DAT225x

Developing an Analysis Services Tabular Model

Lab 03 | Enhancing the Tabular Model

Estimated time to complete this lab is 60 minutes

Overview

In this lab, you will enhance each of the model tables to support an intuitive and friendly experience, and also encapsulate business logic with measures and a KPI.

Note: The four labs in this course are accumulative. You cannot complete this lab if you did not successfully complete **Lab 02**.

It is possible to commence from the solution available in the **F:\Labs\Lab02\Solution** folder, providing that you execute **F:\Labs\Lab02\Assets\Script-01.sql** first.

Getting Started

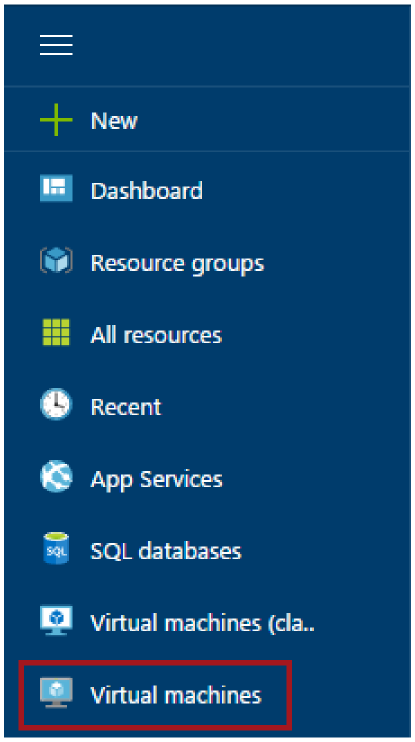
In this exercise, you will get started with the VM created in **Lab 01**.

Getting Started

In this task, you will start the VM, and then connect to it to complete the exercises in this lab.

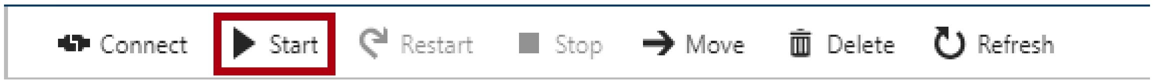
1. Sign in to the **Azure Portal** by using your subscription.

2. In the left pane, select **Virtual Machines**—do not select **Virtual Machines (Classic)**.



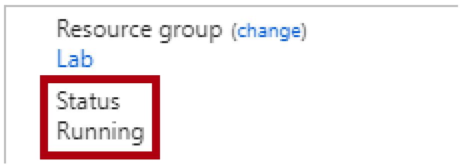
3. In the **Virtual Machines** blade, select the VM you provisioned in **Lab 01**.

4. In the VM blade, click **Start**.



5. Wait for the VM status to update to **Running**.

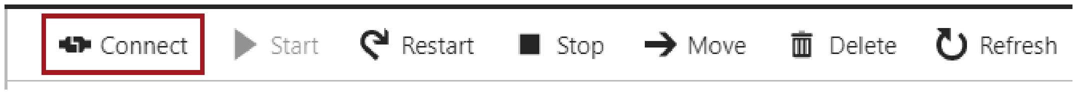
   It usually takes 1-2 minutes for the VM to start.



Resource group (change)
Lab
Status
Running

6. To connect to the VM, click **Connect**.

   Take care not to use the RDP file downloaded in the previous lab. It is likely that a different IP address has be assigned.

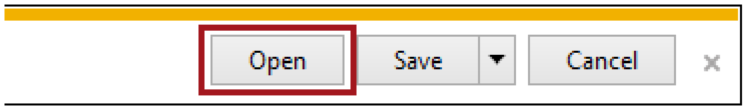


7. In the **Connect to Virtual Machine** pane (located at the right), click **Dowload RDP File**.

   A Remote Desktop File (.rdp) file is downloaded to your computer.

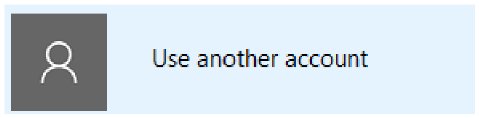
   This file can be used to reconnect to the remote desktop session, but note that if you deallocate the VM and later re-start the VM, it will be likely that a different IP address will be assigned.

8. If prompted by the web browser to open the Remote Desktop File, click **Open**, otherwise, locate the downloaded file, and then double-click it.



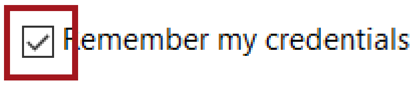
9. If prompted to connect to the unknown publisher, click **Connect**.

10. If prompted, in the **Windows Security** dialog window, click **Use Another Account**.



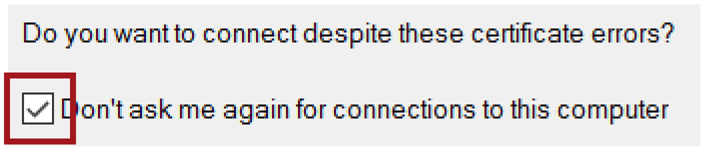
11. Enter the credentials you created for your VM.

12. Check the **Remember My Credentials** checkbox.



13. Click **OK**.

14. In the **Remote Desktop Connection** dialog window, check the **Don't Ask Me Again for Connections to This Computer** checkbox.

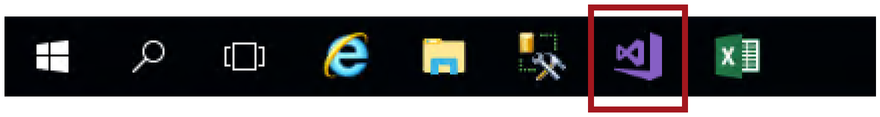


Do you want to connect despite these certificate errors?

☑ Don't ask me again for connections to this computer

15. Click **Yes**.

16. Open **Visual Studio 2017**.



17. On the **File** menu, select **Recent Projects and Solutions** to re-open your project and the model designer.

Exercise 1: Enhancing the Model Interface

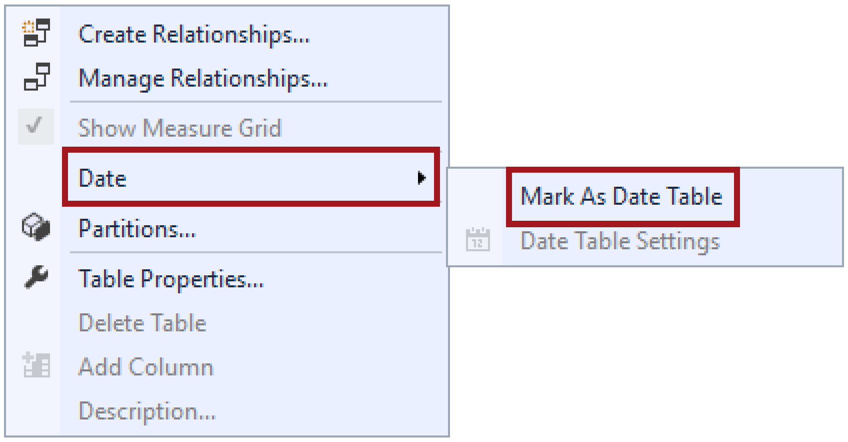
In this exercise, you will enhance the model design to deliver an optimized and user-friendly experience.

To produce an enhanced model, for each table, and as appropriate, you will mark a date table, order columns, create hierarchies to support the ability to drill down and drill up to analyze data at different levels of granularity, and hide columns that should not be used for reporting.

