Power Query #3

Import, Clean, Transform, and Load Data in Power BI Desktop

Table of Contents

Power BI Desktop is a free download 2

What Is Power Query & M Code? 2

Where is Power Query in Power BI Desktop? 2

Data Source Settings in Power BI Desktop 2

Power Query Editor in Power BI Desktop 3

Apply & Close Button 3

Refreshing a Visual or Report Dashboard 4

Csv File Structure 4

Excel File Structure & Excel Tables 4

Access Database File Structure 5

Project Goal for Video 03 6

Steps in Project: 7

- 1) Download files 7
- 2) Import: Table from Csv File 7
- 3) Import: Table from Excel File 10
- 4) Import: Table from Access Database File 12
- 5) Clean: Split Column by Delimiter 14
- 6) Clean: Convert ISO Dates to Real Dates 16
- 7) Cleaning: Replace Values So Relationship in Data Model Can Function Correctly 17
- 8) Load three tables to the Data Model 18

Star Schema Data Model or Semantic Model 19

- 9) Create One-To-Many Relationship 20
- 10) Create the Net Sales DAX Formula Measure (Filter Context & Row Context) 21
- 11) Hide Fields Not Needed in the Reporting Area (No Implicit Measures) 22
- 12) Creating Useful Information; Report / Dashboard 23

SUMX Function 26

One-Step Vs Two-Step Measures 26

Keyboard shortcuts seen in video 26

Power BI Desktop is a free download

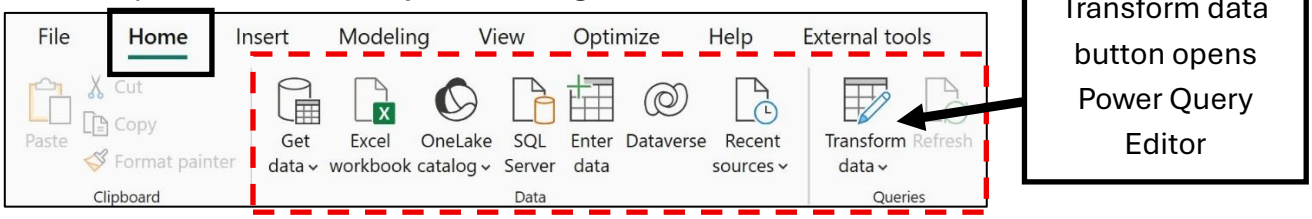
Live link to download Power BI Desktop: https://powerbi.microsoft.com/en-us/desktop/

What Is Power Query & M Code?

- Power Query is a tool in Excel (and Power BI) that allows you to connect to data wherever the data may be (online, database, text files, Excel files) and import, clean and transform the data before loading it to the worksheet, a PivotTable or the Data Model.
- What is so amazing about Power Query is that it can work with data in any location, such as: databases, tables, records, lists, worksheets, Excel Tables, text files, online locations, and many others.
- And once you gather the data into the Power Query Editor, you can use the user interface to create all the steps needed to get your data into the correct structure for the analysis that you need. And, when you use the user interface to add steps to your query, Power Query writes all the formulas and code for you in the computer language M Code.
- In Power Query, the formulas are written with M Code, a case-sensitive, function-based computer language that "mashes up" your data together into to just the structure that you want. The "mash up" term is where Microsoft got the name "M Code".
- As you become proficient with Power Query and M Code, you'll learn to create queries by combining the user interface with your own handwritten M Code—an approach you'll practice throughout this class.

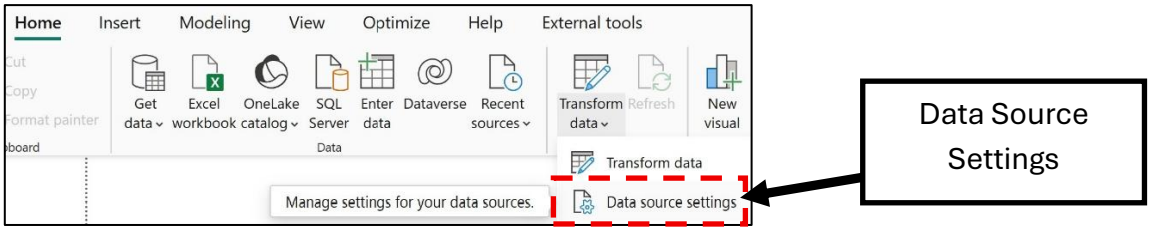
Where is Power Query in Power BI Desktop?

- Power Query is in the Home Tab in the Power BI Desktop Ribbon in the Data and the Queries groups, as shown in the picture below.
- You can connect to data using the commands in the Data group.
- You can open the Power Query Editor using the Transform data button.



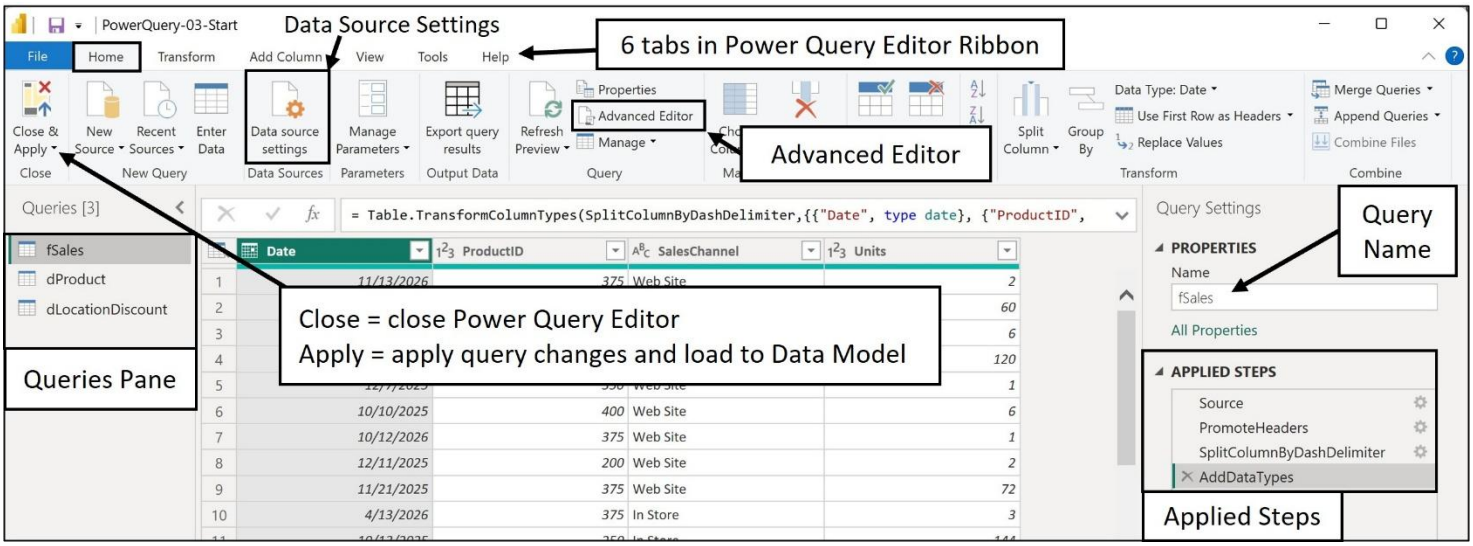
Data Source Settings in Power BI Desktop

- As we learned in video #2, Data Source Settings is where you manage connections to source data, such as on-premises file and folder paths. As shown in the picture below, in Power BI Desktop, these settings can be found on the Home tab, in the Queries group, by selecting the Transform data dropdown arrow and then choosing Data source settings from the menu.



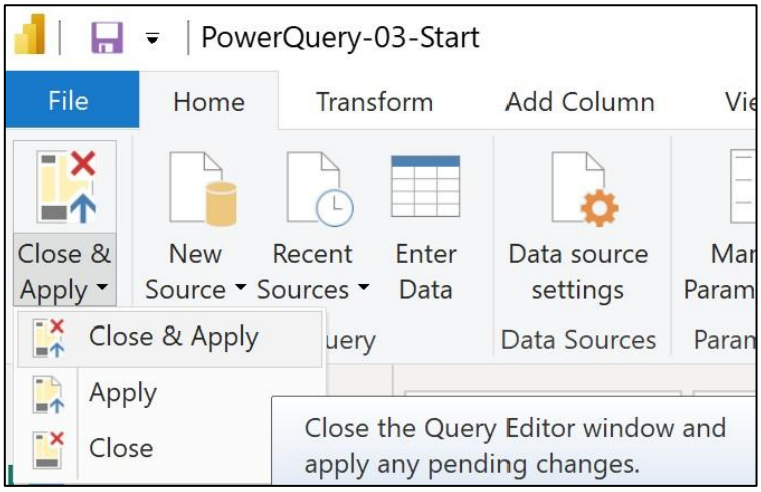
Power Query Editor in Power BI Desktop

- Power Query in Excel and Power BI Desktop are very similar.
- The M Code is identical except for few different functions.
- There are six tabs in the Power BI Desktop Power Query Editor, whereas in Excel there are only four, and some of the command buttons are in different locations.



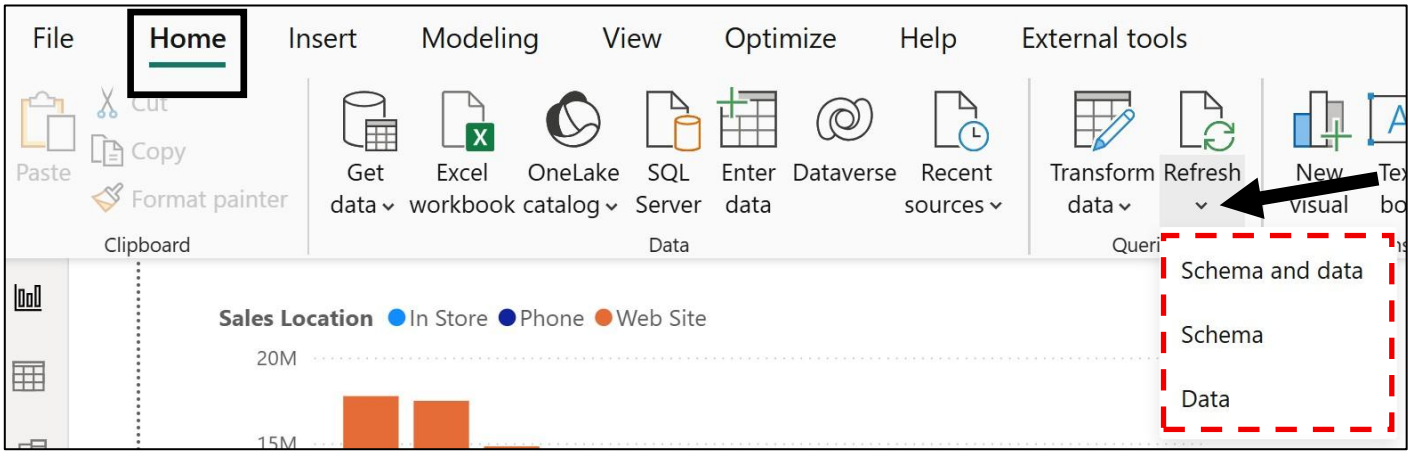
Apply & Close Button

- In Excel you load using the Close & Load option. In Power BI Desktop, you use the Close & Apply button to load to the only location that you can load: the Data Model.
- In the Power BI Desktop Power Query Editor, the Close & Apply button:
  - Close = closes the Power Query Editor window.
  - Apply = applies query changes and loads the queries to the Data Model.
  - Close & Apply = close editor and apply changes.



Refreshing a Visual or Report Dashboard

- The refresh button is in the Power BI Desktop Ribbon in the Home tab, Queries group, Refresh dropdown arrow.
    - Data = refreshes source data only (data in rows and columns), but not the structure of the queries.
    - Schema = refreshes the structure of the queries and is the same as clicking the Apply button.
    - Schema and data = refreshes data and queries.



Csv File Structure

- A **Csv file** contains a single table of data with comma delimiters to separate the bits of data. In the first row, the field names are separated by commas. In subsequent rows the data in each record are separated by commas. When we use Power Query to import the Csv file, it can correctly interpret the file as a table of data. Here is a picture of the Csv file we will use in this video:

```
Date,Description,Units¶
20261113,375-Web·Site,2¶
20261015,250-Web·Site,60¶
20261030,200-Web·Site,6¶
20260615,375-Web·Site,120¶
20251207,350-Web·Site,1¶
```

Excel File Structure & Excel Tables

- Whereas a Csv file contains a single table, **Excel files** can contain many tables and many objects. Power Query makes it easy to pick only the objects that you want from a given file.
- Objects that can be in an Excel file:
    - Worksheets.
    - Excel Tables (Ctrl + T).
    - Defined Names.
    - Print Areas.
    - Advanced Filter automatic Defined Names: Criteria and Extract.
    - Auto Filtered automatic and hidden Define Name: FilterDatabase.
- When you store tables of data in Excel files, you should convert proper tables into Excel Tables using the keyboard Ctrl + T. Although some other Excel objects may contain table-like data and can be

imported with Power Query, **Excel Tables** are the most reliable table structure and are consistently interpreted as tables by Power Query. Here is picture of the Excel Table that we will import:

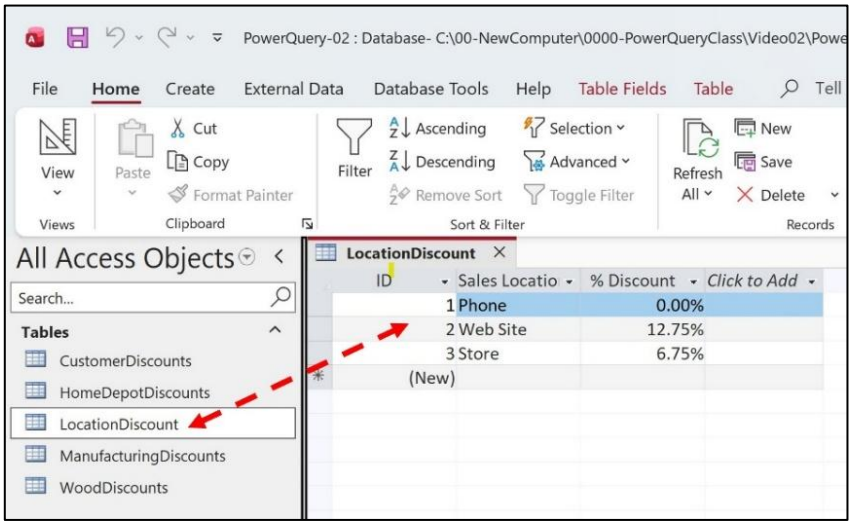


Lookup Table:

ProductID	Boomerang	Category	Flight Range (m)	COGS	Price
200	Quad	Freestyle	20	21.79	43.95
225	Carlota	Freestyle	20	17.28	34.95
250	Sunshine	Fast Catch	20	16.36	29.95
275	GelFast	Fast Catch	20	29.6	49.95
300	Eagle	Beginner	22	8.68	19.95
325	Aspen	Beginner	25	16.64	27.95
350	Tri-Fly	Freestyle	25	11.37	24.95
375	Bellen	Beginner	25	12.67	24.95
400	Yanaki	Mid-Range	30	20.23	36.75
425	SunBell	Mid-Range	35	14.83	29.95
450	Majestic	Long Range	40	21.52	44.95
475	Sunset	Long Range	45	19.66	39.95

Access Database File Structure

- Like Excel files, Access Database files can have many objects such as tables, queries, and reports. Power Query makes it easy to pick only the objects that you want from a given file. In the picture below you can see five tables on the left. We will import the LocationDiscount table.



