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 Power Query features of Power BI Desktop. The ending point in the previous lab will be your starting point for this lab. Your work in this lab will use the data modeling tools of Power BI Desktop and 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 data 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\Solution** into the folder at **C:\Student\Projects**.

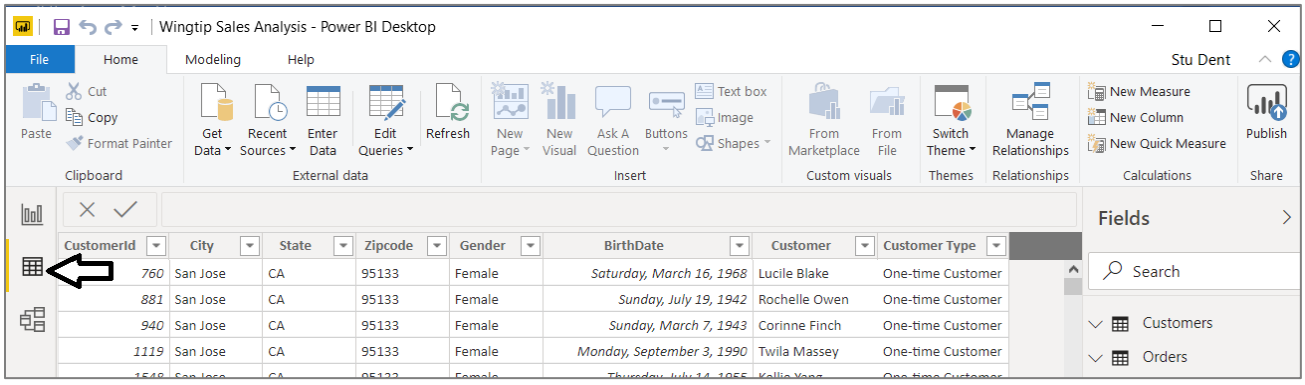
Exercise 1: Configure Table Relationships

In this exercise, you will ensure the table relationships between the tables in the data model have been created correctly.

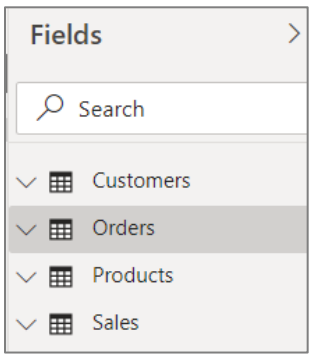
1. Open the **Wingtip Sales Analysis** project.
   a) Launch Power BI Desktop.
   b) Using the Power BI Desktop **File > Open** command, open **Wingtip Sales Analytics.pbix** located at the following path.

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

   c) When the project opens, click the table icon in the left navigation to enter **Data** view.

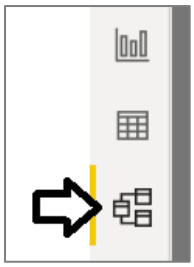


2. Look at the data in each of the four tables in the data model
   a) Click on each of the tables in the **Fields** list one by one to see what the data in each table looks like.

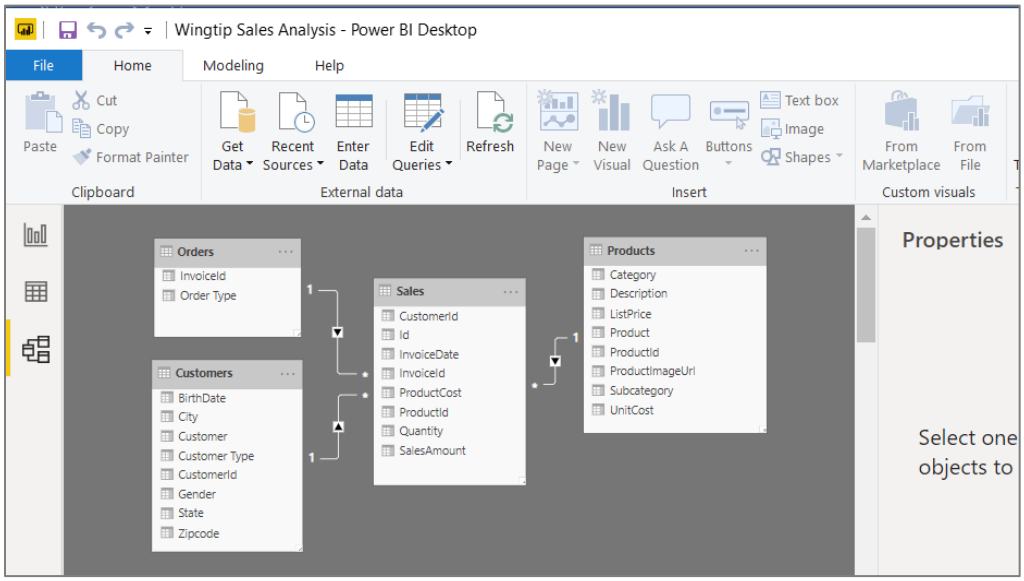


© Critical Path Training. 2020. All Rights Reserved

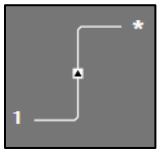
3. Inspect the tables relationships that have been created in the project's data model.
   a) Click the bottom button in the sidebar to navigate to Model view.



   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**.



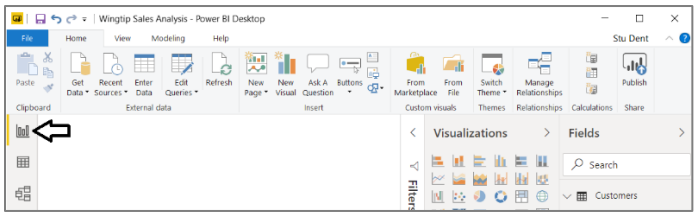
In certain scenarios it might make sense to change the **Cross filter direction** of a table relationship from **Single** to **Both**. Remember that the filter on the one-side table of a one-to-many relationship is automatically propagated to the many-side table. For example, a filter on the **State** column of the **Customers** will be automatically applied to **Sales.** However, the opposite is not true.

The filtering on a table on the many-side of a table relationship will not be automatically propagated to the table on the one-side if the **Cross filter direction** is set to **Single**. For example, you can change the **Cross filter direction** of the relationship between **Sales** and **Products** to **Both** if you want filtering on the **Sales** table to be automatically propagated to the **Products** table. This would allow filtering on the **Customers** table to propagate all the way over to the **Products** table which, in turn, would allow you to filter on customer **State** to see which products have sold in that state.

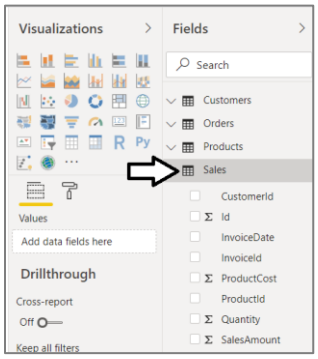
You should be cautious about setting the **Cross filter direction** of a table relationship to **Both**. Table relationships set to **Both** increase the size of your data model and can also degrade performance in larger data models. The disadvantages of using **Both** really only apply to larger data models such as those with tens of millions or hundreds of millions of rows. You will not really see any differences in performance with smaller data models such as the Wingtip Sales data model you are working with in this lab exercise. This lab will only involve table relationships with the **Cross filter direction** is set to **Single** to encourage the best practice for building larger data models.

© Critical Path Training. 2020. All Rights Reserved

4. Inspect the data model from the perspective of a report builder.
   a) Click the report icon in the left navigation to navigate to **Report** view.

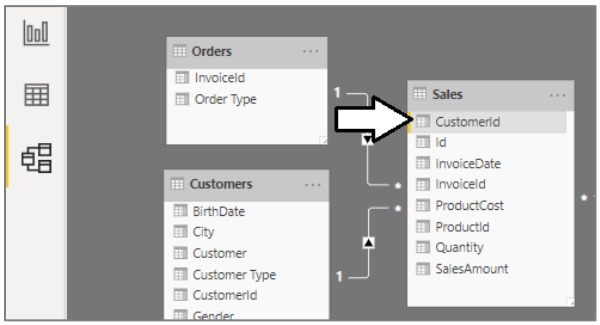


   b) Inside the **Fields** list, use the mouse to expand the fields inside the **Sales** table.

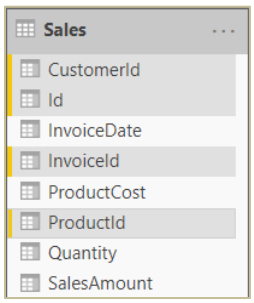


There are several fields in the **Sales** table that will never be used when designing reports such as the ID columns. The data model will be easier for consumers such as report builders to understand if you hide fields which will not be used and add unnecessary clutter.

5. Use **Model** view to update the data model by hiding fields in the **Sales** table that are unnecessary to display in report view.
   a) Navigate to **Model** view in the Power BI Desktop window.
   b) Using the mouse, click on the **CustomerId** column in the **Sales** table to select it.

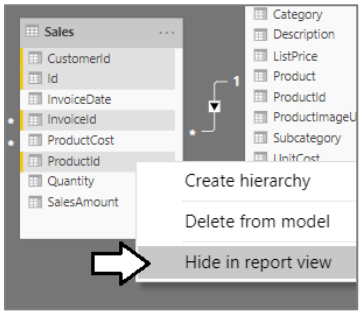


   c) Hold down the **Ctrl** key and use the mouse to additionally select the columns **Id**, **InvoiceId** and **ProductId**.

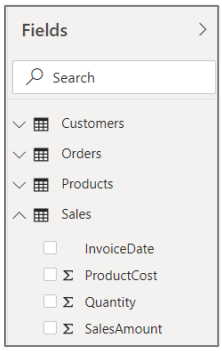


© Critical Path Training. 2020. All Rights Reserved

d) Right-click on one of the selected columns and select the **Hide in report view** command.

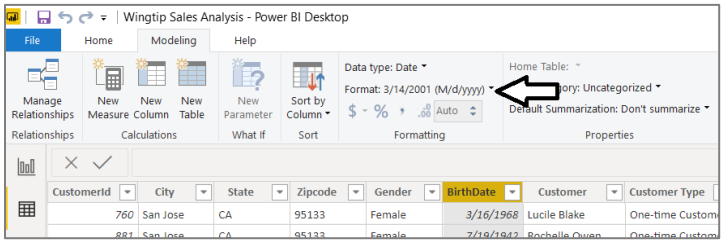


e) Now, navigate back to **Report** view and examine the set of fields displayed in the **Fields** list for the **Sales** table.

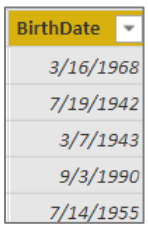


You should be able to see that hiding unnecessary columns from **Report** view makes your data model easier to use. This is especially true in the scenario where you are creating a data model that other less-technical people will be using to build reports & dashboards.

6. 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)**.

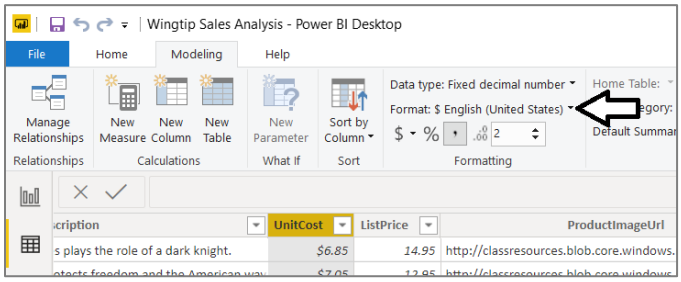


   e) The **BirthDate** column should now reflect the change in formatting.



© Critical Path Training. 2020. All Rights Reserved

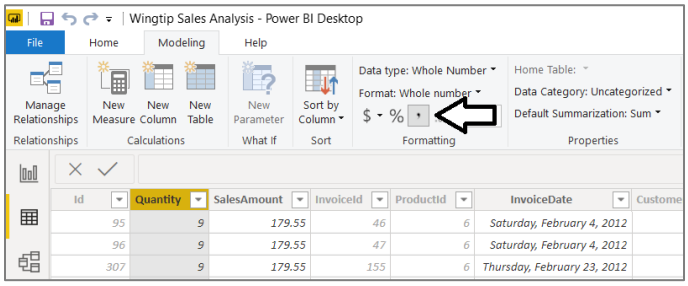
7. Modify the formatting of columns in the **Products** table.
   a) In the **Fields** list on the right, select the **Products** table to display its rows and columns.
   b) Select the **UnitCost** column by clicking on its column header.
   c) Use the **Format** menu button in the ribbon to update 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.



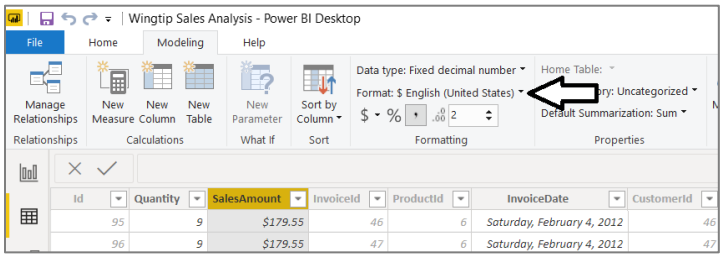
   d) Changing the format setting of the **ListPrice** column to **$ English (United States)** and set the number of decimal places to **2** so it matches the **UnitCost** column.

UnitCost	ListPrice
$6.85	$14.95
$7.05	$12.95
$6.10	$14.95
$2.85	$9.95
$2.85	$9.95

8. 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.



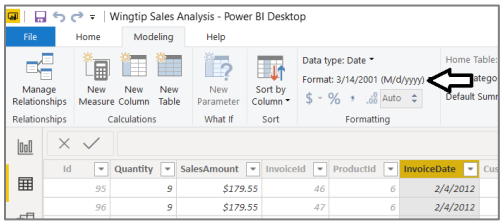
   d) Select the **SalesAmount** column by clicking on its column header.
   e) Modify the formatting of the **SalesAmount** column to **Currency > $ English (United States)** and set the number of decimal places to **2** in the spin control located underneath the **Format** dropdown menu



© Critical Path Training. 2020. All Rights Reserved

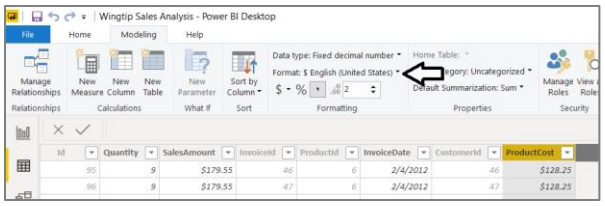
f) Select the **InvoiceDate** column by clicking on its column header.

g) Modify the formatting of the **InvoiceDate** 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.



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

Exercise 2: Create Calculated Columns using DAX

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 the report in the current project.

1. Create a calculated column in the **Customers** table named **Age** to indicate the age of the customer.

a) Navigate to **Data** view.

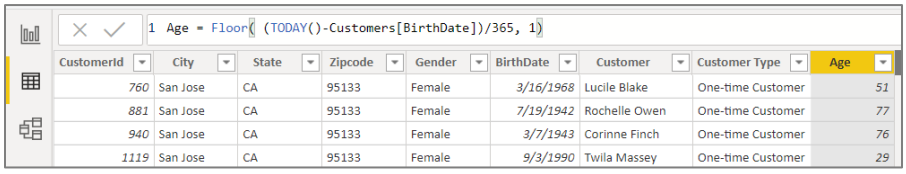
b) Select the **Customers** table in the **Fields** list.

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**.

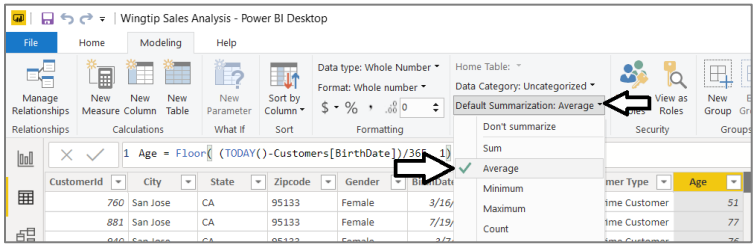
```
Age = Floor( (TODAY()-Customers[BirthDate])/365, 1)
```

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	51
881	San Jose	CA	95133	Female	7/19/1942	Rochelle Owen	One-time Customer	77
940	San Jose	CA	95133	Female	3/7/1943	Corinne Finch	One-time Customer	76
1119	San Jose	CA	95133	Female	9/3/1990	Twila Massey	One-time Customer	29

f) Use the **Default Summarization** dropdown menu to change the **Default Summarization** setting for **Age** to **Average**.