Enhancing the Date Table

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

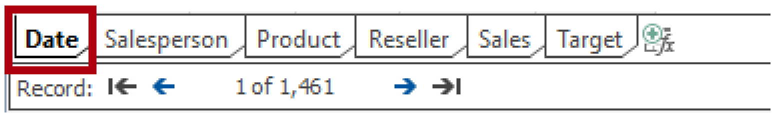
1. To mark the table as a date table, in **Tabular Model Explorer**, expand the **Tables** folder, right-click the **Date** table, and then select **Date | Mark as Date Table**.



2. 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 model. The Excel PivotTable fields pane is one such example that interrogates the model for a date table, and it will surface appropriate time-based filter options based on a marked date table.

3. Go to the **Date** table in Data View.



Lab Check

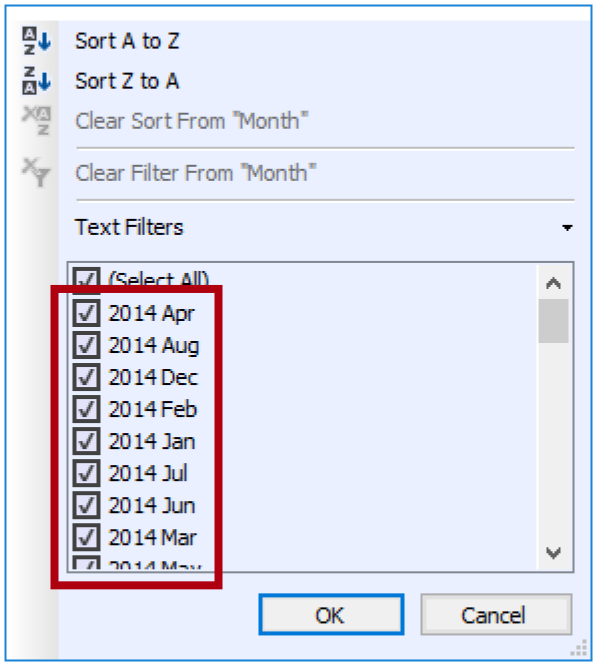
Lab 03 ► Enhancing the Tabular Model

What is the latest (maximum) **Date** stored in the **Date** table?

You may need data from this step to answer a Lab-based Knowledge Check associated with this module.

At this time, we recommend that you open the **Module 3 Lab-based Knowledge Check** portion of the course in EdX to answer the questions as you complete this lab.

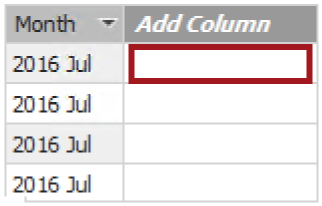
4. To view the **Month** column values, in the **Month** column header, click the down-arrow, and then review the distinct column values, available for filtering the column.



5. Notice that the months are sorted alphabetically, and then click **Cancel**.

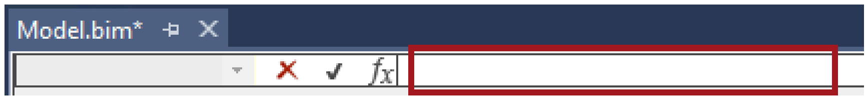
The values of a column can be sorted by the values of a different column from within the same table. You will add a calculated column to produce a month key value, and then sort the Month column by this column.

6. To create a calculated column, select any cell in the **Add Column** column.



Month	Add Column
2016 Jul	
2016 Jul	
2016 Jul	
2016 Jul	

7. In the formula bar (located above the data grid), 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.

For convenience, all expressions in this exercise can be copied from the ***F:\Labs\Lab03\Assets\Snippets.txt*** *file.*

DAX

```
=(YEAR([Date]) * 100) + MONTH([Date])
```

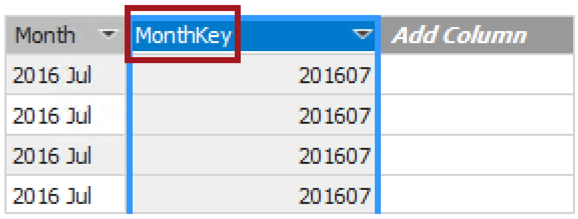
*The **MonthKey** column values stored a number which is the year multiplied by 100 with the month number of year added (e.g., July 2016 is 201607).*

As a model developer, you can often choose between adding a custom column to a query, or a calculated column to a table. Usually, both approaches produce the same result, and the decision to implement a calculation in a specific way could be influenced by your skill, or supported functionality of the chosen expression language.

8. Press **Enter**.

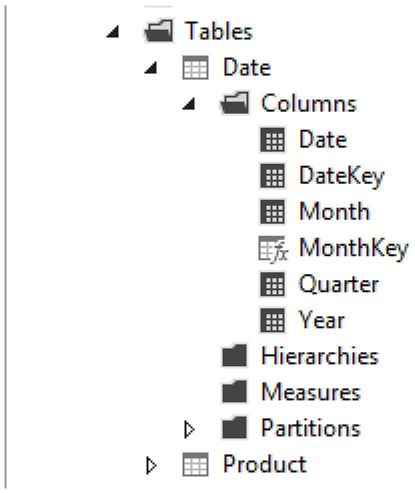
9. To rename the new column, double-click the column header.

10. Replace the text with **MonthKey**, and then press **Enter**.

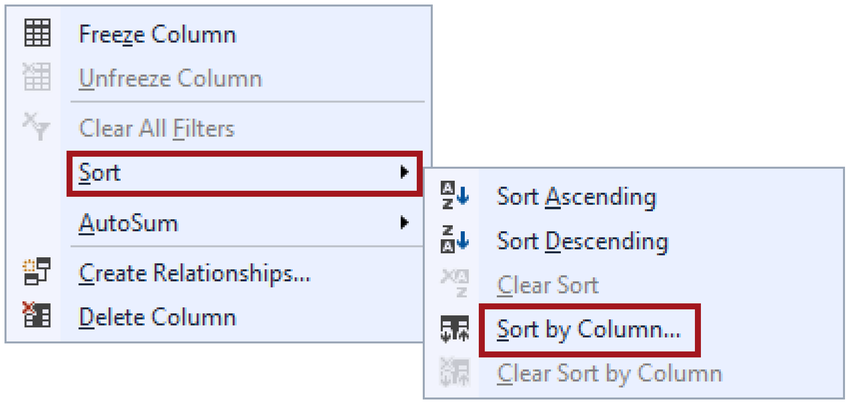


Month	MonthKey	Add Column
2016 Jul	201607	
2016 Jul	201607	
2016 Jul	201607	
2016 Jul	201607	

