✓ 100 XP ▶

Introduction

8 minutes

In this module, you learn about various key features of the Power BI service. This is an introductory course intended to teach you how to build reports on the Power BI Desktop, create operational dashboards, and share content via the Power BI Service.

By the end of this Module, you learn:

- How to load data from Microsoft Excel and Comma-Separated Values (CSV) sources
- How to clean the data to prepare it for reporting
- How to prepare tables in Power Query and load them into the model

Learning these steps prepares you for the modeling exercises in Module 1. Additionally, the results of this Module are the starting point for Module 3.

Access data

In Unit 1, you:

- Import VanArsdel, Ltd. USA sales data
- Import their competitors' USA sales data

In Unit 2, you:

- Import and merge sales data from other countries

In Unit 3, you:

- Clean up all the data

Get data

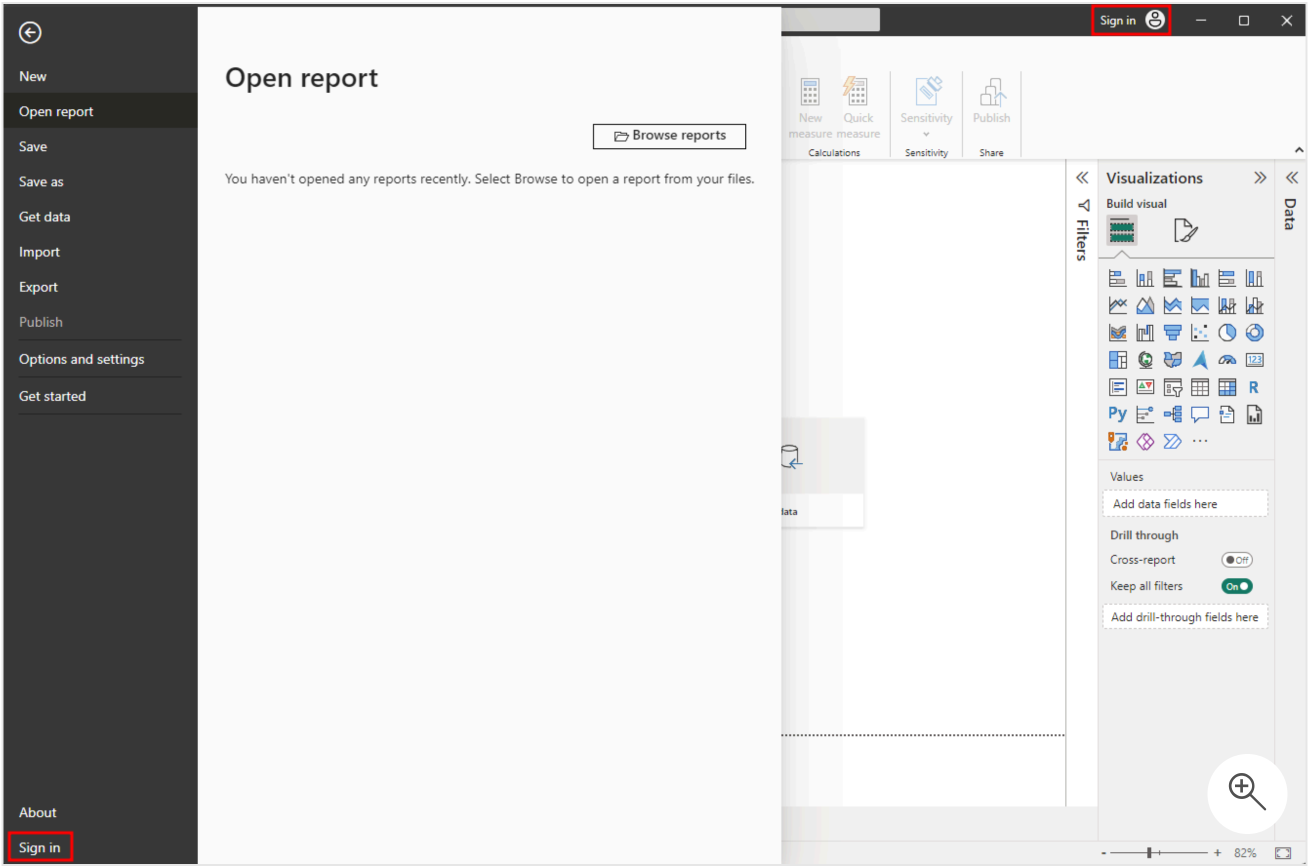
The dataset for this course contains sales data from VanArsdel, Ltd. and other competitors. We have seven years of transaction data by day, product, and zip code for each manufacturer. We're going to analyze data from seven countries.

To find the USA sales data, go to Usages > Data > USSales.csv.

To find sales of all other countries, Data > InternationalSales.

Product, Geography, and Manufacturer information is in a Microsoft Excel file called bi_dimensions.xlsx in the USSales subfolder in the Data folder (/Data/USSales/).

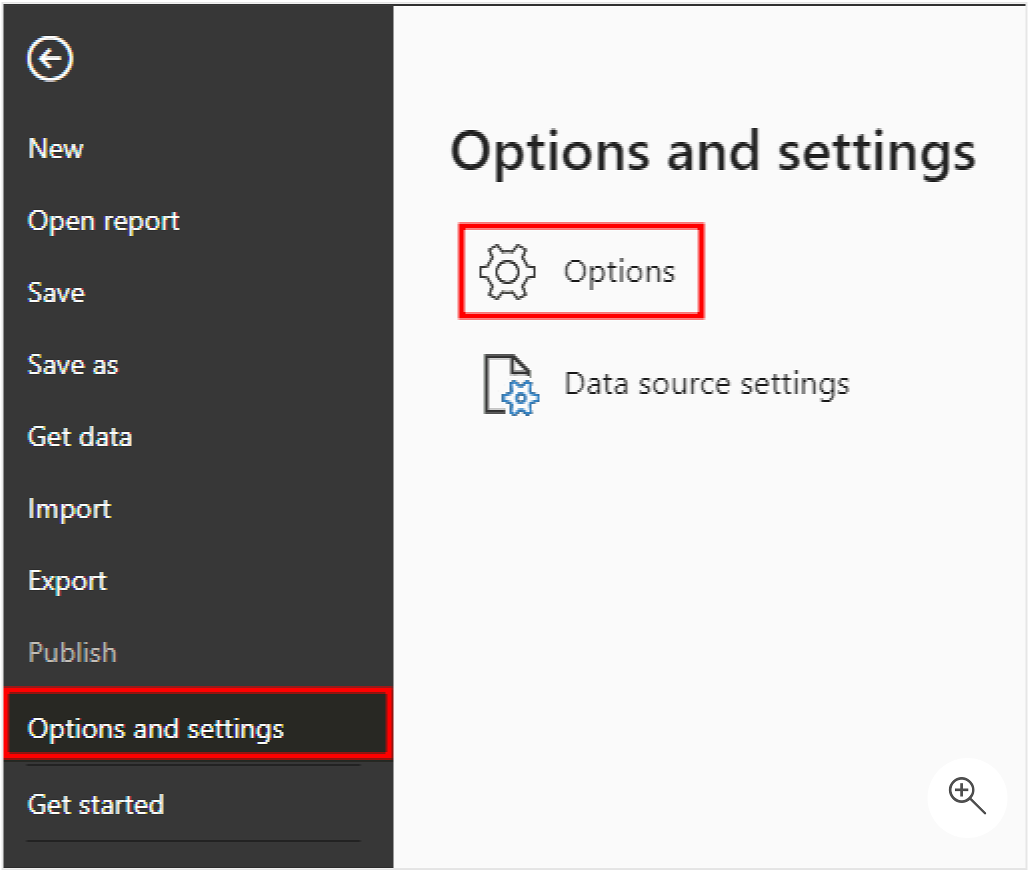
1. If you don't already have the **Power BI Desktop** open, launch it now.
2. If you aren't signed into the **Power BI Desktop**, select the **Sign in** option.
3. **Sign in** using your Power BI credentials.



You haven't opened any reports recently. Select Browse to open a report from your files.

Next, let's set the **Locale** to US English to make it convenient for the rest of this lab.

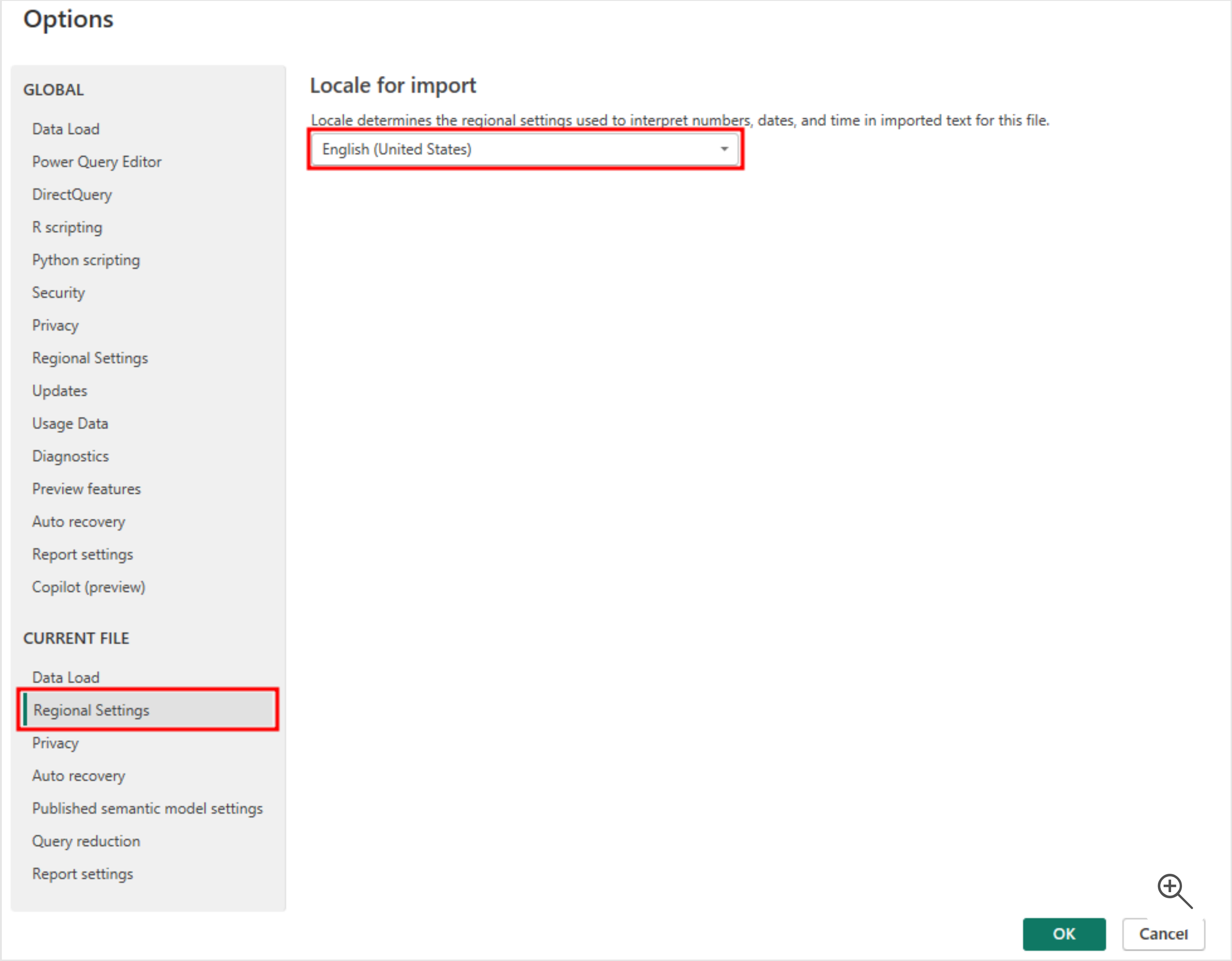
4. From the ribbon, select **File**, then choose **Options and settings**. Then, select **Options**.



5. In the pane to the left of the **Options** dialog box, select **Regional Settings** under **Current File**.

6. From the **Locale** drop-down, select **English (United States)**.

7. Then, select **OK** to close the dialog box.



Options

GLOBAL

Data Load
Power Query Editor
DirectQuery
R scripting
Python scripting
Security
Privacy
Regional Settings
Updates
Usage Data
Diagnostics
Preview features
Auto recovery
Report settings
Copilot (preview)

CURRENT FILE

Data Load
Regional Settings
Privacy
Auto recovery
Published semantic model settings
Query reduction
Report settings

Locale for import

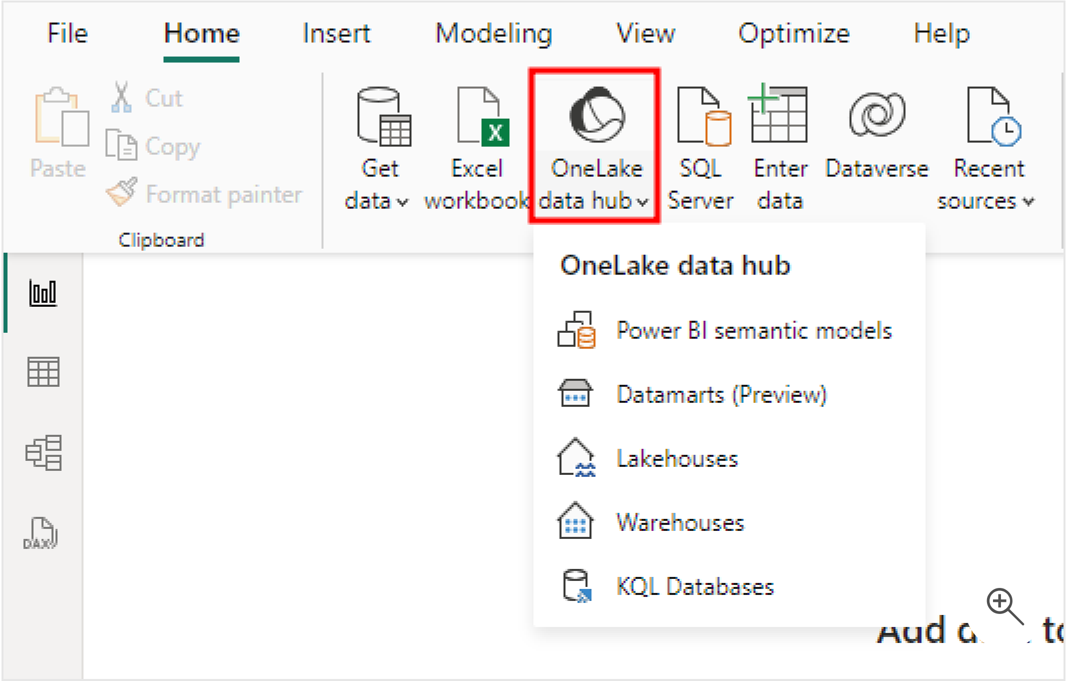
Locale determines the regional settings used to interpret numbers, dates, and time in imported text for this file.

English (United States)

The next step is to load data into **Power BI Desktop**.

Note

Power BI Desktop has the capability to connect to 300+ data sources. The newest sources are part of Microsoft Fabric's OneLake data hub. You will not be using OneLake in today's class but to learn more read here: Tutorial: Fabric for Power BI users.

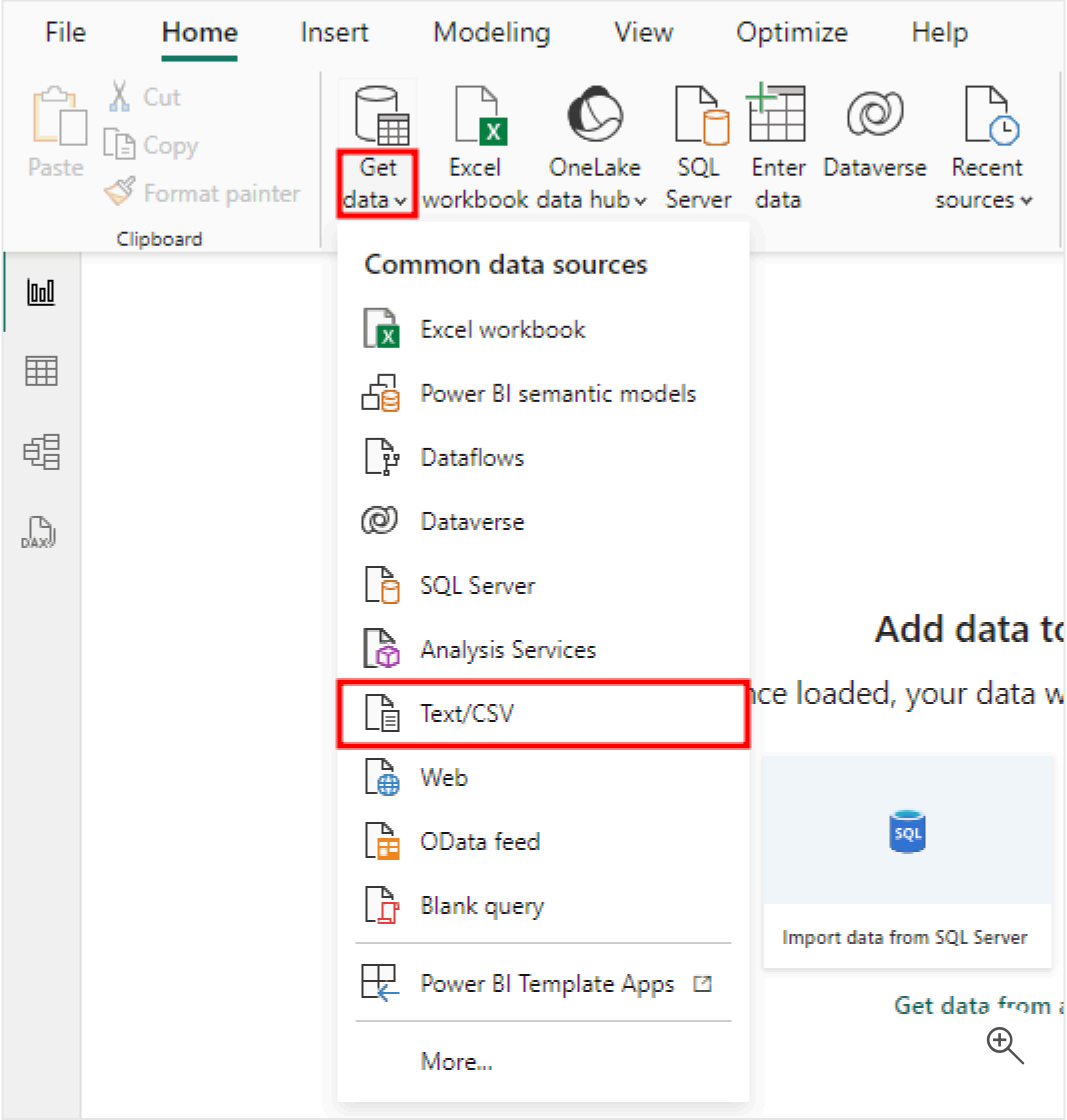


We're using CSV and Excel data files in this lab for simplicity. If you would like a full list of data sources, see: Data sources in Power BI Desktop.

Start by loading **USA Sales data**, which is in a CSV file.

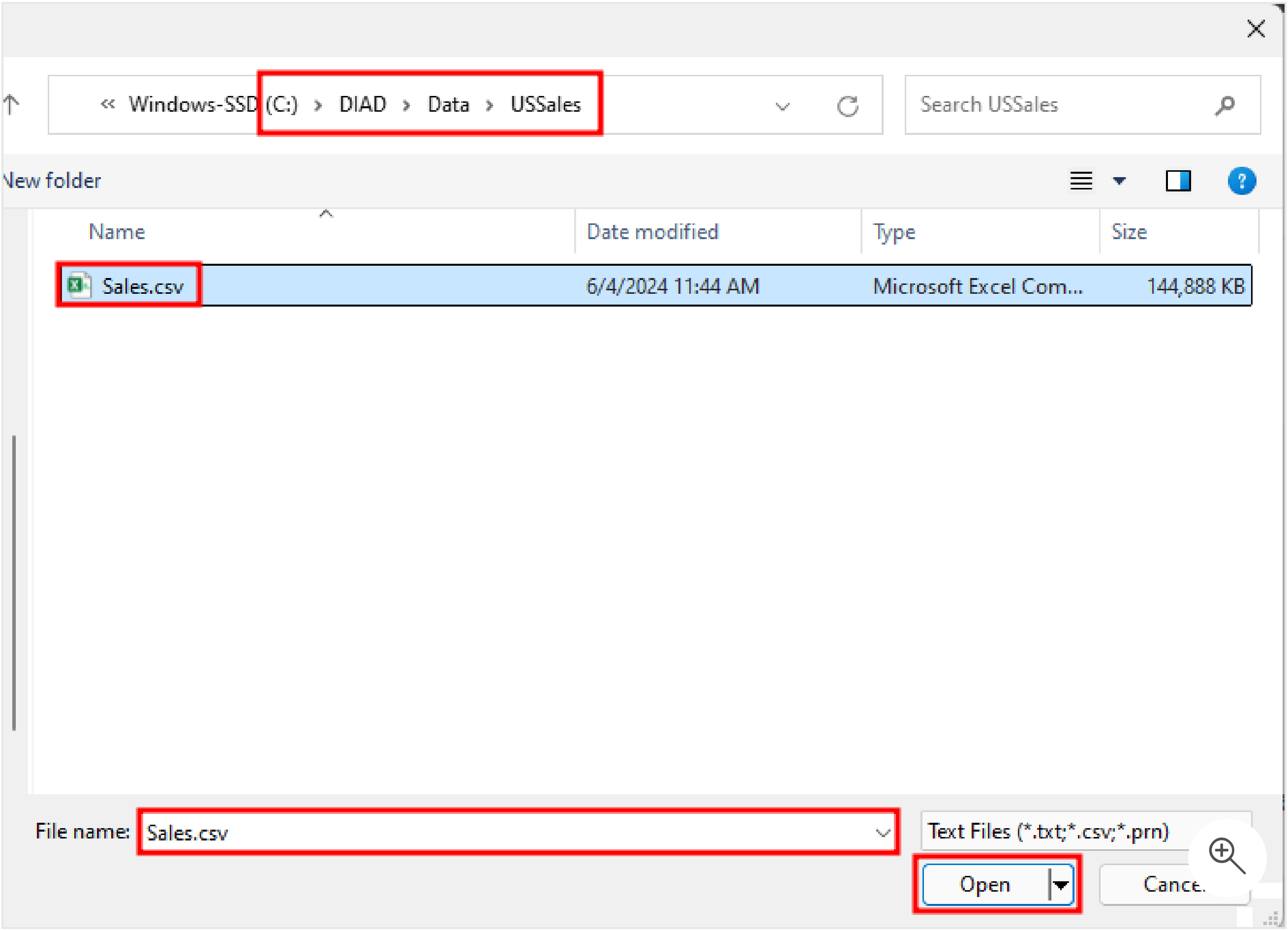
8. From the ribbon at the top of the screen, select the **Home** tab. Then, choose the **Get Data** drop-down (*not the icon*).

9. Select **Text/CSV** from the **Common data sources** list.



10. Browse to the **DIAD** folder (this folder might be called **diad-student-english** if you didn't rename it in Module 1), double-click **Data**, double-click the **USSales** folder, and then select the **Sales.csv** file.

11. Then, select the **Open** button.



Name	Date modified	Type	Size
Sales.csv	6/4/2024 11:44 AM	Microsoft Excel Com...	144,888 KB

File name: Sales.csv

Text Files (*.txt;*.csv;*.prn)

> ⓘ Note
>
> If your folder appears empty then this likely means you forgot to unzip your class files. Navigate to your location where you saved the class files and unzip the files by right-clicking on the .zip file, then select **Extract All**.

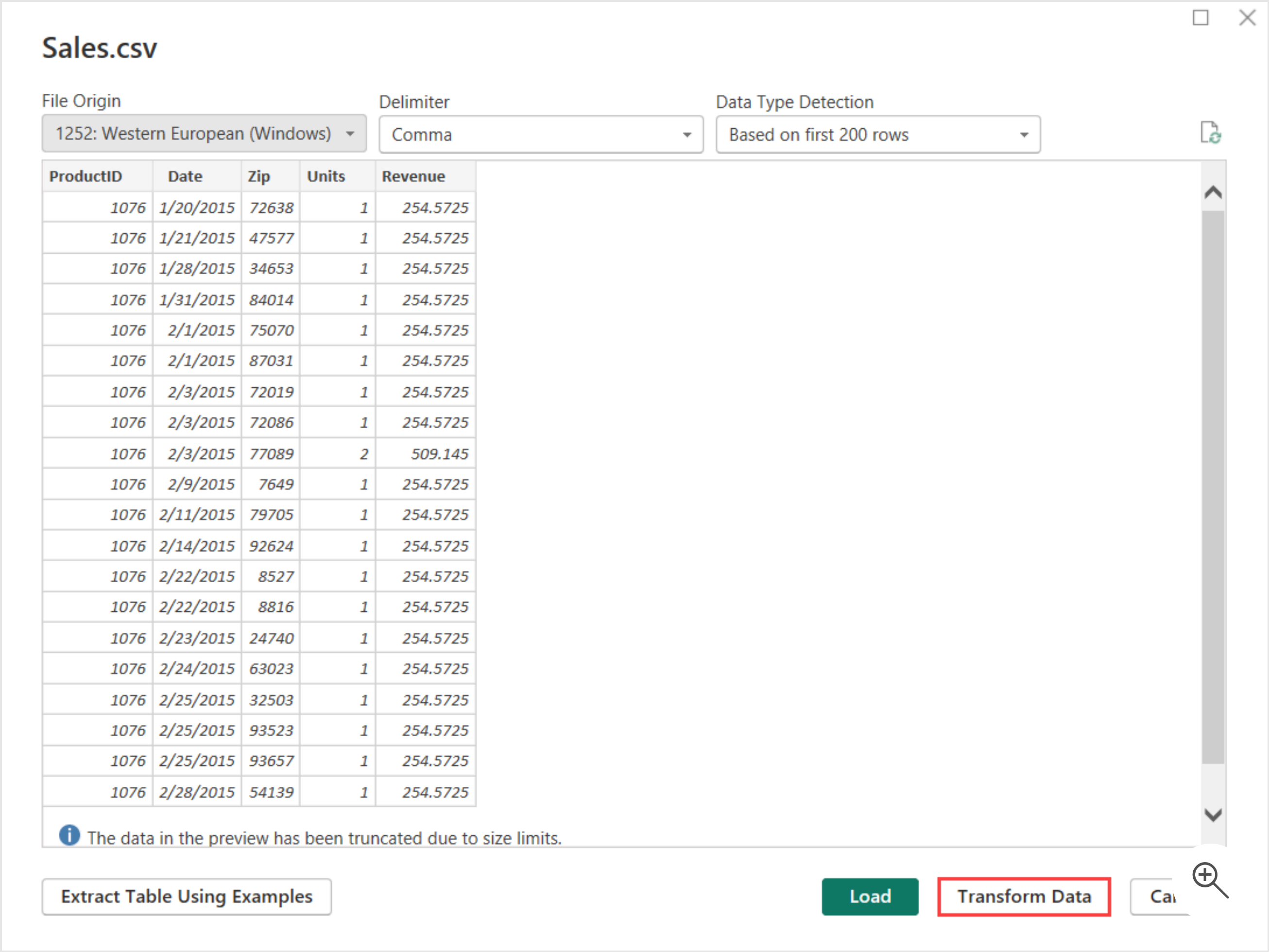
Power BI detects the data type in each column. There are three options for Data Type Detection: based on the first 200 rows, based on the entire dataset, or not detecting the data type. Since our dataset is large and it takes time and resources to scan the complete dataset, we leave the default option of selecting the dataset based on the first 200 rows.

After completing your selection, you have three options: Load, Transform Data, or Cancel.

- **Load** adds the data from the source into Power BI Desktop for you to start creating reports.
- **Transform Data** allows you to perform data shaping operations such as merging columns, adding extra columns, changing data types of columns, and bringing in other data.

- **Cancel** returns you back to the main canvas.

