Writing Advanced DAX Expressions

Lab Time: 60 minutes

Lab Folder: C:\Student\Modules\04_AdvancedDax\Lab

Lab Overview: In this set of lab exercises, you will continue to work on the Power BI Desktop project **Wingtip Sales Model.pbix** that you have been working on over the last few labs. The ending point in the previous lab will be your starting point for this lab. Your work in this lab will involve adding dimensional hierarchies, adding a custom calendar table and writing measures which leverages the built-in DAX support for time intelligence.

Lab Dependencies: This lab assumes you have completed the previous lab titled **Designing a Data Model in Power BI Desktop** in which you worked on a Power BI Desktop project named **Wingtip Sales Model.pbix**. If you would like to begin work on this lab without first completing the previous lab, use the Windows Explorer to copy the lab solution file named **Wingtip Sales Model.pbix** which is located in the student folder at **C:\Student\Modules\03_DataModel\Lab\Solution\** into the folder at **C:\Student\Projects**.

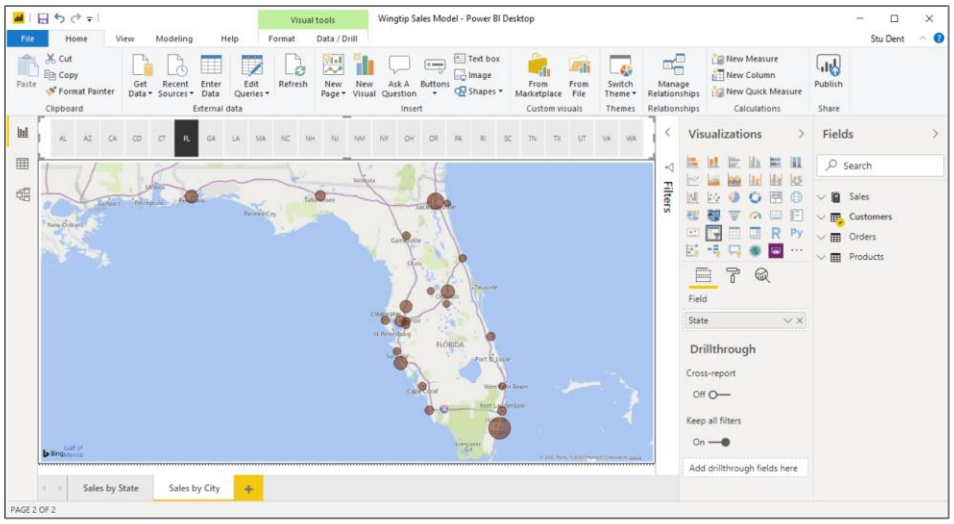
Exercise 1: Extend the Data Model using Dimensional Hierarchies

In this exercise you will modify the **Products** table to add a new dimensional hierarchy named **Product Category** and then you will modify the **Customers** table by adding a new dimensional hierarchy named **Customer Geography**.

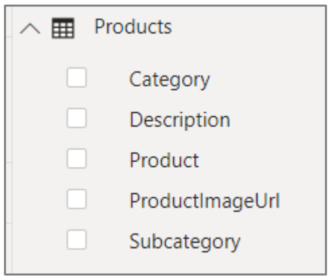
1. Open the **Wingtip Sales Model** project.
   a) Launch Power BI Desktop.
   b) Using the Power BI Desktop **File > Open** command, open **Wingtip Sales Model.pbix** located at the following path.

```
C:\Student\Projects\Wingtip Sales Model.pbix
```

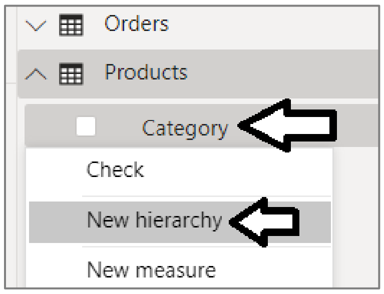
   c) The **Wingtip Sales Model** project should contain two report pages that you created in the previous lab,.



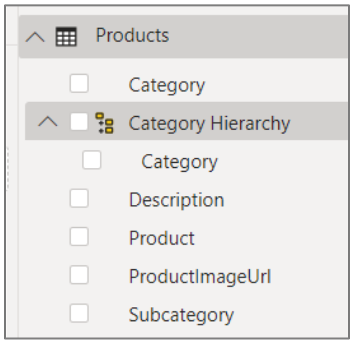
2. Add a new dimensional hierarchy to the **Products** table.
   a) Select the **Products** table in the **Fields** list. It should appear as the one shown in the following screenshot.



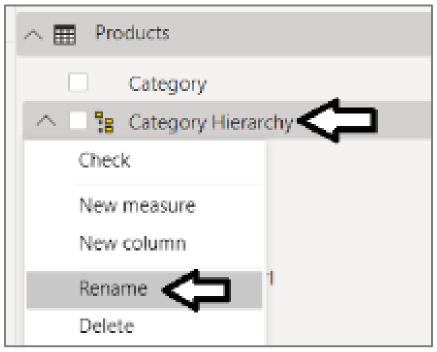
b) Right-click on the **Category** field and then select the **New Hierarchy** menu command.



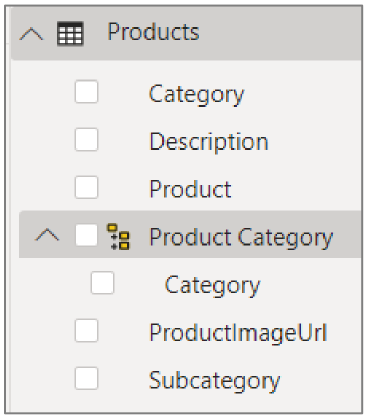
c) You should now see a new dimensional hierarchy in the **Products** table named **Category Hierarchy**.



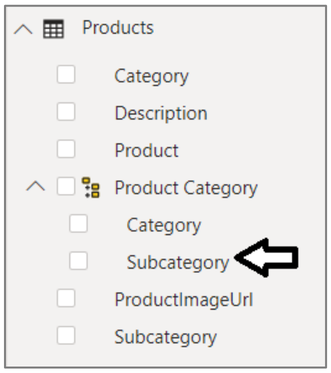
d) Right-click **Category Hierarchy** and select the **Rename** command.



e) Rename the new hierarchy **Product Category**.

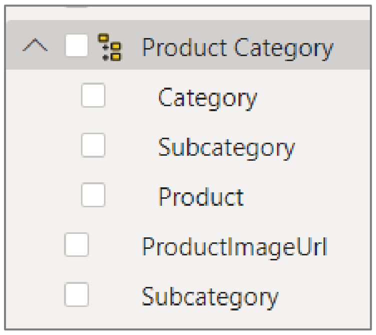


f) Using the mouse, drag-and-drop the **Subcategory** column onto **Product Category** to add this column into the hierarchy.

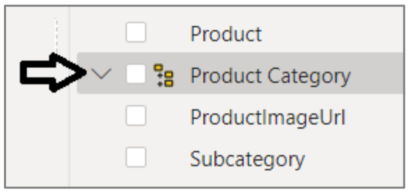


g) Using the mouse, drag-and-drop the **Product** column onto **Product Category** to add this column into the hierarchy.

h) The **Product Category** hierarchy should now contain three fields as shown in the following screenshot.

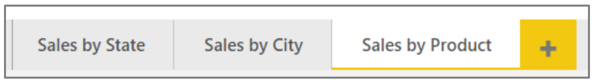


i) Note you can collapse a hierarchy to hide its hierarchy members by clicking the expand/collapse arrow on the left.

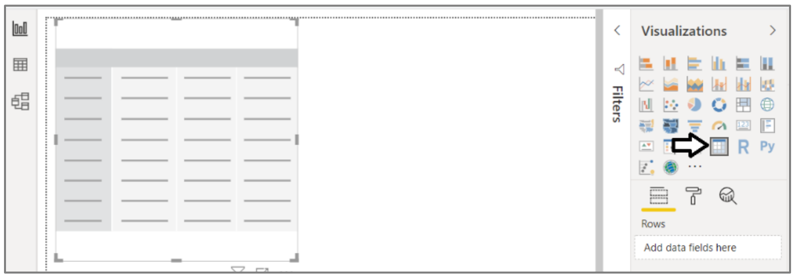


3. Add a new page with a matrix visual to display product sales data.

a) Add new page to the report and rename it to **Sales by Product**.

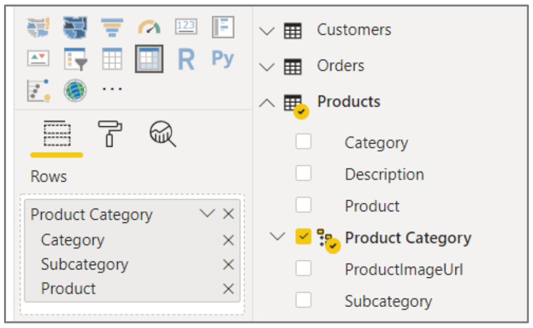


b) Click the **Matrix** icon in the **Visualizations** list to add a new matrix visual to the page.

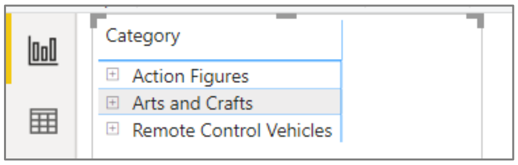


c) With the matrix visual selected, click the **Fields** pane icon.

d) Drag and drop the **Product Category** hierarchy into the **Rows** well for the matrix visual

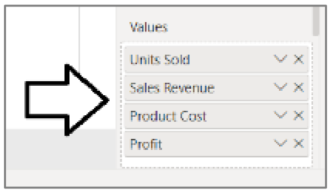


e) The matrix should initially display three rows for the top hierarchy member which is the **Category** column.

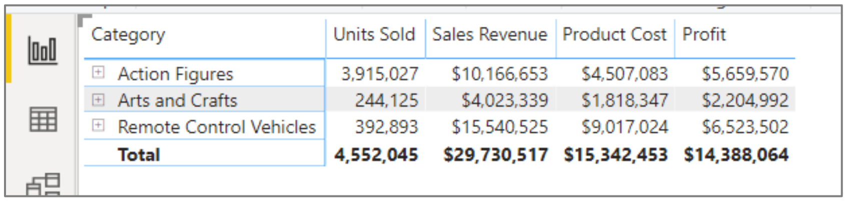


Category
Action Figures
Arts and Crafts
Remote Control Vehicles

f) Populate the **Values** well for the matrix visual by adding the measures **Units Sold**, **Sales Revenue**, **Product Cost** and **Profit**.



g) The matrix visual should display a value per category for each of the four measure as shown in the following screenshot.

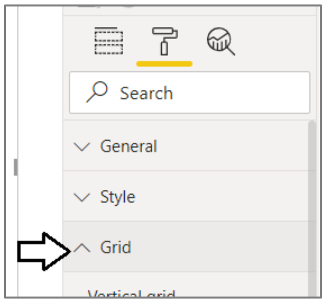


Category	Units Sold	Sales Revenue	Product Cost	Profit
Action Figures	3,915,027	$10,166,653	$4,507,083	$5,659,570
Arts and Crafts	244,125	$4,023,339	$1,818,347	$2,204,992
Remote Control Vehicles	392,893	$15,540,525	$9,017,024	$6,523,502
Total	**4,552,045**	**$29,730,517**	**$15,342,453**	**$14,388,064**

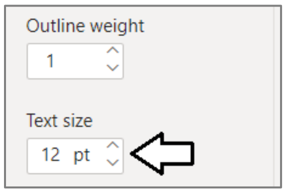
h) Resize the matrix visual to take up to entire width and height of the **Sales by Product** page.

4. Configure the **Matrix** visual to have a larger font size.

a) With the matrix visual selected, expand the **Grid** section in the **Format** pane.

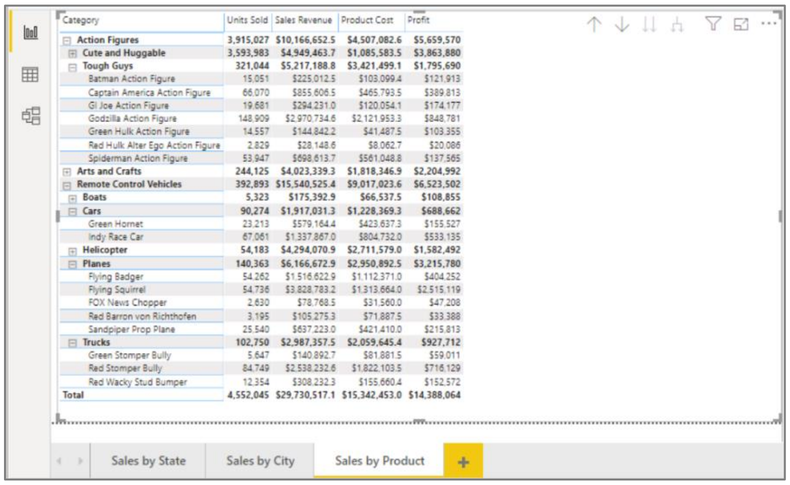


b) Inside the **Grid** section, locate the **Text** site property and increase its value to **12 pt**.



5. Experiment with the pivot-table like behavior of the Matrix visual.

a) Click on the expand/collapse icons to drill into a single subcategory and then into a single product.



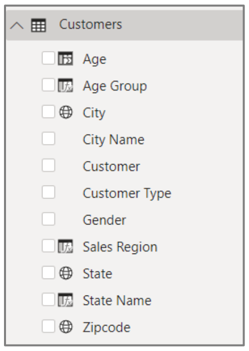
Category	Units Sold	Sales Revenue	Product Cost	Profit
Action Figures	**3,915,027**	**$10,166,652.5**	**$4,507,082.6**	**$5,659,570**
Cute and Huggable	**3,593,983**	**$4,949,463.7**	**$1,085,583.5**	**$3,863,880**
Tough Guys	**321,044**	**$5,217,188.8**	**$3,421,499.1**	**$1,795,690**
Batman Action Figure	15,051	$225,012.5	$103,099.4	$121,913
Captain America Action Figure	66,070	$855,606.5	$465,793.5	$389,813
GI Joe Action Figure	19,681	$294,231.0	$120,054.1	$174,177
Godzilla Action Figure	148,909	$2,970,734.6	$2,121,953.3	$848,781
Green Hulk Action Figure	14,557	$144,842.2	$41,487.5	$103,355
Red Hulk Alter Ego Action Figure	2,829	$28,148.6	$8,062.7	$20,086
Spiderman Action Figure	53,947	$698,613.7	$561,048.8	$137,565
Arts and Crafts	**244,125**	**$4,023,339.3**	**$1,818,346.9**	**$2,204,992**
Remote Control Vehicles	**392,893**	**$15,540,525.4**	**$9,017,023.6**	**$6,523,502**
Boats	**5,323**	**$175,392.9**	**$66,537.5**	**$108,855**
Cars	**90,274**	**$1,917,031.3**	**$1,228,369.3**	**$688,662**
Green Hornet	23,213	$579,164.4	$423,637.3	$155,527
Indy Race Car	67,061	$1,337,867.0	$804,732.0	$533,135
Helicopter	**54,183**	**$4,294,070.9**	**$2,711,579.0**	**$1,582,492**
Planes	**140,363**	**$6,166,672.9**	**$2,950,892.5**	**$3,215,780**
Flying Badger	54,262	$1,516,622.9	$1,112,371.0	$404,252
Flying Squirrel	54,736	$3,828,783.2	$1,313,664.0	$2,515,119
FOX News Chopper	2,630	$78,768.5	$31,560.0	$47,208
Red Barron von Richthofen	3,195	$105,275.3	$71,887.5	$33,388
Sandpiper Prop Plane	25,540	$637,223.0	$421,410.0	$215,813
Trucks	**102,750**	**$2,987,357.5**	**$2,059,645.4**	**$927,712**
Green Stomper Bully	5,647	$140,892.7	$81,881.5	$59,011
Red Stomper Bully	84,749	$2,538,232.6	$1,822,103.5	$716,129
Red Wacky Stud Bumper	12,354	$308,232.3	$155,660.4	$152,572
Total	**4,552,045**	**$29,730,517.1**	**$15,342,453.0**	**$14,388,064**

Now you have learned to build a Pivot-table like experience on top of a dimensional hierarchy. Over the next few steps, you will create a second hierarchy in the **Customers** table named **Customer Geography**.