Project Goal for Video 03

The company sells boomerang products and wants to summarize net sales by boomerang products and sales channel.

1) Import Source Data:

PowerQuery-02.csv
PowerQuery-02.xlsx
PowerQuery-02.accdb

1) Sales Table in Csv File (Fact Table)

```
Date,Description,Units
20261113,375-Web Site,2
20261015,250-Web Site,60
20261030,200-Web Site,6
20260615,375-Web Site,120
20251207,350-Web Site,1
```

2) Lookup Table in an Excel file (Dimension Table)

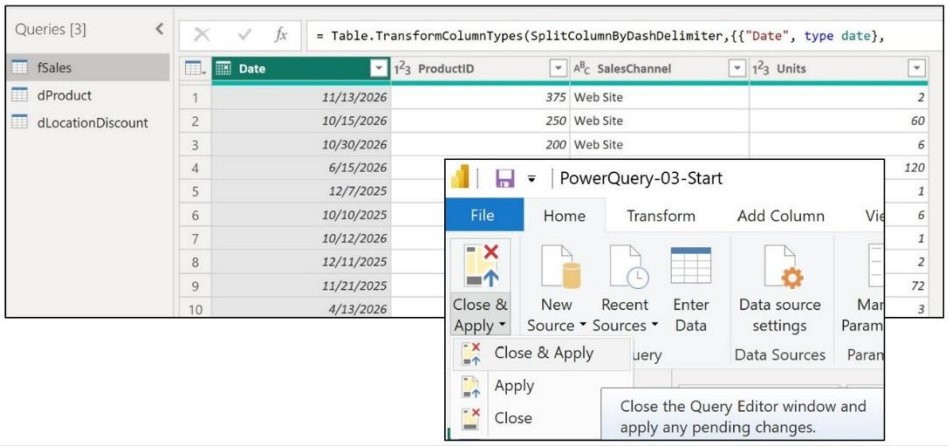
ProductID	Boomerang	Category	Flight Range (m)	COGS	Price
200	Quad	Freestyle	20	21.79	43.95
225	Carlota	Freestyle	20	17.28	34.95
250	Sunshine	Fast Catch	20	16.36	29.95
275	GelFast	Fast Catch	20	29.6	49.95
300	Eagle	Beginner	22	8.68	19.95

3) Lookup Table in an Access database file (Dimension Table)

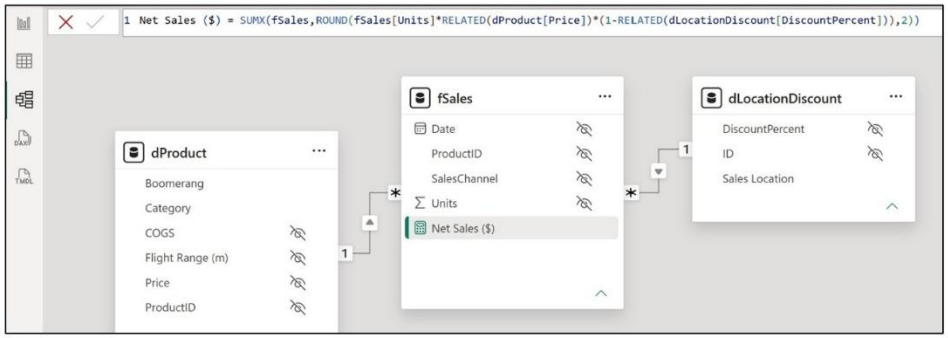
LocationDiscount

ID	Sales Locatio	% Discount	Click to Add
1	Phone	0.00%	
2	Web Site	12.75%	
3	Store	6.75%	

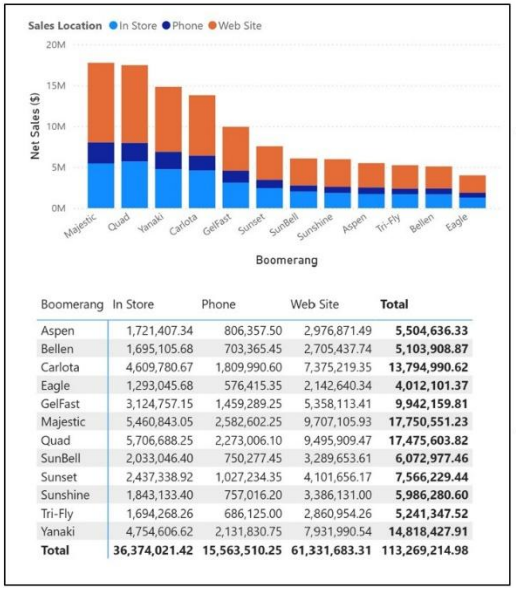
2) Clean, Transform, and Load to Data Model with Apply Button:



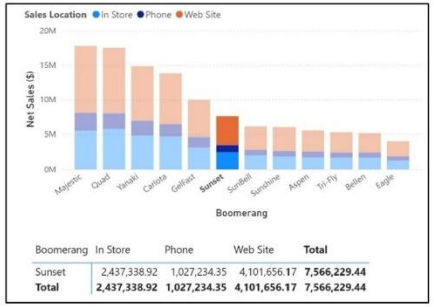
3) Build Data Model with Relationships & a DAX Measure



4) Build Interactive Visualizations



Boomerang	In Store	Phone	Web Site	Total
Aspen	1,721,407.34	806,357.50	2,976,871.49	**5,504,636.33**
Bellen	1,695,105.68	703,365.45	2,705,437.74	**5,103,908.87**
Carlota	4,609,780.67	1,809,990.60	7,375,219.35	**13,794,990.62**
Eagle	1,293,045.68	576,415.35	2,142,640.34	**4,012,101.37**
GelFast	3,124,757.15	1,459,289.25	5,358,113.41	**9,942,159.81**
Majestic	5,460,843.05	2,582,602.25	9,707,105.93	**17,750,551.23**
Quad	5,706,688.25	2,273,006.10	9,495,909.47	**17,475,603.82**
SunBell	2,033,046.40	750,277.45	3,289,653.61	**6,072,977.46**
Sunset	2,437,338.92	1,027,234.35	4,101,656.17	**7,566,229.44**
Sunshine	1,843,133.40	757,016.20	3,386,131.00	**5,986,280.60**
Tri-Fly	1,694,268.26	686,125.00	2,860,954.26	**5,241,347.52**
Yanaki	4,754,606.62	2,131,830.75	7,931,990.54	**14,818,427.91**
Total	**36,374,021.42**	**15,563,510.25**	**61,331,683.31**	**113,269,214.98**



Boomerang	In Store	Phone	Web Site	Total
Sunset	2,437,338.92	1,027,234.35	4,101,656.17	**7,566,229.44**
Total	**2,437,338.92**	**1,027,234.35**	**4,101,656.17**	**7,566,229.44**

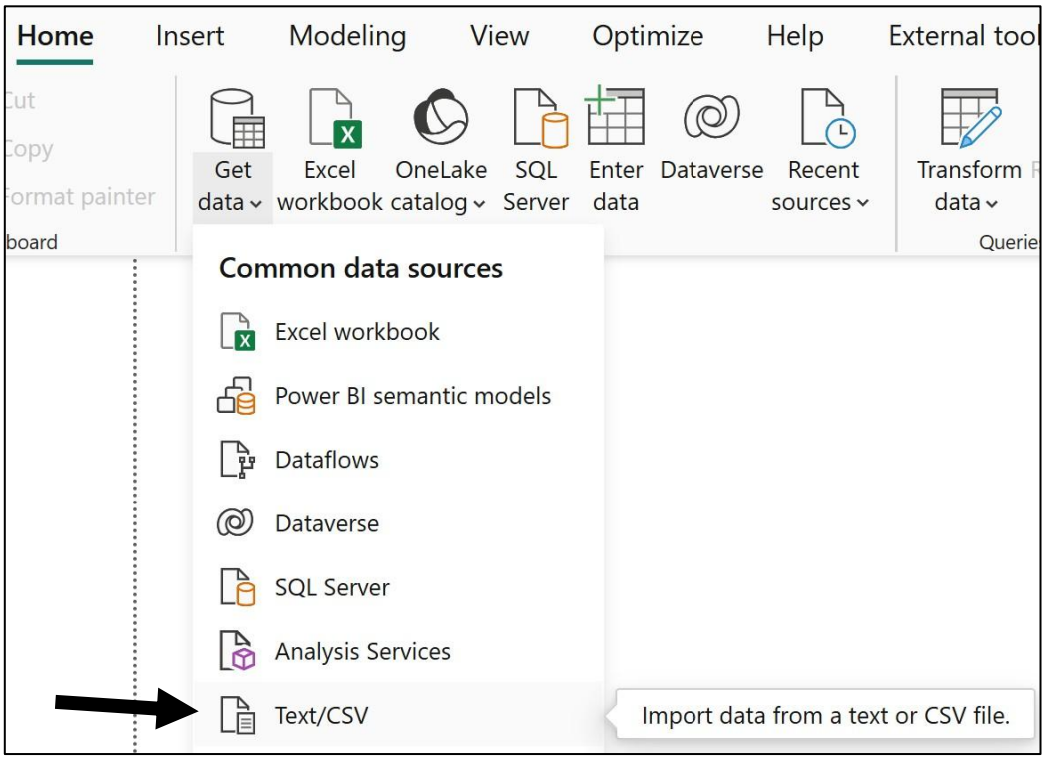
Steps in Project:

1) Download files

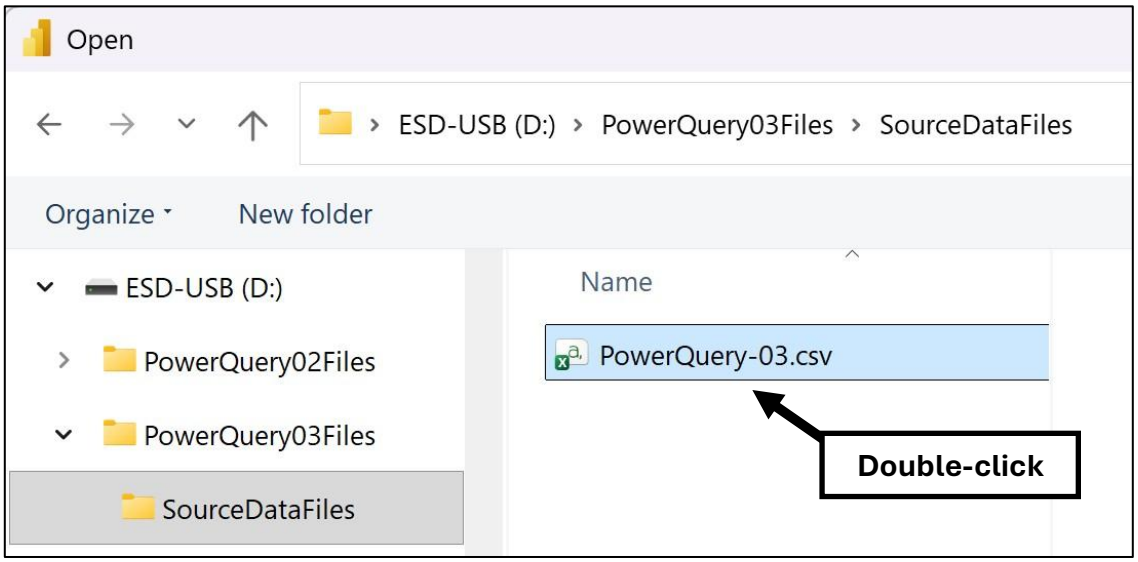
As we learned in the last video, download the zipped folder below Video #3, then unzip it in the same location you used for the previous video.

2) Import: Table from Csv File

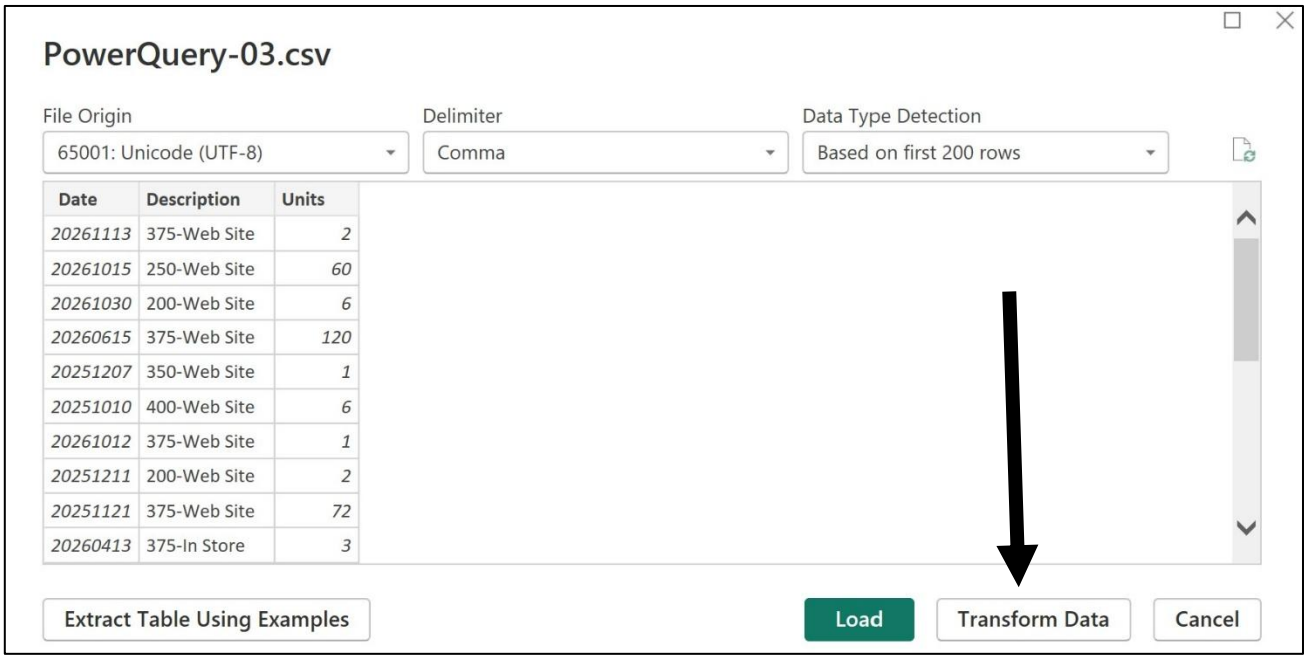
1. In the Power BI Desktop ribbon, on the Home tab, in the Data group, click the Get Data drop-down arrow, and then select Text/CSV from the menu, as shown in the picture below.