12. In the **Sales.csv** dialog window, select the **Transform Data** button.



Sales.csv

File Origin: 1252: Western European (Windows)

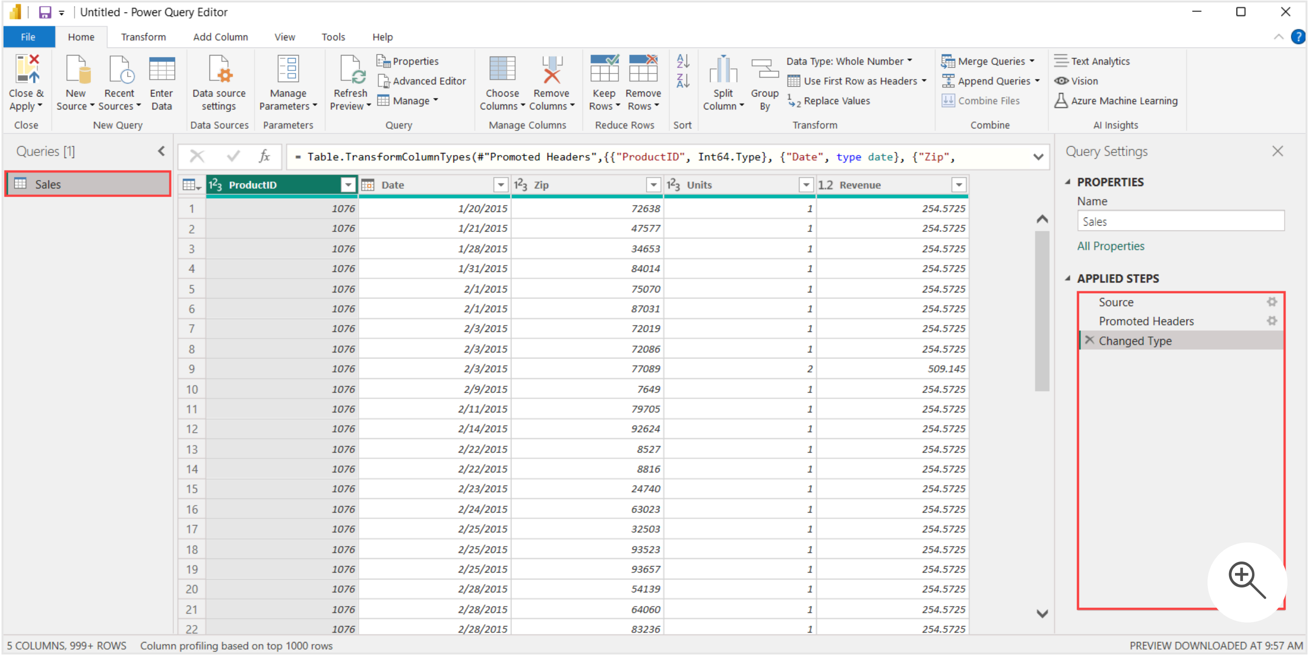
Delimiter: Comma

Data Type Detection: Based on first 200 rows

ProductID	Date	Zip	Units	Revenue
1076	1/20/2015	72638	1	254.5725
1076	1/21/2015	47577	1	254.5725
1076	1/28/2015	34653	1	254.5725
1076	1/31/2015	84014	1	254.5725
1076	2/1/2015	75070	1	254.5725
1076	2/1/2015	87031	1	254.5725
1076	2/3/2015	72019	1	254.5725
1076	2/3/2015	72086	1	254.5725
1076	2/3/2015	77089	2	509.145
1076	2/9/2015	7649	1	254.5725
1076	2/11/2015	79705	1	254.5725
1076	2/14/2015	92624	1	254.5725
1076	2/22/2015	8527	1	254.5725
1076	2/22/2015	8816	1	254.5725
1076	2/23/2015	24740	1	254.5725
1076	2/24/2015	63023	1	254.5725
1076	2/25/2015	32503	1	254.5725
1076	2/25/2015	93523	1	254.5725
1076	2/25/2015	93657	1	254.5725
1076	2/28/2015	54139	1	254.5725

The data in the preview has been truncated due to size limits.

You should be in the **Query Editor** window as shown in the next screenshot. The Query Editor is used to perform data shaping operations. Notice that the sales file you connected shows as a query in the pane to the left of the screen. You can see a preview of the data in the center pane. Power BI predicts the data type of each field (based on the first 200 rows) as indicated by the icons to the right of each column header. In the pane to the right of the screen, steps that the Query Editor performs are recorded in the **APPLIED STEPS** section.



= Table.TransformColumnTypes(#"Promoted Headers",{{"ProductID", Int64.Type}, {"Date", type date}, {"Zip",

	ProductID	Date	Zip	Units	Revenue
1	1076	1/20/2015	72638	1	254.5725
2	1076	1/21/2015	47577	1	254.5725
3	1076	1/28/2015	34653	1	254.5725
4	1076	1/31/2015	84014	1	254.5725
5	1076	2/1/2015	75070	1	254.5725
6	1076	2/1/2015	87031	1	254.5725
7	1076	2/3/2015	72019	1	254.5725
8	1076	2/3/2015	72086	1	254.5725
9	1076	2/3/2015	77089	2	509.145
10	1076	2/9/2015	7649	1	254.5725
11	1076	2/11/2015	79705	1	254.5725
12	1076	2/14/2015	92624	1	254.5725
13	1076	2/22/2015	8527	1	254.5725
14	1076	2/22/2015	8816	1	254.5725
15	1076	2/23/2015	24740	1	254.5725
16	1076	2/24/2015	63023	1	254.5725
17	1076	2/25/2015	32503	1	254.5725
18	1076	2/25/2015	93523	1	254.5725
19	1076	2/25/2015	93657	1	254.5725
20	1076	2/28/2015	54139	1	254.5725
21	1076	2/28/2015	64060	1	254.5725
22	1076	2/28/2015	83236	1	254.5725

5 COLUMNS, 999+ ROWS Column profiling based on top 1000 rows PREVIEW DOWNLOADED AT 9:57 AM

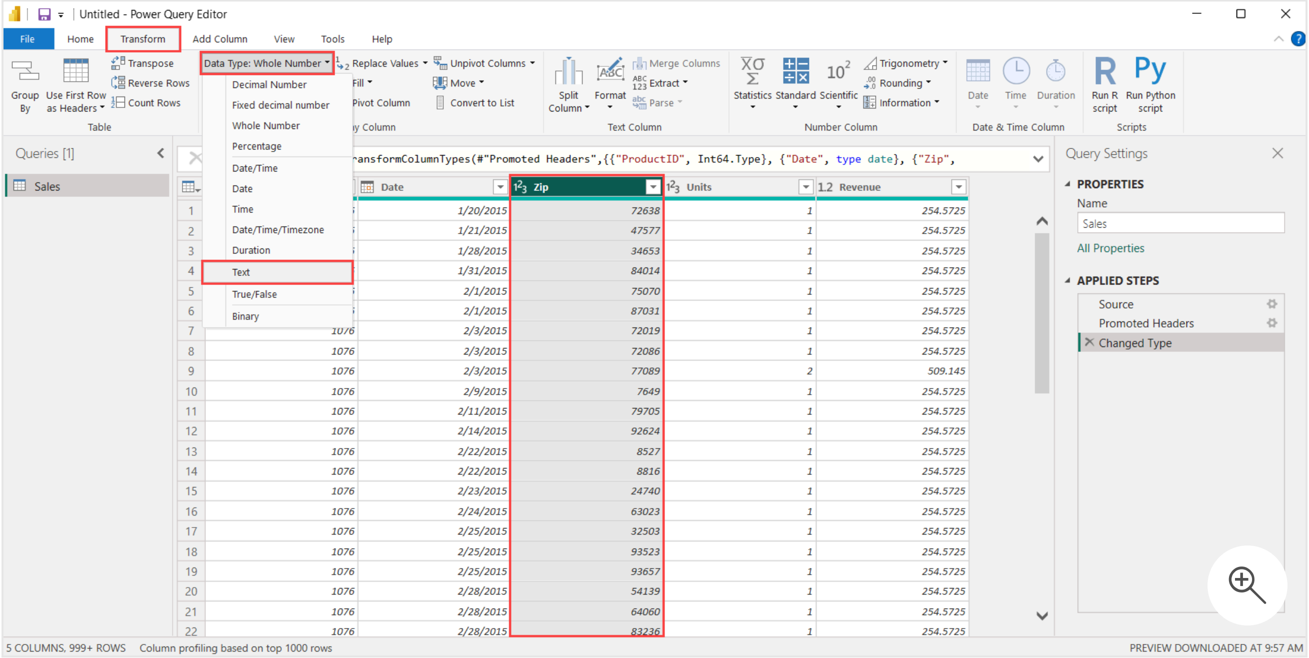
13. Notice that Power BI set the **Zip** column to the data type **Whole Number**. To make sure that the leading zero isn't dropped from Zip codes that start with zero, we format them as **Text**.

    a. To do this, select the **Zip** column.

    b. Then, from the ribbon, select the **Transform** tab.

    c. From the menu at the top of the screen, select the **Data Type** drop-down.

    d. Then choose the **Text** option.



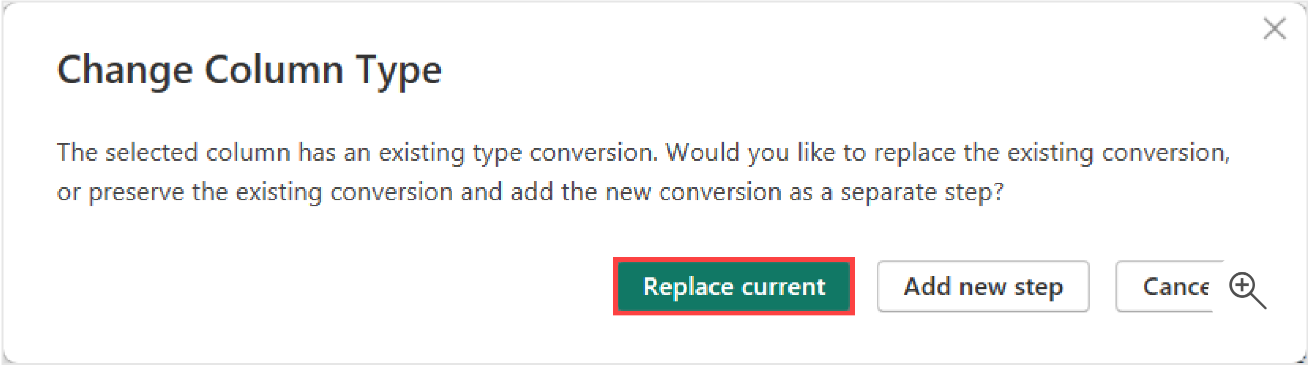
= Table.TransformColumnTypes(#"Promoted Headers",{{"ProductID", Int64.Type}, {"Date", type date}, {"Zip",

		Date	Zip	Units	Revenue
1		1/20/2015	72638	1	254.5725
2		1/21/2015	47577	1	254.5725
3		1/28/2015	34653	1	254.5725
4		1/31/2015	84014	1	254.5725
5		2/1/2015	75070	1	254.5725
6		2/1/2015	87031	1	254.5725
7	1076	2/3/2015	72019	1	254.5725
8	1076	2/3/2015	72086	1	254.5725
9	1076	2/3/2015	77089	2	509.145
10	1076	2/9/2015	7649	1	254.5725
11	1076	2/11/2015	79705	1	254.5725
12	1076	2/14/2015	92624	1	254.5725
13	1076	2/22/2015	8527	1	254.5725
14	1076	2/22/2015	8816	1	254.5725
15	1076	2/23/2015	24740	1	254.5725
16	1076	2/24/2015	63023	1	254.5725
17	1076	2/25/2015	32503	1	254.5725
18	1076	2/25/2015	93523	1	254.5725
19	1076	2/25/2015	93657	1	254.5725
20	1076	2/28/2015	54139	1	254.5725
21	1076	2/28/2015	64060	1	254.5725
22	1076	2/28/2015	83236	1	254.5725

5 COLUMNS, 999+ ROWS Column profiling based on top 1000 rows PREVIEW DOWNLOADED AT 9:57 AM

14. The **Change Column Type** notification box opens. Select the **Replace current** button, which overwrites Power BI's predicted data type.

> ⓘ Important
>
> Missing these last two steps will introduce null values when the Zip field contains both characters and numbers.



Change Column Type

The selected column has an existing type conversion. Would you like to replace the existing conversion, or preserve the existing conversion and add the new conversion as a separate step?

Now that we covered importing data into Power BI Desktop using Power Query, in the next unit, we'll begin the process of loading data from various sources into Power BI.

Next unit: Exercise - Load data from various sources into Power BI

Continue >

✓ 100 XP

Exercise - Load data from various sources into Power BI

10 minutes

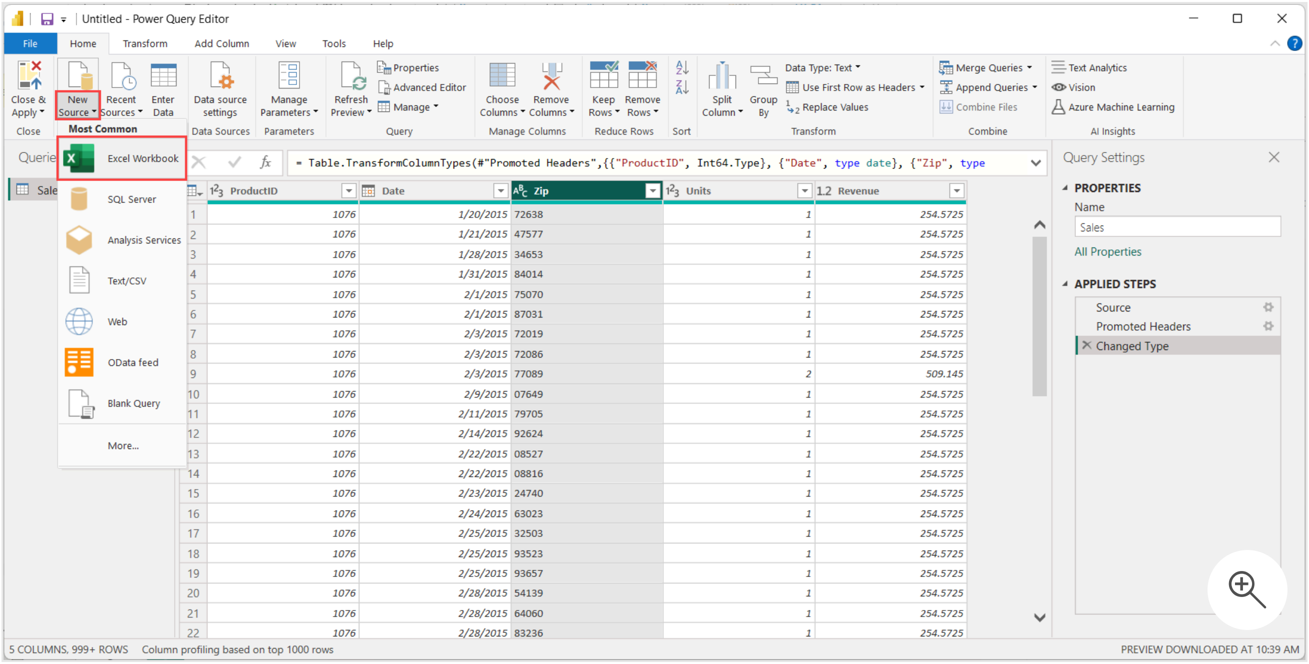
In the previous unit, you were introduced to importing data into Power BI Desktop using Power Query. Now we begin working with various sources, walking through the steps needed to combine these sources into one model. After you learn how to deal with multiple sources, unit 3 will cover how to clean up all this pulled data.

Section 1: Load various sources

Now, let's get the data that's in the Excel source file called **bi-dimensions.xlsx**.

1. From the ribbon at the top of the Power Query Editor, select the **Home** tab.

2. Choose the **New Source** drop-down (*not the icon*), and then select **Excel Workbook**.



```
= Table.TransformColumnTypes(#"Promoted Headers",{{"ProductID", Int64.Type}, {"Date", type date}, {"Zip", type
```

	ProductID	Date	Zip	Units	Revenue
1	1076	1/20/2015	72638	1	254.5725
2	1076	1/21/2015	47577	1	254.5725
3	1076	1/28/2015	34653	1	254.5725
4	1076	1/31/2015	84014	1	254.5725
5	1076	2/1/2015	75070	1	254.5725
6	1076	2/1/2015	87031	1	254.5725
7	1076	2/3/2015	72019	1	254.5725
8	1076	2/3/2015	72086	1	254.5725
9	1076	2/3/2015	77089	2	509.145
10	1076	2/9/2015	07649	1	254.5725
11	1076	2/11/2015	79705	1	254.5725
12	1076	2/14/2015	92624	1	254.5725
13	1076	2/22/2015	08527	1	254.5725
14	1076	2/22/2015	08816	1	254.5725
15	1076	2/23/2015	24740	1	254.5725
16	1076	2/24/2015	63023	1	254.5725
17	1076	2/25/2015	32503	1	254.5725
18	1076	2/25/2015	93523	1	254.5725
19	1076	2/25/2015	93657	1	254.5725
20	1076	2/28/2015	54139	1	254.5725
21	1076	2/28/2015	64060	1	254.5725
22	1076	2/28/2015	83236	1	254.5725

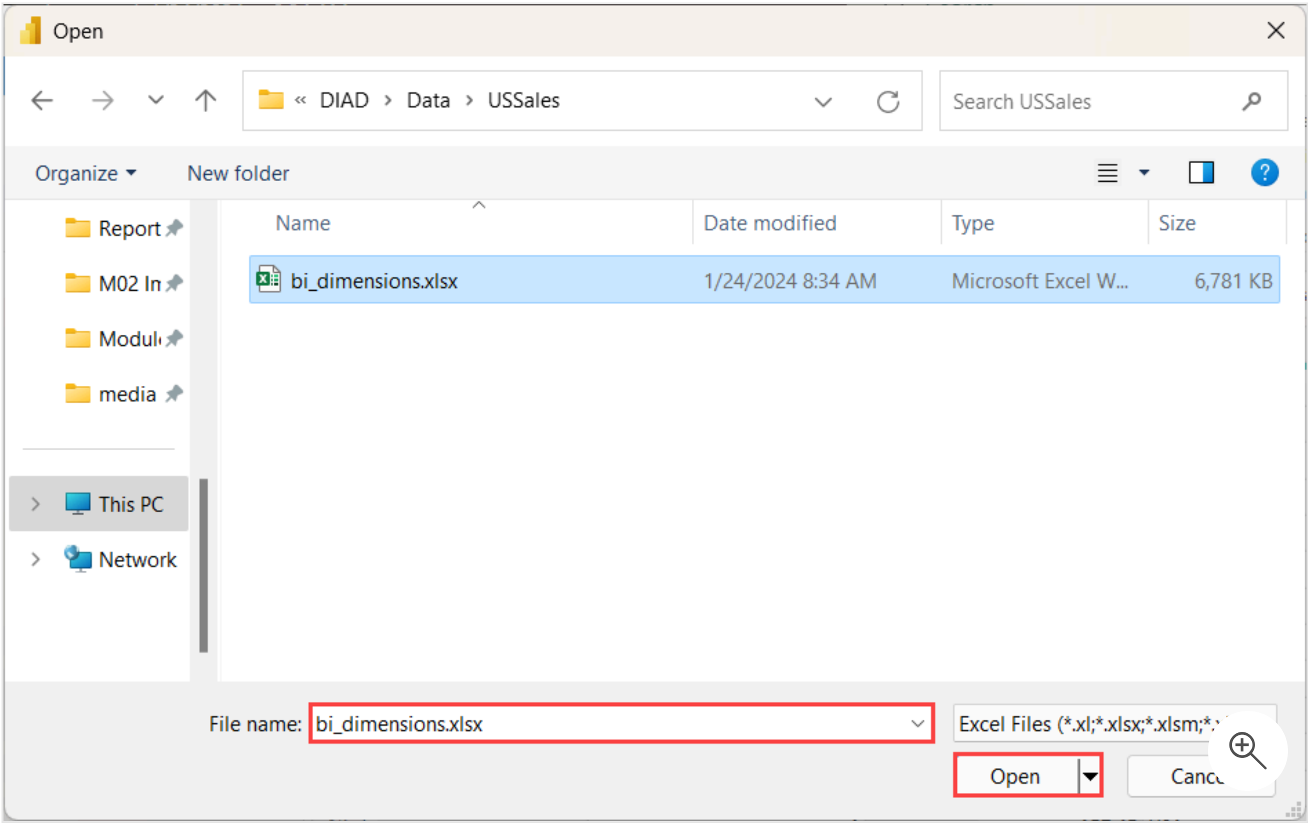
5 COLUMNS, 999+ ROWS Column profiling based on top 1000 rows PREVIEW DOWNLOADED AT 10:39 AM

3. Browse to the **DIAD** folder:

   a. Select **Data**, then the **USSales** folder

b. Next, select the **bi_dimensions.xlsx** file

c. Then select **Open** and the **Navigator** dialog box appears.

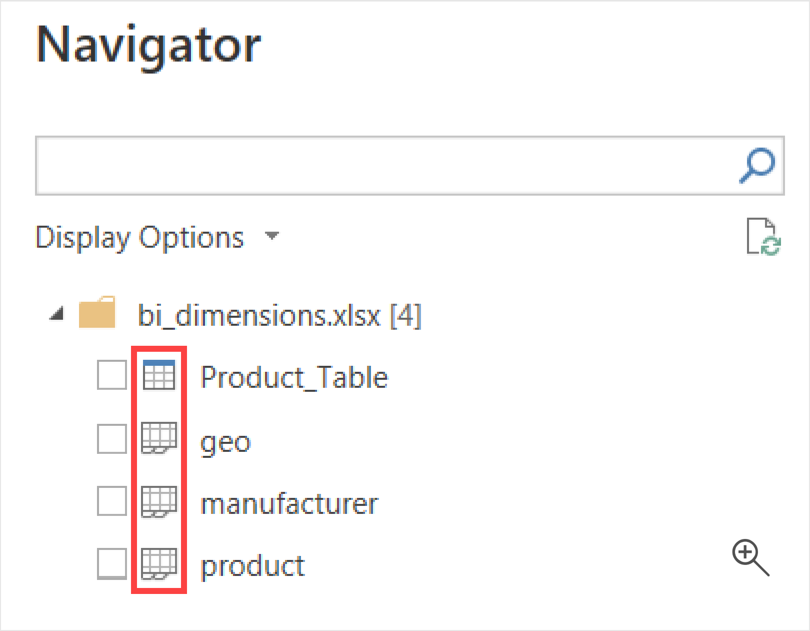


Name	Date modified	Type	Size
bi_dimensions.xlsx	1/24/2024 8:34 AM	Microsoft Excel W...	6,781 KB

File name: bi_dimensions.xlsx

4. The **Navigator** dialog opens. In the list to the left of the dialog, you see three sheets listed that are in the Excel workbook. It also lists **Product_Table**, which is a pre-defined Excel table.

> ⓘ **Note**
>
> Excel Tables are differentiated from worksheets by using different icons.



Navigator

Display Options

bi_dimensions.xlsx [4]

□ Product_Table

□ geo

□ manufacturer

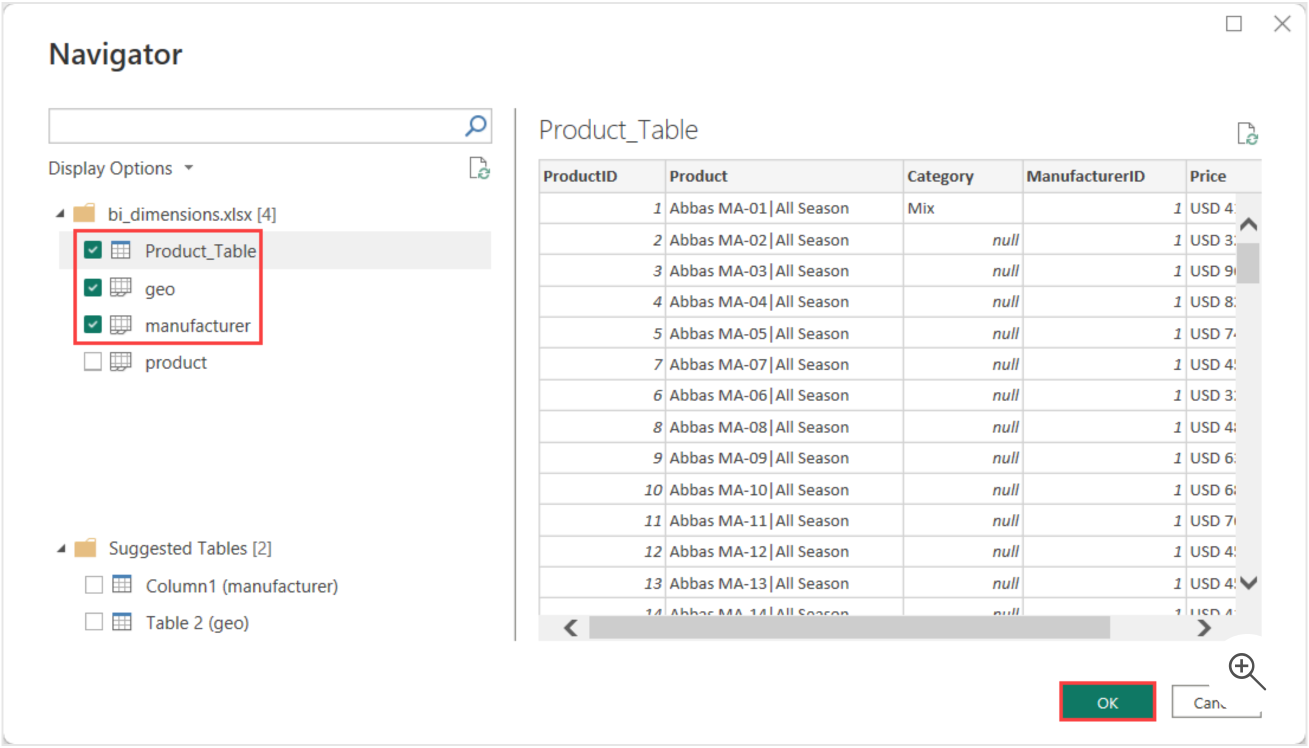
□ product

5. From the list to the left of the dialog, select the checkbox for **geo**. In the preview pane, notice that the first few rows are headers and aren't part of the data. We remove them shortly.

6. Select the checkbox for **manufacturer**. In the preview pane, notice that the last couple of rows are footers and aren't part of the data. We remove them shortly.

7. Select the checkbox for **Product_Table**. Notice that the different icon indicates this data is stored in an Excel table.

8. Make sure that **Product_Table**, **geo** and **manufacturer** are selected in the pane to the left, and then select **OK**.



