

# SQL Server 2016

Lab 07

## Developing a SQL Server 2016 Analysis Services Tabular Model

# Overview

**The estimated time to complete this lab is 90 minutes**

In this lab, you will develop a tabular BI Semantic Model (BISM) based on the **AdventureWorksDW2016** database. Specifically, it will enable the analysis of reseller sales compared to sales quota.

You will work with the entire development lifecycle covering the creation of a project, the importing of data from both a SQL Server relational database and also an OData data service, the enhancement of the model user interface, and also the creation of measures and a Key Performance Indicator (KPI).

Once you have explored and validated the model by using Excel, you will enhance it with perspectives, partitions and a security role.

Finally, you will deploy the project to a tabular instance of SQL Server 2016 Analysis Services.

*In order to break down this long lab into shorter activities, it is possible to commence the lab from any of the seven exercises. This enables you to recommence the lab at any of the seven exercises.*

You will learn how to:

- Create an Analysis Services tabular project
- Import data and creating table relationships
- Enhance the model use interface
- Define measures and a KPI
- Analyze the model in the Excel client
- Define perspectives, partitions and a role
- Deploy and manage the model

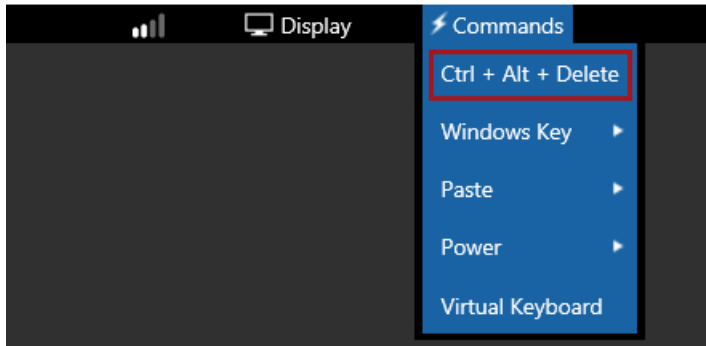
# Connecting to the Virtual Machine

In this exercise, you will use the lab hosting portal to connect to the virtual machine, and optimize the virtual machine environment for your language and location.

## Signing In

In this task, you will sign in to the virtual machine.

1. To sign in to the virtual machine, using the portal menu, click **Commands**, and then select **Ctrl + Alt + Delete**.



2. In the password box, enter **Pass@word1** (do not enter the period), and then click **Submit**.



*If you are not using a US English keyboard, the password you enter may not be correctly received by the virtual machine. You must complete the following task to sign in, and then update the virtual machine language.*

*Note: For lab users with English keyboards, if the @ symbol is above the 2, then your keyboard is a US English keyboard, and you should not complete the following task.*

## Updating the Virtual Machine Language

In this task, you will sign in to the virtual machine by using the on-screen keyboard, and then update the virtual machine language. This is important to ensure that your keyboard characters are correctly received by the virtual machine.

1. Located at the bottom-left of the virtual machine screen, click **Ease of Access**, and then select **On-Screen Keyboard**.



2. Use the on-screen keyboard to enter the password **Pass@word1**. (Do not enter the period.)

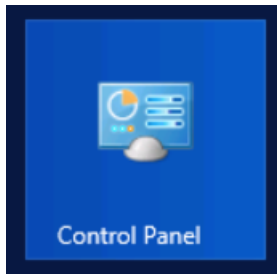
*Tip: To reveal the input password before submitting, click the following.*



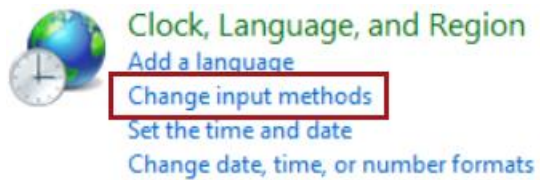
3. Submit the password.
4. Once signed in, to add a new language, on the taskbar, click the **Windows** button.



5. In the **Start** screen, click the **Control Panel** tile.



6. In the **Control Panel** window, from inside the **Clock, Language, and Region** group, click **Change Input Methods**.



7. In the **Language** window, click **Add a Language**.



8. In the **Add Languages** window, locate and select your language, and then click **Add** (or **Open**).

*If the selected language has regional variants, you will be directed to the **Regional Variants** window, in which case, select a variant, and then click **Add**.*

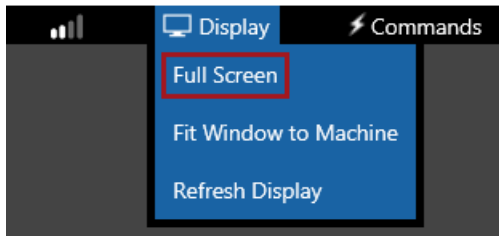
9. Close the **Language** window.
10. In the taskbar, click **ENG**, and then select your language.



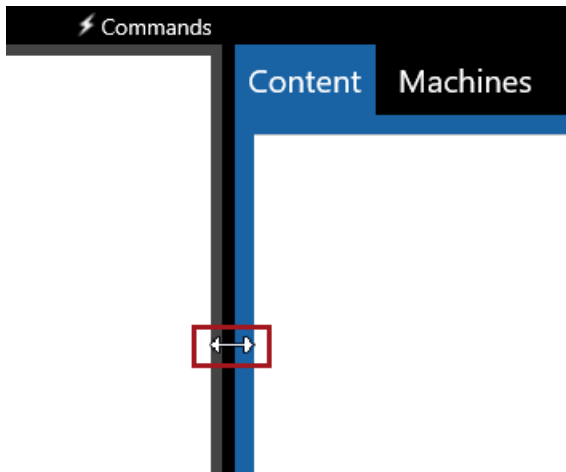
## Changing the Screen Resolution (Optional)

It is recommended that you change the virtual machine screen resolution to take full advantage of your screen size.

1. Using the portal menu, click **Display**, and then select **Full Screen**.



2. First, notice that the portal left pane can be resized.

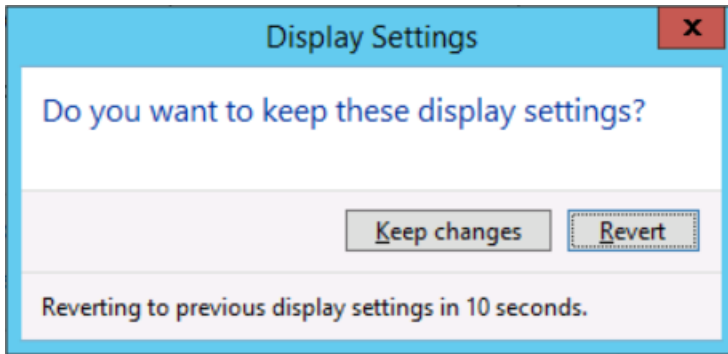


3. Resize the width of the pane, with the aim of arriving at a minimum width that allows the font size of the content (lab manual) to remain easily readable.

*Tip: Less width occupied by the pane allows for more room for the virtual machine screen.*

4. In the virtual machine screen, right-click the desktop, and then select **Screen Resolution**.
5. In the **Screen Resolution** window, in the **Resolution** dropdown list, select a higher resolution.  
*1024 x 768 is the recommended minimum, but use a higher resolution if this fits your screen size.*
6. Click **Apply**.

7. If the entire virtual machine screen is fully visible within the portal, click **Keep Changes**, otherwise, click **Revert**, and try a different resolution.



### Changing the Clock (Optional)

Changing the clock is for your convenience only, and therefore this task is optional.

1. In the taskbar, click the clock, and then select **Change Date and Time Settings**.
2. In the **Date and Time** window, click **Change Time Zone**.
3. In the **Time Zone Settings** window, in the **Time Zone** dropdown list, select your time zone.
4. Click **OK**.
5. In the **Date and Time** window, click **OK**.

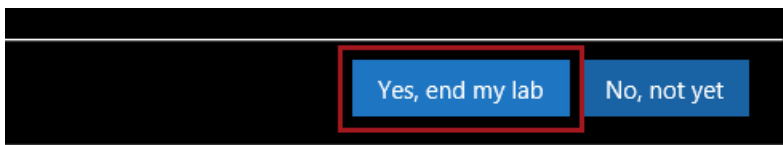
### Ending the Lab Session

In this task, you will explore how to end the lab session—when you are ready to do so.

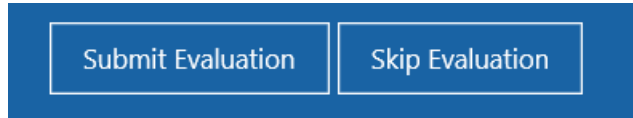
1. At the top right-corner of the portal, click **Exit**, and then select **End Lab**.



2. When prompted to end the lab, click **Yes, End My Lab**.



3. Once redirected to the evaluation form, please take a few moments to complete and submit your evaluation of this lab—your feedback assists us to deliver a great lab experience.

A blue rectangular button bar containing two white-outlined buttons. The left button is labeled 'Submit Evaluation' and the right button is labeled 'Skip Evaluation'.

*You are now ready to commence the lab.*



# 1: Creating the Tabular Model Project

In this exercise, you will prepare the SQL Server database, and then configure the SQL Server Data Tools (SSDT) Analysis Services options that will be applied to new tabular projects. You will then create a tabular project.

## Preparing the AdventureWorksDW2016 Database

In this task, you will execute a script to prepare the **AdventureWorksDW2016** database.

1. To open File Explorer, on the taskbar, click the **File Explorer** shortcut.



2. In the File Explorer window, navigate to the **D:\SQLServer2016BI\Lab07\Assets** folder.
3. Right-click the **Startup.cmd** file, and then select **Open**.
4. In the Command window, when prompted to press any key to continue, press any key.

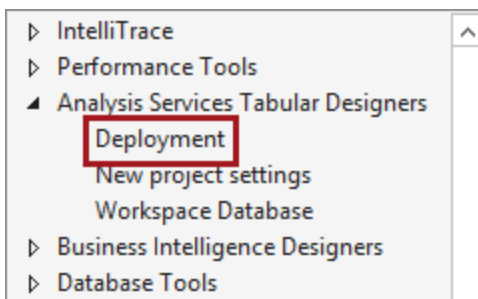
## Configuring the Analysis Services Options

In this task, you will launch SSDT to configure options that will be applied to new tabular projects.

1. To open SQL Server Data Tools (Visual Studio), on the taskbar, click the **Visual Studio** program shortcut.



2. To configure the Analysis Services options, on the **Tools** menu, select **Options**.
3. In the **Options** window, expand the **Analysis Services Tabular Designers** group (you will need to scroll down the list to locate this group), and then select the **Deployment** page.

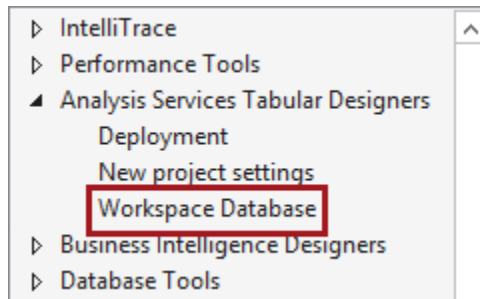


4. In the **Default Deployment Server** dropdown list, select **localhost\TABULAR**.



A screenshot of a software interface showing a dropdown menu for 'Default deployment server:'. The dropdown is open, and 'localhost\TABULAR' is selected and highlighted with a red rectangle. To the right of the dropdown is a button labeled 'Test Connection'.

5. Click **Test Connection**.
6. When the connection test has succeeded, click **OK**.
7. Select the **Workspace Database** page



8. In the **Workspace Server** dropdown list, select **localhost\TABULAR**.



A screenshot of a software interface showing a dropdown menu for 'Workspace server'. The dropdown is open, and 'localhost\TABULAR' is selected and highlighted with a red rectangle. To the right of the dropdown is a button labeled 'Test Connection'.

*When you work with the tabular model designer, you are working with a temporary Analysis Services database that automatically loads on a workspace server. It is also possible to use an integrated workspace which uses an internal Analysis Services instance in the background, and so does not require an instance of Analysis Services to be installed.*

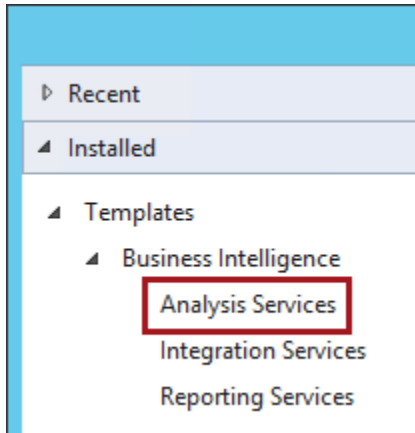
*The Integrated Workspace option is a new capability. To learn more about this capability, read the Analysis Services Team Blog article [Introducing Integrated Workspace Mode for SQL Server Data Tools for Analysis Services Tabular Projects \(SSDT Tabular\)](#).*

9. Click **OK**.

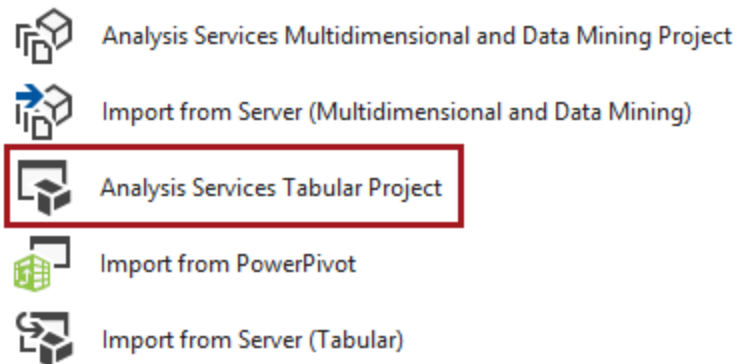
## Creating an Analysis Services Tabular Project

In this task, you will create an Analysis Services tabular project.

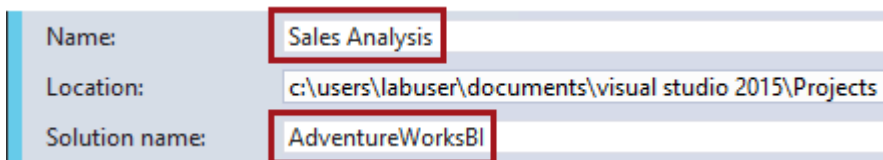
1. To create a solution, on the **File** menu, select **New | Project**.
2. In the **New Project** window, in the left pane, from inside the **Business Intelligence** group, select the **Analysis Services** template.



3. Select the **Analysis Services Tabular Project** template.

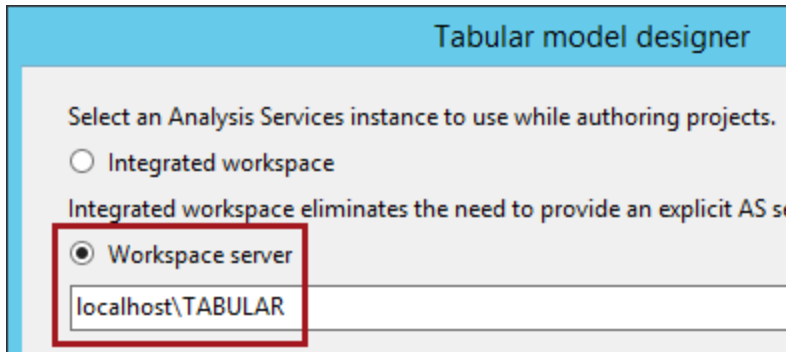


4. In the **Name** box, replace the text with **Sales Analysis**.
5. In the **Solution Name** box, replace the text with **AdventureWorksBI**.

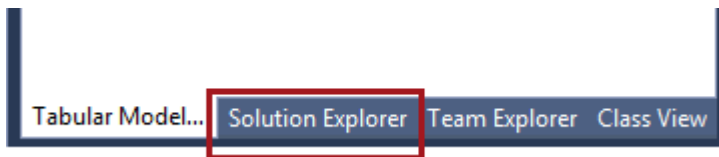


6. Click **OK**.

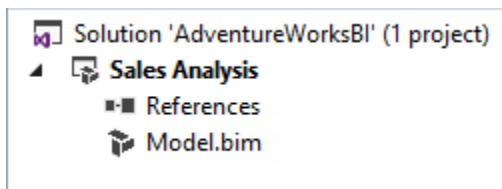
7. In the **Tabular Model Designer** window, notice that the project will use the default workspace server.



8. Click **OK**.
9. In the right pane, select **Solution Explorer**.



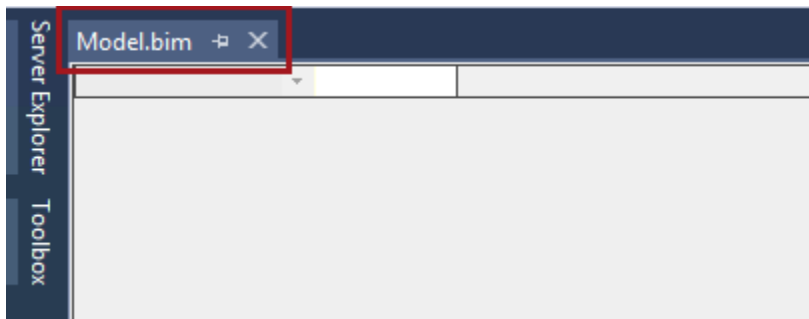
10. Notice that the **Sales Analysis** project consists of a single project item named **Model.bim**.



The **Model.bim** item is the data model that you will develop in this lab.

*Each tabular project consists of a single data model, and no additional data models can be added. When deployed for the first time, the project creates a database on the target Analysis Services instance. You will deploy the project later in this lab.*

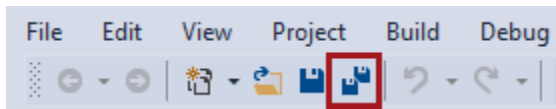
11. Notice that the **Model.bim** item was automatically opened upon project creation.



12. To save the project, on the **File** menu, select **Save All**.

*It is a good practice to regularly save the solution to protect your development effort in case of an unexpected application crash.*

*The **Save All** function is also available on the toolbar.*



## 2: Importing Data and Creating Relationships

In this exercise, you will import data from two data sources. The first will be the Microsoft SQL Server **AdventureWorksDW2016** database. The second will be an OData data service.

The **Table Import Wizard** will be used to configure the connection details, and select the tables (and feed) to import. The interface includes the ability to select the table columns to import and to filter rows. It is important not to include columns or rows that are not required by the model. Including only data that is required by the model will conserve server resources and help speed up query response times.

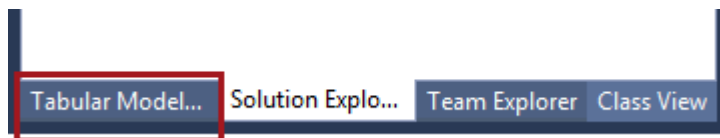
You will learn that the presence of foreign keys in the source data can result in the creation of relationships in the model. When importing data from different data sources there is no way for the Wizard to detect and create relationships, so you will create these manually.

*To commence the lab starting from this exercise, in SSDT, open the **AdventureWorksBI.sln** file from the **D:\SQLServerBI\Lab07\Starter\Ex2** folder.*

### Importing Data from a SQL Server Relational Database

In this task, you will import several tables from a SQL Server relational database.

1. To manage the tabular model design, in the right pane, select **Tabular Model Explorer**.



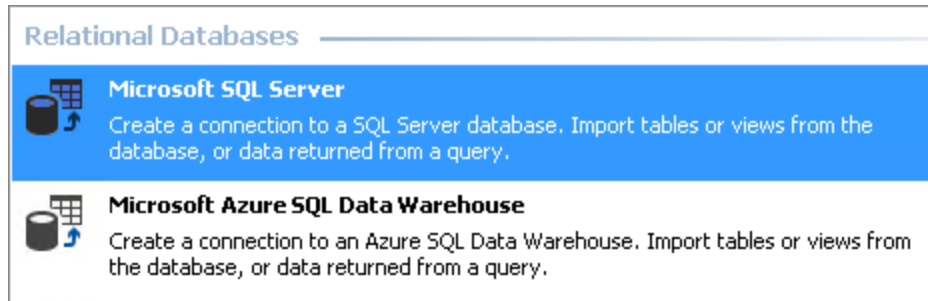
*This pane lets you conveniently navigate through the various metadata objects in a model, including data sources, tables, measures, and relationships.*

*To learn more about this capability, read the Analysis Services Team Blog article [Introducing Tabular Model Explorer for SQL Server Data Tools for Analysis Services Tabular Projects \(SSDT Tabular\)](#).*

2. To launch the Table Import Wizard, in **Tabular Model Explorer**, right-click the **Data Sources** folder, and then select **Import from Data Source**.

*The functions available in **Tabular Model Explorer** are also available on the Visual Studio **Model**, **Table** and **Column** menus.*

3. In the **Table Import Wizard** window, at the **Connect to a Data Source** step, notice that **Microsoft SQL Server** is selected.

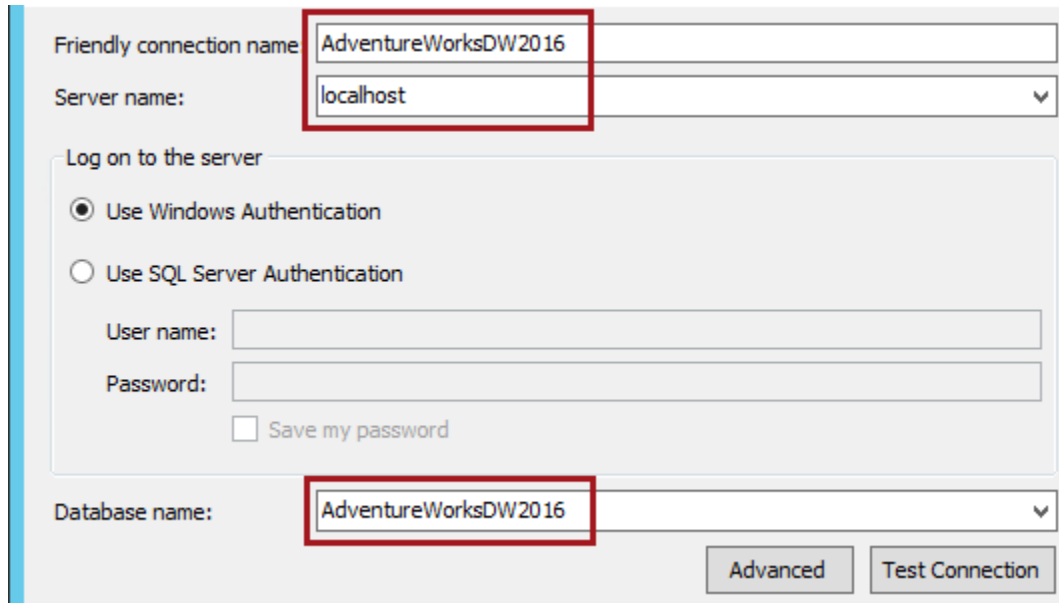


**Relational Databases**

**Microsoft SQL Server**  
Create a connection to a SQL Server database. Import tables or views from the database, or data returned from a query.

