing set display options [page 369]

Sets Editor default options [page 417]

13.4.3 Working with sets in set editing mode

When the information design tool is configured to launch in Set editing mode, the only resources you are allowed to edit are Sets. Therefore, the Local Projects and Repository panes are not visible. Furthermore, the toolbar and menus have been simplified to accommodate for this focus on sets. This is useful to shield Set authors from universe concepts and local/remote resource paradigm. It is meant to reduce learning curve and increase audience to less technical operators. A user can do the following:

- Create, edit, delete, and build sets
- Create, edit, delete, and build groups
- Save and Publish sets and groups. Save and Publish are merged into single operation so use get the perception that saving Set makes it public.

Pressing Ctrl+S will transparently auto-publish the set container if no concurrent authoring issues were found.

Related Information

Starting the information design tool in set editing mode [page 17]

13.4.4 Editing set display options

A set container is designed to contain hundreds, or even thousands of sets. Filtering capabilities have been provided so you can quickly display only the relevant ones for your daily tasks. Filtering criteria include set name, visibility in the Query Panel, publishing status, or even subject.

A subject is the central theme of a set, for example customers, accounts, or products. It is based on a single dimension in the business layer that has a primary key defined in the *Keys* tab of the dimension properties. A set contains data from a single subject and it is the subject that determines the data available to the new set.

1. Open a set container in one of the following ways:
   - Retrieve a set container from *Repository Resources* tab. **Or**
   - Double click a set container from *Local Projects* pane.
2. In the *Sets* pane of the Sets editor, click the *Edit display options* filter icon at the top of the pane.

   A display options box appears. By default, all options are selected. You have the following options:
   - *Static*: Display static sets
   - *Mandatory*: Display mandatory sets
   - *Filter by name*: filter the sets by name
   - *Active*: display sets with Active visibility level
   - *Hidden*: display sets with Hidden visibility level
   - *Deprecated*: display sets with Deprecated visibility level

- *Mandatory*: display sets that are mandatory
- *Non Mandatory*: display sets that are not mandatory
- *Published*: display sets that are already available in the repository
- *Not Published*: display sets that are local to this information design tool
- *Show sets created under*: The subjects available in the set container are listed.When a subject is selected, only sets defined for this subject are displayed in the *Sets* pane.

3. Select one or more display options.
4. Click anywhere outside of the box to close it.
5. You can also relocate the popup window to a more suitable location on then later dock/reattach it back to its initial location via a dedicated button in its title bar. .

Related Information

Defining a static set [page 373]
Defining a temporal set [page 376]
Designing sets [page 364]
Sets Editor default options [page 417]

13.4.5 Using scoring in your sets

You can provide scoring for set members to differentiate between them. This allows you to score different aspects of members of a set and aggregate the scores generated by the different set steps, then analyze the results. The members with the the highest score are the members who were present in most or all of the steps.

The domain expert assigns scores to selected members of a set step.

A score is assigned to a member who is present for that specific step. You can assign a score to any type of set step. The score is totaled per member as the steps are built alongside their containing set.

In temporal sets, scores can change with time, but only the last score from the last built set can be obtained.

13.4.6 Specifying how set filters are used (Consumption tab)

You can set the following options for the set that determine set filter properties in the Query Panel:

Set filter property	Option	Description
Outline	*Position*	Indicates the position of the set filter in the object viewer of the Query Panel. For example `BEFORE Dw Crm Product\Product Id` shows that the system will try to position the set filter immediately above the Product ID objet in the Query Panel.

Set filter property	Option	Description
	Change	Click to select a new position in the Query Panel object browser.
	Reset	Click to reset the default position for a set filter which is immediately under the subject dimension in the object browser. Visibility can also be reset.
	Visibility	The set filter can be set to be: • Active: Visible in the Query Panel • Hidden: Not visible in the Query Panel. This option can be used for a set that is in progress. • Deprecated: Not visible to discourage continued use. SAP BusinessObjects Web Intelligence reports will break until fixed.
Security	*Accessible by users with Object Security level not less than*	You set the availability of the set filter for SAP BusinessObjects Web Intelligence users with the selected security profile (and above).

1. Click a set in the set browser.
2. Click the *Consumption* tab in the *Set Properties* pane to the right of the set browser.

   The default position of the set filter in the Query Panel's object tree view appears in the *Position* field.
3. Select the set filter options as described in the previous table.

13.4.6.1 Re-positioning a set filter in the list of available objects

A set appears as a set filter object in the Query Panel immediately under its subject dimension in the list of available objects. You can modify this default positioning as follows:

1. Open a set in the Set Editor.
2. Click the *Consumption* tab.

   The *Outline* pane displays the current position of the set filter in the Object pane of the Query Panel.
3. Click the *Change* button.

   A list of available objects appears. The current position of the set filter is indicated by the two boxes at the top of the list. You can select either a position after or before a selected object in the list, or as the first or last child of a selected folder.
4. Click an object in the list that you want to use to position the set filter.
5. Select either Before, After, First or Last from the drop down list at the top left of the list, depending on whether you want the set filter to be positioned before or after the selected object in the object tree view.
6. Click *Validate*.
7. Click the *Preview Outline in Query Panel* icon in the tool bar at the top of the Set Browser pane, and browse to the new placement of the set filter.

The set filter is positioned as close as possible the newly indicated place in the available objects list.

13.4.7 Previewing a set as a filter in the query panel

You can preview the set filter for a built set in the available objects pane of the Query Panel.

1. Click the *Preview Outline in Query Panel* icon in the tool bar at the top of the Set Browser pane.

   The Query Panel opens.
2. Expand the folder that contains the subject dimension used for the set.

   By default, the set filter is positioned under its subject dimension in the list of available objects, or as close as possible to the new location you provided in the *Consumption* tab.

Related Information

Re-positioning a set filter in the list of available objects [page 371]

13.4.8 Filtering objects in the set editor

To filter the objects in the *Set Editor*, click the Filtering Options button in the side panel. You can filter by the following:

- *Type* (Static, Temporal, or Limited Temporal)
- *State* (Active, Hidden, or Deprecated)
- *Mandatory* (show mandatory or non-mandatory sets)
- *Published* (Published or Local sets)
- *Security* level (Published, Controlled, Restricted, Confidential, or Private)
- *Membership* (Members with data or without data)
- *Subjects* (select the subjects to filter on)

Related Information

Sets Editor default options [page 417]

13.4.9 Building a set

You build a set to create its list members in the database. When a set is built, its lists of values are compiled into a database column immediately making the new count available to SAP BusinessObjects Web Intelligence

users, if this set was already published. The set container holding all sets in the repository is updated when you republish the set container. Depending on the type of set, there are two ways to build a set:

Build set using	Description
Build	Available for static and temporal sets. Button appears on the *Steps* and *History* tabs in the Sets Editor. When a temporal set is built for the current period, if *Allow multiple processing for current period* is not selected, and if you select *Build*, only data for the current period up to the current date is associated with the set. It is not possible to build the set again in the current period.
Temporal Management	Available for auto-fill temporal sets only. The button appears activated on the *Steps* and *History* tabs in the Sets Editor if you have selected the *Allow Auto-Fill* option in the *Calendar and date selection* dialog box when you created the temporal set. You can choose the following options: • Auto-fill (partial or complete) • Rollback (partial or complete) • Rollback and then Auto-fill *Auto-Fill* associates data for the specified calendar periods you choose.

Depending on the type of set, you can build using one of the following:

- For a static set, click the *Build* button available on the *Steps* or *History* tabs in the Sets Editor.
- For a temporal set, if you want data for the current period, click the *Build* button available on the *Steps* and *History* tabs in the Sets Editor. To build one period at a time (you might want to build periods in the future, for example Reservations), make sure you have not selected 'Stop at current period'.
- For a temporal set, if you want data for one or more specific periods in the past, click *Temporal Management* in the *Steps* and *History* tabs in the Sets Editor. All temporal Sets start building from their Base period onwards. The base period can only be changed if the set has no membership.

Related Information

Building temporal sets with auto-fill [page 378]

13.4.9.1 Defining a static set

Static set members are fixed at a moment in time. It is like a snapshot of your data that captures a state of the population that you are considering for analysis. Once you specify that your new set has static members, you can choose the set members either from a list of object members, or as a subset of combined sets that are already available in the set container. You can choose most universe objects to define the set.

1. Open or retrieve a set container.
2. If no folder is available for the set, create one: click the *Insert Folder* icon in the menu bar of the Set Browser pane and name the folder.
3. Select the set folder and click the arrow to the right of the *Insert Static Set* icon in the menu bar of the Set Browser pane.

4. Select a subject for the new set from the drop down list.

> **i Note**
>
> If you click the icon directly, a set is created from the currently selected subject by default.

A set file with a default name field appears under the folder node.

5. Type a different name for the set if required, and press `Enter` to validate the name.

The Set Properties pane is shown to the right of the Set Browser pane. You choose members for the set using the *Steps* tab as follows:

Choose your set members by	Description	Go here to choose your set members
Selecting the members from an object list.	You select a range of static members from a list of all of the business layer objects available via the new set's Subject .	Inserting a visual data step in a set [page 383]
Combining any of the existing sets in the container to create a new set.	You select the intersection, union, or exclusion between several built sets based on the same subject.	Creating a Sets on Sets Step [page 385]
Using the Query Panel to define complex filters.	You unleash the power of the SAP BusinessObjects Web Intelligence *Query Panel* to define complex filters.	Inserting a Query Panel step [page 387]

6. Select the dimension for your set:

Selecting a Dimension or Measure (Visual Data step only)

To Select a Dimension	To Select a Measure
1. Double-click an object from the Classes and Objects pane. 2. Expand the object and select each value that you want to include in the set. 3. Select a Context, if required, and click *OK*. 4. Select the Step Action (Start, for a new item, Add, Subtract, or Keep, depending on what relationship you want the list to have with the preceding step). For the first step you create, you Start is automatically selected, no other option is available. 5. Click *Finish*.	1. Double-click a Measure from the Classes and Objects pane. 2. Define a Yes/No condition for the measure (for example 'Price less than 300'), and click *Create*. 3. Expand the Measure and select a Context, then click *OK*. 4. Select *No* to reject the filter results from your set, or select *Yes* to keep the filter results. Then click *OK*. 5. Select the Step Action (Start, for a new item, Add, Subtract, or Keep, depending on what relationship you want the list to have with the preceding step). For the first step you create, you Start is automatically selected, no other option is available. 6. Click *Finish*.

13.4.9.1.1 Defining a static set based on a measure (Visual Data step)

In the Visual Data step editor, when you define a set based on a measure, you must define a Yes/No condition on the measure (for example, is the value of the object less than or equal to a defined value).

1. Open or retrieve a set container.
2. If no folder is available for the set, create one: click the *Insert Folder* icon in the menu bar of the Set Browser pane and name the folder.
3. Select the set folder and click the arrow to the right of the *Insert Static Set* icon in the menu bar of the Set Browser pane.
4. Select a subject for the new set from the drop down list.

   > **i Note**
   >
   > If you click the icon directly, a set is created from the currently selected subject by default.

   A set file with a default name field appears under the folder node.
5. Type a different name for the set, if required, and press `Enter` to validate the name.

   The Set Properties pane is shown to the right of the Set Browser pane.
6. In the *Steps* tab, select a Measure and define the Yes/No condition for the measure, for example, is the measure greater than or equal to 50. You can choose to include null values in the set.

13.4.9.2 Temporal sets

Description of the terms and options used in temporal sets.

Temporal sets are based on calendar periods. You often need to create sets that contain information about set members who join the set, leave it, stay in it, or simply visit the set, during specified calendar periods. Calendars are created in set container and available to sets. When you create a set based on a calendar, you also choose a DateTime dimension, for example `Start date`, in the business layer to link to the calendar. This date can then be compared to a period of the calendar when building the set.

> **i Note**
>
> Temporal sets now support prompts for reports. When the user runs a report, you can prompt the user to select the period for which the report will be generated.

The following terms are used in the temporal sets options:

Temporal Set Concepts

Concept	Description
Limited Temporal Sets	When sets are built in the limited temporal mode, by default, only the last two sets are built. The user can define a moving window of periods to be stored in the database. Any leavers outside of this window are deleted from the database to increase storage space.
Rollback	Go back to a specified period (the base period for a complete rollback, or a specified period for a partial rollback) and empty the periods built after this period. A complete rollback is equivalent to a purge operation.
Auto-fill	Auto-fill means that the sets are built and filled with data going forwards from the base period. This can be used to analyze booking reservations, for example. This can also be used after a rollback to rebuild sets for past periods.
Base period	This is the first period to be built.
Active period	This is the last period built.
Current period	The current period is the calendar period corresponding to the present day.
Future period	A future period is any calendar period after the current period
Past period	A past period is any calendar period prior to the current period

13.4.9.2.1 Defining a temporal set

Temporal sets are based on calendar periods. You often need to create sets that contain information about set members who join the set, leave it, stay in it, or simply visit the set, during specified calendar periods. Calendars are created in set container and available to sets. When you create a set based on a calendar, you also choose a DateTime dimension, for example `Start date`, in the business layer to link to the calendar. This date can then be compared to a period of the calendar when building the set.

1. Open or retrieve a set container.
2. If no folder is available for the set, click the *Insert a Folder* icon in the menu bar of the *Sets* pane and name the folder.
3. Select the set folder and click the arrow to the right of the *Insert Temporal Set* icon in the menu bar of the *Sets* pane.
4. Select a subject for the new set from the drop down list.

A calendar and date selection box appears. It lists the calendars defined in the set container and the date dimensions available in the business layer that can be used to associate the calendar with a date reference in your datamart.

> **i Note**
>
> If you click the icon directly, a set is created from the currently selected subject by default.

5. Click a calendar in the list of set calendars available.
6. In the lower pane, browse to and select a date dimension that links your data to the calendar. Refer to Related Links for more details on the options available on this dialog box.
7. If you want to build the sets from the present period onwards, check *Allow Auto-Fill*. Refer to Related Links for more details on the *Auto-Fill* option.
8. Click *OK*.

The set properties pane is shown to the right of the *Sets* pane. It contains a series of tabs.You choose members for the set using the *Steps* pane as follows:

Choose your set members by	Description	Go here to choose your set members
Visually selecting the members.	You select a range of static members from a list of all of the objects available via the set's Subject.	Inserting a visual data step in a set [page 383]
Combining any of the existing sets in the container to create a new set.	You select the intersection, union, or exclusion between several existing sets for the same subject.	Creating a Sets on Sets Step [page 385]
Select the members using the query panel step.	Use the full power of the query panel to create complex queries.	Inserting a Query Panel step [page 387]

See Related links for explanations for the other properties that you can view or set for a set.

13.4.9.2.1.1 Selecting calendar options for a temporal set

A calendar has been added to the Set Container. This is usually done by the administrator when creating the set container in the Repository.

The following information applies to the *Calendar and date selection* dialog box. This box appears when you click the *Insert Temporal Set* button from the *Sets Browser* pane to the left of the *Sets Editor*. Selecting calendar options here is the first step when designing a temporal set. These determine how calendar periods are mapped to your data. You have the following settings:

Calendar options	Description
Pick a set calendar	Calendars defined in the set container are listed.

Calendar options	Description
Pick a date to link set periods	Date/Time dimensions available to the subjects.
Allow Auto-Fill	When selected, the *Auto-Fill* button is enabled with *Build* on the *Steps* and *Time* tabs. If you want to build periods in the past, select the *Auto-Fill* checkbox. If you want to build the present period on a regular basis, do not check *Auto-Fill*.

1. Select a calendar in the *Pick a Set Calendar* list.
2. Expand the subject node and select the Date/Time dimension that provides the data for the calendar.
3. Select *Allow Auto-Fill* if you want to have the *Auto-Fill* option available when you build the set.
4. Click *OK* to continue defining the temporal set.

Related Information

Building temporal sets with auto-fill [page 378]
Defining a temporal set [page 376]
Designing and managing sets using the sets editor [page 367]

13.4.9.2.1.2 Building temporal sets with auto-fill

You can build a temporal set for specified periods rather than the current period. Selecting period options in the *Auto-Fill* dialog box allows you to specify the number of calendar periods that are taken into account when the set is built, starting from the last build date of the set up to the current date period, or another period that you indicate.

Using *Auto-Fill* to build your set ensures that the active calendar period for a set is incremented each time it is built. If you do not build with *Auto-Fill*, then only data from the current calendar period is taken into account.

> **i Note**
>
> The *Auto-Fill* button is enabled if you have selected the *Allow Auto-Fill* option in the *Calendar and date selection* dialog box when you created the temporal set. You can also change the Autofill nature of a temporal set after its creation, if the set has no membership. This can be achieved by purging the set.

> **i Note**
>
> The engine only fully builds the previous period of a non-autofill temporal set.
>
> By design, Autofill temporal sets have all periods in the past fully built, without gaps.

You have the following options:

Auto-Fill Type	Description
Complete	Build in a row all possible periods, from the first non-built period in the past, to the last available period. You cannot build in the future.
Partial	The build is from a selected base period to a selected end period. • You can select an end period date from the *Auto-Fill to* calendar. The number of affected periods is automatically incremented in the *Periods to auto-fill* list. • You can directly select the number of periods to automatically fill data by clicking the increase or decrease arrows in the *Periods to auto-fill* list.

1. Click the*Temporal Management* button on the *Steps* or *History* tabs.

   The *Auto-Fill* dialog box appears.
2. Do one of the following:
   - Click *Complete* to automatically fill data for all periods since the last build.
   - Click *Partial* then click the calendar button at the end of the text box to choose an end date period. Data will be inserted for all the periods up to that date. Or in the *Periods to auto-fill* box, click the increase or decrease buttons to select the number of periods that you want build.
3. Click *Auto-Fill* to build the set using the specified Auto-Fill options.

Related Information

Selecting calendar options for a temporal set [page 377]
Building a set [page 372]

13.4.9.2.1.3 Managing the build history for a temporal set

The Manage Build History dialog allows you to perform the following actions:

Temporal Set Management

Setting	Description
Rollback	• When complete rollback is selected, sets are rolled back to the base period from the current period. All sets are purged. • When partial rollback is selected, all built periods are eligible for rollback. Choose the period to rollback to. The sets for the chosen periods are purged. **i Note** The Auto-fill option is not available.

Setting	Description
Auto-Fill	• When complete auto-fill is selected, sets are built from the base period to the current period. • When partial auto-fill is selected, sets are built from the selected base period for the selected number of periods. **i Note** The Rollback option is not available.
Rollback and Auto-Fill	You can use this to rebuild partial set history or synchronize sets. Rollback, then auto-fill. You have the following options: • Complete rollback then complete auto-fill. • Complete rollback to the base period, then partial auto-fill. The selected number of periods are built from the base period. • Partial rollback, then partial auto-fill. Rollback and purge to a selected base period, then auto-fill subsequent periods to a selected period.

13.4.9.2.1.4 Managing the time settings

In the *Time* tab, you define the temporal set type (complete or limited) and the build options for the set.

In the upper part of the pane, you can choose between two set types:

- *Temporal* sets, where the set contains the full membership history.
- *Limited Temporal*, where the last one or more periods are built Two is the default value, the maximum number of periods is currently 9999. Leavers older than the numbers of periods will be dropped automatically during set refresh. This is useful to reduce volume of set history data.

Information about the calendar assigned to the set folder is displayed.

In the lower part of the pane, you can set the following build options:

- *Allow editing of steps after the set is built* This is useful to lock the set definition and enable consistent comparison over given set periods
- *Allow Auto-Fill*
  - *Stop at current period*
  - *Allow multiple processing in current period* Building during the current period will create incomplete sets, by definition. This can be useful to get the latest memberships.
  - *Base Period* (You can set a specific base period).

> **i Note**
>
> When auto-fill is not allowed, the *Temporal Management* option in the *Steps* and *History* tabs is disabled.

> **i Note**
>
> When *Allow multiple processing in current period* is not selected, once you have built a set for the current period, you cannot repeat the operation for that period. This means if you build a set on the first day of a period, you might have very little data for the set, and you will not be able to build that set again when more data is available.

13.4.9.2.1.5 Build options for the base period

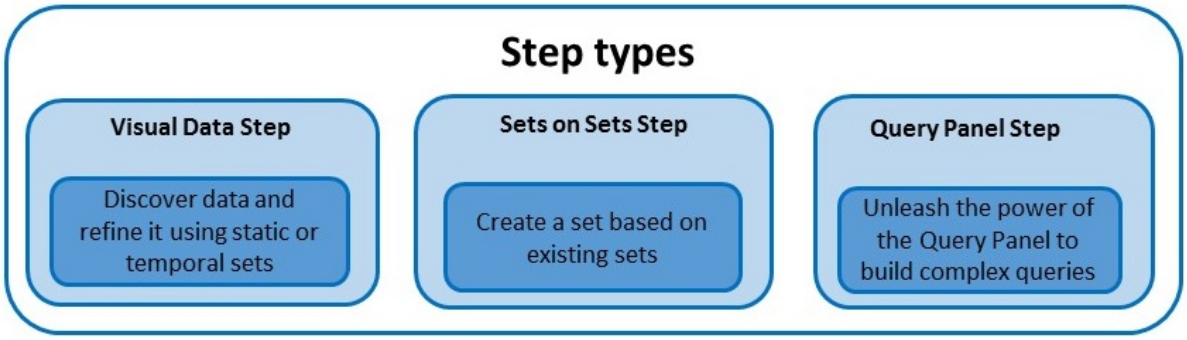
Selecting the base period for a temporal set.

This feature is useful to start a history at a later period than the calendar start. It cannot be changed after first period is built. You must rollback the whole history to enable the option again. Select the first period (the base period) for your temporal set.

The set can be built from this period onwards, or rolled back to this period.

13.4.9.3 Set steps

A set is made up of different steps. There are three types of step that you can insert, depending on the complexity of the filter you want to build:



Related Information

About inserting visual data steps in a set [page 382]
About inserting a Query Panel step in a set [page 386]
About inserting a sets on sets step [page 385]

13.4.9.3.1 About inserting visual data steps in a set

You use the Insert Visual Data step editor to select set members from a list of objects available via the subject. You double-click members or drag and drop them into the set editor pane.

The member objects you can select are attributes or measures. Although you can create a set with a single list of members, the value of building a set is to combine a sequence of lists that respond to questions allowing you to refine your data set.

Adding Custom Values

In cases where values might be missing or not yet available, you can create and insert custom values. For example, if your company will be selling to a region or country for the first time at a later date, you can insert the country and region names as custom values. Custom values are added in the *Visual Data* pane. When the actual values are available, you can delete the custom values by clicking the *Manage Custom Values* in the *Visual Data* pane.

Managing Conditions on Measures

When you select a measure as a step, you must choose a condition for the measure. Members are constrained to a Yes/No condition (for example: Price is Greater Than 200). Setting the condition is mandatory, you can select to keep the data met by the condition, or reject it from your set. You can also add Yes/No conditions in the *Visual Data* pane on objets other than measures, but this is not always automatic nor mandatory..

Adding, Subtracting, or Keeping Steps

When you insert a step in a set, the first step is automatically attributed 'Start' for the set, subsequent steps can be used to add, subtract, or keep data.

Each additional list of members (step) can be combined with the previous one, using the following properties:

Set step property	Description
Start	This is the default step. It is the first list of members that you create, and starts the sequence of lists, each one acting on the previous one to refine your set members.
Add	The list members are added to the set.
Subtract	The list members are subtracted from the set.
Keep	Performs an intersection between the set and the selected members. Removes those members from the set that do not exist in your selected members.

> **i Note**
>
> A Subject with non-unique values is not supported. In the set editor, when a set based on multi-key subject and has a Visual Data step, if the subject values are not unique, the build fails.

Related Information

Parameters and contexts [page 384]
Inserting a visual data step in a set [page 383]
Creating custom values for a visual data step [page 384]
Set step sampling wizard [page 388]
Building a set [page 372]
Hiding a set [page 390]
Publishing a set container [page 391]

13.4.9.3.1.1 Inserting a visual data step in a set

You use the Visual Data editor to select set members from a list of objects available via the subject. The member objects you can select are dimensions, attributes or measures. Although you can create a set with a single list of members, the value of building a set is to combine a sequence of lists that respond to questions allowing you to refine your data set.