Navigator

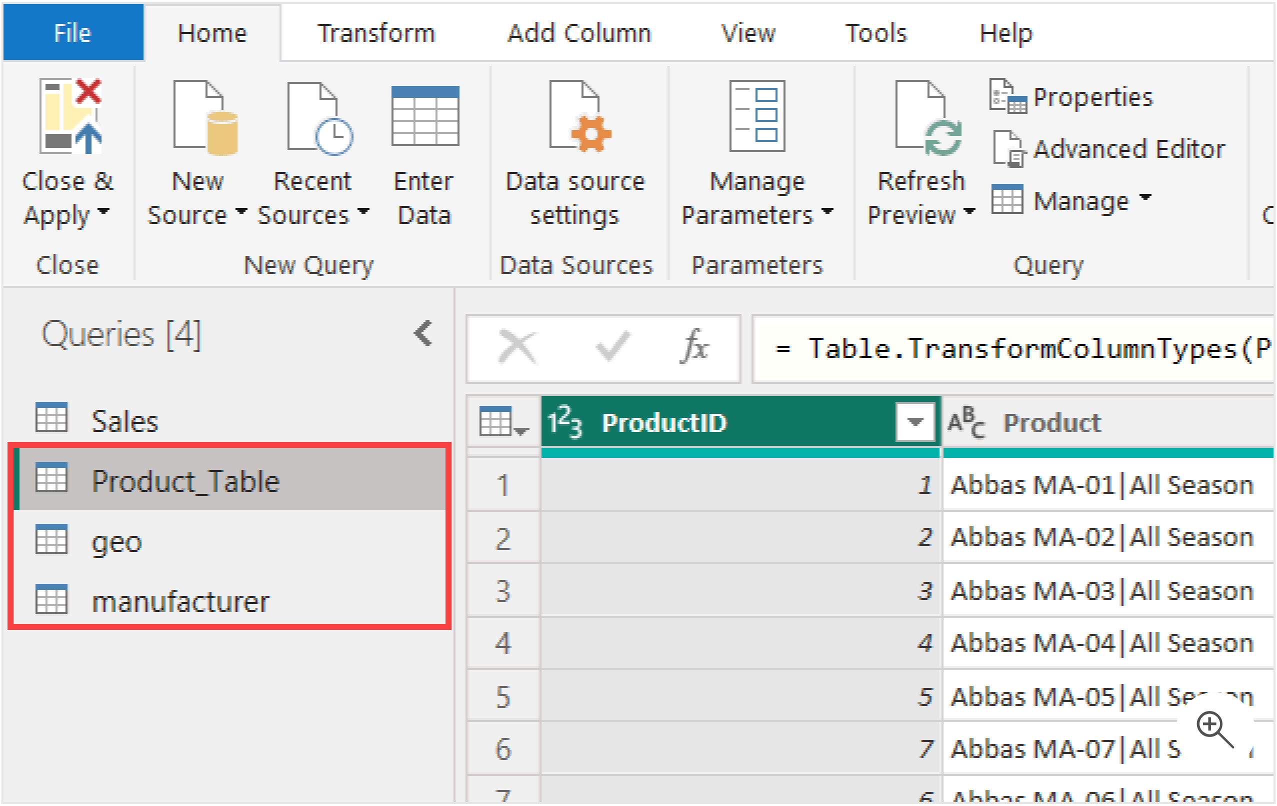
Display Options

- bi_dimensions.xlsx [4]
  - ☑ Product_Table
  - ☑ geo
  - ☑ manufacturer
  - □ product
- Suggested Tables [2]
  - □ Column1 (manufacturer)
  - □ Table 2 (geo)

Product_Table

ProductID	Product	Category	ManufacturerID	Price
1	Abbas MA-01|All Season	Mix	1	USD 4
2	Abbas MA-02|All Season	null	1	USD 3
3	Abbas MA-03|All Season	null	1	USD 9
4	Abbas MA-04|All Season	null	1	USD 8
5	Abbas MA-05|All Season	null	1	USD 7
7	Abbas MA-07|All Season	null	1	USD 4
6	Abbas MA-06|All Season	null	1	USD 3
8	Abbas MA-08|All Season	null	1	USD 4
9	Abbas MA-09|All Season	null	1	USD 6
10	Abbas MA-10|All Season	null	1	USD 6
11	Abbas MA-11|All Season	null	1	USD 7
12	Abbas MA-12|All Season	null	1	USD 4
13	Abbas MA-13|All Season	null	1	USD 4
14	Abbas MA-14|All Season	null	1	USD 4

9. Notice that three sheets are added as queries in the Query Editor: *Product_Table*, *geo*, and *manufacturer*.



Queries [4]

- Sales
- Product_Table
- geo
- manufacturer

```
= Table.TransformColumnTypes(P
```

	ProductID	Product
1	1	Abbas MA-01|All Season
2	2	Abbas MA-02|All Season
3	3	Abbas MA-03|All Season
4	4	Abbas MA-04|All Season
5	5	Abbas MA-05|All Season
6	7	Abbas MA-07|All S
7	6	Abbas MA-06|All Season

Section 2: Add other data

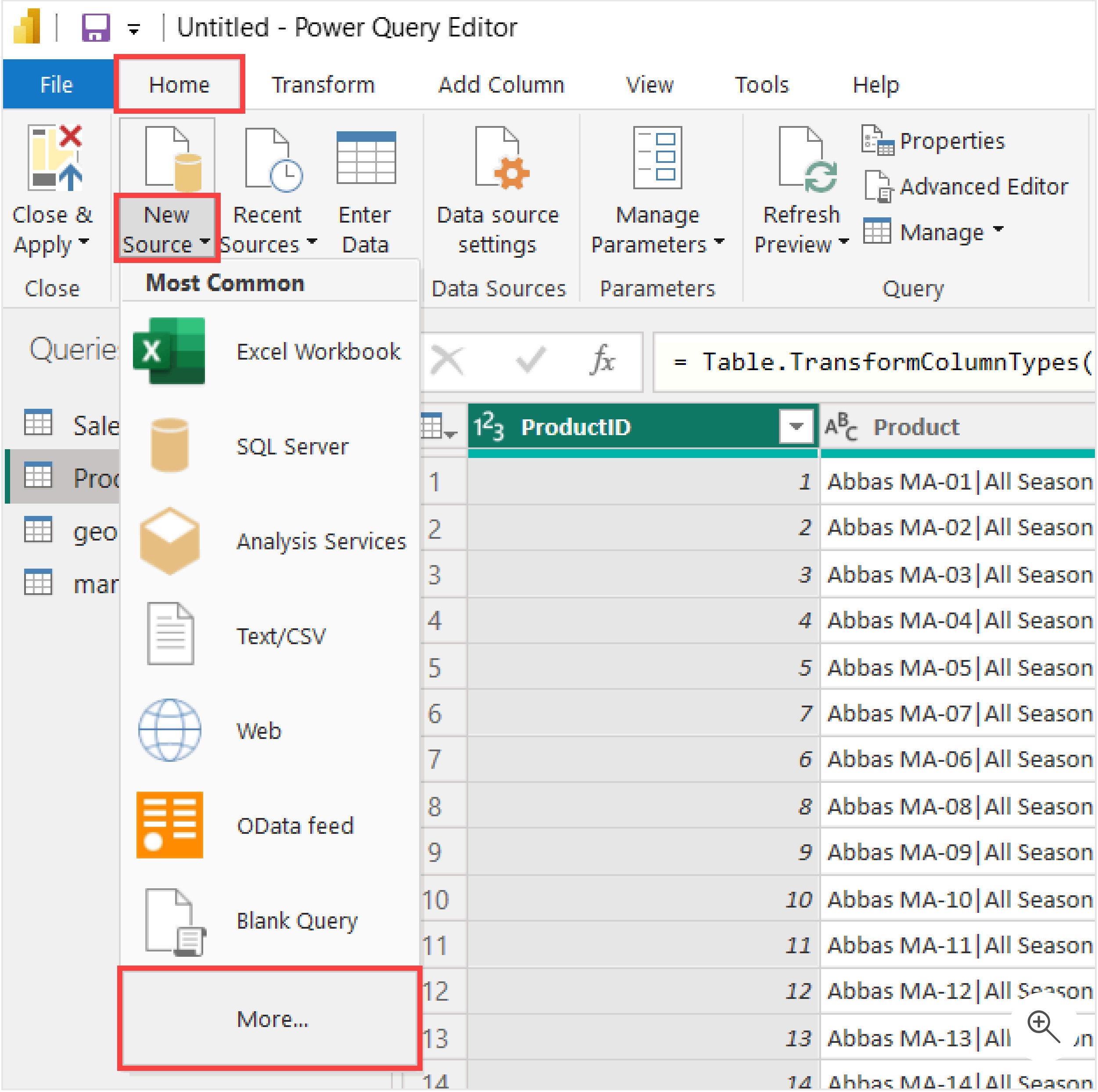
In this scenario, the international subsidiaries agree to provide their sales data so that the company's sales can be analyzed together. You created a folder where they each put their data.

To analyze all the data together, you need to import the new data from each of the subsidiaries and combine it with the US Sales you loaded earlier.

When you loaded the US sales data earlier in this unit, you did so one at a time. However, Power BI Gives you the option to load all the files in a folder together at once. This helps save you some time when you load data.

1. From the **Home** tab of the Query Editor, select the **New Source** drop-down (*not the icon*).

2. Select **More..** from the options list. The **Get Data** dialog opens.

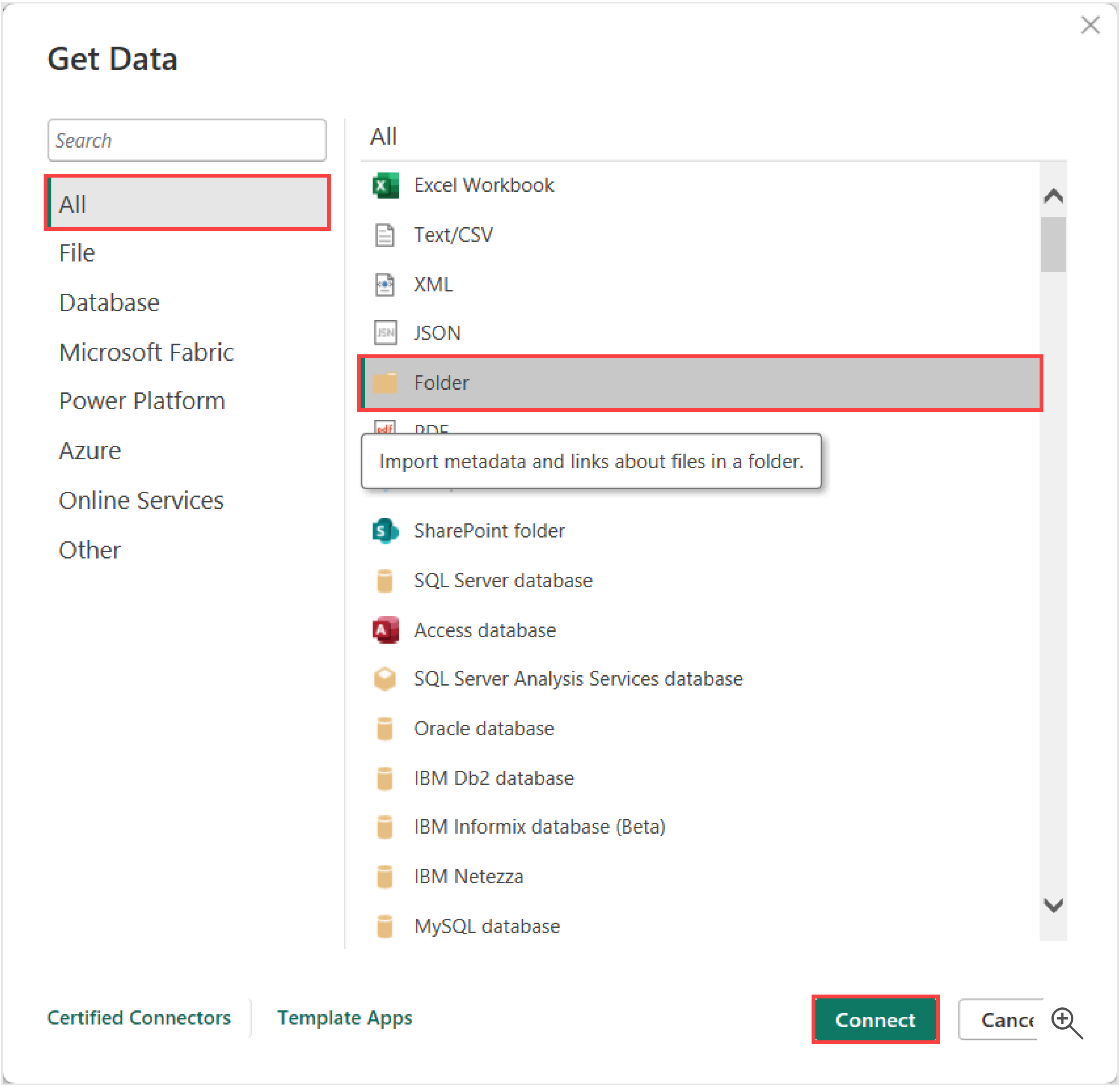


```
= Table.TransformColumnTypes(
```

	ProductID	Product
1	1	Abbas MA-01|All Season
2	2	Abbas MA-02|All Season
3	3	Abbas MA-03|All Season
4	4	Abbas MA-04|All Season
5	5	Abbas MA-05|All Season
6	7	Abbas MA-07|All Season
7	6	Abbas MA-06|All Season
8	8	Abbas MA-08|All Season
9	9	Abbas MA-09|All Season
10	10	Abbas MA-10|All Season
11	11	Abbas MA-11|All Season
12	12	Abbas MA-12|All Season
13	13	Abbas MA-13|All
14	14	Abbas MA-14|All Season

3. In the **Get Data** dialog box, select **Folder** from the **All** list.

4. Then, select the **Connect** button and the **Folder** dialog box opens.



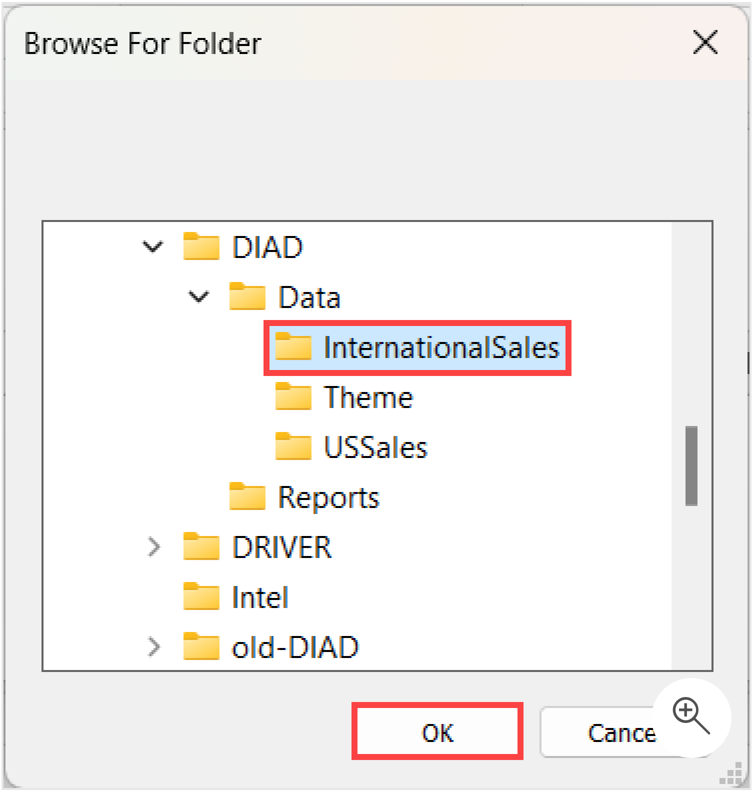
5. In the Folder dialog box, select the **Browse..** button.

6. In the **Browse For Folder** dialog, navigate to the location where you unzipped the class files.

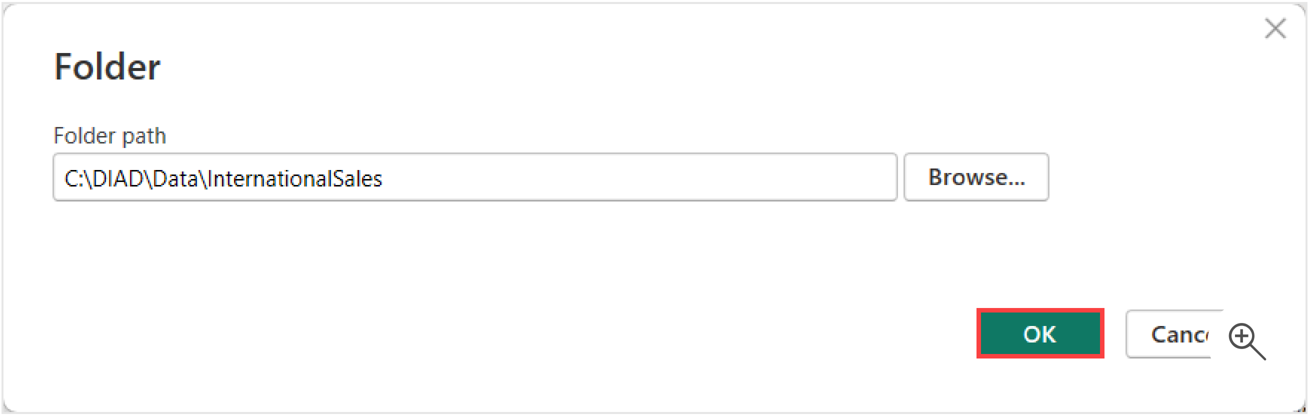
7. Open the **DIAD** folder, then open the **Data** folder.

8. Select the **InternationalSales** folder.

9. Select **OK** to close the **Browse for Folder** dialog box.



10. Then, select **OK** to close the **Folder** dialog box. The selected folder dialog box displays the list of files in the folder.

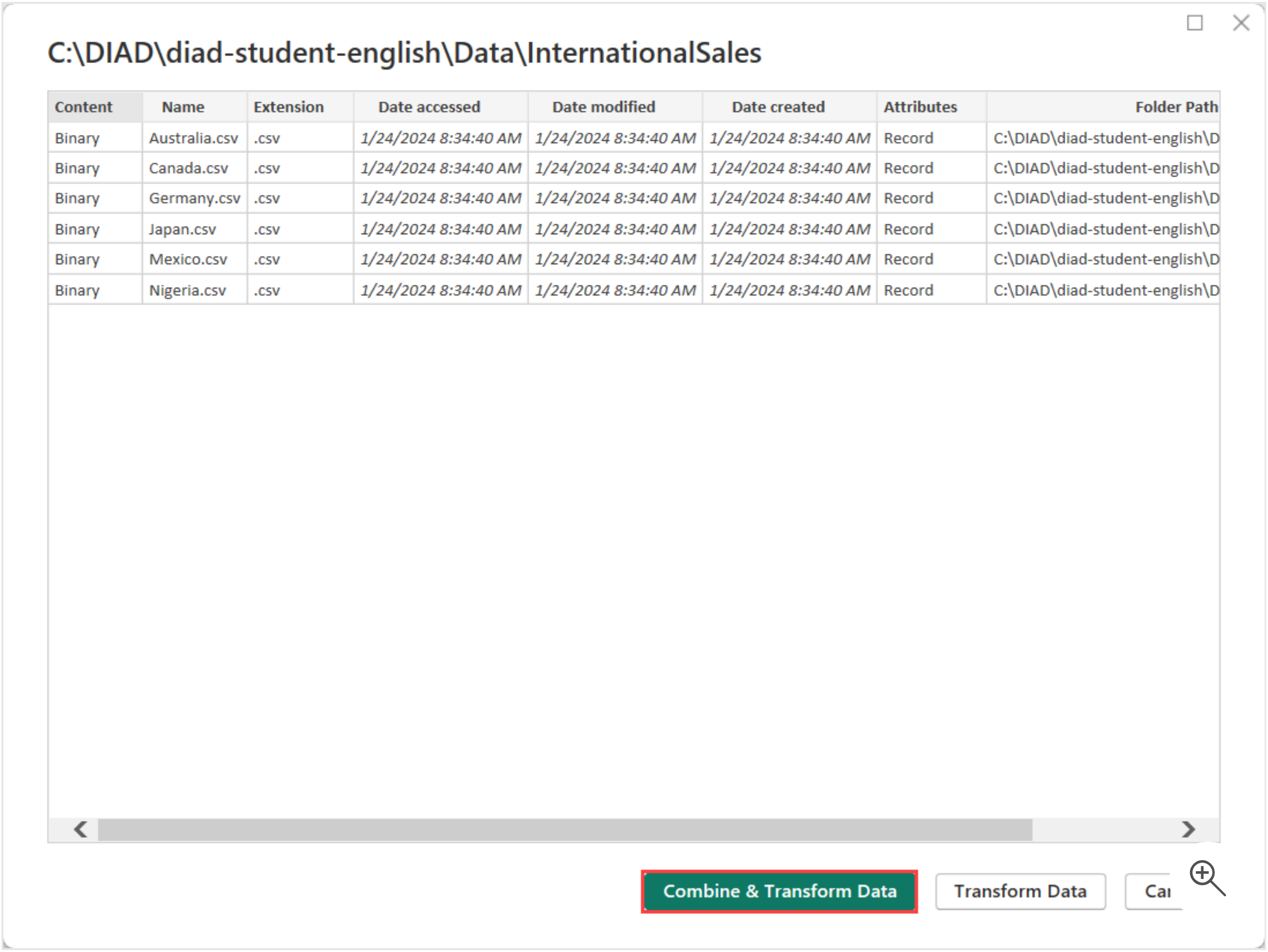


Folder path: C:\DIAD\Data\InternationalSales

ⓘ **Note**

This approach will load all the files located in the folder. This is useful when you have a group that puts files on an FTP (File Transfer Protocol) site each month and you are not always sure of the names of the files or the number of files. All the files must be of the same file type with columns in the same order.

11. Select the **Combine & Transform Data** button at the bottom of the dialog box.



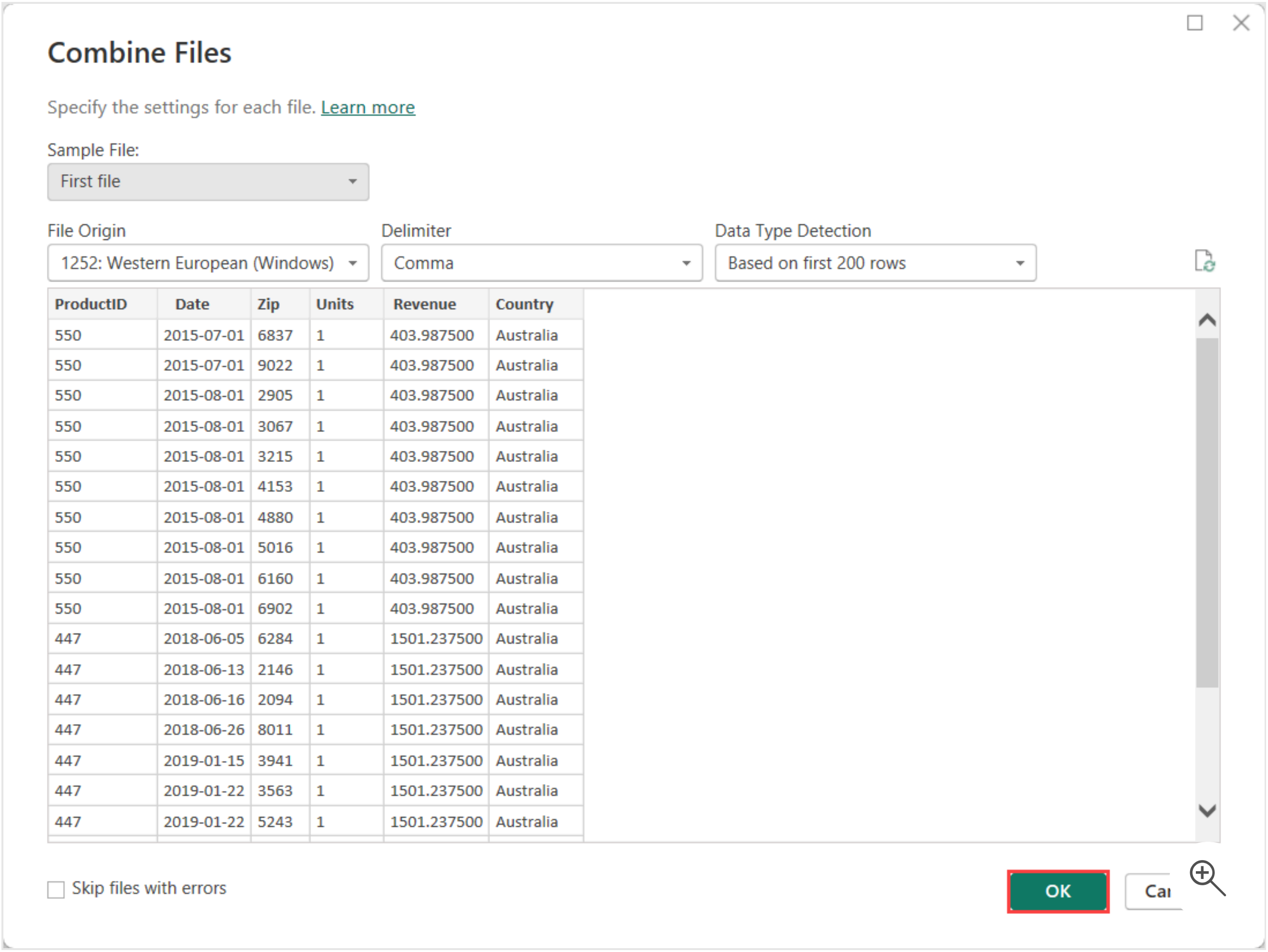
C:\DIAD\diad-student-english\Data\InternationalSales

Content	Name	Extension	Date accessed	Date modified	Date created	Attributes	Folder Path
Binary	Australia.csv	.csv	*1/24/2024 8:34:40 AM*	*1/24/2024 8:34:40 AM*	*1/24/2024 8:34:40 AM*	Record	C:\DIAD\diad-student-english\D
Binary	Canada.csv	.csv	*1/24/2024 8:34:40 AM*	*1/24/2024 8:34:40 AM*	*1/24/2024 8:34:40 AM*	Record	C:\DIAD\diad-student-english\D
Binary	Germany.csv	.csv	*1/24/2024 8:34:40 AM*	*1/24/2024 8:34:40 AM*	*1/24/2024 8:34:40 AM*	Record	C:\DIAD\diad-student-english\D
Binary	Japan.csv	.csv	*1/24/2024 8:34:40 AM*	*1/24/2024 8:34:40 AM*	*1/24/2024 8:34:40 AM*	Record	C:\DIAD\diad-student-english\D
Binary	Mexico.csv	.csv	*1/24/2024 8:34:40 AM*	*1/24/2024 8:34:40 AM*	*1/24/2024 8:34:40 AM*	Record	C:\DIAD\diad-student-english\D
Binary	Nigeria.csv	.csv	*1/24/2024 8:34:40 AM*	*1/24/2024 8:34:40 AM*	*1/24/2024 8:34:40 AM*	Record	C:\DIAD\diad-student-english\D

The **Combine Files** dialog box opens. By default, Power BI again detects the data type based on the first 200 rows. Notice there's an option to select various file Delimiters. The file we're working with is Comma delimited, so let's leave the default **Delimiter** option as **Comma**.

There's also an option to select each individual file in the folder (using the **Sample File** drop-down) to validate the format of the files.

12. Select the **OK** button located at the bottom of the dialog window.



Combine Files

Specify the settings for each file. Learn more

Sample File: First file

File Origin: 1252: Western European (Windows)

Delimiter: Comma

Data Type Detection: Based on first 200 rows

ProductID	Date	Zip	Units	Revenue	Country
550	2015-07-01	6837	1	403.987500	Australia
550	2015-07-01	9022	1	403.987500	Australia
550	2015-08-01	2905	1	403.987500	Australia
550	2015-08-01	3067	1	403.987500	Australia
550	2015-08-01	3215	1	403.987500	Australia
550	2015-08-01	4153	1	403.987500	Australia
550	2015-08-01	4880	1	403.987500	Australia
550	2015-08-01	5016	1	403.987500	Australia
550	2015-08-01	6160	1	403.987500	Australia
550	2015-08-01	6902	1	403.987500	Australia
447	2018-06-05	6284	1	1501.237500	Australia
447	2018-06-13	2146	1	1501.237500	Australia
447	2018-06-16	2094	1	1501.237500	Australia
447	2018-06-26	8011	1	1501.237500	Australia
447	2019-01-15	3941	1	1501.237500	Australia
447	2019-01-22	3563	1	1501.237500	Australia
447	2019-01-22	5243	1	1501.237500	Australia

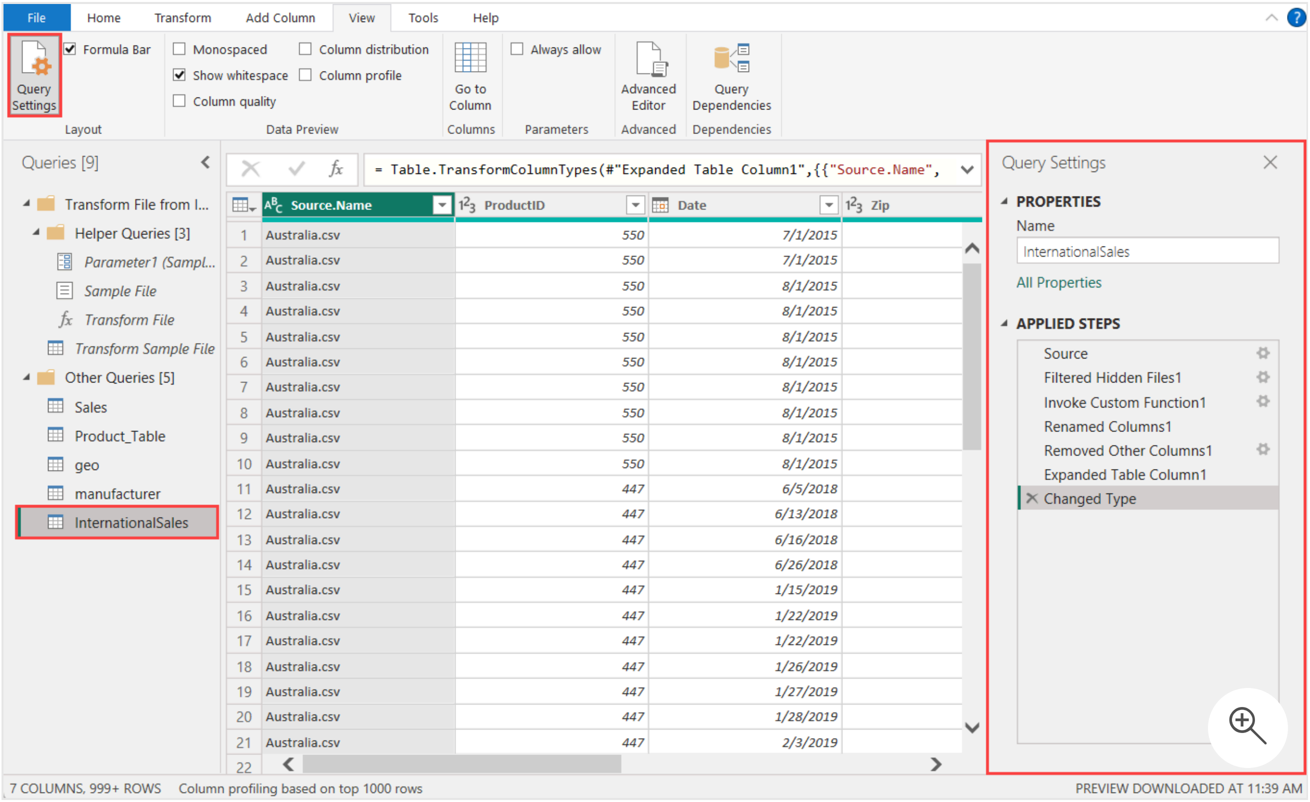
□ Skip files with errors

You're now in the **Power Query Editor** window with a new query named **InternationalSales**.

