## Using PowerPivot in Excel 2013

**Lab Time**: 60 minutes

**Lab Folder**: C:\Student\Modules\ExcelPowerPivot\Lab

**Lab Overview**: In this module you will continue to work with Excel 2013 and PowerPivot along with the Excel workbook and data model that you created in the previous lab. You will enhance the data model containing Wingtip sales data by adding calculated columns, calculated fields and two dimension hierarchies. Along the way, you will create several PivotTables which will leverage the enhancements you make to the data model.

### Exercise 1: Formatting PowerPivot Data in Excel

In this exercise you will format the PowerPivot data that was imported into the workbook in the previous module.

#### Log in to Student Environment

1. Login to the Student VM using the login **WINGTIP\Administrator** and the appropriate password.
   1. If you’re using a local VM provided by the hosting training company, the password will be **Password1**.
   2. If your student VM is hosted by CloudShare, the password for the **WINGTIP\Administrator** account is going to be unique for each student, system-generated by CloudShare. Also note that the CloudShare VM configuration usually logs you into the VM automatically so you do not have to enter the user name and password.

#### Open Existing Workbook

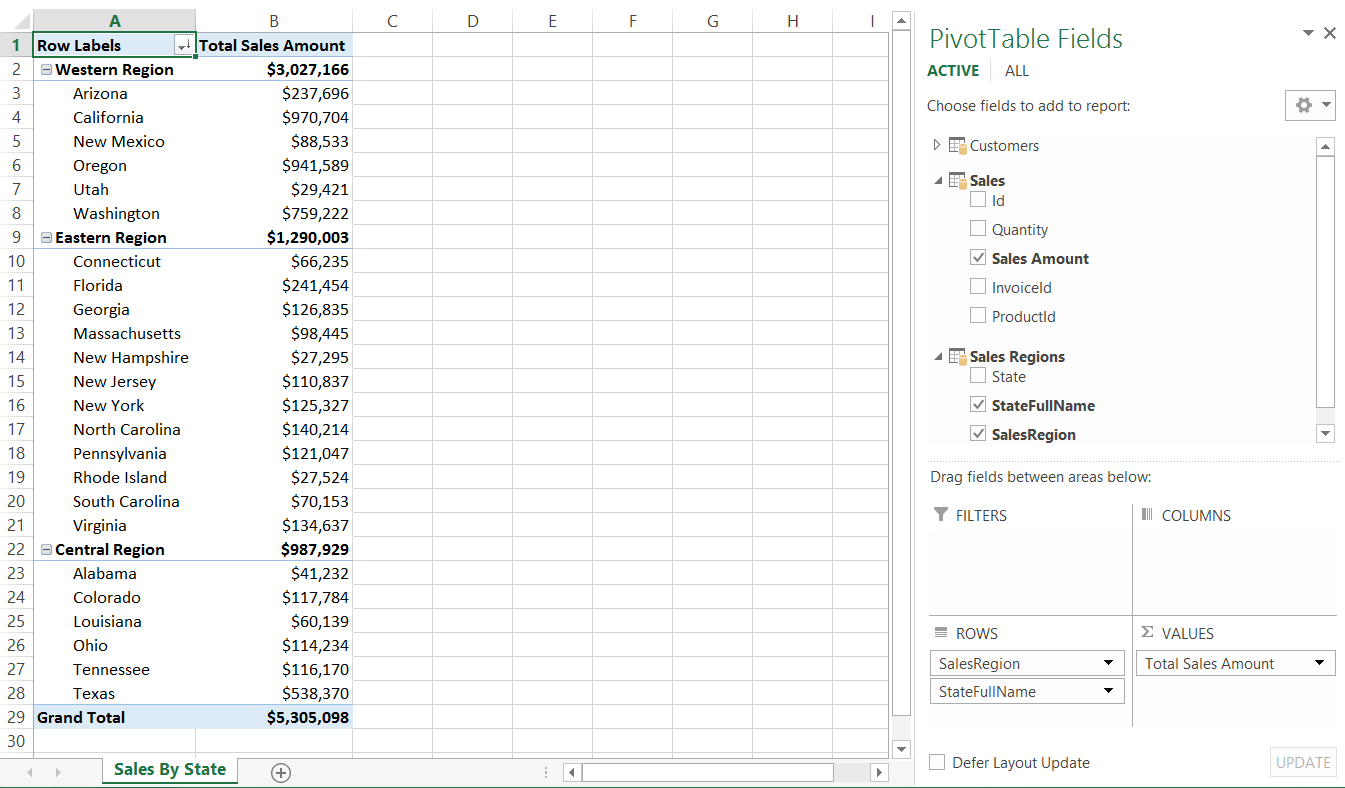
1. Open the Excel workbook titled **WingtipSalesModel.xlsx** created in Module 1.
   1. Navigate to the directory located in your student folder at **C:\Student\Models\**

This lab assumes that you have completed the first lab titled **Getting Started with PowerPivot and the Excel Data Model** in which you created an Excel workbook named **WingtipSalesModel.xlsx** and imported data into the data model using from the Wingtip Sales database. If you would like to begin work on this lab without first completing the Getting Started lab, use the Windows Explorer to copy the lab solution file at **C:\Student\Modules\GettingStarted\Lab\Solution\WingtipSalesModel.xlsx** into the folder at **C:\Student\Models**.

* 1. Double-click on the Excel workbook file named **WingtipSalesModel.xlsx** to open it in Excel 2013.
  2. Once the file is open you may get prompted with a security warning. Click on the **Enable Content** button to enable External Data Connections.

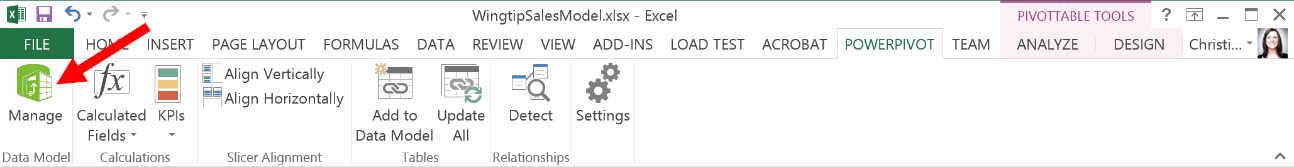


* 1. Once the file is open it should look similar to the image below.