1. Create and name a new set.
2. In the *Steps* tab in the *Set Properties* pane, click the *Insert Visual Data* button.
3. Do one of the following:
   - Double click an object from the *Classes and objects* pane.
   - Drag an object to the *Visual Data Selected Objects* pane.
4. Expand the object and select each value that you want to include in the set.When there are query contexts, the *Query Contexts* pane displays and you select the appropriate context and click *OK*. When there are parameters, the query parameter dialog displays and you edit the query and select or declare the values for the parameter.
5. Select the attribute objects from the list of values.
6. The *Step Action* drop down list in the bottom part of the dialog is set to *Start* by default if this is the first step of the set. If you are adding a list or modifying one, then you can choose *Add*, *Subtract*, or *Keep* depending on what relationship you want the list to have with the preceding step in the *Steps* tab.
7. Add a description for your step. This will help you when managing the different steps in the set you have created.
8. Click *Finish*.

   The list of members is added to the *Steps* tab. If it is the first list defined for the set, then it appears as number *1* with the qualification *Start*.
9. Click the *Insert Visual Data* button to continue adding new lists of members based on the sequence of questions that you are following for the data refinement.

> **i Note**
>
> When you create a new *Visual Data* step and add your first object, the tree of available objects displayed to the left is restricted to those objects that are compatible with both this first object and with the subsequent objects. Only objects that can be reached by a context-free path in the data foundation are eligible.

> **i Note**
>
> When you insert a step in the set, and click on the step, a pane displays at the bottom of the Set editor showing the step details (*Visual Data Details*, *Query Panel Step Details*, or *Sets on Sets Details*).

10. To change position of a step in the set, click the up and down arrows at the top of the *Steps* tab. The step actions are automatically updated.

    The set and its steps are saved in the Local Project. A new or modified set that has not been published is indicated by a red asterisk.

13.4.9.3.1.2 Creating custom values for a visual data step

In cases where values might be missing or not yet available, you can create and insert custom values. For example, if your company will be selling to a region or country for the first time at a later date, you can insert the country and region names as custom values beforehand. These values will appear in membership as soon as they become available in the database, and the set is rebuilt.

1. In the visual data pane of a set step, click on the manage custom values icon.

   The *Visual Data Customization* pane displays with an empty table.
2. Click *Add* and enter the new value (the value type is automatically detected).
3. Press `Enter`, or click *Apply*.

   The new value is displayed in the table of custom values in the *Visual Data Customization* pane.
4. You can add, edit, remove, and change the position of the custom values.
5. Click *OK* when you have finished entering custom values.

   The custom values are displayed at the top of the values list in the *Visual Data* pane. When an object with the same name exists in the database, there is an information note to indicate the value already exists. Click the *Manage Custom Values...* icon and remove the item from the list. When you close the *Visual Data Customization* pane, the item is removed from the *Visual Data* pane.

13.4.9.3.1.3 Parameters and contexts

Parameters and contexts may require setting for Visual Data steps or Sets on Sets steps. When you select a member where parameters require setting, you should select the appropriate context for the step, and edit the parameter as required. Parameters are merged in a set. This means that if you select a country for a parameter in a step, if a subsequent step has a parameter 'Select Region', only the regions from the country selected in

the precedent parameter are displayed as options. When you click the Reset Structure button in the toolbar, all values are cleared.

13.4.9.3.2 About inserting a sets on sets step

You can create sets based on existing sets by using the Sets on Sets Step editor .By creating a Sets on Sets step, you create sets, unions, or intersections from existing sets by using the dedicated Venn Diagram editor.

The Sets on Sets editor automatically detects compatible sets (coming from a compatible subject in a different table). These compatible sets are displayed with the name of their subject in brackets.

You add sets to collections and can define the behavior of each collection. To select an intersection, click on the appropriate area of the Venn Diagram. To select a whole collection, right-click on the collection's non-intersected area in the Venn diagram, and choose *Select whole collection* in the popup menu that displays.

- Empty collections are hidden from diagram.
- The Collection's icon differs for a Union or Intersection.
- When you select an area of the Venn Diagram, the selection is displayed in high contrast surrounded by a dotted line.

You cannot select the following:

- Sets causing circular dependencies
- Sets that are not compatible

Related Information

Parameters and contexts [page 384]
Creating a Sets on Sets Step [page 385]
Set step sampling wizard [page 388]
Building a set [page 372]
Hiding a set [page 390]
Publishing a set container [page 391]
Sets Editor default options [page 417]

13.4.9.3.2.1 Creating a Sets on Sets Step

You can select new set members by adding multiple sets for a subject into a collection, which applies an intersect or union operation on all of its sets. You can then add sets to more collections, combining them all using intersect or union operations to define a new set with members resulting from the complex combination of values.

You assign one of the following properties to your selection of members that links it to the previous step in a set, if any:

Set step property	Description
Start	This is the default step. It is the first list of members that you create, and starts the sequence of lists, each one acting on the previous one to refine your set members.
Add	The list members are added to the set.
Subtract	The list members are subtracted from the set.
Keep	Performs an intersection between the set and the selected members. Removes those members from the set that do not exist in your selected members.

1. In the *Step* tab of the set editor view, click the *Insert Sets on Sets* icon.

   The *Sets on Sets Step* editor displays.
2. To add the first set, double-click on a set in the sets tree view.

   The set is added to the first collection, and displayed in the Venn Diagram pane. To add the set to a different collection, right-click the set and select the collection in the contextual menu, or drag and drop it into the appropriate collection. To change the shape of the Venn Diagram, go to `Windows/Preferences.../Information Design Tool/Sets Editor` and select the shape you want.
3. Double-click on the next set to add the set to the Venn Diagram and to the second collection, or right-click the set and select a different collection for the set.
4. When you have completed your Venn Diagram, click on the intersections containing the membership you are interested in.
5. Write a description for the step.
6. Click Finish.

   The Sets on Sets step is added to your set, and the review pane showing the Venn Diagram is visible at the bottom of the Set Editor.

Related Information

Set step sampling wizard [page 388]
Sets Editor default options [page 417]

13.4.9.3.3 About inserting a Query Panel step in a set

When you create sets using the information design tool, you can insert steps in sets using the *Query Panel*.

You can insert set objects and sets in*Query Panel* steps. This allows you to use the full query power of the *Query Panel* to create more sophisticated set filters.

Related Information

Inserting a Query Panel step [page 387]
Set step sampling wizard [page 388]
Building a set [page 372]
Hiding a set [page 390]
Publishing a set container [page 391]

13.4.9.3.3.1 Inserting a Query Panel step

You use the Query Panel editor to select set members via a list of objects available to the subject and to create powerful set filters for your queries. The objects you can select are all the ones accessible in the universe, including regular filters. You can also select compatible sets.

> **i Note**
>
> You can now enter custom SQL in the Query Panel by using the Expression editor. The query result objects must specify an alias that is associated with sets subject keys. for example "as id".

> **i Note**
>
> When you include a set in the Query Panel step, when you run the query on a .unx universe, the set displays in the Query Panel outline.

1. Create and name a new set.
2. In the *Steps* tab in the *Set Properties* pane, click the *Insert Query Panel* button.

   The Query Panel displays and shows the dimension behind the set's subject (defined when you created the set) in the Result Objects pane. This object cannot be removed, nor the other ones be added in the Result Objects pane.
3. Expand the objects in the Available Objects pane and drag and drop one or more objects to the filter view and define the filter conditions. You can also use sets in the filter conditions, they are displayed on the left: drag and drop a set into the filter pane, as required. You can use existing set and build advanced condition. For instance, you can exclude members from MyPreviousCampaign. When there are query contexts, the *Query Contexts* pane displays and you select the appropriate context and click *OK*. When there are parameters, the Query Prompt dialog displays and you edit the query and select or declare the values for the parameter.
4. Enhance the filter by adding more objects, https://help.sap.com/viewer/3d4f417fd0764f909c0ef7931e19fe1a/4.2.6/en-US/46739fd66e041014910aba7db0e91070.html.
5. The *Step Action* button in the bottom part of the dialog is set to *Start* by default if this is the first step of the set. If you are adding a step or modifying one, then you can choose *Add*, *Subtract*, or *Keep* depending on what relationship you want its membership to have with the preceding step in the *Steps* tab.
6. Add a description for your step. This will help you when managing the different steps in the set you have created.

   If there are any contexts or parameters required for your step, the Contexts pane displays, then the parameters pane (if relevant).

7. Click *Finish*.

   The step is added to the *Steps* tab. If it is the first step defined for the set, then it appears as number *1* with the qualification *Start*. When you add a step to existing steps, it is added in the last position. You can move the steps up or down the list of steps.
8. Continue adding new steps to your set.

   > **i Note**
   >
   > When you insert a step in the set, and click on the step, a pane displays at the bottom of the Set editor showing the step details (*Visual Data Details*, *Query Panel Step Details*, or *Sets on Sets Details*).
9. To change position of a step in the set, click the up and down arrows at the top of the *Steps* tab. The step actions are automatically updated.

   The set and its steps are saved in the Local Project. A new or modified set that has been published is indicated by a red asterisk.

Related Information

Set step sampling wizard [page 388]

13.4.9.3.4 Set step sampling wizard

Use the step sampling wizard to apply sampling to your set step in order to work on a meaningful subset of the total membership.

In order to work with more manageable volumes of data, you can use the sampling wizard to select a percentage of the available rows, or a given number of rows. The sampling feature works for all step types (Visual Data step, Sets on Sets step, Query Panel step). The panel gives you the folllowing information:

Sampling options and functions

Sampling sorting method	Description
None:	None when the sample size is 100%
Random (Percent)	When a percentage is selected, but no ordering has been specified.
Ranked (Percent)	When you have chosen one or more business items to order by.
Ranked, stratified (Percent)	When you have chosen one or more business items to order by and one or more business items to stratify by.

Sampling sorting method	Description
Random, stratified (Percent)	When you have not chosen one or more business items to order by and you have chosen one or more business items to stratify by.
Random (Members in total)	When a number of members is defined, but no ordering has been specified.
Ranked (Members in total)	When you have chosen one or more business items to order by.
Ranked, stratified (Members in total)	When you have chosen one or more business items to order by and a different business item to stratify by.
Random, stratified (Members in total)	When you have not chosen a business item to order by and you have chosen one or more business items to stratify by.

To order your sample, double-click in the *Order by* pane and select select one or more business items. You can preview the results in the preview pane. The order of the columns depends on the order of the business items in the *Order by* and *Stratify by* panes.

When you choose to stratify by, the order of the business items in the Stratify by pane determines the order of the columns in the sample preview pane.

When you have finished defining your sample, click *Finish*, The objects selected for sampling are available at the bottom of the Set edition pane.

Related Information

Set steps [page 381]

13.4.9.4 Defining groups of sets for processing

You can now define groups of sets automatically ordered by the system according to their inter-dependencies (if any). This will decide in which safe sequence they will be processed when they are built as a whole. Processing group allows to group a number of sets together so that they can be processed and reprocessed together and ensure that each parent set has finished processing before the child set starts.

In the Set tree pane, click the Group tile and create a new group.

Add sets to the group, then press Save.

You can schedule the group processing from the CMC. In the parameters section, select the group and select *Schedule*.

> **i Note**
>
> The CMC can only schedule the processing of one group at a time

13.4.9.5 Testing sets by running queries with the set filters

Once you have published your sets you can use the Query Panel to test them as follows:

Use the Query Panel in	Test the set as follows
Information design tool	• From the *Repository Resources* pane, browse to the universe used to create the set container. • Right-click the universe and select *Run Query*, or double-click the universe. • Run queries in the Query Panel using the set filter.
SAP BusinessObjects Web Intelligence	• Create a new document, and select the universe used for the set container as the data source. • Run queries using the set filter as you would any other query filter. Refer to the *SAP BusinessObjects Web Intelligence User's Guide* for information on using SAP BusinessObjects Web Intelligence.

Related Information

Using the Query Panel [page 307]

13.4.9.6 Hiding a set

You can hide a set from SAP BusinessObjects Web Intelligence users. In the *Consumption* tab of the *Set* editor, set the *Visibility* property to *Hidden*. You can build or publish the set container without the reporting tools users being able to see or use the set.

13.4.9.7 Scheduling set refresh dates

You can schedule a refresh time for your sets from the home page of the Central Management Console. You select *Sets* from the *Applications* drop down menu, select the set container, and then schedule a refresh date for a set listed in the *Parameters* page.

Refer to the *Business Intelligence Platform Administrator Guide* for complete information on scheduling refresh dates in the Central Management Console.

13.4.9.8 Viewing the SQL for a set

When a set is defined with set steps, you can view the SQL that will be generated to materialize this set to the database. You can right-click on a set and select *Show Materialization Script*. You can view the SQL for the set membership, but you cannot execute the script from this view.

13.4.9.9 Editing table names in a set

You can edit the table names in a set, but you must edit all table names in that container.

> **i Note**
>
> When you change table names, new tables are created, and the original tables are not deleted in the database. The new tables are associated with the defined *Owner* and *Qualifier*.

The table names are displayed in the *Database* tab of the *Sets Containers Management* dialog. Double-click a table name to start renaming the table. You can undo a name change for a table by right-clicking on the table name and selecting *Undo table renaming*. You can also select *Copy to clipboard* and the whole table structure is copied under text format to the clipboard.

> **i Note**
>
> If you change the *Owner* and *Qualifier* without closing the editor, the changes won't be reflected in the locally edited container. You must close and reopen the editor or click the refresh icon in the *Sets* pane.

13.5 Publishing a set container

You publish a set container to update the contents accessible to Set Designers in the repository. This ensures that the latest definitions of sets are available to SAP BusinessObjects Web Intelligence as set filters in the Query Panel.

1. Ensure that you saved your latest changes to the set container.
2. Right-click the set container in the *Local Projects* pane.
3. Select *Publish Changes* from the contextual menu.

   The sets container in the repository is updated with the latest set changes.

   > **i Note**
   >
   > Calendars and Subjects are unaffected, and can only be updated by Administrators.

14 Tools and Commodities

14.1 Finding and replacing text

You can search for text in the data foundation and business layer, and object names in he data foundation.

14.1.1 Finding and replacing text

You can perform a text-based search for object names in the active data foundation or the active business layer. Use this search to find an object or edit its properties. There are three main search areas:

- Names
- Descriptions
- Scripts, references tables, and objects

The search result is displayed in a table. You can edit one item at a time or select multiple items and replace the object names in one action. Changes made in the data foundation do not impact the object names in the business layer.

Related Information

Where you can search in the data foundation [page 392]
Finding and replacing object names in the data foundation and the business layer [page 394]
Where you can search in the business layer [page 393]

14.1.2 Where you can search in the data foundation

Describes where you can search for a name in the data foundation of the information design tool.

You can run a search or search and replace on the following items in the data foundation:

Finding object names in the data foundation

Search	The search will find results in
In the *Name* of:	• Tables • Derived tables • Alias tables • Calculated columns • Contexts • Parameters • Static lists of values • Lists of values based on SQL
In the *Description* of:	• Tables • Derived tables • Alias tables • Contexts • Parameters • Static lists of values • Lists of values based on SQL
In the Scripts, reference tables, and objects of:	• Table owners, qualifiers • SQL Statement elements: Derived tables, Calculated columns, Joins, Filters on columns • Filters on columns in tables • Parameter prompt text • Column names in a static list of values

Related Information

Finding and replacing object names in the data foundation and the business layer [page 394]

14.1.3 Where you can search in the business layer

Describes where you can search for a name in the business layer of the information design tool

You can run a search or search and replace on the following items in the data foundation:

Finding object names in the business layer

Search	The search will find results in
In the *Name* of:	• Parameters • Static lists of values • Lists of values based on SQL • Lists of values based on queries • Objects, • Navigation paths
In the *Description* of:	• Parameters • Static lists of values • Lists of values based on SQL • Lists of values based on queries • Objects, • Navigation paths
In the Scripts, reference tables, and objects of:	• Prompt text parameters • Column names in static lists of values • SQL statements in lists of values based on SQL • SELECT statement objects • WHERE statement objects • Extra table objects (Replace is not possible) • Object name in query based on a list of values based on a query

Related Information

14.1.4 Finding and replacing object names in the data foundation and the business layer

Use the Find/Replace function to search for specific string, then change this string in selected objects. Search results are displayed with the object names, their type, for example table or dimension, the object property where the string is found, the parent object and in what resources the object is located (in case of linked universe). You can replace text in selected objects that are included in the search result set.

1. Click the *Find/Replace* button in the menu bar at the top of the information design tool window.
2. Enter a text string to search and click *Find*.

   The objects where the string are found are listed. Double-click one object to put the focus on it in the data foundation. Click *Previous* or *Next* to move this focus on the previous or next found object in the data foundation.

3. If you want to replace either a whole string or part of a string, select the objects to be changed in the search result list, type the replacement text in the *Replace with (for selected objects)* box, and click *Replace*, then *Close*.

   The selected object names in the data foundation are updated with the replacement text. You can undo the change by pressing *Ctrl+Z*.

14.2 Showing dependencies between resources

14.2.1 Resource dependencies

For any local resource, you can view its relationship to other local resources, and any dependent universes published in a repository.

Making changes to a resource, such as deleting it from a local project, moving it to another local project, renaming or updating it, may impact other resources that depend on it. You are warned of the impact before you delete or move a resource.

To help you understand the impact of changes and plan your work, commands exist that show the dependencies between resources and their objects.

Dependencies between local resources

The *Show Local Dependencies* command shows the dependencies between resources in a local project.

When you select a resource, two tabs display the dependent and referenced resources: The *Dependent Resources* tab lists resources in the same local project that depend on the selected resource. The *Referenced Resources* tab lists the resources in the same local project that are referenced by the selected resource. See example 1.

The paths to referenced resources are relative, not absolute. That means that if you rename a resource, the referenced resources are assumed to be in the same folder. If the referenced resources are in a different sub-folder, the reference is broken. See example 2.

You can also show local dependencies for any object in the business layer. In this case, for relational business layers, the referenced resources include the data foundation tables and columns that the object is based on.

Example: 1: Show local dependencies

The local project `Demo` contains the following resources:

- **Demo_Local_Connection.cnx**
- **Demo_Data_Foundation.dfx**
- **Demo_for_Accounting.blx**
- **Demo_for_Sales.blx**

You want to list all the resources that would be impacted if you change the **Demo_Data_Foundation**. In the Local Projects View, you select the *Show Local Dependencies* command on **Demo_Data_Foundation.dfx**. The following dependencies are displayed:

Dependent Resources	*Referenced Resources*
\Demo\Demo_for_Accounting.blx **\Demo\Demo_for_Sales.blx**	**\Demo\Demo_Local_Connection.cnx**

The two business layers contain references to the data foundation and might contain invalid references if you delete or change **Demo_Data_Foundation**, so they are listed in the *Dependent Resources* tab.

The data foundation references the connection. Any changes to **Demo_Local_Connection** could impact **Demo_Data_Foundation** and its dependent resources.

Now you want to show the dependencies for **Demo_Local_Connection**:

Dependent Resources	*Referenced Resources*
\Demo\Demo_Data_Foundation.dfx • **\Demo\Demo_for_Accounting.blx** • **\Demo\Demo_for_Sales.blx**	

Notice that **Demo_Data_Foundation** and its two dependent business layers are listed as dependent resources. Since a connection is the first resource created when building a universe, no resources reference the connection.

Example: 2: Renaming resources

The local project **OLAP_Demo** contains a folder with the following resources:

Folder_One

- **OLAP_Local_Connection.cnx**
- **OLAP_Business_Layer.blx**

Rename **OLAP_Business_Layer.blx** to **OLAP_New_Business_Layer.blx** and show local dependencies. In the Local Projects View, you select the *Show Local Dependencies* command on **OLAP_New_Business_Layer.blx**. The following dependencies are displayed:

Dependent Resources	*Referenced Resources*
	\OLAP_Demo\Folder_One **\OLAP_Local_Connection.cnx**

Even though the business layer was renamed, the **OLAP_Local_Connection** is listed as a referenced resource because it is in the same folder.

Now, create **Folder_Two** in the **OLAP_Demo** project and copy **OLAP_New_Business_Layer**.blx to **Folder_Two**. In **Folder_Two**, rename **OLAP_New_Business_Layer**.blx to **OLAP_New2_Business_Layer.blx** and show local dependencies.

Dependent Resources	*Referenced Resources*
	\OLAP_Demo\Folder_Two \OLAP_Local_Connection.cnx

The reference to **OLAP_Local_Connection** is broken because when renaming to **OLAP_New2_Business_Layer.blx**, the information design tool assumes referenced resources are in the same folder.

Dependencies between data foundation and business layer objects

When editing a data foundation, you can show local dependencies for any table or column. A list of dependent business layers displays. You can then display a list of the objects per business layer that depend on the selected data foundation object.

When editing a relational business layer, in the data foundation view, you can select tables and columns and show associated objects. This will highlight in the business layer all objects which reference the selected data foundation objects.

Dependencies between local resources and repository resources

The *Show Repository Dependencies* command lists the universes published in a particular repository that are referenced by the selected local resource.

Related Information

Showing local dependencies [page 397]
Showing local dependencies in the data foundation [page 174]
Showing associated objects [page 295]
Showing repository dependencies [page 398]

14.2.2 Showing local dependencies

To show resources in the local project that depend on a selected resource:

1. In the Local Projects View, select the resource for which you want to show the dependencies.

2. Right-click and select *Show Local Dependencies*.

The *Dependent Resources* tab lists the resources in the same local project that contain references to, or depend on, the selected resource.

The *Referenced Resources* tab lists the resources in the same local project that are referenced by the selected resource.

Related Information

Resource dependencies [page 395]

14.2.3 Showing repository dependencies

To show universes in a repository that depend on a selected resource:

1. In the Local Projects View, select the resource for which you want to show the dependent resources published in a repository.
2. Right-click and select *Show Repository Dependencies*.
3. Select a session for the repository system where the resources are published, and log in.

The published universes in the repository that reference the selected resource are listed.

Related Information

Opening a session [page 80]
Resource dependencies [page 395]

14.3 Checking the integrity

Checking the integrity of a universe or its elements helps you avoid problems when running queries and reports on a published universe. This function allows you to verify aspects of the design of your universe and its elements. The check is performed in two steps: the SQL is first parsed, and then is executed.

You can check the integrity for different objects and resources at any time in the information design tool. This function is available for the following views:

- In the *Local Projects View* for resources such as data foundations, business layers, connections, and shortcuts.
- In the *Editor* for elements in the data foundation and business layer such as tables, contexts, business layer objects, queries, parameters, lists of values.

- In the *Repository Resources View* for published universes.
- In the *Repository Resources View* for published universes when checking the validity of security profiles.

1. Right-click the resource or object of which you wish to check the integrity, and select *Check Integrity*. You can expand the panel to full screen using the expand/collapse button.
2. Select the rules you want to apply in the left-hand pane of the *Check Integrity* dialog box.
3. Click *Check Integrity*.

   The results are listed in the right-hand pane of the *Check Integrity* dialog box. You can change the status of the results of a rule to a higher or lower setting depending on your own standards. You can do so in the information design tool preferences. When a rule has been checked, it can have one of three status:

Icon	Status	Description
	Error	The check has detected an error and can't go on. You must resolve the problem.
	Warning	The check has detected a missing object such as a key or a link. It may cause future issues.
	Information	The check hasn't detected any problem.

4. Save the results in a text file by clicking *Export*.

You can now review the results. Click *OK* to close the dialog box.

> → Tip
>
> You can perform the check automatically whenever you save a resource by setting a background integrity check. For more information, see the related topic about setting preferences for *Check Integrity*.

> → Tip
>
> You can still review the *Check Integrity* after closing the dialog box. It is available in the *Check Integrity Problems Views* until you run the next check integrity. For more information, see the related topic.

To change settings or the status of the messages returned by the check, see the Check integrity settings [page 405] page.

Related Information

Check integrity settings [page 405]
Reviewing check integrity problems [page 400]

14.4 Reviewing check integrity problems

1. From the information design tool main menu, select *Window* > *Check Integrity Problems*.
   The Check Integrity Problems View opens with a list of the results of the latest check integrity.