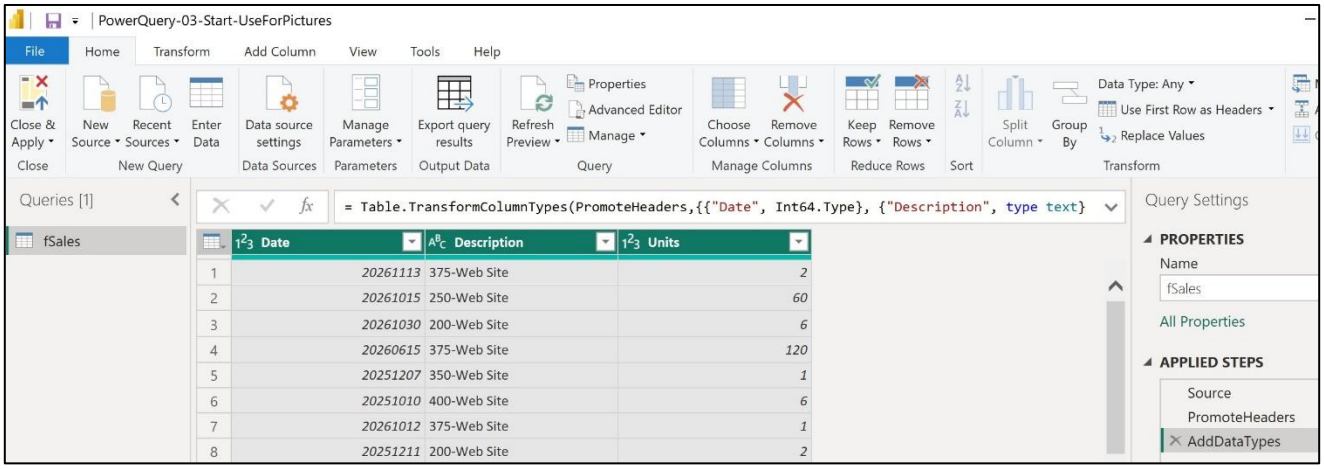
2. As shown in the picture below, in the Open dialog box, navigate to your source data folder, the double-click your "PowerQuery-03.csv" CSV file.



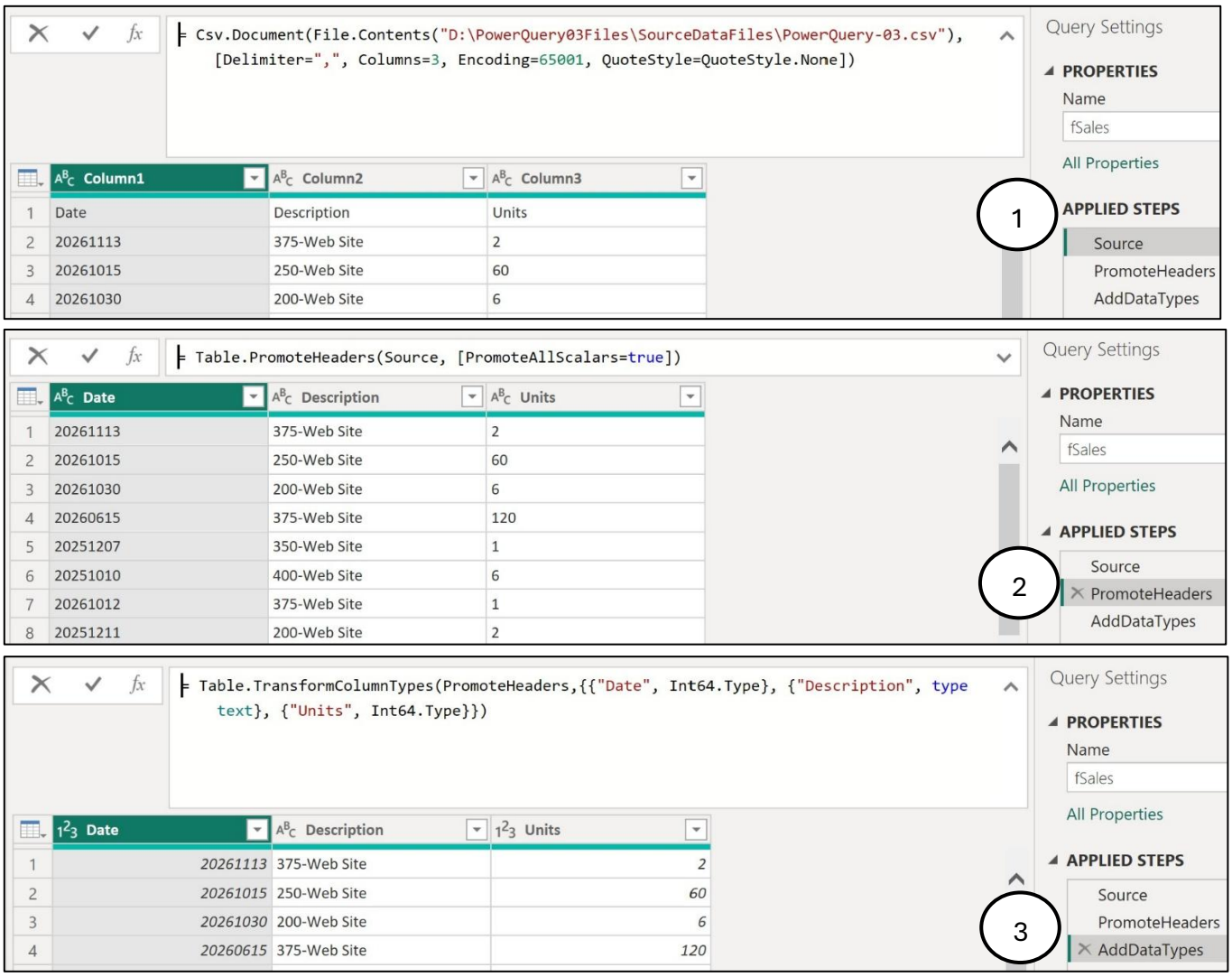
3. As shown below, in the next dialog box, make sure that the delimiter is Comma and then click the Transform Data button to open the Power Query Editor.



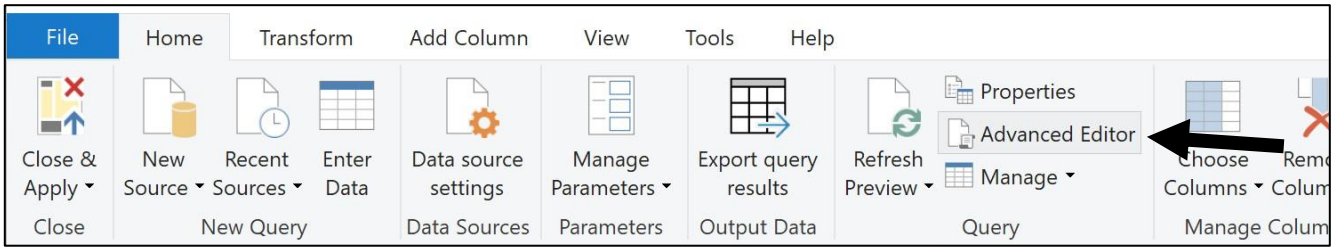
4. As shown below, when Power Query Editor opens, use the Query Settings pane on the right to make these changes: in the Name textbox, name the query “fSales”; in the Applied Steps list, select the second step, press F2, and rename it “PromoteHeaders”; then select the third step, press F2, and rename it “AddDataTypes”. Remember, the query name and query step names are identifiers. Remove all spaces from identifiers so the M code stays clear and uncluttered, without unnecessary quotation marks and pound signs.



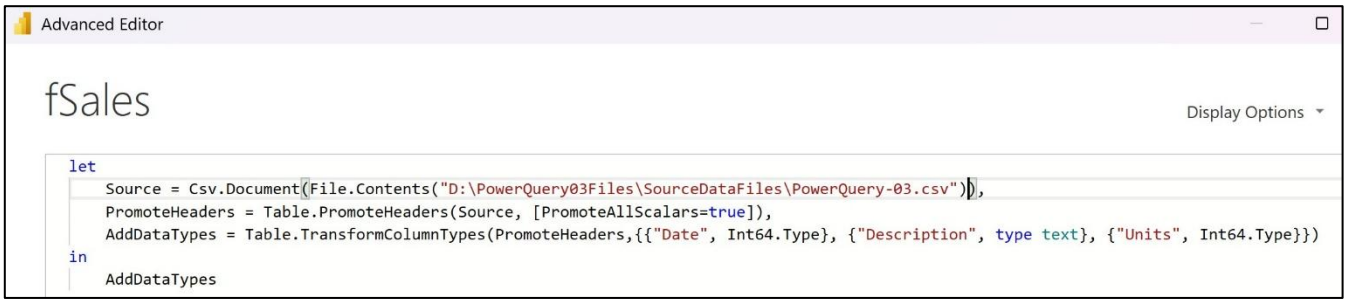
5. As shown in the three pictures on the next page, you can select each step and look at the formula bar to see the formula. The Identifier names for each step and the formulas are all part of the let expression that you can see in full in the Advanced Editor.



6. As shown below, to open the Advanced Editor, In the Home tab, Query group, click the Advanced Editor button.

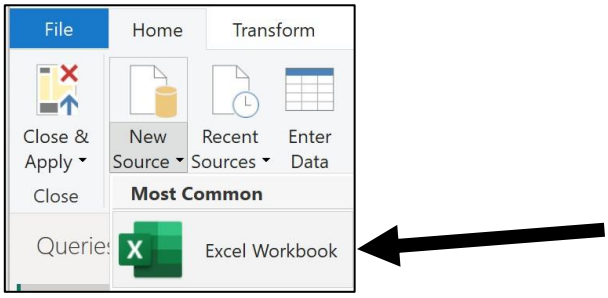


7. In the Advanced Editor, after you remove the record from the second argument of the Csv.Document function, the let expression should look like the picture shown below.

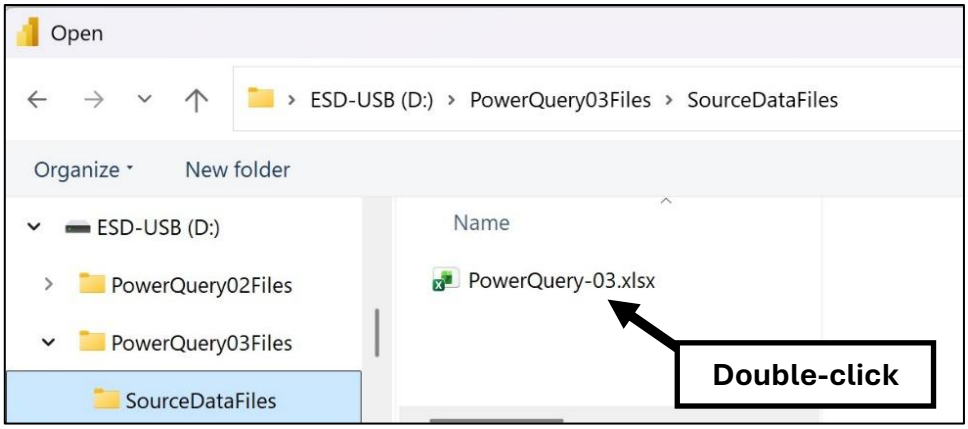


3) Import: Table from Excel File

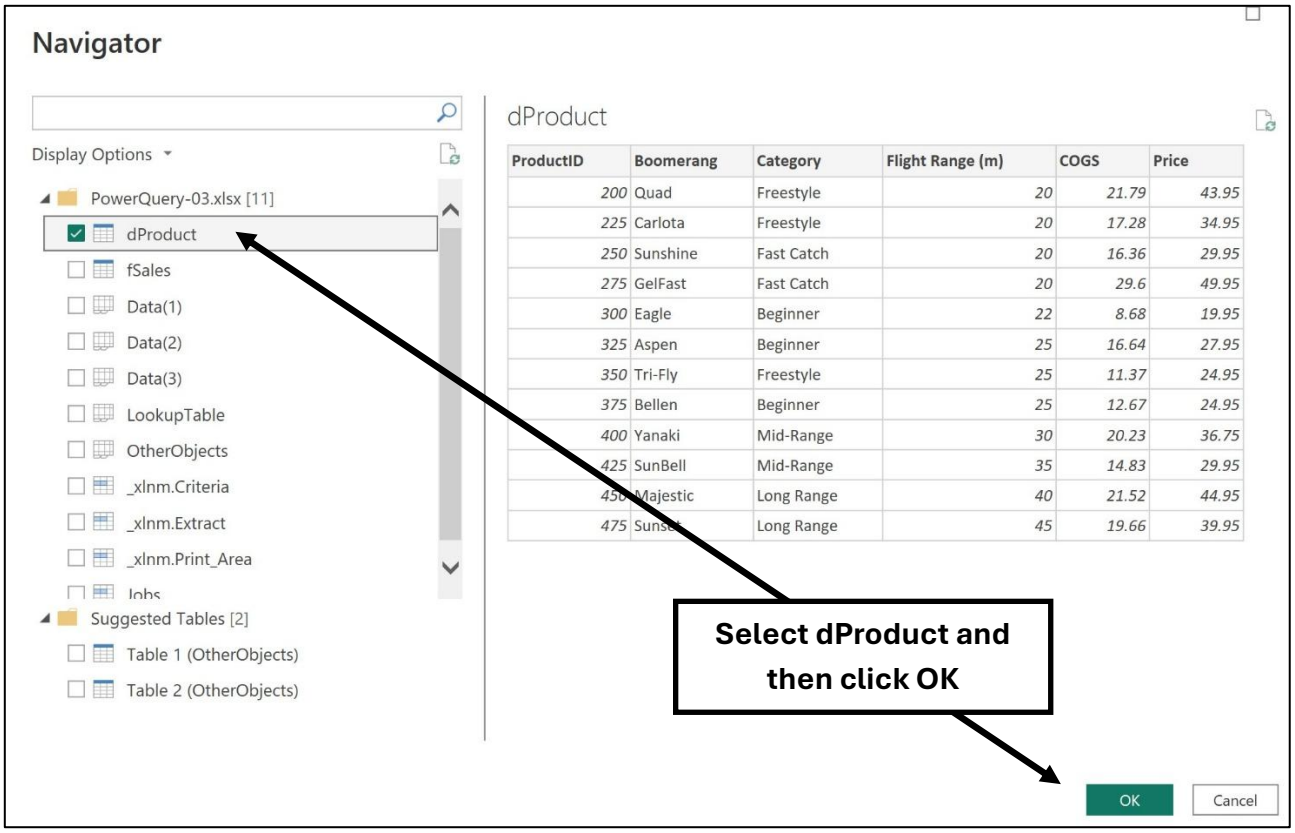
1. The next step is to import the product table from the "PowerQuery-03.xlsx" Excel file. While still in the Power Query Editor, in the Home tab, in the Data group, click the New Sources dropdown arrow and select Excel Workbook, as shown below.