13. If you don't see the **Queries** pane to the left of the screen, select the > (greater than) icon to expand the pane.

14. If you don't see the **Query Settings** pane on the right of the screen, select the **View** tab in the ribbon and choose **Query Settings** to view the pane.

15. Select **InternationalSales** from the query pane on the left.



☑ Formula Bar
□ Monospaced
☑ Show whitespace
□ Column quality
□ Column distribution
□ Column profile
□ Always allow

- Transform File from I...
  - Helper Queries [3]
    - Parameter1 (Sampl...
    - Sample File
    - Transform File
  - Transform Sample File
- Other Queries [5]
  - Sales
  - Product_Table
  - geo
  - manufacturer
  - InternationalSales

```
= Table.TransformColumnTypes(#"Expanded Table Column1",{{"Source.Name",
```

	Source.Name	ProductID	Date	Zip
1	Australia.csv	550	7/1/2015	
2	Australia.csv	550	7/1/2015	
3	Australia.csv	550	8/1/2015	
4	Australia.csv	550	8/1/2015	
5	Australia.csv	550	8/1/2015	
6	Australia.csv	550	8/1/2015	
7	Australia.csv	550	8/1/2015	
8	Australia.csv	550	8/1/2015	
9	Australia.csv	550	8/1/2015	
10	Australia.csv	550	8/1/2015	
11	Australia.csv	447	6/5/2018	
12	Australia.csv	447	6/13/2018	
13	Australia.csv	447	6/16/2018	
14	Australia.csv	447	6/26/2018	
15	Australia.csv	447	1/15/2019	
16	Australia.csv	447	1/22/2019	
17	Australia.csv	447	1/22/2019	
18	Australia.csv	447	1/26/2019	
19	Australia.csv	447	1/27/2019	
20	Australia.csv	447	1/28/2019	
21	Australia.csv	447	2/3/2019	
22				

Name: InternationalSales

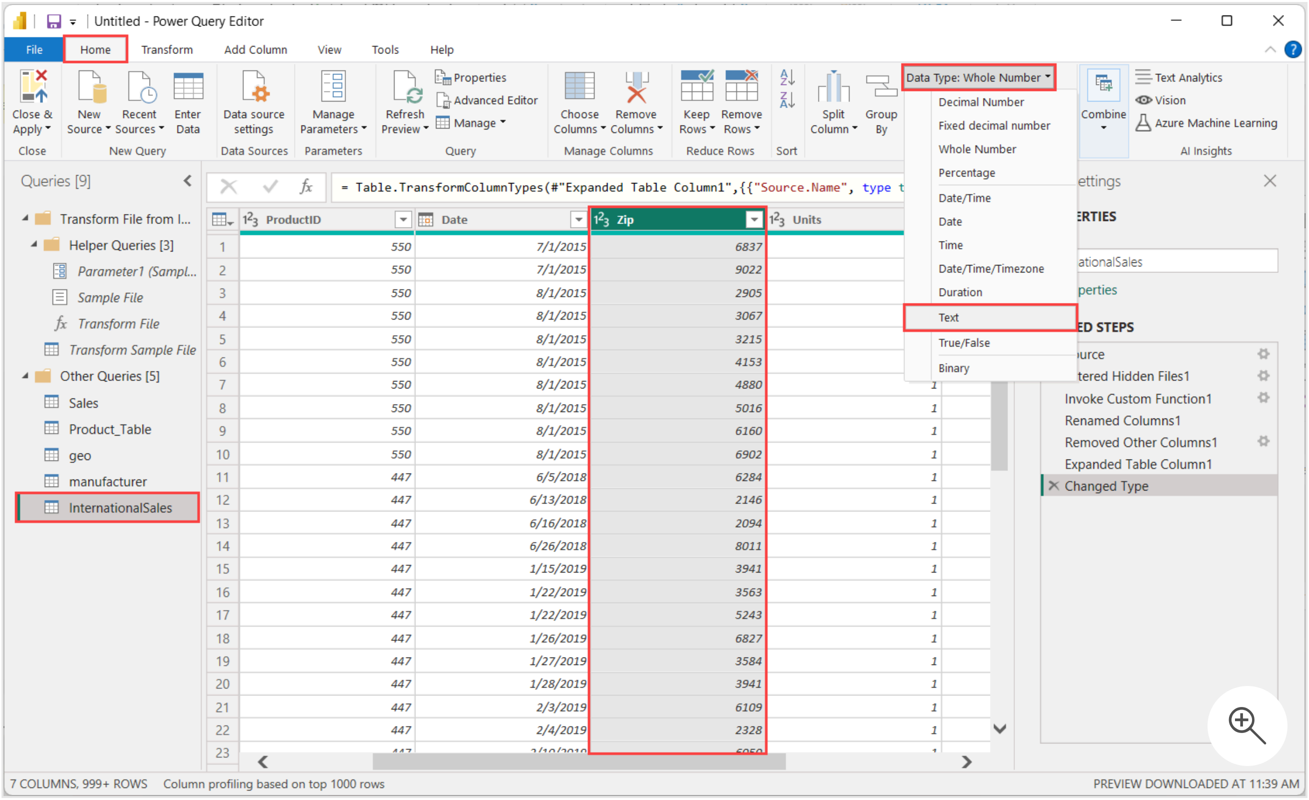
- Source
- Filtered Hidden Files1
- Invoke Custom Function1
- Renamed Columns1
- Removed Other Columns1
- Expanded Table Column1
- Changed Type

7 COLUMNS, 999+ ROWS Column profiling based on top 1000 rows

PREVIEW DOWNLOADED AT 11:39 AM

Notice that the **Zip** column is of the **Whole Number** type. Based on the first 200 rows, Power BI thinks the Zip column consists of whole numbers. But zip code could be alpha numeric in some countries or regions or contain leading zeros. If we don't change the data type, we receive an error when we load the data shortly. So, let's change the Zip column to data type **Text**.

16. Select the **Zip** column in the **InternationalSales** query, and then change the **Data Type** to **Text** using the drop-down under the **Home** tab.



```
= Table.TransformColumnTypes(#"Expanded Table Column1",{{"Source.Name", type t
```

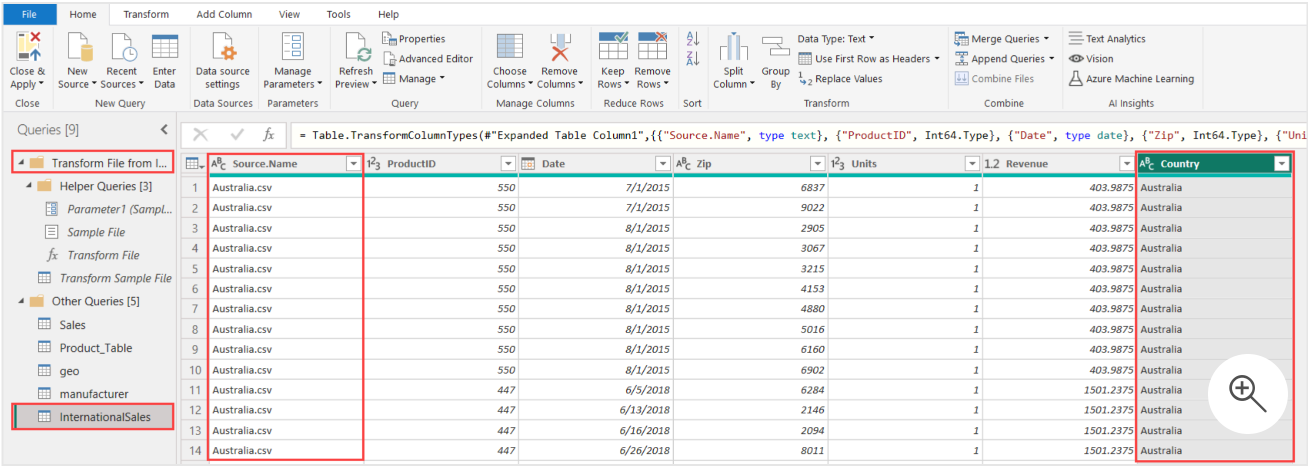
	ProductID	Date	Zip	Units
1	550	7/1/2015	6837	
2	550	7/1/2015	9022	
3	550	8/1/2015	2905	
4	550	8/1/2015	3067	
5	550	8/1/2015	3215	
6	550	8/1/2015	4153	
7	550	8/1/2015	4880	
8	550	8/1/2015	5016	1
9	550	8/1/2015	6160	1
10	550	8/1/2015	6902	1
11	447	6/5/2018	6284	1
12	447	6/13/2018	2146	1
13	447	6/16/2018	2094	1
14	447	6/26/2018	8011	1
15	447	1/15/2019	3941	1
16	447	1/22/2019	3563	1
17	447	1/22/2019	5243	1
18	447	1/26/2019	6827	1
19	447	1/27/2019	3584	1
20	447	1/28/2019	3941	1
21	447	2/3/2019	6109	1
22	447	2/4/2019	2328	1

7 COLUMNS, 999+ ROWS Column profiling based on top 1000 rows PREVIEW DOWNLOADED AT 11:39 AM

17. The **Change Column Type** dialog box opens. Select the **Replace Current** button when prompted.

    In the **Queries** pane, notice that a **Transform File from the InternationalSales** folder is created. This contains the function used to load each of the files from the folder.

    If you compare the **InternationalSales** and the **Sales** table, you see the **InternationalSales** table contains two new columns: **Source.Name** and **Country**.



```
= Table.TransformColumnTypes(#"Expanded Table Column1",{{"Source.Name", type text}, {"ProductID", Int64.Type}, {"Date", type date}, {"Zip", Int64.Type}, {"Uni
```

	Source.Name	ProductID	Date	Zip	Units	Revenue	Country
1	Australia.csv	550	7/1/2015	6837	1	403.9875	Australia
2	Australia.csv	550	7/1/2015	9022	1	403.9875	Australia
3	Australia.csv	550	8/1/2015	2905	1	403.9875	Australia
4	Australia.csv	550	8/1/2015	3067	1	403.9875	Australia
5	Australia.csv	550	8/1/2015	3215	1	403.9875	Australia
6	Australia.csv	550	8/1/2015	4153	1	403.9875	Australia
7	Australia.csv	550	8/1/2015	4880	1	403.9875	Australia
8	Australia.csv	550	8/1/2015	5016	1	403.9875	Australia
9	Australia.csv	550	8/1/2015	6160	1	403.9875	Australia
10	Australia.csv	550	8/1/2015	6902	1	403.9875	Australia
11	Australia.csv	447	6/5/2018	6284	1	1501.2375	Australia
12	Australia.csv	447	6/13/2018	2146	1	1501.2375	Australia
13	Australia.csv	447	6/16/2018	2094	1	1501.2375	Australia
14	Australia.csv	447	6/26/2018	8011	1	1501.2375	Australia

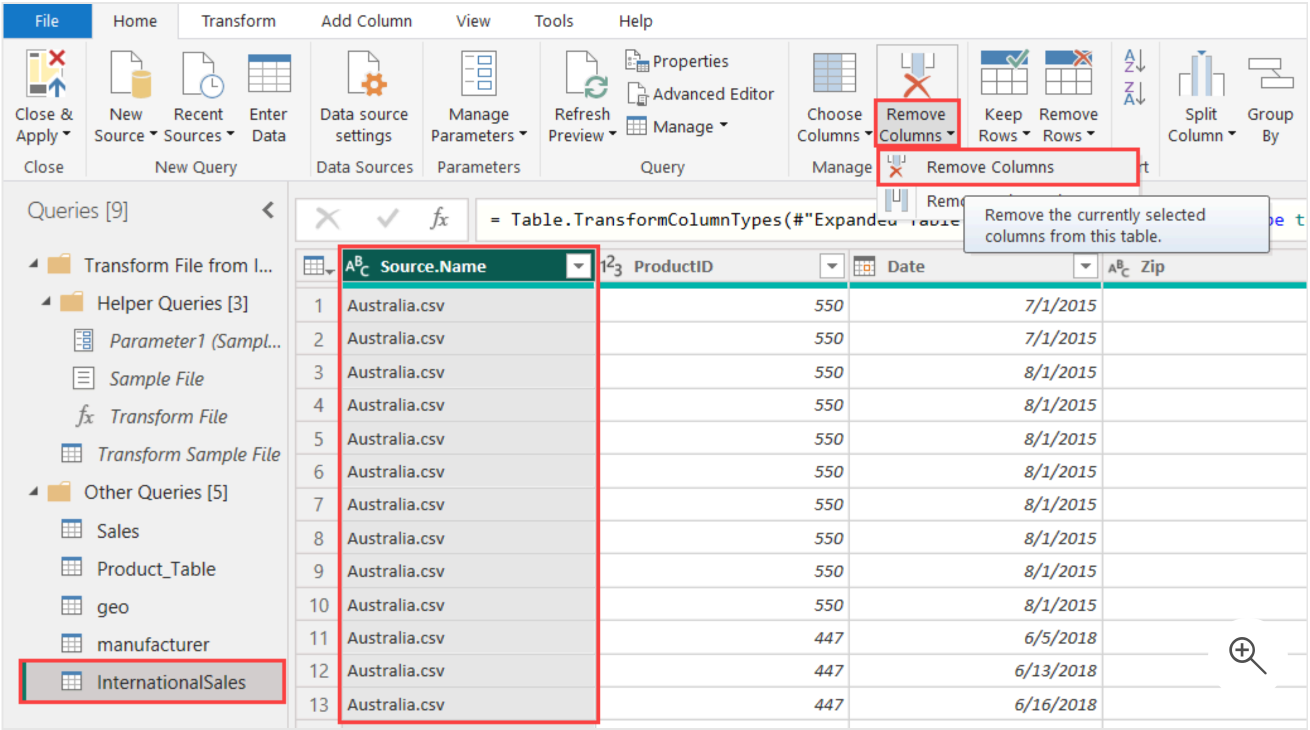
18. We don't need the **Source.Name** column in the **InternationalSales** query. To remove the column from the query:

    a. Select the **Source.Name** column.

b. select the **Home** tab from the ribbon.

c. Choose the **Remove Columns** drop-down.

d. Now, select **Remove Columns** again.



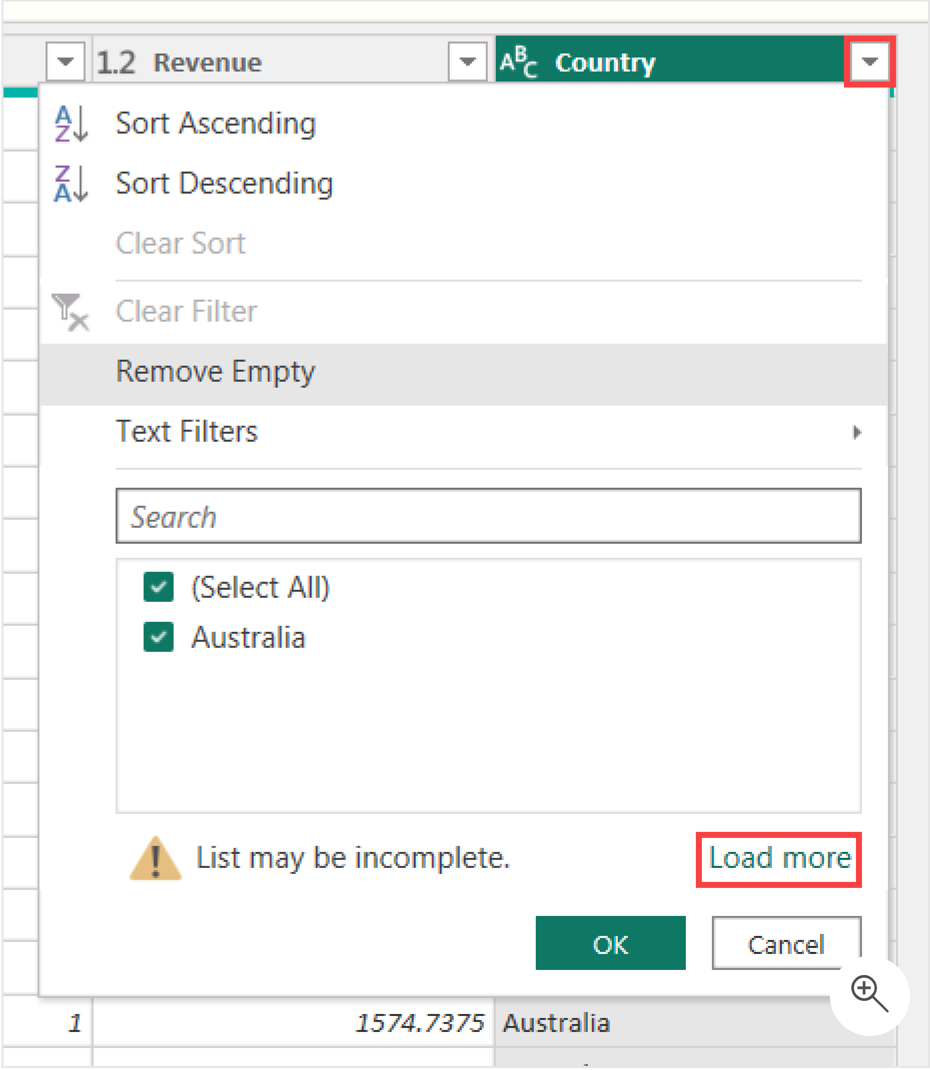
Remove the currently selected columns from this table.

= Table.TransformColumnTypes(#"Expanded Table

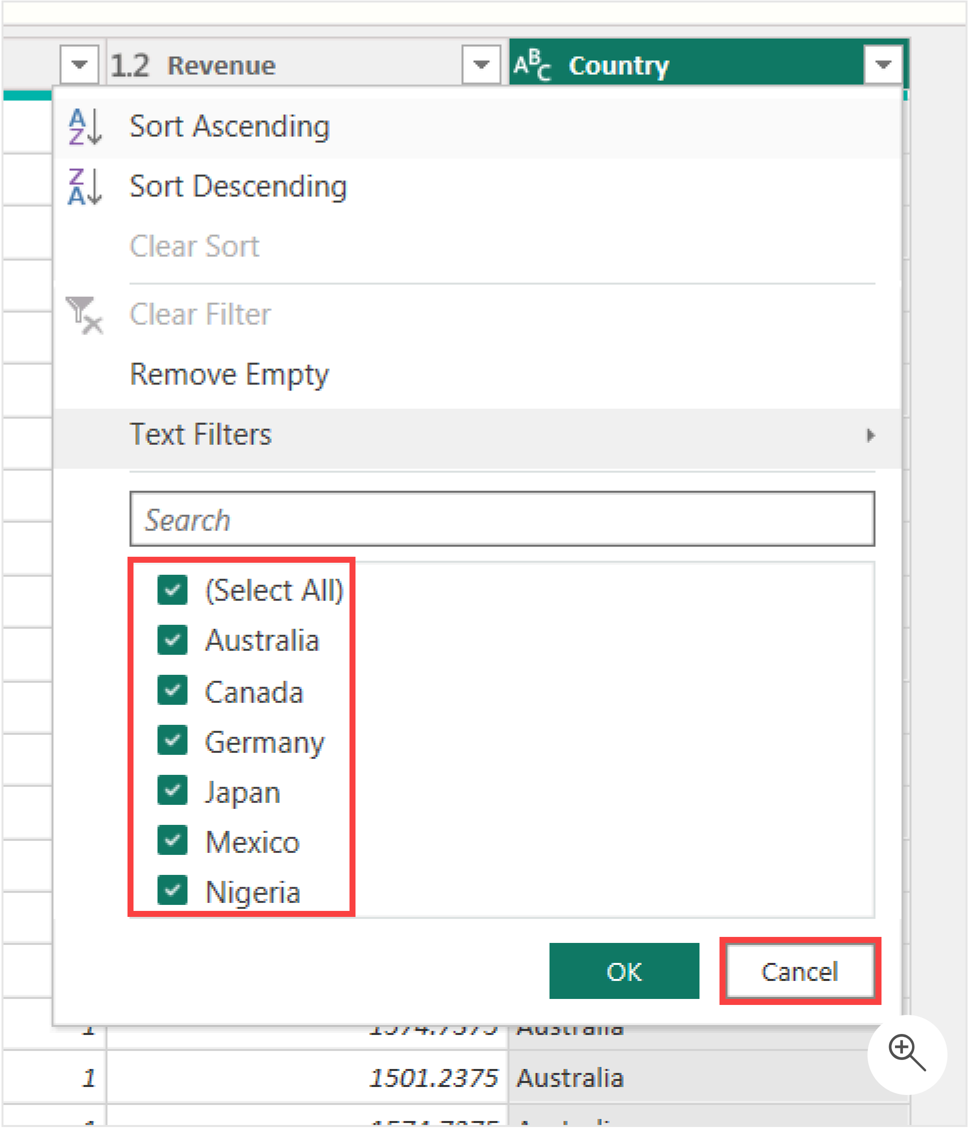
	Source.Name	ProductID	Date	Zip
1	Australia.csv	550	7/1/2015	
2	Australia.csv	550	7/1/2015	
3	Australia.csv	550	8/1/2015	
4	Australia.csv	550	8/1/2015	
5	Australia.csv	550	8/1/2015	
6	Australia.csv	550	8/1/2015	
7	Australia.csv	550	8/1/2015	
8	Australia.csv	550	8/1/2015	
9	Australia.csv	550	8/1/2015	
10	Australia.csv	550	8/1/2015	
11	Australia.csv	447	6/5/2018	
12	Australia.csv	447	6/13/2018	
13	Australia.csv	447	6/16/2018	

ⓘ Note

You may find that Australia is the only country displayed. This is due to the **Power Query Editor** displaying only the first 1000 rows of any data source. To validate you have the data from all country files you can optionally select the drop-down menu next to the **Country** column, then select **Load more**.



You will now see that **Australia**, **Canada**, **Germany**, **Japan**, **Mexico**, and **Nigeria** are all selected.



If you did this optional step, select **Cancel**.

Now that you loaded all the necessary data for the upcoming report, you're ready to start preparing the data. In the next unit, we'll be exploring methods to transform and clean our data using Power BI Desktop.

Next unit: Exercise - Perform common data cleaning practices

Continue ›

✓ 100 XP

Exercise - Perform common data cleaning practices

25 minutes

Data preparation

In this section, we explore methods to transform data. Transforming the data by renaming tables, updating data types, and appending tables together ensures that the data is ready to be used for reporting. In some instances, this means cleaning the data up so that similar sets of data can be combined. In other instances, groups of data are renamed so that end users more easily recognize them and report writing is simplified.

Section 1: Rename tables

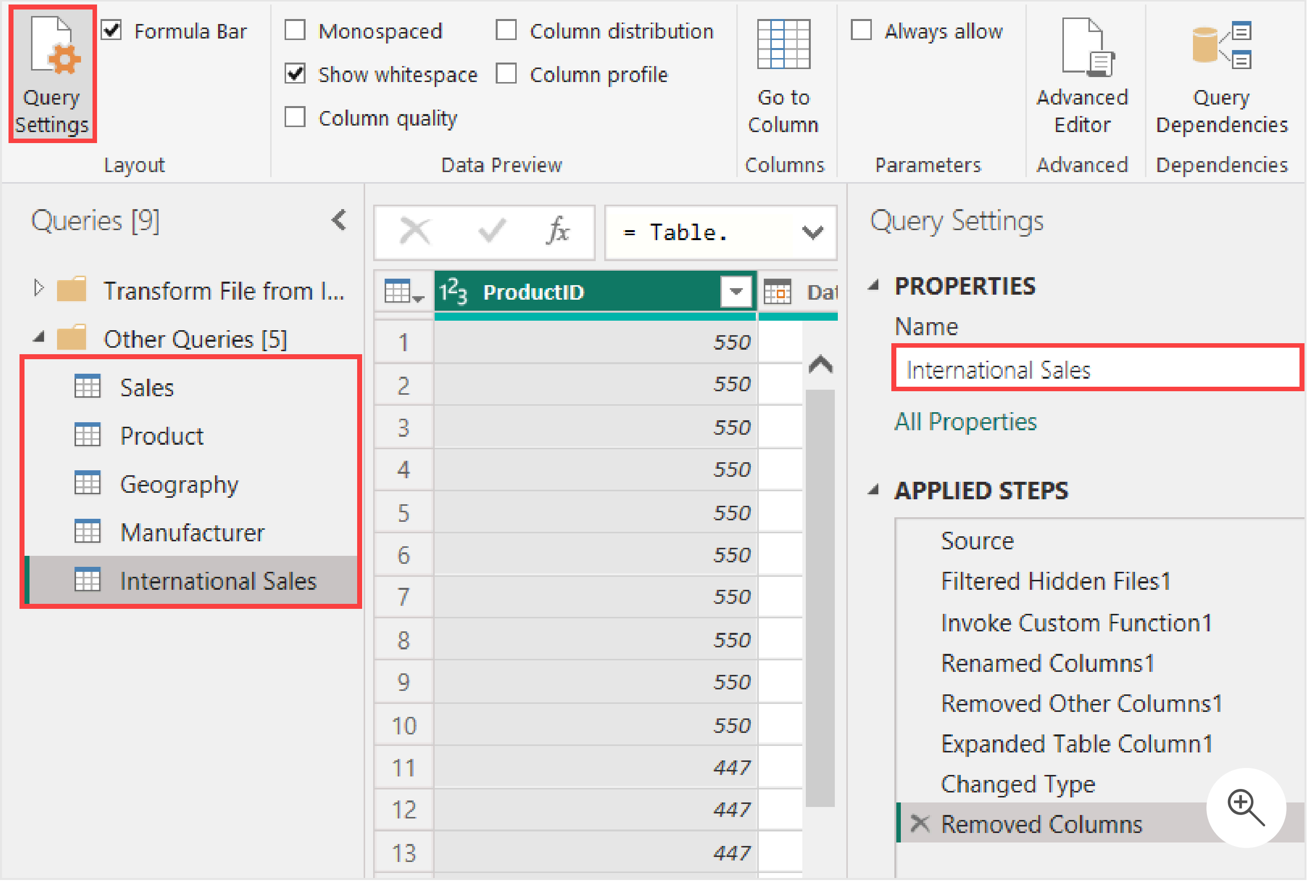
1. In the **Queries** pane, minimize the folder called **Transform Files from InternationalSales**.
2. Next, **rename** the queries listed in the **Queries** pane. Using the text field in the **Properties** section of the **Query Settings** pane, use the new names listed here to change the name of each of the queries listed. After entering the new name in the text field, hit **Enter** on your keyboard to save the new name of the query.