   > **i Note**
   >
   > If check integrity results are available for more than one resource, the *Problems* view displays the results of the resource that is currently active in the editor.

2. To correct a problem, double-click the result in the list.
   The editor opens for the object concerned in the result. For example, if the result concerns a problem with table Customer, the *Data Foundation Editor* opens with the table Customer highlighted.

The list of results remains in the Check Integrity Problems View until you close the view, or run another check integrity.

Related Information

Checking the integrity [page 398]
Check integrity settings [page 405]

14.5 Multilingual universes

The information design tool supports the creation of multilingual universes. This feature enables a multilingual solution using a single universe metadata model:

- The designer creates the universe in the source language in the information design tool.
- Translators translate the metadata in the data foundation and business layer using the translation management tool. For more information about translating metadata, see the related topic.
- Report designers can then build reports once from the same universe that can be displayed in several languages based on user preferences.

Three language parameters impact how labels, metadata, and data are displayed in the information design tool:

- The Product Language determines the language of the user interface of the information design tool. This parameter is set in the information design tool preferences.
- The Preferred Viewing Locale is the user's preferred language for viewing report and query objects in an application. This parameter is set in the information design tool preferences.
  A locale defines a language and a geographical area. Locale abbreviations consist of the language abbreviation followed by the country abbreviation, for example, fr_FR. A locale also defines the way data is sorted and how dates and numbers are formatted. Data is displayed in a fallback locale when viewing a translated document and no translation in the user's Preferred Viewing Locale is available. The fallback locale can be defined (in the translation management tool), or defaults to the dominant locale that is automatically defined for every locale.

- Connection Language: For data sources that support a language parameter, the language parameter is entered when you create or edit a connection. This determines the language of the data.

Metadata source language in the information design tool

The metadata of the data foundation (table and column names) is created in the language of the metadata in the data source. The metadata that you insert into the data foundation can be entered in any language.

For SAP BW connections, the data foundation can be generated automatically in the language specified in the connection language parameter.

The metadata of a relational business layer is created in the language of the data foundation metadata. For OLAP business layers, the metadata is created in the language of the connection language parameter. The metadata that you insert into the business layer can be entered in any language.

Once the business layer is generated, the metadata (when viewed in the Business Layer Editor) remains in that language, even if you change the connection language parameter.

When designing the universe, with the `@Variable` function, you can use the `PREFERRED_VIEWING_LOCALE` and the `DOMINANT_PREFERRED_VIEWING_LOCALE` variables to customize the universe in order to filter multilingual data and retrieve only data in the user's Preferred Viewing Locale at query time.

Multilingual display in the information design tool

The Preferred Viewing Locale (defined in the information design tool preferences) determines the language of the metadata and data in the Query Panel, as long as two conditions are met:

- Translations are available in the language (metadata)
- The connection supports the language parameter (data)

When displaying values from the data source in the Connection Editor, the metadata and data displays in the language of the data source according to the current value of the connection language parameter.

> **i Note**
>
> When you run a query on a translated universe, the language that the objects are displayed in depends on your connection:
>
> - If you run a query connected to the CMS (based on a shortcut connection), the localization is set on the server side (SAP BusinessObjects Web Intelligence) settings.
> - If you run a query using a local connection, the localization uses the information design tool preferences.

> **i Note**
>
> Newly added input parameters are not reflected in the translation management tool.

> **i Note**
>
> The translation of prompts based on the @prompt syntax is not supported for information design tool data foundation and business layers (.dfx and .blx).

Related Information

Translating universe metadata [page 402]
Creating universes [page 60]
Languages used by the information design tool [page 415]

14.5.1 Translating universe metadata

Before you begin, the universe to translate must be created in the source language, and published to a repository or local folder. When the universe is translated, you can also translate the variables that are present in that universe. This includes SAP HANA Universe based on SAP HANA views that contain SAP HANA variables.

This procedure presents how to translate the data foundation and business layer metadata using local files. You can also translate metadata in shared projects. To do so, access the metadata files in the shared project in the repository from the translation management tool. For best results when translating queries, SAP BusinessObjects Web Intelligence gives optimum result. For details on the procedures in the translation management tool, see the *Translation Management Tool User Guide*.

1. In the information design tool, create a local project if you do not already have one.

   When creating the project, note the file path to the directory where the project files are saved in the file system. The default root directory for all projects is workspace.
2. Retrieve the universe into the local project.

   The information design tool saves the .dfx and .blx files in the local project. These files correspond to the data foundation and business layer definitions. These are the files that are used as the source for the translations.

   > **i Note**
   >
   > For OLAP universes, only a .blx file is saved.
3. In the translation management tool, translate the data foundation metadata (for relational universes):
   a. Import the .dfx file from project folder in the local file system.
   b. Translate the metadata.

      When a data foundation based on a SAP HANA view is opened, a branch *Variables* is displayed under the *Prompts* branch. This branch contains both SAP HANA input parameters and SAP HANA variables to translate. You can expand an input control or variable to display the Question field. This field contains the label that has to be translated. At run time, these translations are displayed in the user's Preferred Viewing Locale. Not all variables or input parameters can be translated.

c. Export the translated content to the local file system.

For details on these workflows, see the *Translation Management Tool User Guide*.

4. Follow the same procedure as in previous step to translate the .blx file.
5. In the information design tool, to see the translations, you must have a secured connection shortcut and set the preferred viewing language in the CMS.
6. Re-publish the business layer so that translations are available to universe users.

   For universes published to a repository, you can open the Query Panel on the published universe by right-clicking the universe the Repository Resources View and selecting *Run Query*.

   When you run a query on a translated universe, the language that the objects are displayed in depends on your connection:

   - If you run a query connected to the CMS (based on a secured shortcut connection), the localization is set on the server side (SAP BusinessObjects Web Intelligence) settings.

Related Information

Creating a local project [page 65]
Retrieving a published universe from a repository [page 305]
Retrieving a published universe from the local file system [page 305]
Showing values in a data source [page 162]
Inserting and editing a query in the business layer [page 242]
Languages used by the information design tool [page 415]
Publishing a universe [page 302]
Running a query on a universe published in a repository [page 81]

15 Preferences

Preferences allow you to customize your product. This section covers the wide range of settings available.

To set preferences, go to the information design tool main menu, and select *Window* > *Preferences*.

Related Information

Languages used by the information design tool [page 415]
Low memory warning parameters [page 419]
Business Layer Editor preferences [page 404]
Publish/retrieve options for the business layer [page 416]
Check integrity settings [page 405]
Universe conversion data format [page 409]
Setting connection display preferences for the Data Foundation Editor [page 409]
Display preferences for the data foundation view [page 418]
Setting Summary tab preferences for data foundation views [page 410]
Preferences for inserting tables and joins [page 414]
Setting table and join detection options [page 412]
Setting default join states for contexts [page 413]
Setting performance-related options for the data foundation view [page 413]
Setting the upper case and lower case naming [page 414]
Linking to online tutorials [page 415]
Middleware driver for secured relational connections [page 416]
Sets Editor default options [page 417]
Preferences for the Show Values command [page 417]

15.1 Business Layer Editor preferences

How to set the Business Layer Editor preferences for the information design tool.

The Business Layer Editor preference page lets you change how object names are generated in relational business layers.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and select *Business Layer Editor*.
3. Select how object names are generated in the business layer:

The option applies to the automatic generation of object names when creating relational business layers and inserting data foundation tables into the business layer.

The option does not apply for business layers based on SAP ERP, SAP BW, and SAP HANA, which use a dedicated strategy for naming object during automatic generation.

Option	Description
Transform table and column names to user-friendly names	If selected, generates object names by changing non-letter characters to blanks and capitalizing the first letter of each word. For example, the column name **`region_id`** generates the dimension name **`Region Id`**.
Use table and column names as they are	If selected, generates object names using the table and column names in the data foundation. For example, the column name **`region_id`** generates the dimension name **`region_id`**.
Sort Business Layer or folders content	Select one of the following: ○ *Sort alphabetically*. Objects are sorted alphabetically. No distinction is made between the type of object. ○ *Sort alphabetically (Folders always on top)*. Sorts are performed in the following order: Folders, then Dimensions, then Measures, then Filters. ○ *Sort alphabetically (Folders always on bottom)*. Sorts are performed in the following order: Dimensions, then Measures, then Filters, then Folders. **i Note** When a new object is added, it is not automatically sorted.

4. To restore the default values for preferences on the current page, click *Restore Defaults*.
5. To save the changes and continue editing preferences, click *Apply*.
6. To save the changes and close the *Preferences* dialog box, click *OK*.

The new preferences take effect immediately.

15.2 Check integrity settings

How to set the integrity check preferences for the information design tool.

In the check integrity preference page, you can select rules to be run automatically whenever you save a resource. You can also change the severity of the messages returned by each rule.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and select *Check Integrity*.
3. To select rules to be run automatically when you save a resource:
   a. Select the *Enable background check integrity on save* option.
   b. Select the rules to include in the background check.

      The *Cost* column indicates the relative processing time required to execute the rule.
4. To change the severity of the messages returned by a rule, click in the *Severity* column for the rule. Select a severity from the list.

5. To restore the default values for check integrity preferences, click *Restore Defaults*.
6. To save the changes and continue editing preferences, click *Apply*.
7. To save the changes and close the *Preferences* dialog box, click *OK*.

The background check takes effect immediately.

Related Information

Checking the integrity [page 398]
Check integrity default settings [page 406]

15.2.1 Check integrity default settings

The following table shows the default settings for each verification that the Check Integrity function performs.

The default settings for the Check Integrity function are shown in the table below. They indicate the type of error that is generated. To change settings or the status of the messages returned by the check, see the Check integrity settings [page 405] page.

Check Integrity Default Settings

Category	Verification	Error	Warning	Information
Connections and Dependencies	*Check Connection*		X	
	Check Dependencies		X	
Tables	*Check Alias Table*			X
	Check Calculated Column	X		
	Check Derived Table		X	
	Check Table Primary Key			X
	Check Table Structure	X		
	Check Variables		X	
	Check for Isolated Table			X
Input Columns	*Check Input Column Resolution*			X

Category	Verification	Error	Warning	Information
	Check Parameter Compatibility			X
	Check Parameter Data Type			X
Joins	*Check Cardinality Same as Detected*	X		
	Check Context		X	
	Check Join	X		
	Check Loops		X	
	Check for Undefined Cardinality			X
Business Layer	*Check Business Filter*	X		
	Check Business Object Expression	X		
	Check Business Object Mapping	X		
	Check Business Object Name			
	Check Business Object OLAP Validity	X		
	Check Query		X	
	Check object execution (DB)	X		
Parameters and Lists of Values	*Check Lists of Values*		X	
	Check Parameter			X
Federation Layer	*Data Source Exists Rule*			X
	Federated Table Has Active Mappings Rule			X
	Federated Table Has Columns Rule			X

Category	Verification	Error	Warning	Information
	Input Columns Are Solved Rule			X
	Input Table Exists Rule			X
	Input Table Is In Join Rule			X
	Join Formula Correctness Rule			X
	Join Paths With No Cycles Rule			X
	Mapping Formula Correctness Rule			X
	Mapping Has Core Input Tables Rule			X
	Mapping Has Input Tables Rule			X
	Post-filter Formula Correctness Rule			X
	Pre-filter Formula Correctness Rule			X
Security	*Check Business Security Profile*		X	
	Check Data Security Profile		X	
Sets	*Incorrect Group Dependencies Validity*	X		
	Incorrect Group Validity	X		
	Set Aliases Placement Validity	X		
	Sets Structure Validity	X		
	Subjects Validity			X
	Temporal Sets Validity	X		

Related Information

Checking the integrity [page 398]
Reviewing check integrity problems [page 400]
Check integrity settings [page 405]

15.3 Universe conversion data format

How to set the universe conversion data format for the information design tool.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and select *Convert Universe*.
3. Choose to convert the data type for `Date` to `DateTime` or choose to convert the data type for `Date` to `Date` only.

15.4 Data Foundation Editor preferences

How to set the Data Foundation Editor preferences for the information design tool.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and select *Data Foundation Editor*.

Related Information

Setting connection display preferences for the Data Foundation Editor [page 409]
Setting Summary tab preferences for data foundation views [page 410]
Setting table and join detection options [page 412]
Setting default join states for contexts [page 413]
Setting performance-related options for the data foundation view [page 413]

15.4.1 Setting connection display preferences for the Data Foundation Editor

The *Data Foundation Editor* preference page lets you to change how connections are displayed in the *Data Foundation Editor*. For information on other preferences for the *Data Foundation Editor*, see the Related Topics.

1. From the information design tool main menu, select *Window* > *Preferences* >.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and select *Data Foundation Editor*.
3. Set or clear options for how connections are displayed in the data foundation:

Option	Description
Enable automatic expansion	If selected, and the connection provides default qualifiers and/or owners, the catalog in the *Connections* pane automatically expands the default qualifier/owner.
Show Qualifiers/ Owners	If selected, qualifiers and owners, if available, are displayed by default in the *Connections* pane.
Show only information models (SAP HANA connections)	If selected, in the _SYS_BIC owner of SAP HANA connections, only information models (such as Analytic Views and Calculation Views) are displayed by default in the *Connections* pane.

4. To restore the default values for preferences on the current page, click *Restore Defaults*.
5. To save the changes and continue editing preferences, click *Apply*.
6. To save the changes and close the *Preferences* dialog box, click *OK*.

The new preferences take effect immediately.

Related Information

Display preferences for the data foundation view [page 418]
Setting table and join detection options [page 412]
Setting performance-related options for the data foundation view [page 413]
Connections [page 142]

15.4.2 Setting Summary tab preferences for data foundation views

When you open a data foundation for editing, by default the Master View is displayed, which contains all available tables and joins. The more tables and especially joins in a data foundation, the longer it takes to display the Master view. For large data foundations, it is good practice to define additional views that work on a subset of tables and joins, so you can focus on specific portions of the schema.

You can use the data foundation *Summary* tab to get an overview of all available sub-views, as well as an indication of their loading weight. These color coded indices allow you to select a specific view to edit without having to load the Master View that may contain other views that require more time to display on screen. You can set the following view preferences for the *Summary* tab:

View preferences for the Summary tab	When selected:
Remember the last modified view on opening	If you modified a view the last time you worked with the tool, this view displays.
Hide Summary	When you open a data foundation, the summary is displayed, if you don't want to see the summary, choose this option.
Always display Summary (unless Master is empty)	Choose this option if you want to see the summary. If the *Master* view is empty, the summary won't be shown.
Only display the summary if the number of joins in the Master exceeds	Choose this option and enter a minimum number of joins. When the number of joins in the *Master* view exceeds this number, the summary automatically displays.
Never display Summary	The *Summary* tab is not available for the data foundation. The master view of the data foundation is loaded by default.
Always display Summary	The *Summary* tab is displayed when you select to edit a data foundation. You choose the view to edit or the master view. The views in the *Summary* tab show the following colors for tables, joins, and comments: • Green: less than 100 • Orange: between 100 and 300 • Red: between 300 and 1000 • Black: greater than 1000
Only display Summary when the number of joins exceeds	The *Summary* tab is displayed when the number of joins in a data foundation exceeds the specified limit. When loading a data foundation, it is the time required to parse the joins that has the most impact on the data foundation's availability time in the editor.
Select last edited view on editor start up	The last edited data foundation view is automatically loaded when the data foundation is opened for editing. **i Note** This option is set as the default option.

You set the *Summary* tab preference for views in the data foundation as follows:

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and then expand the *Data Foundation Editor* node.
3. Select *Appearance*.
4. In the *Views* pane, set or clear display options.
5. Click *OK*.

Close and reopen your view in order for the summary tab to be available.

15.4.3 Setting table and join detection options

Set whether or not to automatically detect table keys, row counts, joins, and cardinalities when inserting tables into the data foundation.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and then expand the *Data Foundation Editor* node.
3. Select *Detections*.
4. Set or clear detection options:

Option	Description
Detect keys	If selected, when inserting a table into the data foundation, automatically detects primary and foreign keys in the data source and sets the keys in the data foundation table.
Keep keys defined in data foundation if none detected in database	If selected, when you use the *Detect Keys* command, if no keys are detected for a table in the data source, the keys that were set manually in the data foundation table are kept.
Detect row counts	If selected, when inserting a table into the data foundation, the number of rows in the table is counted and stored in the data foundation.
Detect joins	If selected, when inserting tables into the data foundation, joins are detected automatically (using the selected method) and inserted into the data foundation.
Detect cardinalities	If selected, when inserting tables into the data foundation, the cardinality of joins is detected and set automatically. **i Note** Detect joins must also be selected. For a description of the method used to detect cardinality, see the related topic about cardinality.

5. To restore the default values for preferences on the current page, click *Restore Defaults*.
6. To save the changes and continue editing preferences, click *Apply*.
7. To save the changes and close the *Preferences* dialog box, click *OK*.

The new preferences take effect immediately.

Related Information

Table keys [page 157]
Table row counts [page 158]
Detecting joins [page 178]
Cardinality [page 179]
Setting default join states for contexts [page 413]

15.4.4 Setting default join states for contexts

Application preferences let you set the default join state for contexts when adding joins and contexts to the data foundation.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and then expand the *Data Foundation Editor* node.
3. Select *Detections*.
4. Select the state that new joins will have when they are added to existing contexts:

Option	Description
Neutral	This is the default value. Joins added to the data foundation are neutral in any existing contexts. Neutral joins are not explicitly included or excluded but may be used in a query path.
Excluded	Joins added to the data foundation will be excluded in any existing contexts.
Included	Joins added to the data foundation will be included in any existing contexts.

5. Optionally, select the check box to apply this rule when creating contexts.

   By default, when creating a context, all joins in the data foundation are neutral. If, for example, you set the default state for joins to *Excluded* in step 4, and if you select *Also apply this rule when creating contexts*, then when creating a context, all joins in the data foundation will be excluded from the context.

Related Information

Contexts [page 181]

15.4.5 Setting performance-related options for the data foundation view

The following options enhance the display in the data foundation view. In some cases the enhancements can cause sluggishness when dragging elements within the view. Options can be unselected in cases where this results in a better display-performance benefit.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and then expand the *Data Foundation Editor* node.
3. Select *Performance*.
4. Set or clear performance options:

Option	Description
Use transparency effects	If selected, when dragging a table in the data foundation view, a semi-transparent shadow of the table follows to show the trail.
Use line smoothing	If selected, displays smoothed lines for joins.
Use image enhancement when zooming	If selected, avoids using big pixels when zooming.
Use text line smoothing	If selected, lines in the text are smoothed.
Use fading transitions	This option is not currently used.

5. To restore the default values for preferences on the current page, click *Restore Defaults*.
6. To save the changes and continue editing preferences, click *Apply*.
7. To save the changes and close the *Preferences* dialog box, click *OK*.

The new preferences take effect immediately.

15.4.6 Setting the upper case and lower case naming

You can set the owner and qualifier naming convention for the *Data Foundation Editor* of the information design tool. You can define it if the *Upper Owner and Qualifier* and/or *Lower Owner and Qualifier* actions impact owners and qualifiers names.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and select *Data Foundation Editor*.
3. Select the *Set Upper and Lower Case Action* checkbox to update the owner and qualifier names when you select the appropriate actions.

15.4.7 Preferences for inserting tables and joins

How to define how new tables or joins are displayed when they are added to a data foundation.

When you add new tables to a data foundation, you can select how the tables are displayed and you can select how the joins are added.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog, expand the *Data Foundation Editor* option, then select *Insertions*.
3. In the *Tables* section, select one of the radio buttons to define tables display mode when they are inserted in the data foundation:
   - *Expanded* (this is the default setting)
   - *With joins only*
   - *Collapsed*
4. In the *Joins* section, select one of the radio buttons to define how joins state in contexts when they are inserted to the data foundation:

- *Included*
- *Excluded*
- *Neutral*

15.5 Languages used by the information design tool

How to set the languages used by the information design tool.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and select *Languages*.
3. To change the language of the user interface, select the language from the *Product Languages* list.
4. To change the *Preferred Viewing Locale*, select the language from the list.

   For information about the Preferred Viewing Locale and how it impacts the language display, see the related topic about multilingual universes.
5. To save the changes and continue editing preferences, click *Apply*.
6. To save the changes and close the *Preferences* dialog box, click *OK*.
7. Exit and restart the information design tool for the language change to take effect.

Related Information

Multilingual universes [page 400]

15.6 Linking to online tutorials

You can access online tutorials about the information design tool from the *Help* menu. The *Online Tutorials* preferences page allows you to update the URL address to the tutorials.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and select *Online Tutorials*.
3. Enter the new URL address in *Online Tutorial Address*.
4. To save the change and continue editing preferences, click *Apply*.
5. To save the change and close the *Preferences* dialog box, click *OK*.

The new address takes effect immediately.

Related Information

Help [page 22]

15.7 Publish/retrieve options for the business layer

How to set the publish/retrieve options for the business layer.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and select *Publish*.
3. *Propose Check Integrity before publishing*. Select how to customize the Integrity Check before publishing the business layer, or you can choose *Don't propose Check Integrity before publishing*.
4. Set the retrieve options as required: *Prefix the folder name with 'retrieval'* followed by the date (`retrieval-yyyy-mm-dd-hh-mm-ss`), or *Prefix folder with Universe name* (example: `efashion.unx-yyyy-mm-dd-hh-mm-ss`)

   > **i Note**
   >
   > *Prefix folder with Universe name* is the default setting.

5. To restore the default values for preferences on the current page, click *Restore Defaults*.
6. To save the changes and continue editing preferences, click *Apply*.
7. To save the changes and close the *Preferences* dialog box, click *OK*.

15.8 Middleware driver for secured relational connections

How to set the preference for the middleware driver you want to use for secured relational connections.

The secure connections middleware preference applies only if the *Download connection locally* right is granted in the Central Management Console for the connection.

When running queries on secured relational connections in the information design tool, you can choose to run queries on the server using the server middleware driver, or locally using the local middleware driver.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and select *Secured Connections*.
3. Select the middleware to use:

Option	Description
Server middleware	To use the middleware driver on the repository server.
Local middleware	To use the middleware driver on the local machine.

4. To restore the default value, click *Restore Defaults*.

5. To save the changes and continue editing preferences, click *Apply*.
6. To save the changes and close the *Preferences* dialog box, click *OK*.

Related Information

Secured connections [page 85]

15.9 Preferences for the Show Values command

For the show values commands in the Data Foundation Editor and Business Layer Editor, you can select how to display values.

1. From the information design tool *Window* > *Preferences* . main menu, select
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and select *Show Values*.
3. Select how you want the values to display.
4. mainTo save the change and continue editing preferences, click *Apply*.
5. To save the change and close the *Preferences* dialog box, click *OK*.

The new display option takes effect immediately.

Related Information

Showing values in a data source [page 162]

15.10 Sets Editor default options

How to set the default options for the information design tool Sets Editor.

The *Window* > *Preferences* > *Information Design Tool* > *Sets Editor* option ets you set the following options:

- Allow warnings when rebuilding/purging a set
- Manage numeric and date objects in Visual Data steps
- Choose to use a classic Venn Diagram for Sets on Sets steps

1. From the information design tool main menu, select *Window* > *Preferences* .
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and select *Sets Editor*.
3. Choose to warn the user of the possible consequences when rebuilding or purging a set.

4. Choose to automatically apply a Yes/No condition when adding numeric or date objects to a Visual Data step
5. Choose to use a classic Venn Diagram when creating a Sets on Sets step.
6. To restore the default values for preferences on the current page, click *Restore Defaults*.
7. To save the changes and continue editing preferences, click *Apply*.
8. To save the changes and close the *Preferences* dialog box, click *OK*.

15.11 Default font for scripts and SQL expression editors

How to set the default font for scripts editors and the SQL Expression Editor.

You can select the font and font size for the following text/script editors:

- Select and Where fields in the object description
- SQL Expression Editor

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, click the *Information Design Tool* node.
3. In the *Expression fields font* setting, select the font and font size for the expression editors, then click *Apply*.
4. To save the changes and close the *Preferences* dialog box, click *OK*.