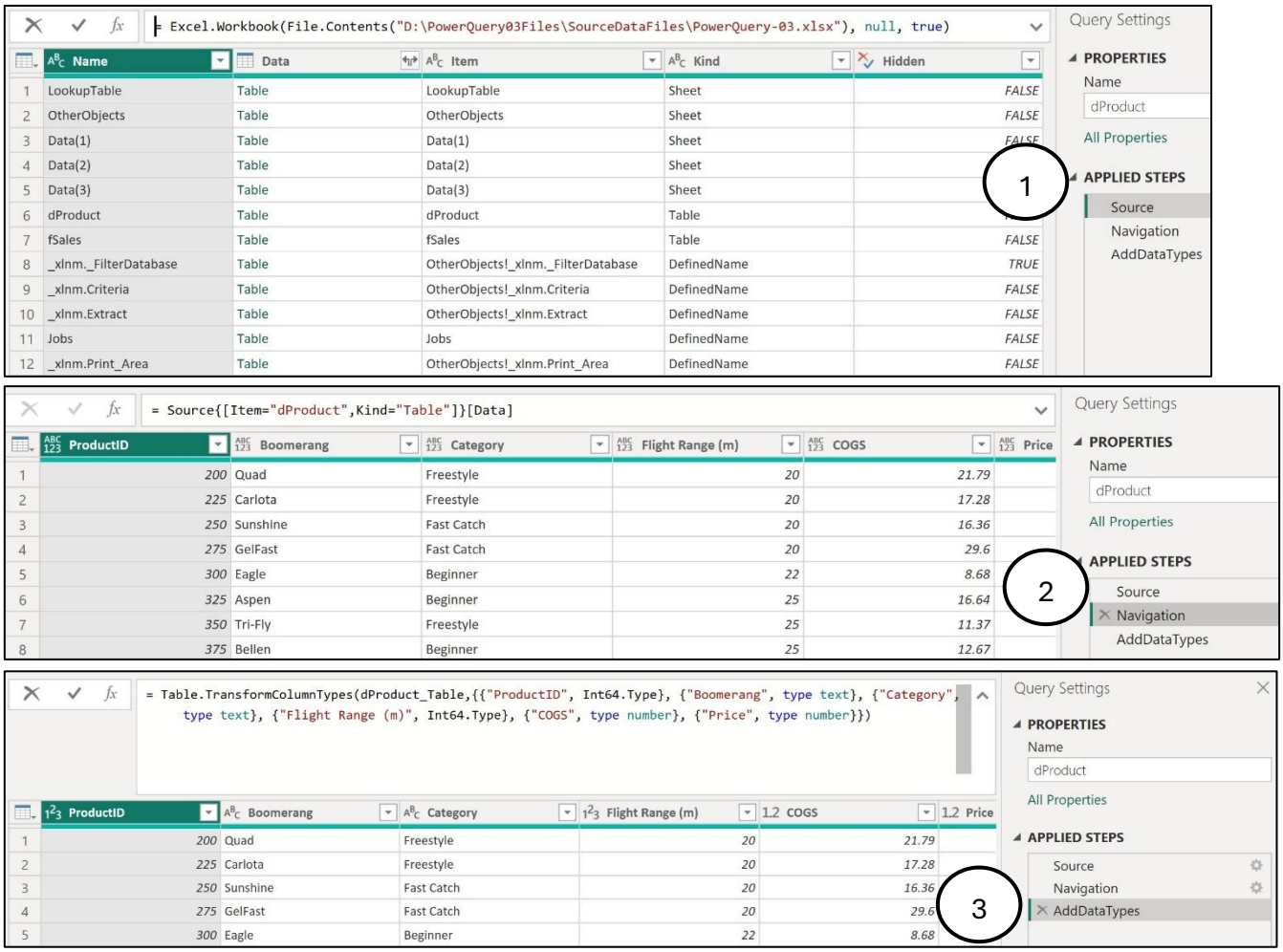
2. As shown below, in the Open dialog box navigate to the Excel file and double click to open.



3. As shown below, in the Navigator dialog box, select the Excel Table object, dProduct, then click OK.



4. As shown below, rename the last query step to “AddDataTypes”, then select each query step to see the M Code formula in the Formula Bar.

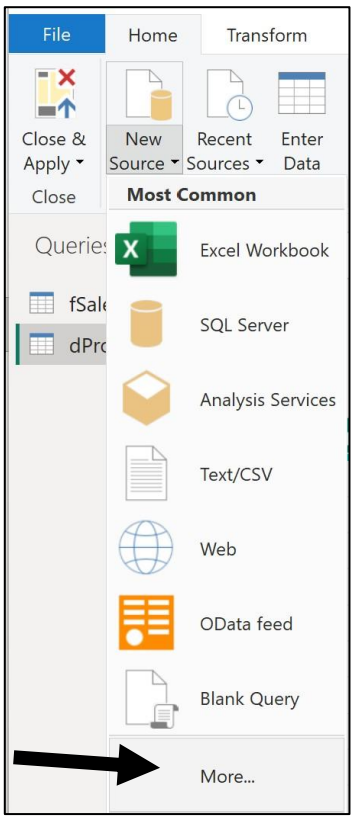


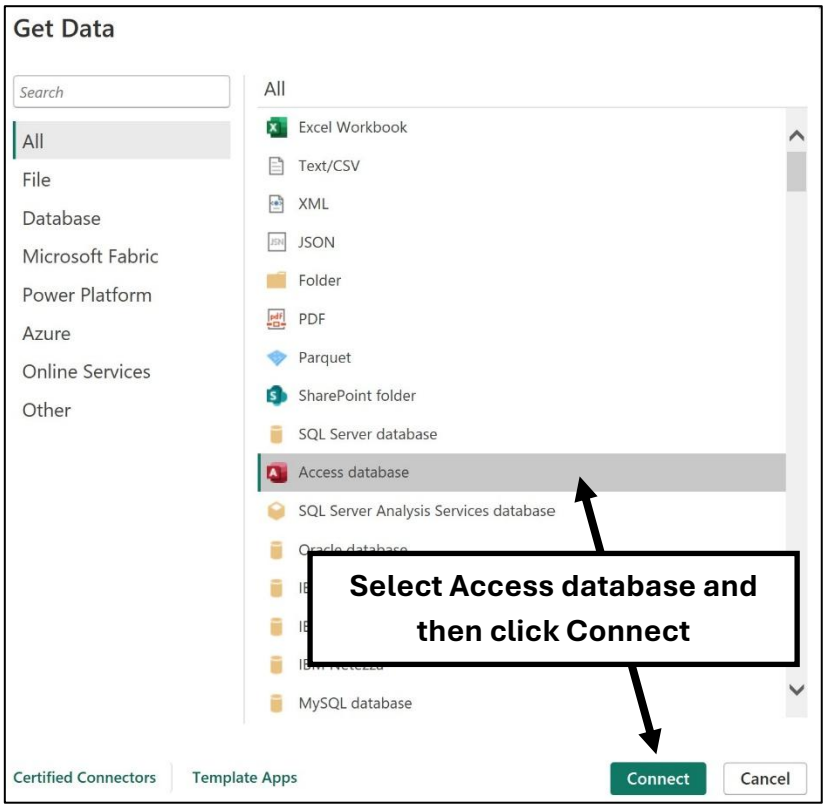
5. As shown below, you can open the Advanced Editor to see the full query let expression.



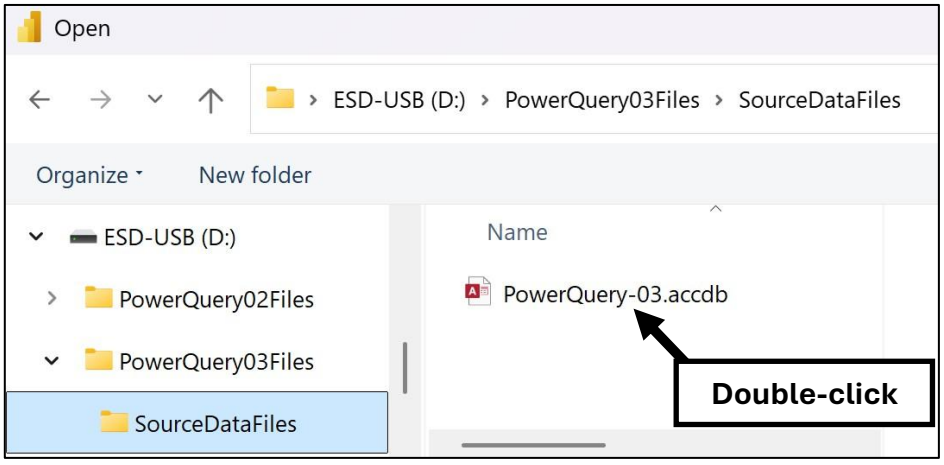
4) Import: Table from Access Database File

1. The next step is to import the location discount table from the "PowerQuery-03.accdb" file. While still in the Power Query Editor, in the Home tab, in the Data group, click the New Sources dropdown arrow and select More..., as shown below on the left. Then as shown on the right, select the Access database and then click Connect.

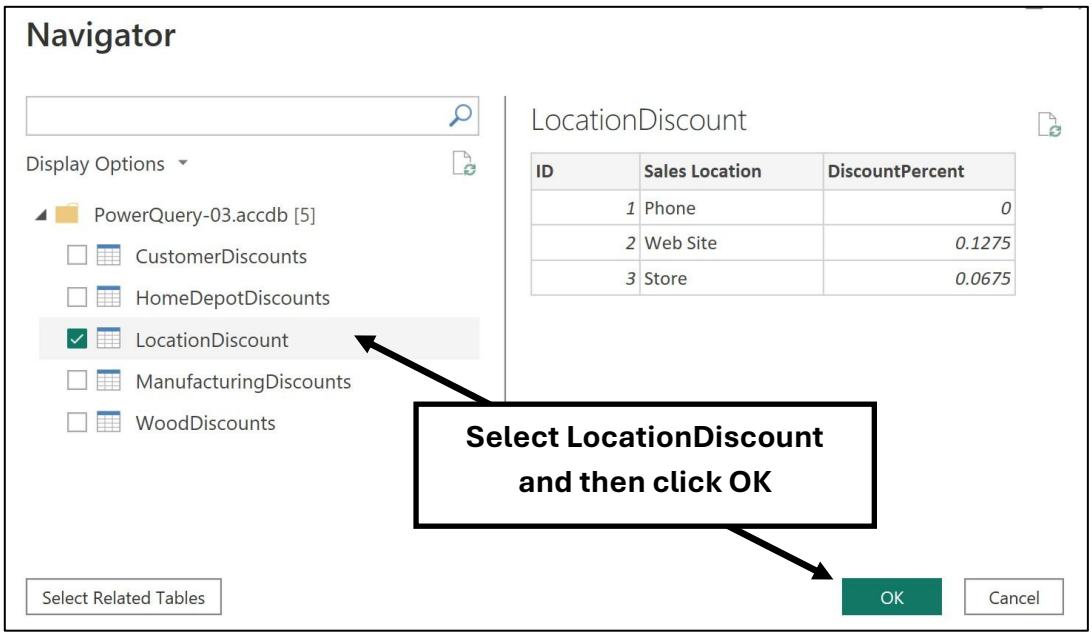




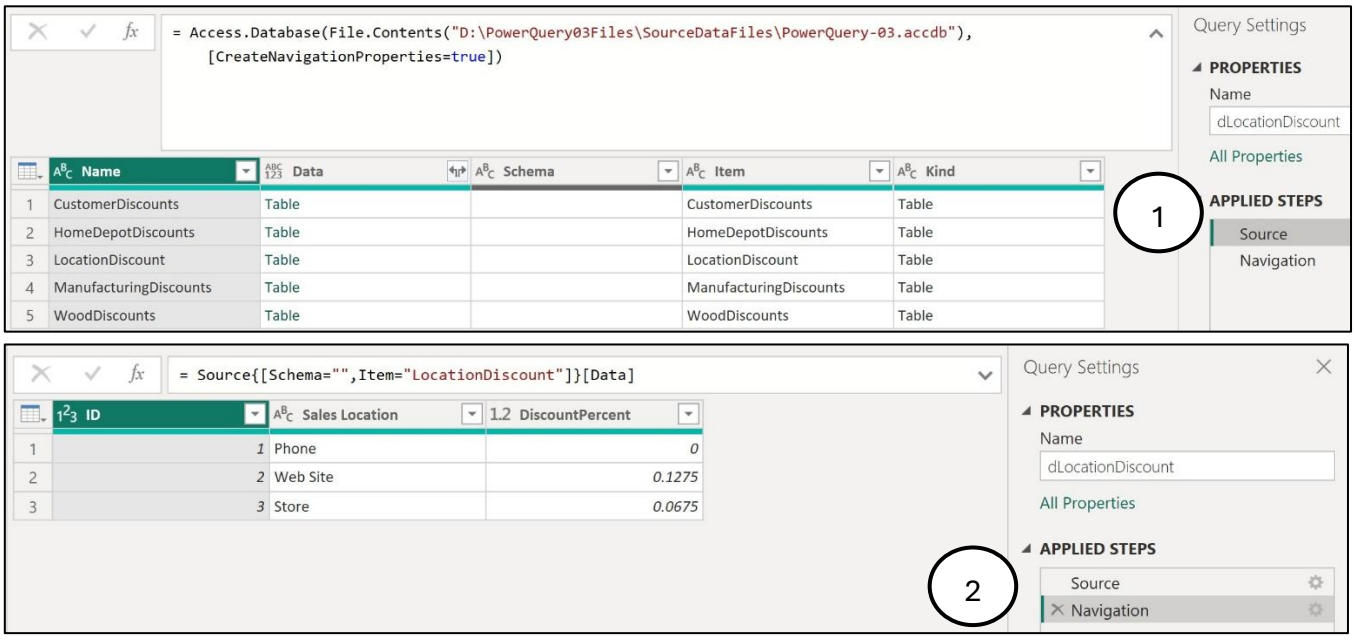
2. In the Open dialog box, double click your access file.



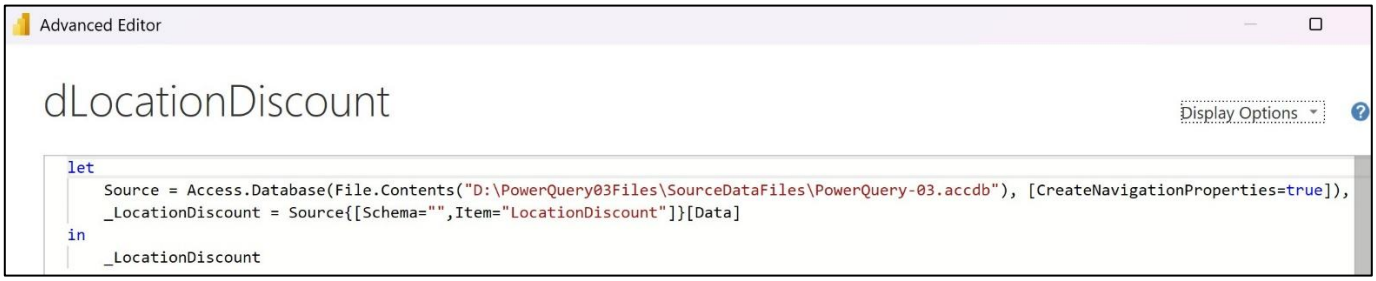
3. As shown on the next page, in the Navigator dialog box, select the Access Table object, LocationDiscount, then click OK.