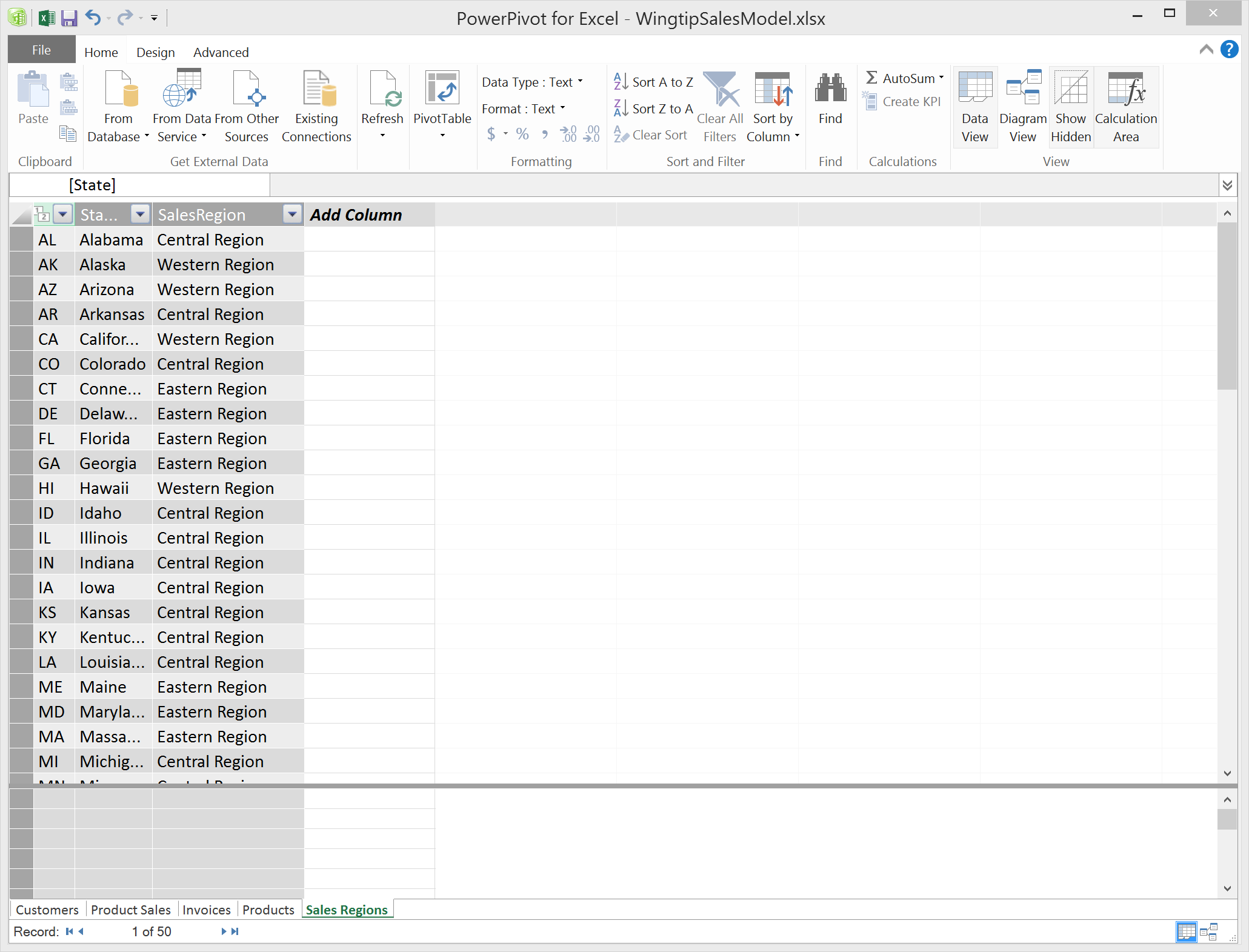


#### Format PowerPivot Data

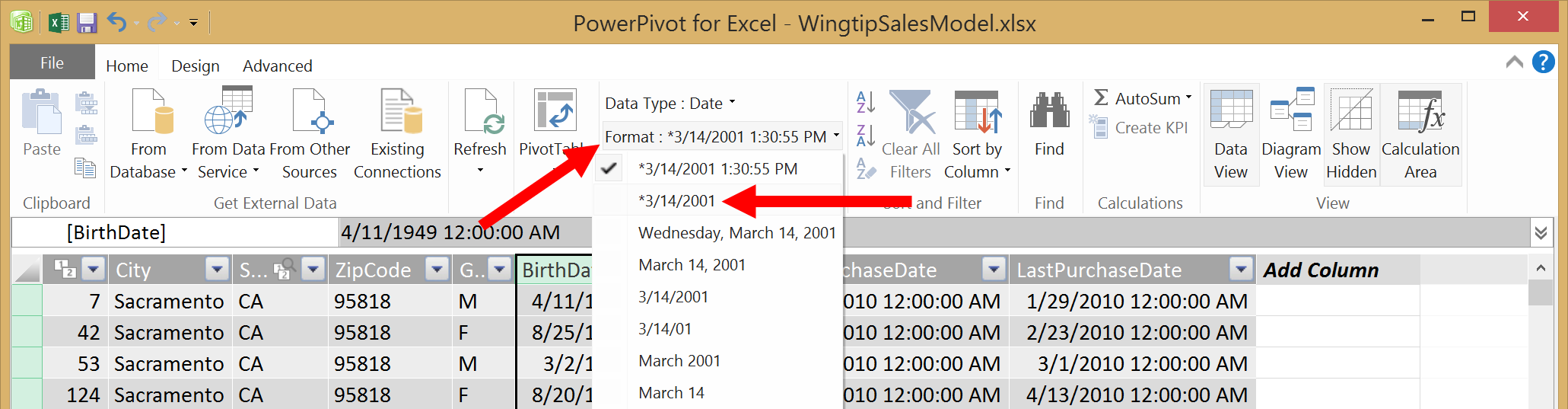
1. Open the PowerPivot window for the current Excel workbook.
   1. Navigate to the **POWERPIVOT** tab in the Excel ribbon.
   2. Click the **Manage** button in the **Data Model** ribbon group to open the PowerPivot window.



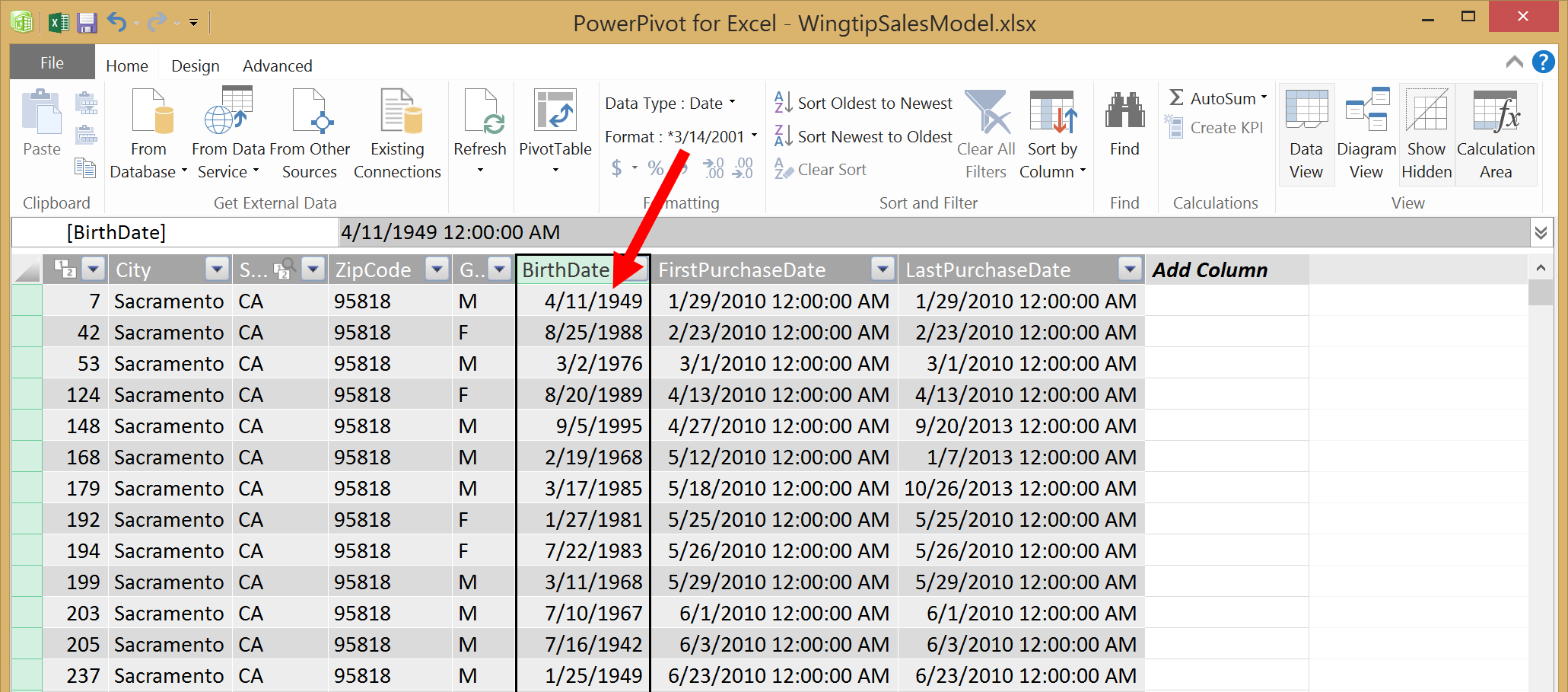
* 1. When the PowerPivot window appears you should see four data different data sheets titled **Customers**, **Sales**, **Purchases**, **Products**, and **Sales Regions**.
  2. Select the **Sales Regions** sheet if it is not already selected.
  3. Within the **Sales Regions** table, resize the column headers to show their contents by dragging cell to resize.



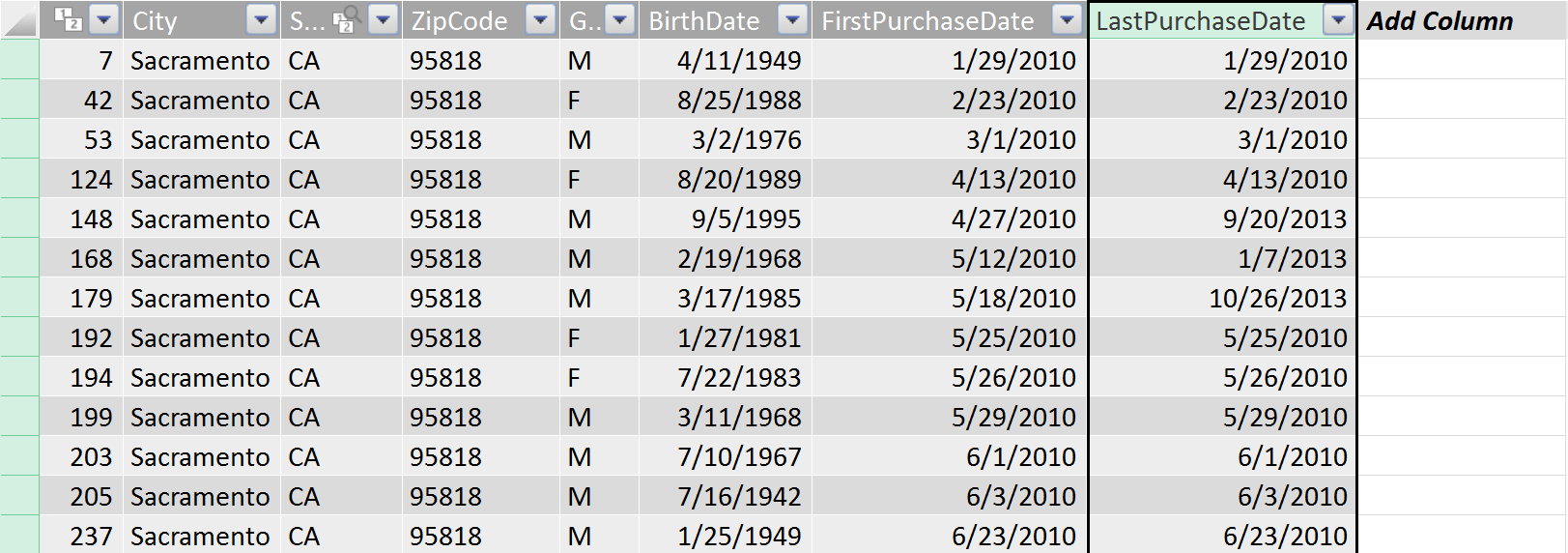
1. Modify the **Customers** table to change format of **BirthDate**, **FirstPurchaseDate**, and **LastPurchaseDate** to short date.
   1. Click on the **Customers** tab to activate the **Customers** sheet.
   2. Expand the column headers so you can see the data in the columns better by double-clicking on the end of each header column or by click-dragging your mouse to resize.
   3. Select the **BirthDate** column and then in the ribbon under the **Formatting** group, click on **Format** and then select the **\*3/14/2001** format option.



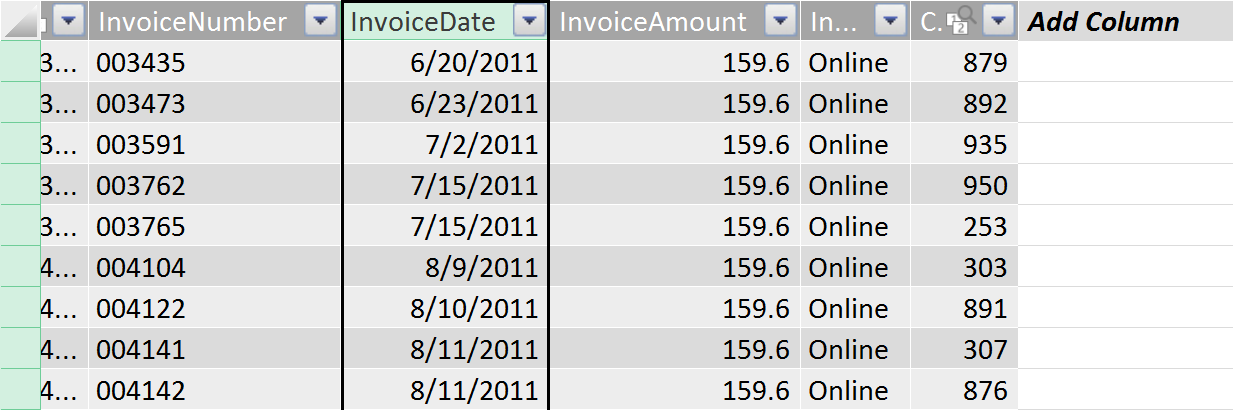
* 1. The **BirthDate** column data is now using the short date format.