15.12 Display preferences for the data foundation view

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog box, expand the *Information Design Tool* node and then expand the *Data Foundation Editor* node.
3. Select *Appearance*.
4. Set or clear display options:

   The options change the display of elements in the data foundation view.

Option	Description
Columns: *Show format*	
Columns: *Show data types*	If selected, an icon showing the data type of the column is displayed in front of the column name. For example, `AB` indicates a string data type, and `12` indicates numeric.
Columns: *Centered*	If selected, column names are centered in the table display. Otherwise, column names are left-justified.
Shadow border	If selected, tables display with a shadow border.

Option	Description
Show row counts	If selected, a row count displays for each table.
Postfix alias names with original table name	If selected, the original table name for an alias table is displayed in parentheses after the alias table name.
Prefix table names with owner and qualifier	If selected, the owner and qualifier names are displayed in front of the table name.
Join Lines	Select the type of join line from the list.
Show full table names in join expressions	This option is not currently used.
Auto scroll and zoom to selection	If selected, when you select an element in the view, the view automatically scrolls and zooms to better center the element in the viewing area.

5. To restore the default values for preferences on the current page, click *Restore Defaults*.
6. To save the changes and continue editing preferences, click *Apply*.
7. To save the changes and close the *Preferences* dialog box, click *OK*.

The new preferences take effect immediately.

15.13 Low memory warning parameters

In order to enhance performances, and to ensure that you don't encounter low memory issues, define the warning message triggering criteria.

Set the low memory parameters to determine the conditions before a low memory warning message displays. You can define the following settings in the *Window* > *Preferences* > *Information Design Tool* pane, in the *Low memory parameters* and *Warning message triggering criteria* sections:

Parameter	Description
Delay (seconds) between each free memory processing	You can set the interval between running the process that frees up unused memory. This interval is in seconds.
Free memory threshold (percent)	This is is the percentage below which the process considers the available memory to be 'low'.
Number of consecutive times below the free memory threshold	When the about memory is below the threshold for this number of successive measurement cycles, a warning message displays.

15.14 Preferences for inserting tables and joins

How to define how new tables or joins are displayed when they are added to a data foundation.

When you add new tables to a data foundation, you can select how the tables are displayed and you can select how the joins are added.

1. From the information design tool main menu, select *Window* > *Preferences*.
2. In the *Preferences* dialog, expand the *Data Foundation Editor* option, then select *Insertions*.
3. In the *Tables* section, select one of the radio buttons to define tables display mode when they are inserted in the data foundation:
   - *Expanded* (this is the default setting)
   - *With joins only*
   - *Collapsed*
4. In the *Joins* section, select one of the radio buttons to define how joins state in contexts when they are inserted to the data foundation:
   - *Included*
   - *Excluded*
   - *Neutral*

16 SQL and MDX reference

This chapter contains technical information about the SQL/MDX editor, and the different functions, parameters, and formulas that you use with the information design tool.

Related Information

The SQL/MDX expression editor [page 421]
SAP BusinessObjects SQL function reference for multisource-enabled universes [page 423]
@Functions [page 489]
SQL generation parameters [page 500]
Formula language for prompts [page 522]

16.1 The SQL/MDX expression editor

The SQL/MDX Expression Editor helps you write valid SQL/MDX expressions.

You can type SQL/MDX directly into the *Expression* box, or drag and drop table names, column names, business objects, functions, and parameters from the resource panes available in the editor. These panes are described in the following table. To display a resource pane, click the icon in the tool bar of the *Expression* pane.

You can display the SQL expression for @ SELECT and @WHERE formula directly in the expression pane: click *Show SQL*. The @SELECT and @WHERE objects are replaced by the SQL code. The expression cannot be edited in *Show SQL* mode, you must click again on the *Show SQL* button to toggle back to edit mode: the SQL for the @SELECT and @WHERE is replaced by the object equivalent and you can edit the expression.

Different icons are available depending on the type of expression you are editing.

Icon	Description
Tables	The list of tables and columns in the data foundation. To see a list of values for a column, click the icon next to column name.

Icon	Description
OLAP Metadata	For OLAP connections, the list of objects in the source cube. To change the display options, click . You can display names, keys, or both. To see a list of members for a level, click the icon next to the level name. Use the icon to search for a string in the object names. **i Note** A list of values for hierarchy attribute objects is not available in the OLAP Metadata pane. To see values for attributes, use the list in the Business Layer pane.
Functions	The list of functions that can be used in the expression. The functions are grouped by type: • *Operators*: Common database operators, for example *, IS NOT NULL. • *Database Functions*: The SQL functions that are valid for the databases in the connections. For multisource-enabled data foundations or business layers, see the related topic about SAP BusinessObjects SQL functions. • *System Variables*: The system variables for which you can retrieve the values assigned using the @Variable function. For more information, see the related topic about @Variable. **i Note** You can also reference User Attributes defined in the Central Management Server using @Variable. • *@Functions*: The @Functions that are valid for this expression. For more information, see the related topic about @Functions.
Business Layer	The list of objects in the business layer. To see a list of members for a level, click the icon next to the level name. Use the icon to search for a string in the object names. Use the toggle button to change how the object-related text is inserted into the expression: • When the button is unselected (default): The @Select function for the object is inserted, for example: `@Select(Account\Account Number)` • When the button is selected: The SQL or MDX expression for the object is inserted, for example: `[Account].[Account Number]`
Parameters	The list of parameters defined in the data foundation and business layer.
List of Values	The list of values defined in the data foundation and business layer.

Click the *Validate* icon in the tool bar of the *Expression* pane to check if the expression you have defined is valid SQL/MDX.

Date formats in SQL expressions

When entering a date value into an SQL expression, you need to use the format that is defined for each data source in the extended PRM file by the SQL generation parameter `USER_INPUT_DATE_FORMAT`.

For example, for ERP data sources, in the corresponding extended PRM file `jco.prm`, the parameter `USER_INPUT_DATE_FORMAT=DATE'yyyy-mm-dd'`. So, the SQL expression might look like this:

WHERE "table_name"."start_date"=DATE'2013-04-10'

For more information about SQL generation parameters and extended PRM files, see the related link.

Related Information

SAP BusinessObjects SQL function reference for multisource-enabled universes [page 423]
@Variable [page 498]
@Functions [page 489]
Tables [page 149]
SQL generation parameters set in the extended PRM [page 514]
Default font for scripts and SQL expression editors [page 418]

16.2 SAP BusinessObjects SQL function reference for multisource-enabled universes

The information design tool provides a set of database functions based on SQL-92. Use these functions when defining SQL expressions for objects in a multisource-enabled data foundation or business layer.

This reference describes the syntax to use. The data federation service translates the SQL to the appropriate syntax for the data source at query run time.

> **i Note**
>
> The SAP BusinessObjects syntax can be different from the syntax of the same function provided by database-specific SQL.

16.2.1 Aggregation functions

16.2.1.1 `Average (avg)`

Description

Returns the average of a set of values.

Syntax

decimal `avg(<set of values>)`

Input

Parameter	Description	Data Type
`<set of values>`	A set of values.	Numeric

Notes

You can use the SQL keyword `DISTINCT` in front of column names.

Examples

Calculates the average of the sums of two columns: `avg`(table.column1 + table.column2)

Calculates the average of the values in a column that contains numbers written as strings:
`avg`((`toInteger`(table.column1))

16.2.1.2 `Count`

Description

Counts the number of values in a set.

Syntax

integer `count(<set of values>)`

Input

Parameter	Description	Data Type
`<set of values >`	A set of values.	All data types (Numeric, String, Boolean, DateTime, Date).

Notes

You can use the SQL keyword `DISTINCT` in front of column names.

Examples

Counts the number of values in a column: `count`(table.column1)

16.2.1.3 `Maximum (max)`

Description

Returns the maximum value in a set.

Syntax

value `max(<set of values>)`

Input

Parameter	Description	Data Type
`<set of values>`	A set of values.	All data types (Numeric, String, DateTime, Date).

Notes

You can use the SQL keyword `DISTINCT` in front of column names.

Examples

Returns the maximum value of a column: `max`(table.column1)

16.2.1.4 `Minimum (min)`

Description

Returns the minimum value in a set.

Syntax

value `min(<set of values>)`

Input

Parameter	Description	Data Type
set of values	A set of values.	All data types (Numeric, String, DateTime, Date).

Notes

You can use the SQL keyword `DISTINCT` in front of column names.

Examples

Returns the minimum value of a column: `min`(table.column1)

16.2.1.5 `Sum`

Description

Returns the sum of a set of values.

Syntax

decimal `sum`(`<set of values>`)

Input

Parameter	Description	Data Type
`<set of values>`	A set of values.	Numeric

Notes

You can use the SQL keyword `DISTINCT` in front of column names.

Examples

Sums the values in a column: `sum`(table.column1)

16.2.2 Cast

Description

Converts a given value to a given data type.

Syntax

value cast(<expression>, AS <data type>)

Input

Parameter	Description	Data Type
<expression>	A numeric expression.	All data types (Numeric, String, Boolean, DateTime, Date).
<data type>	The data type to which to convert the value of <expression>.	A keyword that can have the following values: • NULL • VARCHAR • DOUBLE • DECIMAL • DATE • TIME • TIMESTAMP

16.2.3 If Else

Description

Returns a value based on a given condition:

- If <condition> is true, the function returns the value of <expression1>.
- If <condition> is false, the function returns the value of <expression2>.

Syntax

value `ifElse(<condition>, <expression1>, <expression2>)`

Input

Parameter	Description	Data Type
`<condition>`	A logical expression.	Boolean
`<expression1>`	The value to return if `<condition>` resolves to true.	All data types (Numeric, String, Boolean, DateTime, Date).
`<expression2>`	The value to return if `<condition>` resolves to false.	All data types (Numeric, String, Boolean, DateTime, Date).

16.2.4 `If Null (nvl)`

Description

Returns a value based on whether or not a value is null:

- If `<expression1>` is null, the function returns the value of `<expression2>`.
- If `<expression1>` is not null, the function returns the value of `<expression1>`.

Syntax

value `nvl(<value1>, <value2>)`

Input

Parameter	Description	Data Type
`<expression1>`	Returns the value of `<expression1>` if that values is not null.	All data types (Numeric, String, Boolean, DateTime, Date).

Parameter	Description	Data Type
`<expression2>`	The value to return if `<expression1>` is null.	All data types (Numeric, String, Boolean, DateTime, Date).

16.2.5 `Timestamp Diff`

Description

Returns an integer representing the number of intervals by which the first given timestamp is greater than the second given timestamp.

Syntax

integer `timestampDiff`(`<interval>`, `<timestamp1>`, `<timestamp2>`)

Input

Parameter	Description	Data Type
`<interval>`	An interval constant. This parameter can either be a string or integer constant as follows: • 'SQL_TSI_FRAC_SECOND' or 0 • 'SQL_TSI_SECOND' or 1 • 'SQL_TSI_MINUTE' or 2 • 'SQL_TSI_HOUR' or 3 • 'SQL_TSI_DAY' or 4 • 'SQL_TSI_WEEK' or 5 • 'SQL_TSI_MONTH' or 6 • 'SQL_TSI_QUARTER' or 7 • 'SQL_TSI_YEAR' or 8	String or integer
`<timestamp1>`	A date and time.	DateTime
`<timestamp2>`	A date and time.	DateTime

Notes

- The calculation can be impacted by daylight saving time in the locale for 'SQL_TSI_HOUR'.
- Large differences can cause an error.
- The first day of the week is Sunday.

16.2.6 Database functions

16.2.6.1 CharString

Database character and string functions

This section contains the character and string function descriptions.

16.2.6.1.1 ASCII Code (ascii)

Description

Returns an integer representing the ASCII code value of the leftmost character in the input string.

Syntax

integer ascii(<string>)

Input

Parameter	Description	Data Type
<string>	A string of characters.	String

Notes

Returns null if <string> is null.

16.2.6.1.2 Case

Description

Returns a value depending on which of the given conditions is met.

Syntax

value `CASE <input expression> WHEN <when expression> THEN <then result expression> ELSE <else result expression> END`

Input

Parameter	Description	Data Type
`<input expression>`	An expression that represents a value to be compared to `<when expression>`.	All types. **i Note** `<Input expression>` must have the same data type as `<when expression>`.
`<when expression>`	An expression that represents a value to be compared to `<input expression>`.	All types. **i Note** `<Input expression>` must have the same data type as `<when expression>`.
`<then result expression>`	An expression representing the value to return when `<input expression>` compared to `<when expression>` is true.	All types. **i Note** `<Then result expression>` must have the same data type as `<else result expression>`.

Parameter	Description	Data Type
`<else result expression>`	An expression representing the value to return when `<input expression>` compared to `<when expression>` is not true.	All types. **i Note** `<Then result expression>` must have the same data type as `<else result expression>`.

Notes

- The `case` function implements the standard simple SQL `CASE` statement.
- `WHEN <when expression> THEN <then result expression>` is repeated to provide multiple conditions.

Examples

- `CASE` (table1.column1)
  `WHEN` 'p1' `THEN` 'Product1'
  `WHEN` 'p2' `THEN` 'Product2'
  `WHEN` 'p3' `THEN` 'Product3'
  `ELSE` 'Out of stock'
  `END`
- `CASE` ProductName
  `WHEN` 'laptop' `THEN` 1
  `ELSE` 0
  `END`

16.2.6.1.3 `Catalog`

Description

Returns the default catalog of the connection.

Syntax

string `catalog()`

16.2.6.1.4 `Character (char)`

Description

Returns the character corresponding to the given ASCII code.

Syntax

string `char(<code>)`

Input

Parameter	Description	Data Type
`<code>`	An ASCII code from 0 to 255.	Integer

Notes

Returns null if `<code>` < 0 or > 255.

16.2.6.1.5 `Concat`

Description

Concatenates two strings.

Syntax

string `concat(<string1>, <string2>)`

Input

Parameter	Description	Data Type
`<string1>`	A string.	String
`<string2>`	A string.	String

Notes

Returns null if either `<string1>` or `<string2>` is null.

Examples

`concat`('AB', 'CD') = 'ABCD'

16.2.6.1.6 `Database`

Description

Returns the name of the database.

Syntax

string `database`()

16.2.6.1.7 `Day Name`

Description

Returns a string containing the day of the week of a given date.

Syntax

string `dayName(<date>)`

Input

Parameter	Description	Data Type
`<date>`	A date.	Date or DateTime

Notes

Returns the day name in English in uppercase letters. Possible values are as follows:

- SUNDAY
- MONDAY
- TUESDAY
- WEDNESDAY
- THURSDAY
- FRIDAY
- SATURDAY

16.2.6.1.8 `LPad`

Description

Pads a string on the left side with a second given string to a given length.

Syntax

string `lpad(<string1>, <string2>, <length>)`

Input

Parameter	Description	Data Type
`<string1>`	A string.	String
`<string2>`	A string to insert into `<string1>` on the left side.	String
`<length>`	The total length of the return string after padding.	Integer

Notes

- If `<length>` < the length of `<string1>`, returns `left(<string1>, <length>)`.
- Returns null if either `<string2>` is null or `<length>` <= 0.

16.2.6.1.9 `Left`

Description

Returns the given number of characters from the left of the given string.

Syntax

string `left(<string>, <number of characters>)`

Input

Parameter	Description	Data Type
`<string>`	A string.	String
`<number of characters>`	The number of leftmost characters to return.	Integer

Notes

Returns null if either `<string>` is null or `<number of characters>` is <= 0.

16.2.6.1.10 `Left Remove (ltrim)`

Description

Removes the first sequence of spaces and tabs from the left side of the given string.

Syntax

string `ltrim(<string>)`

Input

Parameter	Description	Data Type
`<string>`	A string.	String

Examples

`ltrim`(' ABCD') = 'ABCD'

`ltrim`(' AB CD ') = 'AB CD '

16.2.6.1.11 `Lowercase (lcase)`

Description

Converts a string to lowercase.

Syntax

string `lcase(<string>)`

Input

Parameter	Description	Data Type
`<string>`	A string.	String

Examples

`lcase`('ABCD') = 'abcd'

`lcase`('Cd123') = 'cd123'

16.2.6.1.12 `Month Name`

Description

Returns a string containing the month name of a given date.

Syntax

string `monthName`(`<date>`)

Input

Parameter	Description	Data Type
`<date>`	A date.	Date or DateTime

Notes

Returns the month name in English in uppercase letters. Possible values are as follows:

- JANUARY
- FEBRUARY
- MARCH
- APRIL

- MAY
- JUNE
- JULY
- AUGUST
- SEPTEMBER
- OCTOBER
- NOVEMBER
- DECEMBER

16.2.6.1.13 `Permute`

Description

Permutes a given string using two templates: the `<reference template>` and the `<new template>`.

First, each character (or block of characters) in the `<reference template>` is assigned to a character (or block of characters) in the given string (`<string1>`). The lengths of `<string1>` and `<reference template>` must be the same.

Next, `<new template>` is used to permute the characters that were assigned in the `<reference template>`.

For example, the character string '22/09/1999', which represents a date, can be converted to '1999-09-22' as follows.

The `<reference template>` is 'DD/MM/YYYY'. The letters are assigned according to their position and grouping. So, 'DD' is the first block of characters, and is assigned the value '22', the first two characters in `<string1>`. The slash character (/) is assigned to the third character in `<string1>`. The next block of characters 'MM' is assigned to '09', and so on.

The `<new template>` is 'YYYY-MM-DD'. The permutation is applied and the resulting string is '1999-09-22'.

Text can also be inserted into `<new template>`, provided that none of the characters are already used in the `<reference template>`. For example, if `<new template>` = 'MM/DD Year: YYYY', the resulting string is '09/22 Year: 1999'.

Syntax

string `permute`(`<string1>`, `<reference template>`, `<new template>`)

Input

Parameter	Description	Data Type
`<string1>`	A string.	String
`<reference template>`	A string representing the pattern of `<string1>`.	String
`<new template>`	A string providing the new pattern for the permutation of `<string1>`.	String

Notes

- To represent a block of characters in the templates, repeat the character in the pattern. For example, 'YYYY' in `<reference template>` matches four characters in `<string1>`.
- The length of `<string1>` must be equal to the length of `<reference template>`, otherwise the function returns an error.

Examples

Change the format of how a date is represented:

- `permute`('02/09/2003', 'DD/MM/YYYY', 'YYYY-MM-DD') = '2003-09-02'
- `permute`('02-09/200', 'DD/MM/YYYY', 'YYYY-MM-DD') = '2003-09-02'
- `permute`('02/09_2003', 'DD/MM/YYYY', 'DL :MM/DD An :YYYY') = 'DL :09/02 An :2003'

Extract a month and year from a string of characters representing one date:

- `permute`('2003-09-02', 'DDYY-MM-YY', 'MM/YY') = '09/03'

Compose a number from an internal code:

- `permute`('03/03/21-0123', 'bbbYY/MM/DD-NNNN', 'YYMMDDNNNN') = '0303210123'

Extract date information from internal code:

- `permute`('2003NL987M08J21', 'YYYYXXXXXXMMXDD', 'YYYY-MM-DD') = '2003-08-21'

16.2.6.1.14 `Replace`

Description

Replaces in a given string the occurrences of the pattern with a replacement string.

Syntax

string `replace(<string>, <pattern>, <replacement string>)`

Input

Parameter	Description	Data Type
`<string>`	A string.	String
`<pattern>`	The string of characters to search for and replace in `<string>`.	String
`<replacement string>`	The string to replace the `<pattern>` in `<string>`	String

Notes

- If `<pattern>` is null, returns `<string>`.
- Does not return null if `<replacement string>` is null.

Example

`replace`('rar', 'a', 'ada') = 'radar'

16.2.6.1.15 `Replace String Exp`

Description

Replaces in a given string all occurrences of the pattern with a replacement string following the syntax of a Java regular expression. For more information, refer to the Pattern documentation for Java regular expressions at `http://java.sun.com/j2se/1.4.2/docs/api/java/util/regex/Pattern.html`.

Syntax

string `replaceStringExp(<string>, <pattern>, <replacement string>)`

Input

Parameter	Description	Data Type
`<string>`	A string.	String
`<pattern>`	The string of characters to search for and replace in `<string>`.	String
`<replacement string>`	The string to replace the `<pattern>` in `<string>`	String

16.2.6.1.16 `Replicate (repeat)`

Description

Returns a string formed by repeating a given character string a given number of times.

Syntax

string `repeat(<string>, <number of replications>)`

Input

Parameter	Description	Data Type
`<string>`	A string.	String
`<number of replications>`	The number of times to repeat `<string>`.	Integer

Notes

Returns null if `<number of replications>` <= 0.

16.2.6.1.17 Rightpart (right)

Description

Returns the given number of characters from the right of the given string.

Syntax

string right(<string>, <number of characters>)

Input

Parameter	Description	Data Type
<string>	A string.	String
<number of characters>	The number of rightmost characters to return.	Integer

Notes

Returns null if either <string> is null or <number of characters> is <= 0.

16.2.6.1.18 Rpad

Description

Pads a string on the right side with a second given string to a given length.

Syntax

string rpad(<string1>, <string2>, <length>)

Input

Parameter	Description	Data Type
`<string1>`	A string.	String
`<string2>`	A string to insert into `<string1>`.	String
`<length>`	The total length of the return string after padding.	Integer

Notes

- If `<length>` < the length of `<string1>`, returns `right`(`<string1>`, `<length>`).
- Returns null if either `<string2>` is null or `<length>` <= 0.

16.2.6.1.19 `Rtrim`

Description

Removes the first sequence of spaces and tabs from the right side of the given string.

Syntax

string `rtrim`(`<string>`)

Input

Parameter	Description	Data Type
`<string>`	A string.	String

Examples

`rtrim`('ABCD ') = 'ABCD'

`rtrim`(' AB CD ') = ' AB CD'

16.2.6.1.20 `Schema`

Description

Returns the default schema (qualifier and owner) of the current connection.

Syntax

string `schema()`

16.2.6.1.21 `Space`

Description

Returns a character string with the given number of blank characters (spaces).

Syntax

string `space(<number of spaces>)`

Input

Parameter	Description	Data Type
`<number of spaces>`	The number of spaces to be returned in the string.	Integer

Notes

Returns null if `<number of spaces>` <=0.

16.2.6.1.22 `Str`

Description

Returns a value set to a specific length.

Syntax

String `str(<length>)`

16.2.6.1.23 `Stuff (insert)`

Description

Replaces the sequence of characters in one given string with a second given string.

Syntax

string `insert(<string1>, <start position>, <number of characters>, <string2>)`

Input

Parameter	Description	Data Type
`<string1>`	A string.	String
`<start position>`	A number representing the position in `<string1>` to start the replacement. Must be in the range of 1 to the length of `<string1>` + 1.	Integer
`<number of characters>`	The number of characters in `<string1>` to replace. Must be in the range of 0 to the length of `<string1>`.	Integer
`<string2>`	The replacement string.	String

Notes

Returns null if either `<start position>` or `<number of characters>` is out of range.

16.2.6.1.24 `Substring`

Description

Returns a substring of a given string.

Syntax

string `substring(<string>, <start position>, <number of characters>)`

Input

Parameter	Description	Data Type
`<string>`	A string.	String
`<start position>`	The start position in `<string>` of the substring. Must be in the range of 1 to the length of `<string>`.	Integer
`<number of characters>`	The number of characters to include in the substring.	Integer

Notes

Returns null in the following situations:

- `<start position>` <= 0
- `<start position>` > the length of `<string>`
- `<string>` is null
- `<number of characters>` <= 0

Examples

`substring`('ABCD', 2, 2) = 'BC'

`substring`('ABCD', 2, 10) = 'BCD'

`substring`('ABCD', 0, 2) = null

16.2.6.1.25 `To Null`

Description

Converts a given value to null.

Syntax

null `toNull`(`<expression>`)

Input

Parameter	Description	Data Type
`<expression>`	A value.	All data types (Numeric, String, Boolean, DateTime, Date).

16.2.6.1.26 `To String`

Description

Converts a given value to a string.

