Designing a Data Model in Power BI Desktop

Lab Time: 60-90 minutes

Lab Folder: C:\Student\Modules\03_DataModeling\Lab

Lab Overview: In this set of lab exercises, you will continue to work on the Power BI Desktop project you started in the previous lab. At this point, you have already imported sales data from a SQL Azure database and transformed it using the query features of Power BI Desktop. The ending point in the previous lab will be your starting point for this lab. You work in this lab will be using the data modeling tools of Power BI Desktop and using DAX to write expressions for calculated columns and measures. Along the way, you will build a few simple reports and add visuals so you can see the effects of your modeling efforts.

Lab Dependencies: This lab assumes you have completed the previous lab titled **Designing Queries to Generate a Data Model in Power BI Desktop** in which you created a Power BI Desktop project named **Wingtip Sales Analysis.pbix**. In the previous lab you should have imported data into the data model using data from the **WingtipSalesDB** database in SQL Azure and transformed the data into a schema that is better suited for data modeling and analysis. 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 Analysis.pbix** which is located in the student folder at **C:\Student\Modules\02_Queries\Lab** into the folder at **C:\Student\Projects**.

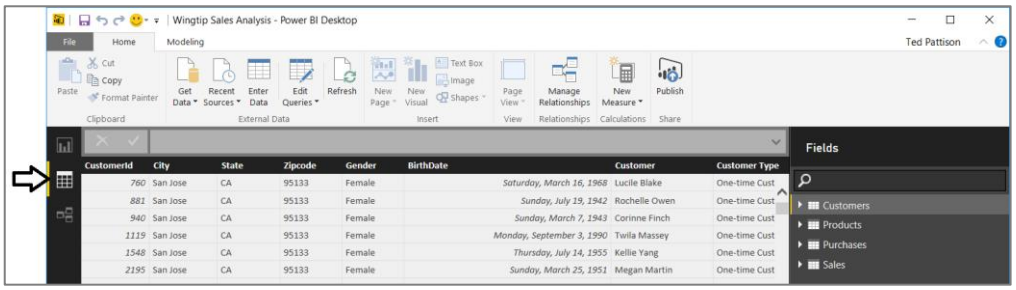
Exercise 1: Configuring Table Relationships

In this exercise, you configure relationships between the tables in the data model so that every relationship si based on a . After that, you will create a simple report to see the effects of your formatting changes.

1. Launch Power BI Desktop.
2. Using the Power BI Desktop **File > Open** command, open **Wingtip Sales Analytics.pbix** located at the following path.

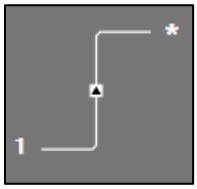
```
C:\Student\Projects\Wingtip Sales Analysis.pbix
```

3. When the project opens, click the table icon in the middle of the sidebar to enter data view mode.



4. Take a moment to review the data in each of the four tables in the data model by clicking on the tables inside the **Fields** list.
5. Inspect the tables relationships that have been created in the project's data model.
   a) Click the bottom button in the sidebar to navigate to relationship view.
   b) You should see that the four tables in a star schema where **Sales** has a relationship with each of the three other tables.

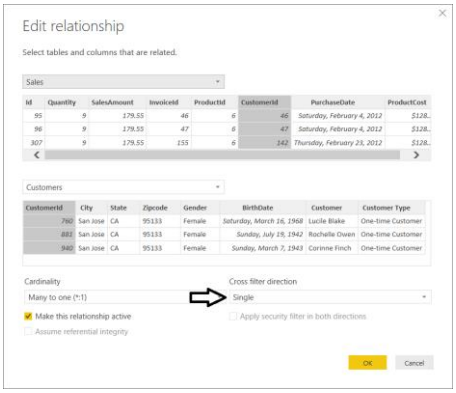
c) Currently, all three table relationships are shown with a single arrow indicating their **Cross filter direction** is set to **Single**.



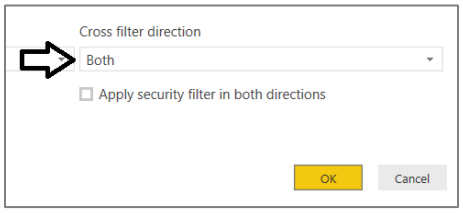
6. Modify the relationship between **Sales** and **Customers** to set **the Cross filter direction** to **Both**.

a) Double-click the relationship line that connects **Customers** to **Sales** to open the relationship in the **Edit relationship** dialog.

b) Locate the dropdown menu with the **Cross filter direction** settings.

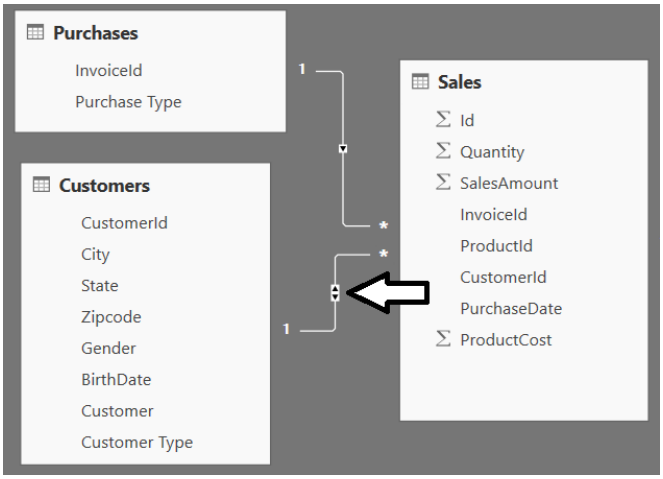


c) Change the **Cross filter direction** settings from **Single** to Both.



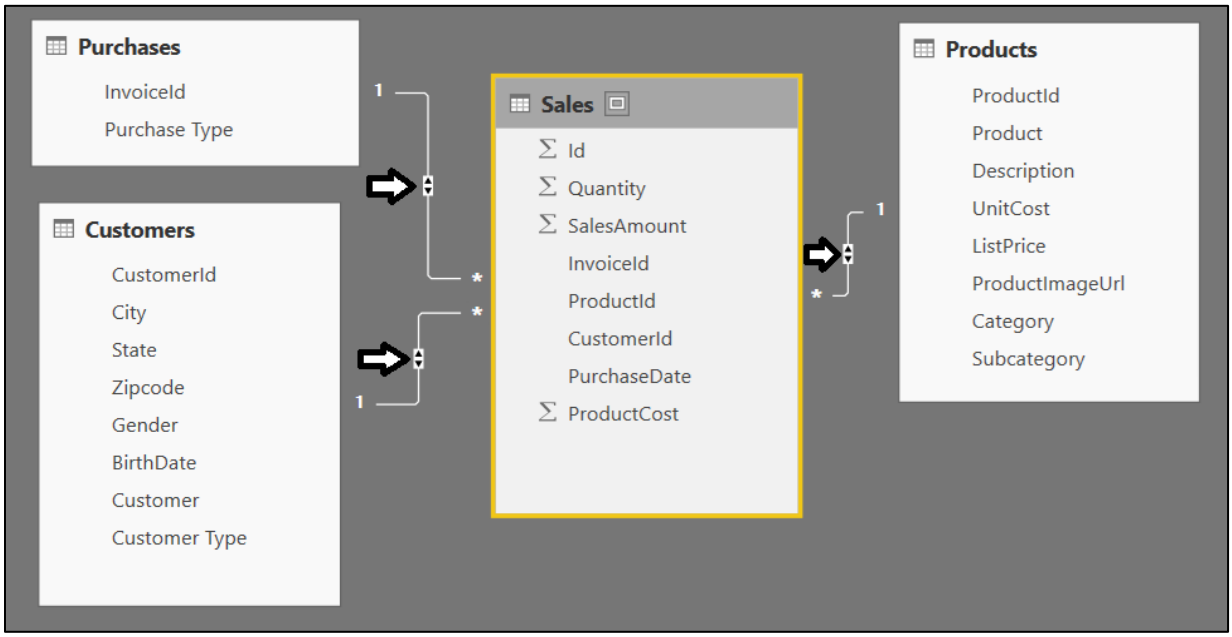
d) Click the **OK** button to close the **Edit** relationship dialog.

e) Verify that the relationship line between Customers and Sales now shows a bidirectional arrow



In the next step you will use the exact same set of steps to modify the Cross filter direction of two more table relationships.

7. Configure the **Cross filter direction** for the other two relationships in the data model.
   a) Modify the relationship between **Sales** and **Purchase** to set the **Cross filter direction** to **Both**.
   b) Modify the relationship between **Sales** and **Products** to set the **Cross filter direction** to **Both**.
   c) Verify that all three relationships now show a bidirectional line indicating a **Cross Filter direction** of **Both**.

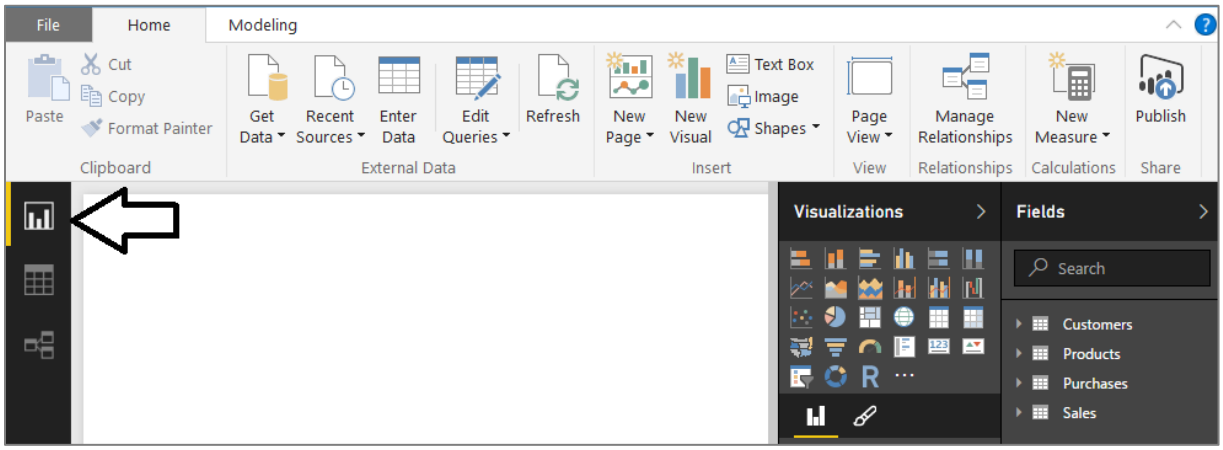


8. Save the work you have done by clicking the Save button in the upper left corner of the Power BI Desktop window.

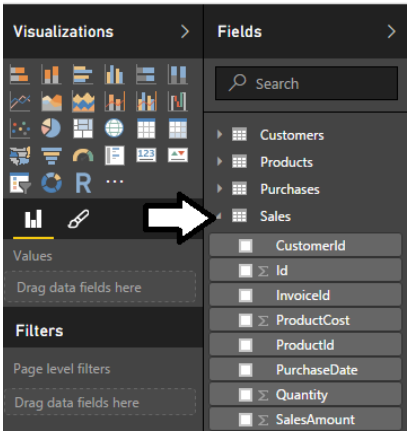
Exercise 2: Hiding and Formatting Columns in the Data Model

In this exercise you will continue to refine your data model by hiding and formatting table columns. After that, you will create a simple report to see the effects of your data modeling work.

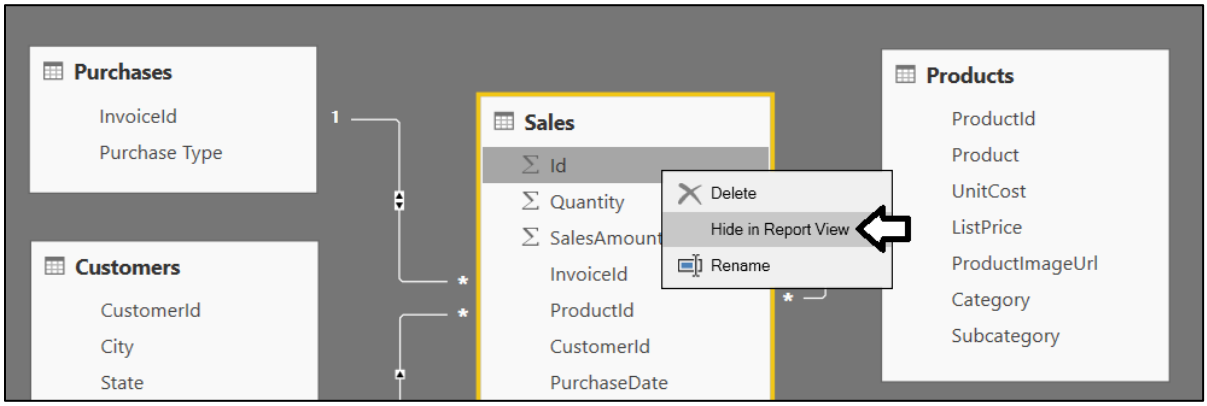
1. Now it's time to inspect the data model from a different perspective. More specifically, you will examine the data model from the perspective of a consumer who is designing reports and creating visuals using the Power BI Desktop report designer.
   a) Click the top button in the sidebar to navigate to report view.



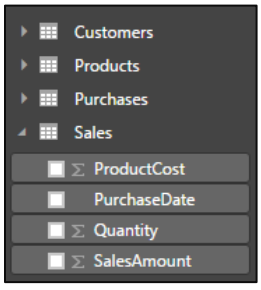
b) Inside the **Fields** list, use the mouse to expand the fields inside the **Sales** table. You can see that there are several fields in the **Sales** table that will never be used when designing reports such as the four identifier columns. The data model will be easier for consumers such as report designers to understand if you hide these types of fields which add unnecessary clutter.



2. Use relationship view to update the data model by hiding fields in the **Sales** table that are unnecessary to display in report view.
   a) Navigate to relationship view in the Power BI Desktop window.
   b) Using the mouse, right-click on the **Id** column in the **Sales** table and select the **Hide in Report View** command.

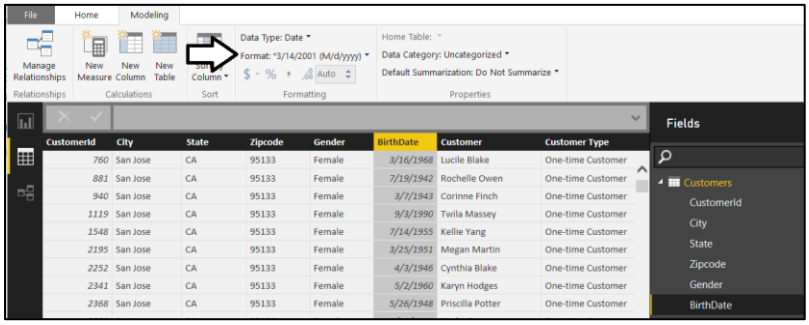


   c) Right-click on the **InvoiceId** column in the **Sales** table and select the **Hide in Report View** command.
   d) Right-click on the **CustomerId** column in the **Sales** table and select the **Hide in Report View** command.
   e) Right-click on the **ProductId** column in the **Sales** table and select the **Hide in Report View** command.
   f) Now, navigate back to report view and examine the set of fields displayed for the **Sales** table.



You should be able to see that hiding unnecessary columns from report view makes your data model easier to use. This is especially true in the scenario where you are creating a data model that other less-technical people will be using to create reports & dashboards.

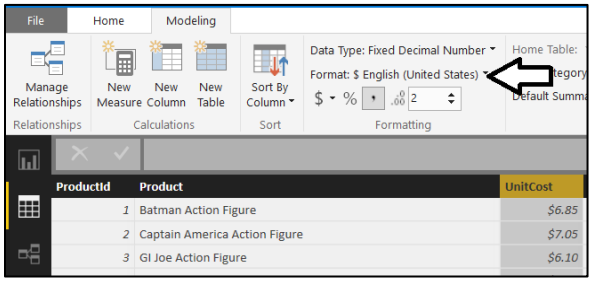
3. Modify the formatting of the **BirthDate** column in the **Customers** table.
   a) In the Power BI Desktop windows, navigate back to data view.
   b) In the **Fields** list on the right, select the **Customers** table to display its rows and columns.
   c) Select the **BirthDate** column.
   d) Modify the **BirthDate** column formatting using the **Format** menu to select a format of **Date Time > 3/14/2001 (M/d/yyyy)**.



   e) The **BirthDate** column should now reflect the change in formatting.

Gender	BirthDate	Customer
Female	3/16/1968	Lucile Blake
Female	7/19/1942	Rochelle Owen
Female	3/7/1943	Corinne Finch
Female	9/3/1990	Twila Massey

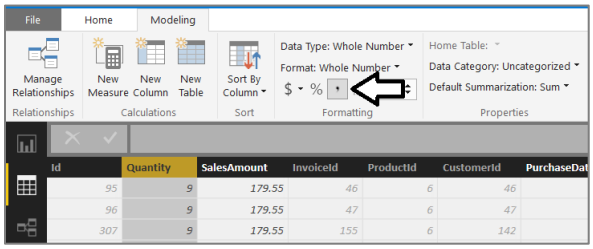
4. Modify the formatting of columns in the **Products** table.
   a) In the **Fields** list on the right, select the **Products** table to display its rows and columns.
   b) Select the **UnitCost** column by clicking on its column header.
   c) Use the **Format** menu button in the ribbon to update the format setting to **Currency > $ English (United States)** and set the number of decimal places to **2** in the spin control located underneath the **Format** dropdown menu.



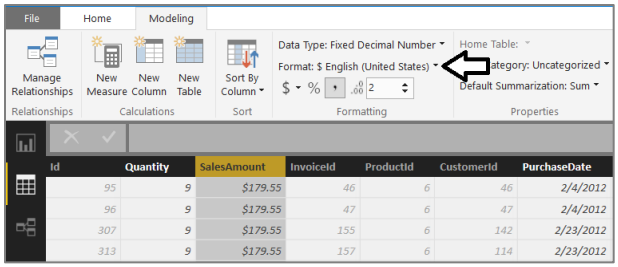
   d) Changing the format setting of the **ListPrice** column to **$ English (United States)** and set the number of decimal places to **2** so it matches the **UnitCost** column.

UnitCost	ListPrice
$6.85	$14.95
$7.05	$12.95
$6.1	$14.95
$2.85	$9.95

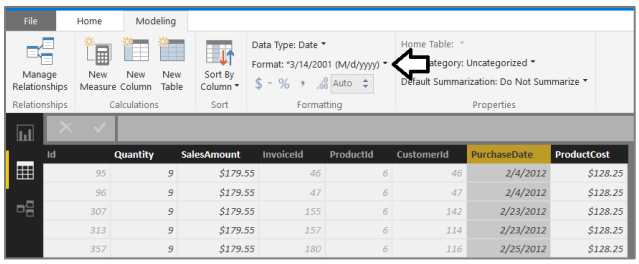
5. Modify the formatting of columns in the **Sales** table.
   a) In the **Fields** list on the right, select the **Sales** table to display its rows and columns.
   b) Select the **Quantity** column by clicking on its column header.
   c) Modify the **Quantity** column by clicking to select the comma button on the ribbon to add a comma separator.



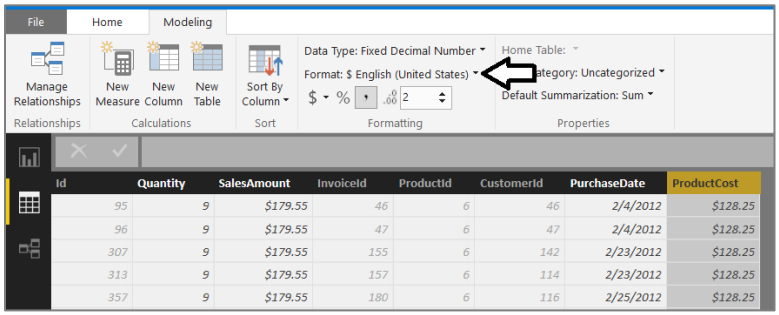
   d) Select the **SalesAmount** column by clicking on its column header.
   e) Modify the formatting of the **SalesAmount** column to **Currency > $ English (United States)** and set the number of decimal places to **2** in the spin control located underneath the **Format** dropdown menu



   f) Select the **PurchaseDate** column by clicking on its column header.
   g) Modify the formatting of the **PurchaseDate** to of **Date Time > 3/14/2001 (M/d/yyyy)**.

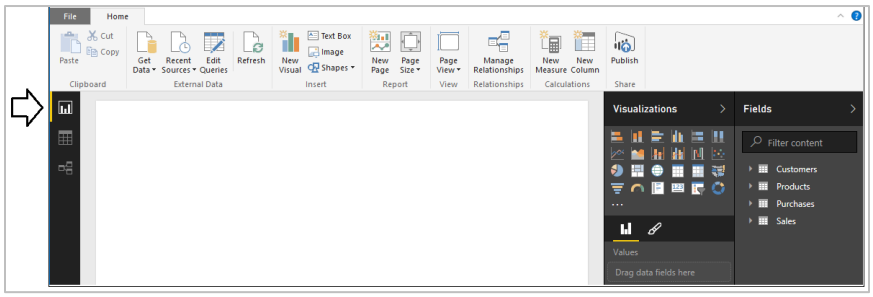


   h) Select the **ProductCost** column by clicking on its column header.
   i) Modify the formatting of the **ProductCost** column to **Currency> $ English (United States)** and set the number of decimal places to **2** in the spin control located underneath the **Format** dropdown menu.

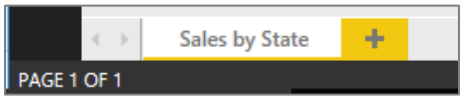


6. See the effect of your formatting by adding a visual to a report.

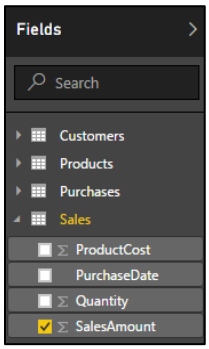
a) Navigate to report view. There should be an empty report for the project with a single page named **Page 1**.



b) Change the name of the page in the report from **Page 1** to **Sales by State**.

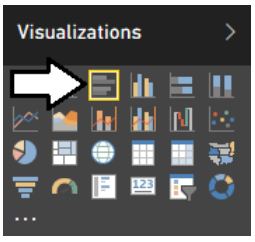


c) Create a new visual in the report by selecting the checkbox for the **SalesAmount** column in the **Fields** list.



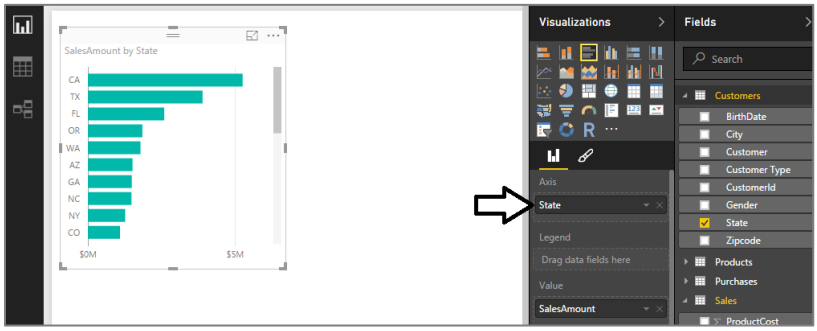
d) When you select the **SalesAmount** column, Power BI Desktop will automatically add a new visual to the report based on the visualization type of **Clustered Column Chart**.

e) Click the **Clustered Bar Chart** button in the **Visualizations** list to change the visualization type to a clustered bar chart.

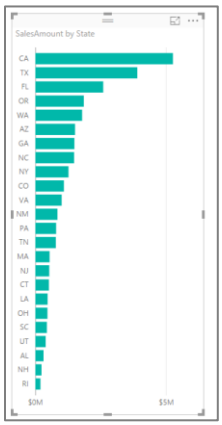


f) Since you haven't added any row labels yet, the clustered bar chart currently displays a single bar showing total sales.

g) Drag and drop the **States** field from the **Fields** list into the **Axis** well of the **Visualizations** pane.



h) Use your mouse to resize the visual so that it can display all the stats without a scrollbar.



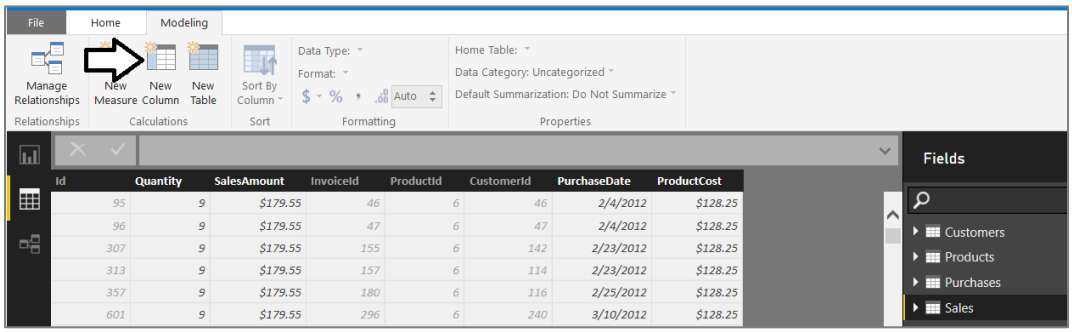
i) Reposition the visual to the bottom left corner of the page as shown in the following screenshot.