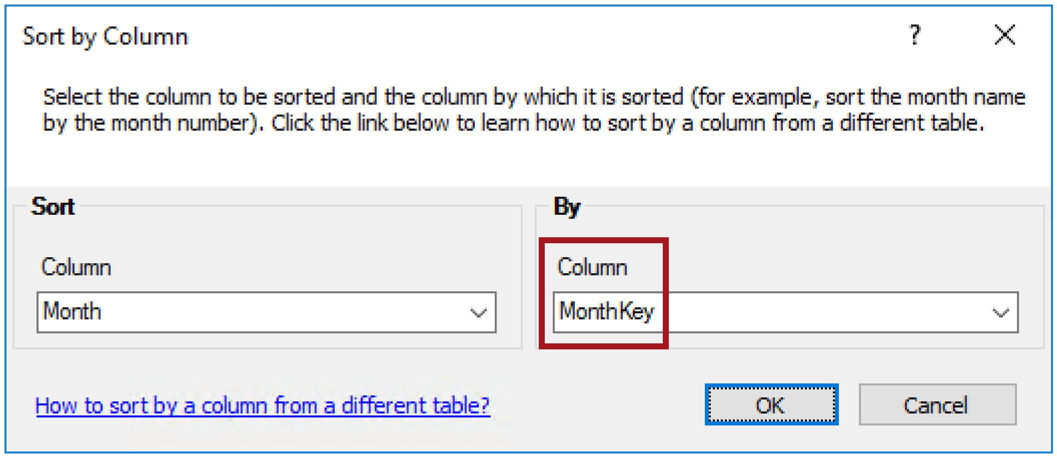
11. To configure the months to sort chronologically, in **Tabular Model Explorer**, expand the **Date** table, and then expand the **Columns** folder.



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



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



Sort by Column

Select the column to be sorted and the column by which it is sorted (for example, sort the month name by the month number). Click the link below to learn how to sort by a column from a different table.

Sort
Column: Month

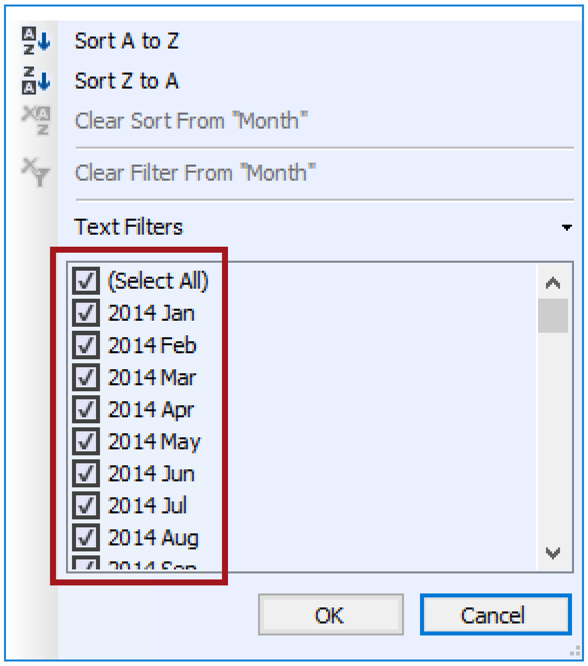
By
Column: MonthKey

How to sort by a column from a different table?

OK Cancel

14. Click **OK**.

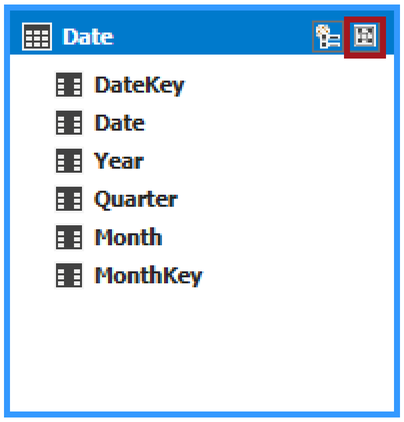
15. In the **Month** column, review the distinct values found in the column, and notice that they are now sorted chronologically.



16. Click **Cancel**.

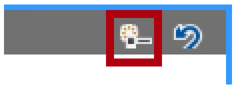
17. Switch to Diagram View.

18. Locate the **Date** table, hover over the top-right corner of the table, and then click the **Maximize** button.

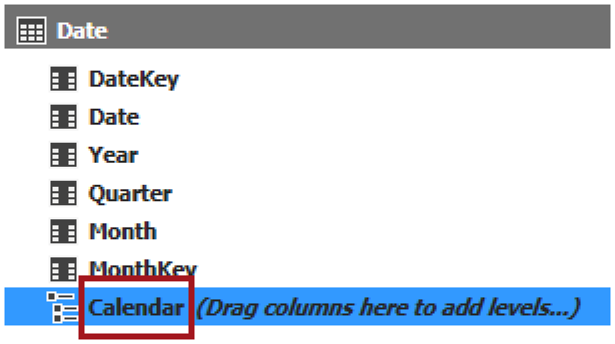


Maximizing the table is a very convenient way to view its definition and to configure it.

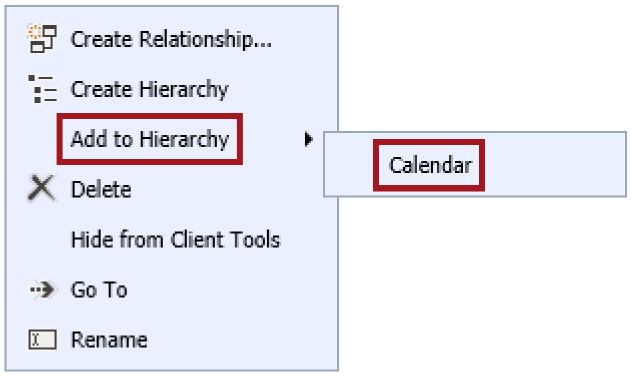
19. In the **Date** table, to create a hierarchy, at the top-right corner, click **Create Hierarchy**.



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



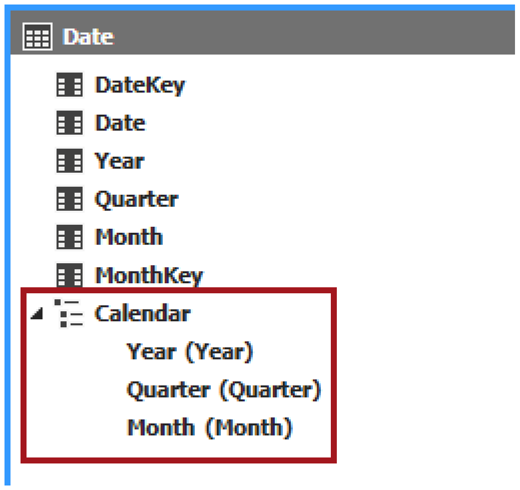
21. To add the **Year** column as the first level of the hierarchy, right-click the **Year** column, and then select **Add to Hierarchy | Calendar**.



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

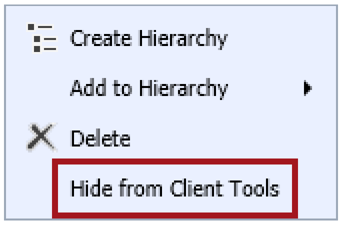
22. Add also the **Quarter** and **Month** columns to the hierarchy.

23. Verify that the **Calendar** hierarchy looks like the following.



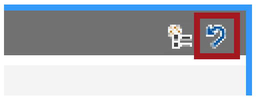
24. To hide specific columns, first select the **DateKey** column, and then while pressing the **Control** key, select the **MonthKey** column.