Expand table

Initial Name	Final Name
Sales	Sales
geo	Geography
manufacturer	Manufacturer

Initial Name	Final Name
Product_Table	Product
InternationalSales	International Sales

The Query Editor window should appear as shown here:



Name

International Sales

Source
Filtered Hidden Files1
Invoke Custom Function1
Renamed Columns1
Removed Other Columns1
Expanded Table Column1
Changed Type
Removed Columns

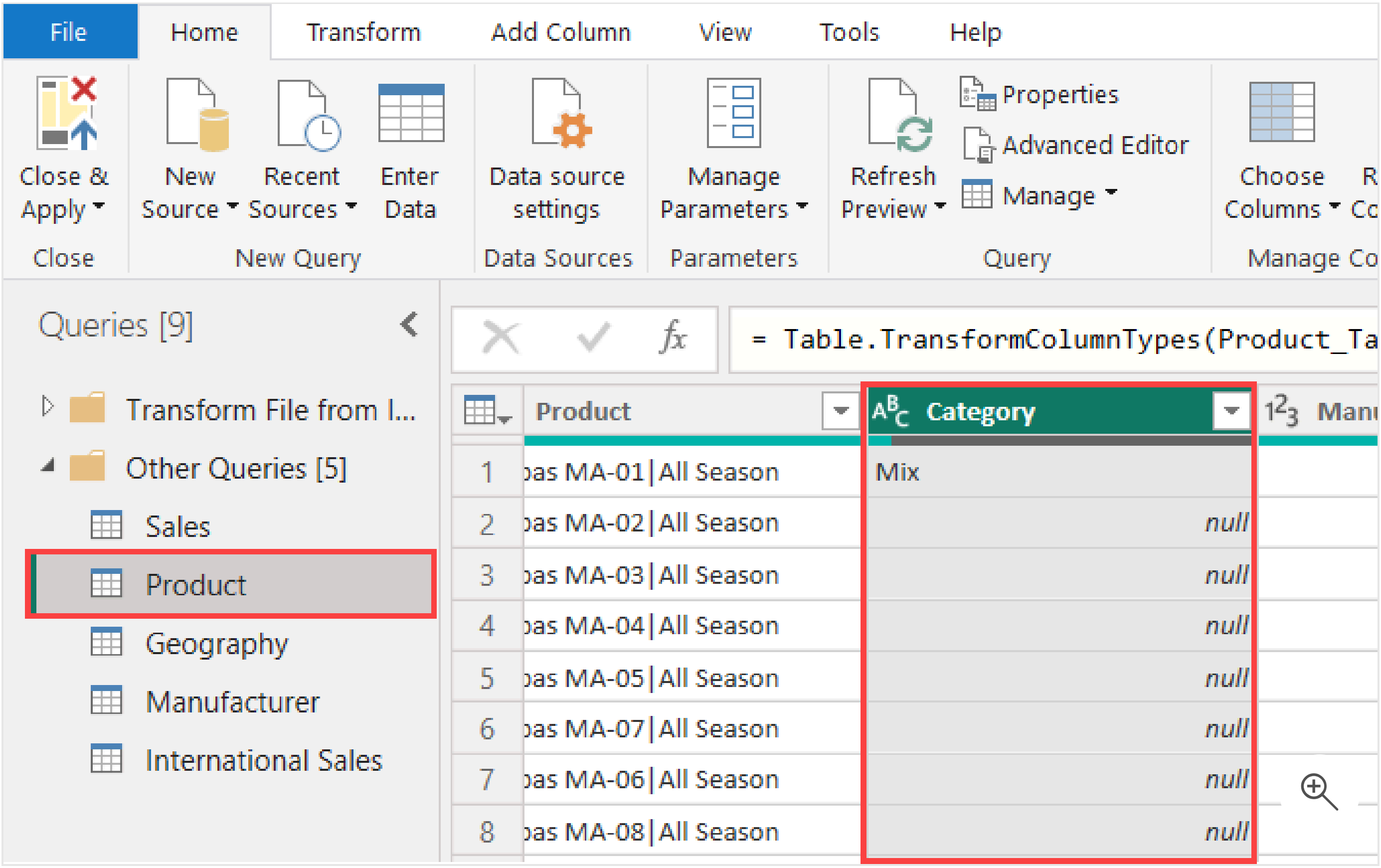
	ProductID	Dat
1	550	
2	550	
3	550	
4	550	
5	550	
6	550	
7	550	
8	550	
9	550	
10	550	
11	447	
12	447	
13	447	

ⓘ Note

It is a best practice to provide descriptive query and column names. These names are used in visuals and in the Q&A section of Power BI, which is covered in a later module.

Section 2: Fill empty values

In our scenario, some of the data isn't in the right format. Power BI provides extensive transformation capabilities to clean and prepare data to meet your needs. Let's start by selecting the **Product** query from the **Queries** pane.



```
= Table.TransformColumnTypes(Product_Ta
```

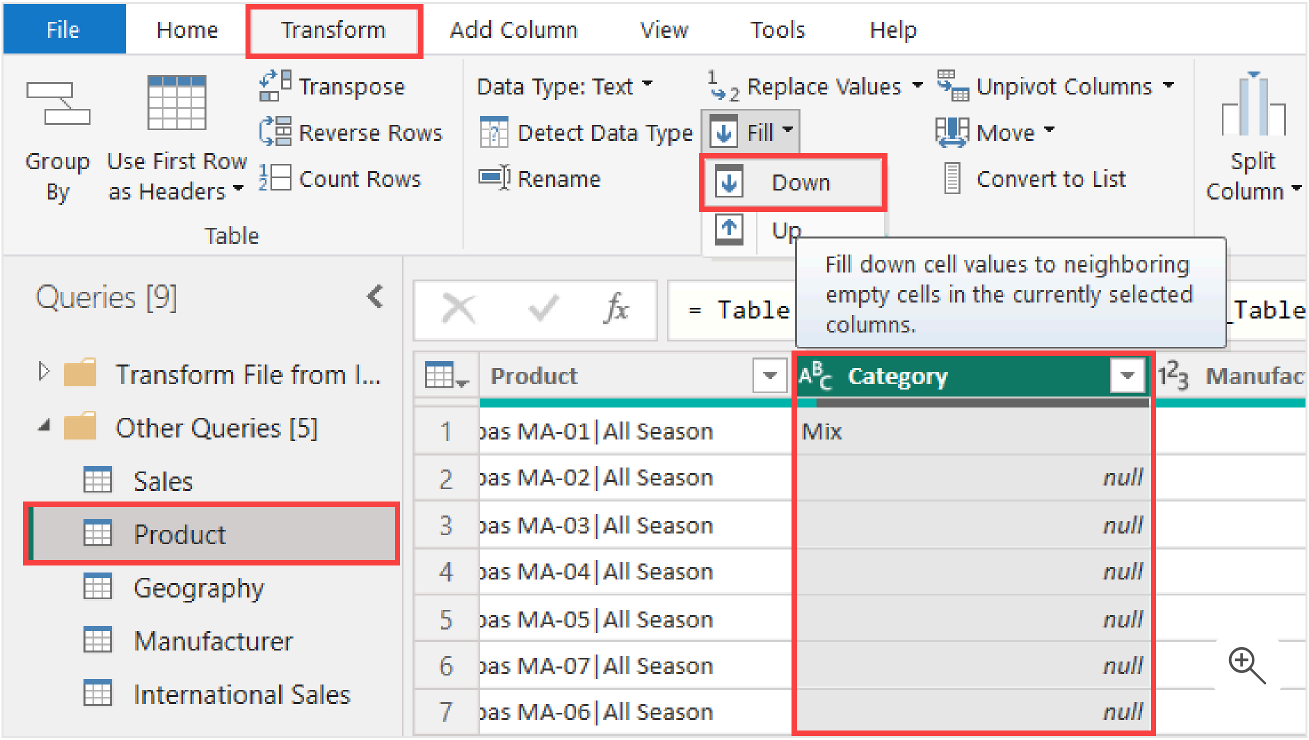
	Product	Category	Manu
1	bas MA-01|All Season	Mix	
2	bas MA-02|All Season	null	
3	bas MA-03|All Season	null	
4	bas MA-04|All Season	null	
5	bas MA-05|All Season	null	
6	bas MA-07|All Season	null	
7	bas MA-06|All Season	null	
8	bas MA-08|All Season	null	

Notice that the **Category** column has numerous **null** values. Hover over the green/gray bar (known as the quality bar) below the column header. This allows you to easily identify errors and empty values in your data previews. It looks like there are values in the Category column only when the value changes. We need to provide data in this column so there are values in each row.

1. With the **Product** query selected from the **Queries** pane, select the **Category** column.

2. From the ribbon, select the **Transform** tab.

3. Choose the **Fill** drop-down, then select the **Down** option.



Fill down cell values to neighboring empty cells in the currently selected columns.

	Product	Category	Manufac
1	ɔas MA-01 | All Season	Mix	
2	ɔas MA-02 | All Season	*null*	
3	ɔas MA-03 | All Season	*null*	
4	ɔas MA-04 | All Season	*null*	
5	ɔas MA-05 | All Season	*null*	
6	ɔas MA-07 | All Season	*null*	
7	ɔas MA-06 | All Season	*null*	

Notice how all the null values are filled with the appropriate **Category** values.

> ⓘ **Note**
>
> The fill down operation takes a column and traverses through the values in it to fill any null values in the next rows until it finds a new value. This process continues on a row-by-row basis until there are no more values in that column.

Section 3: Split columns

In the **Product** query, notice the **Product** column. It looks like the product name and product segment are concatenated into one field with a pipe (|) separator. Let's **split** them into **two** columns. This is useful when we build visuals so we can analyze based on both fields.

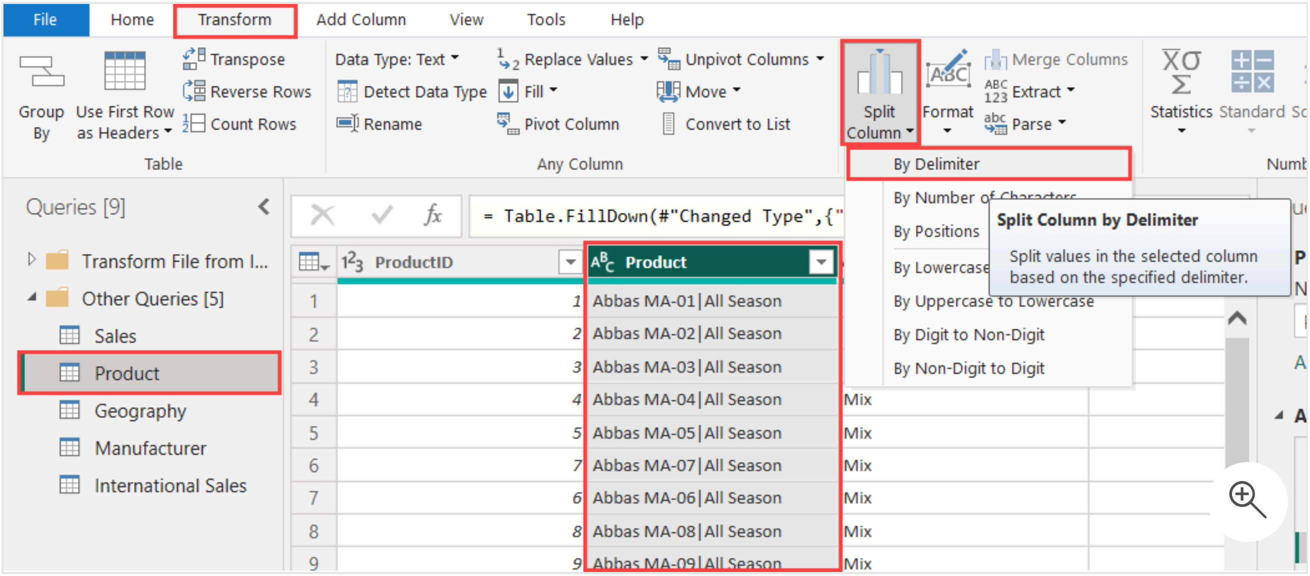
1. From the **Queries** pane to the left, make sure that the **Product** query is selected.

2. Select the **Product** column from the query table.

3. From the ribbon, select the **Transform** tab.

4. Expand the **Split Column** drop-down.

5. Then, select **By Delimiter**. The **Split Column by Delimiter** dialog box opens.



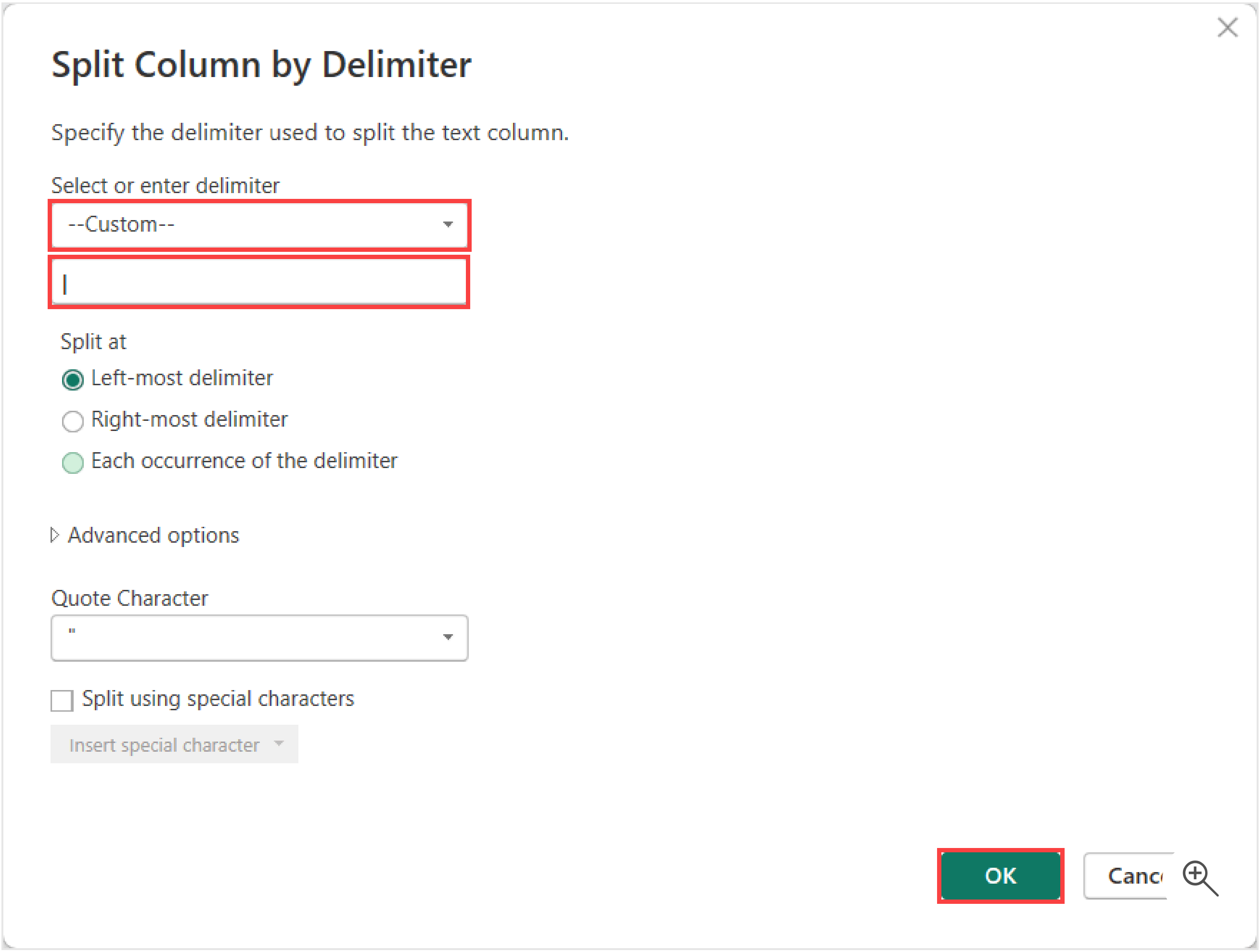
```
= Table.FillDown(#"Changed Type",{"
```

	ProductID	Product	
1	1	Abbas MA-01|All Season	
2	2	Abbas MA-02|All Season	
3	3	Abbas MA-03|All Season	
4	4	Abbas MA-04|All Season	Mix
5	5	Abbas MA-05|All Season	Mix
6	7	Abbas MA-07|All Season	Mix
7	6	Abbas MA-06|All Season	Mix
8	8	Abbas MA-08|All Season	Mix
9	9	Abbas MA-09|All Season	Mix

6. In the dialog box, ensure that **Custom** is selected in the **Select or enter delimiter** drop-down menu.

> ⓘ Note
>
> The **Select or enter delimiter** drop-down menu has some of the standard delimiters like comma, colon, and so on.

7. Notice that in the text box, there's a **hyphen** (-). Power BI assumes we want to split by hyphen. **Remove** the hyphen symbol and enter the **pipe** symbol (|). Then, choose **Left-most delimiter** under **Split at**, and select **OK**.



Split Column by Delimiter

Specify the delimiter used to split the text column.

Select or enter delimiter

--Custom--

|

Split at

● Left-most delimiter

○ Right-most delimiter

○ Each occurrence of the delimiter

Advanced options

Quote Character

"

□ Split using special characters

Insert special character

Note

If the delimiter occurs multiple times, the Split at section provides the option to split only once (either left most or right most) or the option to split the column on each occurrence of the delimiter. In this scenario, the delimiter occurs only once, therefore the Product column is split into two columns.

Section 4: Rename columns

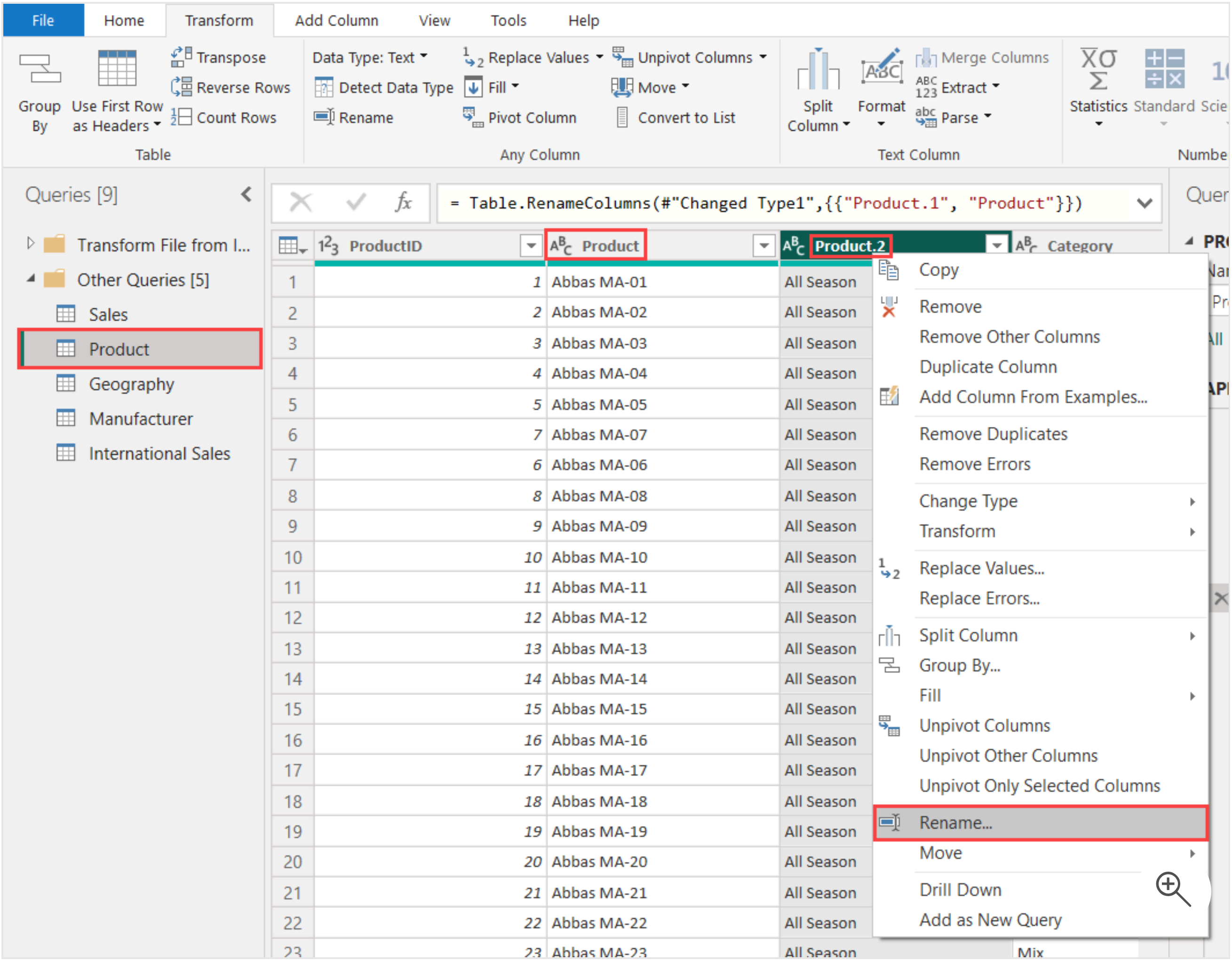
Let's rename the columns now to something more user friendly.

1. Select the **Product.1** column, and then right-click next to the column name.

2. Choose **Rename..** from the options menu.

3. **Rename** the field to **Product**.

4. Use the same steps to rename **Product.2** to **Segment**.



```
= Table.RenameColumns(#"Changed Type1",{{"Product.1", "Product"}})
```

	ProductID	Product	Product.2	Category
1	1	Abbas MA-01	All Season	
2	2	Abbas MA-02	All Season	
3	3	Abbas MA-03	All Season	
4	4	Abbas MA-04	All Season	
5	5	Abbas MA-05	All Season	
6	7	Abbas MA-07	All Season	
7	6	Abbas MA-06	All Season	
8	8	Abbas MA-08	All Season	
9	9	Abbas MA-09	All Season	
10	10	Abbas MA-10	All Season	
11	11	Abbas MA-11	All Season	
12	12	Abbas MA-12	All Season	
13	13	Abbas MA-13	All Season	
14	14	Abbas MA-14	All Season	
15	15	Abbas MA-15	All Season	
16	16	Abbas MA-16	All Season	
17	17	Abbas MA-17	All Season	
18	18	Abbas MA-18	All Season	
19	19	Abbas MA-19	All Season	
20	20	Abbas MA-20	All Season	
21	21	Abbas MA-21	All Season	
22	22	Abbas MA-22	All Season	
23	23	Abbas MA-23	All Season	Mix

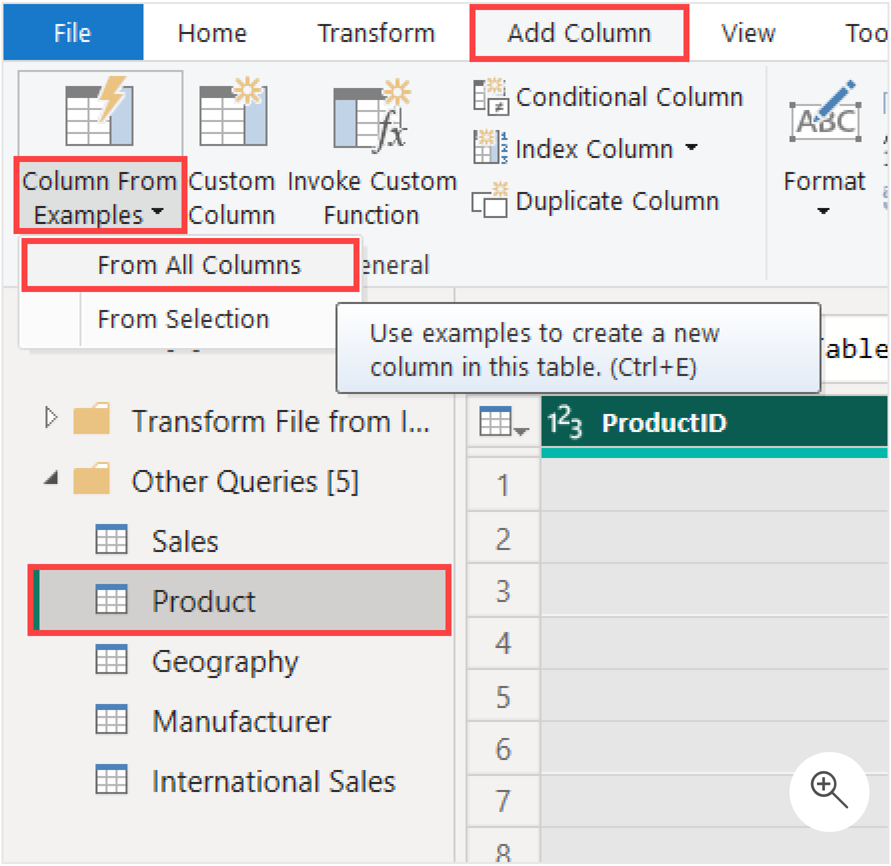
Section 5: Use Column From Examples to split columns

In the **Product** query, notice that the **Price** column has price and currency concatenated (combined) into one field. To do any calculations, we only need the numeric value. Therefore, we need to split this field into two columns. We can use the split feature like earlier or we can use **Column From Examples**. **Column From Examples** is handy in scenarios where the pattern is more complex than simply a delimiter.

1. From the **Queries** pane to the left of the screen, make sure that the **Product** query is selected.

2. From the ribbon at the top of the screen, select the **Add Column** tab.

3. Choose the **Column From Examples** drop-down, and then select **From All Columns**.



4. In the first row of the newly added **Column1**, enter the first **Price** value, **412.13**.

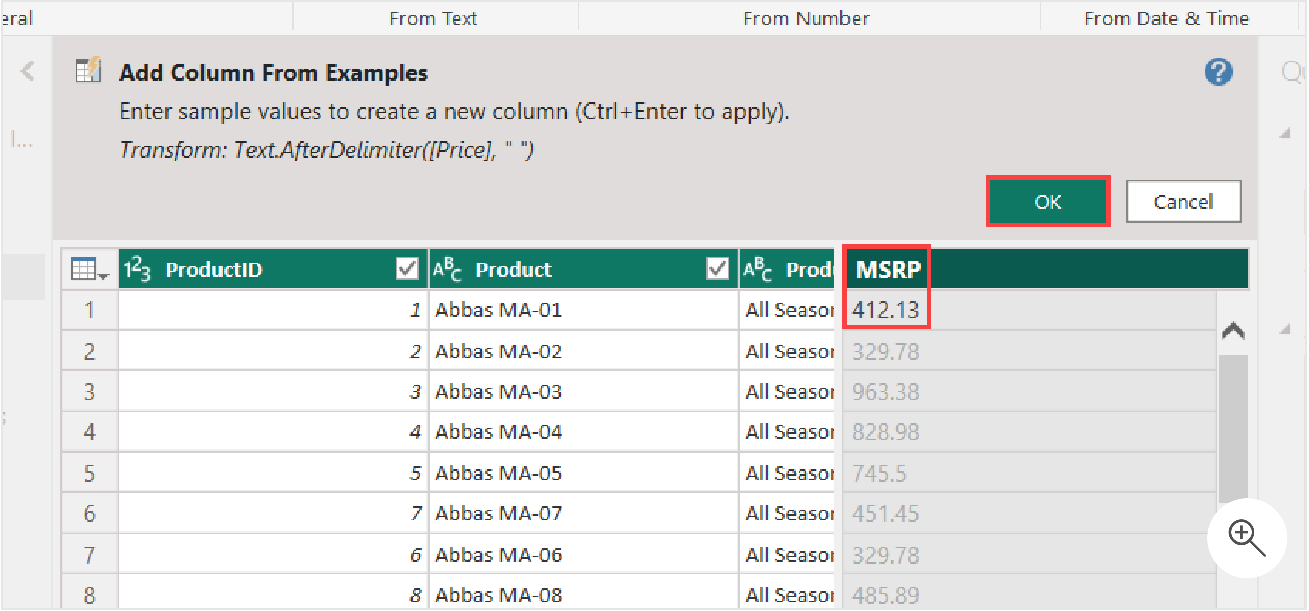
5. Hit **Enter** on your keyboard.

   Notice after you hit Enter, Power BI knows that you want to split the Price column. The formula Power BI uses is displayed as well.

   > ⓘ **Note**
   >
   > A common mistake that can occur here is the **Column From Example** feature may attempt to auto-type **USD 412.13** with the Intellisense feature. **DO NOT** accept this auto-typed value.

6. Double-click the column header of the newly added column in the query table.

7. **Rename** the column to **MSRP** and select **OK** to apply the changes.



Add Column From Examples

Enter sample values to create a new column (Ctrl+Enter to apply).

