All Modules: Analyzing Data with Microsoft Power BI

9 Hr 53 Min Remaining

**Disclaimer** Click to show or hide

# Disclaimer

The content in this lab is presented as designed by Microsoft. Any bugs or errors contained in these materials are maintained by Microsoft. The content provided has been adapted to LODS’ environment, optimizing user experience within the platform. If lab issues are identified that do not pertain directly to LODS’ platform, you can send feedback to Microsoft Learning through the Partner Resource Center at <https://www.microsoft.com/en-us/learning/certification-and-training-help.aspx>.

### Table of Contents

[**Required Lab Setup**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#required-lab-setup)

[**Module 2 - Lab 1 - Prepare Data in Power BI Desktop**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#prepare-data-in-power-bi-desktop)

[**Module 2 - Review Questions**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#module-2-review-questions)

[**Module 3 - Lab 1 - Load Data in Power BI Desktop**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#load-data-in-power-bi-desktop)

[**Module 3 - Review Questions**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#module-3-review-questions)

[**Module 4 - Lab 1 - Model Data in Power BI Desktop, Part 1**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#model-data-in-power-bi-desktop-part-1)

[**Module 4 - Lab 2 - Model Data in Power BI Desktop, Part 2**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#model-data-in-power-bi-desktop-part-2)

[**Module 4 - Review Questions**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#module-4-review-questions)

[**Module 5 - Lab 1 - Create DAX Calculations in Power BI Desktop, Part 1**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#create-dax-calculations-in-power-bi-desktop-part-1)

[**Module 5 - Lab 2 - Create DAX Calculations in Power BI Desktop, Part 2**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#create-dax-calculations-in-power-bi-desktop-part-2)

[**Module 5 - Review Questions**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#module-5-review-questions)

[**Module 7 - Lab 1 - Design a Report in Power BI Desktop, Part 1**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#design-a-report-in-power-bi-desktop-part-1)

[**Module 7 - Lab 2 - Design a Report in Power BI Desktop, Part 2**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#design-a-report-in-power-bi-desktop-part-2)

[**Module 7 - Review Questions**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#module-7-review-questions)

[**Module 8 - Lab 1 - Create a Power BI Dashboard**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#create-a-power-bi-dashboard)

[**Module 8 - Review Questions**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#module-8-review-questions)

[**Module 9 - Lab 1 - Create a Power BI Paginated Report**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#create-a-power-bi-paginated-report)

[**Module 9 - Review Questions**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#module-9-review-questions)

[**Module 10 - Lab 1 - Perform Data Analysis in Power BI Desktop**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#perform-data-analysis-in-power-bi-desktop)

[**Module 10 - Review Questions**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#module-10-review-questions)

# Required Lab Setup

## Before you start

If you need to open a **Starter** or **Solution** in this lab series and you get an Unable to connect error, you first need to connect to the **Lab 06B** Solution and leave it open. Next, open the desired datasheet. When you receive the error click **Edit** and choose the **Sales Report**. Your data will appear.

### Sign in to the lab virtual machine

Hello Zhaolong, log on to [Data Analyst](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10) by pressing [Ctrl+Alt+Delete](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10) and typing in the following credentials:

**Username**: Student

**Password**: Pa55w.rd

SUCCESS! The Lab Files have been downloaded to the D:\ directory.

Once you have received the 'Success!' message, click **Next** to proceed to the lab.

Click [**here**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#toc) to return to the **Table of Contents**.

**Prepare Data in Power BI Desktop**

**The estimated time to complete the lab is 45 minutes**

In this lab you commence the development of a Power BI Desktop solution for the Adventure Works company. It involves connecting to source data, previewing the data, and using data preview techniques to understand the characteristics and quality of the source data.

In this lab you learn how to:

* Open Power BI Desktop
* Set Power BI Desktop options
* Connect to source data
* Preview source data
* Use data preview techniques to better understand the data

**Lab story**

This lab is one of many in a series of labs that was designed as a complete story from data preparation to publication as reports and dashboards. You can complete the labs in any order. However, if you intend to work through multiple labs, for the first 10 labs, we suggest you do them in the following order:

1. Prepare Data in Power BI Desktop
2. Load Data in Power BI Desktop
3. Model Data in Power BI Desktop, Part 1
4. Model Data in Power BI Desktop, Part 2
5. Create DAX Calculations in Power BI Desktop, Part 1
6. Create DAX Calculations in Power BI Desktop, Part 2
7. Design a Report in Power BI Desktop, Part 1
8. Design a Report in Power BI Desktop, Part 2
9. Create a Power BI Dashboard
10. Create a Power BI Paginated Report
11. Perform Data Analysis in Power BI Desktop

**Exercise 1: Prepare Data**

In this exercise you will create eight Power BI Desktop queries. Six queries will source data from SQL Server, and two from CSV files.

**Task 1: Save the Power BI Desktop file**

In this task you will first save the Power BI Desktop file.

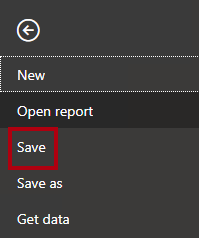
1. To open the Power BI Desktop, on the taskbar, click the Microsoft Power BI Desktop shortcut.



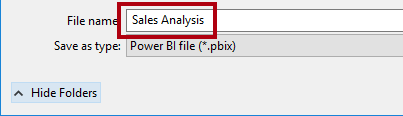
1. To close the getting started window, at the top-right of the window, click **X**.

Picture 3

1. To save the file, click the **File** ribbon tab to open the backstage view.
2. Select **Save**.



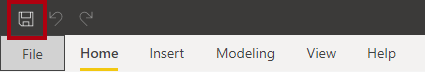
1. In the **Save As** window, navigate to the **D:\DA100\MySolution** folder.
2. In the **File Name** box, enter **Sales Analysis**.



1. Click **Save**.

Picture 17

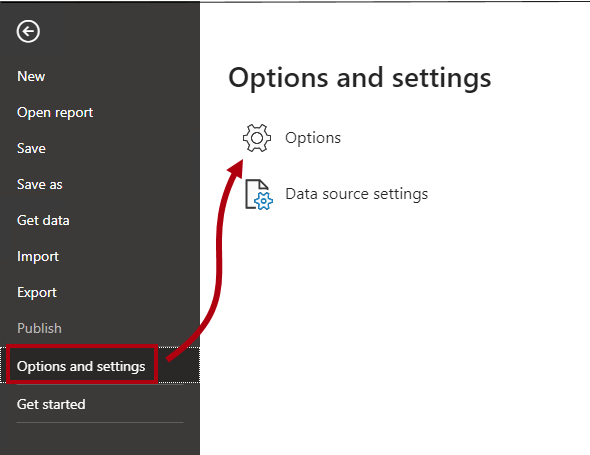
Tip: You can also save the file by click the **Save** icon located at the top-left.



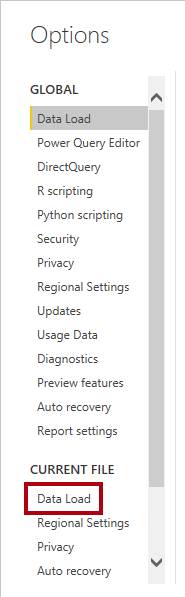
**Task 2: Set Power BI Desktop options**

In this task you will set Power BI Desktop options.

1. In Power BI Desktop, click the **File** ribbon tab to open the backstage view.
2. At the left, select **Options and Settings**, and then select **Options**.

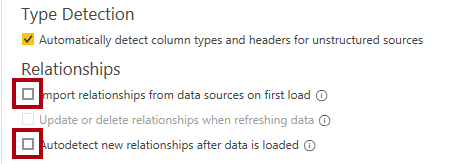


1. In the **Options** window, at the left, in the **Current File** group, select **Data Load**.



The **Data Load** settings for the current file allow setting options that determine default behaviors when modeling.

1. In the **Relationships** group, uncheck the two options that are already checked.



While having these two options enabled can be helpful when developing a data model, you disabled them earlier to support the lab experience. When you create relationships in the **Load Data in Power BI Desktop** lab, you’ll learn why you are adding each one.

1. Click **OK**.

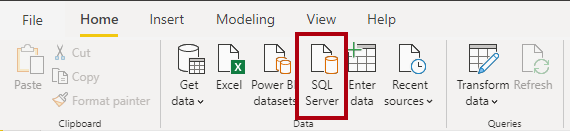


1. Save the Power BI Desktop file.

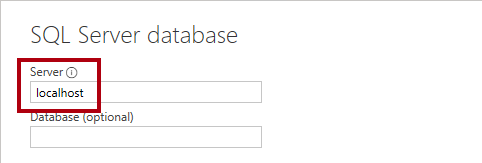
**Task 3: Get data from SQL Server**

In this task you will create queries based on SQL Server tables.

1. On the **Home** ribbon tab, from inside the **Data** group, click **SQL Server**.



1. In the **SQL Server Database** window, in the **Server** box, enter **localhost**.



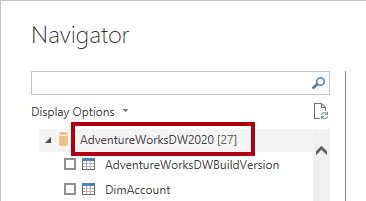
In this lab you’ll connect to the SQL Server database by using **localhost**. This isn’t a recommended practice when creating your own solutions. It’s because gateway data sources cannot resolve **localhost**.

1. Click **OK**.

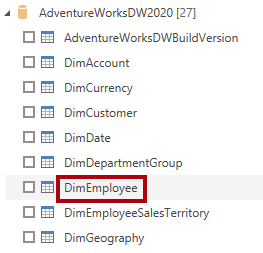


1. In the **Navigator** window, at the left, expand the **AdventureWorksDW2020** database.

The **AdventureWorksDW2020** database is based on the **AdventureWorksDW2017** sample database. It has been modified to support the learning objectives of the course labs.



1. Select—but don’t check—the **DimEmployee** table.



1. In the right pane, notice a preview of the table data.

The preview data allows you to determine the columns and a sample of rows.

1. To create queries, select the checkbox next to the following six tables:
   * DimEmployee
   * DimEmployeeSalesTerritory
   * DimProduct
   * DimReseller
   * DimSalesTerritory
   * FactResellerSales
2. To apply transformations to the data of the selected tables, click **Transform Data**.

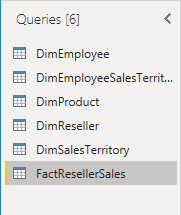
You won’t be transforming the data in this lab. The objectives of this lab focus on exploring and profiling the data in the **Power Query Editor** window.

Picture 30

**Task 4: Preview SQL Server queries**

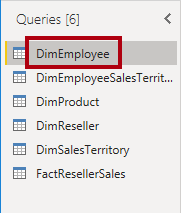
In this task you will preview the data of the SQL Server queries. First, you will learn relevant information about the data. You will also use column quality, column distribution, and column profile tools to understand the data and to assess data quality.

1. In the **Power Query Editor** window, at the left, notice the **Queries** pane.



The **Queries** pane contains one query for each table you checked.

1. Select the first query—**DimEmployee**.



The **DimEmployee** table in the SQL Server database stores one row for each employee. A subset of the rows from this table represents the salespeople, which will be relevant to the model you’ll develop.

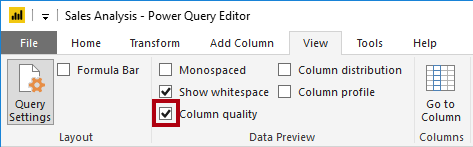
1. At the bottom left, in the status bar, notice the table statistics—the table has 33 columns, and 296 rows.

Picture 36

1. In the data preview pane, scroll horizontally to review all columns.
2. Notice that the last five columns contain **Table** or **Value** links.

These five columns represent relationships to other tables in the database. They can be used to join tables together. You’ll join tables in the **Load Data in Power BI Desktop** lab.

1. To assess column quality, on the **View** ribbon tab, from inside the **Data Preview** group, check **Column Quality**.

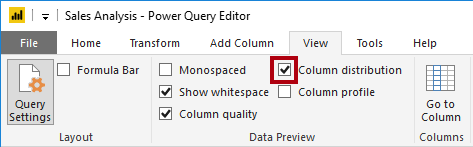


The column quality feature allows you to easily determine the percentage of valid, error, or empty values found in columns.

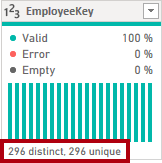
1. For the **Position** column (sixth last column), notice that 94% of rows are empty (null).



1. To assess column distribution, on the **View** ribbon tab, from inside the **Data Preview** group, check **Column Distribution**.

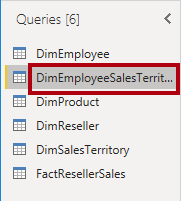


1. Review the **Position** column again, and notice that there are four distinct values, and one unique value.
2. Review the column distribution for the **EmployeeKey** (first) column—there are 296 distinct values, and 296 unique values.



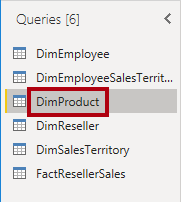
When the distinct and unique counts are the same, it means the column contains unique values. When modeling, it’s important that some model tables have unique columns. These unique columns can be used to create one-to-many relationships, which you will do in the **Model Data in Power BI Desktop, Part 1** lab.

1. In the **Queries** pane, select the **DimEmployeeSalesTerritory** query.



The **DimEmployeeSalesTerritory** table stores one row for each employee and the sales territory regions they manage. The table supports relating many regions to a single employee. Some employees manage one, two, or possibly more regions. When you model this data, you’ll need to define a many-to-many relationship, which you’ll do in the **Model Data in Power BI Desktop, Part 2** lab.

1. In the **Queries** pane, select the **DimProduct** query.

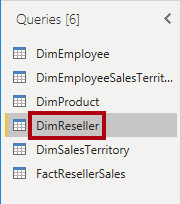


The **DimProduct** table contains one row per product sold by the company.

1. Horizontally scroll to reveal the last columns.
2. Notice the **DimProductSubcategory** column.

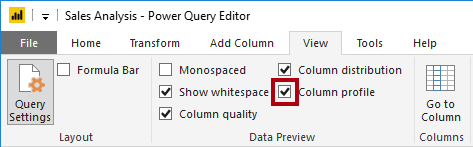
When you add transformations to this query in the **Load Data in Power BI Desktop** lab, you’ll use the **DimProductSubcategory** column to join tables.

1. In the **Queries** pane, select the **DimReseller** query.

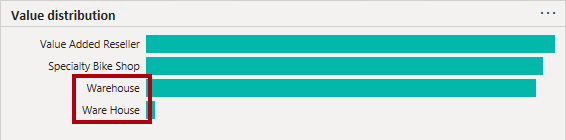


The **DimReseller** table contains one row per reseller. Resellers sell, distribute, or value add to the Adventure Works products.

1. To view column values, on the **View** ribbon tab, from inside the **Data Preview** group, check **Column Profile**.



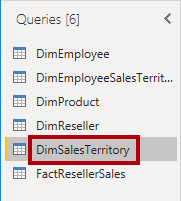
1. Select the **BusinessType** column header.
2. Notice the new pane beneath the data preview pane.
3. Review the column statistics and value distribution in the data preview pane.
4. Notice the data quality issue: there are two labels for warehouse (**Warehouse**, and the misspelled **Ware House**).



1. Hover the cursor over the **Ware House** bar, and notice that there are five rows with this value.

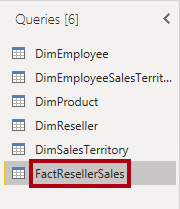
You’ll apply a transformation to relabel these five rows in the **Load Data in Power BI Desktop** lab.

1. In the **Queries** pane, select the **DimSalesTerritory** query.



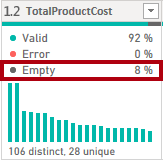
The **DimSalesTerritory** table contains one row per sales region, including **Corporate HQ** (headquarters). Regions are assigned to a country, and countries are assigned to groups. In the **Model Data in Power BI Desktop, Part 1** lab, you’ll create a hierarchy to support analysis at region, country, or group level.

1. In the **Queries** pane, select the **FactResellerSales** query.



The **FactResellerSales** table contains one row per sales order line—a sales order contains one or more line items.

1. Review the column quality for the **TotalProductCost** column, and notice that 8% of the rows are empty.

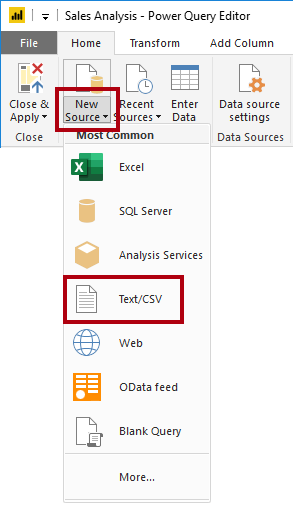


Missing **TotalProductCost** column values is a data quality issue. To address the issue, in the **Load Data in Power BI Desktop** lab, you’ll apply transformations to fill in missing values by using the product standard cost, which is stored in the related **DimProduct** table.

**Task 5: Get data from a CSV file**

In this task you will create a query based on a CSV file.

1. To add a new query, in the **Power Query Editor** window, on the **Home** ribbon tab, from inside the **New Query** group, click the **New Source** down-arrow, and then select **Text/CSV**.

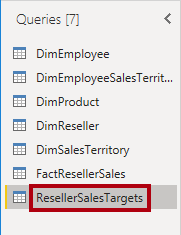


1. In the **Open** window, navigate to the **D:\DA100\Resources** folder, and select the **ResellerSalesTargets.csv** file.
2. Click **Open**.
3. In the **ResellerSalesTargets.csv** window, review the preview data.
4. Click **OK**.



‎

1. In the **Queries** pane, notice the addition of the **ResellerSalesTargets** query.

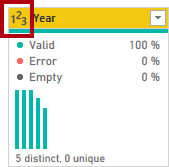


The **ResellerSalesTargets** CSV file contains one row per salesperson, per year. Each row records 12 monthly sales targets (expressed in thousands). Note that the business year for the Adventure Works company commences on July 1.