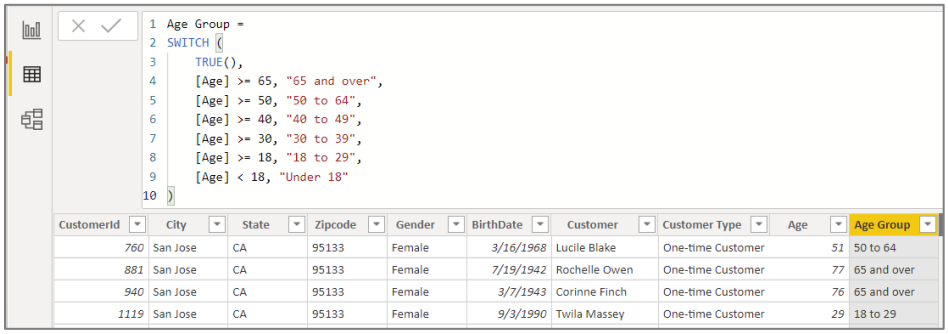


© Critical Path Training. 2020. All Rights Reserved

2. 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 **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 **Age Group**.

```
Age Group =
SWITCH (
    TRUE(),
    [Age] >= 65, "65 and over",
    [Age] >= 50, "50 to 64",
    [Age] >= 40, "40 to 49",
    [Age] >= 30, "30 to 39",
    [Age] >= 18, "18 to 29",
    [Age] < 18, "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	51	50 to 64
881	San Jose	CA	95133	Female	7/19/1942	Rochelle Owen	One-time Customer	77	65 and over
940	San Jose	CA	95133	Female	3/7/1943	Corinne Finch	One-time Customer	76	65 and over
1119	San Jose	CA	95133	Female	9/3/1990	Twila Massey	One-time Customer	29	18 to 29

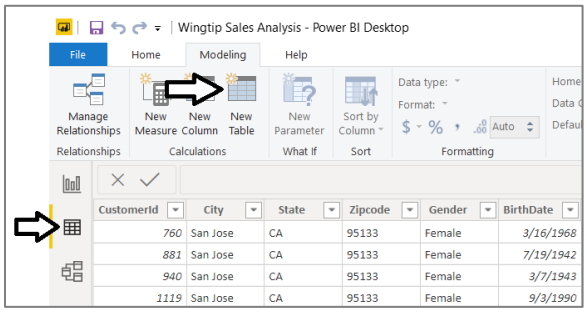
It's a best practice to format DAX expressions with line breaks and indenting to make them easier to read and maintain. Unfortunately, Power BI Desktop offers no built-in features to format DAX expression as you write them. However, there is a free DAX formatting tool for free on the Internet at https://www.daxformatter.com/. This tool allows you to paste your DAX expressions into a web page and it generates the properly-formatted output which you can copy-and-paste back into the DAX expression editor in Power BI Desktop.

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

Exercise 3: Create a Dynamic Lookup Table using DAX

In this exercise, you will extend the data model by adding a new lookup table named **SalesRegions** which assigns each state to a geographic sales region. This will involve creating a new table using a DAX expression which calculates the start date and the end date dynamically. Once you have created the **SalesRegions** table, you must then create a table relationship between the **SalesRegions** table and the **Customers** table to integrate the new table into the 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.

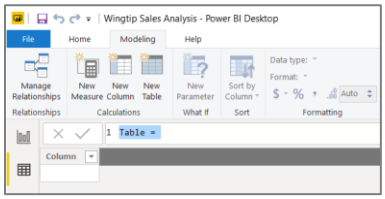
1. Create the **SalesRegions** table using a DAX expression.
   a) Navigate to Data view in the **Wingtip Sales Analysis.pbix** project.
   b) Click the **New Table** button to create a new table using DAX.



CustomerId	City	State	Zipcode	Gender	BirthDate
760	San Jose	CA	95133	Female	3/16/1968
881	San Jose	CA	95133	Female	7/19/1942
940	San Jose	CA	95133	Female	3/7/1943
1119	San Jose	CA	95133	Female	9/3/1990

© Critical Path Training. 2020. All Rights Reserved

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

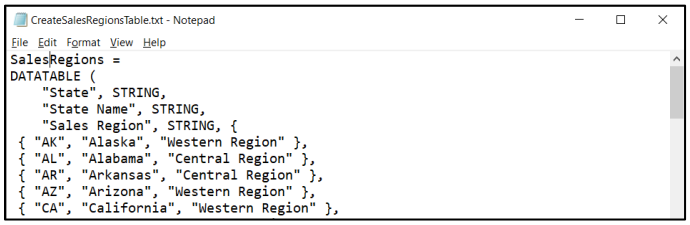


Instead of having you write one heck-of-a-long DAX expression, the lab exercise will have you copy and paste the required DAX expression from a text file in the **Students** folder.

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
```

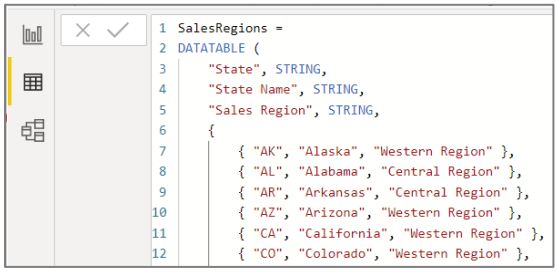
e) Take a moment to inspect the DAX expression inside **CreateSalesRegionsTable.txt**.



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.

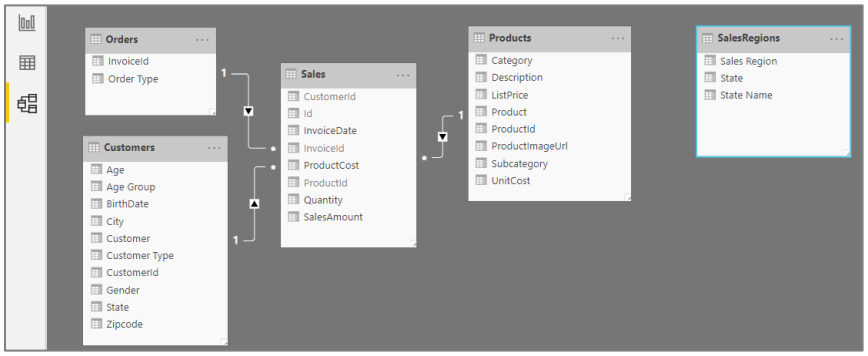
h) Press the **ENTER** key to create the new table named **SalesRegions**.



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

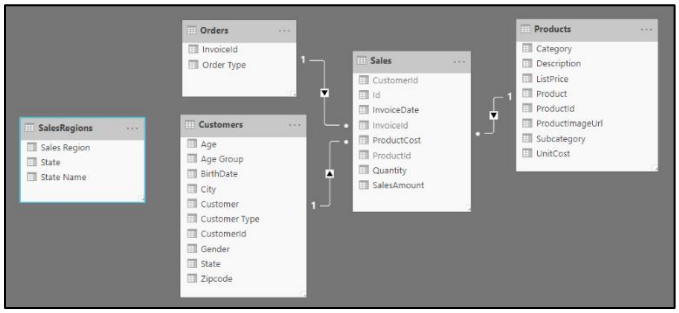
a) Navigate to **Relationship** view.

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



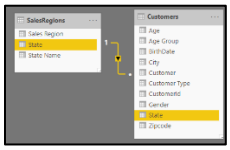
© Critical Path Training. 2020. All Rights Reserved

c) Rearrange layout of tables in **Model** view 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 confirm that a new table 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.

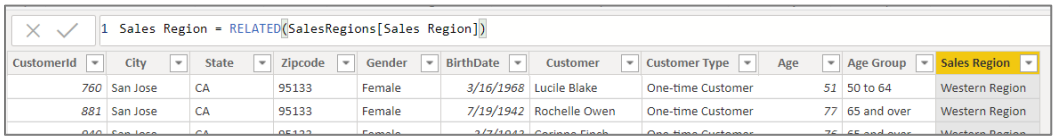
c) Create a new calculated column by clicking the **New Column** button in the ribbon.

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	51	50 to 64	Western Region
881	San Jose	CA	95133	Female	7/19/1942	Rochelle Owen	One-time Customer	77	65 and over	Western Region

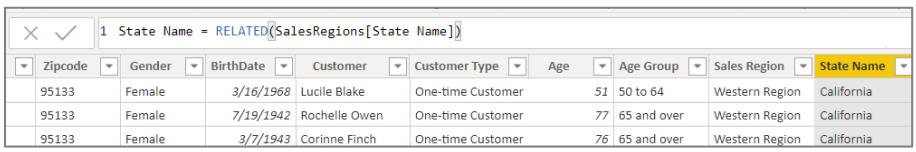
4. Add a calculated column to the **Customers** table named **State Name** to display the full state name.

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.

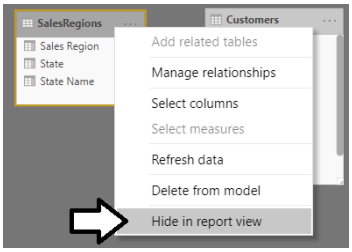


Zipcode	Gender	BirthDate	Customer	Customer Type	Age	Age Group	Sales Region	State Name
95133	Female	3/16/1968	Lucile Blake	One-time Customer	51	50 to 64	Western Region	California
95133	Female	7/19/1942	Rochelle Owen	One-time Customer	77	65 and over	Western Region	California
95133	Female	3/7/1943	Corinne Finch	One-time Customer	76	65 and over	Western Region	California

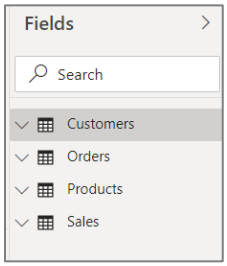
© Critical Path Training. 2020. All Rights Reserved

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.

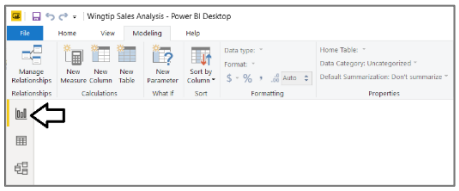
5. Hide the **SalesRegions** table from report view.
   a) Navigate to relationship view.
   b) Right-click on the the **SalesRegions** table



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

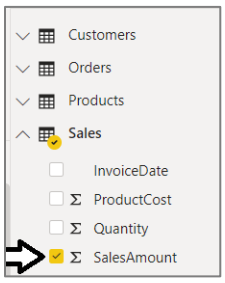


6. See the effect of your formatting by adding a visual to a report.
   a) Navigate to report view. There should be an empty report for the project with a single page named **Page 1**.



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

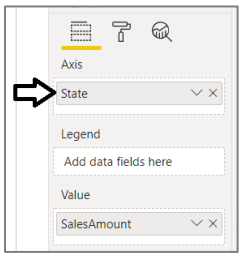
   c) Create a new visual in the report by selecting the checkbox for the **SalesAmount** column in the **Fields** list.



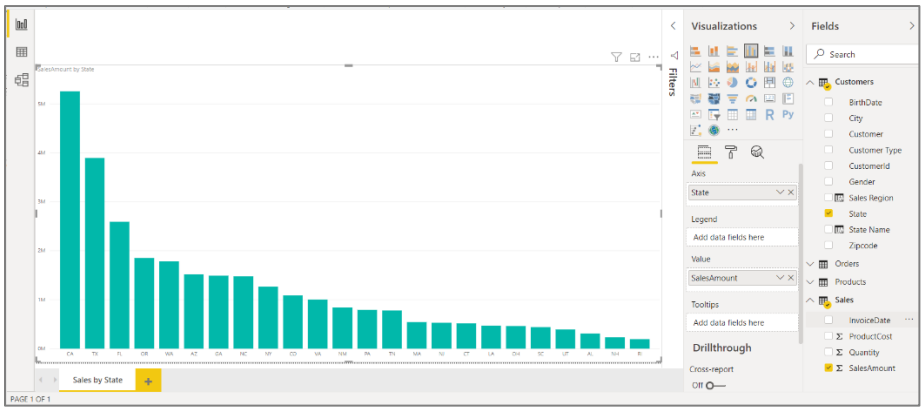
© Critical Path Training. 2020. All Rights Reserved

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**.

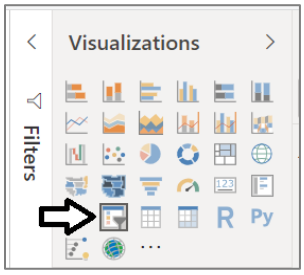
e) Drag and drop the **State** field from the **Fields** list into the **Axis** well in the **Fields** pane.



f) Use your mouse to resize the visual to be as wide as the hosting report page as shown in the following screenshot.

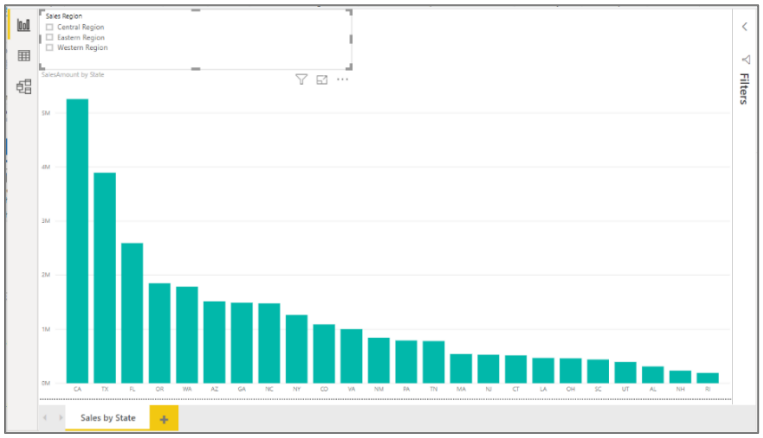


7. Add a slicer visual to the report to filter customer state by sales region.
   a) Navigate to **Report** view if you are not already there.
   b) Make sure that no visuals are selected on the page so that you can create a new visual.
   c) Select the checkbox for the **Sales Region** column in the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



© Critical Path Training. 2020. All Rights Reserved

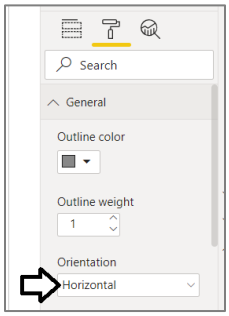
e) Using the mouse, reposition the slicer visual so it appears in the upper, left-hand corner of the page.



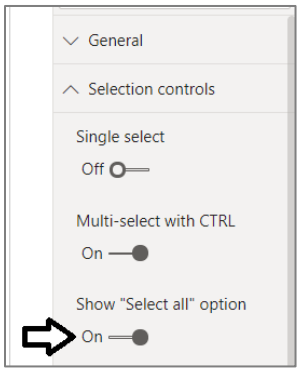
You will now change the alignment of the slicer visual from a vertical layout to a horizontal layout.

f) With the slicer visual selected, navigate to the **General** section of the **Format** pane.

g) Change the **Orientation** property to a value of **Horizontal**.



h) Enable the **Show "Select all" option** property in the **Selection controls** section by setting its value to **On**.



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

© Critical Path Training. 2020. All Rights Reserved

j) Click on the different sales regions in the slicer visual to observe its filtering effect. If you click on **Western Region**, the column chart should only show states assigned to a **Sales Region** of **Western Region**.

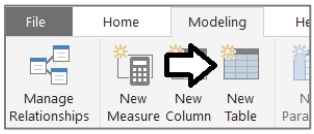
8. Save the work you have done by clicking the **Save** button in the upper left corner of the Power BI Desktop window.

Oh no! There's a problem with the default sort order of the **Sales Region** column. Currently, column values are 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 new table with DAX named **SalesRegionsSort** 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.

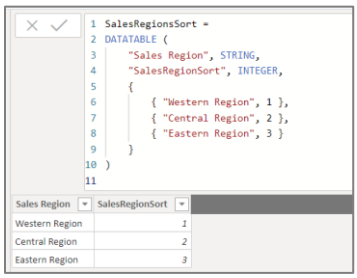
```
C:\Student\Modules\03_DataModeling\Lab\DAX\CreateSalesRegionsSortTable.txt
```

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 }
    }
)
```

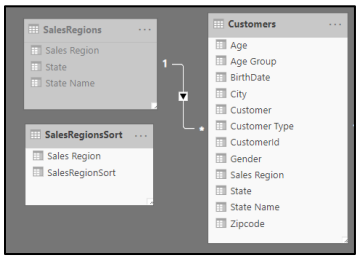
f) Select the entire contents of **CreateSalesRegionsSortTable.txt** and then copy it into the Windows clipboard.

g) Press the ENTER key to submit your DAX expression and to create the new table.

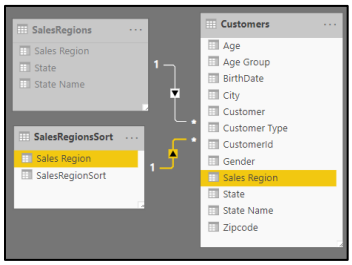


© Critical Path Training. 2020. All Rights Reserved

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



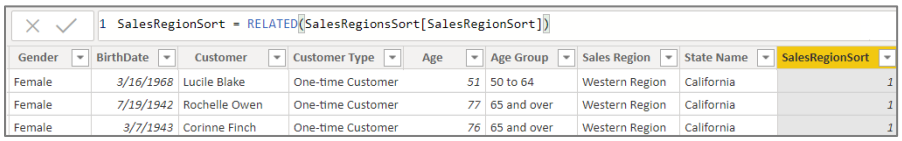
    c) Drag the **Sales Region** column from **SalesRegionsSort** and drop it on the **Sales Region** column in **Customers**.



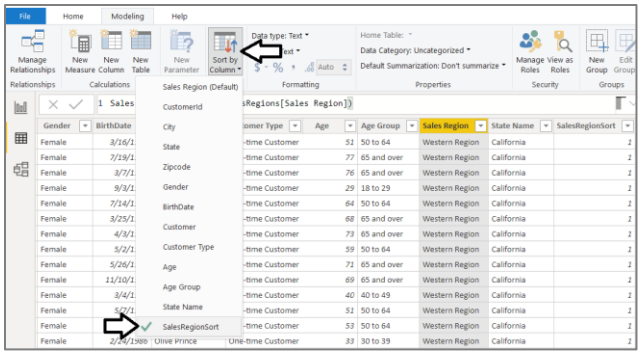
You should be able to verify that a one-to-many relationship has been created between **SalesRegionsSort** and **Customers**.

11. Add the **SalesRegionSort** column to the **Customers** table as a calculated column.
    a) Navigate to Data view and select the **Customers** table in the **Fields** list.
    b) Click the **New Column** button in the ribbon to create a new calculate column.
    c) Type in the following DAX expression to create a new calculated column named **SalesRegionSort**.

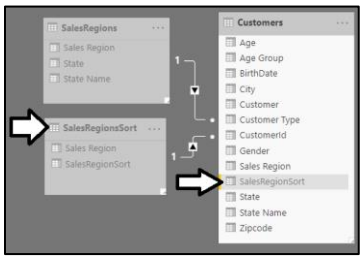
```
SalesRegionSort = RELATED(SalesRegionsSort[SalesRegionSort])
```



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 the **Report** view of your data model by hiding the implementation details of your custom sort order.
    a) Navigate to **Model** 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**.

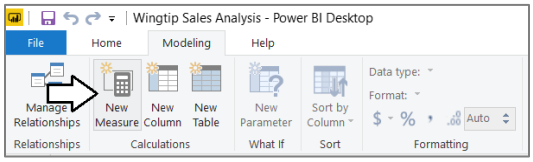


14. Verify your custom sort order is working properly.
    a) Return to Report View and examine the slicer which shows the sales regions.
    b) Verify that the sort order now has **Western Sales** first followed by **Central Region** and then **Eastern Region**.

Exercise 4: Create Measures using DAX

In this exercise you will create four measures named **Sales Revenue, Units Sold, 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.

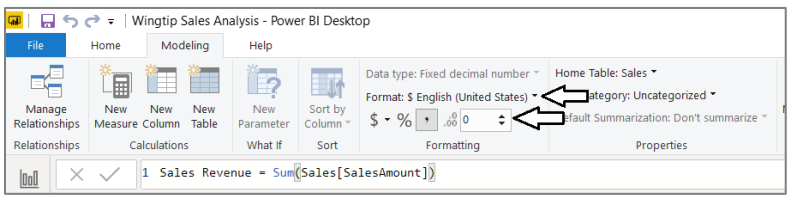
1. Create a measure in the **Sales** table named **Sales Revenue** to perform a sum aggregation on the **SalesAmount** column.
    a) Navigate to **Data** view and select the **Sales** table from the **Fields** list.
    b) Click the **New Measure** button in the **Modeling** tab of the ribbon to add a new measure to the **Sales** table.



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

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

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

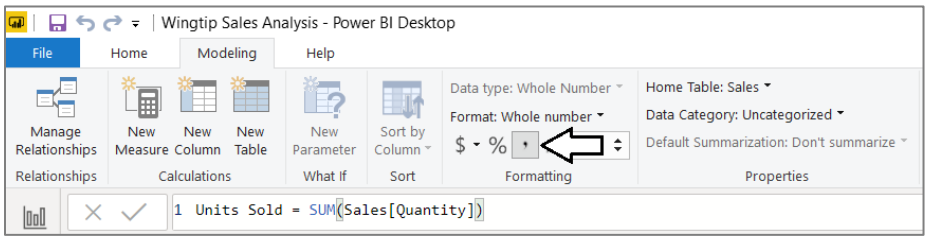


© Critical Path Training. 2020. All Rights Reserved

2. Create a measure in the **Sales** table named **Units Sold** to perform a sum aggregation on the **Quantity** column.
   a) Select the **Sales** table in the **Fields** list and then click the **New Measure** button in the Modeling tab of the ribbon.
   b) Enter to following DAX expression into the formula bar to create the measure named **Units Sold**.

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

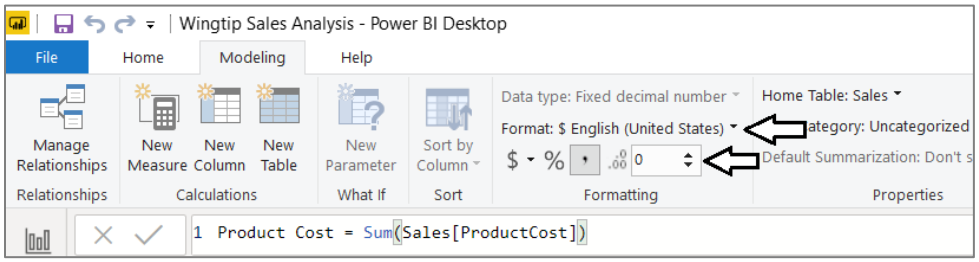
   c) Press the **ENTER** key to add the measure to data model.
   d) Modify the formatting by clicking and selecting the **Comma (,)** button on the ribbon to add a comma separator.



3. Create a measure in the **Sales** table named **Product Cost** to perform a sum aggregation on the **ProductCost** column.
   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])
```