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

Exercise 3: Extending the Data Model by Creating Calculated Columns

In this exercise you will create several calculated columns which will require you to write and test DAX expressions. After creating calculated columns, you will use them to enhance reports in the current project.

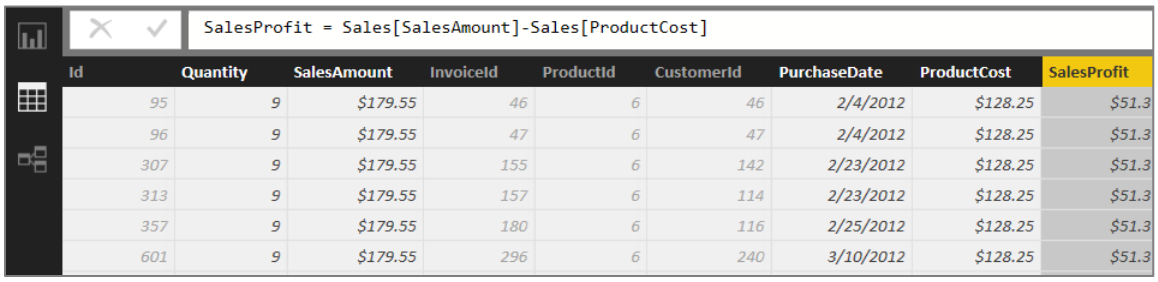
1. Add a calculated column to the **Sales** table named **SalesProfit** to determine profit by calculating the difference between **SalesAmount** and **ProductCost**.
   a) Navigate to data view.
   b) Select the **Sales** table in the **Fields** list.
   c) Create a new calculated column by clicking the **New Column** button in the ribbon.



   d) Enter to following DAX expression into the formula bar to create the calculated column named **SalesProfit**.

```
SalesProfit = Sales[SalesAmount]-Sales[ProductCost]
```

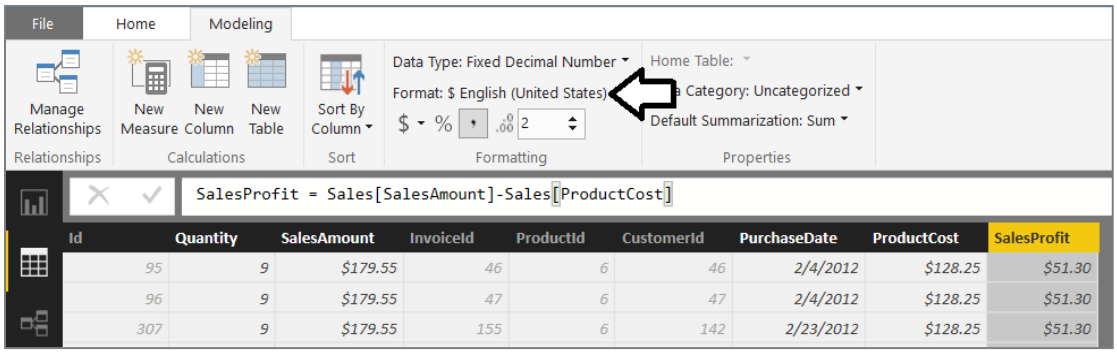
   e) Press the **ENTER** key to add the calculated column to the table. You should be able to see a **SalesProfit** value for each row in the **Sales** table.



Id	Quantity	SalesAmount	InvoiceId	ProductId	CustomerId	PurchaseDate	ProductCost	SalesProfit
95	9	$179.55	46	6	46	2/4/2012	$128.25	$51.3
96	9	$179.55	47	6	47	2/4/2012	$128.25	$51.3
307	9	$179.55	155	6	142	2/23/2012	$128.25	$51.3
313	9	$179.55	157	6	114	2/23/2012	$128.25	$51.3
357	9	$179.55	180	6	116	2/25/2012	$128.25	$51.3
601	9	$179.55	296	6	240	3/10/2012	$128.25	$51.3

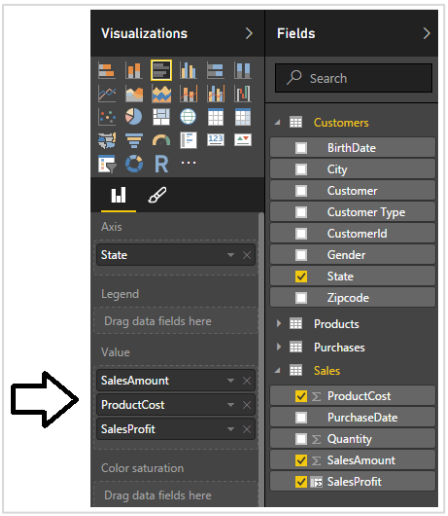
When writing the DAX for a calculated column, you can reference other columns using the table name combined together with the column name (e.g. **Sales[SalesAmount]**) or by using just the column name (e.g. **[SalesAmount]**).

   f) Configure the column's formatting by using the **Format** menu on the ribbon to select **Currency > English (United States)**.

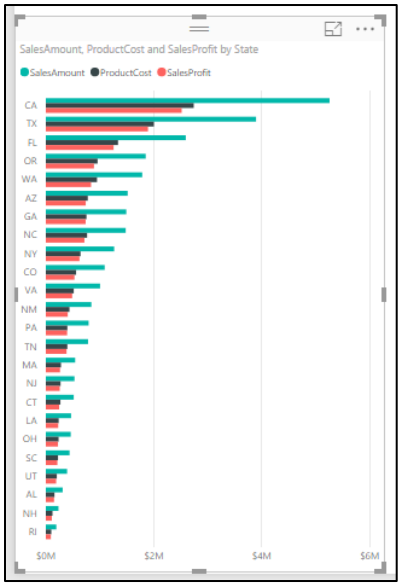


Id	Quantity	SalesAmount	InvoiceId	ProductId	CustomerId	PurchaseDate	ProductCost	SalesProfit
95	9	$179.55	46	6	46	2/4/2012	$128.25	$51.30
96	9	$179.55	47	6	47	2/4/2012	$128.25	$51.30
307	9	$179.55	155	6	142	2/23/2012	$128.25	$51.30

2. Update the clustered bar chart visual you created in the previous exercise.
   a) Navigate to report view.
   b) Select the visual you created in the previous exercise.
   c) Using your mouse, drag and drop the **ProductCost** column from the **Fields** list into the **Value** well in the **Visualizations** pane.
   d) Using your mouse, drag and drop the **SalesProfit** column from the **Fields** list into the **Value** well in the **Visualizations** pane.



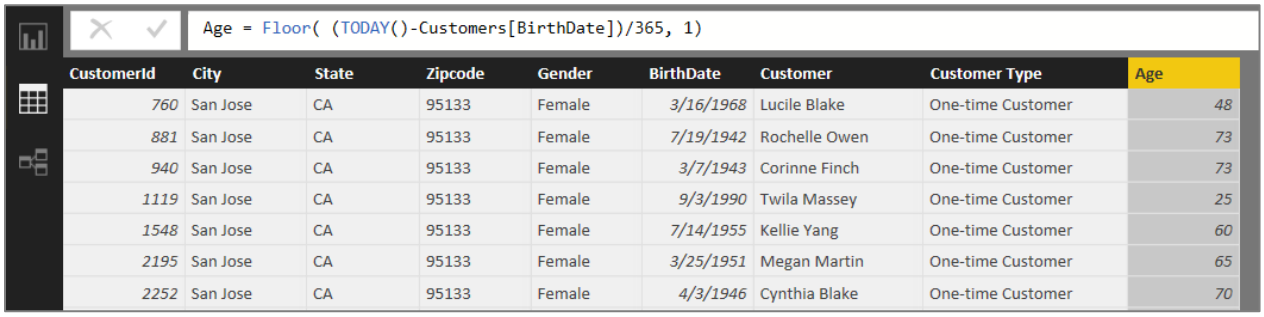
   e) You should now see that the cluster bar chart is showing additional bars for product cost and profit. Note you might have to resize the visual and make it a little taller to get rid of the scrollbars.



3. Add a calculated column to the **Customers** table named **Age** to indicate the age of the customer.
   a) Navigate to data view.
   b) Select the **Customers** table in the **Fields** list.
   c) Create a new calculated column by clicking the **New Column** button in the ribbon.
   d) Enter the following DAX expression into the formula bar to create the calculated column named **Age**.

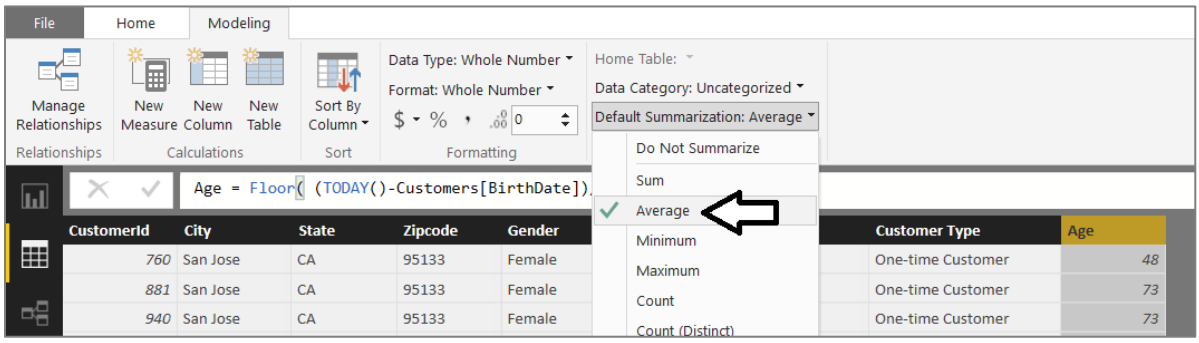
```
Age = Floor( (TODAY()-Customers[BirthDate])/365, 1)
```

e) Press the **ENTER** key to add the calculated column. You should be able to see a whole number for the age of each customer.



CustomerId	City	State	Zipcode	Gender	BirthDate	Customer	Customer Type	Age
760	San Jose	CA	95133	Female	3/16/1968	Lucile Blake	One-time Customer	48
881	San Jose	CA	95133	Female	7/19/1942	Rochelle Owen	One-time Customer	73
940	San Jose	CA	95133	Female	3/7/1943	Corinne Finch	One-time Customer	73
1119	San Jose	CA	95133	Female	9/3/1990	Twila Massey	One-time Customer	25
1548	San Jose	CA	95133	Female	7/14/1955	Kellie Yang	One-time Customer	60
2195	San Jose	CA	95133	Female	3/25/1951	Megan Martin	One-time Customer	65
2252	San Jose	CA	95133	Female	4/3/1946	Cynthia Blake	One-time Customer	70

f) Use the **Default Summarization** dropdown menu in the ribbon to change the default summarization setting for **Age** column from **Sum** to **Average**.



CustomerId	City	State	Zipcode	Gender	Customer Type	Age
760	San Jose	CA	95133	Female	One-time Customer	48
881	San Jose	CA	95133	Female	One-time Customer	73
940	San Jose	CA	95133	Female	One-time Customer	73

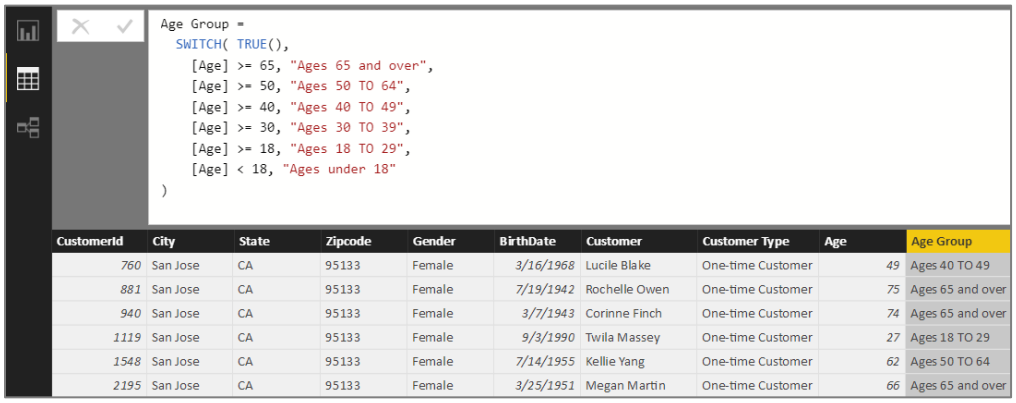
4. Add a calculated column to the **Customers** table named **Age Group** to break customers up into age-based sets.

a) Create a new calculated column in the **Sales** table by clicking the **New Column** button in the ribbon.

b) Enter to following DAX expression into the formula bar to create the calculated column named **Age Group**.

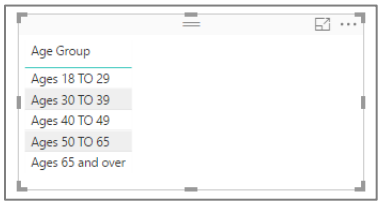
```
Age Group =
  SWITCH( TRUE(),
     [Age] >= 65, "Ages 65 and over",
     [Age] >= 50, "Ages 50 TO 64",
     [Age] >= 40, "Ages 40 TO 49",
     [Age] >= 30, "Ages 30 TO 39",
     [Age] >= 18, "Ages 18 TO 29",
     [Age] < 18, "Ages under 18"
)
```

c) After creating the calculated column, you should be able to verify that the **Age Group** column calculates a value for each customer row which places that customer in a bucket for a particular age group.



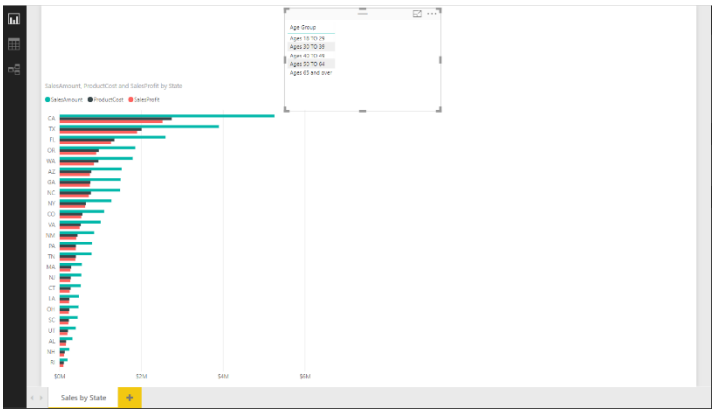
CustomerId	City	State	Zipcode	Gender	BirthDate	Customer	Customer Type	Age	Age Group
760	San Jose	CA	95133	Female	3/16/1968	Lucile Blake	One-time Customer	49	Ages 40 TO 49
881	San Jose	CA	95133	Female	7/19/1942	Rochelle Owen	One-time Customer	75	Ages 65 and over
940	San Jose	CA	95133	Female	3/7/1943	Corinne Finch	One-time Customer	74	Ages 65 and over
1119	San Jose	CA	95133	Female	9/3/1990	Twila Massey	One-time Customer	27	Ages 18 TO 29
1548	San Jose	CA	95133	Female	7/14/1955	Kellie Yang	One-time Customer	62	Ages 50 TO 64
2195	San Jose	CA	95133	Female	3/25/1951	Megan Martin	One-time Customer	66	Ages 65 and over

5. Use the **Age Group** calculated column as a row label in a matrix visual.
   a) Navigate to report view.
   b) Make sure that no visuals as selected on the page.
   c) Create a new **Table** visual in the report by selecting the checkbox for the **Age Group** column in the **Fields** list.

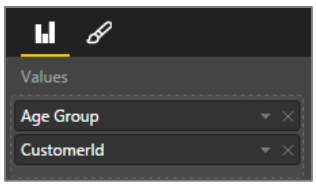


Note that this visual does not display a row for the age demographic value of **Age under 18** even though you added that value to the DAX formula. The reason for this is that the Wingtip Sales database you are working with does not contain any customers under 18 years of age so the age demographic value of **Age under 18** is automatically filtered out of the visual.

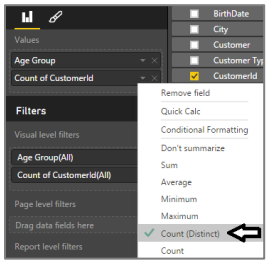
   d) Use the mouse to resize and reposition the visual so it appears in the top center of the page.



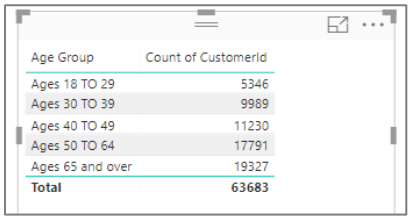
   e) Now it's time to modify the table visual by adding a new column to show the count of customers in each age group. Accomplish this by dragging the **CustomerId** column from the **Customers** table in the **Fields** list and dropping it into the **Values** well in the **Visualizations** pane. When you drop the **CustomerId** column into the **Values** well, make sure to drop it so it appears underneath the **Age Group** column.



   f) Click on the dropdown menu of the **CustomerId** field inside the **Values** well so you can see and change the aggregation that will be performed. Select the aggregation type that is **Count (Distinct)**.

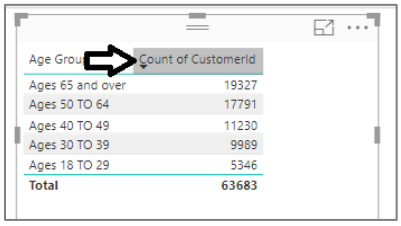


g) Now the new visual should display one row per age group which includes a total count of customers.



Age Group	Count of CustomerId
Ages 18 TO 29	5346
Ages 30 TO 39	9989
Ages 40 TO 49	11230
Ages 50 TO 64	17791
Ages 65 and over	19327
Total	**63683**

h) Click on the column header for the **Count of CustomerId** column to sort the age groups with most customers to the top.

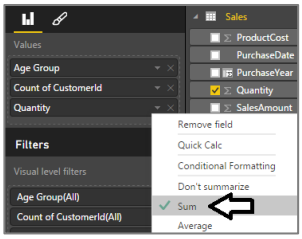


Age Group	Count of CustomerId
Ages 65 and over	19327
Ages 50 TO 64	17791
Ages 40 TO 49	11230
Ages 30 TO 39	9989
Ages 18 TO 29	5346
Total	**63683**

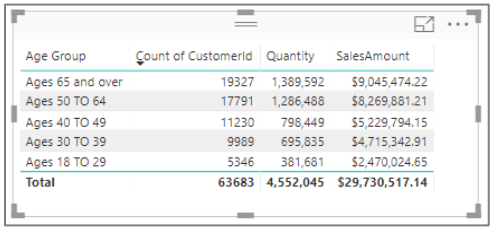
i) Drag the **Quantity** column from the **Sales** table and drop it into the **Values** well in the **Visualizations** pane.

j) Drag the **SalesAmount** column from the **Sales** table and drop it into the **Values** well in the **Visualizations** pane.

k) Configure the summarization type for the **Quantity** field and the **SalesAmount** field to perform a **Sum**.

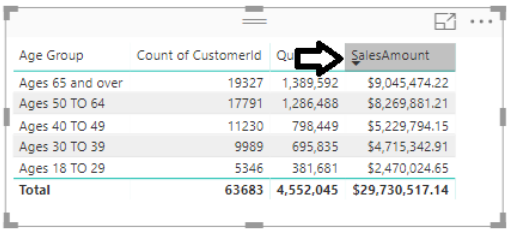


l) The visual should now display a new **Quantity** column as well as a **SalesAmount** column displaying aggregated totals.



Age Group	Count of CustomerId	Quantity	SalesAmount
Ages 65 and over	19327	1,389,592	$9,045,474.22
Ages 50 TO 64	17791	1,286,488	$8,269,881.21
Ages 40 TO 49	11230	798,449	$5,229,794.15
Ages 30 TO 39	9989	695,835	$4,715,342.91
Ages 18 TO 29	5346	381,681	$2,470,024.65
Total	**63683**	**4,552,045**	**$29,730,517.14**

m) Sort the rows in the table visual by clicking in the column header for the **SalesAmount** column so that the **Age Group** with the greatest amount of sales revenue is sorted to the top.



Age Group	Count of CustomerId	Qu	SalesAmount
Ages 65 and over	19327	1,389,592	$9,045,474.22
Ages 50 TO 64	17791	1,286,488	$8,269,881.21
Ages 40 TO 49	11230	798,449	$5,229,794.15
Ages 30 TO 39	9989	695,835	$4,715,342.91
Ages 18 TO 29	5346	381,681	$2,470,024.65
Total	**63683**	**4,552,045**	**$29,730,517.14**

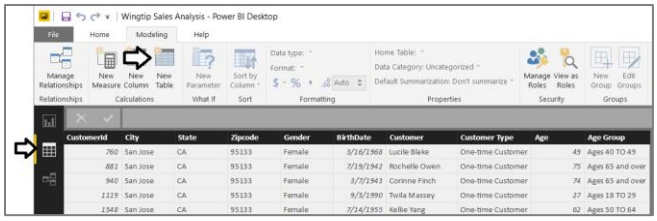
n) Save the work you have done by clicking the Save button in the upper left corner of the Power BI Desktop window.

At this point, you might be bothered by the fact that some of the column header names in this visual are not as pretty as they could be. For example, it would be less confusing to business users if the column displaying a count of customers had a column heading of **Customer Count** instead of **Count of CustomerId**. You will address this issue later in the lab exercise where you begin to extend the data model by creating measures.

Exercise 4: Extending the Data Model with a Lookup Table

In this exercise, you will extend the data model by adding a new lookup table named **SalesRegions** which assigns a geographic sales region to each state. The work to accomplish this will involve dynamically creating a new table named **SalesRegions** using a DAX expression. Once you have created the **SalesRegions** table, you must then create a new relationship between the **SalesRegions** table and the **Customers** table to integrate the new tables into the project's data model. At the end of this exercise, you will also create two new calculated columns in the **Customers** table to pull in related data from the **SalesRegions** table.

1. Create the **SalesRegions** table using a DAX expression.
   a) Navigate to Data view in the **Wingtip Sales Analysis.pbix** project.
   b) Click the **New Table** button to create a new table using DAX.



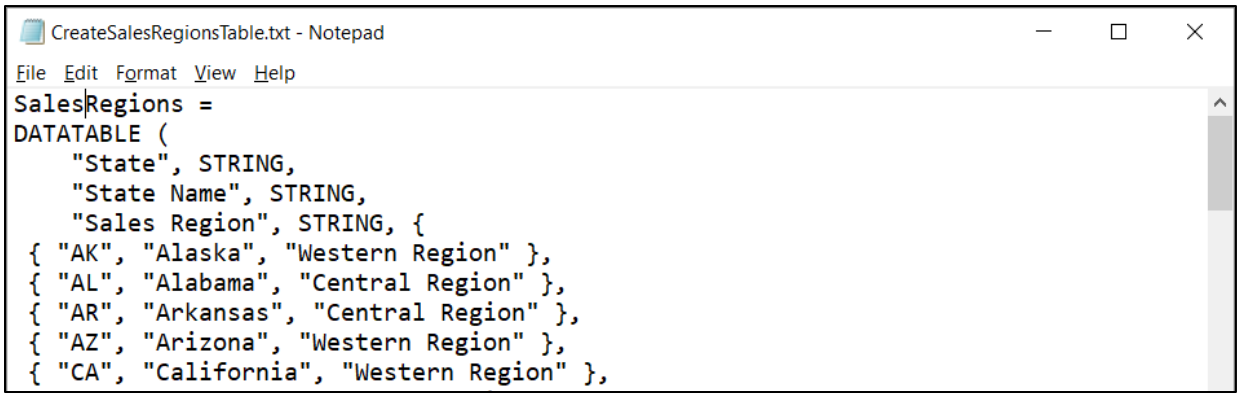
   c) You should now be able to add the DAX expression to create the new table.

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.

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

```
C:\Student\Modules\03_DataModeling\Lab\DAX\CreateSalesRegionsTable.txt
```

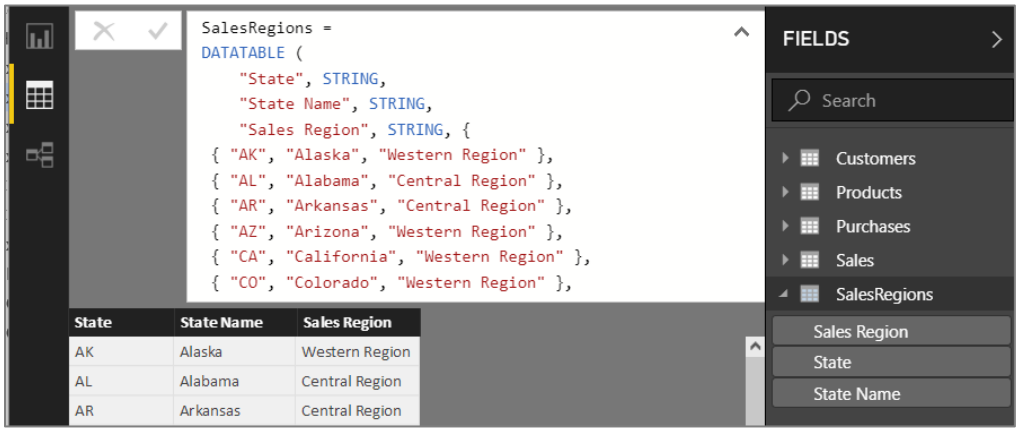
   e) Take a moment to inspect the DAX expression inside **CreateSalesRegionsTable.txt**.