1. Notice that no column contains empty values.

When there isn’t a monthly sales target, a hyphen character is stored instead.

1. Review the icons in each column header, to the left of the column name.



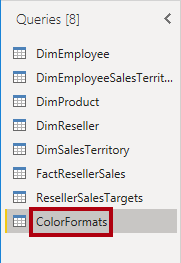
The icons represent the column data type. **123** is whole number, and **ABC** is text.

You’ll apply many transformations to achieve a different shaped result consisting of only three columns: **Date**, **EmployeeKey**, and **TargetAmount** in the **Load Data in Power BI Desktop** lab.

**Task 6: Get additional data from a CSV file**

In this task you will create an additional query based on a different CSV file.

1. Use the steps in the previous task to create a query based on the **D:\DA100\Resources\ColorFormats.csv** file.

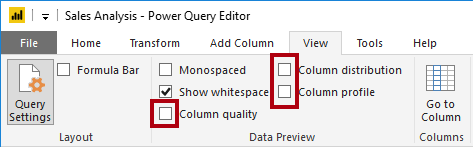


The **ColorFormats** CSV file contains one row per product color. Each row records the HEX codes to format background and font colors. You’ll integrate this data with the **DimProduct** query data in the **Load Data in Power BI Desktop** lab.

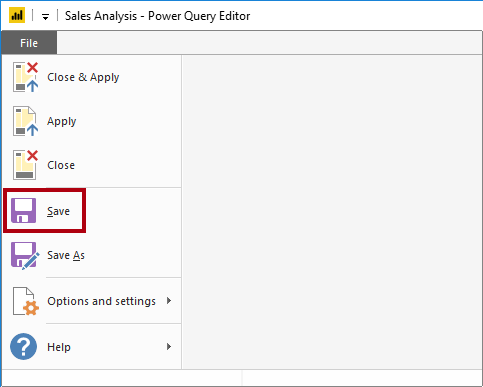
**Task 7: Finish up**

In this task you will complete the lab.

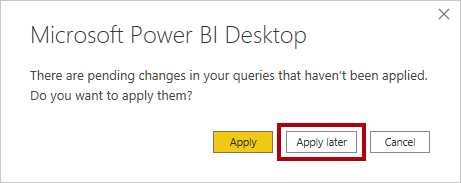
1. On the **View** ribbon tab, from inside the **Data Preview** group, uncheck the three data preview options that were previously enabled in this lab:
   * Column quality
   * Column distribution
   * Column profile



1. To save the Power BI Desktop file, in the **Power Query Editor** window, on the **File** backstage view, select **Save**.



1. When prompted to apply the queries, click **Apply Later**.



Applying the queries will load their data to the data model. You’re not ready to do that, as there are many transformations that must be applied first.

1. If you intend to start the next lab, leave Power BI Desktop open.

You’ll apply various transformations to the queries and then apply the queries to load them to the data model in the **Load Data in Power BI Desktop** lab.

**Congratulations!**

Click **Next** to proceed to the **Review Questions**

Click [**here**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#toc) to return to the **Table of Contents**.

All Modules: Analyzing Data with Microsoft Power BI

9 Hr 48 Min Remaining

**Load Data in Power BI Desktop**

**The estimated time to complete the lab is 45 minutes**

In this lab you will commence apply transformations to each of the queries created in the previous lab. You will then apply the queries to load each as a table to the data model.

In this lab you learn how to:

* Apply various transformations
* Apply queries to load them to the data model

**Lab story**

This lab is one of many in a series of labs that was designed as a complete story from data preparation to publication as reports and dashboards. You can complete the labs in any order. However, if you intend to work through multiple labs, for the first 10 labs, we suggest you do them in the following order:

1. Prepare Data in Power BI Desktop
2. Load Data in Power BI Desktop
3. Model Data in Power BI Desktop, Part 1
4. Model Data in Power BI Desktop, Part 2
5. Create DAX Calculations in Power BI Desktop, Part 1
6. Create DAX Calculations in Power BI Desktop, Part 2
7. Design a Report in Power BI Desktop, Part 1
8. Design a Report in Power BI Desktop, Part 2
9. Create a Power BI Dashboard
10. Create a Power BI Paginated Report
11. Perform Data Analysis in Power BI Desktop

**Exercise 1: Load Data**

In this exercise you will apply transformations to each of the queries created in the previous lab.

**Task 1: Get started**

In this task you will setup the environment for the lab.

*Important: If you are continuing on from the previous lab (and you completed that lab successfully), do not complete this task; instead, continue from the next task.*

1. To open the Power BI Desktop, on the taskbar, click the Microsoft Power BI Desktop shortcut.



1. To close the getting started window, at the top-left of the window, click **X**.

Picture 7

1. To open the starter Power BI Desktop file, click the **File** ribbon tab to open the backstage view.
2. Select **Open Report**.



1. Click **Browse Reports**.

Picture 11

1. In the **Open** window, navigate to the **D:\DA100\Labs\02-load-data-with-power-query-in-power-bi-desktop\Starter** folder.
2. Select the **Sales Analysis** file.
3. Click **Open**.

Picture 12

1. Close any informational windows that may open.
2. Notice the yellow warning message beneath the ribbon.

*The message alerts you to the fact that the queries have not been applied to load as model tables. You’ll apply the queries later in this lab.*

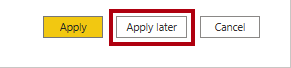
1. To dismiss the warning message, at the right of the yellow warning message, click **X**.

Picture 13

1. To create a copy of the file, click the **File** ribbon tab to open the backstage view.
2. Select **Save As**.



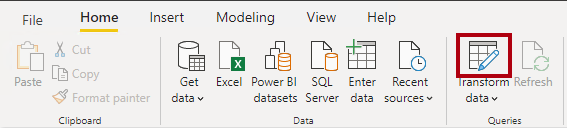
1. If prompted to apply changes, click **Apply Later**.



1. In the **Save As** window, navigate to the **D:\DA100\MySolution** folder.
2. Click **Save**.

Picture 15

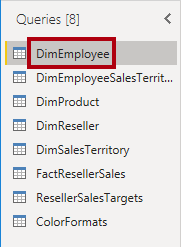
1. To open the **Power Query Editor** window, on the **Home** ribbon tab, from inside the **Queries** group, click the **Transform Data** icon.



**Task 2: Configure the Salesperson query**

In this task you will configure the **Salesperson** query.

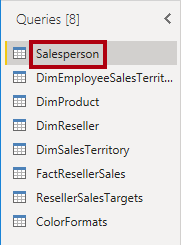
1. In the **Power Query Editor** window, in the **Queries** pane, select the **DimEmployee** query.



1. To rename the query, in the **Query Settings** pane (located at the right), in the **Name** box, replace the text with **Salesperson**, and then press **Enter**.

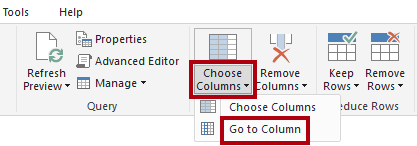
*The query name will determine the model table name. It’s recommended to define concise, yet friendly, names.*

1. In the **Queries** pane, verify that the query name has updated.



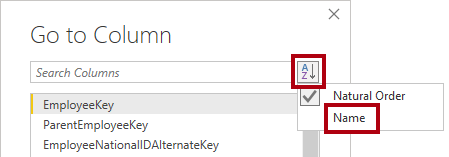
*You’ll now filter the query rows to retrieve only employees who are salespeople.*

1. To locate a specific column, on the **Home** ribbon tab, from inside the **Manage Columns** group, click the **Choose Columns** down-arrow, and then select **Go to Column**.

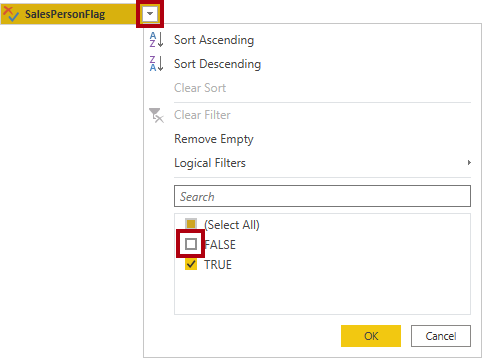


*Tip: This technique is useful when a query contains many columns. If there’s not too many columns, you can simply horizontally scroll to locate the column of interest.*

1. In the **Go to Column** window, to order the list by column name, click the **AZ** sort button, and then select **Name**.



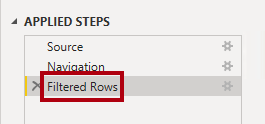
1. Select the **SalesPersonFlag** column, and then click **OK**.
2. To filter the query, in the **SalesPersonFlag** column header, click the down-arrow, and then uncheck **FALSE**.



1. Click **OK**.

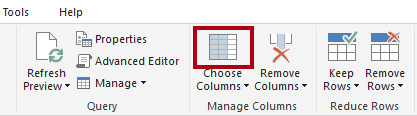
Picture 96

1. In the **Query Settings** pane, in the **Applied Steps** list, notice the addition of the **Filtered Rows** step.

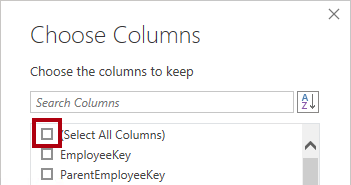


*Each transformation you create results in additional step logic. It’s possible to edit or delete steps. It’s also possible to select a step to preview the query results at that stage of the query transformation.*

1. To remove columns, on the **Home** ribbon tab, from inside the **Manage Columns** group, click the **Choose Columns** icon.



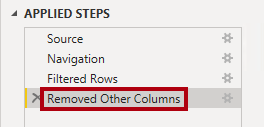
1. In the **Choose Columns** window, to uncheck all columns, uncheck the **(Select All Columns)** item.



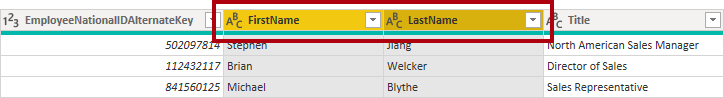
1. To include columns, check the following six columns:
   * EmployeeKey
   * EmployeeNationalIDAlternateKey
   * FirstName
   * LastName
   * Title
   * EmailAddress
2. Click **OK**.



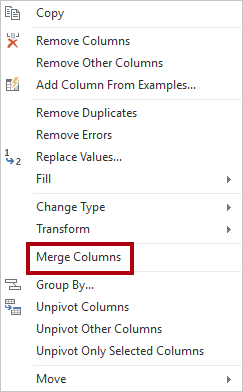
1. In the **Applied Steps** list, notice the addition of another query step.



1. To create a single name column, first select the **FirstName** column header.
2. While pressing the **Ctrl** key, select the **LastName** column.

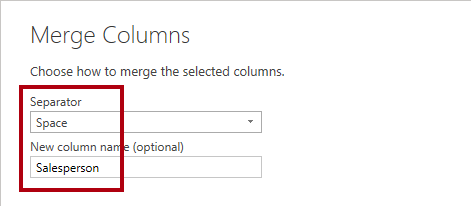


1. Right-click either of the select column headers, and then in the context menu, select **Merge Columns**.



*Many common transformations can be applied by right-clicking the column header, and then choosing them from the context menu. Note, however, more transformations are available in the ribbon.*

1. In the **Merge Columns** window, in the **Separator** dropdown list, select **Space**.
2. In the **New Column Name** box, replace the text with **Salesperson**.



1. Click **OK**.



1. To rename the **EmployeeNationalIDAlternateKey** column, double-click the **EmployeeNationalIDAlternateKey** column header.
2. Replace the text with **EmployeeID**, and then press **Enter**.

*Important: When instructed to rename columns, it’s important that you rename them exactly as described.*

1. Use the previous steps to rename the **EmailAddress** column to **UPN**.

*UPN is an acronym for User Principal Name. The values in this column will be used when you configure row-level security in the* ***Model Data in Power BI Desktop, Part 2*** *lab.*

1. At the bottom-left, in the status bar, verify that the query has five columns and 18 rows.

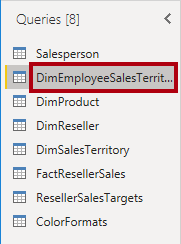
Picture 5638

*Important: It’s important that you do not proceed if your query does not produce the correct result—it won’t be possible to complete later labs. If the query columns or rows don’t match, refer back to the steps in this task to fix any problems.*

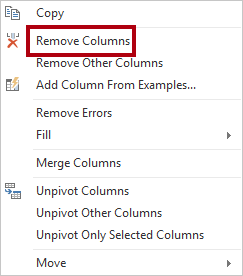
**Task 3: Configure the SalespersonRegion query**

In this task you will configure the **SalespersonRegion** query.

1. In the **Queries** pane, select the **DimEmployeeSalesTerritory** query.



1. In the **Query Settings** pane, rename the query to **SalespersonRegion**.
2. To remove the last two columns, first select the **DimEmployee** column header.
3. While pressing the **Ctrl** key, select the **DimSalesTerritory** column header.
4. Right-click either of the select column headers, and then in the context menu, select **Remove Columns**.



1. In the status bar, verify that the query has two columns and 39 rows.

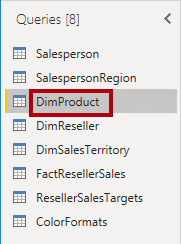
Picture 5641

**Task 4: Configure the Product query**

In this task you will configure the **Product** query.

*Important: When detailed instructions have already been provided, the lab steps will now provide more concise instructions. If you need the detailed instructions, you can refer back to the steps of previous tasks.*

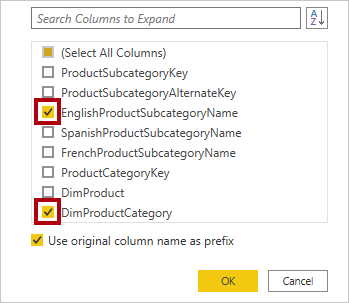
1. Select the **DimProduct** query.



1. Rename the query to **Product**.
2. Locate the **FinishedGoodsFlag** column, and then filter the column to retrieve products that are finished goods (i.e. TRUE).
3. Remove all columns, except the following:
   * ProductKey
   * EnglishProductName
   * StandardCost
   * Color
   * DimProductSubcategory
4. Notice that the **DimProductSubcategory** column represents a related table (it contains **Value** links).
5. In the **DimProductSubcategory** column header, at the right of the column name, click the expand button.

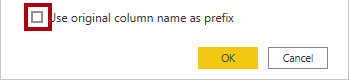
Picture 5644

1. To uncheck all columns, uncheck the **(Select All Columns)** item.
2. Check the **EnglishProductSubcategoryName** and **DimProductCategory** columns.



*By selecting these two columns, a transformation will be applied to join to the* ***DimProductSubcategory*** *table, and then include these columns. The* ***DimProductCategory*** *column is, in fact, another related table in the data source.*

1. Uncheck the **Use Original Column Name as Prefix** checkbox.



*Query column names must always be unique. If left checked, this checkbox would prefix each column with the expanded column name (in this case* ***DimProductSubcategory****). Because it’s known that the selected column names don’t collide with column names in the* ***Product*** *query, the option is deselected.*

1. Click **OK**.

Picture 5648

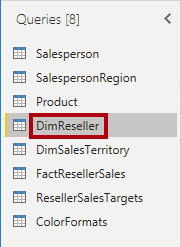
1. Notice that the transformation resulted in the addition of two columns, and that the **DimProductSubcategory** column has been removed.
2. Expand the **DimProductCategory** column, and then introduce only the **EnglishProductCategoryName** column.
3. Rename the following four columns:
   * **EnglishProductName** to **Product**
   * **StandardCost** to **Standard Cost** (include a space)
   * **EnglishProductSubcategoryName** to **Subcategory**
   * **EnglishProductCategoryName** to **Category**
4. In the status bar, verify that the query has six columns and 397 rows.

Picture 5651

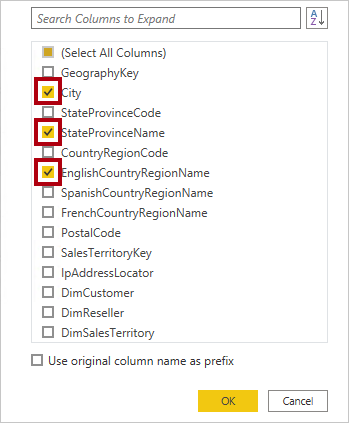
**Task 5: Configure the Reseller query**

In this task you will configure the **Reseller** query.

1. Select the **DimReseller** query.



1. Rename the query to **Reseller**.
2. Remove all columns, except the following:
   * ResellerKey
   * BusinessType
   * ResellerName
   * DimGeography
3. Expand the **DimGeography** column, to include only the following three columns:
   * City
   * StateProvinceName
   * EnglishCountryRegionName

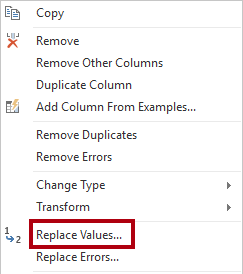


1. In the **Business Type** column header, click the down-arrow, and then review the distinct column values, and notice the incorrect spelling of warehouse.

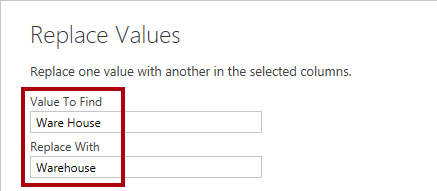


‎

1. Right-click the **Business Type** column header, and then select **Replace Values**.



1. In the **Replace Values** window, configure the following values:
   * In the **Value to Find** box, enter **Ware House**
   * In the **Replace With** box, enter **Warehouse**



1. Click **OK**.



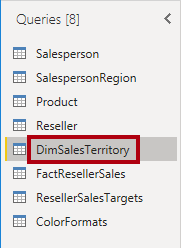
1. Rename the following four columns:
   * **BusinessType** to **Business Type** (include a space)
   * **ResellerName** to **Reseller**
   * **StateProvinceName** to **State-Province**
   * **EnglishCountryRegionName** to **Country-Region**
2. In the status bar, verify that the query has six columns and 701 rows.