4. As shown below, select each query step to see the M Code formula in the Formula Bar. Because the table came from a structured database that assigns data types, the data arrived in Power Query with data types already applied. For this reason, the Access query is the only query that does not have the third query step to assign data types.

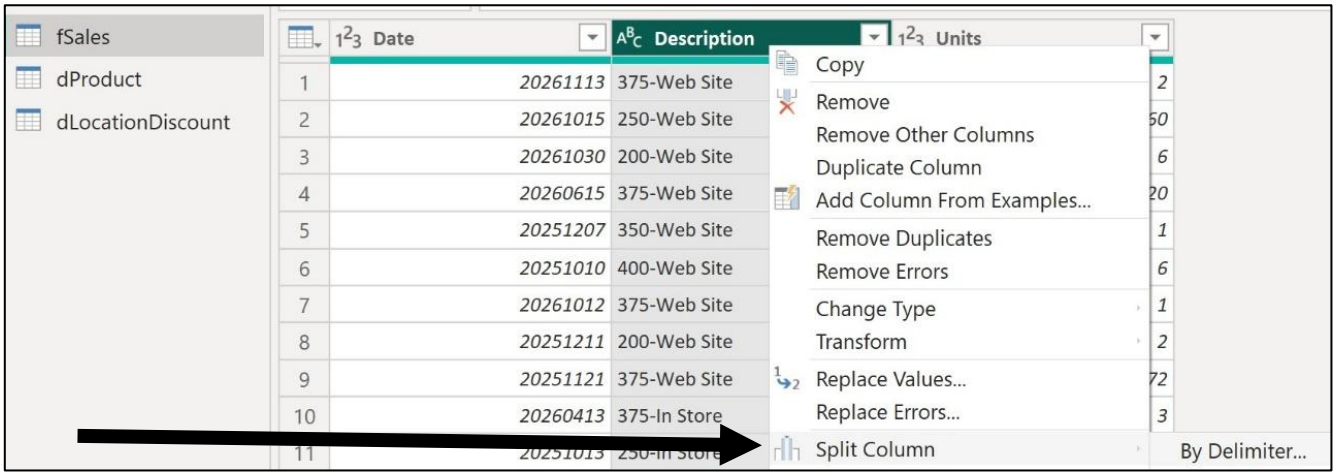


5. As shown below, you can open the Advanced Editor to see the full query let expression.

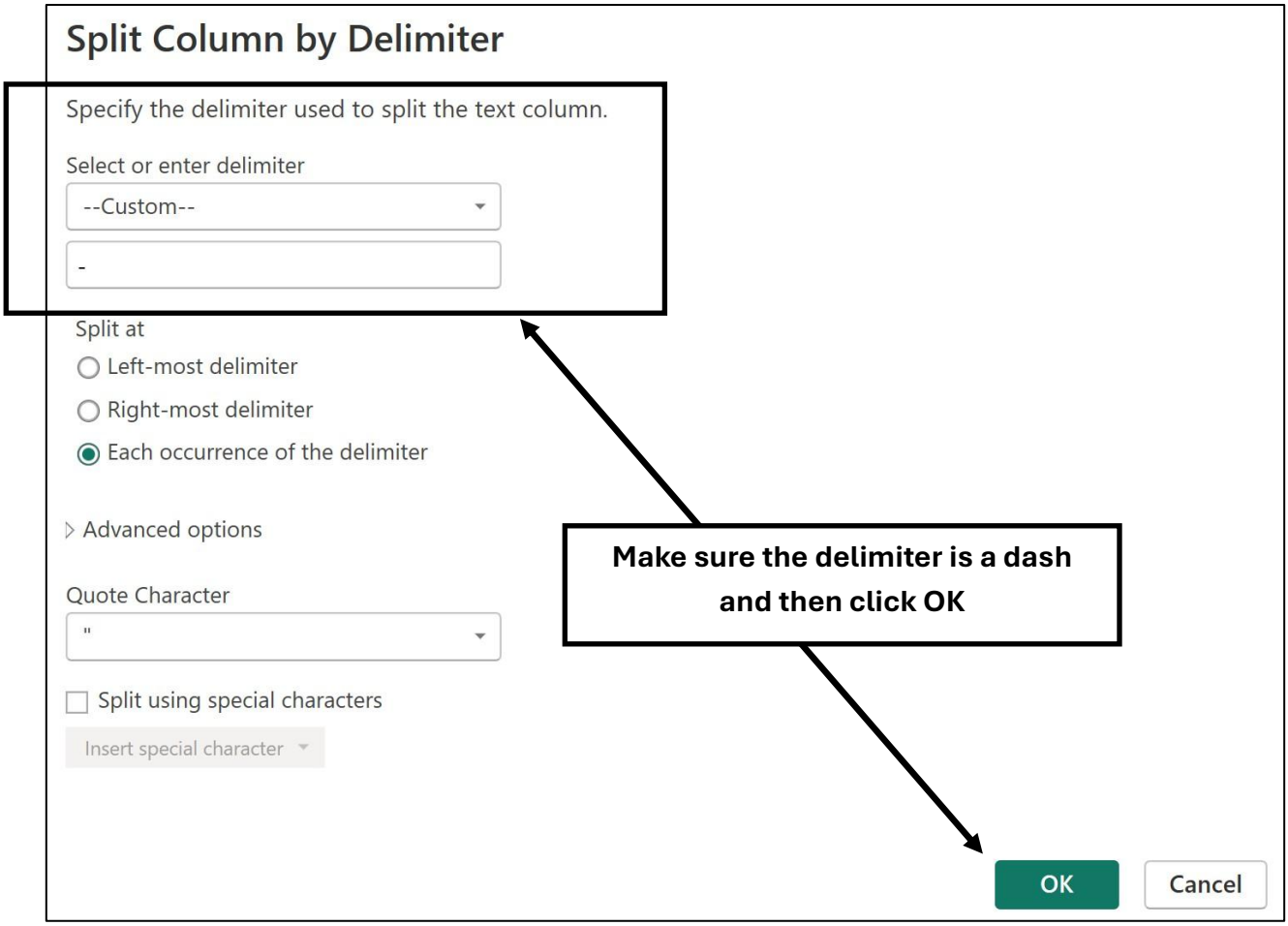


5) Clean: Split Column by Delimiter

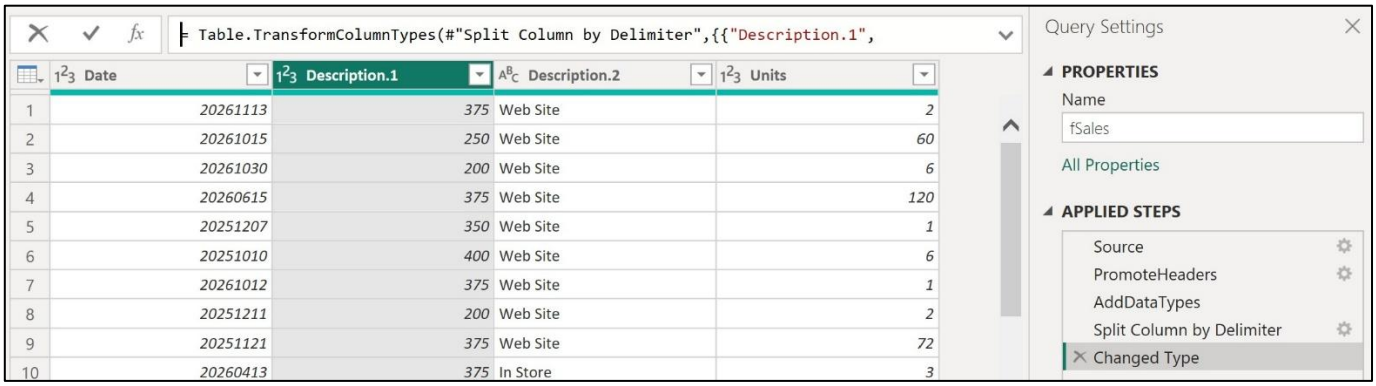
1. As shown below, the Description column in the fSales table contains the product ID and sales channel separated by a dash delimiter. The product ID is needed as a number value in a separate column so that it can be used as a lookup value to get product boomerang name and price. The sales channel is needed as a text value in a separate column so that it can be used as a lookup value to get the discount rate needed to calculate net sales. To spilt the column by the dash delimiter, right-click the Description field name and from the menu hover over Split Column, then from the sub-menu select By Delimiter.



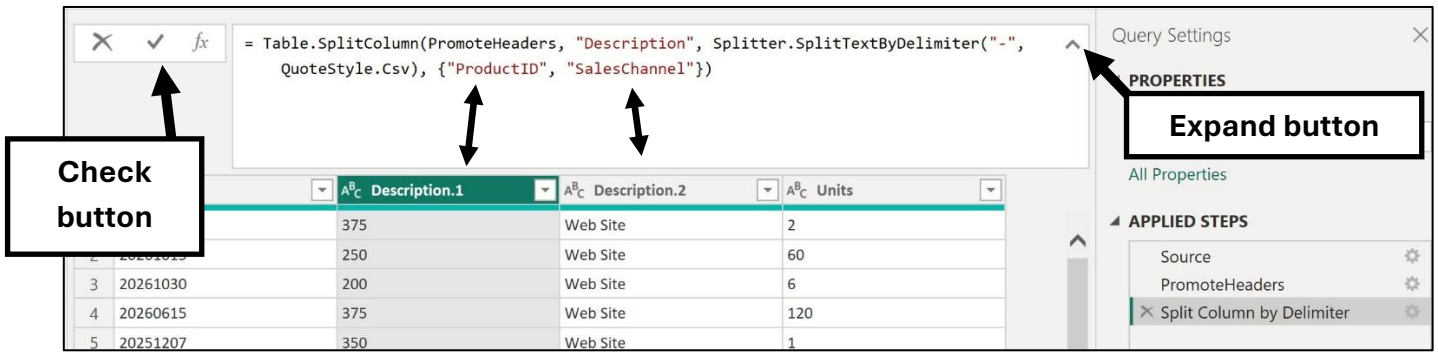
2. In the Split Column by Delimiter dialog box, select the dash delimiter and then click OK.



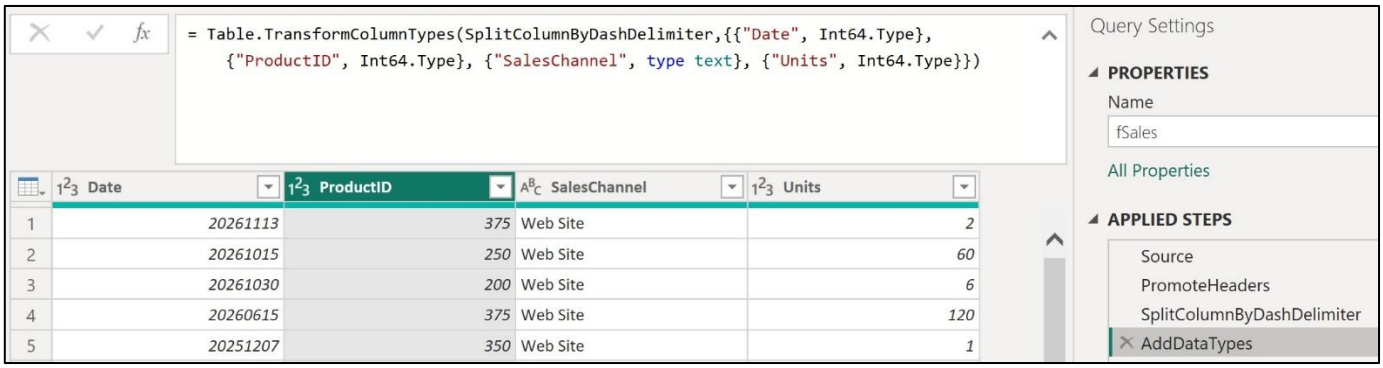
3. As shown below, Two query steps were added: Split Column by Delimiter and Changed Type. We want to keep the Split Column by Delimiter step so that we can edit the M Code in the formula bar and assign more informative field names that Description.1 and Description.2. However, to create more efficient M Code (fewer steps in the let expression), use the red X next to the query steps AddDataTypes and Changed Types to delete both steps.



4. As shown below, select the Split Column by Delimiter step and expand the formula bar. In the fourth argument of the Table.SplitColumn function, edit the default names so they become “ProductID” and “SalesChannel”. The new field names must be text values, and they must be separated by a comma and surrounded by curly brackets, so what you enter into the fourth argument is the correct list value shown here: {"ProductID", "SalesChannel"}. As studied in the last video, everything you type into M Code must be one of the 15 allowable value types, such as text values and list values. When your formula is done, use the check button on the left side of the formula bar to enter the formula.

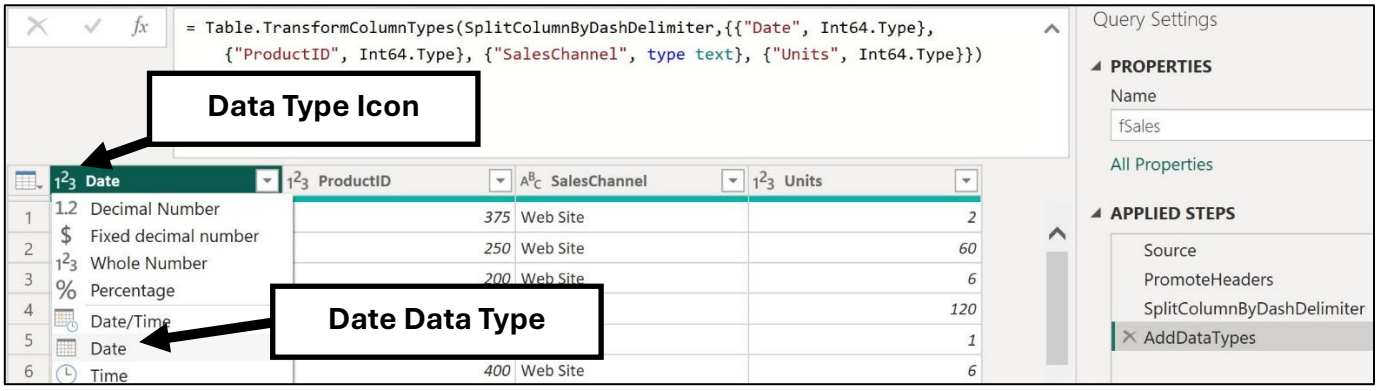


5. After you completing the third query step, add data types, then rename the query steps so that they do not contain spaces.