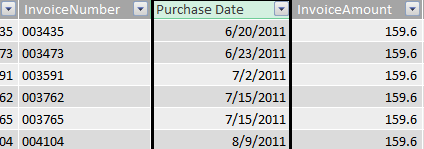
* 1. Repeat the steps for the **FirstPurchaseDate** and the **LastPurchaseDate** columns.
  2. The columns are now updated with the newly desired format.



1. Update the formatting of the columns in the **Purchases** table.
   1. Click on the **Purchases** tab to activate the Invoices sheet.
   2. Select the **InvoiceDate** column and then in the ribbon under the **Formatting** group, click on **Format** and then select the **\*3/14/2001** format option.
   3. The **InvoiceDate** column is now using the short date format.



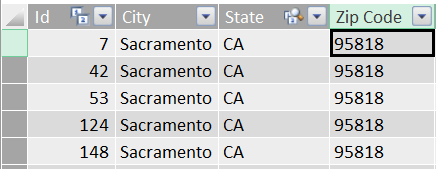
* 1. Rename the **InvoiceDate** column to **Purchase Date**.



* 1. Select the **InvoiceAmount** column and then in the ribbon under the **Formatting** group, click on **Format** and then select the **Currency** format option.
  2. Rename the **InvoiceAmount** column to **Purchase Amount**.



1. Rename the **ZipCode** column in the **Customers** table to be **Zip Code** so it is more human-readable.



1. Save your changes to the Excel workbook file **WingtipSalesModel.xslx**.

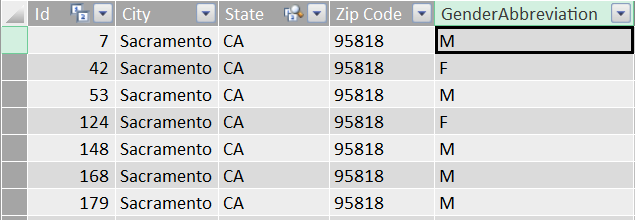
### Exercise 2: Working with Calculated Columns and DAX Expressions

In this exercise you will learn how to create calculated columns using DAX expressions.

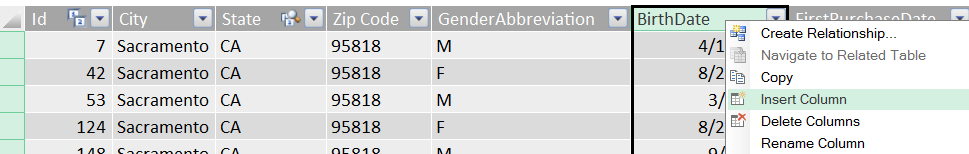
#### Add a Calculated Column to Customer table to display the Full Gender Name

Currently, the **Gender** column displays abbreviated values of "M" and "F" instead of the human-readable value of "Male" and "Female". Over the next few steps, you will create a calculated column to display customer gender values of "Male" and "Female" instead of the shorter abbreviated versions.

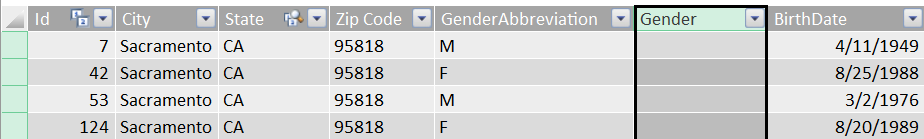
1. Create a new calculated column to display the full gender name.
   1. Navigate back to the **Customers** table by clicking on the **Customers** tab.
   2. Locate the **Gender** column and rename it to **GenderAbbreviation**.



* 1. Note that the column to the right of the **GenderAbbreviation** column is the **BirthDate** column. Right-click on the **BirthDate** column and select the **Insert Column** command to add a new column to the right of the **GenderAbbreviation** column.



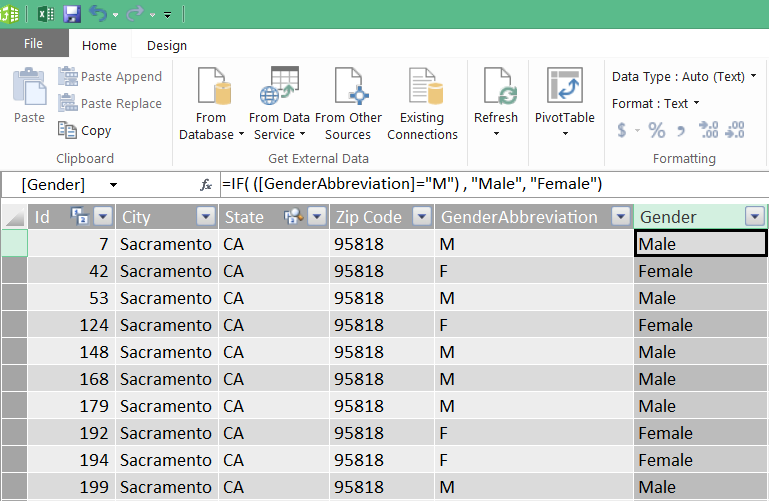
* 1. Rename the new column to **Gender**.



* 1. Enter the following DAX formula to determine the full gender name from the abbreviated gender name.

=IF( ([GenderAbbreviation]="M") , "Male", "Female")

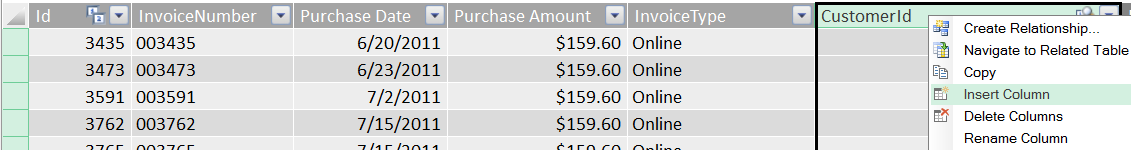
* 1. Verify that each value displayed in the **Gender** column has a value of either "Male" or "Female".



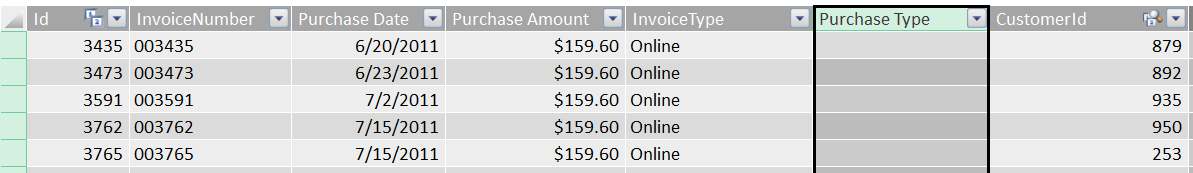
#### Add a Calculated Column to the Purchases table to display the Purchase Type

The **InvoiceType** column contains abbreviated dimension values including "Online", "InPerson" and "MailOrder". Over the next few steps, you will create a calculated column named **Purchase Type** to create more human-readable values such as "Online Purchases", "Store Location Purchases" and "Mail Order Purchases". The key point is here that you are creating more readable dimension column values that will provide better labels for the PivotTables and the reports that you create using this data model.

1. Add a calculated column for **Purchase Type**.
   1. Navigate back to the **Purchases** table by clicking on the **Purchases** tab.
   2. Note that the column to the right of the **InvoiceType** column is the **CustomerId** column. Right-click on the **CustomerId** column and select the **Insert Column** command to add a new column to the right of the **InvoiceType** column.



* 1. Rename the new column to **Purchase Type**.



* 1. Enter the following DAX formula to replace the current values with more human-readable values.

=SWITCH([InvoiceType],

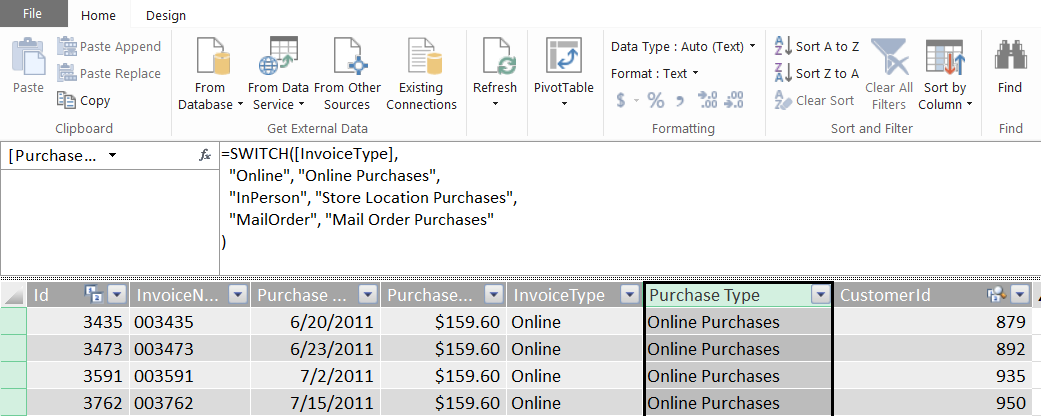
"Online", "Online Purchases",

"InPerson", "Store Location Purchases",

"MailOrder", "Mail Order Purchases"

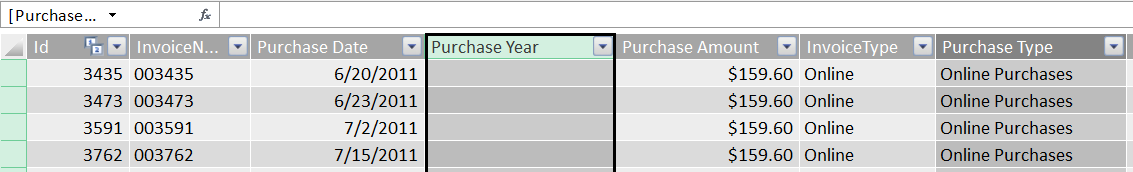
)

* 1. Verify that values of the **Purchase Type** column are displayed as "Online Purchases", "Store Location Purchases" or "Mail Order Purchases". Note you may have to scroll down to the bottom of the **Purchases** table to see rows that contain all three distinct values.



