

Modeling Data with Hierarchies and Time Intelligence

Lab Time: 60 minutes

Lab Folder: C:\Student\Modules\05_TimeIntelligence\Lab

Lab Overview: In this lab, you will continue to work on the Power BI Desktop project you have been working with over the last two previous labs. At this point, you have already spent time designing the data model for the Wingtip Sales Analysis project. In this lab you will continue to extend the data model by adding dimensional hierarchies and a time dimension table named **Calendar**. You will learn how to configure the **Calendar** table so it can be used for time-based financial analysis. This will allow you to create measures using the DAX time intelligence functions.

Lab Dependencies: This lab assumes you have completed the earlier lab **Using the Data Modeling Features of Power BI Desktop** in which you created a Power BI Desktop project named **Wingtip Sales Analysis**. If you would like to begin work on this lab without completing the earlier lab, copy the lab solution file named **Wingtip Sales Analysis.pbix** which is located in the student folder at **C:\Student\Modules\04_DataModeling\Lab\Solution** into the folder at **C:\Student\Projects** using the Windows Explorer.

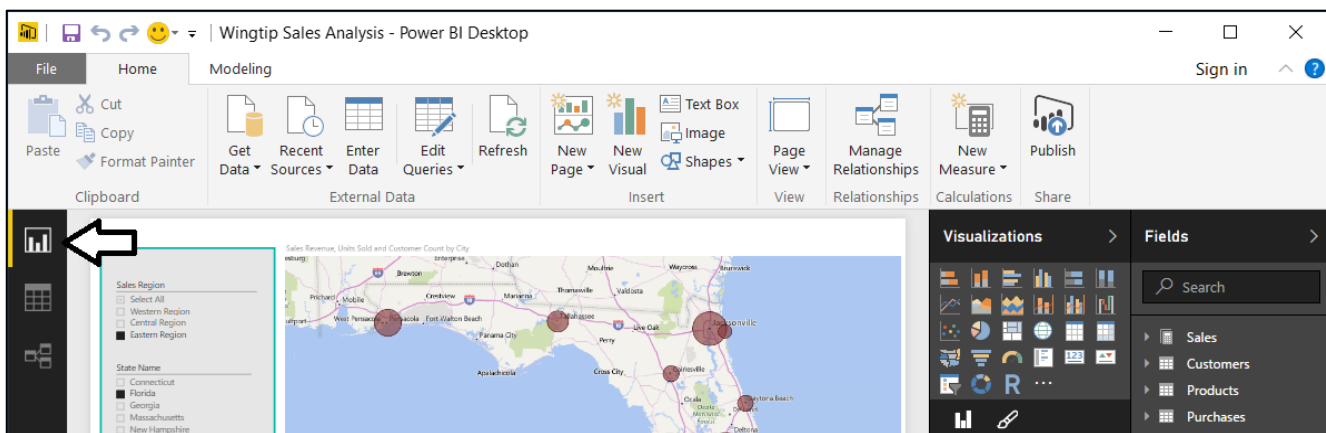
Exercise 1: Create a Dimensional Hierarchy for Product Category

In this exercise you will modify the **Products** table to add a new dimension hierarchy named **Product Category**. After that you will create a new report page and a few supporting measures to calculate percentages that each product category, subcategory and product contribute to overall sales revenue.

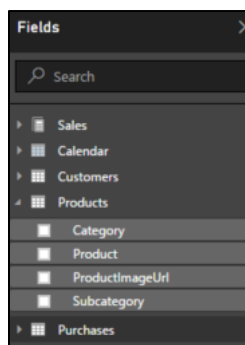
1. Launch Power BI Desktop.
2. Open the Power BI Desktop project named **Wingtip Sales Analytics.pbix** from the previous lab located at the following path.

C:\Student\Projects\wingtip Sales Analysis.pbix

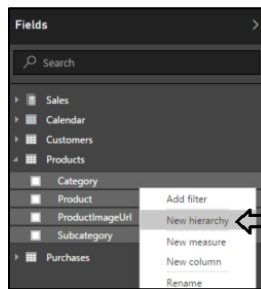
3. Navigate to Report view and inspect the tables in the **Fields** list on the right hand side of the Power BI Desktop window.



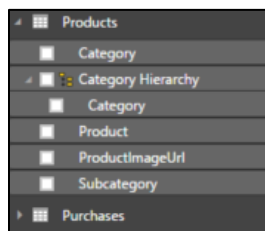
4. Add a new dimensional hierarchy to the **Products** table.
 - a) Inspect the **Products** table in the fields list. It should appear as the one shown in the following screenshot.



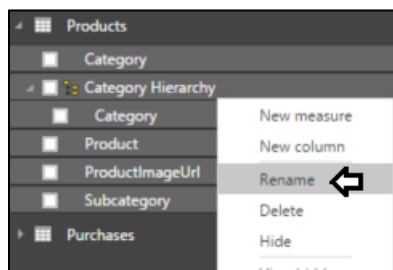
- b) Right-click on the **Category** field and then select the **New Hierarchy** menu command.



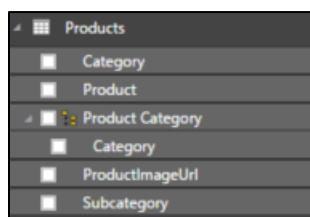
- c) You should now see a new dimensional hierarchy in the fields list named **Category Hierarchy**.



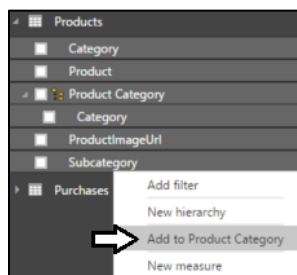
- d) Right-click **Category Hierarchy** and select the **Rename** menu command.



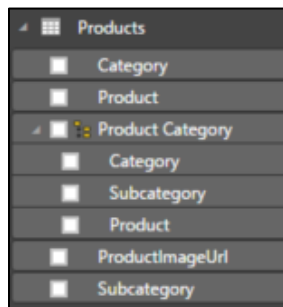
- e) Rename the new hierarchy **Product Category**.



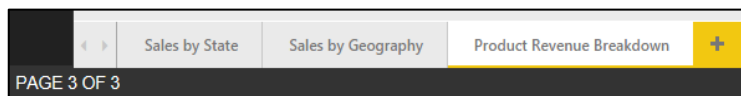
- f) Right-click on the **Subcategory** field and select the **Add to Product Category** menu command.



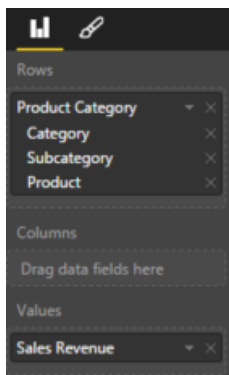
- g) Right-click on the **Product** field and select the **Add to Product Category** menu command.
- h) The **Product Category** hierarchy should now contain three fields as shown in the following screenshot.



- 5. Create a new report page and rename it to **Product Revenue Breakdown**.

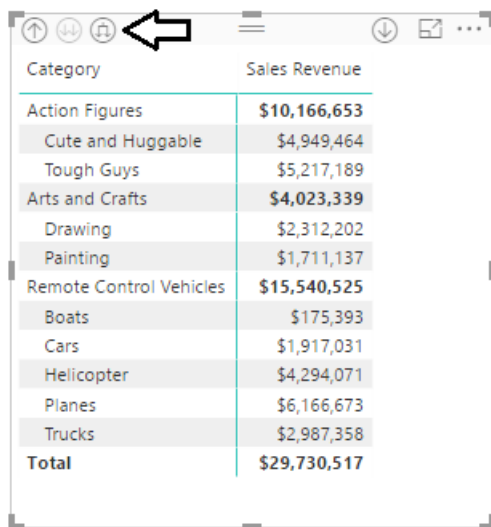


- 6. Create a new matrix visual to display sales revenue broken out by product category, subcategory and product.
 - a) Click the **New Visual** button on the ribbon to add a new visual to the page.
 - b) Change the visual to a matrix by clicking the **Matrix** button in the **Visualizations** list.
 - c) In the **Fields** list, click the checkbox for the **Product Category**.
 - d) Drag and drop the **Sales Revenue** field from the **Sales** table into the **Values** well.

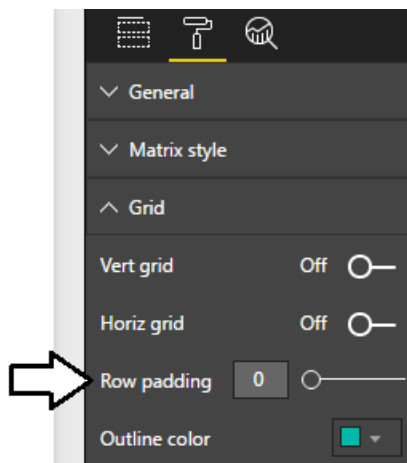


- e) The Matrix visual should now display sales revenue for each product along with totals for each subcategory and category.

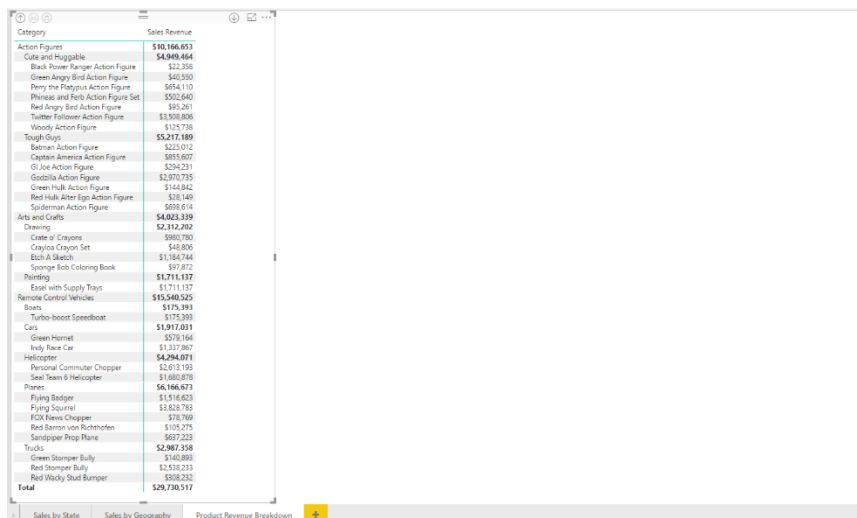
| Category | Sales Revenue |
|-------------------------|---------------------|
| Action Figures | \$10,166,653 |
| Arts and Crafts | \$4,023,339 |
| Remote Control Vehicles | \$15,540,525 |
| Total | \$29,730,517 |



| Category | Sales Revenue |
|-------------------------|---------------------|
| Action Figures | \$10,166,653 |
| Cute and Huggable | \$4,949,464 |
| Tough Guys | \$5,217,189 |
| Arts and Crafts | \$4,023,339 |
| Drawing | \$2,312,202 |
| Painting | \$1,711,137 |
| Remote Control Vehicles | \$15,540,525 |
| Boats | \$175,393 |
| Cars | \$1,917,031 |
| Helicopter | \$4,294,071 |
| Planes | \$6,166,673 |
| Trucks | \$2,987,358 |
| Total | \$29,730,517 |

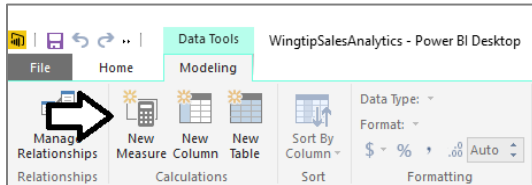


- f) Reposition the visual so it takes up the height and the width of the entire report page. You need extra width for this visual because you will be adding more columns to it later in this exercise.



| Category | Sales Revenue |
|------------------------------------|---------------------|
| Action Figures | \$10,166,653 |
| Cute and Huggable | \$4,949,464 |
| Black Power Ranger Action Figure | \$12,150 |
| Green Angry Bird Action Figure | \$40,550 |
| Perry the Platypus Action Figure | \$554,110 |
| Phineas and Ferb Action Figure Set | \$502,640 |
| Red Angry Bird Action Figure | \$95,261 |
| Twister Follow-up Action Figure | \$1,500,800 |
| Woody Action Figure | \$125,738 |
| Tough Guys | \$5,217,189 |
| Batman Action Figure | \$22,912 |
| Captain America Action Figure | \$855,807 |
| GI Joe Action Figure | \$254,231 |
| Godzilla Action Figure | \$2,970,735 |
| Green Hulk Action Figure | \$144,042 |
| Red Hulk Alter Ego Action Figure | \$30,140 |
| Spiderman Action Figure | \$698,614 |
| Arts and Crafts | \$4,023,339 |
| Drawing | \$2,312,202 |
| Crate of Crayons | \$990,780 |
| Crayola Crayon Set | \$48,806 |
| Eco-A Sketch | \$1,184,744 |
| Sponge Bob Coloring Book | \$97,872 |
| Painting | \$1,711,137 |
| Easel with Supply Trays | \$1,711,137 |
| Remote Control Vehicles | \$15,540,525 |
| Boats | \$175,393 |
| Turbo-boost Speedboat | \$175,393 |
| Cars | \$1,917,031 |
| Green Hornet | \$579,164 |
| Indy Race Car | \$1,337,867 |
| Helicopter | \$4,294,071 |
| Personal Commuter Chopper | \$2,613,193 |
| Stunt Team B Helicopter | \$1,680,878 |
| Planes | \$6,166,673 |
| Flying Badger | \$1,516,623 |
| Flying Squirrel | \$3,652,763 |
| FOX News Chopper | \$78,769 |
| Red Baron von Richtenhofen | \$105,275 |
| Sawdipper Prop Plane | \$607,235 |
| Trucks | \$2,987,358 |
| Green Stomper Bully | \$140,893 |
| Red Stomper Bully | \$1,516,233 |
| Red Wacky Stud Bumper | \$336,232 |
| Total | \$29,730,517 |

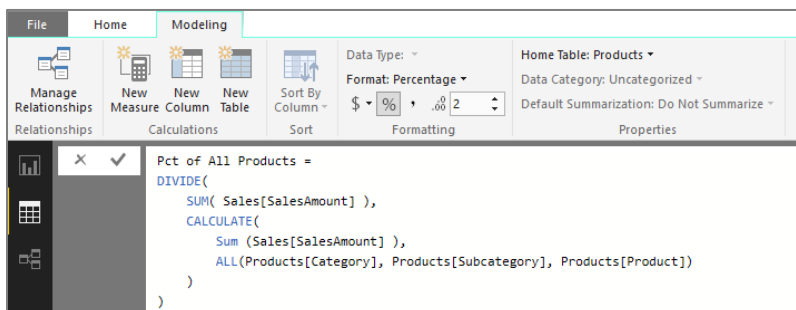
7. Create the **Pct of All Products** measure that calculates the percentage of sales revenue compared to that of all sales revenue.
- Navigate to data view.
 - Select the **Products** table from the **Fields** list.
 - Create a new measure by clicking the **New Measure** button in the ribbon.



