

TECHNICAL INCEPTION AND DATA ANALYZING ENGAGEMENT TRAINING WITH POWER BI

*Turn data into opportunity. Drive better
business decisions across your organization
with Power BI.*

Table of Contents

Power BI - Introduction	1
Power BI - Architecture	1
Power BI - Supported Data Sources	2
All	3
File	3
Database	4
Import vs DirectQuery	5
Advantages of Using DirectQuery	6
Limitations of Using DirectQuery	6
Azure.....	6
Online Services	7
Other	8
What is Power BI Desktop?	9
Views in Power BI Desktop.....	10
Connect to data	11
Transform and clean data, create a model.....	12
Create visuals	13
Create reports	14
Share reports	15
Quick start: Connect to data in Power BI Desktop	16
Prerequisites.....	16
Launch Power BI Desktop	17
Connect to data	18
View data in the Fields pane	21
Tutorial: Shape and combine data in Power BI Desktop	22
Shape data	23
Adjust data	27
Combine data	34
Combine queries	38
Tutorial: Analyze web page data using Power BI Desktop	42
Connect to a web data source.....	42
Shape data in Power Query Editor	46
Import the query into Report View	51
Create a visualization.....	52
Customize the visualization	55

Format the map	55
Change the visualization type	58
Tutorial: Combine sales data from Excel and an OData feed	59
Import Excel product data.....	60
Clean up the products columns	61
Import the OData feed's order data	62
Expand the order data	64
Create a custom calculated column.....	66
Set the new field's data type.....	67
Clean up the orders columns	67
Review the query steps	68
Import the transformed queries	69
Manage the relationship between the datasets	70
Create visualizations using your data.....	73
Interact with your report visuals to analyze further	76
Complete the sales analysis report	77
Tutorial: Create your own measures in Power BI Desktop.....	77
Prerequisites.....	78
Automatic measures	78
Create and use your own measures	80
Quick measures	80
Create a measure	81
Use your measure in the report	85
Use your measure with a slicer	86
Use your measure in another measure	88
Tutorial: Create calculated columns in Power BI Desktop.....	90
Prerequisites.....	91
Create a calculated column with values from related tables	91
Use your new column in a report	95
Create a calculated column that uses an IF function.....	97
What you've learned.....	99
Publish from Power BI Desktop	99
To publish a Power BI Desktop dataset and reports	100
Re-publish or replace a dataset published from Power BI Desktop.....	101

Power BI - Introduction

Power BI is a Data Visualization and Business Intelligence tool that converts data from different data sources to interactive dashboards and BI reports. Power BI suite provides multiple software, connector, and services - Power BI desktop, Power BI service based on SaaS, and mobile Power BI apps available for different platforms. These set of services are used by business users to consume data and build BI reports.

Power BI desktop app is used to create reports, while Power BI Services (Software as a Service - SaaS) is used to publish the reports, and Power BI mobile app is used to view the reports and dashboards.

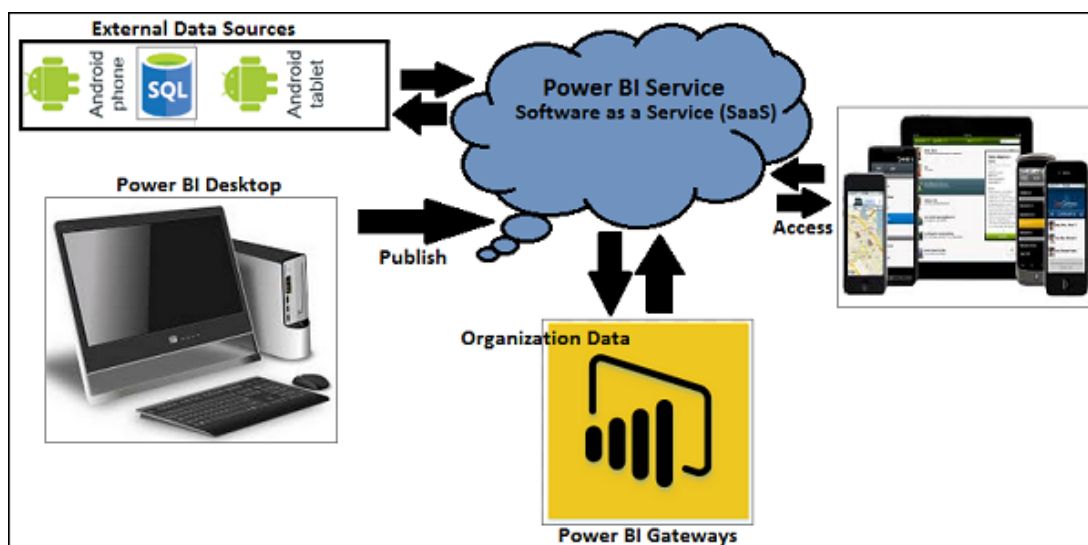
Power BI Desktop is available in both 32-bit and 64-bit versions. To download the latest version, you can use the following link –

<https://powerbi.microsoft.com/en-us/downloads/>

Power BI - Architecture

Power BI includes the following components –

- **Power BI Desktop** – This is used to create reports and data visualizations on the dataset.
- **Power BI Gateway** – You can use Power BI on-premises gateway to keep your data fresh by connecting to your on-premises data sources without the need to move the data. It allows you to query large datasets and benefit from the existing investments.
- **Power BI Mobile Apps** – Using Power BI mobile apps, you can stay connected to their data from anywhere. Power BI apps are available for Windows, iOS, and Android platform.
- **Power BI Service** – This is a cloud service and is used to publish Power BI reports and data visualizations.



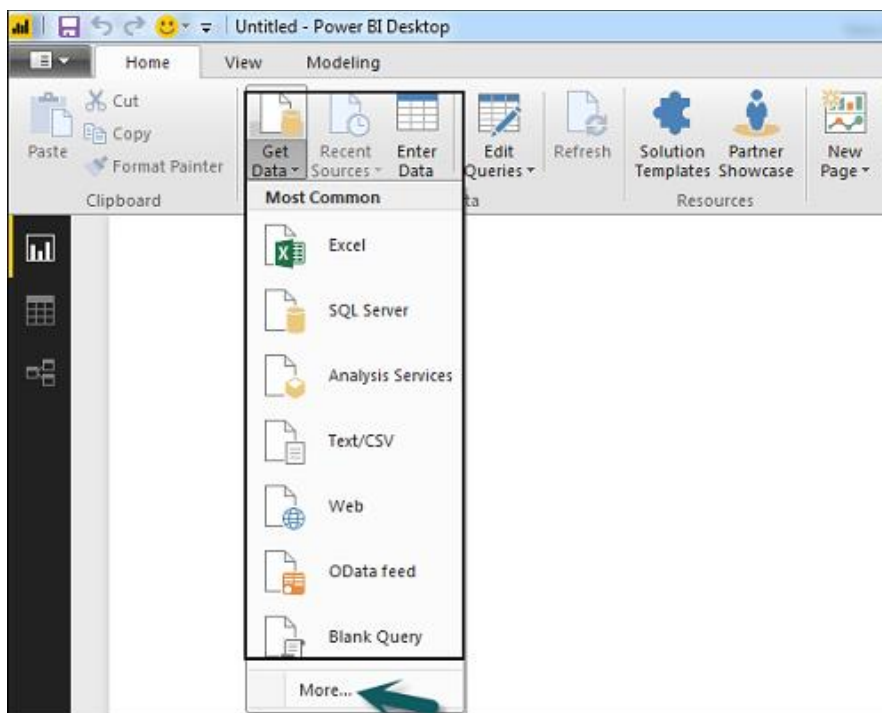
Power BI - Supported Data Sources

Power BI supports large range of data sources. You can click Get data and it shows you all the available data connections. It allows you to connect to different flat files, SQL database, and Azure cloud or even web platforms such as Facebook, Google Analytics, and Salesforce objects. It also includes ODBC connection to connect to other ODBC data sources, which are not listed.

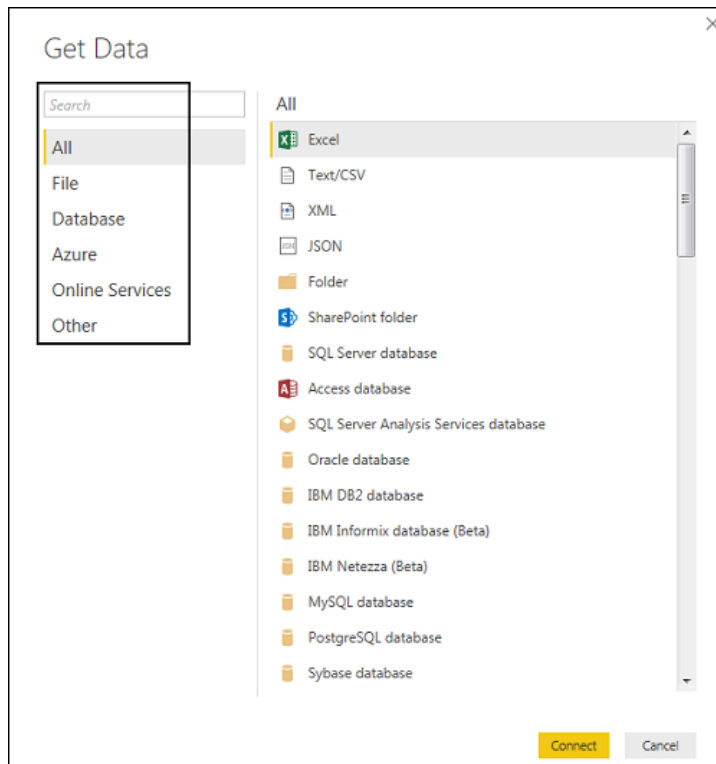
Following are the available data sources in Power BI –

- Flat Files
- SQL Database
- OData Feed
- Blank Query
- Azure Cloud platform
- Online Services
- Blank Query
- Other data sources such as Hadoop, Exchange, or Active Directory

To get data in Power BI desktop, you need to click the Get data option in the main screen. It shows you the most common data sources first. Then, click the More option to see a full list of available data sources.



When you click “More..” tab as shown in the above screenshot, you can see a new navigation window, where on the left side it shows a category of all available data sources. You also have an option to perform a search at the top.



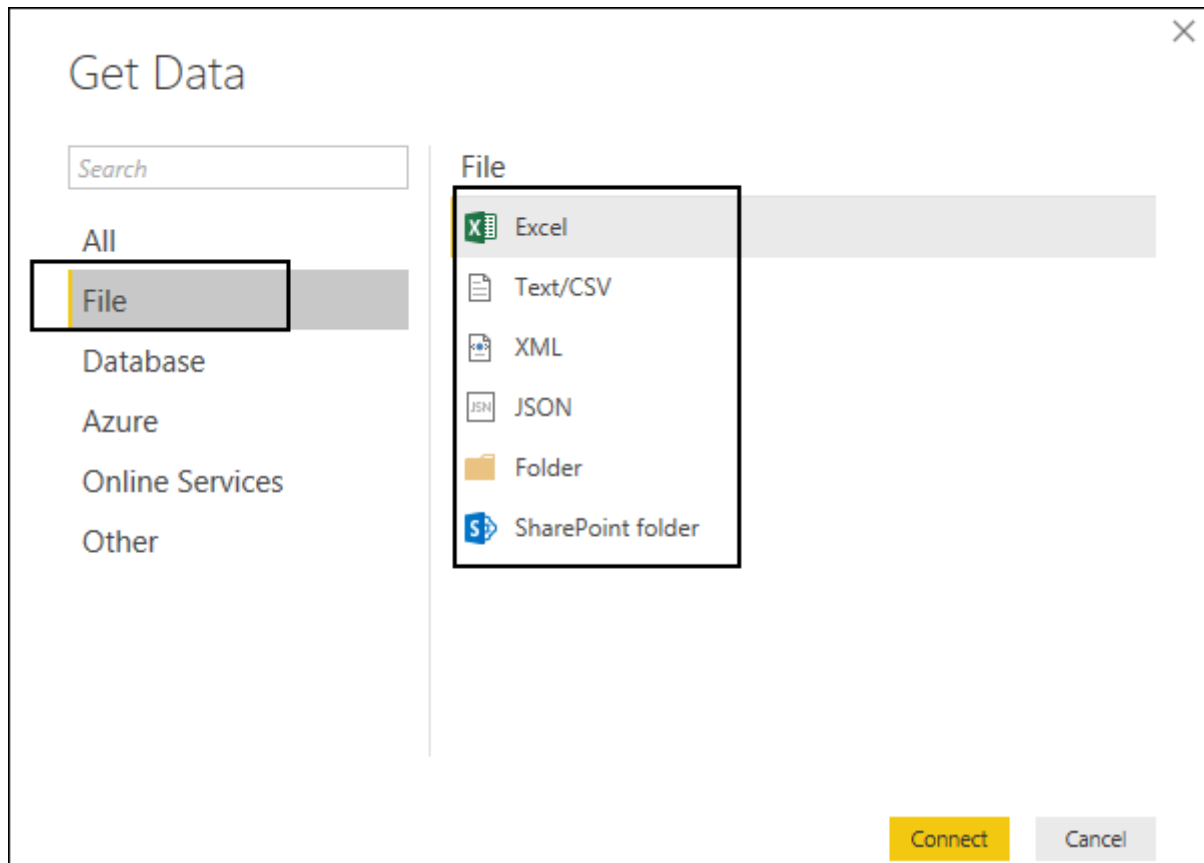
Following are the various **data sources** listed –

All

Under this category, you can see all the available data sources under Power BI desktop.

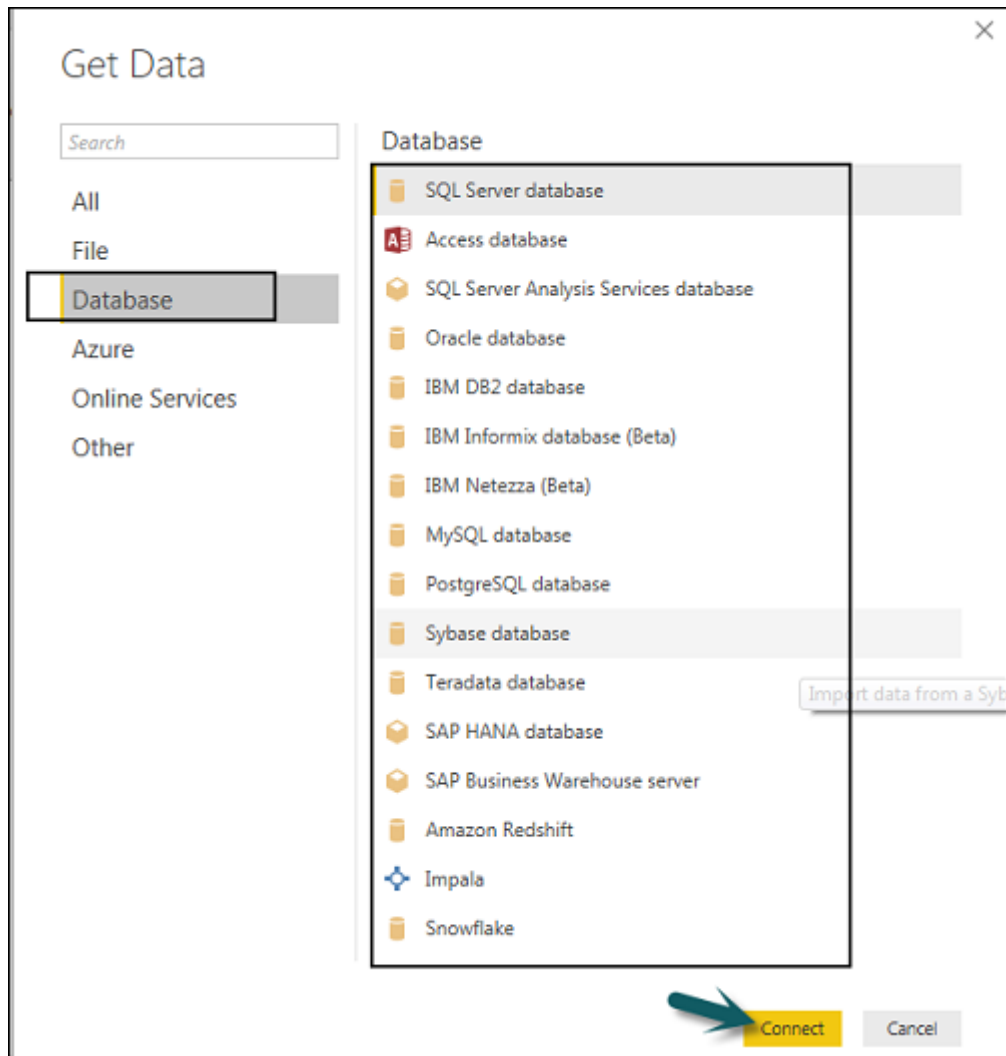
File

When you click File, it shows you all flat file types supported in Power BI desktop. To connect to any file type, select the file type from the list and click Connect. You have to provide the location of the file.



Database

When you click the Database option, it shows a list of all the database connections that you can connect to.



To connect to any database, select a Database type from the list as shown in the above screenshot. Click Connect.

You have to pass Server name/ User name and password to connect. You can also connect via a direct SQL query using Advance options. You can also select Connectivity mode- Import or DirectQuery.

Note – You can't combine import and DirectQuery mode in a single report.

Import vs DirectQuery

DirectQuery option limits the option of data manipulation and the data stays in SQL database. DirectQuery is live and there is no need to schedule refresh as in the Import method.

Import method allows to perform data transformation and manipulation. When you publish the data to PBI service, limit is 1GB. It consumes and pushes data into Power BI Azure backend and data can be refreshed up to 8 times a day and a schedule can be set up for data refresh.

SQL Server database

Server

Database (optional)

Data Connectivity mode ☒ Import ☐ DirectQuery

Advanced options

Command timeout in minutes (optional)

SQL statement (optional, requires database)

☒ Include relationship columns

☐ Navigate using full hierarchy

☐ Enable SQL Server Failover support

OK Cancel

Advantages of Using DirectQuery

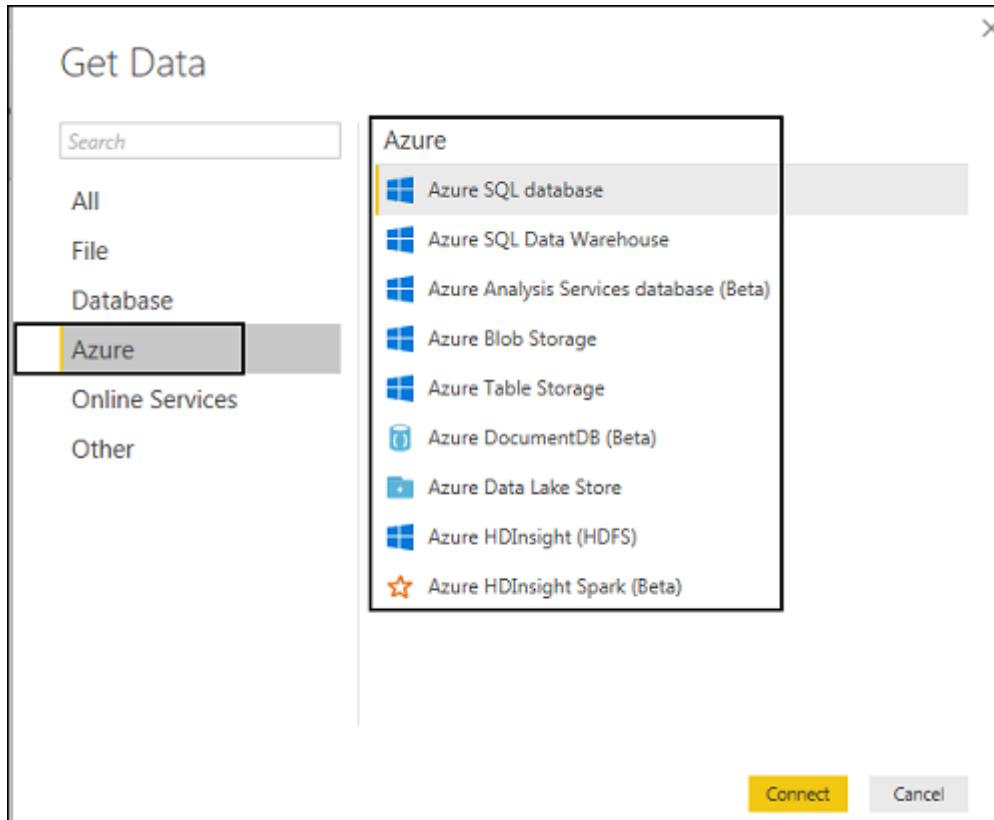
- Using DirectQuery, you can build data visualizations on large datasets, which is not feasible to import in Power BI desktop.
- DirectQuery doesn't apply any 1GB data set limit.
- With the use of DirectQuery, the report always shows current data.

Limitations of Using DirectQuery

- There is a limitation of 1 million row for returning data while using DirectQuery. You can perform aggregation of more number of rows, however, the result rows should be less than 1 million to return the dataset.
- In DirectQuery, all tables should come from a single database.
- When a complex query is used in the Query editor, it throws an error. To run a query, you need to remove the error from the query.
- In DirectQuery, you can use Relationship filtering only in one direction.
- It doesn't support special treatment for time-related data in tables.

Azure

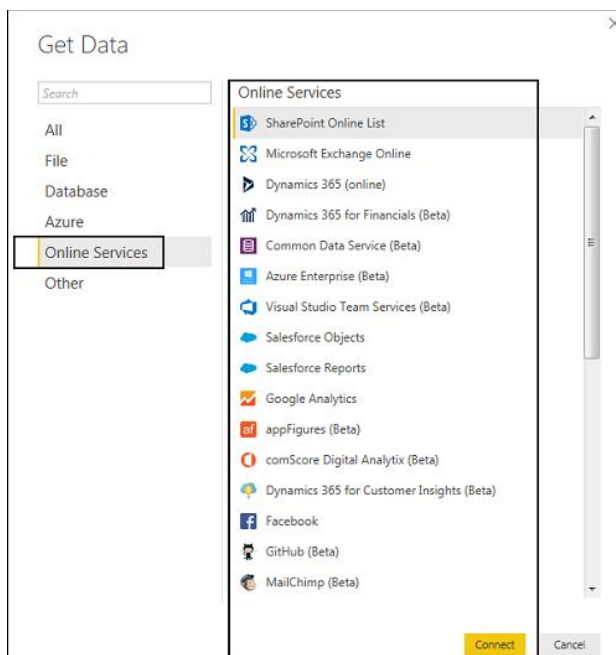
Using the Azure option, you can connect to the database in Azure cloud. Following screenshot shows the various options available under Azure category.

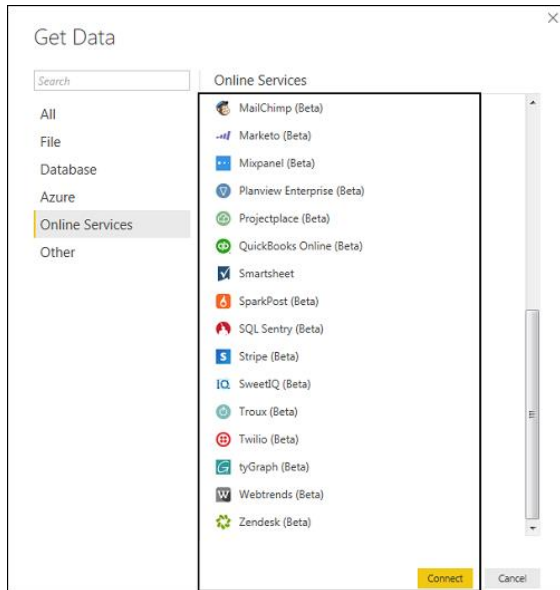


Online Services

Power BI also allows you to connect to different online services such as Exchange, Salesforce, Google Analytics, and Facebook.

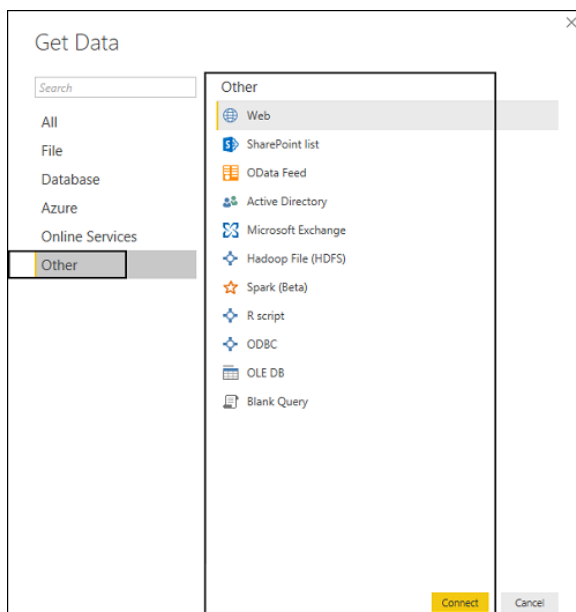
Following screenshots shown the various options available under Online Services.





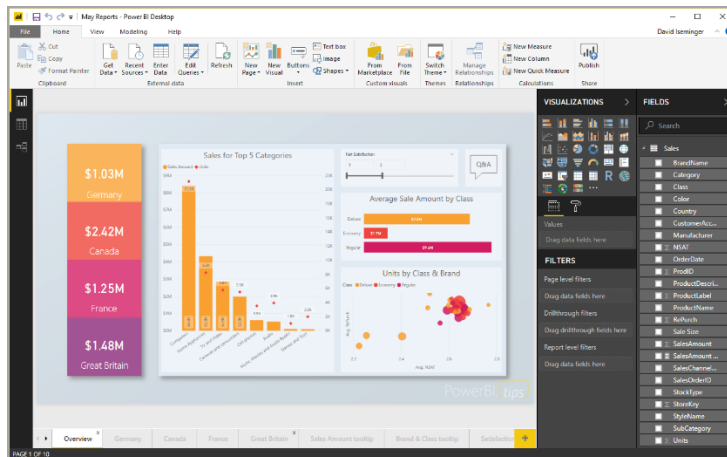
Other

Following screenshot shows the various options available under other category.



What is Power BI Desktop?

Power BI Desktop is a free application you can install on your local computer that lets you connect to, transform, and visualize your data. With **Power BI Desktop**, you can connect to multiple different sources of data, and combine them (often called modeling) into a data model that lets you build visuals, and collections of visuals you can share as reports, with other people inside your organization. Most users who work on Business Intelligence projects use **Power BI Desktop** to create reports, and then use the **Power BI service** to share their reports with others.



The most common uses for **Power BI Desktop** are the following:

- Connect to data
- Transform and clean that data, to create a data model
- Create visuals, such as charts or graphs, that provide visual representations of the data
- Create reports that are collections of visuals, on one or more report pages
- Share reports with others using the **Power BI service**

People most often responsible for such tasks are often considered *data analysts* (sometimes just referred to as *analysts*) or Business Intelligence professionals (often referred to as *report creators*). However, many people who don't consider themselves an analyst or a report creator use **Power BI Desktop** to create compelling reports, or to pull data from various sources and build data models, which they can share with their co-workers and organizations.

Views in Power BI Desktop

There are three views in Power BI Desktop, shown along the left side of the canvas. The views, shown in the order they appear, are the following:

- **Report View** - this is where you create reports and visuals, and where most of your creation time is spent.
- **Data View** - here you can see the tables, measures, and other data used in the data model associated with your report, and transform the data for best use in the report's model.
- **Model View** - in this view you see and manage the relationships among tables in your data model.

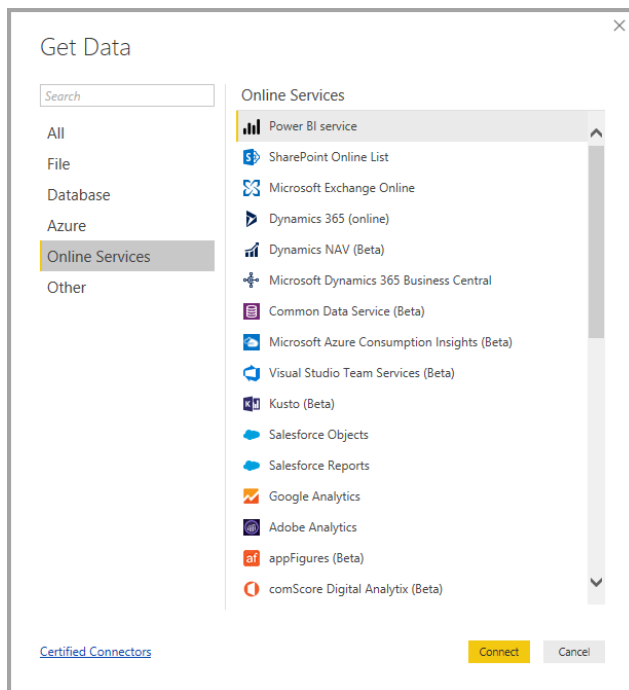
The following image shows the three Views, as displayed along the left side of the canvas:



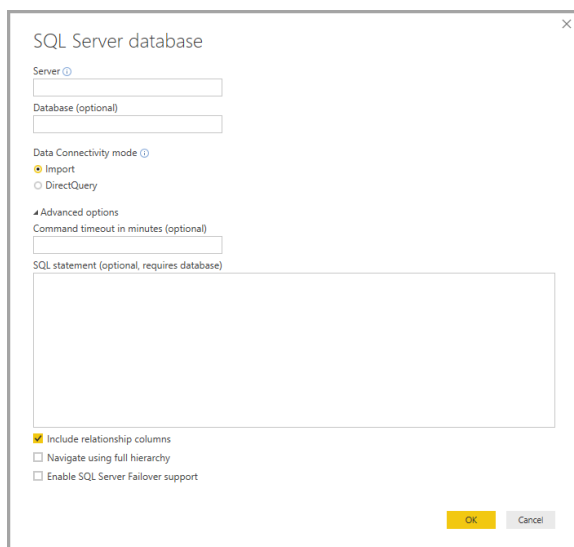
With **Power BI Desktop** you can create complex and visually rich reports, using data from multiple sources, all in one report that you can share with others in your organization.

Connect to data

To get started with **Power BI Desktop**, the first step is to connect to data. There are many different data sources you can connect to from **Power BI Desktop**. To connect to data, simply select the **Home** ribbon, then select **Get Data > More**. The following image shows the **Get Data** window that appears, showing the many categories to which Power BI Desktop can connect.



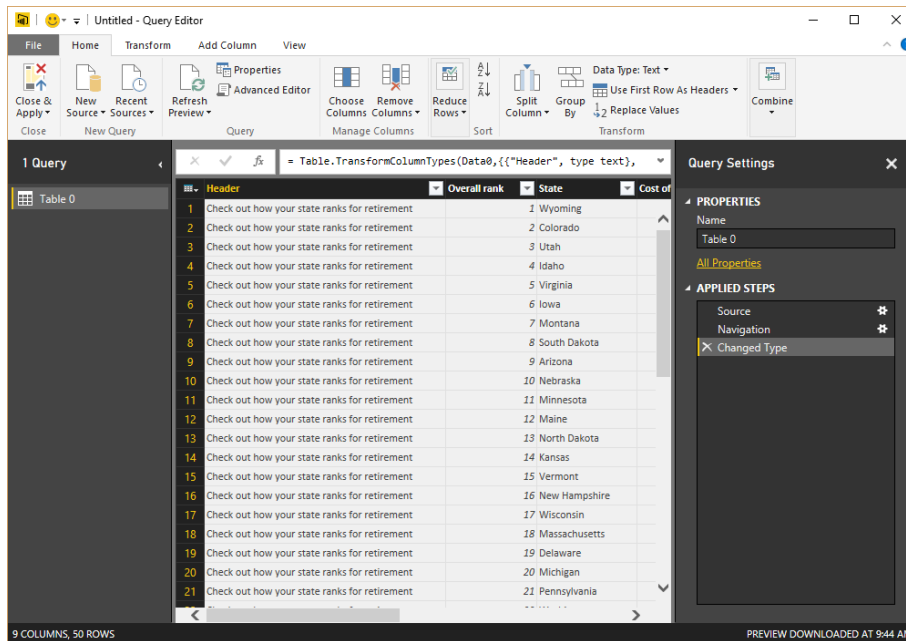
When you select a data type, you're prompted for information, such as the URL and credentials, necessary for Power BI Desktop to connect to the data source on your behalf.



Once you connect to one or more data sources, you may want to transform the data so it's useful for you.

Transform and clean data, create a model

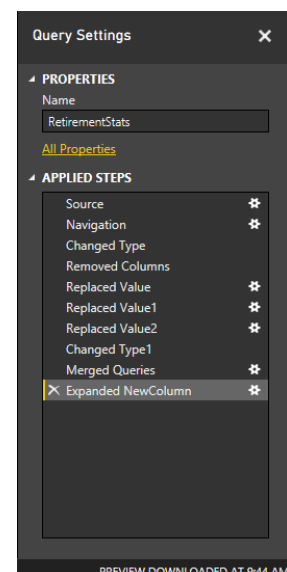
In Power BI Desktop, you can clean and transform data using the built-in **Query Editor**. With Query Editor you can make changes to your data, such as changing a data type, removing columns, or combining data from multiple sources. It's a little bit like sculpting - you can start with a large block of clay (or data), then shave pieces off or add others as needed, until the shape of the data is how you want it.



Each step you take in transforming data (such as rename a table, transform a data type, or delete columns) is recorded by **Query Editor**, and each time this query connects to the data source those steps are carried out so that the data is always shaped the way you specified.

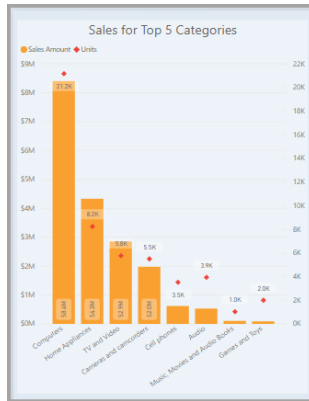
The following image shows the **Query Settings** pane for a query that has been shaped, and turned into a model.

Once your data is how you want it, you can create visuals.

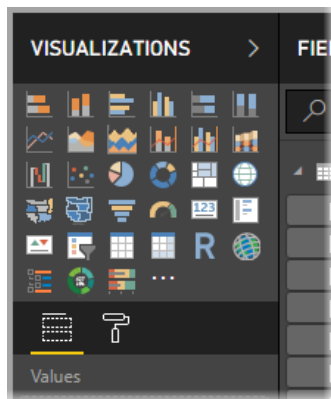


Create visuals

Once you have a data model, you can drag *fields* onto the report canvas to create *visuals*. A *visual* is a graphic representation of the data in your model. The following visual shows a simple column chart.

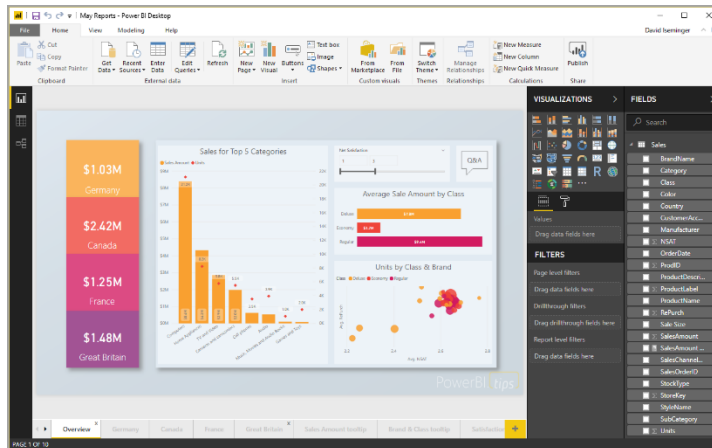


There are many different types of visuals to choose from in Power BI Desktop. To create or change a visual, just select the visual icon from the **Visualizations** pane. If you have a visual selected on the report canvas, the selected visual changes to the type you selected. If no visual is selected, a new visual is created based on your selection.