6. Add a new dimensional hierarchy to the **Customers** table.

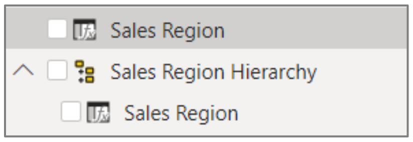
a) Navigate to **Report** view.

b) Inspect the **Customers** table in the **Fields** list. It should appear as the one shown in the following screenshot.

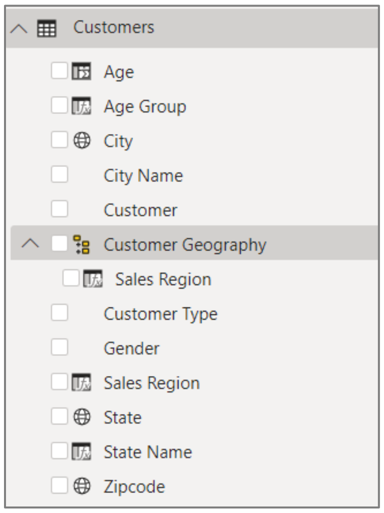


c) Right-click on the **Sales Region** field and then select the **New Hierarchy** menu command.

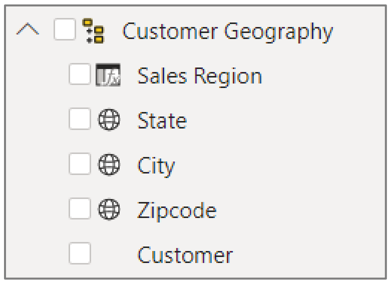
d) You should now see a new dimensional hierarchy in the fields list named **Sales Region Hierarchy**.



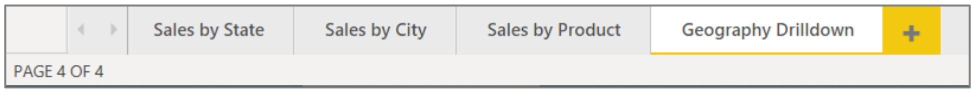
e) Right-click **Sales Region Hierarchy,** select the **Rename** menu command and rename the hierarchy **Customer Geography**.


Customers
Age
Age Group
City
City Name
Customer
Customer Geography
Sales Region
Customer Type
Gender
Sales Region
State
State Name
Zipcode

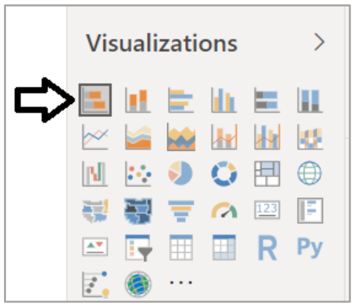
f) Using the mouse, drag-and-drop the **State** column onto **Customer Geography** to add this column into the hierarchy.
g) Using the mouse, drag-and-drop the **City** column onto **Customer Geography** to add this column into the hierarchy.
h) Using the mouse, drag-and-drop the **Zipcode** column onto **Customer Geography** to add this column into the hierarchy.
i) Using the mouse, drag-and-drop the **Customer** column onto **Customer Geography** to add this column into the hierarchy.
j) The **Customer Geography** hierarchy should now contain four fields as shown in the following screenshot.


Customer Geography
Sales Region
State
City
Zipcode
Customer

7. Create a new report page and rename it to **Geography Drilldown**.


Sales by State
Sales by City
Sales by Product
Geography Drilldown
PAGE 4 OF 4

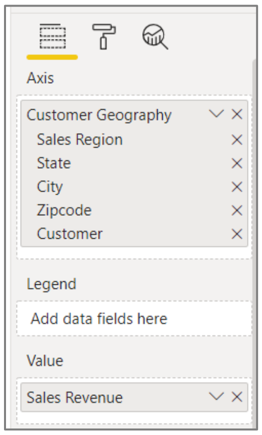
8. Add a new stacked bar chart visual.
a) Click the **Stacked Bar Chart** button on the **Visualizations** list to create a new bar chart visual.



b) Resize the bar chart visual so it takes up the entire height and width of the report page.

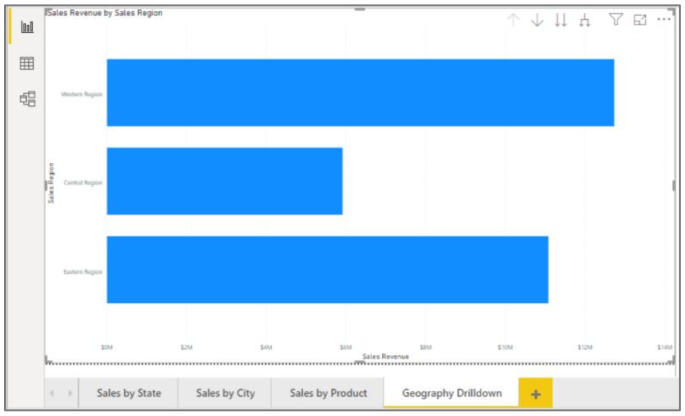
c) Drag and drop the **Customer Geography** hierarchy from the **Customers** table into the **Axis** well.

d) Drag and drop the **Sales Revenue** field from **Sales** table into the **Value** well.



Axis
Customer Geography
Sales Region
State
City
Zipcode
Customer
Legend
Add data fields here
Value
Sales Revenue

e) The bar chart should now appear like the one shown in the following screenshot.



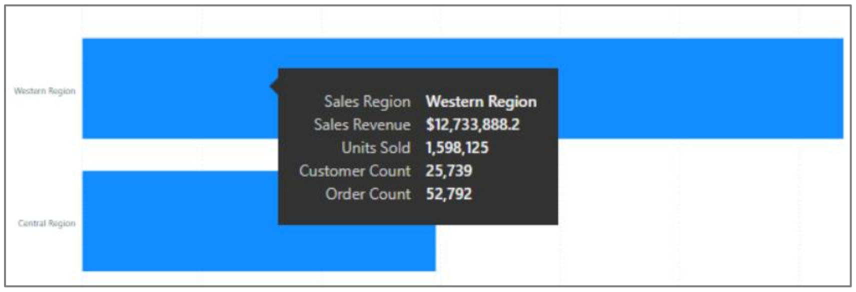
9. Configure **Tooltips** for the bar chart to additionally display **Units Sold** and **Customer Count**.

a) Drag and drop the **Units Sold** field from the **Sales** table into the **Tooltips** well.

b) Drag and drop the **Customer Count** field from the **Sales** table into the **Tooltips** well.

c) Drag and drop the **Order Count** field from the **Sales** table into the **Tooltips** well.

d) Hover over a bar in the bar chart with the mouse to observe the effects of the new tooltip configuration.



Sales Region Western Region
Sales Revenue $12,733,888.2
Units Sold 1,598,125
Customer Count 25,739
Order Count 52,792

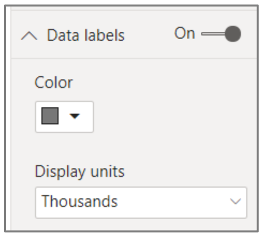
10. Configure **Data labels** for bars inside the bar chart.

a) Make sure the bar chart visual is selected.

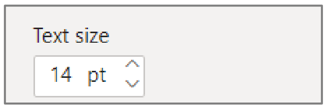
b) Navigate to the **Format** properties pane and expand the **Data labels** section.

c) Set the primary **Data labels** setting from **Off** to **On**.

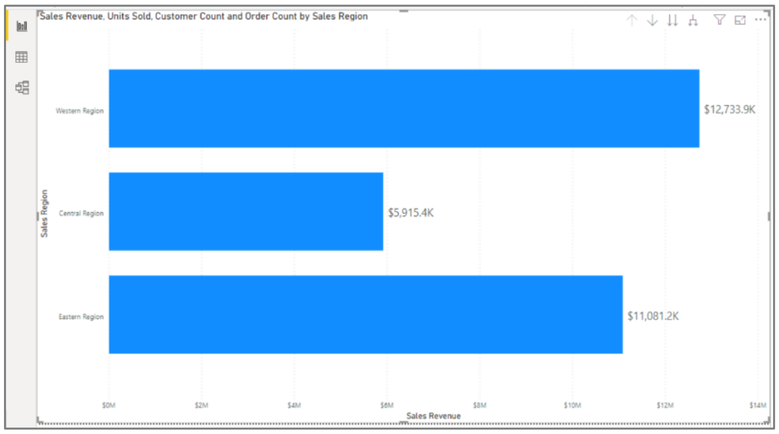
d) Update the **Display Units** property to **Thousands**.


Data labels On
Color
Display units
Thousands

e) Set the Text Size to **14 pt**.

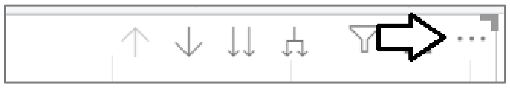

Text size
14 pt

f) The bar chart should now display a data label for each bar showing total sales revenue.

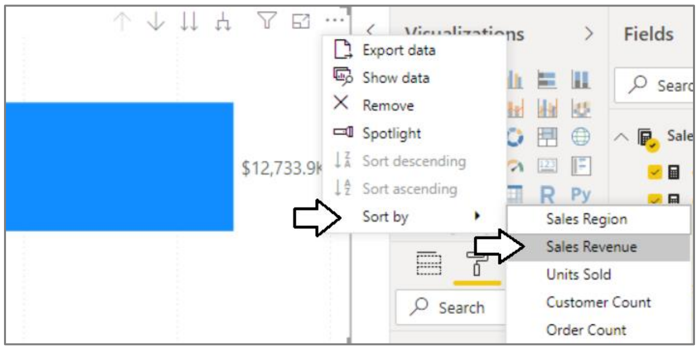


11. Modify the sorting to sorting of the bars in the bar chart to sort by **Sales Revenue** in a descending fashion.

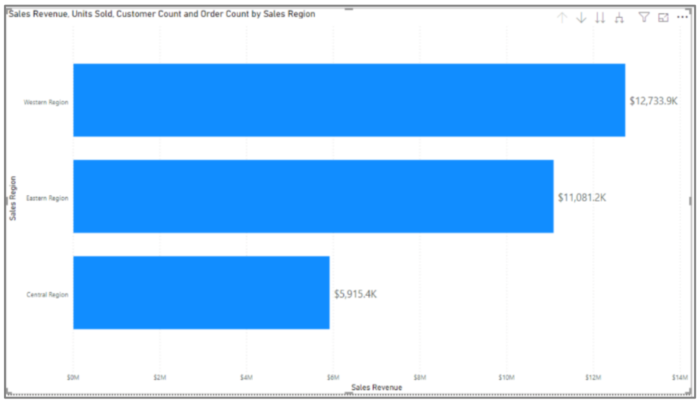
a) Click the ellipse for the visual flyout menu at the top right of the bar chart visual.



b) Select the **Sort by > Sales Revenue** command.

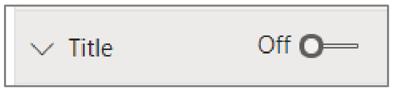


c) The bars in the bar chart should now be sorted with the bars containing the largest sales revenue values at the top.



12. Remove the visual title from the bar chart.

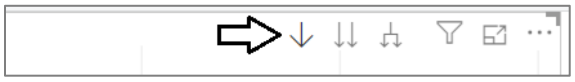
a) With the bar chart selected, go to the **Format** pane and set the **Title** property to **Off**.



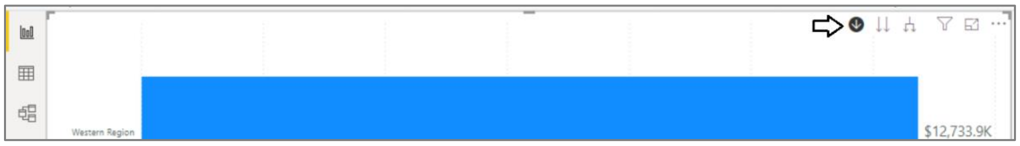
13. Turn on drill down mode for the bar chart visual.

a) Locate the **Drill Down** button with the downward pointing arrow and the circle around it in the top right corner of the visual.

b) Click on the **Drill Down** button once to enable drill down mode.

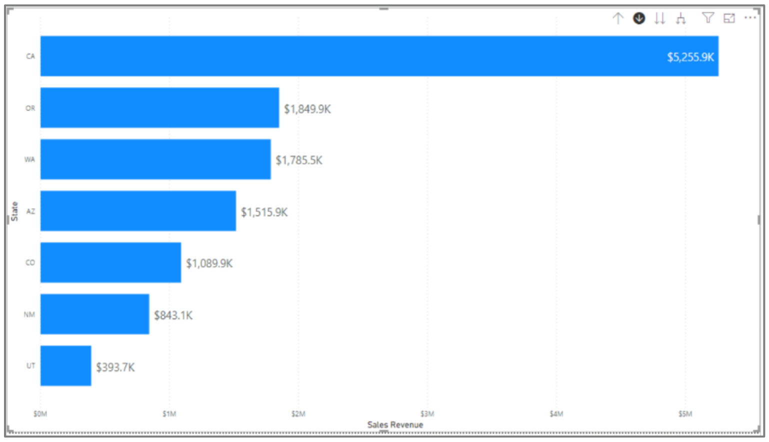


c) Once you have enabled drill down mode, the **Drill Down** button appears as a dark circle with a white downward-facing arrow.

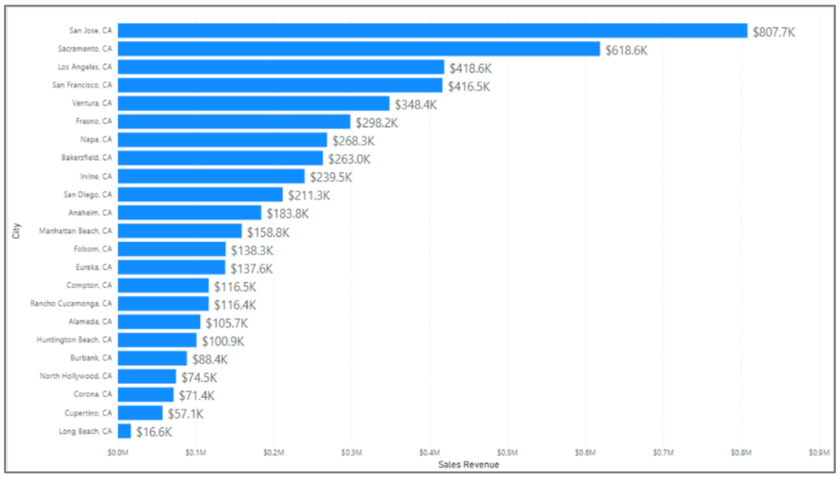


14. Experiment with drill down mode by drilling into the **Customer Geography** hierarchy.

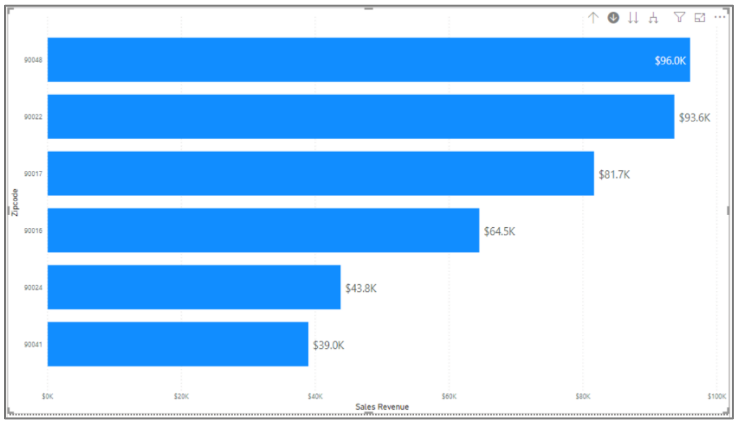
a) Click on the bar for the **Western Region** to drill down into the states for that sales region.



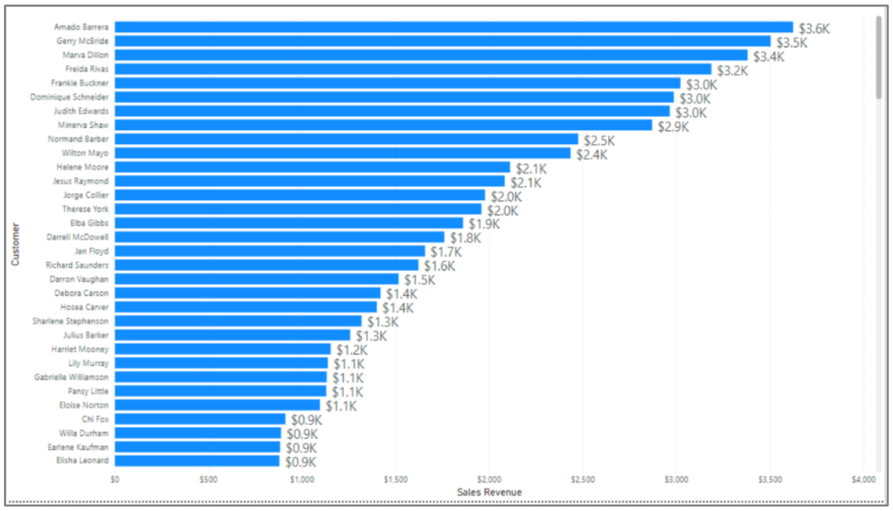
b) Click on the bar for the **CA** to drill down into the cities in California.



c) Click on the bar for a city such as **Los Angeles, CA** to drill down into the zipcodes for that city.