- Enter the following DAX expression into the formula bar to create the measure named **Pct of All Products**.

```
Pct of All Products =  
DIVIDE(  
    SUM( Sales[SalesAmount] ),  
    CALCULATE(  
        Sum ( Sales[SalesAmount] ),  
        ALL( Products[Category], Products[Subcategory], Products[Product] )  
    )  
)
```

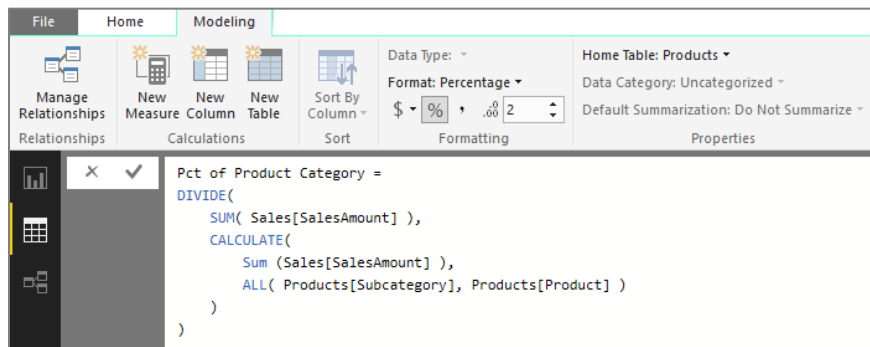
- Press the **ENTER** key to add the measure to the data model.
- Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Percentage**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to **2**.



8. Create the **Pct of Product Category** measure that calculates the percentage of sales revenue within the current product category.
- Select the **Products** table from the **Fields** list.
 - Create a new measure by clicking the **New Measure** button in the ribbon.
 - Enter the following DAX expression into the formula bar to create the measure named **Pct of Product Category**.

```
Pct of Product Category =  
DIVIDE(  
    SUM( Sales[SalesAmount] ),  
    CALCULATE(  
        Sum ( Sales[SalesAmount] ),  
        ALL( Products[Subcategory], Products[Product] )  
    )  
)
```

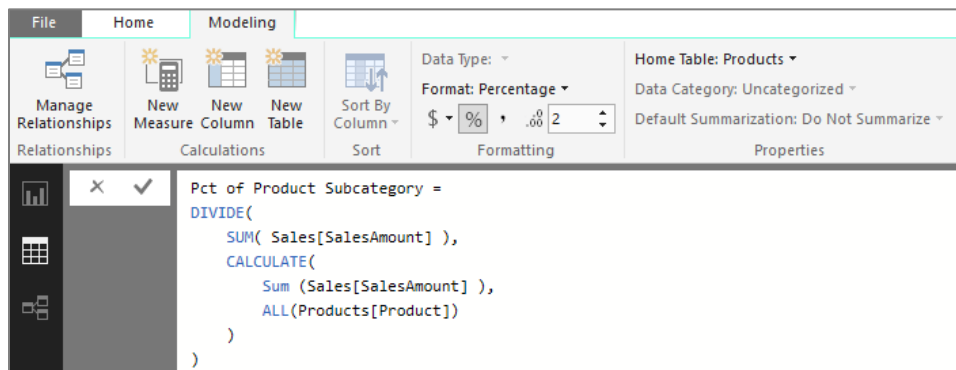
- Press the **ENTER** key to add the measure to the data model.
- Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Percentage**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to **2**.



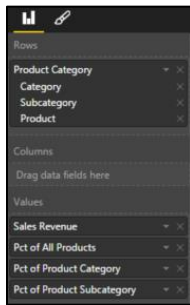
9. Create the **Pct of Product Subcategory** measure that calculates the percentage of sales revenue within the current subcategory.
 - a) Select the **Products** table from the **Fields** list.
 - b) Create a new measure by clicking the **New Measure** button in the ribbon.
 - c) Enter the following DAX expression into the formula bar to create the measure named **Pct of Product Subcategory**.

```
Pct of Product Subcategory =
DIVIDE(
    SUM( Sales[SalesAmount] ),
    CALCULATE(
        Sum ( Sales[SalesAmount] ),
        ALL( Products[Product] )
    )
)
```

- d) Press the **ENTER** key to add the measure to the data model.
- e) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Percentage**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to **2**.



10. Modify the matrix visual on the **Sales Revenue Breakdown** report page to include the three new measures.
 - a) Click the report icon on the top of the sidebar to enter Report view mode.
 - b) Make sure that the active report page is the **Sales Revenue Breakdown** report page.
 - c) Select the matrix visual that you created earlier in this exercise.
 - d) Drag and drop the **Pct of All Products** measure from the **Products** table into the **Values** well.
 - e) Drag and drop the **Pct of Product Category** measure from the **Products** table into the **Values** well.
 - f) Drag and drop the **Pct of Product Subcategory** measure from the **Products** table into the **Values** well.



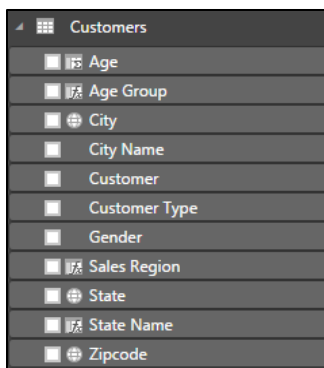
- g) Your visual should now match the one shown in the following screenshot. If you examine the values of the **Pct of Product Subcategory** measure, you should see that the products in a subcategory have percentages that sum to 100% for that subcategory. If you examine the values of the **Pct of Product Category** measure, you should see that the products in a category have percentages that sum to 100% for that category.

| Category | Sales Revenue | Pct of All Products | Pct of Product Category | Pct of Product Subcategory |
|------------------------------------|---------------|---------------------|-------------------------|----------------------------|
| Action Figures | \$10,166,653 | 34.20 % | 100.00 % | 100.00 % |
| Cole and Huggable | \$4,948,464 | 16.65 % | 48.68 % | 100.00 % |
| Black Power Ranger Action Figure | \$22,358 | 0.08 % | 0.22 % | 0.45 % |
| Green Angry Bird Action Figure | \$40,550 | 0.14 % | 0.40 % | 0.82 % |
| Perry the Platypus Action Figure | \$654,110 | 2.20 % | 6.43 % | 13.22 % |
| Phineas and Ferb Action Figure Set | \$502,640 | 1.69 % | 4.94 % | 10.16 % |
| Red Angry Bird Action Figure | \$95,261 | 0.32 % | 0.94 % | 1.92 % |
| Tweeter Followee Action Figure | \$1,500,808 | 11.80 % | 34.51 % | 70.89 % |
| Woody Action Figure | \$125,738 | 0.42 % | 1.24 % | 2.54 % |
| Tough Guys | \$2,217,189 | 17.55 % | 51.32 % | 100.00 % |
| Batman Action Figure | \$225,012 | 0.76 % | 2.21 % | 4.31 % |
| Captain America Action Figure | \$855,607 | 2.88 % | 8.42 % | 16.40 % |
| Q! Joe Action Figure | \$284,231 | 0.99 % | 2.89 % | 5.84 % |
| Godzilla Action Figure | \$2,970,735 | 9.99 % | 29.22 % | 56.94 % |
| Green Hulk Action Figure | \$144,842 | 0.49 % | 1.42 % | 2.78 % |
| Red Hulk Action Figure | \$26,148 | 0.09 % | 0.28 % | 0.54 % |
| Spiderman Action Figure | \$698,614 | 2.35 % | 6.87 % | 13.39 % |
| Arts and Crafts | \$4,603,398 | 13.53 % | 100.00 % | 100.00 % |
| Drawing | \$2,312,262 | 7.18 % | 57.47 % | 100.00 % |
| Crate of Crayons | \$980,780 | 3.30 % | 24.38 % | 42.42 % |
| Crayon Crayon Set | \$48,806 | 0.16 % | 1.21 % | 2.11 % |
| Elch A Sketch | \$1,184,744 | 3.98 % | 29.45 % | 51.24 % |
| Sponge Bob Coloring Book | \$97,872 | 0.33 % | 2.43 % | 4.23 % |
| Painting | \$1,711,157 | 5.76 % | 42.53 % | 100.00 % |
| Easel with Supply Trays | \$1,711,137 | 5.76 % | 42.53 % | 100.00 % |
| Remote Control Vehicles | \$15,540,525 | 52.27 % | 100.00 % | 100.00 % |
| Boats | \$175,393 | 0.59 % | 1.13 % | 100.00 % |
| Turbo-boost Speedboat | \$175,393 | 0.59 % | 1.13 % | 100.00 % |
| Cars | \$1,917,031 | 6.45 % | 12.34 % | 100.00 % |
| Green Hornet | \$575,164 | 1.85 % | 3.73 % | 30.21 % |
| Indy Race Car | \$1,337,867 | 4.50 % | 8.61 % | 69.79 % |
| Helicopter | \$4,284,071 | 14.44 % | 27.63 % | 100.00 % |
| Personal Commuter Chopper | \$2,613,193 | 8.79 % | 16.82 % | 60.86 % |
| See Team 6 Helicopter | \$1,660,878 | 5.65 % | 10.82 % | 39.14 % |
| Planes | \$4,166,473 | 20.74 % | 39.68 % | 100.00 % |
| Flying Squirrel | \$1,516,623 | 5.10 % | 9.76 % | 24.59 % |
| Flying Squirrel | \$1,620,783 | 12.88 % | 24.64 % | 62.09 % |
| FCR Team Chopper | \$75,766 | 0.26 % | 0.51 % | 1.28 % |
| Red Baron von Richtenhofen | \$105,275 | 0.35 % | 0.68 % | 1.71 % |
| Sandpaper Prop Plane | \$807,223 | 2.14 % | 4.10 % | 10.33 % |
| Trucks | \$2,967,354 | 10.05 % | 19.22 % | 100.00 % |
| Green Stomper Bully | \$140,893 | 0.47 % | 0.91 % | 4.72 % |
| Red Stomper Bully | \$2,530,233 | 8.54 % | 16.33 % | 84.97 % |
| Red Wacky Stool Bumper | \$308,232 | 1.04 % | 1.08 % | 10.32 % |
| Total | \$29,730,517 | 100.00 % | 100.00 % | 100.00 % |

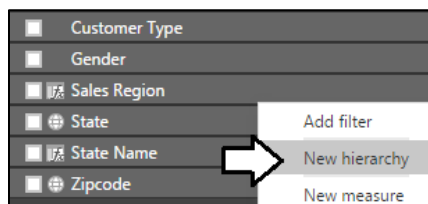
Exercise 2: Create a Dimensional Hierarchy for Customer Geography

In this exercise you will modify the **Customers** table by adding a new dimension hierarchy named **Customer Geography**.

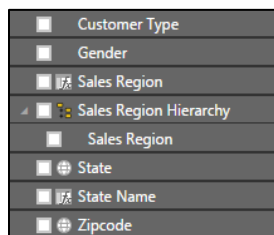
1. Add a new dimensional hierarchy to the **Customers** table.
 - a) If you are not already in Report view, use the left navigation to switch to Report view.
 - b) Inspect the **Customers** table in the fields list. It should appear as the one shown in the following screenshot.



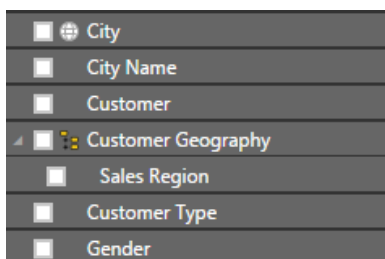
- c) Right-click on the **Sales Region** field and then select the **New Hierarchy** menu command.