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

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

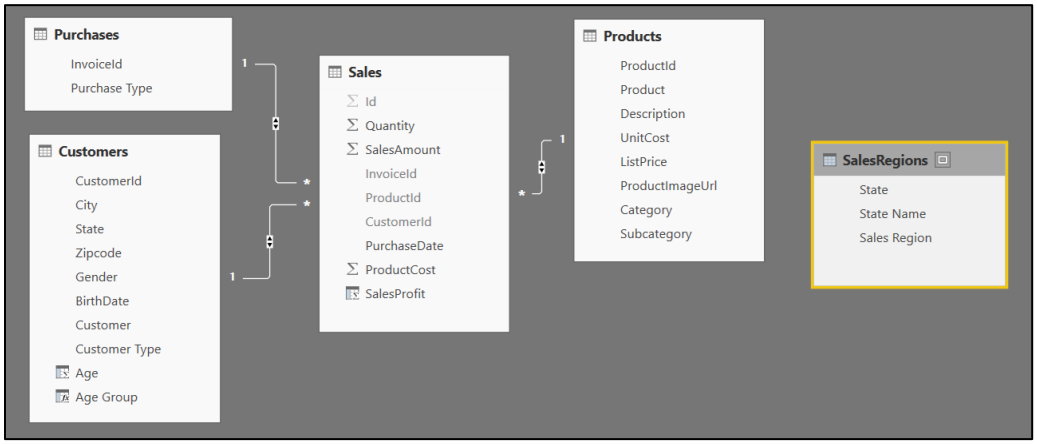
h) Press the **ENTER** key to create the new table named **SalesRegions**.



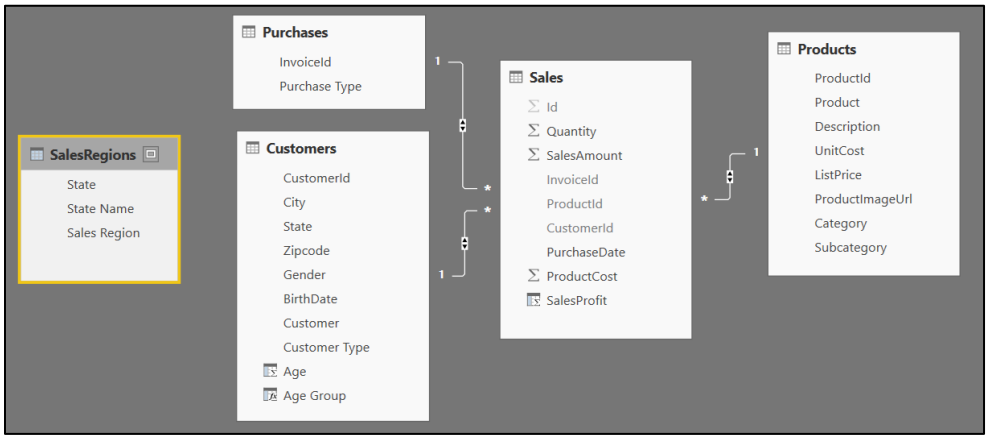
2. Create a table relationship between the **SalesRegions** table and the **Customers** table.

a) Navigate to Relationship view.

b) You should be able to see the new **SalesRegions** table on the right.



c) Rearrange layout of tables by moving **SalesRegions** to the left and all the other tables to the right.



d) Create a new table relationship between **SalesRegions** and **Customers** by clicking and dragging the **State** column from the **SalesRegions** table and dropping it on top of the **State** column in the **Customers** table.

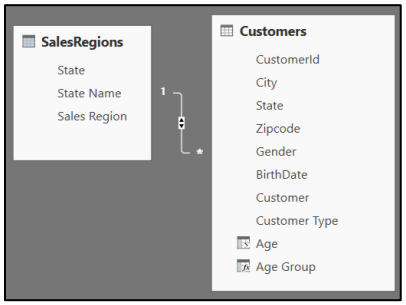
e) You should be able to see that a relationship has already been created between **SalesRegions** and **Customers**.

f) Double-click the line that connects the **SalesRegions** table to the **Customers** table to display the **Edit relationship** dialog.

g) Set the **Cross filter direction** property for the relationship to **Both**.

h) Click the OK button to save your changes and close the **Edit relationship** dialog.

i) Verify that a bidirectional relationship has been created between **SalesRegions** and **Customers**.



3. Add a calculated column to the **Customers** table named **Sales Region** to display the sales region for each state.

a) Navigate to data view.

b) Select the **Customers** table in the **Fields** list.

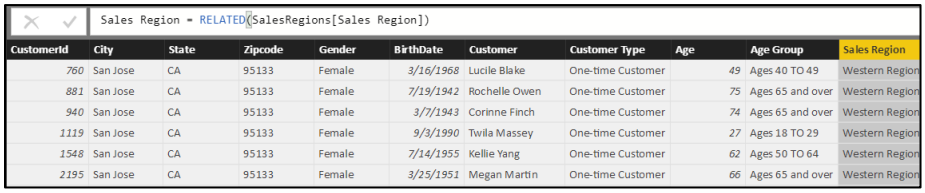
c) Create a new calculated column by clicking the **New Column** button in the ribbon.

Since a relationship exists between **SalesRegions** and **Customers**, you can use the **RELATED** function provided by DAX to create calculated columns in the **Customers** table that pull in data from the **SalesRegions** table.

d) Enter to following DAX expression into the formula bar to create the calculated column named **Sales Region**.

```
Sales Region = RELATED(SalesRegions[Sales Region])
```

e) Press the **ENTER** key to add the calculated column to the table. You should be able to see a value in the **Sales Region** column for each row in the **Customers** table.



CustomerId	City	State	Zipcode	Gender	BirthDate	Customer	Customer Type	Age	Age Group	Sales Region
760	San Jose	CA	95133	Female	3/16/1968	Lucile Blake	One-time Customer	49	Ages 40 TO 49	Western Region
881	San Jose	CA	95133	Female	7/19/1942	Rochelle Owen	One-time Customer	75	Ages 65 and over	Western Region
940	San Jose	CA	95133	Female	3/7/1943	Corinne Finch	One-time Customer	74	Ages 65 and over	Western Region
1119	San Jose	CA	95133	Female	9/3/1990	Twila Massey	One-time Customer	27	Ages 18 TO 29	Western Region
1548	San Jose	CA	95133	Female	7/14/1955	Kellie Yang	One-time Customer	62	Ages 50 TO 64	Western Region
2195	San Jose	CA	95133	Female	3/25/1951	Megan Martin	One-time Customer	66	Ages 65 and over	Western Region

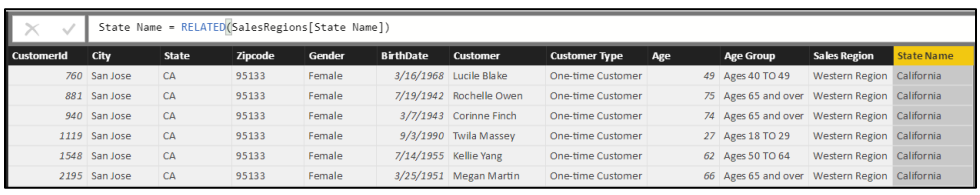
4. Add a calculated column to the **Customers** table named **State Name** to display the full state name.

a) Create a new calculated column in the **Customers** table by clicking the **New Column** button in the ribbon.

b) Enter to following DAX expression into the formula bar to create the calculated column named **State Name**.

```
State Name = RELATED(SalesRegions[State Name])
```

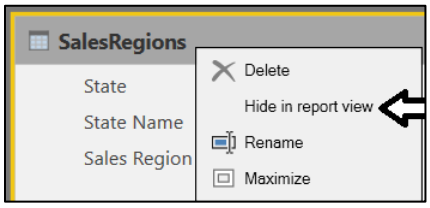
c) Press the **ENTER** key to add the calculated column to the table. You should be able to see a value in the **Sales Region** column for each row in the **Customers** table.



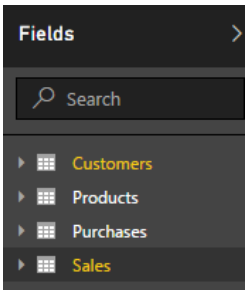
CustomerId	City	State	Zipcode	Gender	BirthDate	Customer	Customer Type	Age	Age Group	Sales Region	State Name
760	San Jose	CA	95133	Female	3/16/1968	Lucile Blake	One-time Customer	49	Ages 40 TO 49	Western Region	California
881	San Jose	CA	95133	Female	7/19/1942	Rochelle Owen	One-time Customer	75	Ages 65 and over	Western Region	California
940	San Jose	CA	95133	Female	3/7/1943	Corinne Finch	One-time Customer	74	Ages 65 and over	Western Region	California
1119	San Jose	CA	95133	Female	9/3/1990	Twila Massey	One-time Customer	27	Ages 18 TO 29	Western Region	California
1548	San Jose	CA	95133	Female	7/14/1955	Kellie Yang	One-time Customer	62	Ages 50 TO 64	Western Region	California
2195	San Jose	CA	95133	Female	3/25/1951	Megan Martin	One-time Customer	66	Ages 65 and over	Western Region	California

Now that you have pulled all the important data from the **SalesRegions** table into the **Customers** table, there is no need to display the **SalesRegions** table in report view. In the next step, you will hide the **SaleRegions** table to simplify view of the data model that is shown in report view.

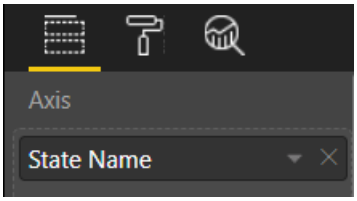
5. Hide the **SalesRegions** table from report view.
   a) Navigate to relationship view.
   b) Right-click on the the **SalesRegions** table



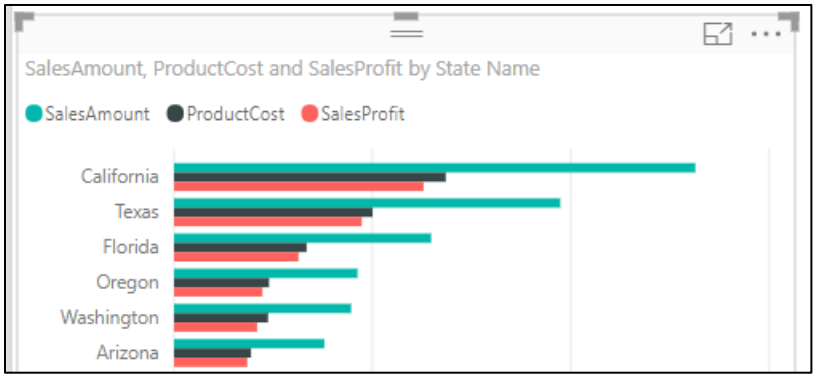
   c) Navigate to Report view and verify that the **SalesRegions** table is not displayed as one of the tables in the **Fields** list.



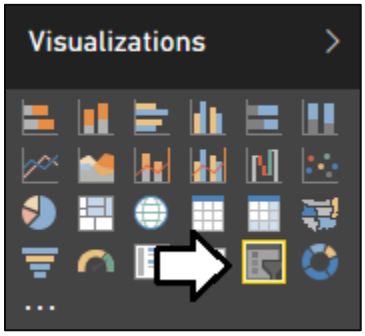
6. Update the Clustered bar chart visual to display axis labels based on the **State Name** column instead of the **State** column.
   a) Navigate back to report view.
   b) Select the Clustered bar chart visual which is currently displaying axis labels based on the **State** column.
   c) Remove the **State** field from the **Axis** well in the **Visualization** pane and replace it with the **State Name** column.



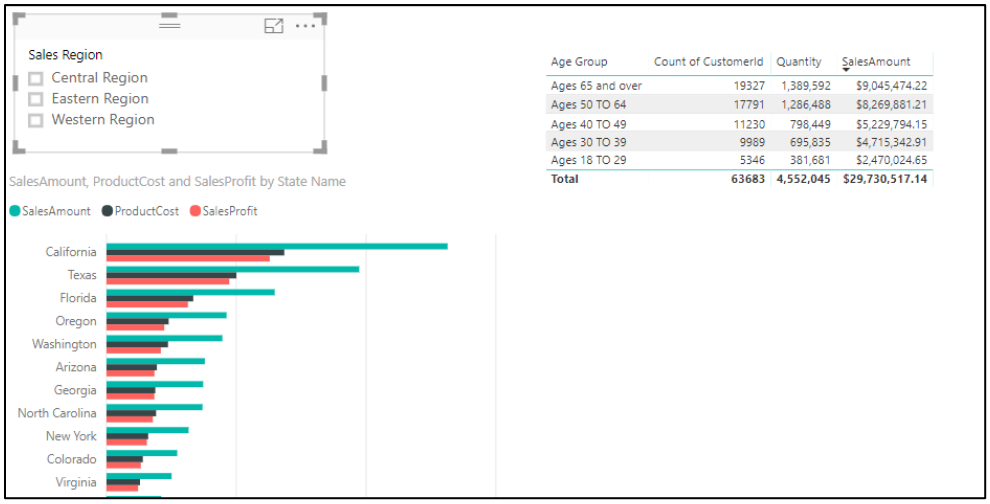
   d) The visual should now display the full state name instead of the two-character state abbreviation.



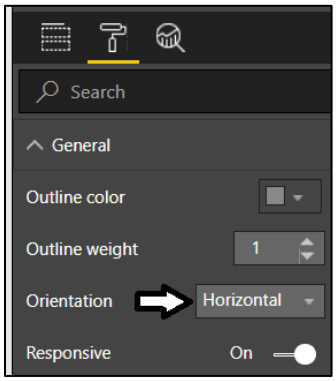
7. Add a slicer visual to the report to filter all displayed results by sales region.
   a) Navigate to report view if you are not already there.
   b) Make sure that no visuals are selected on the page so that you can create a new visual.
   c) Select the checkbox for the **Sales Region** column in the **Fields** list to create a new visual.
   d) Click the **Slicer** button in the **Visualizations** list to change the visual type to a slicer.



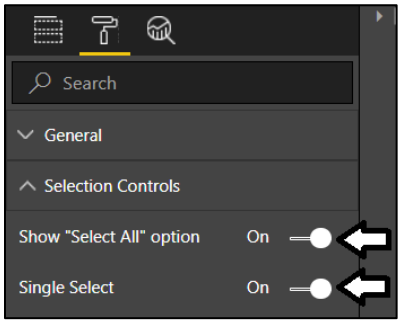
   e) Using the mouse, reposition the slicer visual so it appears in the upper, left-hand corner of the page as shown in the following screenshot.



   f) Change the alignment of the slicer visual from vertical to horizontal. Accomplishing this by first clicking the Edit Pen button in the **Visualization** pane to view the appearance properties of the visual and then by modifying the **Orientation** property to a value of **Horizontal**.

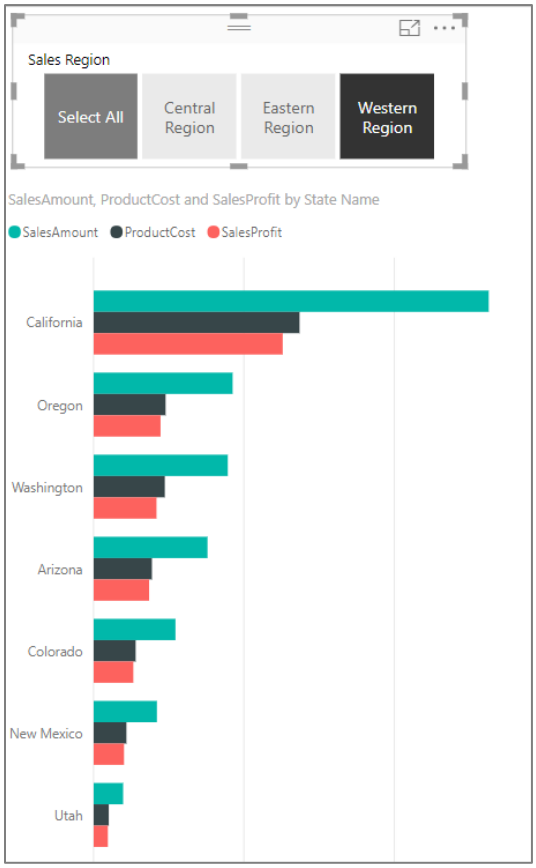


g) Enable the **Select All** property in the **Selection Control** section by setting its value to **On**.



h) The slicer visual should now appear with a horizontal layout with an additional **Select All** node.

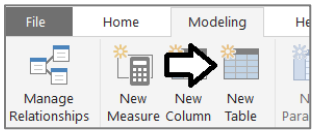
i) Click on the different sales region nodes of the slicer visual to see its filtering effect. If you click on the **Western Region** node, all the other visuals apply a filter to only shows states assigned to the western sales region.



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

Oh no! There's a problem with the default sort order of the **Sales Region** column. The column values are currently sorted alphabetically so **Central Region** is first, **Easter Region** is second and **Western Region** is third. However, your manager has asked you to implement a customized sort order so that **Western Region** appears first, **Central Region** is second and **Eastern Region** is third. To solve this problem, you will create a second table named **SalesRegionsSort** and use it to implement a custom sort order.

9. Create the **SalesRegionsSort** table using a DAX expression.
   a) Navigate to Data view in the **Wingtip Sales Analysis.pbix** project.
   b) Click the **New Table** button to create a new table using DAX.



   c) You should now be in editing mode where you can add the DAX expression to create the new table.
   d) Locate the file named **CreateSalesRegionsSortTable.txt** at the following path and open it with Windows Notepad.

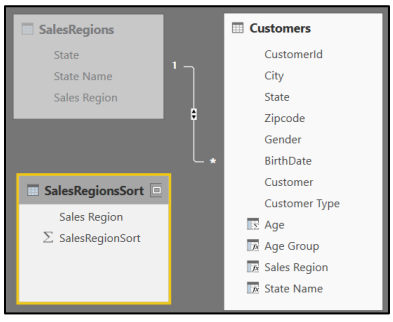
```
C:\Student\Modules\03_DataModeling\Lab\DAX\CreateSalesRegionsSortTable.txt
```

   e) Take a moment to inspect the DAX expression inside **CreateSalesRegionsSortTable.txt**.

```
SalesRegionsSort =
DATATABLE ("Sales Region", STRING, "SalesRegionSort", INTEGER, {
 { "Western Region", 1 },
 { "Central Region", 2 },
 { "Eastern Region", 3 }
})
```

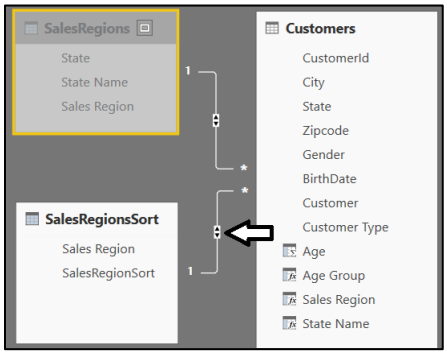
   f) Select the entire contents of **CreateSalesRegionsSortTable.txt** and then copy it into the Windows clipboard.
   g) Press the ENTER key to submit your DAX expression and to create the new table.

10. Create a relationship between the **SalesRegionsSort** table and the **Customers** table.
   a) Navigate back to Relationship view.
   b) Verify you can see the new table named **SalesRegionsSort**.
   c) Using the mouse, move the **SalesRegionsSort** table underneath the **SalesRegions** table to the left of the **Customers** table.



d) Drag the **Sales Region** column from **SalesRegions** and drop it on the **Sales Region** column in **Customers**.

e) Edit this new table relationship so that its **Cross filter direction** is set to **Both**.



11. Add the **SalesRegionSortOrder** column to the **Customers** table as a calculated field.

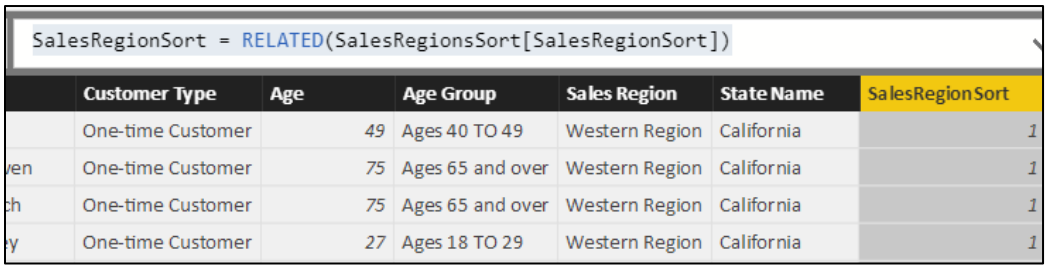
a) Navigate to Data view mode in the Power BI Desktop application window.

b) Select the **Customers** table in the **Fields** list.

c) Click the **New Column** button in the ribbon to create a new calculate column.

d) Type in the following DAX expression to create a new calculated column named **SalesRegionSort**.

```
SalesRegionSort = RELATED(SalesRegionsSort[SalesRegionSort])
```

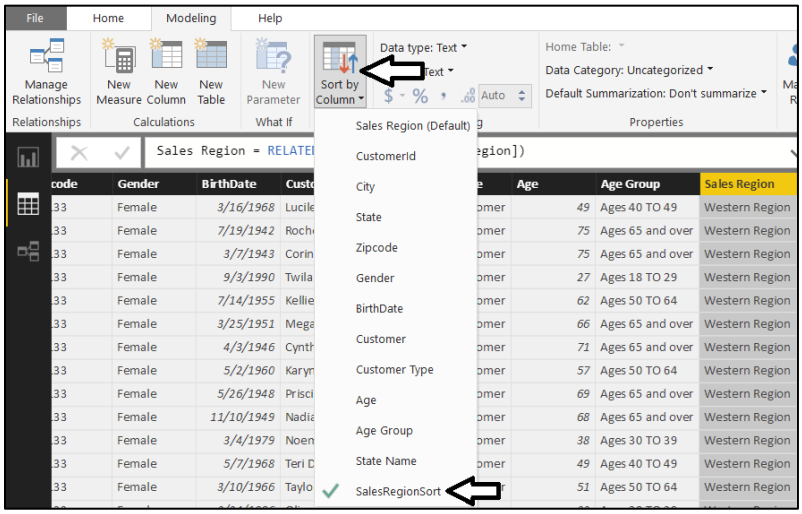


	Customer Type	Age	Age Group	Sales Region	State Name	SalesRegionSort
	One-time Customer	49	Ages 40 TO 49	Western Region	California	1
ven	One-time Customer	75	Ages 65 and over	Western Region	California	1
ch	One-time Customer	75	Ages 65 and over	Western Region	California	1
y	One-time Customer	27	Ages 18 TO 29	Western Region	California	1

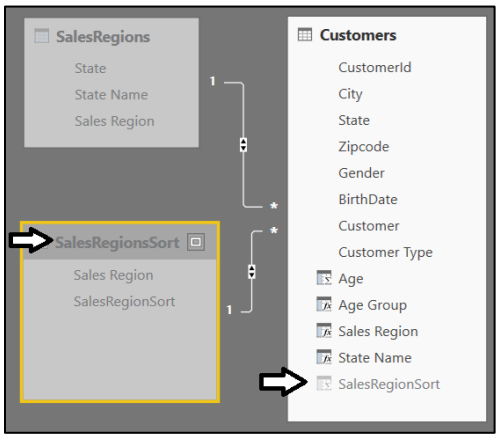
12. Configure a custom sort column for the **Sales Region** field.

a) Select the **Sales Region** field by clicking its column header.

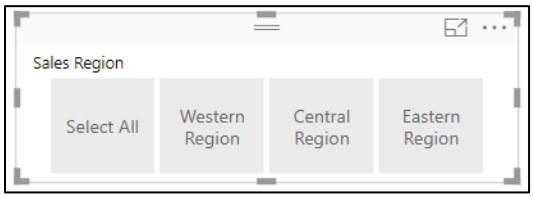
b) Drop by the **Sort By Column** menu in the ribbon and select the **SalesRegionSort** column.



13. Simply your data model by hiding the implementation details of your custom sort order from report view.
    a) Navigate to Relationship view.
    b) Hide the **SalesRegionsSort** table by right-clicking it and selecting **Hide in Report View**.
    c) Hide the **SalesRegionSort** column in the **Customer** table by right-clicking it and selecting **Hide in Report View**.



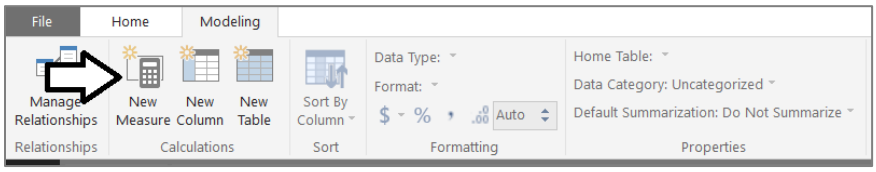
14. Verify your custom sort order is working properly.
    a) Return to Report View and examine the slicer which shows the sales regions.
    b) Verify that the sort order now has **Western Sales** first followed by **Central Region** and then **Eastern Region**.