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



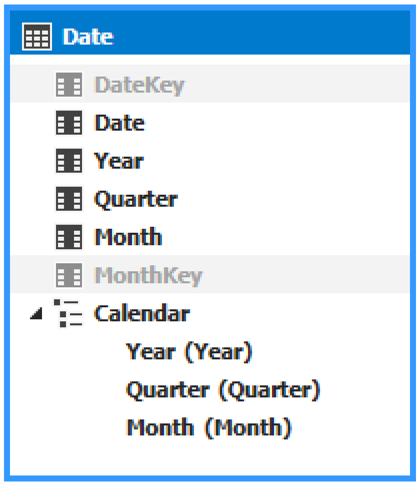
Users exploring and querying the model do not need to access key columns, especially when their purpose is to support relationships between tables, or to sort other columns.

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



27. Resize the **Date** table so that all the table content—including the hierarchy—is visible.

28. Verify that the **Date** table looks like the following.

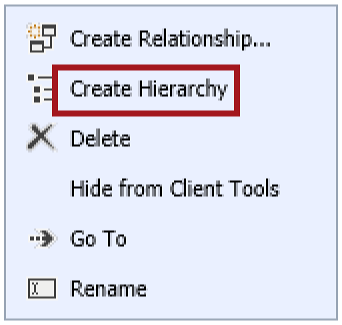


Enhancing the Salesperson Table

In this task, you will extend the **Salesperson** table with a **Salespeople** hierarchy.

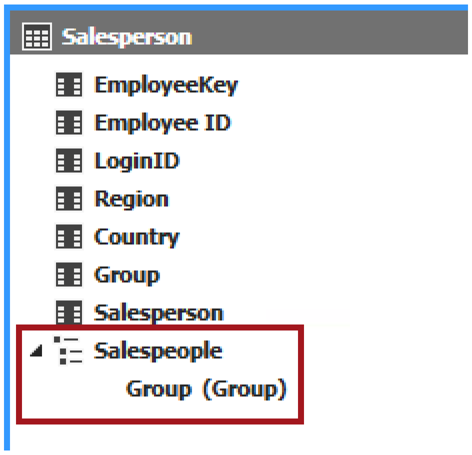
1. Locate and maximize the **Salesperson** table.

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

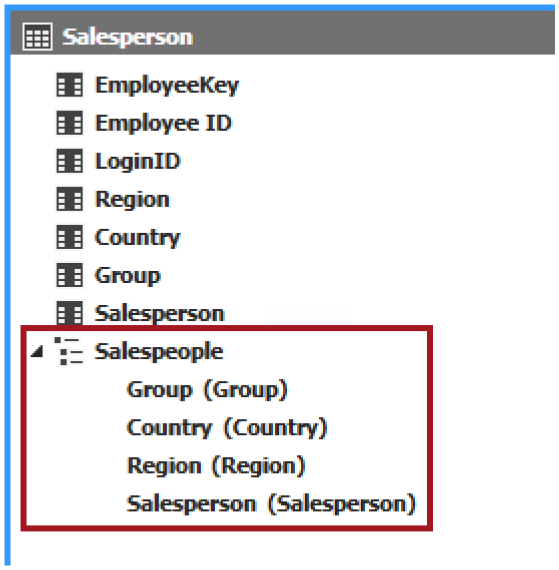


*The selected column, **Group**, will become the first level in the new hierarchy.*

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



4. Add the **Country**, **Region** and **Salesperson** columns—in that order—to the hierarchy.



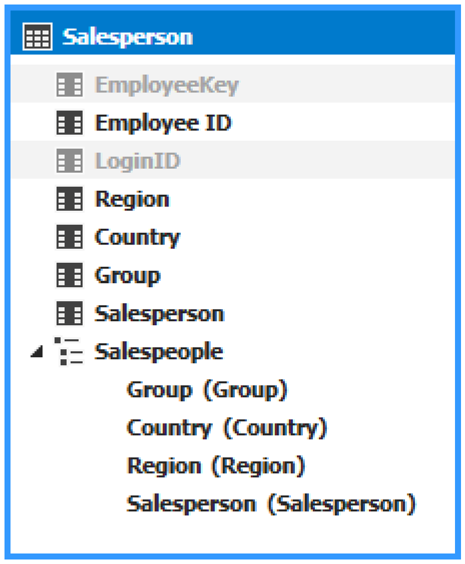
5. Hide the **EmployeeKey** and **LoginID** columns from client tools.

   *The **LoginID** column will be used to enforce row-level security in **Lab 04**.*

6. Minimize the **Salesperson** table.

7. Resize the **Salesperson** table so that all table content is visible.

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



Lab Check

Lab 03 ► Enhancing the Tabular Model

How many salespeople are stored in the **Salesperson** table?

You may need data from this step to answer a Lab-based Knowledge Check associated with this module.

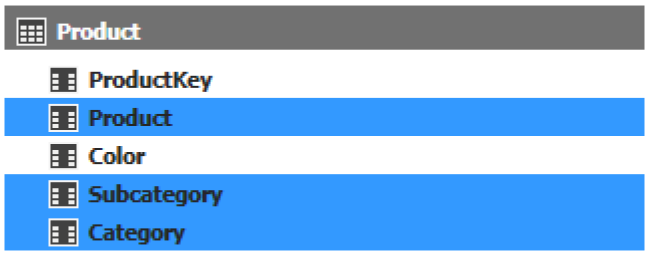
At this time, we recommend that you open the **Module 3 Lab-based Knowledge Check** portion of the course in EdX to answer the questions as you complete this lab.

Enhancing the Product Table

In this task, you will enhance the **Product** table with the **Products** hierarchy.

1. Maximize the **Product** table.

2. As an alternate and more efficient way to create a hierarchy, first select the **Product** column, and then while pressing the **Control** key, select also the **Subcategory** and **Category** columns.



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

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

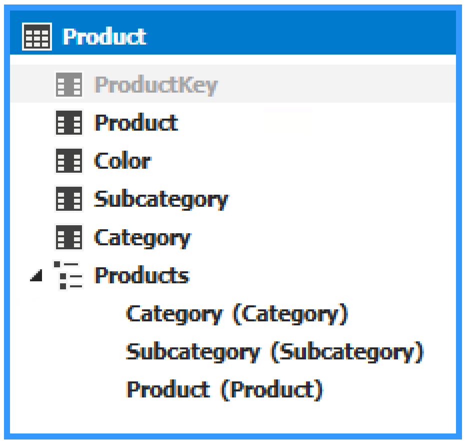
   The behavior for the multi-select method used in this step to create a hierarchy is slightly different from the incremental level addition methods used earlier. When using the multi-select method, the fields will be ordered based on cardinality (the field with fewer members will be the higher level in 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.

5. Hide the **ProductKey** column from the client tools.

6. Minimize the **Product** table.

7. Resize the **Product** table so that all table content is visible.

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



Product

ProductKey
Product
Color
Subcategory
Category
Products
Category (Category)
Subcategory (Subcategory)
Product (Product)

Enhancing the Reseller Table

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

1. Locate and maximize the **Reseller** table.

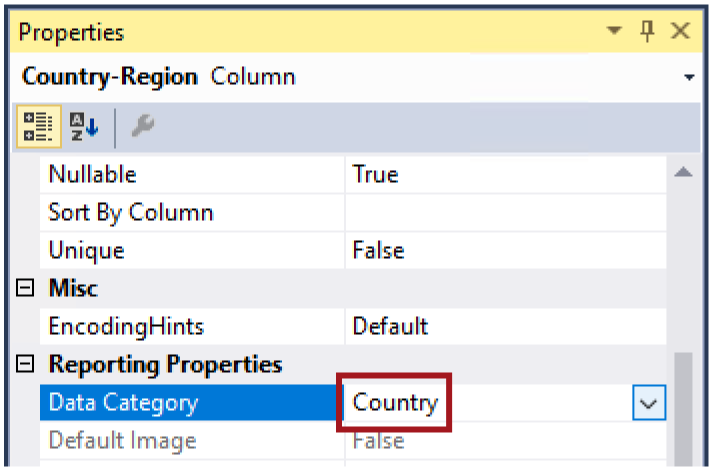
2. Create a hierarchy named **Resellers** based on the **Business Type** and **Reseller** columns (in that order).