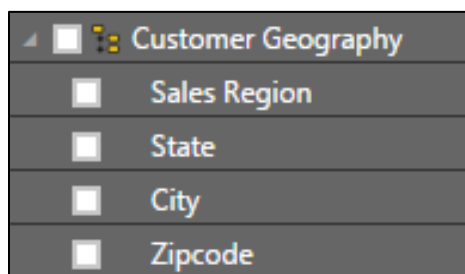
- d) You should now see a new dimensional hierarchy in the fields list named **Sales Region Hierarchy**.



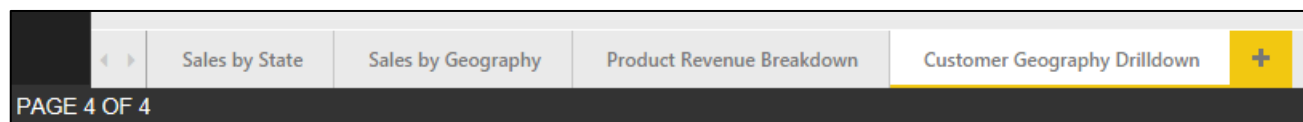
- e) Right-click **Sales Region Hierarchy**, select the **Rename** menu command and rename the hierarchy **Customer Geography**.



- f) Right-click on the **State** field and select the **Add to Customer Geography** menu command.
g) Right-click on the **City** field and select the **Add to Customer Geography** menu command.
h) Right-click on the **Zipcode** field and select the **Add to Customer Geography** menu command.
i) The **Customer Geography** hierarchy should now contain four fields as shown in the following screenshot.

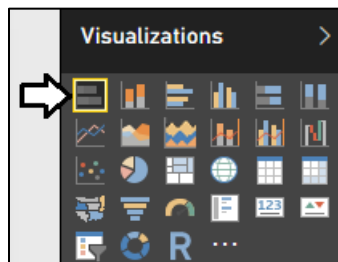


2. Create a new report page and rename it to **Customer Geography Drilldown**.

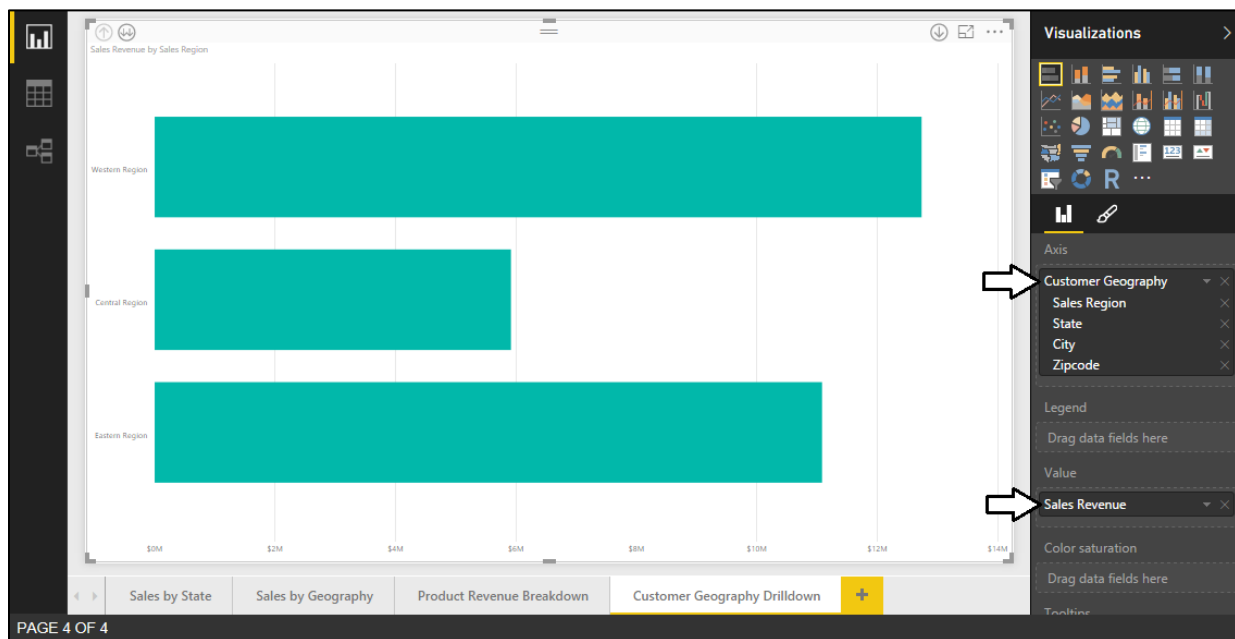


3. Add a new stacked bar chart visual.

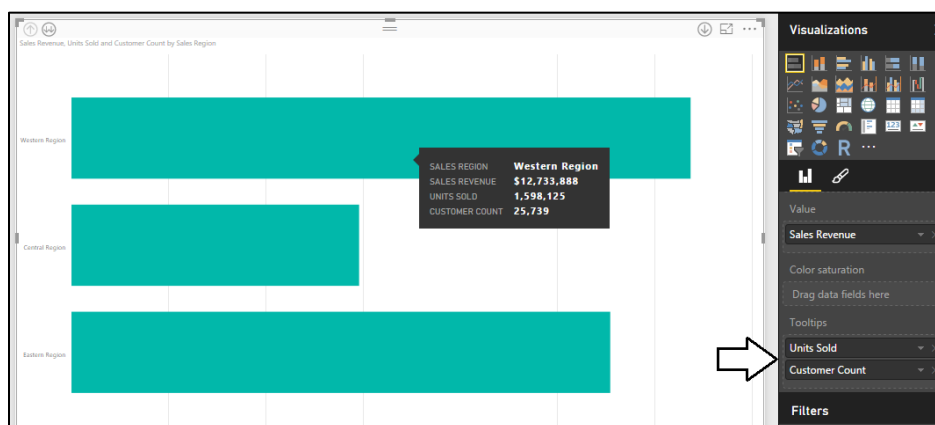
- a) Click the **Stacked Bar Chart** button on the **Visualizations** list to create a new bar chart visual.



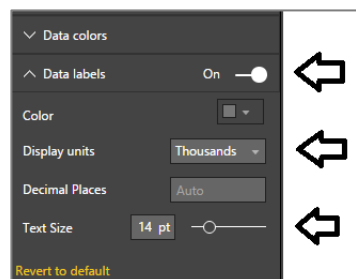
- b) Resize the bar chart visual so it takes up the entire height and width of the report page.
- c) Drag and drop the **Customer Geography** hierarchy from the **Customers** table into the **Axis** well.
- d) Drag and drop the **Sales Revenue** field from **Sales** table into the **Value** well.
- e) The bar chart should now appear like the one shown in the following screenshot.



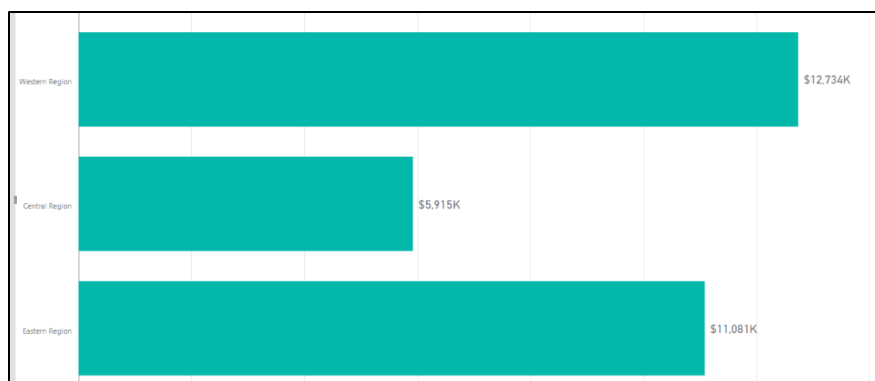
4. Configure **Tooltips** for the bar chart to additionally show **Unit Sold** and **Customer Count**.
 - a) Drag and drop the **Units Sold** field from the **Sales** table into the **Tooltips** well.
 - b) Drag and drop the **Customer Count** field from the **Sales** table into the **Tooltips** well.
 - c) Hover over a bar with the mouse to observe the effects of the new tooltip configuration.



5. 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**.
 - e) Set the Text Size to **14 pt**.



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



6. 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 arrow.



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

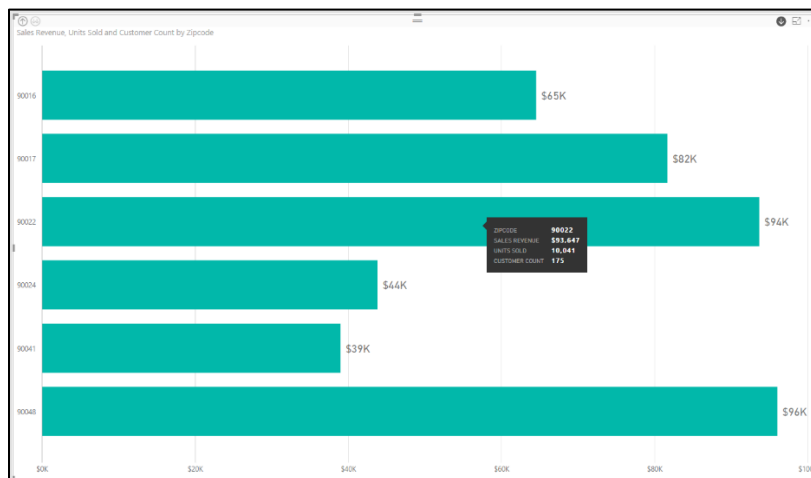
- a) Click on the bar for the **Western Region** to drill down into the states for that sales region.



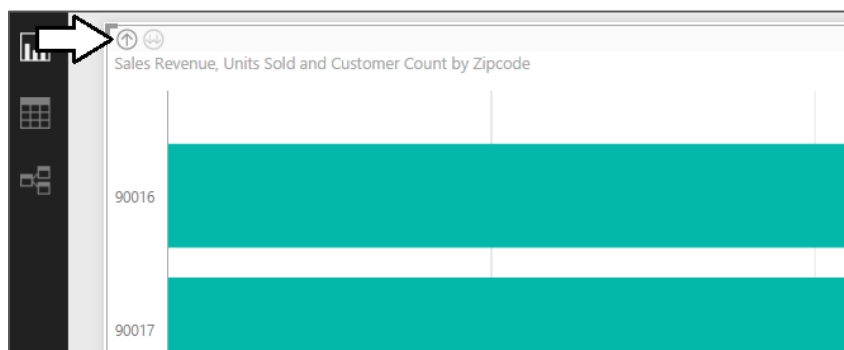
- b) Click on the bar for the **CA** to drill down into the cities in California.



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



d) In the top left corner is a **Drill Up** button. Click on this button three times to return to the top level of the hierarchy.

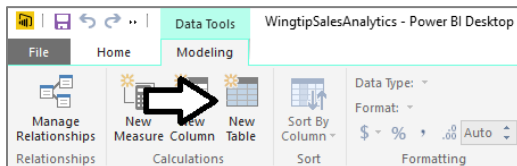


You have now seen how you can set up a visual to drill into and out of a dimensional hierarchy.

Exercise 3: Extend the Data Model by Adding a Calendar Dimension 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 advantage of the time intelligence support that is built into Power Pivot and DAX.

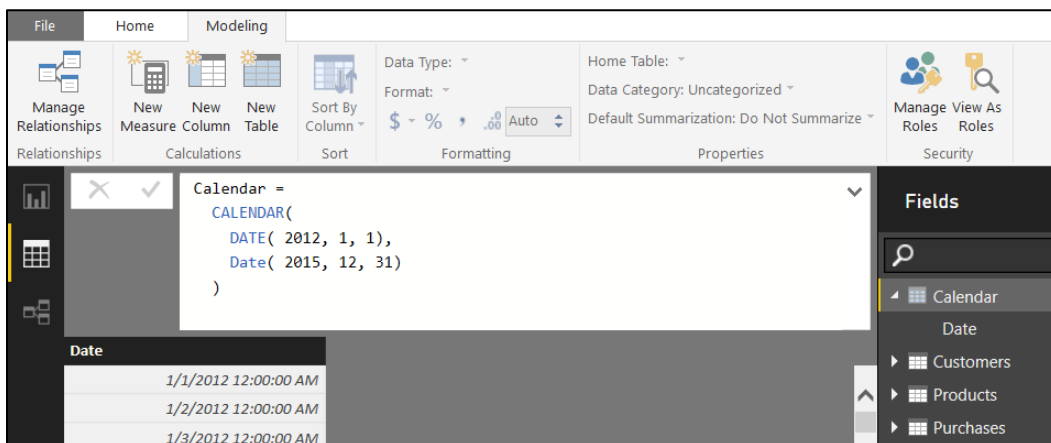
1. Create a new calculated table named **Calendar**.
 - a) Navigate to the **Modeling** tab in the ribbon.
 - b) Click the **New Table** button in the ribbon.



- c) Type the following DAX formula into the formula bar and press the **Enter** key to create the **Calendar** table.

```
Calendar =
CALENDAR(
    DATE( 2012, 1, 1),
    Date( 2015, 12, 31)
)
```

- d) You should be able to verify that the **Calendar** table has been created with a single column named **Date**. You should also be able to verify that there is one row in the **Calendar** table for each day in the calendar years of 2012, 2013, 2014 and 2015.