Exercise 5: Extending the Data Model by Creating Measures

In this exercise you will create four measures named **Units Sold, Sales Revenue, Product Cost** and **Profit** that will perform sum aggregations on the **Sales** table. You will also create a measure named **Customer Count** that performs a distinct count aggregation on the **Customers** table. As you will see, creating measures to use in your report visuals will give you much greater control over the column names, formatting and aggregations that are displayed in your reports.

1. Create a measure named **Units Sold** to perform a sum aggregation on the **Quantity** 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.

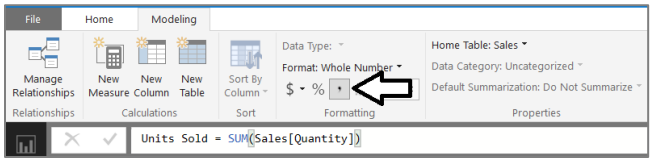


    d) Enter to following DAX expression into the formula bar to create the measure named **Units Sold**.

```
Units Sold = SUM(Sales[Quantity])
```

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

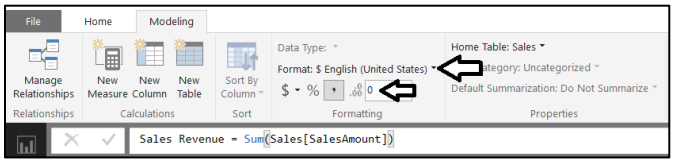
f) Modify the formatting by clicking and selecting the Comma button on the ribbon to add a comma separator.



2. Create a measure named **Sales Revenue** to perform a sum aggregation on the **SalesAmount** column of the **Sales** table.
   a) Create a new measure by clicking the **New Measure** button in the ribbon.
   b) Enter to following DAX expression into the formula bar to create the measure named **Sales Revenue**.

```
Sales Revenue = Sum(Sales[SalesAmount])
```

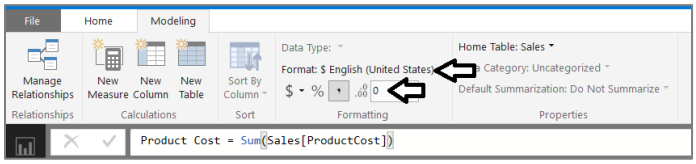
   c) Press the **ENTER** key to add the measure to data model.
   d) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Currency > $ English (United States)**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to zero.



3. Create a measure named **Product Cost** to perform a sum aggregation on the **ProductCost** column of the **Sales** table.
   a) Create a new measure by clicking the **New Measure** button in the ribbon.
   b) Enter to following DAX expression into the formula bar to create the measure named **Product Cost**.

```
Product Cost = Sum(Sales[ProductCost])
```

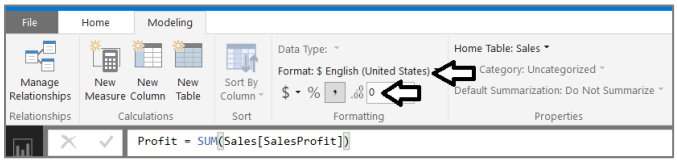
   c) Press the **ENTER** key to add the measure to data model.
   d) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Currency > English (United States)**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to zero.



4. Create a measure named **Profit** to perform a sum aggregation on the **SalesProfit** column of the **Sales** table.
   a) Create a new measure by clicking the **New Measure** button in the ribbon.
   b) Enter to following DAX expression into the formula bar to create the measure named **Profit**.

```
Profit = SUM(Sales[SalesProfit])
```

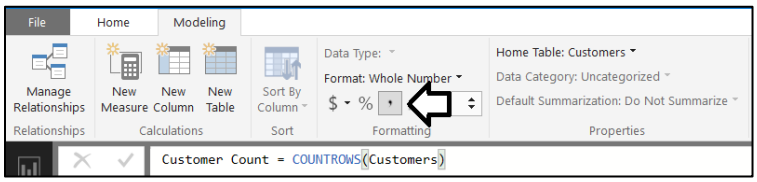
   c) Press the **ENTER** key to add the measure to data model.
   d) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Currency > English (United States)**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to zero.



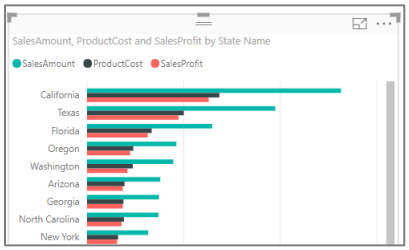
5. Create a measure named **Customer Count** to perform an aggregation to count the number of rows in the **Customers** table.
   a) Make sure the **Sales** table is selected.
   b) Create a new measure by clicking the **New Measure** button in the ribbon.
   c) Enter the following DAX expression into the formula bar to create the measure named **Customer Count**.

```
Customer Count = COUNTROWS(Customers)
```

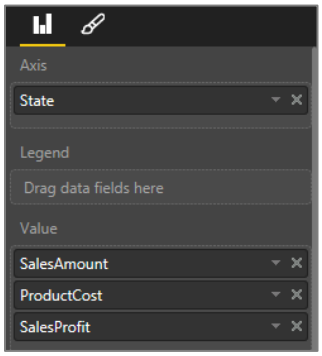
   d) Press the **ENTER** key to add the measure to data model.
   e) Modify the formatting by clicking and selecting the Comma button on the ribbon to add a comma separator.



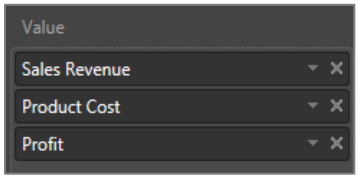
6. Update the visuals in the report to use the new measures you have just created.
   a) Navigate to report view.
   b) Select the clustered bar chart visual.



   c) Examine the properties of this visual in the **Visualizations** pane. The **Value** well should currently contain the three fields named **SalesAmount**, **ProductCost** and **SalesProfit**.



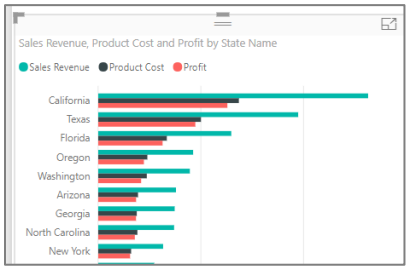
   d) Remove those three fields from the **Value** well and replace them with the new measures named **Sales Revenue**, **Product Cost** and **Profit**.



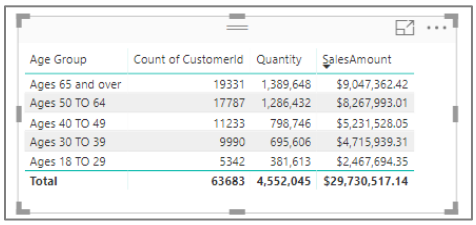
You will notice that the sort order of this visual is based on the **State** field because you removed the **SalesAmount** field.

e) Resort the rows in the visual using **Sales Revenue**.

f) Now this visual looks more polished because it is using measures instead of aggregated columns to produce its values.

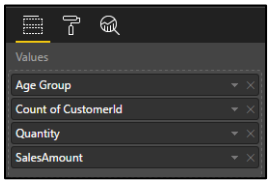


g) Select the table visual which displays a row for each age group and notice how some of the column names contain text) that is not very user-friendly such as **Count of CustomerId**.

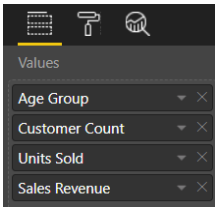


Age Group	Count of CustomerId	Quantity	SalesAmount
Ages 65 and over	19331	1,389,648	$9,047,362.42
Ages 50 TO 64	17787	1,286,432	$8,267,993.01
Ages 40 TO 49	11233	798,746	$5,231,528.05
Ages 30 TO 39	9990	695,606	$4,715,939.31
Ages 18 TO 29	5342	381,613	$2,467,694.35
Total	**63683**	**4,552,045**	**$29,730,517.14**

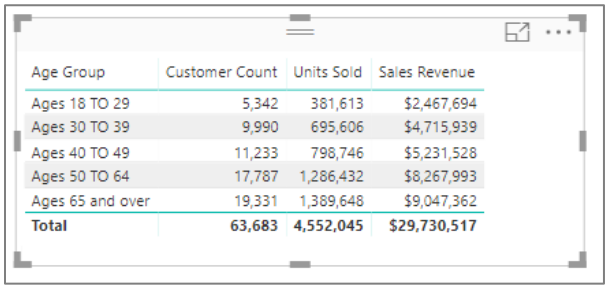
h) Examine the properties of this visual in the **Visualizations** pane. The **Value** well should currently contain the four fields named **Age Group**, **Count of CustomerId**, **Quantity** and **SalesAmount**.



i) Remove all fields from the **Value** well except for Age Group and then add the new measures named **Customer Count**, **Units Sold** and **Sales Revenue**.

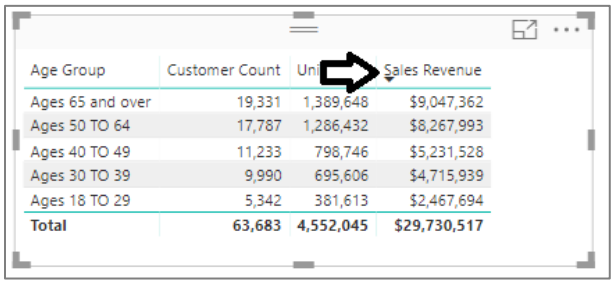


j) The columns of the table visual are now more user-friendly and the formatting of values looks better as well.



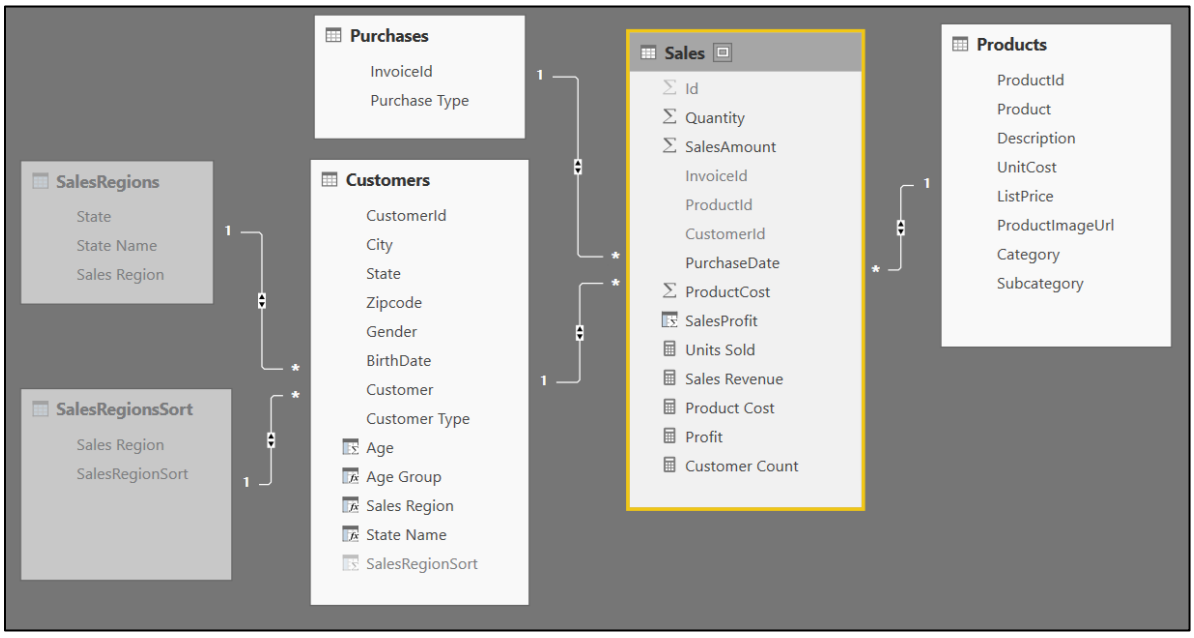
Age Group	Customer Count	Units Sold	Sales Revenue
Ages 18 TO 29	5,342	381,613	$2,467,694
Ages 30 TO 39	9,990	695,606	$4,715,939
Ages 40 TO 49	11,233	798,746	$5,231,528
Ages 50 TO 64	17,787	1,286,432	$8,267,993
Ages 65 and over	19,331	1,389,648	$9,047,362
Total	**63,683**	**4,552,045**	**$29,730,517**

k) Click on the **Sales Revenue** column header to sort the age groups by sales revenue.



Age Group	Customer Count	Un	Sales Revenue
Ages 65 and over	19,331	1,389,648	$9,047,362
Ages 50 TO 64	17,787	1,286,432	$8,267,993
Ages 40 TO 49	11,233	798,746	$5,231,528
Ages 30 TO 39	9,990	695,606	$4,715,939
Ages 18 TO 29	5,342	381,613	$2,467,694
Total	**63,683**	**4,552,045**	**$29,730,517**

7. You will complete your work in this exercise by cleaning up the data model by hiding fields in report view.
   a) Navigate to Relationship View so you can see all the tables in the project's data model.
   b) Resize each table so that is properly displays all its fields.

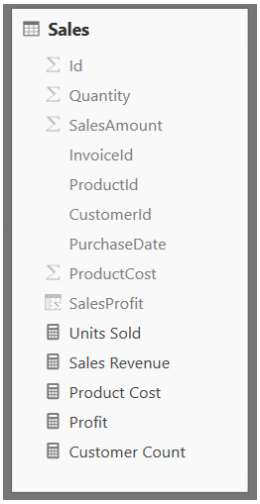


   c) Use the **Hide in Report View** command to hide the following columns.
      i) The **InvoiceId** column in the **Purchases** table.
      ii) The **CustomerId** column and the **BirthDate** column from the **Customers** table.
      iii) The **ProductId** column, the **UnitCost** column and the **ListPrice** column from the **Products** table.

When you hide all the fields in a table except for its measures, Power BI Desktop treats this table as a **fact table**. The main effect this has in Power BI Desktop is that the table will be displayed above in the Fields list when in report view. However, Power BI Desktop does not refresh the view until you close and reopen the Fields list. In the next step, you will close and reopen the Fields list to see the effects of creating a tables which only displays measures in Report View.

a) Configure the **Sales** table as a fact table by hiding every field except for the 5 measures named **Units Sold**, **Sales Revenue**, **Product Cost**, **Profit** and **Customer Count**.

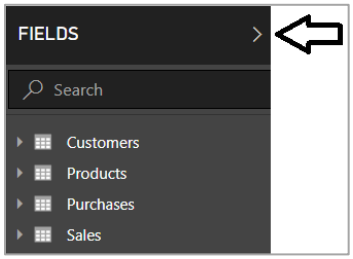
b) Verify that all fields in the Sales table are hidden except for the 5 measures.



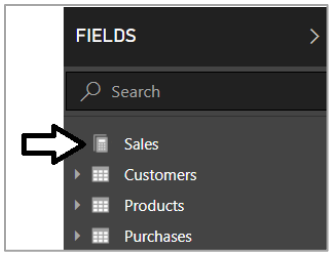
8. Close and reopen the **Fields** list to see the effects of a measure-only table.

a) Return to Report view.

b) Locate the arrow menu in the top right corner of the **Fields** list and click it twice to close and reopen the **Fields** list.



c) Once the **Fields** list has been refreshed, you should see that the **Sales** table has been moved to the top and has been given a calculator icon to distinguish it from the other visible tables in the data model which are acting as dimension tables.



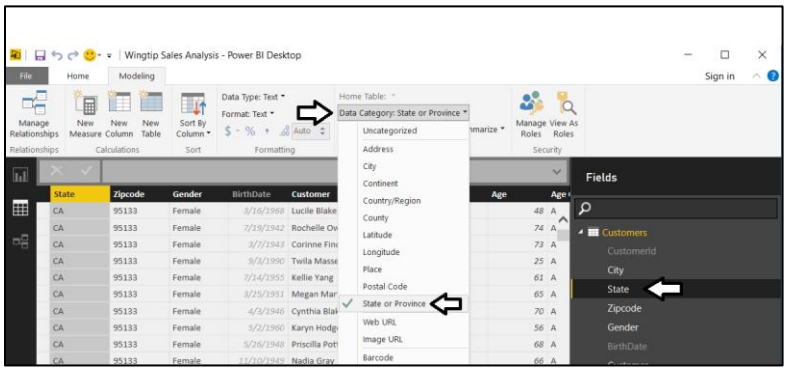
9. Save the current project by clicking the **Save** button in the ribbon.

While this exercise was not complicated, it's important that you know how to simplify a data model so it's easier for others to use.

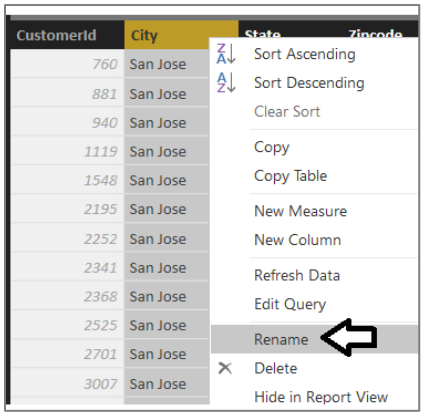
Exercise 6: Extending the Data Model with Geolocation Metadata

In this exercise, you will configure geolocation metadata to fields in the **Customers** table including **State**, **City** and **Zipcode**. After that, you will create a new report page with a map visual to visualize how customer sales data is spread out across various geographic regions within the United States.

1. Configure Geolocation Metadata for the **State** field in the **Customers** table.
   a) Return to Data View.
   b) From the **Fields** list on the right, select the **State** field from the **Customers** table.
   c) Drop down the **Data Category** menu from the ribbon and select **State or Province**.



2. Modify the **City** column to contain the state as well as the city to remove ambiguity when determining geographic locations.
   a) Select the **Customers** table in the **Fields** list.
   b) Right-click on the column header for the **City** column and select the **Rename** command.



   c) Rename the column to **City Name**.

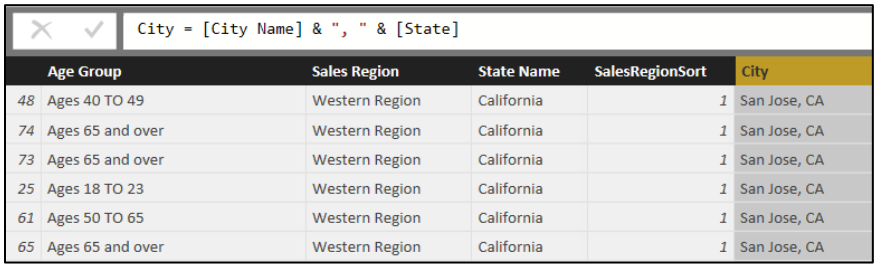
CustomerId	City Name	State	Zipcode
760	San Jose	CA	95133
881	San Jose	CA	95133
940	San Jose	CA	95133
1119	San Jose	CA	95133
1548	San Jose	CA	95133

   d) Navigate to the **Modeling** tab in the ribbon and make sure the Customers table is still selected.
   e) Click the **New Column** tab to create a new calculated column.

f) Add in the following DAX expression to create a new column named **City**.

```
City = [City Name] & ", " & [State]
```

g) The **City** column should display the city and the state which will remove ambiguity when used as a geolocation field.

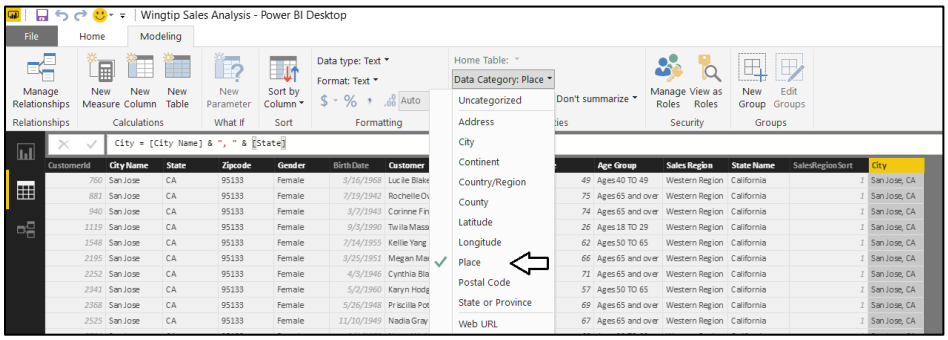


City = [City Name] & ", " & [State]

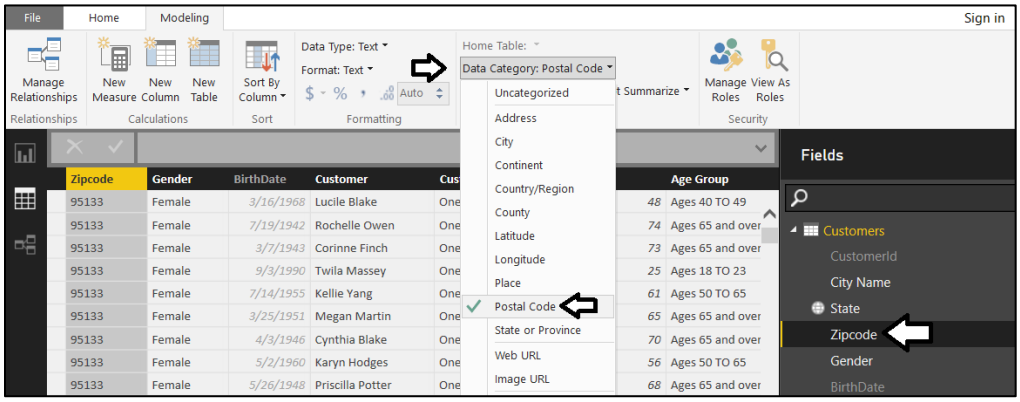
	Age Group	Sales Region	State Name	SalesRegionSort	City
48	Ages 40 TO 49	Western Region	California	1	San Jose, CA
74	Ages 65 and over	Western Region	California	1	San Jose, CA
73	Ages 65 and over	Western Region	California	1	San Jose, CA
25	Ages 18 TO 23	Western Region	California	1	San Jose, CA
61	Ages 50 TO 65	Western Region	California	1	San Jose, CA
65	Ages 65 and over	Western Region	California	1	San Jose, CA

Certain aspects of working with Power BI are not as intuitive as they could be. For example, you need to configure the **Data Category** for the **City** column which contains values such as **Tampa, FL** and **Austin, TX**. The obvious choice of setting the **Data Category** to **City** does not work as expected. Instead, you must set the **Data Category** setting to **Place** for the visual mapping to work correctly.

3. Configure geolocation metadata for the **City** field in the **Customers** table.
   a) Return to Data View.
   b) From the **Fields** list on the right, select the **City** field from the **Customers** table.
   c) Drop down the **Data Category** menu from the ribbon and select **Place**.

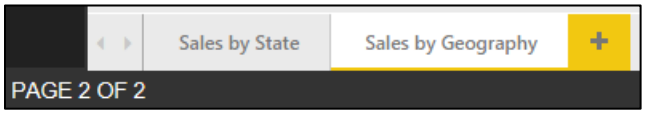


4. Configure geolocation metadata for the **Zipcode** field in the **Customers** table.
   a) Return to Data View.
   b) From the **Fields** list on the right, select the **Zipcode** field from the **Customers** table.
   c) Drop down the **Data Category** menu from the ribbon and select **Postal Code**.

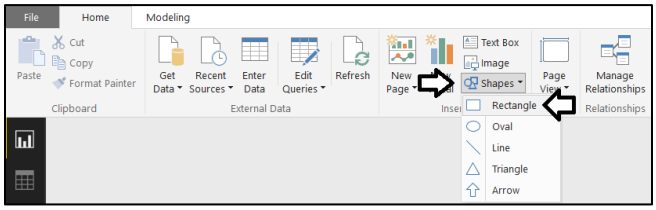


Now that you have configured fields in the **Customers** table with geolocation metadata, it's time to put this metadata to use by creating a new report page with a map visual to visualize how sales revenue is distributed over geographic regions.

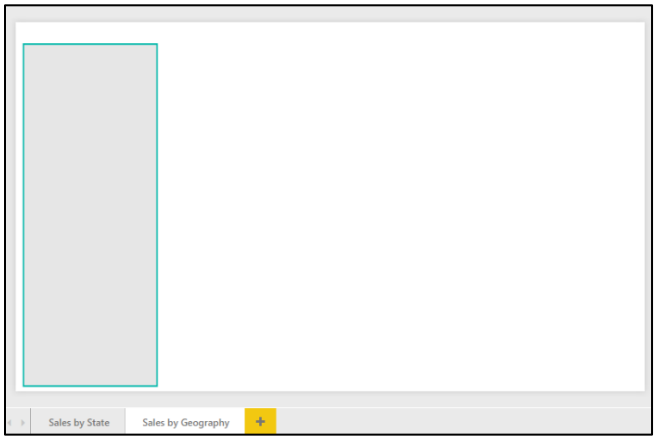
5. Create a new report page.
   a) Return to Report View.
   b) Click on the (**+**) button on the page navigation tab to create a new page.
   c) Rename the new page to **Sales by Geography**.



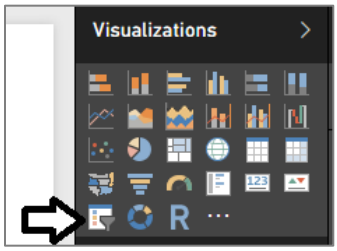
6. Add a background **Rectangle** shape to the report for formatting purposes.
   a) Navigate to the **Home** tab in the ribbon.
   b) Drop down the Shapes menu in the ribbon and select Rectangle to add a new rectangle shape to the report.