**Microsoft Azure SQL Data Warehouse**  
Create a connection to an Azure SQL Data Warehouse. Import tables or views from the database, or data returned from a query.

4. Click **Next**.
5. At the **Connect to a Microsoft SQL Server Database** step, in the **Server Name** dropdown list, enter **localhost** (do not open the dropdown list—it takes a long time for it to discover available instances).
6. In the **Database Name** dropdown list, select the **AdventureWorksDW2016** database.
7. In the **Friendly Connection Name** box, modify the text to **AdventureWorksDW2016**.



Friendly connection name: AdventureWorksDW2016

Server name: localhost

Log on to the server

☒ Use Windows Authentication

☐ Use SQL Server Authentication

User name:

Password:

☐ Save my password

Database name: AdventureWorksDW2016

Advanced Test Connection

8. Click **Next**.

9. At the **Impersonation Information** step, select the **Service Account** option.

☐ Specific Windows user name and password  
Connects to the data source using the credentials of the user named below.

User Name:

Password:

☒ **Service Account**  
Connects to the data source using the credentials of the user running the Analysis Service server.

☐ Current User  
Connects to the data source using a low privilege account.

*A best practice is to use the credentials of a dedicated domain account that has minimum privileges, just to read data.*

10. Click **Next**.
11. At the **Choose How to Import the Data** step, notice the default option to select from a list of tables and views, and then click **Next**.

*You will create a table based on a query later in this exercise.*

12. At the **Select Tables and Views** step, select the **DimEmployee** source table.

<input type="checkbox"/>		DimDepartmentGroup	dbo	
<input checked="" type="checkbox"/>		DimEmployee	dbo	DimEmployee
<input type="checkbox"/>		DimGeography	dbo	

13. In the corresponding **Friendly Name** column, modify the text to **Salesperson**.

<input type="checkbox"/>		DimDepartmentGroup	dbo	
<input checked="" type="checkbox"/>		DimEmployee	dbo	Salesperson
<input type="checkbox"/>		DimGeography	dbo	

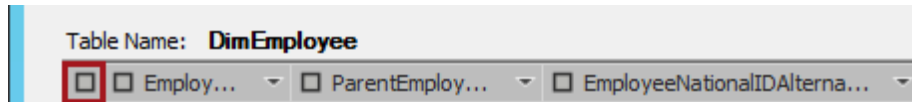
14. To preview and filter the data in the **DimEmployee** table, click **Preview & Filter**.

Select Related Tables **Preview & Filter**

< Back Next > Finish Cancel



- In the **Table Import Wizard** window, to unselect all columns, uncheck the checkbox located in the top-left corner.



- Select the following columns (check the checkbox in each column header).

#### Column

EmployeeKey

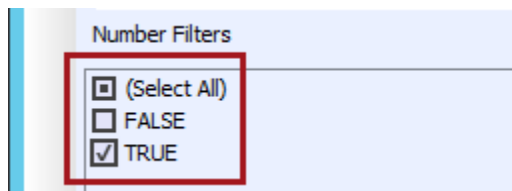
SalesTerritoryKey

FirstName

LastName

LoginID

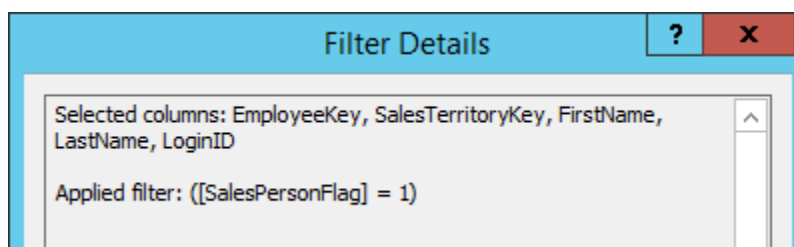
- To filter the table to include only salespeople rows, scroll to the very end of the columns to locate the **SalesPersonFlag** column (the sixth last column).
- In the **SalesPersonFlag** column header, click the down-arrow, unselect (**Select All**), and then select **TRUE**.



- Click **OK**.
- In the **Table Import Wizard** window, click **OK**.
- To review the column selection and filters, in the **DimEmployee** row in the grid, click the **Applied Filters** link.

<input type="checkbox"/>		DimDepartmentGroup	dbo		
<input checked="" type="checkbox"/>		DimEmployee	dbo	Salesperson	<a href="#">Applied filters</a>
<input type="checkbox"/>		DimGeography	dbo		

- Verify that the details in the **Filter Details** window look like the following.



- Click **OK**.

24. Select the **DimProduct** table, modify the **Friendly Name** column to **Product**, and then select only the following columns.

**Column**

---

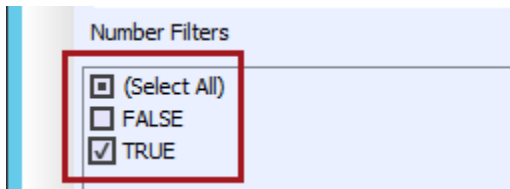
ProductKey

ProductSubcategoryKey

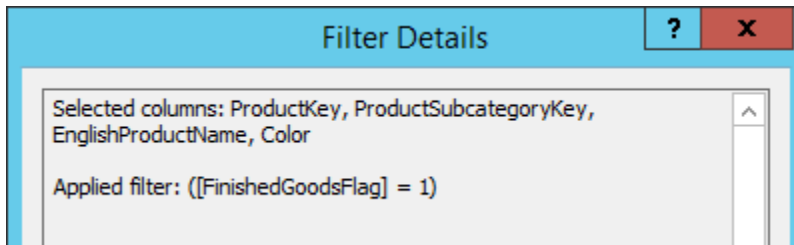
EnglishProductName

Color

25. To filter the table to include only finished goods rows, in the **FinishedGoodsFlag** column header (located to the left of the **Color** column), click the down-arrow, unselect (**Select All**), and then select **TRUE**.



26. Click **OK**.
27. In the **Table Import Wizard** window, click **OK**.
28. Review the column selection and filters, and verify that the details look like the following.



29. Select the **DimProductCategory** table, modify the **Friendly Name** column to **ProductCategory**, and then select only the following columns.

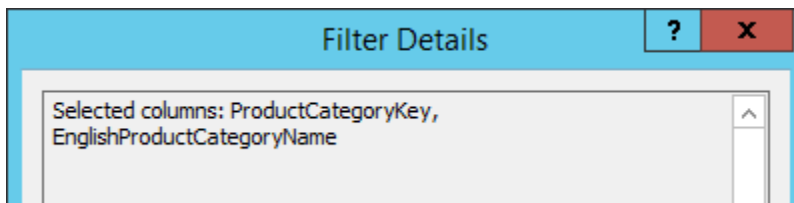
**Column**

---

ProductCategoryKey

EnglishProductCategoryName

30. Review the column selection and filters, and verify that the details look like the following.



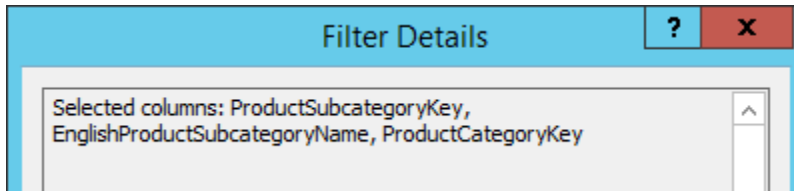
31. Select the **DimProductSubcategory** table, modify the **Friendly Name** column to **ProductSubcategory**, and then select only the following columns.

**Column**

---

ProductSubcategoryKey  
EnglishProductSubcategoryName  
ProductCategoryKey

32. Review the column selection and filters, and verify that the details look like the following.



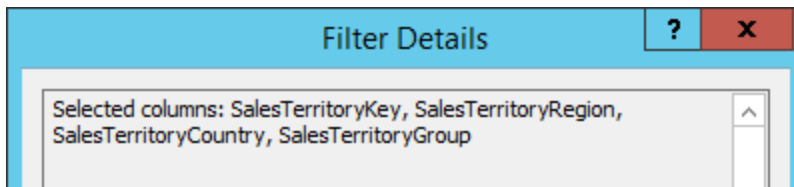
33. Select the **DimSalesTerritory** table, modify the **Friendly Name** column to **SalesTerritory**, and then select only the following columns.

**Column**

---

SalesTerritoryKey  
SalesTerritoryRegion  
SalesTerritoryCountry  
SalesTerritoryGroup

34. Review the column selection and filters, and verify that the details look like the following.



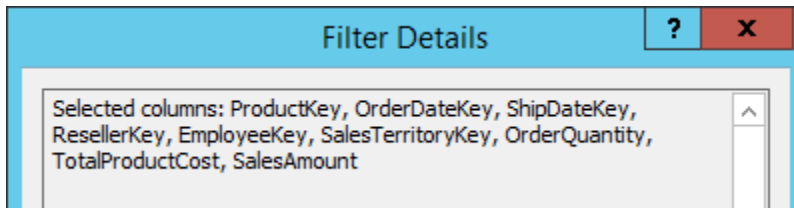
35. Select the **FactResellerSales** table, modify the **Friendly Name** column to **ResellerSales**, and then select only the following columns.

**Column**

---

ProductKey  
OrderDateKey  
ShipDateKey  
ResellerKey  
EmployeeKey  
SalesTerritoryKey  
OrderQuantity  
TotalProductCost  
SalesAmount

36. Review the column selection and filters, and verify that the details look like the following.



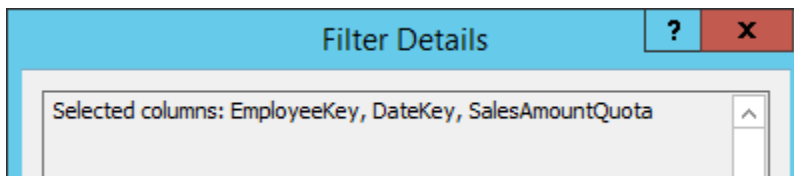
37. Select the **FactSalesQuota** table, modify the **Friendly Name** column to **SalesQuota**, and then select only the following columns.

**Column**

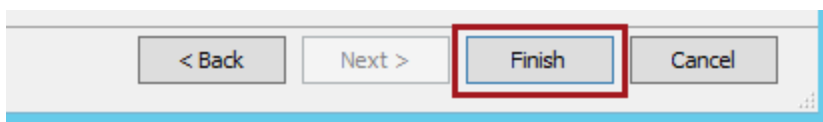
---

EmployeeKey  
DateKey  
SalesAmountQuota

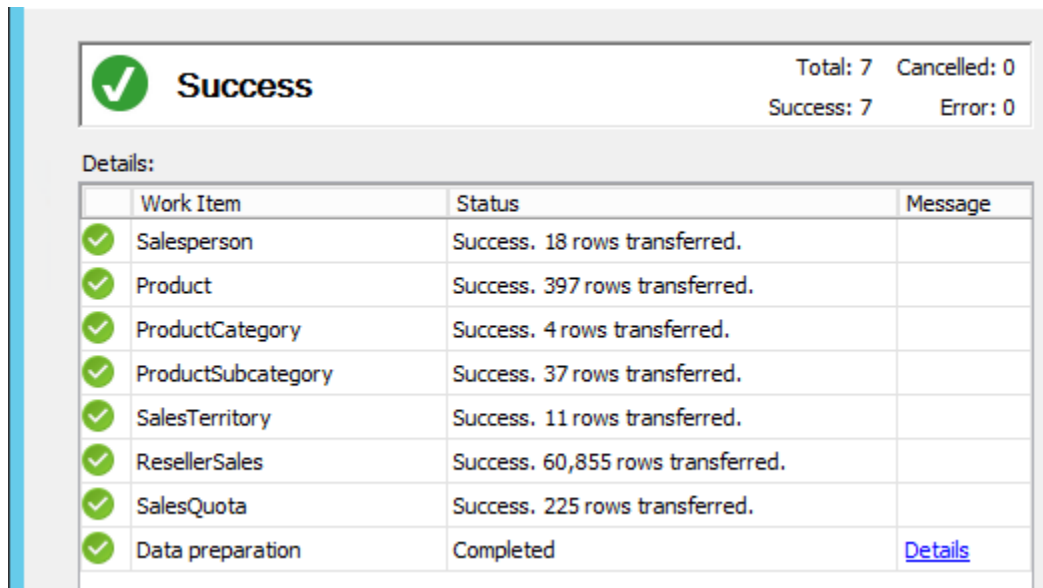
38. Review the column selection and filters, and verify that the details look like the following.



39. In the **Table Import Wizard** window, to add the tables to the model, click **Finish**.



40. At the Importing step, verify that the status of each work item looks like the following.



The screenshot shows a 'Success' dialog box with a green checkmark icon. It displays summary statistics: Total: 7, Cancelled: 0, Success: 7, and Error: 0. Below this is a 'Details:' section containing a table with three columns: Work Item, Status, and Message. The table lists eight work items, all with a status of 'Success' except for 'Data preparation' which is 'Completed'. Each row has a green checkmark in the first column. A 'Details' link is present at the bottom right of the table.

Work Item	Status	Message
✓ Salesperson	Success. 18 rows transferred.	
✓ Product	Success. 397 rows transferred.	
✓ ProductCategory	Success. 4 rows transferred.	
✓ ProductSubcategory	Success. 37 rows transferred.	
✓ SalesTerritory	Success. 11 rows transferred.	
✓ ResellerSales	Success. 60,855 rows transferred.	
✓ SalesQuota	Success. 225 rows transferred.	
✓ Data preparation	Completed	<a href="#">Details</a>

41. Click the **Details** link.

42. In the **Details** window, review the creation of model relationships.

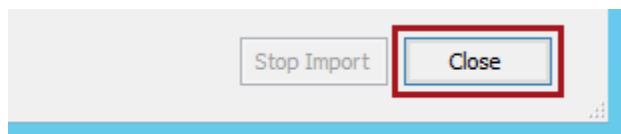
*The Table Import Wizard has created relationships for all foreign keys where the referenced and referencing columns have been included in the table's column selection.*

43. Notice in particular the second last relationship created between the **FactResellerSales** table's **SalesTerritoryKey** column and the **DimSalesTerritory** table's **SalesTerritoryKey** column.

*This relationship is marked as inactive. Only one active path can exist, directly or indirectly, between two tables in the model. An active relationship already exists between the **FactResellerSales** and **DimEmployee** tables, and the **DimEmployee** table has an active relationship to the **DimSalesTerritory** table. Active relationships are used by default in client tools. You will explore this concept in more detail later in this lab.*

44. To close the **Details** window, click **OK**.

45. To close the **Table Import Wizard** window, click **Close**.



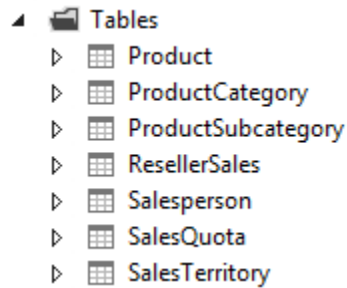
46. Notice the addition of the seven tables inside the model designer.



*All data loaded into the data model is read-only. The only way to modify the data is to modify the source data and then either refresh an individual table or table partition, or refresh the data source that will refresh all tables that are based on that data source.*

*It is also possible to modify the table properties that allow modifying the selection of columns and row filters.*

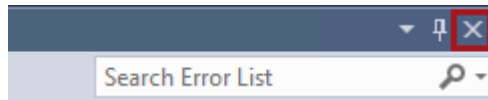
47. In **Tabular Model Explorer**, expand the **Tables** folder, and notice that the same tables are listed.



48. Notice the **Error List** pane at the bottom of the user interface, and review the warning.

49. To close the **Error List** pane, click the **X** located in the top right corner of the window.

*This window opens automatically after certain designer processes. You may want to close this window to maximize the available model designer space.*

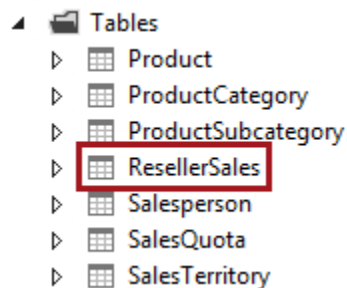


50. To save the project, on the **File** menu, select **Save All**.

### Exploring the ResellerSales Table

In this task, you will explore the data loaded into the **ResellerSales** table.

1. In **Tabular Model Explorer**, select the **ResellerSales** table.



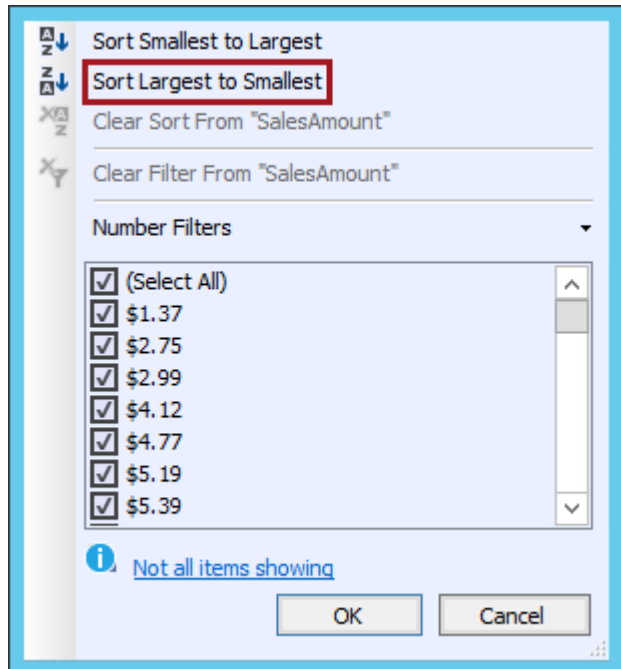
2. Notice that the **ResellerSales** table is now in focus.

Salesperson	Product	ProductCategory	ProductSubcategory	SalesTerritory	<b>ResellerSales</b>	SalesQuota
-------------	---------	-----------------	--------------------	----------------	----------------------	------------

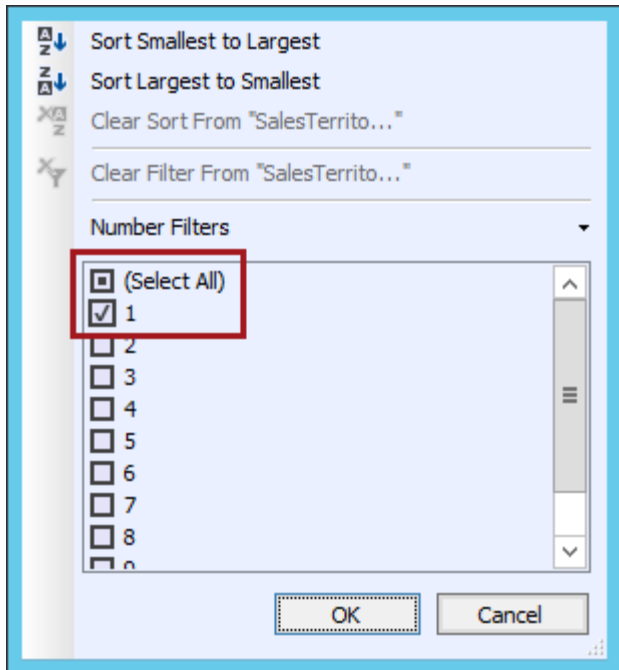
3. At the bottom-left corner, notice that this table has 60,855 rows.

Record: |< < 1 of 60,855 > >|

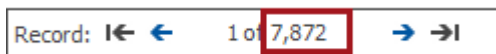
4. To sort the rows by descending **SalesAmount** value, in the **SalesAmount** column header, click the down-arrow, and then select **Sort Largest to Smallest**.



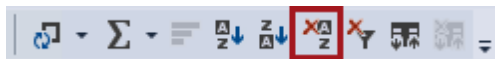
5. To filter the data by a particular region, in the **SalesTerritoryKey** column header, click the down arrow, unselect **(Select All)**, select **1**.



6. Click **OK**.
7. Notice that the filtered table has 7,872 rows.



8. To reset the table sort, on the toolbar, click the **Clear Sort** button.



9. To remove all table filters, on the toolbar, click the **Clear All Filters** button.

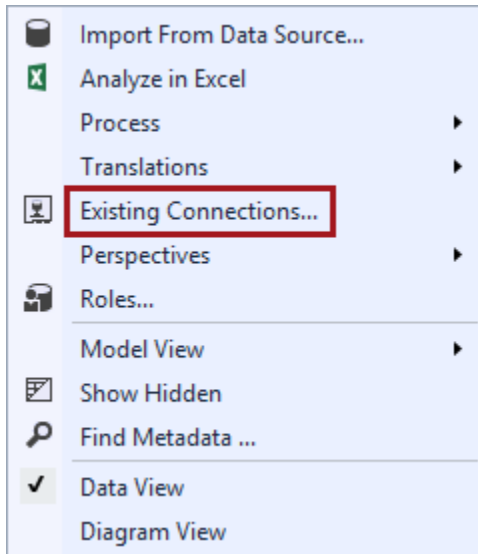




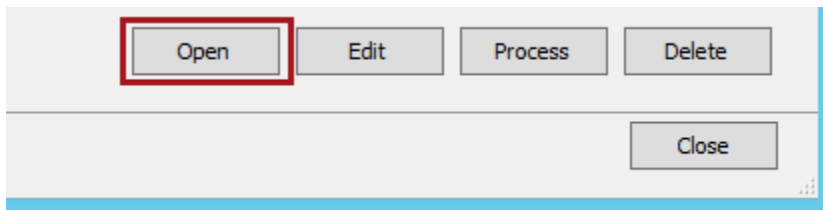
## Importing an Additional Table from an Existing Connection

In this task, you will open the **AdventureWorksDW2016** connection to create a table based on a query.