- e) Modify the DAX expression for this calculated table so you do not have to hardcode literal values for the starting year or the ending year. First, replace the literal value for the starting year 2012 with the following DAX expression.

```
YEAR( MIN(Sales[PurchaseDate]) )
```

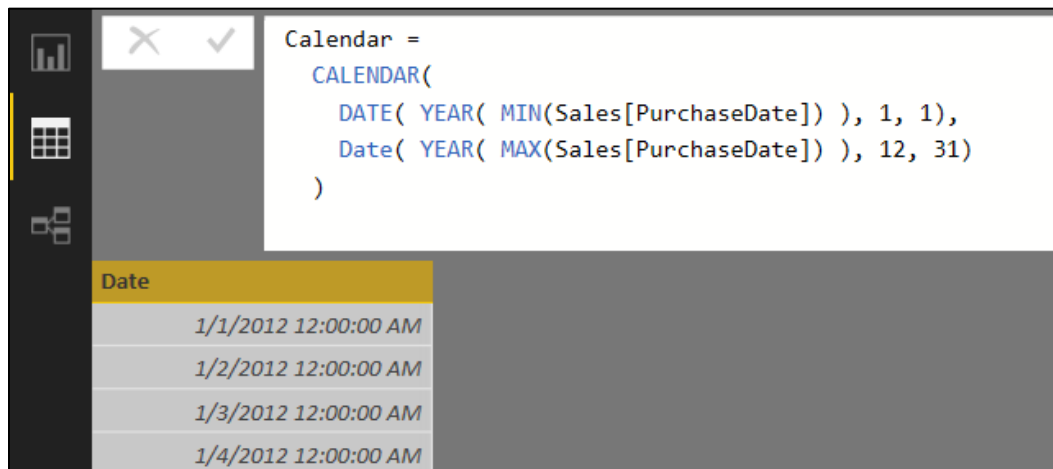
- f) Next, replace the literal value for the ending year 2015 with the following DAX expressions.

```
YEAR( MAX(Sales[PurchaseDate]) )
```

- g) At this point, your DAX expression to create the **Calendar** table should now match the following code listing.

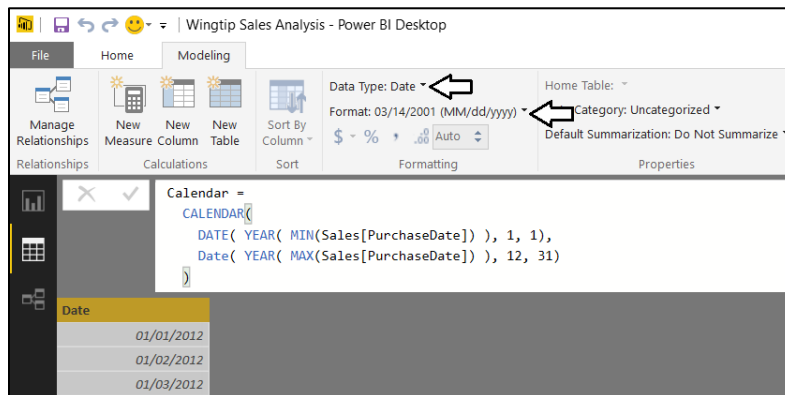
```
Calendar =
CALENDAR(
    DATE( YEAR( MIN(Sales[PurchaseDate]) ), 1, 1),
    Date( YEAR( MAX(Sales[PurchaseDate]) ), 12, 31)
)
```

- h) The **Calendar** table should continue to look as it did before starting at **1/1/2012** and running to **12/31/2015**.

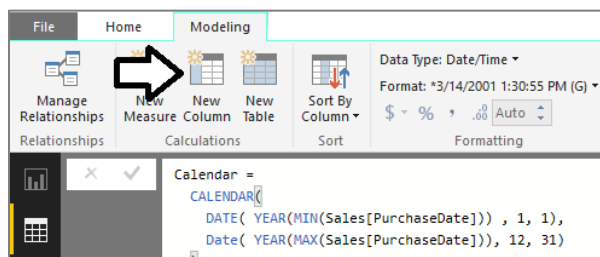


The modification to the DAX expression to calculate the start date and end date dynamically provides flexibility. The new expression will now automatically add new rows for complete years if the **Sales** table is ever updated with purchases that occurred either before the start of 2012 or after the end of 2015.

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)**.



3. Add a new column to the **Calendar** table named **Year** which displays the financial year.
 - a) Create a new calculated column by clicking the **New Column** button in the ribbon.



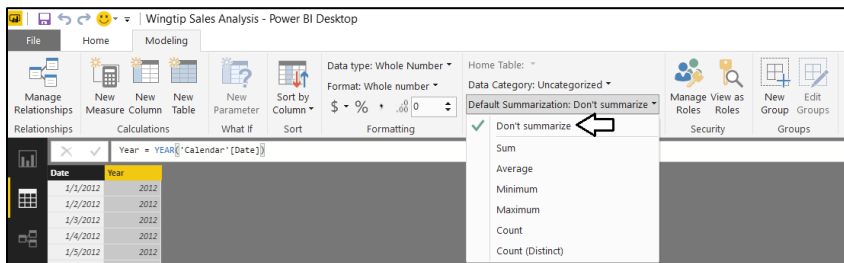
- b) Type in the following DAX expression and press the Enter key.

Year = YEAR('Calendar'[Date])

- c) You should see that the **Calendar** table now contains a **Year** column displaying the year.

| Date | Year |
|----------|------|
| 1/1/2012 | 2012 |
| 1/2/2012 | 2012 |
| 1/3/2012 | 2012 |

- d) Modify the **Default Summarization** property of the **Year** column to **Don't Summarize**.



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.

4. Add a new column to the **Calendar** table named **Quarter** which display the financial year and quarter.
- Create a new calculated column by clicking the **New Column** button in the ribbon.
 - Type in the following DAX expression and press the Enter key.

Quarter = YEAR('Calendar'[Date]) & "-Q" & FORMAT('Calendar'[Date], "q")

- c) The **Quarter** column now displays a value with the year and quarter which can be used in visuals and reports.

| Date | Year | Quarter |
|------------|------|---------|
| 01/01/2012 | 2012 | 2012-Q1 |
| 01/02/2012 | 2012 | 2012-Q1 |
| 01/03/2012 | 2012 | 2012-Q1 |
| 01/04/2012 | 2012 | 2012-Q1 |
| 01/05/2012 | 2012 | 2012-Q1 |

5. Add a new column to the **Calendar** table named **Month** which displays the financial year and month.
- Create a new calculated column by clicking the **New Column** button in the ribbon.
 - Type in the following DAX expression and press the Enter key.

Month = FORMAT('Calendar'[Date], "MMM yyyy")

- c) The column should now display a month for each date as shown in the following screenshot.

| Date | Year | Quarter | Month |
|----------|------|---------|----------|
| 1/1/2012 | 2012 | 2012-Q1 | Jan 2012 |
| 1/2/2012 | 2012 | 2012-Q1 | Jan 2012 |
| 1/3/2012 | 2012 | 2012-Q1 | Jan 2012 |

The **Month** column is a good example of a column whose value will not automatically be sorted in the 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 create an addition column named **MonthSort** whose sole purpose will be to provide assistance to the **Month** column to sort its values chronologically.

6. Add a new sort column to the **Calendar** table named **MonthSort** to control the sort order of the **Month** column.

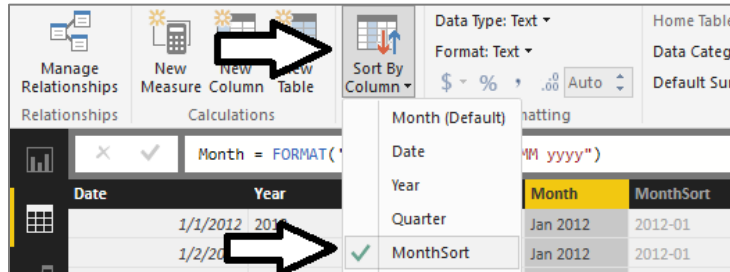
- Create a new calculated column by clicking the **New Column** button in the ribbon.
- Type in the following DAX expression and press the Enter key.

```
MonthSort = FORMAT('Calendar'[Date], "yyyy-MM")
```

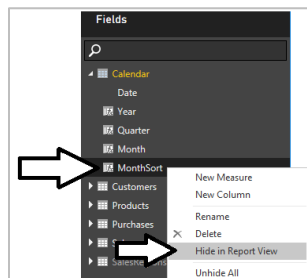
- You should be able to see that the **MonthSort** column produces a text value for each date in the format of **2012-01**. The key aspect of this format is that **MonthSort** values are sorted chronologically when they are sorted alphabetically.

| Date | Year | Quarter | Month | MonthSort |
|----------|------|---------|----------|-----------|
| 1/1/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 |
| 1/2/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 |
| 1/3/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 |

- Configure the **Month** column to use the **MonthSort** column as its sort column. Accomplish this by clicking the column header of the **Month** column to select it and then by dropping down the **Sort By Column** menu button in the ribbon and selecting the **MonthSort** column.



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



7. Add a new column to the **Calendar** table named **Month in Year** to display the financial month without the year.

- Create a new calculated column by clicking the **New Column** button in the ribbon.
- Type in the following DAX expression and press the Enter key.

```
Month in Year = FORMAT('Calendar'[Date], "MMM")
```


- c) The **Month in Year** column should now display the abbreviated month name for each date.

Month in Year = `FORMAT('Calendar'[Date], "MMM")`

| Date | Year | Quarter | Month | MonthSort | Month in Year |
|------------|------|---------|----------|-----------|---------------|
| 01/01/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan |
| 01/02/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan |
| 01/03/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan |
| 01/04/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan |
| 01/05/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan |
| 01/06/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan |

Just like the **Month** column, the **Month in Year** column will need the assistance of a sort column.

8. Add a new sort column to the **Calendar** table named **MonthInYearSort** to control the sort order of the **Month in Year** column.
- Create a new calculated column by clicking the **New Column** button in the ribbon.
 - Type in the following DAX expression and press the Enter key.

MonthInYearSort = `MONTH('Calendar'[Date])`

- c) As you can see, the **MonthInYearSort** column displays an integer value between 1 and 12 to indicate the month.

MonthInYearSort = `MONTH('Calendar'[Date])`

| Date | Year | Quarter | Month | MonthSort | Month in Year | MonthInYearSort |
|----------|------|---------|----------|-----------|---------------|-----------------|
| 1/1/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | 1 |
| 1/2/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | 1 |
| 1/3/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | 1 |

- d) Configure the **Month in Year** column to use the **MonthInYearSort** column as its sort column. Accomplish this by clicking the column header of the **Month in Year** column to select it and then by dropping down the **Sort By Column** menu button in the ribbon and selecting the **MonthInYearSort** column.

The screenshot shows the Power BI ribbon with the 'Modeling' tab selected. The 'Sort By Column' menu is open, and the 'MonthInYearSort' column is selected. The 'Month in Year' column is highlighted in the table below.

| Date | Year | Quarter | Month | MonthSort | Month in Year | MonthInYearSort |
|----------|------|---------|----------|-----------|---------------|-----------------|
| 1/1/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | 1 |
| 1/2/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | 1 |
| 1/3/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | 1 |
| 1/4/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | 1 |
| 1/5/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | 1 |

- Hide the **MonthInYearSort** column by right-clicking it in the **Fields** list and selecting the **Hide in Report View** command.
9. Add a new column to the **Calendar** table named **Day of Week** which display the day of the week.
- Create a new calculated column by clicking the **New Column** button in the ribbon.
 - Type in the following DAX expression and press the Enter key.

Day of Week = `FORMAT('Calendar'[Date], "ddd")`

- c) The **Day of Week** column should now display the name of the day (e.g. Monday) for each date.

| Day of Week = FORMAT('Calendar'[Date], "ddd") | | | | | | | |
|-----------------------------------------------|------|---------|----------|-----------|---------------|-----------------|-------------|
| Date | Year | Quarter | Month | MonthSort | Month in Year | MonthInYearSort | Day of Week |
| 01/01/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan | | Sun |
| 01/02/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan | | Mon |
| 01/03/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan | | Tue |
| 01/04/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan | | Wed |
| 01/05/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan | | Thu |
| 01/06/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan | | Fri |
| 01/07/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | Jan | | Sat |

10. Add a new sort column to the **Calendar** table named **DayOfWeekSort** to control the sort order of the **Day of Week** column.

- Create a new calculated column by clicking the **New Column** button in the ribbon.
- Type in the following DAX expression and press the Enter key.

```
DayOfWeekSort = WEEKDAY('Calendar'[Date], 2)
```

The second argument passes to **WEEKDAY** function determines the starting day for the week. If you pass a value of 1, the starting day for the week will be Sunday. In this case you have passed a value of 2 which will make Monday the first day of the week.