3. Create a second hierarchy named **Geography** based on the **Country-Region**, **State-Province**, **City** and **Reseller** columns (in that order).

4. Hide the **ResellerKey** column from client tools.

5. Select the **Country-Region** column.

6. In the **Properties** window (located at the bottom-right), in the **Reporting Properties** group, set the **Data Category** property to **Country**.



Properties

Country-Region Column

Nullable	True
Sort By Column	
Unique	False
Misc	
EncodingHints	Default
Reporting Properties	
Data Category	Country
Default Image	False

Setting the data category can help reporting tools configure default visual types.

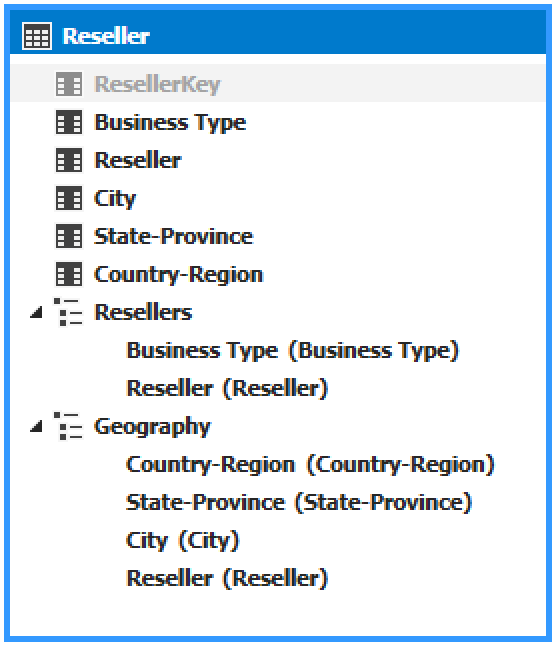
7. Set the following two column data category properties:

Column Name	Data Category
State-Province	StateOrProvince
City	City

8. Minimize the **Reseller** table.

9. Resize the **Reseller** table so that all table content is visible.

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



Reseller
ResellerKey
Business Type
Reseller
City
State-Province
Country-Region
Resellers
Business Type (Business Type)
Reseller (Reseller)
Geography
Country-Region (Country-Region)
State-Province (State-Province)
City (City)
Reseller (Reseller)

Enhancing the Sales Table

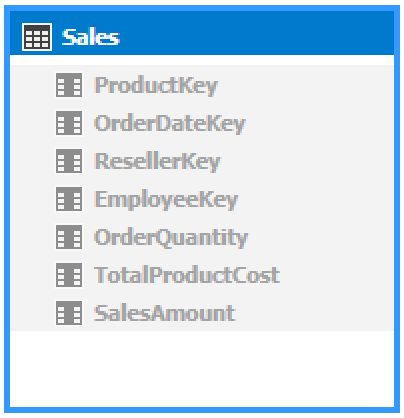
In this task, you will enhance the **Sales** table by hiding all columns from client tools.

1. Locate the **Sales** table.

2. Multi-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 **Sales** table looks like the following.



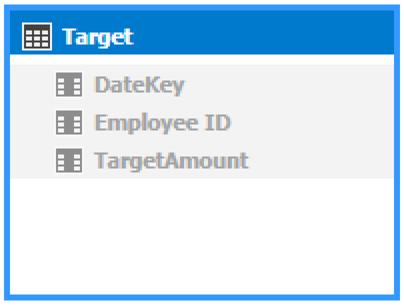
Enhancing the Target Table

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

1. Locate the **Target** table.

2. Multi-select and then hide all columns from client tools. (Do not to hide the table.)

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



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

   All tables (i.e. tables which define business entities), have now been enhanced. In the next exercise, you will enhance fact-type tables which typically consist only of measure (i.e. aggregation logic).

Exercise 2: Adding Aggregation Logic

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

Like calculated columns, measures are DAX expressions. However, they are evaluated within a filter context of the report or PivotTable. Simple measures just aggregate column data. However, measure expressions can be more sophisticated by modifying and overriding filter context, or transforming with 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.

Adding Measures

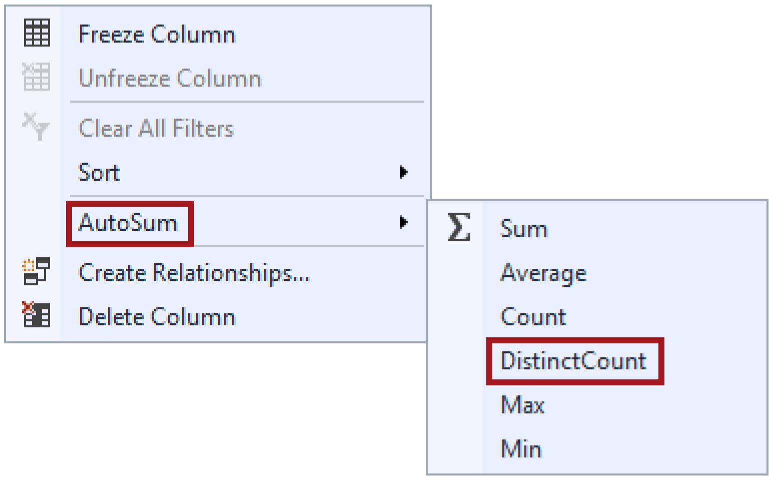
In this task, you will add and format measures to the **Sales** and **Target** tables.

1. Go to the **Sales** table in Data View.

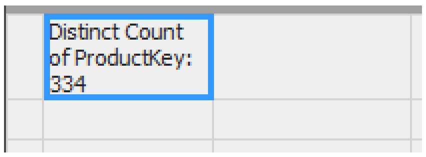
   Calculations can only be added in Data View.

2. To add a measure, in **Tabular Model Explorer**, expand the **Sales** table, and then expand the **Columns** folder.

3. Right-click the **ProductKey** column, and then select **AutoSum | DistinctCount**.



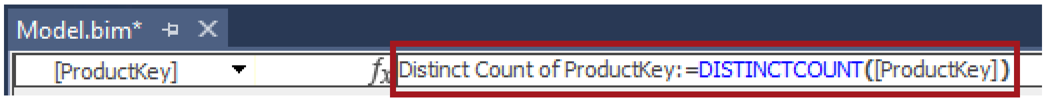
4. In the Measure Grid (at the bottom of the table grid), notice the addition of the **Distinct Count of ProductKey** measure.


Distinct Count of ProductKey: 334

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 on its evaluation.