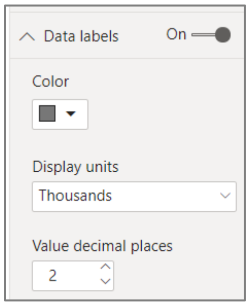
d) Click on the bar for a zipcode to drill down into the top customers in that area.



Now that you can navigate down to see sales revenue at the customer level, you need to increase the precision of the data labels.

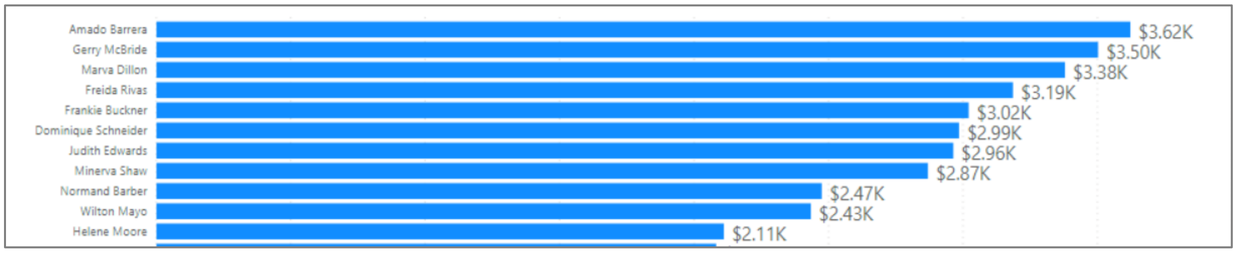
e) With the bar chart selected, expand the **Data labels** section in the **Format** pane.

f) Set the **Value decimal places** property value to **2**.

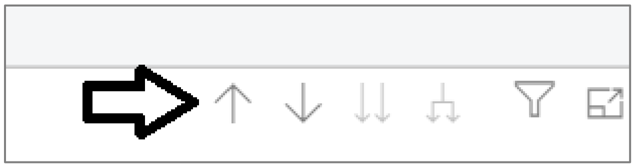


The up and down button arrows for **Value decimal places** does not work if the value is blank. You might have to type in the value of 2.

g) The data labels with customer sales revenue should now show greater precision.



h) Click the **Drill up** icon 4 times to move back up to the top of the hierarchy where the bars are showing sales segions..



You have now seen how you can set up a visual to drill into and out of a dimensional hierarchy.

Exercise 2: Extend the Data Model with a Custom Calendar Table

In this exercise you will create a calculated table named **Calendar** which will play the role of a time dimension table in the data model. The motivation for adding a time dimension table to your data model is that it will allow you to take maximum advantage of the time intelligence functions that are included in the DAX expression language.

1. Create a new calculated table named **Calendar**.

a) Navigate to the **Modeling** tab in the ribbon.

b) Click the **New Table** button in the ribbon.

Instead of having you write one heck-of-a-long DAX expression, the lab exercise will have you copy and paste the required DAX expression from a text file in the **Students** folder.

c) Locate the file named **CreateCalendarTable.txt** at the following path and open it with Windows Notepad.

```
C:\Student\Modules\04_AdvancedDax\Lab\DAX\CreateCalendarTable.txt
```

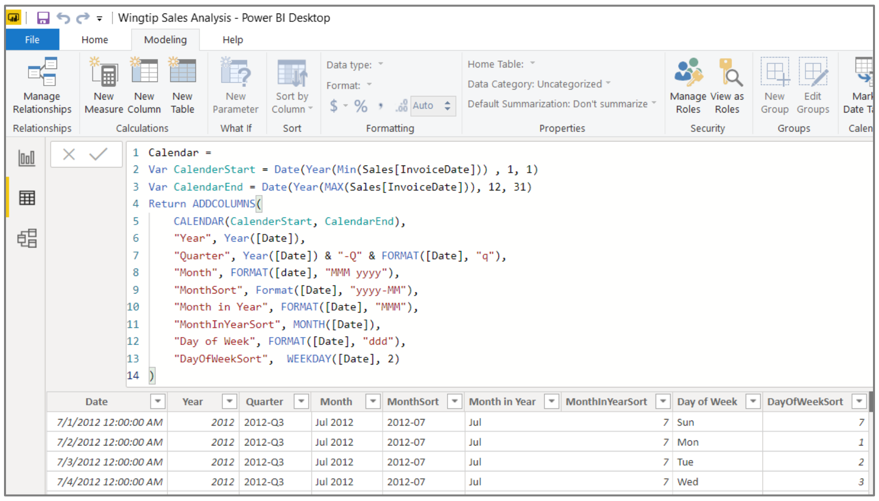
d) Select the entire contents of **CreateCalendarTable.txt** and then copy it into the Windows clipboard.

e) Return to Power BI Desktop and paste in the DAX expression for the new table.

f) Press the **ENTER** key to create the new table named **Calendar**.

```
Calendar =
Var CalenderStart = Date(Year(Min(Sales[InvoiceDate])) , 1, 1)
Var CalendarEnd = Date(Year(MAX(Sales[InvoiceDate])), 12, 31)
Return ADDCOLUMNS(
    CALENDAR(CalenderStart, CalendarEnd),
    "Year", Year([Date]),
    "Quarter", Year([Date]) & "-Q" & FORMAT([Date], "q"),
    "Month", FORMAT([date], "MMM yyyy"),
    "MonthSort", Format([Date], "yyyy-MM"),
    "Month in Year", FORMAT([Date], "MMM"),
    "MonthInYearSort", MONTH([Date]),
    "Day of Week", FORMAT([Date], "ddd"),
    "DayOfWeekSort",  WEEKDAY([Date], 2)
)
```

g) You should be able to verify that the **Calendar** table has been created with several columns including a **Date** column.

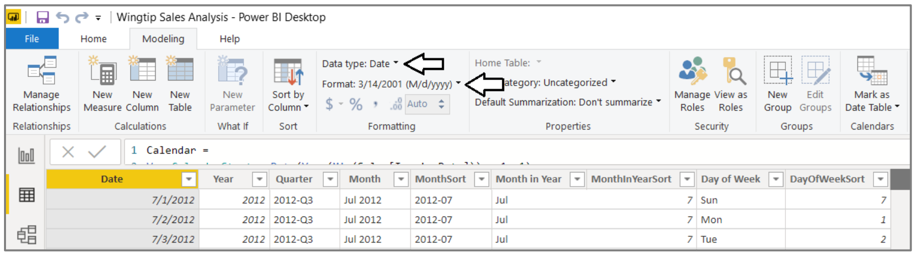


```
Calendar =
Var CalenderStart = Date(Year(Min(Sales[InvoiceDate])) , 1, 1)
Var CalendarEnd = Date(Year(MAX(Sales[InvoiceDate])), 12, 31)
Return ADDCOLUMNS(
    CALENDAR(CalenderStart, CalendarEnd),
    "Year", Year([Date]),
    "Quarter", Year([Date]) & "-Q" & FORMAT([Date], "q"),
    "Month", FORMAT([date], "MMM yyyy"),
    "MonthSort", Format([Date], "yyyy-MM"),
    "Month in Year", FORMAT([Date], "MMM"),
    "MonthInYearSort", MONTH([Date]),
    "Day of Week", FORMAT([Date], "ddd"),
    "DayOfWeekSort",  WEEKDAY([Date], 2)
)
```

Date	Year	Quarter	Month	MonthSort	Month in Year	MonthInYearSort	Day of Week	DayOfWeekSort
7/1/2012 12:00:00 AM	2012	2012-Q3	Jul 2012	2012-07	Jul	7	Sun	7
7/2/2012 12:00:00 AM	2012	2012-Q3	Jul 2012	2012-07	Jul	7	Mon	1
7/3/2012 12:00:00 AM	2012	2012-Q3	Jul 2012	2012-07	Jul	7	Tue	2
7/4/2012 12:00:00 AM	2012	2012-Q3	Jul 2012	2012-07	Jul	7	Wed	3

Note that this DAX expressions to calculate the start date and end date dynamically provides flexibility. This DAX expression for the **Calendar** table automatically add new rows for complete years for **Sales** based on the minimum and maximum date values in the **InvoiceDate** column of the **Sales** table.

2. Change the type and format of the **Date** column.
   a) Select the **Date** column by clicking its column header.
   b) Activate the **Modeling** tab of the ribbon.
   c) Set the **Data Type** property to **Date**.
   d) Set the **Format** property to **03/14/2001 (MM/dd/yyyy)**.



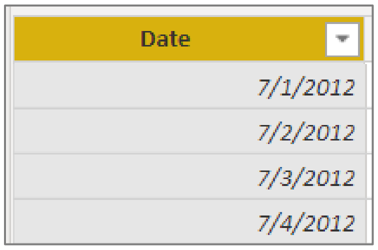
Data type: Date
Format: 3/14/2001 (M/d/yyyy)

```
Calendar =
```

Date	Year	Quarter	Month	MonthSort	Month in Year	MonthInYearSort	Day of Week	DayOfWeekSort
7/1/2012	2012	2012-Q3	Jul 2012	2012-07	Jul	7	Sun	7
7/2/2012	2012	2012-Q3	Jul 2012	2012-07	Jul	7	Mon	1
7/3/2012	2012	2012-Q3	Jul 2012	2012-07	Jul	7	Tue	2

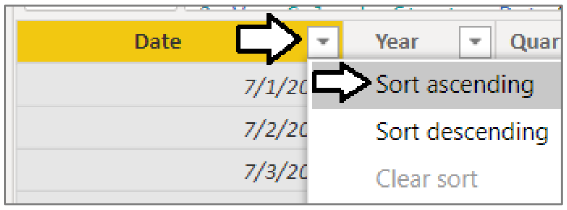
When the tables of a Power BI data model are loaded into memory, they are laid out in columns of data as opposed to rows of data. Therefore, a table in a data model table doesn't really have a pre-defined sort order. Because of this, it sometimes appears a table isn't sorted properly when you examine it in **Data** view. Such is the case when you examine the **Date** column of the **Calendar** table.

3. Sort the rows in the **Calendar** table chronologically.
   a) If you examine the **Date** column, the first date shown is **7/1/2012** instead of the first actual date which is **1/1/2012**.

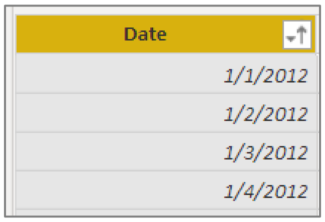


Date
7/1/2012
7/2/2012
7/3/2012
7/4/2012

   b) Click the downward arrow menu on right side of the **Date** column header and select the **Sort ascending** command.



   c) Now you can see the real start date of the **Calendar** table which is **1/1/2012**.



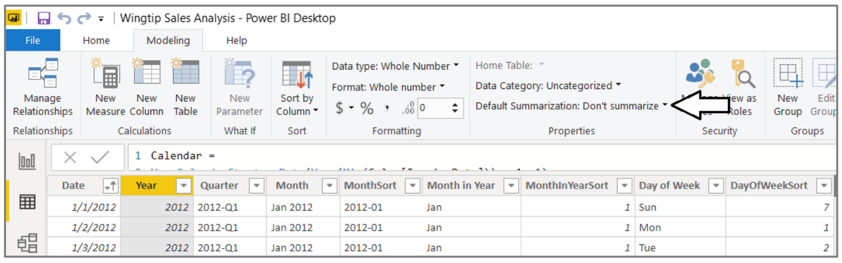
Date
1/1/2012
1/2/2012
1/3/2012
1/4/2012

You should also be able to verify that there's one row in the **Calendar** table for each day in 2012, 2013, 2014 and 2015.

4. Modify the **Default Summarization** property of the **Year** column.

While the **Year** column contains numeric values, it doesn't make sense to perform standard aggregations on this column such as **Sum**, **Average** or **Count**. Setting the column's **Default Summarization** to **Do Not Summarize** will prevent Power BI Desktop from automatically assigning aggregate operations when this field is added to a visual in a report.

   a) Select the **Year** column and sets its **Default Summarization** property to **Don't Summarize**.



Data type: Whole Number
Format: Whole number
Home Table:
Data Category: Uncategorized
Default Summarization: Don't summarize

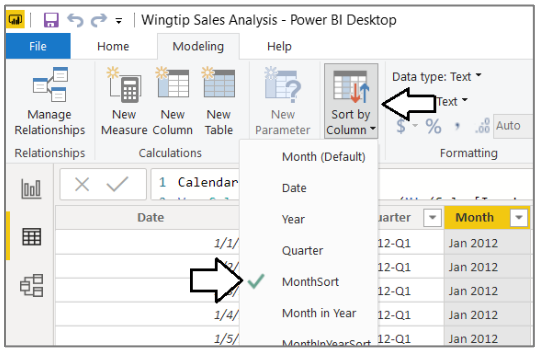
```
1 Calendar =
```

Date	Year	Quarter	Month	MonthSort	Month in Year	MonthInYearSort	Day of Week	DayOfWeekSort
1/1/2012	2012	2012-Q1	Jan 2012	2012-01	Jan	1	Sun	7
1/2/2012	2012	2012-Q1	Jan 2012	2012-01	Jan	1	Mon	1
1/3/2012	2012	2012-Q1	Jan 2012	2012-01	Jan	1	Tue	2

5. Configure the **Month** column to be sorted chronologically.

The **Month** column is a good example of a column whose values will not automatically be sorted in a chronological sort order. The default sort order of a text column like **Month** is to sort month names alphabetically so that April will sort before February, and February will sort before January. Therefore, you will now configure the **Month** column to be sorted by values in **MonthSort** to create a chronological sort order.

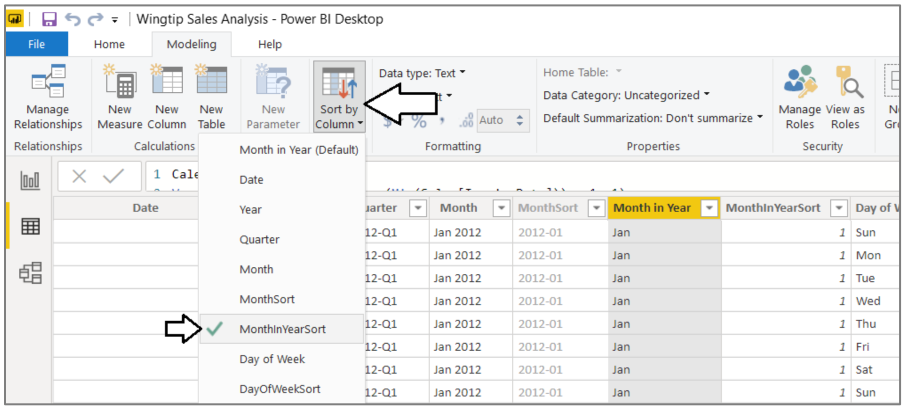
a) Select the **Month** column and then set its **Sort By Column** property to **MonthSort** column.



b) Hide the **MonthSort** column by right-clicking it on the **Fields** list and selecting the **Hide in Report View** menu command.

6. Configure the **Month in Year** column to be sorted chronologically.

a) Select the **Month in Year** column and then set its **Sort By Column** property to **MonthinYearSort** column.