- Now the **DayOfWeekSort** column should return an integer value for each date indicating the day of the week. As you can see, each date that is a Monday has a value of 1.

| DayOfWeekSort = WEEKDAY('Calendar'[Date], 2) | | | | | | | | |
|----------------------------------------------|------|---------|----------|-----------|---------------|-----------------|-------------|---------------|
| Date | Year | Quarter | Month | MonthSort | Month in Year | MonthInYearSort | Day of Week | DayOfWeekSort |
| 1/1/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Sunday | 7 |
| 1/2/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Monday | 1 |
| 1/3/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Tuesday | 2 |
| 1/4/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Wednesday | 3 |
| 1/5/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Thursday | 4 |
| 1/6/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Friday | 5 |
| 1/7/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Saturday | 6 |

- Configure the **Day of Week** column to use the **DayInWeekSort** column as its sort column. Accomplish this by clicking the column header of the **Day of Week** column to select it and then by dropping down the **Sort By Column** menu button in the ribbon and selecting the **DayInWeekSort** column.

| Day of Week (Default) | | | | | | | | |
|-----------------------|------|---------|----------|-----------|---------------|-----------------|-------------|---------------|
| Date | Year | Quarter | Month | MonthSort | Month in Year | MonthInYearSort | Day of Week | DayOfWeekSort |
| 1/1/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Sunday | 7 |
| 1/2/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Monday | 1 |
| 1/3/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Tuesday | 2 |
| 1/4/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Wednesday | 3 |
| 1/5/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Thursday | 4 |
| 1/6/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Friday | 5 |
| 1/7/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Saturday | 6 |
| 1/8/2012 | 2012 | 2012-Q1 | Jan 2012 | 2012-01 | January | | Sunday | 7 |

- Hide the **DayInWeekSort** column by right-clicking it in the **Fields** list and selecting the **Hide in Report View** command.

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.

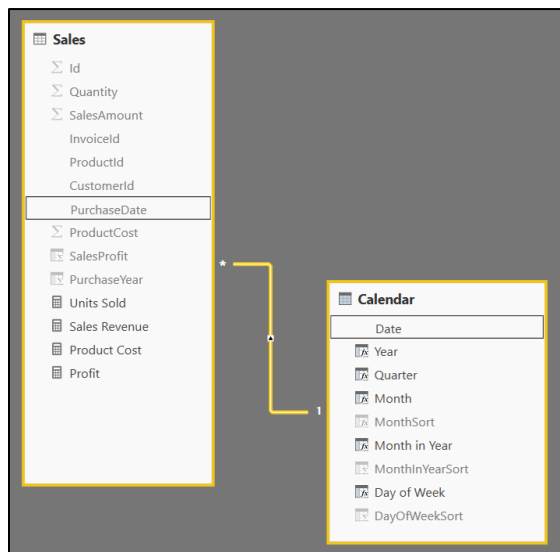
11. Create a relationship between the **Calendar** table and the **Sales** table.

- Navigate to relationship view. You should be able to see that the **Calendar** table is present. You should also be able to verify that the **Calendar** table does not have any existing relationships.

- b) Using the mouse, rearrange the tables in relationship 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.



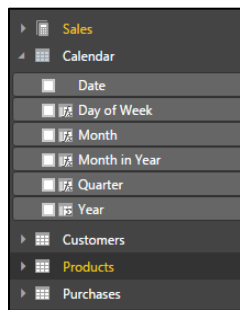
- c) Create a new relationship by performing a drag and drop operation with the mouse to drag the **PurchaseDate** column from the **Sales** table and dropping it on the **Date** column of the **Calendar** table.



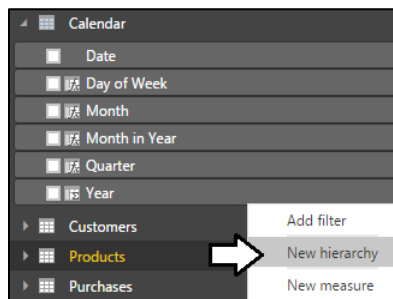
You have now completed the work of adding the **Calendar** table to the data model. Now, you will modify the **Calendar** table to add a new dimension hierarchy named **Calendar Drilldown**.

12. Add a new dimensional hierarchy to the **Calendar** table.

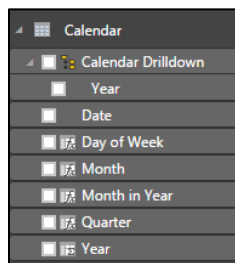
- Use the left navigation to switch to Report View.
- Inspect the **Calendar** table in the fields list. It should appear as the one shown in the following screenshot.



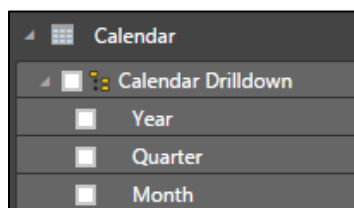
- c) Right-click on the **Year** field and then select the **New Hierarchy** menu command.



- d) You should now see a new dimensional hierarchy in the fields list named **Year Hierarchy**.
e) Right-click **Year Hierarchy** and select the **Rename** menu command.
f) Rename the new hierarchy **Calendar Drilldown**.



- g) Right-click on the **Quarter** field and select the **Add to Calendar Drilldown** menu command.
h) Right-click on the **Month** field and select the **Add to Calendar Drilldown** menu command.
i) The **Calendar Drilldown** hierarchy should now contain three fields as shown in the following screenshot.

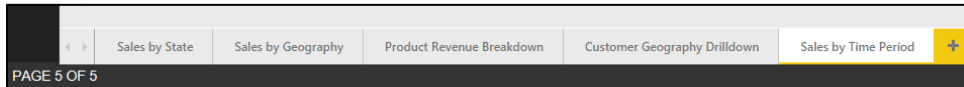


Now you have finished creating the **Calendar** table and you can use it to create reports and to call DAX time intelligence functions.

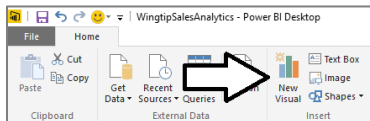
Exercise 4: Design Reports and Visuals using the Calendar Table

In this exercise, you will leverage the **Calendar** table that you created in the previous exercise by creating a few new visuals to display sales revenue totals aggregated over various time intervals.

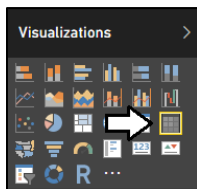
1. Create a new page in the project's report.
 - a) Navigate to report view.
 - b) Add a new page by clicking the (+) button on the right side of the page navigation menu.
 - c) Once the new page has been created, modify its title to **Sales by Time Period**.



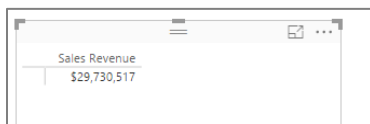
- d) Now that you have created a new page, you can now add a few new visuals.
2. Create a new matrix visual to show sales revenue for specific time periods.
 - a) Click the **New Visual** button on the ribbon to add a new visual to the page.



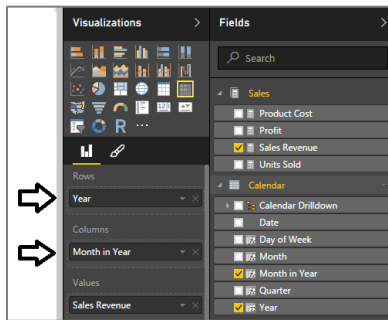
- b) Change the visual to a matrix by clicking the **Matrix** button in the **Visualizations** list.



- c) Select the checkbox next to the **Sales Revenue** measure in the **Fields** list. When you select the **Sales Revenue** measure, the report designer will add it to the **Values** well and the visual will show a single value for total sales revenue across the entire **Sales** table.



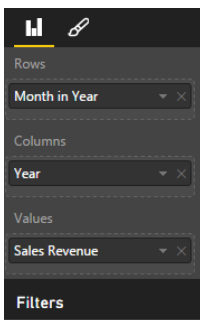
- d) Now it's time to extend the matrix by adding row labels and column labels. First, drag and drop the **Year** column from the **Calendar** table in the **Fields** list into the **Rows** well in the **Visualizations** pane.
- e) Now drag and drop the **Month in Year** column from the **Calendar** table in the **Fields** list into the **Columns** well in the **Visualizations** pane.



- 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 so you can see all the columns.

| Year | Jan | Feb | Mar | Apr | May | Jun | Jul | Aug | Sep | Oct | Nov | Dec | Total |
|-------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|--------------|
| 2012 | \$3,063 | \$33,218 | \$49,213 | \$40,434 | \$83,840 | \$136,670 | \$144,244 | \$197,952 | \$215,097 | \$239,513 | \$376,503 | \$424,240 | \$1,943,986 |
| 2013 | \$307,182 | \$291,942 | \$346,186 | \$380,869 | \$377,376 | \$353,586 | \$391,202 | \$476,884 | \$504,532 | \$577,439 | \$579,507 | \$769,473 | \$5,356,177 |
| 2014 | \$629,969 | \$609,637 | \$628,618 | \$661,588 | \$748,193 | \$814,333 | \$788,469 | \$869,143 | \$890,958 | \$988,789 | \$999,574 | \$1,644,980 | \$10,274,251 |
| 2015 | \$959,863 | \$969,330 | \$675,533 | \$722,456 | \$698,311 | \$785,793 | \$921,994 | \$1,084,189 | \$1,088,863 | \$1,211,810 | \$1,305,029 | \$1,732,932 | \$12,156,103 |
| Total | \$1,900,077 | \$1,904,126 | \$1,699,551 | \$1,805,347 | \$1,907,720 | \$2,090,382 | \$2,245,908 | \$2,628,168 | \$2,699,449 | \$3,017,551 | \$3,260,613 | \$4,571,625 | \$29,730,517 |

- 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 **Month in Year** field and the **Year** field have just switched places.

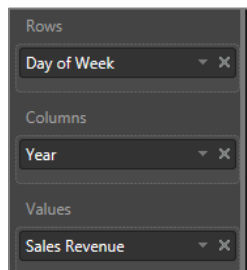


- i) The matrix should now display a row for each month and a column for each year.

| Month in Year | 2012 | 2013 | 2014 | 2015 | Total |
|---------------|-------------|-------------|--------------|--------------|--------------|
| Jan | \$3,063 | \$307,182 | \$629,969 | \$959,863 | \$1,900,077 |
| Feb | \$33,218 | \$291,942 | \$609,637 | \$969,330 | \$1,904,126 |
| Mar | \$49,213 | \$346,186 | \$628,618 | \$675,533 | \$1,699,551 |
| Apr | \$40,434 | \$380,869 | \$661,588 | \$722,456 | \$1,805,347 |
| May | \$83,840 | \$377,376 | \$748,193 | \$698,311 | \$1,907,720 |
| Jun | \$136,670 | \$353,586 | \$814,333 | \$785,793 | \$2,090,382 |
| Jul | \$144,244 | \$391,202 | \$788,469 | \$921,994 | \$2,245,908 |
| Aug | \$197,952 | \$476,884 | \$869,143 | \$1,084,189 | \$2,628,168 |
| Sep | \$215,097 | \$504,532 | \$890,958 | \$1,088,863 | \$2,699,449 |
| Oct | \$239,513 | \$577,439 | \$988,789 | \$1,211,810 | \$3,017,551 |
| Nov | \$376,503 | \$579,507 | \$999,574 | \$1,305,029 | \$3,260,613 |
| Dec | \$424,240 | \$769,473 | \$1,644,980 | \$1,732,932 | \$4,571,625 |
| Total | \$1,943,986 | \$5,356,177 | \$10,274,251 | \$12,156,103 | \$29,730,517 |

- Create a new matrix visual to show sales revenue for specific time periods.
 - Click the **New Visual** button on the ribbon to add a new visual to the page.
 - Make sure this new visual is positioned directly below the first visual that you created.
 - Change the visual to a matrix by clicking the Matrix button in the **Visualizations** list.
 - Drag and drop the **Day of Week** column from the **Calendar** table in the **Fields** list into the **Rows** well.

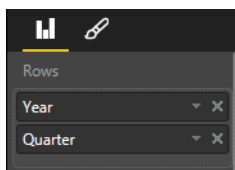
- e) Drag and drop the **Year** column from the **Calendar** table into the **Columns** well.
- f) Drag and drop the **Sales Revenue** measure from the **Calendar** table in the **Fields** list into the **Values** well.



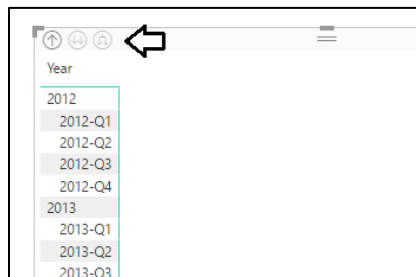
- g) The matrix should now display a row for each day of the week and a column for each year.

| 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 |

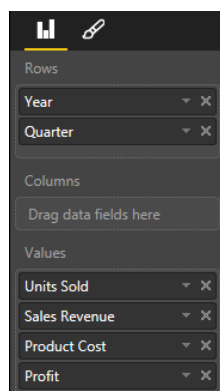
4. Add a third matrix visual to analyze sales data calculated at both the yearly level and the quarterly level.
 - a) Click the **New Visual** button on the ribbon to add a new visual to the page.
 - b) Change the visual to a matrix by clicking the **Matrix** button in the **Visualizations** list.
 - c) Drag and drop the **Year** column from the **Calendar** table into the **Rows** well.
 - d) Drag and drop the **Quarter** column from the **Calendar** table into the **Rows** well.



- e) Click the **Expand All** button so that the Matrix visual displays row labels for the **Year** column and the **Quarter** column.



- f) Drag and drop the **Units Sold** measure from the **Sales** table into the **Values** well.
- g) Drag and drop the **Sales Revenue** measure from the **Sales** table into the **Values** well.
- h) Drag and drop the **Product Cost** measure from the **Sales** table into the **Values** well.
- i) Drag and drop the **Profit** measure from the **Sales** table into the **Values** well.