Picture 5657

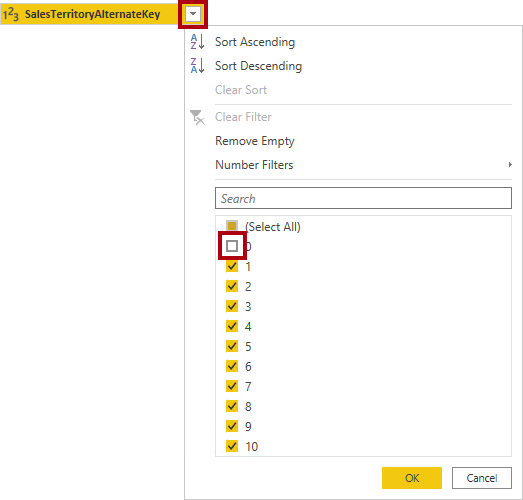
**Task 6: Configure the Region query**

In this task you will configure the **Region** query.

1. Select the **DimSalesTerritory** query.



1. Rename the query to **Region**.
2. Apply a filter to the **SalesTerritoryAlternateKey** column to remove the value 0 (zero).



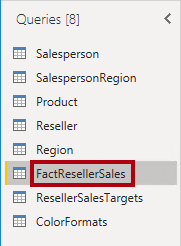
1. Remove all columns, except the following:
   * SalesTerritoryKey
   * SalesTerritoryRegion
   * SalesTerritoryCountry
   * SalesTerritoryGroup
2. Rename the following three columns:
   * **SalesTerritoryRegion** to **Region**
   * **SalesTerritoryCountry** to **Country**
   * **SalesTerritoryGroup** to **Group**
3. In the status bar, verify that the query has four columns and 10 rows.

Picture 5661

**Task 7: Configure the Sales query**

In this task you will configure the **Sales** query.

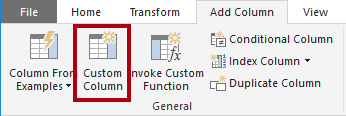
1. Select the **FactResellerSales** query.



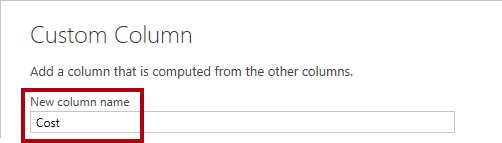
1. Rename the query to **Sales**.
2. Remove all columns, except the following:
   * SalesOrderNumber
   * OrderDate
   * ProductKey
   * ResellerKey
   * EmployeeKey
   * SalesTerritoryKey
   * OrderQuantity
   * UnitPrice
   * TotalProductCost
   * SalesAmount
   * DimProduct

*You may recall in the* ***Prepare Data in Power BI Desktop*** *lab that a small percentage of* ***FactResellerSales*** *rows had missing* ***TotalProductCost*** *values. The* ***DimProduct*** *column has been included to retrieve the product standard cost column to assist fixing the missing values.*

1. Expand the **DimProduct** column, uncheck all columns, and then include only the **StandardCost** column.
2. To create a custom column, on the **Add Column** ribbon tab, from inside the **General** group, click **Custom Column**.



1. In the **Custom Column** window, in the **New Column Name** box, replace the text with **Cost**.



1. In the **Custom Column Formula** box, enter the following expression (after the equals symbol):
2. For your convenience, you can copy the expression from the **D:\DA100\Labs\load-data-with-power-query-in-power-bi-desktop\Assets\Snippets.txt** file.

**Power Query**

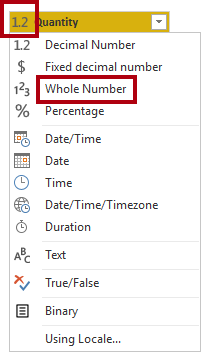
if [TotalProductCost] = null then [OrderQuantity] \* [StandardCost] else [TotalProductCost]

*This expression tests if the* ***TotalProductCost*** *value is missing. If it is, produces a value by multiplying the* ***OrderQuantity*** *value by the* ***StandardCost*** *value; otherwise, it uses the existing* ***TotalProductCost*** *value.*

1. Click **OK**.

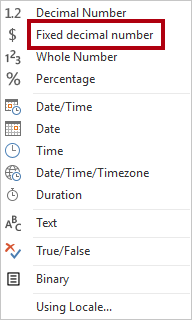


1. Remove the following two columns:
   * TotalProductCost
   * StandardCost
2. Rename the following three columns:
   * **OrderQuantity** to **Quantity**
   * **UnitPrice** to **Unit Price** (include a space)
   * **SalesAmount** to **Sales**
3. To modify the column data type, in the **Quantity** column header, at the left of the column name, click the **1.2** icon, and then select **Whole Number**.



*Configuring the correct data type is important. When the column contains numeric value, it’s also important to choose the correct type if you expect to perform mathematic calculations.*

1. Modify the following three column data types to **Fixed Decimal Number**.
   * Unit Price
   * Sales
   * Cost



*The fixed decimal number data type stores values with full precision, and so requires more storage space that decimal number. It’s important to use the fixed decimal number type for financial values, or rates (like exchange rates).*

1. In the status bar, verify that the query has 10 columns and 999+ rows.

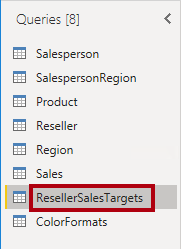
Picture 5669

*A maximum of 1000 rows will be loaded as preview data for each query.*

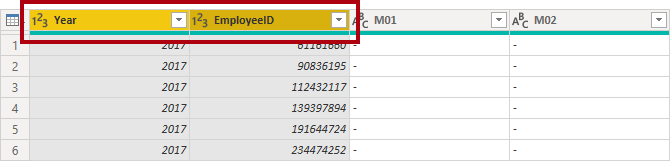
**Task 8: Configure the Targets query**

In this task you will configure the **Targets** query.

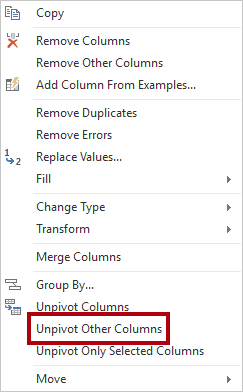
1. Select the **ResellerSalesTargets** query.



1. Rename the query to **Targets**.
2. To unpivot the 12 month columns (**M01**-**M12**), first multi-select the **Year** and **EmployeeID** column headers.



1. Right-click either of the select column headers, and then in the context menu, select **Unpivot Other Columns**.



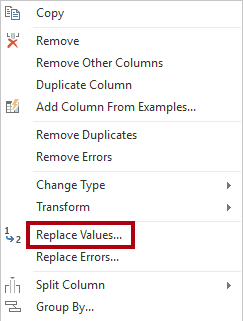
1. Notice that the column names now appear in the **Attribute** column, and the values appear in the **Value** column.
2. Apply a filter to the **Value** column to remove hyphen (-) values.

*You may recall that the hyphen character was used in the source CSV file to represent zero (0).*

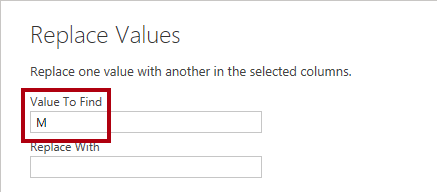
1. Rename the following two columns:
   * **Attribute** to **MonthNumber** (there is no space between the two words—it will be removed later)
   * **Value** to **Target**

*You’ll now apply transformations to produce a date column. The date will be derived from the* ***Year*** *and* ***MonthNumber*** *columns. You’ll create the column by using the* ***Columns From Examples*** *feature.*

1. To prepare the **MonthNumber** column values, right-click the **MonthNumber** column header, and then select **Replace Values**.



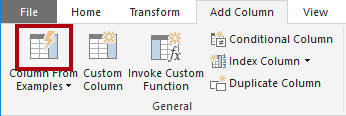
1. In the **Replace Values** window, in the **Value To Find** box, enter **M**.



1. Click **OK**.
2. Modify the **MonthNumber** column data type to **Whole Number**.

Picture 5678

1. On the **Add Column** ribbon tab, from inside the **General** group, click The **Column From Examples** icon.



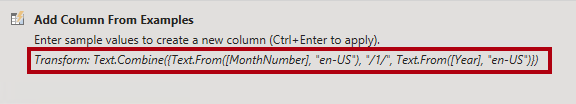
1. Notice that the first row is for year **2017** and month number **7**.
2. In the **Column1** column, in the first grid cell, commence entering **7/1/2017**, and then press **Enter**.

*The virtual machine uses US regional settings, so this date is in fact July 1, 2017.*

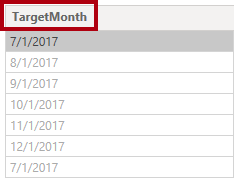
1. Notice that the grid cells update with predicted values.

*The feature has accurately predicted that you are combining values from the* ***Year*** *and* ***MonthNumber*** *columns.*

1. Notice also the formula presented above the query grid.



1. To rename the new column, double-click the **Merged** column header.
2. Rename the column as **TargetMonth**.

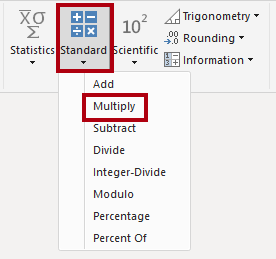


1. To add the new column, click **OK**.

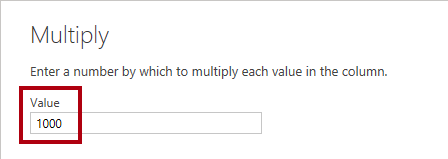


1. Remove the following columns:
   * Year
   * MonthNumber
2. Modify the following column data types:
   * **Target** as fixed decimal number
   * **TargetMonth** as date
3. To multiply the **Target** values by 1000, select the **Target** column header, and then on the **Transform** ribbon tab, from inside the **Number Column** group, click **Standard**, and then select **Multiply**.

*You may recall that the target values were stored as thousands.*



1. In the **Multiply** window, in the **Value** box, enter **1000**.



1. Click **OK**.



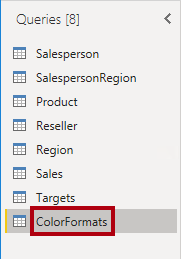
1. In the status bar, verify that the query has three columns and 809 rows.

Picture 5685

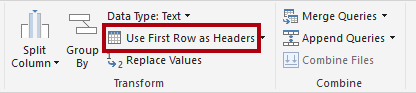
**Task 9: Configure the ColorFormats query**

In this task you will configure the **ColorFormats** query.

1. Select the **ColorFormats** query.



1. Notice that the first row contains the column names.
2. On the **Home** ribbon tab, from inside the **Transform** group, click **Use First Row as Headers**.



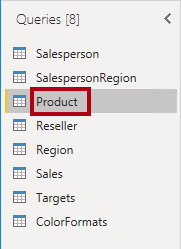
1. In the status bar, verify that the query has three columns and 10 rows.

Picture 5689

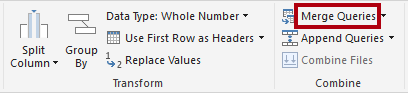
**Task 10: Update the Product query**

In this task you will update the **Product** query by merging the **ColorFormats** query.

1. Select the **Product** query.

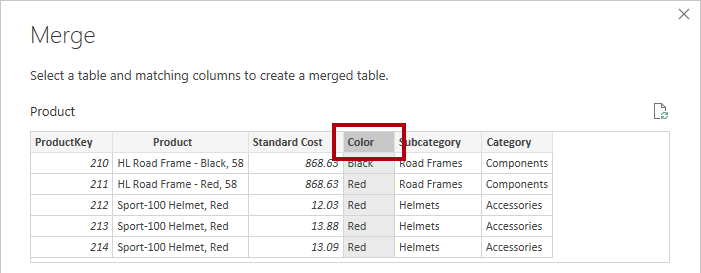


1. To merge the **ColorFormats** query, on the **Home** ribbon tab, from inside the **Combine** group, click **Merge Queries**.

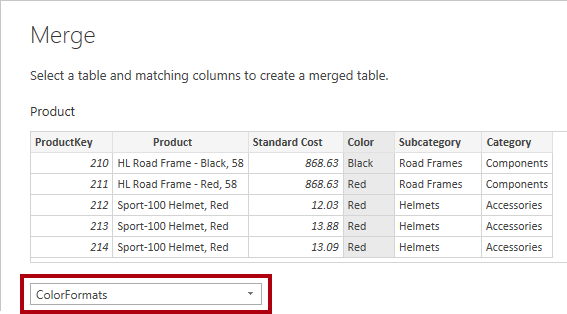


*Merging queries allows integrating data, in this case from different data sources (SQL Server and a CSV file).*

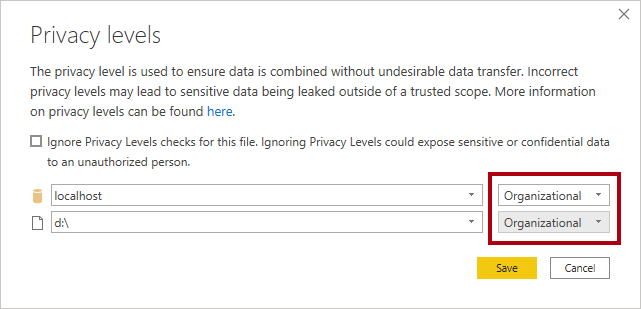
1. In the **Merge** window, in the **Product** query grid, select the **Color** column header.



1. Beneath the **Product** query grid, in the dropdown list, select the **ColorFormats** query.



1. In the **ColorFormats** query grid, select the **Color** column header.
2. When the **Privacy Levels** window opens, for each of the two data sources, in the corresponding dropdown list, select **Organizational**.



*Privacy levels can be configured for data source to determine whether data can be shared between sources. Setting each data source as* ***Organizational*** *allows them to share data, if necessary. Note that Private data sources can never be shared with other data sources. It doesn’t mean that Private data cannot be shared; it means that the Power Query engine cannot share data between the sources.*

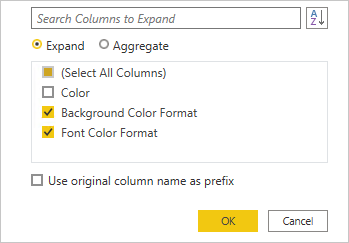
1. Click **Save**.

Picture 5692

1. In the **Merge** window, use the default **Join Kind** - maintaining the selection of Left Outer and click **OK**.



1. Expand the **ColorFormats** column to include the following two columns:
   * Background Color Format
   * Font Color Format



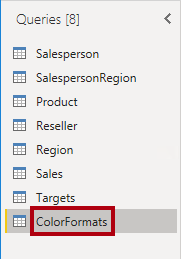
1. In the status bar, verify that the query now has eight columns and 397 rows.

Picture 5695

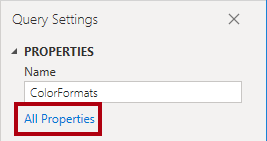
**Task 11: Update the ColorFormats query**

In this task you will update the **ColorFormats** to disable its load.

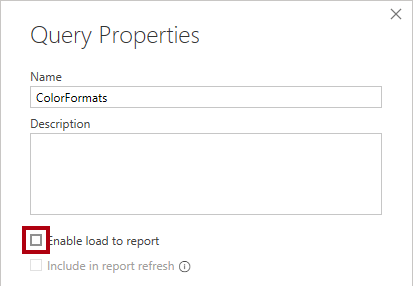
1. Select the **ColorFormats** query.



1. In the **Query Settings** pane, click the **All Properties** link.



1. In the **Query Properties** window, uncheck the **Enable Load To Report** checkbox.



Disabling the load means it will not load as a table to the data model. This is done because the query was merged with the **Product** query, which is enabled to load to the data model.

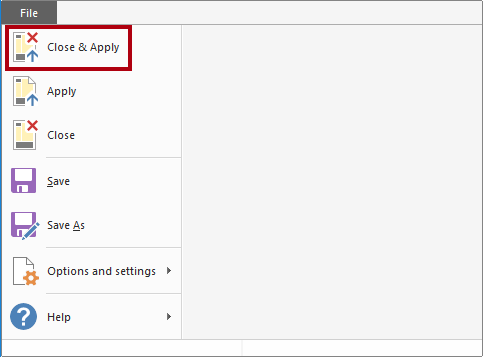
1. Click **OK**.



**Task 12: Finish up**

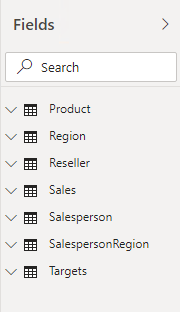
In this task you will complete the lab.

1. Verify that you have eight queries, correctly named as follows:
   * Salesperson
   * SalespersonRegion
   * Product
   * Reseller
   * Region
   * Sales
   * Targets
   * ColorFormats (which will not load to the data model)
2. To load the data model, on the **File** backstage view, select **Close & Apply**.



*All load-enabled queries are now loaded to the data model.*

1. In the **Fields** pane (located at the right), notice the seven tables loaded to the data model.



1. Save the Power BI Desktop file.
2. If you intend to start the next lab, leave Power BI Desktop open.

*You’ll configure data model tables and relationships in the* ***Model Data in Power BI Desktop, Part 1*** *lab.*

**Congratulations!**

Click **Next** to proceed to the **Review Questions**

Click [**here**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#toc) to return to the **Table of Contents**.

**Model Data in Power BI Desktop, Part 1**

**The estimated time to complete the lab is 45 minutes**

In this lab you will commence developing the data model. It will involve creating relationships between tables, and then configuring table and column properties to improve the friendliness and usability of the data model. You will also create hierarchies and create quick measures.