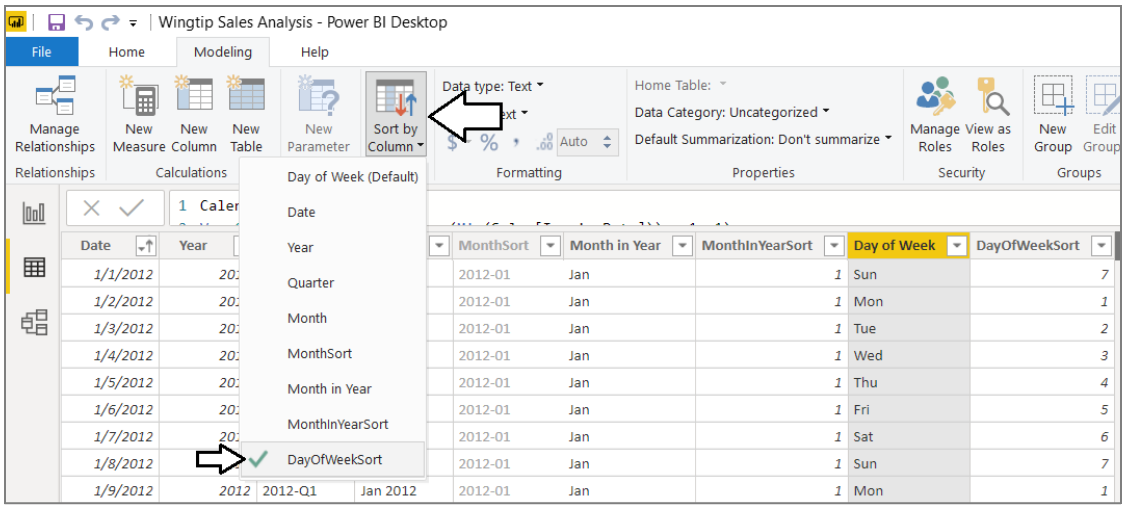


Month	MonthSort	Month in Year	MonthInYearSort	Day of W
Jan 2012	2012-01	Jan	1	Sun
Jan 2012	2012-01	Jan	1	Mon
Jan 2012	2012-01	Jan	1	Tue
Jan 2012	2012-01	Jan	1	Wed
Jan 2012	2012-01	Jan	1	Thu
Jan 2012	2012-01	Jan	1	Fri
Jan 2012	2012-01	Jan	1	Sat
Jan 2012	2012-01	Jan	1	Sun

b) Hide the **MonthInYearSort** column by right-clicking it on the **Fields** list and selecting **Hide in Report View**.

7. Configure the **Day of Week** column to be sorted chronologically.

a) Select the **Day of Week** column and then set its **Sort By Column** property to **DayOfWeekSort** column.

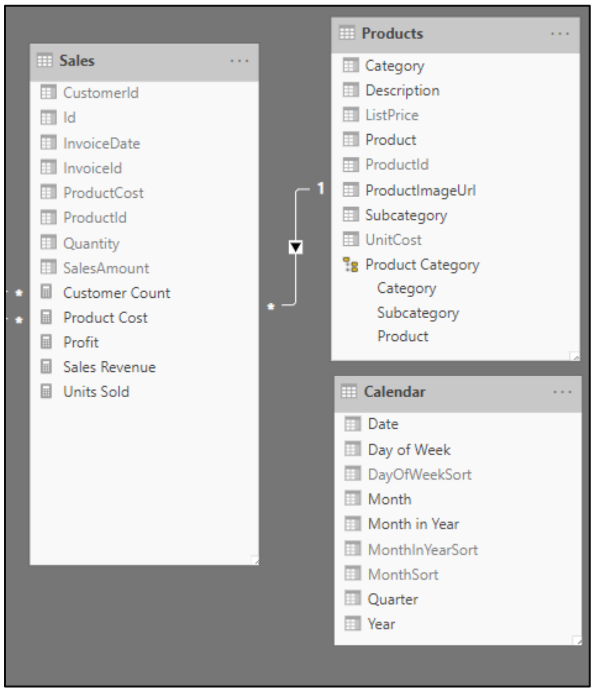


Date	Year	MonthSort	Month in Year	MonthInYearSort	Day of Week	DayOfWeekSort
1/1/2012	20	2012-01	Jan	1	Sun	7
1/2/2012	20	2012-01	Jan	1	Mon	1
1/3/2012	20	2012-01	Jan	1	Tue	2
1/4/2012	20	2012-01	Jan	1	Wed	3
1/5/2012	20	2012-01	Jan	1	Thu	4
1/6/2012	20	2012-01	Jan	1	Fri	5
1/7/2012	20	2012-01	Jan	1	Sat	6
1/8/2012	20	2012-01	Jan	1	Sun	7
1/9/2012	2012	2012-01	Jan	1	Mon	1

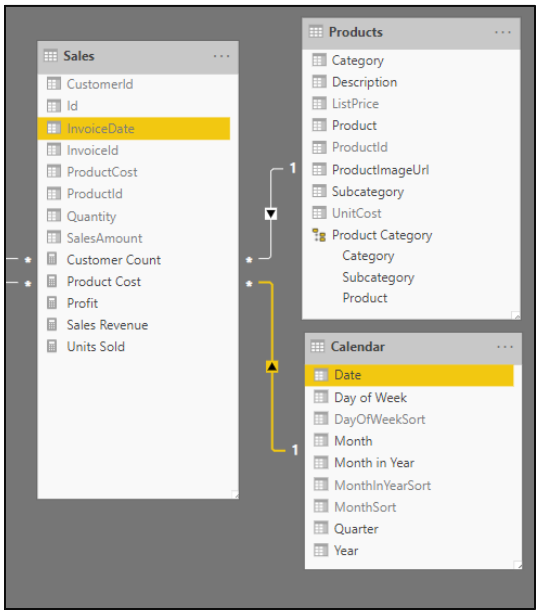
b) Hide the **DayOfWeekSort** column by right-clicking it on the **Fields** list and selecting **Hide in Report View**.

At this point, you have created the **Calendar** table and added all the columns that it requires. The next step to integrate the **Calendar** table into the data model will be to create a relationship between the **Calendar** table and the **Sales** table.

8. Create a relationship between the **Calendar** table and the **Sales** table.
   a) Navigate to **Model** view. You should be able to see that the **Calendar** table is present.
   b) Using the mouse, rearrange the tables in **Model** view to match the following screenshot where the **Calendar** table is positioned directly below the **Products** table and to the immediate right of the **Sales** table.

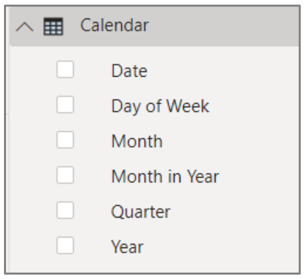


   c) Create a new table relationship by performing a drag and drop operation with the mouse to drag the **InvoiceDate** column from the **Sales** table and to drop it on the **Date** column of the **Calendar** table.

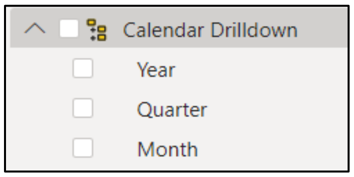


You have now completed the work of adding the **Calendar** table to the data model. Now, you will modify the **Calendar** table to add a new dimension hierarchy named **Calendar Drilldown**.

9. Add a new dimensional hierarchy to the **Calendar** table.
   a) Use the left navigation to switch to **Report** View.
   b) Inspect the **Calendar** table in the **Fields** list. It should appear as the one shown in the following screenshot.



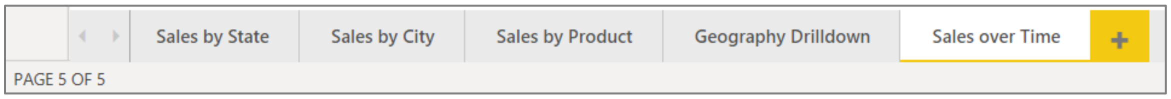
   c) Right-click on the **Year** field and then select the **New Hierarchy** menu command.
   d) Right-click **Year Hierarchy,** select the **Rename** command and rename the hierarchy to **Calendar Drilldown**.
   e) Using the mouse, drag-and-drop the **Quarter** column onto **Calendar Drilldown** to add this column into the hierarchy.
   f) Using the mouse, drag-and-drop the **Month** column onto **Calendar Drilldown** to add this column into the hierarchy.
   g) The **Calendar Drilldown** hierarchy should now contain three hierarchy members as shown in the following screenshot.



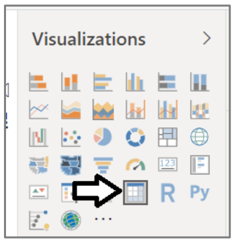
10. Save the work you have done by clicking the **Save** button in the upper left corner of the Power BI Desktop window.

Over the next few steps, you will leverage the **Calendar** table by using it to create a few new visuals to display sales revenue totals aggregated over various time periods.

11. Create a new report page named **Sales over Time**.
   a) Navigate to **Report** view.
   b) Add a new page by clicking the (**+**) button on the right side of the page navigation menu.
   c) Once the new page has been created, modify its title to **Sales over Time**.



12. Create a new matrix visual to show sales revenue for specific time periods.
   a) Click that **Matrix** icon in the **Visualizations** list to add a new matrix visual to the page.

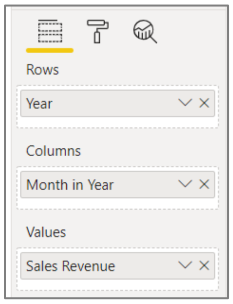


b) With the matrix visual selected, navigate to the **Fields** pane.

c) Add the **Year** column from the **Calendar** table into the **Rows** well.

d) Add the **Month in Year** column from the **Calendar** table into the **Columns** well.

e) Add the **Sales Revenue** measure from the **Sales** table into the **Values** well.



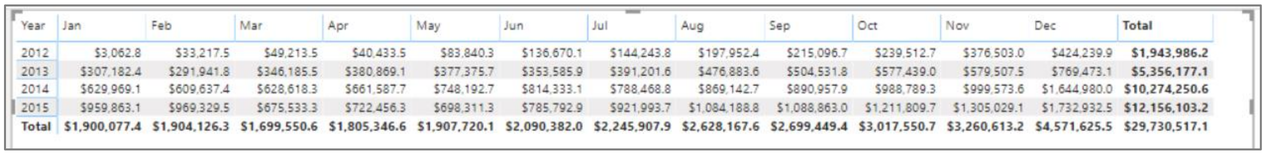
Rows
Year

Columns
Month in Year

Values
Sales Revenue

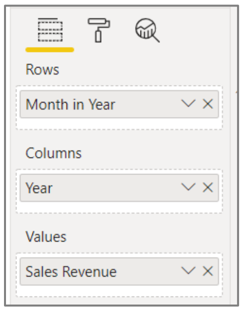
f) The matrix now has a column for each month and a row for each year.

g) Use your mouse to resize the matrix visual to take up the entire width of the page.



Year	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Total
2012	$3,062.8	$33,217.5	$49,213.5	$40,433.5	$83,840.3	$136,670.1	$144,243.8	$197,952.4	$215,096.7	$239,512.7	$376,503.0	$424,239.9	**$1,943,986.2**
2013	$307,182.4	$291,941.8	$346,185.5	$380,869.1	$377,375.7	$353,585.9	$391,201.6	$476,883.6	$504,531.8	$577,439.0	$579,507.5	$769,473.1	**$5,356,177.1**
2014	$629,969.1	$609,637.4	$628,618.3	$661,587.7	$748,192.7	$814,333.1	$788,468.8	$869,142.7	$890,957.9	$988,789.3	$999,573.6	$1,644,980.0	**$10,274,250.6**
2015	$959,863.1	$969,329.5	$675,533.3	$722,456.3	$698,311.3	$785,792.9	$921,993.7	$1,084,188.8	$1,088,863.0	$1,211,809.7	$1,305,029.1	$1,732,932.5	**$12,156,103.2**
Total	**$1,900,077.4**	**$1,904,126.3**	**$1,699,550.6**	**$1,805,346.6**	**$1,907,720.1**	**$2,090,382.0**	**$2,245,907.9**	**$2,628,167.6**	**$2,699,449.4**	**$3,017,550.7**	**$3,260,613.2**	**$4,571,625.5**	**$29,730,517.1**

h) Now experiment by pivoting the matrix visual to display the exact same data using a different layout. Accomplish this by moving the **Month in Year** field into the **Rows** well and then moving the **Year** field into the **Columns** well. In effect, the **Year** column and the **Month in Year** column are switching places.

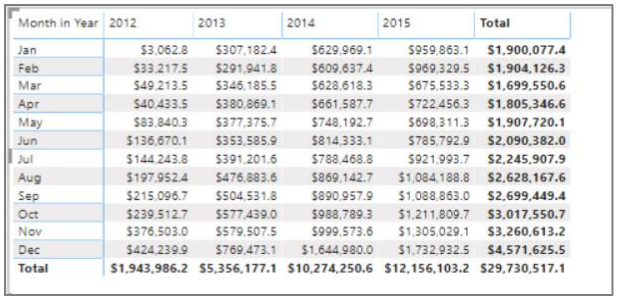


Rows
Month in Year

Columns
Year

Values
Sales Revenue

i) The matrix should now display a row for each month and a column for each year.



Month in Year	2012	2013	2014	2015	Total
Jan	$3,062.8	$307,182.4	$629,969.1	$959,863.1	**$1,900,077.4**
Feb	$33,217.5	$291,941.8	$609,637.4	$969,329.5	**$1,904,126.3**
Mar	$49,213.5	$346,185.5	$628,618.3	$675,533.3	**$1,699,550.6**
Apr	$40,433.5	$380,869.1	$661,587.7	$722,456.3	**$1,805,346.6**
May	$83,840.3	$377,375.7	$748,192.7	$698,311.3	**$1,907,720.1**
Jun	$136,670.1	$353,585.9	$814,333.1	$785,792.9	**$2,090,382.0**
Jul	$144,243.8	$391,201.6	$788,468.8	$921,993.7	**$2,245,907.9**
Aug	$197,952.4	$476,883.6	$869,142.7	$1,084,188.8	**$2,628,167.6**
Sep	$215,096.7	$504,531.8	$890,957.9	$1,088,863.0	**$2,699,449.4**
Oct	$239,512.7	$577,439.0	$988,789.3	$1,211,809.7	**$3,017,550.7**
Nov	$376,503.0	$579,507.5	$999,573.6	$1,305,029.1	**$3,260,613.2**
Dec	$424,239.9	$769,473.1	$1,644,980.0	$1,732,932.5	**$4,571,625.5**
Total	**$1,943,986.2**	**$5,356,177.1**	**$10,274,250.6**	**$12,156,103.2**	**$29,730,517.1**

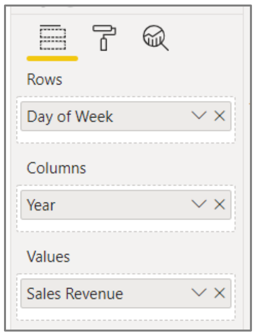
13. Create a new matrix visual to show sales revenue for specific time periods.

a) Add a second matrix visual to the page.

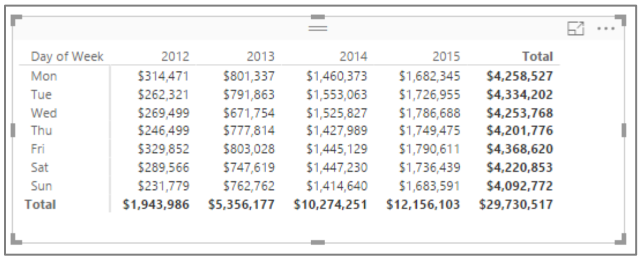
b) Add the **Day of Week** column from the **Calendar** table into the **Rows** well.

c) Add the **Year** column from the **Calendar** table into the **Columns** well.

d) Add the **Sales Revenue** measure from the **Sales** table into the **Values** well.



e) The matrix should now display a row for each day of the week and a column for each year.



Day of Week	2012	2013	2014	2015	Total
Mon	$314,471	$801,337	$1,460,373	$1,682,345	**$4,258,527**
Tue	$262,321	$791,863	$1,553,063	$1,726,955	**$4,334,202**
Wed	$269,499	$671,754	$1,525,827	$1,786,688	**$4,253,768**
Thu	$246,499	$777,814	$1,427,989	$1,749,475	**$4,201,776**
Fri	$329,852	$803,028	$1,445,129	$1,790,611	**$4,368,620**
Sat	$289,566	$747,619	$1,447,230	$1,736,439	**$4,220,853**
Sun	$231,779	$762,762	$1,414,640	$1,683,591	**$4,092,772**
Total	**$1,943,986**	**$5,356,177**	**$10,274,251**	**$12,156,103**	**$29,730,517**

You can see that once you have set up your calendar table, it's easy to layout table visuals and matrix visuals to display summarized financial data over various time periods.