#### Add Calculated Column to the Purchase table for Purchase Year

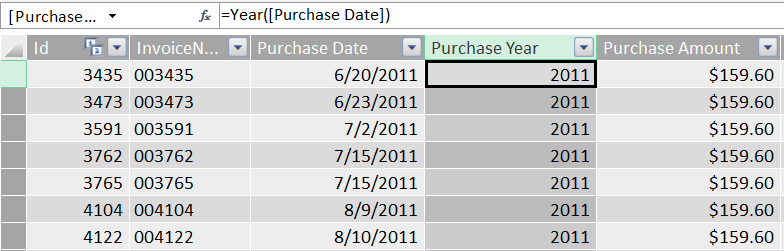
1. Add the **Purchase Year** calculated column.
   1. Navigate back to the **Purchases** table by clicking on the **Purchases** tab if you are not already there.
   2. Create a new calculated column in between the **Purchase Date** column and the **Purchase Amount** column. Give the new column a name of **Purchase Year**.



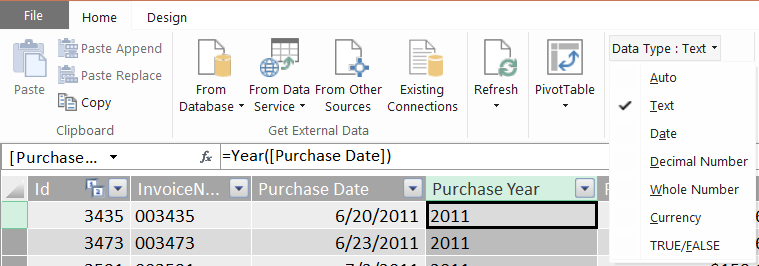
* 1. Enter the following DAX formula to calculate the year from the Purchase Date column.

=Year([Purchase Date])

* 1. Verify that a valid year is displayed for each value in the **Purchase Year** column.



* 1. Change the **Data Type** of the **Purchase Year** column from **Auto** to **Text** using the dropdown menu in the ribbon as shown in the following screenshot.

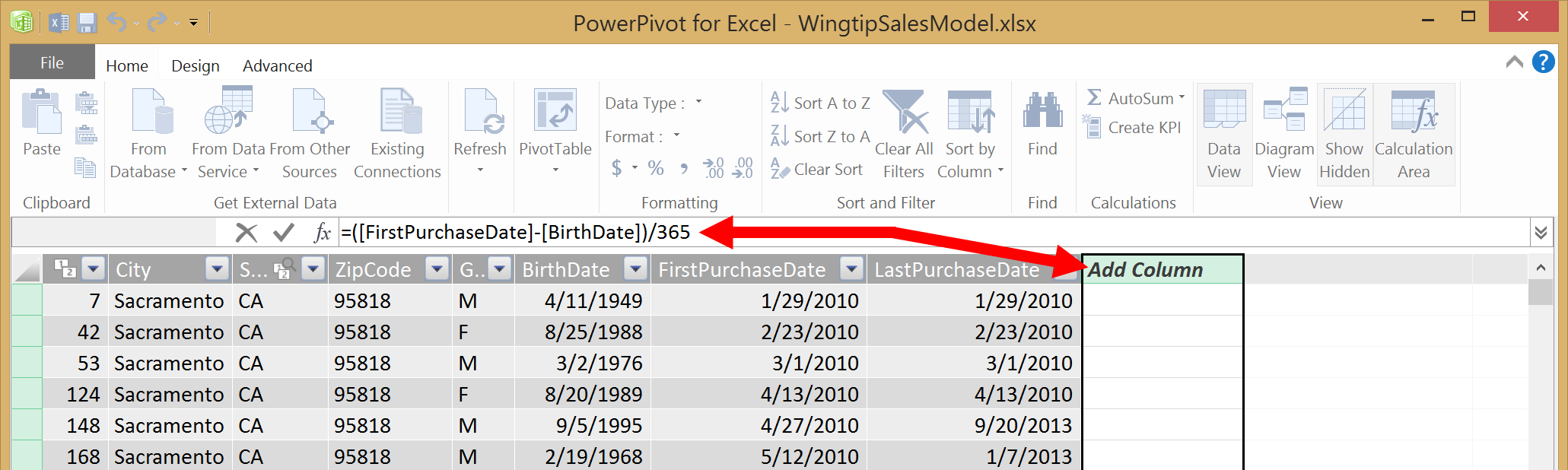


You might be wondering why it is important to modify the data type of the **Purchase Year** column to be a **Text** column. If you leave the **Data Type** setting at its default value of **Auto**, the column type is automatically set to the type **Whole Number**. This can cause confusion because the behavior in Excel when adding the column to a PivotTable assumes that the **Purchase Year** column values are numeric values that can be aggregated using operations such as SUM or AVERAGE. Configuring the **Purchase Year** column with the **Text** data type is a technique that prevents Excel from incorrectly assuming that it can aggregate values from the **Purchase Year** column.

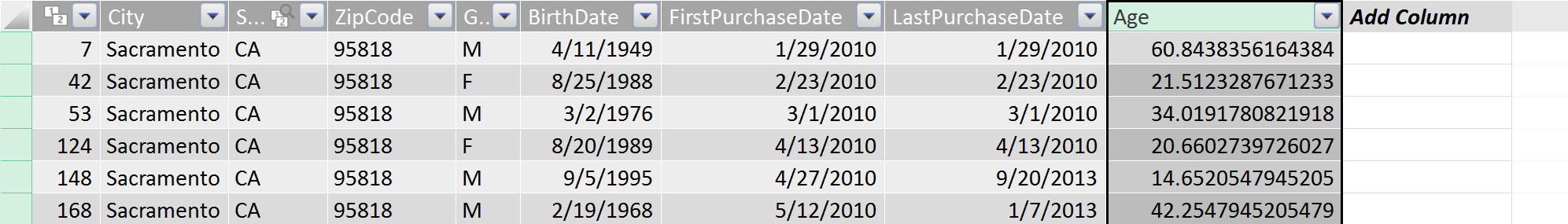
#### Add Calculated Column to Customers table for Customer Age

1. Navigate back to the **Customers** table by clicking on the **Customers** tab.
2. Create a calculated column named **Age** for customer age at first purchase.
   1. Select the column **Add Column** and then in the formula text box type the following DAX expression and then press **Enter**.

=([FirstPurchaseDate]-[BirthDate])/365



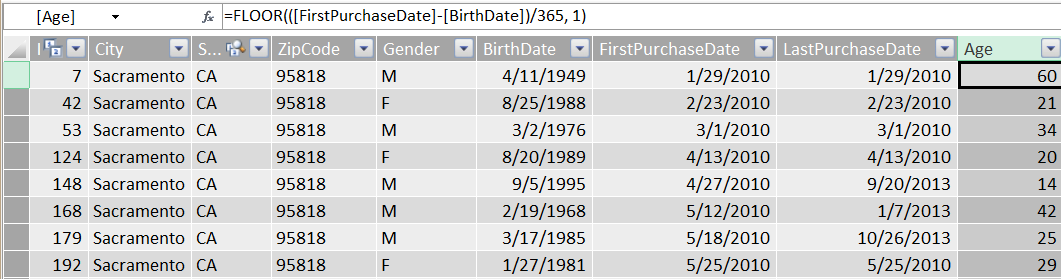
* 1. Rename the column from **CalculatedColumn1** to **Age**.



* 1. Modify the DAX formula to round the value down to the nearest whole number value using the **FLOOR** function.

=FLOOR( ([FirstPurchaseDate]-[BirthDate])/365, 1 )

* 1. The **Age** column is now display a whole number that has been rounded down for the age of each customer.



#### Add Calculated Column to Customers table for Age Demographic

1. Create a calculated column named **Age Demographic** for customer age group.
   1. In the Customers table, select the column **Add Column** and then in the formula text box type the following DAX expression and then press **Enter**.

=If([Age] >= 80, "Customers 80 and up",

If([Age] >= 65, "Customers 65 TO 80",

If([Age] >= 50, "Customers 50 TO 65",

If([Age] >= 40, "Customers 40 TO 49",

If([Age] >= 30, "Customers 30 TO 39",

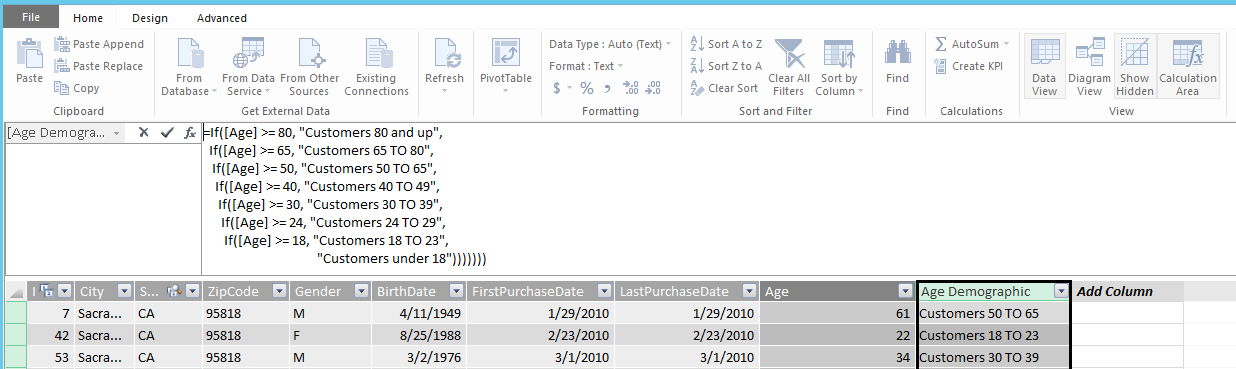
If([Age] >= 24, "Customers 24 TO 29",

If([Age] >= 18, "Customers 18 TO 23",

"Customers under 18")))))))

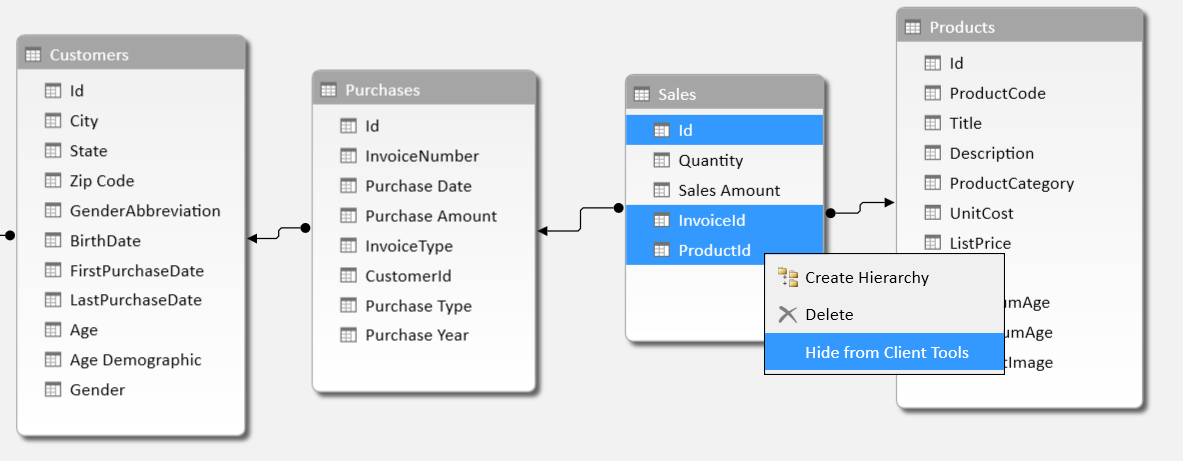
If you would rather copy-and-paste the text for this DAX expression instead of typing it into Excel yourself, you copy it from a file in your student folder at a path of **C:\Student\Modules\ExcelPowerPivot\Lab\AgeDemographicDAX.txt**.