- j) The matrix should now display values for each measure calculated at the quarterly level as well as at the yearly level.

| Year | Units Sold | Sales Revenue | Product Cost | Profit |
|--------------|------------------|---------------------|---------------------|---------------------|
| 2012 | 127,233 | \$1,943,986 | \$979,909 | \$964,077 |
| 2012-Q1 | 5,023 | \$85,494 | \$40,088 | \$45,406 |
| 2012-Q2 | 15,845 | \$260,944 | \$130,287 | \$130,657 |
| 2012-Q3 | 30,979 | \$557,293 | \$269,314 | \$287,979 |
| 2012-Q4 | 75,386 | \$1,040,256 | \$540,222 | \$500,034 |
| 2013 | 995,682 | \$5,356,177 | \$2,510,921 | \$2,845,256 |
| 2013-Q1 | 71,064 | \$945,310 | \$517,474 | \$427,836 |
| 2013-Q2 | 127,830 | \$1,111,831 | \$557,730 | \$554,101 |
| 2013-Q3 | 302,557 | \$1,372,617 | \$571,187 | \$801,430 |
| 2013-Q4 | 494,231 | \$1,926,420 | \$864,530 | \$1,061,889 |
| 2014 | 2,094,582 | \$10,274,251 | \$5,184,002 | \$5,090,249 |
| 2014-Q1 | 492,123 | \$1,868,225 | \$892,244 | \$975,981 |
| 2014-Q2 | 542,615 | \$2,224,114 | \$1,081,051 | \$1,143,063 |
| 2014-Q3 | 417,331 | \$2,548,569 | \$1,332,729 | \$1,215,840 |
| 2014-Q4 | 642,513 | \$3,633,343 | \$1,877,978 | \$1,755,365 |
| 2015 | 1,334,548 | \$12,156,103 | \$6,667,621 | \$5,488,482 |
| 2015-Q1 | 406,989 | \$2,604,726 | \$1,364,369 | \$1,240,357 |
| 2015-Q2 | 216,311 | \$2,206,560 | \$1,219,892 | \$986,669 |
| 2015-Q3 | 308,970 | \$3,095,046 | \$1,724,893 | \$1,370,153 |
| 2015-Q4 | 402,278 | \$4,249,771 | \$2,358,468 | \$1,891,304 |
| Total | 4,552,045 | \$29,730,517 | \$15,342,453 | \$14,388,064 |

- k) Using your mouse, arrange the new matrix visual on the right side of the page at the top to match the following screenshot.

| Month in Year | 2012 | 2013 | 2014 | 2015 | Total |
|---------------|--------------------|--------------------|---------------------|---------------------|---------------------|
| Jan | \$3,063 | \$307,182 | \$629,969 | \$959,863 | \$1,900,077 |
| Feb | \$33,218 | \$291,942 | \$609,637 | \$969,330 | \$1,904,126 |
| Mar | \$49,213 | \$346,186 | \$628,618 | \$675,533 | \$1,699,551 |
| Apr | \$40,434 | \$380,869 | \$661,588 | \$722,456 | \$1,805,347 |
| May | \$83,840 | \$377,376 | \$748,193 | \$698,311 | \$1,907,720 |
| Jun | \$136,670 | \$353,586 | \$814,333 | \$785,793 | \$2,090,382 |
| Jul | \$144,244 | \$391,202 | \$788,469 | \$921,994 | \$2,245,908 |
| Aug | \$197,952 | \$476,884 | \$869,143 | \$1,084,189 | \$2,628,168 |
| Sep | \$215,097 | \$504,532 | \$890,958 | \$1,088,863 | \$2,699,449 |
| Oct | \$239,513 | \$577,439 | \$988,789 | \$1,211,810 | \$3,017,551 |
| Nov | \$376,503 | \$579,507 | \$999,574 | \$1,305,029 | \$3,260,613 |
| Dec | \$428,240 | \$769,473 | \$1,644,980 | \$1,732,932 | \$4,571,625 |
| Total | \$1,943,986 | \$5,356,177 | \$10,274,251 | \$12,156,103 | \$29,730,517 |

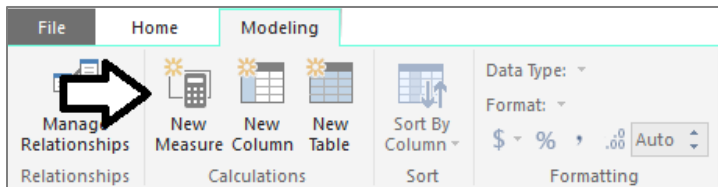
| Day of Week | 2012 | 2013 | 2014 | 2015 | Total |
|--------------|--------------------|--------------------|---------------------|---------------------|---------------------|
| Mon | \$314,471 | \$801,337 | \$1,460,373 | \$1,682,345 | \$4,258,527 |
| Tue | \$262,321 | \$791,863 | \$1,553,063 | \$1,726,955 | \$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 |

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

Exercise 5: 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 use DAX to write an expression which calculate a running total of sales revenue through the entire 4 years of sales activity.

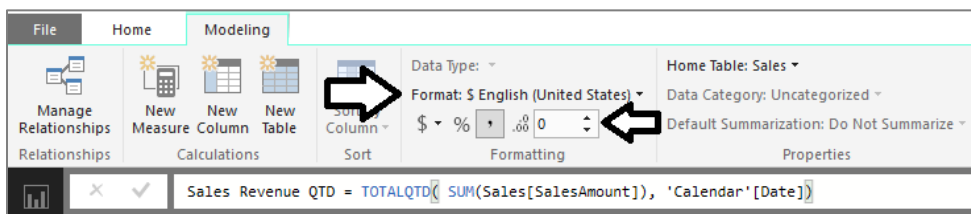
6. Create a measure named **Sales Revenue QTD** that calculates a quarter-to-date aggregate sum on the **SalesAmount** column of the **Sales** table.
 - a) Navigate to data view.
 - b) Select the **Sales** table from the **Fields** list.
 - c) Create a new measure by clicking the **New Measure** button 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( SUM(Sales[SalesAmount]), 'Calendar'[Date])
```

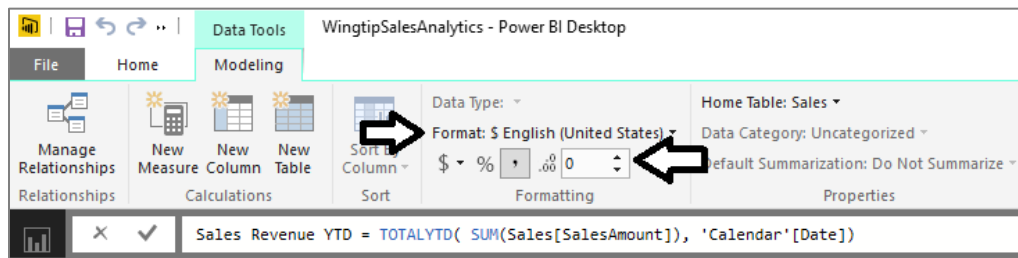
- e) Press the **ENTER** key to add the measure to data model.
 - f) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Currency > English (United States)**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to zero.



7. Create a measure named **Sales Revenue YTD** that calculates a year-to-date aggregate sum on the **SalesAmount** column of the **Sales** table.
 - 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( SUM(Sales[SalesAmount]), 'Calendar'[Date])
```

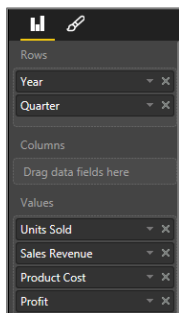
- e) Press the **ENTER** key to add the measure to data model.
 - f) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Currency > English (United States)**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to zero.



8. Create a measure named **Sales Revenue RT** that calculates a running total aggregate sum on the **SalesAmount** column of the **Sales** table.
 - 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(
    SUM(Sales[SalesAmount]),
    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)**. Also use the spinner control below the format menu to set the number of decimal places shown to zero.
9. Use the three new measures in a matrix visual.
 - a) Navigate to report mode and make sure the **Sales by Time Period** report page is active.
 - b) Select the matrix visual you created last that is currently positioned on the right-hand side of the page.
 - c) Examine the bottom of the **Visualizations** pane. Currently, the **Rows** well contains the **Year** column and the **Quarter** column. There are also four other measures in the **Values** well.



- d) Remove all the measures from the **Values** well except for the **Sales Revenue** measure. At this point, your visual should match the one shown in the following screenshot where sales revenue totals as shown at the quarterly level and at the yearly level.

| Year | Sales Revenue |
|--------------|---------------------|
| 2012 | \$1,943,986 |
| 2012-Q1 | \$85,494 |
| 2012-Q2 | \$260,944 |
| 2012-Q3 | \$557,293 |
| 2012-Q4 | \$1,040,256 |
| 2013 | \$5,356,177 |
| 2013-Q1 | \$945,310 |
| 2013-Q2 | \$1,111,831 |
| 2013-Q3 | \$1,372,617 |
| 2013-Q4 | \$1,926,420 |
| 2014 | \$10,274,251 |
| 2014-Q1 | \$1,868,225 |
| 2014-Q2 | \$2,224,114 |
| 2014-Q3 | \$2,548,569 |
| 2014-Q4 | \$3,633,343 |
| 2015 | \$12,156,103 |
| 2015-Q1 | \$2,604,726 |
| 2015-Q2 | \$2,206,560 |
| 2015-Q3 | \$3,095,046 |
| 2015-Q4 | \$4,249,771 |
| Total | \$29,730,517 |

- e) Drag and drop the **Month** column from the **Calendar** table into the **Rows** well below the two other columns.

| Rows |
|---------|
| Year |
| Quarter |
| Month |

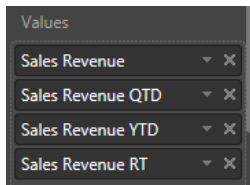
- f) Click the Expand All button so the matrix displays rows for month in addition to quarter and year.
g) You should be able to see that now the visual now has a deeper level of granularity because it is show sales revenue broken out into a separate aggregate value for each month.

| Year | Sales Revenue |
|----------|---------------|
| 2012 | \$1,943,986 |
| 2012-Q1 | \$85,494 |
| Jan 2012 | \$3,063 |
| Feb 2012 | \$33,218 |
| Mar 2012 | \$49,213 |
| 2012-Q2 | \$260,944 |
| Apr 2012 | \$40,434 |
| May 2012 | \$83,840 |
| Jun 2012 | \$136,670 |
| 2012-Q3 | \$557,293 |
| Jul 2012 | \$144,244 |
| Aug 2012 | \$197,952 |
| Sep 2012 | \$215,097 |
| 2012-Q4 | \$1,040,256 |
| Oct 2012 | \$239,513 |
| Nov 2012 | \$376,503 |
| Dec 2012 | \$424,240 |
| 2013 | \$5,356,177 |
| 2013-Q1 | \$945,310 |
| Jan 2013 | \$307,182 |

- h) Set a filter on the matrix visual so that is only displays sales revenue for the calendar years of 2014 and 2015. Accomplish this by setting a filter where the **Year** column is greater than or equal to **2014**.

| Filters |
|----------------------------------|
| Visual level filters |
| Month(All) |
| Quarter(All) |
| Sales Revenue(All) |
| Year |
| is greater than or equal to 2014 |
| Show items when the value: |
| is greater than or equal to |
| 2014 |
| And Or |
| Apply filter |

- i) After setting the filter, click the **Apply Filter** link below to apply your filter to the data shown in the visual.
- j) Resize the matrix visual to take up the entire right-hand side of the page.
- k) Drag and drop the **Sales Revenue QTD** measure from the **Sales** table into the **Values** well.
- l) Drag and drop the **Sales Revenue YTD** measure from the **Sales** table into the **Values** well.
- m) Drag and drop the **Sales Revenue RT** measure from the **Sales** table into the **Values** well.