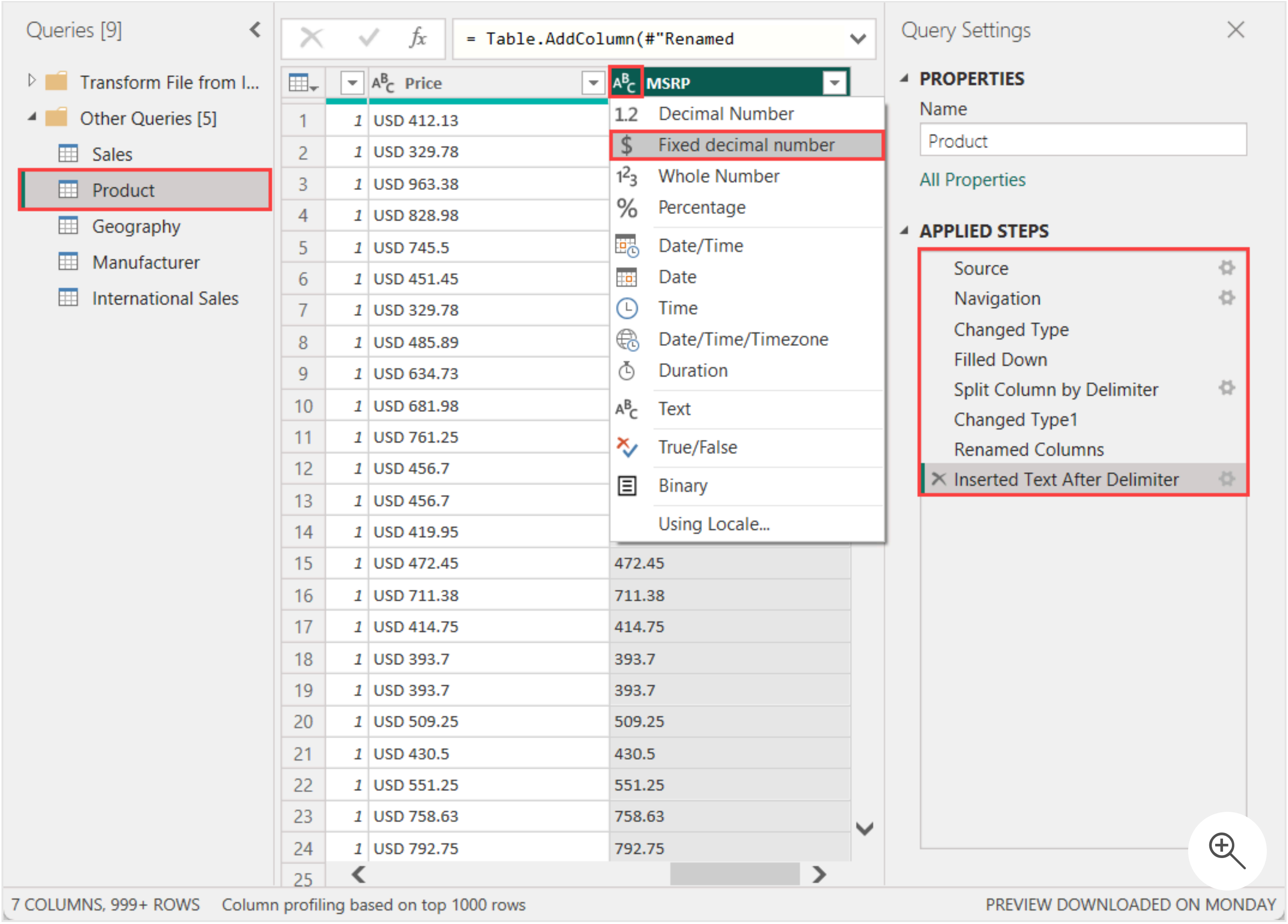
Transform: Text.AfterDelimiter([Price], " ")

	ProductID	Product	Prod	MSRP
1	1	Abbas MA-01	All Seasoı	412.13
2	2	Abbas MA-02	All Seasoı	329.78
3	3	Abbas MA-03	All Seasoı	963.38
4	4	Abbas MA-04	All Seasoı	828.98
5	5	Abbas MA-05	All Seasoı	745.5
6	7	Abbas MA-07	All Seasoı	451.45
7	6	Abbas MA-06	All Seasoı	329.78
8	8	Abbas MA-08	All Seasoı	485.89

Notice that the **MSRP** field has a Data Type of **Text**. The Data Type that it needs to be is **fixed decimal**. Let's change it.

8. Select the **ABC** icon to the left of the **MSRP** column header.

9. From the menu, select **Fixed Decimal Number**. Notice that all the steps we performed on the Product query are being recorded under **APPLIED STEPS** in the right panel.



```
= Table.AddColumn(#"Renamed
```

		Price	MSRP
1	1	USD 412.13	
2	1	USD 329.78	
3	1	USD 963.38	
4	1	USD 828.98	
5	1	USD 745.5	
6	1	USD 451.45	
7	1	USD 329.78	
8	1	USD 485.89	
9	1	USD 634.73	
10	1	USD 681.98	
11	1	USD 761.25	
12	1	USD 456.7	
13	1	USD 456.7	
14	1	USD 419.95	
15	1	USD 472.45	472.45
16	1	USD 711.38	711.38
17	1	USD 414.75	414.75
18	1	USD 393.7	393.7
19	1	USD 393.7	393.7
20	1	USD 509.25	509.25
21	1	USD 430.5	430.5
22	1	USD 551.25	551.25
23	1	USD 758.63	758.63
24	1	USD 792.75	792.75
25			

Query Settings

PROPERTIES

Name: Product

All Properties

APPLIED STEPS

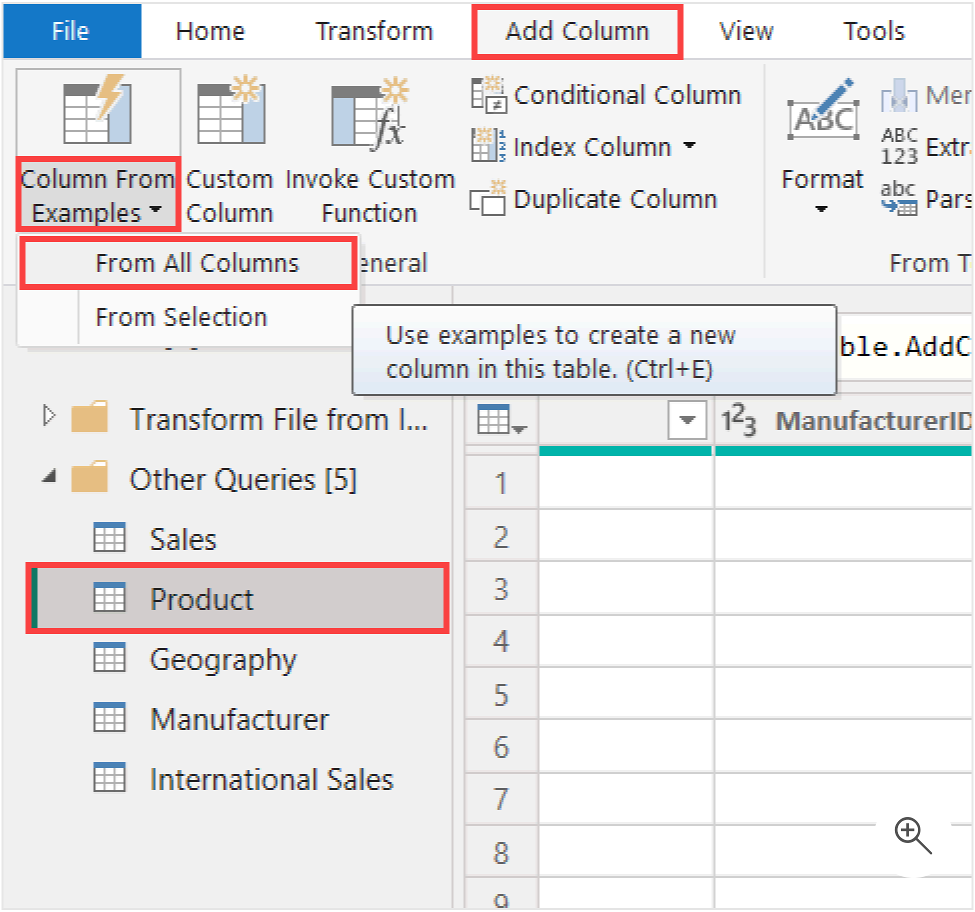
- Source
- Navigation
- Changed Type
- Filled Down
- Split Column by Delimiter
- Changed Type1
- Renamed Columns
- Inserted Text After Delimiter

7 COLUMNS, 999+ ROWS   Column profiling based on top 1000 rows   PREVIEW DOWNLOADED ON MONDAY

Now let's create a **Currency** column in the same way.

10. With the **Product** query selected, from the ribbon, select the **Add Column** tab,

11. Choose the **Column From Examples** drop-down.

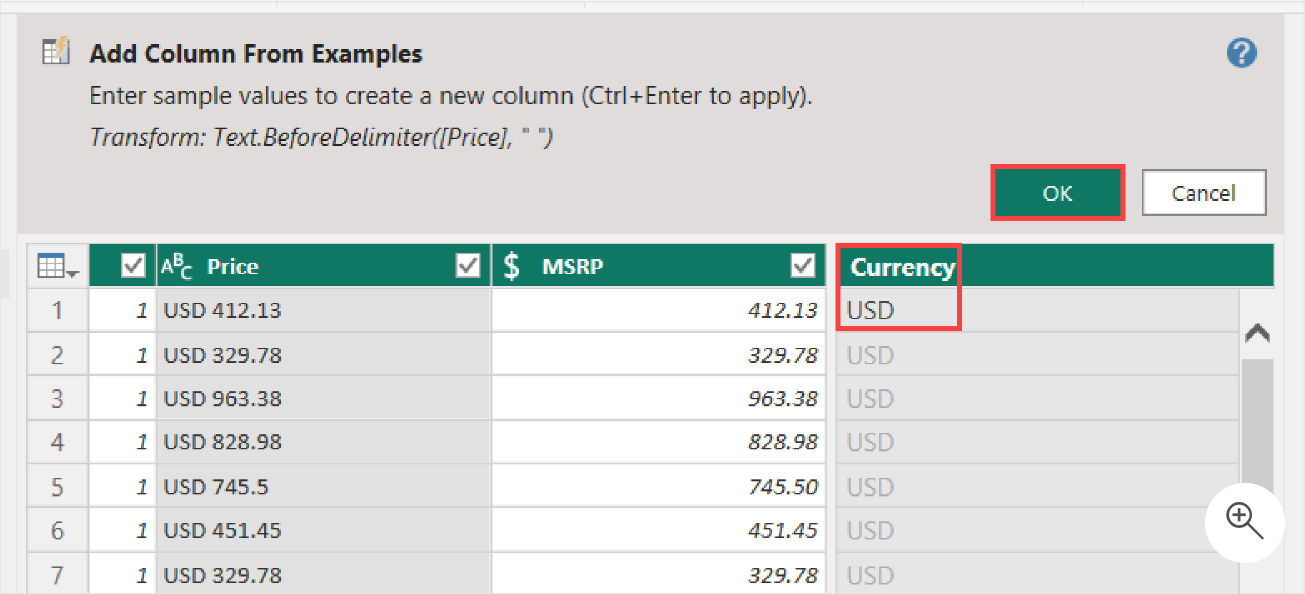


12. Then select **From All Columns**.

13. In the first row of the newly added Column1, enter the first Currency value as `USD` and then hit Enter on your keyboard.

14. **Rename** the column header from **Column1** to **Currency**.

15. Select **OK** to apply the changes.



Add Column From Examples

Enter sample values to create a new column (Ctrl+Enter to apply).

Transform: Text.BeforeDelimiter([Price], " ")

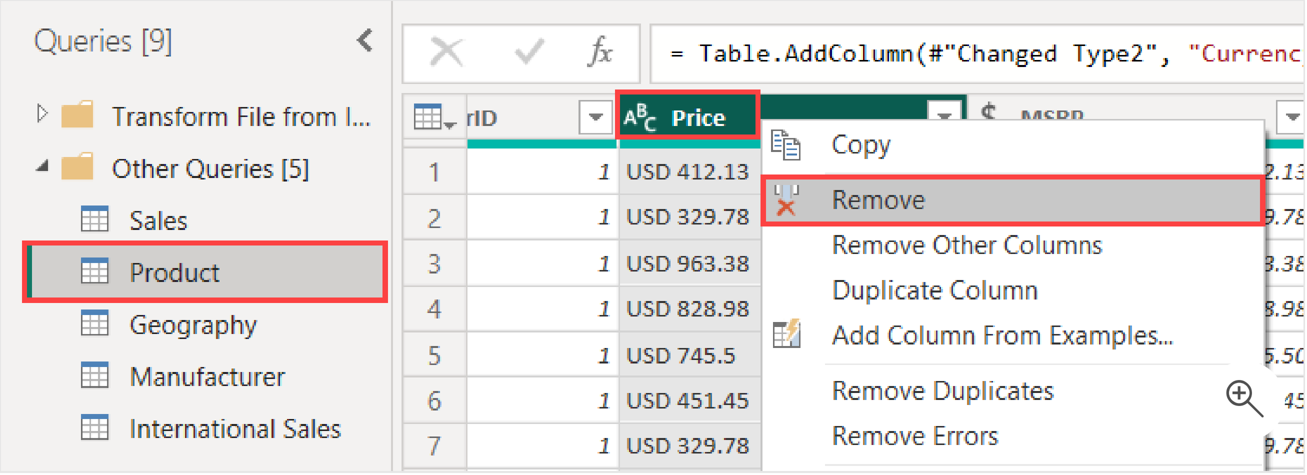
		Price	MSRP	Currency
1	1	USD 412.13	412.13	USD
2	1	USD 329.78	329.78	USD
3	1	USD 963.38	963.38	USD
4	1	USD 828.98	828.98	USD
5	1	USD 745.5	745.50	USD
6	1	USD 451.45	451.45	USD
7	1	USD 329.78	329.78	USD

Notice that after you hit **Enter**, Power BI knows you want to split the **Price** column. The formula it uses is displayed above as well.

Now that we split the **Price** column into the **MSRP** and **Currency** columns, we no longer need the original Price column. Let's remove it.

16. Make sure that you're still viewing the Product query. Right-click on the **Price** column.

17. Select **Remove** from the options menu.



```
= Table.AddColumn(#"Changed Type2", "Currenc
```

	rID	Price
1	1	USD 412.13
2	1	USD 329.78
3	1	USD 963.38
4	1	USD 828.98
5	1	USD 745.5
6	1	USD 451.45
7	1	USD 329.78

Section 6: Remove unwanted rows

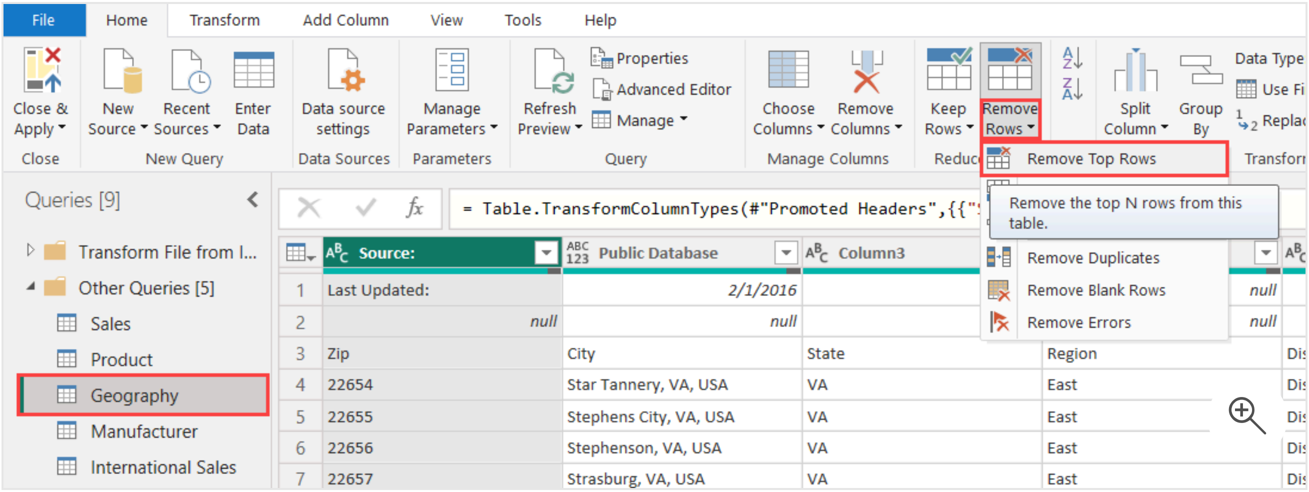
In the **Geography** query, notice that the first two rows are informational. They aren't part of the data. Similarly, in the Manufacturer query, the last couple of rows aren't part of the data. Let's remove them so we have a clean dataset to work with.

1. In the **Queries pane** to the left of the screen, select the **Geography** query.

2. From the ribbon, select the **Home** tab.

3. Choose the **Remove Rows** drop-down.

4. Then, select **Remove Top Rows**.



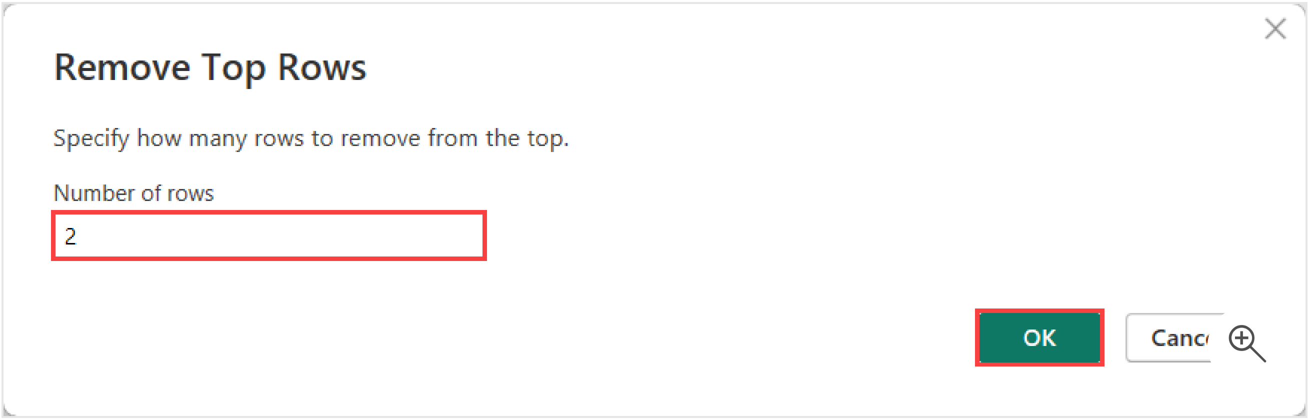
Remove the top N rows from this table.

```
= Table.TransformColumnTypes(#"Promoted Headers",{{"
```

	Source:	Public Database	Column3		
1	Last Updated:	2/1/2016		null	
2	null	null		null	
3	Zip	City	State	Region	Dis
4	22654	Star Tannery, VA, USA	VA	East	Dis
5	22655	Stephens City, VA, USA	VA	East	Dis
6	22656	Stephenson, VA, USA	VA	East	Dis
7	22657	Strasburg, VA, USA	VA	East	Dis

5. The **Remove Top Rows** dialog box opens. Enter **2** in the text box since we want to remove 2 rows, the top informational data row and the blank second row.

6. Then, select **OK**.



Remove Top Rows

Specify how many rows to remove from the top.

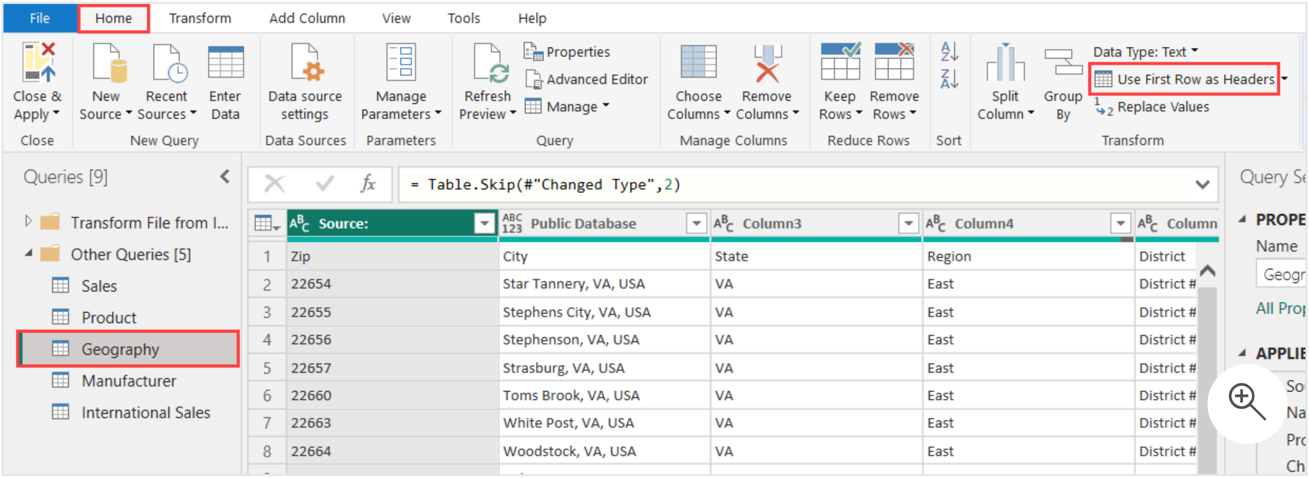
Number of rows

2

Notice the first row in the Geography query contains the column headers. Let's move them into the column header position.

7. Make sure that the **Geography** query is still selected in the Queries pane. From the ribbon at the top of the screen, select the **Home** tab.

8. Then choose **Use First Row as Headers**.



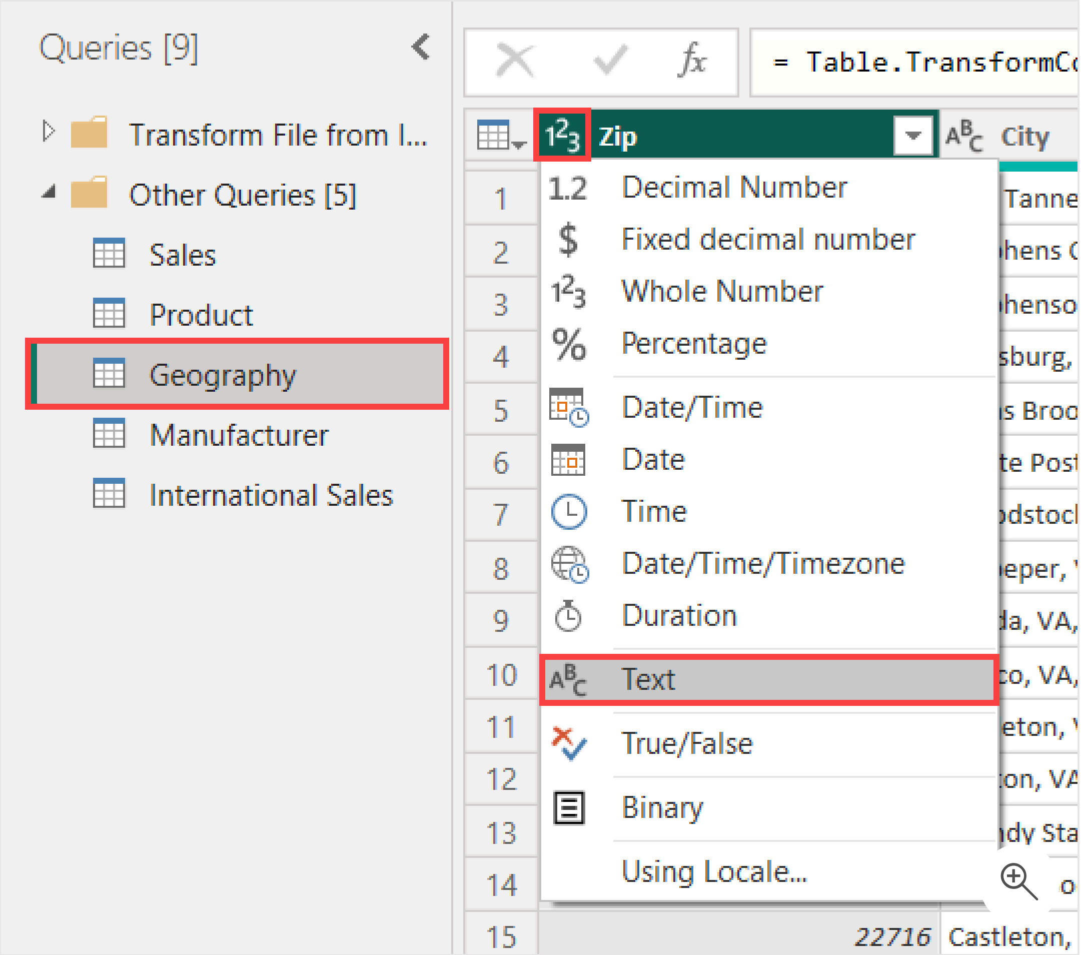
```
= Table.Skip(#"Changed Type",2)
```

	Source:	Public Database	Column3	Column4	Column
1	Zip	City	State	Region	District
2	22654	Star Tannery, VA, USA	VA	East	District #
3	22655	Stephens City, VA, USA	VA	East	District #
4	22656	Stephenson, VA, USA	VA	East	District #
5	22657	Strasburg, VA, USA	VA	East	District #
6	22660	Toms Brook, VA, USA	VA	East	District #
7	22663	White Post, VA, USA	VA	East	District #
8	22664	Woodstock, VA, USA	VA	East	District #

Power BI then predicts the data type of each field again. Notice that the column **Zip** was changed to the **Number** Data Type. Let's change it to **Text** again as we did earlier. If we don't, we see errors when we load the data.

9. Select the **data type** icon to the left of the **Zip** column header.

10. From the options menu, select **Text**.



```
= Table.TransformC
```

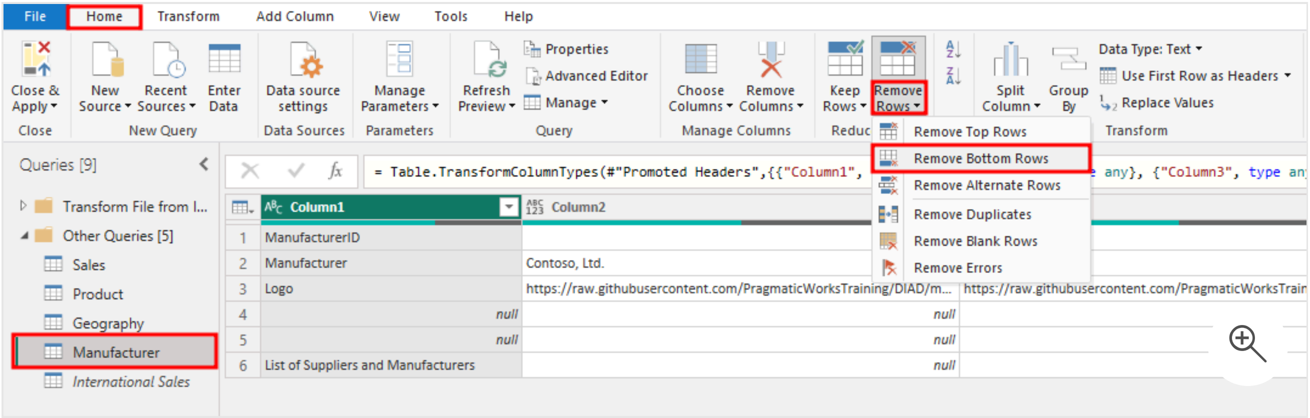
11. Select **Replace Current** in the **Change Column Type** dialog box.

12. From the **Queries** pane, select the **Manufacturer** query. Notice the bottom three rows aren't part of the data. Let's remove them.

13. From the ribbon, select the **Home** tab.

14. Choose the **Remove Rows** drop-down.

15. Then, select **Remove Bottom Rows**.

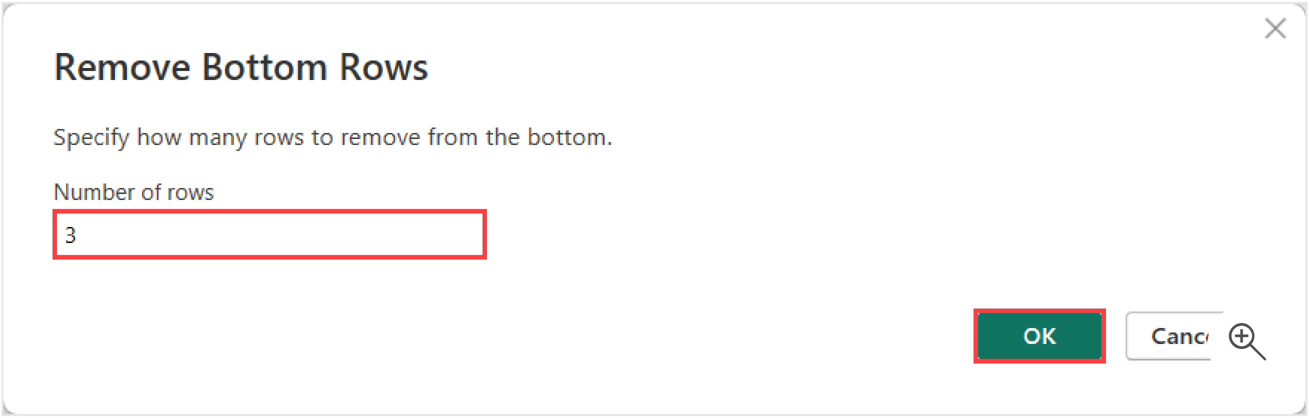


```
= Table.TransformColumnTypes(#"Promoted Headers",{{"Column1", ... any}, {"Column3", type an
```

	Column1	Column2	
1	ManufacturerID		
2	Manufacturer	Contoso, Ltd.	
3	Logo	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...	https://raw.githubusercontent.com/PragmaticWorksTrain
4	null	null	
5	null	null	
6	List of Suppliers and Manufacturers	null	

16. The **Remove Bottom Rows** dialog box opens. Enter **3** in the **Number of rows** text box.

17. Then, select **OK**.



Remove Bottom Rows

Specify how many rows to remove from the bottom.