In this lab you learn how to:

* Create model relationships
* Configure table and column properties
* Create hierarchies
* Create quick measures

**Lab story**

This lab is one of many in a series of labs that was designed as a complete story from data preparation to publication as reports and dashboards. You can complete the labs in any order. However, if you intend to work through multiple labs, for the first 10 labs, we suggest you do them in the following order:

1. Prepare Data in Power BI Desktop
2. Load Data in Power BI Desktop
3. Model Data in Power BI Desktop, Part 1
4. Model Data in Power BI Desktop, Part 2
5. Create DAX Calculations in Power BI Desktop, Part 1
6. Create DAX Calculations in Power BI Desktop, Part 2
7. Design a Report in Power BI Desktop, Part 1
8. Design a Report in Power BI Desktop, Part 2
9. Create a Power BI Dashboard
10. Create a Power BI Paginated Report
11. Perform Data Analysis in Power BI Desktop

**Exercise 1: Create Model Relationships**

In this exercise you will create model relationships.

**Task 1: Get started**

In this task you will setup the environment for the lab.

*Important: If you are continuing on from the previous lab (and you completed that lab successfully), do not complete this task; instead, continue from the next task.*

1. To open the Power BI Desktop, on the taskbar, click the Microsoft Power BI Desktop shortcut.



1. To close the getting started window, at the top-left of the window, click **X**.

Picture 11

1. To open the starter Power BI Desktop file, click the **File** ribbon tab to open the backstage view.
2. Select **Open Report**.



1. Click **Browse Reports**.

Picture 8

1. In the **Open** window, navigate to the **D:\DA100\Labs\03-configure-data-model-in-power-bi-desktop\Starter** folder.
2. Select the **Sales Analysis** file.
3. Click **Open**.

Picture 7

1. Close any informational windows that may open.
2. To create a copy of the file, click the **File** ribbon tab to open the backstage view.
3. Select **Save As**.



1. If prompted to apply changes, click **Apply**.



1. In the **Save As** window, navigate to the **D:\DA100\MySolution** folder.
2. Click **Save**.

Picture 3

**Task 2: Create model relationships**

In this task you will create model relationships.

1. In Power BI Desktop, at the left, click the **Model** view icon.

Picture 1

1. If you do not see all seven tables, scroll horizontally to the right, and then drag and arrange the tables more closely together so they can all be seen at the same time.

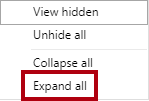
*Tip: You can also use the zoom control located at the bottom of the window.*

*In Model view, it’s possible to view each table and relationships (connectors between tables). Presently, there are no relationships because in the* ***Prepare Data in Power BI Desktop*** *lab, you disabled the data load relationship options.*

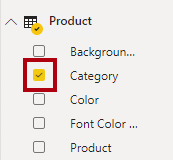
1. To return to Report view, at the left, click the **Report** view icon.

Picture 327

1. To view all table fields, in the **Fields** pane, right-click an empty area, and then select **Expand All**.

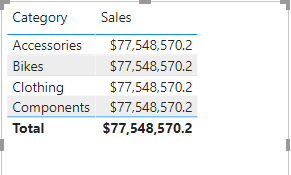


1. To create a table visual, in the **Fields** pane, from inside the **Product** table, check the **Category** field.



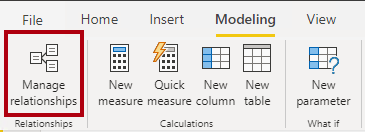
*The labs use a shorthand notation to reference a field. It will look like this:* ***Product | Category****. In this example,* ***Product*** *is the table name and* ***Category*** *is the field name.*

1. To add an additional column to the table, in the **Fields** pane, check the **Sales | Sales** field.
2. Notice that the table visual lists four product categories, and that the sales value is the same for each, and the same for the total.

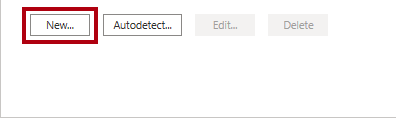


*The issue is that the table is based on fields from different tables. The expectation is that each product category displays the sales for that category. However, because there isn’t a model relationship between these tables, the* ***Sales*** *table is not filtered. You’ll now add a relationship to propagate filters between the tables.*

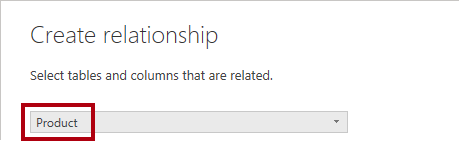
1. On the **Modeling** ribbon tab, from inside the **Relationships** group, click **Manage Relationships**.



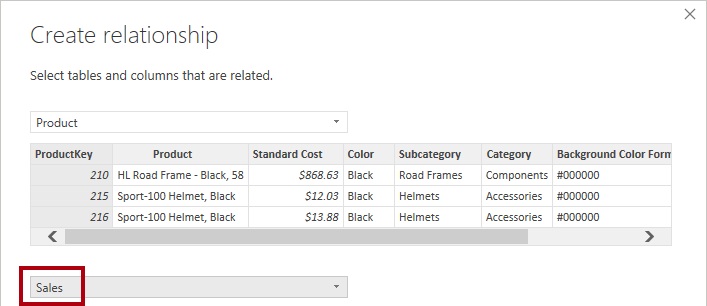
1. In the **Manage Relationships** window, notice that no relationships are yet defined.
2. To create a relationship, click **New**.



1. In the **Create Relationship** window, in the first dropdown list, select the **Product** table.



1. In the second dropdown list (beneath the **Product** table grid), select the **Sales** table.



1. Notice the **ProductKey** columns in each table have been automatically selected.

*The columns were selected because they share the same name and data type.*

1. In the **Cardinality** dropdown list, notice that **One To Many (1:\*)** is selected.

*The cardinality was automatically detected, because Power BI understands that the* ***ProductKey*** *column from the* ***Product*** *table contains unique values. One-to-many relationships are the most common cardinality, and all relationship you create in this lab will be this type. You’ll work with a Many-to-many cardinality in the* ***Model Data in Power BI Desktop, Part 2*** *lab.*

1. In the **Cross Filter Direction** dropdown list, notice that **Single** is selected.

*Single filter direction means that filters propagate from the “one side” to the “many side”. In this case, it means filters applied to the* ***Product*** *table will propagate to the* ***Sales*** *table, but not in the opposite direction. You’ll work with a bi-directional relationship in the* ***Model Data in Power BI Desktop, Part 2*** *lab.*

1. Notice that the **Mark This Relationship Active** is checked.

*Active relationships propagate filters. It’s possible to mark a relationship as inactive so filters don’t propagate. Inactive relationships can exist when there are multiple relationship paths between tables. In this case, model calculations can use special functions to activate them. You’ll work with an inactive relationship in the* ***Model Data in Power BI Desktop, Part 2*** *lab.*

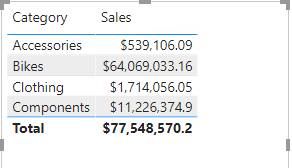
1. Click **OK**.



1. In the **Manage Relationships** window, notice that the new relationship is listed, and then click **Close**.

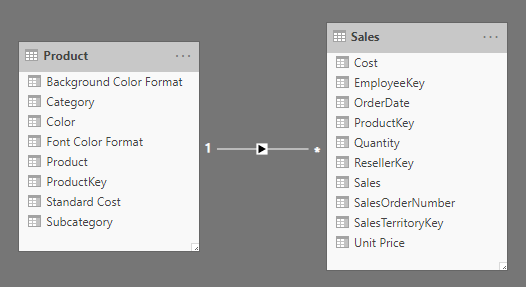


1. In the report, notice that the table visual updated to display different values for each product category.



*Filters applied to the* ***Product*** *table now propagate to the* ***Sales*** *table.*

1. Switch to Model view, and then notice there is now a connector between the two tables (it does not matter if the tables are positioned next to each other).



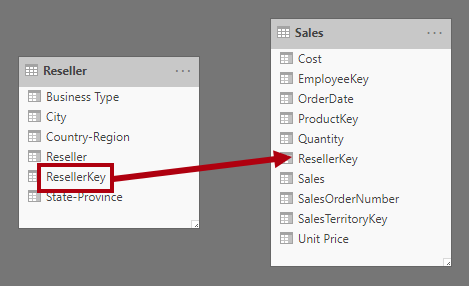
1. In the diagram, notice that you can interpret the cardinality which is represented by the **1** and **\*** indicators.

*Filter direction is represented by the arrow head. A solid line represents an active relationship; a dashed line represents an inactive relationship.*

1. Hover the cursor over the relationship to highlight the related columns.

*There’s an easier way to create a relationship. In the model diagram, you can drag and drop columns to create a new relationship.*

1. To create a new relationship using a different technique, from the **Reseller** table, drag the **ResellerKey** column on to the **ResellerKey** column of the **Sales** table.

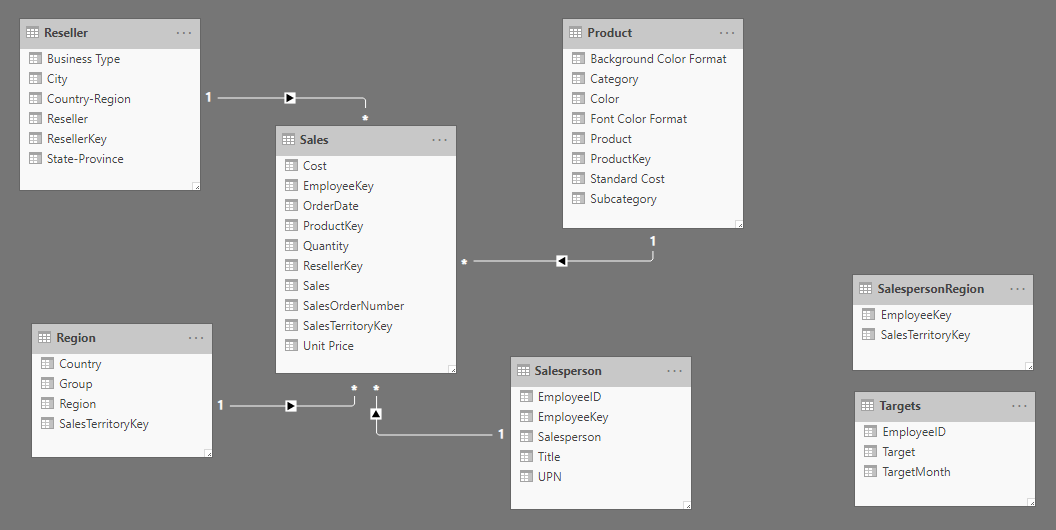


*Tip: Sometimes a column doesn’t want to be dragged. If this situation arises, select a different column, and then select the column you intend to drag again, and then try again. Ensure that you see the new relationship added to the diagram.*

1. Use the new technique to create the following two model relationships:
   * **Region | SalesTerritoryKey** to **Sales | SalesTerritoryKey**
   * **Salesperson | EmployeeKey** to **Sales | EmployeeKey**

*In this lab the* ***SalespersonRegion*** *and* ***Targets*** *tables will remain disconnected. There’s a many-to-many relationship between salespeople and regions, and you’ll work with this advanced scenario in the* ***Model Data in Power BI Desktop, Part 2*** *lab.*

1. In the diagram, arrange the tables so that the **Sales** table is positioned in the center of the diagram, and the related tables are arranged about it. Position the disconnected tables to the side.



1. Save the Power BI Desktop file.

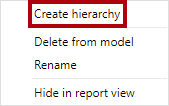
**Exercise 2: Configure Tables**

In this exercise you will configure each table by creating hierarchies, and hiding, formatting, and categorizing columns.

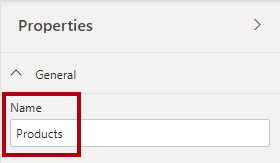
**Task 1: Configure the Product table**

In this task you will configure the **Product** table.

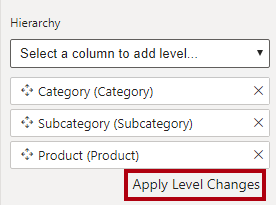
1. In Model view, in the **Fields** pane, if necessary, expand the **Product** table to reveal all fields.
2. To create a hierarchy, in the **Fields** pane, right-click the **Category** column, and then select **Create Hierarchy**.



1. In the **Properties** pane (to the left of the **Fields** pane), in the **Name** box, replace the text with **Products**.

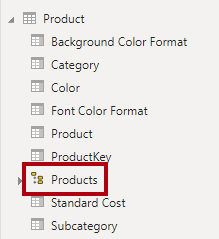


1. To add the second level to the hierarchy, in the **Properties** pane, in the **Hierarchy** dropdown list, select **Subcategory** (you might need to scroll down inside the pane).
2. To add the third level to the hierarchy, in the **Hierarchy** dropdown list, select **Product**.
3. To complete the hierarchy design, click **Apply Level Changes**.

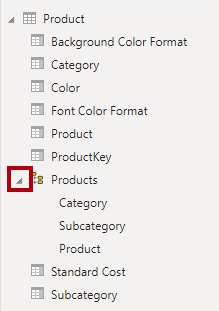


*Tip: Don’t forget to click* ***Apply Level Changes****—it’s a common mistake to overlook this step.*

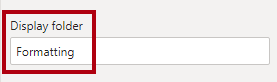
1. In the **Fields** pane, notice the **Products** hierarchy.



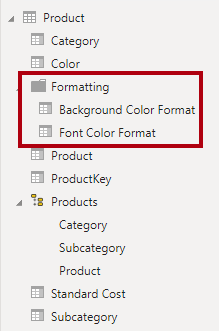
1. To reveal the hierarchy levels, expand the **Products** hierarchy.



1. To organize columns into a display folder, in the **Fields** pane, first select the **Background Color Format** column.
2. While pressing the **Ctrl** key, select the **Font Color Format** column.
3. In the **Properties** pane, in the **Display Folder** box, enter **Formatting** and press **Enter**.



1. In the **Fields** pane, notice that the two columns are now inside a folder.

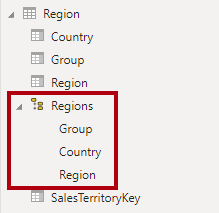


*Display folders are a great way to declutter tables—especially for tables that comprise many fields.*

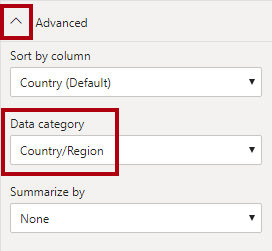
**Task 2: Configure the Region table**

In this task you will configure the **Region** table.

1. In the **Region** table, create a hierarchy named **Regions**, with the following three levels:
   * Group
   * Country
   * Region



1. Select the **Country** column (not the **Country** hierarchy level).
2. In the **Properties** pane, expand the **Advanced** section (at the bottom of the pane), and then in the **Data Category** dropdown list, select **Country/Region**.

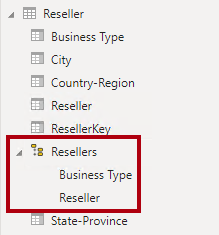


*Data categorization can provide hints to the report designer. In this case, categorizing the column as country or region provides more accurate information to Power BI when it renders a map visualization.*

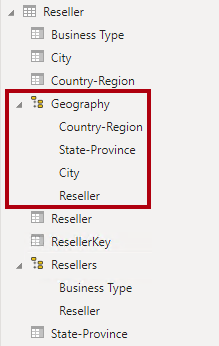
**Task 3: Configure the Reseller table**

In this task you will configure the **Reseller** table.

1. In the **Reseller** table, create a hierarchy named **Resellers**, with the following two levels:
   * Business Type
   * Reseller



1. Create a second hierarchy named **Geography**, with the following four levels:
   * Country-Region
   * State-Province
   * City
   * Reseller



1. Categorize the following three columns:
   * **Country-Region** as **Country/Region**
   * **State-Province** as **State or Province**
   * **City** as **City**

**Task 4: Configure the Sales table**

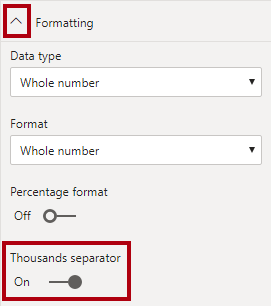
In this task you will configure the **Sales** table.

1. In the **Sales** table, select the **Cost** column.
2. In the **Properties** pane, in the **Description** box, enter: **Based on standard cost**

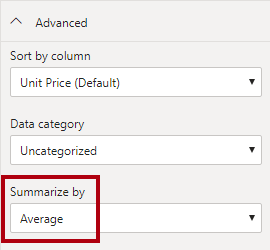


*Descriptions can be applied to tables, columns, hierarchies, or measures. In the* ***Fields*** *pane, description text is revealed in a tooltip when a report author hovers their cursor over the field.*

1. Select the **Quantity** column.
2. In the **Properties** pane, from inside the **Formatting** section, slide the **Thousands Separator** property to **On**.



1. Select the **Unit Price** column.
2. In the **Properties** pane, from inside the **Formatting** section, slide the **Decimal Places** property to **2**.
3. In the **Advanced** group (you may need to scroll down to locate it), in the **Summarize By** dropdown list, select **Average**.

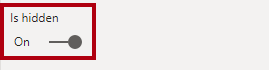


*By default, numeric columns will summarize by summing values together. This default behavior is not suitable for a column like* ***Unit Price****, which represents a rate. Setting the default summarization to average will produce a meaningful result.*

**Task 5: Bulk update properties**

In this task you will update multiple columns using single bulk updates. You will use this approach to hide columns, and format column values.

1. In the **Fields** pane, select the **Product | ProductKey** column.
2. While pressing the **Ctrl** key, select the following 13 columns (spanning multiple tables):
   * Region | SalesTerritoryKey
   * Reseller | ResellerKey
   * Sales | EmployeeKey
   * Sales | ResellerKey
   * Sales | SalesOrderNumber
   * Sales | SalesTerritoryKey
   * Salesperson | EmployeeID
   * Salesperson | EmployeeKey
   * Salesperson | UPN
   * SalespersonRegion | EmployeeKey
   * SalespersonRegion | SalesTerritoryKey
   * Targets | EmployeeID
3. In the **Properties** pane, slide the **Is Hidden** property to **On**.