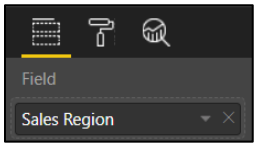
   c) Using the mouse, reposition the rectangle shape so it takes up the full height of the report page and about 20% of the width as shown in the following screenshot.



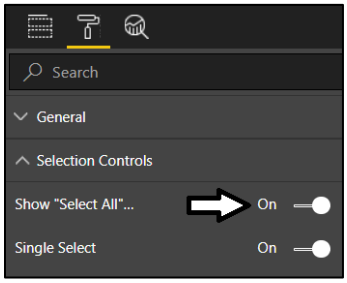
7. Create a new slicer visual to filter by sales region.
   a) Click the **Slicer** button in the **Visualizations** list to add a new slicer visual to the report.



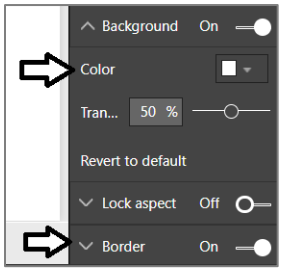
b) Add the **Sales Region** field from the **Customers** table into the **Field** well for the slicer.



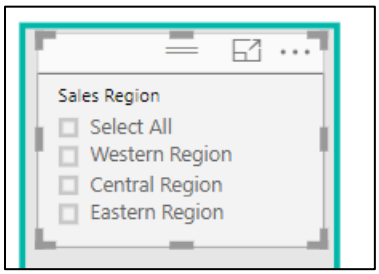
c) With the slicer selected, navigate to the **Selection Controls** sections in the **Format** properties pane and set **Select All** to **On**.



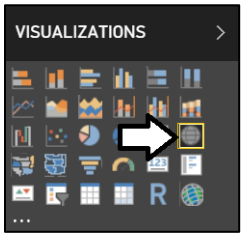
d) Move down in the Format pane and configure the slicer with a background color of white and a border as shown in the following screenshot.



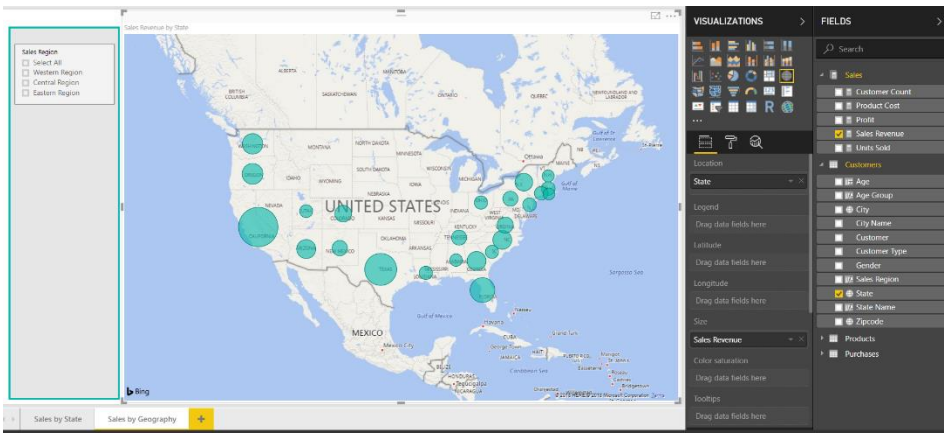
e) Your slicer visual should look like the one shown in the following screenshot.



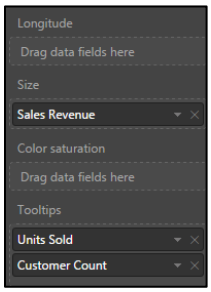
8. Add a new Map visual to display sales revenue by state.
   a) Click the **New Visual** button on the ribbon to create a new visual.
   b) Click the **Map** button in the **Visualizations** list to change the visual into a Map visual.



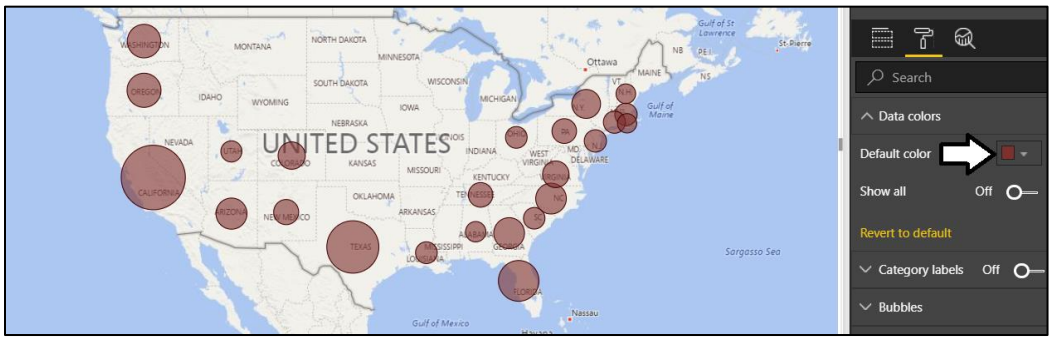
c) Configure the fields for the **Map** visual by adding the **State** field from the **Customers** table into the **Location** well and the **Sales Revenue** field from the **Sales** table into the **Size** well.



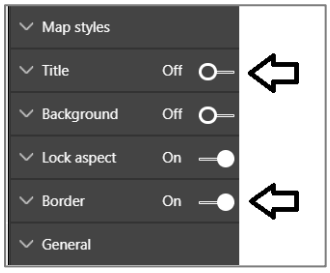
d) Add the **Units Sold** field from the **Sales** table and the **Customer Count** field from the **Customers** table into the **Tooltips** well.



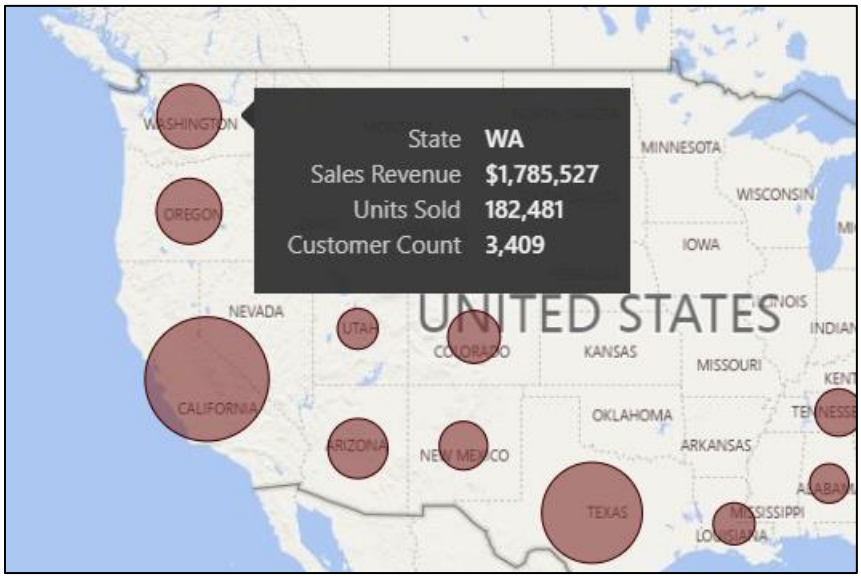
e) With the Map visual selected, navigate to the **Data Colors** section in the **Format** properties pane and change the default color to a dark red to it stands out more clearly on the map visual.



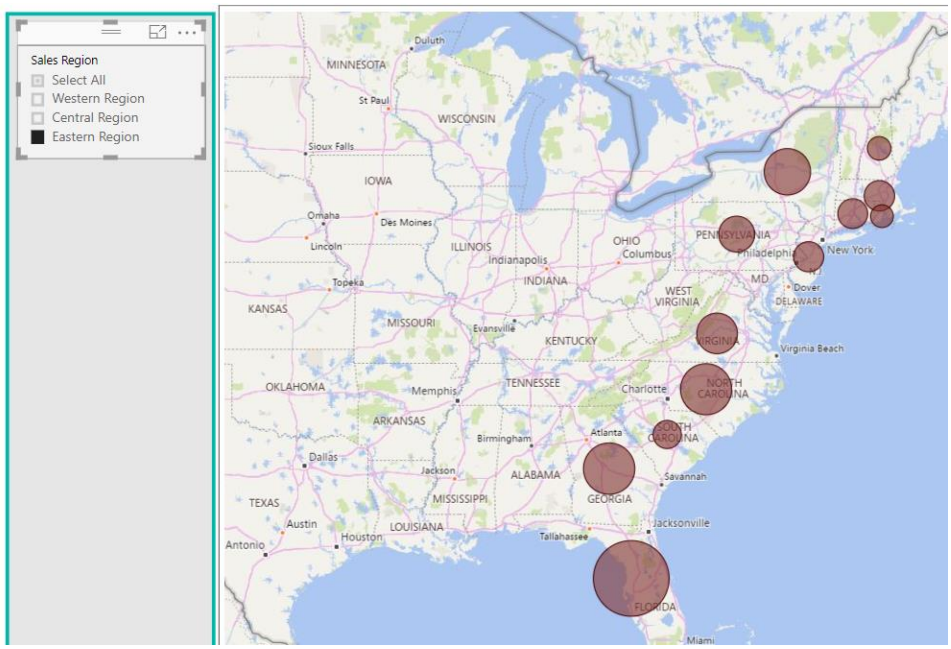
f) Below in the Format pane, disable the **Title** setting and enable the **Border** setting as shown in the following screenshot.



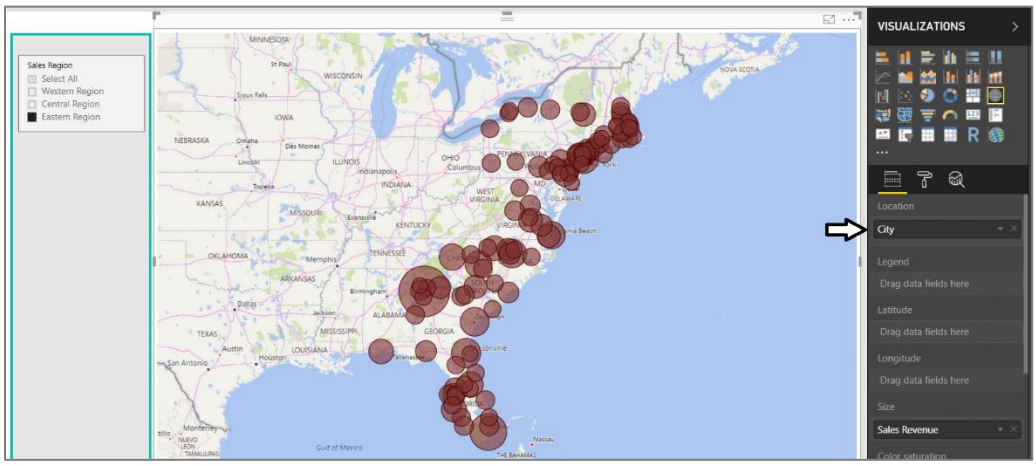
g) Test out the **Tooltips** setting by hovering the mouse over a red dot for a specific state. You should see that the **Tooltips** setting displays **State**, **Sales Revenue**, **Units Sold** and **Customer Count**.



h) Try out the slicer. You should be able to select a single sales region and see the map update to reflect the new filtering.

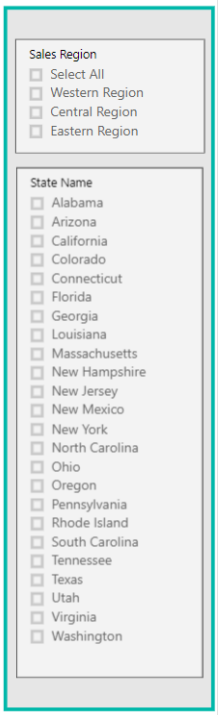


i) With the **Map** visual select, navigate to the **Field** properties pane. Remove the **State** field from the **Location** well and replace it with the **City** field from the **Customers** table. Now the red dots are more granular because they represent cities not states.



9. Add a second slicer below the first slicer to filter by **State**.
   a) Click the **New Visual** button on the ribbon to create a new visual.
   b) Click the **Slicer** button in the **Visualizations** list to change the visual into a slicer visual.
   c) Configure the new slicer by adding the **State Name** field from the **Customers** table into the **Field** well.

   d) In the Format pane, configure the slicer with a background color of white and a border just like you did for the other slicer.
   e) Reposition the new slicer to it appears just below the first slicer as shown in the following screenshot.



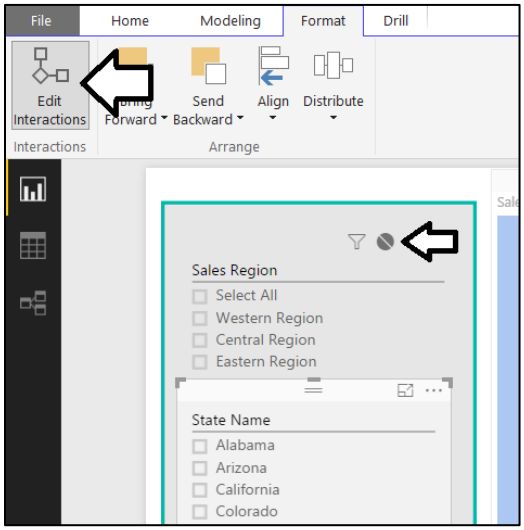
In the next step you will configure the edit interactions between the two slicer visuals. The reason for this is that you do not want the button slicer to place a filter on the top slicer.

f) Make sure the bottom slicer is selected.

g) Navigate to the **Format** tab in the ribbon.

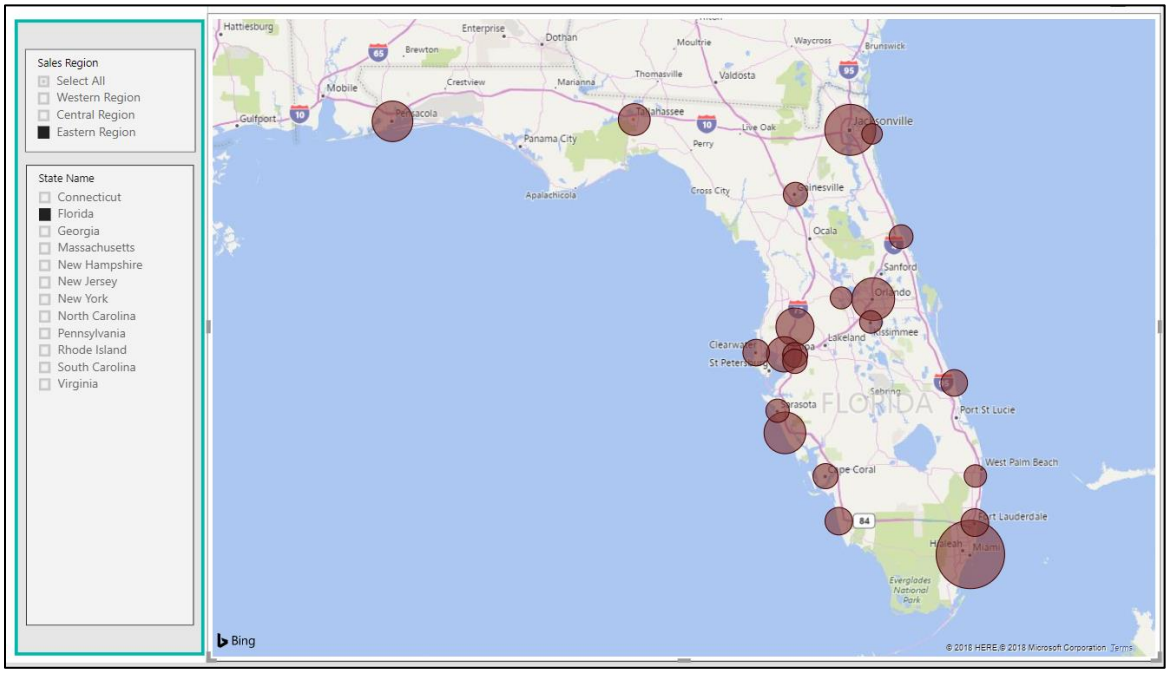
h) Click the **Edit Interactions** button in the ribbon to enter edit interaction mode.

i) In the **Sales Region** slicer, click the circle icon with the line through it to prevent the other slicer from affecting this slicer.



j) Click the **Edit Interactions** button a second time to exit edit interactions mode.

k) Test out the finished report page. You should be able to select a sales region from the top slicer which filters the set of states available in the bottom slicer. You should then be able to select a state and see its cities in the Map visual.

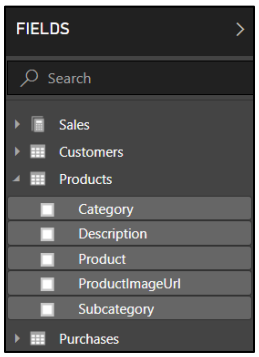


Congratulations for making great progress. You are about half way through this set of lab exercises. Note that the **Solution** folder for this module contains a partial solution named **Wingtip Sales Analysis - Exercise 3.6.pbix** with all the work up to this point.

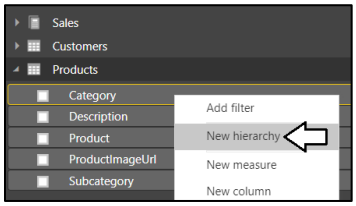
Exercise 7: Create a Dimensional Hierarchy for Product Category

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.

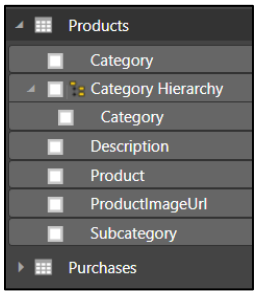
1. Navigate to Report view and inspect the tables in the **Fields** list on the righthand side of the Power BI Desktop window.
2. Add a new dimensional hierarchy to the **Products** table.
   a) Inspect the **Products** table in the fields list. It should appear as the one shown in the following screenshot.



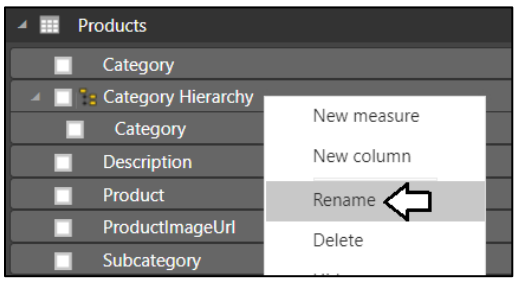
   b) Right-click on the **Category** field and then select the **New Hierarchy** menu command.



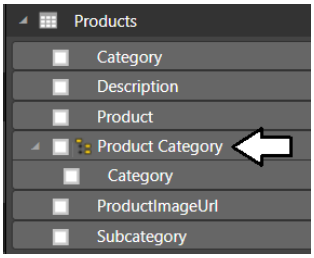
   c) You should now see a new dimensional hierarchy in the fields list named **Category Hierarchy**.



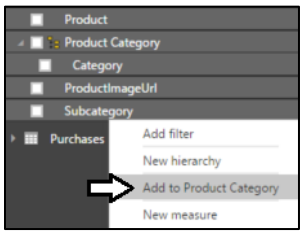
   d) Right-click **Category Hierarchy** and select the **Rename** menu command.



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

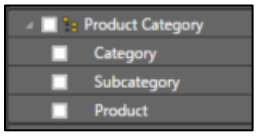


f) Right-click on the **Subcategory** field and select the **Add to Product Category** menu command.

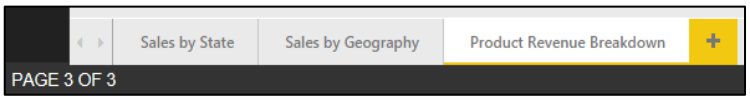


g) Right-click on the **Product** field and select the **Add to Product Category** menu command.

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



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



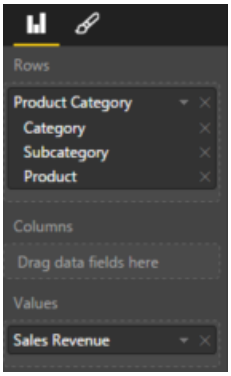
4. Create a new matrix visual to display sales revenue broken out by product category, subcategory and product.

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

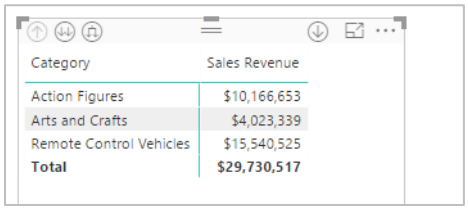
b) Change the visual to a matrix by clicking the **Matrix** button in the **Visualizations** list.

c) In the **Fields** list, click the checkbox for the **Product Category**.

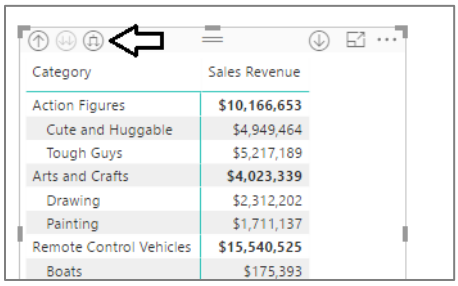
d) Drag and drop the **Sales Revenue** field from the **Sales** table into the **Values** well.



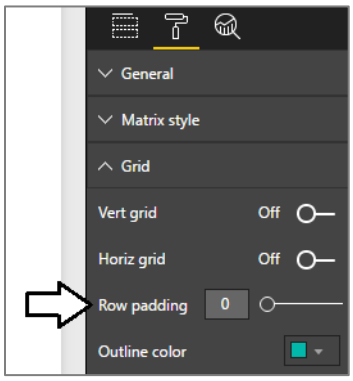
e) The Matrix visual should now display sales revenue for each product along with totals for each subcategory and category.



f) Click on the **Expand All Down** button on the small toolbar at the top right corner of the Matrix visual to display subcategories.

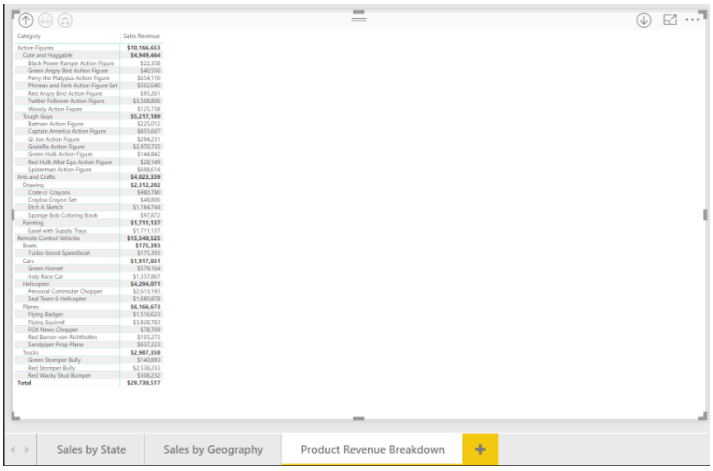


g) With the **Matrix** visual selected, navigate to the **Grid** section in the **Format** pane and update to **Row padding** setting to 0.

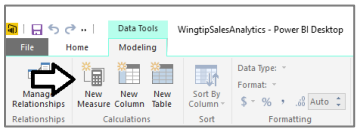


h) Click on the **Expand All Down** button at the top right corner of the Matrix visual one more time to display products.

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



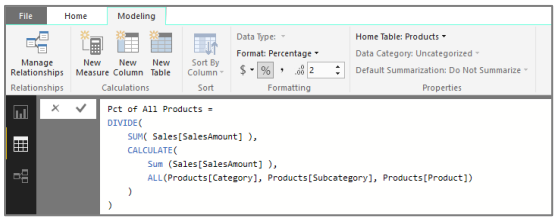
5. Create the **Pct of All Products** measure that calculates the percentage of sales revenue compared to that of all sales revenue.
   a) Navigate to data view.
   b) Select the **Products** table from the **Fields** list.
   c) Create a new measure by clicking the **New Measure** button in the ribbon.



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

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

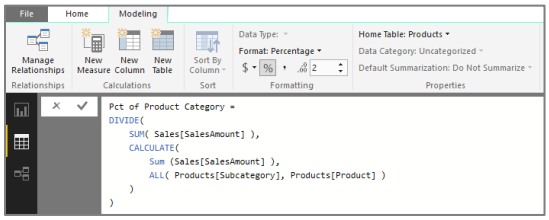
   e) Press the **ENTER** key to add the measure to the data model.
   f) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Percentage**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to **2**.