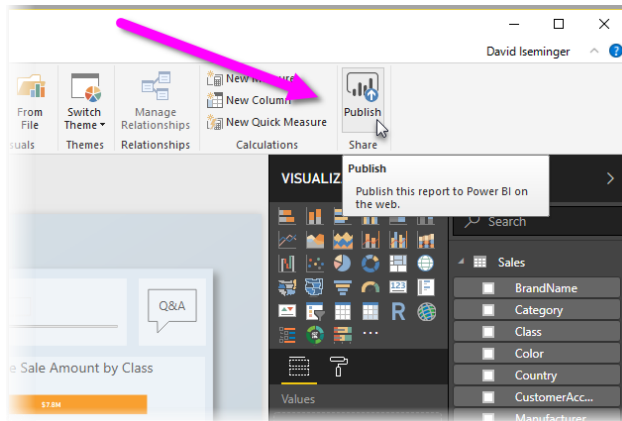
Create reports

More often, you'll want to create a collection of visuals that show various aspects of the data you have used to create your model in Power BI Desktop. A collection of visuals, in one Power BI Desktop file, is called a *report*. A report can have one or more pages, just like an Excel file can have one or more worksheets. In the following image you see the first page of a Power BI Desktop report, named Overview (you can see the tab near the bottom of the image). In this report, there are ten pages.



Share reports

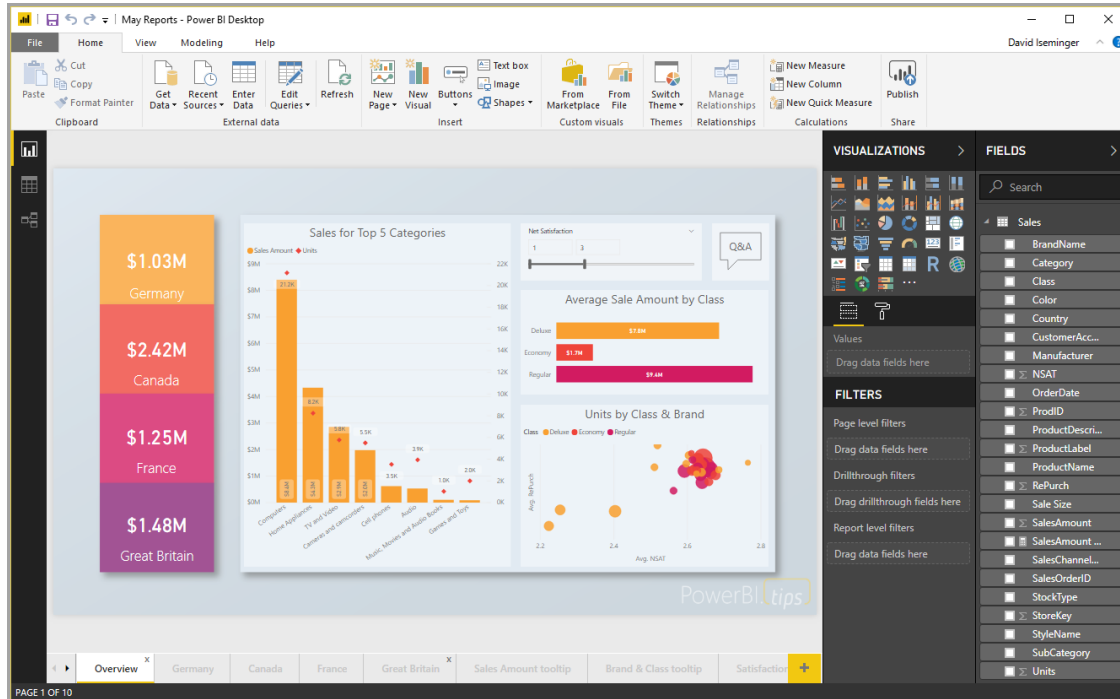
Once a report is ready to share with others, you can **Publish** the report to the **Power BI service**, and make it available to anyone in your organization who has a Power BI license. To publish a Power BI Desktop report, you select the **Publish** button from the **Home** ribbon in Power BI Desktop.



Once you select **Publish**, Power BI Desktop connects you to the **Power BI service** using your Power BI account, and then prompts you to select where in the Power BI service you would like to share the report, such as your workspace, a team workspace, or some other location in the Power BI service. You must have a Power BI license to share reports to the Power BI service.

Quick start: Connect to data in Power BI Desktop

In this quick start, you connect to data using **Power BI Desktop**, which is the first step in building data models and creating reports.



If you're not signed up for Power BI, [sign up for a free trial](#) before you begin.

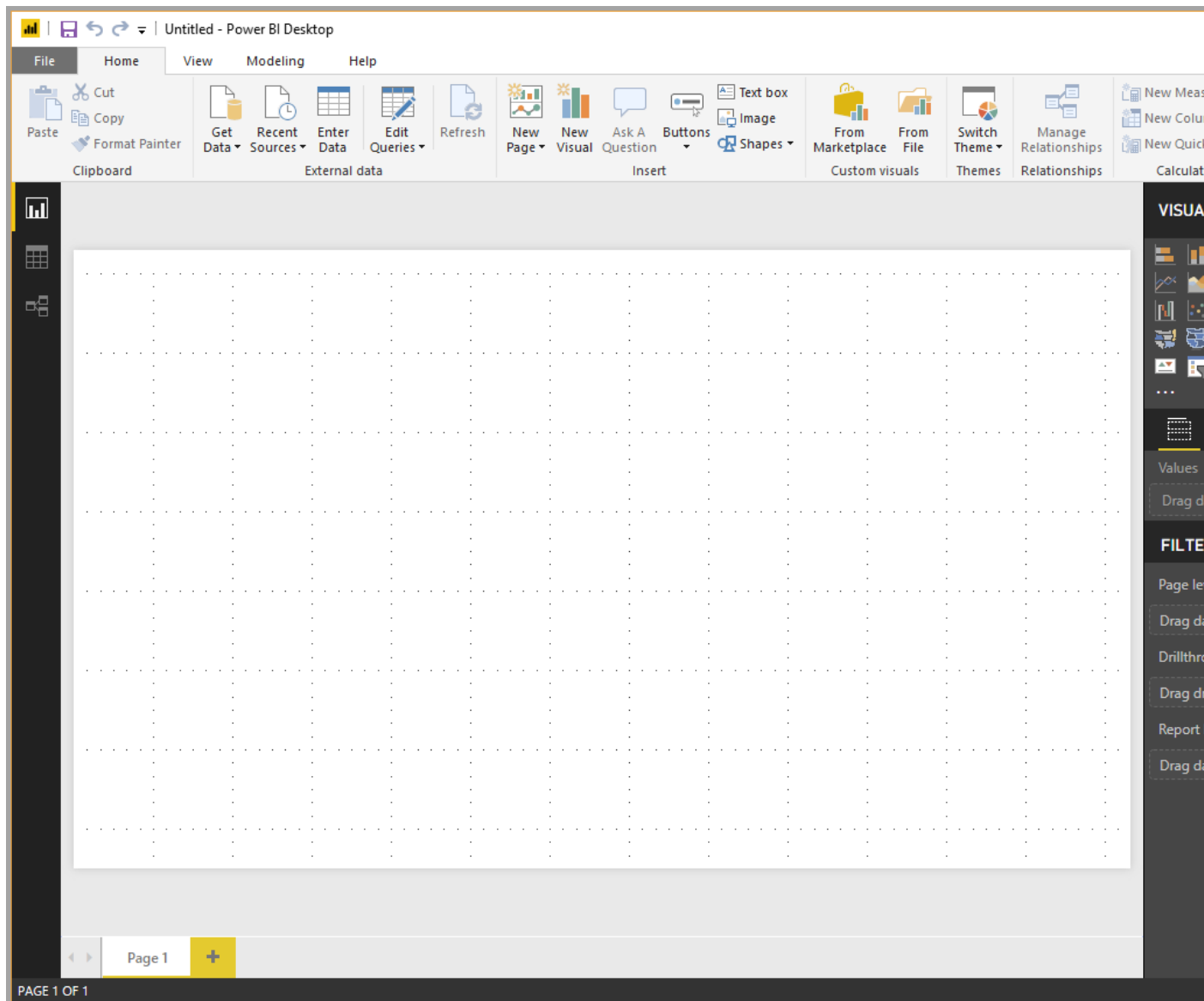
Prerequisites

To complete the steps in this article, you need the following:

- Download and install **Power BI Desktop**, which is a free application that runs on your local computer. You can [download Power BI Desktop](#) directly, or you can get it from [the Microsoft Store](#).
- [Download this sample Excel workbook](#), and create a folder called `C:\PBID-qs` where you can store the Excel file. Subsequent steps in this quick start assume that is the file location for the downloaded Excel workbook.

Launch Power BI Desktop

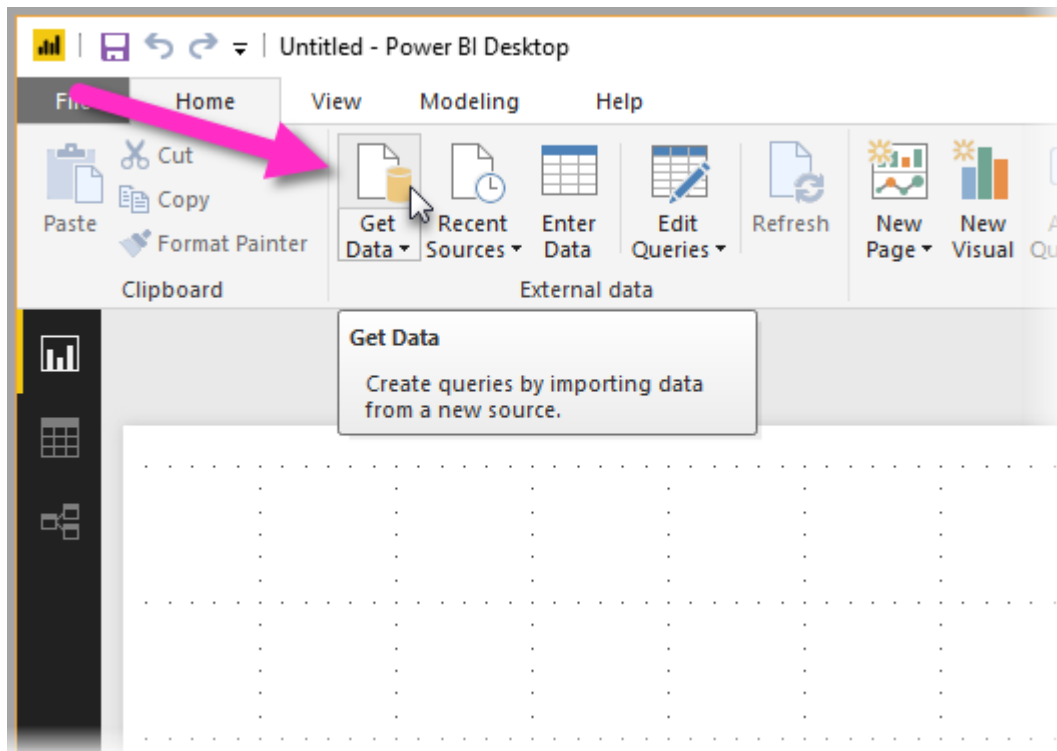
Once you install **Power BI Desktop**, launch the application so it's running on your local computer. You're presented with a Power BI tutorial. Follow the tutorial or click this away and start with a blank canvas, which is where you create visuals and reports from data to which you connect.



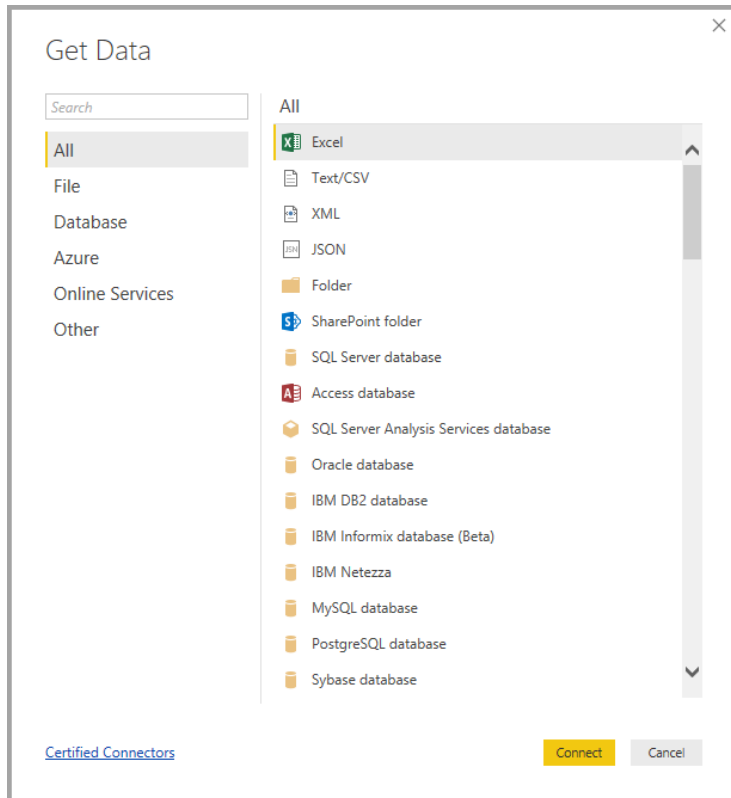
Connect to data

With **Power BI Desktop** you can connect to many different types of data. You can connect to basic data sources such as a Microsoft Excel file, and you can connect to online services that contain all sorts of data such as Salesforce, Microsoft Dynamics, Azure Blob Storage, and many more.

To connect to data, from the **Home** ribbon select **Get Data**.

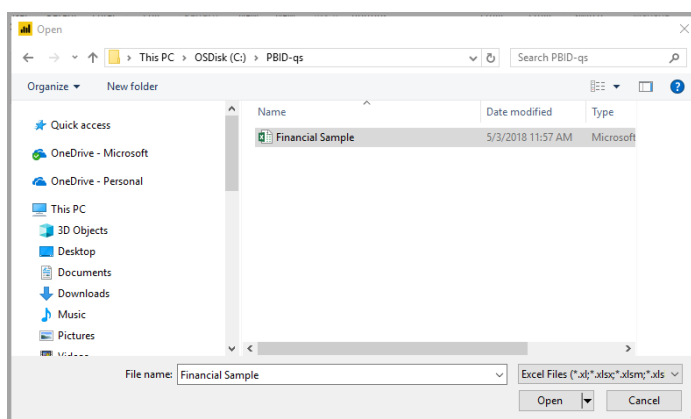


The **Get Data** window appears, where you can choose from the many different data sources to which **Power BI Desktop** can connect. In this quick start we use the Excel workbook that you downloaded, described in the *Prerequisites* section at the beginning of this article.

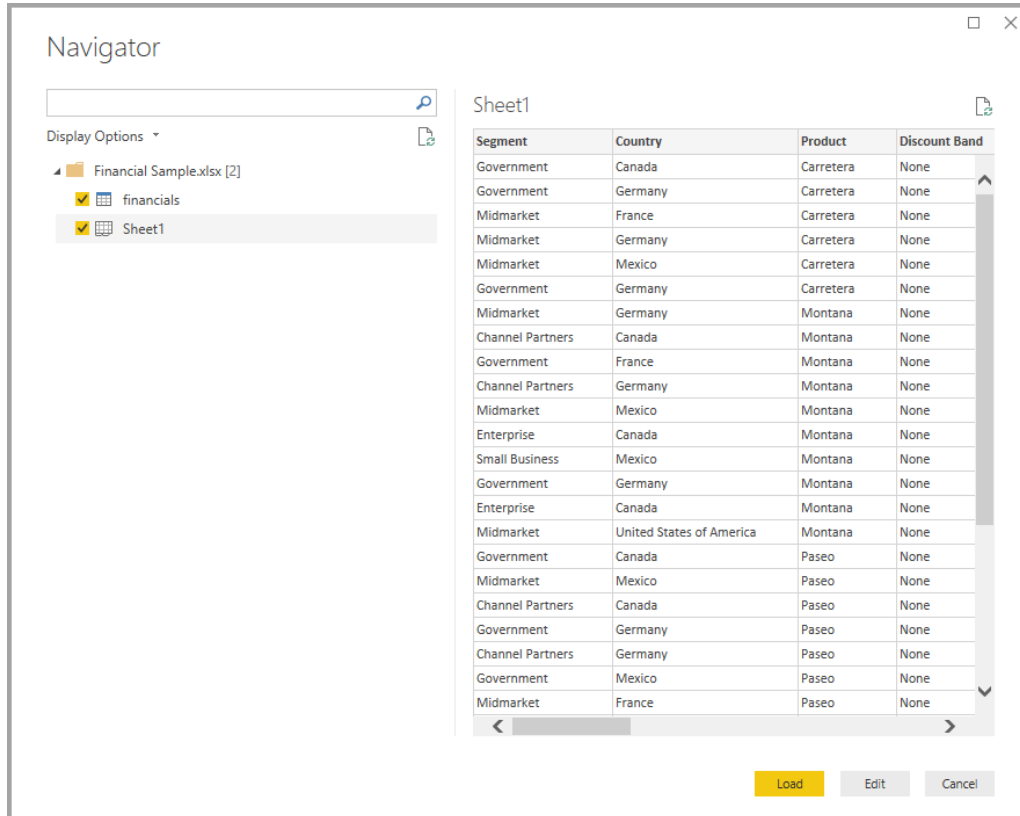


Since this is an Excel file, we select **Excel** from the **Get Data** window, then select the **Connect** button.

We're prompted to provide the location of the Excel file to which we want to connect. The downloaded file is called *Financial Sample* so we select that file, and then select **Open**.



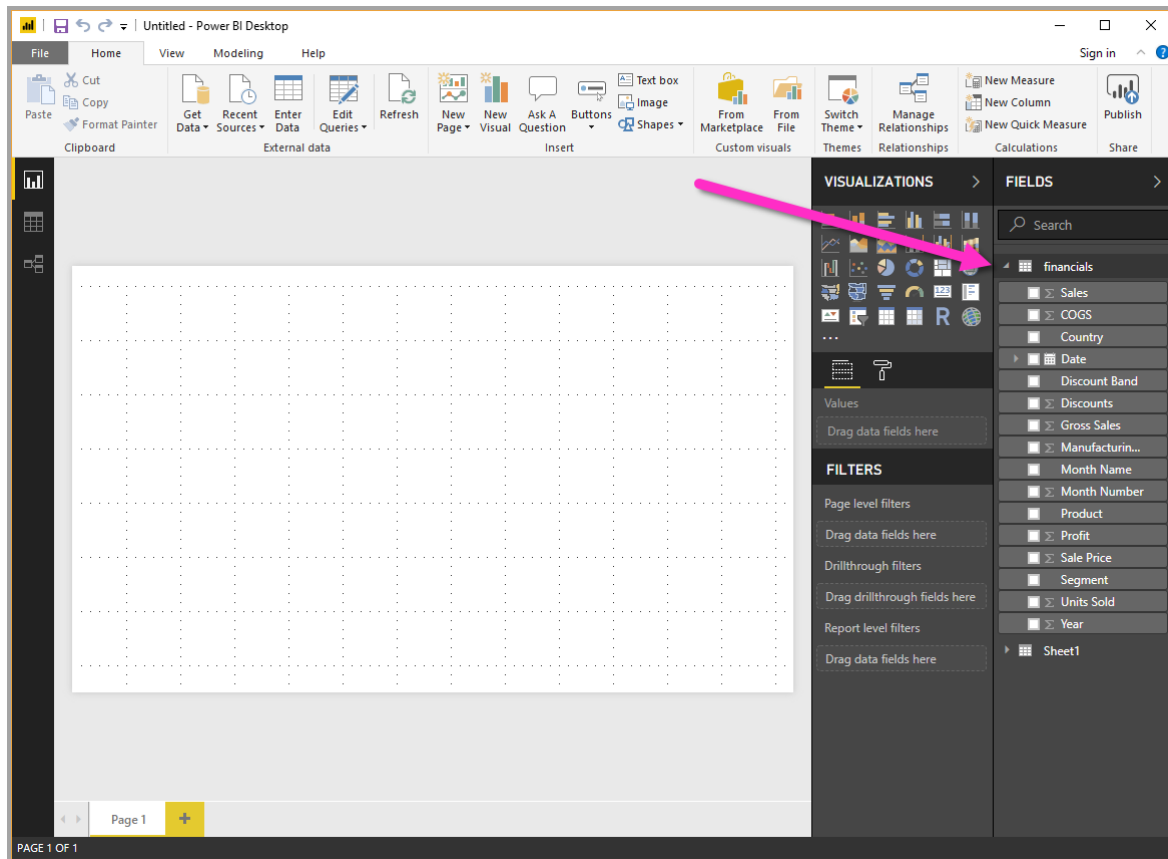
Power BI Desktop then loads the workbook and reads its contents, and shows you the available data in the file using the **Navigator** window, where you can choose which data you would like to load into Power BI Desktop. You select the tables by marking the checkboxes beside each table you want to import. In this case, we'll import both available tables.



Once you've made your selections, select **Load** to import the data into Power BI Desktop.

View data in the Fields pane

Once you've loaded the tables, the **Fields** pane shows you the data. You can expand each table by selecting the triangle beside its name. In the following image, the *financials* table is expanded, showing each of its fields.



And that's it! You've connected to data in **Power BI Desktop**, loaded that data, and now you can see all the available fields within those tables.

Tutorial: Shape and combine data in Power BI Desktop

With Power BI Desktop, you can connect to many different types of data sources, then shape the data to meet your needs, enabling you to create visual reports to share with others. *Shaping* data means transforming the data: renaming columns or tables, changing text to numbers, removing rows, setting the first row as headers, and so on. *Combining* data means connecting to two or more data sources, shaping them as needed, then consolidating them into a useful query.

In this tutorial, you'll learn how to:

- Shape data by using Query Editor.
- Connect to different data sources.
- Combine those data sources, and create a data model to use in reports.

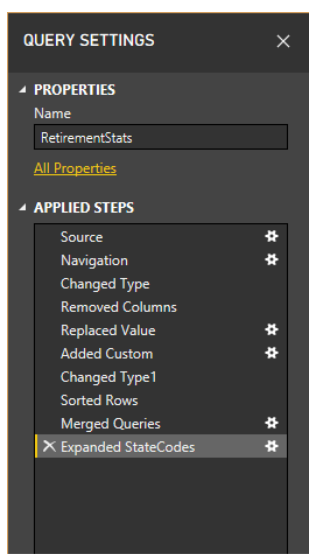
This tutorial demonstrates how to shape a query by using Power BI Desktop, highlighting the most common tasks. The query used here is described in more detail, including how to create the query from scratch, in [Getting Started with Power BI Desktop](#).

Query Editor in Power BI Desktop makes ample use of right-click menus, as well as the **Transform** ribbon. Most of what you can select in the ribbon is also available by right-clicking an item, such as a column, and choosing from the menu that appears.

Shape data

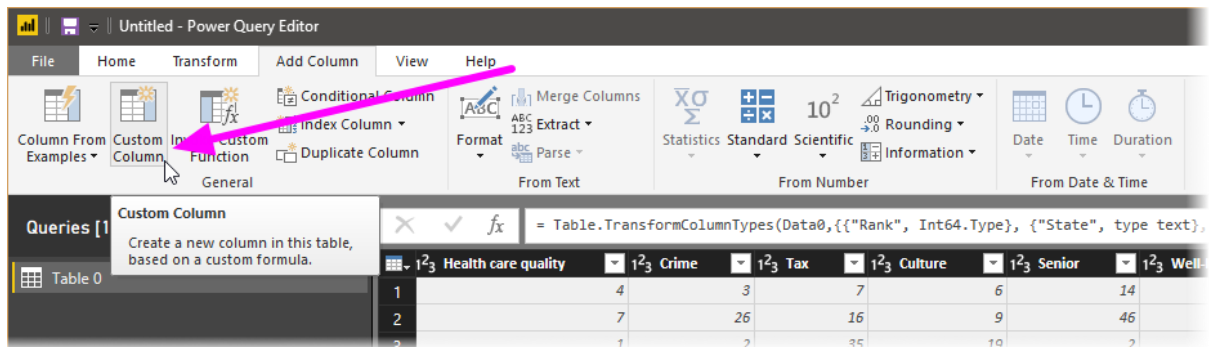
When you shape data in Query Editor, you provide step-by-step instructions for Query Editor to carry out for you to adjust the data as it loads and presents it. The original data source isn't affected; only this particular view of the data is adjusted, or *shaped*.

The steps you specify (such as rename a table, transform a data type, or delete a column) are recorded by Query Editor. Each time this query connects to the data source, Query Editor carries out those steps so that the data is always shaped the way you specify. This process occurs whenever you use Query Editor, or for anyone who uses your shared query, such as on the Power BI service. Those steps are captured, sequentially, in the **Query Settings** pane, under **Applied Steps**. We'll go through each of those steps in the next few paragraphs.



From [Getting Started with Power BI Desktop](#), let's use the retirement data, which we found by connecting to a web data source, to shape that data to fit our needs. We'll add a custom column to calculate rank based on all data being equal factors, and compare this column to the existing column, **Rank**.

1. From the **Add Column** ribbon, select **Custom Column**, which lets you add a custom column.

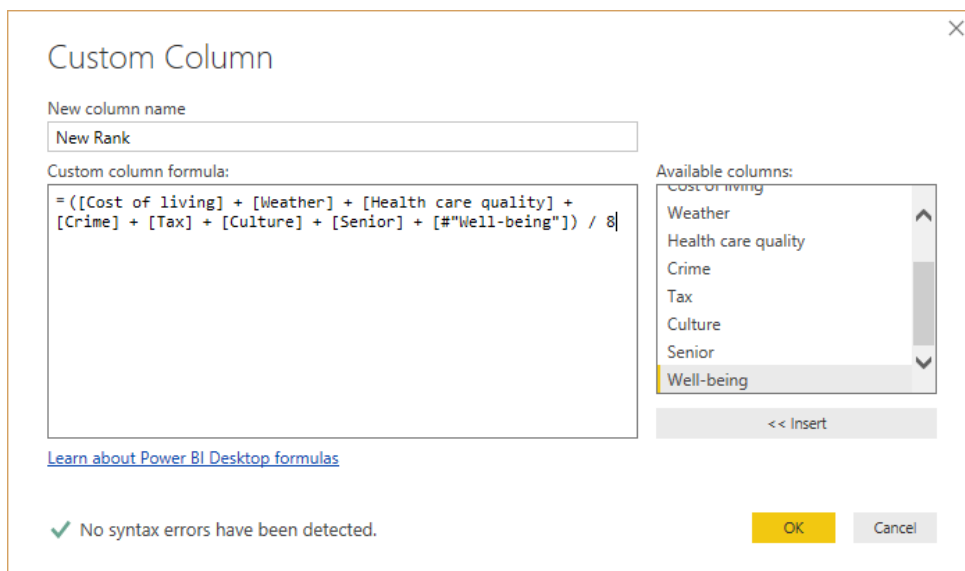


2. In the **Custom Column** window, in **New column name**, enter *New Rank*. In **Custom column formula**, enter the following data:

Copy

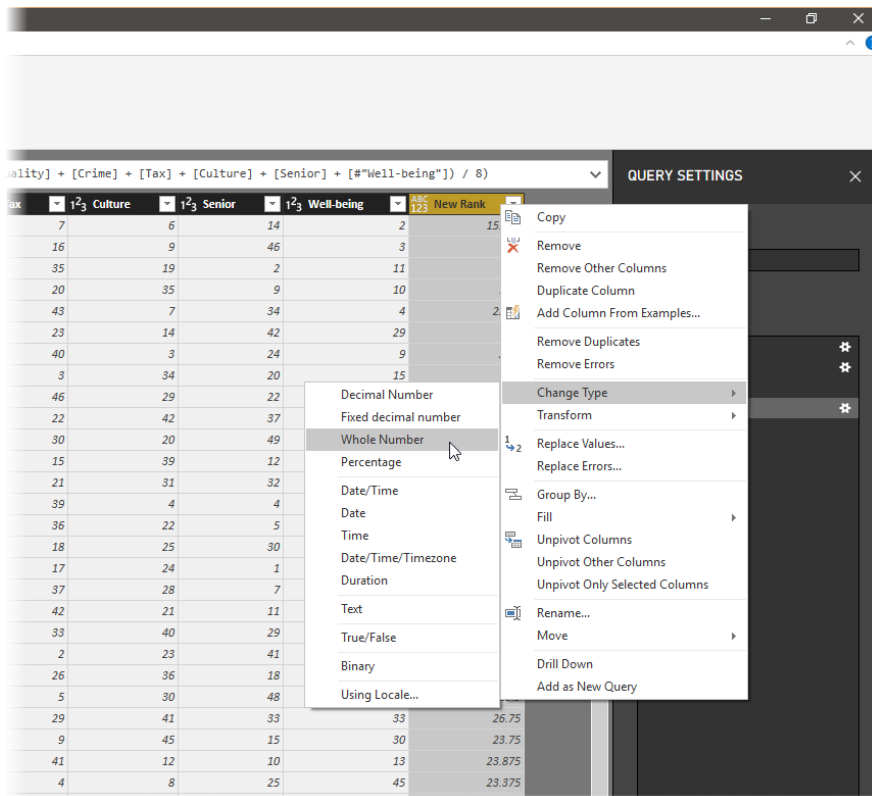
$$([Cost\ of\ living] + [Weather] + [Health\ care\ quality] + [Crime] + [Tax] + [Culture] + [Senior] + [Well-being]) / 8$$

3. Make sure the status message is *No syntax errors have been detected*, and select **OK**.

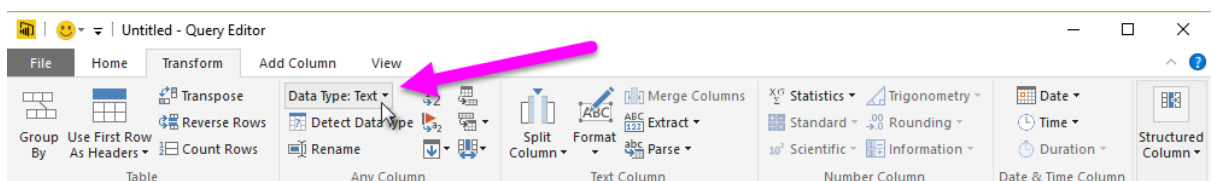


4. To keep column data consistent, transform the new column values to whole numbers. To change them, right-click the column header, and then select **Change Type > Whole Number**.

If you need to choose more than one column, select a column, hold down **SHIFT**, select additional adjacent columns, and then right-click a column header. You can also use the **CTRL** key to choose non-adjacent columns.



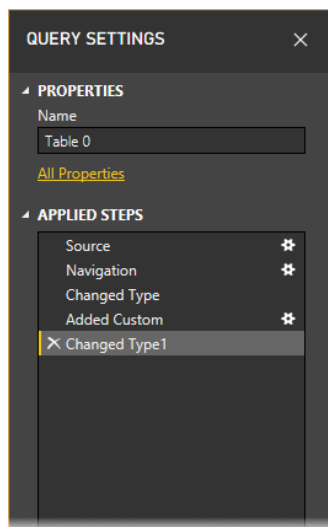
5. To *transform* column data types, in which you transform the current data type to another, select **Data Type Text** from the **Transform** ribbon.



6. In **Query Settings**, the **Applied Steps** list reflects any shaping steps applied to the data. To remove a step from the shaping process, select the **X** to the left of the step.

In the following image, the **Applied Steps** list reflects the added steps so far:

- **Source:** Connecting to the website.
- **Navigation:** Selecting the table.
- **Changed Type:** Changing text-based number columns from *Text* to *Whole Number*.
- **Added Custom:** Adding a custom column.
- **Changed Type1:** The last applied step.



Adjust data

Before we can work with this query, we need to make a few changes to adjust its data:

- Adjust the rankings by removing a column.

We've decided **Cost of living** is a non-factor in our results. After removing this column, we find that the data remains unchanged.

- Fix a few errors.

Because we removed a column, we need to readjust our calculations in the **New Rank** column, which involves changing a formula.

- Sort the data.

Sort the data based on the **New Rank** and **Rank** columns.

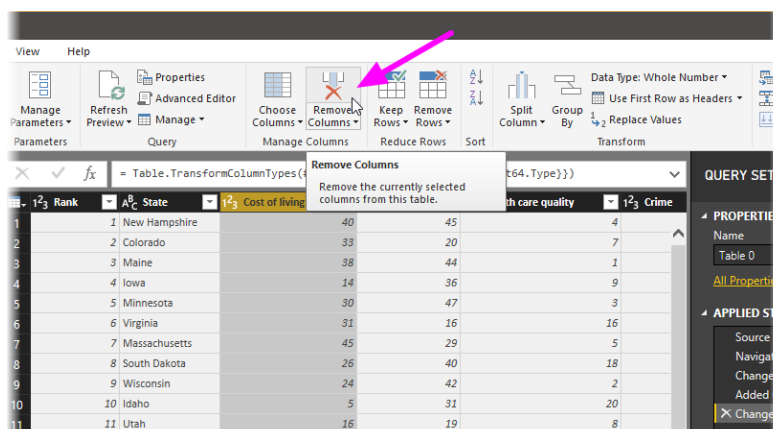
- Replace the data.

We'll highlight how to replace a specific value and the need of inserting an **Applied Step**.

- Change the table name.

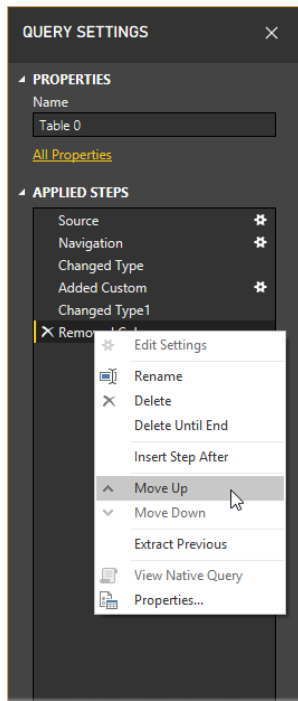
Because **Table 0** isn't a useful descriptor for the table, we'll change its name.

- To remove the **Cost of living** column, select the column, choose the **Home** tab from the ribbon, and then select **Remove Columns**.



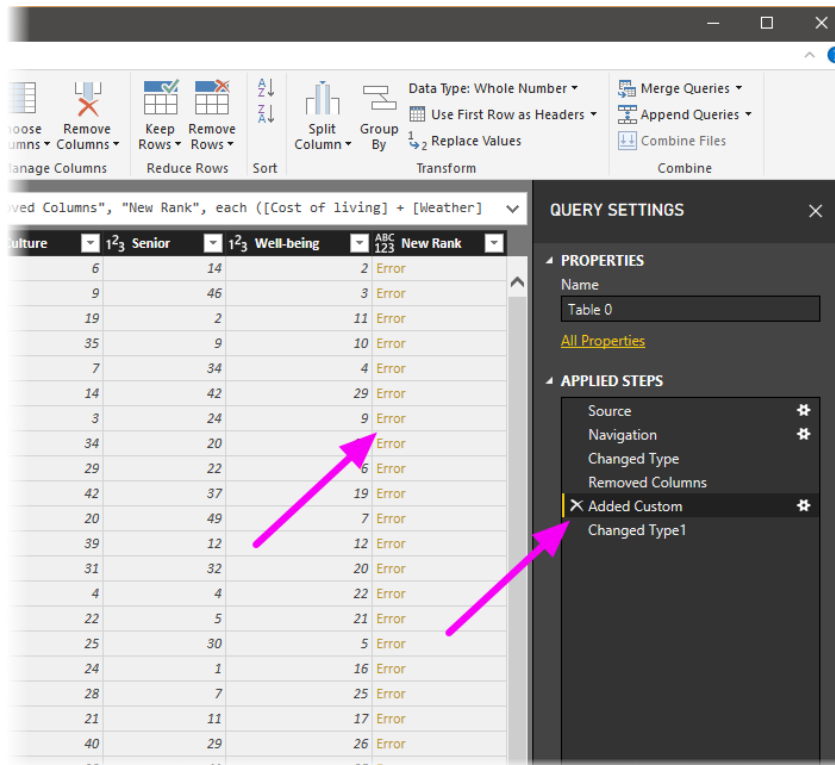
Notice the **New Rank** values haven't changed, due to the ordering of the steps. Because Query Editor records the steps sequentially, yet independently, of each other, you can move each **Applied Step** up or down in the sequence.