6) Clean: Convert ISO Dates to Real Dates

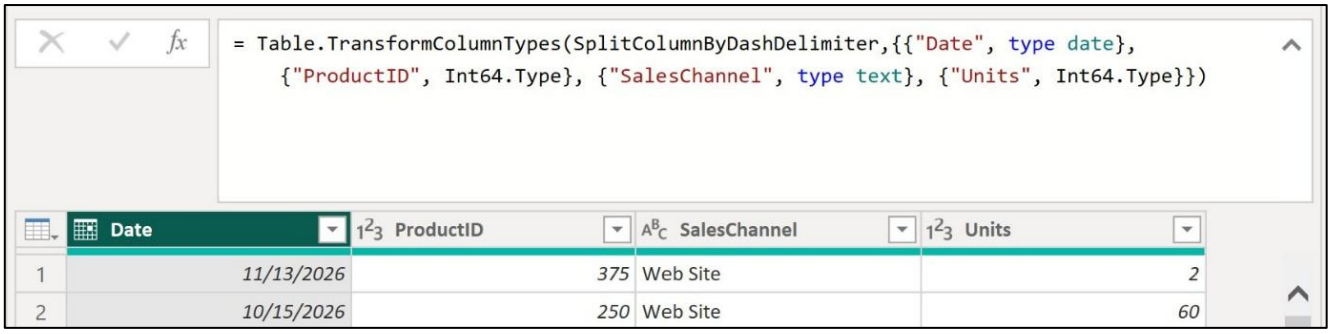
1. The next cleaning task is to convert the ISO Dates to real dates. With the fourth query step selected, as shown below, on the left side of the Date field name, click the 1 2 3 Data Type Icon, then from the menu select Date.



2. As shown below, in the Change Column Type dialog box, select Replace current.



3. As shown below, you can see in the formula bar that the Data Type for the Date column has been changed from Whole Number, Int64.Type, to Date, type date.

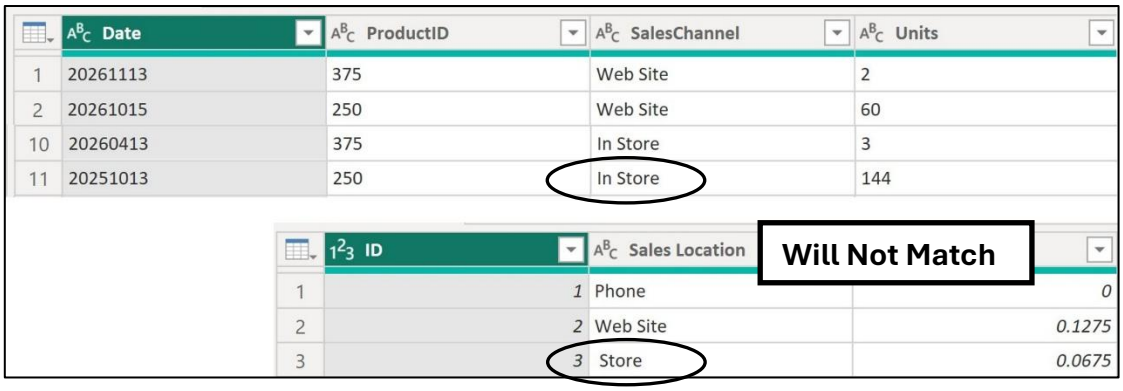


4. As shown below, you can open the Advanced Editor to see the full query let expression for the finished fSales query.

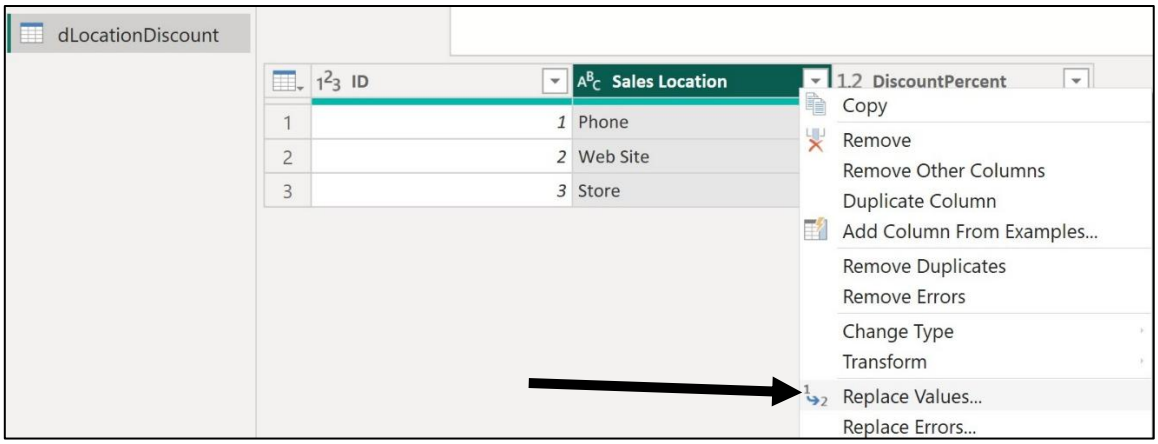
5. 

7) Cleaning: Replace Values So Relationship in Data Model Can Function Correctly

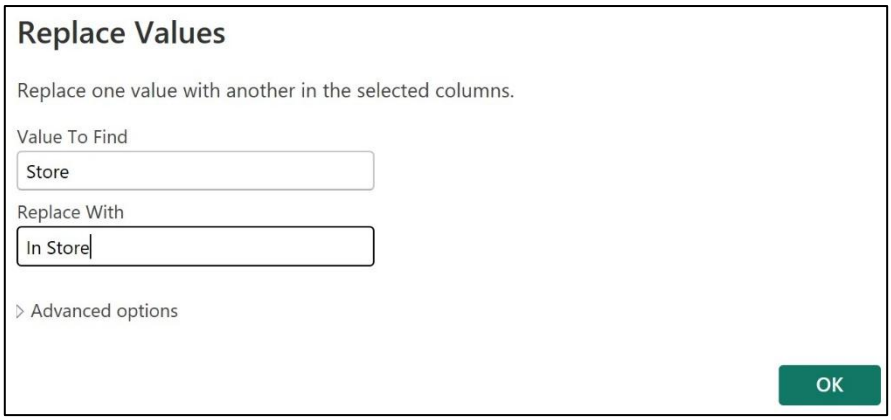
1. As shown in the picture at the top of the next page, the fSales "In Store" lookup value does not match the "Store" value in the dLocationDiscount table. In the Data Model, you need a one-to-many relationship between the Sales Location field in the dLocation table and the SalesChannel field in the fSales table. When lookup values do not match the values in the primary key of the lookup table, the relationship will not work. To fix this problem, use the Replace Values feature in Power Query to replace "Store" with "In Store" in the dLocationDiscount table.



2. As shown below, select the last query step in the dLocationDiscount table, right-click the Sales Location field name, and from the menu select Replace Values.



3. As shown below, in the Value To Find textbox enter "Store", in the Replace With textbox enter "In Store", and click OK. Replace spaces in the new query step name.

4. 

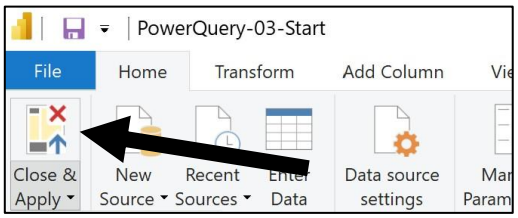
5. As shown below, you can open the Advanced Editor to see the full query let expression.



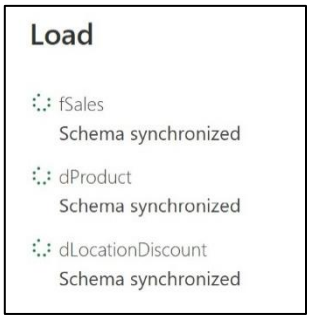
6.

8) Load three tables to the Data Model

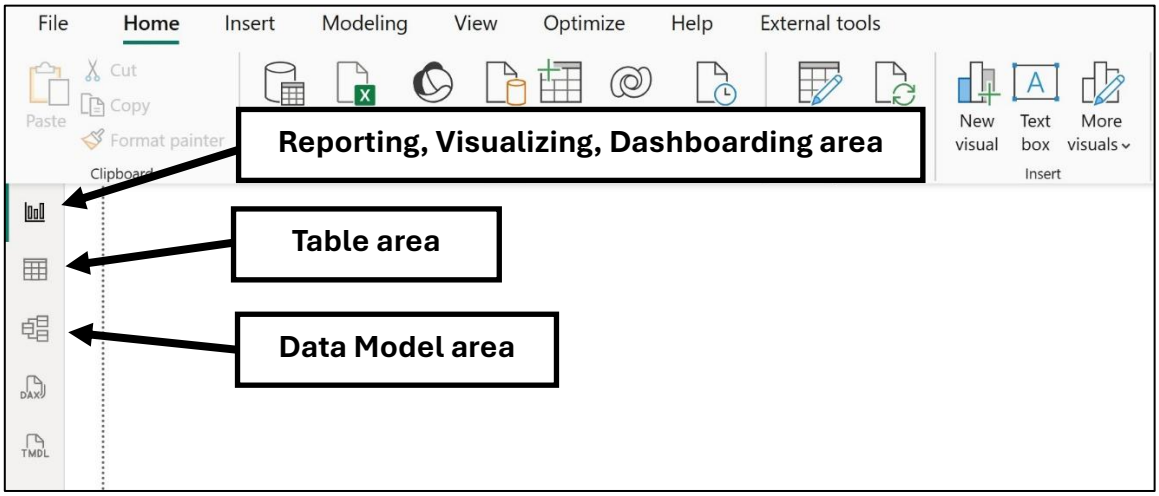
1. Now that the three queries are completed, as shown below, in the Power Query Editor, in the Home tab, in the Close group, click the Close & Apply button. The Apply command runs the query steps for the three queries and loads them as three separate tables into the Data Model. The Close command closes the Power Query Editor.



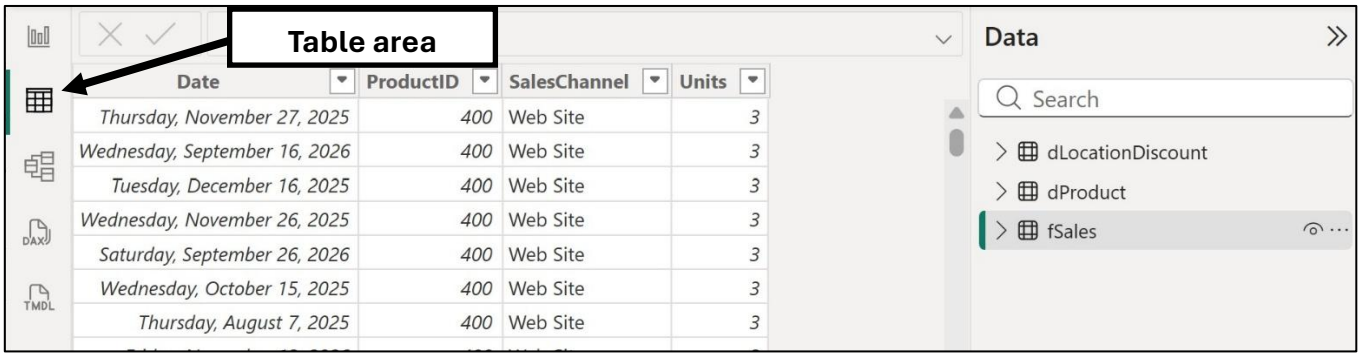
2. As shown below, the Load message box pops up and let's you know that the query steps are being applied and that the Data Model Columnar Database is compressing the tables and getting them ready for you to use.



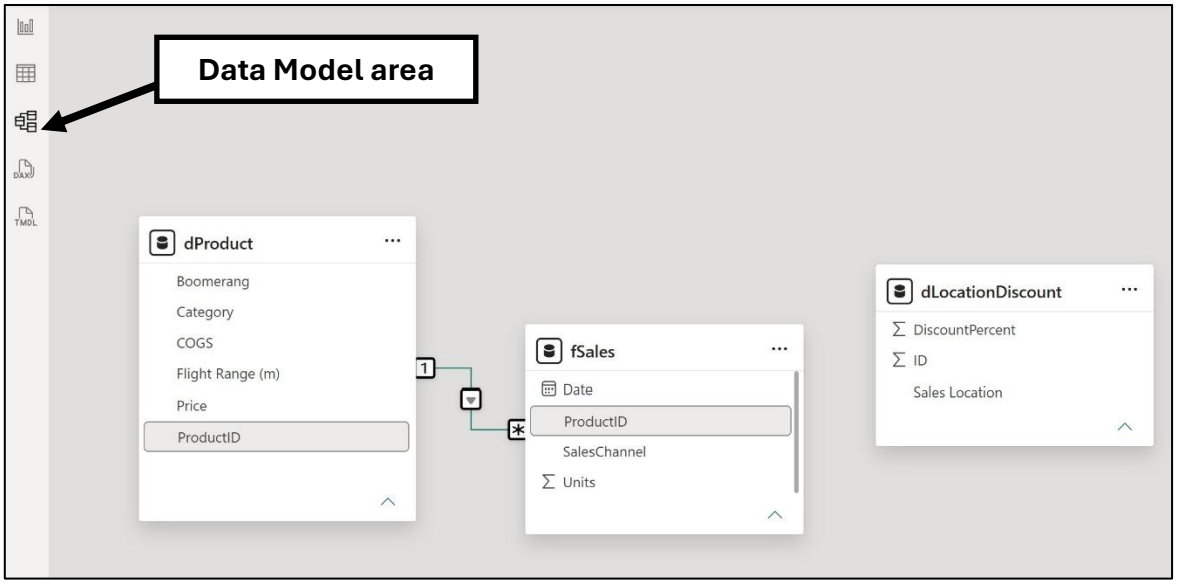
3. After the load is complete, the three areas that we will work in are shown below.



4. As shown below, on the left, click the Table icon. On the right, in the Data pane, you can click on each table to view the results.



5. As shown below, the one-to-many relationship between the ProductID field in the dProduct table and the ProductID field in the fSales table was automatically created.



Star Schema Data Model or Semantic Model

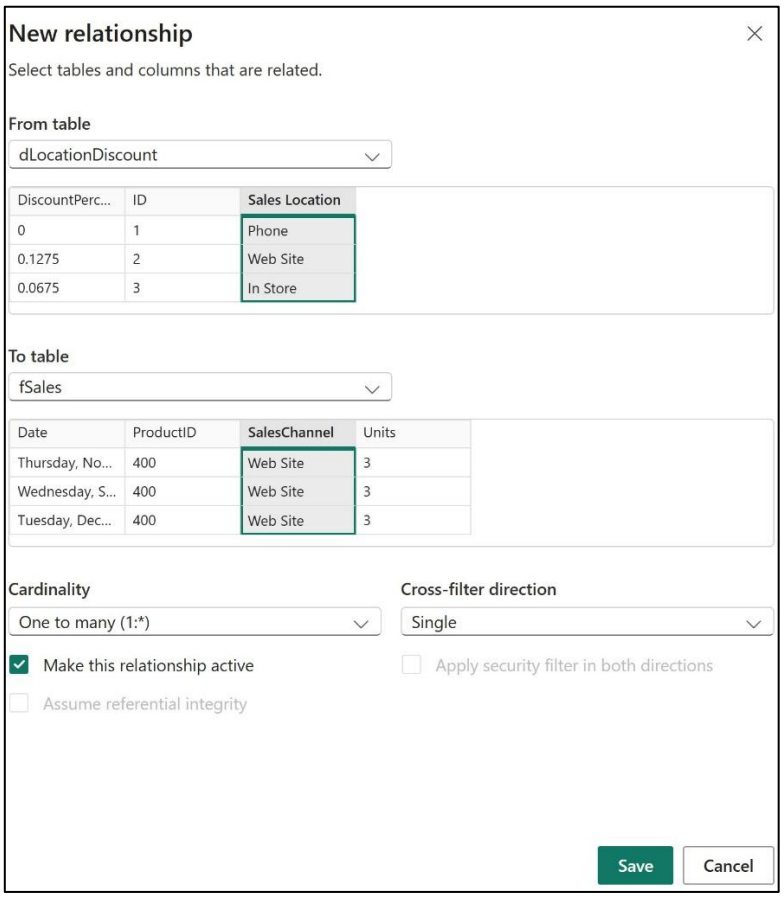
- These models present a mapping of the tables, columns, hierarchies, relationships, measures and other model components that sit between the underlying data sources and the workbooks, reports, dashboards and analysis. Semantic models should match the structure and data security needs of the business. Sematic models serve as the single source of truth that end users can connect to easily and interpret easily so that they can create consistent, data-driven and useful information for decision makers.