* 1. The calculated column should now be created with a name of **CalculatedColumn1**.
  2. Rename the new calculated column from **CalculatedColumn1** to **Age Demographic**.

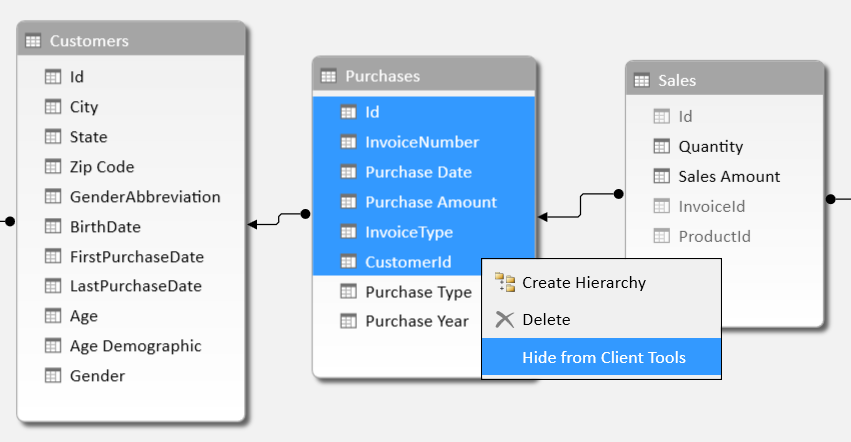


Now it's time to clean up the Data Model by hiding columns that will not be used in any PivotTables or reports.

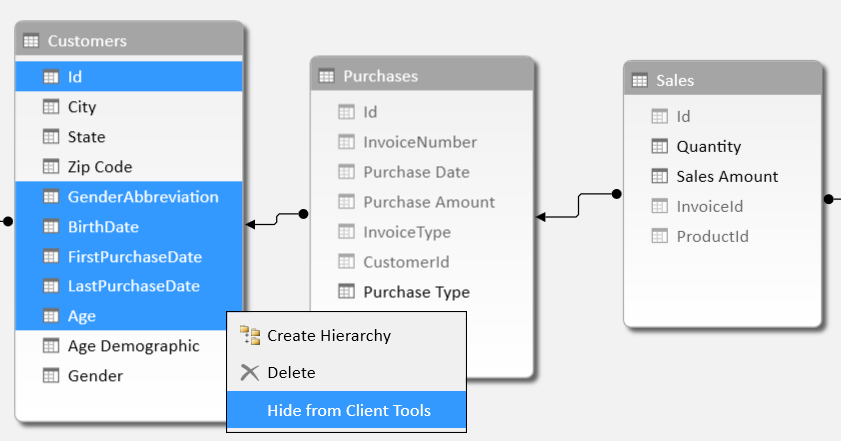
1. Clean up the Data Model by unnecessary hiding columns.
   1. Switch the PowerPivot window from **Data View** to **Diagram View** so you can see all the tables in the data model at once.
   2. Hide all the identifier columns in the **Sales** table including **Id**, **InvoiceId** and **ProductId**. The only two columns that should remain visible are the **Quantity** column and the **Sales Amount** column.



* 1. Hide all fields in the **Purchases** table except for **Purchase Type** and **Purchase Year**.



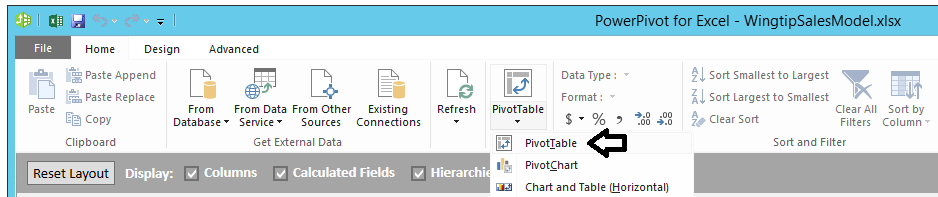
* 1. Hide all the columns in **Customers** table that will not be used to analyze sales results including the columns **Id**, **GenderAbbreviation,** **BirthDate**, **FirstPurchaseDate**, **LastPurchaseDate** and **Age**.



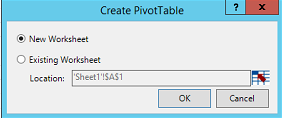
* 1. You have now completed the steps of hiding the unwanted columns from the **Sales** table, the **Purchases** table and the **Customers** table. Next you will see how these changes affects people that are using this data model to create a PivotTable.

#### Create a PivotTable which uses Calculated Columns

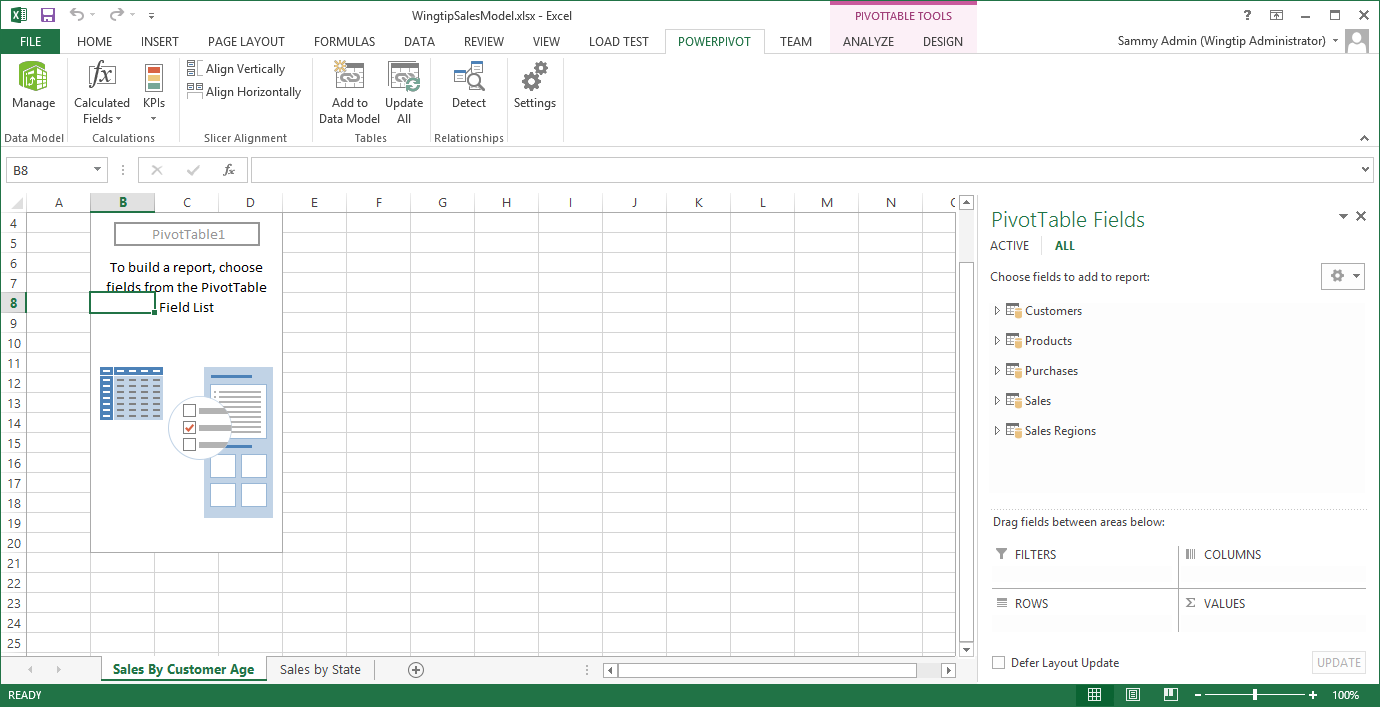
1. Create a new PivotTable that makes use of the calculated columns you just created.
   1. From the **Home** tab in the ribbon of the **POWERPIVOT** window, drop down the **PivotTable** menu control and click **PivotTable** to create a new pivot table in in the workbook **WingtipSalesModel.xslx**.