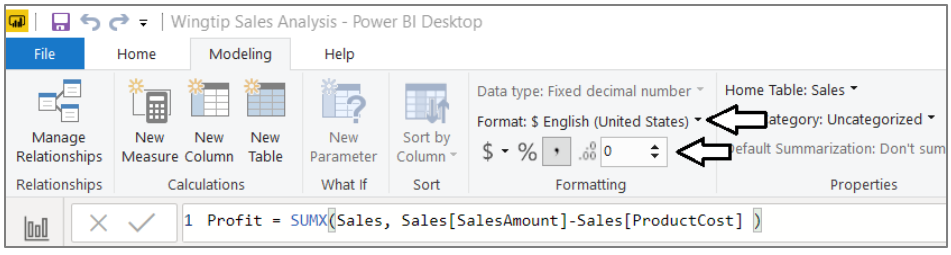
   c) Press the **ENTER** key to add the measure to data model.
   d) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Currency > English (United States)**.
   e) Use the spinner control below the **Format** menu to set the number of decimal places shown to zero.



4. Create a new measure in the **Sales** table named **Profit** to sum the difference between **SalesAmount** and **ProductCost**.
   a) Select the **Sales** table in the **Fields** list and then click the **New Measure** button in the ribbon.
   b) Enter to following DAX expression into the formula bar to create a new measure named **Profit**.

```
Profit = SUMX(Sales, Sales[SalesAmount]-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.

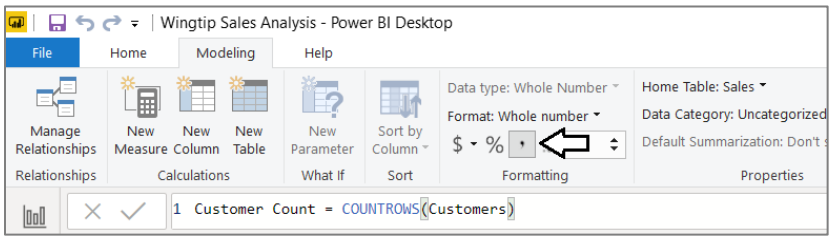


© Critical Path Training. 2020. All Rights Reserved

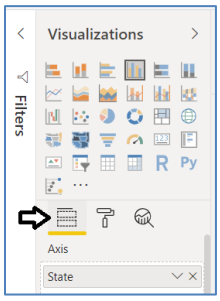
5. Create a measure in the **Sales** table named **Customer Count** to count the number of rows in the **Customers** table.
   a) Make sure the **Sales** table is selected and then click the **New Measure** button in the ribbon.
   b) Enter the following DAX expression into the formula bar to create the measure named **Customer Count**.

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

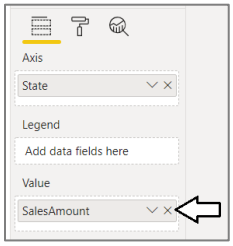
   c) Press the **ENTER** key to add the measure to data model.
   d) Modify the formatting by clicking and selecting the Comma button on the ribbon to add a comma separator.



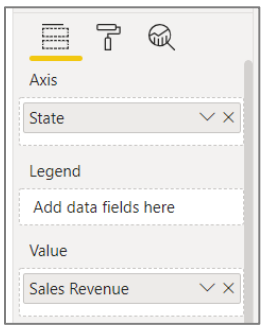
6. Update the Column chart visual to use the new measures you've just created.
   a) Navigate to **Report** view.
   b) Select the column chart visual and then select the **Fields** pane.



   c) Currently, the **Value** well should contain the **SalesAmount** column.

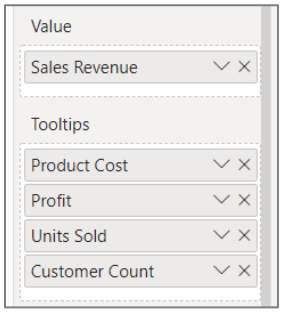


   d) Remove the **SalesAmount** column from the **Values** well and replace it with the **Sales Revenue** measure.

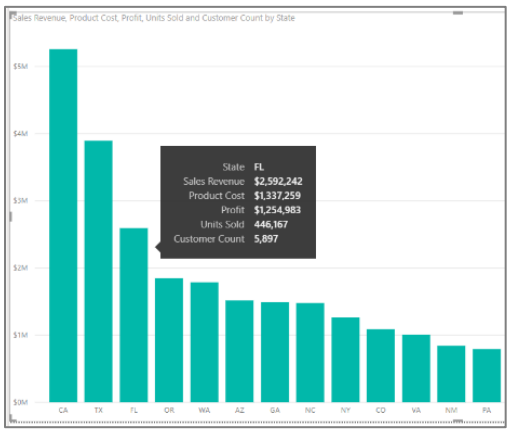


© Critical Path Training. 2020. All Rights Reserved

e) Locate the **Tooltips** well and add the measures named **Product Cost**, **Profit**, **Units Sold** and **Customer Count**.

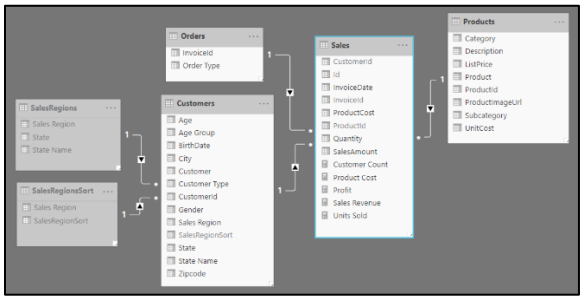


f) Hover your mouse over bars in the bar chart and you should see all the measures displayed in the tool tip.



You will complete your work in this exercise by cleaning up the data model by hiding fields in report view.

7. Simplify your data model by hiding fields that will not be used when building reports.
   a) Navigate to **Model** view so you can see all the tables in the project's data model.
   b) Using the mouse, resize each table in **Model** view so that it properly displays all its fields without the need for a scrollbar.

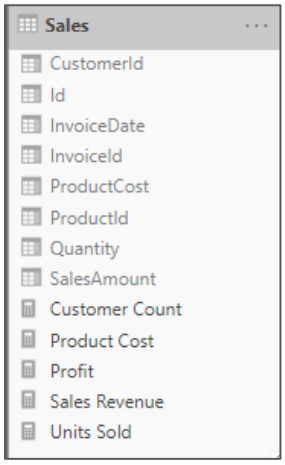


   c) Use the **Hide in Report View** command to hide the following columns.
      i) The **InvoiceId** column in the **Orders** table.
      ii) The **CustomerId** column and the **BirthDate** column from the **Customers** table.
      iii) The **ProductId** column, the **UnitCost** column and the **ListPrice**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

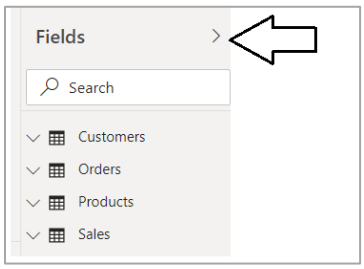
© Critical Path Training. 2020. All Rights Reserved

a) Configure the **Sales** table as a fact table by hiding all columns and only displaying measures.

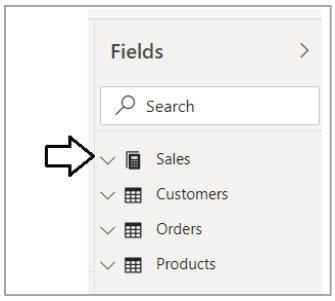


You should be able to verify that all fields in the **Sales** table are hidden except for the 5 measures. Now it's time to see what effect this has to the **Fields** list when you are in **Report** view.

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.

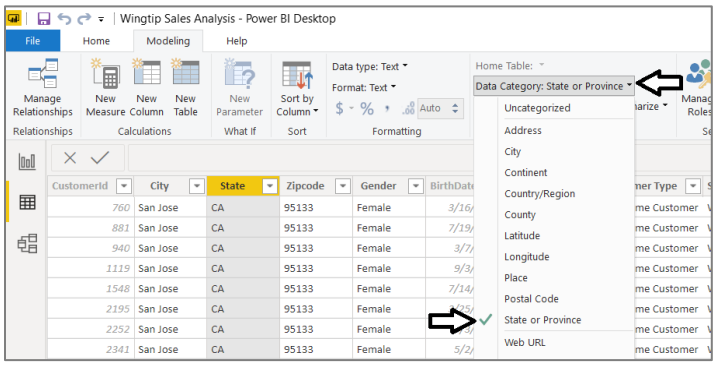
While this exercise was not complicated, it's important that you know how to simplify a data model so it's easier for others to use.

© Critical Path Training. 2020. All Rights Reserved

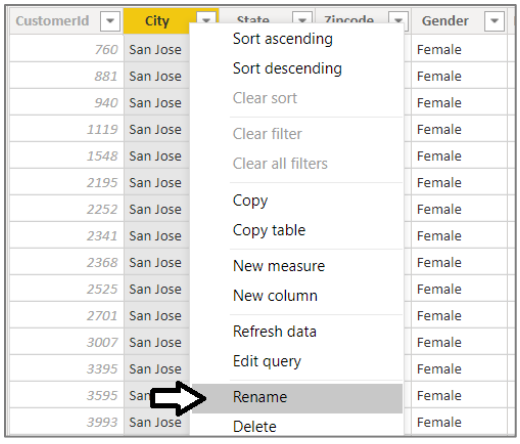
Exercise 5: Configure Geolocation Columns using Data Categories

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.

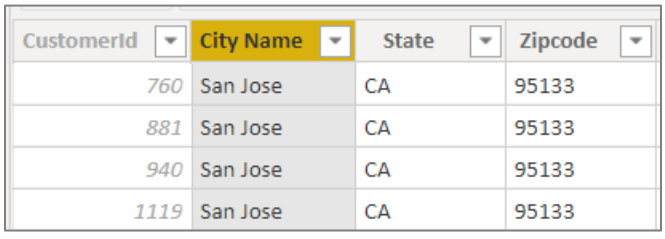
1. Configure geolocation metadata for the **State** field in the **Customers** table.
   a) Return to **Data** View.
   b) From the **Fields** list on the right, select the **State** field from the **Customers** table.
   c) Drop down the **Data Category** menu from the ribbon and select **State or Province**.



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

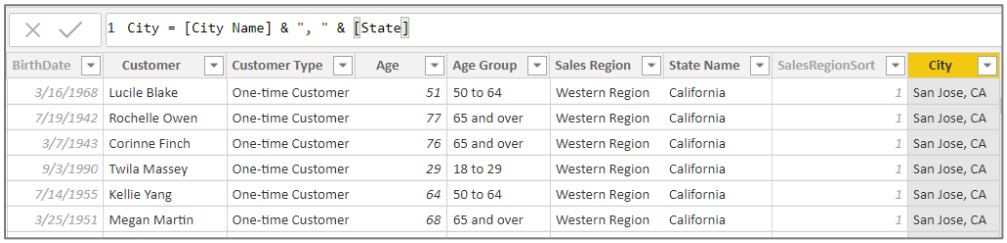
   d) Navigate to the **Modeling** tab in the ribbon and make sure the **Customers** table is still selected.
   e) Click the **New Column** tab to create a new calculated column.

© Critical Path Training. 2020. All Rights Reserved

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

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

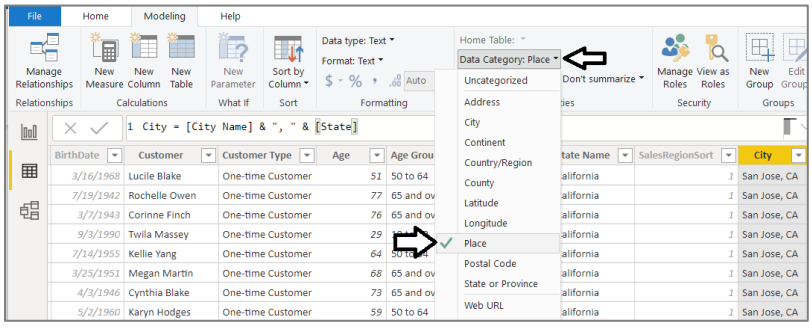
g) The **City** column should display the city and the state which will remove ambiguity when used as a geolocation field.



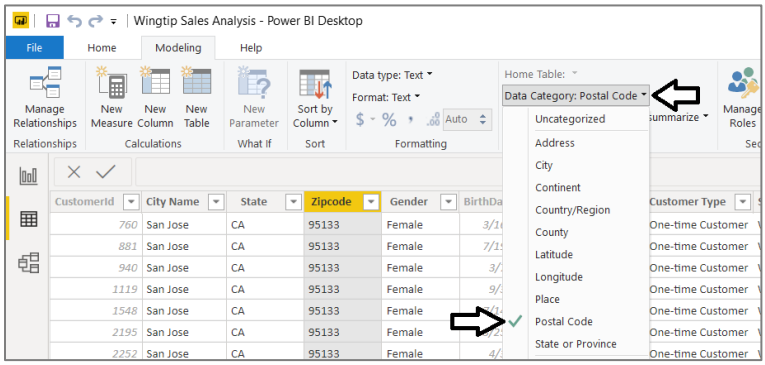
BirthDate	Customer	Customer Type	Age	Age Group	Sales Region	State Name	SalesRegionSort	City
3/16/1968	Lucile Blake	One-time Customer	51	50 to 64	Western Region	California	1	San Jose, CA
7/19/1942	Rochelle Owen	One-time Customer	77	65 and over	Western Region	California	1	San Jose, CA
3/7/1943	Corinne Finch	One-time Customer	76	65 and over	Western Region	California	1	San Jose, CA
9/3/1990	Twila Massey	One-time Customer	29	18 to 29	Western Region	California	1	San Jose, CA
7/14/1955	Kellie Yang	One-time Customer	64	50 to 64	Western Region	California	1	San Jose, CA
3/25/1951	Megan Martin	One-time Customer	68	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**.



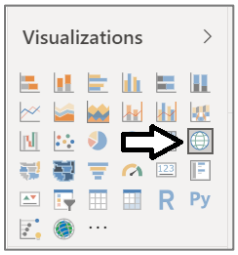
4. Configure geolocation metadata for the **Zipcode** field in the **Customers** table.
   a) Return to **Data** view.
   b) From the **Fields** list on the right, select the **Zipcode** field from the **Customers** table.
   c) Drop down the **Data Category** menu from the ribbon and select **Postal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



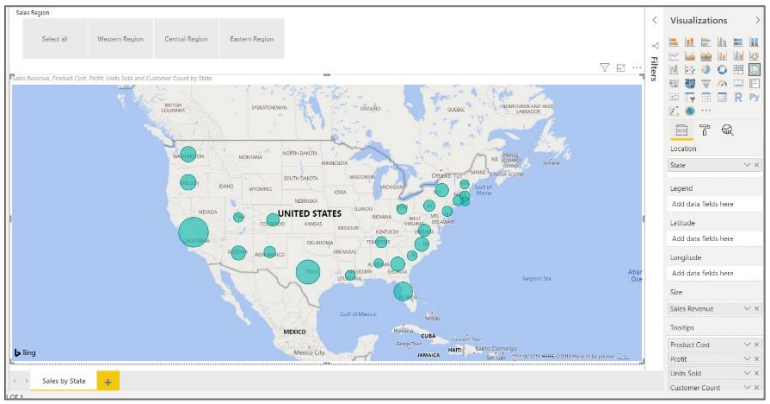
Now that you have configured fields in the **Customers** table with geolocation metadata, it's time to put this metadata to use by creating a new report page with a map visual to visualize how sales revenue is distributed over geographic regions.

© Critical Path Training. 2020. All Rights Reserved

5. Change to the clustered bar chart visual into a **Map** visual.
   a) Return to **Report** view.
   b) Select he column chart visual.
   c) Click the **Map** icon in the **Visualizations** list to change the visual into a **Map** visual.

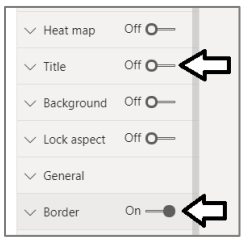


   d) The **Map** visual should display as a bubble map of the United States as shown in the following screenshot.