6. Create the **Pct of Product Category** measure that calculates the percentage of sales revenue within the current product category.
   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 to 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] )
 )
)
```

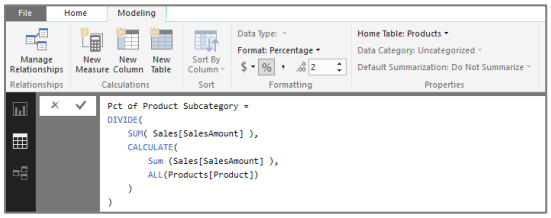
   d) Press the **ENTER** key to add the measure to the data model.
   e) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Percentage**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to **2**.



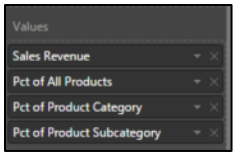
7. Create the **Pct of Product Subcategory** measure that calculates the percentage of sales revenue within the current subcategory.
   a) Select the **Products** table from the **Fields** list.
   b) Create a new measure by clicking the **New Measure** button in the ribbon.
   c) Enter to following 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] )
 )
)
```

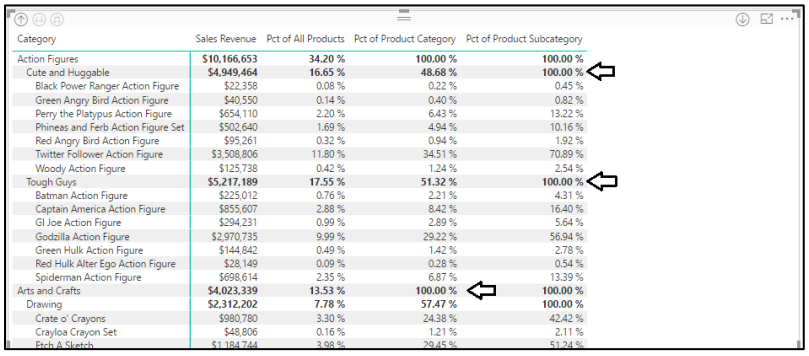
   d) Press the **ENTER** key to add the measure to the data model.
   e) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Percentage**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to **2**.



8. Modify the matrix visual on the **Product 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 **Product 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.

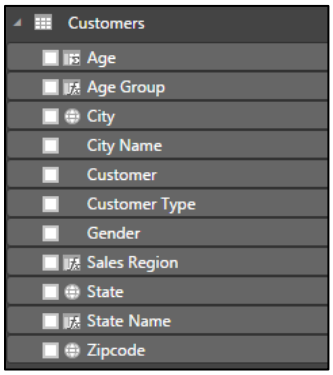


Category	Sales Revenue	Pct of All Products	Pct of Product Category	Pct of Product Subcategory
Action Figures	$10,166,653	34.20 %	100.00 %	100.00 %
Cute and Huggable	$4,949,464	16.65 %	48.68 %	100.00 %
Black Power Ranger Action Figure	$22,358	0.08 %	0.22 %	0.45 %
Green Angry Bird Action Figure	$40,550	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 %
Twitter Follower Action Figure	$3,508,806	11.80 %	34.51 %	70.89 %
Woody Action Figure	$125,738	0.42 %	1.24 %	2.54 %
Tough Guys	$5,217,189	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 %
GI Joe Action Figure	$294,231	0.99 %	2.89 %	5.64 %
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 Alter Ego Action Figure	$28,149	0.09 %	0.28 %	0.54 %
Spiderman Action Figure	$698,614	2.35 %	6.87 %	13.39 %
Arts and Crafts	$4,023,339	13.53 %	100.00 %	100.00 %
Drawing	$2,312,202	7.78 %	57.47 %	100.00 %
Crate o' Crayons	$980,780	3.30 %	24.38 %	42.42 %
Crayloa Crayon Set	$48,806	0.16 %	1.21 %	2.11 %
Etch A Sketch	$1,184,744	3.98 %	29.45 %	51.24 %

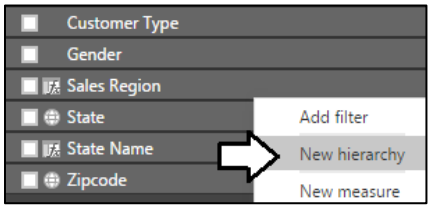
Exercise 8: Create a Dimensional Hierarchy for Customer Geography

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

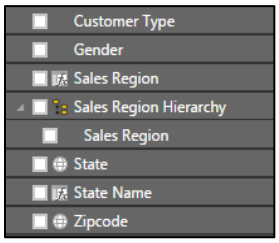
1. Add a new dimensional hierarchy to the **Customers** table.
   a) If you are not already in Report view, use the left navigation to switch to Report view.
   b) Inspect the **Customers** table in the fields list. It should appear as the one shown in the following screenshot.



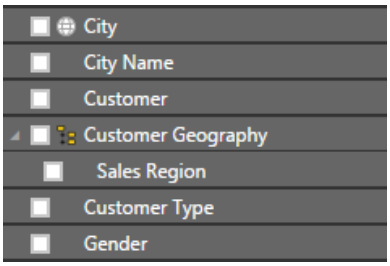
   c) Right-click on the **Sales Region** field and then select the **New Hierarchy** menu command.



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

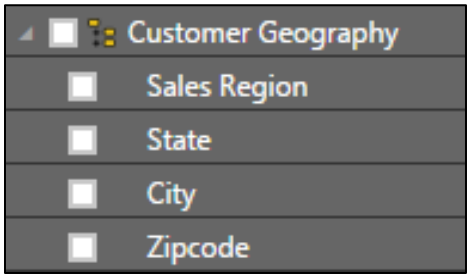


   e) Right-click **Sales Region Hierarchy,** select the **Rename** menu command and rename the hierarchy **Customer Geography**.

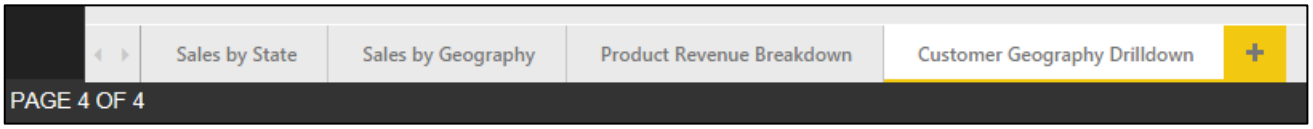


   f) Right-click on the **State** field and select the **Add to Customer Geography** menu command.
   g) Right-click on the **City** field and select the **Add to Customer Geography** menu command.
   h) Right-click on the **Zipcode** field and select the **Add to Customer Geography** menu command.

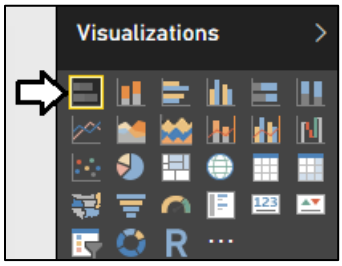
i) The **Customer Geography** hierarchy should now contain four fields as shown in the following screenshot.



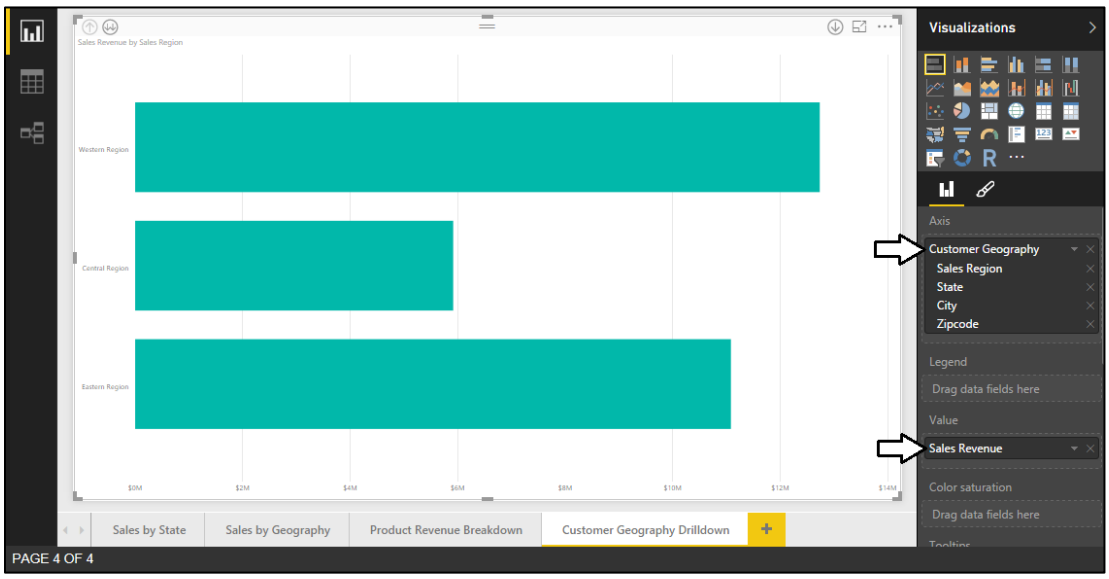
2. Create a new report page and rename it to **Customer Geography Drilldown**.



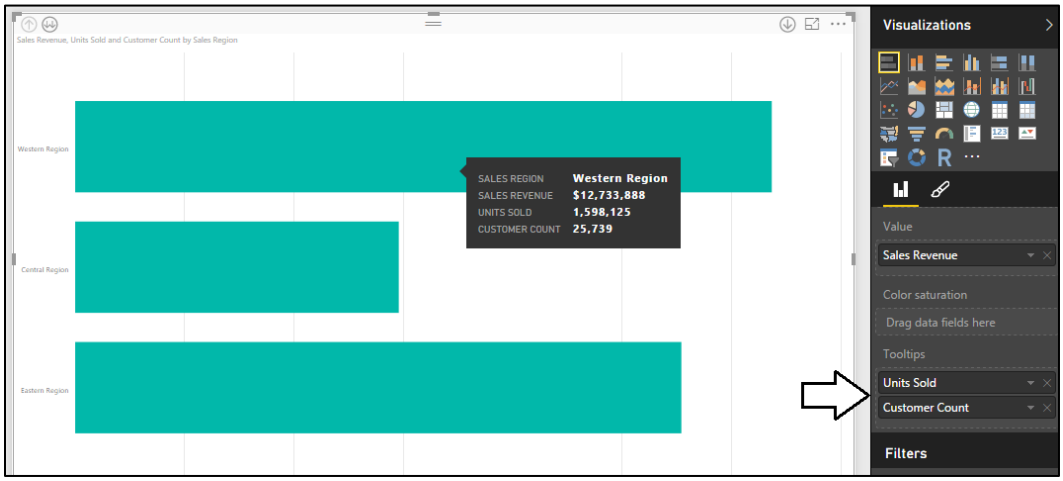
3. Add a new stacked bar chart visual.
   a) Click the **Stacked Bar Chart** button on the **Visualizations** list to create a new bar chart visual.



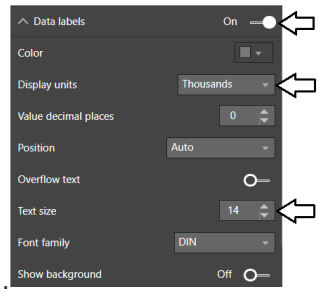
   b) Resize the bar chart visual so it takes up the entire height and width of the report page.
   c) Drag and drop the **Customer Geography** hierarchy from the **Customers** table into the **Axis** well.
   d) Drag and drop the **Sales Revenue** field from **Sales** table into the **Value** well.
  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



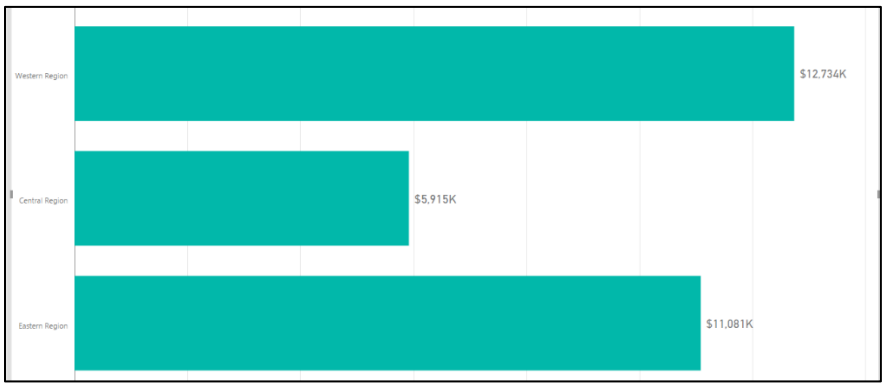
4. Configure **Tooltips** for the bar chart to additionally show **Unit Sold** and **Customer Count**.
   a) Drag and drop the **Units Sold** field from the **Sales** table into the **Tooltips** well.
   b) Drag and drop the **Customer Count** field from the **Sales** table into the **Tooltips** well.
   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**.

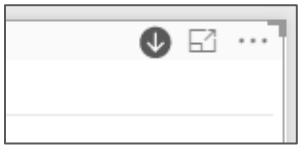


   f) The bar chart should now display a data label for each bar showing total sales revenue.

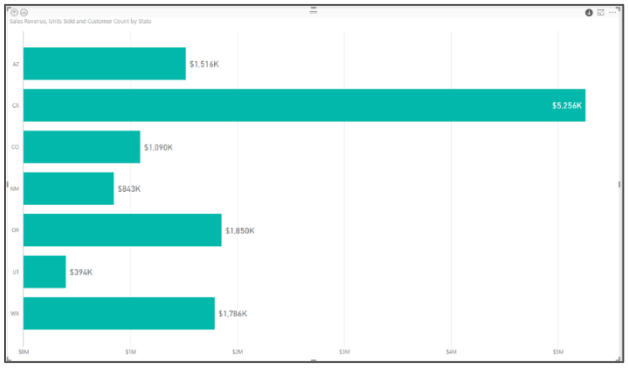


6. Turn on drill down mode for the bar chart visual.
   a) Locate the **Drill Down** button with the downward pointing arrow and the circle around it in the top right corner of the visual.
   b) Click on the **Drill Down** button once to enable drill down mode.

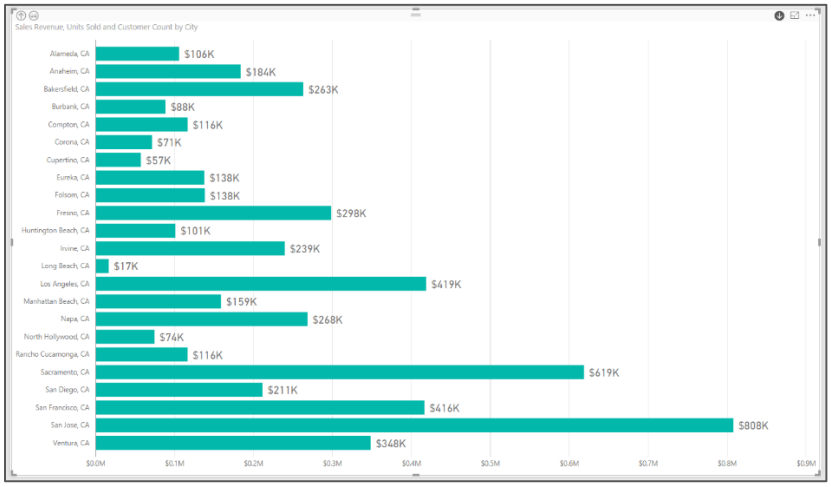
   c) Once you have enabled drill down mode, the Drill Down button appears as a dark circle with a white arrow.



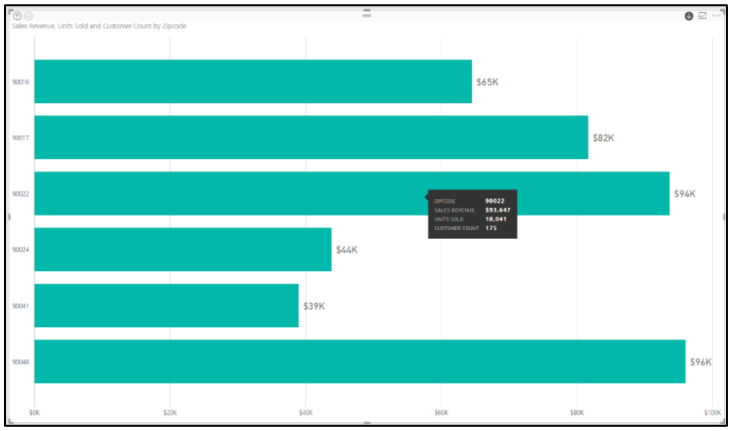
7. Experiment with drill down mode by drilling into the **Customer Geography** hierarchy.
   a) Click on the bar for the **Western Region** to drill down into the states for that sales region.



   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.



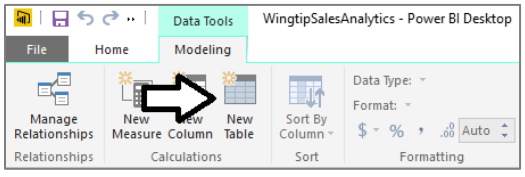
d) In the top left corner is a **Drill Up** button. Click on this button three times to return to the top level of the hierarchy.

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

Exercise 9: 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.

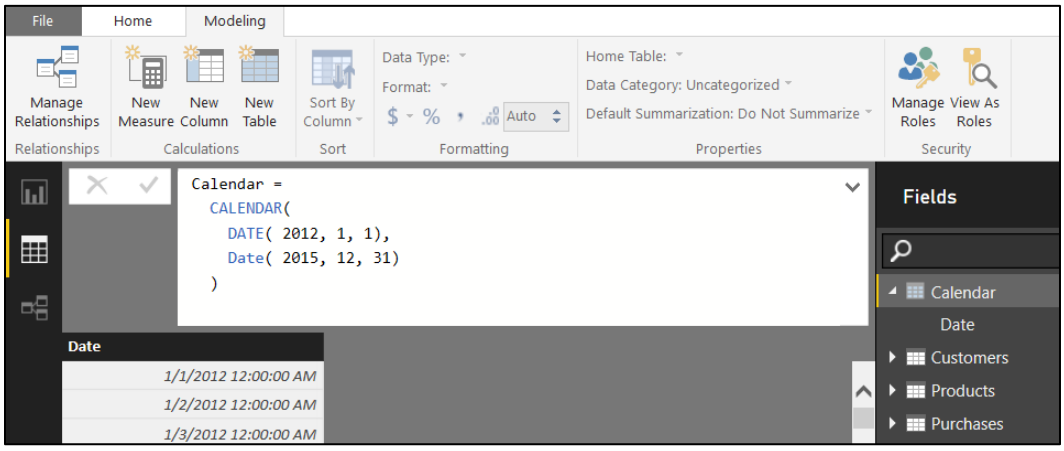
8. Create a new calculated table named **Calendar**.
   a) Navigate to the **Modeling** tab in the ribbon.
   b) Click the **New Table** button in the ribbon.



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 and 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]) )
```

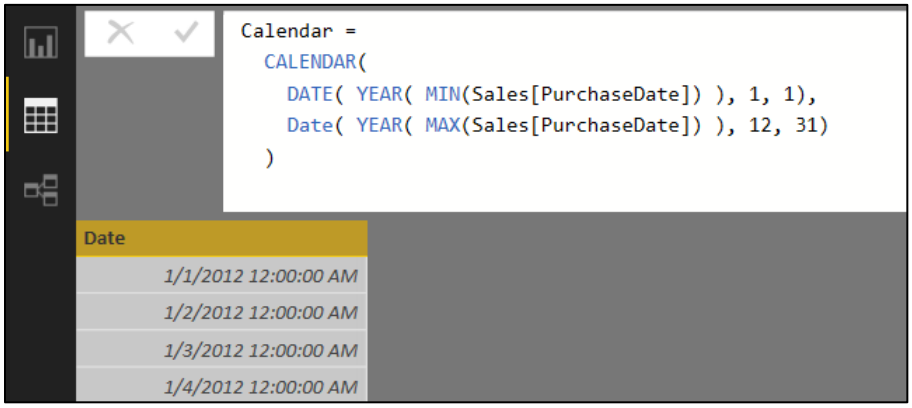
f) Next, replace the literal value for the ending year 2015 with the following DAX expressions.

```
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 **1/1/20012** and running to **12/31/2015**.

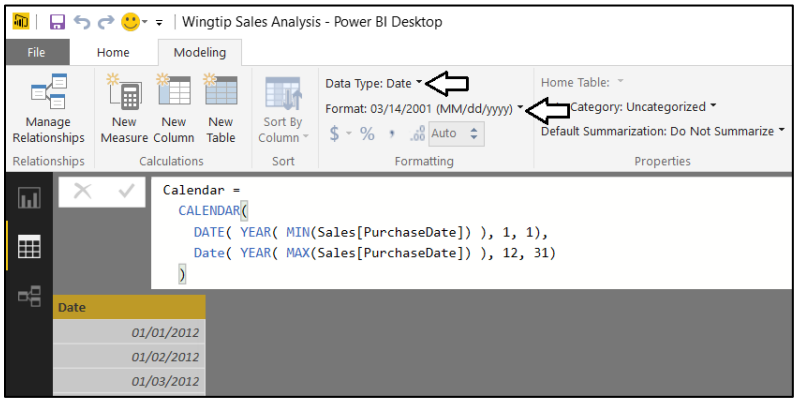


The modification to the DAX expression to calculate the start date and end date dynamically provides flexibility. The new expression will now automatically add new rows for complete years if the **Sales** table is ever updated with purchases that occurred either before the start of 2012 or after the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

9. 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.

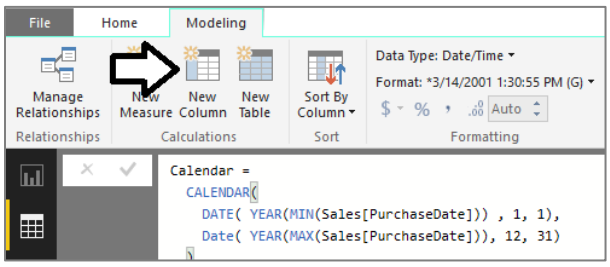
c) Set the **Data Type** property to **Date**.

d) Set the **Format** property to **03/14/2001 (MM/dd/yyyy)**.



10. 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.

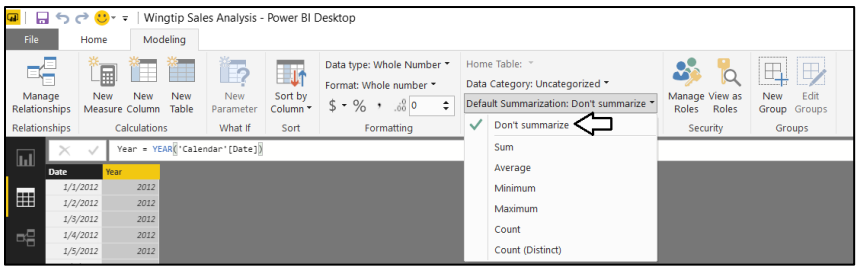


b) Type in the following DAX expression and press the Enter key.

```
Year = YEAR('Calendar'[Date])
```

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

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

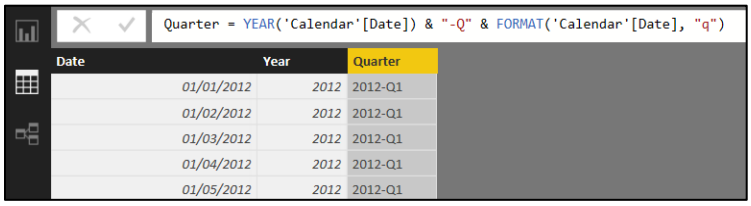


While the **Year** column contains numeric values, it doesn't make sense to perform standard aggregations on this column such as **Sum**, **Average** or **Count**. Setting the column's **Default Summarization** to **Do Not Summarize** will prevent Power BI Desktop from automatically assigning aggregate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

11. Add a new column to the **Calendar** table named **Quarter** which display the financial year and quarter.
    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.

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

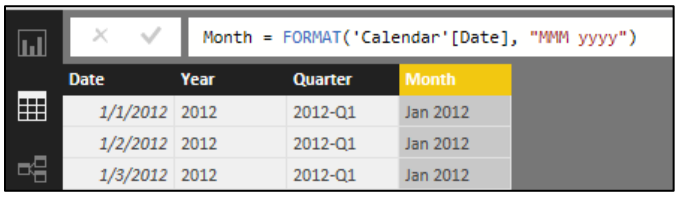
    c) The **Quarter** column now displays a value with the year and quarter which can be used in visuals and reports.



12. Add a new column to the **Calendar** table named **Month** which displays the financial year and month.
    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.

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

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

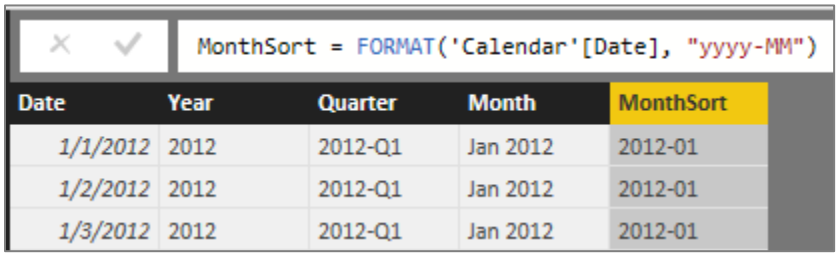


The **Month** column is a good example of a column whose value will not automatically be sorted in the chronological sort order. The default sort order of a text column like **Month** is to sort month names alphabetically so that April will sort before February, and February will sort before January.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

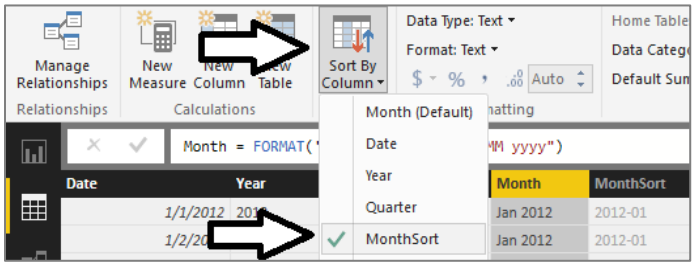
13. Add a new sort column to the **Calendar** table named **MonthSort** to control the sort order of the **Month** column.
    a) Create a new calculated column by clicking the **New Column** button in the ribbon.
    b) Type in the following DAX expression and press the Enter key.

