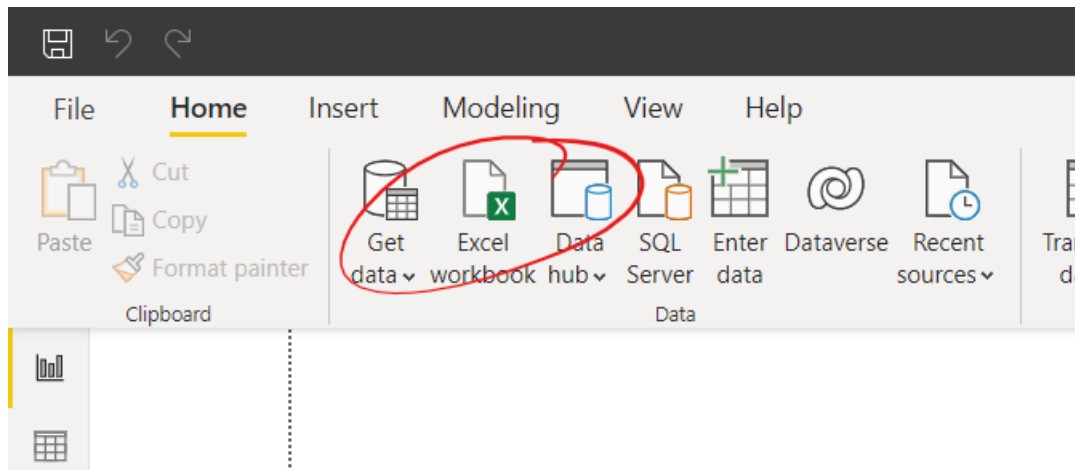


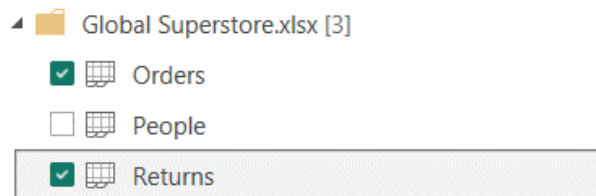
# Advanced Power BI

## 1. Load Data

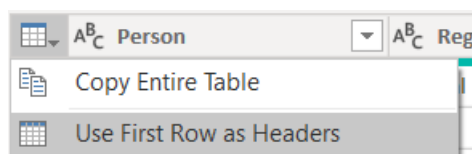
1. Open PowerBI
2. Click on Excel Workbook to load data from Excel



3. Open the file **Global Superstore.xlsx**
4. Check all three tables and click **Transform Data**



5. Select the **Returns** table and select **Use first row as header**



6. Click on **Close & Apply** to load the data into Power BI Desktop

## 2. Relationships


1. Add a table on the first page of your report
2. From the **Returns** table, add **Order ID** and **Returned**

We see that the returns table contains one row for each returned order. However, it doesn't contain any information about the order itself, e.g. the items ordered, or their price, etc, because this information is already stored in the **Orders** table.

To access information from the **Orders** table, we need to tell Power BI that there is a link between these tables.

1. Open the **Model view**



2. Click **Manage Relationships** (  )
3. Click **New**. Select **Orders** as the first table and **Returns** as the second table.
4. Select the **Order ID** columns of both tables.
5. Click **OK**.

We've configured a relationship between the two tables, which means we can use fields from both tables in the same visual.

1. Switch back to the **Report view**



2. Add **Sales** (from **Orders**) to the table you created before.

The table shows now the sales value of each returned order. Note that you if you've added the **Sales** field before configuring the relationship, the table would've shown the total sales of all orders for each row – a useless insight in this context.

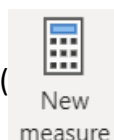
### 3. Measures

Measures are a quick way to create new fields in your data. Measures are defined in a language called DAX and work on aggregated data. They are calculated on demand. That means their values will change, depending on the active filters and any grouping.

Let's illustrate this concept with an example.

1. Select **Orders**.

2. Click on **New measure** (  )

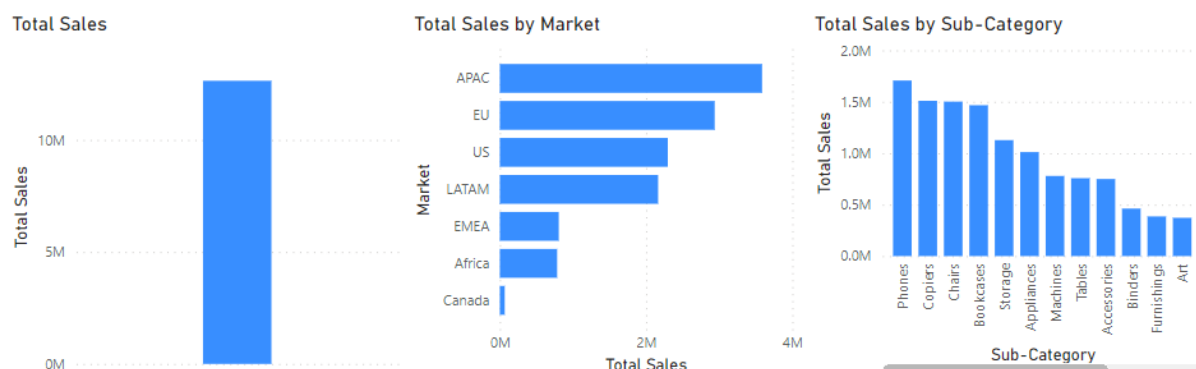


3. Enter this formula:

Total Sales = SUM(Orders[Sales])