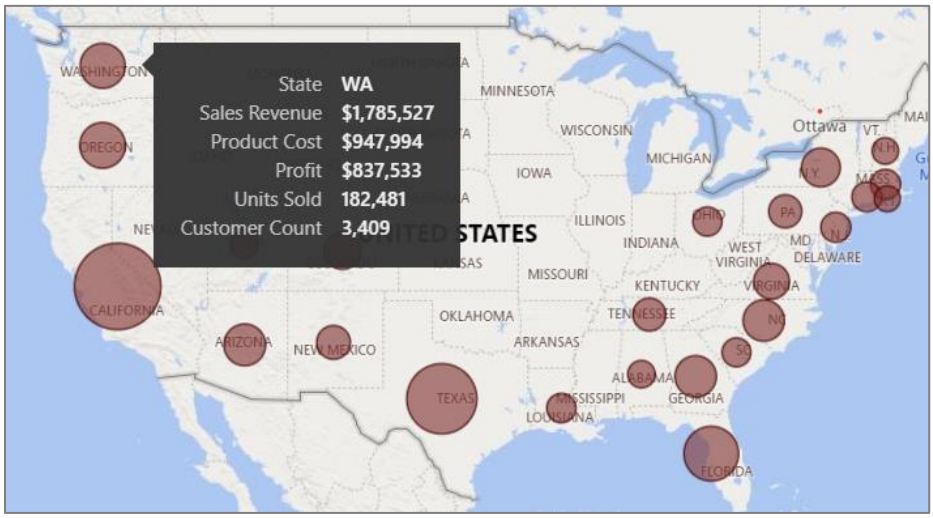
   e) With the **Map** visual selected, navigate to the **Data Colors** section in the **Format** properties pane and change the default color to a dark red so it stands out more clearly on the **Map** visual.

   f) Move down inside the **Format** pane to disable the **Title** setting and enable the **Border** setting.

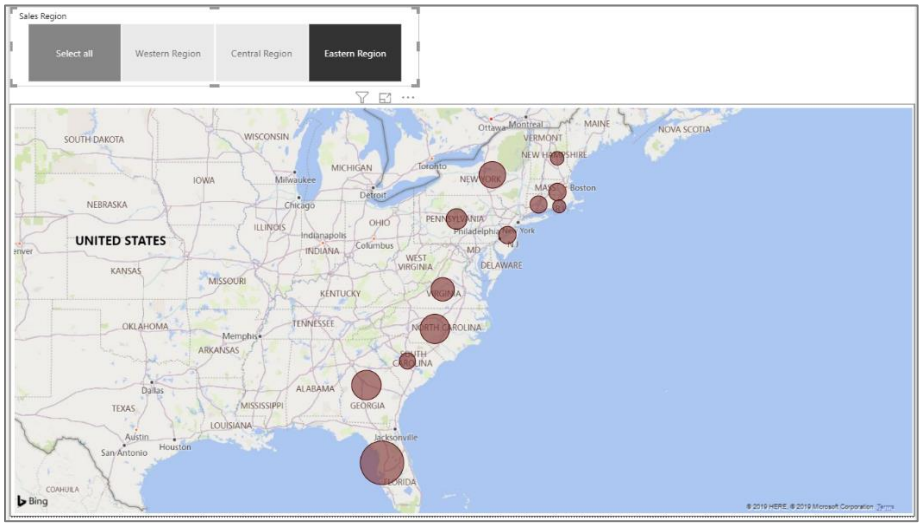


© Critical Path Training. 2020. All Rights Reserved

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

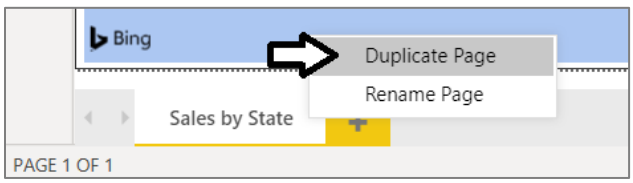


h) Test out the slicer. You should be able to select a single sales region and see the map update to reflect the new filtering.



6. Create a new report page to show sales revenue by city.

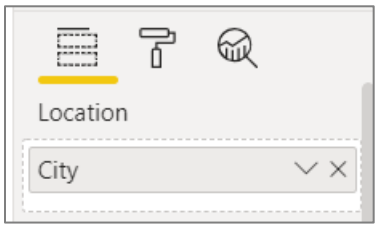
a) Create a new page by right-clicking the **Sales by State** page tab and then selecting the **Duplicate Page** command.



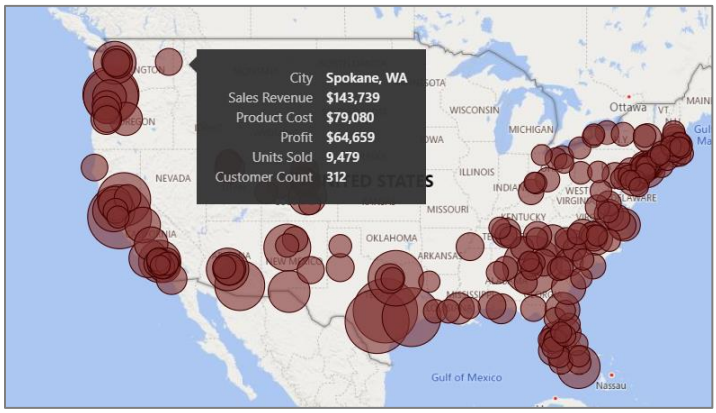
b) Once the new page has been created, rename it to **Sales by City**.

© Critical Path Training. 2020. All Rights Reserved

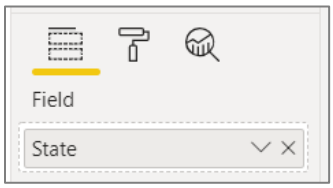
7. Update the **Map** visual on the **Sales by Cities** page to display bubbles for cities instead of states.
   a) Select the **Map visual** and then locate the **Location** well in the **Fields** pane.
   b) Remove the **State** column from the **Location** well and replace it with the **City** column.



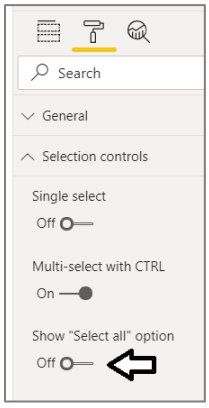
   c) The **Map** visual should now display bubbles for cities instead of states.



8. Update the **Slicer** visual on the **Sales by Cities** page to filter by the **State** column instead of the **Sales Region** column.
   a) Select the **Slicer** visual that is currently configured to filter by **Sales Region**.
   b) In the **Fields** pane. remove **Sales Region** from the **Field** well and replace it with **State**.

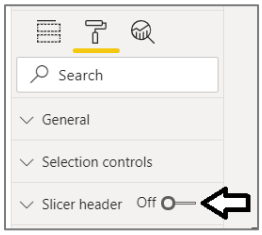


   c) In the **Selection controls** section of the **Format** pane for the slicer visual, set **Show "Select all" option** to **Off**.

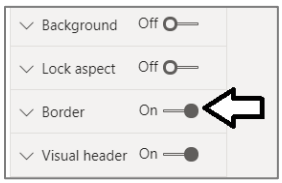


© Critical Path Training. 2020. All Rights Reserved

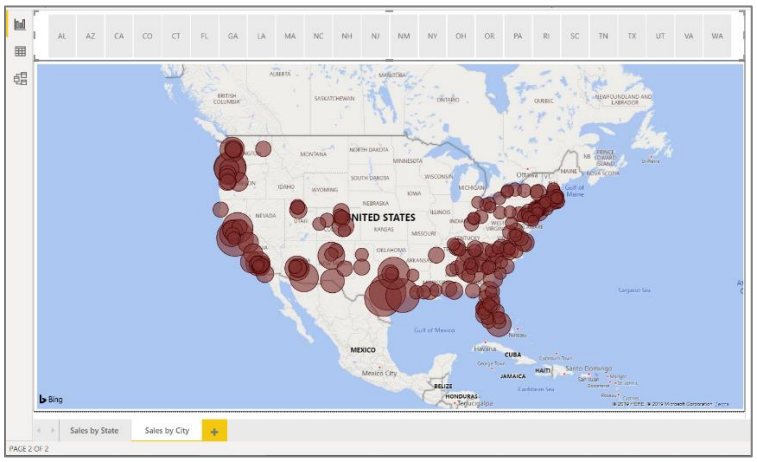
d) Set the **Slicer header** property for the slicer visual to **Off**.



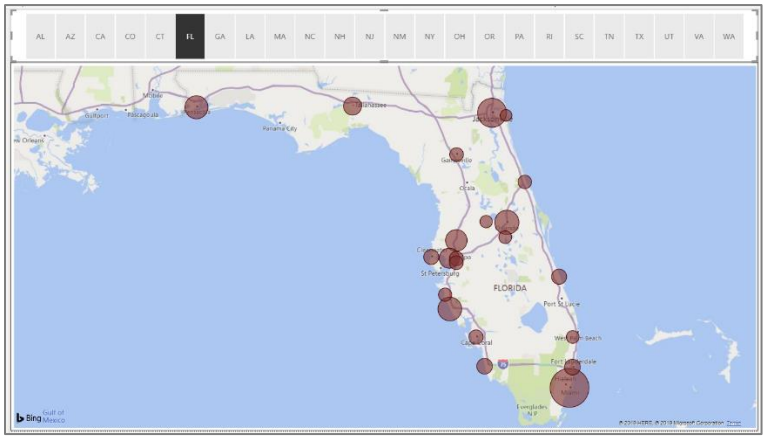
e) Set the **Border** property for the slicer visual to **On**.



f) Reposition the slicer visual and the map visual to match the layout shown in the following screenshot.



g) Use the slicer to select a single state at a time to test your work.



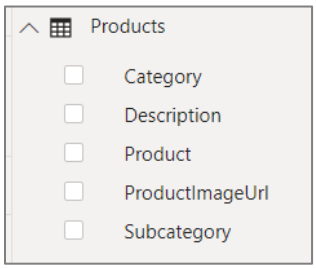
Congratulations for making great progress. You are about half way through this set of lab exercises. Note that the **Solution** folder for this module contains a partial solution named **Wingtip Sales Analysis - Exercise 3.5.pbix** with all the work up to this point.

© Critical Path Training. 2020. All Rights Reserved

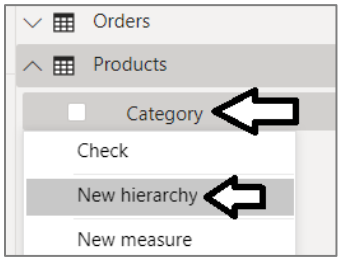
Exercise 6: Extend the Data Model using Dimensional Hierarchies

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

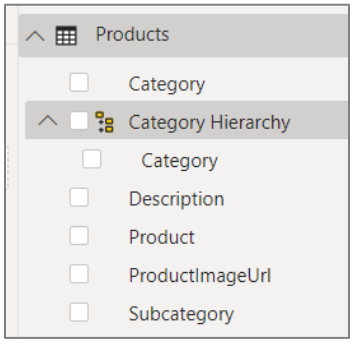
1. Add a new dimensional hierarchy to the **Products** table.
   a) Navigate to **Report** view.
   b) Select the **Products** table in the **Fields** list. It should appear as the one shown in the following screenshot.



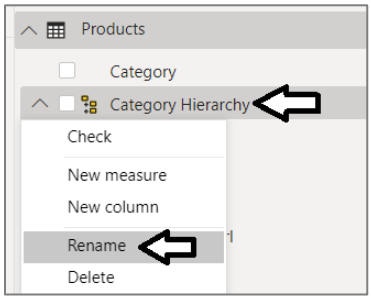
   c) Right-click on the **Category** field and then select the **New Hierarchy** menu command.



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

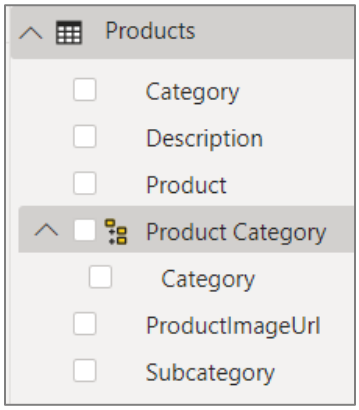


   e) Right-click **Category Hierarchy** and select the **Rename** command.

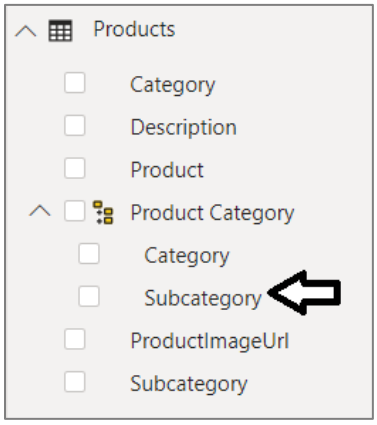


© Critical Path Training. 2020. All Rights Reserved

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

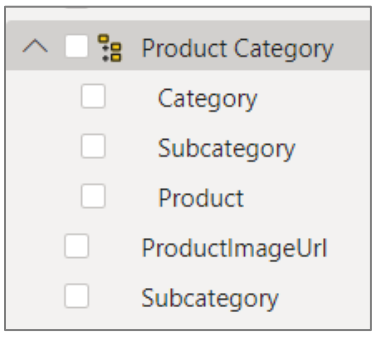


g) Using the mouse, drag-and-drop the **Subcategory** column onto **Product Category** to add this column into the hierarchy.

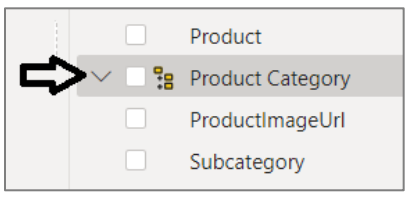


h) Using the mouse, drag-and-drop the **Product** column onto **Product Category** to add this column into the hierarchy.

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

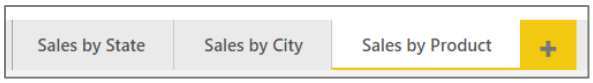


j) Note you can collapse a hierarchy to hide its hierarchy members by clicking the expand/collapse arrow on the left.

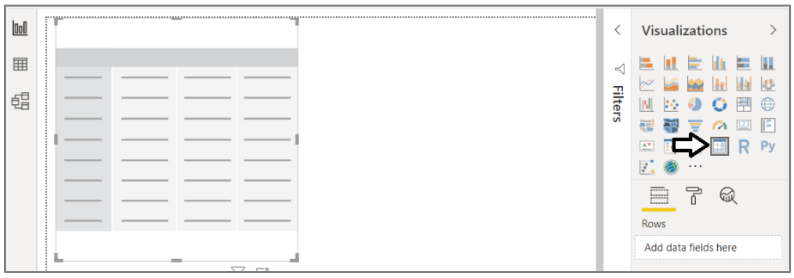


© Critical Path Training. 2020. All Rights Reserved

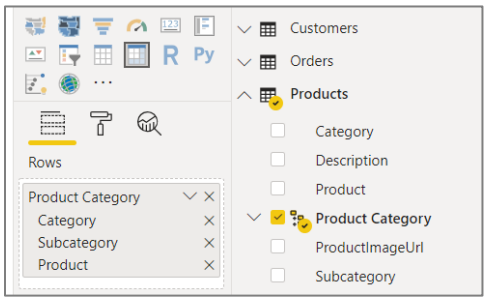
2. Add a new page with a matrix visual to display product sales data.
   a) Add new page to the report and rename it to **Sales by Product**.



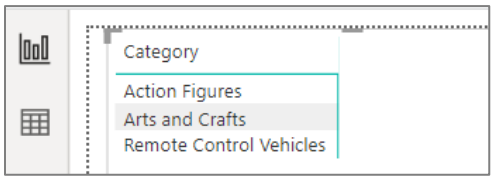
   b) Click the **Matrix** icon in the **Visualizations** list to add a new matrix visual to the page.



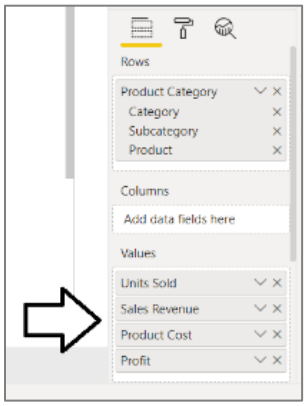
   c) With the matrix visual selected, click the **Fields** pane icon.
   d) Drag and drop the **Product Category** hierarchy into the **Rows** well for the matrix visual



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

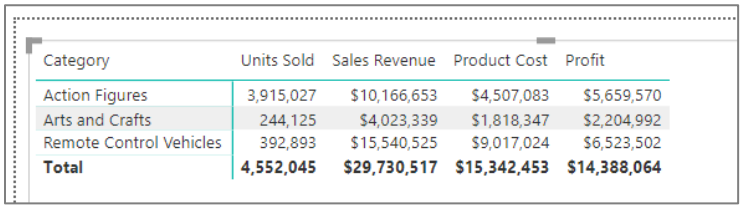


   f) Populate the **Values** well for the matrix visual by adding the measures **Units Sold**, **Sales Revenue**, **Product Cost** and **Profit**.



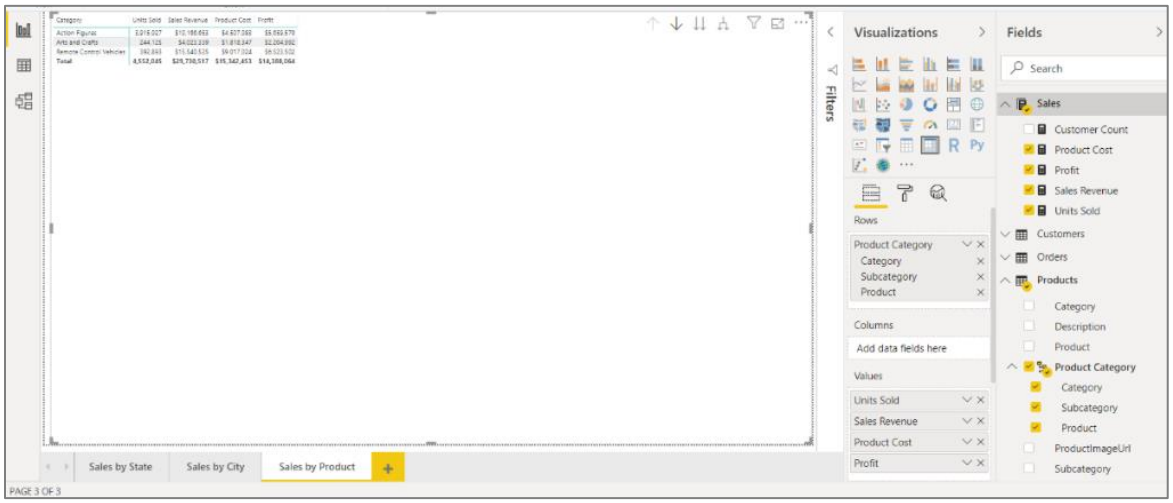
© Critical Path Training. 2020. All Rights Reserved

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



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

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



3. Configure the matrix visual to display all three level of the **Product Category** hierarchy.

a) Locate the toolbar of icons at the top right corner of the matrix visual.

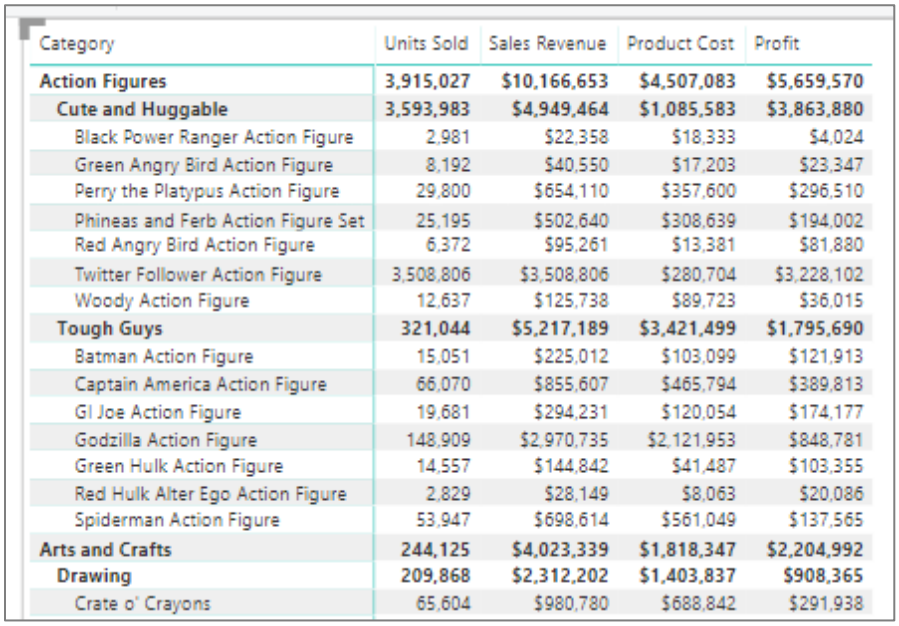
b) Click on the **Expand all** icon in the toolbar to display the next level in the **Product Category** hierarchy.

c) The matrix should now display a second levels of rows for subcategories in addition to the top-level rows for categories.

Category	Units Sold	Sales Revenue	Product Cost	Profit
Action Figures	**3,915,027**	**$10,166,653**	**$4,507,083**	**$5,659,570**
Cute and Huggable	3,593,983	$4,949,464	$1,085,583	$3,863,880
Tough Guys	321,044	$5,217,189	$3,421,499	$1,795,690
Arts and Crafts	**244,125**	**$4,023,339**	**$1,818,347**	**$2,204,992**
Drawing	209,868	$2,312,202	$1,403,837	$908,365
Painting	34,257	$1,711,137	$414,510	$1,296,627
Remote Control Vehicles	**392,893**	**$15,540,525**	**$9,017,024**	**$6,523,502**
Boats	5,323	$175,393	$66,538	$108,855
Cars	90,274	$1,917,031	$1,228,369	$688,662
Helicopter	54,183	$4,294,071	$2,711,579	$1,582,492
Planes	140,363	$6,166,673	$2,950,893	$3,215,780
Trucks	102,750	$2,987,358	$2,059,645	$927,712
Total	**4,552,045**	**$29,730,517**	**$15,342,453**	**$14,388,064**