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

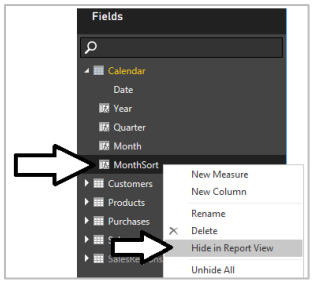
    c) You should be able to see that the **MonthSort** column produces a text value for each date in the format of **2012-01**. The key aspect of this format is that **MonthSort** values are sorted chronologically when they are sorted alphabetically.



d) 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.



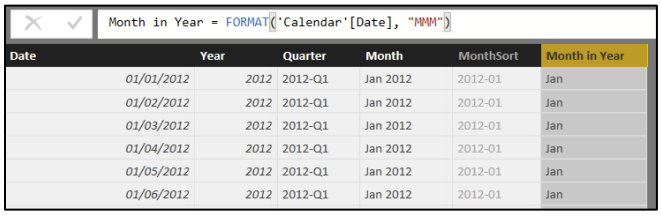
e) Hide the **MonthSort** column by right-clicking it on the **Fields** list and selecting the **Hide in Report View** menu command.



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

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

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



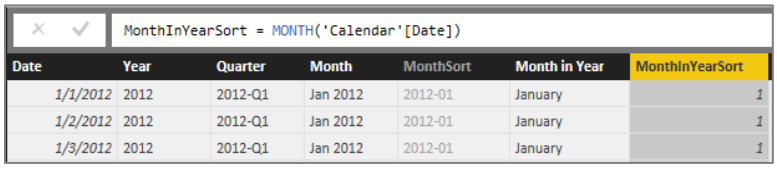
Date	Year	Quarter	Month	MonthSort	Month in Year
01/01/2012	2012	2012-Q1	Jan 2012	2012-01	Jan
01/02/2012	2012	2012-Q1	Jan 2012	2012-01	Jan
01/03/2012	2012	2012-Q1	Jan 2012	2012-01	Jan
01/04/2012	2012	2012-Q1	Jan 2012	2012-01	Jan
01/05/2012	2012	2012-Q1	Jan 2012	2012-01	Jan
01/06/2012	2012	2012-Q1	Jan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

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

15. Add a new sort column to the **Calendar** table named **MonthInYearSort** to control the sort order of the **Month in Year** column.
    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.

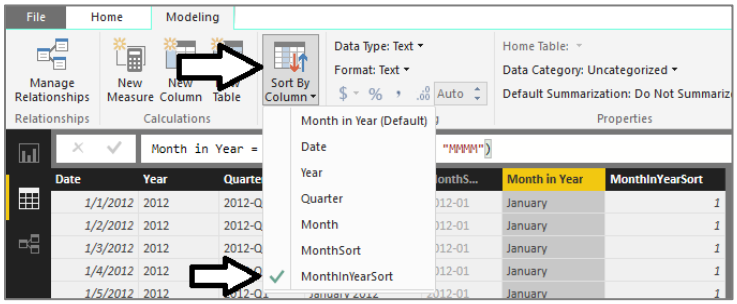
```
MonthInYearSort = MONTH('Calendar'[Date])
```

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



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.



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

16. Add a new column to the **Calendar** table named **Day of Week** which display the day of the week.

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.

```
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.

Date	Year	Quarter	Month	MonthSort	Month in Year	MonthInYearSort	Day of Week
01/01/2012	2012	2012-Q1	Jan 2012	2012-01	Jan	1	Sun
01/02/2012	2012	2012-Q1	Jan 2012	2012-01	Jan	1	Mon
01/03/2012	2012	2012-Q1	Jan 2012	2012-01	Jan	1	Tue
01/04/2012	2012	2012-Q1	Jan 2012	2012-01	Jan	1	Wed
01/05/2012	2012	2012-Q1	Jan 2012	2012-01	Jan	1	Thu
01/06/2012	2012	2012-Q1	Jan 2012	2012-01	Jan	1	Fri
01/07/2012	2012	2012-Q1	Jan 2012	2012-01	Jan	1	Sat

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

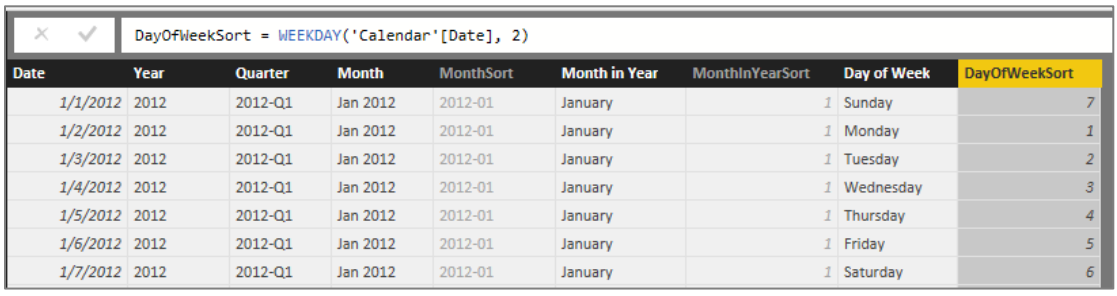
a) Create a new calculated column by clicking the **New Column** button in the ribbon.

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

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

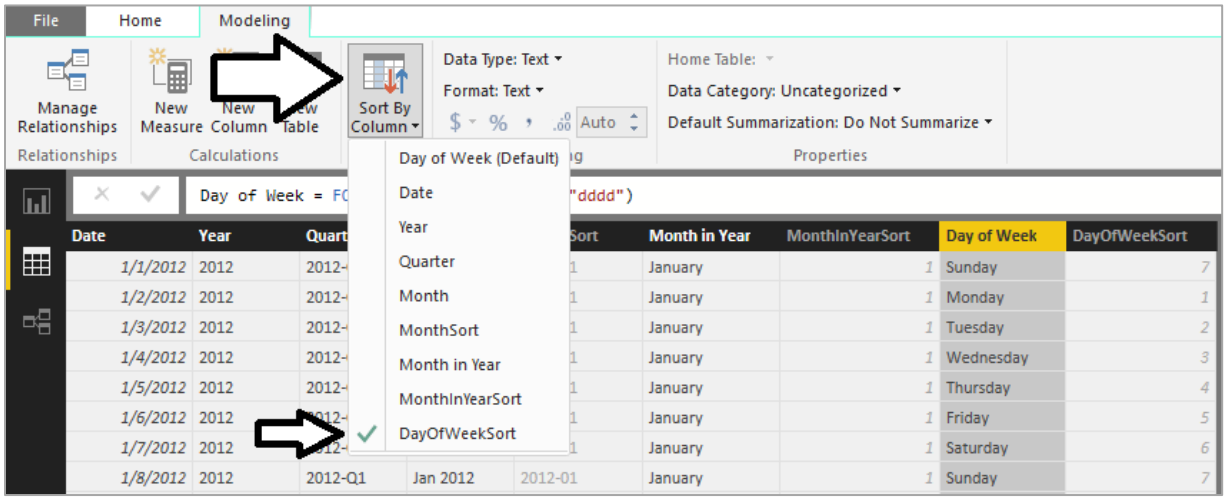
The second argument passes to **WEEKDAY** function determines the starting day for the week. If you pass a value of 1, the starting day for the week will be Sunday. In this case you have passed a value of 2 which will make Monday the first day of the week.

c) 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.



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

d) 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.



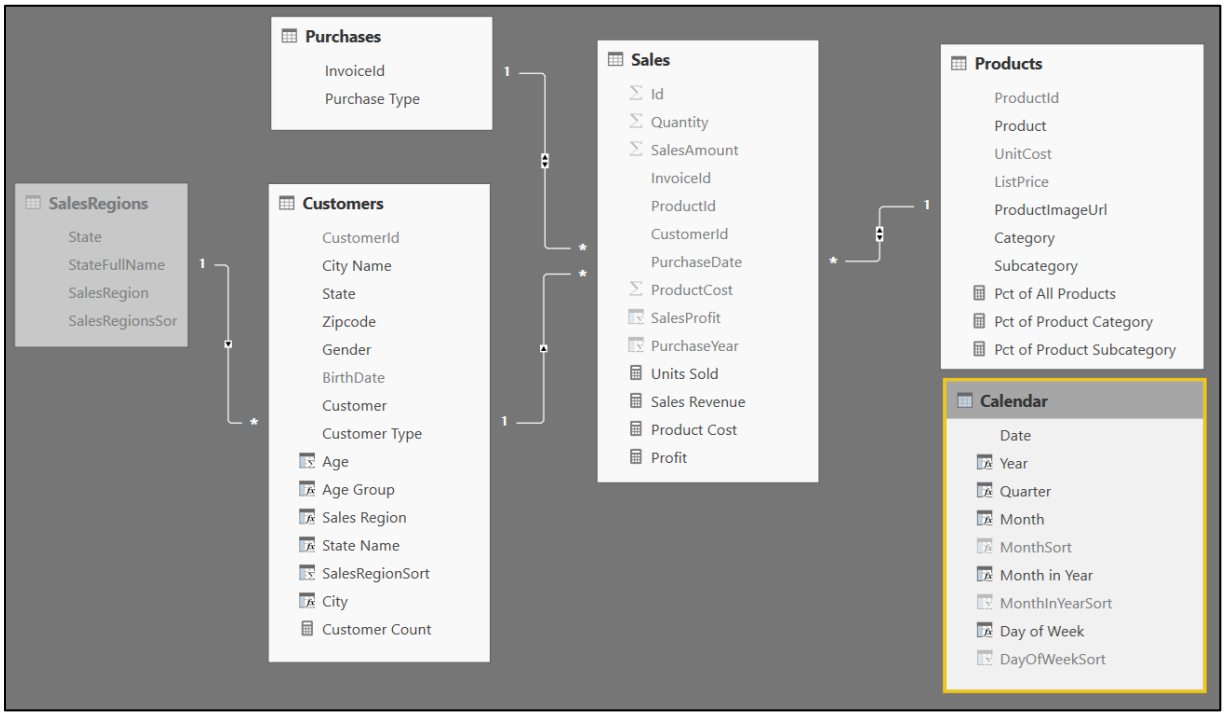
e) Hide the **DayInWeekSort** column by right-clicking it in the **Fields** list and selecting the **Hide in Report View** command.

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.

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

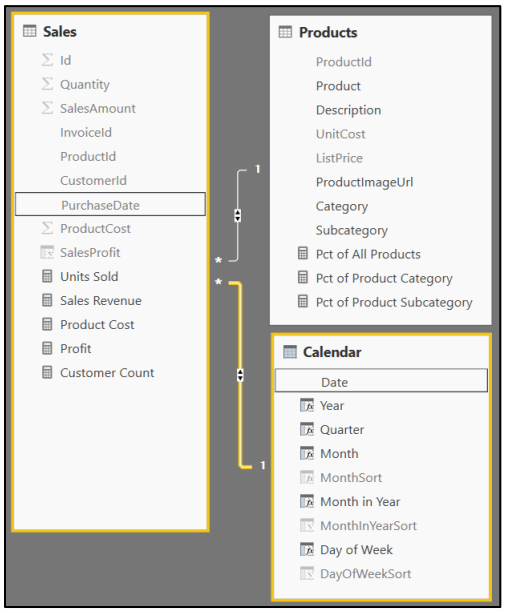
a) Navigate to relationship view. You should be able to see that the **Calendar** table is present. You should also be able to verify that the **Calendar** table does not have any existing relationships.

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.

d) Configure the **Cross filter direction** of the new relationship to **Both**.

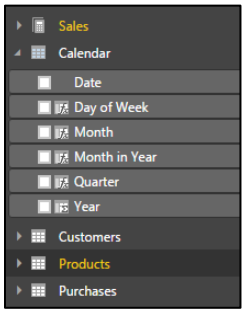


You have now completed the work of adding the **Calendar** table to the data model. Now, you will modify the **Calendar** table to add a new dimension hierarchy named **Calendar Drilldown**.

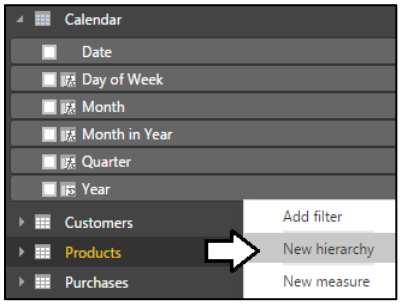
19. Add a new dimensional hierarchy to the **Calendar** table.

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.



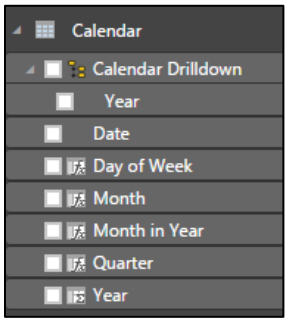
c) Right-click on the **Year** field and then select the **New Hierarchy** menu command.



d) 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.

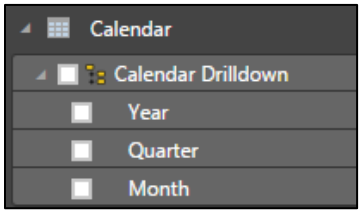
f) Rename the new hierarchy **Calendar Drilldown**.



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.

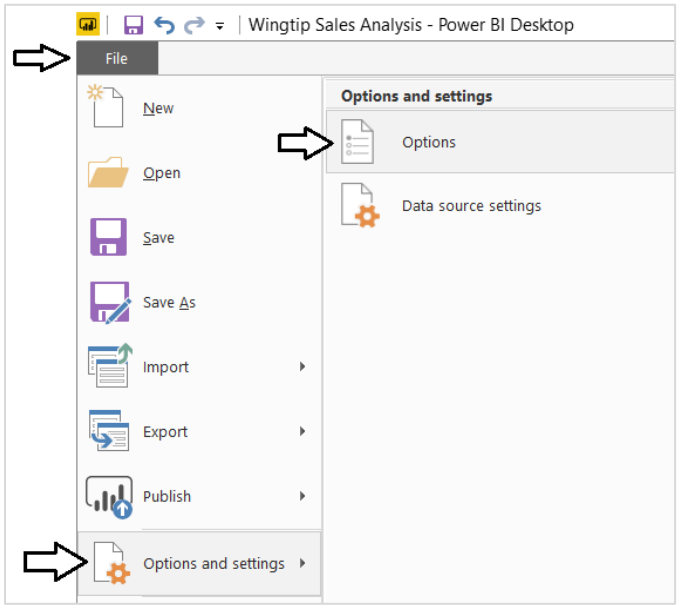
i) The **Calendar Drilldown** hierarchy should now contain three fields as shown in the following screenshot.



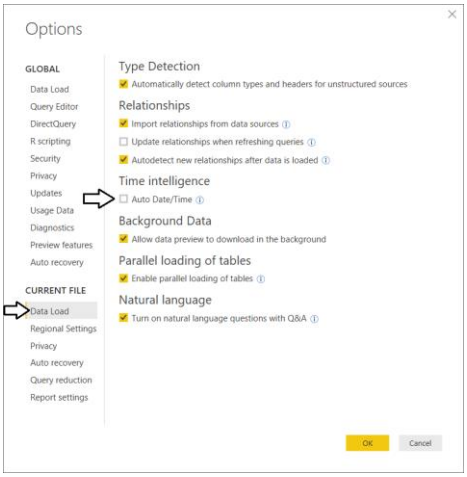
Now you have finished creating the **Calendar** table and you can use it to create reports and to call DAX time intelligence functions. However, before moving on to use your new calendar table, you will disable the Power BI Desktop feature for creating calendar tables automatically.

20. Disable the Auto Date/Time features for the **Wingtip Sales Analysis.pbix** project.

a) From the **File** menu, select the **Options and setting > Options** command to display the **Options** dialog.



b) On the **Options** dialog, select he **Data Load** tab in the left navigation and then uncheck the **Auto Date/Time** option.



c) Click the **OK** button to close the **Options** dialog.

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

Exercise 10: Design Reports and Visuals using the Calendar Table

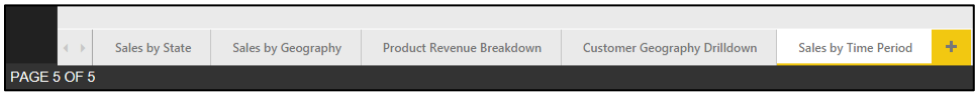
In this exercise, you will leverage the **Calendar** table that you created in the previous exercise by creating a few new visuals to display sales revenue totals aggregated over various time intervals.

21. 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.

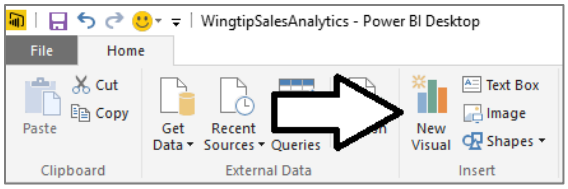
c) Once the new page has been created, modify its title to **Sales by Time Period**.



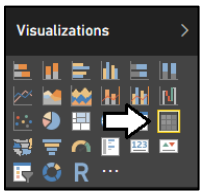
d) Now that you have created a new page, you can now add a few new visuals.

22. 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.



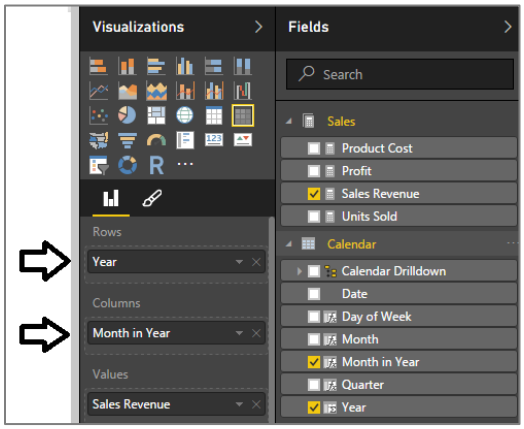
b) Change the visual to a matrix by clicking the **Matrix** button in the **Visualizations** list.



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.

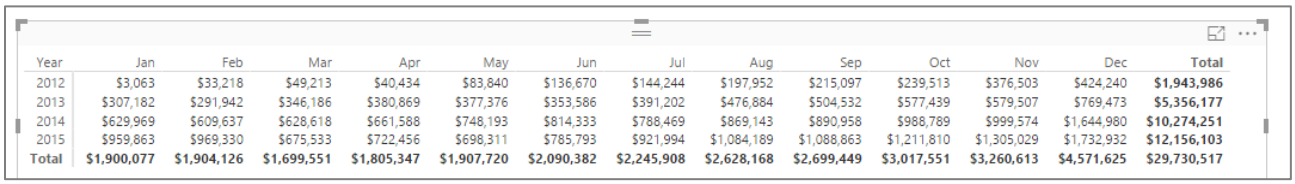
d) Now it's time to extend the matrix by adding row labels and column labels. First, drag and drop the **Year** column from the **Calendar** table in the **Fields** list into the **Rows** well in the **Visualizations** pane.

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.



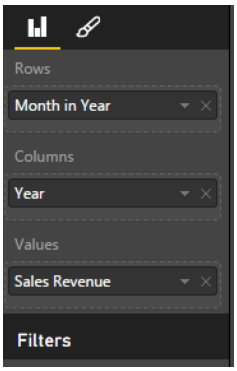
f) The matrix now has a column for each month and a row for each year.

g) Use your mouse to resize the matrix visual so you can see all the columns.

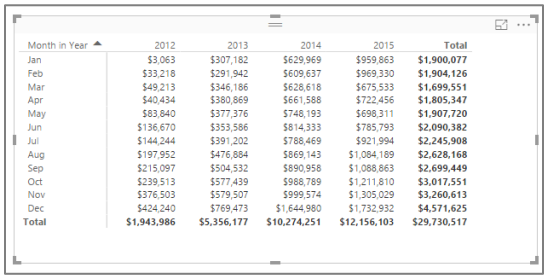


Year	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Total
2012	$3,063	$33,218	$49,213	$40,434	$83,840	$136,670	$144,244	$197,952	$215,097	$239,513	$376,503	$424,240	**$1,943,986**
2013	$307,182	$291,942	$346,186	$380,869	$377,376	$353,586	$391,202	$476,884	$504,532	$577,439	$579,507	$769,473	**$5,356,177**
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.

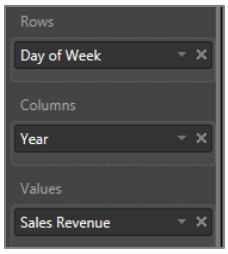


i) The matrix should now display a row for each month and a column for each year.

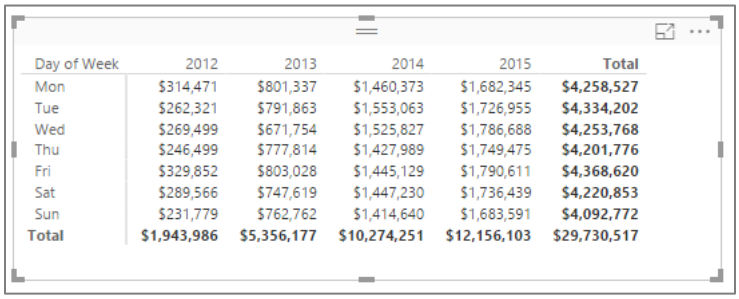


Month in Year	2012	2013	2014	2015	Total
Jan	$3,063	$307,182	$629,969	$959,863	**$1,900,077**
Feb	$33,218	$291,942	$609,637	$969,330	**$1,904,126**
Mar	$49,213	$346,186	$628,618	$675,533	**$1,699,551**
Apr	$40,434	$380,869	$661,588	$722,456	**$1,805,347**
May	$83,840	$377,376	$748,193	$698,311	**$1,907,720**
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**

23. 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) Make sure this new visual is positioned directly below the first visual that you created.
    c) Change the visual to a matrix by clicking the Matrix button in the **Visualizations** list.
    d) 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.

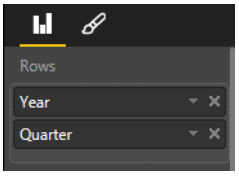


    g) The matrix should now display a row for each day of the week and a column for each year.

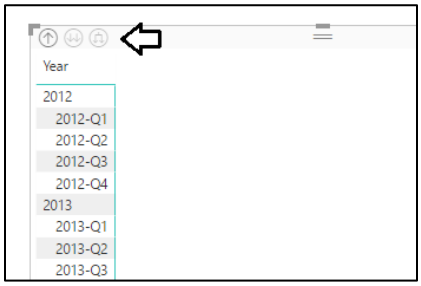


Day of Week	2012	2013	2014	2015	Total
Mon	$314,471	$801,337	$1,460,373	$1,682,345	**$4,258,527**
Tue	$262,321	$791,863	$1,553,063	$1,726,955	**$4,334,202**
Wed	$269,499	$671,754	$1,525,827	$1,786,688	**$4,253,768**
Thu	$246,499	$777,814	$1,427,989	$1,749,475	**$4,201,776**
Fri	$329,852	$803,028	$1,445,129	$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**

24. 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.

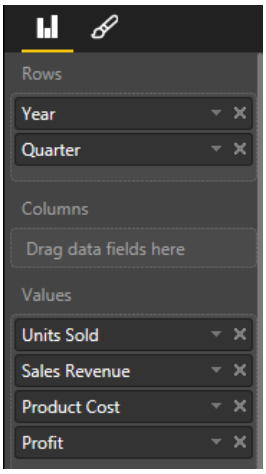


f) Drag and drop the **Units Sold** measure from the **Sales** table into the **Values** well.

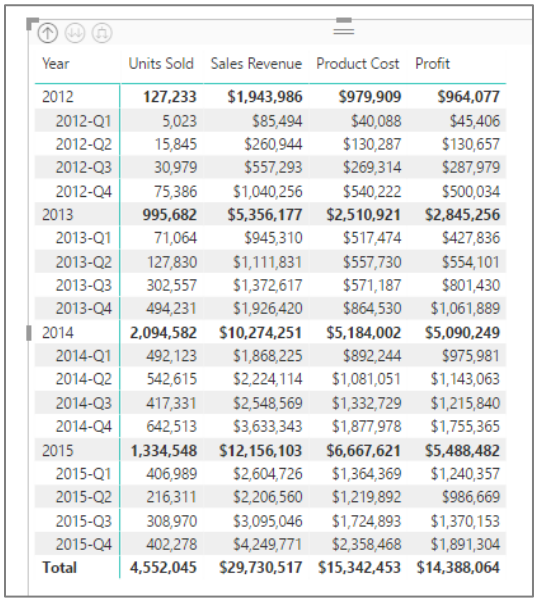
g) Drag and drop the **Sales Revenue** measure from the **Sales** table into the **Values** well.

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.

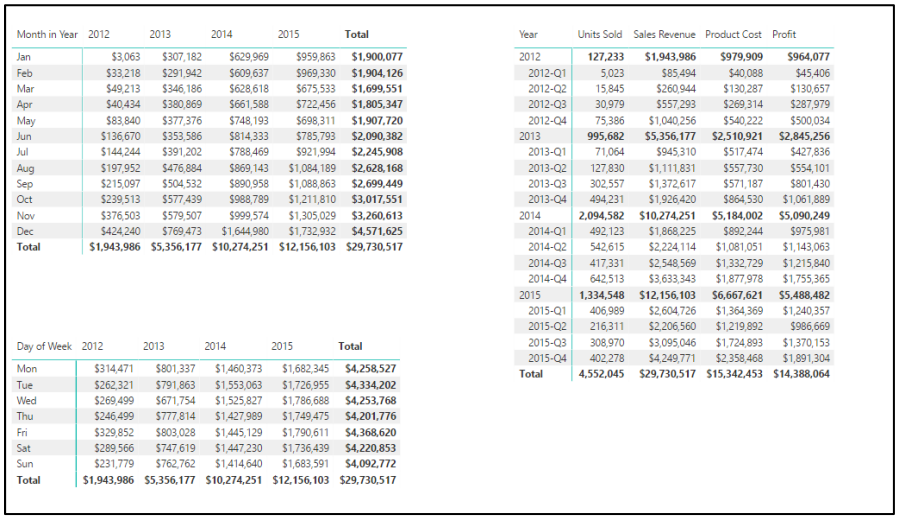


j) The matrix should now display values for each measure calculated at the quarterly level as well as at the yearly level.



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	$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**

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**

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**

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

Exercise 11: Create Measures using DAX Time Intelligence Functions

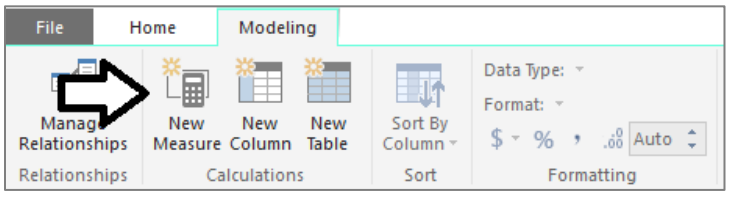
In this exercise, you will leverage various the Time Intelligence functions in DAX to analyze sales revenue using quarter to date (QTD) totals and year to date (YTD) totals. You will also use DAX to write an expression which calculate a running total of sales revenue through the entire 4 years of sales activity.

26. 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([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)**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to zero.

27. Create a measure named **Sales Revenue YTD** that calculates a year-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 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)**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to zero.

28. 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(
    [Sales Revenue],
    FILTER(
      ALL('Calendar'),
      'Calendar'[Date] <= MAX('Calendar'[Date])
    )
  )
```

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

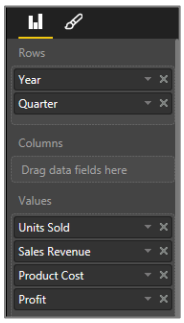
f) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Currency > English (United States)**. Also use the spinner control below the format menu to set the number of decimal places shown to zero.