2. Right-click a step. Query Editor provides a menu that lets you do the following tasks:
 - **Rename**; Rename the step.
 - **Delete**: Delete the step.
 - **Delete Until End**: Remove the current step, and all subsequent steps.
 - **Move Up**: Move the step up in the list.
 - **Move Down**: Move the step down in the list.
3. Move up the last step, **Removed Columns**, to just above the **Added Custom** step.

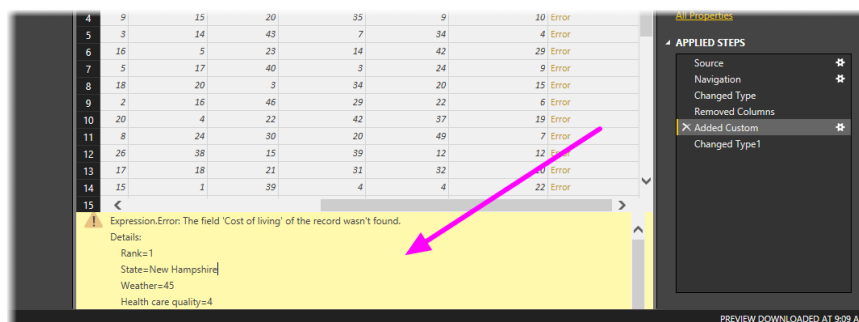


4. Select the **Added Custom** step.

Notice the data now shows *Error*, which we'll need to address.

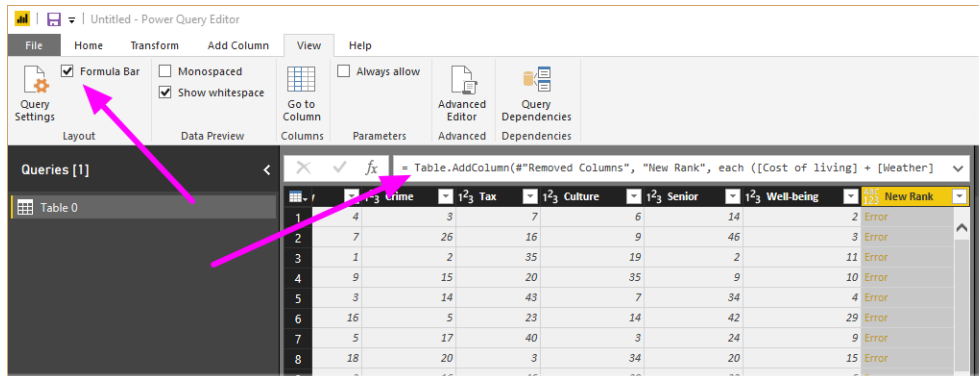


There are a few ways to get more information about each error. If you select the cell without clicking on the word *Error*, Query Editor displays the error information on the bottom of the window.



If you select the word *Error* directly, Query Editor creates an **Applied Step** in the **Query Settings** pane and displays information about the error.

5. Because we don't need to display information about the errors, select **Cancel**.
6. To fix the errors, select the **New Rank** column, then display the column's data formula by selecting the **Formula Bar** checkbox from the **View** tab.



7. Remove the *Cost of living* parameter and decrement the divisor, by changing the formula as follows:

Copy

```
Table.AddColumn("#Removed Columns", "New Rank", each ([Weather] + [Health care quality] + [Crime] + [Tax] + [Culture] + [Senior] + [#"Well-being"]) / 7)
```

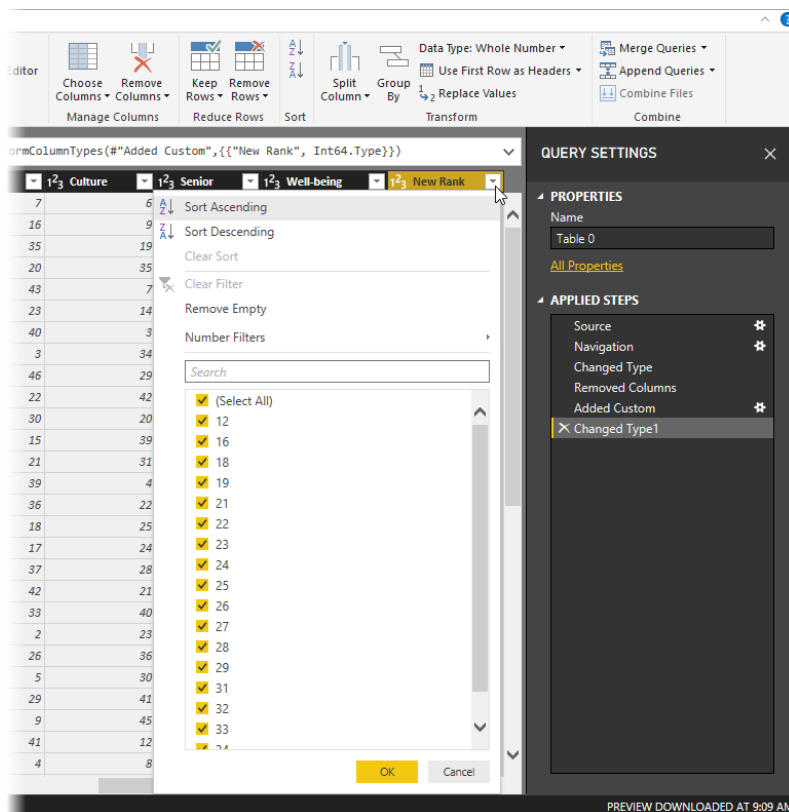
8. Select the green checkmark to the left of the formula box or press **Enter**.

Query Editor replaces the data with the revised values and the **Added Custom** step completes with no errors.

Note

You can also select **Remove Errors**, by using the ribbon or the right-click menu, which removes any rows that have errors. However, we didn't want to do so in this tutorial because we wanted to preserve the data in the table.

1. Sort the data based on the **New Rank** column. First, select the last applied step, **Changed Type1** to display the most recent data. Then, select the drop-down located next to the **New Rank** column header and select **Sort Ascending**.



The data is now sorted according to **New Rank**. However, if you look at the **Rank** column, you'll notice the data isn't sorted properly in cases where the **New Rank** value is a tie. We'll fix it in the next step.

2. To fix the data sorting issue, select the **New Rank** column and change the formula in the **Formula Bar** to the following formula:

Copy

```
= Table.Sort("#Changed Type1",{ "New Rank", Order.Ascending},{ "Rank", Order.Ascending})
```

3. Select the green checkmark to the left of the formula box or press **Enter**.

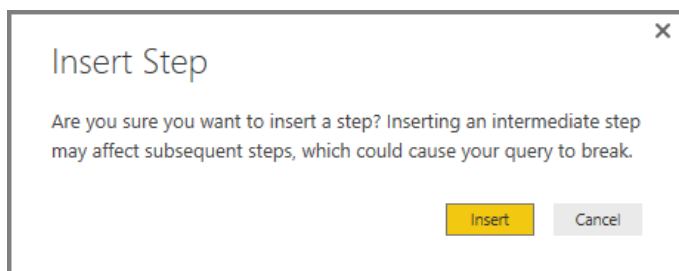
The rows are now ordered in accordance with both **New Rank** and **Rank**. In addition, you can select an **Applied Step** anywhere in the list, and continue shaping the data at that point in the sequence. Query Editor automatically inserts a new step directly after the currently selected **Applied Step**.

- In **Applied Step**, select the step preceding the custom column, which is the **Removed Columns** step. Here we'll replace the value of the **Weather** ranking in Arizona. Right-click the appropriate cell that contains Arizona's **Weather** ranking, and then select **Replace Values**. Note which **Applied Step** is currently selected.

	1 ² Rank	A ^B State	1 ² Weather	1 ² Health care quality	1 ² Crime	1 ² Tax
1	1	New Hampshire	45	4	3	
2	2	Colorado	20	7	26	
3	3	Maine	44	1	2	
4	4	Iowa	36	9	15	
5	5	Minnesota	47	3	14	
6	6	Virginia	16	16	5	
7	7	Massachusetts	29	5	17	
8	8	South Dakota	40	18	20	
9	9	Wisconsin	42	2	16	
10	10	Idaho	31	20	4	
11	11	Utah	19	8	24	
12	12	Arizona	2	26	38	
13	13	Nebraska		17	18	
14	14	Vermont		15	1	
15	15	Pennsylvania		14	12	
16	16	North Dakota		22	11	
17	17	Florida		30	39	
18	18	Delaware		6	40	
19	19	Rhode Island	26	10	9	
20	20	North Carolina	11	12	29	
21	21	Wyoming	41	44	8	
22	22	Michigan	43	19	22	

- Select **Insert**.

Because we're inserting a step, Query Editor warns us about the danger of doing so; subsequent steps could cause the query to break.



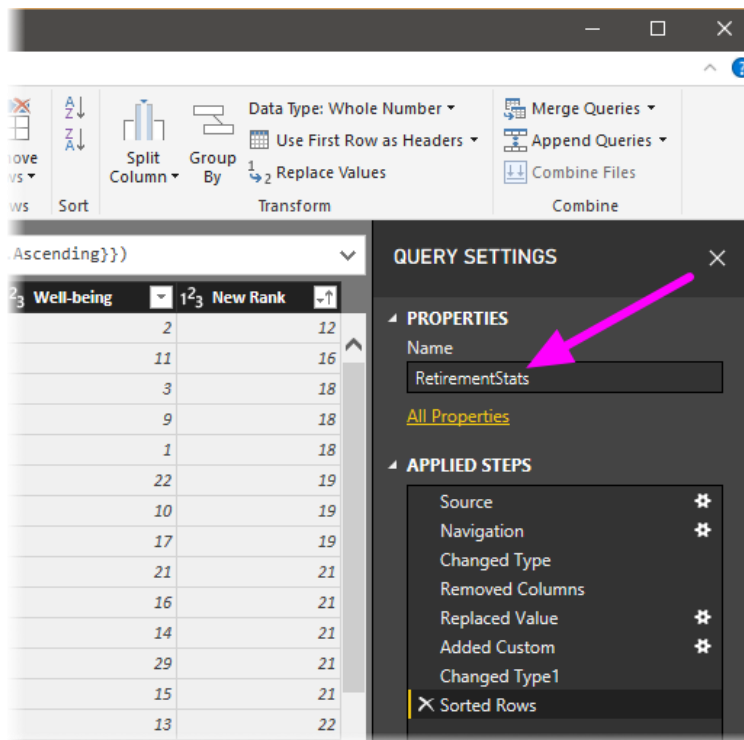
- Change the data value to **51**.

Query Editor replaces the data for Arizona. When you create a new **Applied Step**, Query Editor names it based on the action; in this case, **Replaced Value**. If you have more than one step with the same name in your query, Query Editor adds a number (in sequence) to each subsequent **Applied Step** to differentiate between them.

- Select the last **Applied Step**, **Sorted Rows**.

Notice the data has changed regarding Arizona's new ranking. This change occurs because we inserted the **Replaced Value** step in the correct location, before the **Added Custom** step.

8. Lastly, we want to change the name of that table to something descriptive. In the **Query Settings** pane, under **Properties**, enter the new name of the table, and then select **Enter**. Name this table *RetirementStats*.



When we start creating reports, it's useful to have descriptive table names, especially when we connect to multiple data sources, which are listed in the **Fields** pane of the **Report** view.

We've now shaped our data to the extent we need to. Next let's connect to another data source, and combine data.

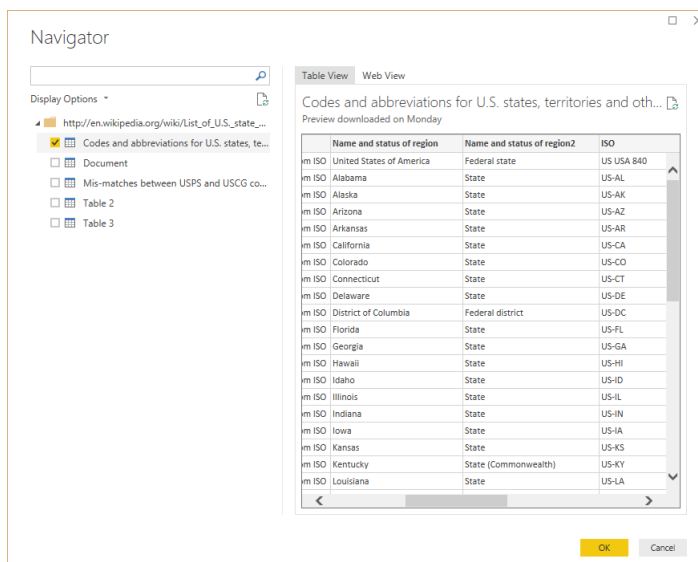
Combine data

The data about various states is interesting, and will be useful for building additional analysis efforts and queries. But there's one problem: most data out there uses a two-letter abbreviation for state codes, not the full name of the state. We need a way to associate state names with their abbreviations.

We're in luck; there's another public data source that does just that, but it needs a fair amount of shaping before we can connect it to our retirement table. TO shape the data, follow these steps:

1. From the **Home** ribbon in Query Editor, select **New Source > Web**.
2. Enter the address of the website for state abbreviations, https://en.wikipedia.org/wiki/List_of_U.S._state_abbreviations, and then select **Connect**.

The Navigator displays the content of the website.



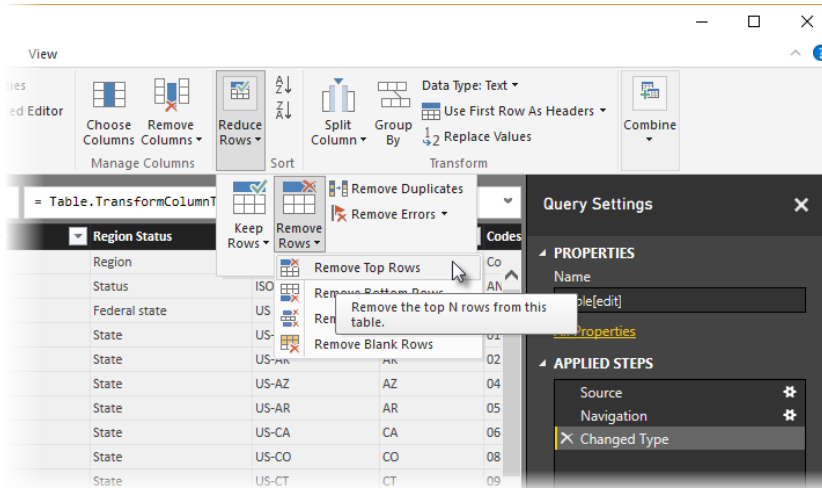
3. Select **Codes and abbreviations**.

Tip

It will take quite a bit of shaping to pare this table's data down to what we want. Is there a faster or easier way to accomplish the steps below? Yes, we could create a *relationship* between the two tables, and shape the data based on that relationship. The following steps are still good to learn for working with tables; however, relationships can help you quickly use data from multiple tables.

To get the data into shape, follow these steps:

1. Remove the top row. Because it's a result of the way that the web page's table was created, we don't need it. From the **Home** ribbon, select **Reduce Rows > Remove Rows > Remove Top Rows**.

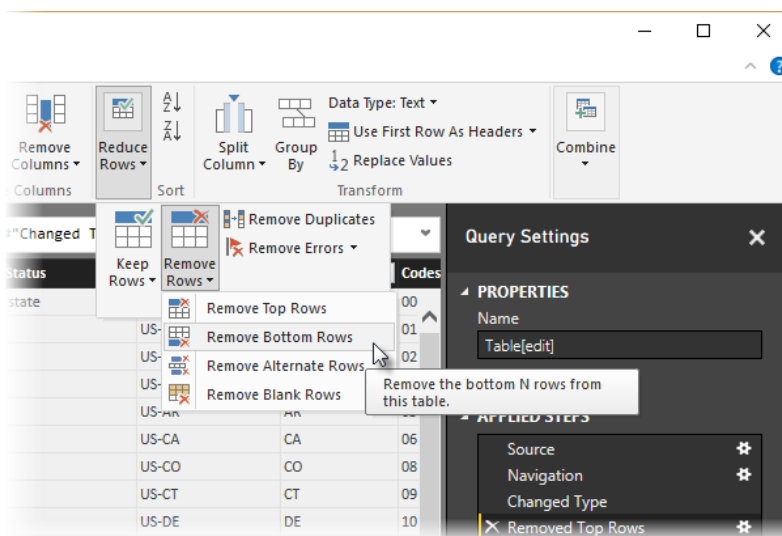


The **Remove Top Rows** window appears, letting you specify how many rows you want to remove.

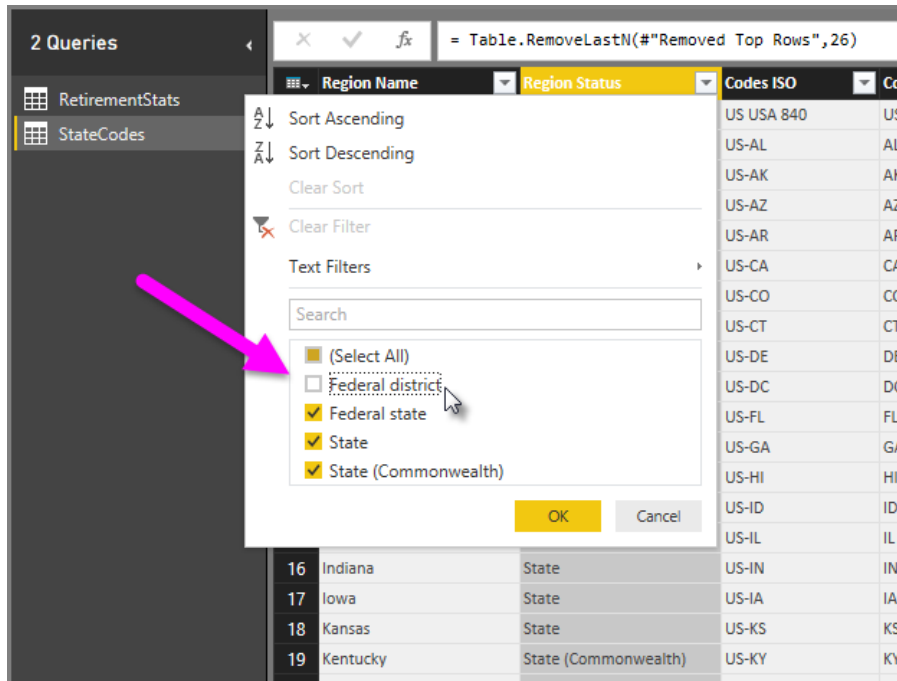
Note

If Power BI accidentally imports the table headers as a row in your data table, you can select **Use First Row As Headers** from the **Home** tab, or from the **Transform** tab in the ribbon, to fix your table.

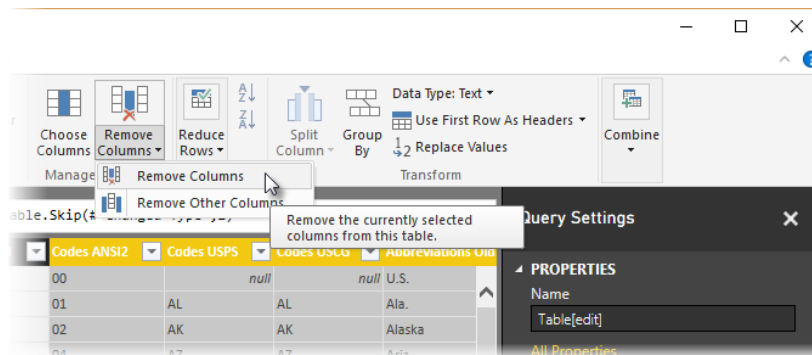
2. Remove the bottom 26 rows. These rows are U.S. territories, which we don't need to include. From the **Home** ribbon, select **Reduce Rows > Remove Rows > Remove Bottom Rows**.



- Because the RetirementStats table doesn't have information for Washington DC, we need to filter it from our list. Select the **Region Status** drop-down, then clear the checkbox beside **Federal district**.



- Remove a few unneeded columns. Because we need only the mapping of each state to its official two-letter abbreviation, we can remove the following columns: **Column1**, **Column3**, **Column4**, and **Column6** through **Column11**. First select **Column1**, then hold down the **CTRL** key and select each of the other columns to be removed. From the **Home** tab on the ribbon, select **Remove Columns > Remove Columns**.



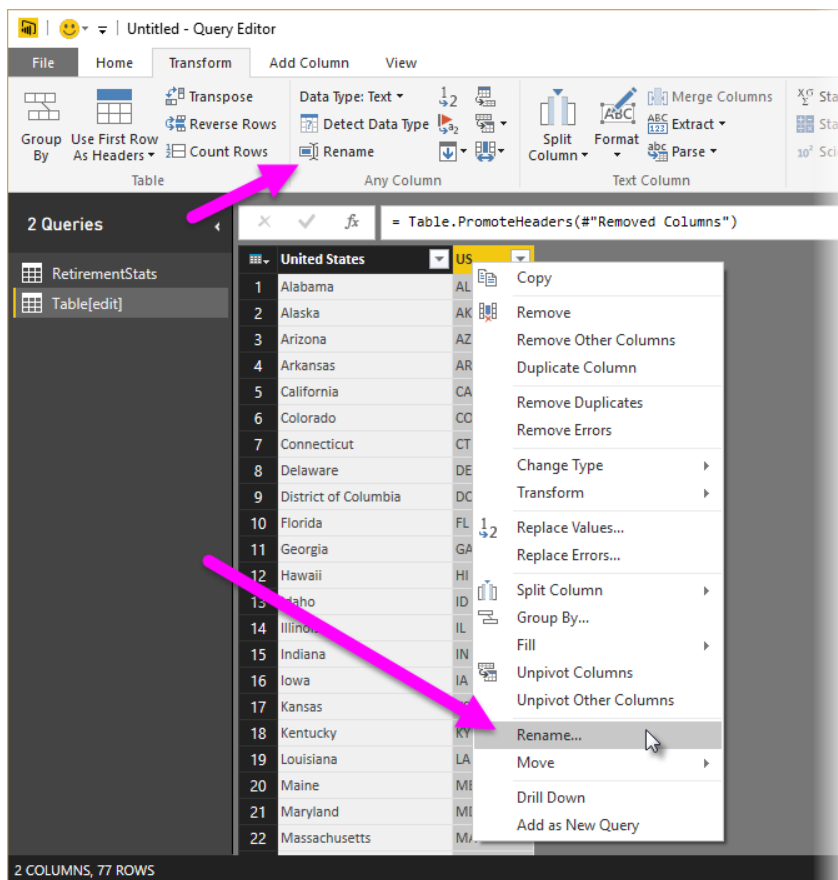
Note

This is a good time to point out that the *sequence* of applied steps in Query Editor is important, and can affect how the data is shaped. It's also important to consider how one step may impact another subsequent step; if you remove a step from the Applied Steps, subsequent steps may not behave as originally intended, because of the impact of the query's sequence of steps.

Note

When you resize the Query Editor window to make the width smaller, some ribbon items are condensed to make the best use of visible space. When you increase the width of the Query Editor window, the ribbon items expand to make the most use of the increased ribbon area.

5. Rename the columns and the table. There are a few ways to rename a column: First, select the column, then either select **Rename** from the **Transform** tab on the ribbon, or right-click and select **Rename**. The following image has arrows pointing to both options; you only need to choose one.



6. Rename the columns to *State Name* and *State Code*. To rename the table, enter the **Name** in the **Query Settings** pane. Name this table *StateCodes*.

Combine queries

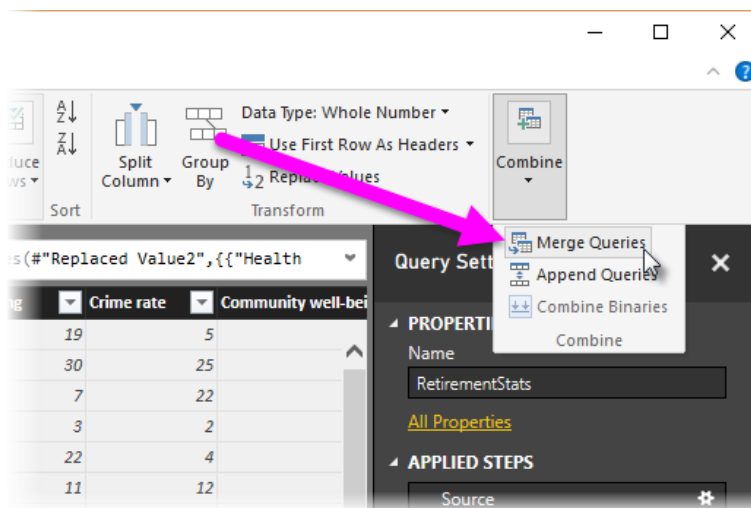
Now that we've shaped the StateCodes table the way we want, let's combine these two tables, or queries, into one. Because the tables we now have are a result of the queries we applied to the data, they're often referred to as *queries*.

There are two primary ways of combining queries: *merging* and *appending*.

- When you have one or more columns that you'd like to add to another query, you *merge* the queries.
- When you have additional rows of data that you'd like to add to an existing query, you *append* the query.

In this case, we want to merge the queries. To do so, follow these steps:

1. From the left pane of Query Editor, select the query *into which* you want the other query to merge. In this case, it's **RetirementStats**.
2. Select **Combine > Merge Queries** from the **Home** tab on the ribbon.

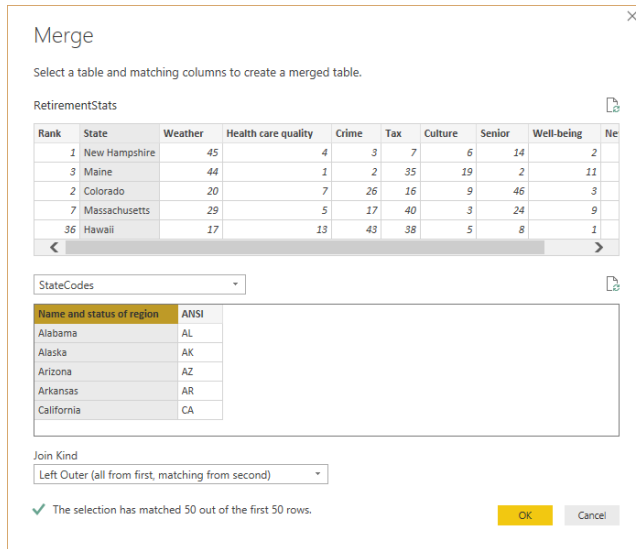


You may be prompted to set the privacy levels, to ensure the data is combined without including or transferring data you don't want transferred.

The **Merge** window appears. It prompts you to select which table you'd like merged into the selected table, and the matching columns to use for the merge.

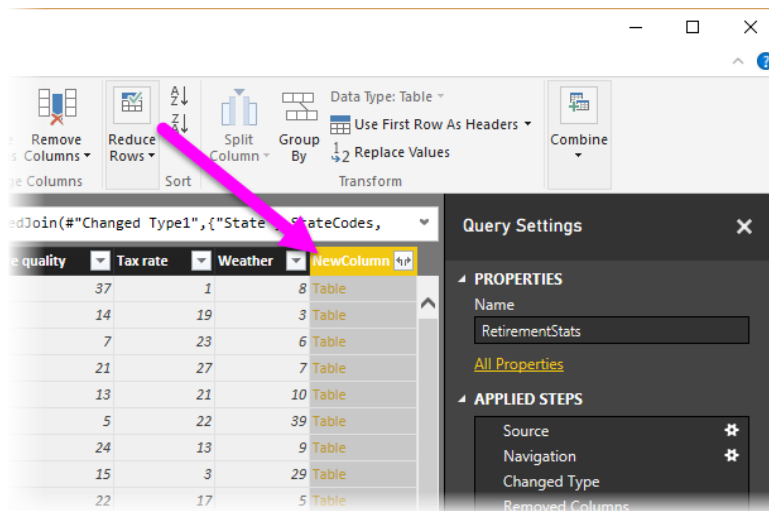
3. Select **State** from the RetirementStats table, then select the **StateCodes** query.

When you select the correct matching columns, the **OK** button is enabled.



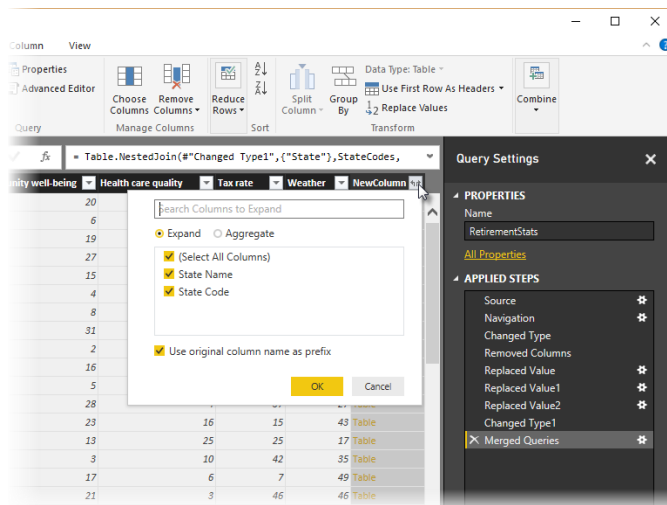
4. Select **OK**.

Query Editor creates a **NewColumn** column at the end of the query, which contains the contents of the table (query) that was merged with the existing query. All columns from the merged query are condensed into the **NewColumn** column, but you can **Expand** the table and include whichever columns you want.



5. To expand the merged table, and select which columns to include, select the expand icon (⌵).

The **Expand** window appears.



6. In this case, we want only the **State Code** column. Select that column, clear **Use original column name as prefix**, and then select **OK**.

If we had left the checkbox selected for **Use original column name as prefix**, the merged column would be named **NewColumn.State Code**.

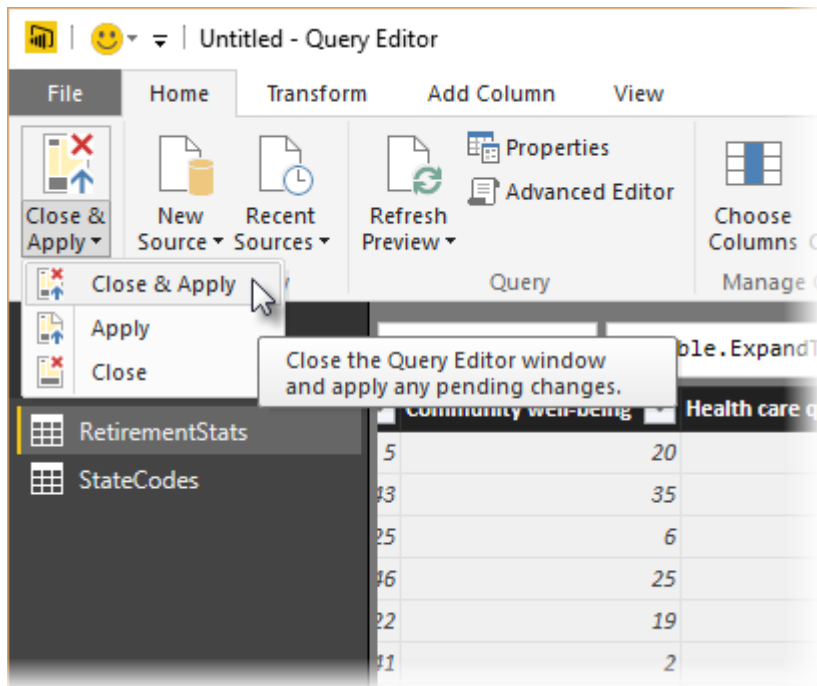
Note

Want to explore how to bring in the NewColumn table? You can experiment a bit, and if you don't like the results, just delete that step from the **Applied Steps** list in the **Query Settings** pane; your query returns to the state prior to applying that **Expand** step. You can do this as many times as you like until the expand process looks the way you want it.

We now have a single query (table) that combines two data sources, each of which has been shaped to meet our needs. This query can serve as a basis for many additional and interesting data connections, such as housing cost statistics, demographics, or job opportunities in any state.

7. To apply your changes and close Query Editor, select **Close & Apply** from the **Home** ribbon tab.

The transformed dataset appears in Power BI Desktop, ready to be used for creating reports.



Tutorial: Analyze web page data using Power BI Desktop

As a long-time soccer fan, you want to report on the UEFA European Championship (Euro Cup) winners over the years. With Power BI Desktop, you can import this data from a web page into a report and create visualizations that show the data. In this tutorial, you learn how to use Power BI Desktop to:

- Connect to a web data source and navigate across its available tables,
- Shape and transform data in the **Power Query Editor**,
- Name a query and import it into a Power BI Desktop report, and
- Create and customize a map and a pie chart visualization.

Connect to a web data source

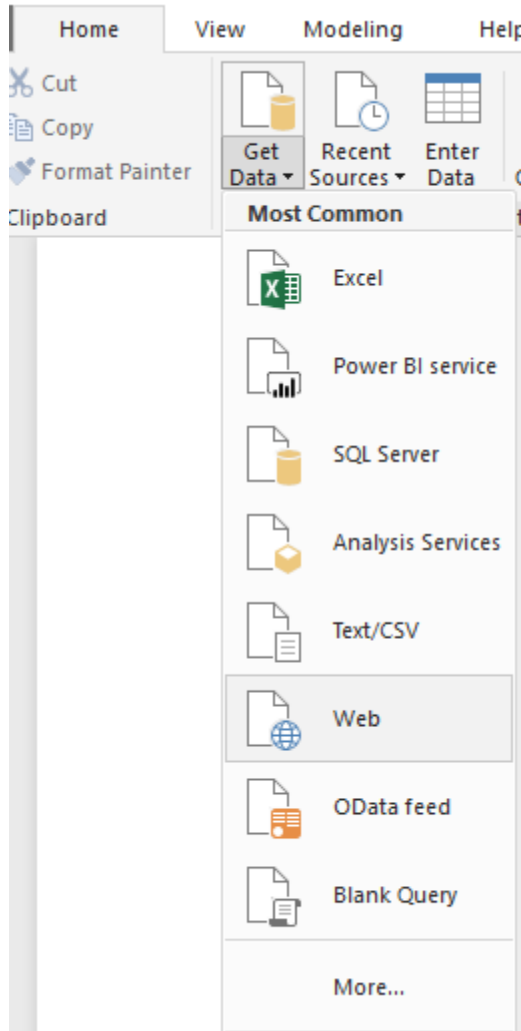
You can get the UEFA winners data from the Results table on the UEFA European Football Championship Wikipedia page at https://en.wikipedia.org/wiki/UEFA_European_Football_Championship.