*The columns were hidden because they’re either used by relationships or will be used in row-level security configuration or calculation logic.*

*You’ll define row-level security in the* ***Model Data in Power BI Desktop, Part 2*** *lab using the* ***UPN*** *column. You’ll use the* ***SalesOrderNumber*** *in a calculation in the* ***Create DAX Calculations in Power BI Desktop, Part 1*** *lab.*

1. Multi-select the following three columns:
   * Product | Standard Cost
   * Sales | Cost
   * Sales | Sales
2. In the **Properties** pane, from inside the **Formatting** section, set the **Decimal Places** property to **0** (zero).



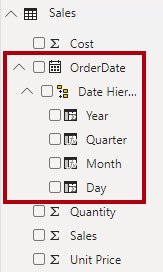
**Exercise 3: Review the Model Interface**

In this exercise you will switch to Report view, and review the model interface.

**Task 1: Review the model interface**

In this task you will switch to Report view, and review the model interface.

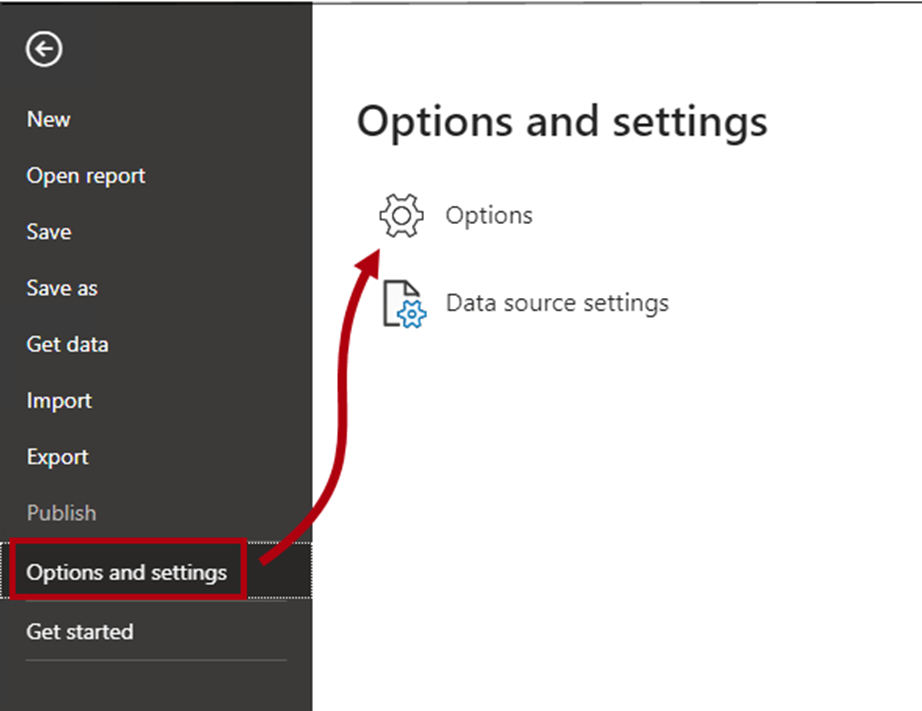
1. Switch to Report view.
2. In the **Fields** pane, notice the following:
   * Columns, hierarchies and their levels are fields, which can be used to configure report visuals
   * Only fields relevant to report authoring are visible
   * The **SalespersonRegion** table is not visible—because all of its fields are hidden
   * Spatial fields in the **Region** and **Reseller** table are adorned with a spatial icon
   * Fields adorned with the sigma symbol (Ʃ) will summarize, by default
   * A tooltip appears when hovering the cursor over the **Sales | Cost** field
3. Expand the **Sales | OrderDate** field, and then notice that it reveals a date hierarchy.



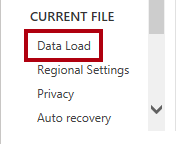
*The* ***Targets | TargetMonth*** *field delivers a similar hierarchy. These hierarchies were not created by you. They were created automatically. There is a problem, however. The Adventure Works financial year commences on July 1 of each year. But, in these automatically created date hierarchies, the date hierarchy year commences on January 1 of each year.*

*You’ll now turn this automatic behavior off. In the* ***Create DAX Calculations in Power BI Desktop, Part 1*** *lab, you’ll use DAX to create a date table, and configure it define the Adventure Works’ calendar.*

1. To turn off auto/date time, click the **File** ribbon tab to open the backstage view.
2. At the left, select **Options and Settings**, and then select **Options**.



1. In the **Options** window, at the left, in the **Current File** group, select **Data Load**.



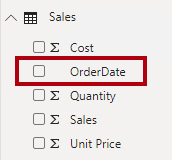
1. In the **Time Intelligence** section, uncheck **Auto Date/Time**.

Picture 362

1. Click **OK**.



1. In the **Fields** pane, notice that the date hierarchies are no longer available.



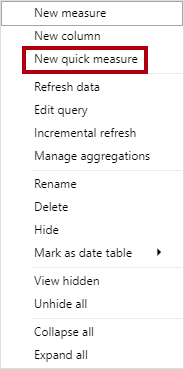
**Exercise 4: Create Quick Measures**

In this exercise you will create two quick measures.

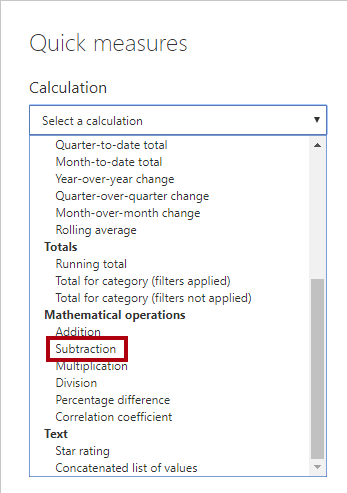
**Task 1: Create quick measures**

In this task you will create two quick measures to calculate profit and profit margin.

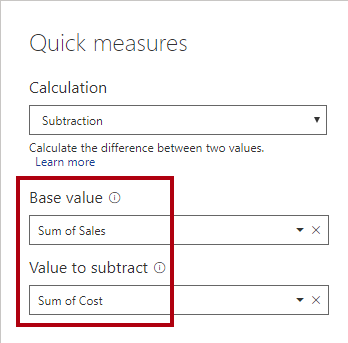
1. In the **Fields** pane, right-click the **Sales** table, and then select **New Quick Measure**.



1. In the **Quick Measures** window, in the **Calculation** dropdown list, from inside the **Mathematical Operations** group, select **Subtraction**.



1. In the **Fields** pane of the **Quick Measures** window, expand the **Sales** table.
2. Drag the **Sales** field into the **Base Value** box.
3. Drag the **Cost** field into the **Value to Subtract** box.

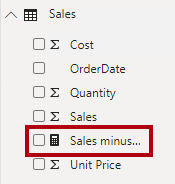


1. Click **OK**.



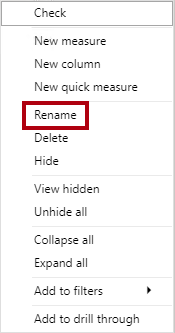
*A quick measure creates the calculation formula for you. They’re easy and fast to create for simple and common calculations. You’ll create measures without using this tool in the* ***Create DAX Calculations in Power BI Desktop, Part 1*** *lab.*

1. In the **Fields** pane, inside the **Sales** table, notice that new measure.



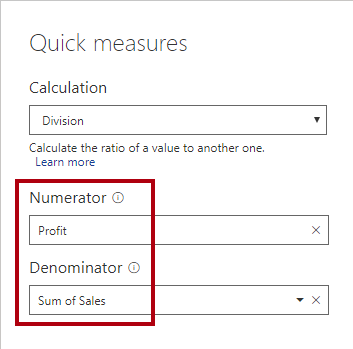
*Measures are adorned with the calculator icon.*

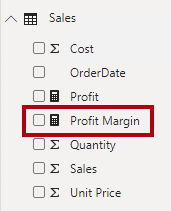
1. To rename the measure, right-click it, and then select **Rename**.



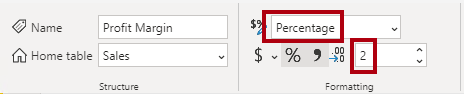
*Tip: To rename a field, you can also double-click it, or select it and press* ***F2****.*

1. Rename the measure to **Profit**, and then press **Enter**.
2. In the **Sales** table, add a second quick measure, based on the following requirements:
   * Use the **Division** mathematical operation
   * Set the **Numerator** to the **Sales | Profit** field
   * Set the **Denominator** to **Sales | Sales** field
   * Rename the measure as **Profit Margin**

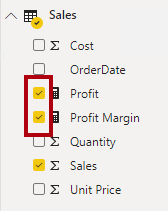




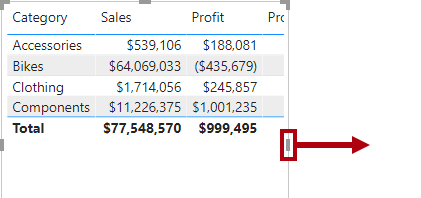
1. Ensure the **Profit Margin** measure is selected, and then on the **Measure Tools** contextual ribbon, set the format to **Percentage**, with two decimal places.



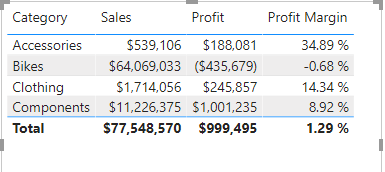
1. To test the two measures, first select the table visual on the report page.
2. In the **Fields** pane, check the two measures.



1. Click and drag the right guide to widen the table visual.



1. Verify that the measures produce reasonable results that are correctly formatted.



**Task 2: Finish up**

In this task you will complete the lab.

1. To remove the table, select the table (by clicking it), and then press the **Delete** key.
2. Save the Power BI Desktop file.
3. If prompted to apply queries, click **Apply Later**.
4. If you intend to start the next lab, leave Power BI Desktop open.

*You’ll enhance the data model by configuring a many-to-many relationship and row-level security in the* ***Model Data in Power BI Desktop, Part 2*** *lab.*

**Congratulations**

You have successfully completed this exercise. Click **Next** to advance to the next lab.

Click [**here**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#toc) to return to the **Table of Contents**.

All Modules: Analyzing Data with Microsoft Power BI

9 Hr 44 Min Remaining

**Model Data in Power BI Desktop, Part 2**

**The estimated time to complete the lab is 45 minutes**

In this lab you will create a many-to-many relationship between the **Salesperson** table and the **Sales** table.

In this lab you learn how to:

* Configure many-to-many relationships

**Lab story**

This lab is one of many in a series of labs that was designed as a complete story from data preparation to publication as reports and dashboards. You can complete the labs in any order. However, if you intend to work through multiple labs, for the first 10 labs, we suggest you do them in the following order:

1. Prepare Data in Power BI Desktop
2. Load Data in Power BI Desktop
3. Model Data in Power BI Desktop, Part 1
4. Model Data in Power BI Desktop, Part 2
5. Create DAX Calculations in Power BI Desktop, Part 1
6. Create DAX Calculations in Power BI Desktop, Part 2
7. Design a Report in Power BI Desktop, Part 1
8. Design a Report in Power BI Desktop, Part 2
9. Create a Power BI Dashboard
10. Create a Power BI Paginated Report
11. Perform Data Analysis in Power BI Desktop

**Exercise 1: Create a Many-to many Relationship**

In this exercise you will create a many-to-many relationship between the **Salesperson** table and the **Sales** table.

**Task 1: Get started**

In this task you will setup the environment for the lab.

*Important: If you are continuing on from the previous lab (and you completed that lab successfully), do not complete this task; instead, continue from the next task.*

1. To open the Power BI Desktop, on the taskbar, click the Microsoft Power BI Desktop shortcut.



1. To close the getting started window, at the top-left of the window, click **X**.

Picture 7

1. To open the starter Power BI Desktop file, click the **File** ribbon tab to open the backstage view.
2. Select **Open Report**.



1. Click **Browse Reports**.

Picture 5

1. In the **Open** window, navigate to the **D:\DA100\Labs\04-configure-data-model-in-power-bi-desktop-advanced\Starter** folder.
2. Select the **Sales Analysis** file.
3. Click **Open**.

Picture 4

1. Close any informational windows that may open.
2. To create a copy of the file, click the **File** ribbon tab to open the backstage view.
3. Select **Save As**.



1. If prompted to apply changes, click **Apply**.



1. In the **Save As** window, navigate to the **D:\DA100\MySolution** folder.
2. Click **Save**.

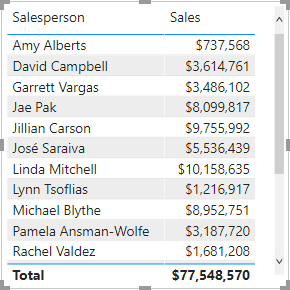
Picture 2

**Task 2: Create a many-to-many relationship**

In this task you will create a many-to-many relationship between the **Salesperson** table and the **Sales** table.

1. In Power BI Desktop, in Report view, in the **Fields** pane, check the follow two fields to create a table visual:
   * Salesperson | Salesperson
   * Sales | Sales

*The labs use a shorthand notation to reference a field. It will look like this:* ***Salesperson | Salesperson*** *. In this example,* ***Salesperson*** *is the table name and* ***Salesperson*** *is the field name.*



*The table displays sales made by each salesperson. However, there’s another relationship between salespeople and sales. Some salespeople belong to one, two, or possibly more sales regions. In addition, sales regions can have multiple salespeople assigned to them.*

*From a performance management perspective, a salesperson’s sales (based on their assigned territories) need to be analyzed and compared with sales targets. You’ll create relationships to support this analysis in the next exercise.*

1. Notice that Michael Blythe has sold almost $9 million.
2. Switch to Model view.

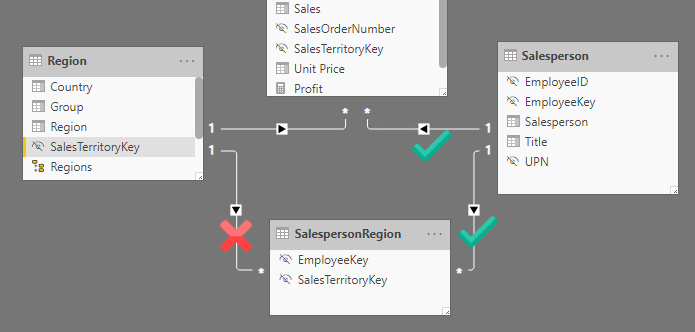
Picture 10

1. Drag the **SalespersonRegion** table to position it between the **Region** and **Salesperson** tables.
2. Use the drag-and-drop technique to create the following two model relationships:
   * **Salesperson | EmployeeKey** to **SalespersonRegion | EmployeeKey**
   * **Region | SalesTerritoryKey** to **SalespersonRegion | SalesTerritoryKey**

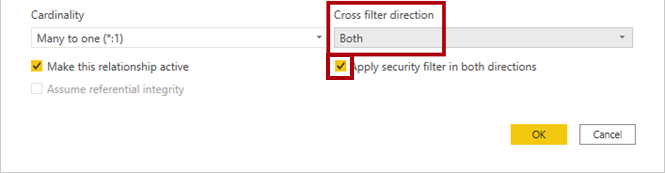
*The* ***SalespersonRegion*** *table can be considered to be a bridging table.*

1. Switch to Report view, and then notice that the visual has not updated—the sales result for Michael Blythe has not changed.
2. Switch back to Model view, and then follow the relationship filter directions (arrowhead) from the **Salesperson** table.

*Consider that the* ***Salesperson*** *table filters the* ***Sales*** *table. It also filters the* ***SalespersonRegion*** *table, but it does not continue by propagating filters to the* ***Region*** *table (the arrowhead is pointing the wrong direction).*



1. To edit the relationship between the **Region** and **SalespersonRegion** tables, double-click the relationship.
2. In the **Edit Relationship** window, in the **Cross Filter Direction** dropdown list, select **Both**.
3. Check the **Apply Security Filter in Both Directions** checkbox.



1. Click **OK**.



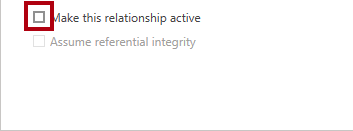
1. Notice that the relationship has a double arrowhead.

Picture 382

1. Switch to Report view, and then notice that the sales values have still not changed.

*The issue now relates to the fact that there are two possible filter propagation paths between the* ***Salesperson*** *and* ***Sales*** *tables. This ambiguity is internally resolved, based on a “least number of tables” assessment. To be clear, you shouldn’t design models with this type of ambiguity—the issue will be addressed in part later in this lab, and by the completion of the* ***Create DAX Calculations in Power BI Desktop, Part 1*** *lab.*

1. Switch to Model view.
2. To force filter propagation via the bridging table, edit (double-click) the relationship between the **Salesperson** and **Sales** tables.
3. In the **Edit Relationship** window, uncheck the **Make This Relationship Active** checkbox.



1. Click **OK**.

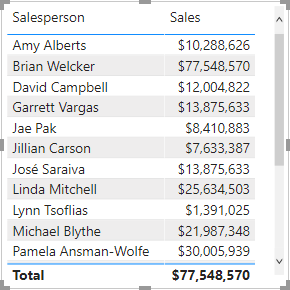


*The filter propagation will now follow the only active path.*

1. In the diagram, notice that the inactive relationship is represented by a dashed line.

Picture 5697

1. Switch to Report view, and then notice that the sales for Michael Blythe is now nearly $22 million.



1. Notice also, that the sales for each salesperson—if added—would exceed the table total.

*It’s a common observation of a many-to-many relationship due to the double, triple, etc. counting of regional sales results. Consider Brian Welcker, the second salesperson listed. His sales amount equals the total sales amount. It’s the correct result simply due to the fact the he’s the Director of Sales; his sales are measured by the sales of all regions.*

*While the many-to-many relationship is now working, it’s now not possible to analyze sales made by a salesperson (because the relationship is inactive). You’ll be able to reactive the relationship when you introduce a calculated table that to represent salesperson for performance analysis (of their regions) in the* ***Create DAX Calculations in Power BI Desktop, Part 1*** *lab.*

1. Switch to Modeling view, and then in the diagram, select the **Salesperson** table.
2. In the **Properties** pane, in the **Name** box, replace the text with **Salesperson (Performance)**.