1. In **Tabular Model Explorer**, right-click the **Sales Analysis** model, and then select **Existing Connections**.

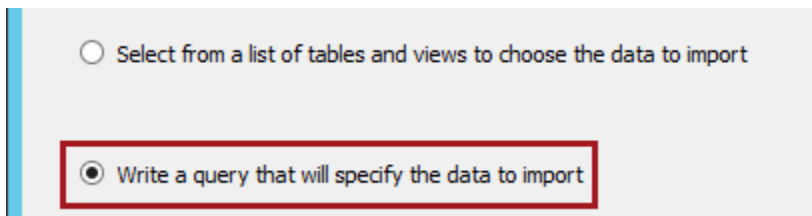


2. In the **Existing Connections** window, notice that the **AdventureWorksDW2016** connection is selected, and then click **Open**.



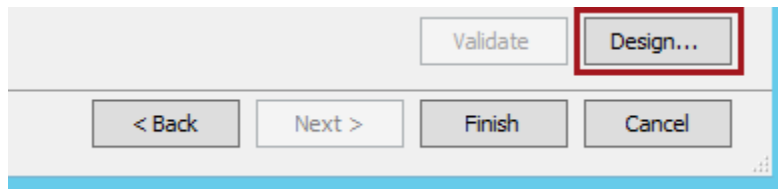
*Opening a connection re-launches the Table Import Wizard by using that connection.*

3. In the **Table Import Wizard** window, at the **Choose How to Import the Data** step, select the **Write a Query That Will Specify the Data to Import** option.



4. Click **Next**.
5. At the **Specify a SQL Query** step, in the **Friendly Query Name** box, modify the text to **Reseller**.

6. To author a query, click **Design**.



7. In the **Table Import Wizard** window, in the **Database View** pane (located at the left), expand the **Tables**, and then expand the **DimReseller** table.
8. Select only the following columns (in this order).

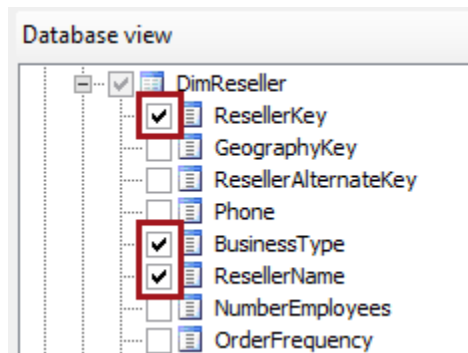
**Column**

---

ResellerKey

BusinessType

ResellerName



9. Expand the **DimGeography** table, and then select only the following columns (in this order).

**Column**

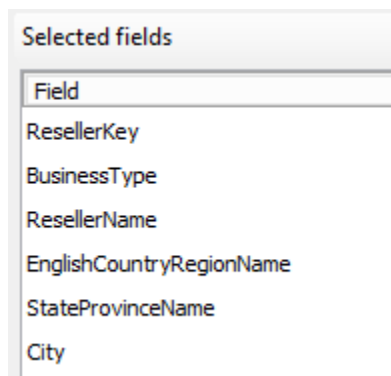
---

EnglishCountryRegionName

StateProvinceName

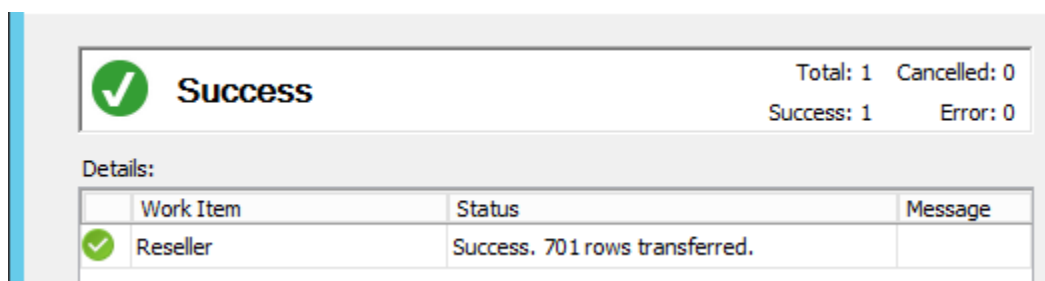
City

11. Verify that the **Selected Fields** pane looks like the following.

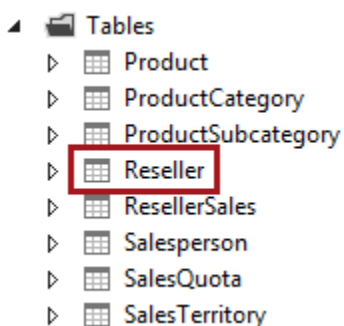


*The Auto Detect feature has automatically identified the relationship between the two tables and will construct an appropriate join clause to relate the data in these tables.*

12. Click **OK**.
13. In the **Table Import Wizard** window, notice that the query statement that has been entered into the **SQL Statement** box.
14. Click **Finish**.
15. At the **Importing** step, verify that the status looks like the following.



16. Click **Close**.
17. In **Tabular Model Explorer**, notice the addition of the **Reseller** table.



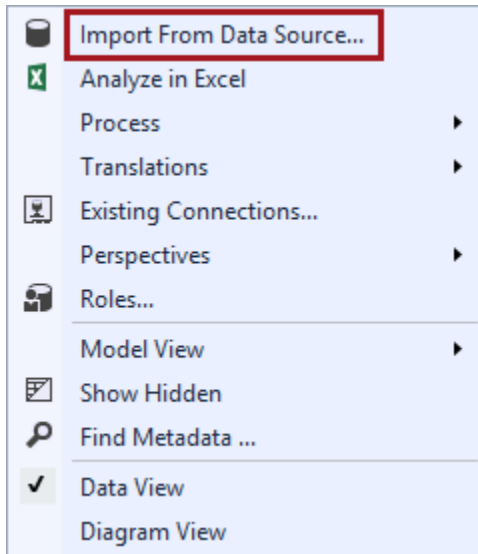
*Relationships do not exist between the tables imported in the first import process and the **Reseller** table. You will add them in a later task of this exercise.*

18. To save the project, on the **File** menu, select **Save All**.

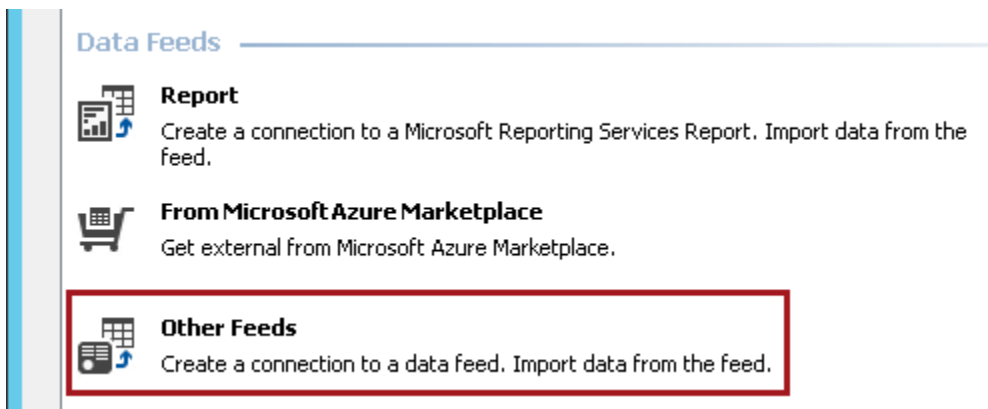
## Importing Data from an OData Data Service

In this task, you will use the Table Import Wizard to import date columns from the **CorporateDate** data feed.

1. To launch the Table Import Wizard, in **Tabular Model Explorer**, right-click the **Sales Analysis** model, and then select **Import from Data Source**.

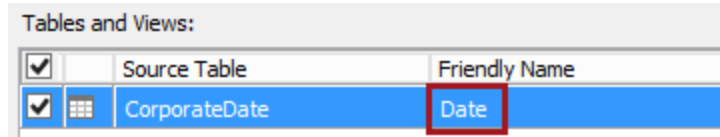


2. In the **Table Import Wizard** window, at the **Connect to a Data Source** step, scroll to the end of the list, and then select **Other Feeds**.



3. Click **Next**.
4. At the **Connect to a Data Feed** step, in the **Friendly Connection Name** box, modify the text to **AdventureWorks Feeds**.
5. In the **Data Feed URL** box, enter **http://localhost:11111/AdventureWorksFeeds/DataService.svc**.  
*For convenience, the URL can be copied from the D:\SQLServer2016BI\Lab07\Assets\Snippets.txt file.*
6. Click **Next**.
7. At the **Impersonation Information** step, select the **Service Account** option, and then click **Next**.

8. At the **Select Tables and Views** step, notice that the only table—**CorporateDate**—is selected.
9. In the corresponding **Friendly Name** column, modify the text to **Date**.

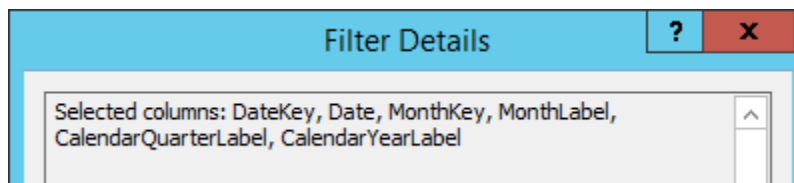


10. To preview and filter the data in the **CorporateDate** table (feed), click **Preview & Filter**.
11. Select only the following columns.

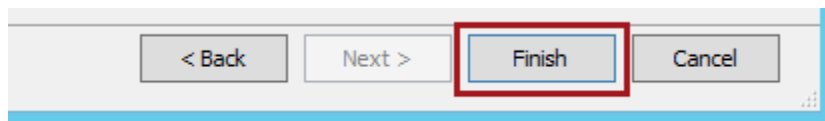
#### Column

DateKey  
Date  
MonthKey  
MonthLabel  
CalendarQuarterLabel  
CalendarYearLabel

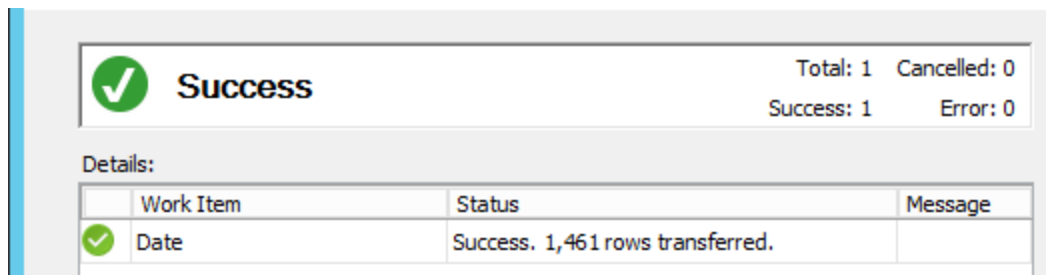
12. Review the column selection and filters, and verify that the details look like the following.



13. In the **Table Import Wizard** window, to add the table to the model, click **Finish**.

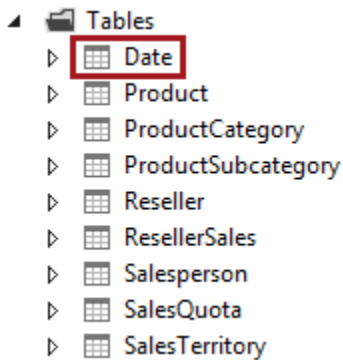


14. At the Importing step, verify that the status looks like the following.



15. To close the **Table Import Wizard** window, click **Close**.

16. In **Tabular Model Explorer**, notice the addition of the **Date** table.



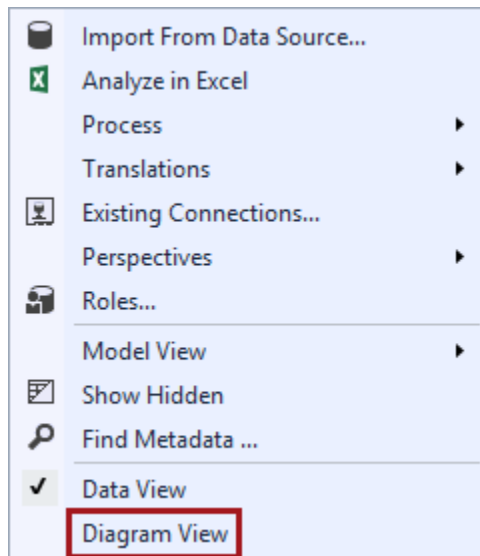
*Relationships do not exist between the tables imported from the **AdventureWorksDW2016** database and the **CorporateDate** feed. You will add them in the next task.*

17. To save the project, on the **File** menu, select **Save All**.

### Adding Relationships to the Date Table in Data View

In this task, you will define relationships between the **ResellerSales** and **Date** tables, and between the **SalesQuota** and **Date** tables.

1. To view the model in Diagram View, in **Tabular Model Explorer**, right-click the **Sales Analysis** model, and then select **Diagram View**.



*Tip: It is also possible to toggle between Data View and Diagram View by clicking the icons located at the bottom-right corner.*



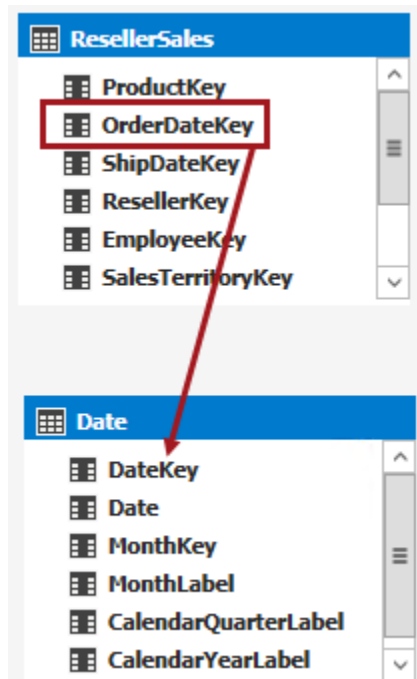
*Diagram View is a useful way to visualize the model's tables and relationships. Beyond the model visualization, this view exposes useful functionality to enhance the design of the data model.*

Oftentimes this functionality is available in Data View (the default view consisting of the tables and rows).

Note that calculated columns and measures can only be defined in Data View, and hierarchies can only be defined in Diagram View.

When appropriate, you can choose to work in the view that is most productive for you.

2. To create a relationship between the **ResellerSales** table and the **Date** table, first reposition the **Date** table so that it is directly beneath the **ResellerSales** table. (You may need to horizontally scroll to locate the **Date** table.)
3. In the **ResellerSales** table, drag the **OrderDateKey** column onto the **DateKey** column of the **Date** table.



4. Create a second relationship, between the same tables by dragging the **ShipDateKey** column onto the **DateKey** column of the **Date** table.
5. Create a third relationship, between the **SalesQuota** table and the **Date** table, relating the two **DateKey** columns.
6. Create a fourth relationship, between the **Reseller** table and the **ResellerSales** table, relating the two **ResellerKey** columns.
7. To review the model relationships, in **Tabular Model Explorer**, right-click the **Relationships** folder, and then select **Manage Relationships**.

- Verify that there 11 relationships listed, and notice the **Active** column values.

Manage Relationships			
<div>Create Edit Delete</div>			
Active	Table 1	Cardinality	Filter Direction
Yes	Product [ProductSubcategoryKey]	Many to One (*:1)	<< To Product
Yes	ProductSubcategory [ProductCategoryKey]	Many to One (*:1)	<< To ProductSubcategory
Yes	ResellerSales [EmployeeKey]	Many to One (*:1)	<< To ResellerSales
Yes	ResellerSales [OrderDateKey]	Many to One (*:1)	<< To ResellerSales
Yes	ResellerSales [ProductKey]	Many to One (*:1)	<< To ResellerSales
Yes	ResellerSales [ResellerKey]	Many to One (*:1)	<< To ResellerSales
No	ResellerSales [SalesTerritoryKey]	Many to One (*:1)	<< To ResellerSales
No	ResellerSales [ShipDateKey]	Many to One (*:1)	<< To ResellerSales
Yes	Salesperson [SalesTerritoryKey]	Many to One (*:1)	<< To Salesperson
Yes	SalesQuota [DateKey]	Many to One (*:1)	<< To SalesQuota
Yes	SalesQuota [EmployeeKey]	Many to One (*:1)	<< To SalesQuota

Two relationships are marked as inactive. Recall that only one active path can exist between two tables. The first relationship you created is automatically configured to be active. The second, and any subsequently created relationships, will be configured to be inactive.

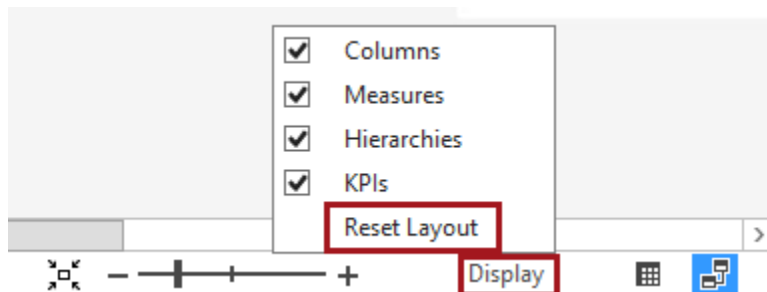
You can use the **Manage Relationships** window or the **Diagram View** to edit relationships and change their active status.

- Click **Close**.

## Exploring the Model in Diagram View

In this task, you will explore the model tables in Diagram View.

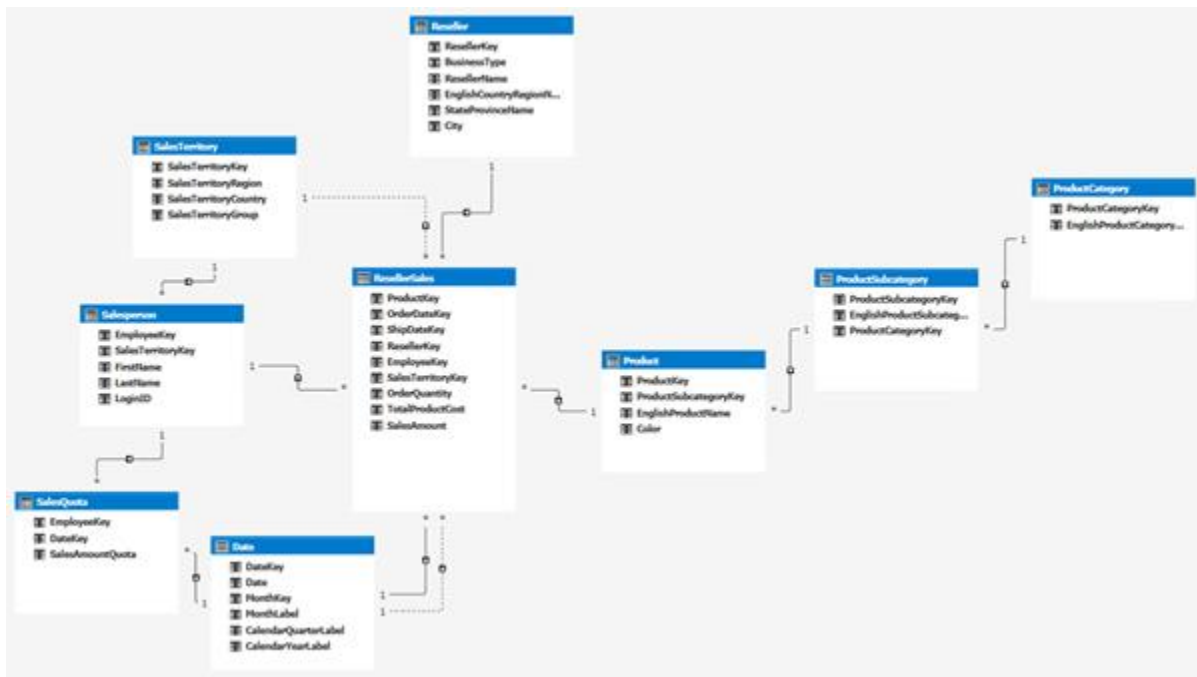
- To arrange the tables, at the bottom right-corner, click **Display**, and then select **Reset Layout**.



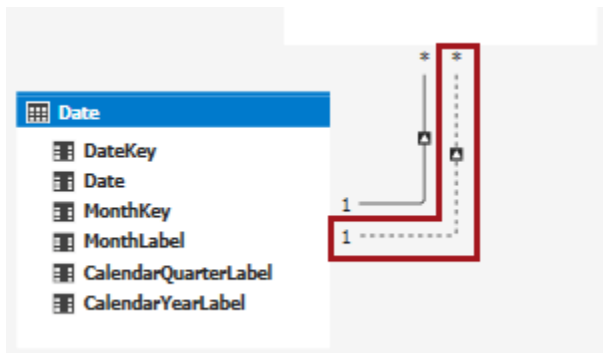
- When prompted to confirm the action, click **Reset Layout**.



3. Reposition and resize each table so that the model is easy to understand.



4. Notice that the second relationship to the **Date** table you created is a dotted line, representing an inactive relationship.



5. Hover over several relationships to highlight the columns involved in the relationship.
6. To save the project, on the **File** menu, select **Save All**.

# 3: Enhancing the Model User Interface

In this exercise, you will enhance the data model user interface to better support end user exploration and querying.

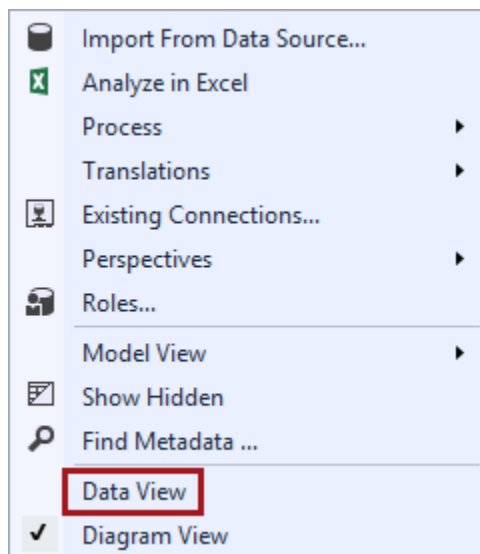
To produce an enhanced data model, for each table, and as appropriate, you will rename columns, create calculated columns based on expressions, create hierarchies to support the ability to drill down and drill up to analyze data at different levels of granularity, and hide columns.