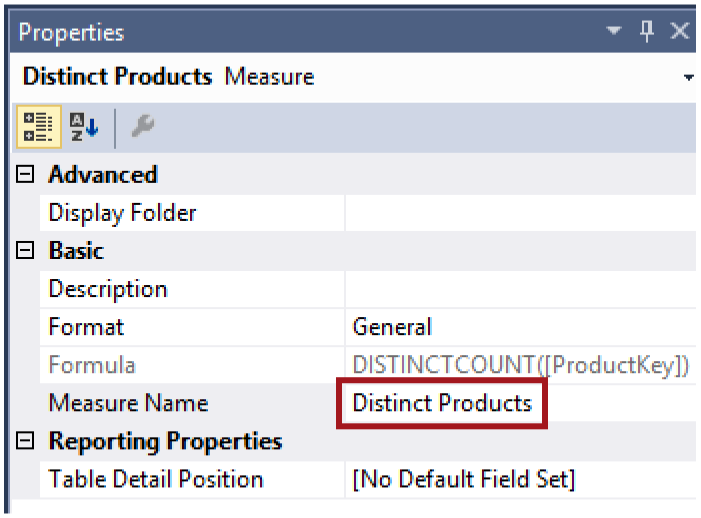
5. In the formula bar, notice the expression that defines the measure, and notice also that the measure name followed by a colon (:) precedes the expression.



```
Distinct Count of ProductKey:=DISTINCTCOUNT([ProductKey])
```

6. In the Measure Grid, select the **Distinct Count of ProductKey** measure.

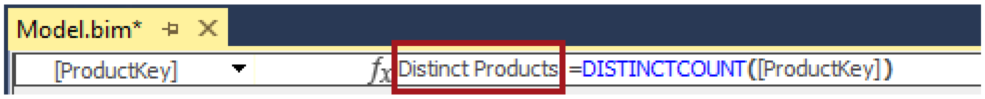
7. In the **Properties** window, modify the **Measure Name** property to **Distinct Products**.



Distinct Products Measure

Advanced	
Display Folder	
Basic	
Description	
Format	General
Formula	DISTINCTCOUNT([ProductKey])
Measure Name	Distinct Products
Reporting Properties	
Table Detail Position	[No Default Field Set]

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



```
Distinct Products:=DISTINCTCOUNT([ProductKey])
```

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 in the table.

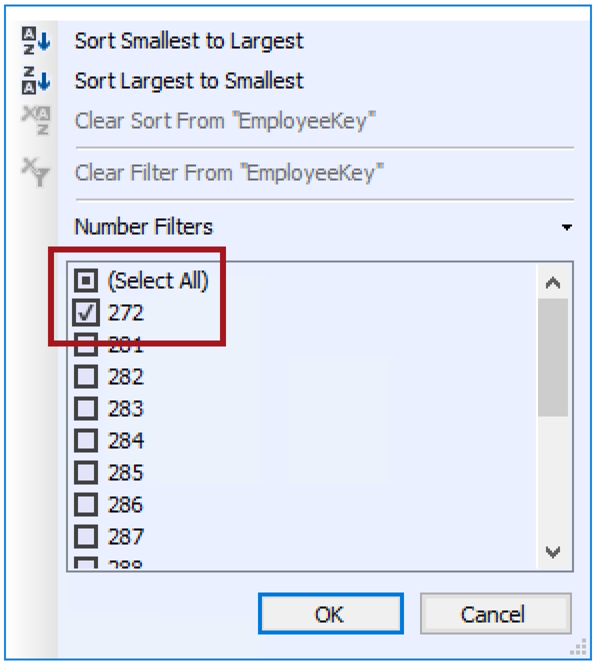
9. In the **Properties** window, modify the following properties of the **Distinct Products** measure:

Property	Value
Format	Whole Number
Show Thousand Separator	True

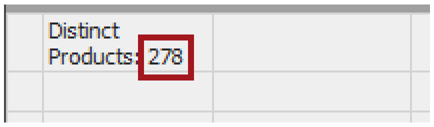
10. In the Measure Grid, notice that the measure also displays a value, **334**.

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

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

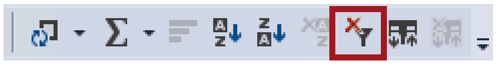


12. Notice that the value of the measure has changed to **278**.

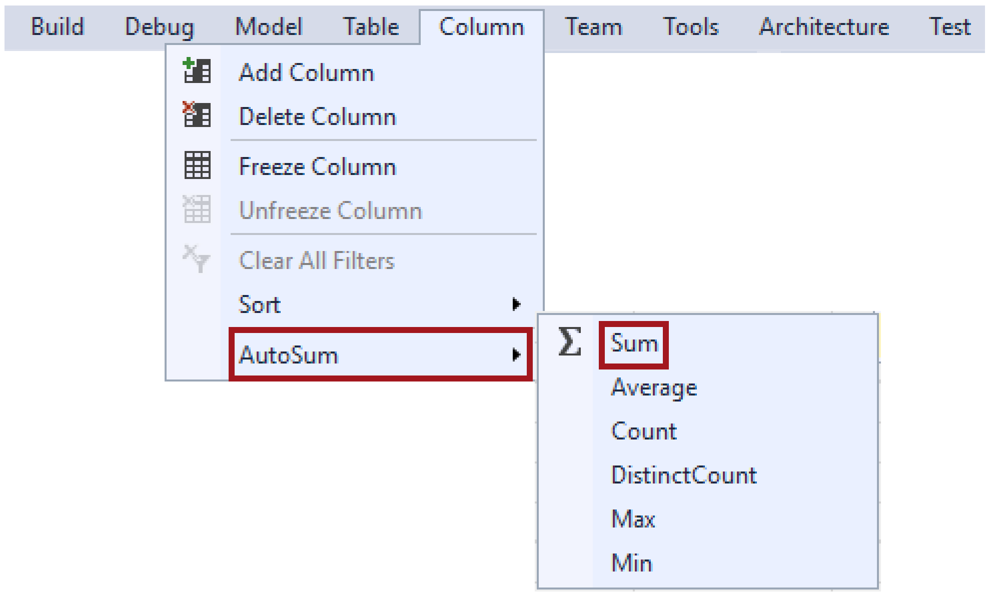


    The table filters can help test the measure expressions, however filters applied to other tables in the model designer will not be considered. The true test of a measure is in a tool like Power BI Desktop, or an Excel PivotTable, where filter context can be set by using columns from different tables.

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



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



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

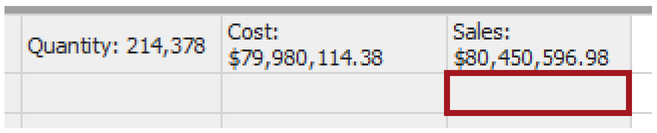
16. In the **Properties** window, modify the following properties:

Property	Value
Measure Name	Units
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.



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

    *For convenience, the measure expressions defined in this exercise can be copied from the **F:\Lab\Lab03\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

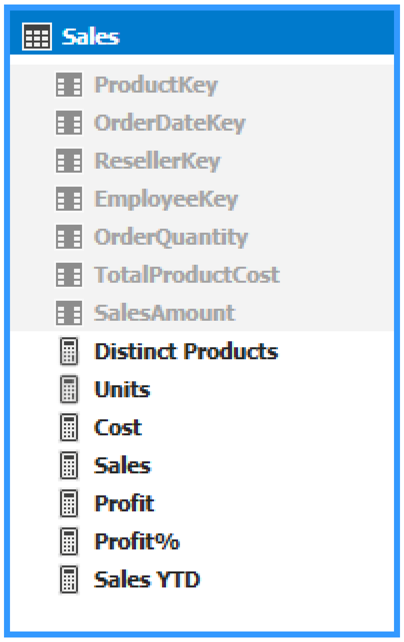
    ```
    Sales YTD:=TOTALYTD([Sales], 'Date'[Date])
    ```

    *This measure uses the TOTALYTD function to calculate year-to-date sales values based on the dates in the related **Date** table.*

    *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 in an Excel report in the next exercise.*

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

24. Verify that the **Sales** table looks like the following.



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

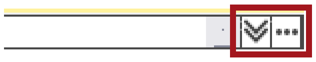
26. Go to the **Target** table in Data View.