*The renamed table now reflects its purpose: it’s used to report and analyze the performance of salespeople based on the sales of their assigned sales regions.*

**Task 3: Relate the Targets table**

In this task you will create a relationship to the **Targets** table

1. Create a relationship from the **Salesperson (Performance) | EmployeeID** column and the **Targets | EmployeeID** column.
2. In Report view, add the **Targets | Target** field to the table visual.
3. Resize the table visual so all columns are visible.



*It’s now possible to visualize sales and targets—but take care for two reasons. First, there’s no filter on a time period, and so targets also include future target amounts. Second, targets are not additive, and so the total should not be displayed. They can either disabled by formatting the visual or removed by using calculation logic. You’ll follow the second approach by creationg a target measure in the* ***Create DAX Calculations in Power BI Desktop, Part 2*** *lab that’ll return BLANK when more than one salesperson is filtered.*

**Task 4: Finish up**

In this task you will complete the lab.

1. Save the Power BI Desktop file.
2. If prompted to apply queries, click **Apply Later**.
3. If you intend to start the next lab, leave Power BI Desktop open.

*You’ll enhance the data model with calculations using DAX in the* ***Create DAX Calculations in Power BI Desktop, Part 2*** *lab.*

**Congratulations!**

Click **Next** to proceed to the **Review Questions**

Click [**here**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#toc) to return to the **Table of Contents**.

**Model Data in Power BI Desktop, Part 2**

**The estimated time to complete the lab is 45 minutes**

In this lab you will create a many-to-many relationship between the **Salesperson** table and the **Sales** table.

In this lab you learn how to:

* Configure many-to-many relationships

**Lab story**

This lab is one of many in a series of labs that was designed as a complete story from data preparation to publication as reports and dashboards. You can complete the labs in any order. However, if you intend to work through multiple labs, for the first 10 labs, we suggest you do them in the following order:

1. Prepare Data in Power BI Desktop
2. Load Data in Power BI Desktop
3. Model Data in Power BI Desktop, Part 1
4. Model Data in Power BI Desktop, Part 2
5. Create DAX Calculations in Power BI Desktop, Part 1
6. Create DAX Calculations in Power BI Desktop, Part 2
7. Design a Report in Power BI Desktop, Part 1
8. Design a Report in Power BI Desktop, Part 2
9. Create a Power BI Dashboard
10. Create a Power BI Paginated Report
11. Perform Data Analysis in Power BI Desktop

**Exercise 1: Create a Many-to many Relationship**

In this exercise you will create a many-to-many relationship between the **Salesperson** table and the **Sales** table.

**Task 1: Get started**

In this task you will setup the environment for the lab.

*Important: If you are continuing on from the previous lab (and you completed that lab successfully), do not complete this task; instead, continue from the next task.*

1. To open the Power BI Desktop, on the taskbar, click the Microsoft Power BI Desktop shortcut.



1. To close the getting started window, at the top-left of the window, click **X**.

Picture 7

1. To open the starter Power BI Desktop file, click the **File** ribbon tab to open the backstage view.
2. Select **Open Report**.



1. Click **Browse Reports**.

Picture 5

1. In the **Open** window, navigate to the **D:\DA100\Labs\04-configure-data-model-in-power-bi-desktop-advanced\Starter** folder.
2. Select the **Sales Analysis** file.
3. Click **Open**.

Picture 4

1. Close any informational windows that may open.
2. To create a copy of the file, click the **File** ribbon tab to open the backstage view.
3. Select **Save As**.



1. If prompted to apply changes, click **Apply**.



1. In the **Save As** window, navigate to the **D:\DA100\MySolution** folder.
2. Click **Save**.

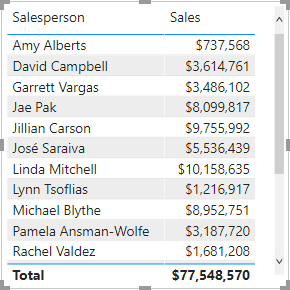
Picture 2

**Task 2: Create a many-to-many relationship**

In this task you will create a many-to-many relationship between the **Salesperson** table and the **Sales** table.

1. In Power BI Desktop, in Report view, in the **Fields** pane, check the follow two fields to create a table visual:
   * Salesperson | Salesperson
   * Sales | Sales

*The labs use a shorthand notation to reference a field. It will look like this:* ***Salesperson | Salesperson*** *. In this example,* ***Salesperson*** *is the table name and* ***Salesperson*** *is the field name.*



*The table displays sales made by each salesperson. However, there’s another relationship between salespeople and sales. Some salespeople belong to one, two, or possibly more sales regions. In addition, sales regions can have multiple salespeople assigned to them.*

*From a performance management perspective, a salesperson’s sales (based on their assigned territories) need to be analyzed and compared with sales targets. You’ll create relationships to support this analysis in the next exercise.*

1. Notice that Michael Blythe has sold almost $9 million.
2. Switch to Model view.

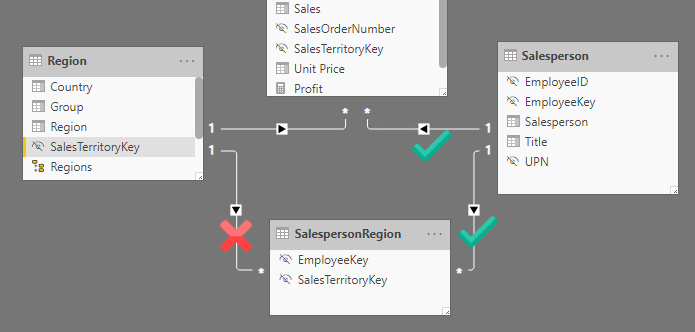
Picture 10

1. Drag the **SalespersonRegion** table to position it between the **Region** and **Salesperson** tables.
2. Use the drag-and-drop technique to create the following two model relationships:
   * **Salesperson | EmployeeKey** to **SalespersonRegion | EmployeeKey**
   * **Region | SalesTerritoryKey** to **SalespersonRegion | SalesTerritoryKey**

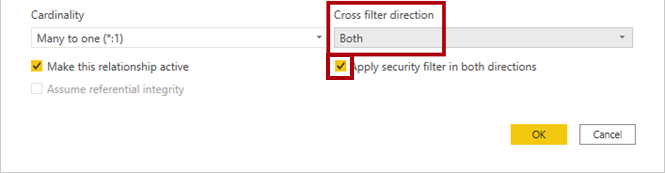
*The* ***SalespersonRegion*** *table can be considered to be a bridging table.*

1. Switch to Report view, and then notice that the visual has not updated—the sales result for Michael Blythe has not changed.
2. Switch back to Model view, and then follow the relationship filter directions (arrowhead) from the **Salesperson** table.

*Consider that the* ***Salesperson*** *table filters the* ***Sales*** *table. It also filters the* ***SalespersonRegion*** *table, but it does not continue by propagating filters to the* ***Region*** *table (the arrowhead is pointing the wrong direction).*



1. To edit the relationship between the **Region** and **SalespersonRegion** tables, double-click the relationship.
2. In the **Edit Relationship** window, in the **Cross Filter Direction** dropdown list, select **Both**.
3. Check the **Apply Security Filter in Both Directions** checkbox.



1. Click **OK**.



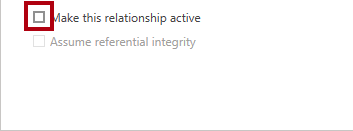
1. Notice that the relationship has a double arrowhead.

Picture 382

1. Switch to Report view, and then notice that the sales values have still not changed.

*The issue now relates to the fact that there are two possible filter propagation paths between the* ***Salesperson*** *and* ***Sales*** *tables. This ambiguity is internally resolved, based on a “least number of tables” assessment. To be clear, you shouldn’t design models with this type of ambiguity—the issue will be addressed in part later in this lab, and by the completion of the* ***Create DAX Calculations in Power BI Desktop, Part 1*** *lab.*

1. Switch to Model view.
2. To force filter propagation via the bridging table, edit (double-click) the relationship between the **Salesperson** and **Sales** tables.
3. In the **Edit Relationship** window, uncheck the **Make This Relationship Active** checkbox.



1. Click **OK**.

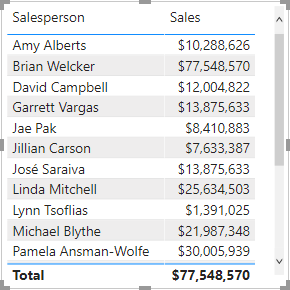


*The filter propagation will now follow the only active path.*

1. In the diagram, notice that the inactive relationship is represented by a dashed line.

Picture 5697

1. Switch to Report view, and then notice that the sales for Michael Blythe is now nearly $22 million.



1. Notice also, that the sales for each salesperson—if added—would exceed the table total.

*It’s a common observation of a many-to-many relationship due to the double, triple, etc. counting of regional sales results. Consider Brian Welcker, the second salesperson listed. His sales amount equals the total sales amount. It’s the correct result simply due to the fact the he’s the Director of Sales; his sales are measured by the sales of all regions.*

*While the many-to-many relationship is now working, it’s now not possible to analyze sales made by a salesperson (because the relationship is inactive). You’ll be able to reactive the relationship when you introduce a calculated table that to represent salesperson for performance analysis (of their regions) in the* ***Create DAX Calculations in Power BI Desktop, Part 1*** *lab.*

1. Switch to Modeling view, and then in the diagram, select the **Salesperson** table.
2. In the **Properties** pane, in the **Name** box, replace the text with **Salesperson (Performance)**.

*The renamed table now reflects its purpose: it’s used to report and analyze the performance of salespeople based on the sales of their assigned sales regions.*

**Task 3: Relate the Targets table**

In this task you will create a relationship to the **Targets** table

1. Create a relationship from the **Salesperson (Performance) | EmployeeID** column and the **Targets | EmployeeID** column.
2. In Report view, add the **Targets | Target** field to the table visual.
3. Resize the table visual so all columns are visible.



*It’s now possible to visualize sales and targets—but take care for two reasons. First, there’s no filter on a time period, and so targets also include future target amounts. Second, targets are not additive, and so the total should not be displayed. They can either disabled by formatting the visual or removed by using calculation logic. You’ll follow the second approach by creationg a target measure in the* ***Create DAX Calculations in Power BI Desktop, Part 2*** *lab that’ll return BLANK when more than one salesperson is filtered.*

**Task 4: Finish up**

In this task you will complete the lab.

1. Save the Power BI Desktop file.
2. If prompted to apply queries, click **Apply Later**.
3. If you intend to start the next lab, leave Power BI Desktop open.

*You’ll enhance the data model with calculations using DAX in the* ***Create DAX Calculations in Power BI Desktop, Part 2*** *lab.*

**Congratulations!**

Click **Next** to proceed to the **Review Questions**

Click [**here**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#toc) to return to the **Table of Contents**.

All Modules: Analyzing Data with Microsoft Power BI

5 Hr 27 Min Remaining

**Create DAX Calculations in Power BI Desktop, Part 1**

**The estimated time to complete the lab is 45 minutes**

In this lab you will create calculated tables, calculated columns, and simple measures using Data Analysis Expressions (DAX).

In this lab you learn how to:

* Create calculated tables
* Create calculated columns
* Create measures

**Lab story**

This lab is one of many in a series of labs that was designed as a complete story from data preparation to publication as reports and dashboards. You can complete the labs in any order. However, if you intend to work through multiple labs, for the first 10 labs, we suggest you do them in the following order:

1. Prepare Data in Power BI Desktop
2. Load Data in Power BI Desktop
3. Model Data in Power BI Desktop, Part 1
4. Model Data in Power BI Desktop, Part 2
5. Create DAX Calculations in Power BI Desktop, Part 1
6. Create DAX Calculations in Power BI Desktop, Part 2
7. Design a Report in Power BI Desktop, Part 1
8. Design a Report in Power BI Desktop, Part 2
9. Create a Power BI Dashboard
10. Create a Power BI Paginated Report
11. Perform Data Analysis in Power BI Desktop

**Exercise 1: Create Calculated Tables**

In this exercise you will create two calculated tables. The first will be the **Salesperson** table, to allow a direct relationship between it and the **Sales** table. The second will be the **Date** table.

**Task 1: Get started**

In this task you will setup the environment for the lab.

*Important: If you are continuing on from the previous lab (and you completed that lab successfully), do not complete this task; instead, continue from the next task.*

1. To open the Power BI Desktop, on the taskbar, click the Microsoft Power BI Desktop shortcut.



1. To close the getting started window, at the top-left of the window, click **X**.

Picture 49

1. To open the starter Power BI Desktop file, click the **File** ribbon tab to open the backstage view.
2. Select **Open Report**.



1. Click **Browse Reports**.

Picture 47

1. In the **Open** window, navigate to the **D:\DA100\Labs\05-create-dax-calculations-in-power-bi-desktop\Starter** folder.
2. Select the **Sales Analysis** file.
3. Click **Open**.

Picture 35

1. Close any informational windows that may open.
2. To create a copy of the file, click the **File** ribbon tab to open the backstage view.
3. Select **Save As**.



1. If prompted to apply changes, click **Apply**.



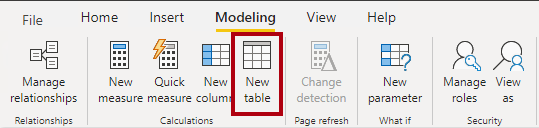
1. In the **Save As** window, navigate to the **D:\DA100\MySolution** folder.
2. Click **Save**.

Picture 13

**Task 2: Create the Salesperson table**

In this task you will create the **Salesperson** table (direct relationship to **Sales**).

1. In Power BI Desktop, in Report view, on the **Modeling** ribbon, from inside the **Calculations** group, click **New Table**.



1. In the formula bar (which opens directly beneath the ribbon when creating or editing calculations), type **Salesperson =**, press **Shift+Enter**, type **'Salesperson (Performance)'**, and then press **Enter**.

Picture 4

*For your convenience, all DAX definitions in this lab can be copied from the snippets file, located in* ***D:\DA100\Labs\05-create-dax-calculations-in-power-bi-desktop\Assets\Snippets.txt****.*

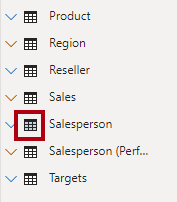
*A calculated table is created by first entering the table name, followed by the equals symbol (=), followed by a DAX formula that returns a table. Note that the table name cannot already exist in the data model.*

*The formula bar supports entering a valid DAX formula. It includes features like auto-complete, Intellisense and color-coding, enabling you to quickly and accurately enter the formula.*

*This table definition creates a copy of the* ***Salesperson (Performance)*** *table. It copies the data only, however model properties like visibility, formatting, etc. are not copied.*

*Tip: You’re encouraged to enter “white space” (i.e. carriage returns and tabs) to layout formulas in an intuitive and easy-to-read format—especially when formulas are long and complex. To enter a carriage return, press* ***Shift+Enter****. “White space” is optional.*

1. In the **Fields** pane, notice that the table icon is a shade of blue (denoting a calculated table).



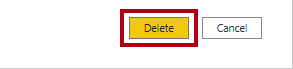
*Calculated tables are defined by using a DAX formula which returns a table. It’s important to understand that calculated tables increase the size of the data model because they materialize and store values. They’re recomputed whenever formula dependencies are refreshed, as will be the case for this data model when new (future) date values are loaded into tables.*

*Unlike Power Query-sourced tables, calculated tables can’t be used to load data from external data sources. They can only transform data based on what has already been loaded into the data model.*

1. Switch to Model view.
2. Notice that the **Salesperson** table is available (take care, it might be hidden from view, in which case scroll horizontally to locate it).
3. Create a relationship from the **Salesperson | EmployeeKey** column to the **Sales | EmployeeKey** column.
4. Right-click the inactive relationship between the **Salesperson (Performance)** and **Sales** tables, and then select **Delete**.

Picture 2

1. When prompted to confirm the deletion, click **Delete**.



1. In the **Salesperson** table, multi-select the following columns, and then hide them (set the **Is Hidden** property to **Yes**):
   * EmployeeID
   * EmployeeKey
   * UPN
2. In the model diagram, select the **Salesperson** table.
3. In the **Properties** pane, in the **Description** box, enter: **Salesperson related to Sales**

*You may recall that descriptions appear as tooltips in the* ***Fields*** *pane when the user hovers their cursor over a table or field.*

1. For the **Salesperson (Performance)** table, set the description to: **Salesperson related to region(s)**

*The data model now provides two alternatives when analyzing salespeople. The* ***Salesperson*** *table allows analyzing sales made by a salesperson, while the* ***Salesperson (Performance)*** *table allows analyzing sales made in the sales region(s) assigned to the salesperson.*

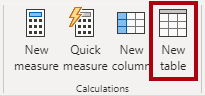
**Task 3: Create the Date table**

In this task you will create the **Date** table.

1. Switch to Data view.

Picture 29

1. On the **Home** ribbon tab, from inside the **Calculations** group, click **New Table**.



1. In the formula bar, enter the following:

**DAX**

Date =

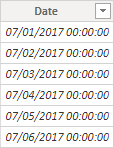
CALENDARAUTO(6)

Picture 6

*The CALENDARAUTO() function returns a single-column table consisting of date values. The “auto” behavior scans all data model date columns to determine the earliest and latest date values stored in the data model. It then creates one row for each date within this range, extending the range in either direction to ensure full years of data is stored.*

*This function can take a single optional argument that is the last month number of a year. When omitted, the value is 12, meaning that December is the last month of the year. In this case, 6 is entered, meaning that June is the last month of the year.*

1. Notice the column of date values.



*The dates shown are formatted using US regional settings (i.e. mm/dd/yyyy).*

1. At the bottom-left corner, in the status bar, notice the table statistics, confirming that 1826 rows of data have been generated, which represents five full years’ data.