*To commence the lab starting from this exercise, in SSDT, open the **AdventureWorksBI.sln** file from the **D:\SQLServerBI\Lab07\Starter\Ex3** folder. To process the model, on the **Model** menu, select **Process / Process All**.*

## Enhancing the Date Table

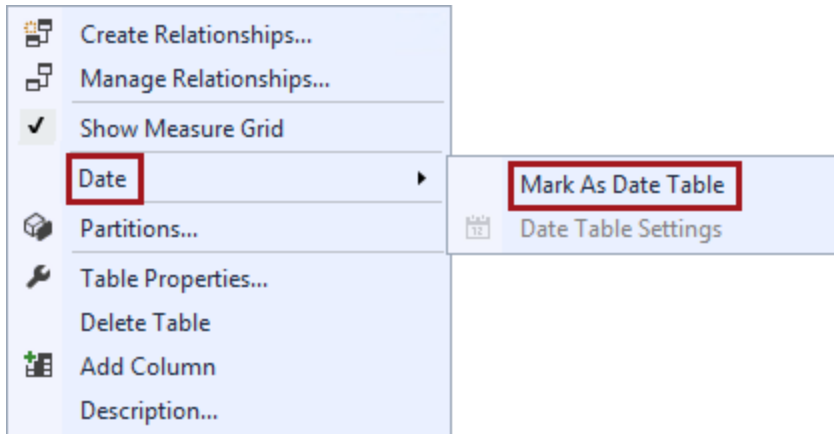
In this task, you will extend the **Date** table with a **Calendar** hierarchy.

1. To view the model in Data View, in **Tabular Model Explorer**, right-click the **Sales Analysis** model, and then select **Data View**.



2. In **Tabular Model Explorer**, select the **Date** table.

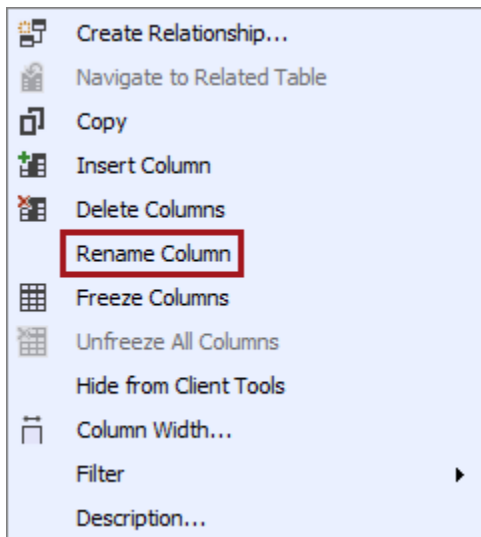
3. To mark the table as a date table, in **Tabular Model Explorer**, right-click the **Date** table, and then select **Date | Mark as Date Table**.



4. In the **Mark as Date Table** window, in the **Date** dropdown list, notice that the **Date** column is selected, and then click **OK**.

*Marking a date table will help client applications understand how time is defined in the data model. The Excel PivotTable Fields pane is one such example that interrogates the data model for a date table, and it will surface appropriate time-based filter options based on a marked date table.*

5. To rename the **MonthLabel** column, right-click the **MonthLabel** column header, and then select **Rename Column**.

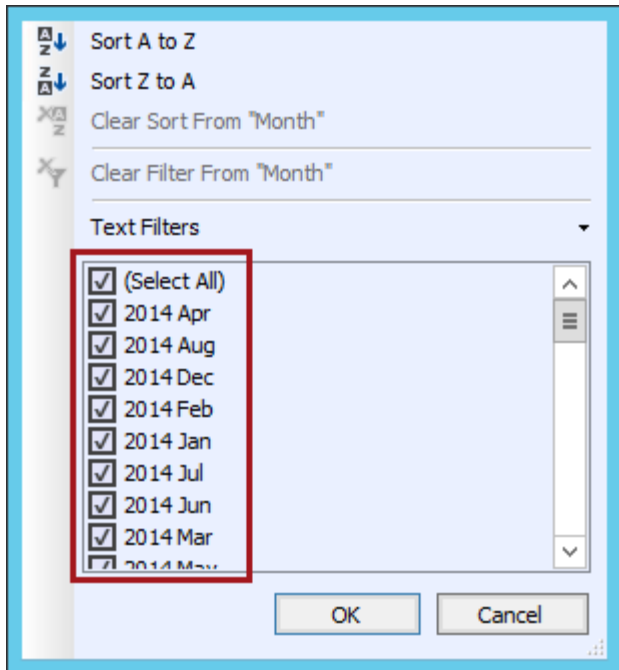


*Tip: You can also double-click the column header to rename the column.*

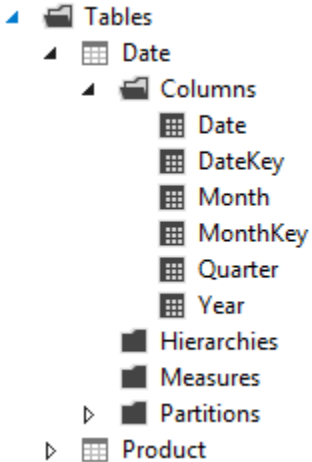
6. Replace the column header text with **Month**, and then press **Enter**.
7. Rename also the following two columns.

Column Name	New Column Name
CalendarQuarterLabel	Quarter
CalendarYearLabel	Year

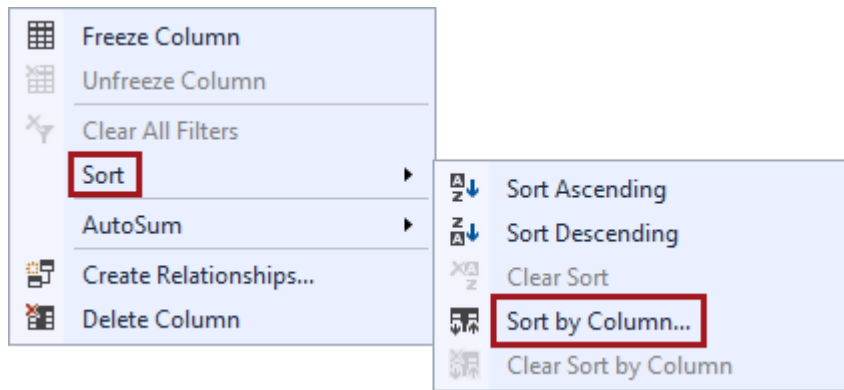
8. To view the **Month** column values, in the **Month** column header, click the down-arrow, and then review the distinct column values, available for filtering, found in the column.



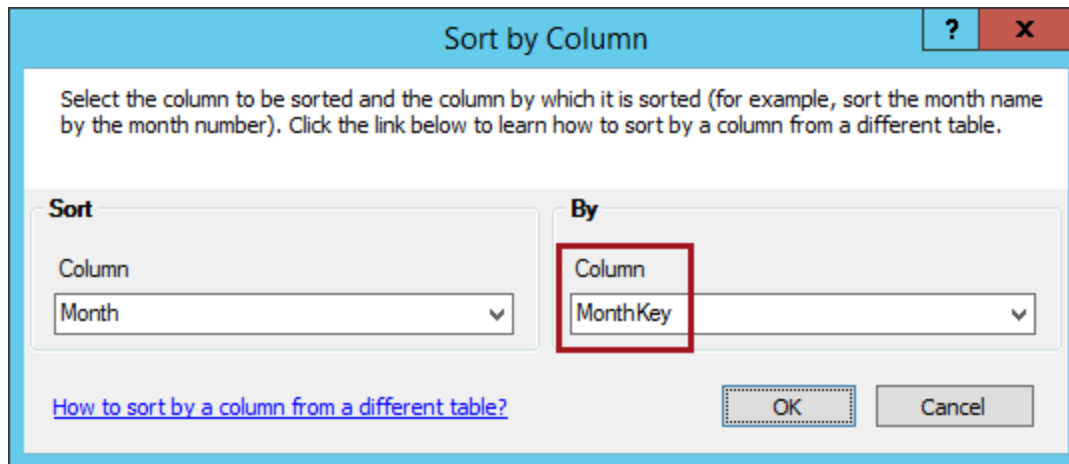
9. Notice that the months are sorted alphabetically, and then click **Cancel**.
10. To configure the months to sort chronologically, in **Tabular Model Explorer**, expand the **Date** table, and then expand the **Columns** folder.



11. Right-click the **Month** column, and then select **Sort | Sort by Column**.



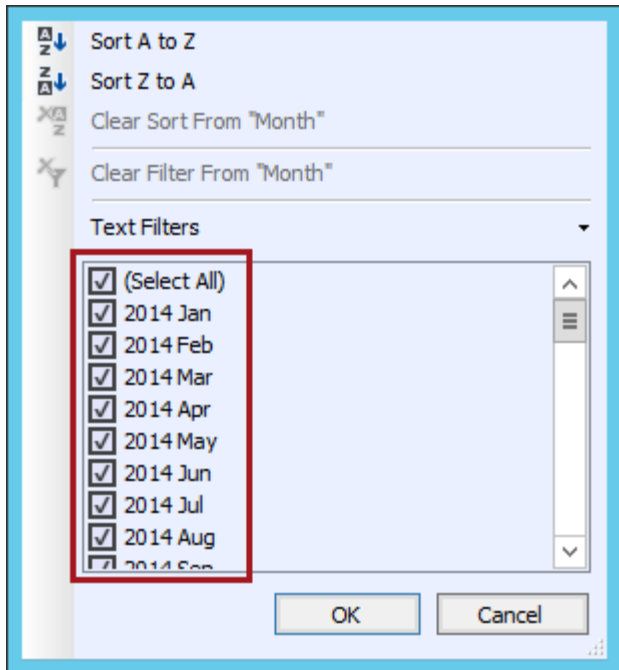
12. In the **Sort by Column** window, in the second dropdown list, select the **MonthKey** column.



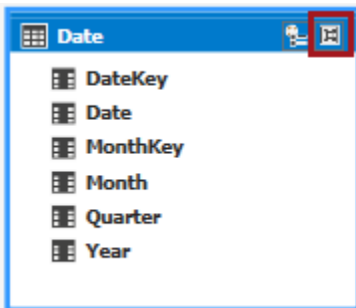
The **MonthKey** column values stored a number which is the year multiplied by 100 with the month number of year added (e.g., July 2016 is 201607).

13. Click **OK**.

14. In the **Month** column, review the distinct values found in the column, and notice that they are now sorted chronologically.



15. Switch to Diagram View.
16. Locate the **Date** table, hover over the top-right corner of the table, and then click the **Maximize** button.

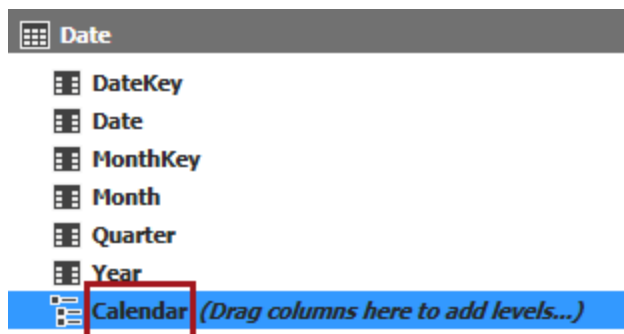


*Maximizing the table is a very convenient way to view its definition and to configure it.*

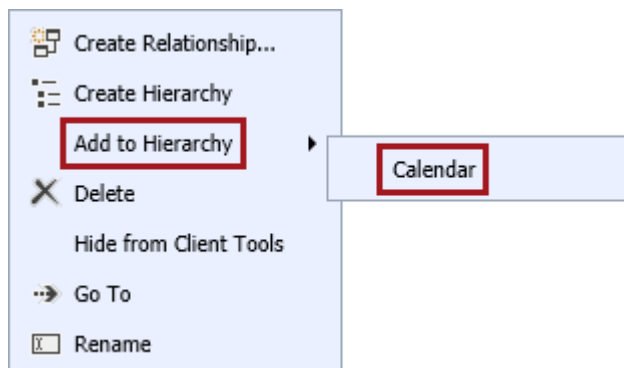
17. In the **Date** table, to create a hierarchy, at the top right corner, click **Create Hierarchy**.



18. When the hierarchy is added to the table, replace the default name with **Calendar**, and then press **Enter**.

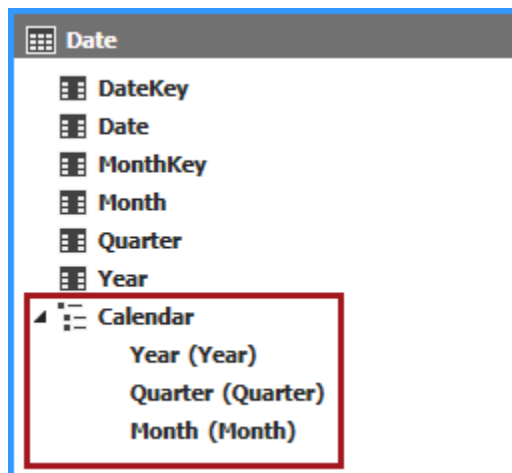


19. To add the **Year** column as the first level of the hierarchy, right-click the **Year** column, and then select **Add to Hierarchy | Calendar**.



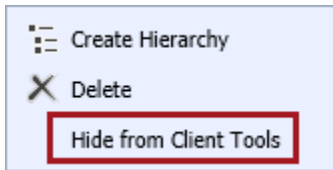
*Tip: It is also possible to drag and drop columns into the hierarchy.*

20. Add also the **Quarter** and **Month** columns to the hierarchy.
21. Verify that the **Calendar** hierarchy looks like the following.



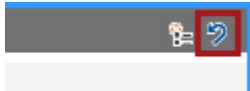
22. To hide all columns, select the **DateKey** column, and then while pressing the **Shift** key, select the **Year** column.

23. Right-click the selected columns, and then select **Hide from Client Tools**.

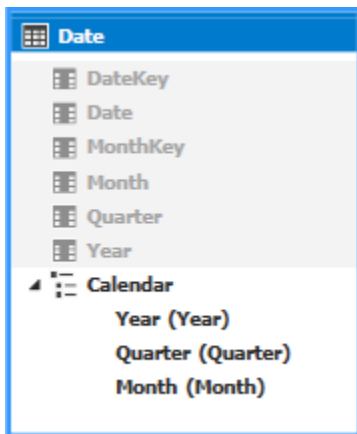


*Users exploring and querying this data model do not need to access the columns directly. Year, quarter and month members are now available from the **Calendar** hierarchy which is now the only visible object in this table.*

24. To minimize the table, at the top-right corner, click **Restore**.



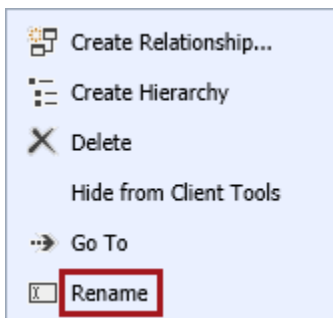
25. Resize the **Date** table so that all of the table content, including the hierarchy, is visible.
26. Verify that the **Date** table looks like the following.



## Enhancing the SalesTerritory Table

In this task, you will extend the **SalesTerritory** table with a **Regions** hierarchy.

1. Locate and maximize the **SalesTerritory** table.
2. Right-click the **SalesTerritoryRegion** column, and then select **Rename**.

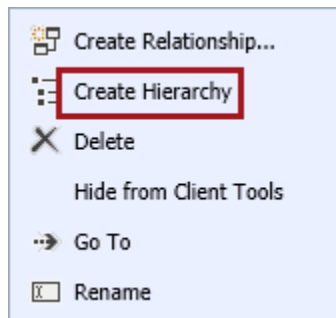




3. Replace the text with **Region**, and then press **Enter**.
4. Rename also the following two columns.

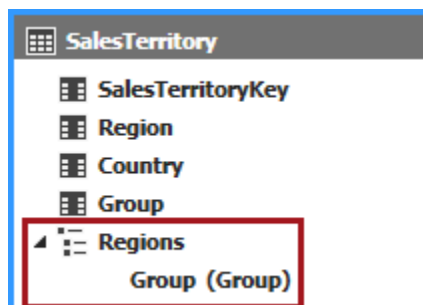
Column Name	New Column Name
SalesTerritoryCountry	Country
SalesTerritoryGroup	Group

5. As an alternate way to create a hierarchy, right-click the **Group** column, and then select **Create Hierarchy**.



The selected column, **Group**, will become the first level in the new hierarchy.

6. When the hierarchy is added to the table, replace the default name with **Regions**, and then press **Enter**.

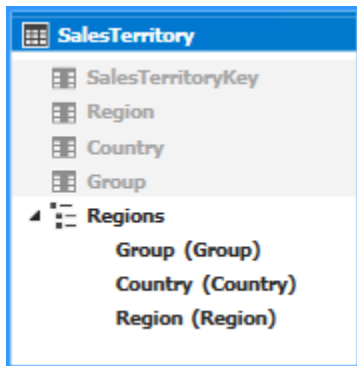


7. Add the **Country** and **Region** columns to the hierarchy.



8. Hide all columns for client tools.

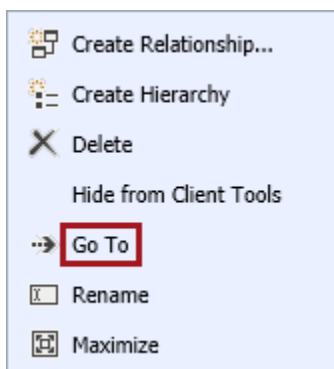
9. Minimize the **SalesTerritory** table.
10. Resize the **SalesTerritory** table so that all table content is visible.
11. Verify that the **SalesTerritory** table looks like the following.



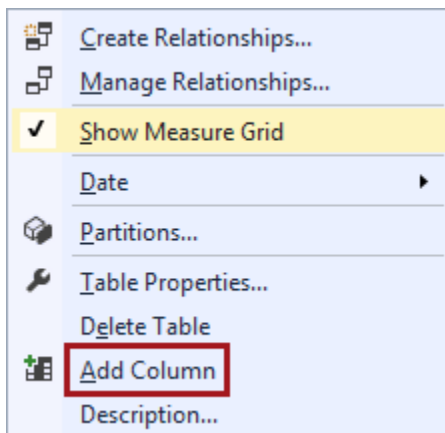
### Enhancing the Salesperson Table

In this task, you will enhance the **Salesperson** table with a calculated column to display the salesperson's full name.

1. To edit the **Salesperson** table in Grid View, right-click the header of the **Salesperson** table, and then select **Go to**.



2. To create a column, in **Tabular Model Explorer**, right-click the **Salesperson** table, and then select **Add Column**.



3. In the formula bar, enter the following formula.

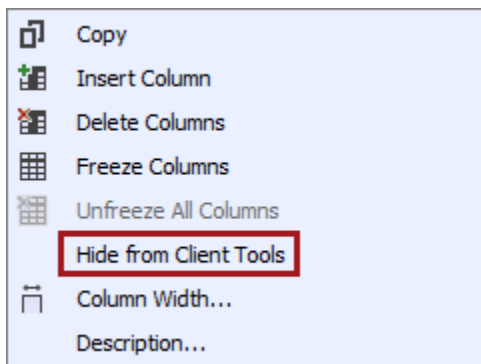


*To inject the column references into the expression, when you are ready to enter the column name, simply click anywhere inside the column.*

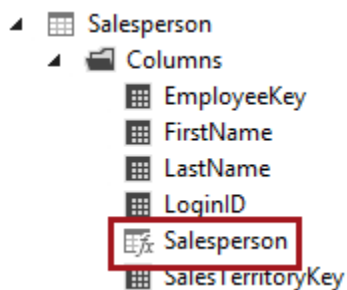
#### DAX

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

4. Press **Enter**.
5. Rename the new column as **Salesperson**.
6. To hide a range of columns, first select the **EmployeeKey** column, and then while pressing the **Shift** key, select the **LoginID** column.
7. Right-click the selected columns, and then select **Hide from Client Tools**.

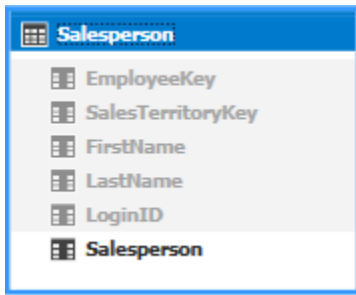


8. In **Tabular Model Explorer**, review the columns of the **Salesperson** table, and notice that calculated columns are adorned with a different icon.



9. Switch to Diagram View.

10. Verify that the **Salesperson** table looks like the following.



Column Name	Data Type	Nullable	Primary Key	Foreign Key
EmployeeKey	int	No	No	Yes
SalesTerritoryKey	int	No	No	Yes
FirstName	varchar	Yes	No	No
LastName	varchar	Yes	No	No
LoginID	varchar	Yes	No	No
Salesperson	varchar	Yes	No	No

## Enhancing the Product Table

In this task, you will enhance the **Product** table with calculated columns to introduce the related **Subcategory** and **Category** columns. You will then create the **Products** hierarchy.

1. Right-click the header of the **Product** table, and then select **Go to**.
2. Rename the **EnglishProductName** column as **Product**.
3. Switch to Data View, and then add a new column based on the following expression.

*For convenience, the calculated column expressions defined in this exercise can be copied from the **D:\SQL Server 2016 BI\Lab07\Assets\Snippets.txt** file.*

*To paste the clipboard content into the formula bar, right-click inside the formula box, select **Paste**, and then press **Enter**.*

### DAX

```
=RELATED(ProductSubcategory[EnglishProductSubcategoryName])
```

*This expression navigates the relationship to the **ProductSubcategory** table to retrieve the **EnglishProductSubcategoryName** column value.*

4. Rename the new column as **Subcategory**.
5. Add another column based on the following expression.

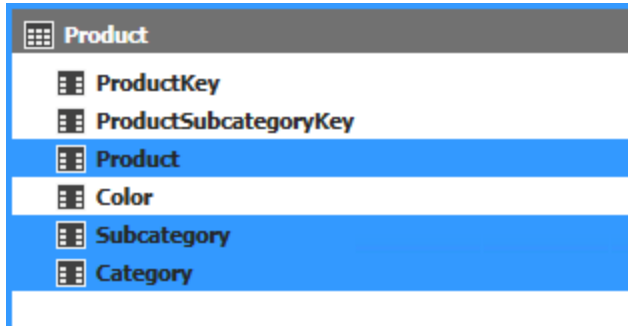
### DAX

```
=RELATED(ProductCategory[EnglishProductCategoryName])
```

*This expression navigates two relationships, first to the **ProductSubcategory** table, then the **ProductCategory** table, to lookup the **EnglishProductCategoryName** column value.*

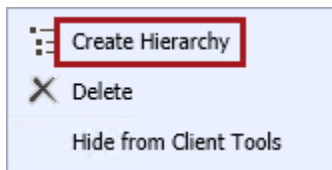
6. Rename the column as **Category**.
7. Switch to Diagram View.
8. To hide the **ProductCategory** table, right-click the **ProductCategory** table header, and then select **Hide from Client Tools**.
9. Hide also the **ProductSubcategory** table.
10. Locate and maximize the **Product** table.