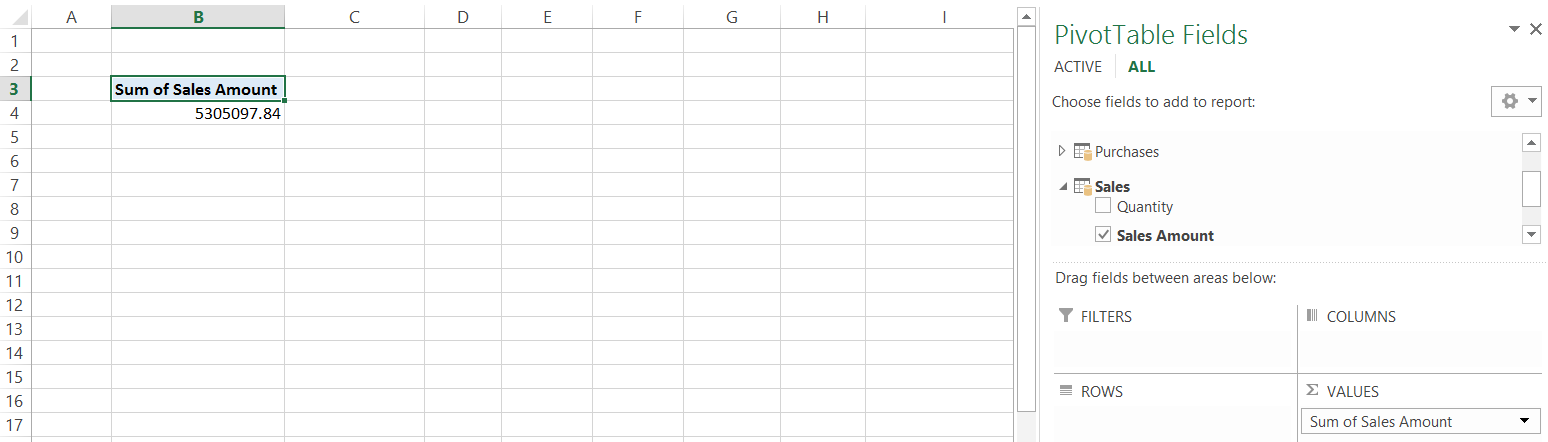
* 1. When the **Create PivotTable** dialog appears, select **New Worksheet** and click the **OK** button.



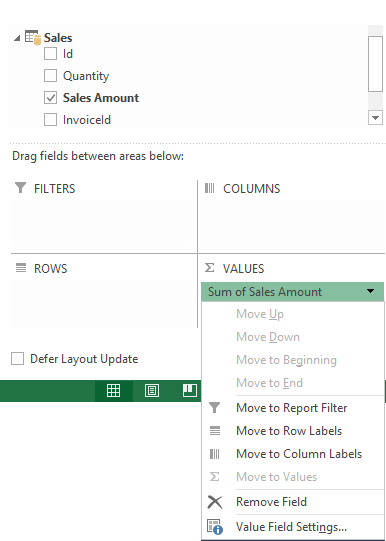
* 1. Once the new worksheet with the new PivotTable is added to the workbook, rename the worksheet **Sales by Customer Age**.



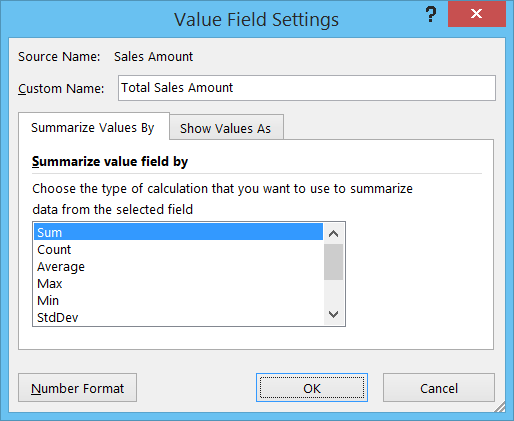
* 1. On the right-hand side of the workbook in the **PivotTable Fields** section of the task pane, expand the **Sales** table.
  2. Select the **Sales Amount** column from the **Sales** table. When you select the **Sales Amount** column, you should see the column show up in the **VALUES** section down below with a default name of **Sum of Sales Amount**.



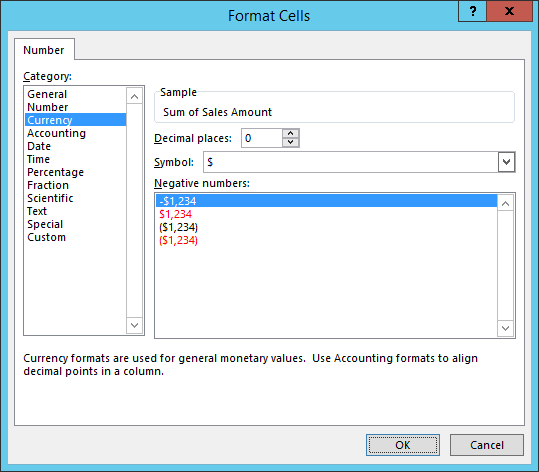
* 1. In the **VALUES** section in the bottom right of the PowerPivot window, click on **Sum of Sales Amount** to drop down the menu control and click the **Value Field Settings…** command to display the **Field Value Settings** dialog.



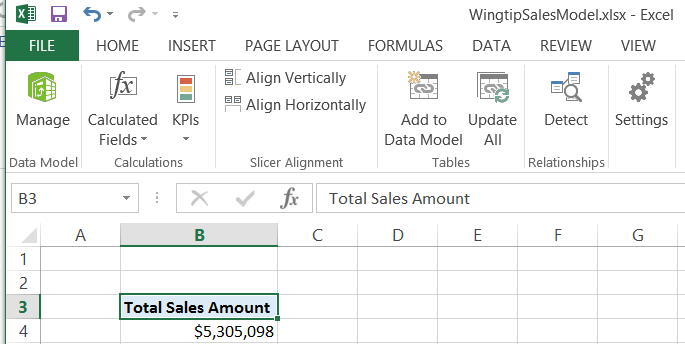
* 1. In the **Value Field Settings** dialog, change the **Custom Name** property from **Sum of Sales Amount** to **Total Sales Amount**.



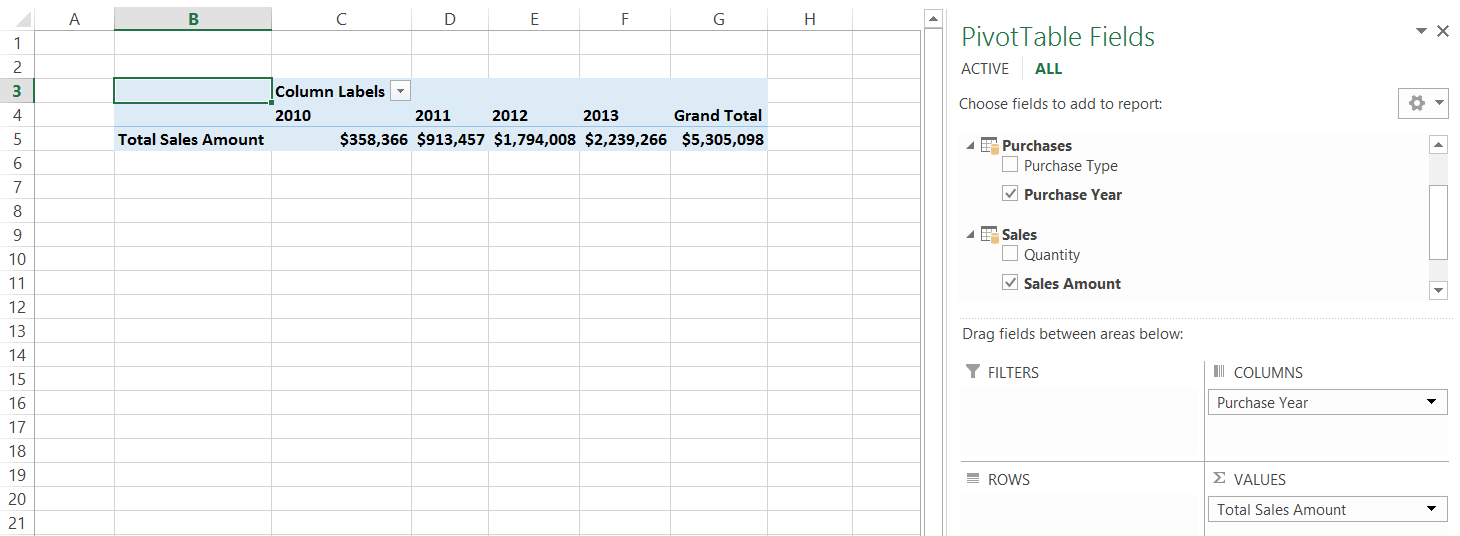
* 1. In the **Value Field Settings** dialog, click to **Number Format** button to display the **Format Cells** dialog.
  2. In the **Format Cells** dialog, set the **Category** to **Currency** and assign a value of **0** for the number of **Decimal places**.



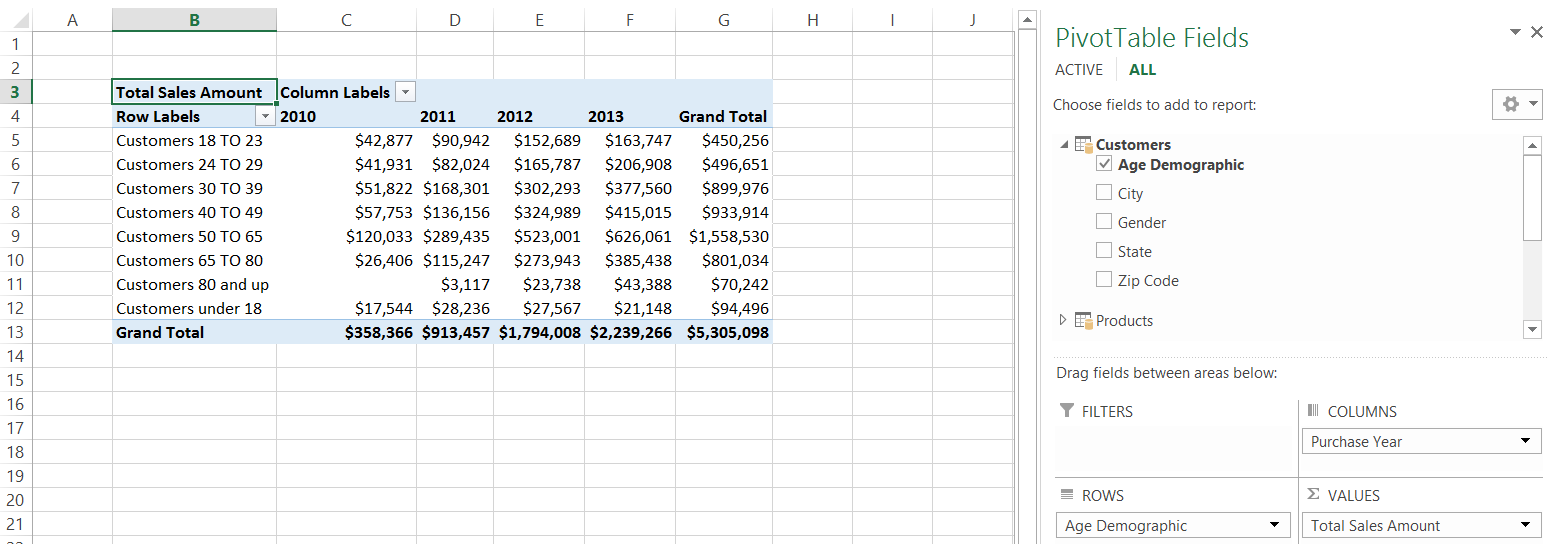
* 1. Click **OK** to close the **Format Cells** dialog.
  2. Click **OK** to close the **Value Field Settings** dialog.
  3. You should now see the result in the PivotTable has been renamed to **Sales Revenue** and is formatted as currency.