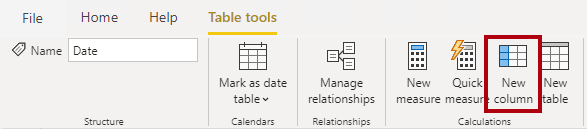
Picture 9

**Task 4: Create calculated columns**

In this task you will add additional columns to enable filtering and grouping by different time periods. You will also create a calculated column to control the sort order of other columns.

*For your convenience, all DAX definitions in this lab can be copied from the snippets file, located in* ***D:\DA100\Labs\05-create-dax-calculations-in-power-bi-desktop\Assets\Snippets.txt****.*

1. On the **Table Tools** contextual ribbon, from inside the **Calculations** group, click **New Column**.



1. In the formula bar, type the following (or copy from the snippets file), and then press **Enter**:

**DAX**

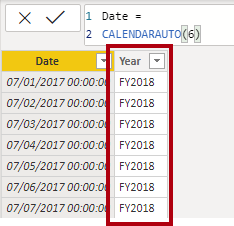
Year =

"FY" & YEAR('Date'[Date]) + IF(MONTH('Date'[Date]) > 6, 1)

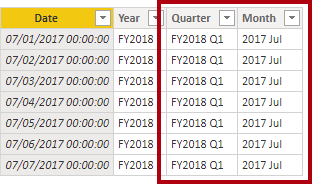
*A calculated column is created by first entering the column name, followed by the equals symbol (=), followed by a DAX formula that returns a single-value result. The column name cannot already exist in the table.*

*The formula uses the date’s year value but adds one to the year value when the month is after June. It’s how fiscal years at Adventure Works are calculated.*

1. Verify that the new column was added.



1. Use the snippets file definitions to create the following two calculated columns for the **Date** table:
   * Quarter
   * Month

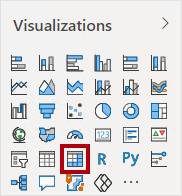


1. To validate the calculations, switch to Report view.
2. To create a new report page, at the bottom-left, click the plus icon.

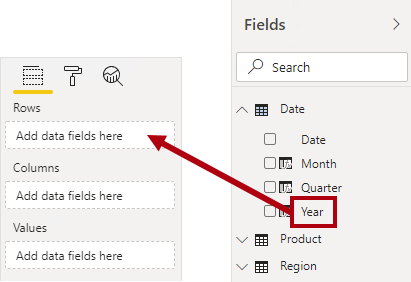


1. To add a matrix visual to the new report page, in the **Visualizations** pane, select the matrix visual type.

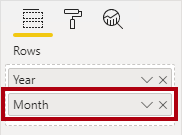
*Tip: You can hover the cursor over each icon to reveal a tooltip describing the visual type.*



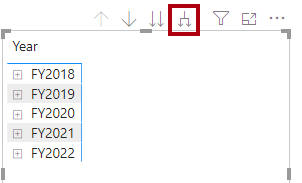
1. In the **Fields** pane, from inside the **Date** table, drag the **Year** field into the **Rows** well/area.



1. Drag the **Month** field into the **Rows** well/area, directly beneath the **Year** field.



1. At the top-right of the matrix visual (or bottom, depending on the location of the visual), click the forked-double arrow icon (which will expand all years down one level).



1. Notice that the years expand to months, and that the months are sorted alphabetically rather than chronologically.



*By default, text values sort alphabetically, numbers sort from smallest to largest, and dates sort from earliest to latest.*

1. To customize the **Month** field sort order, switch to Data view.
2. Add the **MonthKey** column to the **Date** table.

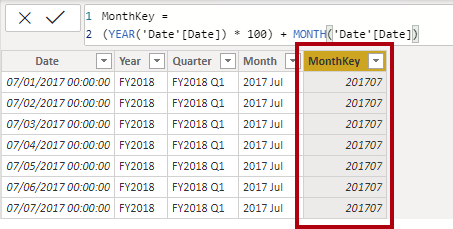
**DAX**

MonthKey =

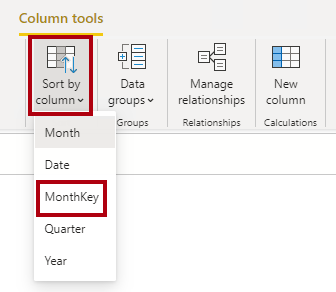
(YEAR('Date'[Date]) \* 100) + MONTH('Date'[Date])

*This formula computes a numeric value for each year/month combination.*

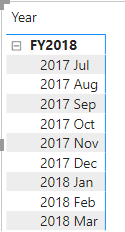
1. In Data view, verify that the new column contains numeric values (e.g. 201707 for July 2017, etc.).



1. Switch back to Report view.
2. In the **Fields** pane, ensure that the **Month** field is selected (when selected, it will have a dark gray background).
3. On the **Column Tools** contextual ribbon, from inside the **Sort** group, click **Sort by Column**, and then select **MonthKey**.



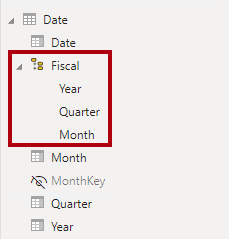
1. In the matrix visual, notice that the months are now chronologically sorted.



**Task 5: Complete the Date table**

In this task you will complete the design of the **Date** table by hiding a column and creating a hierarchy. You will then create relationships to the **Sales** and **Targets** tables.

1. Switch to Model view.
2. In the **Date** table, hide the **MonthKey** column (set **Is Hidden** to **Yes**).
3. In the **Date** table, create a hierarchy named **Fiscal**, with the following three levels:
   * Year
   * Quarter
   * Month

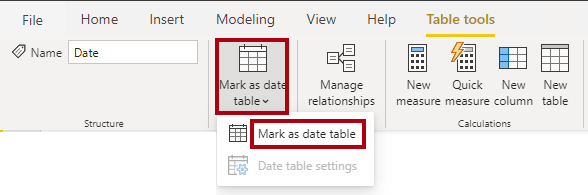


1. Create the following two model relationships:
   * **Date | Date** to **Sales | OrderDate**
   * **Date | Date** to **Targets | TargetMonth**
2. Hide the following two columns:
   * Sales | OrderDate
   * Targets | TargetMonth

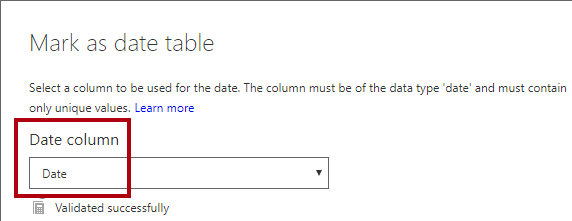
**Task 6: Mark the Date table**

In this task you will mark the **Date** table as a date table.

1. Switch to Report view.
2. In the **Fields** pane, select the **Date** table (not the **Date** field).
3. On the **Table Tools** contextual ribbon, from inside the **Calendars** group, click **Mark as Date Table**, and then select **Mark as Date Table**.



1. In the **Mark as Date Table** window, in the **Date Column** dropdown list, select **Date**.



1. Click **OK**.



1. Save the Power BI Desktop file.

*Power BI Desktop now understands that this table defines date (time). It’s important when relying on time intelligence calculations. You’ll work with time intelligence calculations in the* ***Create DAX Calculations in Power BI Desktop, Part 2*** *lab.*

*Note that this design approach for a date table is suitable when you don’t have a date table in your data source. If you have a data warehouse, it would be appropriate to load date data from its date dimension table rather than “redefining” date logic in your data model.*

**Exercise 2: Create Measures**

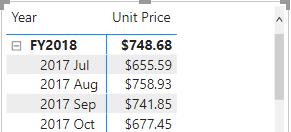
In this exercise you will create and format several measures.

**Task 1: Create simple measures**

In this task you will create simple measures. Simple measures aggregate values in a single column or count rows of a table.

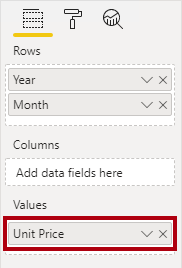
1. In Report view, on **Page 2**, in the **Fields** pane, drag the **Sales | Unit Price** field into the matrix visual.

*The labs use a shorthand notation to reference a field. It will look like this:* ***Sales | Unit Price****. In this example,* ***Sales*** *is the table name and* ***Unit Price*** *is the field name.*

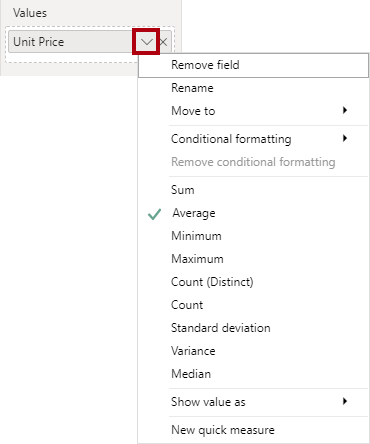


*You may recall that in the* ***Model Data in Power BI Desktop, Part 2*** *lab, you set the* ***Unit Price*** *column to summarize by* ***Average****. The result you see in the matrix visual is the monthly average unit price (sum of unit price values divided by the count of unit prices).*

1. In the visual fields pane (located beneath the **Visualizations** pane), in the **Values** field well/area, notice that **Unit Price** is listed.

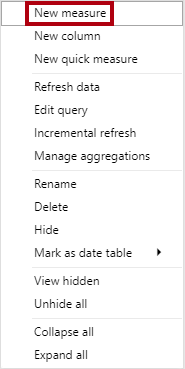


1. Click the down-arrow for **Unit Price**, and then notice the available menu options.



*Visible numeric columns allow report authors at report design time to decide how column values will summarize (or not). It can result in inappropriate reporting. Some data modelers don’t like leaving things to chance, however, and choose to hide these columns and instead expose aggregation logic defined in measures. It’s the approach you will now take in this lab.*

1. To create a measure, in the **Fields** pane, right-click the **Sales** table, and then select **New Measure**.



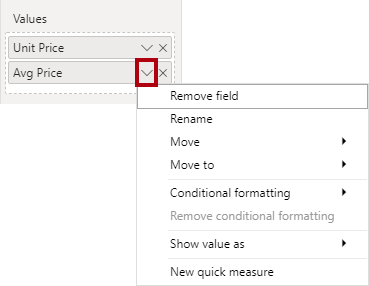
1. In the formula bar, add the following measure definition:

**DAX**

Avg Price =

‎AVERAGE(Sales[Unit Price])

1. Add the **Avg Price** measure to the matrix visual.
2. Notice that it produces the same result as the **Unit Price** column (but with different formatting).
3. In the **Values** well, open the context menu for the **Avg Price** field, and notice that it is not possible to change the aggregation technique.



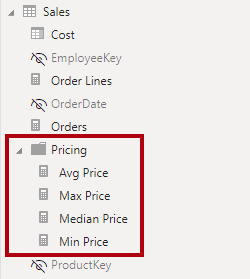
*It’s not possible to modify the aggregation behavior of a measure.*

1. Use the snippets file definitions to create the following five measures for the **Sales** table:
   * Median Price
   * Min Price
   * Max Price
   * Orders
   * Order Lines

*The DISTINCTCOUNT() function used in the* ***Orders*** *measure will count orders only once (ignoring duplicates). The COUNTROWS() function used in the* ***Order Lines*** *measure operates over a table.*

*In this case, the number of orders is calculated by counting the distinct* ***SalesOrderNumber*** *column values, while the number of order lines is simply the number of table rows (each row is a line of an order).*

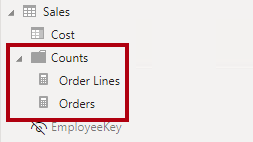
1. Switch to Model view, and then multi-select the four price measures: **Avg Price**, **Max Price**, **Median Price**, and **Min Price**.
2. For the multi-selection of measures, configure the following requirements:
   * Set the format to two decimal places
   * Assign to a display folder named **Pricing**



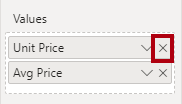
1. Hide the **Unit Price** column.

*The* ***Unit Price*** *column is now not available to report authors. They must use the pricing measures you’ve added to the model. This design approach ensures that report authors won’t inappropriately aggregate prices, for example, by summing them.*

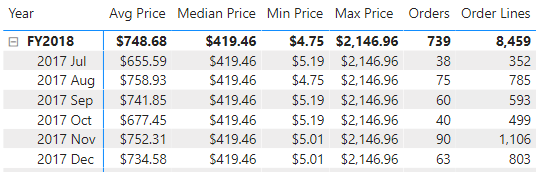
1. Multi-select the **Order Lines** and **Orders** measures, and then configure the following requirements:
   * Set the format use the thousands separator
   * Assign to a display folder named **Counts**



1. In Report view, in the **Values** well/area of the matrix visual, for the **Unit Price** field, click **X** to remove it.



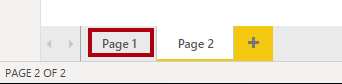
1. Increase the size of the matrix visual to fill the page width and height.
2. Add the following five measures to the matrix visual:
   * Median Price
   * Min Price
   * Max Price
   * Orders
   * Order Lines
3. Verify that the results looks sensible and are correctly formatted.



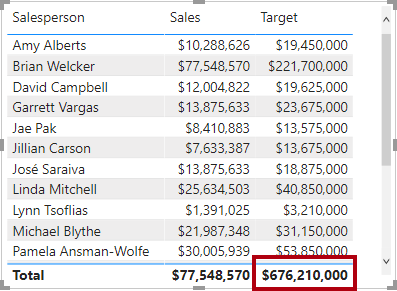
**Task 2: Create additional measures**

In this task you will create additional measures that use more complex formulas.

1. In Report view, select **Page 1**.

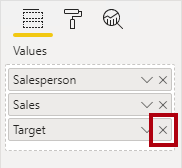


1. Review the table visual, noticing the total for the **Target** column.



*You may recall from a previous lab that there’s a many-to-many relationship between salespeople and regions. This means that summing the target values together doesn’t make sense because salespeople targets are set for each salesperson based on their sales region assignment(s). A target value should only be shown when a single salesperson is filtered. You’ll now implement a measure now to do just that.*

1. Select the table visual, and then in the **Visualizations** pane, remove the **Target** field.



1. Rename the **Targets | Target** column as **Targets | TargetAmount**.

*Tip: There are several ways to rename the column in Report view: In the* ***Fields*** *pane, you can right-click the column, and then select* ***Rename****—or, double-click the column, or press* ***F2****.*

*You’re about to create a measure named* ***Target****. It’s not possible to have a column and measure in the same table with the same name.*

1. Create the following measure on the **Targets** table:

**DAX**

Target =

IF(

HASONEVALUE('Salesperson (Performance)'[Salesperson]),

SUM(Targets[TargetAmount])

)

*The HASONEVALUE() function tests whether a single value in the* ***Salesperson*** *column is filtered. When true, the expression returns the sum of target amounts (for just that salesperson). When false, BLANK is returned.*

1. Format the **Target** measure for zero decimal places.

*Tip: You can use the* ***Measure Tools*** *contextual ribbon.*

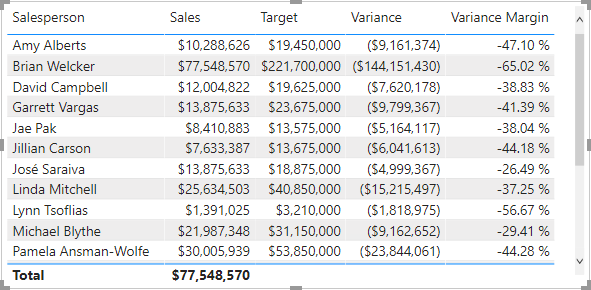
1. Hide the **TargetAmount** column.

*Tip: You can right-click the column in the* ***Fields*** *pane, and then select* ***Hide****.*

1. Add the **Target** measure to the table visual.
2. Notice that the **Target** column total is now BLANK.

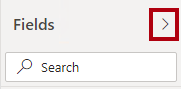


1. Use the snippets file definitions to create the following two measures for the **Targets** table:
   * Variance
   * Variance Margin
2. Format the **Variance** measure for zero decimal places.
3. Format the **Variance Margin** measure as percentage with two decimal places.
4. Add the **Variance** and **Variance Margin** measures to the table visual.
5. Resize the table visual so all columns and rows can be seen.



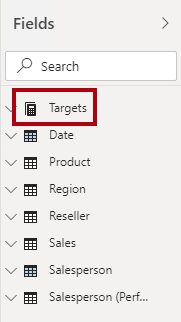
*While it appears all salespeople are not meeting target, remember that the table visual isn’t yet filtered by a specific time period. You’ll produce sales performance reports that filter by a user-selected time period in the* ***Design a Report in Power BI Desktop, Part 1*** *lab.*

1. At the top-right corner of the **Fields** pane, collapse and then expand open the pane.



*Collapsing and re-opening the pane resets the content.*

1. Notice that the **Targets** table now appears at the top of the list.



*Tables that comprise only visible measures are automatically listed at the top of the list.*

**Task 3: Finish up**

In this task you will complete the lab.

1. Save the Power BI Desktop file.
2. If you intend to start the next lab, leave Power BI Desktop open.

*You’ll enhance the data model with more advanced calculations using DAX in the* ***Create DAX Calculations in Power BI Desktop, Part 2*** *lab.*

**Congratulations**

You have successfully completed this exercise. Click **Next** to advance to the next lab.

Click [**here**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#toc) to return to the **Table of Contents**.

All Modules: Analyzing Data with Microsoft Power BI

3 Hr 36 Min Remaining

**Create DAX Calculations in Power BI Desktop, Part 2**

**The estimated time to complete the lab is 45 minutes**

In this lab you will create measures with DAX expressions involving filter context manipulation.