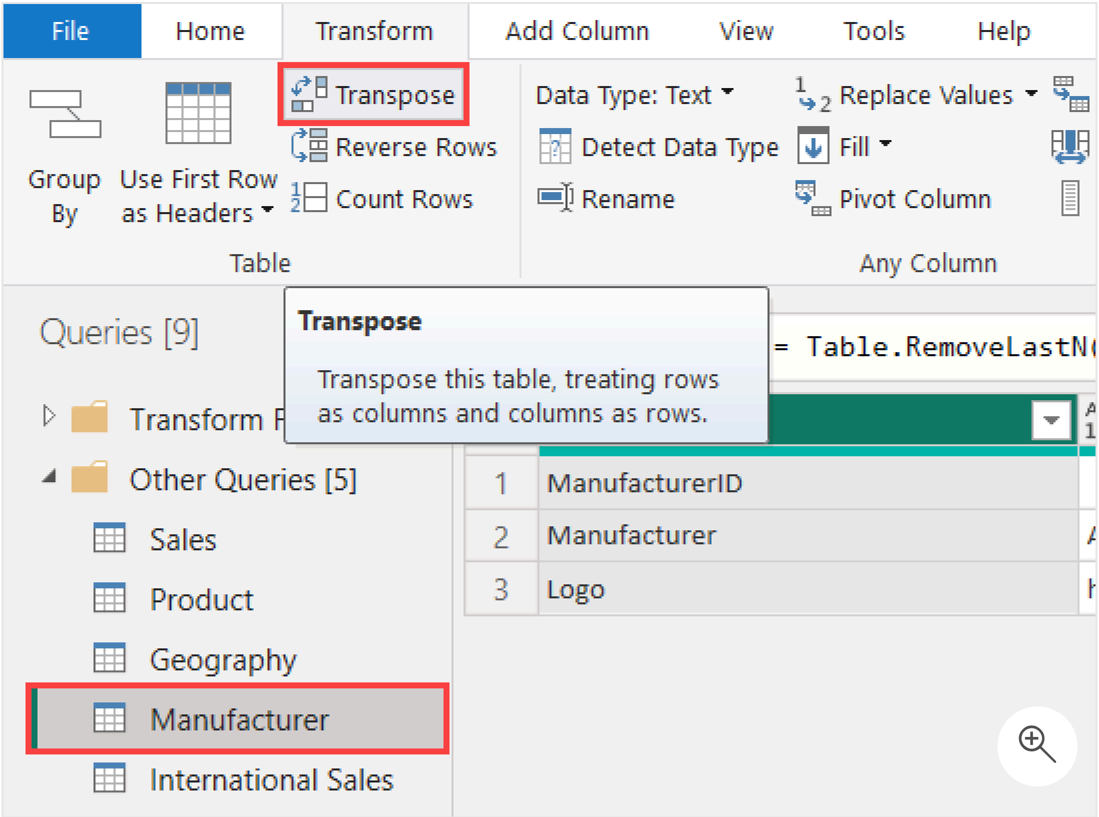
Number of rows

3

Section 7: Transpose data

1. From the **Queries pane** to the left of the screen, select the **Manufacturer** query. Notice that the **ManufacturerID**, **Manufacturer**, and **Logo** data are laid across in rows. Also notice that the header isn't useful. We need to transpose the table to meet our needs. Transposing a table treats the rows as columns and columns as rows, effectively inverting the layout of a table.

2. From the ribbon at the top of the screen, select the **Transform** tab, then choose **Transpose**.



Transpose

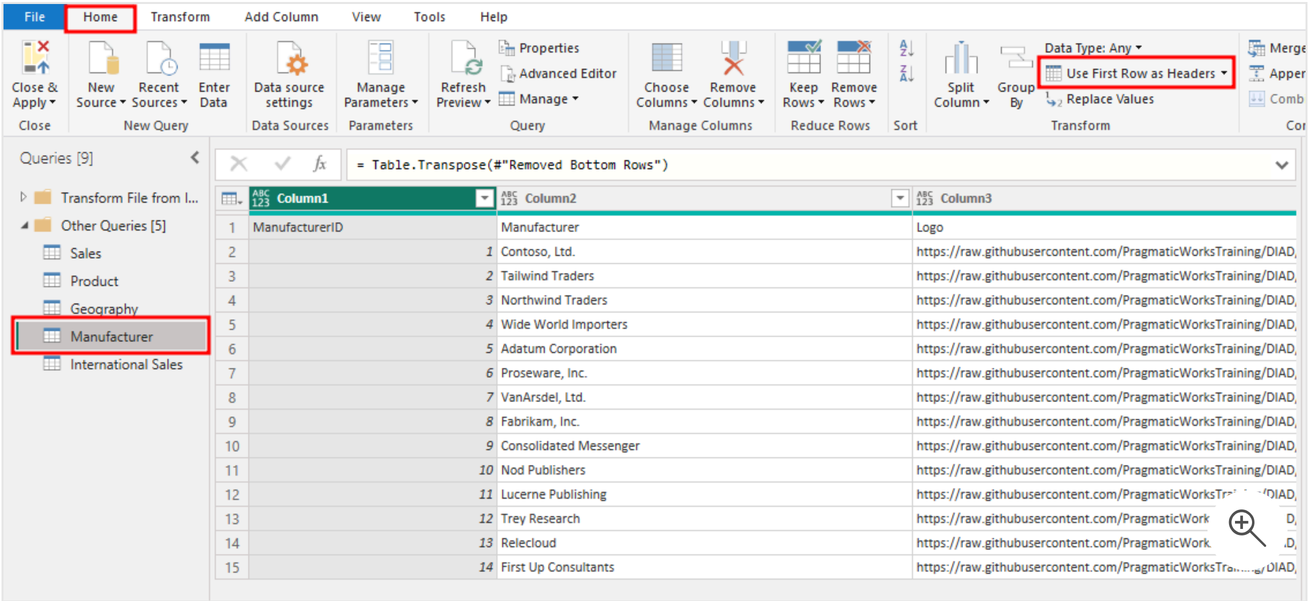
Transpose this table, treating rows as columns and columns as rows.

`= Table.RemoveLastN(`

1	ManufacturerID
2	Manufacturer
3	Logo

Notice that this transposes the data into columns. Now we need the first row to be the header.

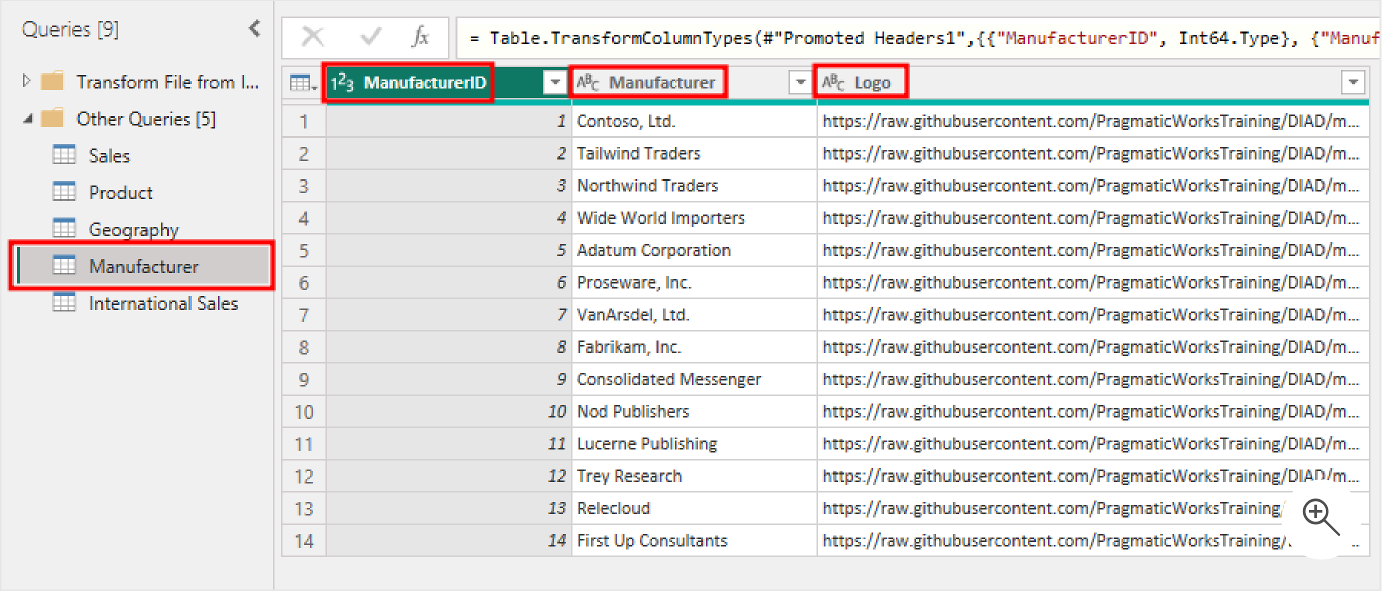
3. From the ribbon at the top of the screen, select the **Home** tab, and then choose the **Use First Row as Headers** button.



`= Table.Transpose(#"Removed Bottom Rows")`

	Column1	Column2	Column3
1	ManufacturerID	Manufacturer	Logo
2	1	Contoso, Ltd.	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD,
3	2	Tailwind Traders	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD,
4	3	Northwind Traders	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD,
5	4	Wide World Importers	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD,
6	5	Adatum Corporation	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD,
7	6	Proseware, Inc.	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD,
8	7	VanArsdel, Ltd.	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD,
9	8	Fabrikam, Inc.	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD,
10	9	Consolidated Messenger	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD,
11	10	Nod Publishers	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD,
12	11	Lucerne Publishing	https://raw.githubusercontent.com/PragmaticWorksTr DIAD,
13	12	Trey Research	https://raw.githubusercontent.com/PragmaticWork D,
14	13	Relecloud	https://raw.githubusercontent.com/PragmaticWork D,
15	14	First Up Consultants	https://raw.githubusercontent.com/PragmaticWorksTra DIAD,

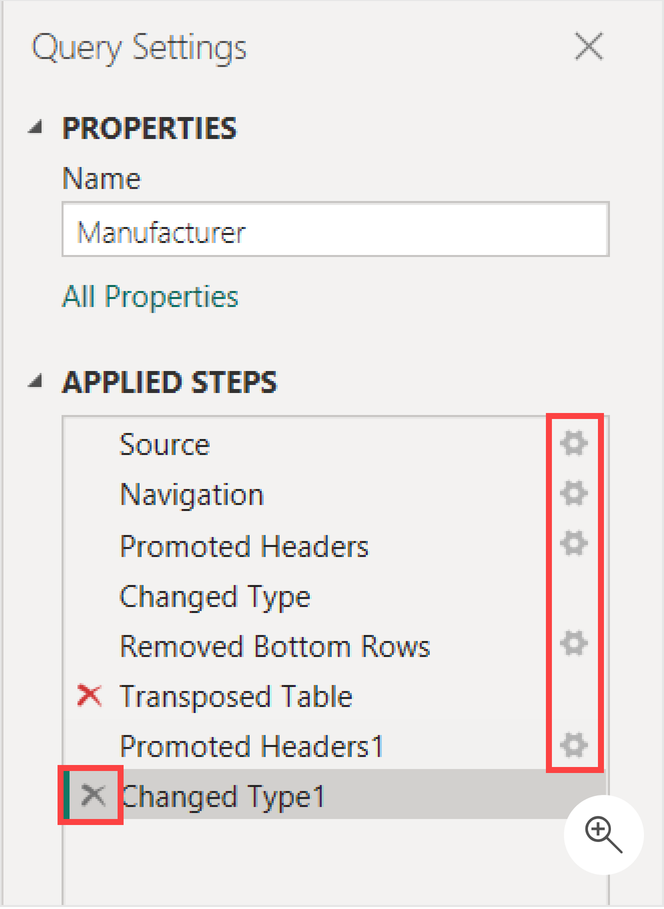
Notice that now the **Manufacturer** table is laid out the way we need it with a header and values along columns.



```
= Table.TransformColumnTypes(#"Promoted Headers1",{{"ManufacturerID", Int64.Type}, {"Manuf
```

	ManufacturerID	Manufacturer	Logo
1	1	Contoso, Ltd.	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...
2	2	Tailwind Traders	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...
3	3	Northwind Traders	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...
4	4	Wide World Importers	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...
5	5	Adatum Corporation	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...
6	6	Proseware, Inc.	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...
7	7	VanArsdel, Ltd.	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...
8	8	Fabrikam, Inc.	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...
9	9	Consolidated Messenger	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...
10	10	Nod Publishers	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...
11	11	Lucerne Publishing	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...
12	12	Trey Research	https://raw.githubusercontent.com/PragmaticWorksTraining/DIAD/m...
13	13	Relecloud	https://raw.githubusercontent.com/PragmaticWorksTraining/
14	14	First Up Consultants	https://raw.githubusercontent.com/PragmaticWorksTraining/.

Also, notice that with the **Query Settings** pane, under **APPLIED STEPS**, you see the list of transformations and steps that were applied. You can navigate through each change made to the data by selecting the step. Steps can also be deleted by choosing the **X** that appears to the left of the step. The properties of each step can be reviewed by selecting the **gear** to the right of the step.



Query Settings

PROPERTIES

Name

Manufacturer

All Properties

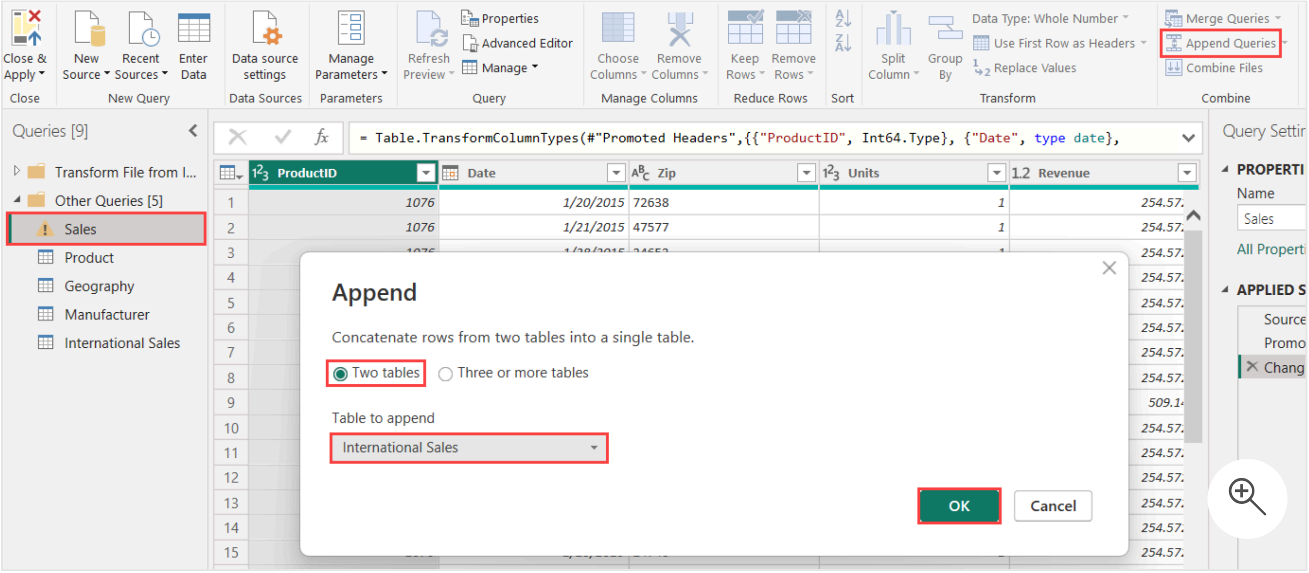
APPLIED STEPS

- Source
- Navigation
- Promoted Headers
- Changed Type
- Removed Bottom Rows
- Transposed Table
- Promoted Headers1
- Changed Type1

Section 8: Append queries

To analyze the sales of all countries, it's convenient to have a single Sales table. To do this, you need to use the **Append Queries** feature. With Append Queries, we can add all the rows from the **International Sales** query to the **Sales** query.

1. In the **Queries** pane to the left of the screen, select the **Sales** query.
2. From the ribbon at the top of the screen, select the **Home** tab, and then choose **Append Queries** button.
3. The **Append** dialog box opens. You can append Two tables or Three or more tables. Leave **Two tables** selected since we're appending just two tables.
4. From the **Table to append** drop-down, select **International Sales**.
5. Then, select **OK**.



```
= Table.TransformColumnTypes(#"Promoted Headers",{{"ProductID", Int64.Type}, {"Date", type date},
```

	ProductID	Date	Zip	Units	Revenue
1	1076	1/20/2015	72638	1	254.57
2	1076	1/21/2015	47577	1	254.57
3					254.57
4					254.57
5					254.57
6					254.57
7					254.57
8					254.57
9					509.1
10					254.57
11					254.57
12					254.57
13					254.57
14					254.57
15					254.57

Append

Concatenate rows from two tables into a single table.

● Two tables ○ Three or more tables

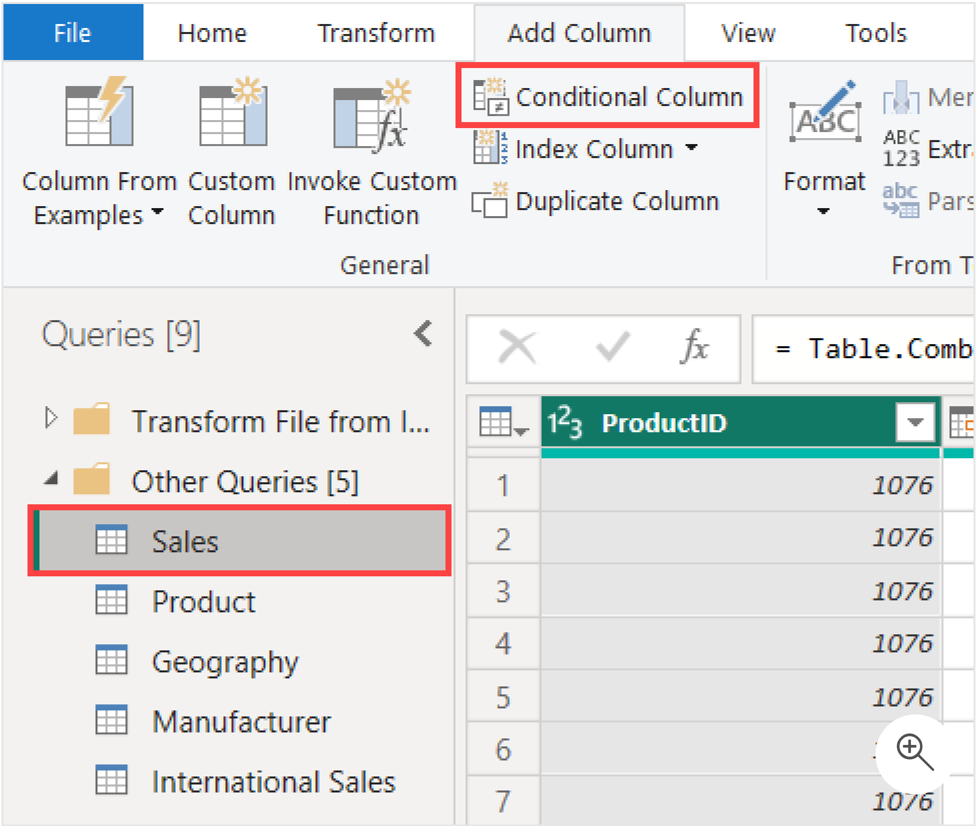
Table to append

International Sales

You now see a new column in the **Sales** table called **Country**. Since the **International Sales** query had the extra column for **Country**, the Power Query Editor added the **Country** column to the newly updated **Sales** table when it loaded the values from the **International Sales** query.

You might also notice that there are **null** values in the **Country** column by default for the **Sales** table rows. This is because that column didn't exist for the table with USA data. We now add the value **USA** as a data shaping operation.

6. From the ribbon at the top of the screen, select the **Add Column** tab, and then choose the **Conditional Column** button.



7. In the **Add Conditional Column** dialog box, enter the name of the column as **CountryName**.

8. Select **Country** from the **Column Name** drop-down menu.

9. Choose **equals** from the **Operator** drop-down menu.

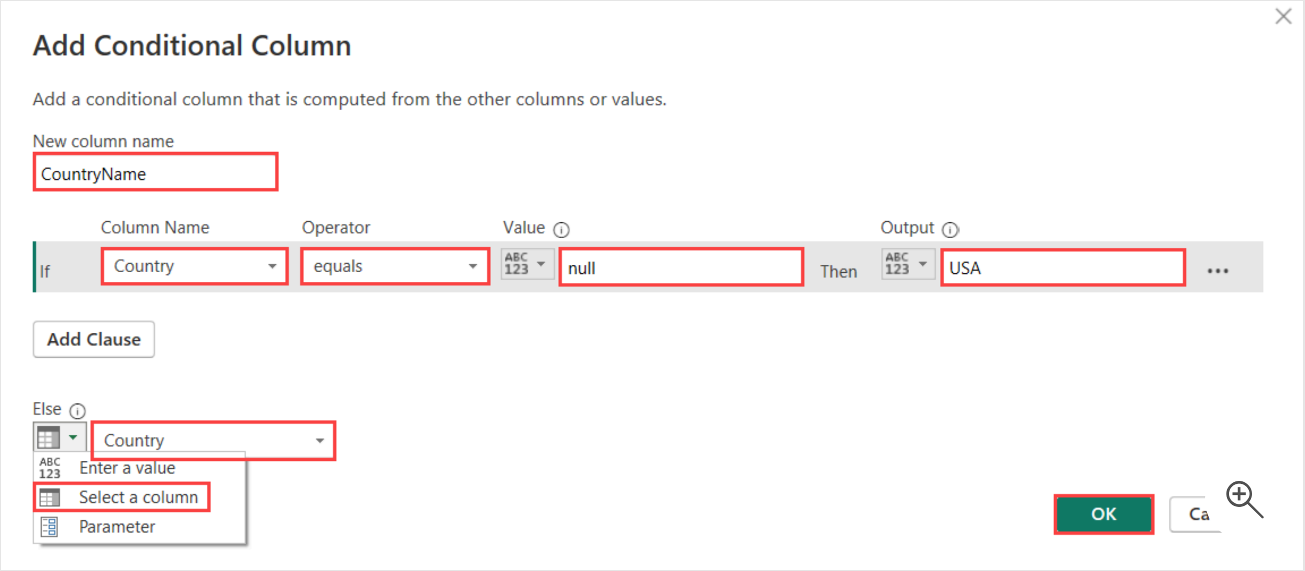
10. Enter **null** in the **Value** text box.

11. Enter **USA** in the **Output** text box.

12. Select the value drop-down menu under **Else**, and then choose the **Select a column** option.

13. Choose **Country** from the column drop-down menu.

14. Then select **OK**.



Add Conditional Column

Add a conditional column that is computed from the other columns or values.

New column name: CountryName

	Column Name	Operator	Value		Output
If	Country	equals	null	Then	USA

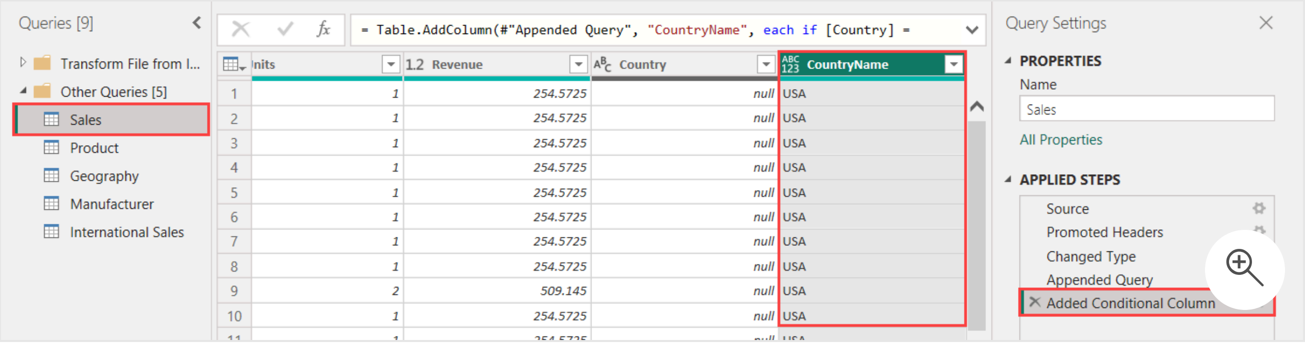
Add Clause

Else: Country

This reads: *If the current Country value is equal to null, then the value should return USA; otherwise, if the value isn't null, then use the current Country value.*

> **Note**
>
> A common mistake on the previous step is that the **Else** may not be set correctly. Please double check that your **Else** part of the conditional column matches the screenshot above.

15. You see the **CountryName** column in the Query editor window. Notice that in the **APPLIED STEPS** list, it's added to the list the action you completed.



```
= Table.AddColumn(#"Appended Query", "CountryName", each if [Country] =
```

	nits	1.2 Revenue	Country	CountryName
1	1	254.5725	null	USA
2	1	254.5725	null	USA
3	1	254.5725	null	USA
4	1	254.5725	null	USA
5	1	254.5725	null	USA
6	1	254.5725	null	USA
7	1	254.5725	null	USA
8	1	254.5725	null	USA
9	2	509.145	null	USA
10	1	254.5725	null	USA

The original **Country** column containing the null values is no longer needed and can be removed from the final table for analysis.

16. In the **Sales** query, right-click on the **Country** column.

17. Select **Remove** from the options menu.

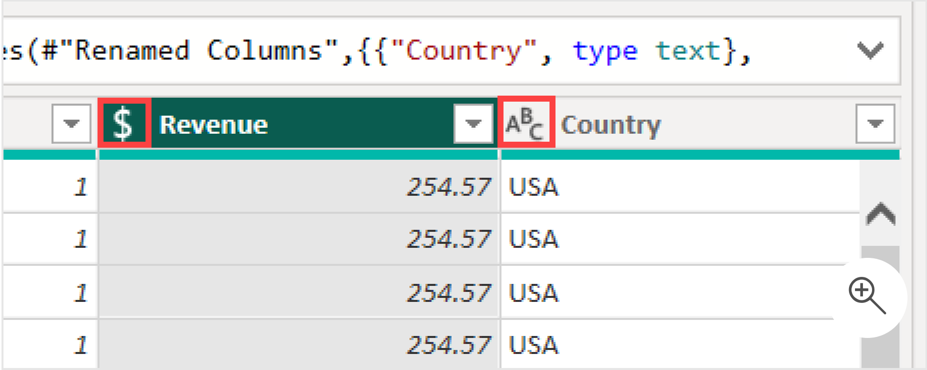
With this column now removed, we can now **rename** the **CountryName** column to **Country**.

18. Right-click on the **CountryName** column and **rename** it to **Country**.

19. Select the **Data Type** icon to the left of the **Country** column header and change the **Data Type** to **Text**.

20. Next, select the **Data Type icon** to the left of the **Revenue** column header.

21. Change the **Data Type** to **Fixed decimal number**. We do this because it's a currency field.



```
:s(#"Renamed Columns",{{"Country", type text},
```

	$ Revenue	ABC Country
1	254.57	USA
1	254.57	USA
1	254.57	USA
1	254.57	USA

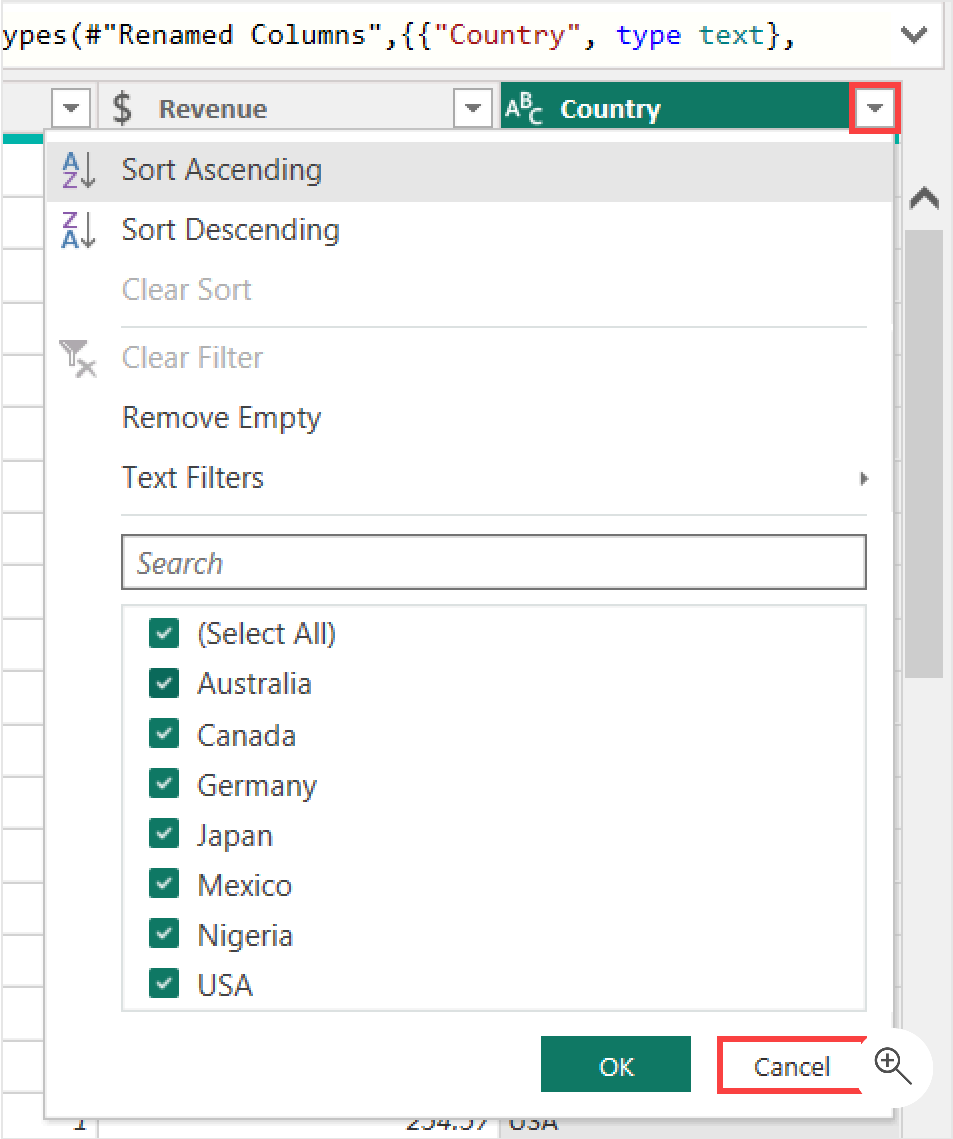
> ⓘ Note
>
> The difference between a Fixed decimal number and a Decimal number is related to the length and precision of the decimal places. For more information, see Number types.