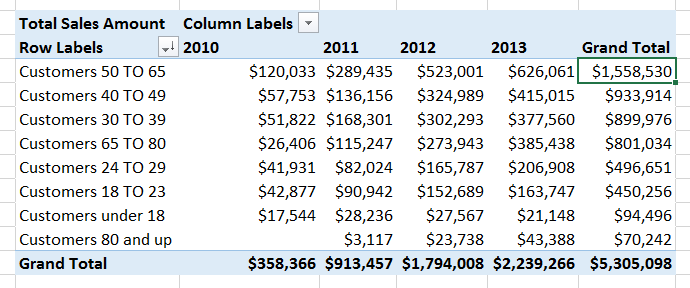
* 1. In the **PivotTable Fields** section of the task pane, add the **Purchase Year** column from the **Purchases** table into the **COLUMNS** section at the bottom of the task pane.



* 1. Add the **Age Demographic** column from the **Customers** table into the **ROWS** section at the bottom of the task pane. The PivotTable should now look like the one shown in the following screenshot.



* 1. Right-click on a cell inside the **Grand Total** column and select the menu command to sort the PivotTable so that the age groups with higher sales revenue are sorted to the top.



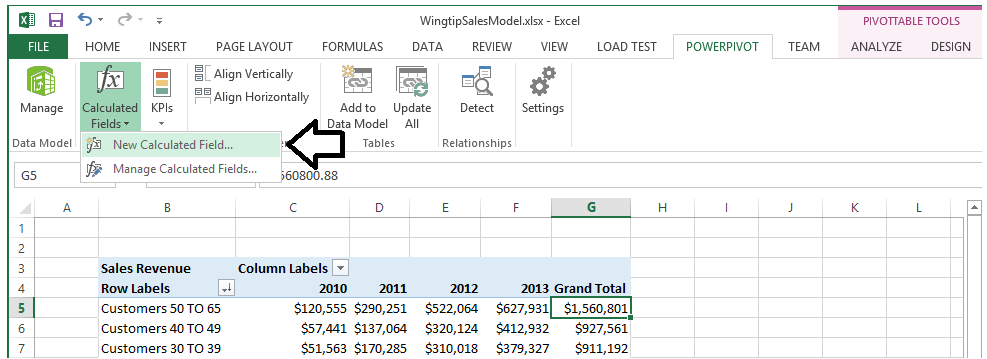
* 1. Save our changes to **WingtipSalesModel.xslx**.

### Exercise 3: Creating Calculated Fields on the Sales Table

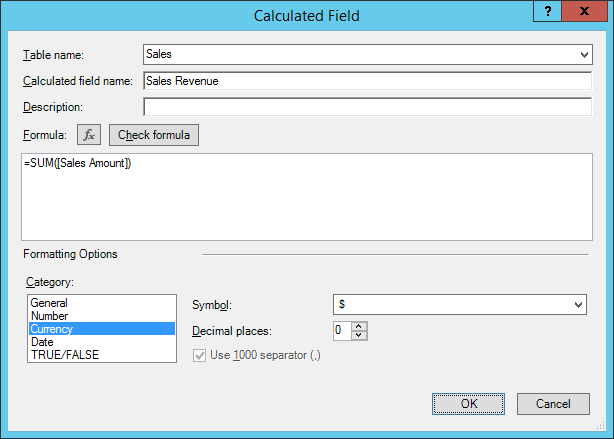
In this exercise you will create two calculated fields on the **Sales** table named **Sales Revenue** and **Units Sold**. The calculated field named **Sales Revenue** will be based on the aggregate sum of the **Sales Amount** column and the calculated field named **Units Sold** will be based will be based on the aggregate sum of the **Quantity** column.

#### Create Calculated Fields on the Sales Table to Calculate Sales Revenue and Units Sold

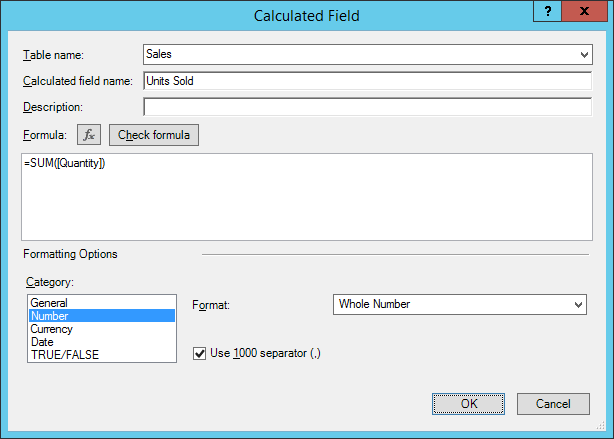
1. Create a calculated field for **Sales Revenue** on **Sales** table.
   1. Make sure you are at the Excel application windows as opposed to the PowerPivot window.
   2. Navigate to the **POWERPIVOT** tab in the ribbon.
   3. In the **Calculations** ribbon group, drop down the **Calculated Fields** menu.
   4. Click the **New Calculated Field…** menu command.



* 1. Fill in the form in the Calculated Field dialog as follows:
     1. **Table name**: Sales
     2. **Calculated field name**: Sales Revenue
     3. **Formula**: =SUM([Sales Amount])
     4. **Formatting Options Category**: Currency
     5. **Decimal places**: 0
  2. When the **Calculated Field** dialog looks like the following screenshot, click **OK** to create the new calculated field.

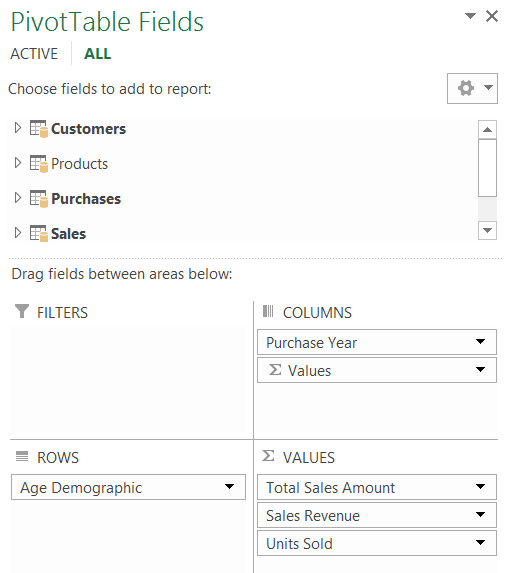


1. Create a calculated field for **Units Sold** on **Sales** table.
   1. Make sure you are at the Excel application windows as opposed to the PowerPivot window.
   2. Navigate to the **POWERPIVOT** tab in the ribbon.
   3. In the **Calculations** ribbon group, drop down the **Calculated Fields** menu.
   4. Click the **New Calculated Field…** menu command.
   5. Fill in the form in the Calculated Field dialog as follows:
      1. **Table name**: Sales
      2. **Calculated field name**: Units Sold
      3. **Formula**: =SUM([Quantity])
      4. **Formatting Options Category**: Number
      5. **Format**: Whole Number
      6. **Use 1000s separator**: checked
   6. When the **Calculated Field** dialog looks like the following screenshot, click **OK** to create the new **Units Sold** calculated field.

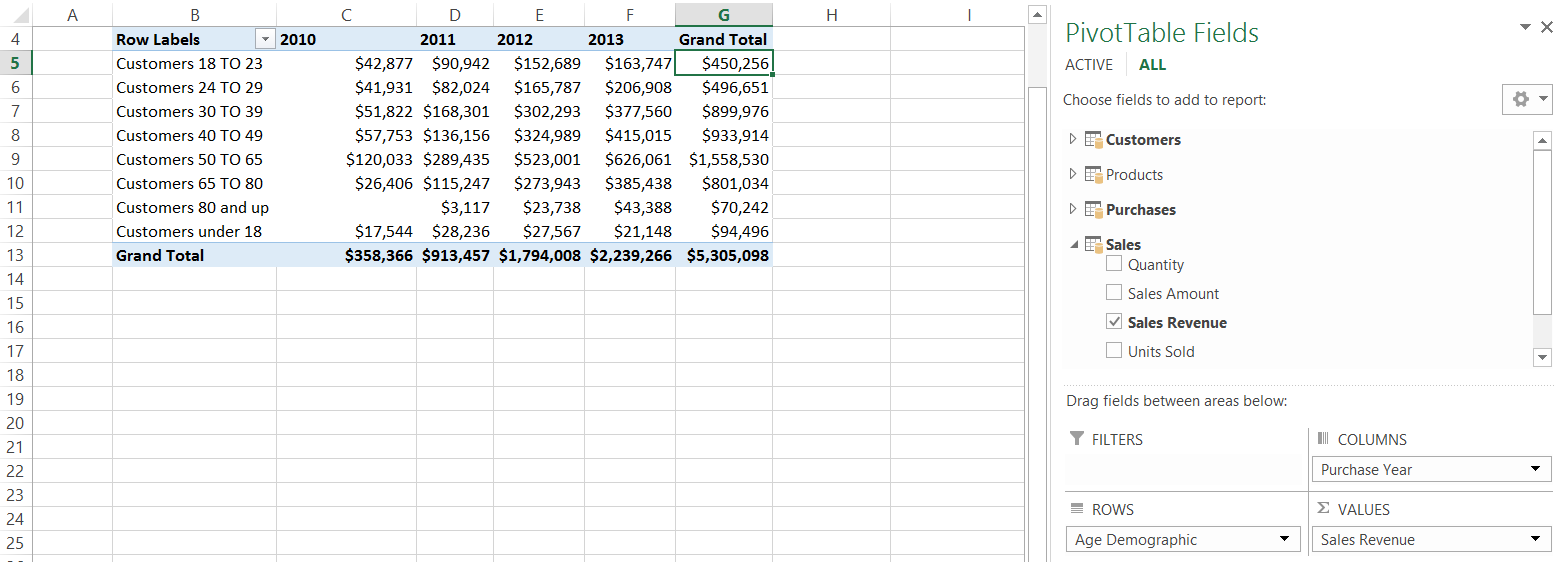


#### Update existing PivotTables to use the Sales Revenue calculated field

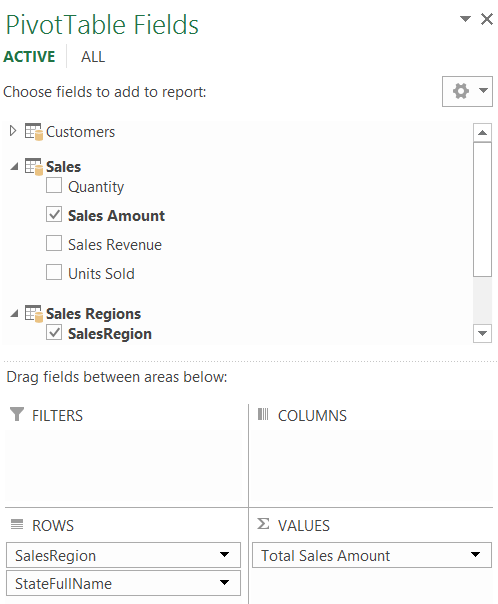
1. Examine the PivotTable on the worksheet named **Sales by Customer Age**. You should observe that Excel automatically added the calculated fields named **Sales Revenue** and **Units Sold** to the **VALUES** well when the calculated fields were created.