In this lab you learn how to:

* Use the CALCULATE() function to manipulate filter context
* Use Time Intelligence functions

**Lab story**

This lab is one of many in a series of labs that was designed as a complete story from data preparation to publication as reports and dashboards. You can complete the labs in any order. However, if you intend to work through multiple labs, for the first 10 labs, we suggest you do them in the following order:

1. Prepare Data in Power BI Desktop
2. Load Data in Power BI Desktop
3. Model Data in Power BI Desktop, Part 1
4. Model Data in Power BI Desktop, Part 2
5. Create DAX Calculations in Power BI Desktop, Part 1
6. **Create DAX Calculations in Power BI Desktop, Part 2**
7. Design a Report in Power BI Desktop, Part 1
8. Design a Report in Power BI Desktop, Part 2
9. Create a Power BI Dashboard
10. Create a Power BI Paginated Report
11. Perform Data Analysis in Power BI Desktop

**Exercise 1: Work with Filter Context**

In this exercise you will create measures with DAX expressions involving filter context manipulation.

**Task 1: Get started**

In this task you will setup the environment for the lab.

*Important: If you are continuing on from the previous lab (and you completed that lab successfully), do not complete this task; instead, continue from the next task.*

1. To open the Power BI Desktop, on the taskbar, click the Microsoft Power BI Desktop shortcut.



1. To close the getting started window, at the top-left of the window, click **X**.

Picture 11

1. To open the starter Power BI Desktop file, click the **File** ribbon tab to open the backstage view.
2. Select **Open Report**.



1. Click **Browse Reports**.

Picture 9

1. In the **Open** window, navigate to the **D:\DA100\Labs\06-create-dax-calculations-in-power-bi-desktop-advanced\Starter** folder.
2. Select the **Sales Analysis** file.
3. Click **Open**.

Picture 8

1. Close any informational windows that may open.
2. To create a copy of the file, click the **File** ribbon tab to open the backstage view.
3. Select **Save As**.



1. If prompted to apply changes, click **Apply**.



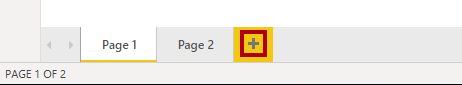
1. In the **Save As** window, navigate to the **D:\DA100\MySolution** folder.
2. Click **Save**.

Picture 2

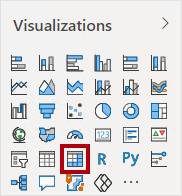
**Task 2: Create a matrix visual**

In this task you will create a matrix visual to support testing your new measures.

1. In Power BI Desktop, in Report view, create a new report page.



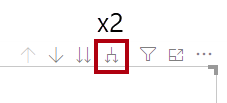
1. On **Page 3**, add a matrix visual.



1. Resize the matrix visual to fill the entire page.
2. To configure the matrix visual fields, from the **Fields** pane, drag the **Region | Regions** hierarchy, and drop it inside the visual.

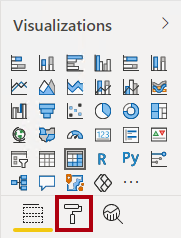
*The labs use a shorthand notation to reference a field or hierarchy. It will look like this:* ***Region | Regions****. In this example,* ***Region*** *is the table name and* ***Regions*** *is the hierarchy name.*

1. Add also the **Sales | Sales** field.
2. To expand the entire hierarchy, at the top-right of the matrix visual, click the forked-double arrow icon twice.

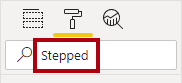


*You may recall that the* ***Regions*** *hierarchy has the levels* ***Group****,* ***Country****, and* ***Region****.*

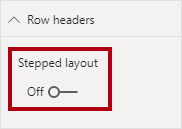
1. To format the visual, beneath the **Visualizations** pane, select the **Format** pane.



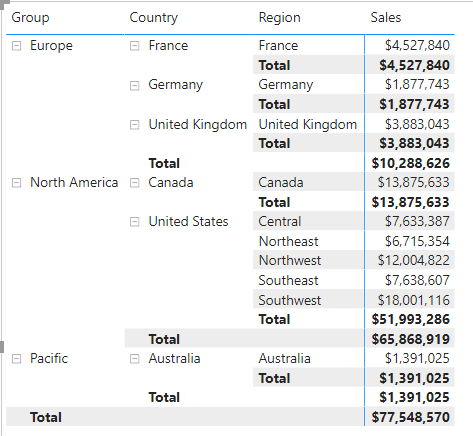
1. In the **Search** box, enter **Stepped**.



1. Set the **Stepped Layout** property to **Off**.



1. Verify that the matrix visual now has four column headers.



*At Adventure Works, the sales regions are organized into groups, countries, and regions. All countries—except the United States—have just one region, which is named after the country. As the United States is such a large sales territory, it’s divided into five sales regions.*

*You’ll create several measures in this exercise, and then test them by adding them to the matrix visual.*

**Task 3: Manipulate filter context**

In this task you will create several measures with DAX expressions that use the CALCULATE() function to manipulate filter context.

1. Add a measure to the **Sales** table, based on the following expression:

*For your convenience, all DAX definitions in this lab can be copied from the* ***D:\DA100\Labs\06-create-dax-calculations-in-power-bi-desktop-advanced\Assets\Snippets.txt*** *file.*

**DAX**

Sales All Region =

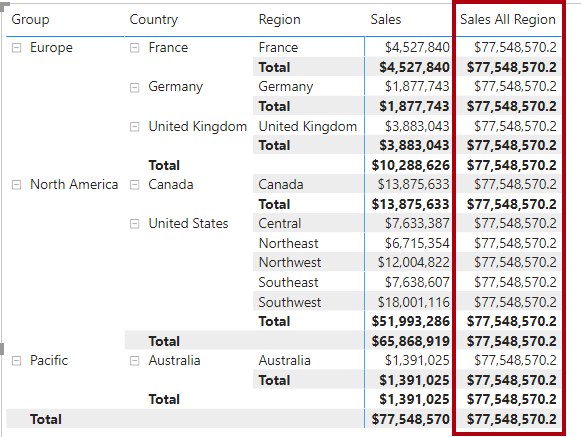
CALCULATE(SUM(Sales[Sales]), REMOVEFILTERS(Region))

*The CALCULATE() function is a powerful function used to manipulate the filter context. The first argument takes an expression or a measure (a measure is just a named expression). Subsequent arguments allow modifying the filter context.*

*The REMOVEFILTERS() function removes active filters. It can take either no arguments, or a table, a column, or multiple columns as its argument.*

*In this formula, the measure evaluates the sum of the* ***Sales*** *column in a modified filter context, which removes any filters applied to the columns of the* ***Region*** *table.*

1. Add the **Sales All Region** measure to the matrix visual.



1. Notice that the **Sales All Region** measure computes the total of all region sales for each region, country (subtotal) and group (subtotal).

*The new measure is yet to deliver a useful result. When the sales for a group, country, or region is divided by this value it will produce a useful ratio known as “percent of grand total”.*

1. In the **Fields** pane, ensure that the **Sales All Region** measure is selected (when selected, it will have a dark gray background), and then in the formula bar, replace the measure name and formula with the following formula:

*Tip: To replace the existing formula, first copy the snippet. Then, click inside the formula bar and press* ***Ctrl+A*** *to select all text. Then, press* ***Ctrl+V*** *to paste the snippet to overwrite the selected text. Then press* ***Enter****.*

**DAX**

Sales % All Region =

‎DIVIDE(

‎ SUM(Sales[Sales]),

‎ CALCULATE(

‎ SUM(Sales[Sales]),

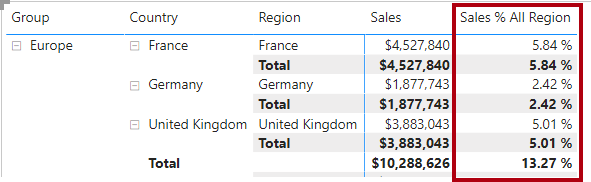
‎ REMOVEFILTERS(Region)

‎ )

‎)

*The measure has been renamed to accurately reflect the updated formula. The DIVIDE() function divides the* ***Sales*** *measure (not modified by filter context) by the* ***Sales*** *measure in a modified context, which removes any filters applied to the* ***Region*** *table.*

1. In the matrix visual, notice that the measure has been renamed and that a different values now appear for each group, country, and region.
2. Format the **Sales % All Region** measure as a percentage with two decimal places.
3. In the matrix visual, review the **Sales % All Region** measure values.



1. Add another measure to the **Sales** table, based on the following expression, and format as a percentage:

**DAX**

Sales % Country =

‎DIVIDE(

‎ SUM(Sales[Sales]),

‎ CALCULATE(

‎ SUM(Sales[Sales]),

‎ REMOVEFILTERS(Region[Region])

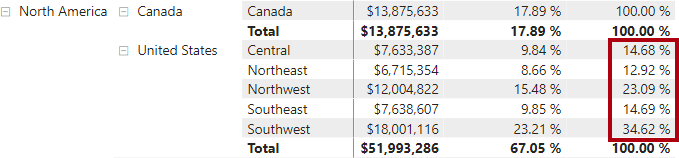
‎ )

‎)

1. Notice that the **Sales % Country** measure formula differs slightly from the **Sales % All Region** measure formula.

*The difference is that the denominator modifies the filter context by removing filters on the* ***Region*** *column of the* ***Region*** *table, not all columns of the* ***Region*** *table. It means that any filters applied to the group or country columns are preserved. It’ll achieve a result that represents the sales as a percentage of country.*

1. Add the **Sales % Country** measure to the matrix visual.
2. Notice that only the United States’ regions produce a value which is not 100%.



*You may recall that only the United States has multiple regions. All other countries comprise a single region, which explains why they are all 100%.*

1. To improve the readability of this measure in visual, overwrite the **Sales % Country** measure with this improved formula.

**DAX**

Sales % Country =

‎IF(

‎ ISINSCOPE(Region[Region]),

‎ DIVIDE(

‎ SUM(Sales[Sales]),

‎ CALCULATE(

‎ SUM(Sales[Sales]),

‎ REMOVEFILTERS(Region[Region])

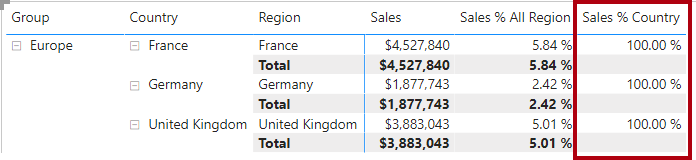
‎ )

‎ )

‎)

*Embedded within the IF() function, the ISINSCOPE() function is used to test whether the region column is the level in a hierarchy of levels. When true, the DIVIDE() function is evaluated. The absence of a false part means that blank is returned when the region column is not in scope.*

1. Notice that the **Sales % Country** measure now only returns a value when a region is in scope.



1. Add another measure to the **Sales** table, based on the following expression, and format as a percentage:

**DAX**

Sales % Group =

‎DIVIDE(

‎ SUM(Sales[Sales]),

‎ CALCULATE(

‎ SUM(Sales[Sales]),

‎ REMOVEFILTERS(

‎ Region[Region],

‎ Region[Country]

‎ )

‎ )

‎)

*To achieve sales as a percentage of group, two filters can be applied to effectively remove the filters on two columns.*

1. Add the **Sales % Group** measure to the matrix visual.
2. To improve the readability of this measure in visual, overwrite the **Sales % Group** measure with this improved formula.

**DAX**

Sales % Group =

‎IF(

‎ ISINSCOPE(Region[Region])

‎ || ISINSCOPE(Region[Country]),

‎ DIVIDE(

‎ SUM(Sales[Sales]),

‎ CALCULATE(

‎ SUM(Sales[Sales]),

‎ REMOVEFILTERS(

‎ Region[Region],

‎ Region[Country]

‎ )

‎ )

‎ )

‎)

1. Notice that the **Sales % Group** measure now only returns a value when a region or country is in scope.
2. In Model view, place the three new measures into a display folder named **Ratios**.



1. Save the Power BI Desktop file.

*The measures added to the* ***Sales*** *table have modified filter context to achieve hierarchical navigation. Notice that the pattern to achieve the calculation of a subtotal requires removing some columns from the filter context, and to arrive at a grand total, all columns must be removed.*

**Exercise 2: Work with Time Intelligence**

In this exercise you will create a sales year-to-date (YTD) measure and sales year-over-year (YoY) growth measure.

**Task 1: Create a YTD measure**

In this task you will create a sales YTD measure.

1. In Report view, on **Page 2**, notice the matrix visual that displays various measures with years and months grouped on the rows.
2. Add a measure to the **Sales** table, based on the following expression, and formatted to zero decimal places:

**DAX**

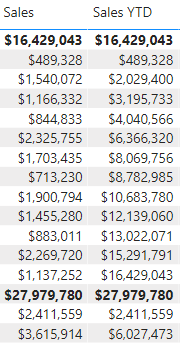
Sales YTD =

‎TOTALYTD(SUM(Sales[Sales]), 'Date'[Date], "6-30")

*The TOTALYTD() function evaluates an expression—in this case the sum of the* ***Sales*** *column—over a given date column. The date column must belong to a date table marked as a date table, as was done in the* ***Create DAX Calculations in Power BI Desktop, Part 1*** *lab.*

*The function can also take a third optional argument representing the last date of a year. The absence of this date means that December 31 is the last date of the year. For Adventure Works, June in the last month of their year, and so “6-30” is used.*

1. Add the **Sales** field and the **Sales YTD** measure to the matrix visual.
2. Notice the accumulation of sales values within the year.



*The TOTALYTD() function performs filter manipulation, specifically time filter manipulation. For example, to compute YTD sales for September 2017 (the third month of the fiscal year), all filters on the* ***Date*** *table are removed and replaced with a new filter of dates commencing at the beginning of the year (July 1, 2017) and extending through to the last date of the in-context date period (September 30, 2017).*

*Note that many Time Intelligence functions are available in DAX to support common time filter manipulations.*

**Task 2: Create a YoY growth measure**

In this task you will create a sales YoY growth measure.

1. Add an additional measure to the **Sales** table, based on the following expression:

**DAX**

Sales YoY Growth =

‎VAR SalesPriorYear =

‎ CALCULATE(

‎ SUM(Sales[Sales]),

‎ PARALLELPERIOD(

‎ 'Date'[Date],

‎ -12,

‎ MONTH

‎ )

‎ )

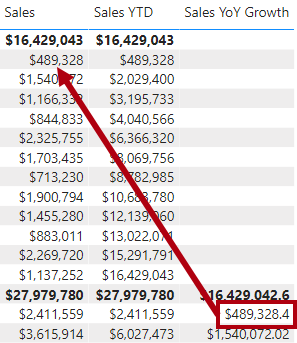
‎RETURN

‎ SalesPriorYear

*The* ***Sales YoY Growth*** *measure formula declares a variable. Variables can be useful for simplifying the formula logic, and more efficient when an expression needs to be evaluated multiple times within the formula (which will be the case for the YoY growth logic). Variables are declared by a unique name, and the measure expression must then be output after the* ***RETURN*** *keyword.*

*The* ***SalesPriorYear*** *variable is assigned an expression which calculates the sum of the* ***Sales*** *column in a modified context that uses the PARALLELPERIOD() function to shift 12 months back from each date in filter context.*

1. Add the **Sales YoY Growth** measure to the matrix visual.
2. Notice that the new measure returns BLANK for the first 12 months (because there were no sales recorded before fiscal year 2017).
3. Notice that the **Sales YoY Growth** measure value for **2018 Jul** is the **Sales** value for **2018 Jul**.



*Now that the “difficult part” of the formula has been tested, you can overwrite the measure with the final formula which computes the growth result.*

1. To complete the measure, overwrite the **Sales YoY Growth** measure with this formula, formatting it as a percentage with two decimal places:

**DAX**

Sales YoY Growth =

‎VAR SalesPriorYear =

‎ CALCULATE(

‎ SUM(Sales[Sales]),

‎ PARALLELPERIOD(

‎ 'Date'[Date],

‎ -12,

‎ MONTH

‎ )

‎ )

‎RETURN

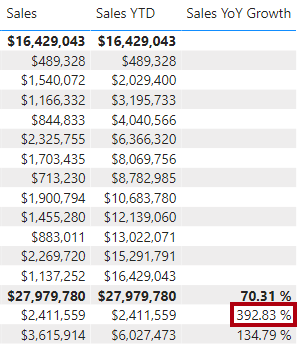
‎ DIVIDE(

‎ (SUM(Sales[Sales]) - SalesPriorYear),

‎ SalesPriorYear

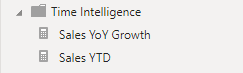
‎ )

1. In the formula, in the **RETURN** clause, notice that the variable is referenced twice.
2. Verify that the YoY growth for **2018 Jul** is **392.83%**.



*It means that July 2018 sales ($2,411,559) represents a nearly 400% (almost 4x) improvement over the sales achieved at the same time of the prior year ($489,328).*

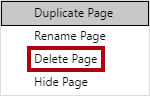
1. In Model view, place the two new measures into a display folder named **Time Intelligence**.



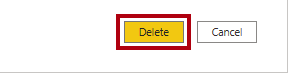
**Task 3: Finish up**

In this task you will complete the lab.

1. To clean up the solution ready for report development, at the bottom-left, right-click the **Page 2** tab, and then select **Delete** page.



1. When prompted to delete the page, click **Delete**.



1. Delete **Page 3** also.
2. On the remaining page, to clear the page, select the table visual, and the press the **Delete** key.
3. Save the Power BI Desktop file.
4. If you intend to start the next lab, leave Power BI Desktop open.

*You’ll create a report based on the data model in the* ***Design a Report in Power BI Desktop, Part 1*** *lab.*

**Congratulations!**

Click **Next** to proceed to the **Review Questions**

Click [**here**](https://labclient.labondemand.com/Instructions/17692e00-0eeb-41ae-a7d8-954484e1f8f0?rc=10#toc) to return to the **Table of Contents**.