Syntax

string `toString`(`<expression>`)

Input

Parameter	Description	Data Type
`<expression>`	A value.	All data types (Numeric, String, Boolean, DateTime, Date).

Examples

`toString`(45) = '45'

`toString` (-45) = '-45'

`toString`(45.9) = '45.9'

`toString` (-45.9) = '-45.9'

`toString`(Date value for September 9, 2002) = '2002-09-09'

`toString`(DateTime value for September 9, 2002 23:08:08) = '2002-03-03 23:08:08'

`toString`(Boolean value 1) = 'true'

`toString`(Boolean value 0) = 'false'

16.2.6.1.27 `Trim`

Description

Removes the spaces and tabs from the left and right sides of the given string.

Syntax

string `trim`(`<string>`)

Input

Parameter	Description	Data Type
`<string>`	A string.	String

16.2.6.1.28 `Uppercase (ucase)`

Description

Converts a string to uppercase.

Syntax

string `ucase(<string>)`

Input

Parameter	Description	Data Type
`<string>`	A string.	String

Examples

`ucase`('abcd) = 'ABCD'

16.2.6.1.29 `User`

Description

Returns the user name as defined in the connection parameters.

Syntax

string `user()`

16.2.6.2 DateTime

16.2.6.2.1 `Convert`

Description

Converts a given value to a given data type.

Syntax

value `convert(<expression>, <data type>)`

Input

Parameter	Description	Data Type
`<expression>`	A value or expression.	All data types (Numeric, String, Boolean, DateTime, Date).
`<data type>`	The data type to which to convert the value.	A string that can have the following values: • `NULL` • `INTEGER` • `DOUBLE` • `DECIMAL` • `DATE` • `TIME` • `TIMESTAMP`

16.2.6.2.2 `ConvertDate`

Description

Converts a string of a given format to date.

Syntax

value `convertDate(<String>: <String>)`

Input

Parameter	Description	Data Type
`<string>`	A string.	String

16.2.6.2.3 `Current Date (curDate)`

Description

Returns the current date.

Syntax

date `curDate()`

16.2.6.2.4 `Current Time (curTime)`

Description

Returns the current time.

Syntax

time `curTime()`

16.2.6.2.5 `Decrement Days`

Description

Decrements a given date by the given number of days.

Syntax

date `decrementDays(<date>, <number of days>)`

Input

Parameter	Description	Data Type
`<date>`	A date.	Date or DateTime
`<number of days>`	The number of days to decrement the date.	Integer

16.2.6.2.6 `Increment Days`

Description

Increments a given date by the given number of days.

Syntax

date `incrementDays(<date>, <number of days>)`

Input

Parameter	Description	Data Type
`<date>`	A date.	Date or DateTime
`<number of days>`	The number of days to increment the date.	Integer

16.2.6.2.7 `Now`

Description

Returns the current date and time.

Syntax

dateTime `now`()

16.2.6.2.8 `Timestamp Add`

Description

Returns a timestamp calculated by adding the given number of intervals to the given timestamp.

Syntax

dateTime `timestampAdd`(`<interval>`, `<count>`, `<timestamp>`)

Input

Parameter	Description	Data Type
`<interval >`	An interval constant. This parameter can either be a string or integer constant as follows: • 'SQL_TSI_FRAC_SECOND' or 0 • 'SQL_TSI_SECOND' or 1 • 'SQL_TSI_MINUTE' or 2 • 'SQL_TSI_HOUR' or 3 • 'SQL_TSI_DAY' or 4 • 'SQL_TSI_WEEK' or 5 • 'SQL_TSI_MONTH' or 6 • 'SQL_TSI_QUARTER' or 7 • 'SQL_TSI_YEAR' or 8	String or integer
`<count >`	The number of intervals to add to the timestamp.	Integer
`<timestamp>`	A date and time.	DateTime

Notes

The calculation can be impacted by daylight saving time in the locale for 'SQL_TSI_HOUR'.

16.2.6.2.9 To Date

Description

Converts a character string to a date.

Syntax

date toDate(<string>)

Input

Parameter	Description	Data Type
`<string>`	A string containing a date value in the format: yyyy-mm-dd, where yyyy is the year, mm is the month, and dd is the day. For example, 2003-09-07 and 2003-11-29.	String

Notes

- If `<string>` does not use the correct format, an error is returned.
- No restrictions are imposed on the month, day, or year values. If the month is greater than 12 or the day does not exist in the corresponding month, the function uses the internal calendar to convert to the correct date.

Examples

`toDate`('2003-02-12') = February 12, 2003

`toDate`('2003-02-29') = March 1, 2003

`toDate`('2002-14-12') = February 12, 2003

`toDate`('1994-110-12') = February 12, 2003

16.2.6.2.10 `To Time`

Description

Converts a given value to a time.

Syntax

time `toTime`(`<expression>`)

Input

Parameter	Description	Data Type
`<expression>`	A value. If the value is a string, the input must be in the format: hh:mm:ss where hh is the hour, mm is the minutes, and ss is the seconds. For example, 23:09:07 and 03:11:23.	String, Date, Time, or Date-Time

Notes

- If `<expression>` does not use the correct format, an error is returned.
- No restrictions are imposed on the hour, minutes, or seconds values. If the minutes or seconds are greater than 60, or if the hour is greater than 24, the function uses the internal clock to convert to the correct time.

Examples

`toTime`('02:10:09') = '02:10:09'

`toTime`('0:450:29) = '07:30:29'

`toTime`('25:14:180') = '01:17:00'

16.2.6.2.11 `To Timestamp`

Description

Converts a given value to a date and time.

Syntax

time `toTimestamp`(`<expression>`)

Input

Parameter	Description	Data Type
`<expression>`	A value. If the value is a string, the input must be in the format: yyyy-mm-dd hh:mm:ss.ssss, where yyyy is the year, mm is the month, dd is the day, hh is the hour, mm is the minutes, ss is the seconds, and ssss is the milliseconds (optional). For example, 2003-09-07 23:09:07 and 2003-11-29 03:11:23.0.	String, Date, Time, or Date-Time

Notes

- If `<expression>` does not use the correct format, an error is returned.
- No restrictions are imposed on the month, day, or year values. If the month is greater than 12 or the day does not exist in the corresponding month, the function uses the internal calendar to convert to the correct date.
- No restrictions are imposed on the hour, minutes, or seconds values. If the minutes or seconds are greater than 60, or if the hour is greater than 24, the function uses the internal clock to convert to the correct time.

Examples

`toTimestamp`('2003-02-12 02:10:09') = '2003-02-12 02:10:09.0'

`toTimestamp`('2003-02-29 02:10:09') = '2003-03-01 02:10:09.0'

`toTimestamp`('2002-14-12 02:10:09') = '2003-02-12 02:10:09.0'

`toTimestamp`('1994-110-12 02:10:09') = '2003-02-12 02:10:09.0'

`toTimestamp`('2003-02-12 0:450:29') = '2003-02-12 07:30:29.0'

`toTimestamp`('2002-09-09 25:14:180') = '2002-09-09 01:17:00.0'

16.2.6.3 Logical

Description

The list of logical functions

16.2.6.3.1 `Contains Only Digits`

Description

Returns true (1) if the given string contains only digits. Otherwise the function returns false (0).

Syntax

boolean `containsOnlyDigits(<string>)`

Input

Parameter	Description	Data Type
`<string>`	A string.	String

16.2.6.3.2 `Is Like`

Description

Checks a string for a matching pattern. Returns true (1) if the function finds a match for the given pattern in the given string.

Syntax

boolean `isLike(<string1>, <pattern>, <escape character>)`

Input

Parameter	Description	Data Type
`<string1>`	A string.	String
`<pattern>`	A string containing the pattern you are trying to match in `<string1>`. The pattern can contain wildcard characters: • The underscore character (_) matches any single character. • The percent sign character (%) matches any string of characters. To match an underscore or percent sign in `<string1>`, define an escape character in `<escape character>` and precede the underscore or percent sign in `<pattern>` with the escape character.	String
`<escape character>` (optional)	A character that allows you to match wildcard characters in `<string1>`.	String

Notes

- Returns null if either `<string1>` or `<pattern>` is null.
- If `<escape character>` is specified and is null, returns null.
- If `<escape character>` is specified, every occurrence of the escape character in `<pattern>` must be followed by either an underscore or percent sign.

Examples

`isLike` ('ABCD', 'AB%') = true

`isLike` ('ABCD', 'AB_D') = true

`isLike` ('10000', '100%') = true

`isLike` ('10000', '100\%', '\') = false

`isLike` ('status: 100%', '100\%', '\') = true

16.2.6.3.3 `To Boolean`

Description

Converts a given value to a Boolean value.

Syntax

boolean `toBoolean(<expression>)`

Input

Parameter	Description	Data Type
`<expression>`	A value or expression.	String or Boolean

Examples

`toBoolean`('true') = 1

`toBoolean`('TrUe') = 1

`toBoolean`('tru') = 0

`toBoolean`('False') = 0

`toBoolean`('F') = 0

`toBoolean`('f') = 0

16.2.6.4 Numeric

Description

The list of numeric functions

16.2.6.4.1 Absolute (abs)

Description

Returns the absolute value of a given integer value.

Syntax

numeric abs(<expression>)

Input

Parameter	Description	Data Type
<expression>	A numeric expression.	Numeric

Notes

- Returns null if the input <expression> is null.
- If <expression> is equal to the most negative value possible for an integer (-2 to the power of 31), that same negative value is returned.

16.2.6.4.2 Angle Tangent 2 (atan2)

Description

Returns the angle in radians whose tangent is <angle1>/<angle2>.

Syntax

numeric atan2(<angle1>, <angle2>)

Input

Parameter	Description	Data Type
`<angle1 >`	An angle.	Numeric
`<angle2>`	An angle.	Numeric

Notes

Returns null if both `<angle1>` and `<angle2>` = 0.

Examples

`atan2`(x,y) converts rectangular coordinates (x, y) to polar (r, theta). This method computes the phase theta by computing an arc tangent of y/x in the range of -Pi to Pi.

16.2.6.4.3 `Arc Cosine (acos)`

Description

Returns the arc cosine of a given numeric expression.

Syntax

numeric `acos(<expression>)`

Input

Parameter	Description	Data Type
`<expression>`	An expression in the range from 0 to Pi.	Numeric

Notes

Returns null if `abs(<expression>)` > 1.

16.2.6.4.4 `Arc Sine (asin)`

Description

Returns the arc sine of a given numeric expression.

Syntax

numeric `asin(<expression>)`

Input

Parameter	Description	Data Type
`<expression >`	An expression in the range from -Pi/2 to Pi/2.	Numeric

Notes

Returns null if `abs(<expression>)` > 1.

16.2.6.4.5 `Arc Tangent (atan)`

Description

Returns the arc tangent of a given numeric expression.

Syntax

numeric `atan(<expression>)`

Input

Parameter	Description	Data Type
`<expression>`	An expression in the range from -Pi/2 to Pi/2.	Numeric

16.2.6.4.6 `Average`

Description

Returns the average value.

Syntax

value `avg(<Sum of values>: <Numeric>)`

16.2.6.4.7 `Ceil (ceiling)`

Description

Returns the value of a number rounded up to the nearest integer.

Syntax

numeric `ceiling(<expression>)`

Input

Parameter	Description	Data Type
`<expression>`	A numeric expression.	Numeric

Notes

The type of the value returned is not converted. Therefore, `ceiling`(1.9) = 2.0. If you want to convert the value to an integer, use the conversion function `toInteger`.

16.2.6.4.8 `Charindex (pos) (locate)`

Description

Returns the position of a search string in a given character string.

Syntax

integer `pos`(`<search string>`, `<string>`, `<start position>`)

integer `locate`(`<search string>`, `<string>`, `<start position>`)

Input

Parameter	Description	Data Type
`<search string>`	The string you want to find the position of in `<string>`.	String
`<string>`	The string you want to search.	String
`<start position>`	The position in `<string>` where you want to start searching. If `<start position>` is not specified, the default start is position 1.	Integer

Notes

Returns 0 if the search string is not found.

Returns 0 if `<start position>` is longer than the length of `<string>`.

If `<start position>` <= 0, the search starts at position 1.

Examples

`pos`('cd','abcd') = 3

`pos`('abc', 'abcd') = 1

`pos`('cd', 'abcdcd') = 3

`pos`('cd', 'abcdcd', 3) = 3

`pos`('cd', 'abcdcd', 4) = 5

`pos`('ef', 'abcd') = 0

16.2.6.4.9 `Cosine (cos)`

Description

Returns the cosine of an angle.

Syntax

numeric `cos(<angle>)`

Input

Parameter	Description	Data Type
`<angle>`	An angle in radians.	Numeric

16.2.6.4.10 `Cotangent (cot)`

Description

Returns the cotangent of an angle in radians.

Syntax

numeric `cot(<angle>)`

Input

Parameter	Description	Data Type
`<angle>`	An angle in radians.	Numeric

Notes

Returns null if `sin(<angle>)` = 0.

16.2.6.4.11 `Day Of Month`

Description

Returns an integer from 1 to 31 representing the day of the month of a given date.

Syntax

integer `dayOfMonth(<date>)`

Input

Parameter	Description	Data Type
`<date>`	A date.	Date or DateTime

16.2.6.4.12 `Day Of Week`

Description

Returns an integer between 1 and 7 representing the day of the week of a given date. The first day of the week is Sunday.

Syntax

integer `dayOfWeek(<date>)`

Input

Parameter	Description	Data Type
`<date>`	A date.	Date or DateTime

16.2.6.4.13 `Day Of Year`

Description

Returns an integer between 1 and 366 representing the day of the year of a given date.

Syntax

integer `dayOfYear(<date>)`

Input

Parameter	Description	Data Type
`<date>`	A date.	Date or DateTime

16.2.6.4.14 `Degrees`

Description

Converts an angle measured in radians to an approximately equivalent angle measured in degrees.

Syntax

numeric `degrees(<angle>)`

Input

Parameter	Description	Data Type
`<angle>`	An angle in radians.	Numeric

16.2.6.4.15 `Exp`

Description

Returns the value of the mathematical constant `e` raised to the given exponent.

Syntax

numeric `exp(<exponent>)`

Input

Parameter	Description	Data Type
`<exponent>`	The exponential power.	Numeric

Examples

`exp`(10) = `e` to the power of 10 = 22,026.4658.

16.2.6.4.16 `Floor`

Description

Returns the value of a number rounded down to the nearest integer.

Syntax

numeric `floor`(`<expression>`)

Input

Parameter	Description	Data Type
`<expression>`	A numeric expression.	Numeric

Notes

The type of the value returned is not converted. Therefore, `floor`(1.9) = 1.0. If you want to convert the value to an integer, use the conversion function `toInteger`.

16.2.6.4.17 `Hexa To Int`

Description

Converts the hexadecimal value given by a string to integer.

Syntax

integer `hexaToInt(<sting>)`

Input

Parameter	Description	Data Type
`<string>`	A string containing a hexadecimal value.	String

Examples

```
hexaToInt('AF') = 175
```

16.2.6.4.18 `Hour`

Description

Returns an integer from 0 to 23 representing the hour of a given time.

Syntax

integer `hour(<time>)`

Input

Parameter	Description	Data Type
`<time>`	A time.	DateTime

16.2.6.4.19 `Int To Hexa`

Description

Converts a given integer to hexadecimal. The hexadecimal value is returned in a string.

Syntax

string `intToHexa(<value>)`

Input

Parameter	Description	Data Type
`<value>`	An integer.	Integer

Notes

- To ensure that the input value is of data type integer, you can use the `toInteger` function: `intToHexa(toInteger (<value>))`.
- If `<value>` < 0, the function returns 'FFFFFFFF'.

16.2.6.4.20 `Length`

Description

Returns the length of a given string. Spaces are counted.

Syntax

integer `length(<string>)`

Input

Parameter	Description	Data Type
`<string>`	A string.	String

16.2.6.4.21 `Locate`

Description

Returns the position of a character in a character string.

Syntax

value `locate(<Search>: <String>), (<Character string>: <String>)`

16.2.6.4.22 `Log`

Description

Returns the natural logarithm of the given value.

Syntax

double `log(<expression>)`

Input

Parameter	Description	Data Type
`<expression>`	A numeric expression > 0.	Double

Notes

Returns null if `<expression>` is <= 0.

16.2.6.4.23 `Log10`

Description

Returns the common logarithm (base 10) of the given value.

Syntax

double `log10(<expression>)`

Input

Parameter	Description	Data Type
`<expression>`	A numeric expression > 0.	Double

Notes

Returns null if `<expression>` is <= 0.

16.2.6.4.24 `Maximum`

Description

Selects the maximum value.

Syntax

value `max(<Set of values>: <All>)`

16.2.6.4.25 `Minute`

Description

Returns an integer from 0 to 59 representing the minutes of a given date and time.

Syntax

integer `minute(<time>)`

Input

Parameter	Description	Data Type
`<time>`	A date and time.	DateTime

16.2.6.4.26 `Minimum`

Description

Selects the minimum value.

Syntax

value `min(<Set of values>: <All>)`

16.2.6.4.27 `Mod`

Description

Returns the remainder of the division of two integers: value1 / value2.

Syntax

integer `mod(<value1>, <value2>)`

Input

Parameter	Description	Data Type
`<value1>`	Value of the numerator.	Numeric
`<value2>`	Value of the divisor not equal to 0.	Numeric

Notes

Returns null if `<value2>` = 0.

16.2.6.4.28 `Number of the Month (month)`

Description

Returns an integer from 1 to 12 representing the month of a given date.

Syntax

integer `month(<date>)`

Input

Parameter	Description	Data Type
`<date>`	A date.	Date or DateTime

16.2.6.4.29 `Number of the Week (week)`

Description

Returns an integer from 1 to 53 representing the week in the year of a given date.

Syntax

integer `week(<date>)`

Input

Parameter	Description	Data Type
`<date>`	A date.	Date or DateTime

Notes

The first day of the week is Sunday. The first week in the year must contain at least one day. If January 1 is a Saturday, the following rules apply:

- January 1 is week 1.
- January 2 to 8 is week 2.
- December 25 to 31 is week 53.

16.2.6.4.30 `Pi`

Description

Returns the constant value of Pi.

Syntax

numeric `pi()`

16.2.6.4.31 Power

Description

Returns the value of a number raised to the power of the given exponent.

Syntax

numeric power(<value>, <exponent>)

Input

Parameter	Description	Data Type
<value>	The base value.	Numeric
<exponent>	The exponent.	Integer

Notes

Returns null if <value> = 0 and <exponent> > 0.

16.2.6.4.32 Quarter

Description

Returns an integer from 1 to 4 representing the quarter of a given date. The value 1 represents January 1 through March 31.

Syntax

integer quarter(<date>)

Input

Parameter	Description	Data Type
`<date>`	A date.	Date or DateTime

16.2.6.4.33 `Radians`

Description

Converts an angle measured in degrees to an approximately equivalent angle measured in radians.

Syntax

numeric `radians(<angle>)`

Input

Parameter	Description	Data Type
`<angle>`	An angle in degrees.	Numeric

16.2.6.4.34 `Random (rand)`

Description

Returns a random number between 0 and 1. You can optionally provide a seed integer to initialize the random number generator.

Syntax

numeric `rand(<value>)`

Input

Parameter	Description	Data Type
`<value>` (optional)	Seed value for the random number generator.	Integer

16.2.6.4.35 `Round`

Description

Returns a number rounded to the given number of decimal places.

Syntax

numeric `round`(`<expression>`, `<number of places>`)

Input

Parameter	Description	Data Type
`<expression>`	The base value to be rounded.	Numeric
`<number of places>`	The number of decimal places to round to.	Numeric

Notes

- The function rounds towards the nearest neighboring integer unless both neighbors are equidistant, in which case it rounds away from zero.
- The type of the value returned is not converted. Therefore, `round`(1.9) = 2.0. If you want to convert the value to an integer, use the conversion function `toInteger`.

16.2.6.4.36 `Rpos`

Description

Returns the position of the last occurrence of a search string in a given character string.

Syntax

integer `rpos(<search string>, <string>)`

Input

Parameter	Description	Data Type
`<search string>`	The string you want to find the last occurrence of in `<string>`.	String
`<string>`	The string you want to search.	String

Notes

Returns 0 if the search string is not found.

Examples

`rpos`('cd','abcd') = 3

`rpos`('cd', 'abcdcd') = 5

`rpos`(abc', 'abcdcd') = 1

`rpos`('ef', 'abcd') = 0

16.2.6.4.37 `Second`

Description

Returns an integer from 0 to 59 representing the seconds of a given date and time.

Syntax

integer `second(<time>)`

Input

Parameter	Description	Data Type
`<time>`	A date and time.	DateTime

16.2.6.4.38 `Sign`

Description

Returns the positive (1), zero (0), or negative (-1) sign of a given number.

Syntax

numeric `sign(<value>)`

Input

Parameter	Description	Data Type
`<value>`	A numeric value.	Integer

16.2.6.4.39 `Sine (sin)`

Description

Returns the sine of an angle.

Syntax

numeric `sin(<angle>)`

Input

Parameter	Description	Data Type
`<angle>`	An angle in radians.	Numeric

16.2.6.4.40 `Sqrt`

Description

Returns the square root of a number.

Syntax

numeric `sqrt(<expression>)`

Input

Parameter	Description	Data Type
`<expression>`	An expression >= 0.	Numeric

Notes

Returns null if expression < 0.

16.2.6.4.41 `Sum`

Description

Returns the sum of the values.

Syntax

```
sum(<Set of values>: <Numeric>)
```

16.2.6.4.42 Tangent (tan)

Description

Returns the tangent of an angle.

Syntax

numeric `tan(<angle>)`

Input

Parameter	Description	Data Type
`<angle>`	An angle in radians.	Numeric

Notes

Returns null if `cos(<angle>)` = 0.

16.2.6.4.43 To Decimal

Description

Converts a given value to a decimal.

Syntax

decimal `toDecimal(<expression>)`

Input

Parameter	Description	Data Type
`<expression>`	A value. If the value is a string, it must be in decimal number format and use the period character (.) as the decimal separator.	Numeric or String

16.2.6.4.44 `To Double`

Description

Converts a given value to a decimal.

Syntax

double `toDouble(<expression>)`

Input

Parameter	Description	Data Type
`<expression>`	A value. If the value is a string, the input must be in decimal number format and use the period character (.) as the decimal separator.	Numeric or String

16.2.6.4.45 `To Integer`

Description

Converts a given value to an integer.

Syntax

integer `toInteger(<expression>)`

Input

Parameter	Description	Data Type
`<expression>`	A value. If the value is a string, the input must be in number format.	Numeric or String

16.2.6.4.46 `Trunc`

Description

Returns a number truncated to a given number of decimal places.

Syntax

numeric `trunc(<expression>, <number of places>)`

Input

Parameter	Description	Data Type
`<expression>`	The base value to be truncated.	Decimal
`<number of places>`	The number of decimal places to remain after truncating.	Integer

Notes

- If `<number of places>` is omitted, the number is truncated to 0 decimal places.
- If `<number of places>` is negative, the function starts at the digit the number of places to the left of the decimal point and sets to zero all the digits to the right of that position.

Examples

`trunc`(10.1234, 1) = 10.1

`trunc`(10.1234, 2) = 10.2

`trunc`(1862.1234, -1) = 1860

`trunc`(1862.1234, -2) = 1800

16.2.6.4.47 `Year`

Description

Returns an integer representing the year of a given date.

Syntax

integer `year`(`<date>`)

Input

Parameter	Description	Data Type
`<date>`	A date.	Date or DateTime

16.3 @Functions

@Functions are special functions that provide more flexible methods for specifying the query script for an object. Select the related topic to see more about an @Function.

Related Information

@Aggregate_Aware [page 490]
@DerivedTable [page 490]
@Execute [page 491]

@Prompt [page 492]
@Select [page 497]
@Variable [page 498]
@Where [page 499]

16.3.1 @Aggregate_Aware

Use the `@Aggregate_Aware` function in the SQL definition of a business layer object to make the object aggregate-aware. When the object is included in a query, the aggregate tables listed as parameters in the `@Aggregate_Aware` function are queried first.