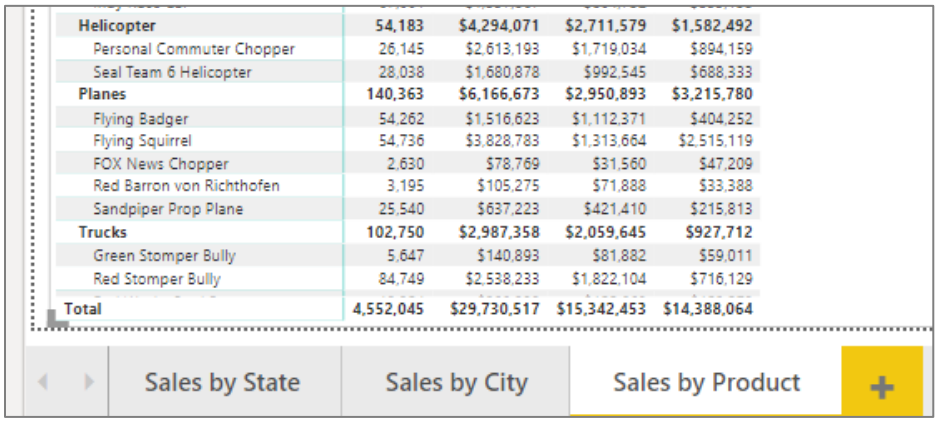
© Critical Path Training. 2020. All Rights Reserved

d) Click on the **Expand all** icon in the toolbar once more to display all three levels of the **Product Category** hierarchy.



Category	Units Sold	Sales Revenue	Product Cost	Profit
Action Figures	**3,915,027**	**$10,166,653**	**$4,507,083**	**$5,659,570**
Cute and Huggable	**3,593,983**	**$4,949,464**	**$1,085,583**	**$3,863,880**
Black Power Ranger Action Figure	2,981	$22,358	$18,333	$4,024
Green Angry Bird Action Figure	8,192	$40,550	$17,203	$23,347
Perry the Platypus Action Figure	29,800	$654,110	$357,600	$296,510
Phineas and Ferb Action Figure Set	25,195	$502,640	$308,639	$194,002
Red Angry Bird Action Figure	6,372	$95,261	$13,381	$81,880
Twitter Follower Action Figure	3,508,806	$3,508,806	$280,704	$3,228,102
Woody Action Figure	12,637	$125,738	$89,723	$36,015
Tough Guys	**321,044**	**$5,217,189**	**$3,421,499**	**$1,795,690**
Batman Action Figure	15,051	$225,012	$103,099	$121,913
Captain America Action Figure	66,070	$855,607	$465,794	$389,813
GI Joe Action Figure	19,681	$294,231	$120,054	$174,177
Godzilla Action Figure	148,909	$2,970,735	$2,121,953	$848,781
Green Hulk Action Figure	14,557	$144,842	$41,487	$103,355
Red Hulk Alter Ego Action Figure	2,829	$28,149	$8,063	$20,086
Spiderman Action Figure	53,947	$698,614	$561,049	$137,565
Arts and Crafts	**244,125**	**$4,023,339**	**$1,818,347**	**$2,204,992**
Drawing	**209,868**	**$2,312,202**	**$1,403,837**	**$908,365**
Crate o' Crayons	65,604	$980,780	$688,842	$291,938

e) Note that when the **Matrix** visual displays all three levels, all the rows just barely fit within the height of the page.

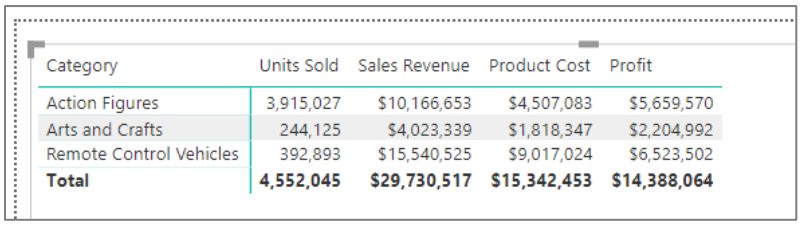


Helicopter	**54,183**	**$4,294,071**	**$2,711,579**	**$1,582,492**
Personal Commuter Chopper	26,145	$2,613,193	$1,719,034	$894,159
Seal Team 6 Helicopter	28,038	$1,680,878	$992,545	$688,333
Planes	**140,363**	**$6,166,673**	**$2,950,893**	**$3,215,780**
Flying Badger	54,262	$1,516,623	$1,112,371	$404,252
Flying Squirrel	54,736	$3,828,783	$1,313,664	$2,515,119
FOX News Chopper	2,630	$78,769	$31,560	$47,209
Red Barron von Richthofen	3,195	$105,275	$71,888	$33,388
Sandpiper Prop Plane	25,540	$637,223	$421,410	$215,813
Trucks	**102,750**	**$2,987,358**	**$2,059,645**	**$927,712**
Green Stomper Bully	5,647	$140,893	$81,882	$59,011
Red Stomper Bully	84,749	$2,538,233	$1,822,104	$716,129
Total	**4,552,045**	**$29,730,517**	**$15,342,453**	**$14,388,064**

Sales by State | Sales by City | Sales by Product | +

f) Click the **Drill up** icon twice to hide all the rows except for the top-level category rows.

g) The matrix visual should now only be displaying three rows for top-level categories.

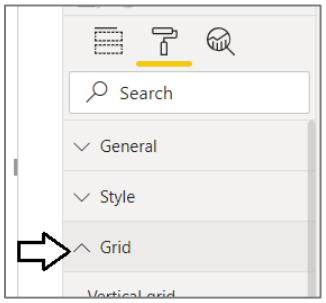


Category	Units Sold	Sales Revenue	Product Cost	Profit
Action Figures	3,915,027	$10,166,653	$4,507,083	$5,659,570
Arts and Crafts	244,125	$4,023,339	$1,818,347	$2,204,992
Remote Control Vehicles	392,893	$15,540,525	$9,017,024	$6,523,502
Total	**4,552,045**	**$29,730,517**	**$15,342,453**	**$14,388,064**

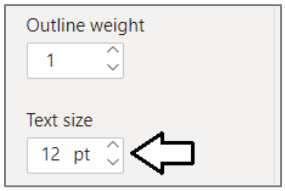
© Critical Path Training. 2020. All Rights Reserved

When Power BI was first introduced, many customer complained that there was not a core visual that acted in the same manner as a Pivot Table in Microsoft Excel. Over the last 3 years, Microsoft has responded by adding a series of updates to the **Matrix** visual to make it behave more like an Excel Pivot Table. You will now configure the matrix to exhibit this Pivit table-like behavior.

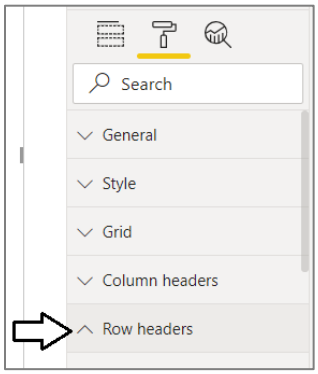
4. Configure the **Matrix** visual to behave like an Excel pivot table.
   a) With the matrix visual selected, expand the **Grid** section in the **Format** pane.



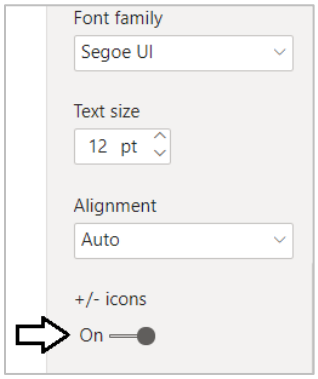
   b) Inside the **Grid** section, locate the **Text** site property and increase its value to **12 pt**.



   c) With the matrix still selected, expand the **Row headers** section.

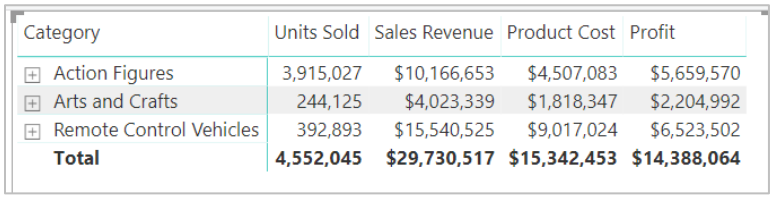


   d) Inside the **Row headers** section, enable the **+/- icons** property by setting it to **On**.



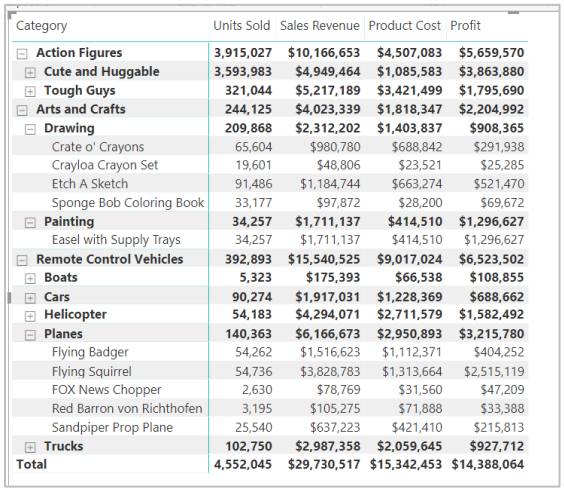
© Critical Path Training. 2020. All Rights Reserved

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



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

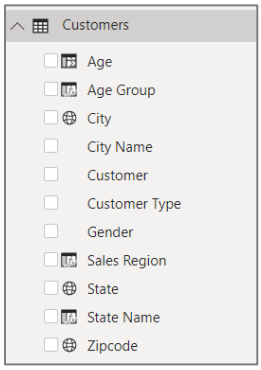
f) Click on the expand/collapse icons to drill into a single subcategory and then into a single product.



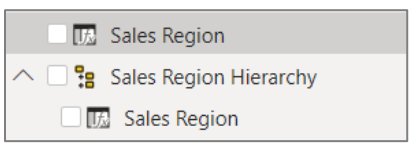
Category	Units Sold	Sales Revenue	Product Cost	Profit
⊟ **Action Figures**	**3,915,027**	**$10,166,653**	**$4,507,083**	**$5,659,570**
⊞ **Cute and Huggable**	**3,593,983**	**$4,949,464**	**$1,085,583**	**$3,863,880**
⊞ **Tough Guys**	**321,044**	**$5,217,189**	**$3,421,499**	**$1,795,690**
⊟ **Arts and Crafts**	**244,125**	**$4,023,339**	**$1,818,347**	**$2,204,992**
⊟ **Drawing**	**209,868**	**$2,312,202**	**$1,403,837**	**$908,365**
Crate o' Crayons	65,604	$980,780	$688,842	$291,938
Crayloa Crayon Set	19,601	$48,806	$23,521	$25,285
Etch A Sketch	91,486	$1,184,744	$663,274	$521,470
Sponge Bob Coloring Book	33,177	$97,872	$28,200	$69,672
⊟ **Painting**	**34,257**	**$1,711,137**	**$414,510**	**$1,296,627**
Easel with Supply Trays	34,257	$1,711,137	$414,510	$1,296,627
⊟ **Remote Control Vehicles**	**392,893**	**$15,540,525**	**$9,017,024**	**$6,523,502**
⊞ **Boats**	**5,323**	**$175,393**	**$66,538**	**$108,855**
⊞ **Cars**	**90,274**	**$1,917,031**	**$1,228,369**	**$688,662**
⊞ **Helicopter**	**54,183**	**$4,294,071**	**$2,711,579**	**$1,582,492**
⊟ **Planes**	**140,363**	**$6,166,673**	**$2,950,893**	**$3,215,780**
Flying Badger	54,262	$1,516,623	$1,112,371	$404,252
Flying Squirrel	54,736	$3,828,783	$1,313,664	$2,515,119
FOX News Chopper	2,630	$78,769	$31,560	$47,209
Red Barron von Richthofen	3,195	$105,275	$71,888	$33,388
Sandpiper Prop Plane	25,540	$637,223	$421,410	$215,813
⊞ **Trucks**	**102,750**	**$2,987,358**	**$2,059,645**	**$927,712**
Total	**4,552,045**	**$29,730,517**	**$15,342,453**	**$14,388,064**

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

5. Add a new dimensional hierarchy to the **Customers** table.
   a) Navigate to **Report** view.
   b) Inspect the **Customers** table in the **Fields** list. It should appear as the one shown in the following screenshot.

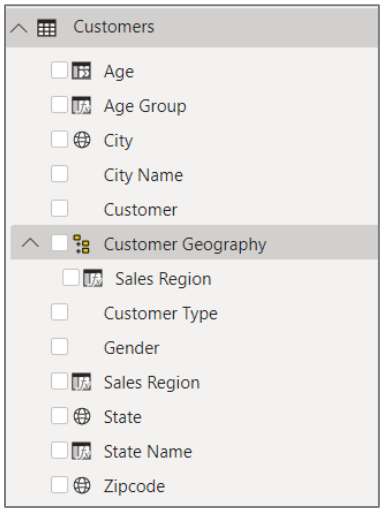


   c) Right-click on the **Sales Region** field and then select the **New Hierarchy** menu command.
   d) You should now see a new dimensional hierarchy in the fields list named **Sales Region Hierarchy**.



© Critical Path Training. 2020. All Rights Reserved

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



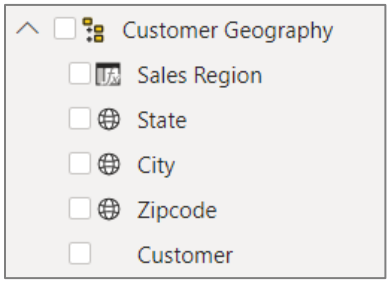
f) Using the mouse, drag-and-drop the **State** column onto **Customer Geography** to add this column into the hierarchy.