27. In the Measure Grid, in any cell, add the following measure, and then format the measure as **Currency**.

DAX

```
Target:=
IF(
     ISFILTERED('Date'[Month]),
     SUMX(
          VALUES('Date'[Quarter]),
          CALCULATE(
               SUM([TargetAmount]),
               ALLEXCEPT('Date', 'Date'[Quarter])
          ) / 3 * CALCULATE(DISTINCTCOUNT('Date'[Month]))
     ),
     SUM([TargetAmount])
)
```

Tip: You can expand the formula bar by clicking the double-down-arrow at the right of the formula bar, and also drag the bottom edge to enlarge the bar area. You can also click the ellipsis to open the ***DAX Editor*** *window.*



*The data in the **Target** table is stored at quarter granularity, and yet the data in the **Sales** 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 **SUMX** function is used to aggregate in a specific way to ensure it sums the quarter values and divides by three. This is an example of how complex a DAX expression may become in order to produce the required result.*

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

DAX

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

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

DAX

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

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

Adding a KPI

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

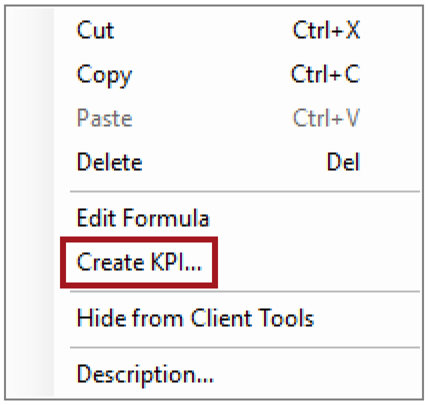
1. Add the following measure beneath the last, and format the measure as **Currency**.

DAX

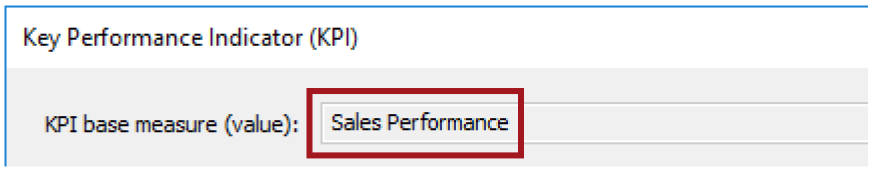
```
Sales Performance:=[Sales]
```

*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**.



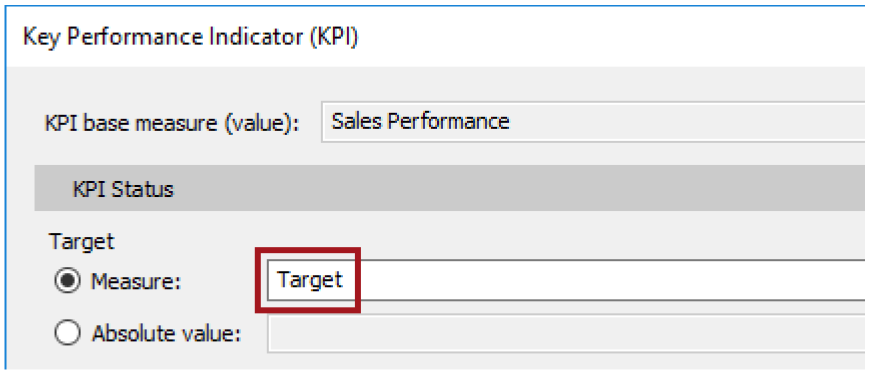
3. In the **Key Performance Indicator (KPI)** window, notice that the KPI base measure (value) is based on the **Sales Performance** measure.



Key Performance Indicator (KPI)

KPI base measure (value): Sales Performance

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



Key Performance Indicator (KPI)

KPI base measure (value): Sales Performance

KPI Status

Target

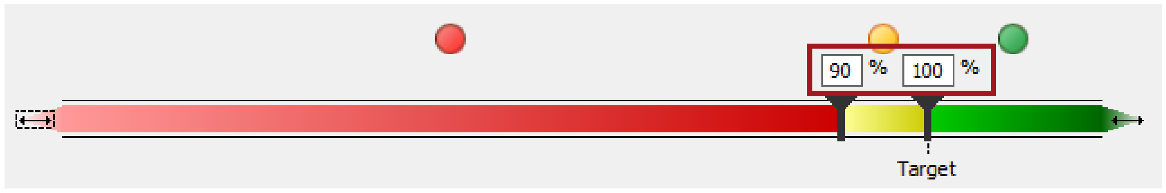
[✓ Measure:] Target

Absolute value:

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

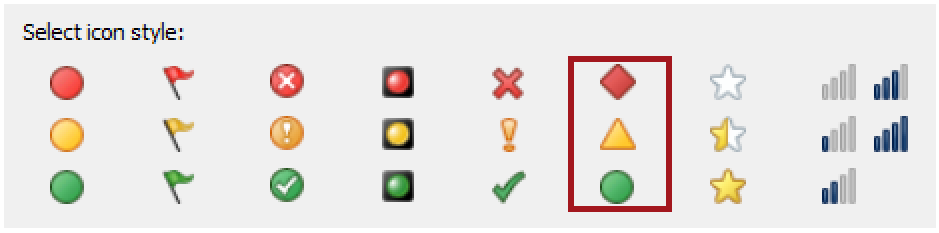
A value of 100% or more means that the sales are on, or exceeding, target. A value less than 0% means that sales are not meeting target.

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



90 % 100 %

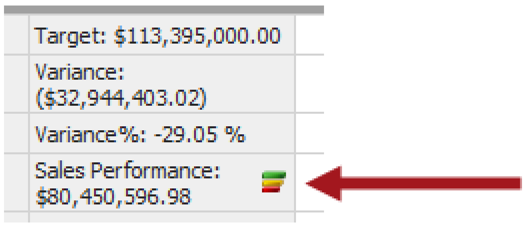
6. In the icon style gallery, select the sixth from the left.



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

7. Click **OK**.

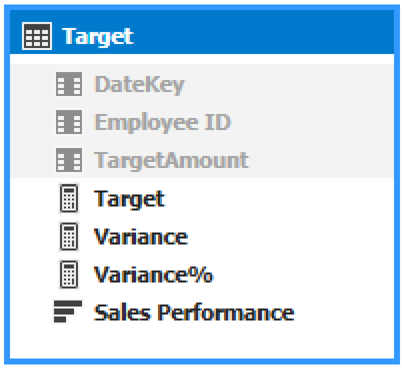
8. In the Measure Grid, notice the icon added to the **Sales Performance** measure to denote that it has been configured as a KPI.