11. As an alternate way to create a hierarchy, select the **Product** column, and then while pressing the **Ctrl** key, select also the **Subcategory** and **Category** columns.



12. Right-click the selected columns, and then select **Create Hierarchy**.

*Sometimes the command is not available on the context menu. In this case, re-select the columns and try again.*



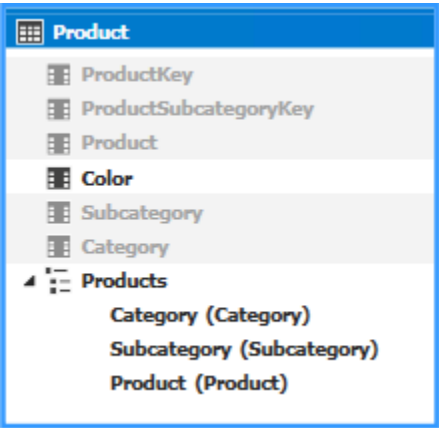
13. Set the name of the hierarchy as **Products**.

*A hierarchy can only add levels based on columns in the same table. This is the reason why you added calculated columns to introduce the related subcategory and category values.*

*The behavior for the multi-select method used in this step to create a hierarchy is slightly different from the incremental level addition methods used earlier. When using the multi-select method, the fields will be ordered based on cardinality (the field with fewer members will be the higher level in the hierarchy). This is to be interpreted to be a “suggested” order of levels only since it this may not necessarily be the correct order.*

14. Hide all columns from the client tools, except the **Color** column.
15. Minimize the **Product** table.
16. Resize the **Product** table so that all table content is visible.

17. Verify that the **Product** table looks like the following.



### Enhancing the Reseller Table

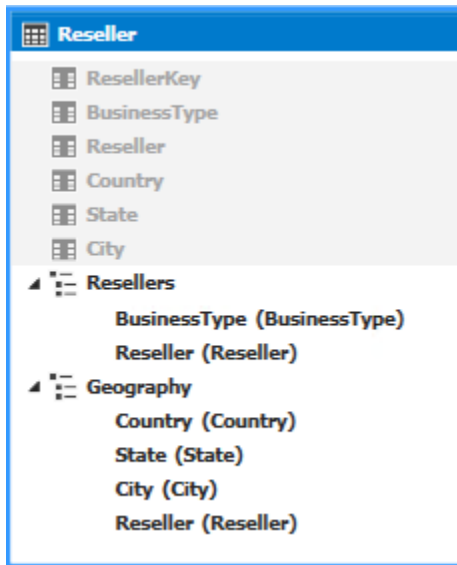
In this task, you will enhance the **Reseller** table by creating two hierarchies.

1. Locate and maximize the **Reseller** table.
2. Rename the following three columns.

Column Name	New Column Name
ResellerName	Reseller
EnglishCountryRegionName	Country
StateProvinceName	State

3. Create a hierarchy named **Resellers** based on the **BusinessType** and **Reseller** columns (in that order).
4. Create a second hierarchy named **Geography** based on the **Country**, **State**, **City** and **Reseller** columns (in that order).
5. Hide all columns from client tools.
6. Minimize the **Reseller** table.
7. Resize the **Reseller** table so that all table content is visible.

8. Verify that the **Reseller** table looks like the following.



ResellerKey	BusinessType	Reseller	Country	State	City
Resellers					
BusinessType (BusinessType)		Reseller (Reseller)			
Geography					
Country (Country)					
State (State)					
City (City)					
Reseller (Reseller)					

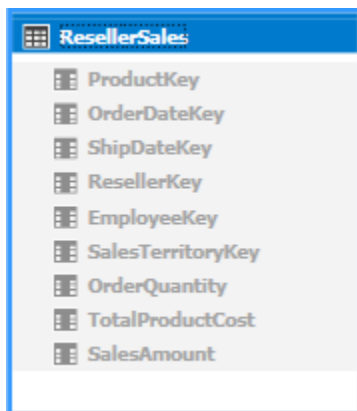
### Enhancing the ResellerSales Table

In this task, you will enhance the **ResellerSales** table by hiding all columns from client tools.

1. Locate the **ResellerSales** table.
2. Select and then hide all columns from client tools. (Do not to hide the table.)

*Generally, when enhancing the design of a fact table it is common to hide the dimension keys and measure columns, and then define explicit measures using aggregate functions. You will do this in the next exercise.*

3. Verify that the **ResellerSales** table looks like the following.



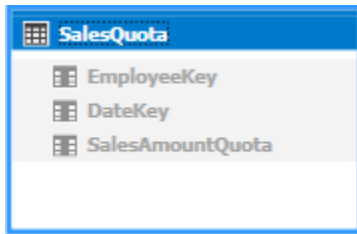
ProductKey	OrderDateKey	ShipDateKey	ResellerKey	EmployeeKey	SalesTerritoryKey	OrderQuantity	TotalProductCost	SalesAmount
------------	--------------	-------------	-------------	-------------	-------------------	---------------	------------------	-------------

### Enhancing the SalesQuota Table

In this task, you will enhance the **SalesQuota** table by hiding all columns from client tools.

1. Locate the **SalesQuota** table.
2. Select and then hide all columns from client tools. (Do not to hide the table.)

3. Verify that the **ResellerSales** table looks like the following.



The screenshot shows a table named 'SalesQuota' with three columns: 'EmployeeKey', 'DateKey', and 'SalesAmountQuota'. Each column has a key icon to its left, indicating it is a primary key. The table is displayed in a light gray box with a blue header bar.

SalesQuota	
EmployeeKey	
DateKey	
SalesAmountQuota	

4. To save the project, on the **File** menu, select **Save All**.



## 4: Adding Measures and a KPI

In this exercise, you will add measures to the **ResellerSales** and **SalesQuota** tables, and add the **Sales Performance** KPI.

Like calculated columns, measures are based on DAX expressions. However, they are evaluated within a query context of the report or PivotTable. Simple measures just aggregate column data. However, measure expressions can also be more sophisticated to modify and override filter context, and to perform time intelligence (e.g. YTD calculations).

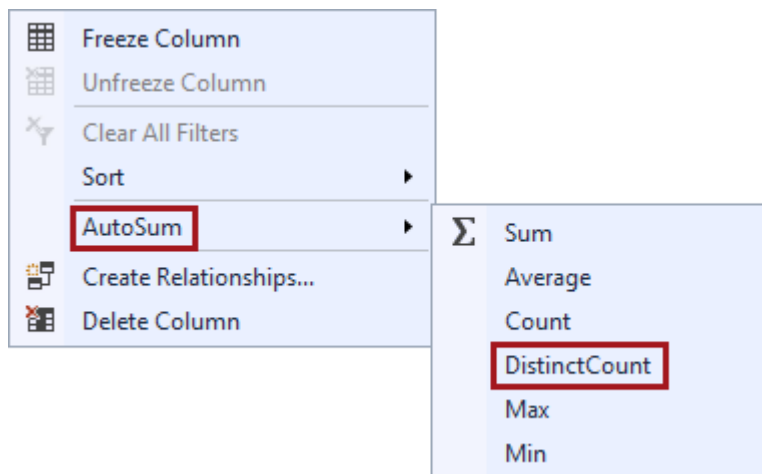
KPIs are based on measures and support the ability to define an object that delivers Value, Goal and Status metrics.

*To commence the lab starting from this exercise, in SSDT, open the **AdventureWorksBI.sln** file from the **D:\SQLServerBI\Lab07\Starter\Ex4** folder. To process the model, on the **Model** menu, select **Process | Process All**.*

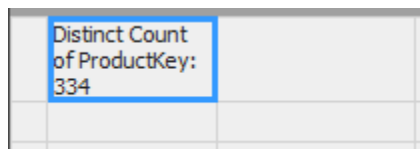
### Adding Measures to the ResellerSales Table

In this task, you will add and format measures to the **ResellerSales** table.

1. In Diagram View, right-click the header of the **ResellerSales** table, and then select **Go to**.
2. To add a measure, in **Tabular Model Explorer**, expand the **Product** table, and then expand the **Columns** folder.
3. Right-click the **ProductKey** column, and then select **AutoSum | DistinctCount**.



4. Notice the Measure Grid at the bottom of the table grid, and also the **Distinct Count of ProductKey** measure.



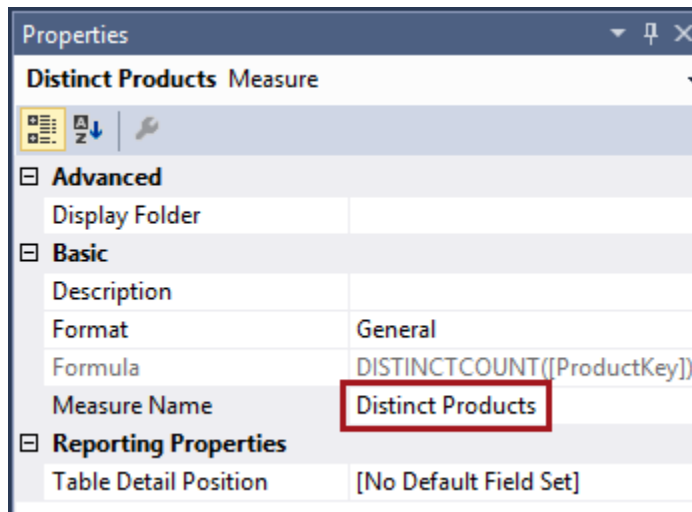
*Tip: You can right-click the table to show or hide the Measure Grid for each table.*

When adding a measure in this way it will be placed in the grid below the column it is based on. Note that the location of the measure within the Measure Grid does not matter. The column used to define the measure, or the sequence of measures within a column, does not impact on how it is evaluated, and you can move a measure to any location of the grid without impacting the formula.

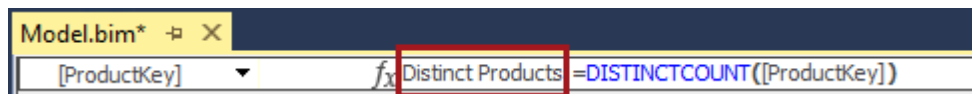
5. In the formula bar, notice the expression that defines the measure, and notice also that the measure name followed by a colon (:) precedes the expression.



6. In the Measure Grid, select the **Distinct Count of ProductKey** measure.
7. In the **Properties** window (located at the bottom-right), modify the **Measure Name** property to **Distinct Products**.



8. In the formula bar, notice the updated name that precedes the expression.



You can choose to modify the measure name in either location.

Note that measure names must be unique within the model. Also, it is not possible to have a measure with the same name as a column.

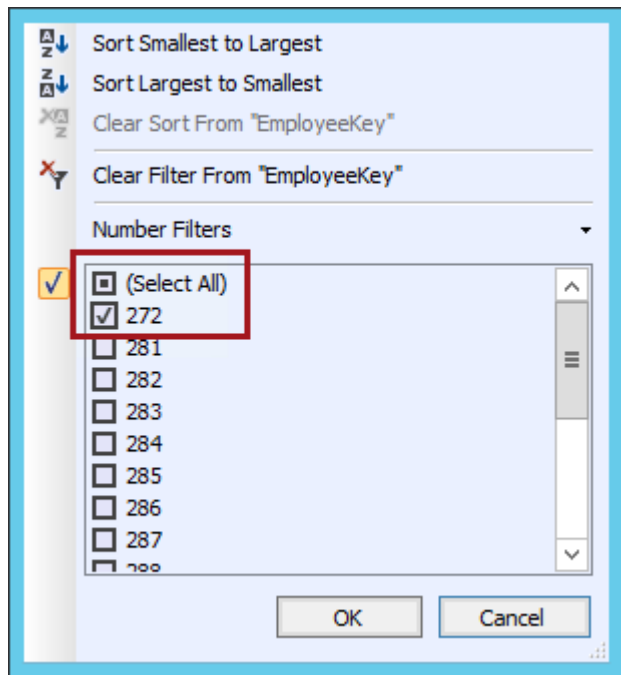
9. In the **Properties** window, modify the following properties of the **Distinct Products** measure.

Property	Value
Format	Whole Number
Show Thousand Separator	True

10. In the Measure Grid, notice that the measure also displays a value, **334**.

This represents the distinct number of products shown for all reseller sales.

11. Filter the **EmployeeKey** column to filter on the value **272**.

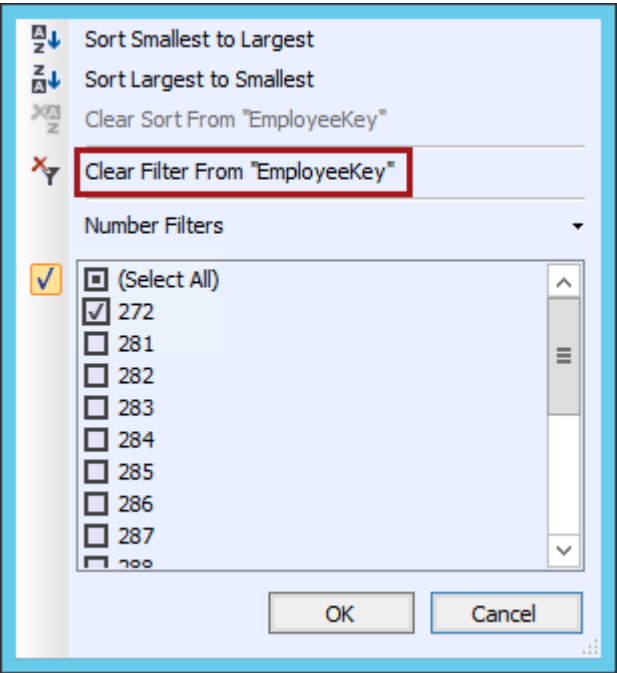


12. Notice that the value of the measure has changed to **278**.

Distinct Products:	278
--------------------	-----

*The table filters can help test the measure expressions, however filters applied to other tables in the model designer will not be considered. The true test of a measure is in a tool like an Excel PivotTable where filter context can be set by using columns from different tables.*

13. Clear the table filter.



14. To add a measures based on the **OrderQuantity**, **TotalProductCost** and **SalesAmount** columns, multi-select the columns, and then on the **Column** menu, select **AutoSum | Sum**.

15. In the Measure Grid, select the **Sum of OrderQuantity** measure.

16. In the **Properties** window, modify the following properties.

*Presently there is a bug which switches focus to a different object in the designer after applying a configuration. If this behavior occurs, be sure to save your solution and then close Visual Studio. Then, re-open the solution and continue.*

Property	Value
Measure Name	Quantity
Format	Whole Number
Show Thousand Separator	True

17. Rename the following two measures.

Measure Name	New Measure Name
Sum of TotalProductCost	Cost
Sum of SalesAmount	Sales

18. To add a measure based on an expression, in the Measure Grid, select the cell beneath the **Sales** measure.

Quantity: 214,378	Cost: \$79,980,114.38	Sales: \$80,450,596.98

19. In the formula bar, enter the following expression.

*For convenience, the measure expressions defined in this exercise can be copied from the **D:\SQL Server 2016 BI\Lab07\Assets\Snippets.txt** file.*

**DAX**

```
Profit:=[Sales] - [Cost]
```

20. In the **Properties** window, set the **Format** property to **Currency**.
21. Add the following measure beneath the last, and format the measure as **Percentage**.

**DAX**

```
Profit%:=DIVIDE([Profit], [Sales])
```

*The **DIVIDE** function divides two expressions, providing that the second argument results in a non-zero number. If the second argument results in zero or blank (missing), then the function will return blank.*

22. Add the following measure beneath the last, and format the measure as **Currency**.

**DAX**

```
Sales YTD (Ordered):=TOTALYTD([Sales], 'Date'[Date])
```

*This measure uses the **TOTALYTD** function to calculate year-to-date sales values based on the dates in the related **Date** table. Recall that there are two relationships between the **ResellerSales** and **Date** tables. The active relationship is based on the **OrderDateKey** and the inactive relationship is based on the **ShipDateKey**.*

*The active relationship is always used by default. You will create one more measure in this table to also calculate year-to-date sales values using the **ShipDateKey** relationship.*

*Also, when you create this measure, in the calculation area, you will see the result of **(blank)**. A value cannot be displayed for this measure as it must be filtered by the date table. You will test the measure by using an Excel PivotTable in Exercise 5.*

23. Add the following measure beneath the last, and format the measure as **Currency**.

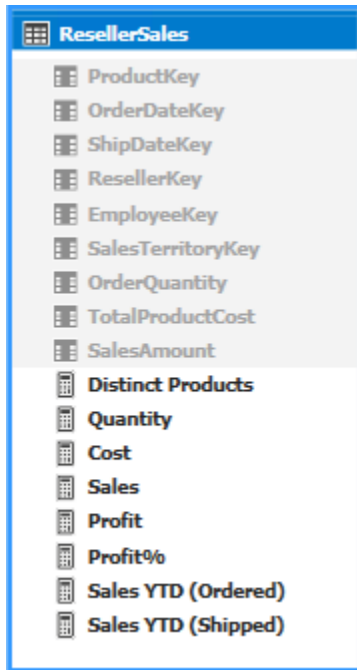
**DAX**

```
Sales YTD (Shipped):=TOTALYTD(CALCULATE([Sales],  
USERRELATIONSHIP(ResellerSales[ShipDateKey], 'Date'[DateKey])), 'Date'[Date])
```

*The **CALCULATE** function is used to force the evaluation of the **Sales** measure in the context of the relationship between the **ShipDateKey** and **Date** table's **DateKey** column. In other words, you are specifying a formula that will navigate via the inactive relationship.*

24. To review the table definition, switch to Diagram View.

25. Verify that the **ResellerSales** table looks like the following.



ProductKey	OrderDateKey	ShipDateKey	ResellerKey	EmployeeKey	SalesTerritoryKey	OrderQuantity	TotalProductCost	SalesAmount	Distinct Products	Quantity	Cost	Sales	Profit	Profit%	Sales YTD (Ordered)	Sales YTD (Shipped)
------------	--------------	-------------	-------------	-------------	-------------------	---------------	------------------	-------------	-------------------	----------	------	-------	--------	---------	---------------------	---------------------

### Adding Measures to the SalesQuota Table

In this task, you will add and format measures to the **SalesQuota** table.

1. Locate and right-click the **SalesQuota** table, and then select **Go to**.
2. In the Measure Grid, in any cell, add the following measure, and then format the measure as **Currency**.

#### DAX

```
Quota:=IF(ISFILTERED('Date'[Month]), CALCULATE(SUM([SalesAmountQuota]),  
ALL('Date'[Month])) / 3, SUM([SalesAmountQuota]))
```

*The data in the **SalesQuota** table is stored at quarter granularity, and yet the data in the **ResellerSales** table is stored at month granularity.*

*This expression is using the IF function test whether only a single month is being used to filter the measure. If it is a single month, then the CALCULATE function is used to evaluate the sum of the **SalesAmountQuota** column for all months. As the month is only accessible by navigating the **Calendar** hierarchy, it is known that specific Year and Quarter will be in context. Essentially the CALCULATE function is returning the value for the quarter that the month belongs to. This value is then divided by three.*

*This expression is only accurate if each quarter contains three months, which is the case with the data loaded into the **Date** table in Exercise 2.*

3. Add the following measure beneath the last, and format the measure as **Currency**.

#### DAX

```
Variance:=[Sales] - [Quota]
```

4. Add the following measure beneath the last, and format the measure as **Percentage**.

#### DAX

```
Variance%:=DIVIDE([Variance], [Quota])
```

### Adding a KPI to the SalesQuota Table

In this task, you will add the **Sales Performance** KPI to the **SalesQuota** table.

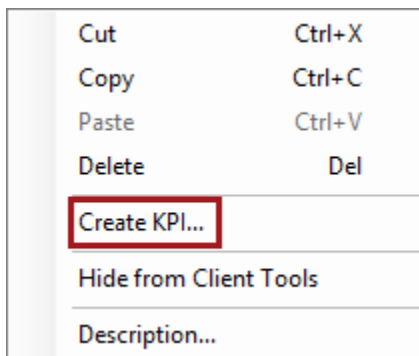
1. Add the following measure beneath the last, and format the measure as **Currency**.

#### DAX

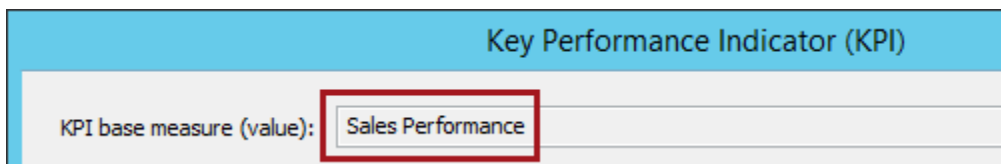
```
Sales Performance:=[Sales]
```

*This new measure is a direct reference to the **Sales** measure. It will become the base measure used to create a KPI.*

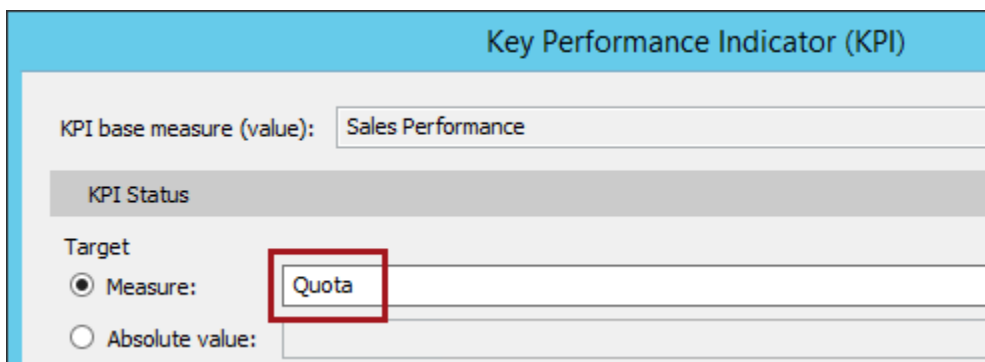
2. Right-click the **Sales Performance** measure, and then select **Create KPI**.