The syntax is:

```
@Aggregate_Aware(sum(<Aggregate table 1>), ... sum(<Aggregate table n>))
```

`<Aggregate table 1>` is the aggregate table with the highest level of aggregation, and `<Aggregate table n>` is the aggregate table with the lowest level.

For more information about aggregate awareness in the universe, see the related topic.

Related Information

Aggregate awareness [page 262]

16.3.2 @DerivedTable

Use the `@DerivedTable` function in the definition of nested derived tables. A nested derived table (also known as a 'derived table on a derived table') is a table that is derived from at least one existing derived table.

The syntax of the `@DerivedTable` function is:

```
@DerivedTable(<Derived table name>)
```

`<Derived table name>` is the name of the derived table you want to reference. The `@DerivedTable` function is only used in the definition of derived tables in the data foundation.

> **i Note**
>
> In database-specific SQL (multisource-enabled data foundations), all tables referenced must be from the same connection.

16.3.3 @Execute

The `@Execute` function lets you define a preliminary query that provides a list of values in a `SELECT` predicate to be included in the main query. The `@Execute` function is based on standard SQL and so applies to relational data sources. The syntax of the `@Execute` function is:

`@Execute(<List of values>)`

`< List of values>` is a list of values pre-defined in the business layer or data foundation. The list of values definition provides the preliminary query. Most often the `@Execute` function is then included in a filter or `WHERE` clause to apply the preliminary query to limit the values returned in the main query.

The list of values can be any of the following types:

- List of values based on custom SQL
- Static list of values
- List of values based on a query that includes business layer objects

The following limitations apply:

- The list of values cannot be based on a custom hierarchy.
- The list of values can contain only objects that are active in the business layer (not hidden or deprecated).
- The SQL defining the list of values cannot contain the @Execute function.
- The @Execute function cannot be used in the definition of an @Prompt function.

For more information on inserting a list of values, see the related topic.

Example: Filter on products

This example creates a query filter that limits query results to products with sales that are two times above the product category average.

First, create the list of values that returns the product IDs of products with sales above the category average. The list of values name is **`Products_Above_Avg`**, and the data type is numeric. The following SQL defines the list of values:

```
 WITH
PA as
(
 SELECT L.PRODUCT_ID, sum(L.NET_SALES) AS SALES
 FROM SO_LINE L
 GROUP BY L.PRODUCT_ID
 ),
 CA as
 SELECT P.CATEGORY_ID, avg(A.SALES) AS SALES
 FROM PRODUCT P, PA A
 WHERE P.PRODUCT_ID = A.PRODUCT_ID
 GROUP BY P.CATEGORY_ID
 )
 SELECT PA.PRODUCT_ID
 FROM PA, CA, PRODUCT P
 WHERE PA.PRODUCT_ID = P.PRODUCT_ID
 AND P.CATEGORY_ID = CA.CATEGORY_ID
 AND PA.SALES > ( CA.SALES * 2)
```

Next, insert a native filter into the business layer that invokes the preliminary query using the `@Execute` function in the `WHERE` clause. Because the `@Execute` function can return multiple values, use the `IN` operator in the filter definition:

```
PRODUCT.PRODUCT_ID IN
 @Execute(Products_Above_Avg)
```

When the filter is included in a query, the `@Execute` function is replaced with the resulting list of product IDs, for example:

```
PRODUCT.PRODUCT_ID in (2, 5, 20, 33, 35)
```

Example: Include a security predicate

This example inserts a column filter that returns sales data only for the geographical region of the current user.

First, create the list of values in the data foundation that returns the authorized country codes for the current user. The list of values name is **`Authorized_Countries`**, and the data type is numeric. This example assumes the database administrator has set up a table called **`user_geography`** in the database that associates authorized countries with each user. The following SQL defines the list of values:

```
 SELECT country_id
 FROM user_geography
 WHERE user_name = @Variable('BOUSER')
```

Next, insert a column filter into the data foundation table **`Sales`**. Because the `@Execute` function can return multiple values, use the `IN` operator in the filter definition.

```
          Sales.country_id
          IN @Execute(Authorized_Countries)
```

When a user includes the **`Sales`** table in a query, the `@Execute` function in the column filter is replaced by the list of authorized country codes for that user.

Related Information

Inserting and editing filters [page 276]
Inserting a column filter [page 179]

16.3.4 @Prompt

Use the `@Prompt` function to insert a prompt into a query. Prompts can be used to restrict the data when a user creates a report. You use the `@Prompt` function in the SQL `SELECT` statement or `WHERE` clause, or in the MDX

expression for an object. It forces a user to enter one or more values (or select from a list of values) for a restriction when that object is used in a query. When the user runs the query, a prompt box appears asking for a value to be entered or selected.

Prompts are useful when you want to force a restriction in the query script, but do not want to preset the value of the condition.

The `@Prompt` function is allowed in the following expressions:

- Joins
- Calculated columns (except in database-specific SQL in multisource-enabled data foundations)
- Derived tables
- Business objects in the business layer

You can insert a `@Prompt` definition in the following ways:

- Define a named parameter for the prompt and reference the parameter in the `@Prompt` function, for example:
  `@Prompt(<Parameter name>)`
  `<Parameter name>` is a parameter pre-defined in the data foundation or business layer. For more information, see the related topic about parameters.
- Type the prompt definition into the SQL or MDX expression of the object. For more information on the syntax and parameters of the `@Prompt` function, see the related topic.

Related Information

Parameters [page 238]
@Prompt syntax [page 493]

16.3.4.1 @Prompt syntax

The syntax for the `@Prompt` function is as follows:

```
@Prompt('<message>',
'<type>',
'<folder\business layer object>'|'<list of values>'|{'<value_1>','<value_2>',...},
Mono|Multi:Any|Leaf,
free|constrained|primary_key,
persistent|not_persistent,
{'<default_value_1>',...'<default_value_n>'},
,,'<formula_expression>')
```

The function parameters and the possible values are described in the following table. The parameters are separated by commas. You must specify at least the first two parameters. If you want to specify additional parameters, you must include the intervening commas for the optional parameters.

> **i Note**
>
> The two commas before `<formula_expression>` are required as placeholders for arguments not supported in the @Prompt defined in the information design tool.

Parameter	Description
'`<message>`'	Text of the prompt message. This parameter is mandatory. The text appears in the prompt box when the user runs the query. The text must be enclosed in single quotes, for example, 'Choose a Region'. For proper function of the prompt, the prompt text should be unique within the universe.
'`<type>`'	The data type of the prompt. This parameter is mandatory. The user's response is interpreted with the data type you specify. The list of values and default values also have this data type. It can be one of the following: • 'A' for alphanumeric string. • 'K' for keyword. This type is also an alphanumeric string, however the responses to the prompt will be not be surrounded by quotes in the query script at run time. • 'N' for number. • 'D' for date. • 'DT' for date-time. The specified `<type>` must be enclosed in single quotes. The `<type>` parameter can be a pair of data types to indicate a name and key. The syntax is '`<name_type>`:`<key_type>`', for example: 'A':'N' where the first type is the data type of the name that the user sees in the list of values, and the second type is the data type of the primary key that is used by the query. Both the `<name_type >`and the `<key_type>` can be any of the available data types. **i Note** To use this option, you must make sure the object and the prompt are index aware: • Define a primary key for the object in the business layer. • Specify `primary key` for the fifth parameter in the `@Prompt` function. In this case, if the list of values or default values parameters are used, they must contain a list of pairs of values.

Parameter	Description
'<folder\business layer object>' | '<list of values>' | {'<value_1>','<value_2>',... }	The list of values the user can choose from when prompted. This parameter is optional. You can specify a list of values in three ways: • The default list of values associated with an object in the business layer (a dimension, measure, attribute, hierarchy, or hierarchy level). Enter the full path and object name in the business layer between single quotes, for example: '**Myconnection\dimproduct\productname**' In this example, **productname** is the business layer object name. The object must be index aware, that is, a primary key is defined for the object in the business layer. For more information, see the related topic on defining keys. • A named list of values defined in the business layer or data foundation. Enter the name of the list of values between single quotes, for example: '**G7_Countries**'. If the list of values is hierarchical with named levels, you can specify the level to use for the prompt, for example: '**Country_Region_City_List**':'**Region**' In this example, **Country_Region_City_List** is the name of the list of values and **Region** is the target level. If the list of values is a multiple column list of values with named columns, you can specify the column to use for the prompt, for example: **'Country_Region_City_List':'Region'**. In this example, **Country_Region_City_List** is the name of the list of values and **Region** is the target column. • A hard-coded list of values or name/key pairs. The values in a pair are separated by a colon. Each value is enclosed in single quotes. The pairs of values are separated by a comma. The whole list is enclosed in braces: The syntax for a single value: {'<value>'} The syntax for several single values: {'<value_1>','<value_2>',... ,'<value_n>'} The syntax for a pair of values: {'<name_value>':'<key_value>'} The syntax for pairs of values: {'<name_value_1>':'<key_value_1>','<name_value_2>':'<key_value_2>',...,'<name_value_n>' :'<key_value_n>'}. For example: {'**France**':'**FR**', '**Germany**':'**DE**','**Spain**':'**ES**', '**United Kingdom**':'**UK**'} **i Note** If the list of values is index aware (a primary key has been defined for the object in the business layer, or you use {name, key} pairs for the list of values, specify `primary key` for the fifth parameter in the `@prompt` function.

Parameter	Description

`Mono` |
`Multi`
`:Any` | `Leaf`
	The selection mode. If not specified, `Mono` is default.
	• Use `Mono` if the user can select only one value from the list of values.
	• Use `Multi` if the user can select multiple values from the list of values.
	You can optionally specify the hierarchical selection mode for hierarchical lists of values. If not specified, `Leaf` is default:
	• Use `Any` if the user can select any member/value at any level of the hierarchical list of values.
	• Use `Leaf` if the user can select only the leaf members/values from the hierarchical list of values.

`free` | `constrained` | `primary_key`	The entry constraint type. If not specified, `free` is default. • Use `free` if the user can enter a value, or select from the list of values. • Use `constrained` if the user must select values from the list of values. • Use `primary_key` when you are using an index aware object or {name, key} pairs. The associated key value for the object is used in the query rather than the entered or displayed name value.
`persistent` | `not_persistent`	Whether or not the last values are displayed. If not specified, `not_persistent` is default. Use `persistent` if, when refreshing a document, the last values used in the prompt are displayed by default, even when default values are defined. Use `not_persistent` if, when refreshing a document, no values used are displayed in the prompt by default.
{'`<default value>`'}	One or more default values presented to the user. This parameter is optional. Enter default values in the following ways: • For a single value: {'**`France`**'} • For a pair of values: {'**`France`**':'**`FR`**'} • For two pairs of values: {'**`France`**':'**`FR`**','**`Germany`**':**`DE`**'} • For hierarchical values, use \ to separate the hierarchy level values: {'**`Europe`**':'2'\'**`France`**'\'**`Marseille`**'\'**`CSP Systems`**','**`Europe`**':'2'\'**`Germany`**'\'**`Berlin`**'} When refreshing a document, these values are displayed by default, but if the `persistent` option is set, then the last values used in the prompt are used instead of the default values. If you specify the `primary_key` parameter in the prompt definition, you must provide the key values.

Parameter	Description
`<formula_expression>`	The formula to return a dynamic default value for a parameter. You use the semantic layer formula language to define the prompt. Refer to Related Links for information on the syntax and conventions for building a formula for dynamic default values. • In the `<formula_expression>`, you escape a quote (') with a double quote ('). • The `<formula_expression>` is only valid if the prompt constraint option is `free` or `constrained`. If `primary_key` is included, then an error is returned for an invalid prompt definition. For example: Sample Code `@Prompt('Select the min date:', 'D', , mono, free, not_persistent, , user:0,, 'currentDate() - 7')`

Related Information

Defining keys for dimensions and dimension attributes [page 269]

16.3.5 @Select

Use the `@Select` function in the definition of an object in the business layer to re-use the `SELECT` statement of another object. The syntax of the `@Select` function is:

`@Select(<Folder name>\<Object name>)`

`<Folder name>\<Object name>` specifies the full path of another object in the business layer.

For example, you define a business layer object **`Promotional_Service_Line`** as `@Select`(**`Resort\Service_Line`**). The `SELECT` statement defined for **`Service_Line`** is used for the definition of **`Promotional_Service_Line`**.

> **Note**
>
> Ensure that the `<Object name>` does not contain parentheses, for example `@Select(Time period\(Year))`. When parsed, the special character *§* is added before and after the *( )* in the `@Select` definition which is recognized as a syntax error by Check Integrity. It is not recomended that you use parentheses in object names when naming objects in the universe.

Using the `@Select` function lets you maintain only one instance of the SQL or MDX expression, and ensures the consistency of related object definitions in the business layer. However, `@Select` creates an object dependency. If you delete the source object, you must manually update the object that uses the `@Select` function.

16.3.6 @Variable

Use the `@Variable` function in an SQL or MDX expression (usually in the `WHERE` clause) to retrieve the value assigned to a system variable or User Attribute. The syntax of the `@Variable` function is:

```
@Variable('<Variable name>'[,DELIMITER=default | no_quote])
```

`<Variable name>` must be in single quotes. Possible variables are described in the following table:

Variable Name and Description	Examples
Variables containing information about the user's authorization: • `BOUSER`: User name entered by the user to log into the SAP BusinessObjects BI platform. • `DBUSER`: User name used for authorization when connecting to the data source. This user name can be defined in the Central Management Console as part of the user's secondary credentials.	For example, to restrict data retrieved in a query to the current user, use the `BOUSER` variable in the `WHERE` clause: `WHERE Employees.Employee_Name = @Variable('BOUSER')`
Variables containing information about the current report or query: • `DOCNAME`: The document name. • `DOCID`: The document identifier. (If the document is published in the repository, the DOCID value corresponds to the document ID in the repository. If the document is not published in the repository, the DOCID value is EMPTY.) • `DPNAME`: The Data Provider name. • `DPTYPE`: The Data Provider type. • `UNVNAME`: The universe name. • `UNVID`: The universe identifier.	For example, these variables can be referenced in the `BEGIN_SQL` parameter that will be executed before the `SELECT` statement. This can be used for audit purposes concerning the use of the database (For example: To determine which report query or which universe is used most frequently).
Variables containing information about the user's current language settings: • `PREFERRED_VIEWING_LOCALE`: The user's preferred locale for viewing report and query objects in an application. • `DOMINANT_PREFERRED_VIEWING_LOCALE`: A pre-defined fallback locale that is used when no fallback locale is defined for the resource.	The following query retrieves the product names in the language determined by the user's Preferred Viewing Locale. The database must contain a column identifying the locale of the data. To see a list of locales, their abbreviations and dominant locales, refer to the *Translation Management Tool User Guide*. `SELECT Product_Name FROM Product WHERE Product.Locale = @Variable('PREFERRED_VIEWING_LOCALE')`

Variable Name and Description	Examples
User Attributes defined in the User Attribute Management area of the Central Management Console (CMC).	To reference a User Attribute, specify the internal name for the attribute as it is defined in the CMC. `@Variable` returns the value of the attribute for the current user. For example, the User Attribute **`MYCOUNTRY`** contains the value of the country of each user in the CMC. Specify the attribute's internal name surrounded by single quotes: `@Variable('SI_MYCOUNTRY')` The attribute's internal name is defined when the attribute is created in the CMC.

> **i Note**
>
> If the `<variable name>` specified in the `@Variable` function is unknown to the system, the user is prompted for a value. In this case, the `@Variable` function behaves in the same way as a single-value `@Prompt` function with the following settings:
>
> `@Prompt('<Variable name>','A',,Mono,free)`

The `DELIMITER` parameter specifies how the returned value for the variable is delimited in the query script. The parameter's default value is `DELIMITER=default`. This means the value is delimited by quotes for relational SQL data sources, and no delimiter for OLAP MDX data sources.

If you specify `DELIMITER=no_quote`, this means that no delimiter character is added around the value in the script.

The `@Variable` function is allowed in the following expressions:

- Joins
- Calculated columns
- Derived tables
- Object definitions in the business layer
- BEGIN_SQL and END_SQL statements
- Connection properties, such as the ConnectInit property (except the `DELIMITER` parameter, which is not supported in connection properties)

In database-specific SQL (multisource-enabled data foundations), all referenced tables or columns must be from the same connection.

16.3.7 @Where

Use the `@Where` function in the SQL definition of an object in the business layer to re-use the `WHERE` clause of another object. The syntax of the `@Where` function is:

`@Where(<Folder name>\<Object name>)`

`<Folder name>\<Object name>` specifies the full path of another object in the business layer.

For example, you define the `WHERE` clause of business layer object **`Resort_Service_Line`** as `@Where`(**`dimResort\Resort`**). The `WHERE` statement defined for the object **`Resort`** is used for the definition of **`Resort_Service_Line`**.

Using the `@Where` function lets you maintain only one instance of the SQL `WHERE` clause, and ensures the consistency of related object definitions in the business layer. However, `@Where` creates an object dependency. If you delete the source object, you must manually update the object that uses the `@Where` function.

16.4 SQL generation parameters

SQL generation parameters affect the generation of the query script. The parameters all have default values. Default values can be overridden in the data foundation properties. Some parameters (concerning lists of values) can also be overridden in the business layer properties. At query time, the query server will use the values it finds in the following order:

1. The value in the business layer, if it is set.
2. The value in the data foundation, if it is set.
3. The default value.

The following reference describes the parameters that affect the generation of the query script. The parameters are listed alphabetical order in two groups:

- SQL parameters that you set in the user interface of the information design tool. These are SQL parameters that are common to most data access drivers. Each parameter is valid for the universe in which it is set.
- SQL parameters that you set in the extended data access parameter (PRM) files. These are connection-specific parameters that are listed in the extended PRM file for the target data access driver.

Related Information

SQL generation parameters reference [page 500]
SQL generation parameters set in the extended PRM [page 514]
Data foundation properties [page 129]
Properties [page 218]

16.4.1 SQL generation parameters reference

The following reference describes the SQL generation parameters that can be overridden in the data foundation properties and business layer properties.

16.4.1.1 ANSI92

`ANSI92` = Yes|No

Values	`Yes/No`
Default	`No`
Description	Specifies whether the SQL generated complies to the `ANSI92` standard. `Yes`: Enables the SQL generation compliant to `ANSI92` standard. `No`: SQL generation behaves according to the `PRM` parameter `OUTER_JOIN_GENERATION`.

16.4.1.2 AUTO_UPDATE_QUERY

`AUTO_UPDATE_QUERY` = Yes|No

Values	`Yes/No`
Default	`No`
Description	Determines what happens when an object in a query is not available to a user profile. `Yes`: Query is updated and the object is removed from the query. `No`: Object is kept in the query.

16.4.1.3 BEGIN_SQL

`BEGIN_SQL` = <String>

Values	String
Default	Empty string

Description	BEGIN_SQL is used to prefix SQL statements for accounting, prioritization, and workload management. The parameter applies to any SQL generation, including document generation and list of values queries.
	
	BEGIN_SQL is supported in Web Intelligence , LiveOffice, Crystal Reports for Enterprise, and QaaWS. It is ignored by Desktop Intelligence.
	
	Example for Teradata:
	
	`BEGIN_SQL=SET QUERY_BAND='string' for transaction;`
	
	This parameter requires a string that contains one or more name-value pairs, separated by a semicolon, all inside single quotes. All SQL statements are prefixed with the parameter that follows BEGIN_SQL. The name-value pairs entered in this parameter are written in the GetQueryBandPairs system table.
	
	Example of three name-value pairs:
	
	`BEGIN_SQL=SET QUERY_BAND='UserID=Jones;JobID=980;AppID=TRM' for transaction;`
	
	You can also use the @Variable function as the value in the name-value pair, the returned value is enclosed in single quotes: `BEGIN_SQL=SET QUERY_BAND='USER='@Variable('BOUSER');Document='@Variable('DPNAME')';' for transaction;`

16.4.1.4 BLOB_COMPARISON

BLOB_COMPARISON = Yes|No

Values	Yes/No
Default	No
Can be edited?	No
Description	Species if a query can be generated with a DISTINCT statement when a BLOB file is used in the SELECT statement. It is related to the setting No Duplicate Row in the query properties. Yes: The DISTINCT statement can be used within the query. No: The DISTINCT statement cannot be used within the query even if the query setting No Duplicate Row is on.

16.4.1.5 BOUNDARY_WEIGHT_TABLE

`BOUNDARY_WEIGHT_TABLE` = Integer 32bits [0-9]

Values	Integer 32bits [0-9, or a negative integer]
Default	-1
Description	Allows you to optimize the `FROM` clause when tables have many rows. If the table size (number of rows) is greater than the entered value, the table is declared as a subquery: `FROM (SELECT col1, col2,....., coln, ,...., FROM Table_Name WHERE simple condition).` A simple condition is defined as not having a subquery. -1, 0, or any negative number means that this optimization is not used.
Limitations	Optimization is not implemented when: • The operator `OR` is in the query condition • Only one table is involved in the SQL • The query contains an outer join • No condition is defined on the table that is being optimized • The table being optimized is a derived table.

16.4.1.6 CUMULATIVE_OBJECT_WHERE

`CUMULATIVE_OBJECT_WHERE` = Yes|No

Values	`Yes|No`
Default	`No`

Description	This parameter applies to filtered objects only. Specifies how to combine the objects `WHERE` clause with the query condition on those objects.
	
	`Yes`: Specifies that `WHERE` clauses are combined with the main query condition with the `AND` operator.
	
	`No` : Specifies that the object's `WHERE` clause is combined with the condition for this object.
	
	Example:
	
	If the condition is find all French clients different from John or American cities different from New York, the SQL is:
	
	`Yes`:
	
	`(customer.first_name <> 'John') OR (city.city <> 'New York AND customer_country.country = 'France' AND city_country.country = 'USA'`
	
	`No`:
	
	`(customer.first_name <> 'John' AND customer_country.country = 'France' ) OR (city.city <> 'New York' AND city_country.country = 'USA' )`

16.4.1.7 DISABLE_ARRAY_FETCH_SIZE_OPTIMIZATION

`DISABLE_ARRAY_FETCH_SIZE_OPTIMIZATION` = Yes|No

Values	`Yes/No`
Default	`No`
Description	An optimization algorithm can be used to optimize the size of the returned arrays instead of using the default setting. `No`: All queries run on the universe will benefit from the optimization. `Yes`: Queries use the default value set.

16.4.1.8 DISTINCT_VALUES

`DISTINCT_VALUES` = GROUPBY|DISTINCT

Values	GROUPBY | DISTINCT
Default	DISTINCT
Description	Specifies whether SQL is generated with a DISTINCT or GROUP BY clause for objects in the business layer, and in lists of values. In the Query Panel, query generation takes the value of DISTINCT_VALUES into account only when the option *Retrieve duplicate rows* is unselected in the query properties. DISTINCT: The SQL is generated with a DISTINCT clause, for example: SELECT DISTINCT cust_name FROM Customer GROUPBY: The SQL is generated with a GROUP BY clause, for example: SELECT cust_name FROM Customer GROUP BY Customer.cust_name

16.4.1.9 END_SQL

END_SQL = String

Values	String
Default	<empty string>
Description	The statement specified in this parameter is added at the end of each SQL statement.
Example	• For a SAP HANA data source, you can pass the user name of a user executing a query from the universe to SAP HANA by inserting the @Variable('BOUSER') parameter as follows: END_SQL=-- @Variable('BOUSER') • For IBM DB2 databases, you can use the following: END_SQL=FOR SELECT ONLY. The server will read blocks of data much faster. • Another example: END_SQL='write ' UNVID To Usage_Audit.Querieded_universe writes a universe id to an audit table, which can be used to record other data such as user and tables queried.

16.4.1.10 EVAL_WITHOUT_PARENTHESIS

EVAL_WITHOUT_PARENTHESIS = Yes|No

Values	Yes | No
Default	No

Description	By default, the function `@Select`(folder\object) is replaced by the `SELECT` statement for the object <folder\object> enclosed within brackets. For example, when combining two `@Select` statements, `@Select`(objet1) *`@Select`(objet2). If the SQL(object1) = A-B and SQL(object2) =C, then the operation is (A-B) * (C). You avoid the default adding of brackets by setting `EVAL_WITHOUT_PARENTHESIS = Yes`. The operation is then A - B * C. `Yes`: Brackets are removed from the `SELECT` statement for a function `@Select`(folder\object) `No`: Brackets are added around the Select statement for the function `@Select`(folder\object).