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

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

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



Target
DateKey
Employee ID
TargetAmount
Target
Variance
Variance%
Sales Performance

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

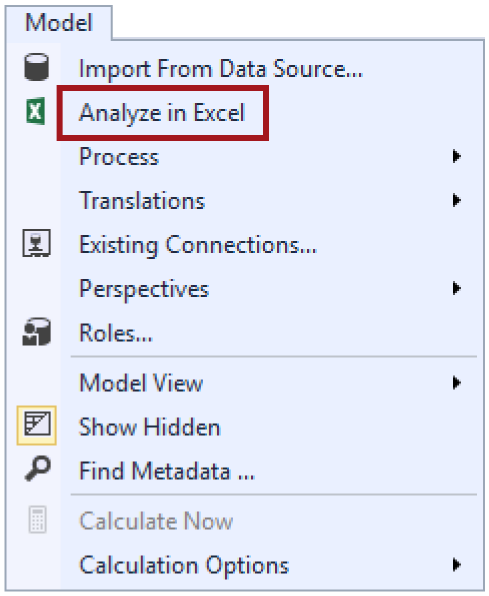
Exercise 3: Exploring the Model Interface

In this exercise, you will explore the model interface with Excel.

Exploring the Model Interface

In this task, you will explore the model interface with Excel.

1. To analyze in Excel, on the **Model** menu, select **Analyze in Excel**.



2. In the **Analyze in Excel** window, click **OK**.

3. In Excel, if prompted to activate Office, click **Cancel**.

4. In the **PivotTable Fields** pane (located at the right), review the fields available in each of the tables.

5. Notice the following:

   - The fields groups consist of "measure groups", KPIs, and tables
   - The "measure groups" are fact-type tables (**Sales** and **Target**) and they contain only measures (because all columns were hidden)
   - The **Sales Performance** KPI is available in the KPIs group, and it consists of a **Value**, **Goal** and **Status** metric
   - The remaining tables represent dimension-type tables, and if the table has one or more hierarchies, then these are listed, with visible columns available in the **More Fields** folder

6. Close Excel, and do not save any changes.

   You have now completed the lab. In the next lab, you will manage the tabular model by creating partitions, and a role to enforce row-level security, and then deploying the Tabular Project.

   If you are not immediately continuing with the next lab, you should complete the ***Finishing Up*** *exercise to shut down and stop the VM.*

Finishing Up

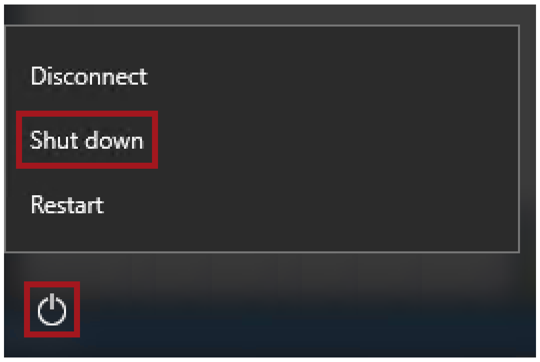
In this exercise, you will shut down and stop the VM.

Finishing Up

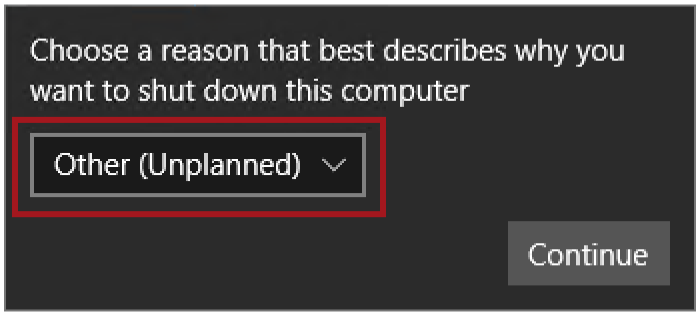
In this task, you will shut down and stop the VM.

1. Close all open applications.

2. Press the **Windows** key, and then in the **Start** page, located at the bottom-left, click the **Power** button, and then select **Shut Down**.



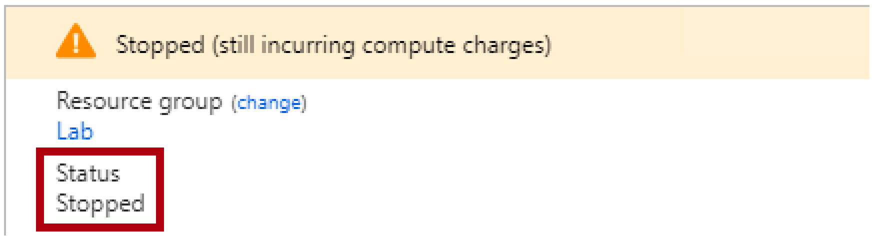
3. When prompted to choose a reason, to accept the default.


Choose a reason that best describes why you want to shut down this computer

Other (Unplanned)

4. Click **Continue**.

5. In the **Azure Portal** Web browser page, wait until the status of the VM updates to **Stopped**.


Stopped (still incurring compute charges)

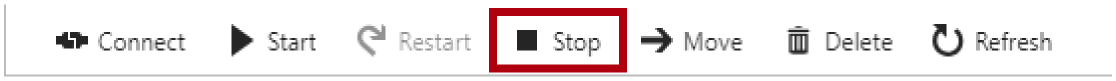
Resource group (change)
Lab

Status
Stopped

In this state, however, the VM is still billable.

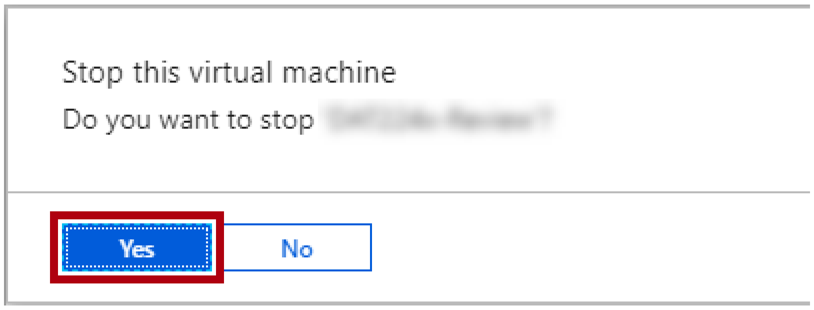
6. Optionally, to deallocate the VM, click **Stop**.

   Deallocation will take some minutes to complete, and also extends the time required to restart the VM. Consider deallocating the VM if you want to reduce costs, or if you choose to complete the next lab after an extended period.

   The deallocation can take several minutes to complete.

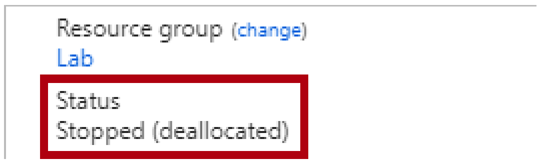
7. When prompted to stop the virtual machine, click **Yes**.

   Stop this virtual machine

   Do you want to stop [redacted]

8. Verify that the VM status updates to **Stopped (Deallocated)**.

   Resource group (change)
   Lab

   Status
   Stopped (deallocated)

   In this state, the VM is now not billable—except for a relatively smaller storage cost.

   Note that a deallocated VM will likely acquire a different IP address the next time it is started.

9. Sign out of the Azure Portal.