Results [edit]									
See also: <i>List of UEFA European Championship finals</i>									
Year	Host	Winner	Score	Runner-up	Third place	Score	Fourth place	Number of teams	
1960 Details	 France	 Soviet Union	2-1 (a.e.t.)	 Yugoslavia	 Czechoslovakia	2-0	 France	4	
1964 Details	 Spain	 Spain	2-1	 Soviet Union	 Hungary	3-1 (a.e.t.)	 Denmark	4	
1968 Details	 Italy	 Italy	1-1 (a.e.t.) 2-0 (replay)	 Yugoslavia	 England	2-0	 Soviet Union	4	
1972 Details	 Belgium	 West Germany	3-0	 Soviet Union	 Belgium	2-1	 Hungary	4	
1976 Details	 Yugoslavia	 Czechoslovakia	2-2 (a.e.t.) (5-3 p)	 West Germany	 Netherlands	3-2 (a.e.t.)	 Yugoslavia	4	
1980 Details	 Italy	 West Germany	2-1	 Belgium	 Czechoslovakia	1-1 ^(a) (9-8 p)	 Italy	8	
Year	Host	Winner	Score	Runner-up	Losing semi-finalists ^(a)				Number of teams
1984 Details	 France	 France	2-0	 Spain	 Denmark and Portugal				8
1988 Details	 West Germany	 Netherlands	2-0	 Soviet Union	 Italy and West Germany				8
1992 Details	 Sweden	 Denmark	2-0	 Germany	 Netherlands and Sweden				8
1996 Details	 England	 Germany	2-1 (a.g.g.a.t.)	 Czech Republic	 England and France				16
2000 Details	 Belgium & Netherlands	 France	2-1 (a.g.g.a.t.)	 Italy	 Netherlands and Portugal				16
2004 Details	 Portugal	 Greece	1-0	 Portugal	 Czech Republic and Netherlands				16
2008 Details	 Austria & Switzerland	 Spain	1-0	 Germany	 Russia and Turkey				16
2012 Details	 Poland & Ukraine	 Spain	4-0	 Italy	 Germany and Portugal				16
2016 Details	 France	 Portugal	1-0 (a.e.t.)	 France	 Germany and Wales				24
2020 Details	 Pan-European								24

Note that Web connections are only established using basic authentication. Web sites requiring authentication may not work properly with the Web connector.

To import the data:

1. In the Power BI Desktop **Home** ribbon tab, drop down the arrow next to **Get Data**, and then select **Web**.



Note

You can also select the **Get Data** item itself, or select **Get Data** from the Power BI **Get started** dialog, then select **Web** from the **All** or **Other** section of the **Get Data** dialog box, and then select **Connect**.

2. In the **From Web** dialog box, paste the URL https://en.wikipedia.org/wiki/UEFA_European_Football_Championship into the **URL** text box, and then select **OK**.

From Web

☒ Basic

☐ Advanced

URL

OK

Cancel

After connecting to the Wikipedia web page, the Power BI **Navigator** dialog box shows a list of available tables on the page. You can select any of the table names to preview its data. The **Results[edit]** table has the data you want, although it is not exactly in the shape you want. You will reshape and clean up the data before loading it into your report.

Navigator

Display Options ▾

http://en.wikipedia.org/wiki/UEFA_European_F...

☐ Document
 ☐ Records and statistics[edit]
 ☒ Results[edit]
 ☐ Summary[edit]
 ☐ Table 1
 ☐ Table 10
 ☐ Table 11
 ☐ Table 12
 ☐ Table 13
 ☐ Table 14
 ☐ Table 5
 ☐ Table 6
 ☐ Table 7
 ☐ Table 8
 ☐ Table 9
 ☐ UEFA European Championship

Table View

Web View

Results[edit]

Year	Host		Final Winner	Final
Year	Host		<i>null</i> Final	Final
Year	Host		<i>null</i> Winner	Score
1960 Details	France		<i>null</i> Soviet Union	2–1 (a
1964 Details	Spain		<i>null</i> Spain	2–1
1968 Details	Italy		<i>null</i> Italy	1–1 (a
1972 Details	Belgium		<i>null</i> West Germany	3–0
1976 Details	Yugoslavia		<i>null</i> Czechoslovakia	2–2 (a
1980 Details	Italy		<i>null</i> West Germany	2–1
Year	Host		<i>null</i> Final	Final
Year	Host		<i>null</i> Winner	Score
1984 Details	France		<i>null</i> France	2–0
1988 Details	West Germany		<i>null</i> Netherlands	2–0
1992 Details	Sweden		<i>null</i> Denmark	2–0
1996 Details	England		<i>null</i> Germany	2–1 (a
2000 Details	Belgium & Netherlands		<i>null</i> France	2–1 (a
2004 Details	Portugal		<i>null</i> Greece	1–0
2008 Details	Austria & Switzerland		<i>null</i> Spain	1–0
2012 Details	Poland & Ukraine		<i>null</i> Spain	4–0
2016 Details	France		<i>null</i> Portugal	1–0 (a
2020 Details	Pan-European		<i>null</i>	<i>null</i>

<

>

Load

Edit

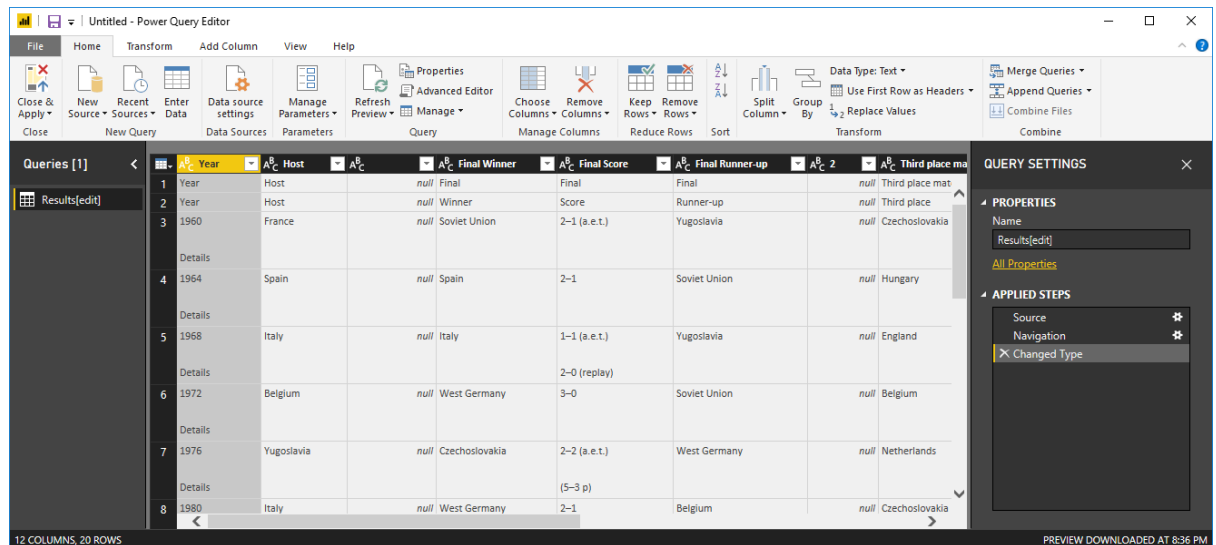
Cancel

Note

The **Preview** pane shows the most recent table selected, but all selected tables will load into the **Power Query Editor** when you select **Edit** or **Load**.

3. Select the **Results[edit]** table in the **Navigator** list, and then select **Edit**.

A preview of the table opens in the **Power Query Editor**, where you can apply transformations to clean up the data.



Shape data in Power Query Editor

You want to make the data easier to scan by displaying only the years and the countries that won. You can use the **Power Query Editor** to perform these data shaping and cleansing steps.

First, remove all the columns except **Year** and **Final Winners** from the table.

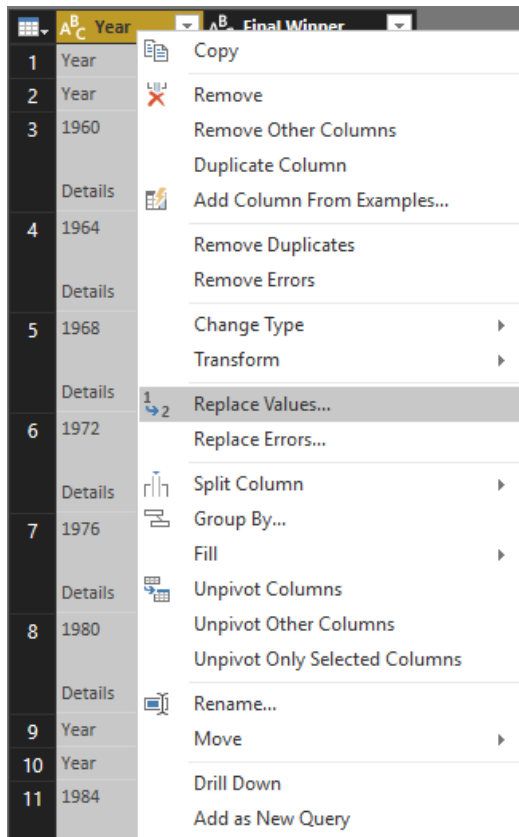
1. In the **Power Query Editor** grid, select the **Year** and **Final Winners** columns (hold down the **Ctrl** key to select multiple items).
2. Right-click and select **Remove Other Columns** from the dropdown, or select **Remove Columns > Remove Other Columns** from the **Manage Columns** group in the **Home** ribbon tab, to remove all other columns from the table.

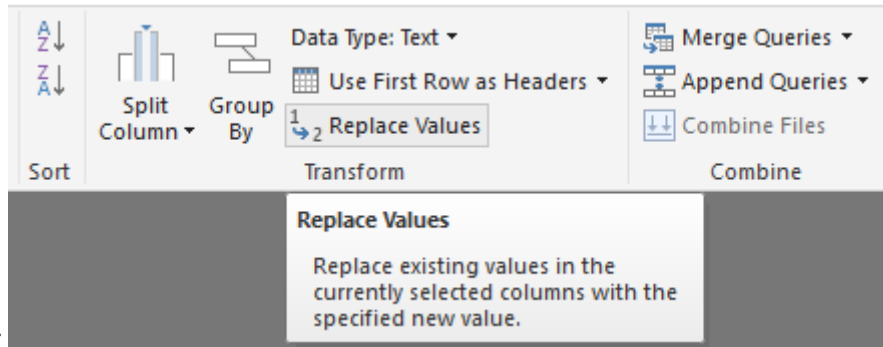
The screenshot displays the Power Query Editor interface. At the top, a table with 5 columns is visible: Year, Host, Final Winner, Final Score, and Final Runner-Up. The 'Year' and 'Final Winner' columns are selected. A right-click context menu is open over the 'Final Winner' column, showing options like Copy, Remove Columns, and **Remove Other Columns** (highlighted with a pink box). Below the table, the 'Manage Columns' ribbon tab is active, showing the 'Remove Other Columns' button (also highlighted with a pink box).

	Year	Host	Final Winner	Final Score	Final Runner-Up
1	Year	Host	null	Final	
2	Year	Host	null	Winner	
3	1960	France	null	Soviet Union	
	Details				
4	1964	Spain	null	Spain	
	Details				
5	1968	Italy	null	Italy	
	Details				
6	1972	Belgium	null	West Germany	
	Details				
7	1976	Yugoslavia	null	Czechoslovakia	
	Details				
8	1980	Italy	null	West Germany	

Next, remove the extra word **Details** from the **Year** column cells.

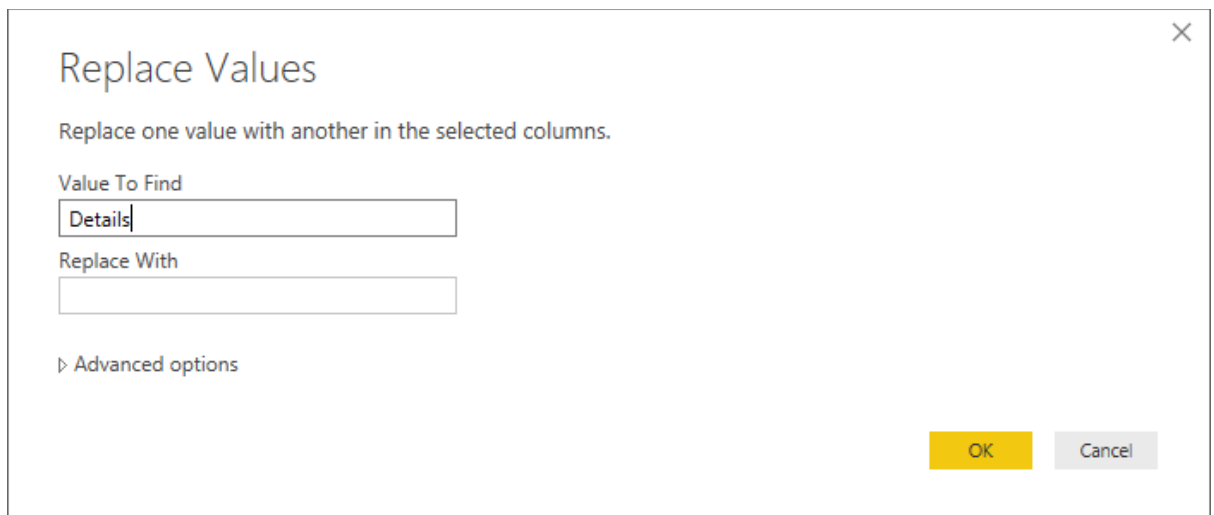
1. Select the **Year** column.
2. Right-click, and select **Replace Values** from the dropdown, or select **Replace Values** from the **Transform** group in the **Home** tab of the ribbon (also found in the **Any Column** group in the **Transform** tab).





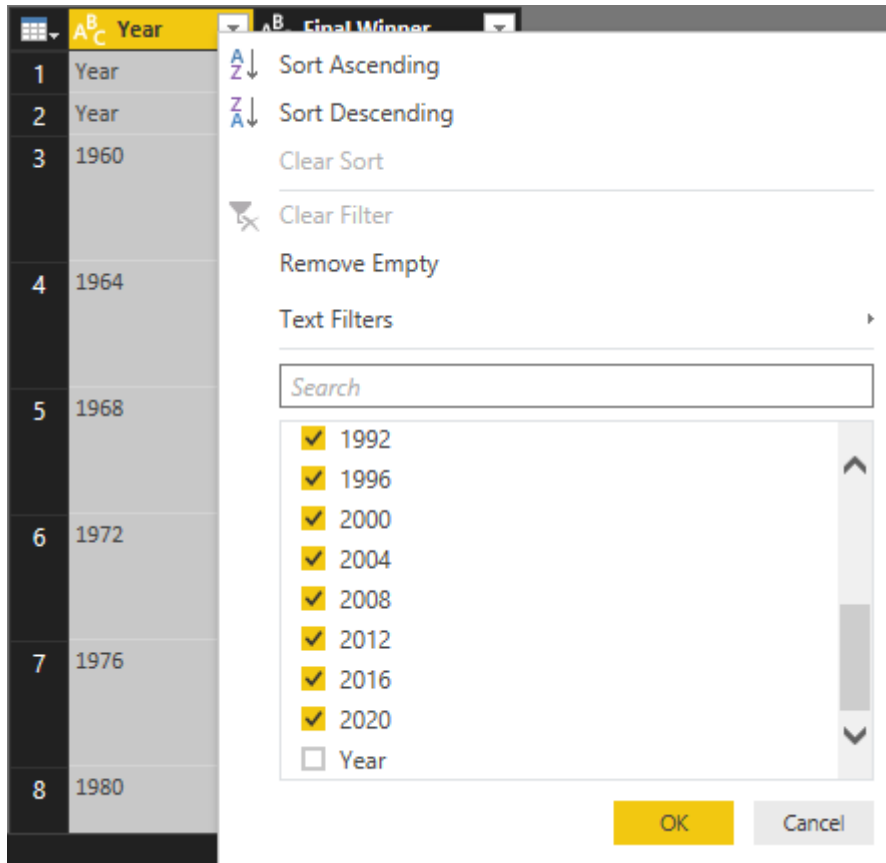
or

3. In the **Replace Values** dialog box, type **Details** in the **Value To Find** text box, leave the **Replace With** text box empty, and then select **OK** to delete the word "Details" from the **Year** entries.



Some **Year** cells only contain the word "Year" rather than year values. You can filter the **Year** column to only display rows that do not contain the word "Year".

1. Select the filter drop-down arrow on the **Year** column.
2. In the drop-down, scroll down and clear the checkbox next to the **Year** option, and then select **OK**, to remove the rows that only have the word "Year" in the **Year** column.



Now that you have cleaned up the data in the **Year** column, you can work on the **Final Winner** column. Since you are only looking at the final winners data now, you can rename this column to **Country**. To rename the column:

1. Double-click or tap and hold in the **Final Winner** column header, or
 - Right-click the **Final Winners** column header, and select **Rename** from the dropdown, or
 - Select the **Final Winners** column and select **Rename** from the **Any Column** group in the **Transform** tab of the ribbon.

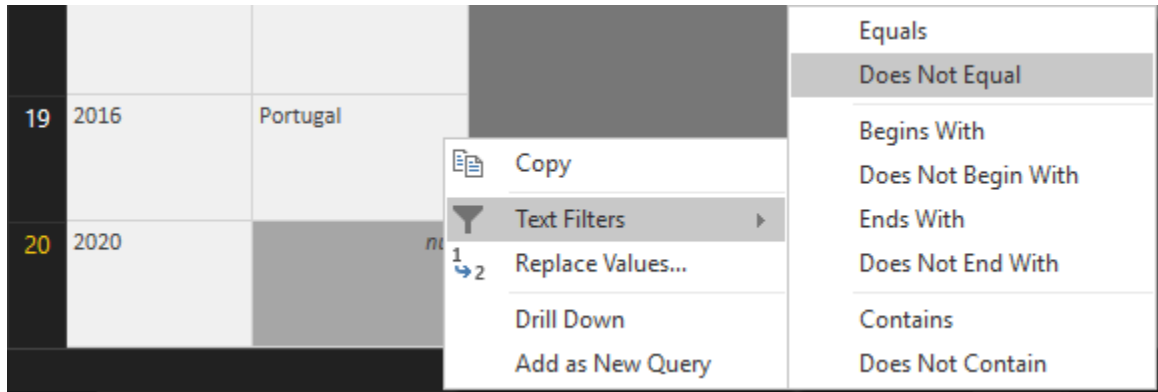
The screenshot shows a Power BI Desktop interface. A table with columns 'Year' and 'Final Winner' is displayed. A context menu is open over the 'Final Winner' column, with the 'Rename...' option highlighted. Below the table, the 'Transform' ribbon is visible, and the 'Rename' button is highlighted.

	Year	Final Winner
1	Year	Final
2	Year	Winner
3	1960	Soviet Union
	Details	
4	1964	Spain
	Details	
5	1968	Italy
	Details	
6	1972	West Germany
	Details	
7	1976	Czechoslovakia
	Details	
8	1980	West Germany
	Details	
9	Year	Final
10	Year	Winner
11	1984	France

2. Type **Country** in the header and press **Enter** to rename the column.

You also want to filter out rows like "2020" that have null values in the **Country** column. You could use the filter menu as you did with the **Year** values, or you can:

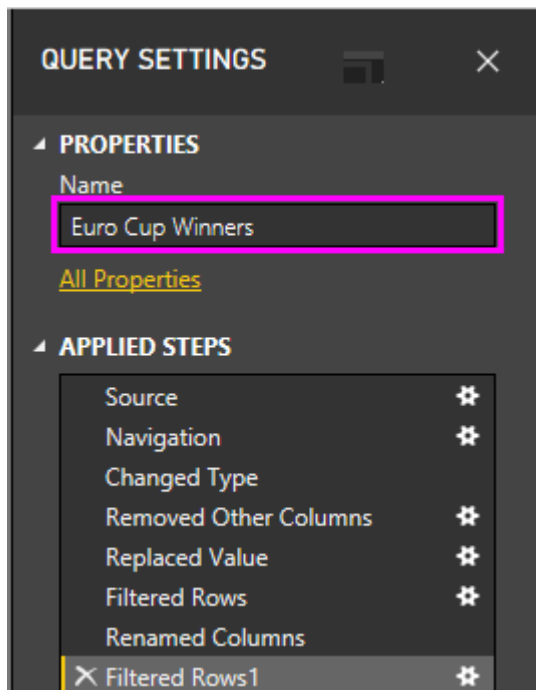
1. Right-click on the **Country** cell in the **2020** row, which has the value *null*.
2. Select **Text Filters** > **Does not Equal** in the context menu to remove any rows that contain that cell's value.



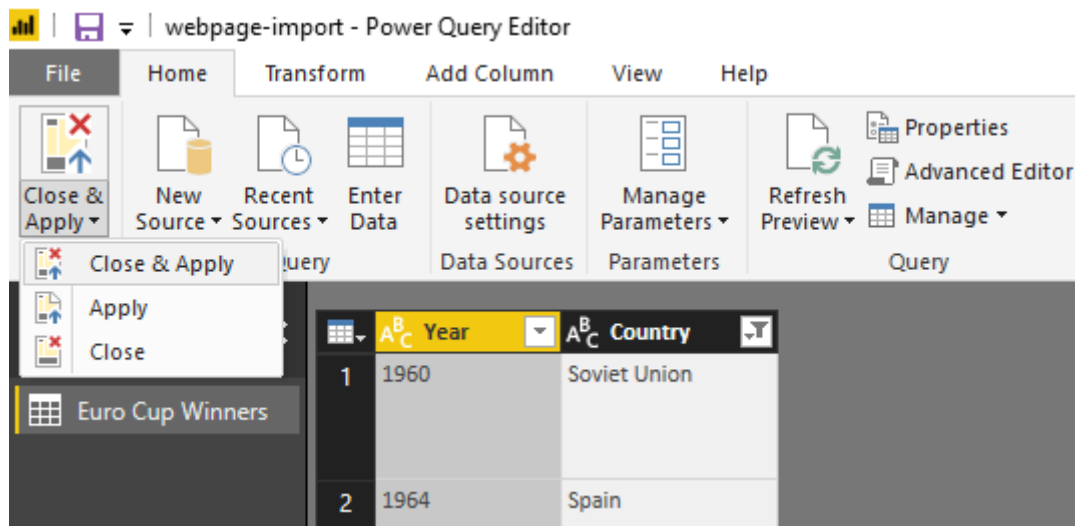
Import the query into Report View

Now that you've shaped the data the way you want, you're ready to name your query "Euro Cup Winners" and import it into your report.

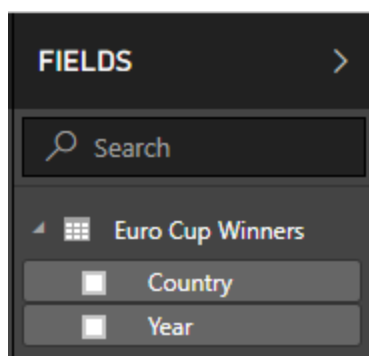
1. In the **Query Settings** pane, in the **Name** text box, type **Euro Cup Winners**, and then press **Enter**.



2. Select **Close & Apply** > **Close & Apply** from the **Home** tab of the ribbon.



The query loads into the Power BI Desktop **Report View**, where you can see it in the **Fields** pane.



Tip

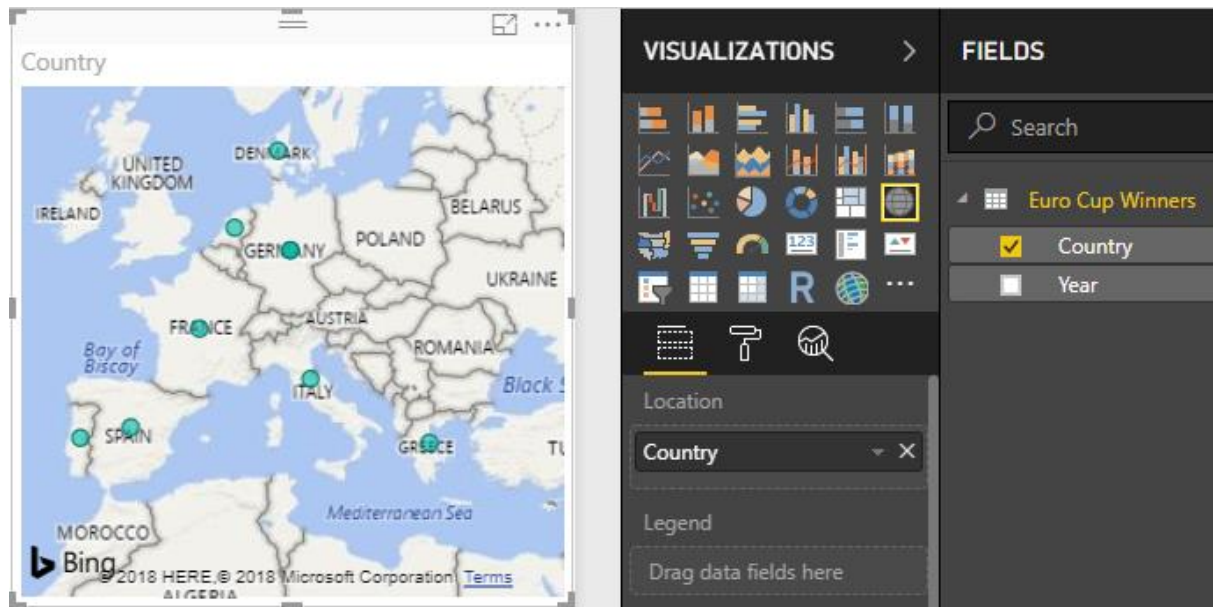
You can always get back to the **Power Query Editor** to edit and refine your query by:

- Selecting the **More options** ellipsis (...) next to **Euro Cup Winners** in the **Fields** pane, and selecting **Edit Query** from the dropdown, or
- Selecting **Edit Queries > Edit Queries** in the **External data** group of the **Home** ribbon tab in Report view.

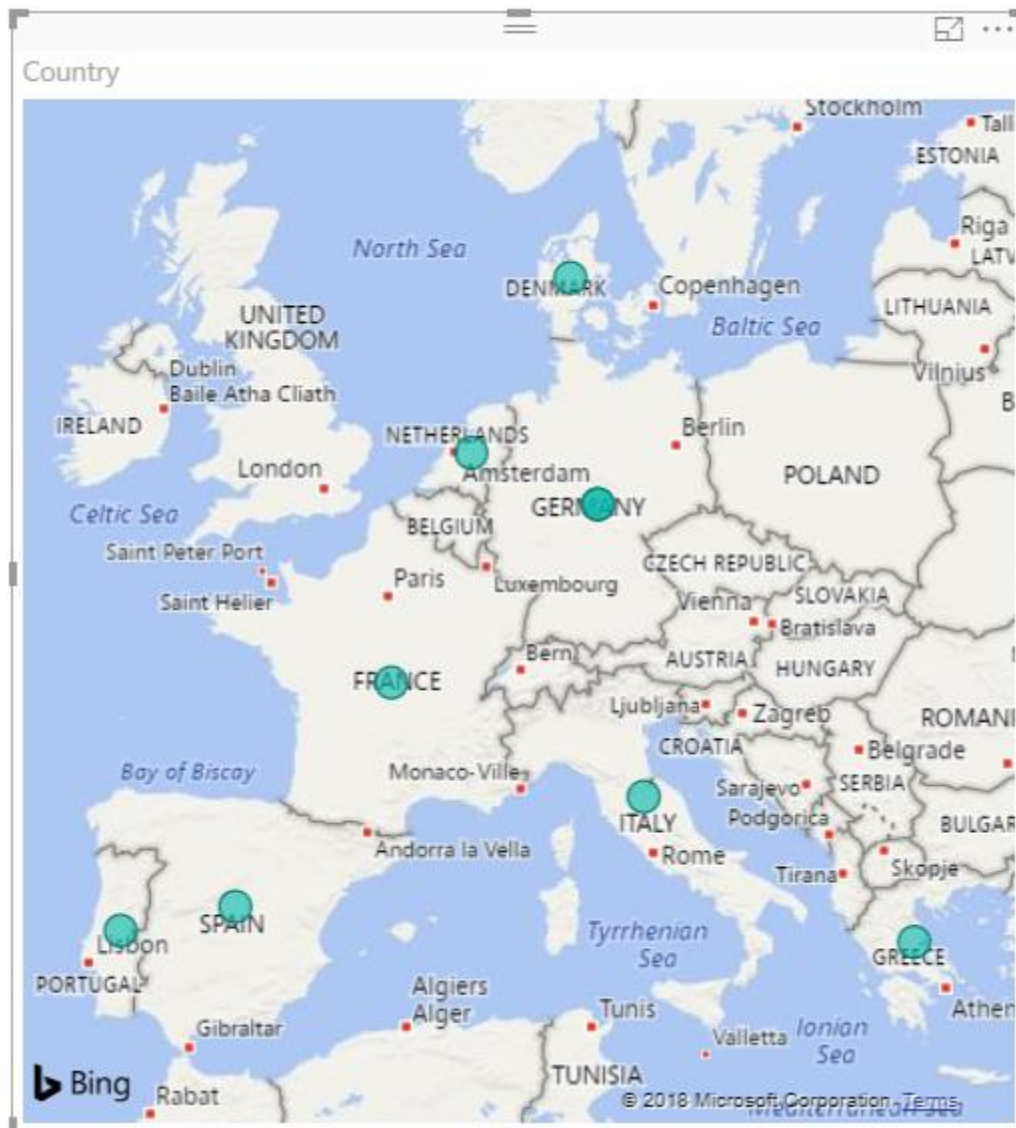
Create a visualization

To create a visualization based on your data:

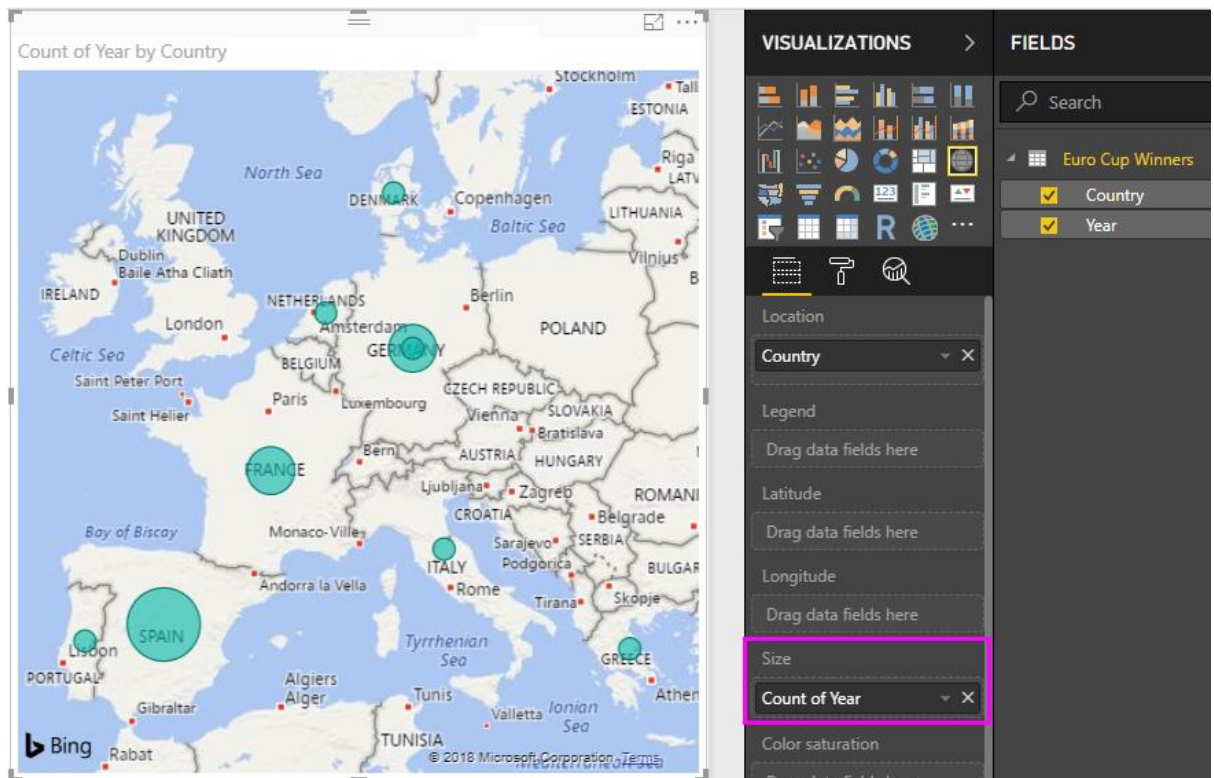
1. Select the **Country** field in the **Fields** pane, or drag it to the report canvas. Power BI Desktop recognizes the data as country names, and automatically creates a **Map** visualization.



2. Enlarge the map by dragging the handles in the corners so all the winning country names are visible.



- The map shows identical data points for every country that won a Euro Cup tournament. To make the size of each data point reflect how often the country has won, drag the **Year** field to **Drag data fields here** under **Size** in the lower part of the **Visualizations** pane. The field automatically changes to a **Count of Year** measure, and the map visualization now shows larger data points for countries that have won more tournaments.



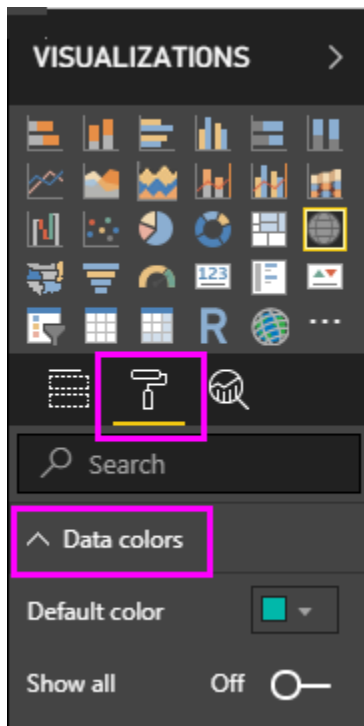
Customize the visualization

As you can see, it is very easy to create visualizations based on your data. It's also easy to customize your visualizations to better present the data in ways that you want.

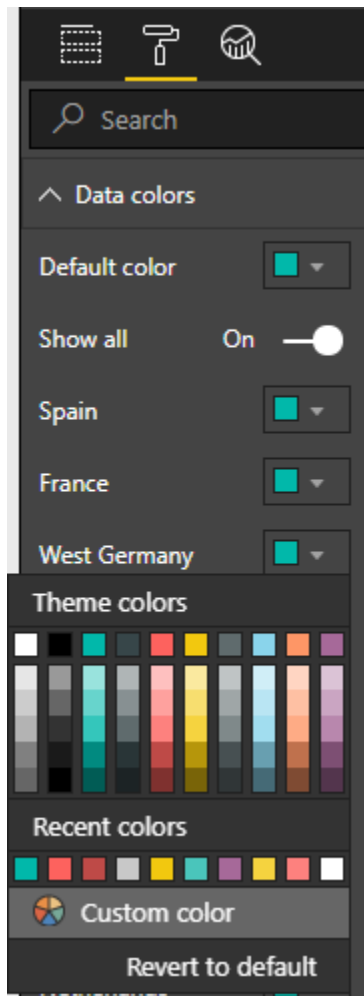
Format the map

You can change the appearance of a visualization by selecting it and then selecting the **Format** (paint roller) icon in the **Visualizations** pane. For example, the "Germany" data point(s) in your visualization could be misleading, because West Germany won two tournaments and Germany won one, and the map superimposes the two points rather than separating or adding them together. You can color these two points differently to highlight this. You can also give the map a more descriptive and attractive title.

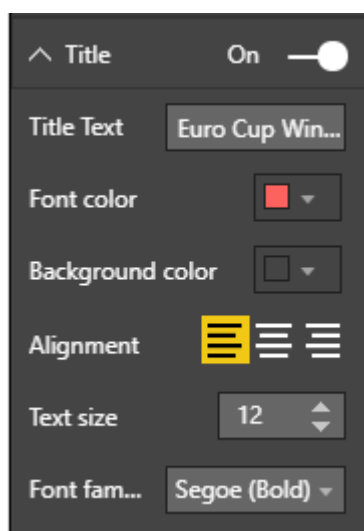
1. With the visualization selected, select the **Format** icon, and then select **Data colors** to expand the data color options.