Exercise 3: Create Measures using DAX Time Intelligence Functions

In this exercise, you will leverage various the Time Intelligence functions in DAX to analyze sales revenue using quarter to date (QTD) totals and year to date (YTD) totals. You will also write a DAX expression to calculate a running total of sales revenue through the entire 4 years of sales activity. After that, you will create new measures to calculate the growth of sales revenue on a month-by-month basis.

1. Create a measure named **Sales Revenue QTD** that calculates a quarter-to-date aggregate sum on the **SalesAmount** column.

   a) Navigate to **Data** view.

   b) Select the **Sales** table from the **Fields** list.

   c) Create a new measure by clicking the **New Measure** button in the ribbon.

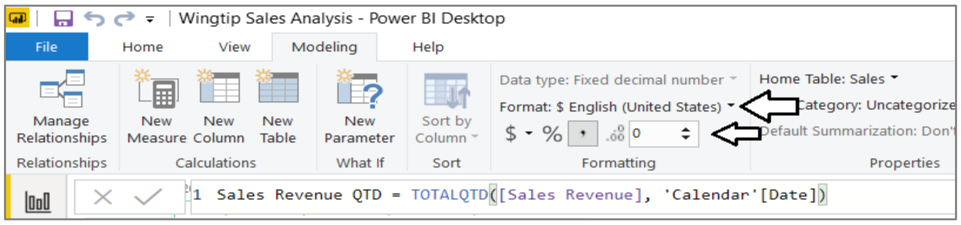
   d) Enter to following DAX expression into the formula bar to create the new measure named **Sales Revenue QTD**.

```
Sales Revenue QTD = TOTALQTD([Sales Revenue], 'Calendar'[Date])
```

   e) Press the **ENTER** key to add the measure to data model.

   f) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Currency > English (United States)**.

   g) Use the spinner control below the **Format** menu to set the number of decimal places shown to zero.



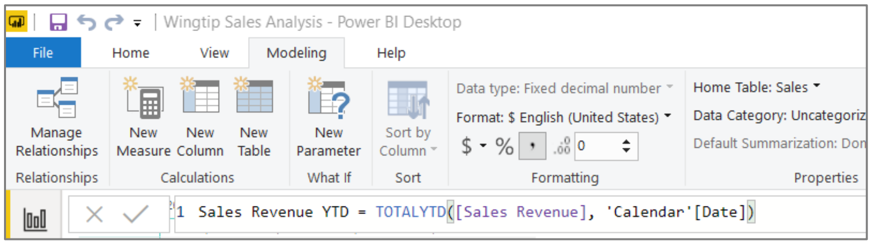
Data type: Fixed decimal number
Format: $ English (United States)
$ - % 0

```
Sales Revenue QTD = TOTALQTD([Sales Revenue], 'Calendar'[Date])
```

2. Create a measure named **Sales Revenue YTD** that calculates a year-to-date aggregate sum using the **Sales Revenue** measure.
   a) Navigate to **Data** view.
   b) Select the **Sales** table from the **Fields** list.
   c) Create a new measure by clicking the **New Measure** button in the ribbon.
   d) Enter to following DAX expression into the formula bar to create the new measure named **Sales Revenue YTD**.

```
Sales Revenue YTD = TOTALYTD([Sales Revenue], 'Calendar'[Date])
```

   e) Press the **ENTER** key to add the measure to data model.
   f) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Currency > English (United States)**.
   g) Use the spinner control below the **Format** menu to set the number of decimal places shown to zero.



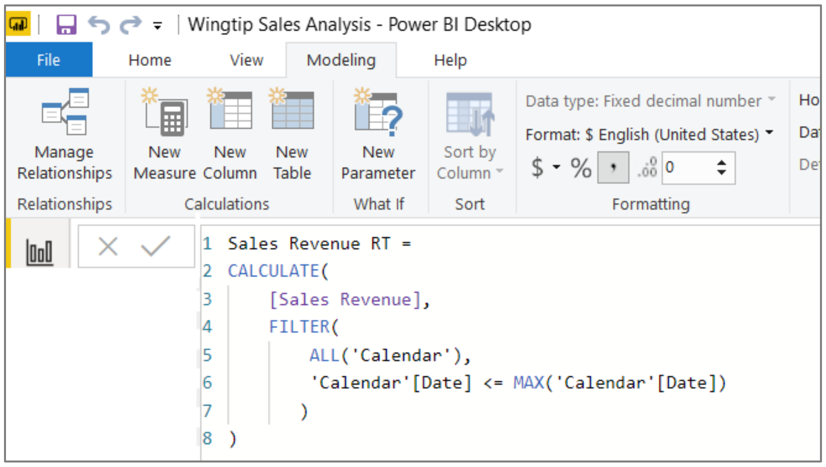
Data type: Fixed decimal number
Format: $ English (United States)
$ - % , .00 0
Home Table: Sales
Data Category: Uncategoriz
Default Summarization: Don

```
1  Sales Revenue YTD = TOTALYTD([Sales Revenue], 'Calendar'[Date])
```

3. Create a measure named **Sales Revenue RT** that calculates a running total aggregate sum the **Sales Revenue** measure.
   a) Navigate to **Data** view.
   b) Select the **Sales** table from the **Fields** list.
   c) Create a new measure by clicking the **New Measure** button in the ribbon.
   d) Enter to following DAX expression into the formula bar to create the new measure named **Sales Revenue RT**.

```
Sales Revenue RT =
CALCULATE(
    [Sales Revenue],
    FILTER(
        ALL('Calendar'),
        'Calendar'[Date] <= MAX('Calendar'[Date])
    )
)
```

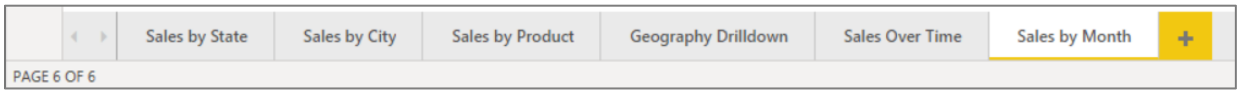
   e) Press the **ENTER** key to add the measure to data model.
   f) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Currency > English (United States)**.
   g) Use the spinner control below the format menu to set the number of decimal places shown to zero.



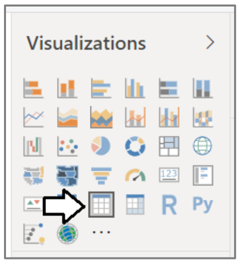
Data type: Fixed decimal number
Format: $ English (United States)
$ - % , .00 0

```
Sales Revenue RT =
CALCULATE(
    [Sales Revenue],
    FILTER(
        ALL('Calendar'),
        'Calendar'[Date] <= MAX('Calendar'[Date])
    )
)
```

4. Create a new report page named **Sales by Month**.
   a) Navigate to **Report** view.
   b) Add a new page by clicking the (**+**) button on the right side of the page navigation menu.
   c) Once the new page has been created, modify its title to **Sales by Month**.

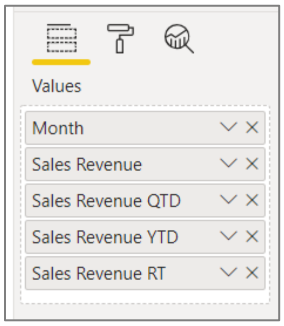


5. Create a new table visual to display data using your new measures.
   a) Add a new table visual on the right-hand side of the page

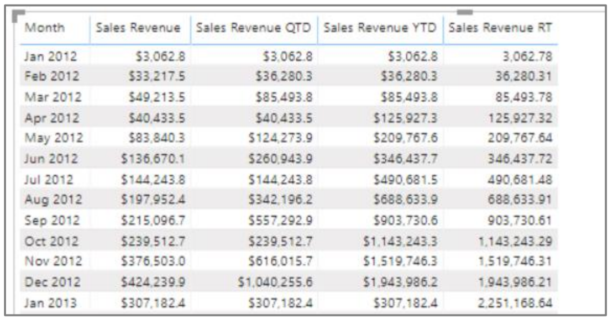


.

   b) With the table visual selected, navigate to **Fields** pane
   c) Add the **Month** column from **Calendar** table into the **Values** well.
   d) Add the measures **Sales Revenue**, **Sales Revenue QTD**, **Sales Revenue YTD**, **Sales Revenue RT** into the **Values** well.



   e) Inspect the data displayed in the table visual and verify the measures for sales revenue QTD, YTD and RT look correct.



Month	Sales Revenue	Sales Revenue QTD	Sales Revenue YTD	Sales Revenue RT
Jan 2012	$3,062.8	$3,062.8	$3,062.8	3,062.78
Feb 2012	$33,217.5	$36,280.3	$36,280.3	36,280.31
Mar 2012	$49,213.5	$85,493.8	$85,493.8	85,493.78
Apr 2012	$40,433.5	$40,433.5	$125,927.3	125,927.32
May 2012	$83,840.3	$124,273.9	$209,767.6	209,767.64
Jun 2012	$136,670.1	$260,943.9	$346,437.7	346,437.72
Jul 2012	$144,243.8	$144,243.8	$490,681.5	490,681.48
Aug 2012	$197,952.4	$342,196.2	$688,633.9	688,633.91
Sep 2012	$215,096.7	$557,292.9	$903,730.6	903,730.61
Oct 2012	$239,512.7	$239,512.7	$1,143,243.3	1,143,243.29
Nov 2012	$376,503.0	$616,015.7	$1,519,746.3	1,519,746.31
Dec 2012	$424,239.9	$1,040,255.6	$1,943,986.2	1,943,986.21
Jan 2013	$307,182.4	$307,182.4	$307,182.4	2,251,168.64

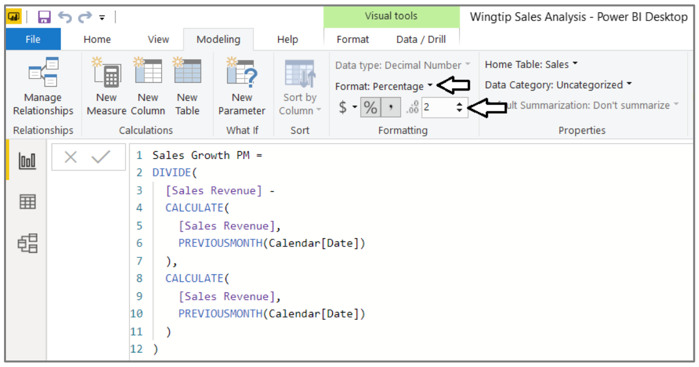
   f) Save the work you have done by clicking the **Save** button in the upper left corner of the Power BI Desktop window.

You have now learned how to use Time Intelligence functions in DAX together with a calendar table. Now you will move on to the final exercise where you will create additional measures to monitor sales growth. Over the next few steps, you will create new measures to calculate the growth of sales revenue on a month-by-month basis.

6. Create a measure named **Sales Growth PM** that calculates the percentage increase between sales revenue for the current month and sales revenue for the previous month.
   a) Navigate to **Data** view.
   b) Select the **Sales** table from the **Fields** list.
   c) Create a new measure by clicking the **New Measure** button in the ribbon.
   d) Enter to following DAX expression into the formula bar to create the measure named **Sales Growth PM**.

```
Sales Growth PM =
DIVIDE(
  [Sales Revenue] -
  CALCULATE(
    [Sales Revenue],
    PREVIOUSMONTH(Calendar[Date])
  ),
  CALCULATE(
    [Sales Revenue],
    PREVIOUSMONTH(Calendar[Date])
  )
)
```

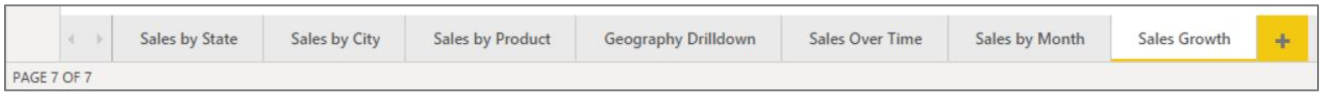
   e) Press the **ENTER** key to add the calculated column to data model.
   f) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Percentage**.
   g) Use the spinner control below the **Format** menu to set the number of decimal places shown to **2**.



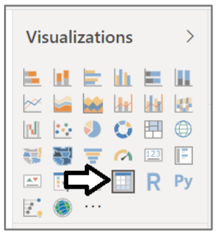
Data type: Decimal Number
Format: Percentage
$ % , .00 2
Home Table: Sales
Data Category: Uncategorized
Default Summarization: Don't summarize

```
Sales Growth PM =
DIVIDE(
  [Sales Revenue] -
  CALCULATE(
    [Sales Revenue],
    PREVIOUSMONTH(Calendar[Date])
  ),
  CALCULATE(
    [Sales Revenue],
    PREVIOUSMONTH(Calendar[Date])
  )
)
```

7. Create a new page in the project's report.
   a) Navigate to **Report** view.
   b) Add a new page by clicking the (**+**) button on the right of the page navigation menu.
   c) Once the new page has been created, modify its title to **Sales Growth**.



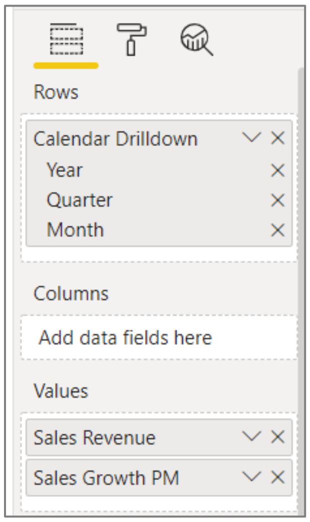
8. Create a new matrix visual to show month-to-month sales revenue growth in 2014 and 2015.
   a) Click the matrix icon in the **Visualizations** list to add a new matrix visual to the page.



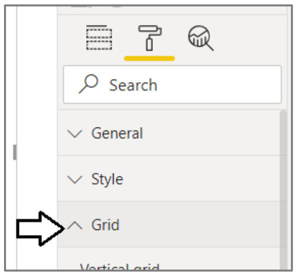
b) Drag and drop the **Calendar Drilldown** hierarchy from the **Calendar** table into the **Rows** well.

c) Drag and drop the **Sales Revenue** measure from the **Sales** table into the **Values** well.

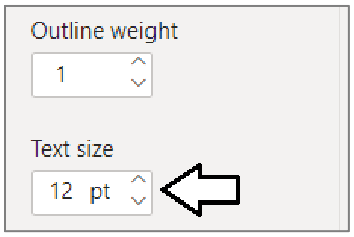
d) Drag and drop the **Sales Growth PM** measure from the **Sales** table into the **Values** well.



e) With the matrix visual selected, expand the **Grid** section in the **Format** pane.

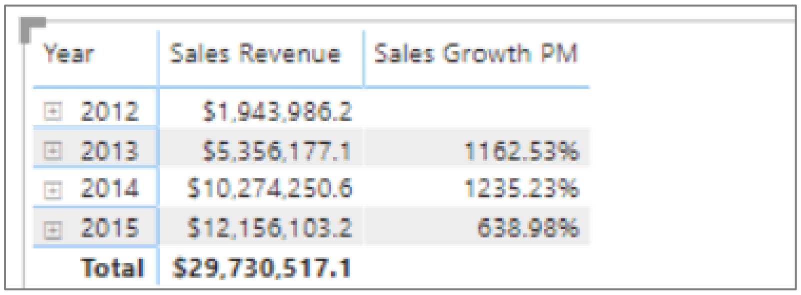


f) Inside the **Grid** section, locate the **Text** site property and increase its value to **12 pt**.


Outline weight
1

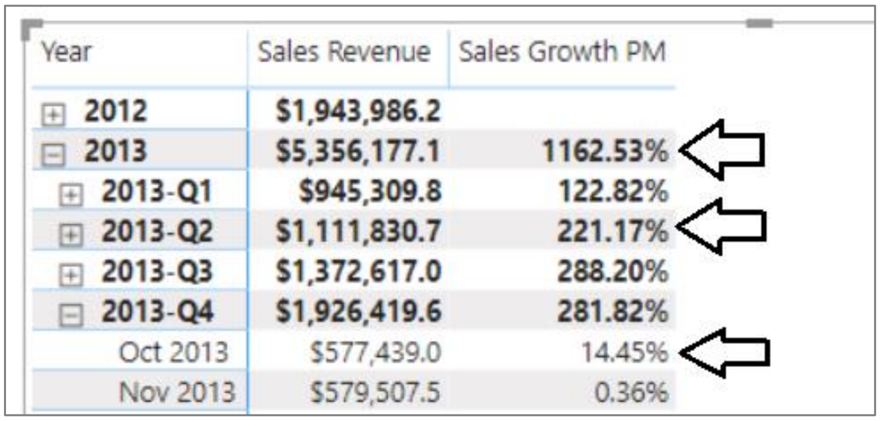
Text size
12 pt

g) The matrix rows should now display an expand/collapse icons at the left



Year	Sales Revenue	Sales Growth PM
⊞ 2012	$1,943,986.2	
⊞ 2013	$5,356,177.1	1162.53%
⊞ 2014	$10,274,250.6	1235.23%
⊞ 2015	$12,156,103.2	638.98%
Total	**$29,730,517.1**	

h) Inspect the values produced by the **Sales Growth PM** measure at the level of year, quarter and month.