Now we create three visuals, all using the same measure but showing different values

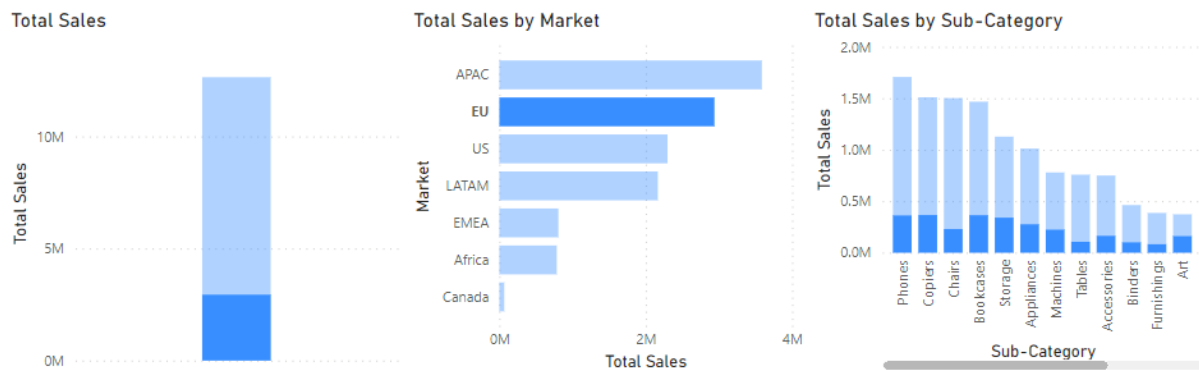
4. Create a **clustered column chart**
5. Add **Total Sales** to **Y-axis**
6. Create a **clustered bar chart**
7. Add **Total Sales** to **X-axis** and **Market** to **Y-axis**
8. Add a **clustered column chart**
9. Add **Total Sales** to **Y-axis** and **Sub-Category** to **X-axis**



Though each visual uses the same measure and, therefore, the same DAX formula, the visuals produce different results. For instance, the first visual shows the Total Sales measure for the entire dataset. In this dataset, Total Sales is USD12.6 million. In the second visual, Total Sales is broken down by market. For instance, in *EU*, Total Sales is USD3.9 million. In the third visual, Total Sales is broken down by Sub-Category.

With Power BI, even though the measure was only defined once, it can be used in these visuals in different ways. Each of the totals is accurate and performs quickly. It is the context of how the DAX measure is used that calculates these totals accurately.

Interactions between visuals will also change how the DAX measure is calculated. For instance, if you select the second visual and then select *EU*, the results appear as shown in the following screenshot.



This scenario is a simple way to explain how context works with DAX. Many other factors affect how DAX formulas are evaluated. Slicers, page filters, and more can affect how a DAX formula is calculated and displayed.


## 4. Running Totals

A running total is the summation of a sequence of numbers, which is updated every time a new number is added. A commonly used running total is sales year-to-date, i.e. the total sales in the current year. DAX makes calculating year-to-date values easy with the DATESYTD function.

1. Create a new measure. Use this formula:

YTD Sales = YTD Sales = `CALCULATE(SUM(Orders[Sales]), DATESYTD(Orders[Order Date]), ALLSELECTED(Orders))`

Add a Matrix (  )

2. Drag **Order Date** to **Rows**. Remove anything except **Month**
3. Drag **Order Date** to **Columns**. Remove anything except **Year**
4. Drag **YTD Sales** to **Values**
5. Click **Format your visual** (  )
6. Turn **Column subtotals** and **Row subtotals** off

Your table should look like this:

Month	2011	2012	2013	2014
January	98,898.49	135,780.72	199,185.91	241,268.56
February	190,050.65	236,290.94	366,425.56	426,105.91
March	335,780.01	399,367.71	565,019.59	689,206.68
April	452,695.78	560,419.98	742,840.90	931,978.55
May	599,443.61	768,784.87	1,003,339.47	1,220,379.59
June	814,650.99	1,024,960.57	1,399,859.08	1,622,193.65
July	930,161.41	1,170,197.35	1,629,788.03	1,880,899.33
August	1,137,742.90	1,473,340.30	1,956,276.82	2,337,519.28
September	1,427,957.36	1,762,729.46	2,332,896.07	2,818,676.52
October	1,627,028.62	2,015,669.31	2,626,302.71	3,241,443.15
November	1,925,525.16	2,339,181.73	3,000,292.07	3,796,722.18
December	2,259,450.90	2,677,438.69	3,405,746.45	4,299,865.87

## 5. Columns

Measures work for most scenarios and are generally preferred, but, as we've seen, they work on aggregated data. Sometimes, data cannot be aggregated, and we need to calculate a value row by row.

If this is the case, we can use **columns**.

1. Right-click the **Orders** table and click **New column**
2. Enter the following formula:
3. `Returned = CONTAINS>Returns, Returns[Order ID], Orders[Order ID])`
4. Add a **Stacked Bar Chart**
5. Add **Sales** to **X-Axis** and **Returned** (from Orders) to Legend.

## 6. Further information about DAX

[Write DAX formulas for Power BI Desktop models - Training | Microsoft Learn](#) provides a good introduction into DAX.

[DAX function reference - DAX | Microsoft Learn](#) is the reference documentation with all available DAX functions.

## 7. Exercise

Re-create the report below. Use the data provided in **Sample Telecom Extract.csv**.