2. Turn **Show All** to **On**, and then select the dropdown next to **West Germany** and choose a yellow color.

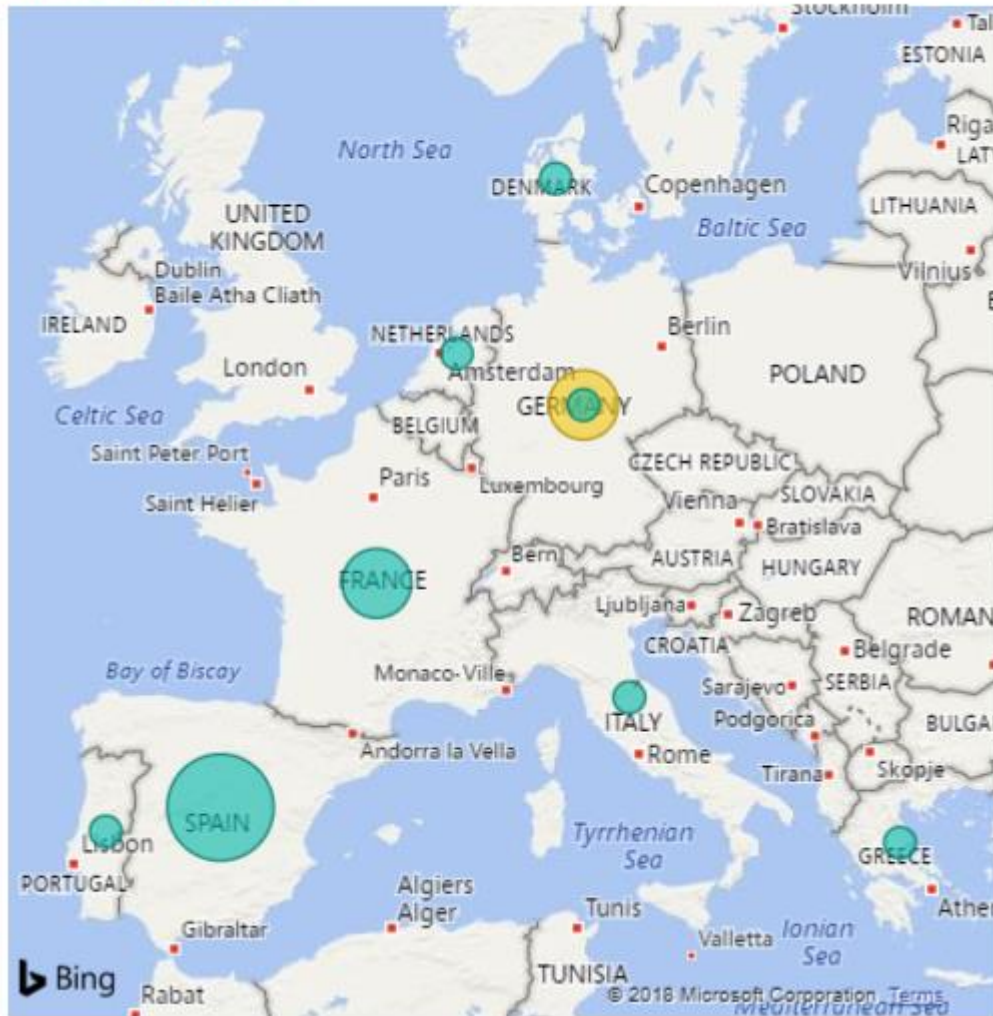


3. Select **Title** to expand the title options, and in the **Title text** field, type **Euro Cup Winners** in place of the current title.
4. Change **Font color** to red, **Text size** to **12**, and **Font family** to **Segoe (Bold)**.



Your map visualization now looks like this:

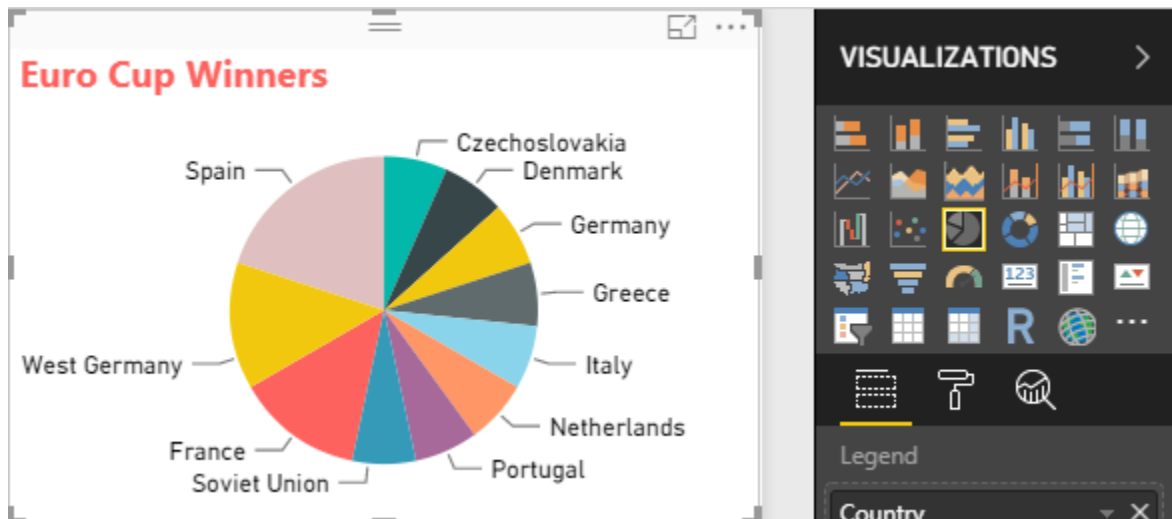
Euro Cup Winners



Change the visualization type

You can change the type of a visualization by selecting it and then selecting a different icon at the top of the **Visualization** pane. For example, your map visualization is missing the data for the Soviet Union and Czechoslovakia, because those countries no longer exist on the world map. Another type of visualization like a treemap or pie chart may be more accurate, because it shows all the values.

To change the map to a pie chart, select the map and then select the **Pie chart** icon in the **Visualization** pane.

**Tip**

- You can use the **Data colors** formatting options to make "Germany" and "West Germany" the same color.
- To group the countries with the most wins together on the pie chart, select the ellipsis (...) at the upper right of the visualization, and then select **Sort by Count of Year** from the dropdown.

Power BI Desktop provides a seamless end-to-end experience, from getting data from a wide range of data sources and shaping it to meet your analysis needs, to visualizing this data in rich and interactive ways. Once your report is ready, you can [upload it to Power BI](#) and create dashboards based on it, which you can share with other Power BI users.

Tutorial: Combine sales data from Excel and an OData feed

It's common to have data in multiple data sources. For example, you could have two databases, one for product information, and another for sales information. With **Power BI Desktop**, you can combine data from different sources to create interesting, compelling data analyses and visualizations.

In this tutorial, you combine data from two data sources:

1. An Excel workbook with product information
2. An OData feed containing orders data

You're going to import each dataset and do transformation and aggregation operations. Then, you'll use the two source's data to produce a sales analysis report with interactive visualizations. Later, you can apply these techniques to SQL Server queries, CSV files, and other data sources in Power BI Desktop.

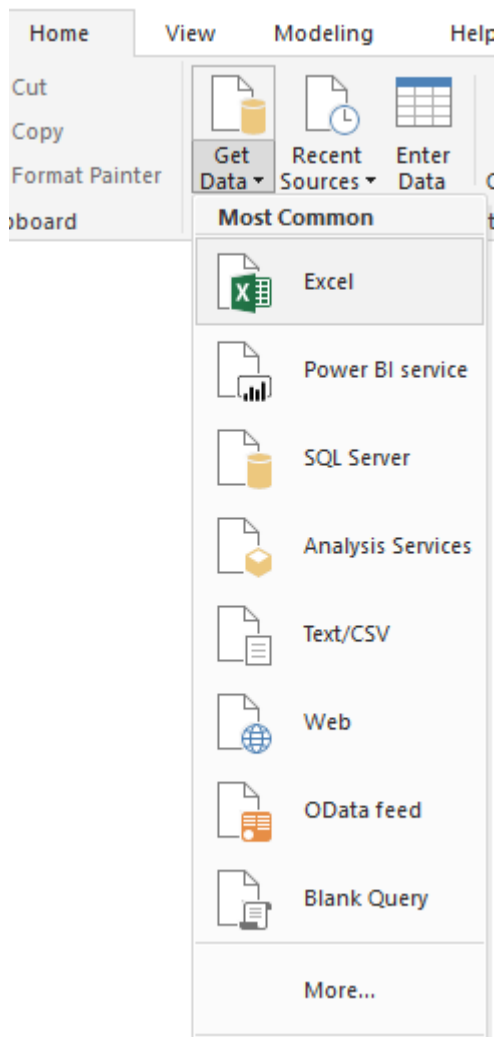
Note

In Power BI Desktop, there are often a few ways to accomplish a task. For example, you can right-click or use a **More options** menu on a column or cell to see additional ribbon selections. Several alternate methods are described in the steps below.

Import Excel product data

First, import the Products.xlsx Excel workbook's product data into Power BI Desktop.

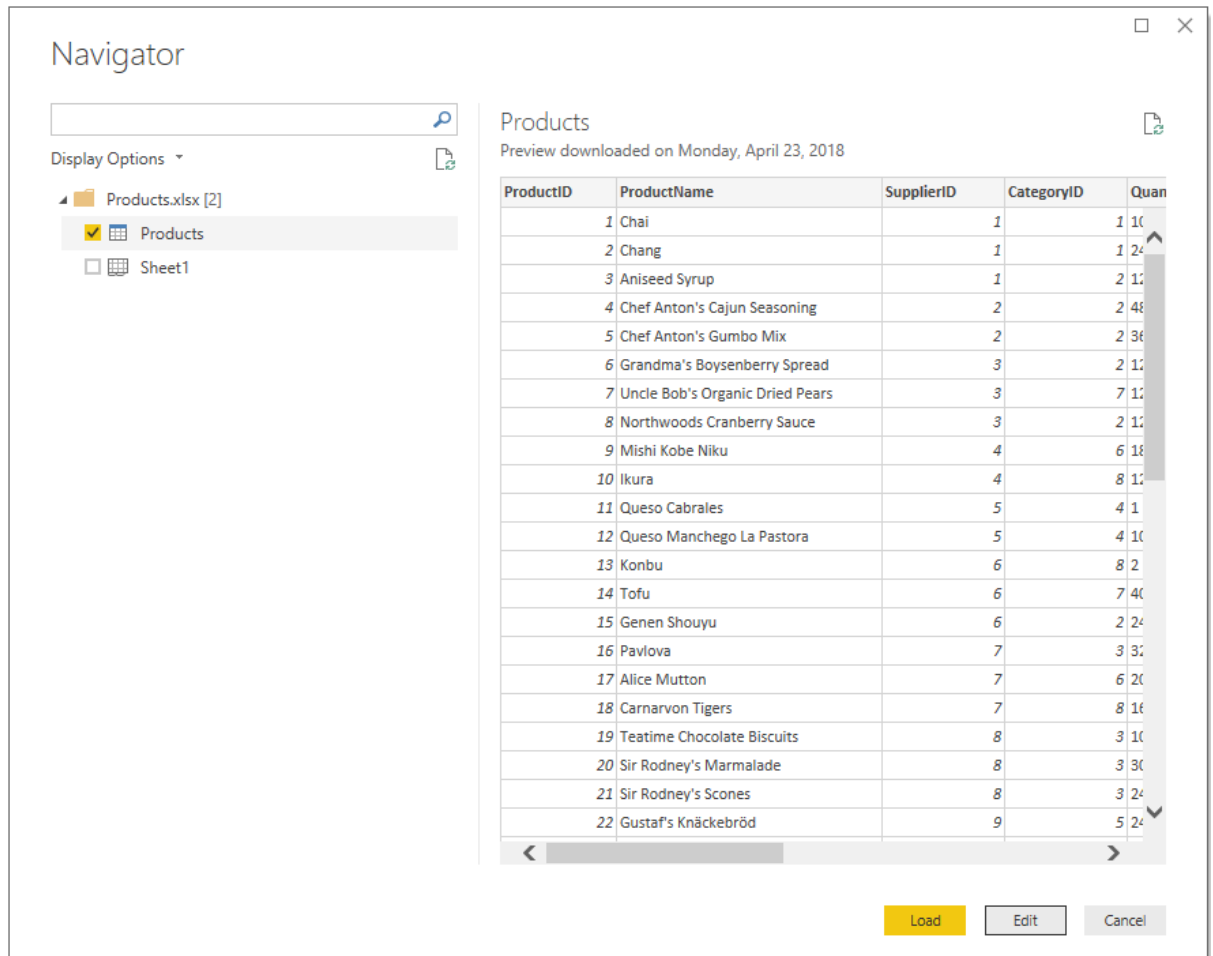
1. [Download the Products.xlsx Excel workbook](#) and save it as **Products.xlsx**.
2. Select the dropdown arrow next to **Get Data** in the Power BI Desktop ribbon's **Home** tab, and then, select **Excel** from the **Most Common** dropdown.



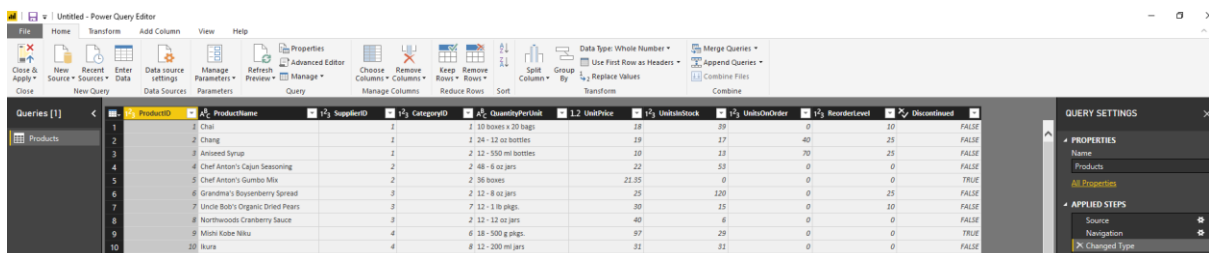
Note

You can also select the **Get Data** item itself, or select **Get Data** from the Power BI **Get started** dialog, then select **Excel** or **File > Excel** in the **Get Data** dialog box, and then select **Connect**.

3. In the **Open** dialog box, navigate to and select the **Products.xlsx** file, and then select **Open**.
4. In the **Navigator** pane, select the **Products** table and then select **Edit**.



A table preview opens in the **Power Query Editor**, where you can apply transformations to clean up the data.



Note

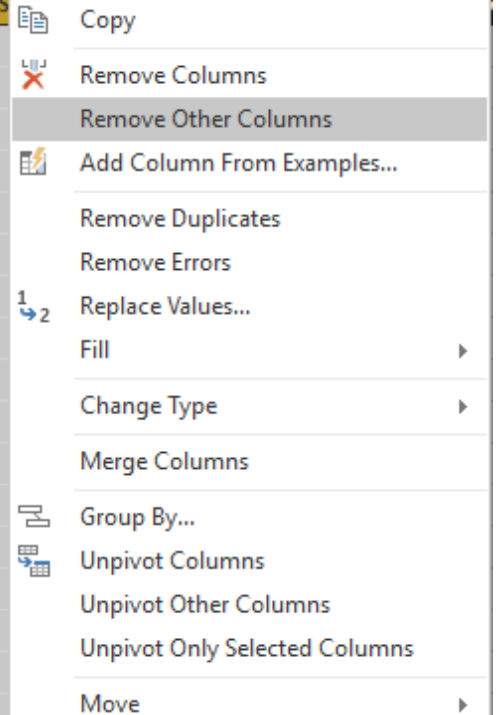
You can also open the **Power Query Editor** by selecting **Edit Queries > Edit Queries** from the **Home** ribbon in Power BI Desktop, or by right-clicking or choosing **More options** next to any query in **Report View**, and selecting **Edit Query**.

Clean up the products columns

Your combined report will use the Excel workbook's **ProductID**, **ProductName**, **QuantityPerUnit**, and **UnitsInStock** columns. You can remove the other columns.

1. In **Power Query Editor**, select the **ProductID**, **ProductName**, **QuantityPerUnit**, and **UnitsInStock** columns. You can use **Ctrl+Click** to select more than one column, or **Shift+Click** to select columns next to each other.
2. Right-click any of the selected headers. Select **Remove Other Columns** from the dropdown. You can also select **Remove Columns > Remove Other Columns** from the **Manage Columns** group in the **Home** ribbon tab.

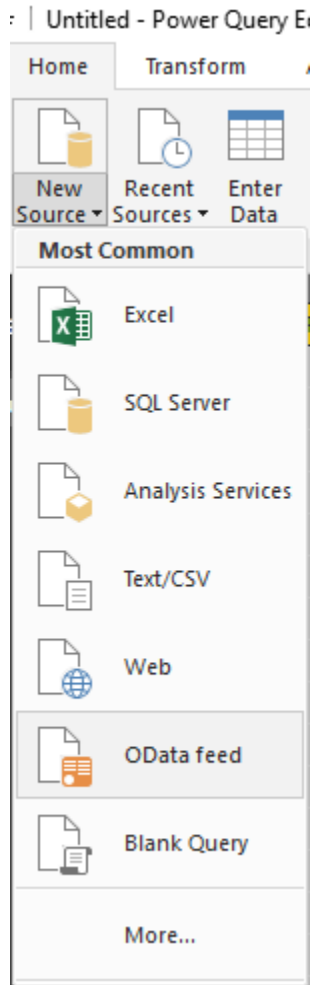
	A ^B C QuantityPerUnit	1.2 UnitPrice	1 ² 3 UnitsInS
1	10 boxes x 20 bags		18
1	24 - 12 oz bottles		19
2	12 - 550 ml bottles		10
2	48 - 6 oz jars		22
2	36 boxes	21.35	
2	12 - 8 oz jars	25	
7	12 - 1 lb pkgs.	30	
2	12 - 12 oz jars	40	
6	18 - 500 g pkgs.	97	
8	12 - 200 ml jars	31	
4	1 kg pkg.	21	
4	10 - 500 g pkgs.	38	
8	2 kg box	6	
7	40 - 100 g pkgs.	23.25	
2	24 - 250 ml bottles	15.5	
3	32 - 500 g boxes	17.45	
6	20 - 1 kg tins	39	



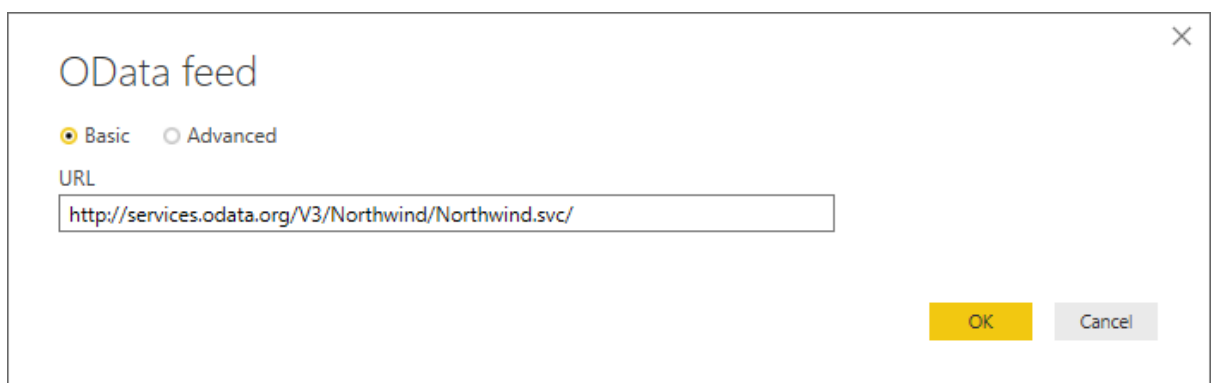
Import the OData feed's order data

Next, import the order data from the sample Northwind sales system OData feed.

1. In **Power Query Editor**, select **New Source** and then, from the **Most Common** dropdown, select **OData feed**.



2. In the **OData feed** dialog box, paste the Northwind OData feed URL, <https://services.odata.org/V3/Northwind/Northwind.svc/>. Select **OK**.



3. In the **Navigator** pane, select the **Orders** table, and then select **OK** to load the data into **Power Query Editor**.

Navigator

Display Options

<http://services.odata.org/V3/Northwind/No...>

- ☐ Alphabetical_list_of_products
- ☐ Categories
- ☐ Category_Sales_for_1997
- ☐ Current_Product_Lists
- ☐ Customer_and_Suppliers_by_Cities
- ☐ CustomerDemographics
- ☐ Customers
- ☐ Employees
- ☐ Invoices
- ☐ Order_Details
- ☐ Order_Details_Extendeds
- ☐ Order_Subtotals
- ☒ **Orders**
- ☐ Orders_Qries
- ☐ Product_Sales_for_1997
- ☐ Products
- ☐ Products_Above_Average_Prices
- ☐ Products_by_Categories
- ☐ Regions

Select Related Tables

Orders

OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate
10248	VINET	5	7/4/1996 12:00:00 AM	8/1/199
10249	TOMSP	6	7/5/1996 12:00:00 AM	8/16/199
10250	HANAR	4	7/8/1996 12:00:00 AM	8/5/199
10251	VICTE	3	7/8/1996 12:00:00 AM	8/5/199
10252	SUPRD	4	7/9/1996 12:00:00 AM	8/6/199
10253	HANAR	3	7/10/1996 12:00:00 AM	7/24/199
10254	CHOPS	5	7/11/1996 12:00:00 AM	8/8/199
10255	RICSU	9	7/12/1996 12:00:00 AM	8/9/199
10256	WELLI	3	7/15/1996 12:00:00 AM	8/12/199
10257	HILAA	4	7/16/1996 12:00:00 AM	8/13/199
10258	ERNSH	1	7/17/1996 12:00:00 AM	8/14/199
10259	CENTC	4	7/18/1996 12:00:00 AM	8/15/199
10260	OTTIK	4	7/19/1996 12:00:00 AM	8/16/199
10261	QUEDE	4	7/19/1996 12:00:00 AM	8/16/199
10262	RATTC	8	7/22/1996 12:00:00 AM	8/19/199
10263	ERNSH	9	7/23/1996 12:00:00 AM	8/20/199
10264	FOLKO	6	7/24/1996 12:00:00 AM	8/21/199
10265	BLONP	2	7/25/1996 12:00:00 AM	8/22/199
10266	WARTH	3	7/26/1996 12:00:00 AM	9/6/199
10267	FRANK	4	7/29/1996 12:00:00 AM	8/26/199
10268	GROSR	8	7/30/1996 12:00:00 AM	8/27/199
10269	WHITC	5	7/31/1996 12:00:00 AM	8/14/199
10270	WARTH	1	8/1/1996 12:00:00 AM	8/29/199

OK
Cancel

Note

In **Navigator**, you can select any table name, without selecting the checkbox, to see a preview.

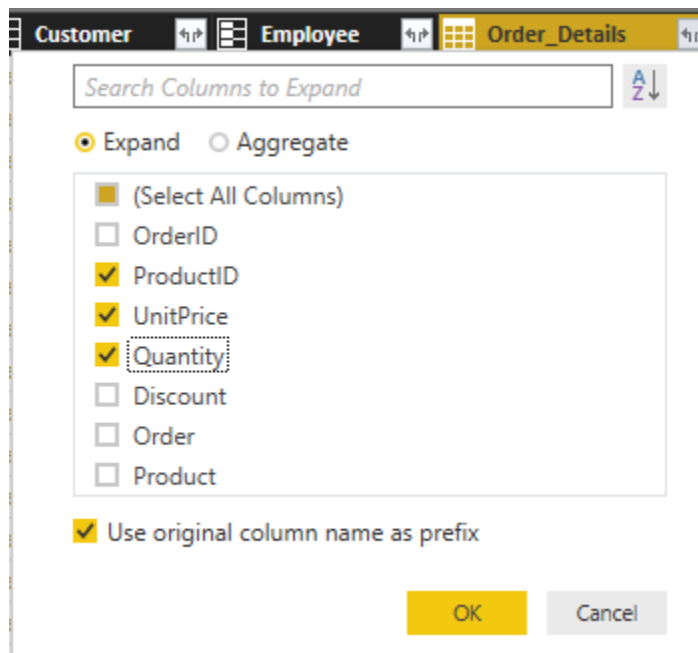
Expand the order data

You can use table references to build queries when connecting to data sources with multiple tables, such as relational databases or the Northwind OData feed. The **Orders** table contains references to several related tables. You can use the **Expand** operation to add the **ProductID**, **UnitPrice**, and **Quantity** columns from the related **Order_Details** table into the subject (**Orders**) table.

1. Scroll to the right in the **Orders** table until you see the **Order_Details** column. It contains references to another table and not data.

Order_Details
Table
Table
Table
Table
Table
Table
Table
Table

2. Select the **Expand** icon () in the **Order_Details** column header.
3. In the **Expand** drop-down:
 1. Select (**Select All Columns**) to clear all columns.
 2. Select **ProductID**, **UnitPrice**, and **Quantity**, and then select **OK**.



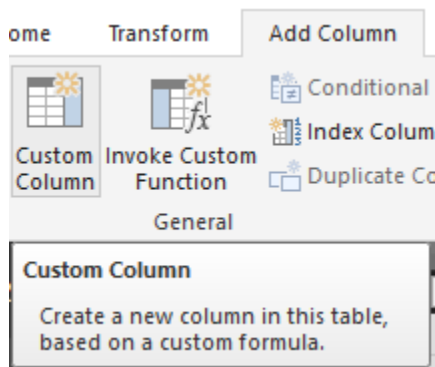
After you expand the **Order_Details** table, three new nested table columns replace the **Order_Details** column. There are new rows in the table for each order's added data.

A ^B ShipCountry	Customer	Employee	1 ² Order_Details.ProductID	1.2 Order_Details.UnitPrice	1.2 Order_Details.Quantity
France	Record	Record	11	14	12
France	Record	Record	42	9.8	10
France	Record	Record	72	34.8	5
Germany	Record	Record	14	18.6	9
Germany	Record	Record	51	42.4	40
Brazil	Record	Record	41	7.7	10
Brazil	Record	Record	51	42.4	35
Brazil	Record	Record	65	16.8	15
France	Record	Record	22	16.8	6
France	Record	Record	57	15.6	15
France	Record	Record	65	16.8	20

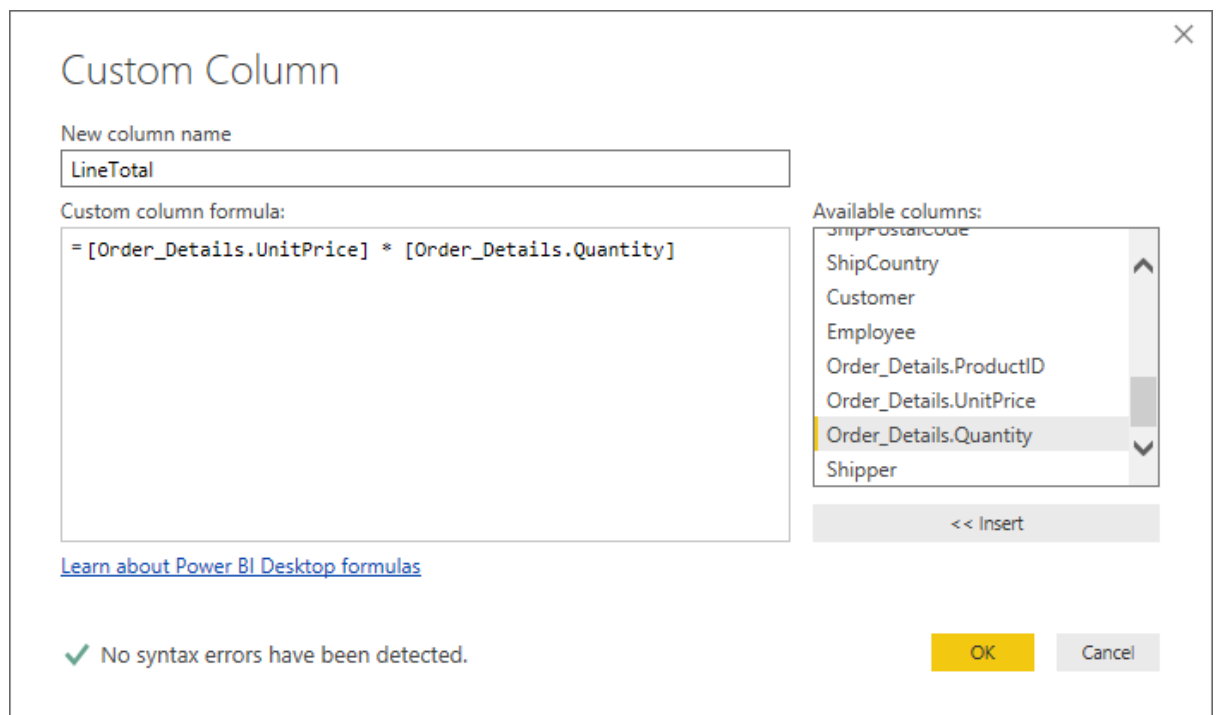
Create a custom calculated column

Power Query Editor lets you create calculations and custom fields to enrich your data. You'll create a custom column that multiplies the unit price by item quantity to calculate the total price for each order's line item.

1. In the Power Query Editor's **Add Column** ribbon tab, select **Custom Column**.



2. In the **Custom Column** dialog box, type **LineTotal** in the **New column name** field.
3. In the **Custom column formula** field after the **=**, enter **[Order_Details.UnitPrice] * [Order_Details.Quantity]**. (You can also select the field names from the **Available columns** scroll box and select **<< Insert**, instead of typing them.)
4. Select **OK**.

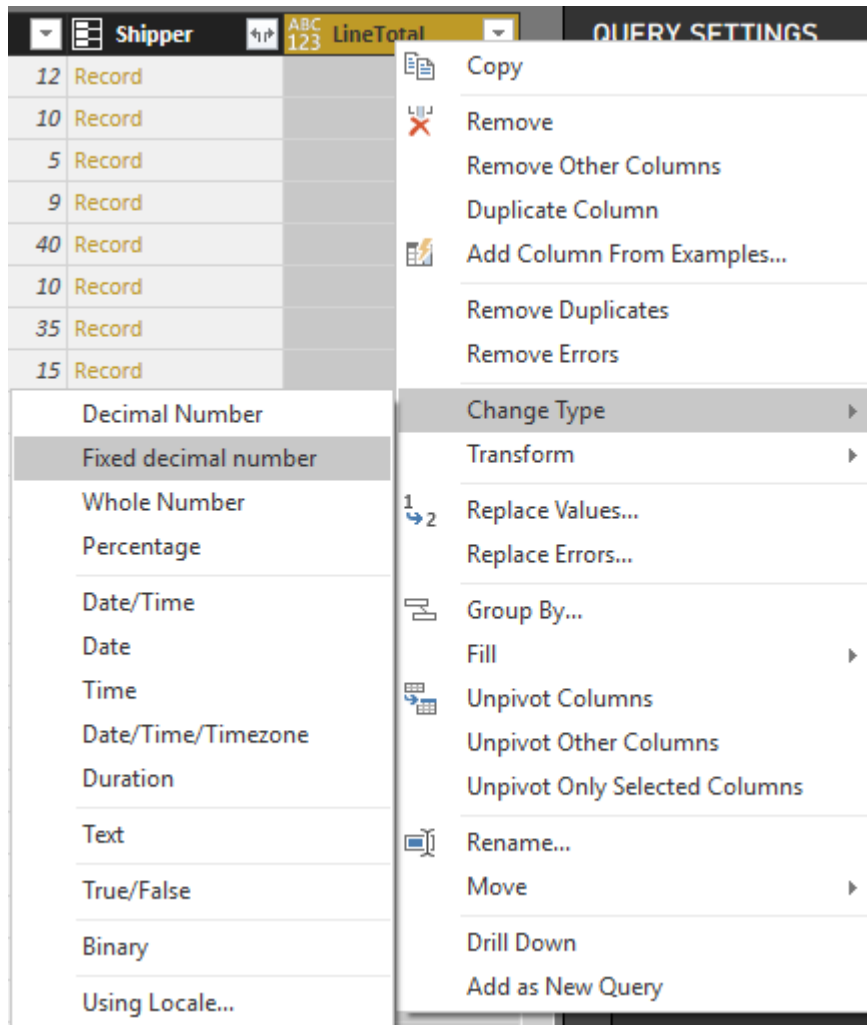


The new **LineTotal** field appears as the last column in the **Orders** table.

Set the new field's data type

When Power Query Editor connects to data, it makes a best guess as to each field's data type for display purposes. A header icon indicates each field's assigned data type. You can also look under **Data Type** in the **Home** ribbon tab's **Transform** group.

Your new **LineTotal** column has an **Any** data type, but it has currency values. To assign a data type, right-click the **LineTotal** column header, select **Change Type** from the dropdown, and then select **Fixed decimal number**.



Note

You can also select the **LineTotal** column, then select the dropdown arrow next to **Data Type** in the **Transform** area of the **Home** ribbon tab, and then select **Fixed decimal number**.

Clean up the orders columns

To make your model easier to work with in reports, you can delete, rename, and reorder some columns.

Your report is going to use the following columns:

- **OrderDate**
- **ShipCity**
- **ShipCountry**
- **Order_Details.ProductID**
- **Order_Details.UnitPrice**
- **Order_Details.Quantity**
- **LineTotal**

Select these columns and use **Remove Other Columns** as you did with the Excel data. Or, you can select the non-listed columns, right-click on one of them, and select **Remove Columns**.

You can rename the columns prefixed with "**Order_Details.**" to make them easier to read:

1. Double-click or tap and hold each column header, or right-click the column header, and select **Rename** from the dropdown.
2. Delete the **Order_Details.** prefix from each name, and then press **Enter**.

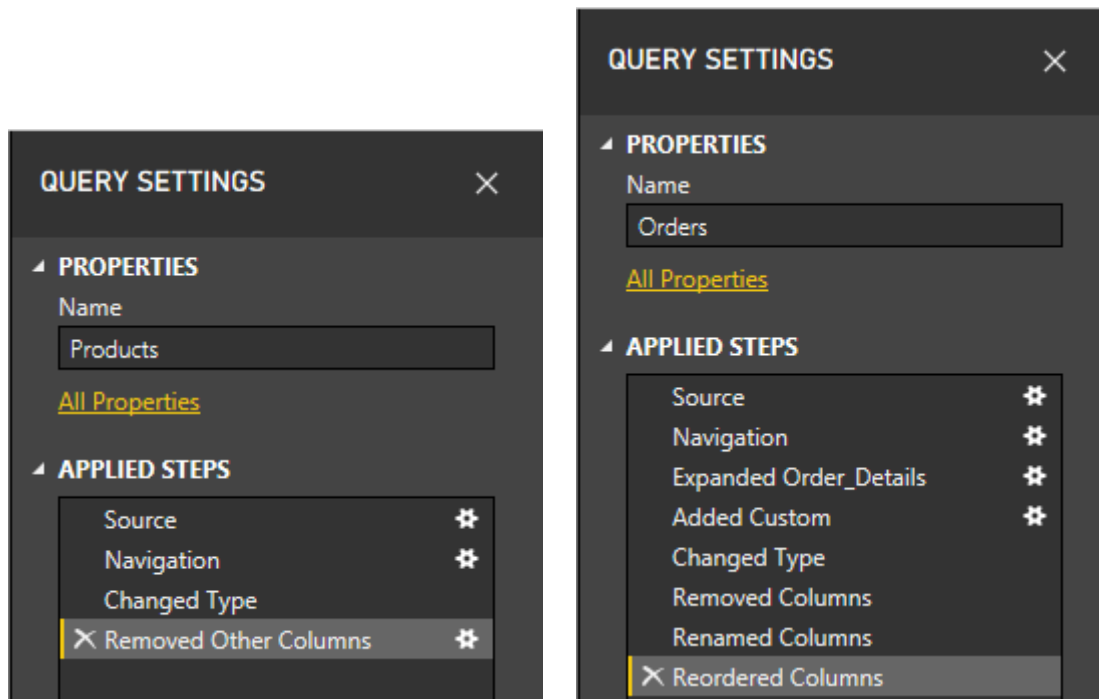
Finally, to make the **LineTotal** column easier to access, drag and drop it to the left, just to the right of the **ShipCountry** column.