- n) The matrix visual should now display three new columns for the three measure you added to the **Values** well.

| Year | Sales Revenue | Sales Revenue QTD | Sales Revenue YTD | Sales Revenue RT |
|----------|---------------|-------------------|-------------------|------------------|
| 2014 | \$10,274,251 | \$3,633,343 | \$10,274,251 | \$17,574,414 |
| 2014-Q1 | \$1,868,225 | \$1,868,225 | \$1,868,225 | \$9,168,388 |
| Jan 2014 | \$629,969 | \$629,969 | \$629,969 | \$7,930,132 |
| Feb 2014 | \$609,637 | \$1,239,606 | \$1,239,606 | \$8,539,770 |
| Mar 2014 | \$628,618 | \$1,868,225 | \$1,868,225 | \$9,168,388 |
| 2014-Q2 | \$2,224,114 | \$2,224,114 | \$4,092,338 | \$11,392,502 |
| Apr 2014 | \$661,588 | \$661,588 | \$2,529,812 | \$9,829,976 |
| May 2014 | \$748,193 | \$1,409,780 | \$3,278,005 | \$10,578,168 |
| Jun 2014 | \$814,333 | \$2,224,114 | \$4,092,338 | \$11,392,502 |
| 2014-Q3 | \$2,548,569 | \$2,548,569 | \$6,640,908 | \$13,941,071 |
| Jul 2014 | \$788,469 | \$788,469 | \$4,880,807 | \$12,180,970 |
| Aug 2014 | \$869,143 | \$1,657,611 | \$5,749,950 | \$13,050,113 |
| Sep 2014 | \$890,958 | \$2,548,569 | \$6,640,908 | \$13,941,071 |

- o) Now imagine your boss asks you to determine in what month the company reached 10 million dollars in total sales revenue. By looking at down the list of values for the **Sales Revenue RT** measure, you can see that the company finally hit \$10,000,000 in sales revenue in May of 2014.

| Year | Sales Revenue | Sales Revenue QTD | Sales Revenue YTD | Sales Revenue RT |
|----------|---------------|-------------------|-------------------|------------------|
| 2014 | \$10,274,251 | \$3,633,343 | \$10,274,251 | \$17,574,414 |
| 2014-Q1 | \$1,868,225 | \$1,868,225 | \$1,868,225 | \$9,168,388 |
| Jan 2014 | \$629,969 | \$629,969 | \$629,969 | \$7,930,132 |
| Feb 2014 | \$609,637 | \$1,239,606 | \$1,239,606 | \$8,539,770 |
| Mar 2014 | \$628,618 | \$1,868,225 | \$1,868,225 | \$9,168,388 |
| 2014-Q2 | \$2,224,114 | \$2,224,114 | \$4,092,338 | \$11,392,502 |
| Apr 2014 | \$661,588 | \$661,588 | \$2,529,812 | \$9,829,976 |
| May 2014 | \$748,193 | \$1,409,780 | \$3,278,005 | \$10,578,168 |
| Jun 2014 | \$814,333 | \$2,224,114 | \$4,092,338 | \$11,392,502 |
| 2014-Q3 | \$2,548,569 | \$2,548,569 | \$6,640,908 | \$13,941,071 |
| Jul 2014 | \$788,469 | \$788,469 | \$4,880,807 | \$12,180,970 |



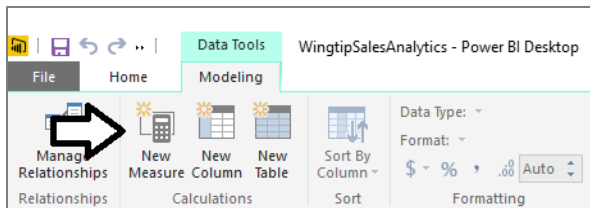
10. Save the work you have done by clicking the **Save** button in the upper left corner of the Power BI Desktop window.

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.

Exercise 6: Create Measures to Monitor Growth in Sales Revenue

In this exercise you will create new measures to calculate the growth of sales revenue on a month-by-month basis. After that you will create additional measures that will act as KPIs to monitor the health of sales growth and provide visual indications as to how each month has done when compared to the previous month.

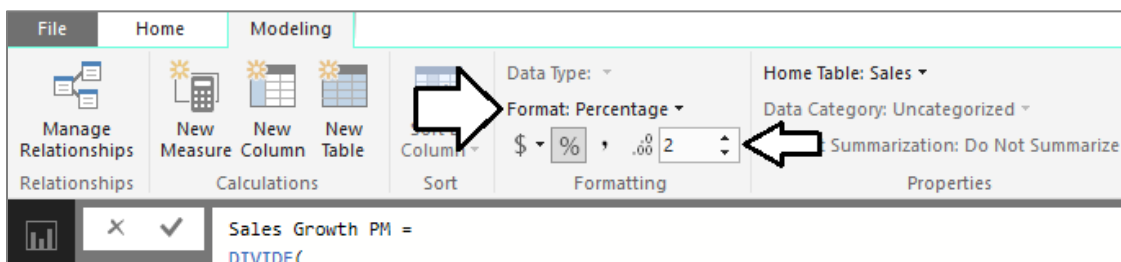
1. Create a measure named **Sales Growth PM** that calculates the percentage increase between sales revenue for the current month and sales revenue for the previous month.
 - a) Navigate to data view.
 - b) Select the **Sales** table from the **Fields** list.
 - c) Create a new measure by clicking the **New Measure** button in the ribbon.



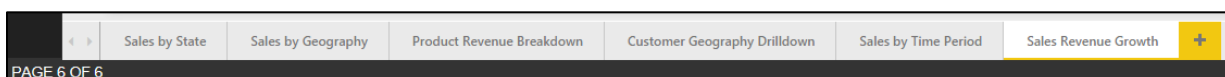
- d) Enter the following DAX expression into the formula bar to create the measure named **Sales Growth PM**.

```
Sales Growth PM =
DIVIDE(
    SUM(Sales[SalesAmount]) -
    CALCULATE(
        SUM(Sales[SalesAmount]),
        PREVIOUSMONTH(Calendar[Date])
    ),
    CALCULATE(
        SUM(Sales[SalesAmount]),
        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**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to **2**.

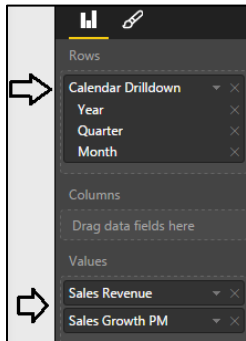


2. Create a new page in the project's report.
 - a) Navigate to report view.
 - b) Add a new page by clicking the (+) button on the right of the page navigation menu.
 - c) Once the new page has been created, modify its title to **Sales Revenue Growth**.

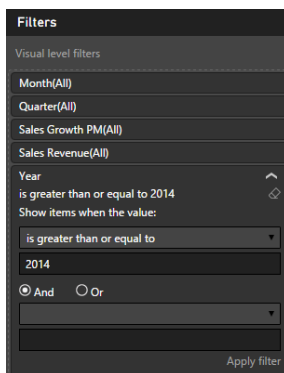


3. Create a new matrix visual to show month-to-month sales revenue growth in 2014 and 2015.

- Click the **New Visual** button on the ribbon to add a new visual to the page.
- Change the visual to a matrix by clicking the **Matrix** button in the **Visualizations** list.
- Drag and drop the **Calendar Drilldown** column hierarchy from the **Calendar** table into the **Rows** well.
- Drag and drop the **Sales Revenue** measure from the **Sales** table into the **Value** well.
- Drag and drop the **Sales Growth PM** measure from the **Sales** table into the **Value** well.
- The **Rows** well and the **Values** well for the matrix visual should match the following screenshot.



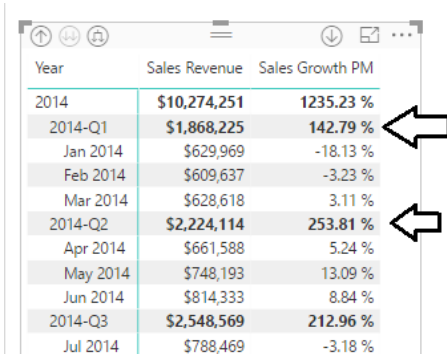
- Move down in the **Visualizations** page to the **Filters** section. Set a filter for **Year** where value is greater or equal 2014.



- Click the **Apply filter** button at the bottom of the **Filters** section for the filter to take effect.
- On the matrix, click the **Expand All** button to show **Year**, **Quarter** and **Month**.
- Use your mouse to resize the matrix visual so you can see all the rows and columns. Give the matrix visual a width that is half the width of the page so you can add additional columns over the next few steps without having to resize the visual again.

| Year | Sales Revenue | Sales Growth PM |
|----------|---------------|-----------------|
| 2014 | \$10,274,251 | 1235.23 % |
| 2014-Q1 | \$1,868,225 | 142.79 % |
| Jan 2014 | \$629,969 | -18.13 % |
| Feb 2014 | \$609,637 | -3.23 % |
| Mar 2014 | \$628,618 | 3.11 % |
| 2014-Q2 | \$2,224,114 | 253.81 % |
| Apr 2014 | \$661,588 | 5.24 % |
| May 2014 | \$748,193 | 13.09 % |
| Jun 2014 | \$814,333 | 8.84 % |
| 2014-Q3 | \$2,548,569 | 212.96 % |
| Jul 2014 | \$788,469 | -3.18 % |
| Aug 2014 | \$869,143 | 10.23 % |
| Sep 2014 | \$890,958 | 2.51 % |
| 2014-Q4 | \$3,633,343 | 307.80 % |
| Oct 2014 | \$988,789 | 10.98 % |
| Nov 2014 | \$999,574 | 1.09 % |

- k) Inspect the values produced by the **Sales Growth PM** measure. You can see that a value has been calculated for each month. You should also notice that the matrix currently displays values for the **Sales Growth PM** measure in the **Total** row. The values in the **Total** row are calculated at the quarterly level and not at the month level.



| Year | Sales Revenue | Sales Growth PM |
|----------|---------------|-----------------|
| 2014 | \$10,274,251 | 1235.23 % |
| 2014-Q1 | \$1,868,225 | 142.79 % |
| Jan 2014 | \$629,969 | -18.13 % |
| Feb 2014 | \$609,637 | -3.23 % |
| Mar 2014 | \$628,618 | 3.11 % |
| 2014-Q2 | \$2,224,114 | 253.81 % |
| Apr 2014 | \$661,588 | 5.24 % |
| May 2014 | \$748,193 | 13.09 % |
| Jun 2014 | \$814,333 | 8.84 % |
| 2014-Q3 | \$2,548,569 | 212.96 % |
| Jul 2014 | \$788,469 | -3.18 % |

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 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. Now that you have seen the 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 context where the time interval is at a granularity other than at the monthly level.

4. Modify the DAX expression for the **Sales Growth PM** measure.

- Navigate to data view.
- 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.
- 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])
```

- 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]))
```

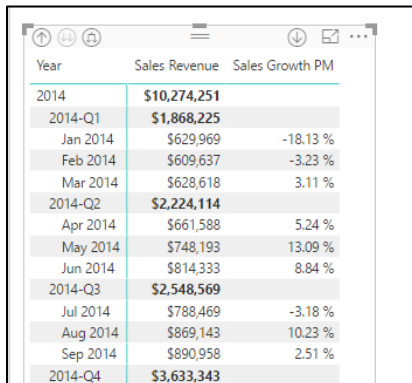
- You will need to ensure that both these expressions are true before the **Sales Growth PM** measure evaluates to a value other than a blank value. You can write the following DAX expression using the DAX **&&** operator to return true when both inner conditions are true

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

- 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(
        SUM(Sales[SalesAmount]) -
        CALCULATE(
            SUM(Sales[SalesAmount]),
            PREVIOUSMONTH(Calendar[Date])
        ),
        CALCULATE(
            SUM(Sales[SalesAmount]),
            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 that the **Sales Growth PM** measure is now returning blank values in the **Total** row for the quarterly evaluation.



| Year | Sales Revenue | Sales Growth PM |
|----------|---------------|-----------------|
| 2014 | \$10,274,251 | |
| 2014-Q1 | \$1,868,225 | |
| Jan 2014 | \$629,969 | -18.13 % |
| Feb 2014 | \$609,637 | -3.23 % |
| Mar 2014 | \$628,618 | 3.11 % |
| 2014-Q2 | \$2,224,114 | |
| Apr 2014 | \$661,588 | 5.24 % |
| May 2014 | \$748,193 | 13.09 % |
| Jun 2014 | \$814,333 | 8.84 % |
| 2014-Q3 | \$2,548,569 | |
| Jul 2014 | \$788,469 | -3.18 % |
| Aug 2014 | \$869,143 | 10.23 % |
| Sep 2014 | \$890,958 | 2.51 % |
| 2014-Q4 | \$3,633,343 | |

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

5. Create a measure named **Sales Growth PM Eval** that inspects the value of the **Sales Growth PM** measure and evaluates to a short string value to indicate the health of the sales growth value.
 - 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 the following DAX expression into the formula bar to create the measure named **Sales Growth PM Eval**.