3. In the **Key Performance Indicator (KPI)** window, notice that the KPI base measure (value) is based on the **Sales Performance** measure.



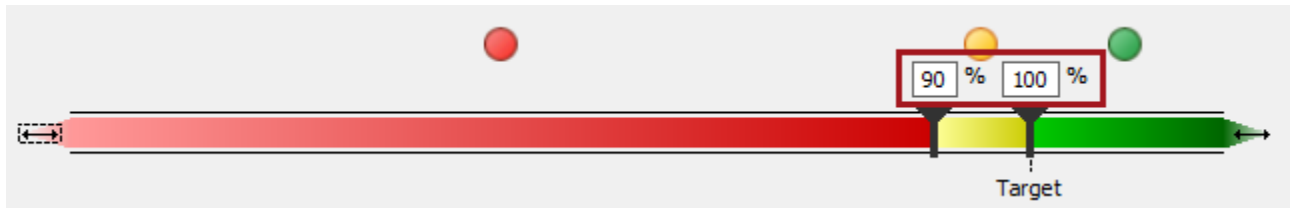
4. In the **KPI Status** section, in the **Measure** dropdown list, select **Quota**.



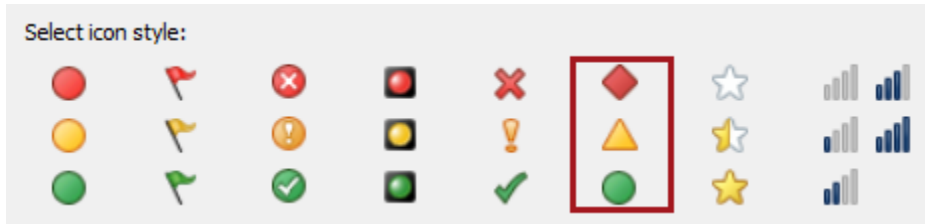
*This configuration will produce a status based on the ratio of **Sales** over **Quota**.*

A value of 100% or more means that the sales are on, or exceeding, target. A value less than 0% means that sales are not meeting target.

- Set the threshold box values to **90** and **100**.



- In the icon style gallery, select the sixth from the left.

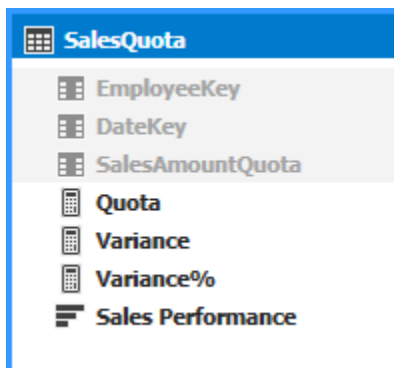


*Tip: The combination of shape and color is helpful for users with visual impairment.*

- Click **OK**.
- In the Measure Grid, notice the icon added to the **Sales Performance** measure to denote that it has been configured as a KPI.

Quota:	\$113,395,000.00
Variance:	(\$32,944,403.02)
Variance%:	-29.05 %
Sales Performance:	\$80,450,596.98

- To review the table definition, switch to Diagram View.
- Verify that the **SalesQuota** table looks like the following.



- To save the project, on the **File** menu, select **Save All**.



# 5: Analyzing the Model in the Excel Client

In this exercise, you will explore and validate the completed model in Excel.

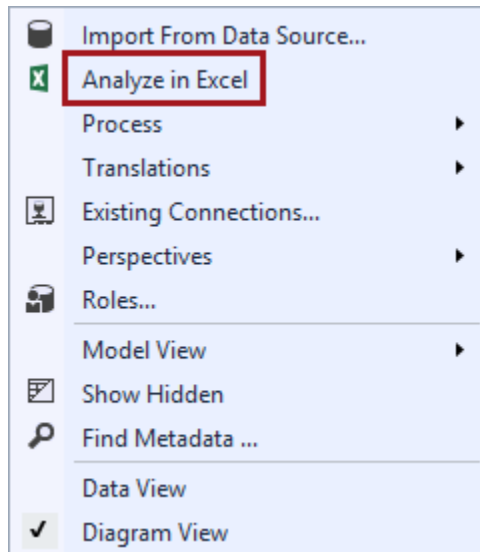
When developing a model in SSDT, a workspace database is created on the nominated workspace server. When analyzing the model in Excel, a workbook connection connects to this database.

*To commence the lab starting from this exercise, in SSDT, open the **AdventureWorksBI.sln** file from the **D:\SQLServerBI\Lab07\Starter\Ex5** folder. To process the model, on the **Model** menu, select **Process / Process All**.*

## Exploring and Validating the Model in the Excel

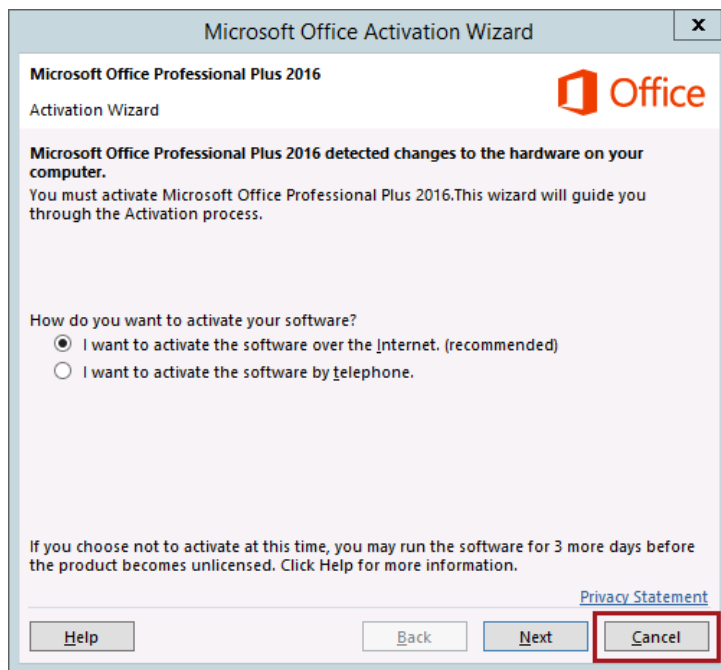
In this task, you will explore and validate the model in Excel by producing and interacting with a PivotTable report.

1. In **Tabular Model Explorer**, right-click the **Sales Analysis** model, and then select **Analyze in Excel**.

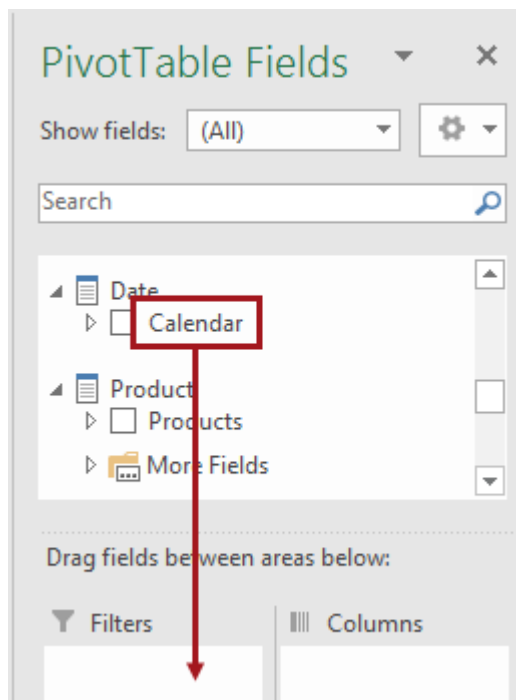


2. In the **Analyze in Excel** window, to connect to the model using your identity, click **OK**.

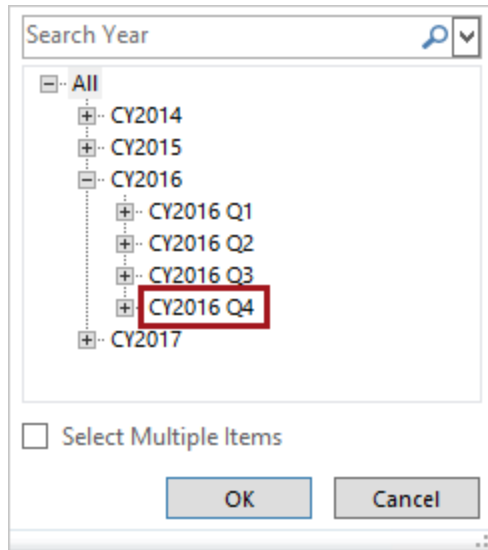
3. If prompted to activate Office, click **Cancel**.



4. In Excel, in the **PivotTable Fields** pane (located at the right), review the model structure consisting of measures, KPIs, tables and hierarchies.
5. To test the model, in the **PivotTable Fields** pane, from inside the **Date** table, drag the **Calendar** hierarchy to the **Filters** drop zone.



- In the **Calendar** PivotTable filter (cell **B1**), click the down-arrow, expand the **All | CY2016** members, and then select the **CY2016 Q4** member.



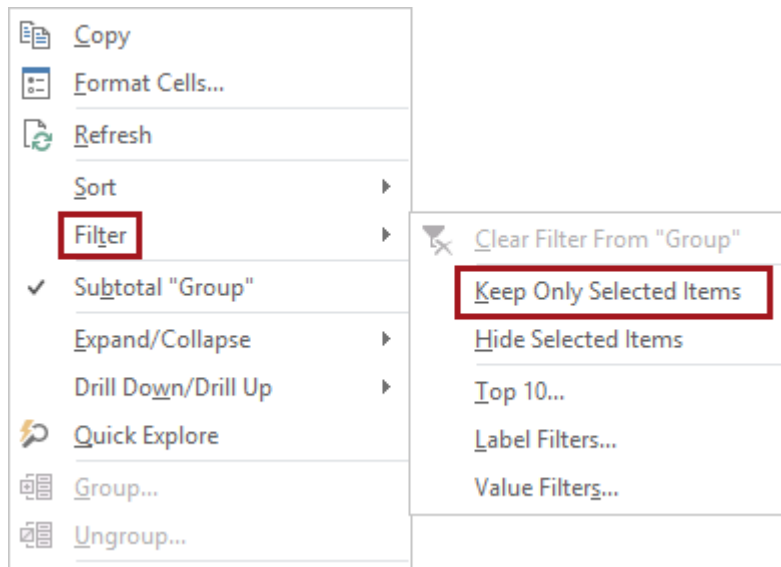
- Click **OK**.
- In the **PivotTable Fields** pane, from inside the **SalesTerritory** table, select the **Regions** hierarchy to add it to the **Rows** drop zone.
- In the **PivotTable Fields** pane, in this order, select the following fields.

Type	Field
Measure	ResellerSales   Sales
Measure	SalesQuota   Quota
Measure	SalesQuota   Variance
KPI	KPIs   Sales Performance   Status
Measure	ResellerSales   Profit%
Measure	ResellerSales   Distinct Products

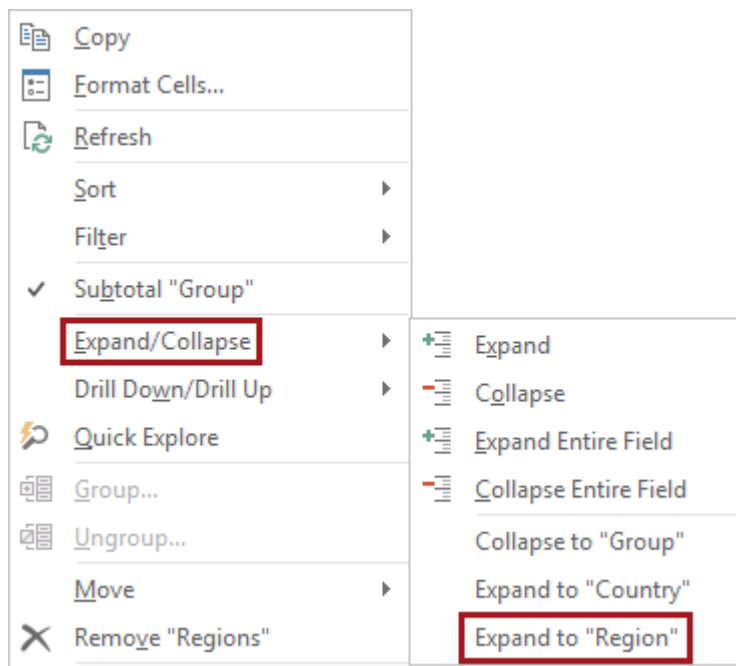
- Verify that the PivotTable report looks like the following.

	A	B	C	D	E	F	G
1	Calendar	CY2016 Q4					
2							
3	Row Labels	Sales	Quota	Variance	Sales Performance Status	Profit%	Distinct Products
4	Europe	\$2,300,279.97	\$2,425,000.00	(\$124,720.03)		0.36 %	147
5	NA	\$200,894.23	\$170,000.00	\$30,894.23		1.19 %	101
6	North America	\$6,051,229.01	\$5,900,000.00	\$151,229.01		-0.70 %	149
7	Pacific	\$385,157.65		\$385,157.65		-1.19 %	91
8	Grand Total	\$8,937,560.86	\$8,495,000.00	\$442,560.86		-0.40 %	149

11. To display only the **North America** member, right-click the **North America** member (cell **A6**), and then select **Filter | Keep Only Selected Items**.

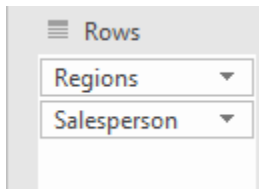


12. To expand all of the **Regions** hierarchy levels, right-click the **North America** member (cell **A4**), and then select **Expand/Collapse | Expand to "Region"**.



13. Notice that the **Northwest** region is under performing.
14. Right-click the **Northwest** member (cell **A10**), and then select **Filter | Keep Only Selected Items**.

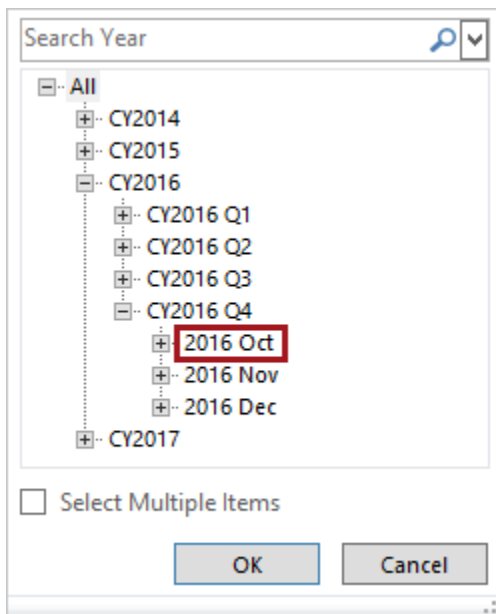
15. To introduce the salespeople to the report, in the **PivotTable Fields** pane, from inside the **Salesperson** table, drag the **Salesperson** hierarchy into the **Rows** drop zone, directly beneath the **Regions** hierarchy.



16. Verify that the **PivotTable Fields** pane looks like the following.

	A	B	C	D	E	F	G
1	Calendar	CY2016 Q4					
2							
3	Row Labels	Sales	Quota	Variance	Sales Performance Status	Profit%	Distinct Products
4	North America	\$1,193,395.54	\$1,350,000.00	(\$156,604.46)		-0.29 %	145
5	United States	\$1,193,395.54	\$1,350,000.00	(\$156,604.46)		-0.29 %	145
6	Northwest	\$1,193,395.54	\$1,350,000.00	(\$156,604.46)		-0.29 %	145
7	David Campbell	\$380,814.59	\$300,000.00	\$80,814.59		1.28 %	131
8	Pamela Ansman-Wolfe	\$362,722.07	\$600,000.00	(\$237,277.93)		1.63 %	105
9	Tete Mensa-Annan	\$449,858.88	\$450,000.00	(\$141.12)		-3.16 %	124
10	Grand Total	\$1,193,395.54	\$1,350,000.00	(\$156,604.46)		-0.29 %	145

17. Notice the **CY2016 Q4** quarter **Quota** values for the three **Northwest** salespeople.
18. To filter at month level, in the **Calendar** report filter (cell **B1**), click the down-arrow, expand the selected **CY2016 Q4** member, notice the chronologically ordered months, select the **2016 Oct** member.



19. Click **OK**.

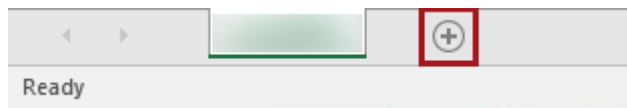
20. Now notice that the month level **Quota** values are one third of the quarter level values.

	A	B	C	D	E	F	G
1	Calendar	2016 Oct					
2							
3	Row Labels	Sales	Quota	Variance	Sales Performance Status	Profit%	Distinct Products
4	North America	\$385,824.68	\$450,000.00	(\$64,175.32)		0.94 %	118
5	United States	\$385,824.68	\$450,000.00	(\$64,175.32)		0.94 %	118
6	Northwest	\$385,824.68	\$450,000.00	(\$64,175.32)		0.94 %	118
7	David Campbell	\$128,326.97	\$100,000.00	\$28,326.97		2.81 %	74
8	Pamela Ansman-Wolfe	\$154,571.13	\$200,000.00	(\$45,428.87)		1.84 %	68
9	Tete Mensa-Annan	\$102,926.58	\$150,000.00	(\$47,073.42)		-2.75 %	43
10	Grand Total	\$385,824.68	\$450,000.00	(\$64,175.32)		0.94 %	118

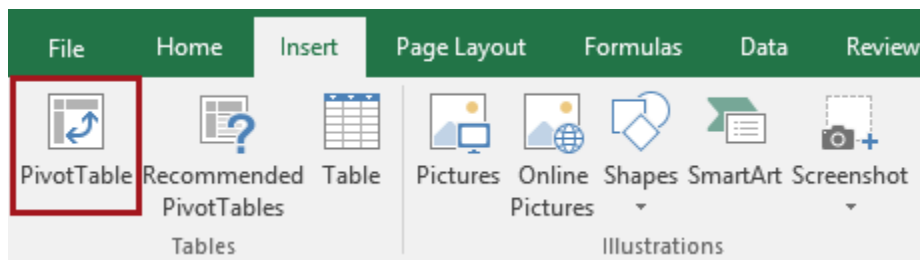
## Testing the YTD Measures

In this task, you will create a report using the two YTD measures created in Exercise 4.

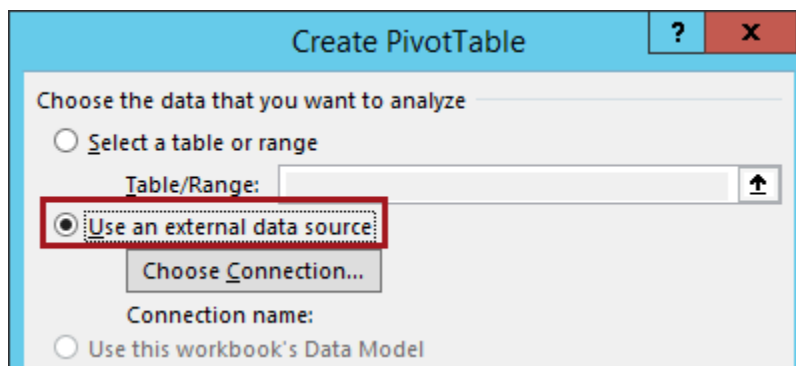
1. To create a new worksheet, click the **New Worksheet** button.



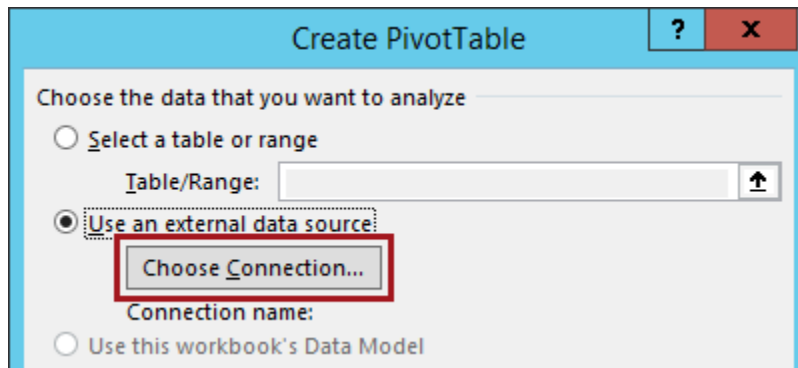
2. To create a new PivotTable report, on the **Insert** ribbon tab, from inside the **Tables** group, click **PivotTable**.



3. In the **Create PivotTable** window, select the **Use an External Data Source** option.



- Click **Choose Connection**.



- In the **Existing Connections** window, accept the default selection (that represents the connection to the model), and then click **Open**.
- In the **Create PivotTable** window, click **OK**.
- In the **PivotTable Fields** pane, from inside the **Date** table, drag the **Calendar** hierarchy to the **Rows** drop zone.
- In the PivotTable report, right-click the **CY2015** member (cell **A3**), and then select **Expand/Collapse | Expand to "Month"**.
- To hide the quarter members, right-click any month member, and then select **Show/Hide Fields | Quarter**.
- In the **PivotTable Fields** pane, from inside the **ResellerSales** table, select the **Sales YTD (Ordered)** and also the **Sales YTD (Shipped)** measures.

	A	B	C
1	Row Labels	Sales YTD (Ordered)	Sales YTD (Shipped)
2	+ CY2014	\$16,288,441.77	\$15,682,252.86
3	- CY2015	\$27,921,670.52	\$27,656,682.28
4	2015 Jan	\$2,393,689.53	\$1,996,250.03
5	2015 Feb	\$5,994,880.24	\$5,050,928.80
6	2015 Mar	\$8,880,239.44	\$8,152,028.64
7	2015 Apr	\$10,682,393.65	\$10,539,749.98
8	2015 May	\$13,736,209.98	\$12,996,957.02
9	2015 Jun	\$15,921,423.19	\$15,517,798.15
10	2015 Jul	\$17,238,965.03	\$17,276,532.52
11	2015 Aug	\$19,623,811.62	\$19,234,493.73
12	2015 Sep	\$21,187,766.70	\$21,087,603.55
13	2015 Oct	\$23,053,045.13	\$22,814,059.00
14	2015 Nov	\$25,933,797.81	\$25,377,407.82
15	2015 Dec	\$27,921,670.52	\$27,656,682.28
16	+ CY2016	\$36,240,484.70	\$35,475,524.74
17	+ CY2017		\$1,636,137.11
18	Grand Total		\$1,636,137.11

11. Notice the accumulation of sales over months, and the different values for each measure.

## Finishing Up

In this task, you will close Excel.