Year	Sales Revenue	Sales Growth PM
2012	$1,943,986.2	
2013	$5,356,177.1	1162.53%
2013-Q1	$945,309.8	122.82%
2013-Q2	$1,111,830.7	221.17%
2013-Q3	$1,372,617.0	288.20%
2013-Q4	$1,926,419.6	281.82%
Oct 2013	$577,439.0	14.45%
Nov 2013	$579,507.5	0.36%

The **Sales Growth PM** measure was written to perform calculations on a month-to-month basis. However, there is currently a problem whenever this measure is evaluated in a filter context based on a larger time interval such as a quarter or a year. More specifically, the **Sales Growth PM** measure is currently producing a large and erroneous value when it is evaluated in the context of a quarter or a year. Now that you understand problem, it’s time to modify the DAX expression for the **Sales Growth PM** measure to return a blank value whenever the measure is evaluated in a filter context where the time interval is at a granularity other than at the monthly level.

9. Modify the DAX expression for the **Sales Growth PM** measure.
   a) Navigate to **Data** view.
   b) Select the **Sales Growth PM** measure of the **Sales** table from the **Fields** list. When you select the **Sales Growth PM** measure in the **Fields** list, you should then be able to see and modify its DAX expression in the formula bar.
   c) Before you can modify the DAX expression for the **Sales Growth PM** measure, you must be able to use the ISFILTERED function provided by DAX. You can write the following DAX expression to determine whether the current evaluation context is filtering at the month level.

```
ISFILTERED(Calendar[Month])
```

   d) You can also write the following DAX expression to make sure that the current evaluation context is not filtering at a more granular level such as at the **Date** level.

```
NOT(ISFILTERED(Calendar[Date]))
```

   e) You will need to ensure that both these expressions are true before the **Sales Growth PM** measure evaluates to a value other than a blank value. You can write the following DAX expression using the DAX **&&** operator to return true when both inner conditions are true

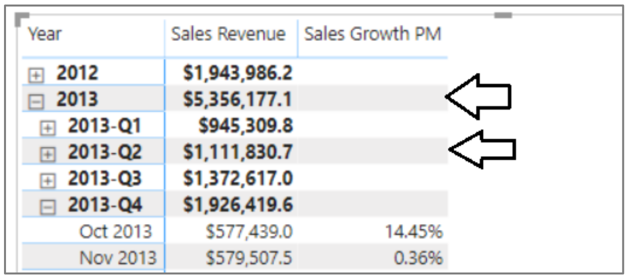
```
ISFILTERED(Calendar[Month]) && NOT(ISFILTERED(Calendar[Date]))
```

   f) Update the DAX expression for the **Sales Growth PM** measure to match the following code listing.

```
Sales Growth PM =
IF(
  ( ISFILTERED(Calendar[Month]) && NOT(ISFILTERED(Calendar[Date])) ),
  DIVIDE(
    [Sales Revenue] -
    CALCULATE(
      [Sales Revenue],
      PREVIOUSMONTH(Calendar[Date])
    ),
    CALCULATE(
      [Sales Revenue],
      PREVIOUSMONTH(Calendar[Date])
    )
  ),
  BLANK()
)
```

g) Navigate back to **Report** view and inspect the effects of your changes to the visual on the **Sales Revenue Growth** page.

h) You should see the **Sales Growth PM** measure is now returning blank values for the quarterly and yearly evaluations.



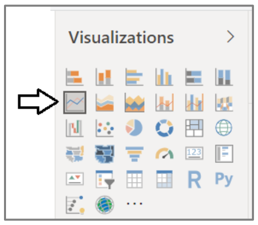
Year	Sales Revenue	Sales Growth PM
⊞ 2012	$1,943,986.2	
⊟ 2013	$5,356,177.1	
⊞ 2013-Q1	$945,309.8	
⊞ 2013-Q2	$1,111,830.7	
⊞ 2013-Q3	$1,372,617.0	
⊟ 2013-Q4	$1,926,419.6	
Oct 2013	$577,439.0	14.45%
Nov 2013	$579,507.5	0.36%

☺ It is widely-accepted among BI experts and BI novices alike that a blank value is always preferable to a large, erroneous value.

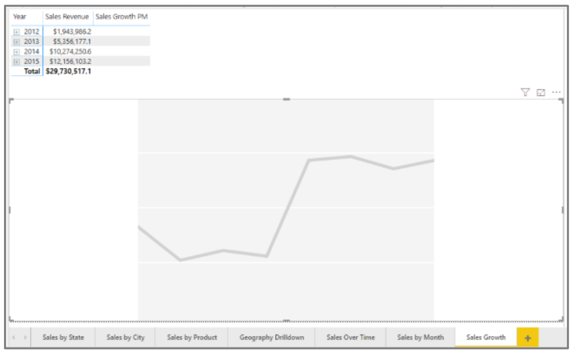
10. Add a line chart visual to the bottom of the report page to show how sales revenue has grown from month to month.

a) Click on the whitespace on the report to make sure that the existing matrix visual is not currently selected.

b) Click on the **Line chart** button in the **Visualizations** list to create a new Line chart visual.



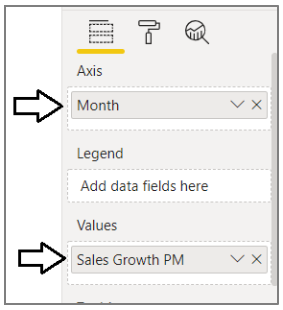
c) Reposition the new visual so it takes up the entire page width at the bottom of the page.



Year	Sales Revenue	Sales Growth PM
⊞ 2012	$1,943,986.2	
⊞ 2013	$5,356,177.1	
⊞ 2014	$10,274,250.6	
⊞ 2015	$12,156,103.2	
Total	$29,730,517.1	

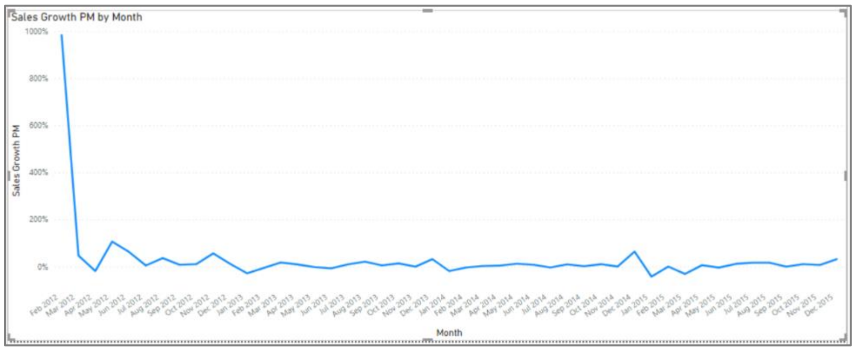
d) Drag and drop the **Month** field from the **Calendar** table into the **Axis** well.

e) Drag and drop the **Sales Growth PM** measure from the **Sales** table into the **Values** well.

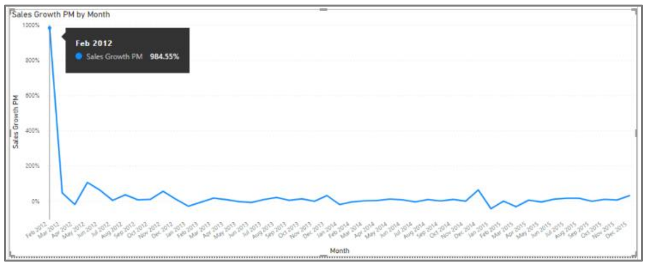


Axis: Month
Legend: Add data fields here
Values: Sales Growth PM

f) You should now see a Line chart which shows the month-to-month growth in sales revenue from 2012 through 2015.



11. Observe the problem with the monthly sales growth in February of 2012.
   a) Examine the sales growth in **Feb 2012** which has a value of 985%.



The problem with this sales growth figure for Feb 2012 has to do with the fact that the previous month is not a complete month but instead only contains 4 days of sales data from January 28 to January 31. Over the next few steps you will refine your DAX code to add additional logic so that the **Sales Growth PM** measure returns a blank value when the previous month is not a full month.

12. Create a measure named **Previous Month Is Valid** to indicate whether the previous month is a complete month or not.
   a) Navigate to data view and select the **Sales** table from the **Fields** list.
   b) Create a new measure by clicking the **New Measure** button in the ribbon.
   c) Enter to following DAX expression into the formula bar to create the measure named **Previous Month Is Valid**.

```
Previous Month Is Valid =
  FIRSTDATE(PREVIOUSMONTH('Calendar'[Date])) >= FIRSTDATE(ALL(Sales[InvoiceDate]))
```

Note that the new measure named **Previous Month Is Valid** will not be used directly in any report. Instead, you have created this measure to call from the DAX code you write in other measures. Since you will only reference this measure from other measures, it makes sense to hide this measure from **Report** view.

13. Once you have created the **Previous Month Is Valid** measure, right click on it in the fields list and click **Hide in Report View**.
14. Update the DAX code for the **Sales Growth PM** measure to return a blank value when the previous month is incomplete.
   a) Select the **Sales Growth PM** measure so you can see its DAX in the formula editor.
   b) Currently, the **If** statement at the top has two conditions.

```
( ISFILTERED(Calendar[Month]) && NOT(ISFILTERED(Calendar[Date])) )
```

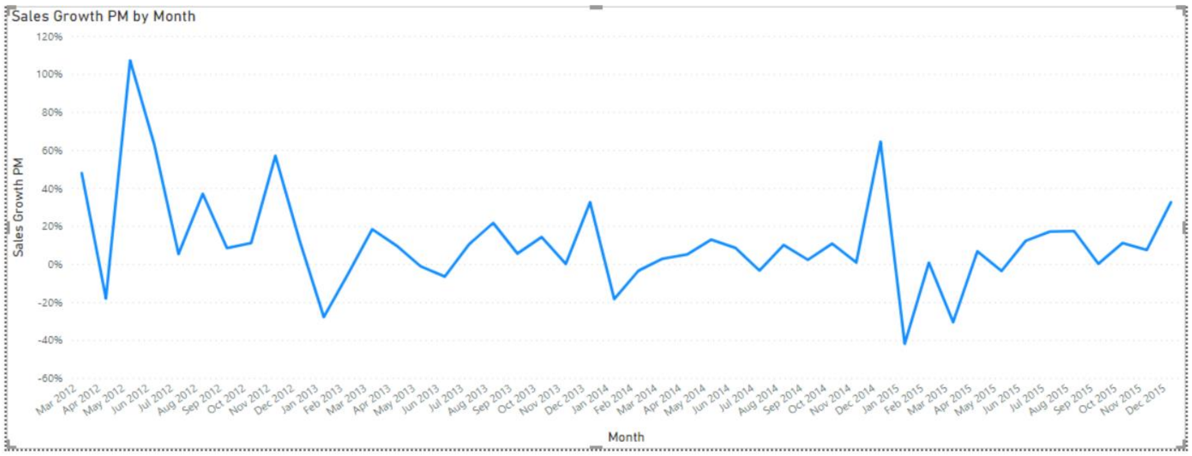
   c) Update the **If** statement to add a third condition to ensure the **Previous Month Is Valid** measure is true.

```
(
  ISFILTERED(Calendar[Month]) &&
  NOT(ISFILTERED(Calendar[Date])) &&
  [Previous Month Is Valid]
)
```

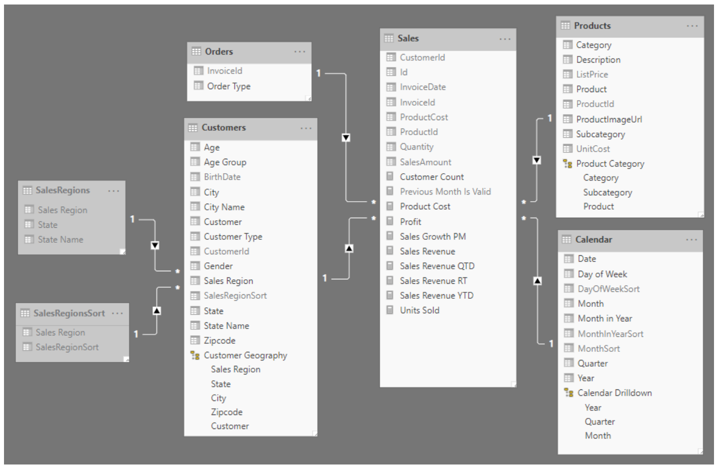
d) The DAX expression for the **Sales Growth PM** measure should now match the following code listing.

```
Sales Growth PM =
IF(
  (
    ISFILTERED(Calendar[Month]) &&
    NOT(ISFILTERED(Calendar[Date])) &&
    [Previous Month Is Valid]
  ),
  DIVIDE(
    [Sales Revenue] -
    CALCULATE(
      [Sales Revenue],
      PREVIOUSMONTH(Calendar[Date])
    ),
    CALCULATE(
      [Sales Revenue],
      PREVIOUSMONTH(Calendar[Date])
    )
  ),
  BLANK()
)
```

15. Return to report view and see the effects of the changes you just made to the **Sales Growth PM** measure. You should see that the spike in the first month is gone and the line chart provides a better view of sales revenue growth of the four years of sales data.



16. Complete your work on the data model by rearranging the table layout in **Model** view so you can see every table and field.



17. Save your work to the **Wingtip Sales Model** project by clicking the **Save** button in the ribbon.

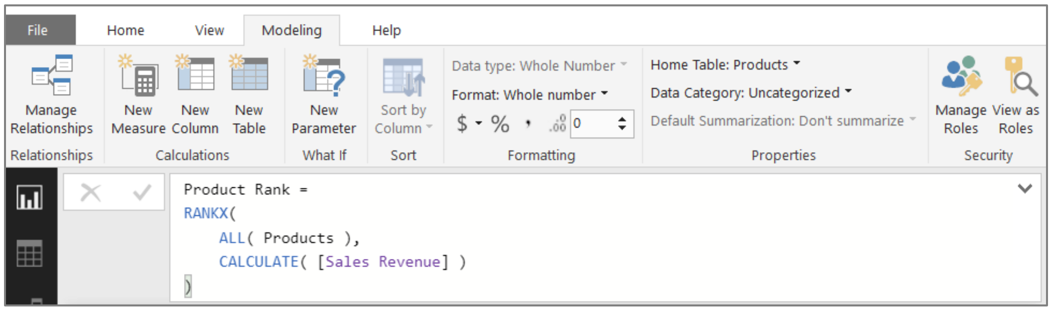
Exercise 4: Create the Top 5 Products Report

In this exercise you will create a measure named **Product Rank** that ranks products according to their total sales revenue. You will then work to create a report that displays the top 5 selling products. Along the way, you will design this report to be interactive allowing the user to filter on a specific year and/or a specific product category to see what products are the best sellers.

1. Create a new measure named **Product Rank** to determine the top selling products.
   a) Navigate to data view.
   b) Select the **Sales** table from the **Fields** list.
   c) Create a new measure by clicking the **New Measure** button in the ribbon.
   d) Enter to following DAX expression into the formula bar to create the measure named **Product Rank**.

```
Product Rank =
RANKX(
 ALL( Products ),
 CALCULATE( [Sales Revenue] )
)
```

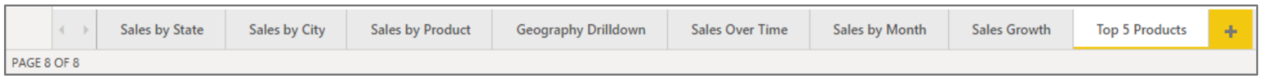
   e) Press the **ENTER** key to add the measure to the data model.
   f) Ensure the formatting for this measure is set to **Whole Number** as shown in the following screenshot.