g) Using the mouse, drag-and-drop the **City** column onto **Customer Geography** to add this column into the hierarchy.

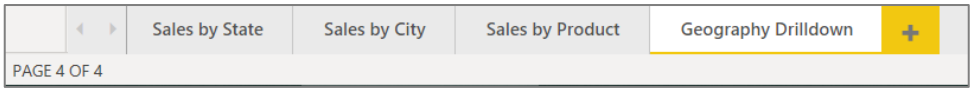
h) Using the mouse, drag-and-drop the **Zipcode** column onto **Customer Geography** to add this column into the hierarchy.

i) Using the mouse, drag-and-drop the **Customer** column onto **Customer Geography** to add this column into the hierarchy.

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

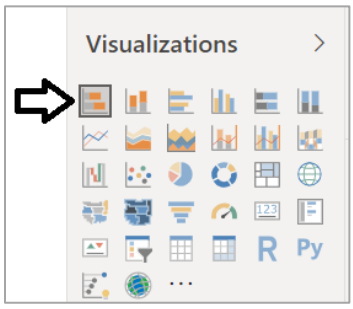


6. Create a new report page and rename it to **Geography Drilldown**.



7. 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.

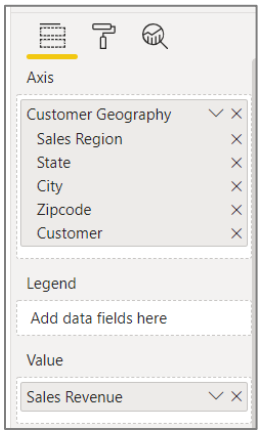


© Critical Path Training. 2020. All Rights Reserved

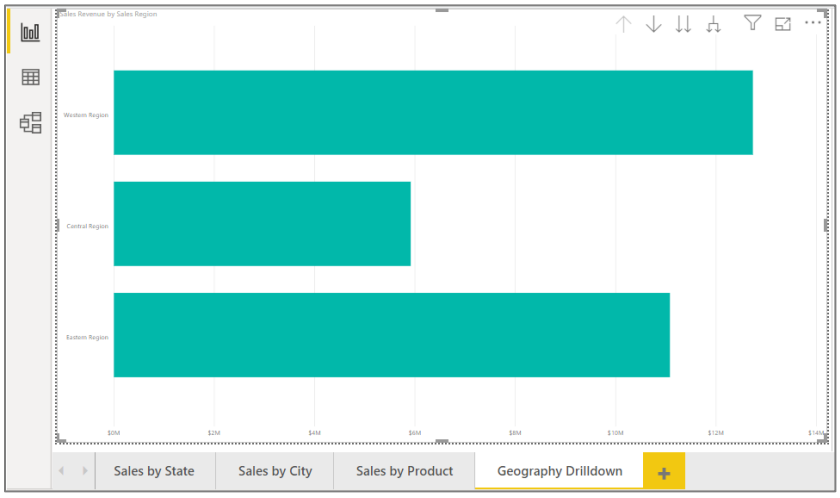
b) Resize the bar chart visual so it takes up the entire height and width of the report page.

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.

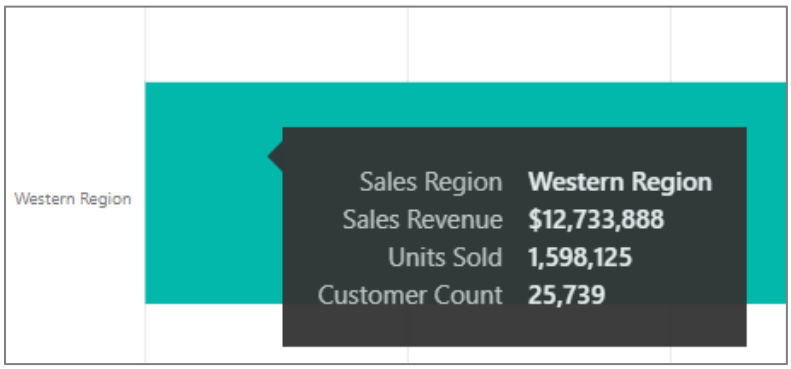


8. Configure **Tooltips** for the bar chart to additionally display **Units Sold** and **Customer Count**.

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

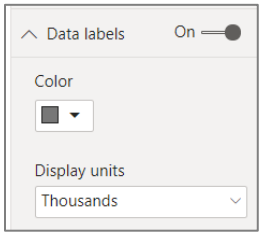
b) Drag and drop the **Customer Count** field from the **Sales** table into the **Tooltips** well.

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



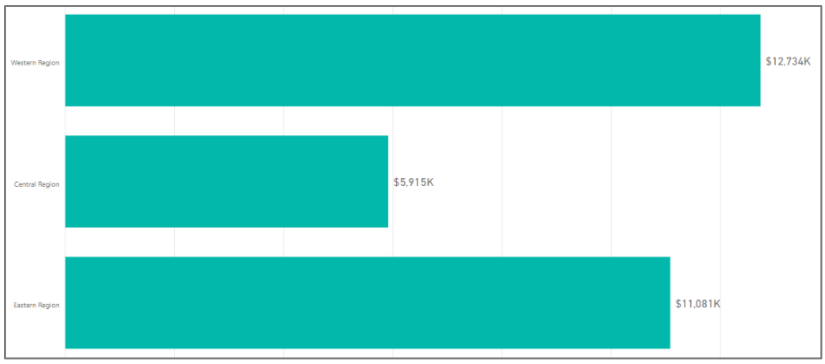
© Critical Path Training. 2020. All Rights Reserved

9. 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**.



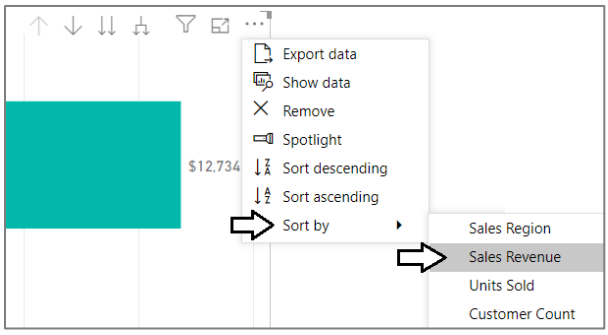
   e) Set the Text Size to **14 pt**.

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



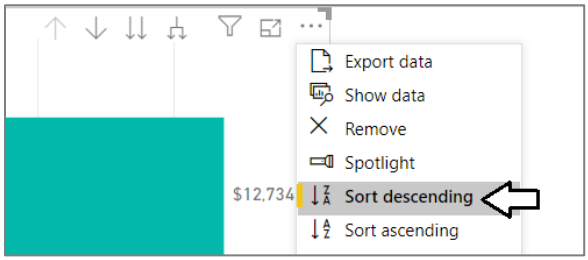
10. Modify the sorting to sorting of the bars in the bar chart to sort by **Sales Revenue** in a descending fashion.
   a) Click the ellipse for the visual flyout menu at the top right of the bar chart visual.

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



© Critical Path Training. 2020. All Rights Reserved

c) Drop down the visual flyout menu a second time and select the **Sort descending** command.



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

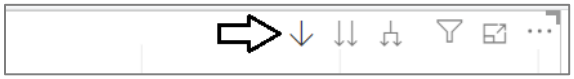
11. Remove the visual title from the bar chart.

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

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

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

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

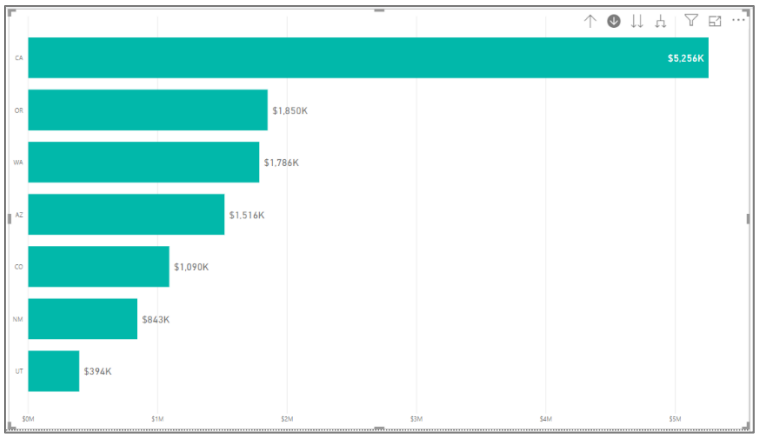


c) Once you have enabled drill down mode, the **Drill Down** button appears as a dark circle with a white downward-facing arrow.

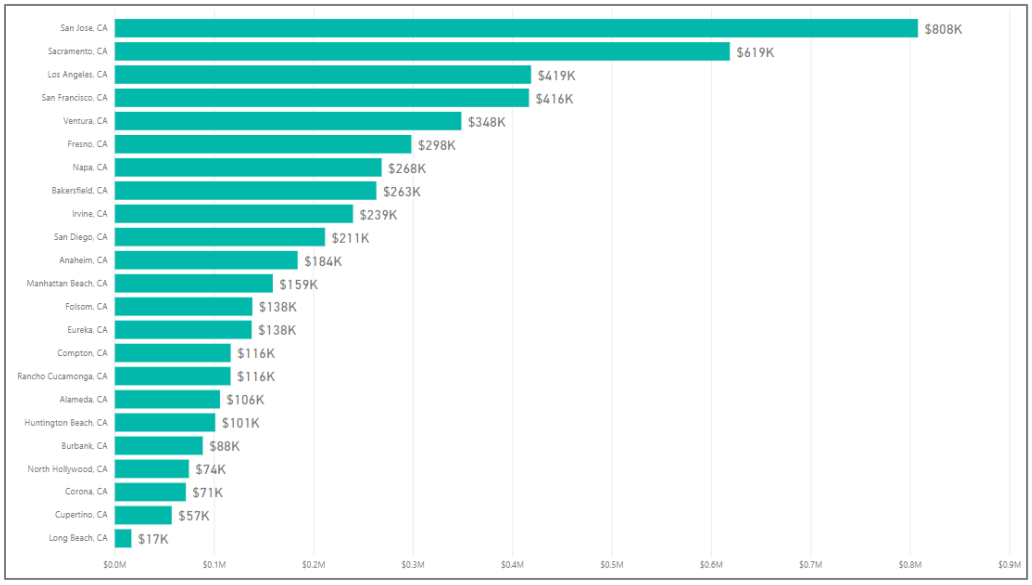
© Critical Path Training. 2020. All Rights Reserved

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

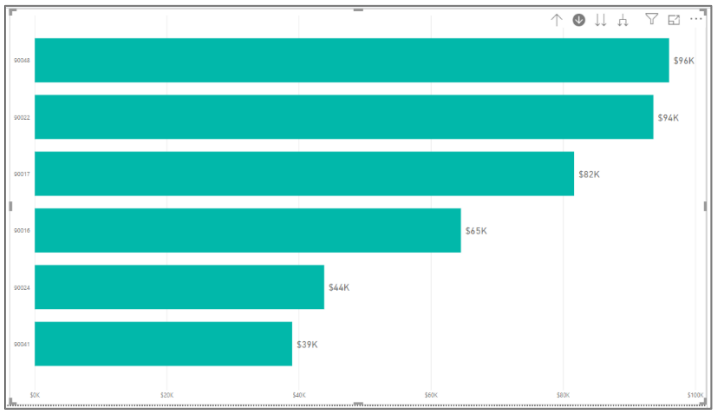
a) Click on the bar for the **Western Region** to drill down into the states for that sales region.



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

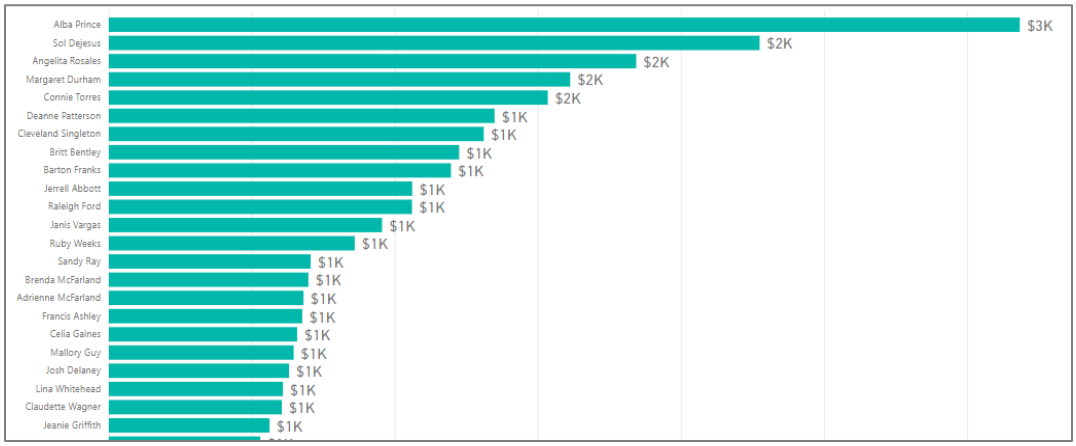


c) Click on the bar for a city such as **Los Angeles, CA** to drill down into the zipcodes for that city.



© Critical Path Training. 2020. All Rights Reserved

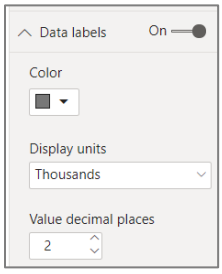
d) Click on the bar for a zipcode to drill down into the top customers in that area.



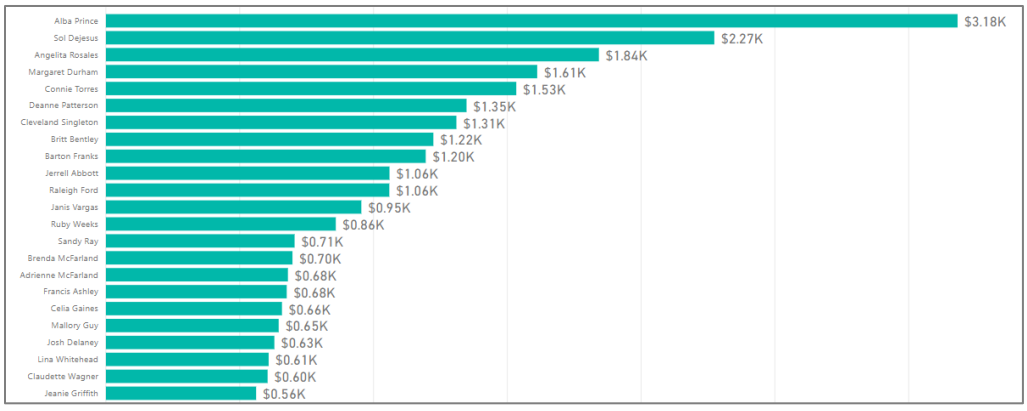
Now that you can navigate down to see sales revenue at the customer level, you need to increase the precision of the data labels.

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

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



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



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

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

© Critical Path Training. 2020. All Rights Reserved

Exercise 7: Extend the Data Model with a Custom Calendar Table

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

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

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

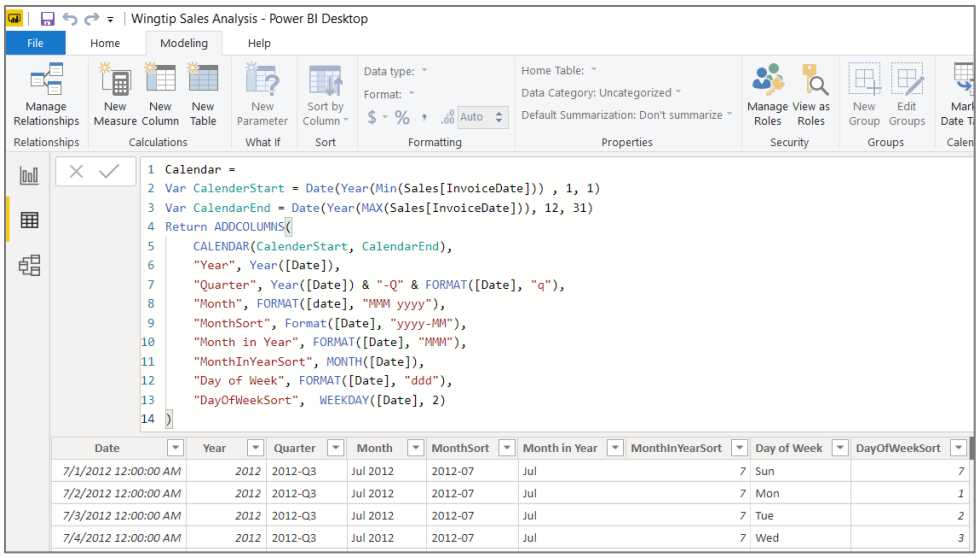
   c) Locate the file named **CreateCalendarTable.txt** at the following path and open it with Windows Notepad.

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

   d) Select the entire contents of **CreateCalendarTable.txt** and then copy it into the Windows clipboard.
   e) Return to Power BI Desktop and paste in the DAX expression for the new table.
   f) Press the **ENTER** key to create the new table named **Calendar**.

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

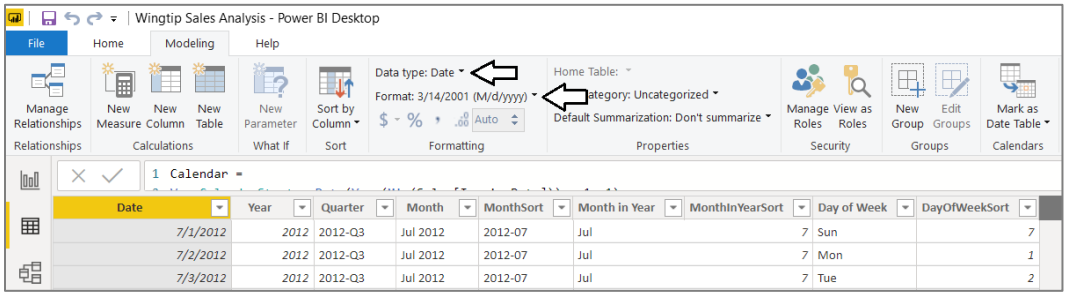
   g) You should be able to verify that the **Calendar** table has been created with several columns including a **Date** column.



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

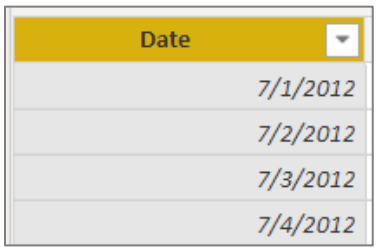
© Critical Path Training. 2020. All Rights Reserved

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

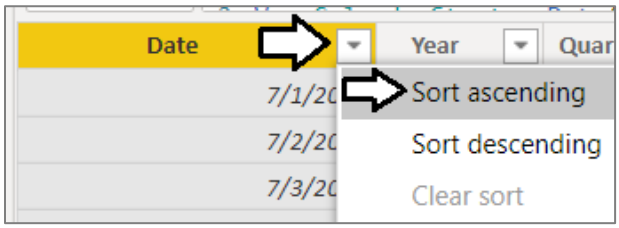


When the tables of a Power BI data model are loaded into memory, they are laid out in columns of data as opposed to rows of data. Therefore, a table in a data model table doesn't really have a pre-defined sort order. Because of this, it sometimes appears a table isn't sorted properly when you examine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

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



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



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

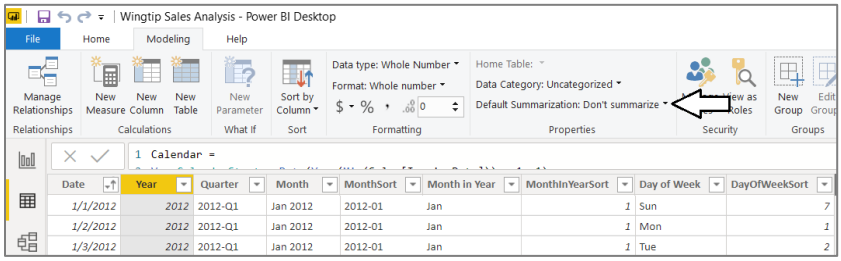
You should also be and to verify that there's one row in the **Calendar** table for each day in 2012, 2013, 2014 and 2015.

© Critical Path Training. 2020. All Rights Reserved

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

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

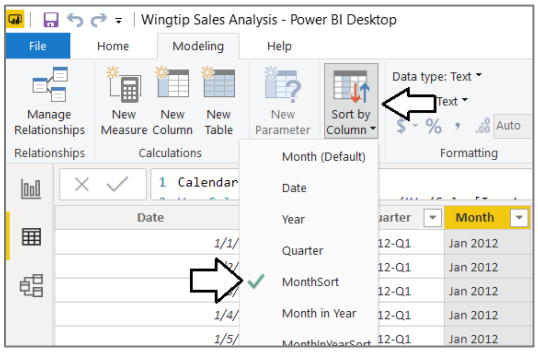
a) Select the **Year** column and sets its **Default Summarization** property to **Don't Summarize**.



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

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

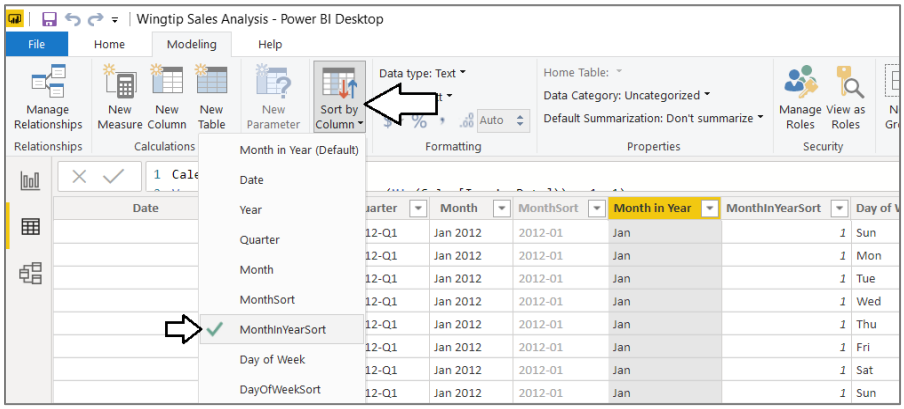
a) Select the **Month** column and then set its **Sort By Column** property to **MonthSort** column.



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

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

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

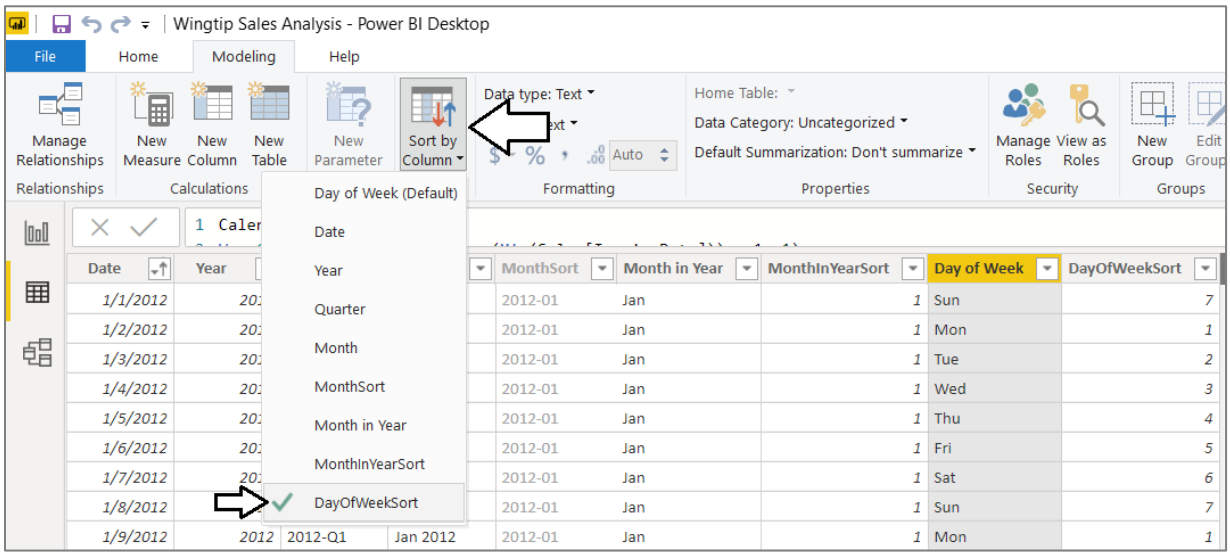


© Critical Path Training. 2020. All Rights Reserved

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

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

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



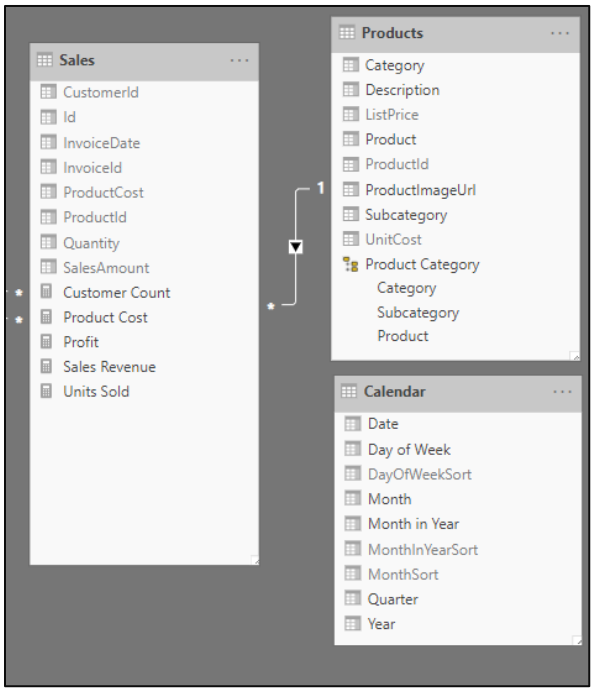
b) Hide the **DayOfWeekSort** column by right-clicking it on the **Fields** list and selecting **Hide in Report View**.

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