1. To close Excel, at the top-right corner, click **X**.



2. When prompted to save the changes, click **Don't Save**.



# 6: Defining Perspectives, Partitions and a Role

In this exercise, you will define perspectives, partitions and a role.

Perspectives allow delivering different versions of the data model that show/hide different fields. This is particularly useful when a single model covers multiple subject areas. Rather than connecting to the entire model, a user can choose to connect to a perspective and thereby be presented with a subject-specific subset of the model. Note that perspectives are not security mechanisms.

In this data model you will create two perspectives. One to show sales-specific fields, and a second to show only monitoring-specific fields.

By default, a table is based on a single partition. However, it is possible to create additional partitions to simplify the management and data refresh of the table. Typically, fact table partitions are based on time periods, and a clear advantage of implementing multiple partitions is that only the current period's partition needs to be refreshed.

You will create four partitions in the **ResellerSales** table for each calendar year from 2014 to 2017.

Roles grant permissions to users, and can support row-level security. In fact, a DAX expression can be used to evaluate which rows in a table are available to the role.

You will create a role to allow salespeople to read the model, but only allow them to see data related to other salespeople in their own region.

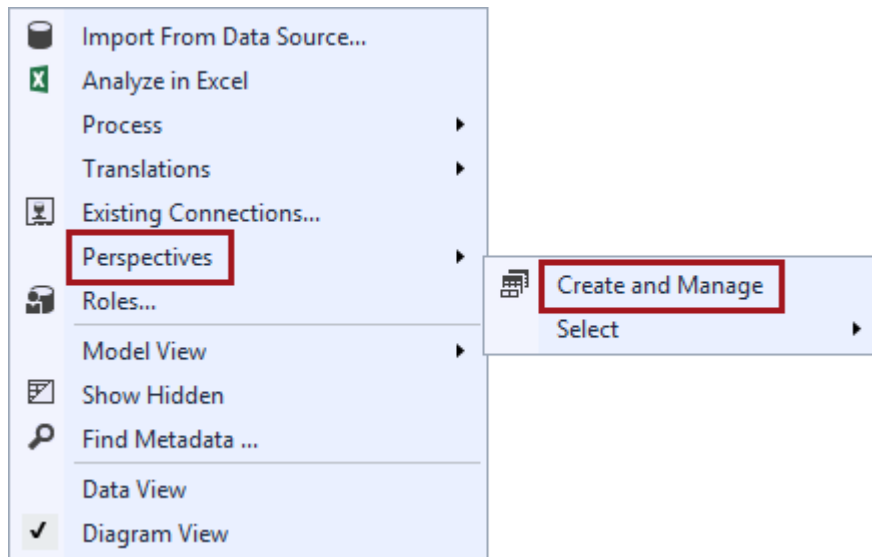
*To commence the lab starting from this exercise, in SSDT, open the **AdventureWorksBI.sln** file from the **D:\SQL Server BI\Lab07\Starter\Ex6** folder. To process the model, on the **Model** menu, select **Process | Process All**.*

## Defining Model Perspectives

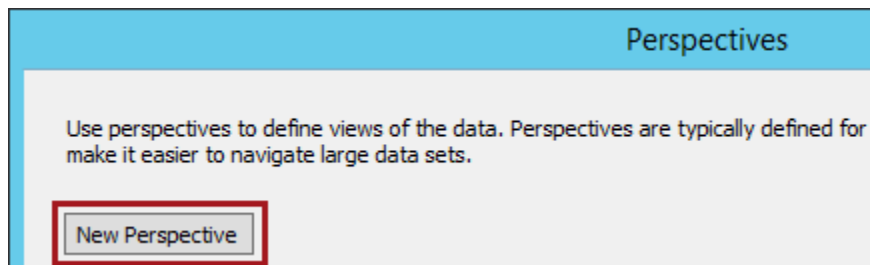
In this task, you will define two model perspectives, namely Sales and Monitoring.

1. Switch to SQL Server Data Tools (Visual Studio).

2. In **Tabular Model Explorer**, right-click the **Sales Analysis** model, and then select **Perspectives | Create and Manage**.



3. In the **Perspectives** window, click **New Perspective**.



4. In the new perspective column, replace the name with **Sales**, and then press **Enter**.
5. Collapse the **Date** table.

Fields	Sales
- Tables	<input type="checkbox"/>
- Date *	<input type="checkbox"/>
Date *	<input type="checkbox"/>
DateKey *	<input type="checkbox"/>
Month *	<input type="checkbox"/>
MonthKey *	<input type="checkbox"/>
Quarter *	<input type="checkbox"/>
Year *	<input type="checkbox"/>
Calendar	<input type="checkbox"/>
+ Product	<input type="checkbox"/>
+ ProductCategory	<input type="checkbox"/>

6. Select only the following six tables.

**Table**

---

Date

Product

Reseller

ResellerSales

Salesperson

SalesTerritory

7. To add another perspective, click **New Perspective**.

8. Name the new perspective **Monitoring**, and then select only the following five tables.

**Table**

---

Date

ResellerSales

Salesperson

SalesQuota

SalesTerritory

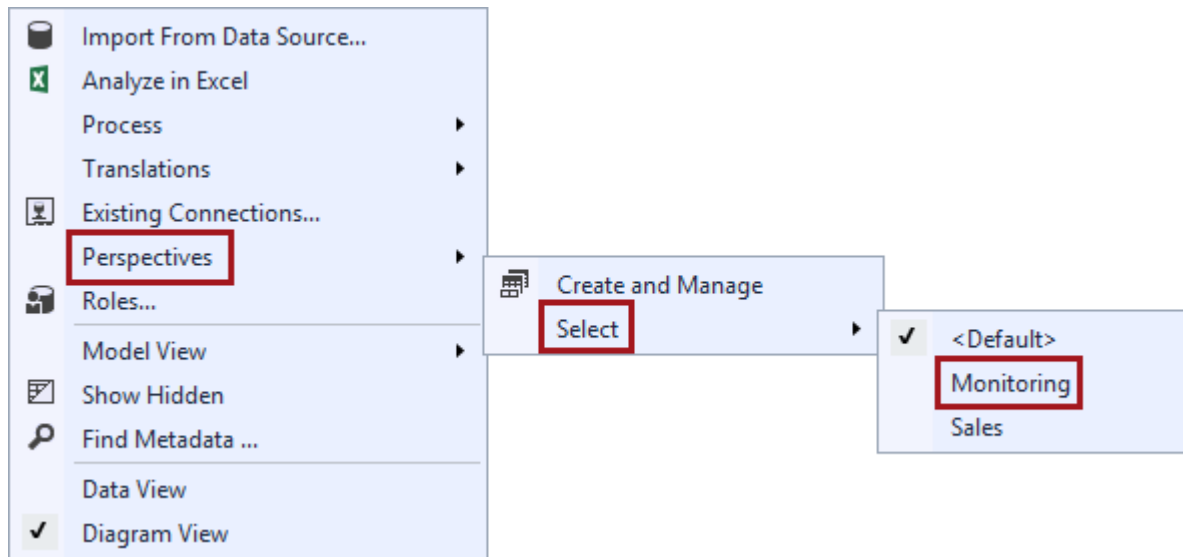
9. Verify that the perspectives look like the following.

Fields	Sales	Monitoring
- Tables	<input type="checkbox"/>	<input type="checkbox"/>
+ Date *	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
+ Product	<input checked="" type="checkbox"/>	<input type="checkbox"/>
+ ProductCategory	<input type="checkbox"/>	<input type="checkbox"/>
+ ProductSubcategory	<input type="checkbox"/>	<input type="checkbox"/>
+ Reseller *	<input checked="" type="checkbox"/>	<input type="checkbox"/>
+ ResellerSales *	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
+ Salesperson	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
+ SalesQuota *	<input type="checkbox"/>	<input checked="" type="checkbox"/>
+ SalesTerritory *	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

10. Click **OK**.

11. Ensure the model designer is in Diagram View.

12. In **Tabular Model Explorer**, right-click the **Sales Analysis** model, and then select **Perspectives | Select | Monitoring**.



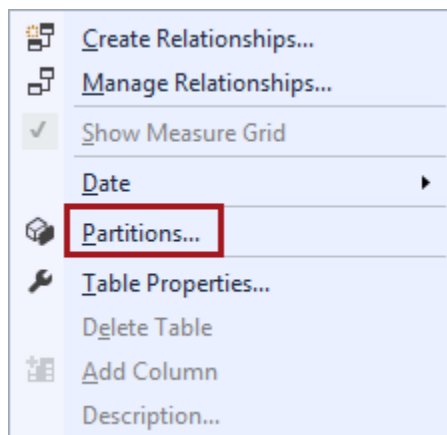
13. Review the subset of tables in the diagram.

You will use the **Monitoring** perspective to analyse in Excel later in this exercise.

### Defining Partitions for the ResellerSales Table

In this task, you will define partitions for the **ResellerSales** table to allow its data to be refreshed for individual years.

1. In **Tabular Model Explorer**, right-click the **ResellerSales** table, and then select **Partitions**.

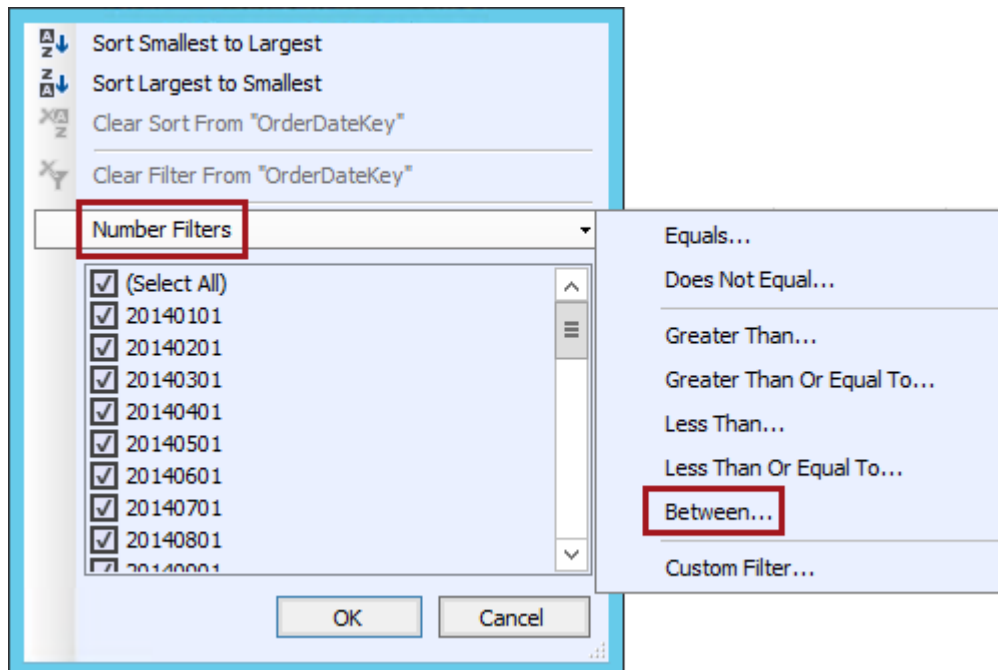


2. In the **Partition Manager** window, notice that there is one partition named **ResellerSales**.

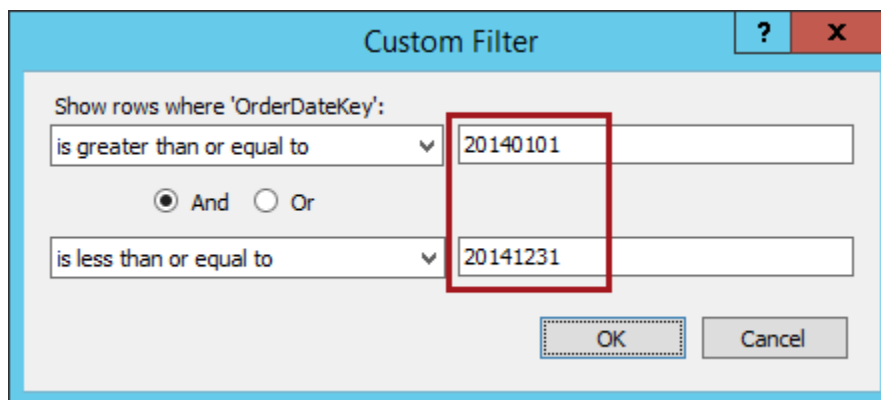
Partition Name	Last Processed
ResellerSales	

3. In the **Partition Name** box, modify the text to **ResellerSales CY2014**.

4. In the **OrderDateKey** column header, click the down-arrow, and then select **Number Filters | Between**.



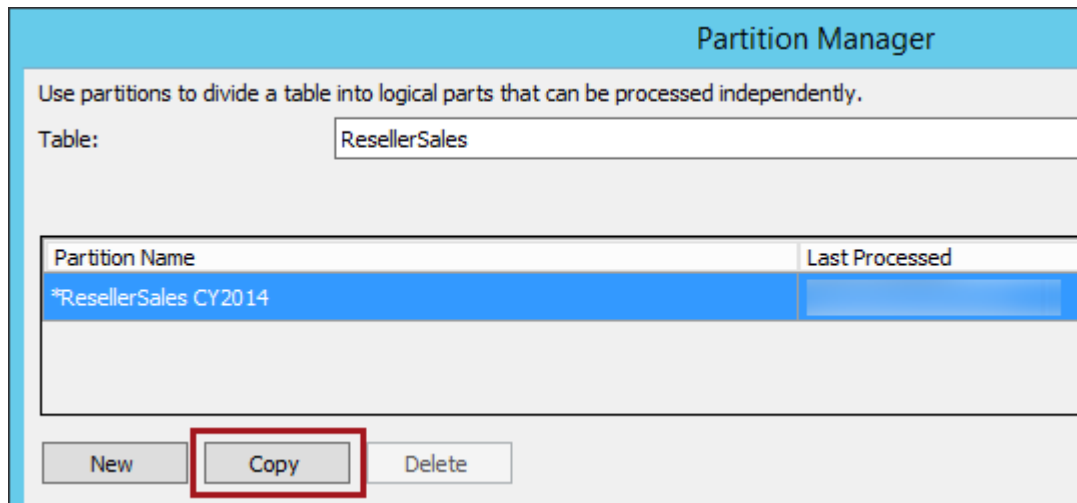
5. In the **Custom Filter** window, enter the values **20140101** and **20141231**.



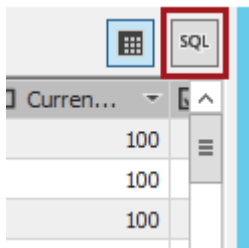
6. Click **OK**.

*Defining multiple partitions for a table allows the efficient removal of data from the data model, and also allows refreshing data at partition level. There is no need to refresh partitions, particularly historical ones, where the source data has not changed.*

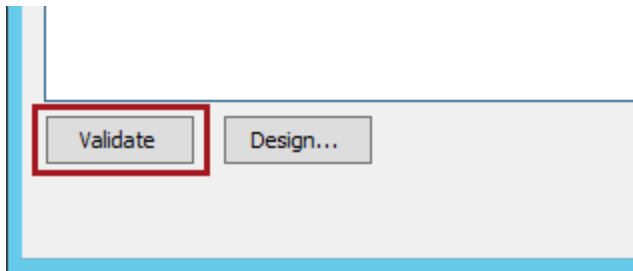
7. To add a new partition based on the existing one, click **Copy**.



8. In the **Partition Name** box, modify the text to **ResellerSales CY2015**.
9. To switch to the query editor, click the **Query Editor** button (located halfway down the right side of the window.)



10. Modify the **OrderDateKey** values in the WHERE clause to **20150101** and **20151231**, respectively.
11. To validate the query, click **Validate**.

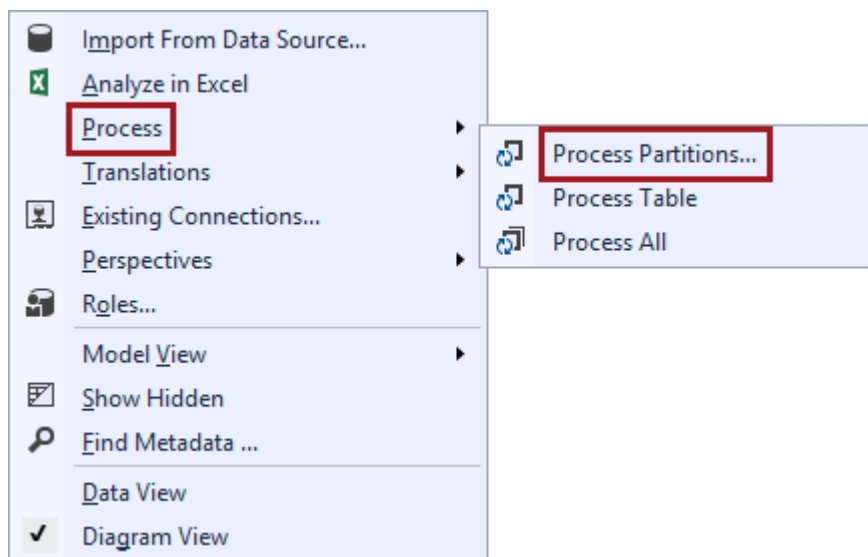


12. Verify that the SQL statement is valid.
13. Create two additional partitions based on the following.

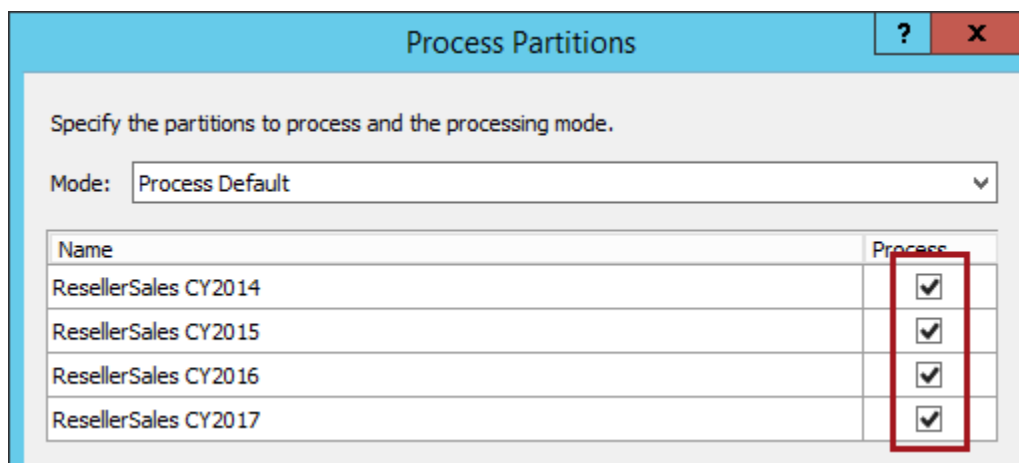
Partition Name	OrderDateKey From	OrderDateKey To
ResellerSales CY2016	20160101	20161231
ResellerSales CY2017	20170101	20171231

14. Click **OK**.

15. To process the **ResellerSales** table, in the model diagram, ensure that the **ResellerSales** table is selected.
16. In **Tabular Model Explorer**, right-click the **Sales Analysis** model, and then select **Process | Process Partitions**.








17. In the **Process Partitions** window, select all four partitions.



18. Click **OK**.

19. Verify that the status of each work item looks like the following.

 <b>Success</b>		4 Total	0 Cancelled
		4 Success	0 Error
Details:			
	Work Item	Status	Message
	ResellerSales CY2014	Success. 8,459 rows transferred.	
	ResellerSales CY2015	Success. 21,670 rows transferred.	
	ResellerSales CY2016	Success. 30,726 rows transferred.	
	ResellerSales CY2017	Success. 0 rows transferred.	

*If you add up the rows for each partition they come to 60,855—the number contained in the original partition.*

20. Click **Close**.

*Partitions can also be managed by administrators using the Object Explorer in SQL Server Management Studio. In addition to the ability to create partitions, delete partitions, define the partition queries and refresh partitions, it is also possible to merge partitions.*

*It is not uncommon that partitions are created, defined and managed by administrators using SQL Server Management Studio.*

21. To save the project, on the **File** menu, select **Save All**.

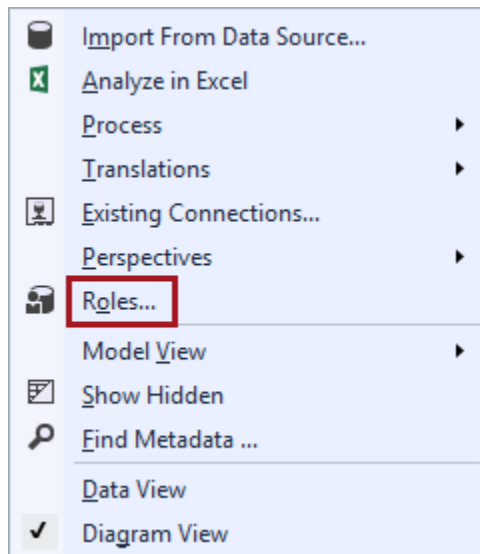
## Defining Role-Based Security

In this task, you will define a role to limit retrieving data for the sales territory to which the salesperson is assigned.