16.4.1.11 FILTER_IN_FROM

`FILTER_IN_FROM = Yes|No`

Values	`Yes/No`
Default	`No`
Description	Determines if the generated SQL includes query filters in the FROM clause whenever possible. **i Note** This setting is only applicable if the SQL generation parameter `ANSI92` is set to `Yes`. This parameter is useful when querying tables that have outer joins defined. For example, an outer join on tables Customer and Reservations returns all customers, even those without reservations. A query filter in the WHERE clause might filter out customers without reservations. If the `FILTER_IN_FROM` parameter is set to `Yes`, the generated SQL includes query filters in the FROM clause whenever possible in order to preserve the null values returned by the outer join. `Yes`: When SQL is generated, query filters are put in the FROM clause whenever possible. `No`: When SQL is generated, query filters are put in the WHERE clause.

16.4.1.12 FORCE_SORTED_LOV

`FORCE_SORTED_LOV` = Yes|No

Values	`Yes` | `No`
Default	`No`
Description	Retrieves a list of values that is sorted. `Yes`: Specifies that the list of values is sorted. `No`: Specifies that the list of values is not sorted.

16.4.1.13 GROUPBY_PRIMARY_KEY

`GROUPBY_PRIMARY_KEY = YES | NO`

Values	`YES	NO`
Default	`YES`
Description	Allows you to deactivate the use of the primary key in the `GROUP BY` clause. By default, if data for an index aware object is retrieved, the SQL is optimized by using the primary key in the `GROUP BY` clause. `YES`: Favors the use of the primary key over the column name in the `GROUP BY` clause. `NO`: Does not use the primary key in the `GROUP BY` clause.

16.4.1.14 INNERJOIN_IN_WHERE

`INNERJOIN_IN_WHERE` = Yes|No

Values	`Yes` | `No`
Default	`No`. You must manually add the parameter to activate it.

Description	Allows you to force the system to generate SQL syntax with all the inner joins in the `WHERE` clause when `ANSI92` is set to yes. This is only possible if a query contains only inner joins (Does not contain FULL OUTER, RIGHT OUTER, or LEFT OUTER joins). `Yes`: If `ANSI92` is set to yes, the system generates `ANSI92` join syntax in the `FROM` clause except when the query contains only inner joins. In this case, the inner joins go into the `WHERE` clause. `No`: If `ANSI92` is set to `Yes`, the system generates `ANSI` 92 join syntax in the `FROM` clause.

16.4.1.15 JOIN_BY_SQL

`JOIN_BY_SQL` = Yes|No

Values	`Yes` | `No`
Default	`No`
Description	Specifies how multiple `SQL` statements are handled. Multiple statements can be combined (provided that the database permits this). `Yes`: Specifies that multiple SQL statements are combined. `No`: Specifies that multiple SQL statements are not combined. This is the default value.

16.4.1.16 MAX_INLIST_VALUES

`MAX_INLIST_VALUES` = [0-99]

Values	Integer: min-1, max depends on DB
Default	-1
Description	Allows you to set the maximum number of values you may enter in a condition when you use the `IN LIST` operator. 99: Specifies that you may enter up to 99 values when you create a condition using the `IN LIST` operator. The maximum authorized value you may enter depends on your database. The value of -1 means that there is no restriction on the number of values returned, except that imposed by the database.

16.4.1.17 MAX_PARALLEL_QUERIES_FOR_MSU_UNIVERSE

`MAX_PARALLEL_QUERIES_FOR_MSU_UNIVERSE` = [1-64]

Values	Integer: min 1, max 64
Default	4
Description	Indicates the maximum number of queries that can be executed for this multiple source data foundation concurrently when a document with multiple data providers is refreshed. This applies to the capability of a SAP BusinessObjects Web Intelligence document to perform parallel data refresh jobs in reports based on multiple data providers. . If this parameter is not set in the universe, then the default value will be 4. If this parameter is set to 0, then parallel queries are disactivated in SAP BusinessObjects Web Intelligence for this multi-source universe

16.4.1.18 NO_NULL_YIELDS_IN_SUBQUERY

`NO_NULL_YIELDS_IN_SUBQUERY` = Yes|No

Values	`Yes/No`
Default	`No`
Description	Determines whether SQL scripts can contain non NULL values for filters based on a subquery. `No`: SQL scripts are generated without controlling if fields have or have not NULL values for filters based on a subquery. `Yes`SQL scripts are generated ensuring fields with non NULL values are included for filters based on a subquery.

16.4.1.19 PREVENT_QUERY_WITHOUT_MEASURE

`PREVENT_QUERY_WITHOUT_MEASURE` = Yes|No

Values	`Yes/No`
Default	`No`

Description	Determines if a query that does not include a measure as a result object, can be created and executed on the universe. `Yes`: An error is returned if the query does not contain a measure. `No`: Default value. There is no restriction on queries that don't contain measures.

16.4.1.20 REPLACE_COMMA_BY_CONCAT

`REPLACE_COMMA_BY_CONCAT`= Yes|No

Values	`Yes` | `No`
Default	`No`
Description	In previous versions of the universe design tool, a comma could be used to separate multiple fields in an object Select statement. The comma was treated as a concatenation operator. For universes that already use the comma in this way you can set `REPLACE_COMMA_BY_CONCAT` to `No` to keep this behavior. In the current version of the universe design tool, this parameter is set to `Yes` by default, so that any expressions using a comma in this way are automatically changed to use concatenation syntax. `Yes`: Comma is replaced by the concatenation expression when multi field object is found. `No`: Keep the comma as it is.

16.4.1.21 SELFJOINS_IN_WHERE

`SELFJOINS_IN_WHERE` = Yes|No

Values	`Yes` | `No`
Default	`No`

Description	Self-joins are usually included in the FROM clause. This allows you to force the system to generate SQL syntax with all the conditions of a self-join in the `WHERE` clause. The `ANSI92` parameter must be set to `Yes` for this parameter to be taken into account. You must manually add the parameter to the list to activate it. `Yes`: The conditions of a self-join go in the `WHERE` clause of the SQL query. `No`: The syntax for self-joins is generated according to the ANSI 92 convention, and conditions for a self-join go in the `ON` clause of the table join definition in the `FROM` clause of the SQL query.

16.4.1.22 SHORTCUT_BEHAVIOR

`SHORTCUT_BEHAVIOR` = ShortestPath|Global|Successive

Values	`ShortestPath|Global|Successive`
Default	`ShortestPath`
Description	Specifies how shortcut joins are applied. `ShortestPath`: applies shortcuts so as to obtain the smallest number of tables in the query. `Successive`: applies shortcuts one after the other. If a shortcut removes a table involved in a potential successive shortcut, the successive shortcut is not applied. `Global`: applies all shortcuts. If the resulting query creates a Cartesian product, no shortcut joins are applied. **i Note** This parameter was formerly listed as `GLOBAL_SHORTCUTS` in the `PRM` files. The value `Global` corresponds to `Yes`, and `Successive` corresponds to `No`.

16.4.1.23 SMART_AGGREGATE

`SMART_AGGREGATE` = Yes|No

Values	`Yes|No`
Default	`No`

Description	Determines how aggregate tables are used for smart measures that are based on an aggregate tables. This ensures that a universe object based on a ratio is correctly aggregated. By default the system takes the advantage of the pre-calculated values from the aggregated tables, if these table are not consistent during time (different time periods), you use this parameter to ensure the most detailed aggregate tables are used. This parameter is not visible in the universe parameter list (by default not activated). The universe designer must manually insert it in the parameter list before activating it (value `Yes`). `Yes`: Any additional grouping set query should be based on the aggregate table of the initial query for the smart measure based on aggregate table. `No`: The system takes the most appropriate aggregate table.

16.4.1.24 THROUGH_AGGREGATE_AWARE

`THROUGH_AGGREGATE_AWARE = Yes|No|Optimized`

Values	`Yes/No|Optimized`
Default	`Optimized` **i Note** For universes converted from .unv, the default value is `No`.
Description	Determines if aggregate awareness is taken into account when testing the compatibility of query objects. This parameter allows you to possibly improve the results of queries on converted .unv universes that fail for the .unv universe. `Yes`: Aggregate awareness is taken into account when testing the compatibility of objects in the query. In some cases this allows the query to succeed in situations where there are incompatible objects (split queries) with aggregate aware objects. `No`: Compatibility testing of objects uses the behavior for .unv universes. `Optimized`: Compatibility testing of objects uses the behavior for .unv universes. If it fails, a new compatibility testing is automatically started in taking into account the aggregate awareness.

16.4.1.25 THOROUGH_PARSE

`THOROUGH_PARSE` = Yes|No

Values	`Yes` | `No`
Default	`No`
Description	Specifies the methodology used for default Parsing in the Query pane and individual object parsing. Yes: `PREPARE, DESCRIBE`, and `EXECUTE` statements are used to parse SQL for objects. `Prepare+DescribeCol+Execute` `No: PREPARE` and `DESCRIBE` statements are used to parse SQL for objects.

16.4.1.26 TRUST_CARDINALITIES

`TRUST_CARDINALITIES` = Yes|No

Values	`Yes` | `No`
Default	`No`
Description	Allows you to optimize the SQL in case of inflated results. `Yes`: For queries that include a measure, all conditions that inflate the measure and do not appear in the Result Objects, are transformed to sub queries to ensure that tables that may return false results for the measure are not included in the query. `No`: No optimization is implemented.

16.4.1.27 UNICODE_STRINGS

`UNICODE_STRINGS` = Yes|No

Values	`Yes` | `No`
Default	`No`

Description	Specifies whether the current universe can manipulate Unicode strings or not. Only applies to Microsoft SQL Server and Oracle 9. If the database character set in the `SBO` file is set as Unicode, then it is necessary to modify the SQL generation to handle specific Unicode column types like `NCHAR` and `NVARCHAR`. `Yes`: Conditions based on strings are formatted in the SQL according to the value for a parameter `UNICODE_PATTERN` in the `PRM` file, for example for Microsoft SQL Server `(sqlsrv.prm) :UNICODE_PATTERN=N$` The condition Customer_name='Arai ' becomes Customer_name=N'Arai'. Note: When you create a prompt with `@Prompt` syntax based on Unicode value, the data type should be 'U' not 'C' `No`: All conditions based on strings are formatted in the standard SQL. For example the condition Customer_name='Arai ' remains Customer_name='Arai'

16.4.1.28 USE_ENHANCED_QUERY_STRIPPING

`USE_ENHANCED_QUERY_STRIPPING` = Yes|No

Values	`Yes|No`
Default	`No`
Description	Specifies the query stripping mode for relational universes. If set to Yes, the system optimizes only the SELECT and GROUP BY clauses to avoid fetching unused data but does not modify the other clauses to respect the original query semantic. If set to No or not set, the system generates optimized queries by completely ignoring the stripped objects with their corresponding tables and joins. **i Note** If aggregate awareness is defined in the business layer (using the @Aggregate_aware function in the definition of business layer objects), enhanced query stripping is used regardless of the value of USE_ENHANCED_QUERY_STRIPPING. For more information, see Query stripping [page 220].

16.4.2 SQL generation parameters set in the extended PRM

The following reference describes the SQL generation parameters that you set in the extended data access parameter (PRM) file for the target data access driver. Extended PRM files are located in the the following directory, where `<RDBMS>` is the network layer or middleware name:

```
<BIP_INSTALL_DIR>\SAP BusinessObjects Enterprise XI 4.0\dataAccess\connectionServer
\<RDBMS>\extensions\qt
```

For more information about PRM files, see the *Data Access Guide*.

Related Information

CASE_SENSITIVE [page 515]
COMMA [page 515]
CONCAT [page 516]
DELIMIT_IDENTIFIERS [page 516]
DELIMIT_LOWERCASE [page 516]
EXTERN_SORT_EXCLUDE_DISTINCT [page 517]
GROUPBY_WITH_ALIAS [page 517]
IDENTIFIER_DELIMITER [page 517]
OUTERJOINS_GENERATION [page 518]
OVER_CLAUSE [page 520]
OWNER [page 520]
QUALIFIER [page 520]
UNICODE_PATTERN [page 521]
USER_INPUT_DATE_FORMAT [page 521]
USER_INPUT_NUMERIC_SEPARATOR [page 522]

16.4.2.1 CASE_SENSITIVE

```
<Parameter Name="CASE_SENSITIVE">NO</Parameter>
```

Description	Specifies if the database is case-sensitive. This parameter is used with Oracle.
Values	`YES`: the database is case-sensitive. `NO`: the database is not case-sensitive.
Default	`NO`

16.4.2.2 COMMA

```
<Parameter Name="COMMA">||' '||</Parameter>
```

Description	Specifies what database concatenation operator should be used to replace a comma for objects that have the following syntax: Tab.Col1, Tab.Col2. This parameter is used with all data access drivers.
Values	`||' '||` `+' '+`
Default	`||' '||`
Result	Tab.Col1||' '||Tab.Col2

16.4.2.3 CONCAT

```
<Parameter Name="CONCAT">||</Parameter>
```

Description	Specifies the concatenation operator. The parameter is used with all data access drivers.
Values	double pipe (`||`) or plus sign (+)
Default	`||`

16.4.2.4 DELIMIT_IDENTIFIERS

```
<Parameter Name="DELIMIT_IDENTIFIERS">YES</Parameter>
```

Description	Specifies if database identifiers can be quoted. Identifiers are quoted using the delimiter specified in the `IDENTIFIER_DELIMITER` parameter.
Values	`YES`: identifiers can be quoted. `NO`: identifiers cannot be quoted.
Default	`YES`
Result	Table name="my_table"

16.4.2.5 DELIMIT_LOWERCASE

```
<Parameter Name="DELIMIT_LOWERCASE"></Parameter>
```

Description	Specifies if lowercase identifiers are delimited with quotes.
Values	`YES`: the lowercase identifiers are delimited with quotes. `NO`: the lowercase identifiers are not delimited with quotes.

16.4.2.6 EXTERN_SORT_EXCLUDE_DISTINCT

```
<Parameter Name="EXTERN_SORT_EXCLUDE_DISTINCT">YES</Parameter>
```

Description	Specifies if the application generates a SELECT DISTINCT when a query contains an ORDER BY clause.
Values	`YES`: a SELECT DISTINCT is not generated when the query contains an ORDER BY clause. `NO`: a DISTINCT is generated when the query contains an ORDER BY clause.
Default	`YES`

16.4.2.7 GROUPBY_WITH_ALIAS

```
<Parameter Name="GROUPBY_WITH_ALIAS">YES</Parameter>
```

Description	Specifies if the database can create a GROUP BY clause that contains aliases in the SELECT statement.
Values	`YES`: it allows you to create a GROUP BY clause with aliases in the SELECT statement. `NO`: it does not let you create a GROUP BY clause with aliases in the SELECT statement.
Default	`YES`

16.4.2.8 IDENTIFIER_DELIMITER

```
<Parameter Name="IDENTIFIER_DELIMITER">"</Parameter>
```

Description	Specifies the following features: • Table or column names that contain spaces or special characters are enclosed within quotes if `BACK_QUOTE_SUPPORTED` parameter is activated. • Tables or column names regardless of their characters are enclosed within quotes if `DELIMIT_IDENTIFIERS` parameter is activated. To use this parameter, either `BACK_QUOTE_SUPPORTED` or `DELIMIT_IDENTIFIERS` must be set to `YES`. This is the default value of both parameters.
Values	`"` (double quote): table or column names that contain spaces or special characters are enclosed in double quotes. `'` (single quote): table or column names that contain spaces or special characters are enclosed in single quotes. This value can only be used with Microsoft Access.
Default	`"`
Result	Table name="My Table"

16.4.2.9 OUTERJOINS_GENERATION

```
<Parameter Name="OUTERJOINS_GENERATION">ANSI92</Parameter>
```

Description	Specifies the SQL syntax for outer joins. The value `ANSI 92` generates an outer join in the FROM clause. Other values generate the outer join in the WHERE clause. When you modify this setting, you should check join properties to verify that the outer join expression is valid, and that the cardinalities are correct. ANSI92 does not support any manual customization in the join syntax. **i Note** The PRM file `OUTERJOINS_GENERATION` parameter relates to the universe `ANSI92` setting in the following way: • If the PRM file `OUTERJOINS_GENERATION` parameter is set to `ANSI_92` and the universe `ANSI92` parameter is set to `NO`, the PRM parameter overrides the universe setting and outer joins conform to ANSI92 behavior. • If the PRM file `OUTERJOINS_GENERATION` parameter is set to `USUAL`, then the universe `ANSI92` setting takes precedence, and outer joins conform to ANSI92 depending on whether the universe `ANSI92` setting is `YES` or `NO`.

Values	The primary values for `OUTERJOINS_GENERATION` are: • `ANSI_92`: the default outer join behavior conforms to the ANSI92 standard regardless of the `ANSI92` parameter value in the universe. • `NO`: outer joins are not supported. • `USUAL`: the database-specific outer join behavior is used. This behavior is overridden if the `ANSI92` parameter is set to `YES`. Other settings are available depending on the database. See the defaults below.
Default	`ANSI_92`: default value for Oracle, Microsoft SQL Server 2005 and Sybase. `DB2`: default value for IBM DB2. `FULL_ODBC`: default value for Microsoft SQL Server. `INFORMIX`: default value for IBM Informix. `INGRES`: default value for Teradata. `NO`: default value for ODBC. `USUAL`: default value for HP Neoview, Netezza, IBM Red Brick and MS SQL Server 2000.

Examples of OUTERJOINS_GENERATION parameter settings

Setting = `USUAL`:

```
FROM T1, T2
WHERE T1.col1(+) = T2.col2
```

Setting = `DB2`:

```
FROM T2 LEFT OUTER JOIN T1
  ON T1.col1 = T2.col2
```

Setting = `ODBC`:

```
FROM {oj T1 LEFT OUTER JOIN T2 ON T1.col1=T2.col2}
 Where (T2.col3 = T3.col1)
```

Setting = `INFORMIX`

```
FROM T2
OUTER T1
WHERE T1.col1=T2.col2
```

Setting = `FULL-ODBC`

```
FROM {oj T1 RIGHT OUTER JOIN T2 ON T2.col2=T1.col1
 T2 INNER JOIN 3 on T2.col3 = T3.col1}
```

Setting = ANSI_92:

```
SELECT DISTINCT
  t1.col1,
  t2.col2
FROM
  (t1 RIGHT OUTER JOIN t2 ON (t1.col1=t2.col2) )
```

16.4.2.10 OVER_CLAUSE

```
<Parameter Name="OVER_CLAUSE">YES</Parameter>
```

Description	Allows SAP BusinessObjects applications to include RISQL functions when generating SQL. The supported RISQL functions for the database are listed in the ANALYTIC_FUNCTIONS parameter.
Values	YES: applications can include RISQL functions when generating SQL. NO: applications cannot include RISQL functions when generating SQL.
Default	YES

16.4.2.11 OWNER

```
<Parameter Name="OWNER">YES</Parameter>
```

Description	Specifies if the database supports the owner name as prefix for tables.
Values	YES: the database supports prefixing tables with the owner name. NO: the database does not support prefixing tables with the owner name.
Default	YES

16.4.2.12 QUALIFIER

```
<Parameter Name="QUALIFIER">NO</Parameter>
```

Description	Specifies if the database supports the qualifier name as prefix for tables.

Values	`YES`: the database supports prefixing tables with the qualifier name. `NO`: the database does not support prefixing tables with the qualifier name.
Default	RDBMS-dependent.

16.4.2.13 UNICODE_PATTERN

```
<Parameter Name="UNICODE_PATTERN">UNISTR($)</Parameter>
```

Description	Only applies when the universe SQL generation parameter `UNICODE_STRINGS` is set to `YES`. All conditions based on strings are then formatted with this string value. This is used with Microsoft SQL Server and Oracle only.
Values	`N$`: for Microsoft SQL Server `UNISTR($)`: for Oracle

16.4.2.14 USER_INPUT_DATE_FORMAT

```
<Parameter Name="USER_INPUT_DATE_FORMAT">'dd-MM-yyyy HH:mm:ss'</Parameter>
```

Description	Specifies the default date and hour formats generated in the WHERE clause of a SQL statement.
Values	`{\d 'yyyy-mm-dd'}`: default date format with ODBC. `'DD-MM-YYYY HH:MM:SS'`: default date and hour formats with Oracle. `'MM/DD/YYYY'`: default date format with IBM Informix. `'yyyy-mm-dd HH:mm:ss'`: default date and hour formats with Microsoft SQL Server and for most IBM DB2 servers. `'mm/dd/yyyy hh:m:s am/pm'`: default date and hour formats with Sybase. `'yyyy-mm-dd'`: default date format with a Sybase gateway. **i Note** If you need to use time or timestamp variables with ODBC, you must replace the default date format value with `{\t 'hh:mm:ss'}` or `{\t\s 'yyyy-mm-dd hh:mm:ss'}` in the `odbc.sbo` file.
Default	See values above.

16.4.2.15 USER_INPUT_NUMERIC_SEPARATOR

```
<Parameter Name="USER_INPUT_NUMERIC_SEPARATOR">.</Parameter>
```

Description	Specifies the default decimal separator that is used in the generated SQL script.
Values	'.' (period)
Default	'.'

16.5 Formula language for prompts

You can set a formula expression for a universe prompt parameter to return dynamic default values for prompts at runtime. The formula expression can be implemented either directly in an @Prompt, or in the definition of a parameter for an object in the business layer or data foundation. This section describes the syntax available for the formula language you use to create dynamic default prompts.

Related Information

Setting dynamic default values for prompts by editing a parameter definition [page 522]
Setting dynamic default values directly in a universe @prompt [page 523]
Basic Elements for Prompt Formula Language [page 524]
Operator Expressions for Prompt Formula Language [page 527]
Function Expressions for Prompt Formula Language [page 530]
If then Else Expression for Prompt Formula Language [page 535]

16.5.1 Setting dynamic default values for prompts by editing a parameter definition

You can implement the formula for dynamic default values using the formula editor available from the Parameters and Lists of Values tab in the browing pane of the *Business Layer Editor*.

You can define dynamic default values for prompts, based on number, string, and date/time functions for the current year or period, for example:

- CurrentDate()
- DatesBetween(date1, date2)
- ToNumber (FormatDate (CurrentDate ();"yyyymm"))

- ToNumber (FormatDate (CurrentDate ();"yyyy"))

These dynamic default values are then used by SAP Web Intelligence. You implement, validate, and refresh dynamic prompt values as follows:

Formula language for dynamic prompts	Description
Implemention	You enable the Parameter options *Set Default Values* and *Formula* to set a formula expression for the universe prompt parameter to get dynamic default values for prompts at runtime.
Validation	When you set a formula expression for dynamic default values for a prompt, the information design tool automatically validates the expression before saving it for the prompt. If the formula expression validation fails, an error message is returned, and the dynamic expression will not be committed. When you edit a formula expression for dynamic default values for a prompt, you can check it on demand. The information design tool validates a formula expression before saving it to the prompt.
Refresh	Refreshing data with dynamic default values for prompts Once dynamic default values on universe prompts have been defined in the information design tool, there are two ways to consume them at the runtime in SAP Web Intelligence, and other SAP Business Intelligence applications. • Users create a query with business objects in which prompts with dynamic default values are inserted at the universe level. • Users create a query with query filters that use universe prompt parameters with dynamic default values

Related Information

Inserting and editing a parameter [page 239]

16.5.2 Setting dynamic default values directly in a universe @prompt

In the information design tool, you can also set dynamic default values from the universe @prompt syntax:

```
@Prompt('<question>', 'A'|'N'|'D'|'K', '[lov]', mono|multi, free|constrained,
persistent|not_persistent, '{default_values}', user:<position>, optional,
'<formula_expression>')
```

- In the <formula_expression>, you escape quote (') with double quote ('').
- The formula for prompt default values is valid if the prompt constraint option was “Free” or “Constrained”. If “Primary_Key” is used, an error is returned: Error: Invalid Prompt Definition. For example, users could

enter the following expression: `@Prompt('Select the min date:', 'D', , mono, free, not_persistent, , user:0,, 'currentDate() - 7')`

16.5.3 Basic Elements for Prompt Formula Language

This section contains descriptions for the basic elements available to build a formula expression for dynamic prompts. The following elements are covered:

- Data types
- Constants for first day of the week
- Time period values
- Literal expressions
- Comment expressions

Related Information

Data Types [page 524]
Constants for First Day of the Week [page 525]
Time Period Values [page 525]
Literal Expressions [page 526]
Comment Expressions [page 527]

16.5.3.1 Data Types

The following data types are supported:

- STRING: String and Long Text data.
- NUMBER: Integer, Long, Double, Big integer, and Big decimal.
- DATE: Only Calendar Date part without a time part.
- DATETIME: Both Date and Time data without time zone.
- BOOLEAN: Only used in filter condition expressions.
- ANY: Unspecified data type can be a BOOLEAN, STRING, NUMBER, DATE or DATETIME type.