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

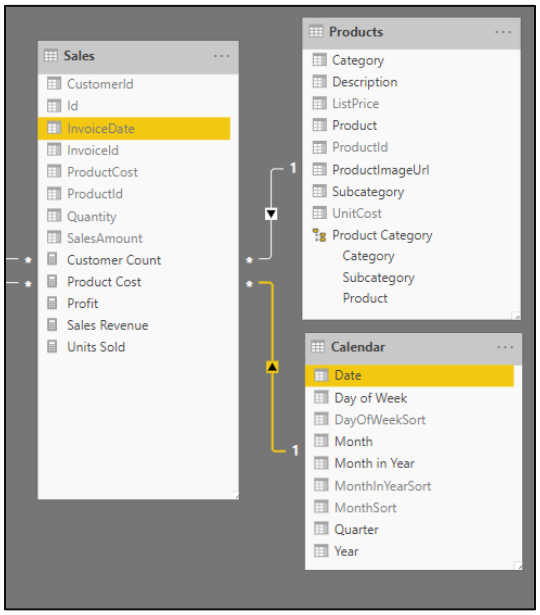
a) Navigate to **Model** view. You should be able to see that the **Calendar** table is present.

b) Using the mouse, rearrange the tables in **Model** view to match the following screenshot where the **Calendar** table is positioned directly below the **Products** table and to the immediate right of the **Sales** table.



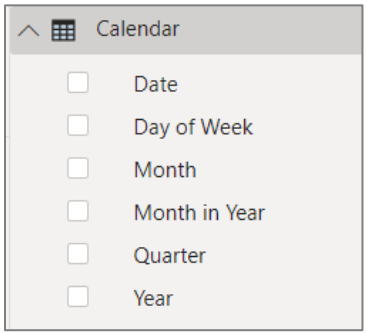
© Critical Path Training. 2020. All Rights Reserved

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

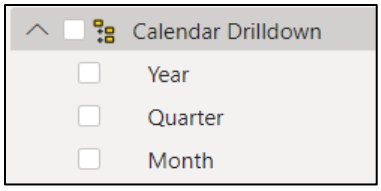


You have now completed the work of adding the **Calendar** table to the data model. Now, you will modify the **Calendar** table to add a new dimension hierarchy named **Calendar Drilldown**.

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



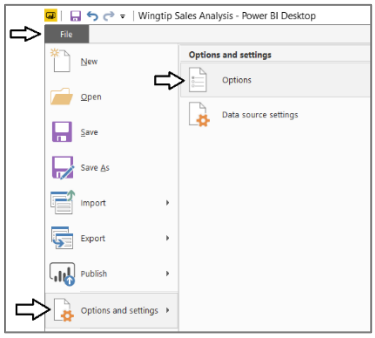
   c) Right-click on the **Year** field and then select the **New Hierarchy** menu command.
   d) Right-click **Year Hierarchy,** select the **Rename** command and rename the hierarchy to **Calendar Drilldown**.
   e) Using the mouse, drag-and-drop the **Quarter** column onto **Calendar Drilldown** to add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



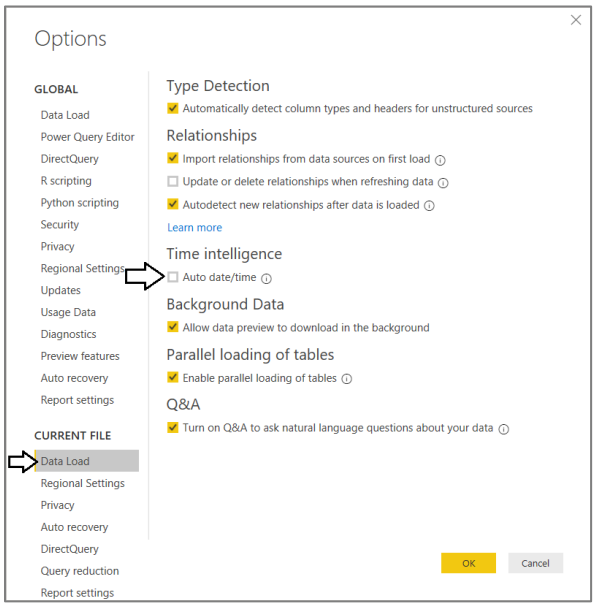
© Critical Path Training. 2020. All Rights Reserved

Now you have finished creating the **Calendar** table, you can now use it to create reports and call time intelligence functions. However, before beginning to use the calendar table, you will disable the Power BI Desktop feature for creating date hierarchies automatically.

10. 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 the **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.

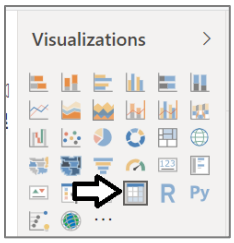
11. Save the work you have done by clicking the **Save** button in the upper left corner of the Power BI Desktop window.

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

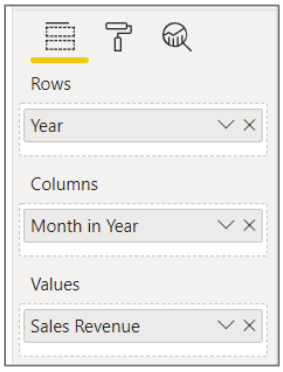
12. Create a new report page named **Sales over Time**.
    a) Navigate to **Report** view.
    b) Add a new page by clicking the (**+**) button on the right side of the page navigation menu.
    c) Once the new page has been created, modify its title to **Sales over Time**.

© Critical Path Training. 2020. All Rights Reserved

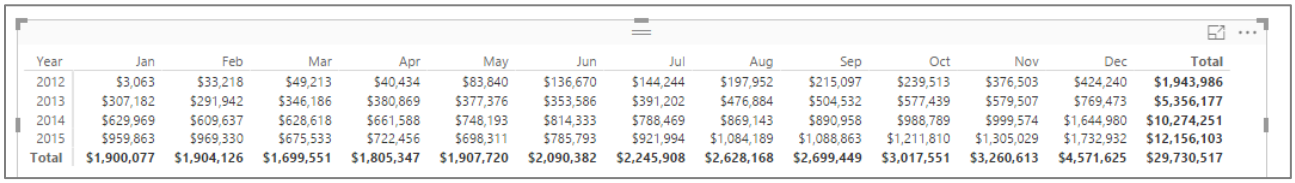
13. Create a new matrix visual to show sales revenue for specific time periods.
    a) Click that **Matrix** icon in the **Visualizations** list to add a new matrix visual to the page.



    b) With the matrix visual selected, navigate to the **Fields** pane.
    c) Add the **Year** column from the **Calendar** table into the **Rows** well.
    d) Add the **Month in Year** column from the **Calendar** table into the **Columns** well.
    e) Add the **Sales Revenue** measure from the **Sales** table into the **Values** well.

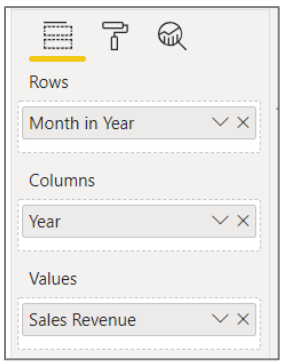


    f) The matrix now has a column for each month and a row for each year.
    g) Use your mouse to resize the matrix visual so you can see all the columns.



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 **Year** column and the **Month in Year** column are switching places.

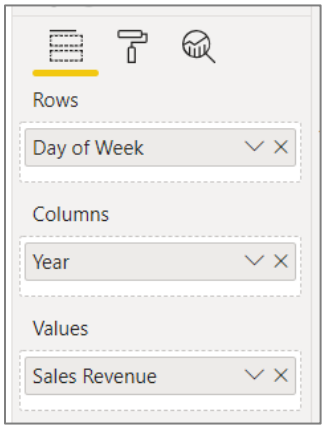


© Critical Path Training. 2020. All Rights Reserved

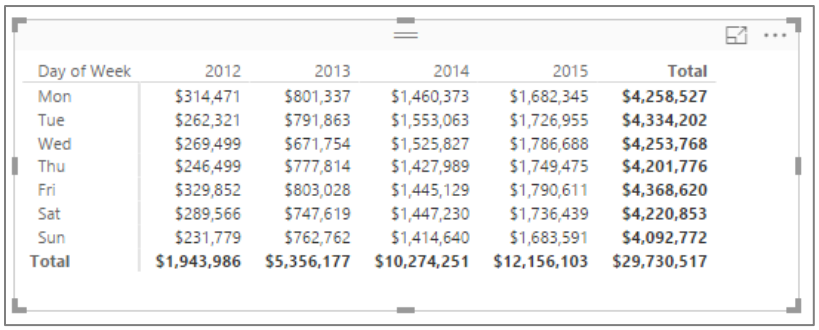
i) The matrix should now display a row for each month and a column for each year.

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**

14. Create a new matrix visual to show sales revenue for specific time periods.
    a) Add a second matrix visual to the page.
    b) Add the **Day of Week** column from the **Calendar** table into the **Rows** well.
    c) Add the **Year** column from the **Calendar** table into the **Columns** well.
    d) Add the **Sales Revenue** measure from the **Sales** table into the **Values** well.



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



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

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

© Critical Path Training. 2020. All Rights Reserved

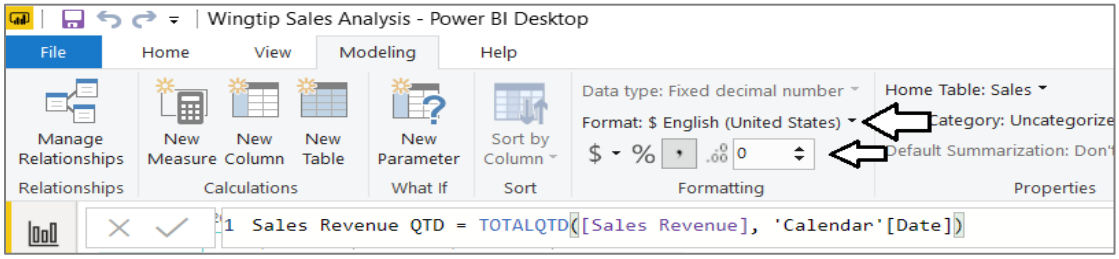
Exercise 8: Create Measures using DAX Time Intelligence Functions

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

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

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

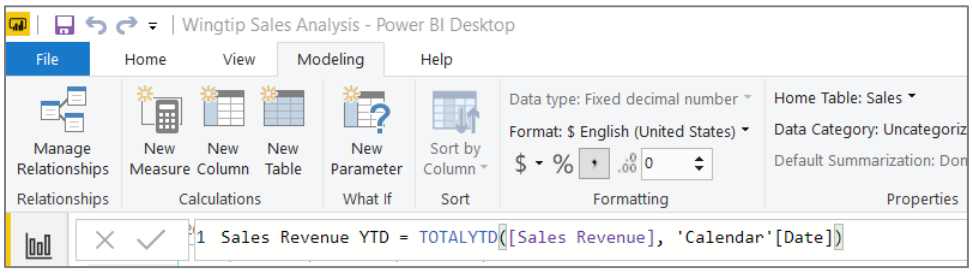
   e) Press the **ENTER** key to add the measure to data model.
   f) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Currency > English (United States)**.
   g) Use the spinner control below the **Format** menu to set the number of decimal places shown to zero.



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

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

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



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

© Critical Path Training. 2020. All Rights Reserved

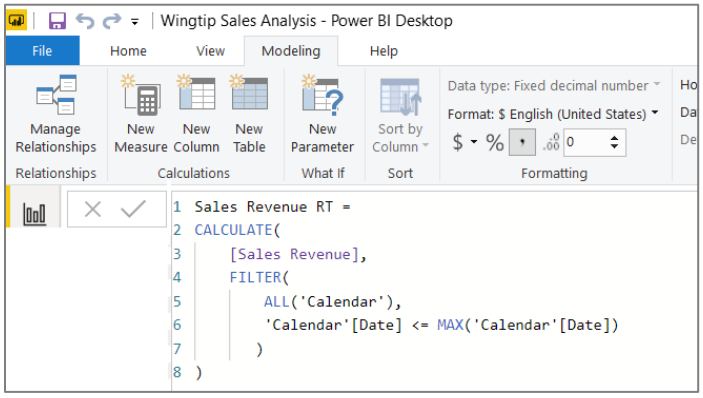
d) Enter to following DAX expression into the formula bar to create the new measure named **Sales Revenue RT**.

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

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

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

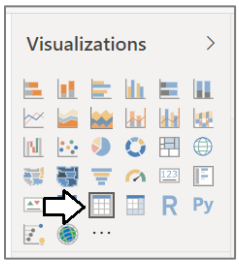
g) Use the spinner control below the format menu to set the number of decimal places shown to zero.



4. Create a new table visual to display data using your new measures.

a) Navigate to **Report** view and make sure the **Sales over Time** page is active.

b) Add a new table visual on the right-hand side of the page

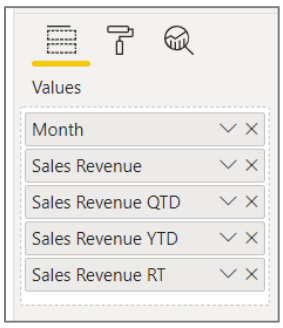


.

c) With the table visual selected, navigate to **Fields** pane

d) Add the **Month** column from **Calendar** table into the **Values** well.

e) Add the measures **Sales Revenue**, **Sales Revenue QTD**, **Sales Revenue YTD**, **Sales Revenue RT** into the **Values** well.



© Critical Path Training. 2020. All Rights Reserved

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

Month	Sales Revenue	Sales Revenue QTD	Sales Revenue YTD	Sales Revenue RT
Jan 2012	$3,063	$3,063	$3,063	$3,063
Feb 2012	$33,218	$36,280	$36,280	$36,280
Mar 2012	$49,213	$85,494	$85,494	$85,494
Apr 2012	$40,434	$40,434	$125,927	$125,927
May 2012	$83,840	$124,274	$209,768	$209,768
Jun 2012	$136,670	$260,944	$346,438	$346,438
Jul 2012	$144,244	$144,244	$490,681	$490,681
Aug 2012	$197,952	$342,196	$688,634	$688,634
Sep 2012	$215,097	$557,293	$903,731	$903,731
Oct 2012	$239,513	$239,513	$1,143,243	$1,143,243
Nov 2012	$376,503	$616,016	$1,519,746	$1,519,746
Dec 2012	$424,240	$1,040,256	$1,943,986	$1,943,986
Jan 2013	$307,182	$307,182	$307,182	$2,251,169
Feb 2013	$291,942	$599,124	$599,124	$2,543,110
Mar 2013	$346,186	$945,310	$945,310	$2,889,296