When the data is refreshed, it processes through all the **APPLIED STEPS** that you created.

The newly named **Country** column has names for **all countries**, including the USA. You can validate this by selecting the drop-down menu next to the **Country** column to see the unique values.

22. At first, you only see USA data. Select the **drop-down arrow** to the right of the **Country** column header. Select **Load more** to validate your data from **all seven** countries.

23. Select **Cancel** to close this filter. You *don't* need to apply this filter to the data.



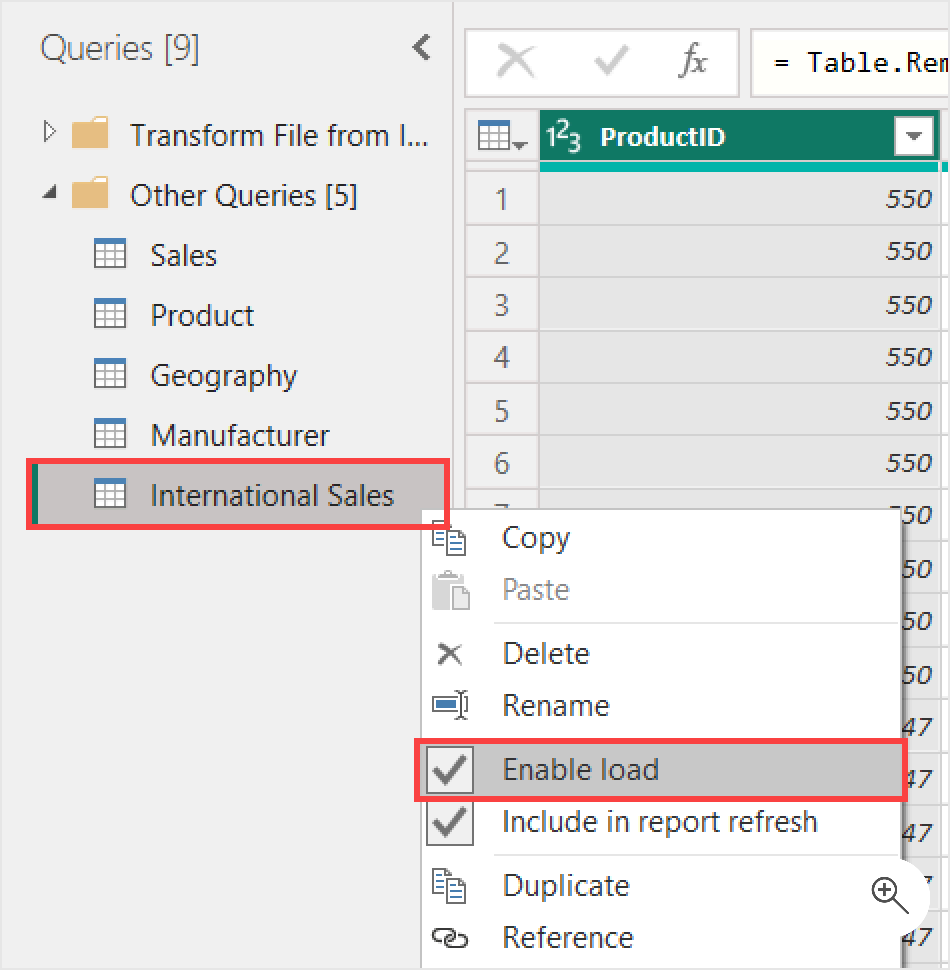
```
ypes(#"Renamed Columns",{{"Country", type text},
```

Now that the **International Sales** data is appended to the **Sales** query, in order to avoid duplicating data we should suppress the **International Sales** table from loading into the data model.

24. From the **Queries** pane to the left of the screen, select the **International Sales** query.

25. Right-click on the **International Sales** query, and then choose **Enable Load** to **deselect** this setting. This disables loading of the International Sales query into the data model.

    You should see the name of this query become italicized in the Queries pane after deselecting the Enable load option.



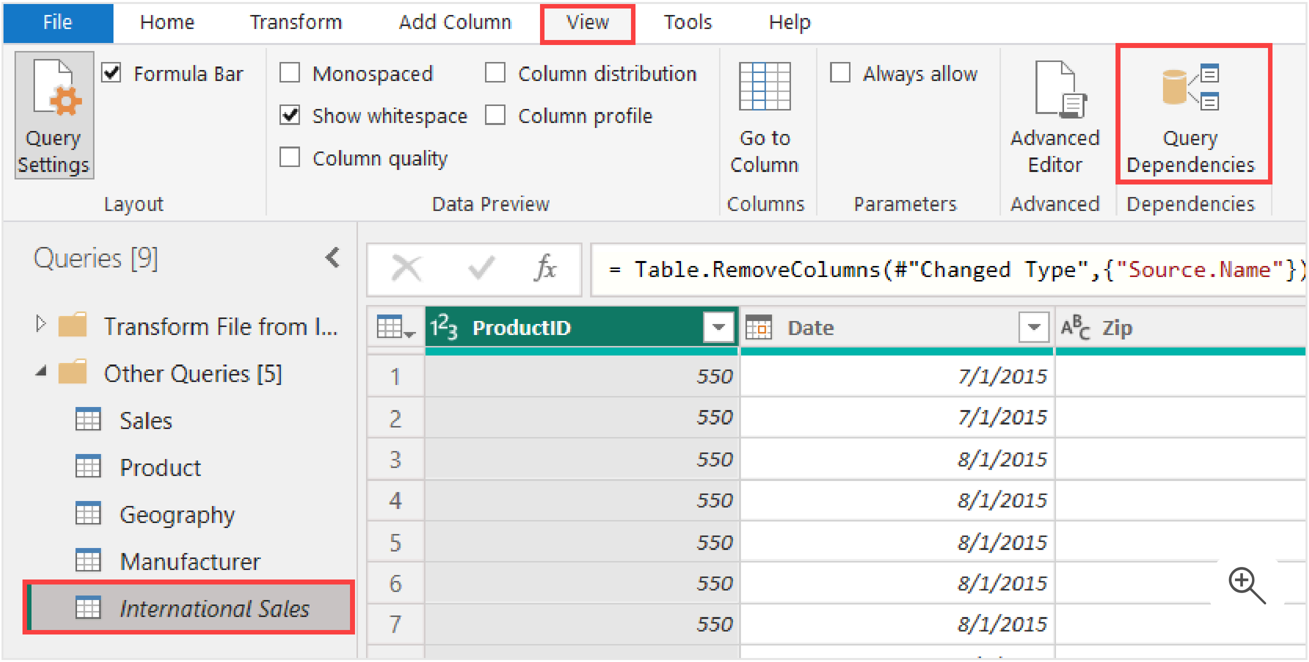
ⓘ Note

The appropriate data from the International Sales table will load onto the Sales table each time the model is refreshed. By removing the International Sales table, we are preventing duplicate data from loading into the model and increasing its file size. In some instances, storing very large amounts of data affects the data model performance.

26. You might receive a message about Possible Data Loss Warning. If so, select **Continue** when this warning appears.

27. Next, while the **International Sales** query is still selected, choose the **View** tab from the ribbon.

28. Select the **Query Dependencies** button.



☑ Formula Bar

□ Monospaced

☑ Show whitespace

□ Column quality

□ Column distribution

□ Column profile

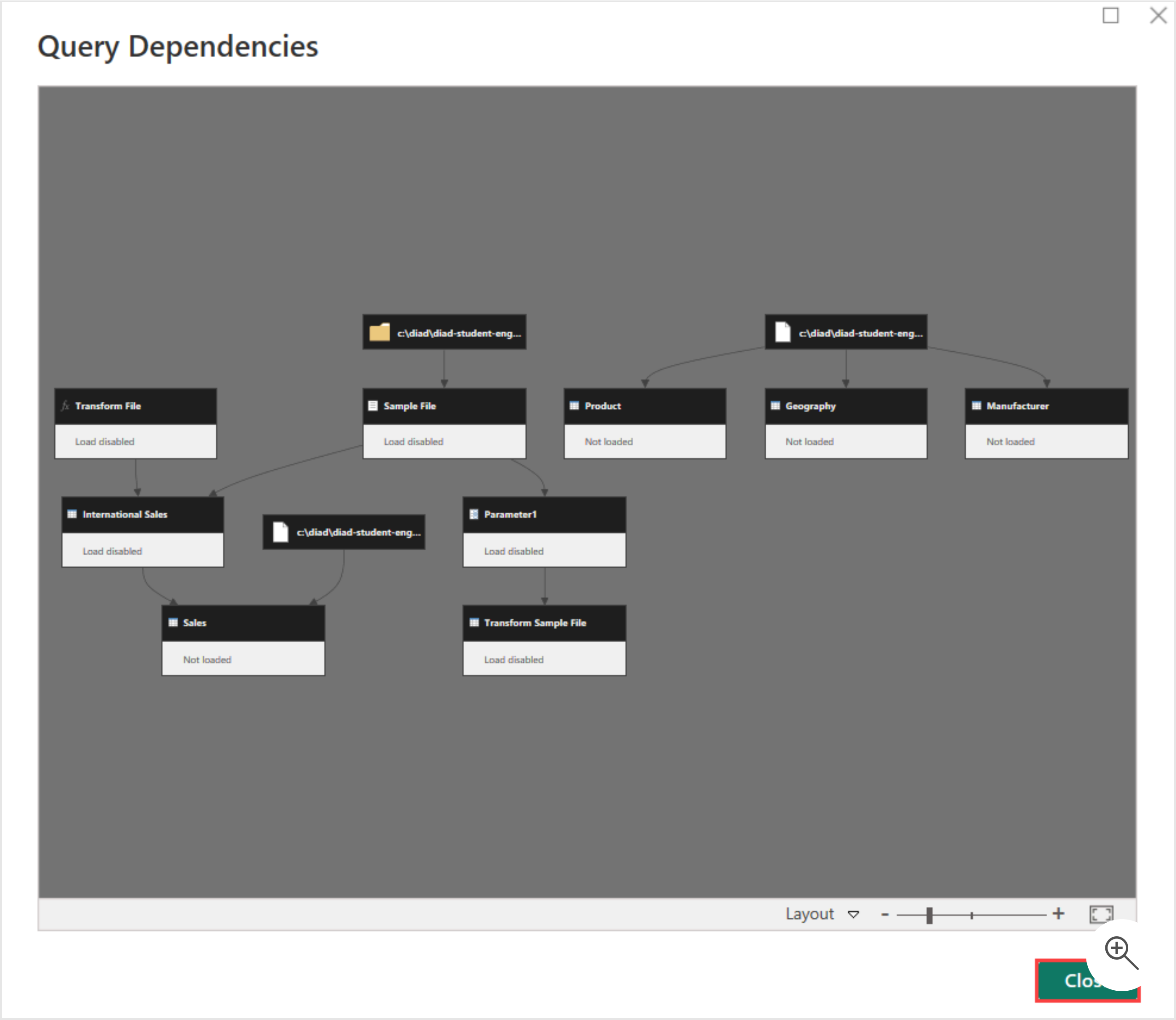
□ Always allow

```
= Table.RemoveColumns(#"Changed Type",{"Source.Name"}
```

	ProductID	Date	Zip
1	550	7/1/2015	
2	550	7/1/2015	
3	550	8/1/2015	
4	550	8/1/2015	
5	550	8/1/2015	
6	550	8/1/2015	
7	550	8/1/2015	

This opens the **Query Dependencies** dialog box. The dialog box shows the source of each query and its dependencies. For example, we see that the **Sales** query has a **CSV file source** and a dependency on the **International Sales** query. This is useful information to share knowledge with your team members.

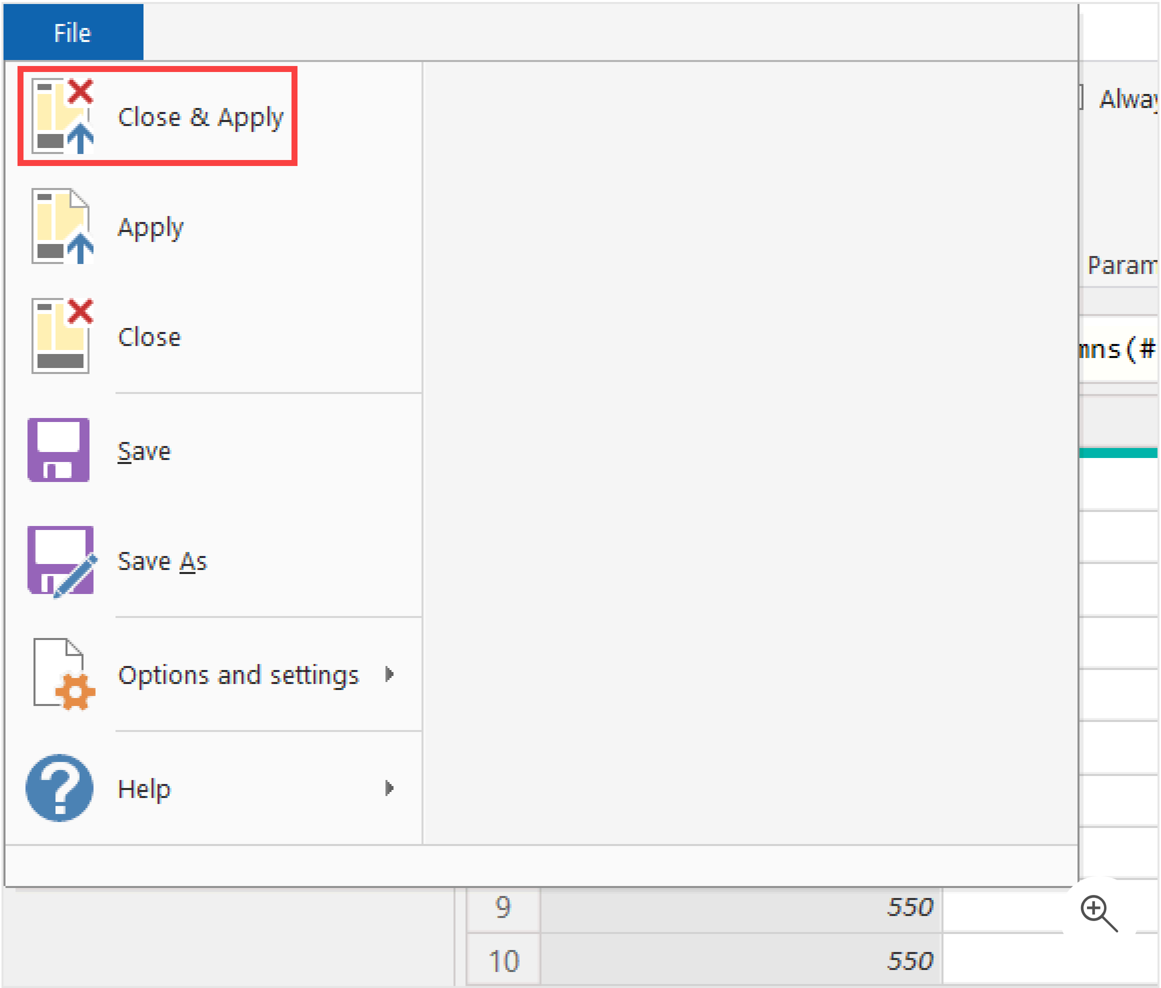
29. Select **Close** at the bottom of the dialog box.



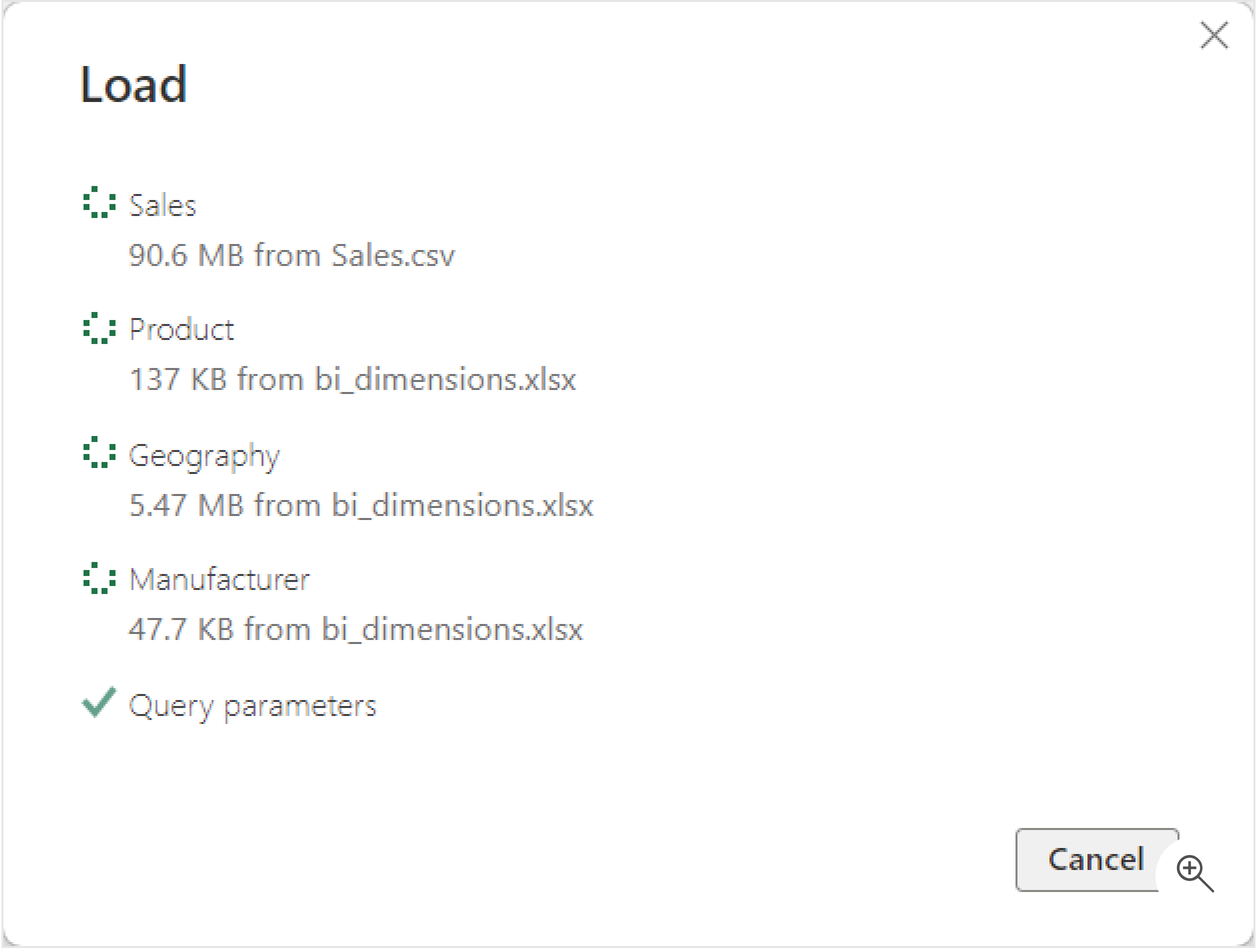
> ⓘ Note
>
> You can zoom in and out of the **Query Dependencies** view as needed.

You successfully completed import and data shaping operations and are ready to load the data into the Power BI Desktop data model to visualize the data.

30. From the ribbon at the top of the screen, select the **File** tab, then choose **Close & Apply**. This closes out the Power Query window and applies all changes.



All the data is loaded in memory in the Power BI Desktop. You see the progress dialog box with the number of rows being loaded in each table as shown in the Figure. Once the load completes, the results of this Power BI Desktop file are used in Module 3.



Load

Sales

90.6 MB from Sales.csv

Product

137 KB from bi_dimensions.xlsx

Geography

5.47 MB from bi_dimensions.xlsx

Manufacturer

47.7 KB from bi_dimensions.xlsx

✓ Query parameters

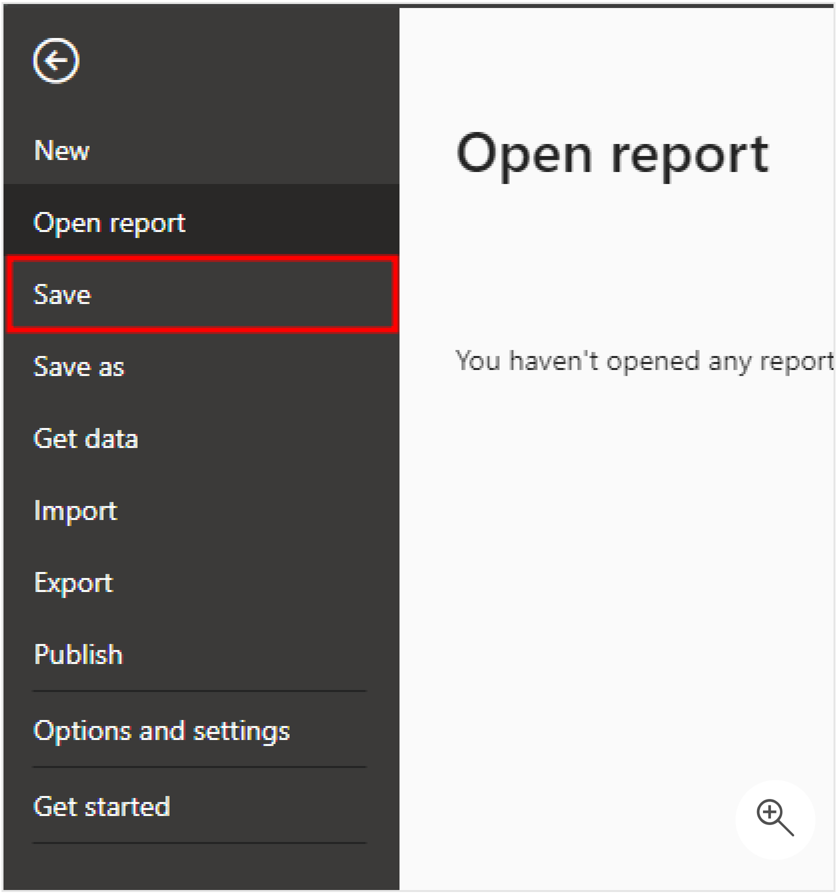
ⓘ **Note**

It may take several minutes to load all the tables.

31. Once the data finishes loading, select the **File** tab from the ribbon at the top of the screen.

32. Then, from the options menu to the left, select **Save** to save the file.

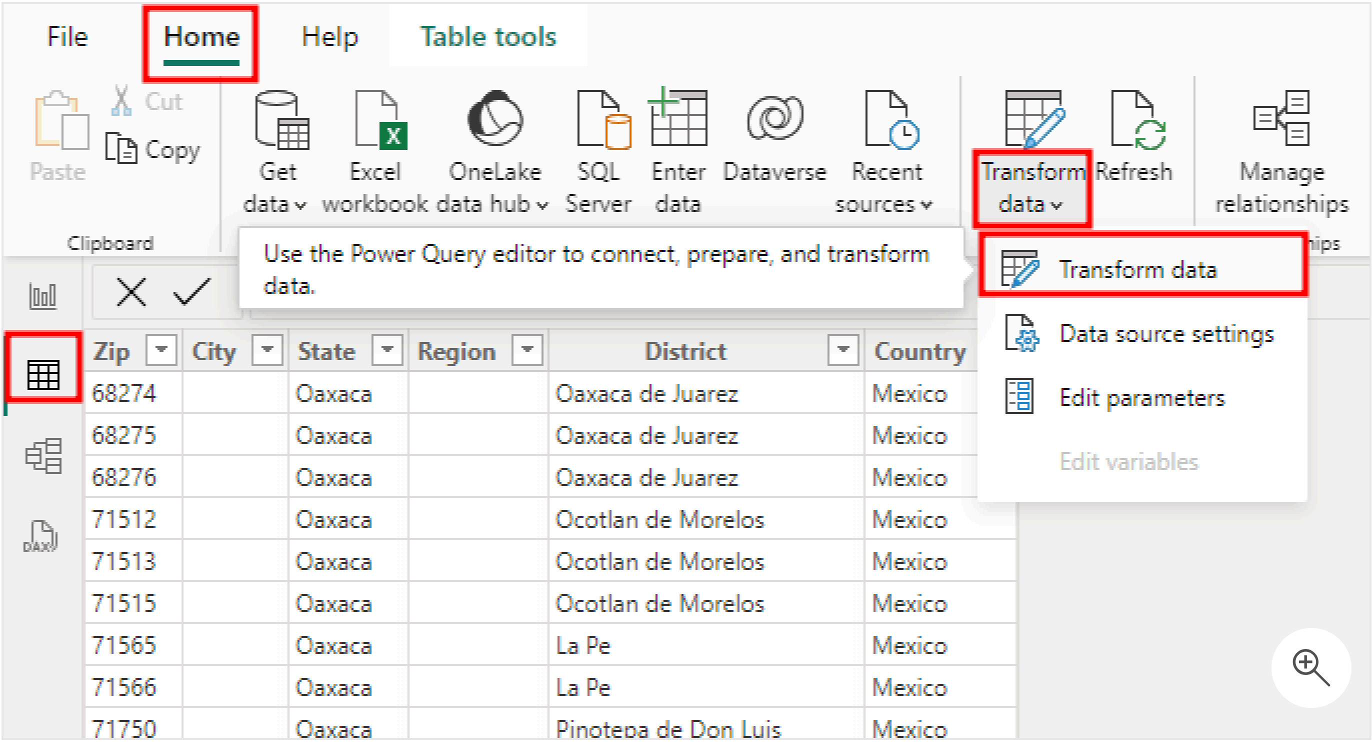
33. Name the file **MyFirstPowerBIModel**. Save the file in the **DIAD Reports (DIADReports)** folder.



Open report

You haven't opened any report

In the navigation pane to the left of the screen, select the Data icon to view the data that was loaded. If you need to return to the Power Query editor again, navigate to **Home** -> **Transform data** -> **Transform data**



Use the Power Query editor to connect, prepare, and transform data.

Zip	City	State	Region	District	Country
68274		Oaxaca		Oaxaca de Juarez	Mexico
68275		Oaxaca		Oaxaca de Juarez	Mexico
68276		Oaxaca		Oaxaca de Juarez	Mexico
71512		Oaxaca		Ocotlan de Morelos	Mexico
71513		Oaxaca		Ocotlan de Morelos	Mexico
71515		Oaxaca		Ocotlan de Morelos	Mexico
71565		Oaxaca		La Pe	Mexico
71566		Oaxaca		La Pe	Mexico
71750		Oaxaca		Pinotepa de Don Luis	Mexico

Next unit: Check your knowledge

Continue >

✓ 100 XP ▶

Summary

1 minute

In this module, you learned methods for loading data from sources like Microsoft Excel Workbooks and Comma-Separated Values (CSV) files, loading data from multiple sources at once, and using the Power Query Editor for various data cleaning practices.

Learned concepts:

- How to use the Get Data function for loading common data sources like a CSV
- The process of loading multiple tables from one Excel Workbook
- Using the connection of a folder to load multiple files and combine them into one table
- Implementing methods from the Power Query Editor to transform data
    - Renaming tables and columns
    - Filling empty values
    - Splitting columns
    - Changing column data types
    - Using Columns from Examples
    - Removing unwanted rows
    - Transposing data
    - Appending queries
    - Using conditional columns
    - Disabling the load function of a query
- Applying query editor changes and saving the file
- How to re-enter the Power Query Editor

Module incomplete:

Go back to finish ›