	OrderDate	ShipCity	ShipCountry	LineTotal	ProductID	UnitPrice	Quantity
1	7/4/1996 12:00:00 AM	Reims	France	168	11	14	12
2	7/4/1996 12:00:00 AM	Reims	France	98	42	9.8	10
3	7/4/1996 12:00:00 AM	Reims	France	174	72	34.8	5
4	7/5/1996 12:00:00 AM	Münster	Germany	167.4	14	18.6	9
5	7/5/1996 12:00:00 AM	Münster	Germany	1696	51	42.4	40
6	7/8/1996 12:00:00 AM	Rio de Janeiro	Brazil	77	41	7.7	10
7	7/8/1996 12:00:00 AM	Rio de Janeiro	Brazil	1484	51	42.4	35
8	7/8/1996 12:00:00 AM	Rio de Janeiro	Brazil	252	65	16.8	15
9	7/8/1996 12:00:00 AM	Lyon	France	100.8	22	16.8	6
10	7/8/1996 12:00:00 AM	Lyon	France	234	57	15.6	15
11	7/8/1996 12:00:00 AM	Lyon	France	336	65	16.8	20

Review the query steps

Your Power Query Editor actions to shape and transform data are recorded. Each action appears on the right in the **Query Settings** pane under **Applied Steps**. You can step back through the **Applied Steps** to review your steps, and edit, delete, or rearrange them if necessary. However, changing preceding steps is risky as that can break later steps.

Select each of your queries in the **Queries** list on the left side of Power Query Editor, and review the **Applied Steps** in **Query Settings**. After applying the previous data transformations, the **Applied Steps** for your two queries should look like this:

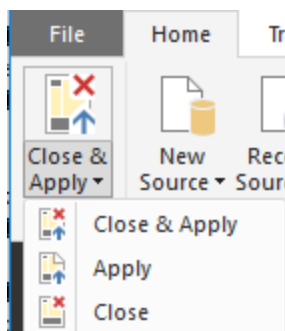


Tip

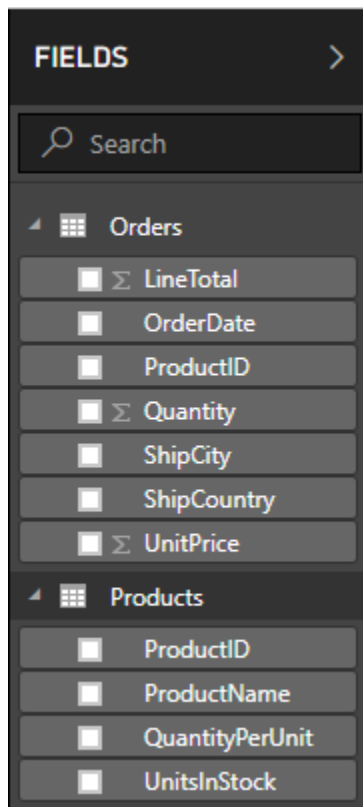
Underlying the Applied Steps are formulas written in the **Power Query Language**, also known as the [M language](#). To see and edit the formulas, select **Advanced Editor** in the **Query** group of the Home tab of the ribbon.

Import the transformed queries

When you're satisfied with your transformed data and ready to import it into Power BI Desktop Report View, select **Close & Apply > Close & Apply** in the **Home** ribbon tab's **Close** group.



Once the data is loaded, the queries appear in the **Fields** list in the Power BI Desktop Report View.

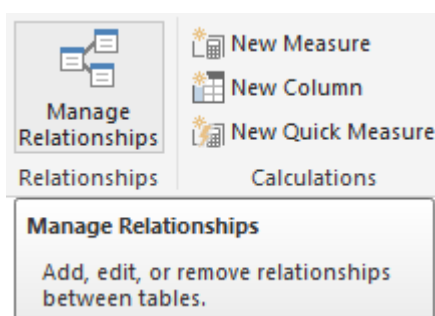


Manage the relationship between the datasets

Power BI Desktop doesn't require you to combine queries to report on them. However, you can use the relationships between datasets, based on common fields, to extend, and enrich your reports. Power BI Desktop may detect relationships automatically, or you can create them in the Power BI Desktop **Manage Relationships** dialog. For more information, see [Create and manage relationships in Power BI Desktop](#).

The shared **ProductID** field creates a relationship between this tutorial's Orders and Products datasets.

1. In Power BI Desktop Report View, select **Manage Relationships** in the **Home** ribbon tab's **Relationships** area.



2. In the **Manage relationships** dialog, you can see that Power BI Desktop has already detected and listed an active relationship between the Products and Orders tables. To view the relationship, select **Edit**.

Manage relationships

Active	From: Table (Column)	To: Table (Column)
<input checked="" type="checkbox"/>	Orders (ProductID)	Products (ProductID)

New... Autodetect... Edit... Delete

Close

The **Edit Relationship** dialog opens, showing details about the relationship.

×

Edit relationship

Select tables and columns that are related.

Orders

OrderDate	ShipCity	ShipCountry	LineTotal	ProductID	UnitPrice	Quantity
10/8/1996 12:00:00 AM	Boise	USA	\$291.9	16	13.9	21
10/8/1996 12:00:00 AM	Boise	USA	\$1,008	35	14.4	70
10/8/1996 12:00:00 AM	Boise	USA	\$288	46	9.6	30

Products

ProductID	ProductName	QuantityPerUnit	UnitsInStock
1	Chai	10 boxes x 20 bags	39
2	Chang	24 - 12 oz bottles	17
3	Aniseed Syrup	12 - 550 ml bottles	13

Cardinality

Many to one (*:1)

Cross filter direction

Single

☒ Make this relationship active
 ☐ Apply security filter in both directions

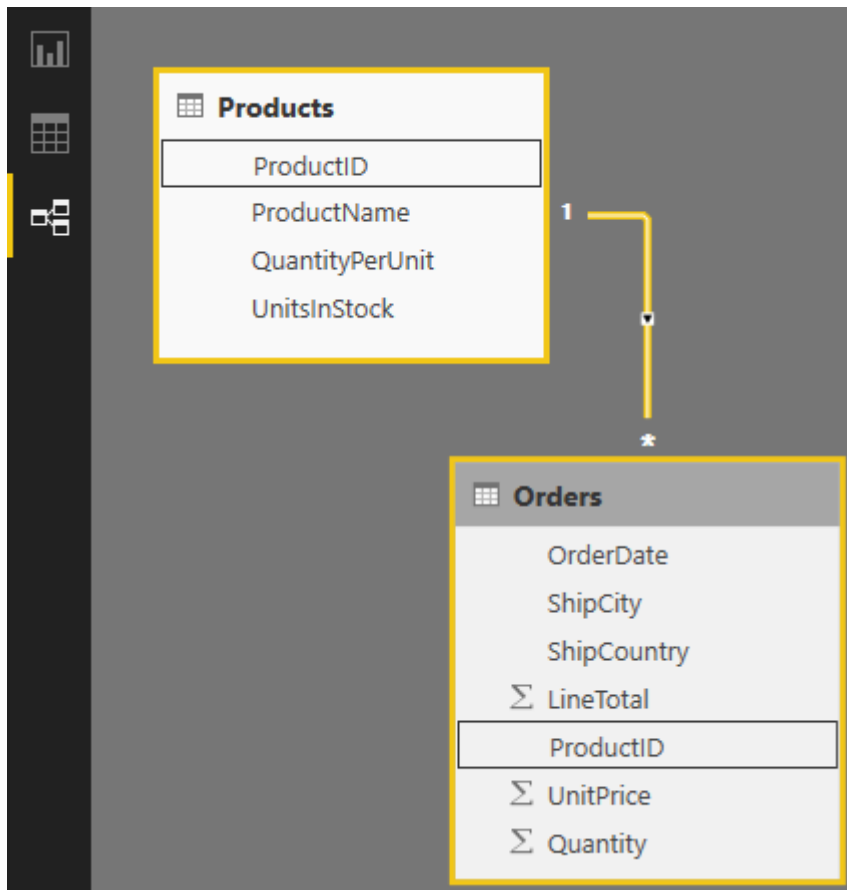
☐ Assume referential integrity

OK

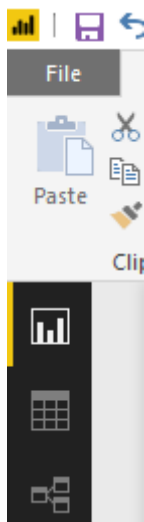
Cancel

- Power BI Desktop has autodetected the relationship correctly, so you can select **Cancel** and then **Close**.

In Power BI Desktop, on the left side, select **Model** to view and manage query relationships. Double-click the arrow on the line connecting the two queries to open the **Edit relationship** dialog and view or change the relationship.



To get back to Report View from Relationships View, select the **Report** icon.



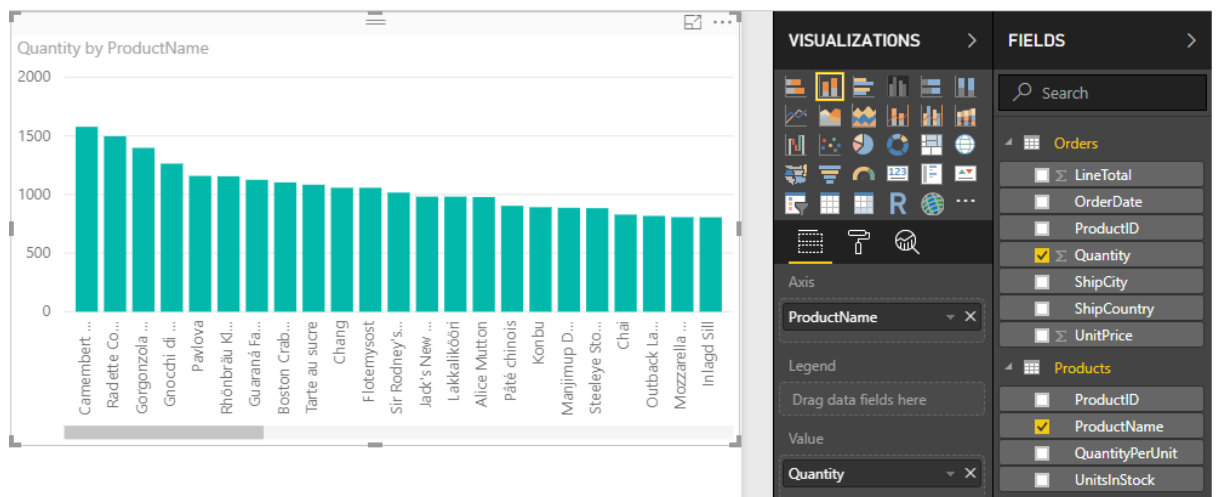
Create visualizations using your data

You can create different visualizations in Power BI Desktop Review View to gain data insights. Reports can have multiple pages, and each page can have multiple visuals. You and others can interact with your visualizations to help analyze and understand data. For more information, see [Interact with a report in Editing view in Power BI service](#).

You can use both of your data sets, and the relationship between them, to help visualize and analyze your sales data.

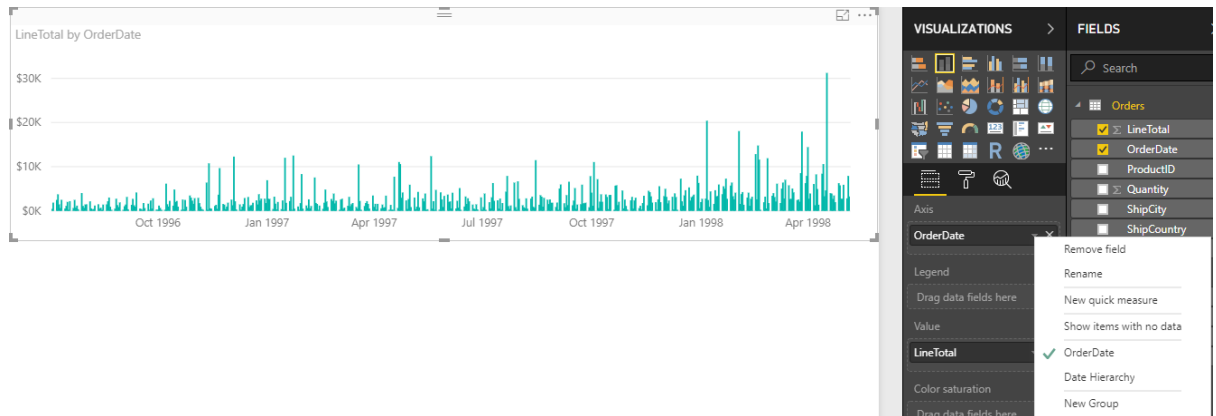
First, create a stacked column chart that uses fields from both queries to show the quantity of each product ordered.

1. Select the **Quantity** field from **Orders** in the **Fields** pane at the right, or drag it onto a blank space on the canvas. A stacked column chart is created showing the total quantity of all products ordered.
2. To show the quantity of each product ordered, select **ProductName** from **Products** in the **Fields** pane, or drag it onto the chart.
3. To sort the products by most to least ordered, select the **More options** ellipsis (...) at the visualization's upper right, and then select **Sort By Quantity**.
4. Use the handles at the corners of the chart to enlarge it so more product names are visible.



Next, create a chart showing order dollar amounts (**LineTotal**) over time (**OrderDate**).

1. With nothing selected on the canvas, select **LineTotal** from **Orders** in the **Fields** pane, or drag it to a blank space on the canvas. The stacked column chart shows the total dollar amount of all orders.
2. Select the stacked chart, then select **OrderDate** from **Orders**, or drag it onto the chart. The chart now shows line totals for each order date.
3. Drag the corners to resize the visualization and see more data.

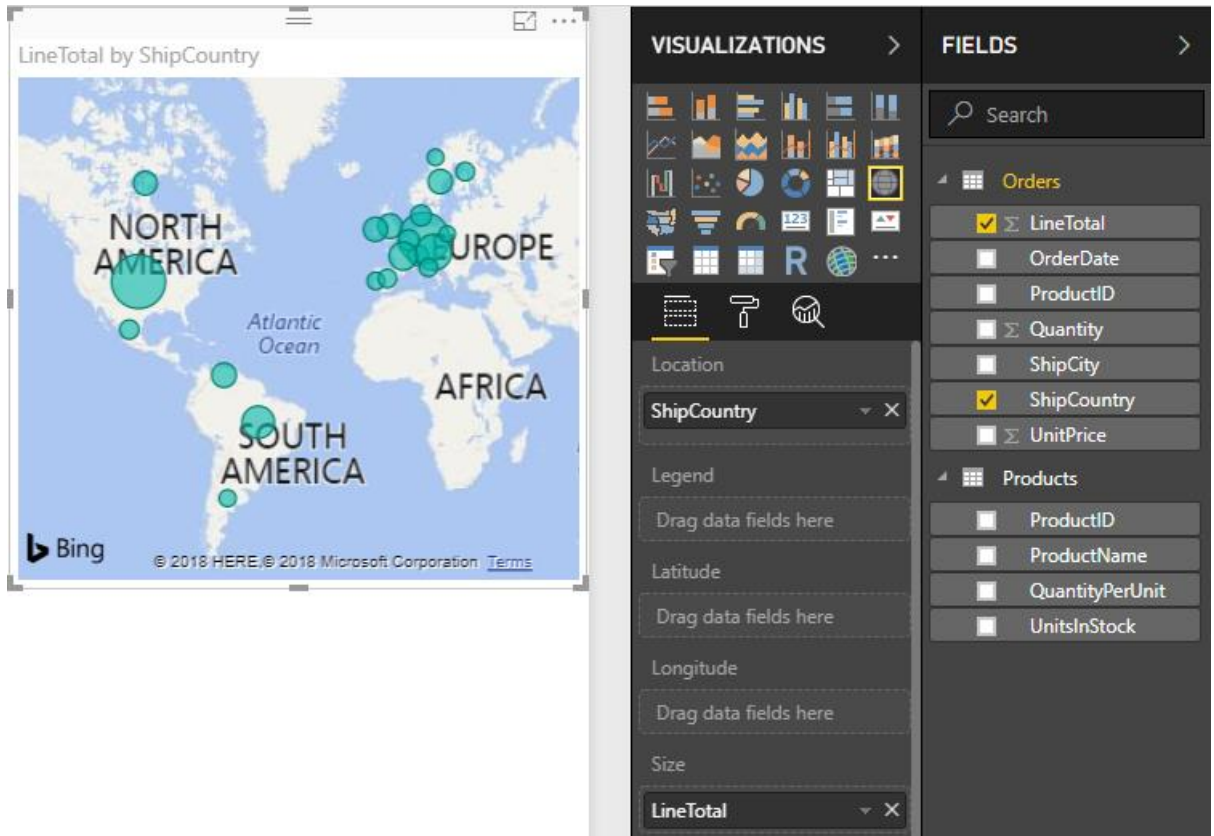


Tip

If you only see Years on the chart (only three data points), drop down the arrow next to **OrderDate** in the **Axis** field of the **Visualizations** pane, and select **OrderDate** instead of **Date Hierarchy**.

Finally, create a map visualization showing order amounts from each country.

1. With nothing selected on the canvas, select **ShipCountry** from **Orders** in the **Fields** pane, or drag it to a blank space on the canvas. Power BI Desktop detects that the data is country names. It then automatically creates a map visualization, with a data point for each country with orders.
2. To make the data point sizes reflect each country's order amounts, drag the **LineTotal** field onto the map. You can also drag it to **Drag data fields here** under **Size** in the **Visualizations** pane. The sizes of the circles on the map now reflect the dollar amounts of the orders from each country.

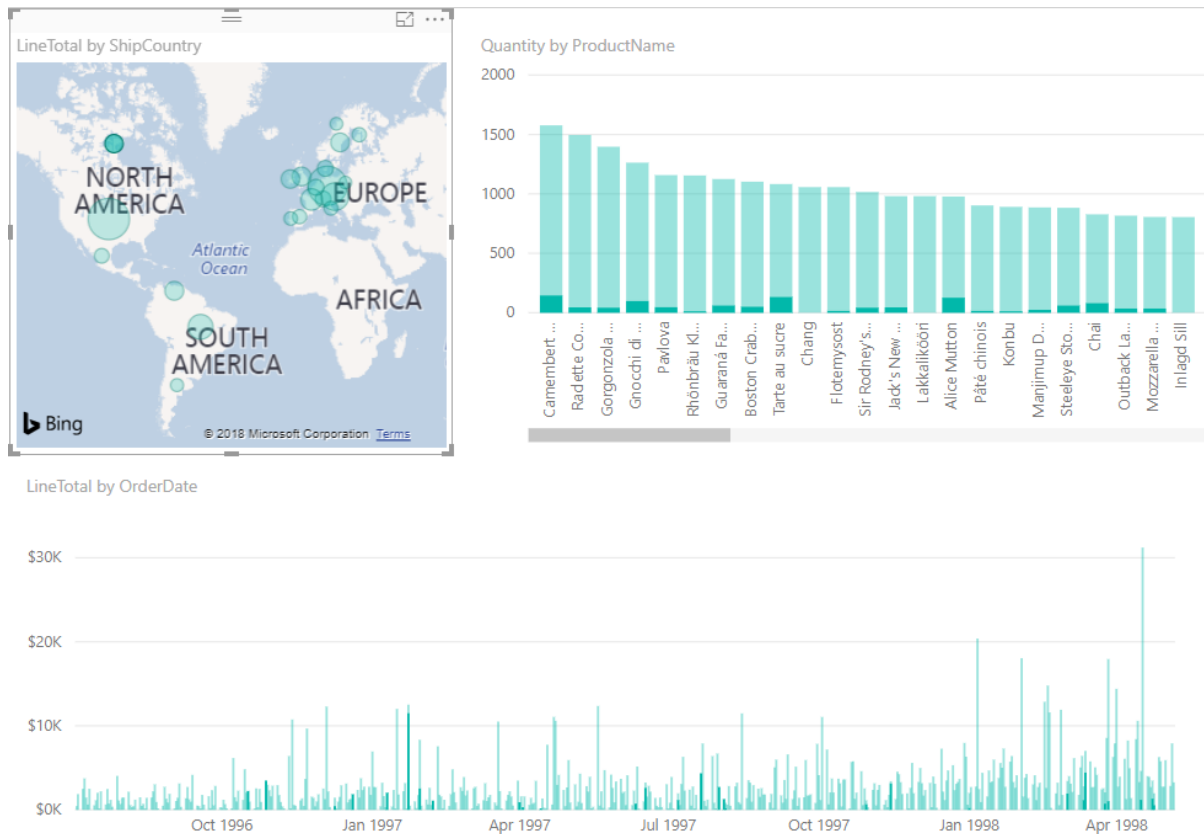


Interact with your report visuals to analyze further

In Power BI Desktop, you can interact with visuals that cross-highlight and filter each other to uncover further trends. For more information, see [Filtering and Highlighting in Power BI Reports](#).

Because of the relationship between your queries, interactions with one visualization affect all the other visualizations on the page.

On the map visualization, select the circle centered in **Canada**. The other two visualizations filter to highlight the Canadian line totals and order quantities.



Select a **Quantity by ProductName** chart product to see the map and the date chart filter to reflect that product's data. Select a **LineTotal by OrderDate** chart date to see the map and the product chart filter to show that date's data.

Tip

To deselect a selection, select it again, or select one of the other visualizations.

Complete the sales analysis report

Your completed report combines data from the Products.xlsx Excel file and the Northwind OData feed in visuals that help you analyze different countries' order information, timeframes, and products. When your report is ready, you can [upload it to Power BI service](#) to share it with other Power BI users.

Tutorial: Create your own measures in Power BI Desktop

By using measures, you can create some of the most powerful data analysis solutions in Power BI Desktop. Measures help you by performing calculations on your data as you interact with your reports. This tutorial will guide you through understanding measures and creating your own basic measures in Power BI Desktop.

Prerequisites

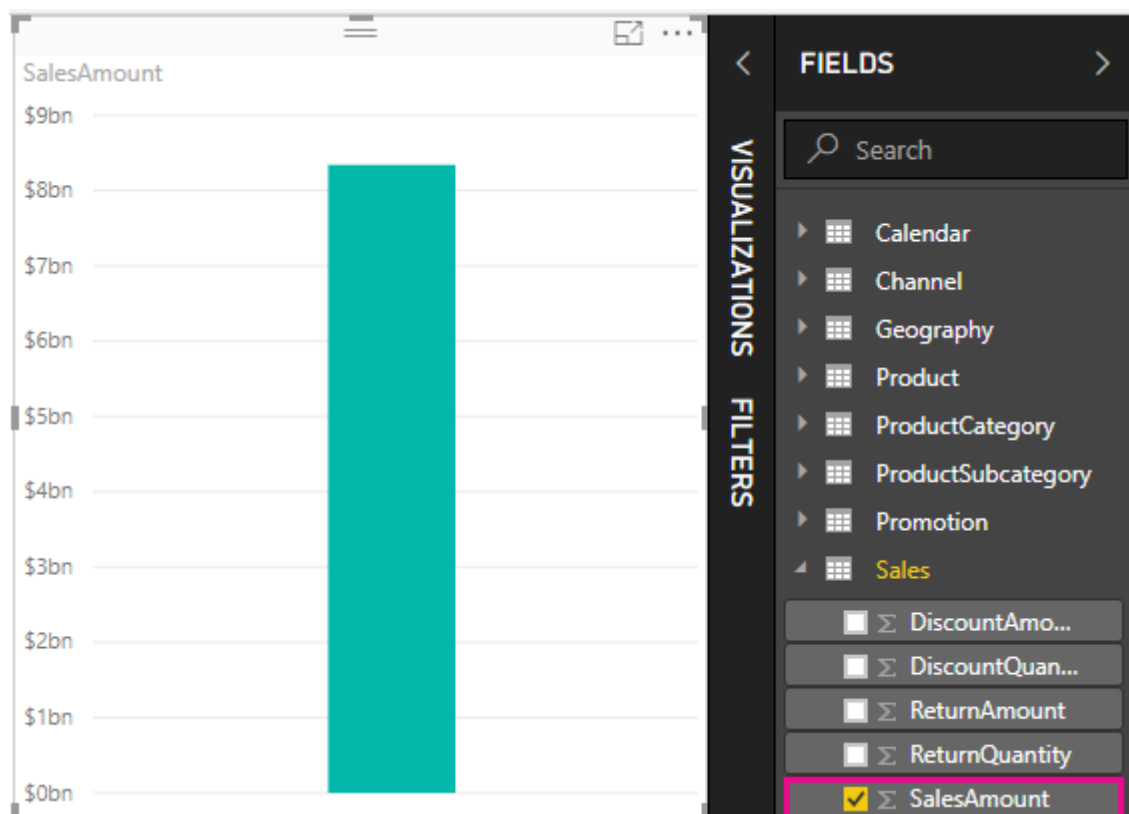
- This tutorial is intended for Power BI users already familiar with using Power BI Desktop to create more advanced models. You should already be familiar with using Get Data and Query Editor to import data, work with multiple related tables, and add fields to the report canvas. If you're new to Power BI Desktop, be sure to check out [Getting Started with Power BI Desktop](#).
- This tutorial uses the [Contoso Sales Sample for Power BI Desktop](#) file, which includes online sales data from the fictitious company, Contoso. Because this data is imported from a database, you can't connect to the datasource or view it in Query Editor. Download and extract the file on your computer.

Automatic measures

When Power BI Desktop creates a measure, it's most often created for you automatically. To see how Power BI Desktop creates a measure, follow these steps:

1. In Power BI Desktop, select **File > Open**, browse to the *Contoso Sales Sample for Power BI Desktop.pbix* file, and then select **Open**.
2. In the **Fields** pane, expand the **Sales** table. Then, either select the check box next to the **SalesAmount** field or drag **SalesAmount** onto the report canvas.

A new column chart visualization appears, showing the sum total of all values in the **SalesAmount** column of the **Sales** table.

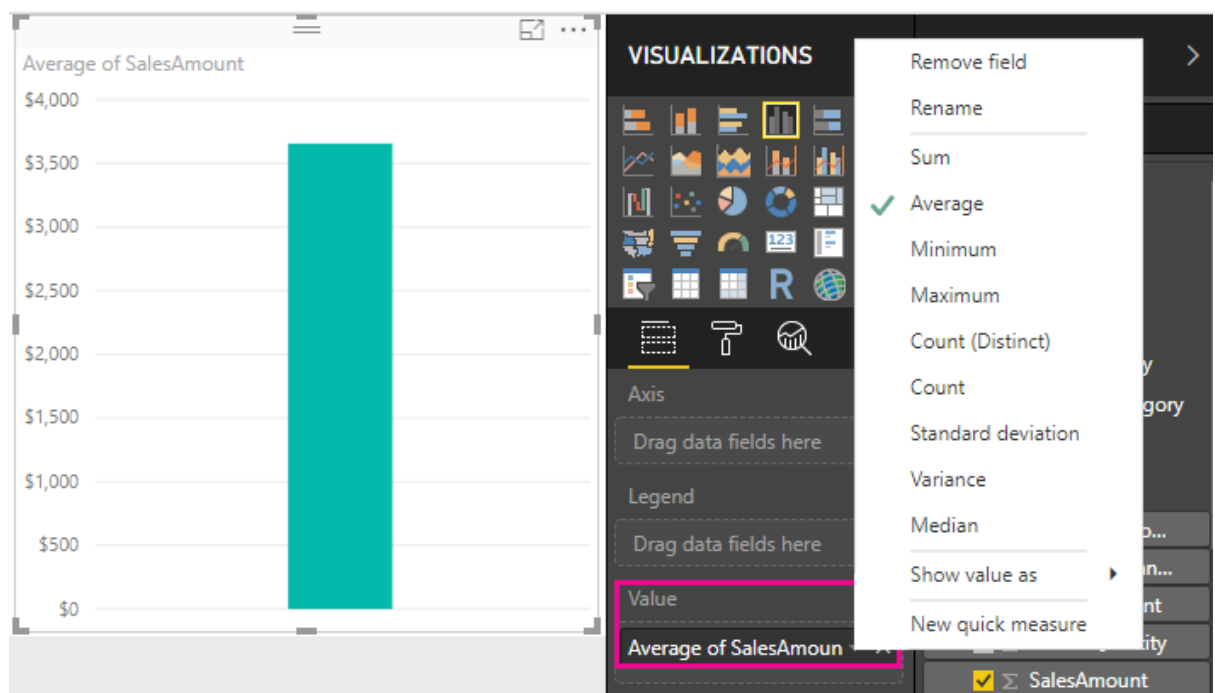


Any field (column) in the **Fields** pane with a sigma icon Σ is numeric, and its values can be aggregated. Rather than display a table with many values (two million rows for **SalesAmount**), Power BI Desktop automatically creates and calculates a measure to aggregate the data if it detects a numeric datatype. Sum is the default aggregation for a numeric datatype, but you can easily apply different aggregations like average or count. Understanding aggregations is fundamental to understanding measures, because every measure performs some type of aggregation.

To change the chart aggregation, follow these steps:

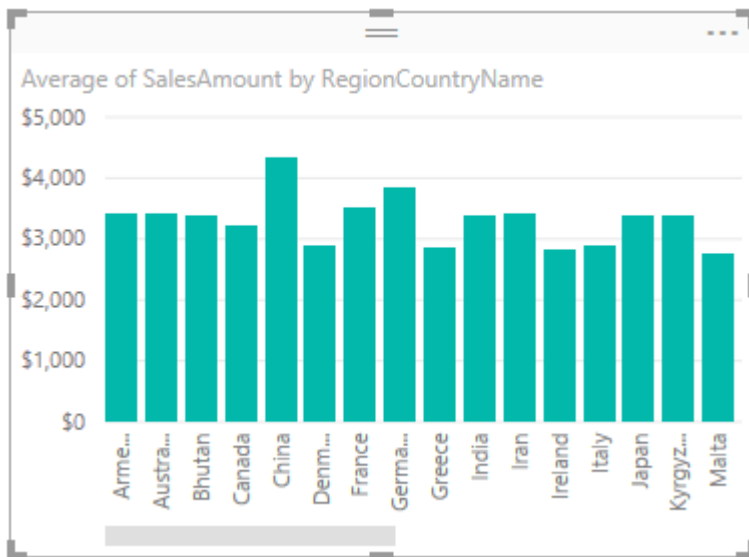
1. Select the **SalesAmount** visualization in the report canvas.
2. In the **Value** area of the **Visualizations** pane, select the down arrow to the right of **SalesAmount**.
3. From the menu that appears, select **Average**.

The visualization changes to an average of all sales values in the **SalesAmount** field.



Depending on the result you want, you can change the type of aggregation. However, not all types of aggregation apply to every numeric datatype. For example, for the **SalesAmount** field, Sum and Average are useful, and Minimum and Maximum have their place as well. However, Count doesn't make sense for the **SalesAmount** field, because while its values are numeric, they're really currency.

Values calculated from measures change in response to your interactions with your report. For example, if you drag the **RegionCountryName** field from the **Geography** table onto your existing **SalesAmount** chart, it changes to show the average sales amounts for each country.



When the result of a measure changes because of an interaction with your report, you've affected your measure's *context*. Every time you interact with your report visualizations, you're changing the context in which a measure calculates and displays its results.

Create and use your own measures

In most cases, Power BI Desktop automatically calculates and returns values according to the types of fields and aggregations you choose. However, in some cases you might want to create your own measures to perform more complex, unique calculations. With Power BI Desktop, you can create your own measures with the Data Analysis Expressions (DAX) formula language.

DAX formulas use many of the same functions, operators, and syntax as Excel formulas. However, DAX functions are designed to work with relational data and perform more dynamic calculations as you interact with your reports. There are over 200 DAX functions that do everything from simple aggregations like sum and average to more complex statistical and filtering functions. There are many resources to help you learn more about DAX. After you've finished this tutorial, see [DAX basics in Power BI Desktop](#).

When you create your own measure, it's called a *model* measure, and it's added to the **Fields** list for the table you select. Some advantages of model measures are that you can name them whatever you want, making them more identifiable; you can use them as arguments in other DAX expressions; and you can make them perform complex calculations quickly.

Quick measures

Starting with the February 2018 release of Power BI Desktop, many common calculations are available as *quick measures*, which write the DAX formulas for you based on your inputs in a window. These quick, powerful calculations are also great for learning DAX or seeding your own customized measures.

Create a quick measure using one of these methods:

- From a table in the **Fields** pane, right-click or select **More options (...)**, and then select **New quick measure** from the list.
- Under **Calculations** in the **Home** tab of the Power BI Desktop ribbon, select **New Quick Measure**.

For more information about creating and using quick measures, see [Use quick measures](#).

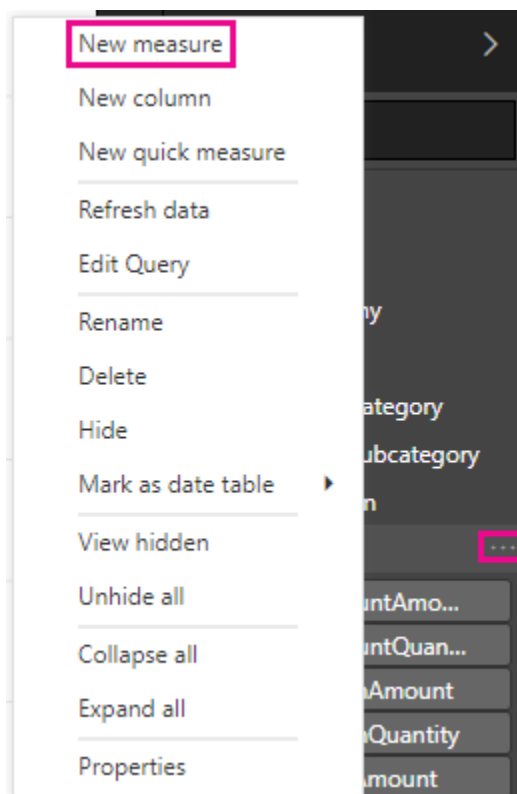
Create a measure

Suppose you want to analyze your net sales by subtracting discounts and returns from total sales amounts. For the context that exists in your visualization, you need a measure that subtracts the sum of DiscountAmount and ReturnAmount from the sum of SalesAmount. There's no field for Net Sales in the **Fields** list, but you have the building blocks to create your own measure to calculate net sales.

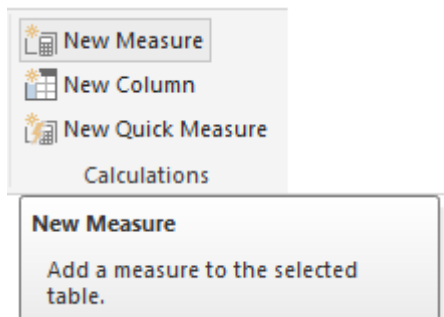
To create a measure, follow these steps:

1. In the **Fields** pane, right-click the **Sales** table, or hover over the table and select **More options (...)**.
2. From the menu that appears, select **New measure**.