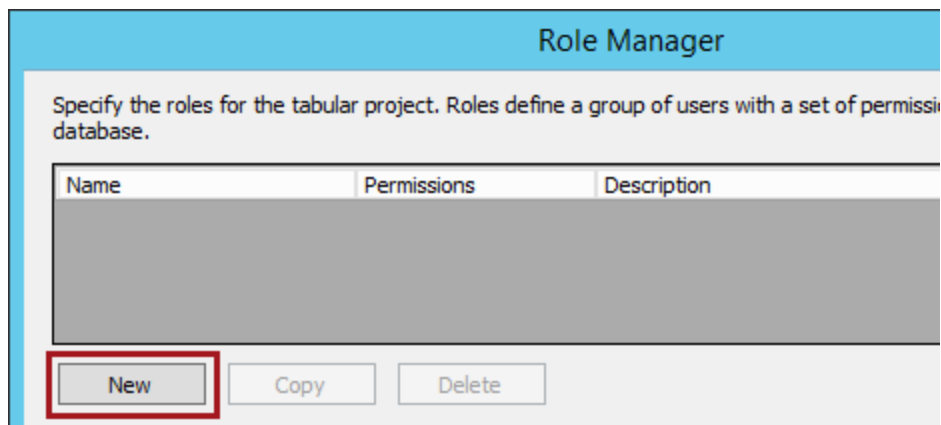
1. Go to the **Salesperson** table in Data View.
2. If necessary, widen the **LoginID** column to view all the values in the column.
3. Notice that the salesperson **Pamela Ansman-Wolfe** has your login ID (used to login to the lab virtual machine). Also, notice that this salesperson belongs to the **SalesTerritoryKey** with the value **1**.
4. To navigate to the **SalesTerritory** table, right-click the **SalesTerritoryKey** column header, and then select **Navigate to Related Table**.
5. Notice that the region for **SalesTerritoryKey** value of **1** is **Northwest**.



6. In **Tabular Model Explorer**, right-click the **Sales Analysis** model, and then select **Roles**.

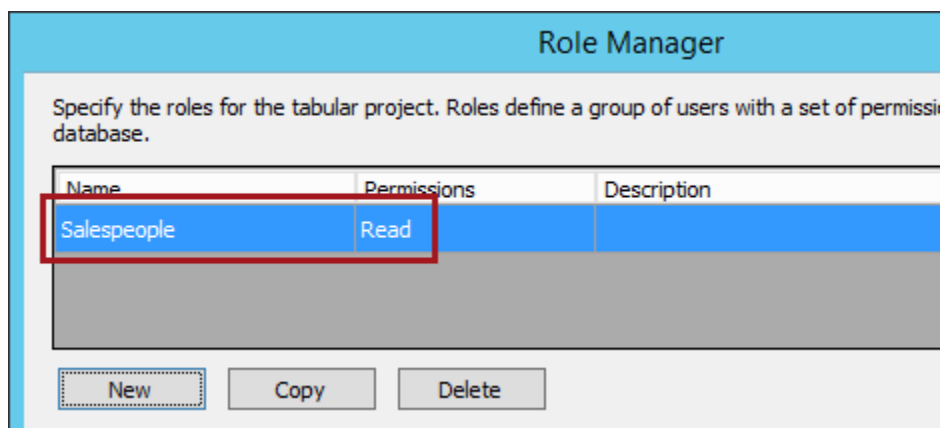


7. In the **Role Manager** window, click **New**.



8. In the **Name** box, replace the text with **Salespeople**.

9. In the **Permissions** dropdown list, select **Read**.



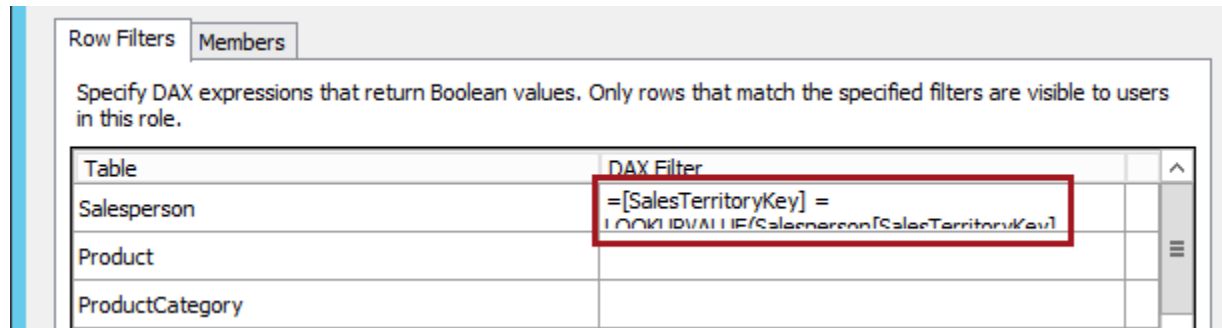
10. In the **Row Filters** tab, in the **Salesperson** table row, in the **DAX Filter** box, enter the following expression, and then press **Enter**.

*For convenience, the expressions defined in this exercise can be copied from the **D:\SQL Server 2016 BI\Lab07\Assets\Snippets.txt** file.*

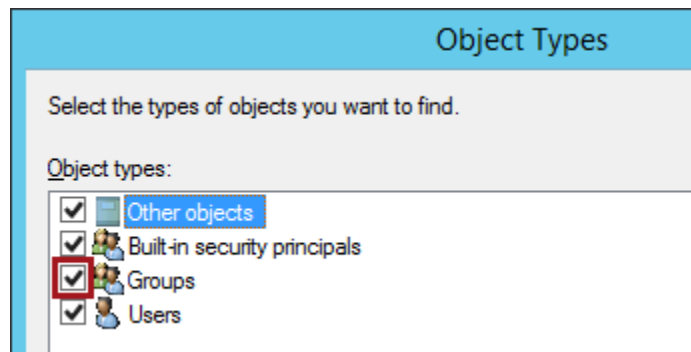
#### DAX

```
=[SalesTerritoryKey] = LOOKUPVALUE(Salesperson[SalesTerritoryKey],  
Salesperson[LoginID], USERNAME())
```

*This expression uses the LOOKUPVALUE function to retrieve the **SalesTerritoryKey** value for the current user. This way the role will allow salespeople to see data related to other salespeople within their own region. This is considered a dynamic filter.*



11. Select the **Members** tab.
12. Click **Add**.
13. In the **Select Users or Groups** window, click **Object Types**.
14. In the **Object Types** window, select the **Groups** object.



15. Click **OK**.
16. In the **Enter the Object Names to Select** box, enter **Salespeople**.

*The **Salespeople** group is maintained by the network administrator to include all AdventureWorks salespeople, regardless of sales territory.*

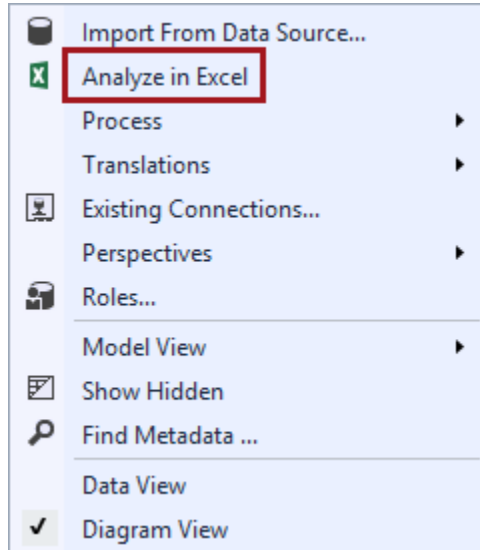
17. Click **Check Names**.
18. Once the group name has resolved, click **OK**.
19. In the **Role Manager** window, click **OK**.

20. To save the project, on the **File** menu, select **Save All**.

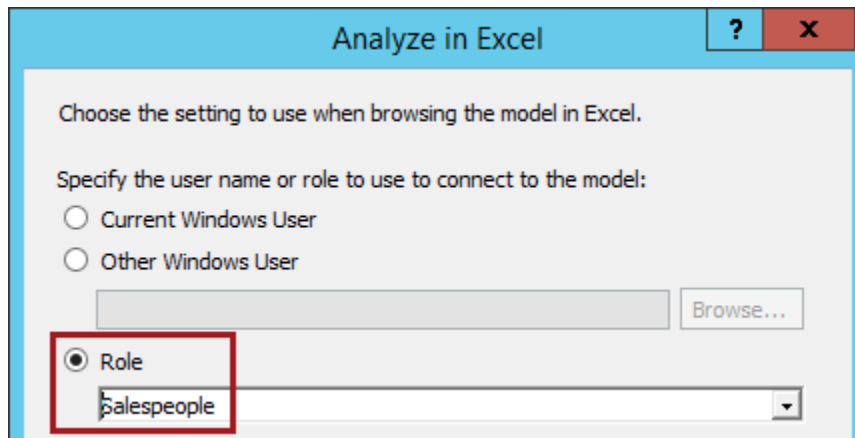
## Testing Perspectives and Permissions

In this task, you will test both the **Monitoring** perspective and the **Salespeople** role.

1. In **Tabular Model Explorer**, right-click the **Sales Analysis** model, and then select **Analyze in Excel**.

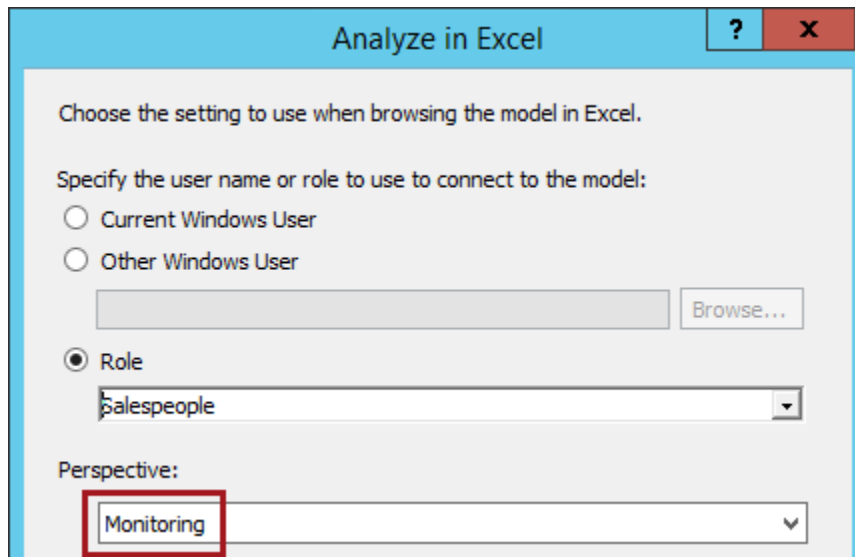


2. In the **Analyze in Excel** window, select the **Role** option.
3. In the dropdown list, check the **Salespeople** role.



4. Click **OK**.

5. In the **Perspective** dropdown list, select the **Monitoring** perspective.



6. Click **OK**.
7. In Excel, in the **PivotTable Fields** pane, from inside the **SalesTerritory** table, select the **Regions** hierarchy to add it to the **Rows** drop zone.
8. Right-click the **Europe** member (cell **A2**), and then select **Expand/Collapse | Expand to "Region"**.
9. Right-click the **France** member (cell **A4**), and then select **Show/Hide Fields | Group**.
10. Hide the **Country** level also.
11. Verify that the PivotTable looks like the following.

	A
1	Row Labels
2	France
3	Germany
4	United Kingdom
5	NA
6	Canada
7	Central
8	Northeast
9	Northwest
10	Southeast
11	Southwest
12	Australia
13	Grand Total

12. In the **PivotTable Fields** pane, from inside the **SalesQuota** measure group, select the **Quota** and **Variance** measures.

13. Notice that only values for the **Northwest** region are visible.

*This is the sales territory that your login account is related to.*

14. Notice also that the **Grand Total** display a visual total (i.e. not the total for all regions).

	A	B	C
1	Row Labels ▼	Quota	Variance
2	Northwest	\$18,855,000.00	(\$2,363,677.41)
3	Grand Total	\$18,855,000.00	(\$2,363,677.41)

## Finishing Up

In this task, you will close Excel.

1. Close Excel.
2. When prompted to save the changes, click **Don't Save**.

# 7: Deploying and Managing the Model

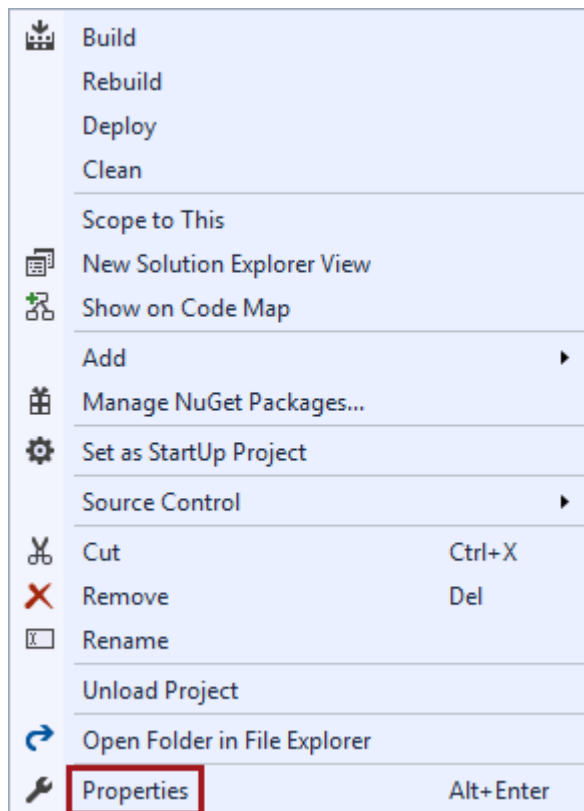
In this exercise, you will deploy the project to the server and explore the functionality supported in SQL Server Management Studio Object Explorer to manage the database and model.

*To commence the lab starting from this exercise, in SSDT, open the **AdventureWorksBI.sln** file from the **D:\SQLServerBI\Lab07\Starter\Ex7** folder. To process the model, on the **Model** menu, select **Process | Process All**.*

## Deploying the Tabular Project

In this task, you will deploy the project.

1. Switch to SQL Server Data Tools (Visual Studio).
2. To review the project deployment properties, in **Solution Explorer**, right-click the **Sales Analysis** project, and then select **Properties**.

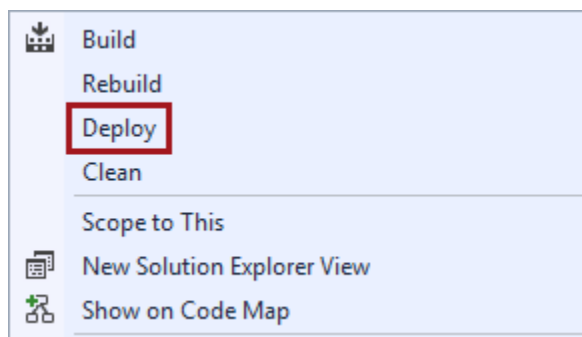


3. In the **Sales Analysis Property Pages** window, notice the **Server** property value.


Deployment Options	
Processing Option	Default
Transactional Deployment	False
Deployment Server	
Server	localhost\TABULAR
Edition	Developer
Database	Sales Analysis
Model Name	Model
Version	13.0

*This property was set when the project was created. It was based on the options configured in Exercise 1.*

4. Notice that the **Database** property is set to the name of the project.
5. Click **Cancel**.
6. To deploy the project, in **Solution Explorer**, right-click the **Sales Analysis** project, and then select **Deploy**.













7. In the **Deploy** window, verify that the status of each work item looks like the following.

 **Success**

10 Total 0 Cancelled  
10 Success 0 Error

Details:

	Work Item	Status	Message
	Deploy metadata	Success. Metadata deployed.	
	Salesperson	Success. 18 rows transferred.	
	Product	Success. 397 rows transferred.	
	ProductCategory	Success. 4 rows transferred.	
	ProductSubcategory	Success. 37 rows transferred.	
	SalesTerritory	Success. 11 rows transferred.	
	ResellerSales	Success. 60,855 rows transferred.	
	SalesQuota	Success. 225 rows transferred.	
	Reseller	Success. 701 rows transferred.	
	Date	Success. 1,461 rows transferred.	

*The database is now available for querying by the AdventureWorks salespeople.*

8. Click **Close**.
9. To close SQL Server Data Tools, on the **File** menu, select **Exit**.

## Exploring the Deployed Database

In this task, you will use SQL Server Management Studio to explore the deployed database.

1. To open SQL Server Management Studio (SSMS), on the taskbar, click the **SQL Server Management Studio** shortcut.



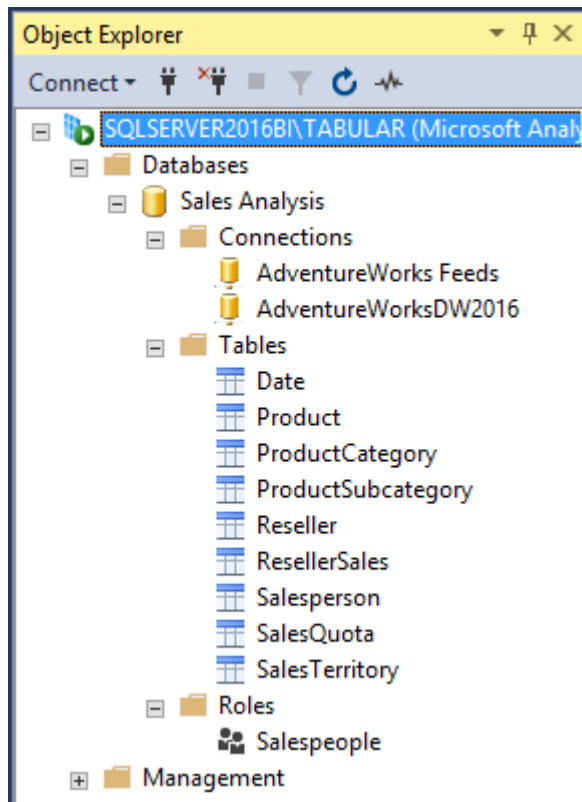
2. In the **Connect to Server** window, configure the following properties.

Credentials	Value
Server Type	Analysis Services
Server Name	SQLSERVER2016BITABULAR

3. Click **Connect**.
4. In **Object Explorer** (located at the left), expand the **Databases** folder.



- Expand **Sales Analysis**, and then expand each of the child folders.



- Right-click several of the connections, tables and the role to learn about the context menu options available to the administrator.

*Once the database is deployed, common tasks include backing up the database; modifying connection settings; processing data sources, tables and partitions; creating, deleting, and merging partitions; and, creating and managing roles to grant permissions to users.*

*The data sources, tables and table partitions can be processed according to a recurring schedule by creating a SQL Server Agent job.*

- To close SSMS, on the **File** menu, select **Exit**.

# Summary

In this lab, you developed a tabular BI Semantic Model (BISM) based on the **AdventureWorksDW2016** database. Specifically, it enabled the analysis of reseller sales compared to sales quota.

You worked with the entire development lifecycle covering the creation of a project, the importing of data from both a SQL Server relational database and also an OData data service, the enhancement of the model user interface, and also the creation of measures and a Key Performance Indicator (KPI).

Once you explored and validated the model by using Excel, you enhanced it with perspectives, partitions and a security role.

Finally, you deployed the project to a tabular instance of SQL Server 2016 Analysis Services.

# Terms of Use

© 2016-2017 Microsoft Corporation. All rights reserved.

By using this hands-on lab, you agree to the following terms:

The technology/functionality described in this hands-on lab is provided by Microsoft Corporation in a “sandbox” testing environment for purposes of obtaining your feedback and to provide you with a learning experience. You may only use the hands-on lab to evaluate such technology features and functionality and provide feedback to Microsoft. You may not use it for any other purpose. Without written permission, you may not modify, copy, distribute, transmit, display, perform, reproduce, publish, license, create derivative works from, transfer, or sell this hands-on lab or any portion thereof.

**COPYING OR REPRODUCTION OF THE HANDS-ON LAB (OR ANY PORTION OF IT) TO ANY OTHER SERVER OR LOCATION FOR FURTHER REPRODUCTION OR REDISTRIBUTION WITHOUT WRITTEN PERMISSION IS EXPRESSLY PROHIBITED.**

**THIS HANDS-ON LAB PROVIDES CERTAIN SOFTWARE TECHNOLOGY/PRODUCT FEATURES AND FUNCTIONALITY, INCLUDING POTENTIAL NEW FEATURES AND CONCEPTS, IN A SIMULATED ENVIRONMENT WITHOUT COMPLEX SET-UP OR INSTALLATION FOR THE PURPOSE DESCRIBED ABOVE. THE TECHNOLOGY/CONCEPTS REPRESENTED IN THIS HANDS-ON LAB MAY NOT REPRESENT FULL FEATURE FUNCTIONALITY AND MAY NOT WORK THE WAY A FINAL VERSION MAY WORK. WE ALSO MAY NOT RELEASE A FINAL VERSION OF SUCH FEATURES OR CONCEPTS. YOUR EXPERIENCE WITH USING SUCH FEATURES AND FUNCTIONALITY IN A PHYSICAL ENVIRONMENT MAY ALSO BE DIFFERENT.**

**FEEDBACK** If you give feedback about the technology features, functionality and/or concepts described in this hands-on lab to Microsoft, you give to Microsoft, without charge, the right to use, share and commercialize your feedback in any way and for any purpose. You also give to third parties, without charge, any patent rights needed for their products, technologies and services to use or interface with any specific parts of a Microsoft software or service that includes the feedback. You will not give feedback that is subject to a license that requires Microsoft to license its software or documentation to third parties because we include your feedback in them. These rights survive this agreement.

**MICROSOFT CORPORATION HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS WITH REGARD TO THE HANDS-ON LAB, INCLUDING ALL WARRANTIES AND CONDITIONS OF MERCHANTABILITY, WHETHER EXPRESS, IMPLIED OR STATUTORY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. MICROSOFT DOES NOT MAKE ANY ASSURANCES OR REPRESENTATIONS WITH REGARD TO THE ACCURACY OF THE RESULTS, OUTPUT THAT DERIVES FROM USE OF THE VIRTUAL LAB, OR SUITABILITY OF THE INFORMATION CONTAINED IN THE VIRTUAL LAB FOR ANY PURPOSE.**

**DISCLAIMER** This lab contains only a portion of new features and enhancements in Microsoft Power BI. Some of the features might change in future releases of the product.

# Document Version

#	Date	Author	Comments
1	28-DEC-2016	Peter Myers	SQL Server 2016 SP1 Visual Studio Enterprise v14.0.25123.00 Update 2 SQL Server Data Tools v14.0.61021.0 SQL Server Analysis Services designer v13.0.1701.8 SQL Server Management Studio v13.0.16100.1 Office Professional Plus 2016 v16.0.7571.7063 64-bit
2	30-JUN-2017	Peter Myers	SQL Server 2016 SP1 CU3 Visual Studio 2015 Enterprise v14.0.25431.01 Update 3 SQL Server Data Tools v14.0.61021.0 SQL Server Analysis Services designer v13.0.1701.8 SQL Server Management Studio v14.0.17119.0 Office Professional Plus 2016 v16.0.8201.2102 64-bit