29. Use the three new measures in a matrix visual.

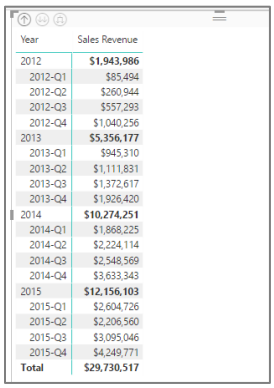
a) Navigate to report mode and make sure the **Sales by Time Period** report page is active.

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.

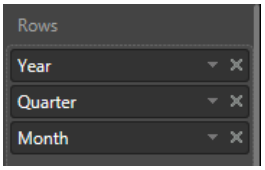


d) Remove all the measures from the **Values** well except for the **Sales Revenue** measure. At this point, your visual should match the one shown in the following screenshot where sales revenue totals as shown at the quarterly level and at the yearly level.



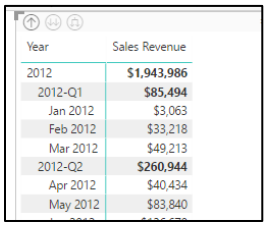
Year	Sales Revenue
2012	$1,943,986
2012-Q1	$85,494
2012-Q2	$260,944
2012-Q3	$557,293
2012-Q4	$1,040,256
2013	$5,356,177
2013-Q1	$945,310
2013-Q2	$1,111,831
2013-Q3	$1,372,617
2013-Q4	$1,926,420
2014	$10,274,251
2014-Q1	$1,868,225
2014-Q2	$2,224,114
2014-Q3	$2,548,569
2014-Q4	$3,633,343
2015	$12,156,103
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.



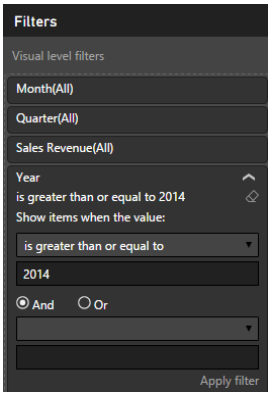
f) Click the Expand All button so the matrix displays rows for month in addition to quarter and year.

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

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**.



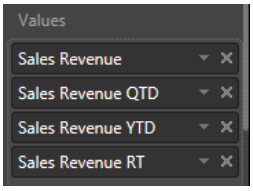
i) After setting the filter, click the **Apply Filter** link below to apply your filter to the data shown in the visual.

j) Resize the matrix visual to take up the entire right-hand side of the page.

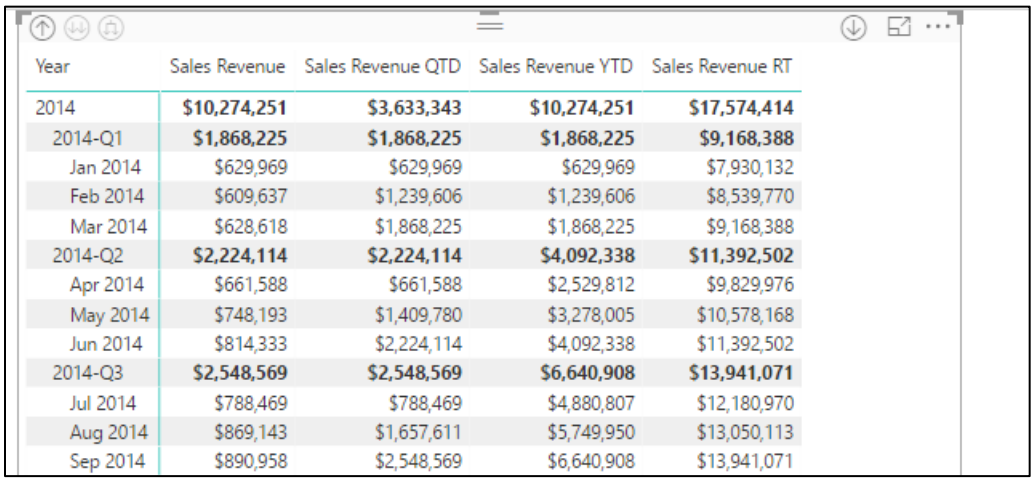
k) Drag and drop the **Sales Revenue QTD** measure from the **Sales** table into the **Values** well.

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.

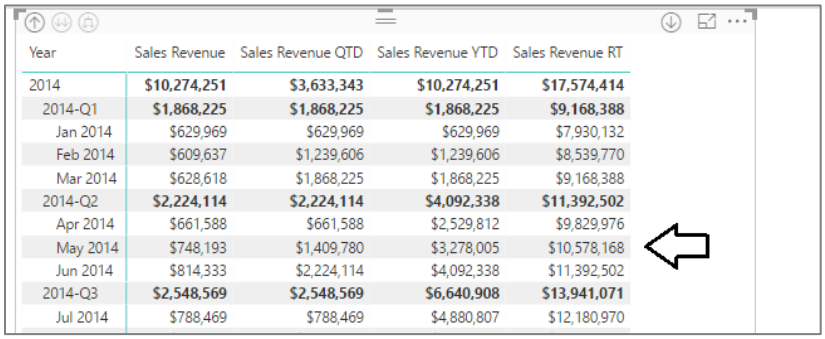


n) The matrix visual should now display three new columns for the three measure you added to the **Values** well.



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

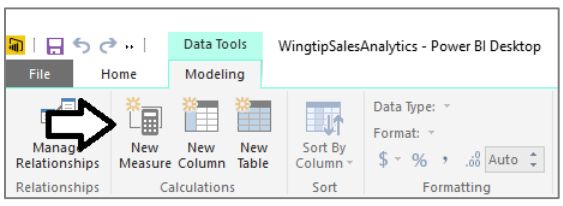
30. Save the work you have done by clicking the **Save** button in the upper left corner of the Power BI Desktop window.

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 12: 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.

31. 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.

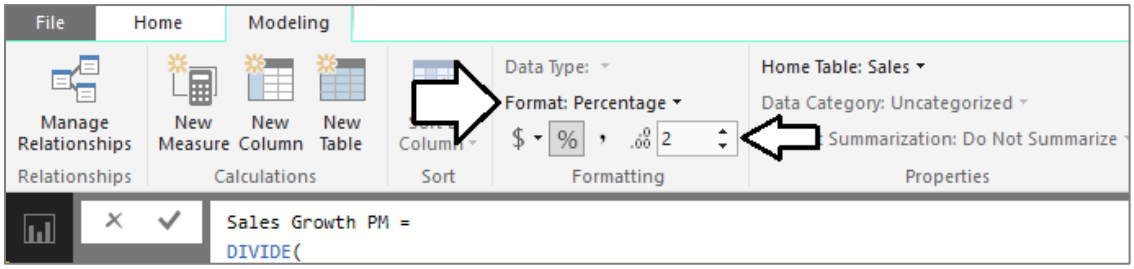


d) Enter to following DAX expression into the formula bar to create the measure named **Sales Growth PM**.

```
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**. Also use the spinner control below the **Format** menu to set the number of decimal places shown to **2**.

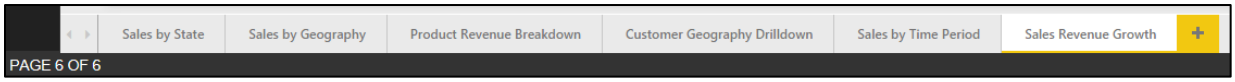


32. Create a new page in the project's report.

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**.



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

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

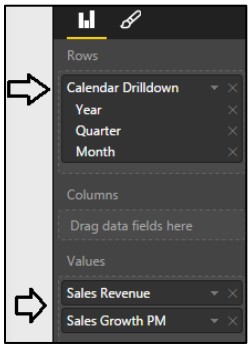
b) Change the visual to a matrix by clicking the **Matrix** button in the **Visualizations** list.

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

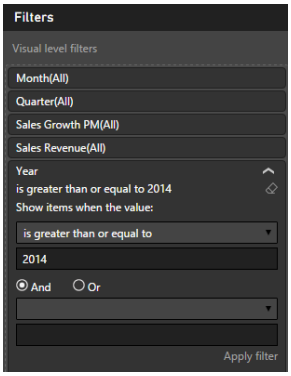
d) Drag and drop the **Sales Revenue** measure from the **Sales** table into the **Value** well.

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

f) The **Rows** well and the **Values** well for the matrix visual should match the following screenshot.



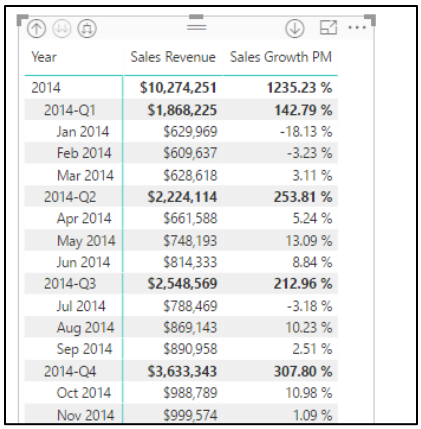
g) Move down in the **Visualizations** page to the **Filters** section. Set a filter for **Year** where value is greater or equal 2014.



h) Click the **Apply filter** button at the bottom of the **Filters** section for the filter to take effect.

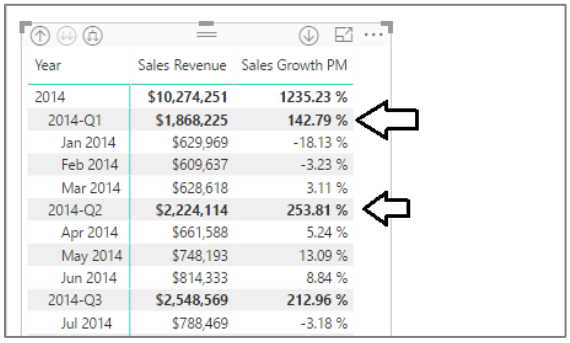
i) On the matrix, click the **Expand All** button to show **Year**, **Quarter** and **Month**.

j) 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.

34. 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

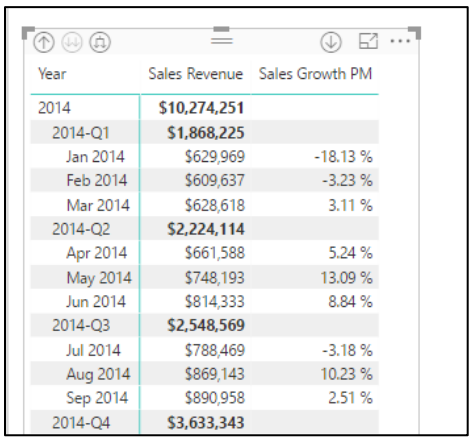
```
ISFILTERED(Calendar[Month]) && NOT(ISFILTERED(Calendar[Date]))
```

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 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.

35. 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.

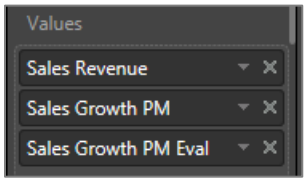
c) Create a new measure by clicking the **New Measure** button in the ribbon.

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

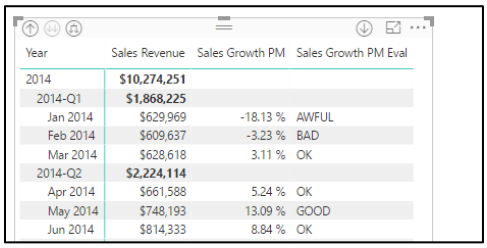
```
Sales Growth PM Eval =
IF(
  ISNUMBER([Sales Growth PM]),
  SWITCH(TRUE(),
    ([Sales Growth PM] >= 0.2), "EXCELLENT",
    ([Sales Growth PM] >= 0.1), "GOOD",
    ([Sales Growth PM] >= 0), "OK",
    ([Sales Growth PM] >= -0.1), "BAD",
    ([Sales Growth PM] <  -0.1), "AWFUL"
  )
)
```

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.

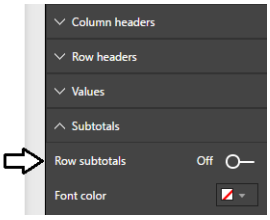


g) You should now see values for the **Sales Growth PM Eval** measure which indicate the health of sales revenue growth.

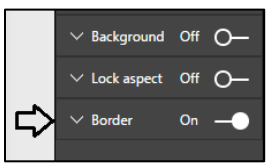


Year	Sales Revenue	Sales Growth PM	Sales Growth PM Eval
2014	$10,274,251		
2014-Q1	$1,868,225		
Jan 2014	$629,969	-18.13 %	AWFUL
Feb 2014	$609,637	-3.23 %	BAD
Mar 2014	$628,618	3.11 %	OK
2014-Q2	$2,224,114		
Apr 2014	$661,588	5.24 %	OK
May 2014	$748,193	13.09 %	GOOD
Jun 2014	$814,333	8.84 %	OK

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

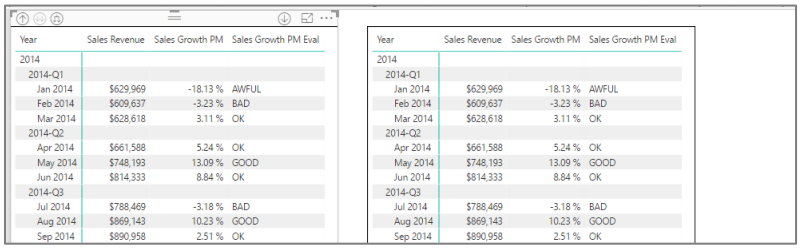


i) At the bottom of the **Format** properties pane, update the Border property from **Off** to **On**.



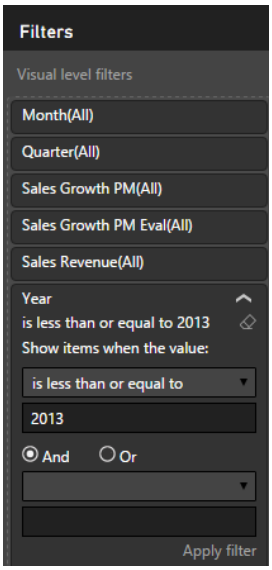
j) Save the work you have done by clicking the **Save** button in the upper left corner of the Power BI Desktop window.

36. Clone the matrix visual by copying it to the Windows clipboard and pasting it to make a copy.
    a) Select the matrix visual and copy it to the Windows clipboard.
    b) Perform a paste operation to create an identical copy of the matrix visual.
    c) 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

37. 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**.



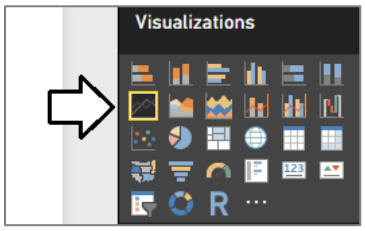
    d) Click the **Apply filter** button for the filter to take effect.
    e) The visual on the left should now display sales data for years of 2012 and 2013 while the visual on the right display sales data for 2014 and 2015.

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

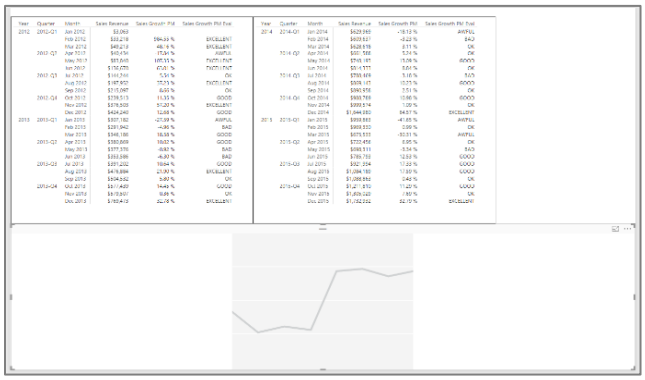
Year	Sales Revenue	Sales Growth PM	Sales Growth PM Eval
2014			
2014-Q1			
Jan 2014	$629,969	-18.13 %	AWFUL
Feb 2014	$609,637	-3.23 %	BAD
Mar 2014	$628,618	3.11 %	OK
2014-Q2			
Apr 2014	$661,588	5.24 %	OK
May 2014	$748,193	13.09 %	GOOD
Jun 2014	$814,333	8.84 %	OK
2014-Q3			
Jul 2014	$788,469	-3.18 %	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.

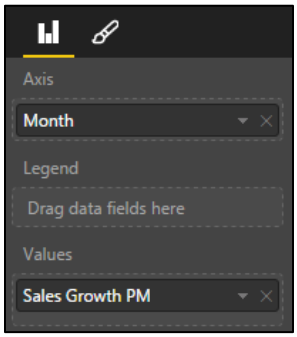
38. Add a line chart visual to the bottom of the report page to show how sales revenue has grown from month to month.
    a) Click on the whitespace on the report to make sure that neither of the two visuals are selected.
    b) Click on the Line chart button in the **Visualizations** list to create a new Line chart visual.



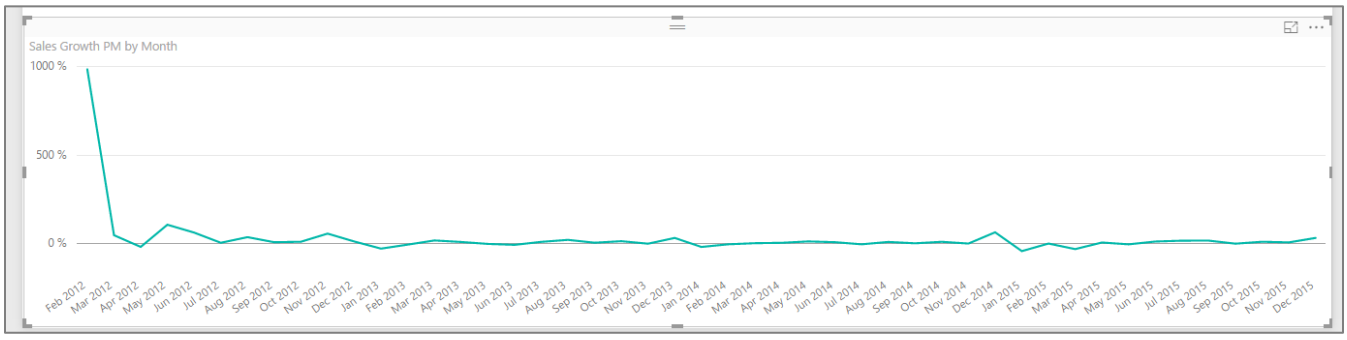
    c) Reposition the new visual so it takes up the entire width of the page at the bottom of the report.



    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.



39. 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.

40. 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[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.

41. Once you have created the **Previous Month Is Valid** measure, right click on it in the fields list and click **Hide in Report View**.
42. 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])) )
```

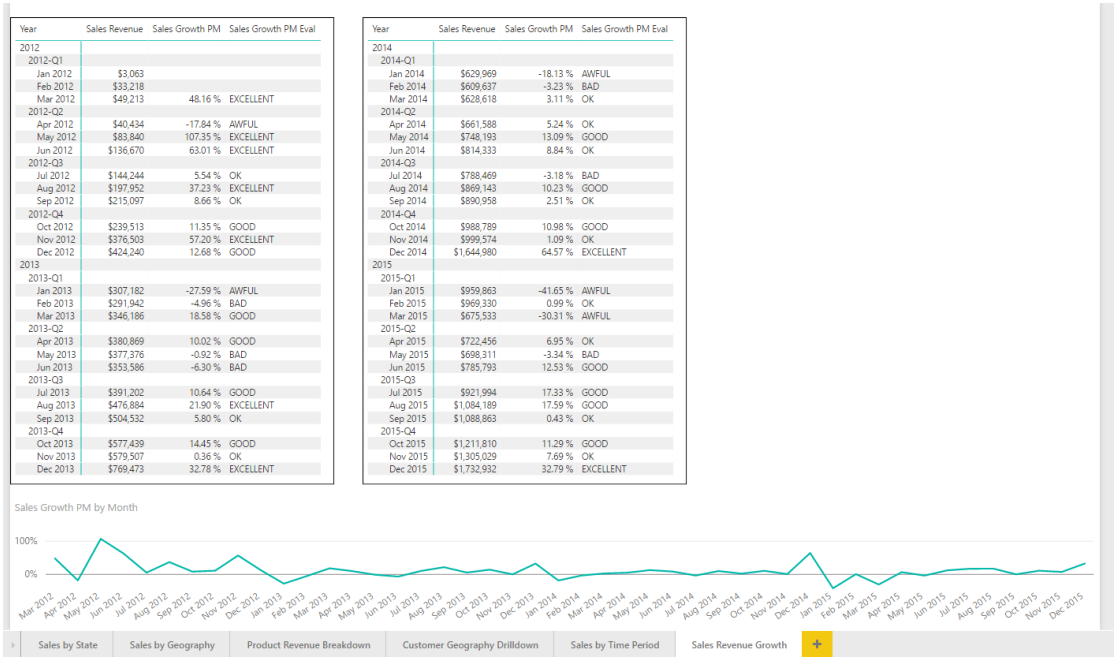
    c) Update the **If** statement to add a third condition to ensure the **Previous Month Is Valid** measure is true.

```
(
  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(
    [Sales Revenue] -
    CALCULATE(
      [Sales Revenue],
      PREVIOUSMONTH(Calendar[Date])
    ),
    CALCULATE(
      [Sales Revenue],
      PREVIOUSMONTH(Calendar[Date])
    )
  ),
  BLANK()
)
```

43. 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.



Year	Sales Revenue	Sales Growth PM	Sales Growth PM Eval
2012			
2012-Q1			
Jan 2012	$3,063		
Feb 2012	$33,218		
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
2013-Q2			
Apr 2013	$380,869	10.02 %	GOOD
May 2013	$377,376	-0.92 %	BAD
Jun 2013	$353,586	-6.30 %	BAD
2013-Q3			
Jul 2013	$391,202	10.64 %	GOOD
Aug 2013	$476,884	21.90 %	EXCELLENT
Sep 2013	$504,532	5.80 %	OK
2013-Q4			
Oct 2013	$577,439	14.45 %	GOOD
Nov 2013	$579,507	0.36 %	OK
Dec 2013	$769,473	32.78 %	EXCELLENT

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
May 2015	$698,311	-3.34 %	BAD
Jun 2015	$785,793	12.53 %	GOOD
2015-Q3			
Jul 2015	$921,994	17.33 %	GOOD
Aug 2015	$1,084,189	17.59 %	GOOD
Sep 2015	$1,088,863	0.43 %	OK
2015-Q4			
Oct 2015	$1,211,810	11.29 %	GOOD
Nov 2015	$1,305,029	7.69 %	OK
Dec 2015	$1,732,932	32.79 %	EXCELLENT

44. 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.