This action saves your new measure in the **Sales** table, where it's easy to find.



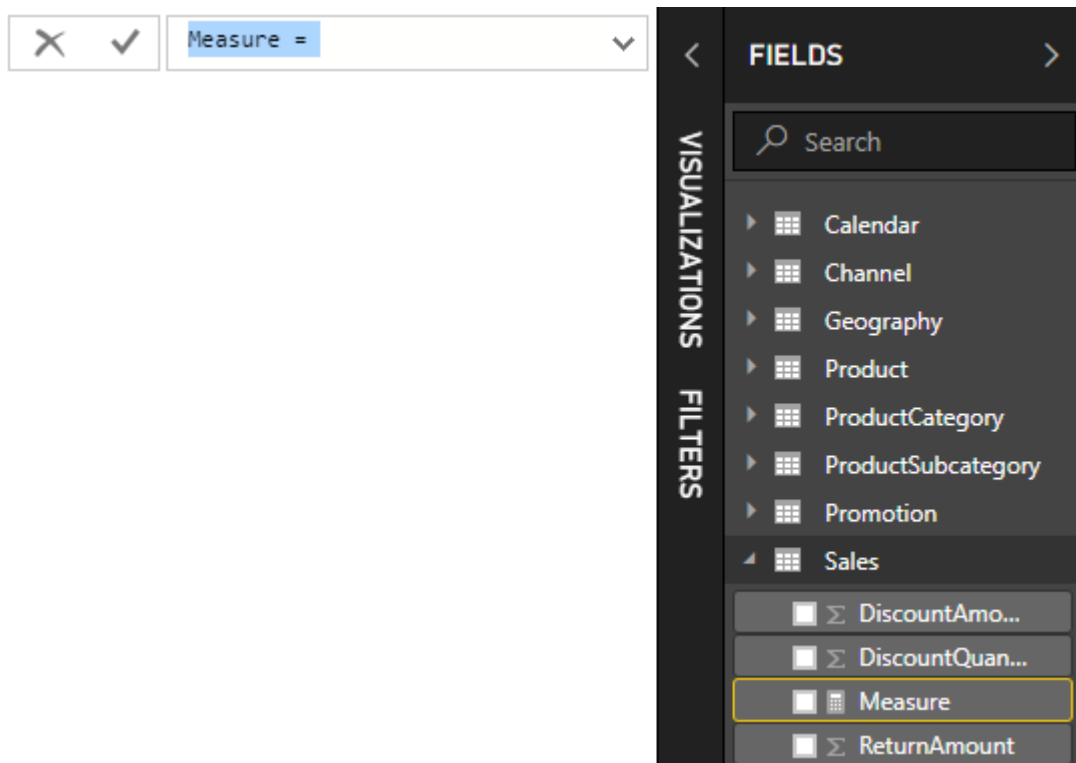
You can also create a new measure by selecting **New Measure** in the **Calculations** group on the **Home** tab of the Power BI Desktop ribbon.



Tip

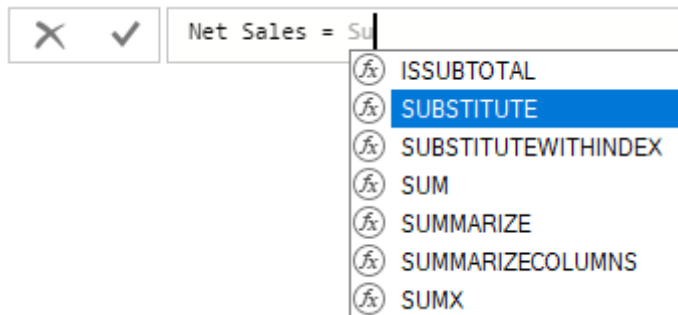
When you create a measure from the ribbon, you can create it in any of your tables, but it's easier to find if you create it where you plan to use it. In this case, select the **Sales** table first to make it active, and then select **New measure**.

The formula bar appears along the top of the report canvas, where you can rename your measure and enter a DAX formula.

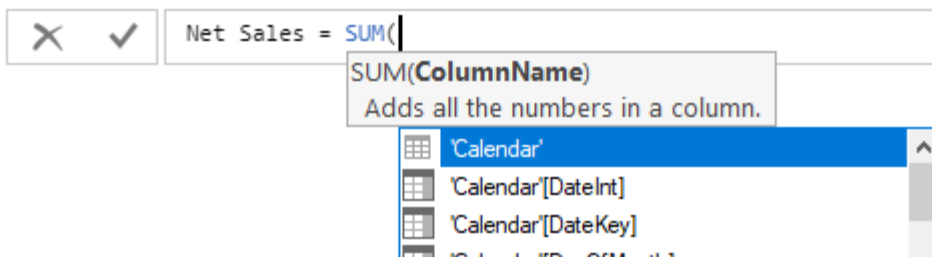


- By default, each new measure is named *Measure*. If you don't rename it, additional new measures are named *Measure 2*, *Measure 3*, and so on. Because we want this measure to be more identifiable, highlight *Measure* in the formula bar, and then change it to *Net Sales*.

4. Begin entering your formula. After the equals sign, start to type *Sum*. As you type, a drop-down suggestion list appears, showing all the DAX functions, beginning with the letters you type. Scroll down, if necessary, to select **SUM** from the list, and then press **Enter**.

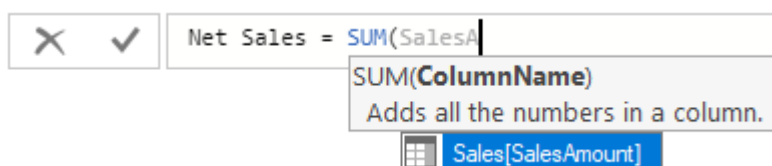


An opening parenthesis appears, along with a drop-down suggestion list of the available columns you can pass to the SUM function.



5. Expressions always appear between opening and closing parentheses. For this example, your expression contains a single argument to pass to the SUM function: the **SalesAmount** column. Begin typing *SalesAmount* until **Sales(SalesAmount)** is the only value left in the list.

The column name preceded by the table name is called the fully qualified name of the column. Fully qualified column names make your formulas easier to read.



6. Select **Sales[SalesAmount]** from the list, and then enter a closing parenthesis.

Tip

Syntax errors are most often caused by a missing or misplaced closing parenthesis.

7. Subtract the other two columns inside the formula:

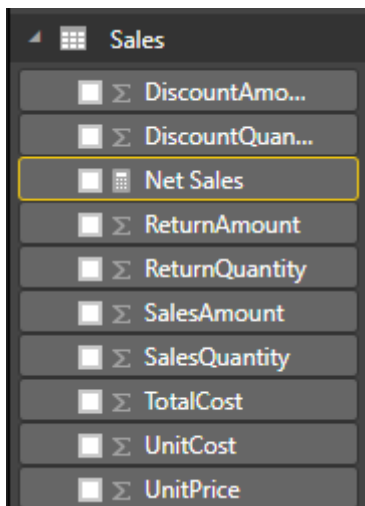
- a. After the closing parenthesis for the first expression, type a space, a minus operator (-), and then another space.
- b. Enter another SUM function, and start typing *DiscountAmount* until you can choose the **Sales[DiscountAmount]** column as the argument. Add a closing parenthesis.
- c. Type a space, a minus operator, a space, another SUM function with **Sales[ReturnAmount]** as the argument, and then a closing parenthesis.



Net Sales = SUM(Sales[SalesAmount]) - SUM(Sales[DiscountAmount]) - SUM(Sales[ReturnAmount])

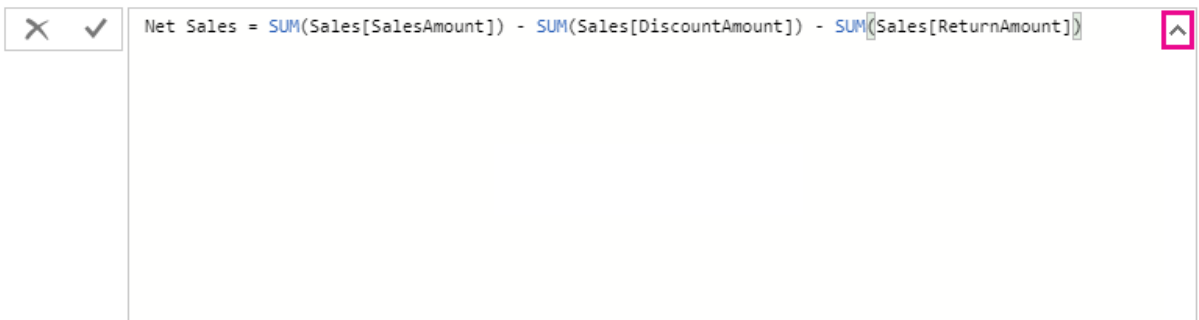
8. Press **Enter** or select **Commit** (checkmark icon) in the formula bar to complete and validate the formula.

The validated **Net Sales** measure is now ready to use in the **Sales** table in the **Fields** pane.



9. If you run out of room for entering a formula or want it on separate lines, select the down arrow on the right side of the formula bar to provide more space.

The down arrow turns into an up arrow and a large box appears.



Net Sales = SUM(Sales[SalesAmount]) - SUM(Sales[DiscountAmount]) - SUM(Sales[ReturnAmount])

10. Separate parts of your formula by pressing **Alt + Enter** for separate lines, or pressing **Tab** to add tab spacing.

```

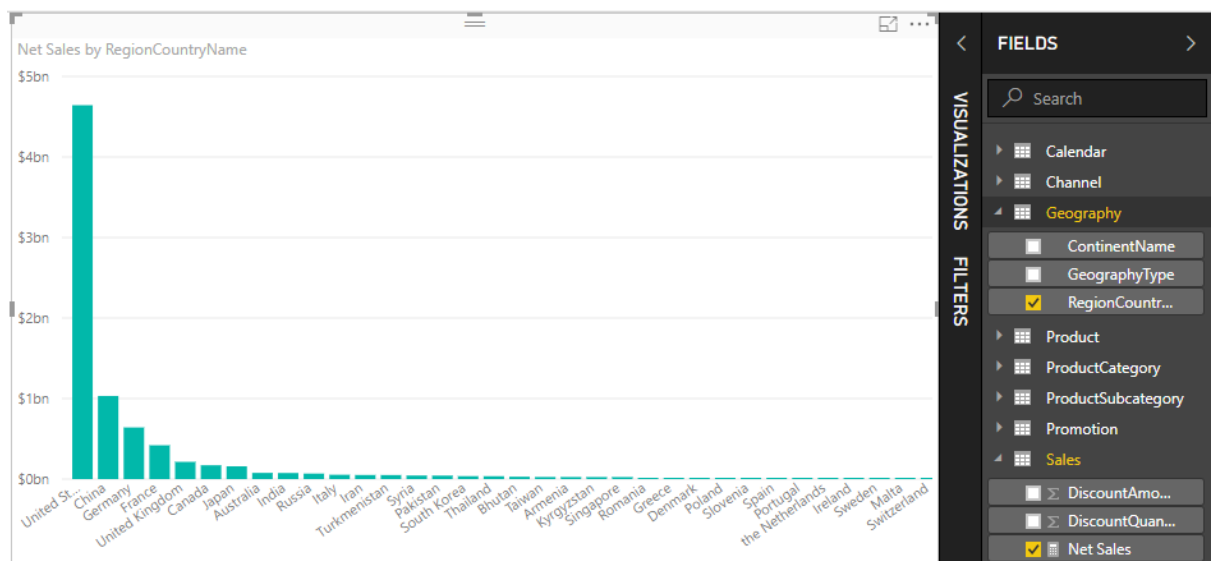
Net Sales = SUM(Sales[SalesAmount])
- SUM(Sales[DiscountAmount])
- SUM(Sales[ReturnAmount])
    
```

Use your measure in the report

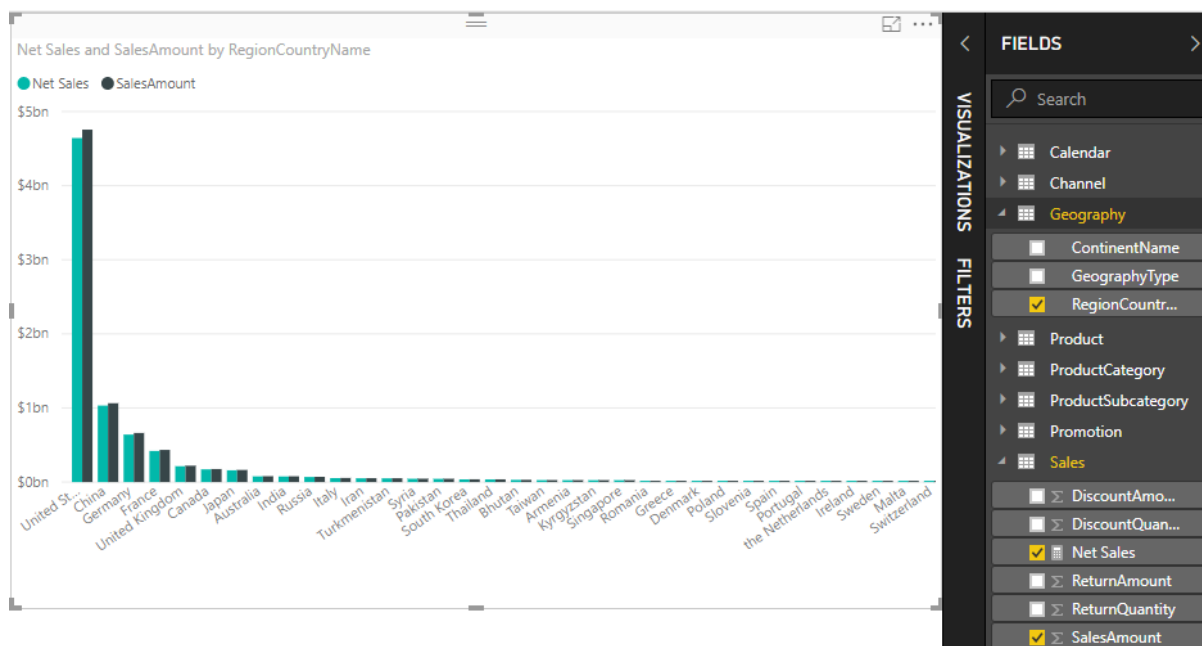
Add your new **Net Sales** measure to the report canvas, and calculate net sales for whatever other fields you add to the report.

To look at net sales by country:

1. Select the **Net Sales** measure from the **Sales** table, or drag it onto the report canvas.
2. Select the **RegionCountryName** field from the **Geography** table, or drag it onto the **Net Sales** chart.



3. To see the difference between net sales and total sales by country, select the **SalesAmount** field or drag it onto the chart.



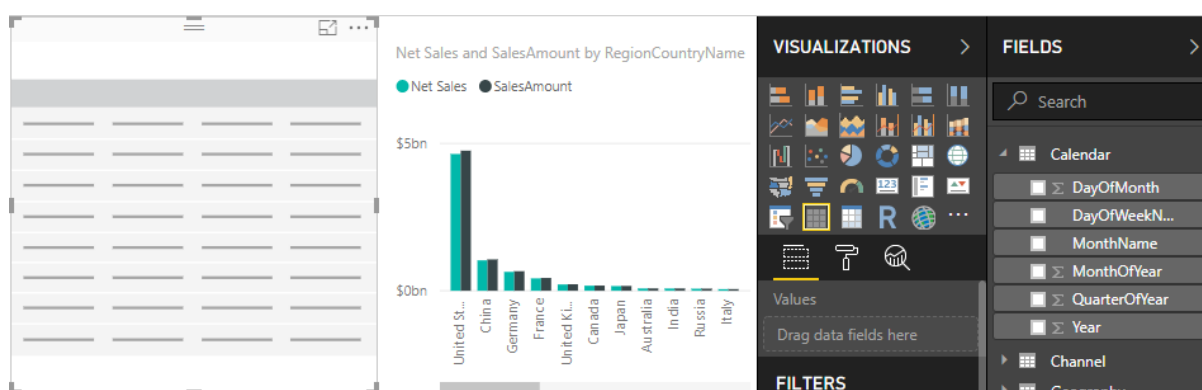
The chart now uses two measures: **SalesAmount**, which Power BI summed automatically, and the **Net Sales** measure, which you manually created. Each measure was calculated in the context of another field, **RegionCountryName**.

Use your measure with a slicer

Add a slicer to further filter net sales and sales amounts by calendar year:

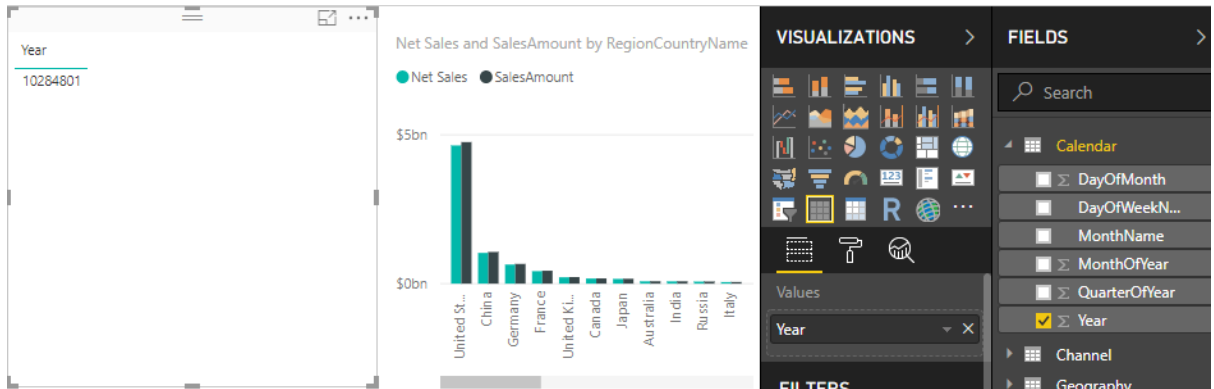
1. Select a blank area next to the chart. In the **Visualizations** pane, select the **Table** visualization.

This action creates a blank table visualization on the report canvas.

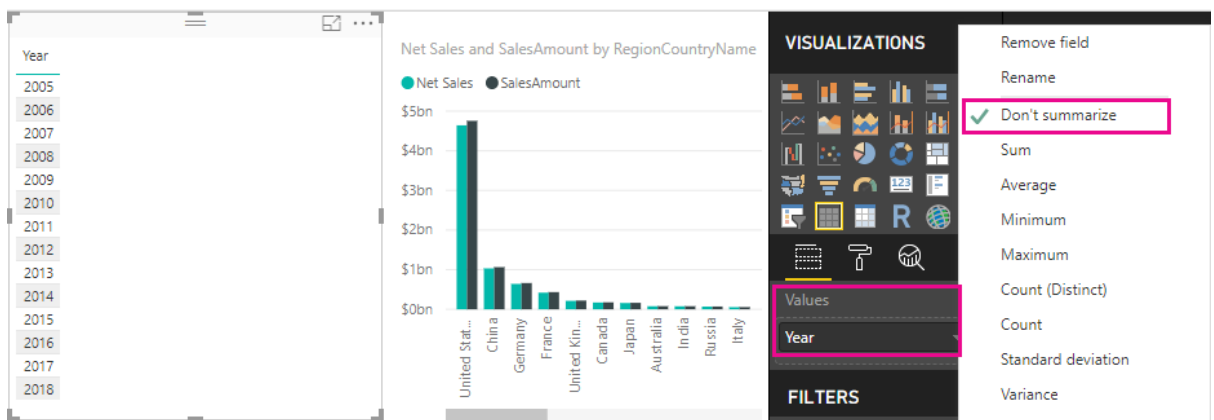


2. Drag the **Year** field from the **Calendar** table onto the new blank table visualization.

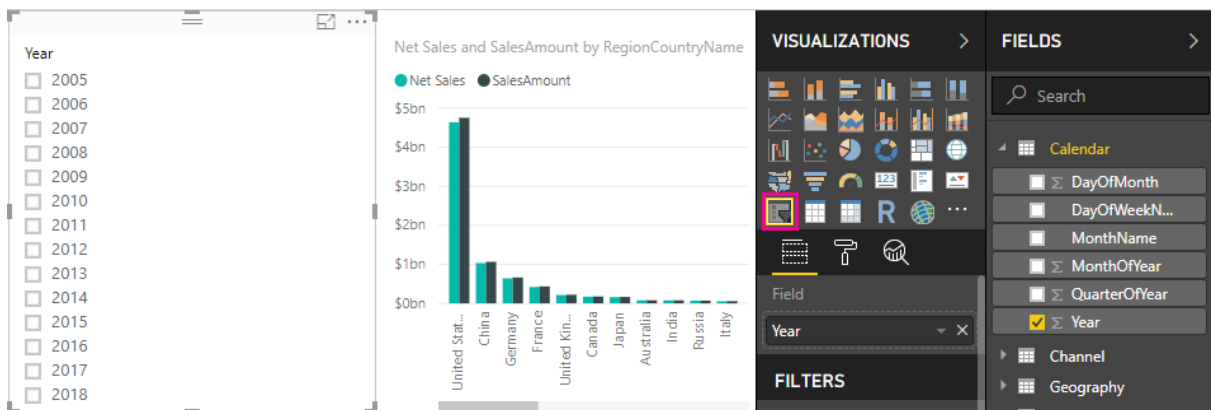
Because **Year** is a numeric field, Power BI Desktop sums up its values. This summation doesn't work well as an aggregation; we'll address that in the next step.



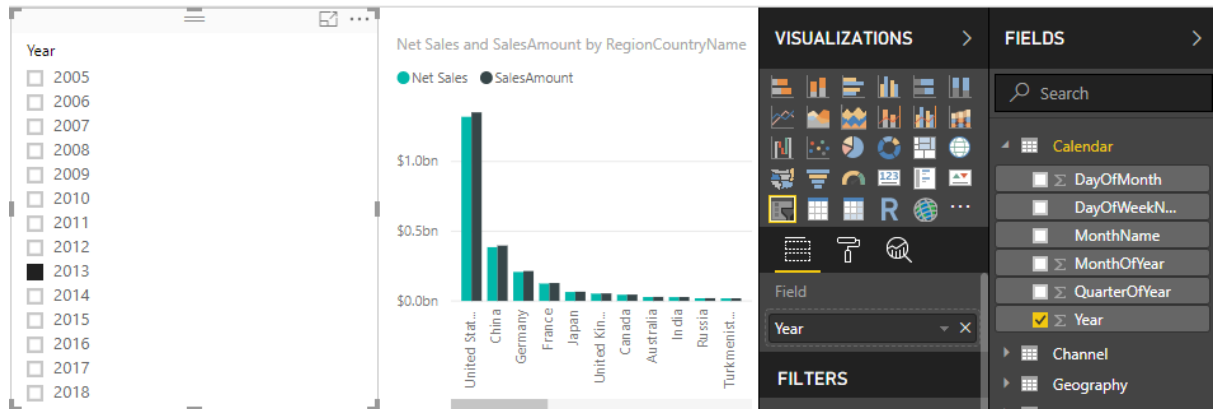
3. In the **Values** box in the **Visualizations** pane, select the down arrow next to **Year**, and then select **Don't summarize** from the list. The table now lists individual years.



4. Select the **Slicer** icon in the **Visualizations** pane to convert the table to a slicer. If the visualization displays a slider instead of a list, select **List** from the down arrow in the slider.



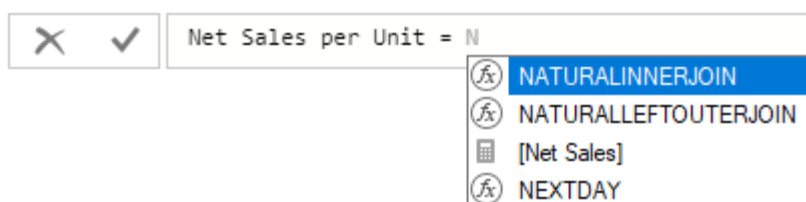
5. Select any value in the **Year** slicer to filter the **Net Sales and Sales Amount by RegionCountryName** chart accordingly. The **Net Sales** and **SalesAmount** measures recalculate and display results in the context of the selected **Year** field.



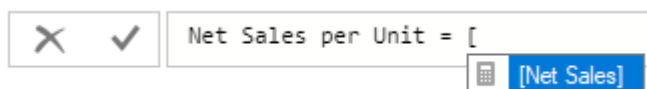
Use your measure in another measure

Suppose you want to find out which products have the highest net sales amount per unit sold. You'll need a measure that divides net sales by the quantity of units sold. Create a new measure that divides the result of your **Net Sales** measure by the sum of **Sales[SalesQuantity]**.

1. In the **Fields** pane, create a new measure named **Net Sales per Unit** in the **Sales** table.
2. In the formula bar, begin typing *Net Sales*. The suggestion list shows what you can add. Select **[Net Sales]**.



3. You can also reference measures by just typing an opening bracket ([). The suggestion list shows only measures to add to your formula.



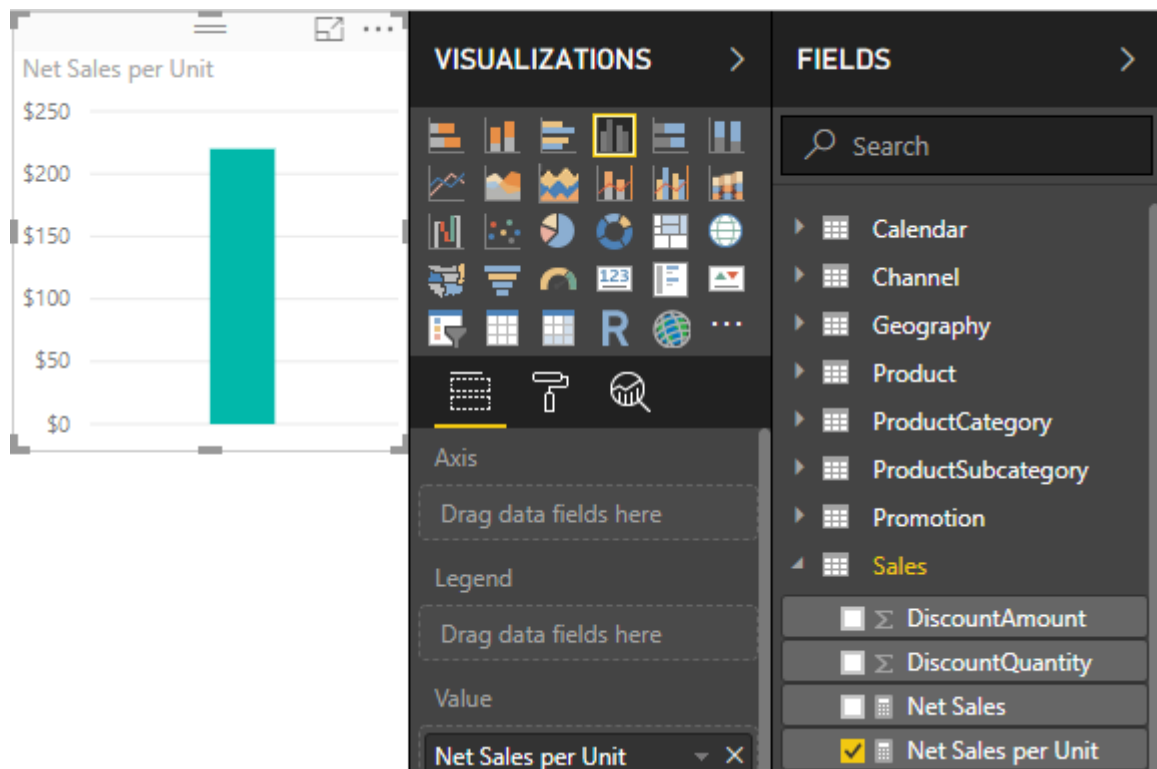
4. Enter a space, a divide operator (/), another space, a SUM function, and then type *Quantity*. The suggestion list shows all the columns with *Quantity* in the name. Select **Sales[SalesQuantity]**, type the closing parenthesis, and press **ENTER** or select **Commit** (checkmark icon) to validate your formula.

The resulting formula should appear as:

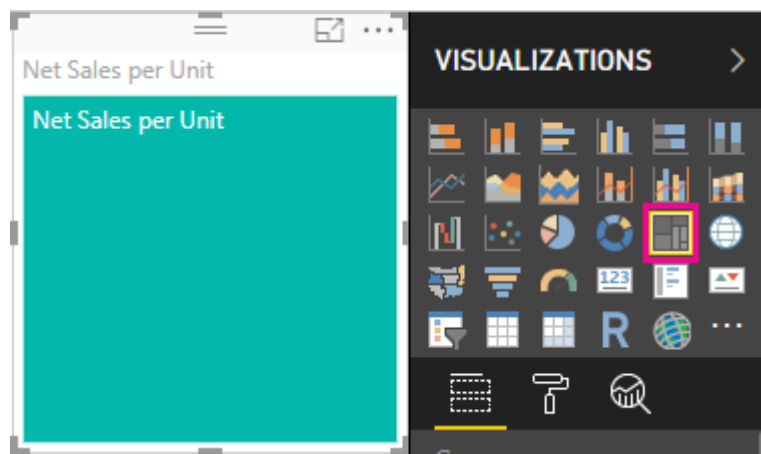
Net Sales per Unit = [Net Sales] / SUM(Sales[SalesQuantity])

5. Select the **Net Sales per Unit** measure from the **Sales** table, or drag it onto a blank area in the report canvas.

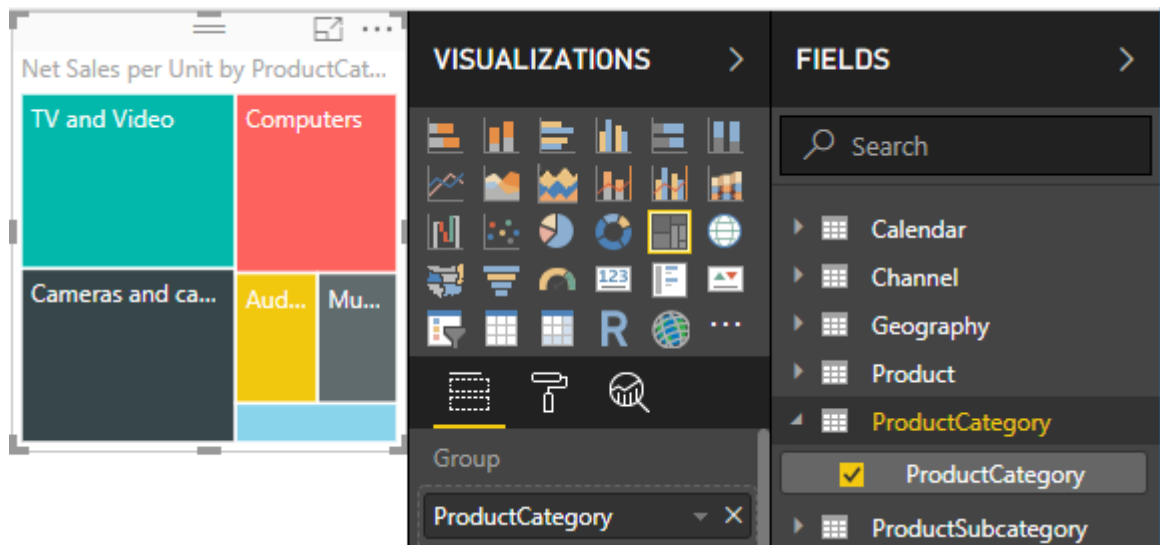
The chart shows the net sales amount per unit over all products sold. This chart isn't very informative; we'll address it in the next step.



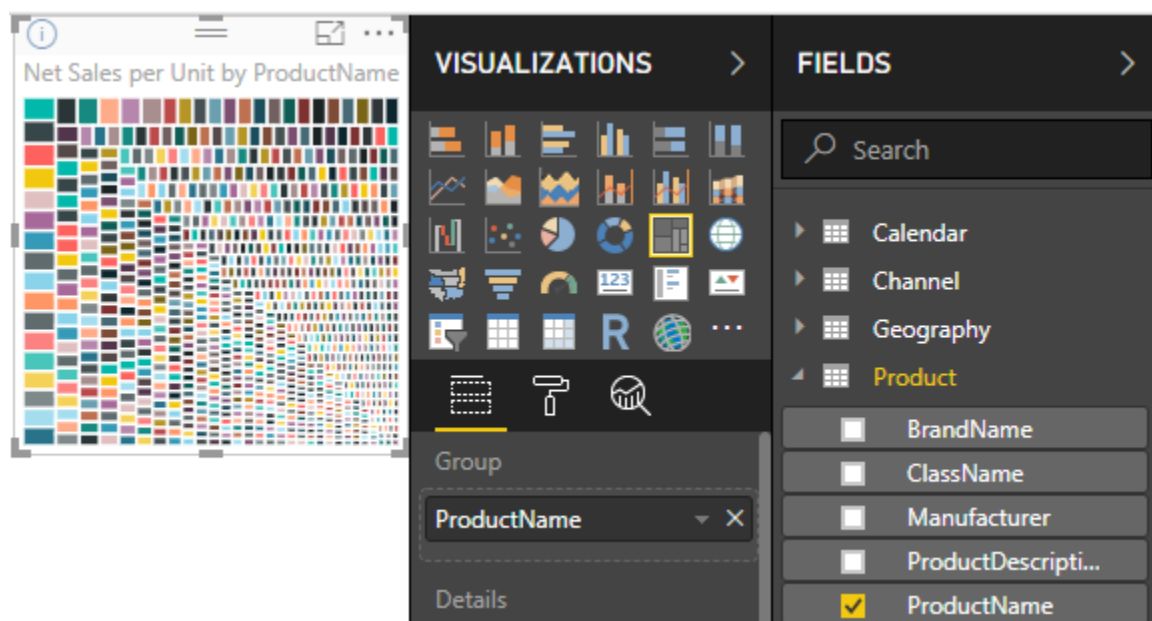
6. For a different look, change the chart visualization type to **Treemap**.



7. Select the **Product Category** field, or drag it onto the treemap or the **Group** field of the **Visualizations** pane. Now you have some good info!



8. Try removing the **ProductCategory** field, and dragging the **ProductName** field onto the chart instead.



Ok, now we're just playing, but you have to admit that's cool! Experiment with other ways to filter and format the visualization.

Tutorial: Create calculated columns in Power BI Desktop

Sometimes the data you're analyzing doesn't contain a particular field you need to get the results you're after. This is where *calculated columns* come in. Calculated columns use Data Analysis Expressions (DAX) formulas to define a column's values, anything from putting together text values from a couple of different columns to calculating a numeric value from other values. For example, let's say your data has **City** and **State** fields, but you want a single **Location** field that has both, like "Miami, FL". This is precisely what calculated columns are for.

Calculated columns are similar to [measures](#) in that both are based on DAX formulas, but they differ in how they are used. You often use measures in a visualization's **Values** area, to calculate results based on other fields. You use calculated columns as new **Fields** in the rows, axes, legends, and group areas of visualizations.

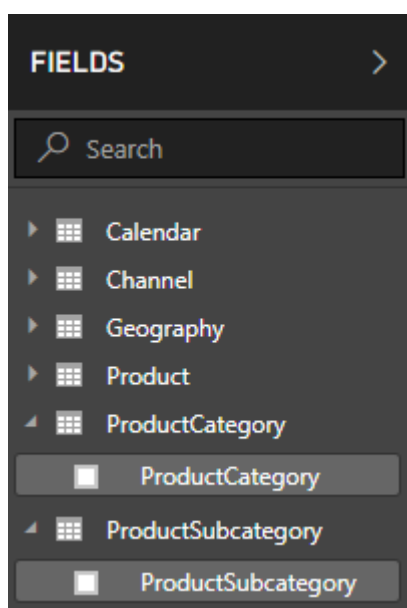
This tutorial will guide you through understanding and creating some calculated columns and using them in report visualizations in Power BI Desktop.