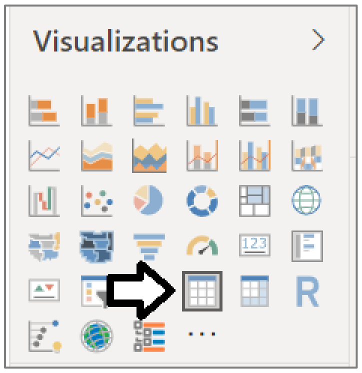
Data type: Whole Number
Format: Whole number
Home Table: Products
Data Category: Uncategorized
Default Summarization: Don't summarize

```
Product Rank =
RANKX(
    ALL( Products ),
    CALCULATE( [Sales Revenue] )
)
```

2. Create a new report page named **Top 5 Products**
   a) Navigate to report view.
   b) Create a new report page and rename it to **Top 5 Products**.



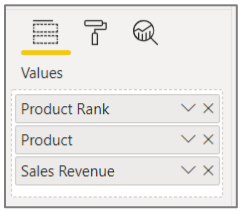
3. Add a new table visual to display the top 5 products.
   a) Click the **New Visual** button on the ribbon to add a new visual to the page.
   b) Change the visual to a table by clicking the **Table** button in the **Visualizations** list.



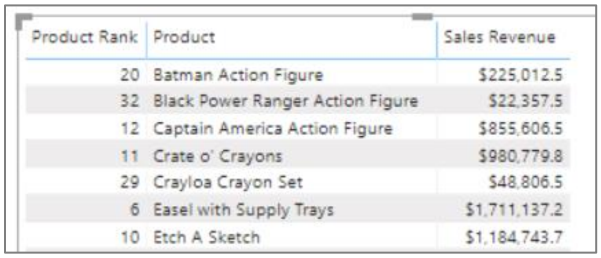
c) Drag and drop the **Product Rank** measure from the **Sales** table into the **Values** well.

d) Drag and drop the **Product** column from the **Products** table into the **Rows** well.

e) Drag and drop the **Sales Revenue** measure from the **Sales** table into the **Rows** well.

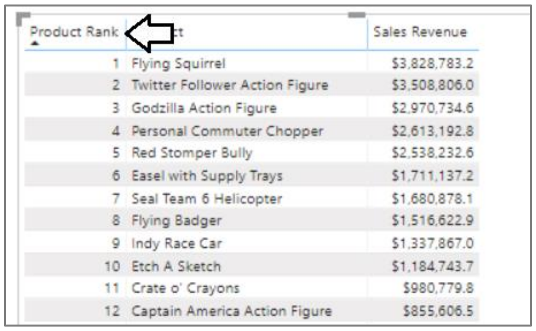

Values
Product Rank
Product
Sales Revenue

f) The new visual should now match the visual shown in the following screenshot.



Product Rank	Product	Sales Revenue
20	Batman Action Figure	$225,012.5
32	Black Power Ranger Action Figure	$22,357.5
12	Captain America Action Figure	$855,606.5
11	Crate o' Crayons	$980,779.8
29	Crayloa Crayon Set	$48,806.5
6	Easel with Supply Trays	$1,711,137.2
10	Etch A Sketch	$1,184,743.7

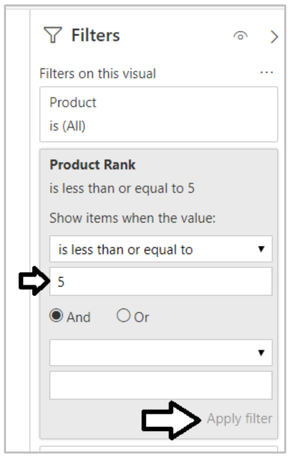
g) Click on the **Product Rank** column header twice to sort the visual so the products with the lowest ranks are sorted to the top.



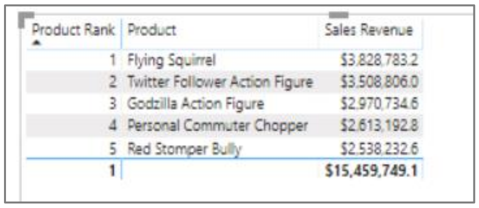
Product Rank	Product	Sales Revenue
1	Flying Squirrel	$3,828,783.2
2	Twitter Follower Action Figure	$3,508,806.0
3	Godzilla Action Figure	$2,970,734.6
4	Personal Commuter Chopper	$2,613,192.8
5	Red Stomper Bully	$2,538,232.6
6	Easel with Supply Trays	$1,711,137.2
7	Seal Team 6 Helicopter	$1,680,878.1
8	Flying Badger	$1,516,622.9
9	Indy Race Car	$1,337,867.0
10	Etch A Sketch	$1,184,743.7
11	Crate o' Crayons	$980,779.8
12	Captain America Action Figure	$855,606.5

h) Inspect the **Visual level filters** well of the **Filters** section of the Field properties pane and locate **Product Rank**.

i) Configure the **Product Rank** filter for the table visual to only display products with a **Product Rank** value of 5 or lower as shown in the following screenshot and then click the **Apply Filter** link to apply the filter to the visual.

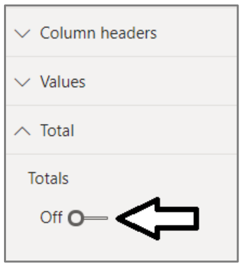

Filters
Filters on this visual
Product
is (All)
Product Rank
is less than or equal to 5
Show items when the value:
is less than or equal to
5
[✓ And] Or
Apply filter

j) Your visual should now display the top 5 selling products as shown in the following screenshot. You should be able to observe that the visual is displaying the **Totals** row at the bottom which needs to be removed.



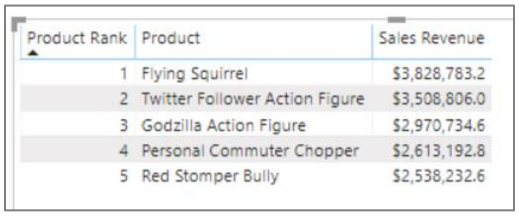
Product Rank	Product	Sales Revenue
1	Flying Squirrel	$3,828,783.2
2	Twitter Follower Action Figure	$3,508,806.0
3	Godzilla Action Figure	$2,970,734.6
4	Personal Commuter Chopper	$2,613,192.8
5	Red Stomper Bully	$2,538,232.6
1		$15,459,749.1

k) Locate the **Totals** property for the table visual in the **Format** pane and set its value to **Off**.



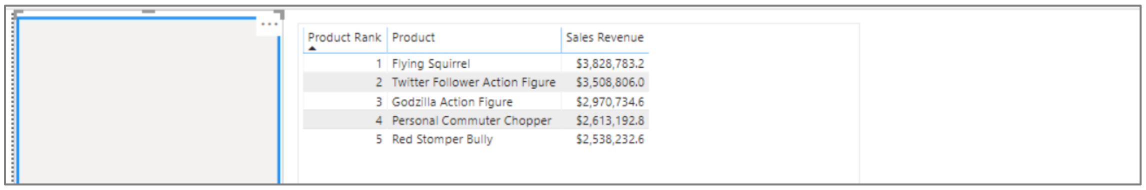
Totals
Off

l) Your top 5 products visual should now look better when it is displayed without the **Totals** row.



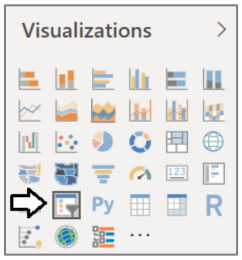
Product Rank	Product	Sales Revenue
1	Flying Squirrel	$3,828,783.2
2	Twitter Follower Action Figure	$3,508,806.0
3	Godzilla Action Figure	$2,970,734.6
4	Personal Commuter Chopper	$2,613,192.8
5	Red Stomper Bully	$2,538,232.6

4. Create a rectangle shape to provide background formatting for the report page.
   a) Drop down the **Shapes** menu and select the **Rectangle** command to add a new shape to the report.
   b) Using the mouse, resize the rectangle share to take up the full height of the report page and about 20% of the width.

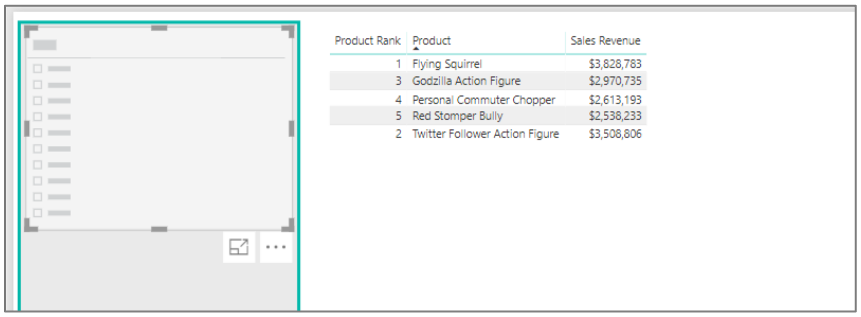


Product Rank	Product	Sales Revenue
1	Flying Squirrel	$3,828,783.2
2	Twitter Follower Action Figure	$3,508,806.0
3	Godzilla Action Figure	$2,970,734.6
4	Personal Commuter Chopper	$2,613,192.8
5	Red Stomper Bully	$2,538,232.6

5. Add a new slicer visual to the page to filter the top 5 products visual by **Year**.
   a) Click the **New Visual** button on the ribbon to add a new visual to the page.
   b) Change the visual to a slicer by clicking the Slicer button in the **Visualizations** list.

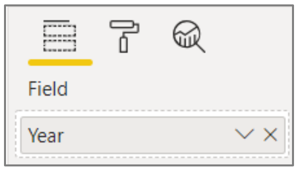


c) Position the slicer on top of the rectangle.

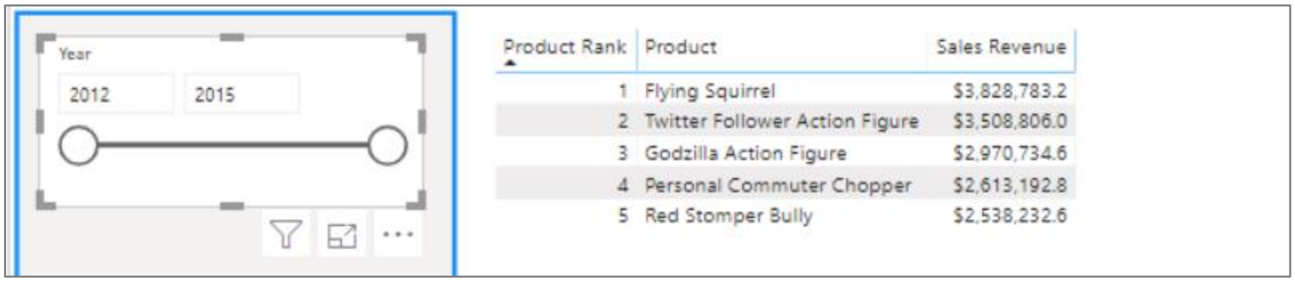


Product Rank	Product	Sales Revenue
1	Flying Squirrel	$3,828,783
3	Godzilla Action Figure	$2,970,735
4	Personal Commuter Chopper	$2,613,193
5	Red Stomper Bully	$2,538,233
2	Twitter Follower Action Figure	$3,508,806

d) Drag and drop the **Year** column from the **Calendar** table into the **Values** well.

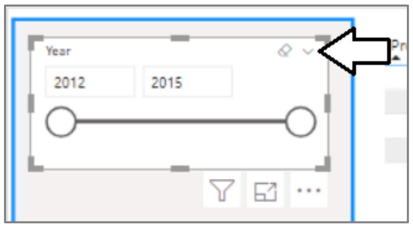


e) The slicer visual will default to a user interface experience with slider since year is a whole number.

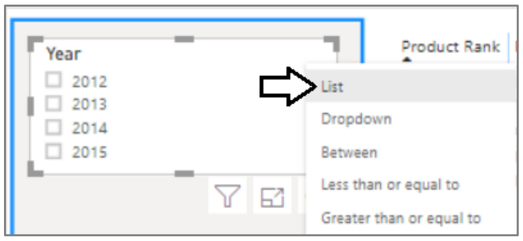


Product Rank	Product	Sales Revenue
1	Flying Squirrel	$3,828,783.2
2	Twitter Follower Action Figure	$3,508,806.0
3	Godzilla Action Figure	$2,970,734.6
4	Personal Commuter Chopper	$2,613,192.8
5	Red Stomper Bully	$2,538,232.6

f) Click the slicer dropdown menu at the top right of the slicer to change the user interface experience.

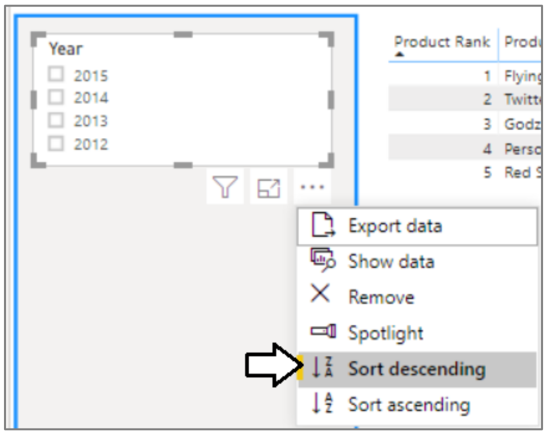


g) Select the **List** option for the slicer's user interface experience.



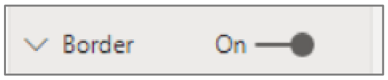
Now, that you have configured the slicer as a list, you will reverse the sort order so that later years are listed on top.

h) Drop down use the standard visuals menu and select the **Sort Descending** command.

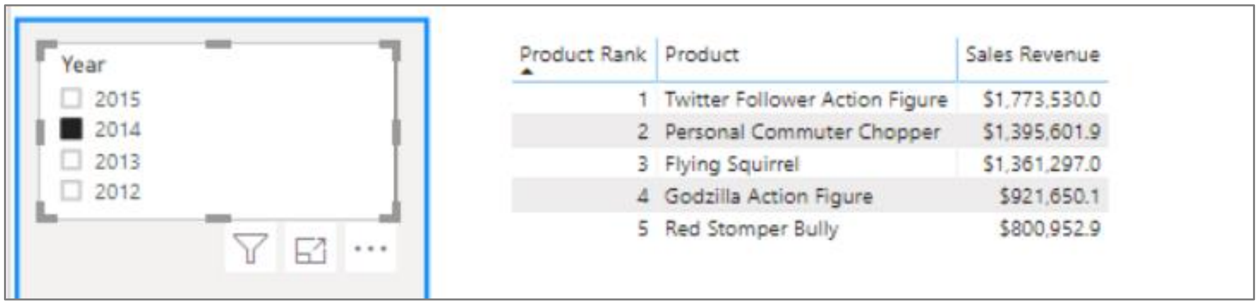


i) The **Year** slicer visual should now show the latest year (i.e. 2015) at the top of the list of years.

j) With the slicer visual selected, navigate to the Format pane and set the Border property to **On**.



k) Experiment using the **Year** slicer by selecting individual years. You should see that the top 5 products visual updates whenever you select a different year.



Product Rank	Product	Sales Revenue
1	Twitter Follower Action Figure	$1,773,530.0
2	Personal Commuter Chopper	$1,395,601.9
3	Flying Squirrel	$1,361,297.0
4	Godzilla Action Figure	$921,650.1
5	Red Stomper Bully	$800,952.9

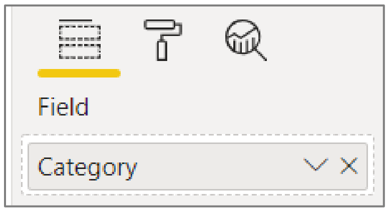
l) When you are done, reset he Year slicer so that no year is selected.

6. Add a second slicer visual to the **Top 5 Products** page to filter by **Category**.

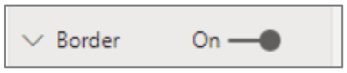
a) Click the **New Visual** button on the ribbon to add a new visual to the page.

b) Change the visual to a slicer by clicking the Slicer button in the **Visualizations** list.

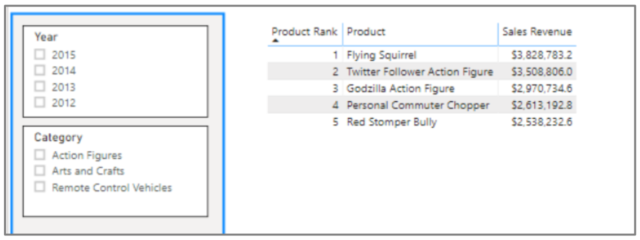
c) Drag and drop the **Category** column from the **Products** table into the **Values** well.



d) With the Category slicer visual selected, navigate to the Format pane and set the Border property to **On**.