```
Sales Growth PM Eval =
IF(
    ISNUMBER([Sales Growth PM]),
    SWITCH(TRUE(),
        ([Sales Growth PM] >= 0.2), "EXCELLENT",
        ([Sales Growth PM] >= 0.1), "GOOD",
        ([Sales Growth PM] >= 0), "OK",
        ([Sales Growth PM] >= -0.1), "BAD",
        ([Sales Growth PM] < -0.1), "AWFUL"
    )
)
```

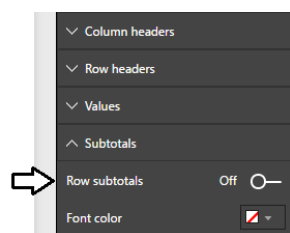
- e) Navigate to report view and select the matrix visual on the **Sales Revenue Growth** page.
 - f) Drag and drop the **Sales Growth PM Eval** measure from the **Sales** table into the **Values** well in the **Visualizations** pane.



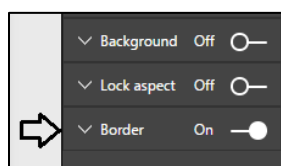
- g) You should now see values for the **Sales Growth PM Eval** measure which indicate the health of sales revenue growth.

| Year | Sales Revenue | Sales Growth PM | Sales Growth PM Eval |
|----------|---------------|-----------------|----------------------|
| 2014 | \$10,274,251 | | |
| 2014-Q1 | \$1,868,225 | | |
| Jan 2014 | \$629,969 | -18.13 % | AWFUL |
| Feb 2014 | \$609,637 | -3.23 % | BAD |
| Mar 2014 | \$628,618 | 3.11 % | OK |
| 2014-Q2 | \$2,224,114 | | |
| Apr 2014 | \$661,588 | 5.24 % | OK |
| May 2014 | \$748,193 | 13.09 % | GOOD |
| Jun 2014 | \$814,333 | 8.84 % | OK |
| 2014-Q3 | \$2,548,569 | | |
| Jul 2014 | \$788,469 | -3.18 % | BAD |
| Aug 2014 | \$869,143 | 10.23 % | GOOD |
| Sep 2014 | \$890,958 | 2.51 % | OK |
| 2014-Q4 | \$3,633,343 | | |
| Oct 2014 | \$988,789 | 10.98 % | GOOD |
| Nov 2014 | \$999,574 | 1.09 % | OK |
| Dec 2014 | \$1,644,980 | 64.57 % | EXCELLENT |
| 2015 | \$12,156,103 | | |
| 2015-Q1 | \$2,604,726 | | |
| Jan 2015 | \$959,863 | -41.65 % | AWFUL |

h) Turn off **Row subtotals** for the matrix.



i) At the bottom of the **Format** properties pane, update the Border property from **Off** to **On**.



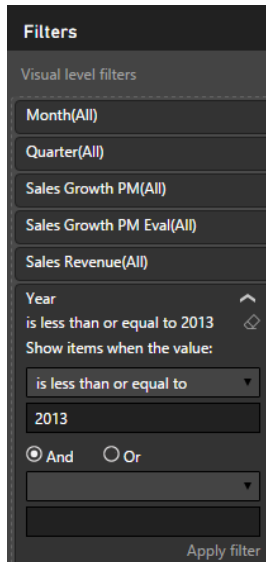
j) Save the work you have done by clicking the **Save** button in the upper left corner of the Power BI Desktop window.

6. Clone the matrix visual by copying it to the Windows clipboard and pasting it to make a copy.

- Select the matrix visual and copy it to the Windows clipboard.
- Perform a paste operation to create an identical copy of the matrix visual.
- Position the two visuals side by side to the left as shown in the following screenshot.

| Year | Sales Revenue | Sales Growth PM | Sales Growth PM Eval |
|----------|---------------|-----------------|----------------------|
| 2014 | | | |
| 2014-Q1 | | | |
| Jan 2014 | \$629,969 | -18.13 % | AWFUL |
| Feb 2014 | \$609,637 | -3.23 % | BAD |
| Mar 2014 | \$628,618 | 3.11 % | OK |
| 2014-Q2 | | | |
| Apr 2014 | \$661,588 | 5.24 % | OK |
| May 2014 | \$748,193 | 13.09 % | GOOD |
| Jun 2014 | \$814,333 | 8.84 % | OK |
| 2014-Q3 | | | |
| Jul 2014 | \$788,469 | -3.18 % | BAD |
| Aug 2014 | \$869,143 | 10.23 % | GOOD |
| Sep 2014 | \$890,958 | 2.51 % | OK |
| 2014-Q4 | | | |
| Oct 2014 | \$988,789 | 10.98 % | GOOD |
| Nov 2014 | \$999,574 | 1.09 % | OK |
| Dec 2014 | \$1,644,980 | 64.57 % | EXCELLENT |
| 2015 | | | |
| 2015-Q1 | | | |
| Jan 2015 | \$959,863 | -41.65 % | AWFUL |
| Feb 2015 | \$969,330 | 0.99 % | OK |
| Mar 2015 | \$675,533 | -30.31 % | AWFUL |
| 2015-Q2 | | | |
| Apr 2015 | \$722,456 | 6.95 % | OK |

7. Update the matrix visual on the left to display financial data for the years of 2012 and 2013.
 - a) Select the matrix visual on the left.
 - b) Locate the **Filters** section the **Field** property pane.
 - c) Update the filter for the **Year** column where value **is less than or equal to 2013**.

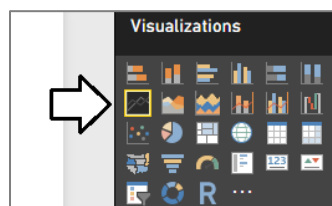


- d) Click the **Apply filter** button for the filter to take effect.
- e) The visual on the left should now display sales data for years of 2012 and 2013 while the visual on the right display sales data for 2014 and 2015.

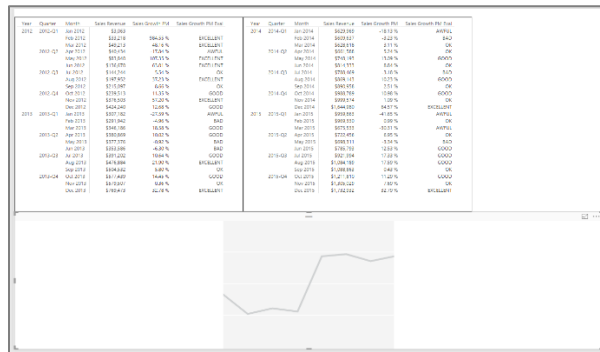
| Year | Sales Revenue | Sales Growth PM | Sales Growth PM Eval |
|----------|---------------|-----------------|----------------------|
| 2012 | | | |
| 2012-Q1 | | | |
| Jan 2012 | \$3,063 | | |
| Feb 2012 | \$33,218 | 984.55 % | EXCELLENT |
| Mar 2012 | \$49,213 | 48.16 % | EXCELLENT |
| 2012-Q2 | | | |
| Apr 2012 | \$40,434 | -17.84 % | AWFUL |
| May 2012 | \$83,840 | 107.35 % | EXCELLENT |
| Jun 2012 | \$136,670 | 63.01 % | EXCELLENT |
| 2012-Q3 | | | |
| Jul 2012 | \$144,244 | 5.54 % | OK |
| Aug 2012 | \$197,952 | 37.23 % | EXCELLENT |
| Sep 2012 | \$215,097 | 8.66 % | OK |
| 2012-Q4 | | | |
| Oct 2012 | \$239,513 | 11.35 % | GOOD |
| Nov 2012 | \$376,503 | 57.20 % | EXCELLENT |
| Dec 2012 | \$424,240 | 12.68 % | GOOD |
| 2013 | | | |
| 2013-Q1 | | | |
| Jan 2013 | \$307,182 | -27.59 % | AWFUL |
| Feb 2013 | \$291,942 | -4.96 % | BAD |
| Mar 2013 | \$346,186 | 18.58 % | GOOD |

| Year | Sales Revenue | Sales Growth PM | Sales Growth PM Eval |
|----------|---------------|-----------------|----------------------|
| 2014 | | | |
| 2014-Q1 | | | |
| Jan 2014 | \$629,969 | -18.13 % | AWFUL |
| Feb 2014 | \$609,637 | -3.23 % | BAD |
| Mar 2014 | \$628,618 | 3.11 % | OK |
| 2014-Q2 | | | |
| Apr 2014 | \$661,588 | 5.24 % | OK |
| May 2014 | \$748,193 | 13.09 % | GOOD |
| Jun 2014 | \$814,333 | 8.84 % | OK |
| 2014-Q3 | | | |
| Jul 2014 | \$788,469 | -3.18 % | BAD |
| Aug 2014 | \$869,143 | 10.23 % | GOOD |
| Sep 2014 | \$890,958 | 2.51 % | OK |
| 2014-Q4 | | | |
| Oct 2014 | \$988,789 | 10.98 % | GOOD |
| Nov 2014 | \$999,574 | 1.09 % | OK |
| Dec 2014 | \$1,644,980 | 64.57 % | EXCELLENT |
| 2015 | | | |
| 2015-Q1 | | | |
| Jan 2015 | \$959,863 | -41.65 % | AWFUL |
| Feb 2015 | \$969,330 | 0.99 % | OK |
| Mar 2015 | \$675,533 | -30.31 % | AWFUL |

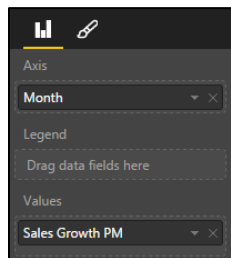
- f) Save the work you have done by clicking the **Save** button in the upper left corner of the Power BI Desktop window.
8. Add a line chart visual to the bottom of the report page to show how sales revenue has grown from month to month.
 - a) Click on the whitespace on the report to make sure that neither of the two visuals are selected.
 - b) Click on the Line chart button in the **Visualizations** list to create a new Line chart visual.



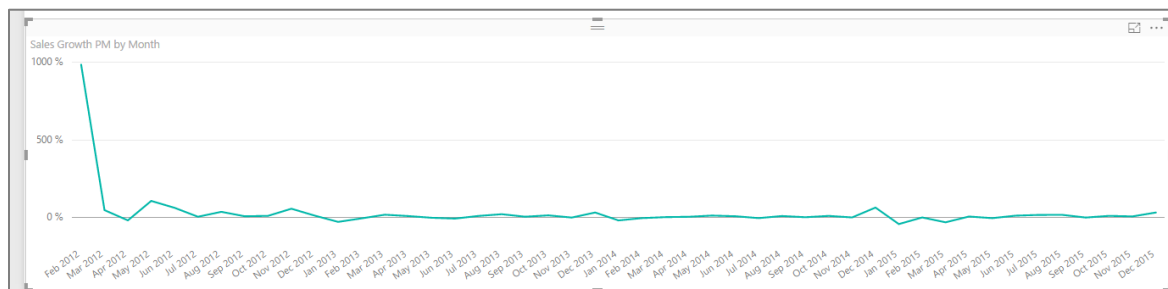
- c) Reposition the new visual so it takes up the entire width of the page at the bottom of the report.



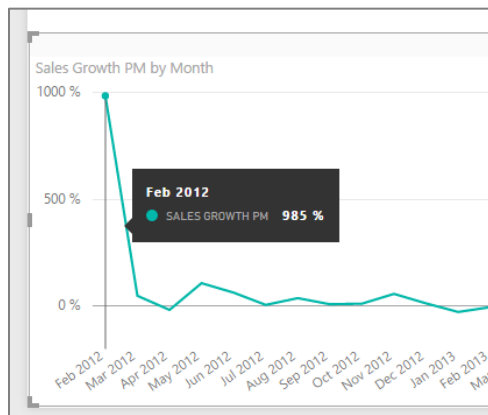
- d) Drag and drop the **Month** field from the **Calendar** table into the **Axis** well.
e) Drag and drop the **Sales Growth PM** field from the **sales** table into the **Values** well.



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



9. 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 write more DAX code to add additional logic so that the Sales Growth PM measure returns a blank value when the previous month does not contain all the days for a full month.

10. 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.
 - 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 the 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[PurchaseDate]))
```

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.

11. Once you have created the **Previous Month Is Valid** measure, right click on it in the fields list and click **Hide in Report View**.
12. 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.

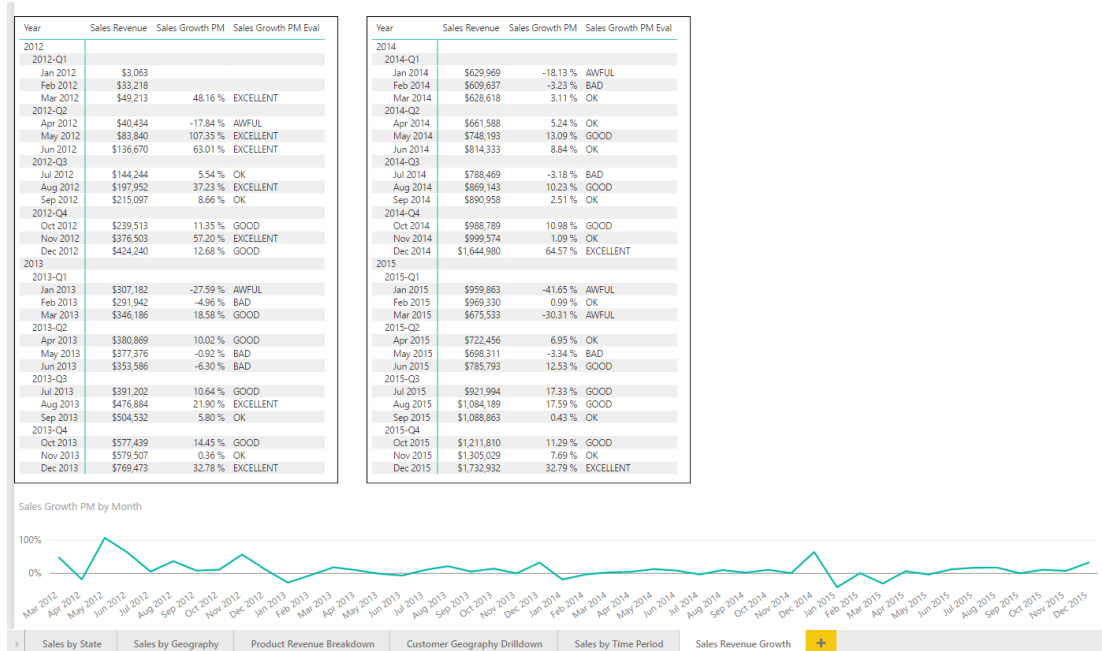
```
(
  ISFILTERED(Calendar[Month]) &&
  NOT(ISFILTERED(Calendar[Date])) &&
  [Previous Month Is Valid]
)
```

- d) The DAX 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(
    SUM(Sales[SalesAmount]) -
    CALCULATE(
```

```
SUM(Sales[SalesAmount]),
PREVIOUSMONTH(Calendar[Date])
),
CALCULATE(
SUM(Sales[SalesAmount]),
PREVIOUSMONTH(Calendar[Date])
),
BLANK()
)
```

13. 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.



14. Save your work to the **Wingtip Sales Analysis** project by clicking the **Save** button in the ribbon.

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