Invariant keywords:

Name	Syntax	Description
+ - * /	+ - * /	Arithmetic operators
= < <= > >=	= < <= > >=	Comparison operators

Name	Syntax	Description
( )	( )	Open and close parenthesis
DOUBLE_QUOTE	"	Quote: Delimiter for String literal
SHARP	#	Delimiter for Date/Datetime literal
-	-	Date separator
:	:	Time separator
DOT	.	Decimal separator Note: In the formula language for parameters, the decimal separator does not depend on the locale setting: it is always the DOT (.) as the decimal separator.
SEMICOLON	;	list/argument separator

16.5.3.2 Constants for First Day of the Week

The following are the supported constants for the first day of the week, used in functions such as Weekday and DatesBetween:

- FRIDAY
- MONDAY
- SATURDAY
- SUNDAY
- THURSDAY
- TUESDAY
- WEDNESDAY

16.5.3.3 Time Period Values

The following are the supported time period values used in functions such as RelativeDate and DatesBetween:

Time period value	Description
YearPeriod	Time period in years
QuarterPeriod	Time period in quarters
MonthPeriod	Time period in months

Time period value	Description
SemesterPeriod	Time period in semesters
DayPeriod	Time period in days
WeekdayPeriod	Time period in week days (result depending on firstDayOfWeek)
WeekPeriod	Time period in weeks
HourPeriod	Time period in hours
MinutePeriod	Time period in minutes
SecondPeriod	Time period in seconds

16.5.3.4 Literal Expressions

Literals	Description
Text	• "France" • "French are \"winners\"" where \ is the escape character.
Numeric	The number format doesn't depend on the user locale settings. The DOT(.) is fixed as the decimal separator. Expression examples: • 123 • 123.45 where . as the decimal separator • -123.45 where – is the negative sign • 1.6E3 where E is the exponent symbol
Date/DateTime	The Date/Datetime format doesn't depend on the user locale settings: • For the date literal, the format is fixed as #yyyy-MM-dd# • For the date time literal, the format is fixed as #yyyy-MM-dd hh:mm:ss# Expression examples: • #1999-10-23# • #1999-10-23 10:21:98#

16.5.3.5 Comment Expressions

Comment	Syntax
/*...*/ (Comment)	/* text_of_comment */
// (Comment)	// text_of_comment

16.5.4 Operator Expressions for Prompt Formula Language

This section contains descriptions for operators available to build a formula expression for dynamic prompts. The following operator expressions are covered:

- Operator rules
- Arithmetic functions
- Comparison operators
- Wildcard characters for match patterns
- Logical operators

Related Information

Operator Rules [page 527]
Arithmetic Functions [page 528]
Comparison Operators [page 528]
Wildcard Characters for Match Patterns [page 529]
Logical Operators [page 529]

16.5.4.1 Operator Rules

The rules for using unary and binary operators are the following:

Operator	Rules for use
Unary	operator operand This operator can have only one numeric operand: • + numeric_expression • - numeric_expression
Binary	expression operator expression This operator defines the way two expressions are combined to yield a single result.

Operator Precedence:

In the following expression, multiplication has a higher precedence than addition, so the system first multiplies 2 by 3 and then adds the result to 1.

1+2*3

16.5.4.2 Arithmetic Functions

Operator	Return Type	Syntax
ADD	Number	numeric_expression + numeric_expression
SUBTRACT	Number	numeric_expression - numeric_expression
MULTIPLY	Number	numeric_expression * numeric_expression
DIVIDE	Number	numeric_expression / numeric_expression

Expression examples:

- 1 + 2 - 3 * 4 / 5
- (Round(1.2) + Abs(-2.3)) / (3.4 * -0.5)
- "price = " + "price " + " * " + FormatNumber(Sqrt(4))
- CurrentDate() + 2 (□ RelativeDate(CurrentDate();2)

16.5.4.3 Comparison Operators

Operator	Return Type	Syntax
EQUAL	Boolean	expression = expression

Operator	Return Type	Syntax
NOT_EQUAL	Boolean	expression != expression
LESS	Boolean	expression < expression
LESS_EQUAL	Boolean	expression <= expression
GREATER	Boolean	expression > expression
GREATER_EQUAL	Boolean	expression >= expression
INLIST	Boolean	expression InList (expression [;...; expression])
BETWEEN	Boolean	expression Between (expression1; expression2)
MATCH	Boolean	Match (expression; pattern_expression)

Expression examples:

- CurrentUser() ="myname"
- CurrentUser() !="myname"
- Sqrt(4) >= 1000
- CurrentDate() = #1999-10-27#
- "France" InList ("Canada"; "France")
- "Belgium" Between ("Canada"; "France")
- Match ("France"; "Fr*")

16.5.4.4 Wildcard Characters for Match Patterns

Wildcard characters	Description
*	To match zero or more characters (for Match function).
?	To match a single character (for Match function).

16.5.4.5 Logical Operators

Operator	Return Type	Syntax
AND	Boolean	boolean_expression AND boolean_expression

Operator	Return Type	Syntax
OR	Boolean	boolean_expression OR boolean_expression

Expression examples:

- Match ("France"; "Fr*") AND CurrentUser() ="otsoungu"
- "France" InList ("Canada"; "France") OR CurrentDate() = #1999-10-27#

16.5.5 Function Expressions for Prompt Formula Language

This section contains descriptions for functions available to build a formula expression for dynamic prompts. The following types of functions are covered:

- Numeric
- Text
- Date Time
- Conversion
- Logical

Related Information

Numeric Functions [page 530]
Text Functions [page 531]
Date Time Functions [page 532]
Conversion Functions [page 533]
Logical Functions [page 534]

16.5.5.1 Numeric Functions

Function	Return Type	Syntax
ABS	Number	Abs (numeric_expression)
CEIL	Number	Ceil (numeric_expression)
COS	Number	Cos (numeric_expression)
EXP	Number	Exp (numeric_expression)

Function	Return Type	Syntax
FLOOR	Number	Floor (numeric_expression)
LN	Number	Ln (numeric_expression)
LOG	Number	Log (numeric_expression; integer_expression)
LOG10	Number	Number Log10 (numeric_expression)
MOD	Number	Mod (numeric_expression; numeric_expression)
POWER	Number	Number Power (numeric_expression; integer_expression)
ROUND	Number	Number Round (numeric_expression; interger_expression)
SIGN	Number	Number Sign (numeric_expression)
SIN	Number	Sin (numeric_expression)
SQRT	Number	Sqrt (numeric_expression)
TAN	Number	Tan (numeric_expression)

Expression examples:

- Cos(2)
- Power(4;2)
- Log(Cos(2))

16.5.5.2 Text Functions

Function	Return Type	Syntax
CONCAT	String	Concatenation (string_expression ; string_expression
LEFT	String	Left (string_expression ; integer_expression)
LOWER	String	Lower (string_expression)
RIGHT	String	Right (string_expression ; integer_expression)
TRIM	String	Trim (string_expression)
RTRIM	String	RightTrim (string_expression)

Function	Return Type	Syntax
LTRIM	String	LeftTrim (string_expression)
UPPER	String	Upper (string_expression)
SUBSTR	String	Substr (string_expression ;start_position [; length])
LENGTH	String	Length (string_expression)
INSTR	String	Instr (string_expression; substring [; start_position])
REPLACE	String	Replace(string_expression; find ; replace)

Expression examples:

- Upper("France")
- LeftTrim(" France ")
- If Instr("La France is beautiful", Upper("France")) > 0 Then ... Else ...
- Concatenation(CurrentUser()," France")
- Concatenation(CurrentUser(); Concatenation(" " ; "France"))

16.5.5.3 Date Time Functions

Function	Return Type	Syntax
CURDATE	Date	Date CurrentDate()
CURDATETIME	Datetime	CurrentDatetime()
DAY	Number	DayNumberOfMonth (date | datetime)
MONTH	Number	MonthNumberOfYear (date | datetime)
QUARTER	Number	Quarter (date | datetime)
SEMESTER	Number	Semester (date | datetime)
YEAR	Number	Year (date | datetime)
HOUR	Number	Hour (date | datetime)
MINUTE	Number	Minute (date | datetime)
SECOND	Number	Second (date | datetime)
WEEKNUM	Number	Week (date | datetime)

Function	Return Type	Syntax
WEEKDAY	Number	DayNumberOfWeek (date | datetime [; firstDayOfWeek])
DATE	Date	Date (year ;month ; day)
DATETIME	Datetime	Datetime (year ;month ; day; hour ; minute ; second)
RELATIVEDATE	Date	RelativeDate (date | datetime; amount [; timePeriod])
DATEDIFF	Number	DatesBetween (date | datetime ; date | datetime [; timePeriod [; firstDayOf-Week]])

Expression examples:

- Date(2013;10;19)
- Year(Date(2013;10;19))
- Quarter(Date(2013;10;19))
- Hour(Datetime(2013;10;19;10;20;34))
- RelativeDate(#2013-09-19#; 2; DayPeriod)
- DatesBetween(#2010-3-31#;#2010-5-1#; WeekPeriod; Thursday)

16.5.5.4 Conversion Functions

Function	Return Type	Syntax
FORMATDATE	String	FormatDate (date_expression | datetime_expression [; format])
FORMATNUMBER	String	FormatNumber (numeric_expression [; format])
TONUMBER	Numeric	ToNumber (string_expression)
TODATE	Date	ToDate (string_expression [; format])

Expression examples:

- FormatNumber(23)
- FormatDate(Date(23,10,1999); 'dd/MM/yy')
- ToNumber("12")
- ToDate("27/2/99";"dd/MM/yy")
- ToDate("27/2/99 12:00:00";"dd/MM/yy hh:mm:ss")

16.5.5.5 Logical Functions

Function	Return Type	Syntax
NOT	Boolean	Not (boolean_expression)
ISNULL	Boolean	IsNull (expression)
ISERROR	Boolean	IsError (expression)

Expression example:

Not("Belgium" Between ("Canada"; "France"))

16.5.6 System Functions for Prompt Formula Language

Function	Return Type	Syntax
CURUSER	String	CurrentUser()
DBUSER	String	DatabaseUser()
DOCID	String	DocumentIdentifier()
DOCNAME	String	DocumentName()
DOMINANTPVL	String	GetDominantPreferredViewingLocale()
DPNAME	String	DataProvider()
DPTYPE	String	DataProviderType()
PVL	String	GetPreferredViewingLocale()
UNVID	String	UniverseIdentifier()
UNVNAME	String	UniverseName()
USERATTRIBUTE	String	UserAttribute(attributeId)

Expression examples:

- CurrentUser()
- DatabaseUser()
- UniverseName()
- UserAttribute("SI_MYCOUNTRY")

16.5.7 If then Else Expression for Prompt Formula Language

Function	Return Value	Syntax
IF-THEN-ELSE	input_type	IF boolean_expression expression ELSE expression

Expression examples:

Example
If (Day(CurrentDate()) <= 4) Then CurrentDate() Else RelativeDate(currentDate();2)

If (GetPreferredViewingLocale() = 'fr_FR') Then "France"
Else If (GetPreferredViewingLocale() = 'en_US') Then "United States"
Else
"Unknown locale"

17 Reporting and running queries in the BI repository

As an administrator, you need to understand and optimize the usage of the Business Intelligence platform. The CMS reporting sample kit includes the CMS database driver which allows you to visualize and report on the metadata objects of the CMS database. You can now use a universe and native reporting clients to query the metadata objects of the CMS repository database. These metadata objects include Business Intelligence platform information, such as:

- Connections
- Documents
- Schedules
- Universes
- Users

You can import the CMS reporting sample that contains predefined objects to help you create reports and dashboards using the following SAP BusinessObjects data analysis and reporting applications :

- SAP BusinessObjects Web Intelligence
- SAP Crystal Reports for Enterprise

> **i Note**
>
> The information design tool now supports the use of a cloned CMS.

Related Information

Overview of CMS reporting [page 536]

17.1 Overview of CMS reporting

For a quick and easy way to start reporting on the CMS, you can work with the CMS reporting sample kit.

Here are the main stages to create a CMS report:

- Import the CMS reporting sample: You use Promotion Management in the CMC to import the CMS reporting sample.
- Create a CMS report: With SAP BusinessObjects Web Intelligence, you can create a CMS report using the CMS sample universe as a data source.

See Related Information for an end to end procedure that gives a more detailed overview of the creation process.

Related Information

CMS reporting sample kit [page 537]
Creating a CMS report [page 540]

17.1.1 CMS reporting sample kit

You need to use the CMS reporting sample kit to start building documents for CMS reporting. The CMS database driver is integrated in the Business Intelligence platform and the CMS reporting sample can be found in the following location:

```
C:\Program Files\SAP BusinessObjects\SAP BusinessObjects Enterprise XI 4.0\Samples
\BI on BI.
```

This sample includes the following:

- Connection (BI platform CMS system database.cns)
- Universe (BI platform CMS system database.unx)
- Web Intelligence samples

Find more information about CMS reporting on the SAP Community network.

17.1.1.1 Importing the CMS reporting sample kit with Promotion Management

Before you begin, make sure that you have access to the CMS reporting sample which is in the following location:

```
C:\Program Files\SAP BusinessObjects\SAP BusinessObjects Enterprise XI 4.0\Samples
\BI on BI
```

You use the Promotion Management tool in the Central Management Console (CMC) to import the CMS reporting sample.

1. In the Central Management Console, click *Promotion Management*.
2. Click *Import* > *Import File*.
3. Select *File System*
4. Click *Choose file* to select the sample.
5. In the *New Job* pane, select *Login to a New CMS* for the *Destination* field.
6. Enter the login parameters then click *Login* > *Create*.
7. In the *Promotion Jobs* pane, right-click the sample then select *Promote*.
8. In the *Promote* dialog box, click *Promote*.

When the *Promotion Status* of the CMS reporting sample is *Success*, you have successfully imported the sample into your Business Intelligence 4.2 system. To use the sample universe for CMS reporting, see the related topic.

Related Information

CMS reporting sample kit [page 537]

17.1.1.2 The CMS sample universe

The CMS sample universe includes a predefined universe that supports common reporting scenarios. According to your analysis and reporting needs, you can edit and enhance the predefined universe. You can also find a list of predefined queries in the *Queries* pane. These queries can serve as a tutorial for the universe features.

Some of the most useful queries and what they mean are listed in the table.

Useful queries to run on the CMS universe

Query	Description
Sample-User-Relationship-Detail	Allows you to see in which group a user belongs to.
Sample-FolderPath (Universe)	Allows you to find the location of an universe.
Sample-ScheduleInfo-Relationships	Allows you to visualize the actions scheduled by the users.
Sample-QT-Properties with Filter (Server)	Allows you to visualize the properties of an InfoObject.

17.1.1.3 Extending the CMS sample universe

You can create a linked universe to expand the CMS sample universe. A linked universe is a .UNX universe that contains a link to a designated core universe in the CMS. For more information on creating a linked universe, see the related topic.

In this case, the CMS sample universe acts as a core universe so the linked universe can use the CMS sample universe's data foundation and business layer as pre-fabricated building blocks. Once you have created the linked universe, you can save its data foundation and business layer inherited from the CMS sample universe as new files so they have a life cycle independent of the CMS sample universe.

You can use the CMS database connection of the CMS sample universe or another connection compatible with the CMS database.

You can add tables, create joins linking the core data foundation tables with the new ones, and add new components to the business layer in the same way that you do for any other universe. Any changes in the core components are automatically propagated to the linked universe when it is checked into the CMS. For more information on working with the data foundation of a linked universe, see the related topic.

How to expand the CMS sample universe

To expand the CMS sample universe, you can	Go here for more information
Create a linked universe	Creating a new linked universe [page 246]
Insert alias tables	Inserting alias tables [page 166]
Insert a join	Inserting and editing a join [page 176]
Work with the data foundation of a linked universe	Working with the data foundation of a linked universe [page 252]

17.2 CMS database connection

You use a CMS database driver to create a secure connection to the CMS database. You can either use the default connection available in the CMS reporting sample or you can create your own CMS database connection.

For the CMS database connection, you need to use a relational connection. The following table describes the parameters of a relational connection.

Parameters for a relational connection

Parameter	Description
Authentication Mode	The method used to authenticate the user's login credentials when accessing the data source: • *Use specified user name and password*: Uses the *User Name* and *Password* parameters defined for the connection. You can access the data source from either an on premise system or a distant system. **i Note** Make sure that the user has the rights to see the content of this session. • *Use Session Token*: Uses the current user session. You can only see the content you are allowed to see and work with. You can only access the data source from an on premise system. **i Note** For security matters, this authentication mode is the recommended choice.
System ID	The name of the CMS

Parameter	Description
User Name	The user name to access the data source if *Authentication Mode* is *Use specified user name and password*.
Password	The password to access the data source if *Authentication Mode* is *Use specified user name and password*.

17.3 Creating a CMS report

With SAP BusinessObjects Web Intelligence, you can create a CMS report using the CMS sample universe as a data source.

1. Open Web Intelligence and click the *New* icon in the *File* toolbar.
2. Select the CMS sample universe.

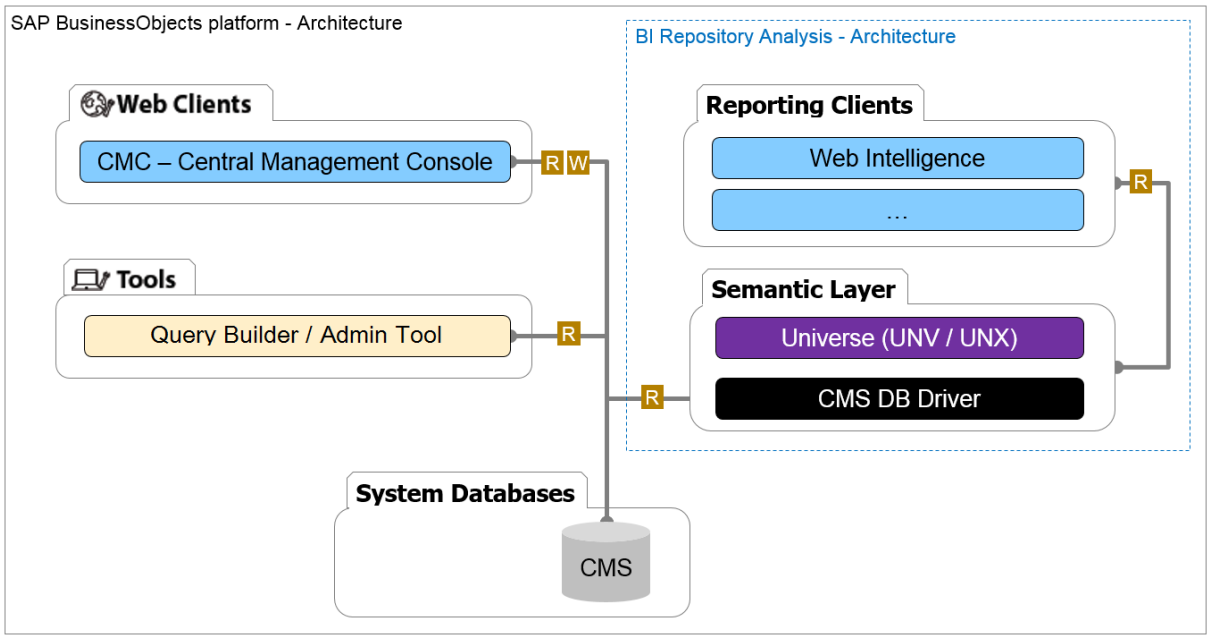
   If you are using the Web Intelligence Rich Client, click *Select*.

   The *Query Panel* opens.
3. Select and drag dimensions and measures that you want to include in the query into the *Result Objects* pane.
4. Select the objects on which you want to define query filters and drag them to the *Query Filters* pane. To create a quick filter on an object, select the object in the *Result Objects* pane then click the *Add a quick filter* icon in the *Result Objects* toolbar.
5. Click *Run Query*.

17.4 CMS architecture and structure

The following schemas provide you more information on the CMS system database.

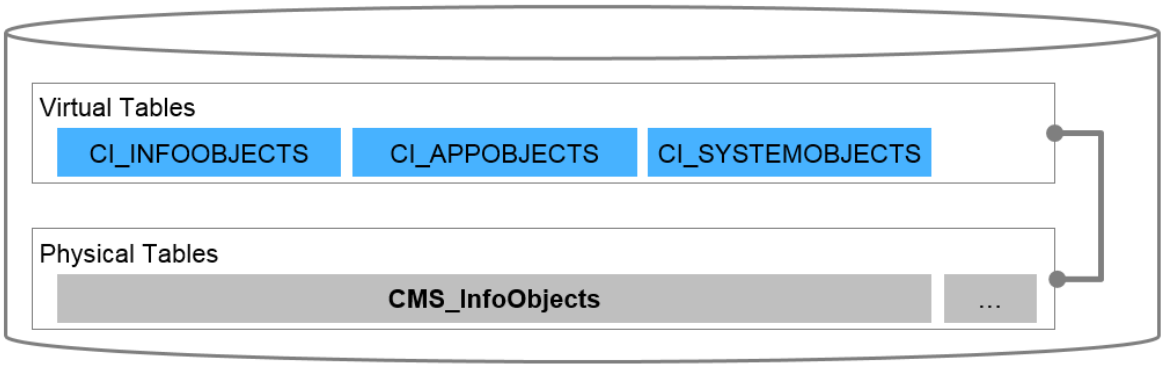
The following schema helps you understand the architecture of the SAP BusinessObjects platform.



SAP Business Objects platform - Architecture

Components	Description
CMC - Central Management Console	A web based tool that you use to configure security settings and manage the following items: • User • Content • Server
CMS system database	A database that stores the following BI platform information: • User • Server • Document • Configuration • Authentication The CMS system database is maintained by the Central Management Server (CMS) and can be referred as the system repository.
Query Builder (also referred as Admin Tool)	A web-based tool that you use to query the BusinessObjects repository and get the required information that cannot be found in CMC.
BI Repository Analysis	This solution uses the Semantic Layer of the BI platform (Universe and CMS DB Driver) to query the CMS.

The following schema describes the structure of the CMS System Database.



CMS System Database

The CMS System Database uses the following virtual tables to access the physical database table CMS_InfoObjects:

- CI_INFOOBJECTS
- CI_APPOBJECTS
- CI_SYSTEMOBJECTS

Here are the most important InfoObjects stored in the physical database table CMS_InfoObjects:

InfoObject	Description
SI_NAME	The name of the object
SI_KIND	The kind of object
SI_OWNER	The user name of the owner
SI_OWNERID	The user ID of the owner
SI_CHILDREN	The number of children
SI_CUID	CUIDs are Cluster Unique Identifiers that uniquely identify an InfoObject
SI_UNIVERSE	The universes used by the document

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering a SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Gender-Related Language

We try not to use gender-specific word forms and formulations. As appropriate for context and readability, SAP may use masculine word forms to refer to all genders.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

www.sap.com/contactsap

© 2020 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Please see https://www.sap.com/about/legal/trademark.html for additional trademark information and notices.

THE BEST RUN SAP