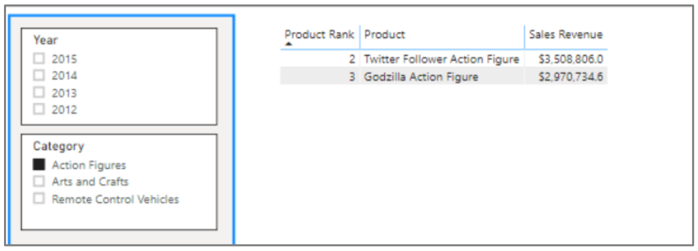


e) Reposition the new slicer visual to match the page layout shown in the following screenshot.



Product Rank	Product	Sales Revenue
1	Flying Squirrel	$3,828,783.2
2	Twitter Follower Action Figure	$3,508,806.0
3	Godzilla Action Figure	$2,970,734.6
4	Personal Commuter Chopper	$2,613,192.8
5	Red Stomper Bully	$2,538,232.6

f) Experiment using the **Category** slicer by selecting individual product categories. You should see that there is now a problem with the report because the visual with the top 5 products doesn't show 5 products.



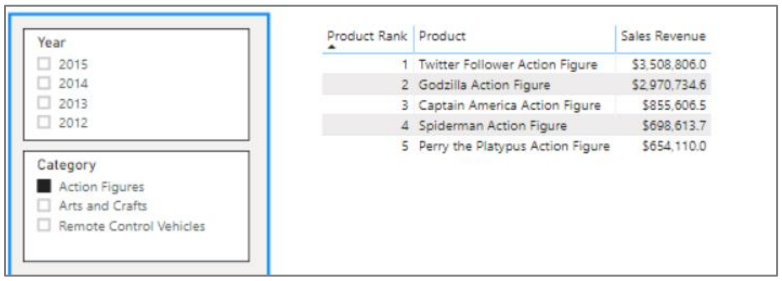
Product Rank	Product	Sales Revenue
2	Twitter Follower Action Figure	$3,508,806.0
3	Godzilla Action Figure	$2,970,734.6

The problem you are facing here has to do with the manner in which the **Product Rank** measure is filtering during its evaluation. The problem is that the measure does not correctly filter by the product category column when determining the top 5 products. Therefore, you must modify the DAX expression for the **Product Rank** measure in order to calculate the top 5 selling products within a specific category when that category is selected in the slicer.

7. Modify the DAX expressions for the **Product Rank** measure to correct the filter problem with product category.
   a) Navigate to data view.
   b) Expand the **Sales** table from the **Fields** list.
   c) Select the **Product Rank** measure in the **Sales** table so you can view and modify its DAX expression in the formula bar.
   d) Modify the DAX expression for the **Product Rank** measure to match the following code listing.

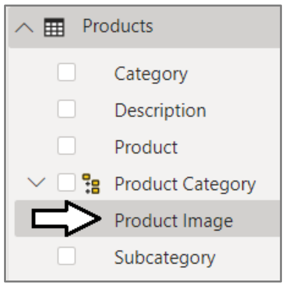
```
Product Rank =
IF(
  HASONEVALUE(Products[Product]),
  RANKX(
    ALL( Products[Subcategory], Products[Product] ),
    CALCULATE( [Sales Revenue] )
  )
)
```

8. Test the changes you made to the **Product Rank** measure.
   a) Navigate to report view.
   b) Test the measure by selecting different categories using the **Category** slicer. At this point, the page filtering should be working correctly as you should see 5 top products when selecting a product category.

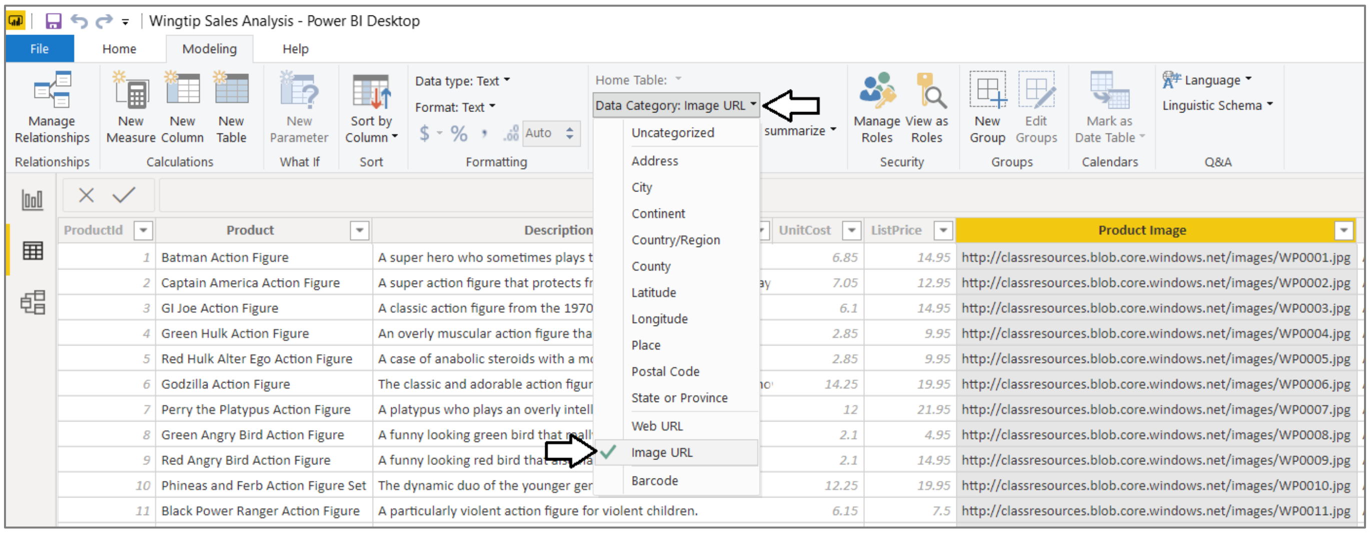


Product Rank	Product	Sales Revenue
1	Twitter Follower Action Figure	$3,508,806.0
2	Godzilla Action Figure	$2,970,734.6
3	Captain America Action Figure	$855,606.5
4	Spiderman Action Figure	$698,613.7
5	Perry the Platypus Action Figure	$654,110.0

9. Add support the data model to provide a product image into the report.
   a) Navigate to Data View and then inspect the fields inside the **Products** table.
   b) Right-click on the **ProductImageUrl** field and select the **Rename** command.
   c) Rename the field to the more user-friendly name of **Product Image**.



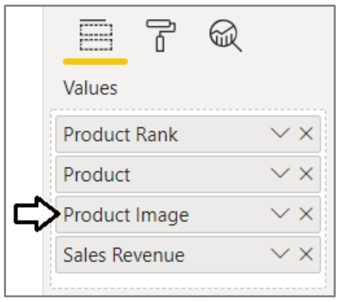
   d) Make sure you have the **Product Image** field selected in the **Fields** list.
   e) Navigate to the **Modeling** tab in the ribbon.
   f) Drop down the **Data Category** dropdown menu and select **Image URL**.



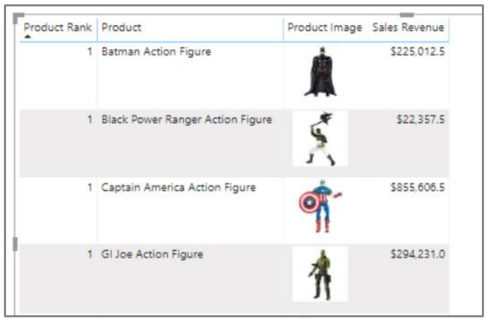
Data type: Text
Format: Text
Home Table:
Data Category: Image URL

ProductId	Product	Description	UnitCost	ListPrice	Product Image
1	Batman Action Figure	A super hero who sometimes plays t	6.85	14.95	http://classresources.blob.core.windows.net/images/WP0001.jpg
2	Captain America Action Figure	A super action figure that protects fr	7.05	12.95	http://classresources.blob.core.windows.net/images/WP0002.jpg
3	GI Joe Action Figure	A classic action figure from the 1970	6.1	14.95	http://classresources.blob.core.windows.net/images/WP0003.jpg
4	Green Hulk Action Figure	An overly muscular action figure tha	2.85	9.95	http://classresources.blob.core.windows.net/images/WP0004.jpg
5	Red Hulk Alter Ego Action Figure	A case of anabolic steroids with a m	2.85	9.95	http://classresources.blob.core.windows.net/images/WP0005.jpg
6	Godzilla Action Figure	The classic and adorable action figur	14.25	19.95	http://classresources.blob.core.windows.net/images/WP0006.jpg
7	Perry the Platypus Action Figure	A platypus who plays an overly intell	12	21.95	http://classresources.blob.core.windows.net/images/WP0007.jpg
8	Green Angry Bird Action Figure	A funny looking green bird that reall	2.1	4.95	http://classresources.blob.core.windows.net/images/WP0008.jpg
9	Red Angry Bird Action Figure	A funny looking red bird that	2.1	14.95	http://classresources.blob.core.windows.net/images/WP0009.jpg
10	Phineas and Ferb Action Figure Set	The dynamic duo of the younger ger	12.25	19.95	http://classresources.blob.core.windows.net/images/WP0010.jpg
11	Black Power Ranger Action Figure	A particularly violent action figure for violent children.	6.15	7.5	http://classresources.blob.core.windows.net/images/WP0011.jpg

10. Add the product image to the report.
    a) Return to **Report** view.
    b) Make sure the table visual is selected.
    c) Drag and drop the **Product Image** field from the **Products** table into the **Values** well. When you add the **Product Image** field in the **Values** well, place it in between the **Product** field and the **Sales Revenue** field as shown in the following screenshot.



d) When you see the effects your change, you will notice there's a problem because every product has a rank of 1. Therefore, the table visual now displays all 32 products instead of just 5 products which are the best sellers.



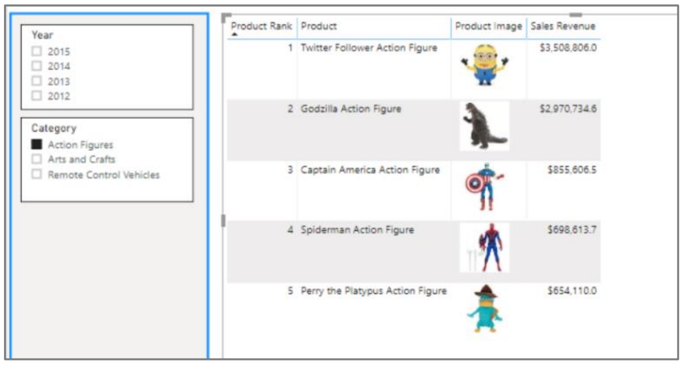
Product Rank	Product	Product Image	Sales Revenue
1	Batman Action Figure		$225,012.5
1	Black Power Ranger Action Figure		$22,357.5
1	Captain America Action Figure		$855,606.5
1	GI Joe Action Figure		$294,231.0

What's the problem here? It has to do with how the **RANKX** function works when the **Product Image** field is added into the filter context inside the table visual. In particular, the **RANKX** function is now calculating the ranking separately for each group of products that share the same product image. Since no two products share the same product image, each product gets a ranking of 1.

11. Modify the DAX for the **Product Rank** measure to ignore the **Product Image** field whenever it's added to the filter context.
    a) Navigate to data view.
    b) Expand the **Products** table from the **Fields** list.
    c) Select the **Product Rank** measure in the **Products** table so you can view and modify its DAX expression in the formula bar.
    d) Modify the DAX expression for the **Product Rank** measure by adding the **Product Image** field to the call to the **All** function.

```
Product Rank =
IF(
  HASONEVALUE(Products[Product]),
  RANKX(
    ALL( Products[Subcategory], Products[Product], Products[Product Image] ),
    CALCULATE( [Sales Revenue] )
  )
)
```

    e) Press Enter to save your DAX changes to the **Product Rank** field.
    f) Return to report view and inspect how your changes have affected the table visual with the top 5 products.
    g) You should see that now the product ranking is working the way it should even when there is a filter on product category.



Year
☐ 2015
☐ 2014
☐ 2013
☐ 2012

Category
■ Action Figures
☐ Arts and Crafts
☐ Remote Control Vehicles

Product Rank	Product	Product Image	Sales Revenue
1	Twitter Follower Action Figure		$3,508,806.0
2	Godzilla Action Figure		$2,970,734.6
3	Captain America Action Figure		$855,606.5
4	Spiderman Action Figure		$698,613.7
5	Perry the Platypus Action Figure		$654,110.0

At this point the **Product Rank** measure is working correctly. However, you will update the DAX for this measure one more time to so you can the effects of a valuable DAX function named AllSelected.

12. Modify the DAX for the **Product Rank** measure to use the DAX **AllSelected** function.
    a) Navigate to data view.
    b) Expand the **Products** table from the **Fields** list.

c) Select the **Product Rank** measure in the **Products** table so you can view and modify its DAX expression in the formula bar.

d) Modify the DAX expression for the **Product Rank** measure by adding the **Product Image** field to the call to the **All** function.

```
Product Rank =
IF(
  HASONEVALUE(Products[Product]),
  RANKX(
    ALLSELECTED(Products),
    CALCULATE( [Sales Revenue] )
  )
)
```

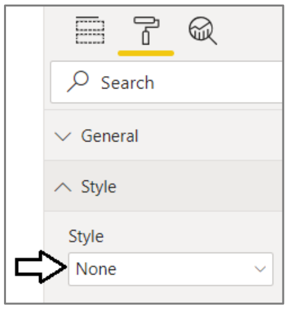
e) Press Enter to save your DAX changes to the **Product Rank** field.

f) Return to report view.

g) You should see that now the product ranking is working the way it should even when there is a filter on product category.

13. Style the Top 5 Products table visual.

a) In the **Table style** section, change the **Style** to **None**.

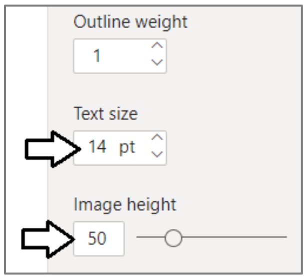


Style
None

b) Navigate to the **Grid** section in the **Format** properties pane.

c) Set the **Text Size** to **14 pt**.

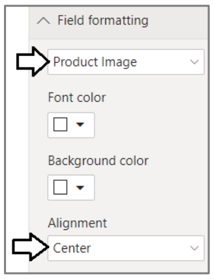
d) Set the **Image Height** to **50**.


Outline weight
1
Text size
14 pt
Image height
50

e) Navigate to the **Field Formatting** section in the **Format** properties pane.

f) Set the **Field** to be formatted to **Product Image**.

g) Set the field's **Alignment** property to **Center**.

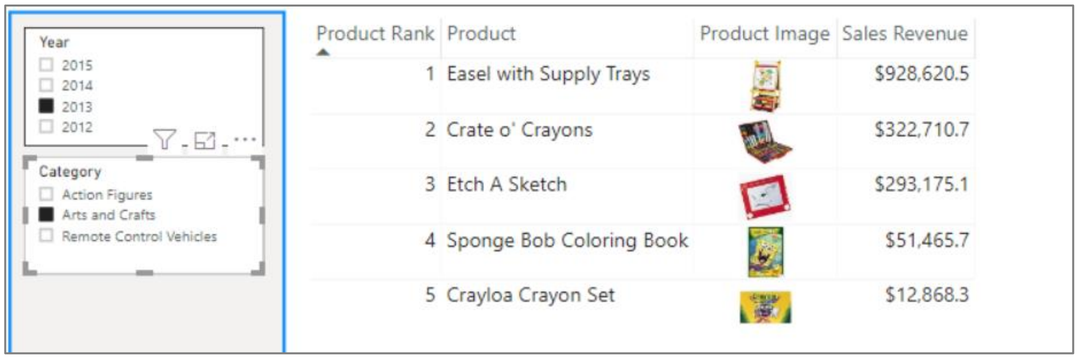


Product Image

Alignment
Center

h) At this point, you are done formatting the top 5 products table

14. Experiment with the report interaction to drill into a year and category when determining the top 5 selling products.



Year
☐ 2015
☐ 2014
■ 2013
☐ 2012

Category
☐ Action Figures
■ Arts and Crafts
☐ Remote Control Vehicles

Product Rank	Product	Product Image	Sales Revenue
1	Easel with Supply Trays		$928,620.5
2	Crate o' Crayons		$322,710.7
3	Etch A Sketch		$293,175.1
4	Sponge Bob Coloring Book		$51,465.7
5	Crayloa Crayon Set		$12,868.3

15. Save the work you have done by clicking the **Save** button in the upper left corner of the Power BI Desktop window.

Congratulations. You have now reached the end of this lab.