9) Create One-To-Many Relationship

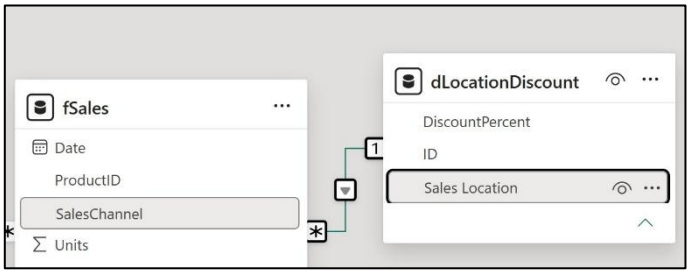
1. To create the the one-to-many relationship between the Sales Location field in the dLocationDiscount table and the SalesChannel field in the fSales table as shown below, click and drag the Sales Location field in the dLocationDiscount table over the SalesChannel field in the fSales table, and drop.



2. The New relationship dialog box pops up. As shown below, make sure your dialog box matches the one below (it should be all automatic), and then click Save.

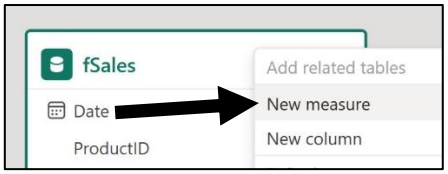


3. As shown below, the new relationship shows the **1 to *** icon to indicate that it has been created.



10) Create the Net Sales DAX Formula Measure (Filter Context & Row Context)

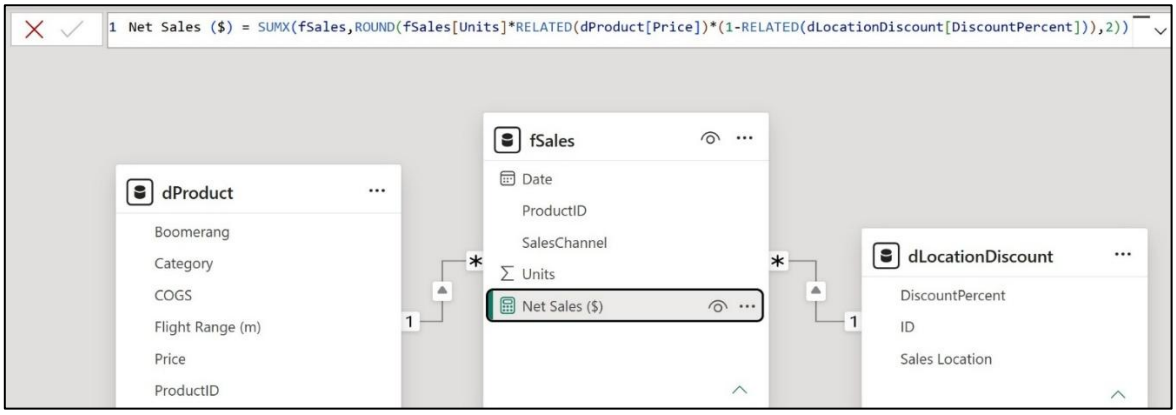
1. To create the DAX Measure for net sales, in the Data Model area, right-click the grey area in the fSales table, as shown below, and select New Measure.



2. In the formula bar type the DAX Measure as shown below. This Measure uses the SUMX function. In the first argument of SUMX is the fSales table. In the second argument you create the formula to calculate net sales. This formula iterates down every row of the fSales table to calculate the value for each row. That column is then added by SUMX to get the net sales. The formula in the second argument of SUMX uses the RELATED function twice, and through the One-To-Many Relationships is able to lookup the correct values for each row in the fSales table and use those two values with the rest of the formula, including the ROUND function, to calculate net sales.

```
Net Sales ($) =
SUMX (
    fSales,
    ROUND (
        fSales[Units] * RELATED ( dProduct[Price] )
            * ( 1 - RELATED ( dLocationDiscount[DiscountPercent] ) ),
        2
    )
)
```

3. The finished formula in the Data Model is shown below. Notice that the Measure is listed in the fSales table.

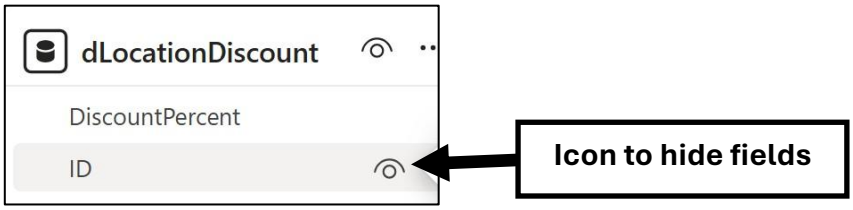


4. **Row Context** allows column reference to see the values in each row of the table. Row Context happens in: 1) Calculated Columns, and in 2) Iterator Functions such as SUMIFS & others.

5. **Filter Context** happens when you drop Measures into visuals and reports. The conditions and criteria from the reports and filters will filter the underlying dimension tables, the filter is passed through the relationships to the fact table, and the underlying fact table contains only rows that match the conditions. In this way, DAX Measures can handle big data calculations better than formulas in the worksheet. For more about DAX formulas you can watch my DAX video at YouTube: DAX Video from Mike excelisfun Girvin

11) Hide Fields Not Needed in the Reporting Area (No Implicit Measures)

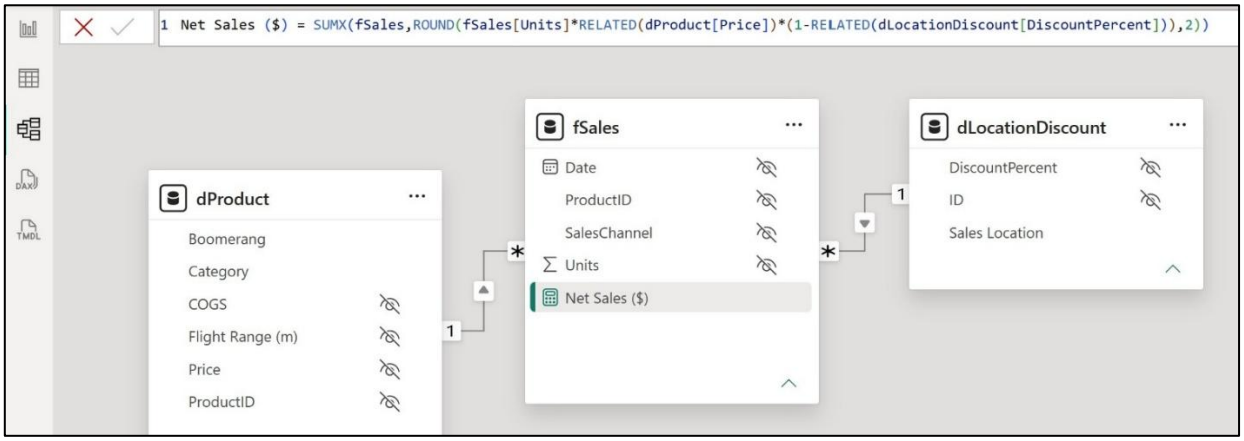
1. Don't drag data columns into visuals because they create hidden Implicit Measures and hidden data tables. Instead, create Explicit Measures (ones that you create) and then hide all the data columns. When you drag and drop data and attribute fields from Fact tables, it is very bad modeling techniques and leads to significant inefficiencies.
2. When building a Data Model, it is important to hide fields that you do not need in the reporting area. This means that you do not want them to show up in the field list in the reporting area. The fields that you do not want are primary ket fields and number and date fields from the fact table. If you drag number or date fields from a fact table into a visual, you can get hidden Implicit Measures and Date Tables which can cause problems. As shown below, you can hide fields using the hide icon to the right of each field name.



3. After you click the hide icon, the icon shows a diagonal like to indicate that the field is hidden in report view. Hide all the fields except Boomerang and Category in the dProduct table, Net Sales in the fSales table, and Sales Location in the dLocationDiscount table. The picture below shows the results.

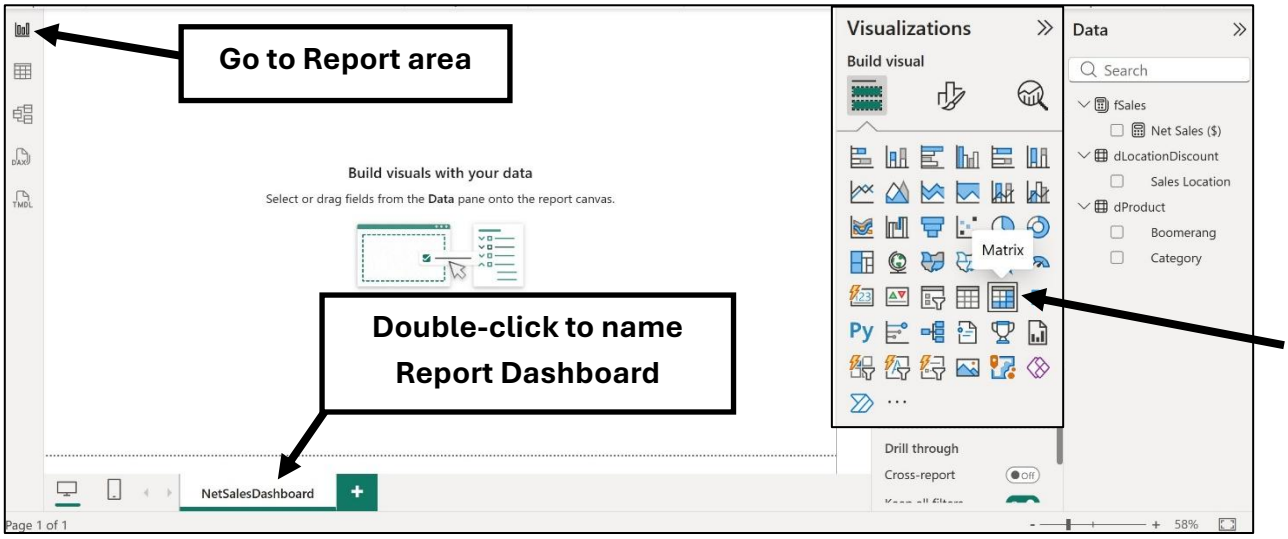


4. The finished Data Model is shown below.

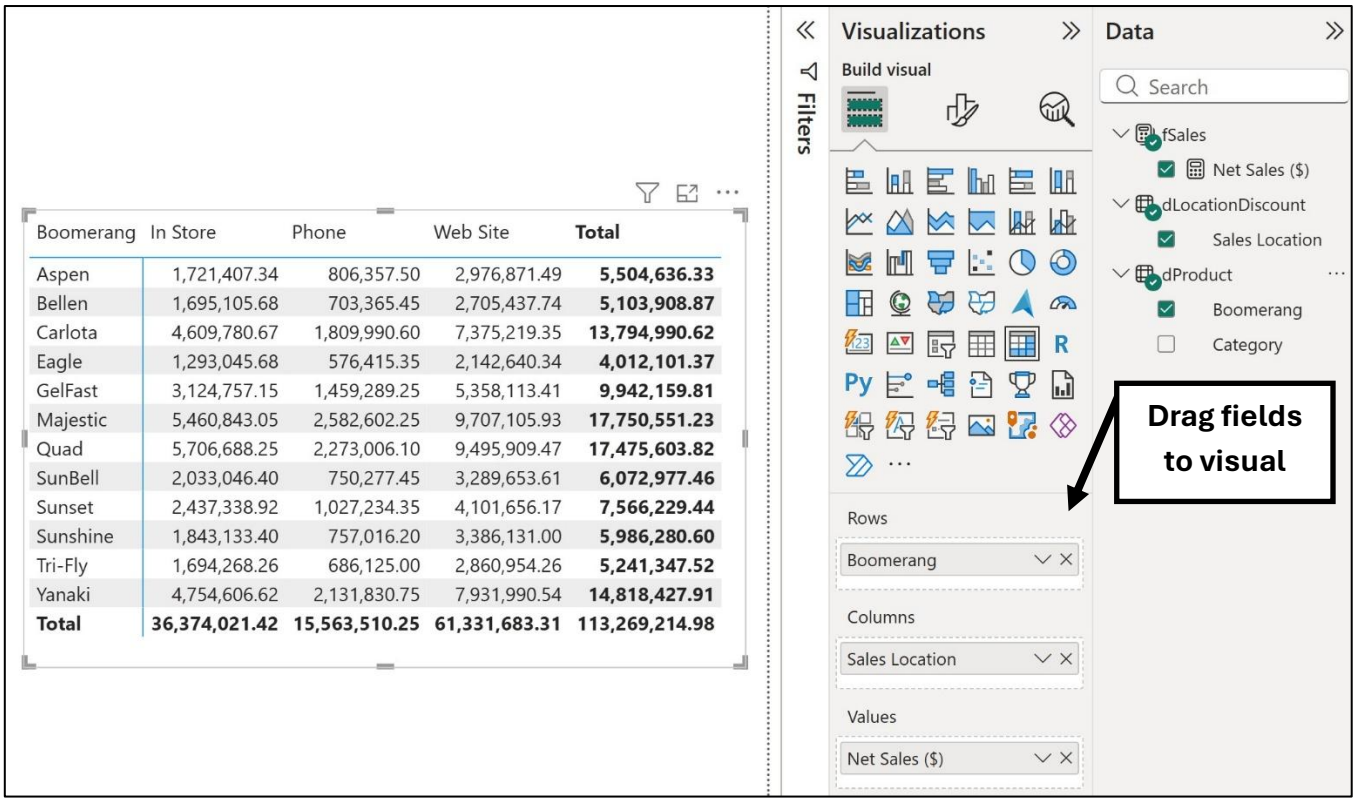


12) Creating Useful Information; Report / Dashboard

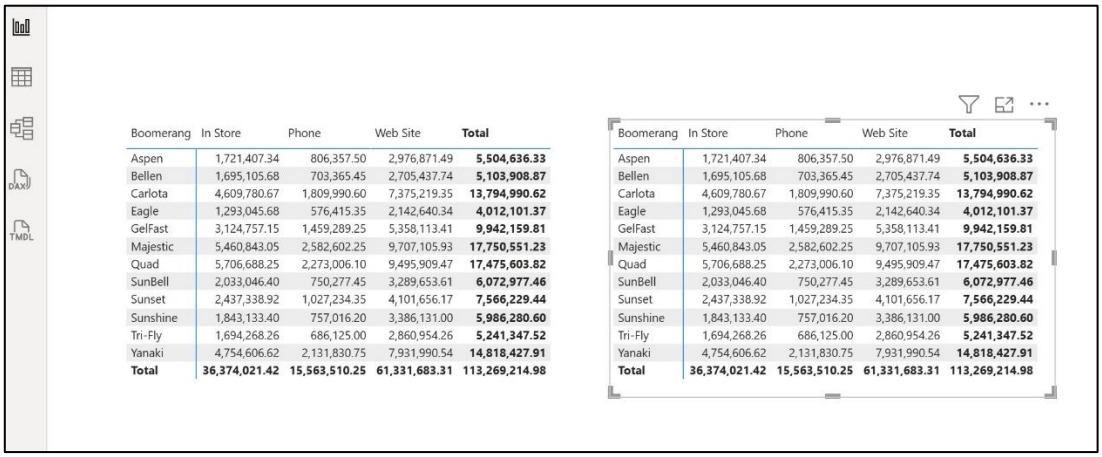
1. As shown below, go to the reporting area and double-click the report tab and name the report / dashboard: “NetSalesDashboard”. In the report area (white canvas), click to activate the report area, and then in the Visualization pane, click the Matrix visual.