Prerequisites

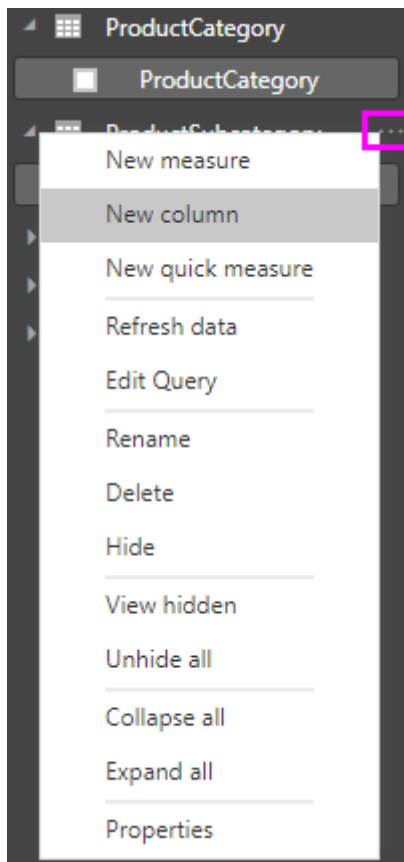
- This tutorial is intended for Power BI users already familiar with using Power BI Desktop to create more advanced models. You should already know how to use **Get Data** and the **Power Query Editor** to import data, work with multiple related tables, and add fields to the Report canvas. If you're new to Power BI Desktop, be sure to check out [Getting Started with Power BI Desktop](#).
- The tutorial uses the [Contoso Sales Sample for Power BI Desktop](#), the same sample used for the [Create your own measures in Power BI Desktop](#) tutorial. This sales data from the fictitious company Contoso, Inc. was imported from a database, so you won't be able to connect to the data source or view it in the Power Query Editor. Download and extract the file on your own computer, and then open it in Power BI Desktop.

Create a calculated column with values from related tables

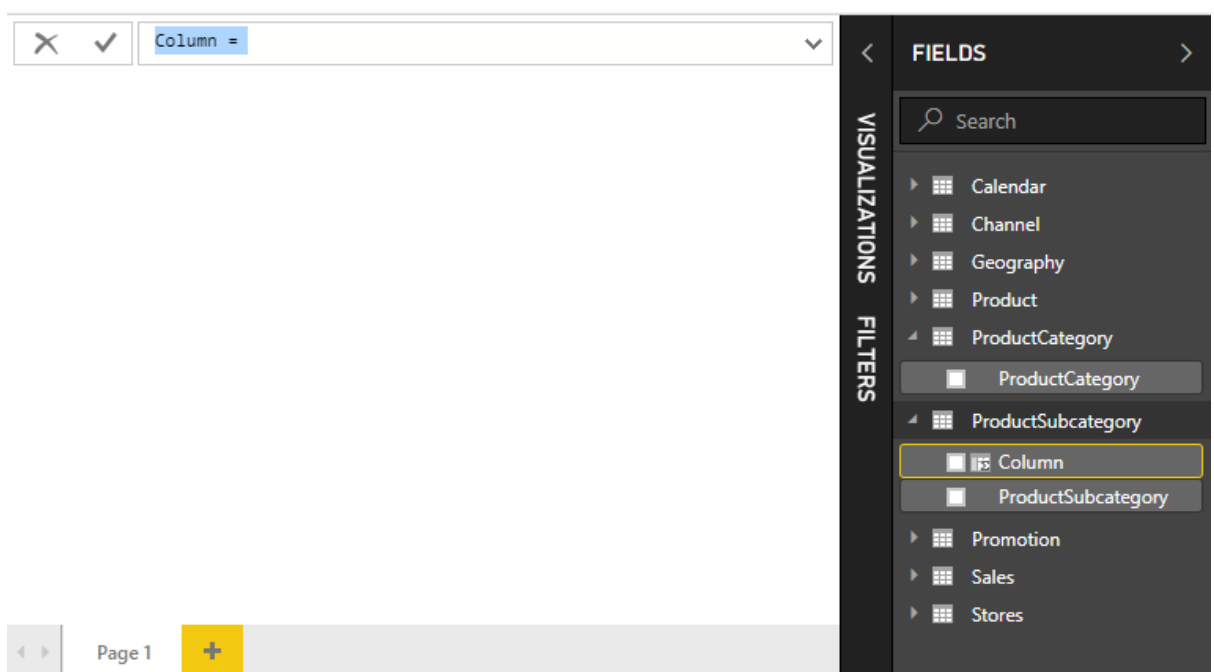
In your Sales Report, you want to display product categories and subcategories as single values, like "Cell phones – Accessories", "Cell phones – Smartphones & PDAs", and so on. There's no field in the **Fields** list that gives you that data, but there is a **ProductCategory** field and a **ProductSubcategory** field, each in its own table. You can create a calculated column that combines values from these two columns. DAX formulas can leverage the full power of the model you already have, including relationships between different tables that already exist.



1. Select **More options** (...), or right-click, on the **ProductSubcategory** table in the Fields list, and then select **New Column**. This creates your new column in the ProductSubcategory table.

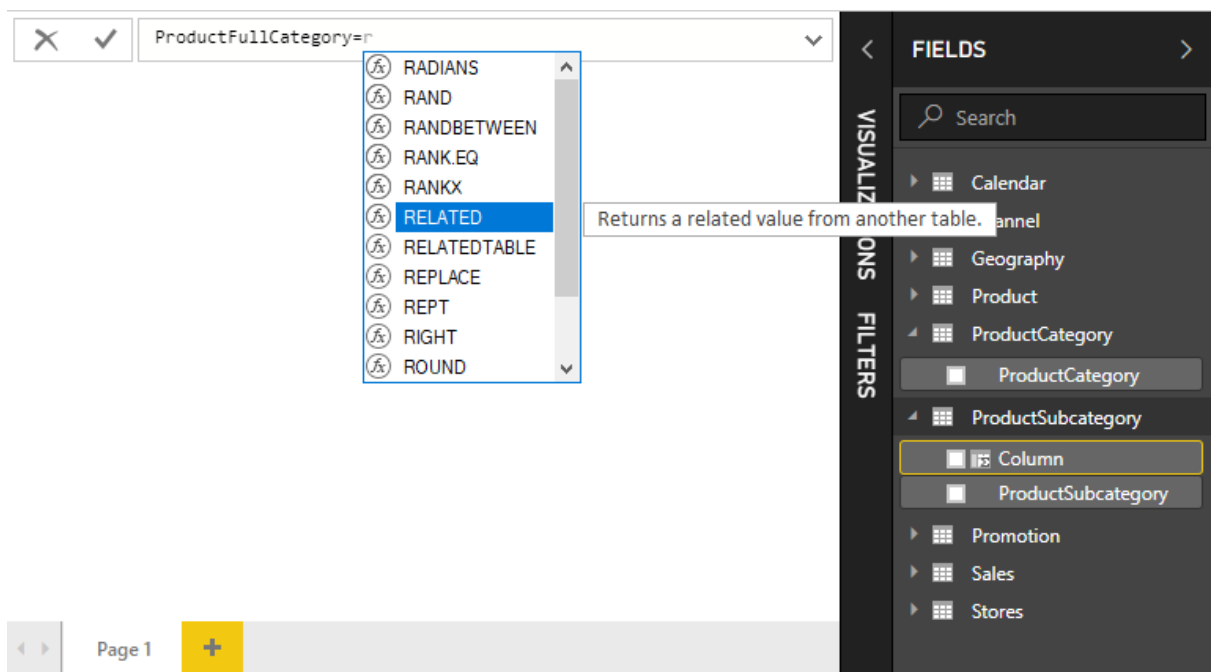


The formula bar appears along the top of the Report canvas, ready for you to name your column and enter a DAX formula.

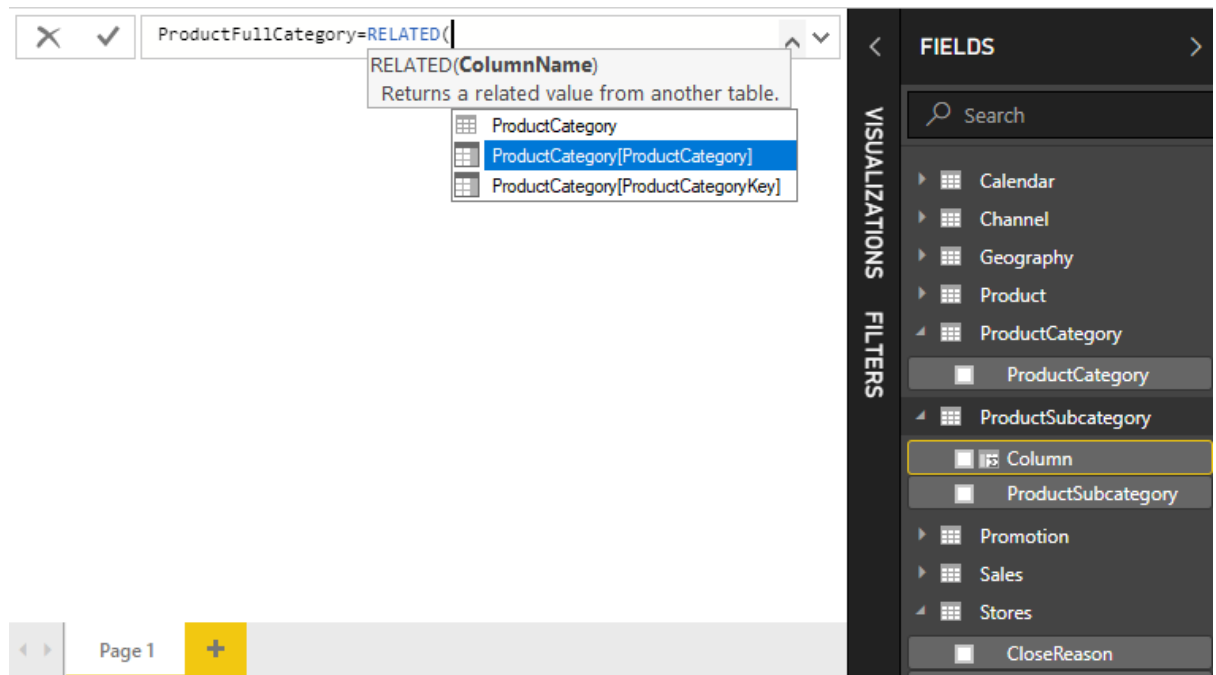


2. By default, a new calculated column is simply named Column. If you don't rename it, additional new columns will named Column 2, Column 3, and so on. You want your column to be more identifiable, so since the **Column** name is already highlighted in the formula bar, rename it by typing **ProductFullCategory**, and then type an equals (=) sign.
3. You want the values in your new column to start with the ProductCategory name. Because this column is in a different but related table, you can use the [RELATED](#) function to help you get it.

After the equals sign, type **r**. A dropdown suggestion list shows all of the DAX functions beginning with the letter R. Selecting each function shows a description of its effect. As you type, the suggestion list scales closer to the function you need. Select **RELATED**, and then press **Enter**.



An opening parenthesis appears, along with another suggestion list of the related columns you can pass to the RELATED function, with descriptions and details on expected parameters.



4. You want the **ProductCategory** column from the **ProductCategory** table. Select **ProductCategory[ProductCategory]**, press **Enter**, and then type a closing parenthesis.

Tip

Syntax errors are most often caused by a missing or misplaced closing parenthesis, although sometimes Power BI Desktop will add it for you.

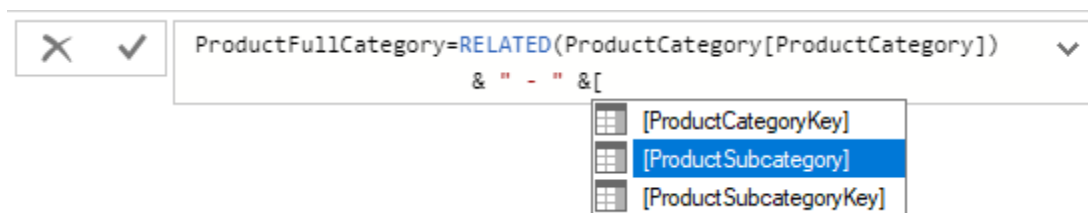
5. You want dashes and spaces to separate the ProductCategories and ProductSubcategories in the new values, so after the closing parenthesis of the first expression, type a space, ampersand (&), double-quote ("), space, dash (-), another space, another double-quote, and another ampersand. Your formula should now look like this:

```
ProductFullCategory = RELATED(ProductCategory[ProductCategory]) & " - " &
```

Tip

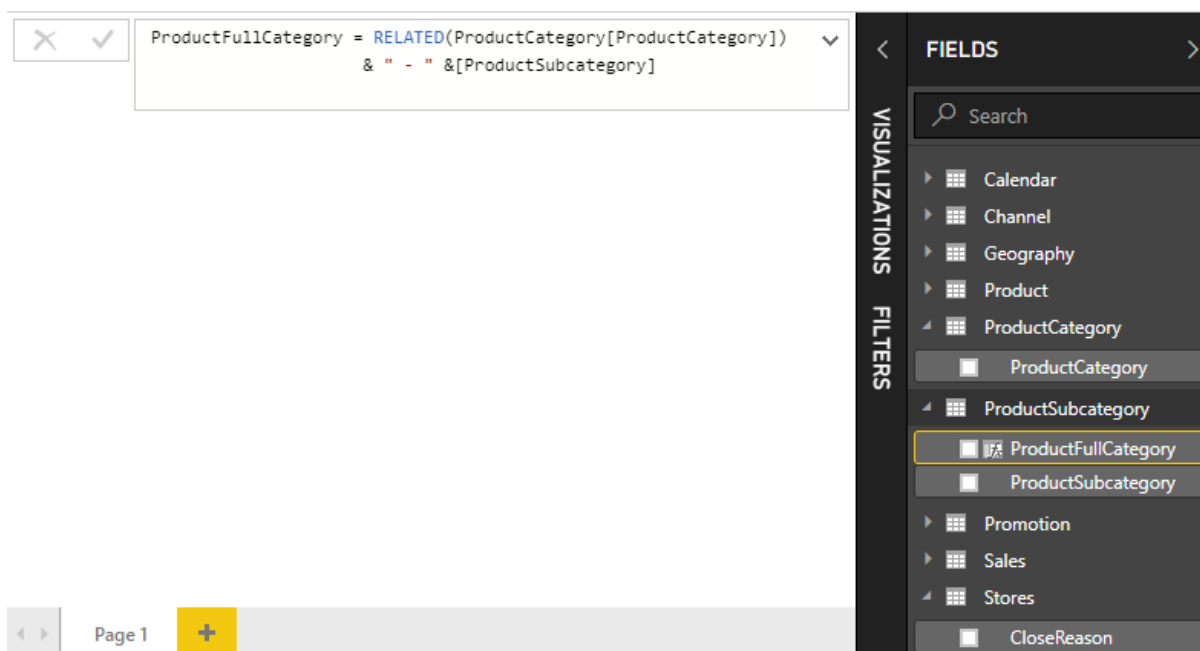
If you need more room, select the down chevron on the right side of the formula bar to expand the formula editor. In the editor, press **Alt + Enter** to move down a line, and **Tab** to move things over.

6. Enter an opening bracket ([), and then select the **[ProductSubcategory]** column to finish the formula.



You didn't need to use another RELATED function to call the ProductSubcategory table in the second expression, because you are creating the calculated column in this table. You can enter [ProductSubcategory] with the table name prefix (fully-qualified) or without (non-qualified).

- Complete the formula by pressing **Enter** or selecting the checkmark in the formula bar. The formula validates, and the **ProductFullCategory** column name appears in the **ProductSubcategory** table in the Fields list.



Note

In Power BI Desktop, calculated columns get a special icon in the field list, showing that they contain formulas. In the Power BI service (your Power BI site), there's no way to change formulas, so calculated columns don't have icons.

Use your new column in a report

Now you can use your new ProductFullCategory column to look at SalesAmount by ProductFullCategory.

- Select or drag the **ProductFullCategory** column from the **ProductSubcategory** table onto the Report canvas to create a table showing all ProductFullCategory names.

The screenshot shows the Power BI interface with the 'ProductFullCategory' field selected in the 'VALUES' pane and the 'SalesAmount' field selected in the 'FILTERS' pane. The 'FIELDS' pane on the right shows the hierarchy of fields including ProductCategory, ProductSubcategory, and Sales.

2. Select or drag the **SalesAmount** field from the **Sales** table into the table to show the Sales Amount for each Product Full Category.

The screenshot shows the Power BI interface with the 'SalesAmount' field selected in the 'VALUES' pane and the 'ProductFullCategory' field selected in the 'FILTERS' pane. The 'FIELDS' pane on the right shows the hierarchy of fields including ProductCategory, ProductSubcategory, and Sales.

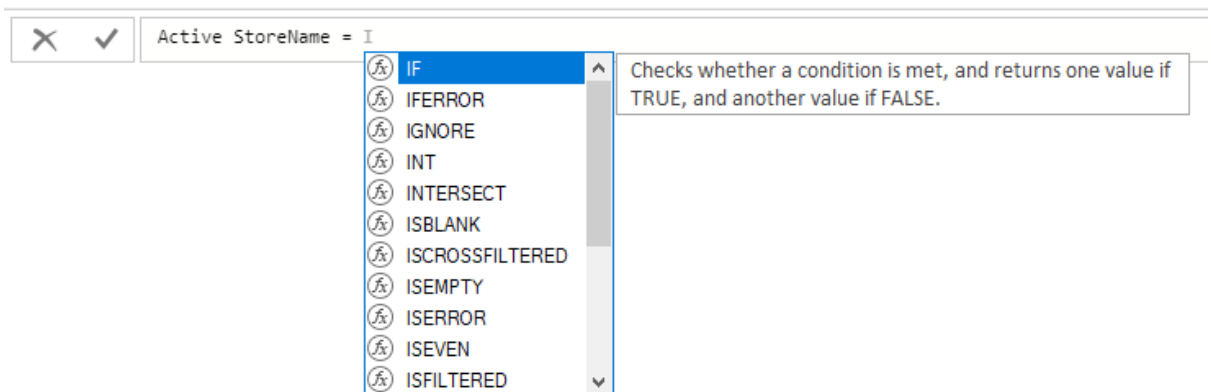
ProductFullCategory	SalesAmount
Audio - Bluetooth Headphones	\$41,907,488.9135
Audio - MP4&MP3	\$65,190,616.3964
Audio - Recording Pen	\$44,516,259.001
Cameras and camcorders - Camcorders	\$1,335,302,769.92
Cameras and camcorders - Cameras & Camcorders Accessories	\$51,643,775.8286
Cameras and camcorders - Digital Cameras	\$365,082,489.926
Cameras and camcorders - Digital SLR Cameras	\$809,994,738.386
Cell phones - Cell phones Accessories	\$120,017,198.2572
Cell phones - Home & Office Phones	\$48,333,160.9812
Cell phones - Smart phones & PDAs	\$423,389,457.48
Cell phones - Touch Screen Phones	\$300,493,447.58
Computers - Computers Accessories	\$111,023,802.1072
Computers - Desktops	\$508,196,937.084
Computers - Laptops	\$933,130,593.776
Computers - Monitors	\$268,114,052.77
Computers - Printers, Scanners & Fax	\$281,762,342.2
Computers - Projectors & Screens	\$1,107,199,413.48
Music, Movies and Audio Books - Movie DVD	\$165,804,705.9811
TV and Video - Car Video	\$306,818,844.52
TV and Video - Home Theater System	\$709,120,510.209
TV and Video - Televisions	\$307,373,914.4742
TV and Video - VCR & DVD	\$36,807,845.561

Create a calculated column that uses an IF function

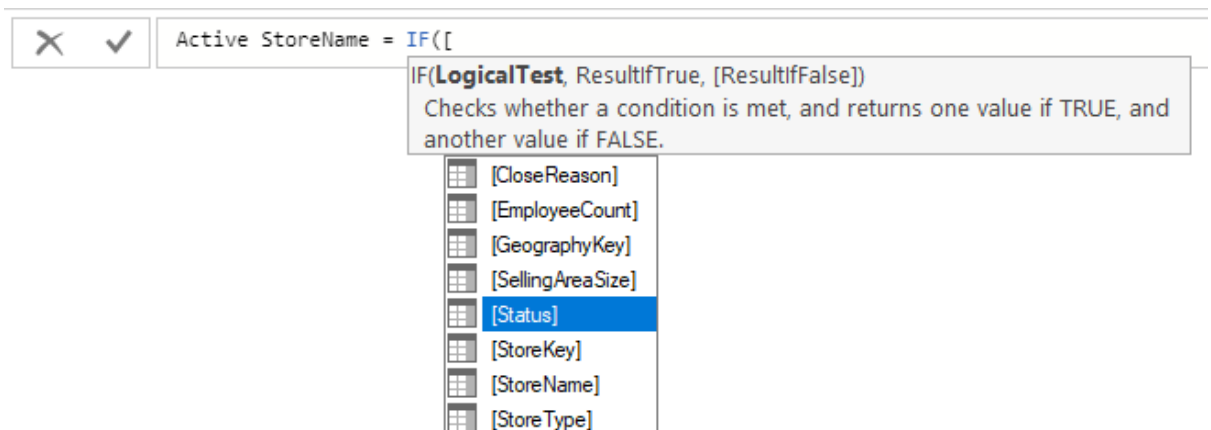
The Contoso Sales Sample contains sales data for both active and inactive stores. You want to ensure that Active store sales are clearly separated from Inactive store sales in your report by creating an Active StoreName field. In the new Active StoreName calculated column, each Active store will appear with the store's full name, while inactive stores will be grouped together under "Inactive".

Fortunately, the Stores table has a column named **Status**, with values of "On" for active stores and "Off" for inactive stores, which we can use to create values for our new Active StoreName column. Your DAX formula will use the logical **IF** function to test each store's Status and return a particular value depending on the result. If a store's Status is "On", the formula will return the store's name. If it's "Off", the formula will assign an Active StoreName of "Inactive".

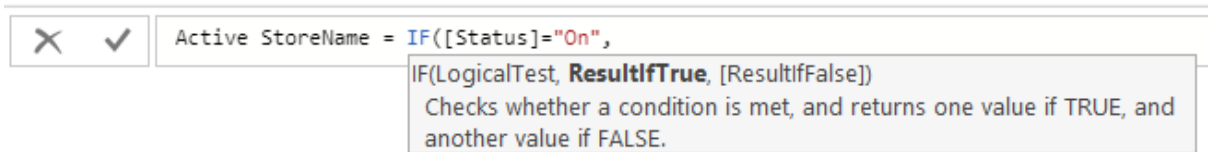
1. Create a new calculated column in the **Stores** table and name it **Active StoreName** in the formula bar.
2. After the = sign, begin typing **IF**. The suggestion list will show what you can add. Select **IF**.



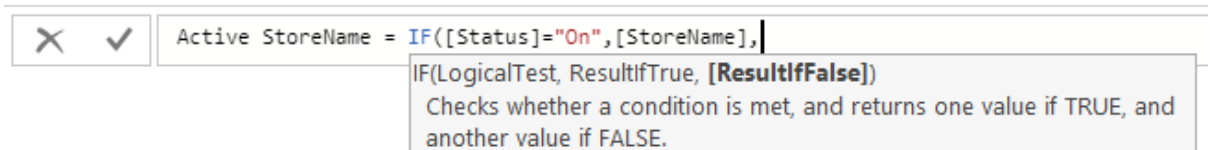
3. The first argument for IF is a logical test of whether a store's Status is "On". Type an opening bracket [, which lists columns from the Stores table, and select **[Status]**.



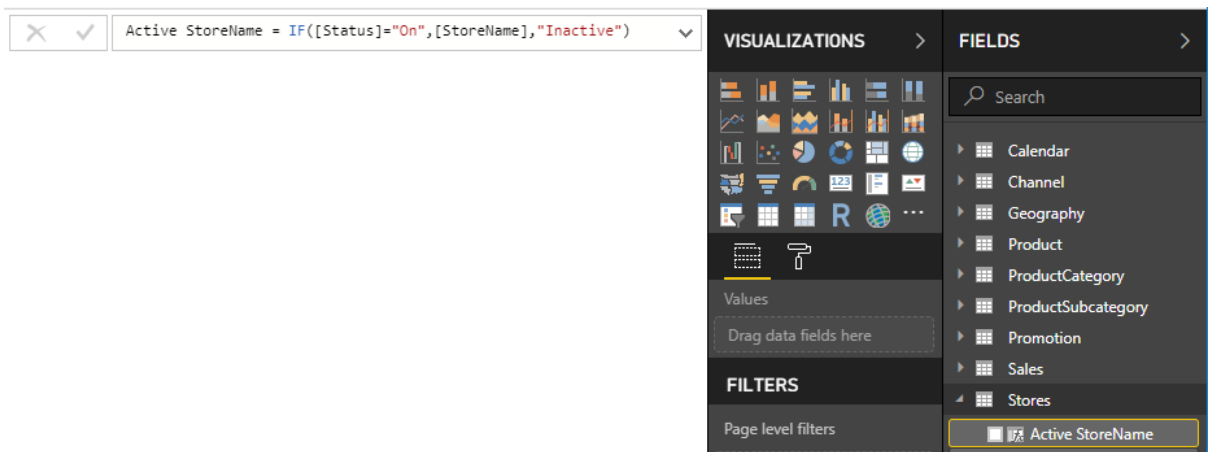
- Right after **[Status]**, type **"On"**, and then type a comma (,) to end the argument. The tooltip suggests that you now need to add a value to return when the result is TRUE.



- If the store's status is "On", you want to show the store's name. Type an opening bracket ([) and select the **[StoreName]** column, and then type another comma. The tooltip now indicates that you need to add a value to return when the result is FALSE.



- You want the value to be *Inactive*, so type **"Inactive"**, and then complete the formula by pressing **Enter** or selecting the checkmark in the formula bar. The formula validates, and the new column's name appears in the **Stores** table in the Fields list.



- You can use your new **Active StoreName** column in visualizations just like any other field. To show **SalesAmounts** by **Active StoreName**, select the **Active StoreName** field or drag it onto the canvas, and then select the **SalesAmount** field or drag it into the table. In this table, active stores appear individually by name, but inactive stores are grouped together at the end as *Inactive*.

The screenshot displays the Microsoft Power BI Desktop interface. On the left, a data table is shown with two columns: 'Active StoreName' and 'SalesAmount'. The table lists various store names and their corresponding sales amounts, with a 'Total' row at the bottom. The 'Inactive' row is highlighted with a pink border. On the right, the 'Visualizations' pane shows a grid of visualization icons, with the 'Table' icon selected. Below the icons, the 'Values' section shows 'Active StoreName' and 'SalesAmount' as the selected fields. The 'Filters' section shows 'Active StoreName (All)' and 'SalesAmount (All)' as the selected filters. On the far right, the 'Fields' pane shows a list of fields, with 'SalesAmount' and 'Active StoreName' selected.

Active StoreName	SalesAmount
Contoso Vailletta Store	\$14,550,1
Contoso Vancouver No.1 St...	\$16,614,3
Contoso Vancouver No.2 St...	\$16,079,3
Contoso Venezia Store	\$1,215
Contoso Veradale Store	\$15,620
Contoso Vineland Store	\$15,727,3
Contoso Virginia Beach Store	\$15,460
Contoso Wapato Store	\$16,427,3
Contoso Warsaw Store	\$15,142,7
Contoso Waterbury Store	\$15,104,3
Contoso Waukesha No.1 Store	\$16,032,4
Contoso Waukesha No.2 Store	\$16,448,3
Contoso West Yorkshire Store	\$15,165
Contoso Westminster Store	\$15,266,3
Contoso Wheat Ridge Store	\$16,117
Contoso Winchester Store	\$15,563,9
Contoso Worcester No.1 Store	\$15,388
Contoso Yakima Store	\$16,266
Contoso Yerevan Store	\$26,084,9
Contoso Yokohama Store	\$25,311,7
Contoso York Store	\$14,926,0
Inactive	\$189,962,3
Total	\$8,341,224,3

What you've learned

Calculated columns can enrich your data and provide easier insights. You've learned how to create calculated columns in the field list and formula bar, use suggestion lists and tooltips to help construct your formulas, call DAX functions like **RELATED** and **IF** with the appropriate arguments, and use your calculated columns in report visualizations.

Publish from Power BI Desktop

When you publish a **Power BI Desktop** file to the **Power BI service**, the data in the model and any reports you created in **Report** view are published to your Power BI workspace. You'll see a new dataset with the same name, and any reports in your Workspace navigator.

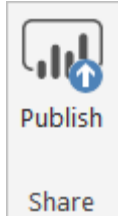
Publishing from **Power BI Desktop** has the same effect as using **Get Data** in Power BI to connect to and upload a **Power BI Desktop** file.

Note

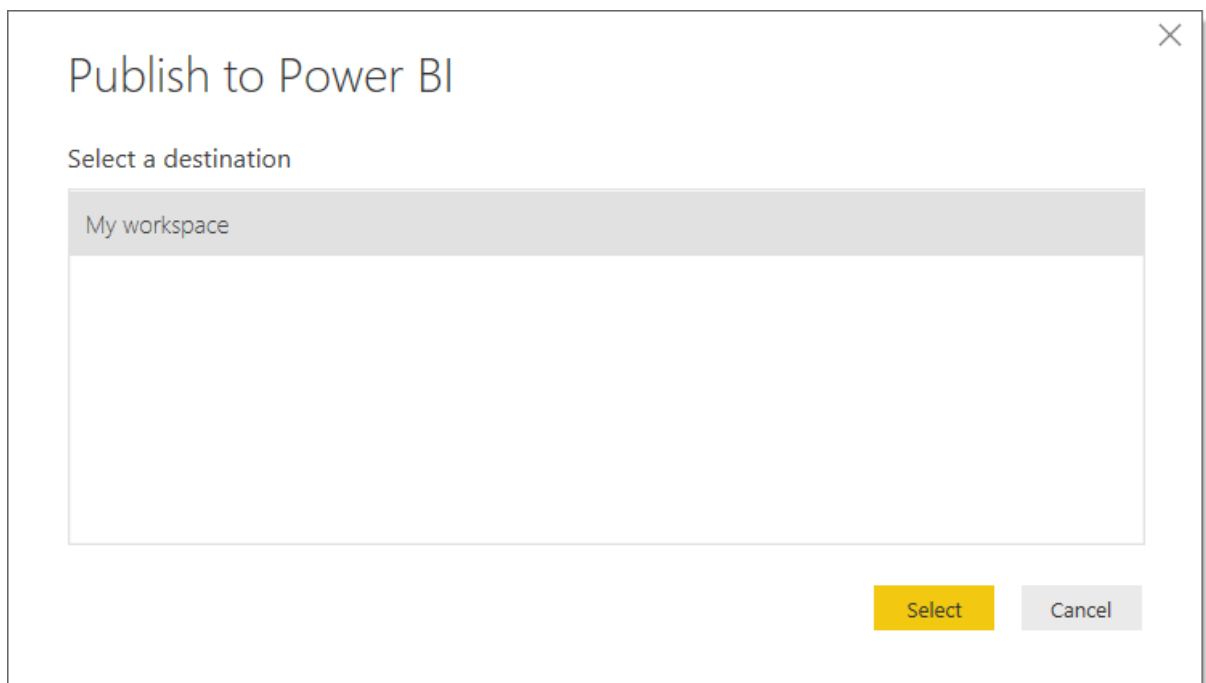
Any changes you make to the report in Power BI, for example, add, delete, or change visualizations in reports, will not be saved back to the original **Power BI Desktop** file.

To publish a Power BI Desktop dataset and reports

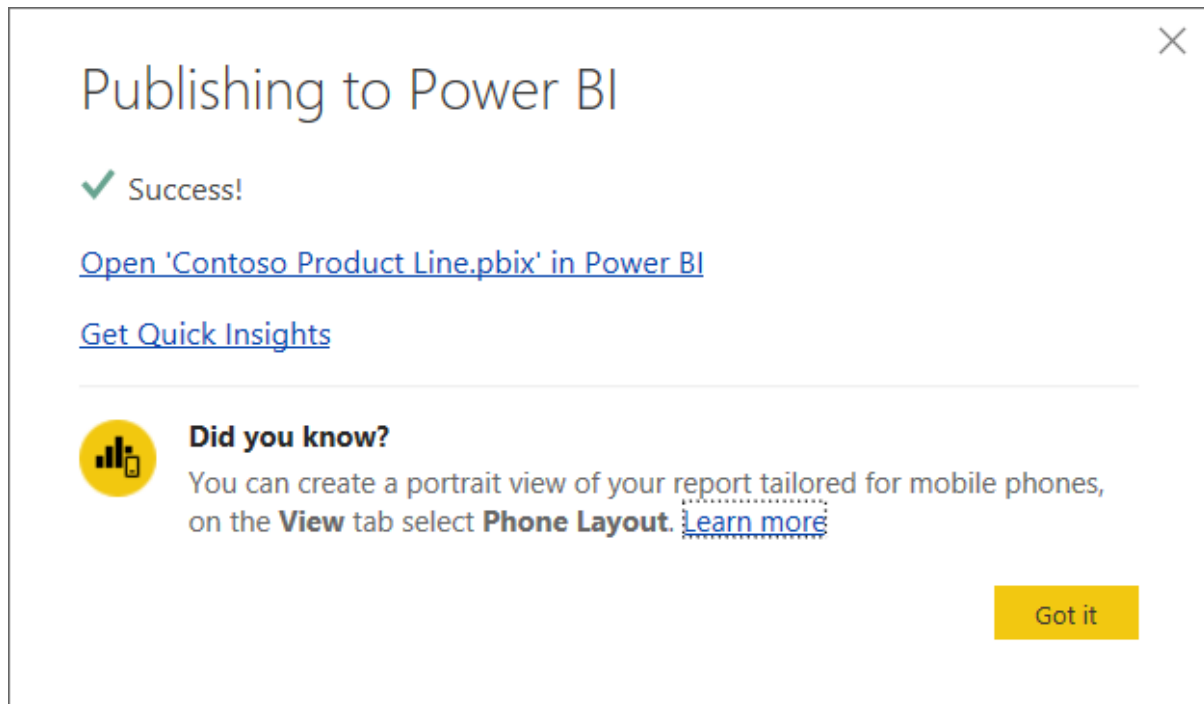
1. In Power BI Desktop select **File > Publish > Publish to Power BI** or click **Publish** on the ribbon.



2. Sign in to Power BI.
3. Select the destination.



When complete, you receive a link to your report. Click the link to open the report in your Power BI site.



Re-publish or replace a dataset published from Power BI Desktop

When you publish a **Power BI Desktop** file, the dataset and any reports you created in **Power BI Desktop** are uploaded to your Power BI site. When you re-publish your **Power BI Desktop** file, the dataset in your Power BI site will be replaced with the updated dataset from the **Power BI Desktop** file.

This is all straightforward, but there are a few things you should know:

- If you already have two or more datasets in Power BI with the same name as the **Power BI Desktop** file, publish could fail. Make sure you have only one dataset in Power BI with the same name. You can also rename the file and publish, creating a new dataset with same name as the file.
- If you rename or delete a column or measure, any visualizations you already have in Power BI with that field could be broken.
- Power BI ignores some format changes of existing columns. For example, if you change a column's format from 0.25 to 25%.
- If you have a refresh schedule configured for your existing dataset in Power BI and you add new data sources to your file and then re-publish, you'll have to sign into them in *Manage Data Sources* prior to the next scheduled refresh.
- When you republish a dataset published from **Power BI Desktop** and have a refresh schedule defined, a dataset refresh is initiated as soon as you republish.