Apr 2013	$380,869	$380,869	$1,326,179	$3,270,165

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

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

5. Create a measure named **Sales Growth PM** that calculates the percentage increase between sales revenue for the current month and sales revenue for the previous month.

a) Navigate to **Data** view.

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

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

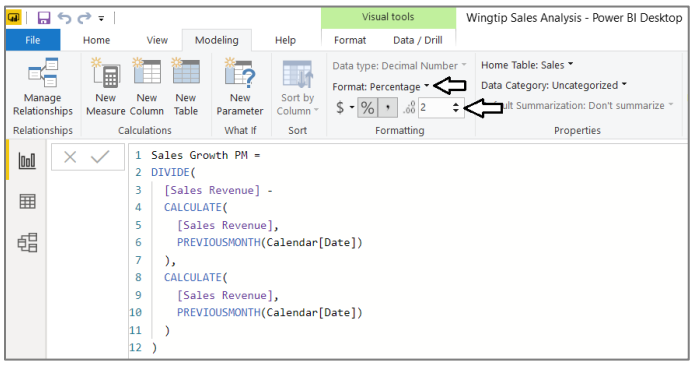
d) Enter to following DAX expression into the formula bar to create the measure named **Sales Growth PM**.

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

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

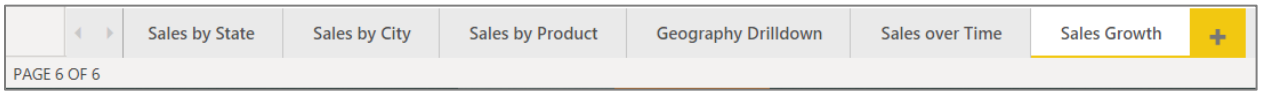
f) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Percentage**.

g) Use the spinner control below the **Format** menu to set the number of decimal places shown to **2**.

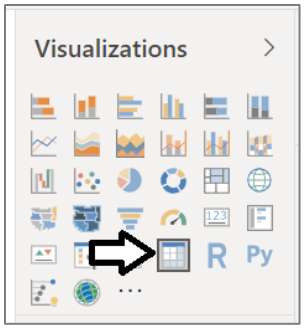


© Critical Path Training. 2020. All Rights Reserved

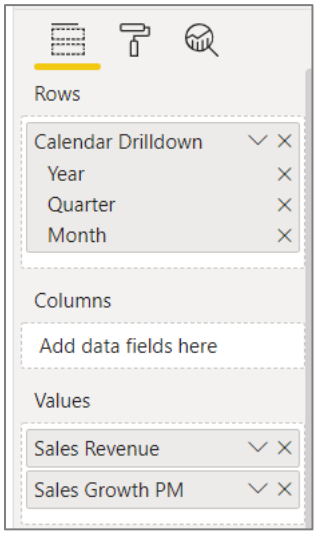
6. Create a new page in the project's report.
   a) Navigate to **Report** view.
   b) Add a new page by clicking the (**+**) button on the right of the page navigation menu.
   c) Once the new page has been created, modify its title to **Sales Growth**.



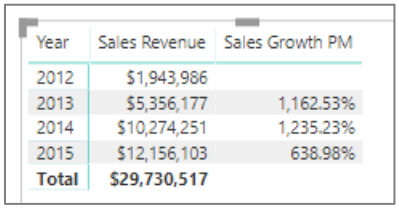
7. Create a new matrix visual to show month-to-month sales revenue growth in 2014 and 2015.
   a) Click the matrix icon in the **Visualizations** list to add a new matrix visual to the page.



   b) Drag and drop the **Calendar Drilldown** hierarchy from the **Calendar** table into the **Rows** well.
   c) Drag and drop the **Sales Revenue** measure from the **Sales** table into the **Values** well.
   d) Drag and drop the **Sales Growth PM** measure from the **Sales** table into the **Values** well.



   e) You should now have a matrix displaying one row per year like the visual shown in the following screenshot.

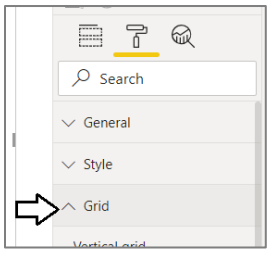


Year	Sales Revenue	Sales Growth PM
2012	$1,943,986	
2013	$5,356,177	1,162.53%
2014	$10,274,251	1,235.23%
2015	$12,156,103	638.98%
Total	**$29,730,517**	

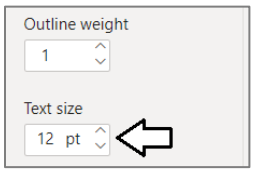
© Critical Path Training. 2020. All Rights Reserved

8. Configure the matrix visual to behave like an Excel pivot table.

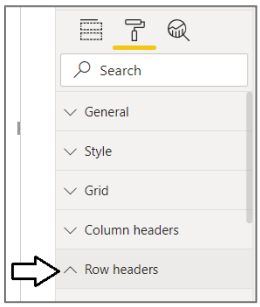
a) With the matrix visual selected, expand the **Grid** section in the **Format** pane.



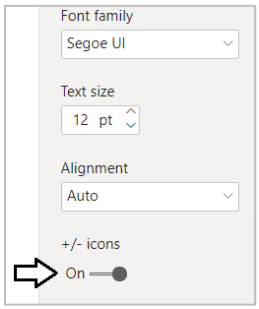
b) Inside the **Grid** section, locate the **Text** site property and increase its value to **12 pt**.



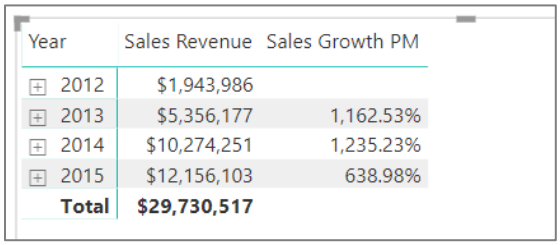
c) With the matrix still selected, expand the **Row headers** section.



d) Inside the **Row headers** section, enable the **+/- icons** property by setting it to **On**.



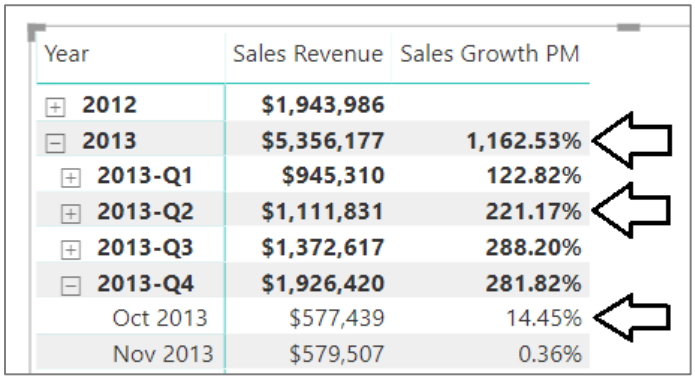
e) The matrix rows should now display an expand/collapse icons at the left



Year	Sales Revenue	Sales Growth PM
2012	$1,943,986	
2013	$5,356,177	1,162.53%
2014	$10,274,251	1,235.23%
2015	$12,156,103	638.98%
Total	**$29,730,517**	

© Critical Path Training. 2020. All Rights Reserved

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



Year	Sales Revenue	Sales Growth PM
⊞ 2012	$1,943,986	
⊟ 2013	$5,356,177	1,162.53%
⊞ 2013-Q1	$945,310	122.82%
⊞ 2013-Q2	$1,111,831	221.17%
⊞ 2013-Q3	$1,372,617	288.20%
⊟ 2013-Q4	$1,926,420	281.82%
Oct 2013	$577,439	14.45%
Nov 2013	$579,507	0.36%

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

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

a) Navigate to **Data** view.

b) Select the **Sales Growth PM** measure of the **Sales** table from the **Fields** list. When you select the **Sales Growth PM** measure in the **Fields** list, you should then be able to see and modify its DAX expression in the formula bar.

c) Before you can modify the DAX expression for the **Sales Growth PM** measure, you must be able to use the ISFILTERED function provided by DAX. You can write the following DAX expression to determine whether the current evaluation context is filtering at the month level.

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

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

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

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

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

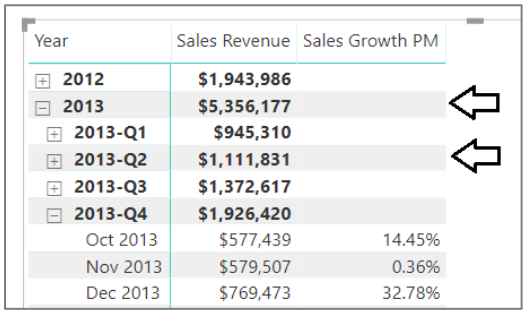
f) Update the DAX expression for the **Sales Growth PM** measure to match the following code listing.

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

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

© Critical Path Training. 2020. All Rights Reserved

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



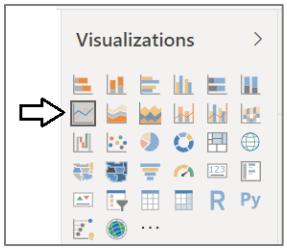
Year	Sales Revenue	Sales Growth PM
2012	$1,943,986	
2013	$5,356,177	
2013-Q1	$945,310	
2013-Q2	$1,111,831	
2013-Q3	$1,372,617	
2013-Q4	$1,926,420	
Oct 2013	$577,439	14.45%
Nov 2013	$579,507	0.36%
Dec 2013	$769,473	32.78%

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

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

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

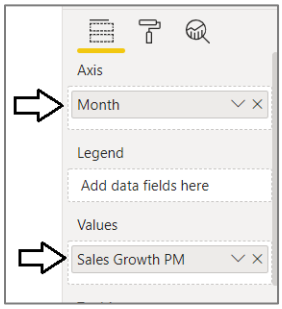
b) Click on the **Line chart** button in the **Visualizations** list to create a new Line chart visual.



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

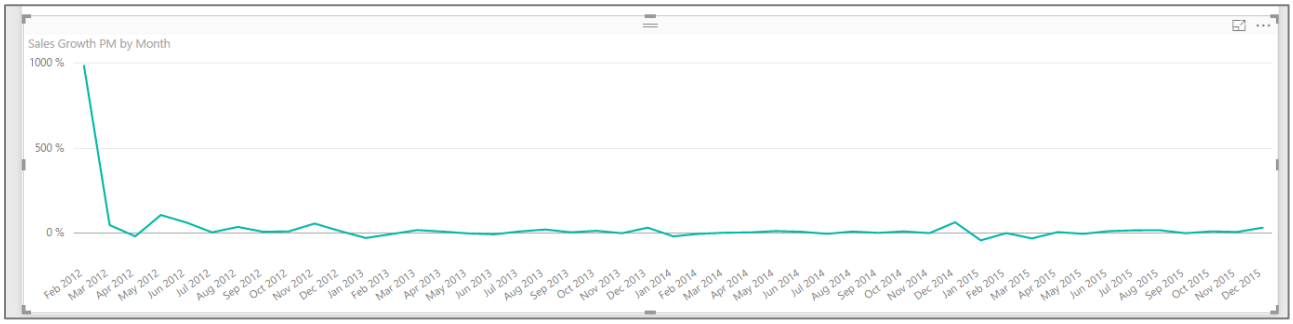
d) Drag and drop the **Month** field from the **Calendar** table into the **Axis** well.

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



© Critical Path Training. 2020. All Rights Reserved

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



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

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

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

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

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

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

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

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

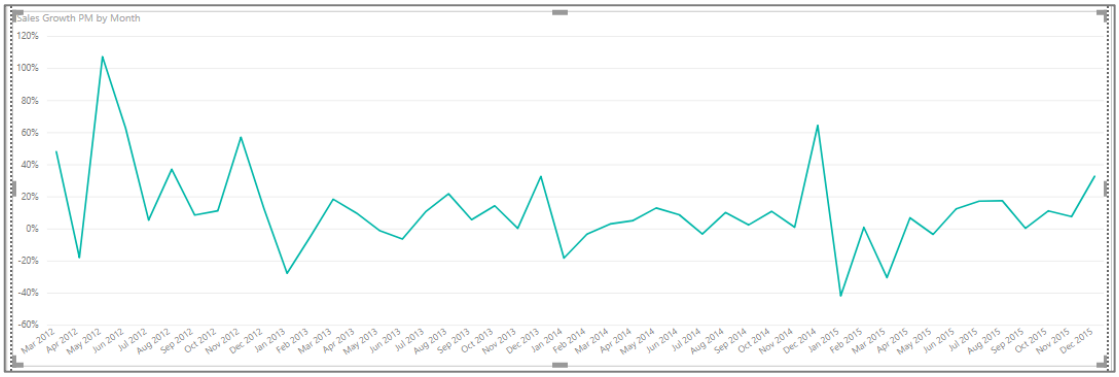
```
(
  ISFILTERED(Calendar[Month]) &&
  NOT(ISFILTERED(Calendar[Date])) &&
  [Previous Month Is Valid]
)
```

© Critical Path Training. 2020. All Rights Reserved

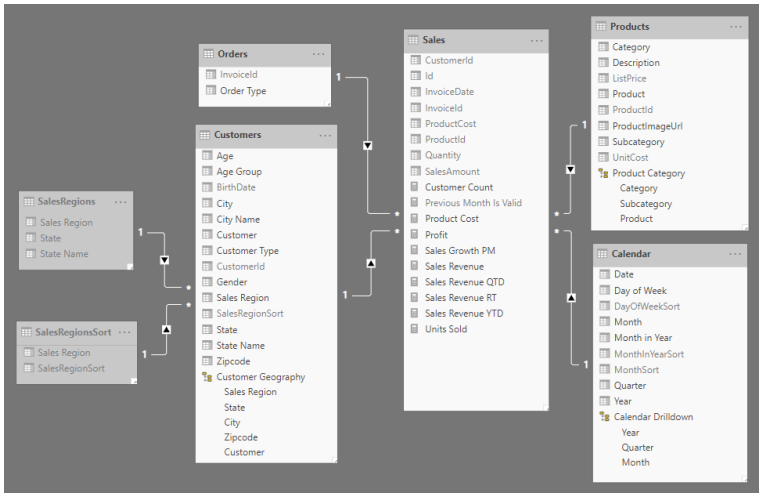
d) The DAX expression for the **Sales Growth PM** measure should now match the following code listing.

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

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



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



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

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

© Critical Path Training. 2020. All Rights Reserved