2. With the visual selected, as shown below, from the Data pane, drag fields to the visualization pane.



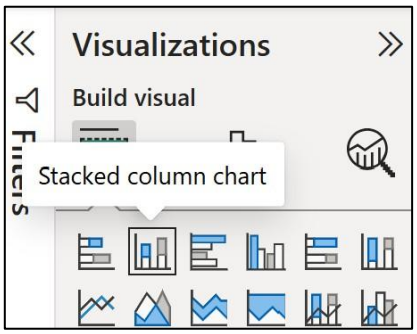
3. Copy and paste the matrix visual. As shown below, move the second matrix to the side.



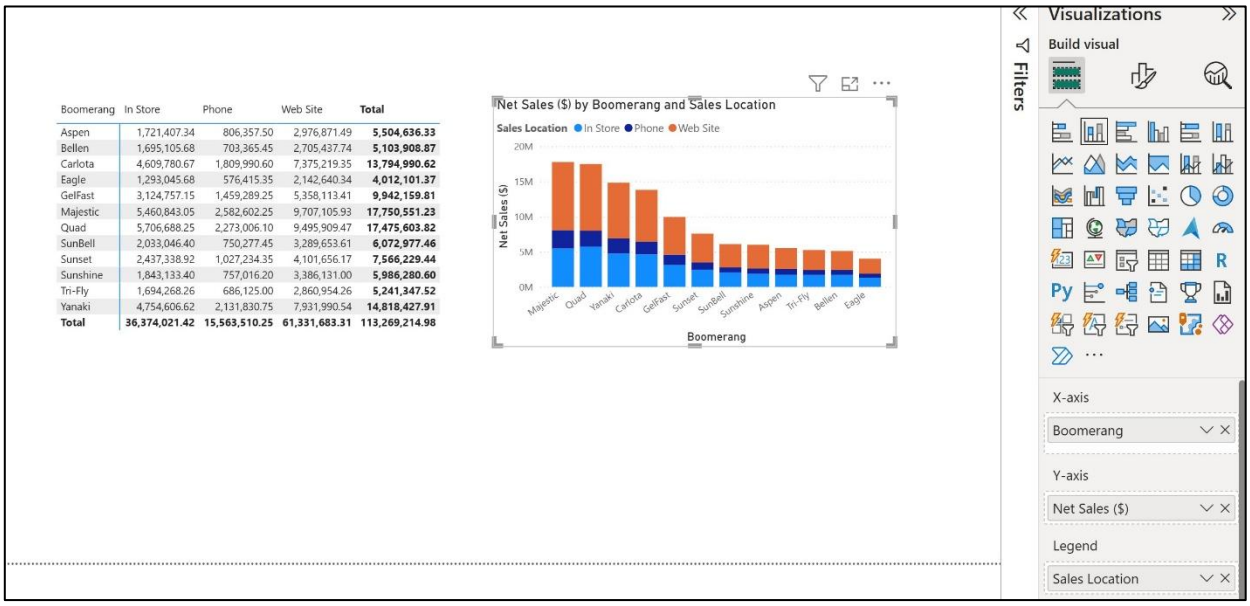
Boomerang	In Store	Phone	Web Site	Total
Aspen	1,721,407.34	806,357.50	2,976,871.49	**5,504,636.33**
Bellen	1,695,105.68	703,365.45	2,705,437.74	**5,103,908.87**
Carlota	4,609,780.67	1,809,990.60	7,375,219.35	**13,794,990.62**
Eagle	1,293,045.68	576,415.35	2,142,640.34	**4,012,101.37**
GelFast	3,124,757.15	1,459,289.25	5,358,113.41	**9,942,159.81**
Majestic	5,460,843.05	2,582,602.25	9,707,105.93	**17,750,551.23**
Quad	5,706,688.25	2,273,006.10	9,495,909.47	**17,475,603.82**
SunBell	2,033,046.40	750,277.45	3,289,653.61	**6,072,977.46**
Sunset	2,437,338.92	1,027,234.35	4,101,656.17	**7,566,229.44**
Sunshine	1,843,133.40	757,016.20	3,386,131.00	**5,986,280.60**
Tri-Fly	1,694,268.26	686,125.00	2,860,954.26	**5,241,347.52**
Yanaki	4,754,606.62	2,131,830.75	7,931,990.54	**14,818,427.91**
Total	**36,374,021.42**	**15,563,510.25**	**61,331,683.31**	**113,269,214.98**

Boomerang	In Store	Phone	Web Site	Total
Aspen	1,721,407.34	806,357.50	2,976,871.49	**5,504,636.33**
Bellen	1,695,105.68	703,365.45	2,705,437.74	**5,103,908.87**
Carlota	4,609,780.67	1,809,990.60	7,375,219.35	**13,794,990.62**
Eagle	1,293,045.68	576,415.35	2,142,640.34	**4,012,101.37**
GelFast	3,124,757.15	1,459,289.25	5,358,113.41	**9,942,159.81**
Majestic	5,460,843.05	2,582,602.25	9,707,105.93	**17,750,551.23**
Quad	5,706,688.25	2,273,006.10	9,495,909.47	**17,475,603.82**
SunBell	2,033,046.40	750,277.45	3,289,653.61	**6,072,977.46**
Sunset	2,437,338.92	1,027,234.35	4,101,656.17	**7,566,229.44**
Sunshine	1,843,133.40	757,016.20	3,386,131.00	**5,986,280.60**
Tri-Fly	1,694,268.26	686,125.00	2,860,954.26	**5,241,347.52**
Yanaki	4,754,606.62	2,131,830.75	7,931,990.54	**14,818,427.91**
Total	**36,374,021.42**	**15,563,510.25**	**61,331,683.31**	**113,269,214.98**

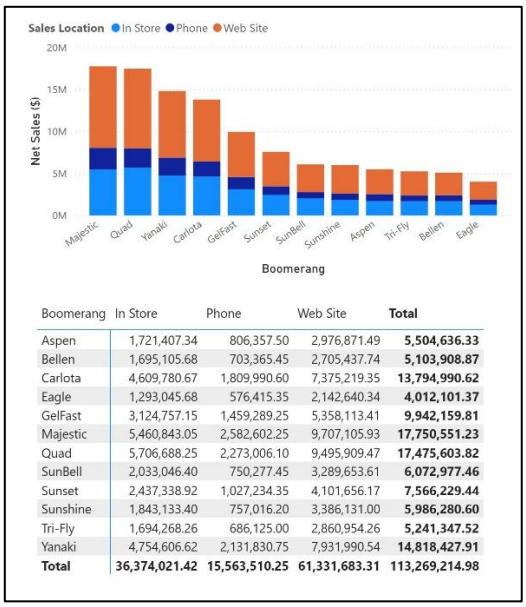
4. With the second matrix selected, in the Visualization pane, as shown below, click the Stacked column chart (this converts the matrix to the chart).



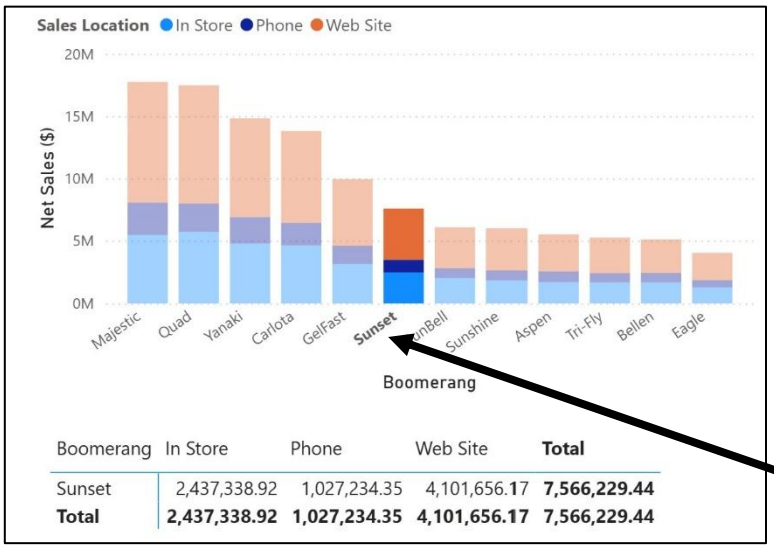
5. The results is shown below.



6. You can move the visuals around, and as shown on the next page, when you select elements in the visuals, the visuals become filtered. This interaction between visuals is a key benefit of using Power BI Desktop.

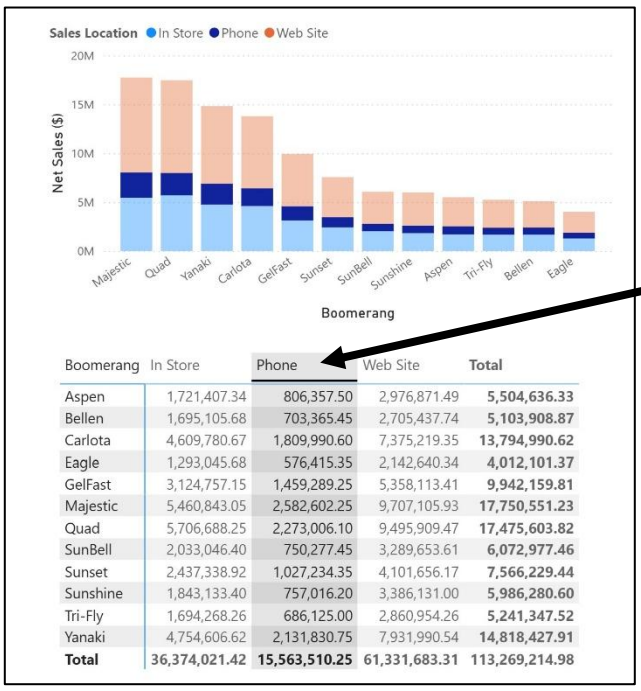


Boomerang	In Store	Phone	Web Site	Total
Aspen	1,721,407.34	806,357.50	2,976,871.49	**5,504,636.33**
Bellen	1,695,105.68	703,365.45	2,705,437.74	**5,103,908.87**
Carlota	4,609,780.67	1,809,990.60	7,375,219.35	**13,794,990.62**
Eagle	1,293,045.68	576,415.35	2,142,640.34	**4,012,101.37**
GelFast	3,124,757.15	1,459,289.25	5,358,113.41	**9,942,159.81**
Majestic	5,460,843.05	2,582,602.25	9,707,105.93	**17,750,551.23**
Quad	5,706,688.25	2,273,006.10	9,495,909.47	**17,475,603.82**
SunBell	2,033,046.40	750,277.45	3,289,653.61	**6,072,977.46**
Sunset	2,437,338.92	1,027,234.35	4,101,656.17	**7,566,229.44**
Sunshine	1,843,133.40	757,016.20	3,386,131.00	**5,986,280.60**
Tri-Fly	1,694,268.26	686,125.00	2,860,954.26	**5,241,347.52**
Yanaki	4,754,606.62	2,131,830.75	7,931,990.54	**14,818,427.91**
Total	**36,374,021.42**	**15,563,510.25**	**61,331,683.31**	**113,269,214.98**



Boomerang	In Store	Phone	Web Site	Total
Sunset	2,437,338.92	1,027,234.35	4,101,656.17	**7,566,229.44**
Total	**2,437,338.92**	**1,027,234.35**	**4,101,656.17**	**7,566,229.44**

In the horizontal axis, select the product name, Sunset



Boomerang	In Store	Phone	Web Site	Total
Aspen	1,721,407.34	806,357.50	2,976,871.49	**5,504,636.33**
Bellen	1,695,105.68	703,365.45	2,705,437.74	**5,103,908.87**
Carlota	4,609,780.67	1,809,990.60	7,375,219.35	**13,794,990.62**
Eagle	1,293,045.68	576,415.35	2,142,640.34	**4,012,101.37**
GelFast	3,124,757.15	1,459,289.25	5,358,113.41	**9,942,159.81**
Majestic	5,460,843.05	2,582,602.25	9,707,105.93	**17,750,551.23**
Quad	5,706,688.25	2,273,006.10	9,495,909.47	**17,475,603.82**
SunBell	2,033,046.40	750,277.45	3,289,653.61	**6,072,977.46**
Sunset	2,437,338.92	1,027,234.35	4,101,656.17	**7,566,229.44**
Sunshine	1,843,133.40	757,016.20	3,386,131.00	**5,986,280.60**
Tri-Fly	1,694,268.26	686,125.00	2,860,954.26	**5,241,347.52**
Yanaki	4,754,606.62	2,131,830.75	7,931,990.54	**14,818,427.91**
Total	**36,374,021.42**	**15,563,510.25**	**61,331,683.31**	**113,269,214.98**

In the column variable, select the sales channel, Phone

SUMX Function

The arguments for the SUMX are as follows:

SUMX(table, expression)

or

SUMX(table, formula that would have been a calculated column)

The SUMX function runs the formula down every row in table to calculate intermediate values and then adds to get a sum total.

One-Step Vs Two-Step Measures

One-Step:

The one-step method involves not creating a Calculated Column in the fact table, but instead you use a function like SUMX with a table in the first argument and a formula in the second argument to create the Measure. The SUMX function runs the formula across each row in the table, calculates the intermediate values and then adds the values. Intermediate values are calculated each time you drop Measure in visual or when you change filters and may slow down Measure calculation time.

Two-Step:

The two-step method involves creating a Calculated Column in the fact table (step one) and then creating a Measure with a function like SUM (second step). Calculated Column values are added to the in-memory RAM Columnar Database and will increase the amount of RAM used.

Which to use?

Unless you have very large amounts of data, you can use either method. The rule that I use is that if the Measure takes a long time to calculate when I drop it into a visual or when I change a filter, then I experiment to see if the two-step method will produce faster results.

Keyboard shortcuts seen in video

1. Open Windows Explorer: **⊞ + E**
2. In the Power Query Editor to rename an identifier, such as a query step or query name use: **F2**