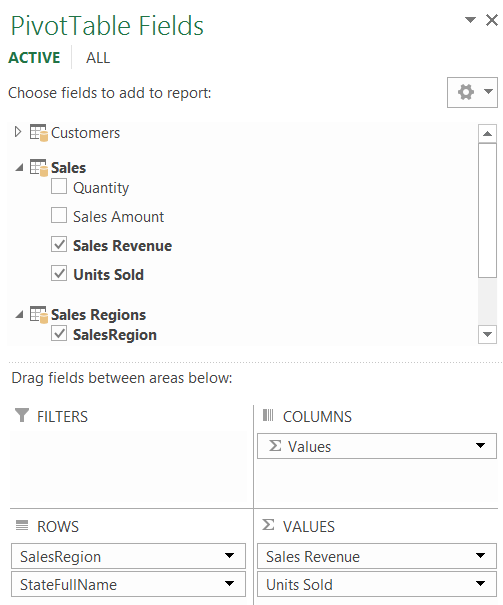
1. Remove the aggregate values named **Total Sales Amount** and **Units Sold** from the **VALUES** well. When you are done with this step, the **Sales Revenue** calculated field should be the only aggregated value inside the **VALUES** well.



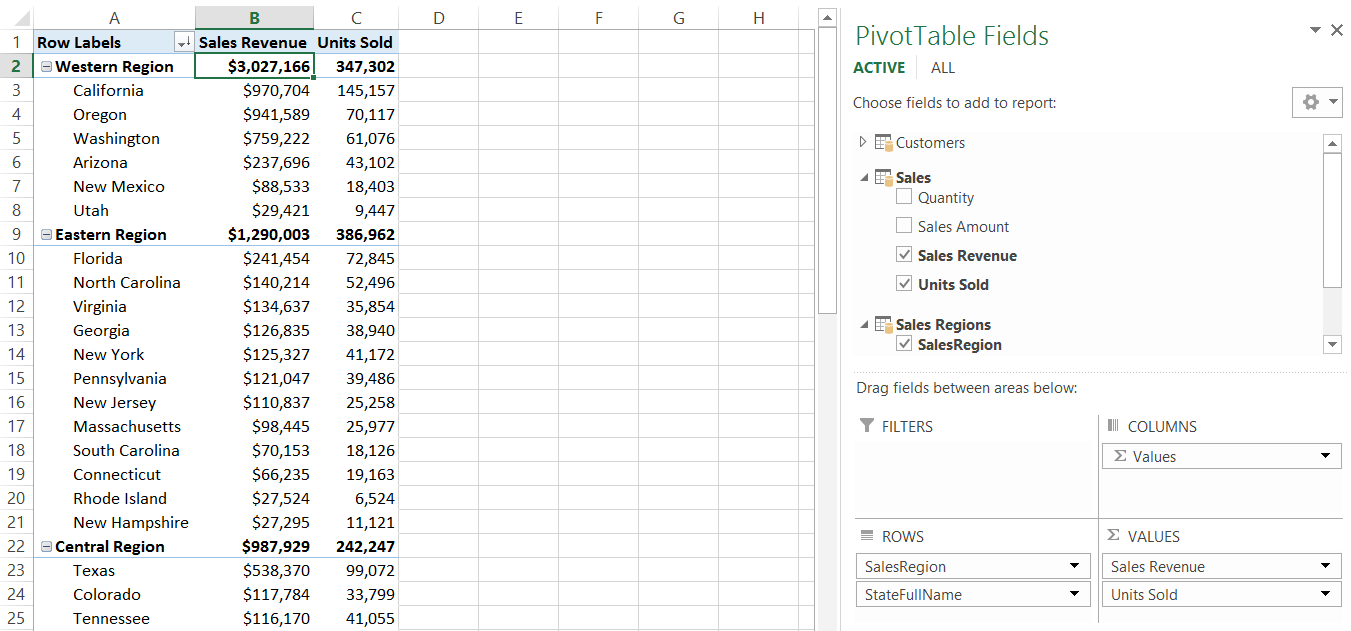
1. Use the tabs at the bottom of the workbook to navigate to the worksheet named **Sales By State**. Once you have navigated to the **Sales By State** worksheet, select one of the cells inside the PivotTable.
2. If you look at the **VALUES** well at the bottom of the task pane on the right, you should see it still contains the aggregate value that you named **Total Sales Amount**.



1. Remove **Total Sales Amount** value from the **VALUES** well and replace it with **Sales Revenue** and **Units Sold**.



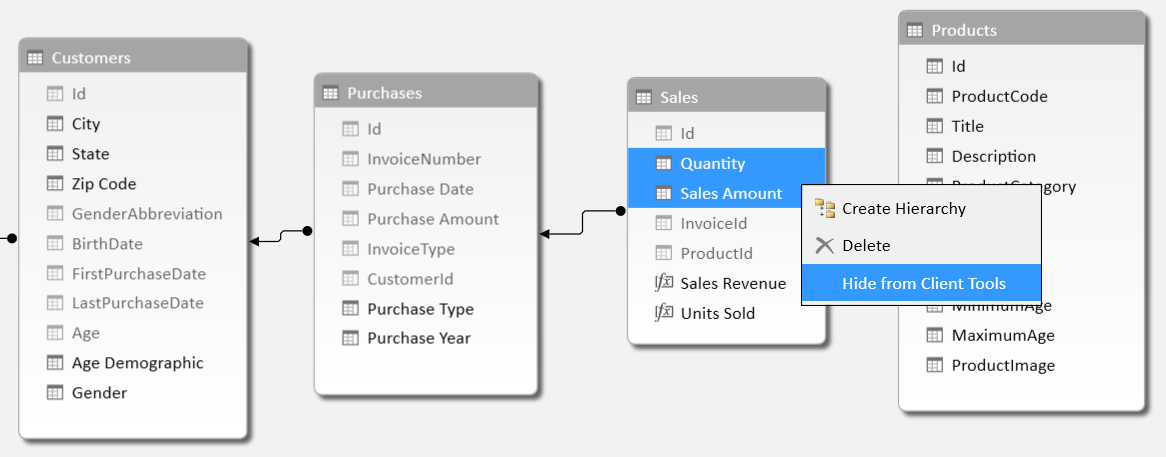
1. The Pivot Table on the **Sales By State** worksheet should now show state-by-state aggregated values for both sales revenue and the number of units sold.
2. Apply sorting to the rows in the PivotTable so that sales regions and states with higher sales revenue are sorted to the top. When you are done, your PivotTable should match the one shown in the following screenshot.



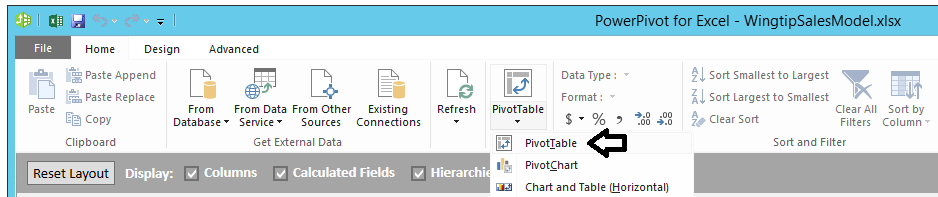
1. Save your changes to **WingtipSalesModel.xslx**.

#### Hide all columns in the Sales table except for the calculated fields

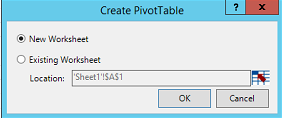
1. Return to the PowerPivot window for **WingtipSalesModel.xslx**.
2. Switch the PowerPivot window from **Data View** to **Diagram View** so you can see all the tables in the data model at once.
3. In the **Sales** table, hide the **Quantity** column and the **Sales Amount** column. After you complete this step, the only two columns that are still visible in the sales tables should be the calculated columns named **Sales Revenue** and **Units Sold**.



1. Create a new PivotTable that makes use of the two calculated columns named **Sales Revenue** and **Units Sold**.
   1. From the **Home** tab in the ribbon of the PowerPivot window, drop down the **PivotTable** menu control and click **PivotTable** to create a new pivot table in in the workbook **WingtipSalesModel.xslx**.

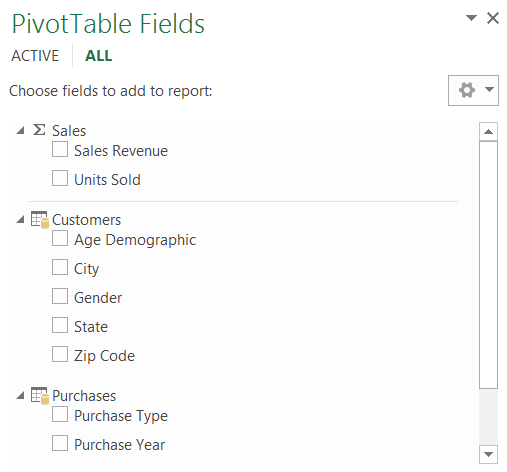


* 1. When the **Create PivotTable** dialog appears, select **New Worksheet** and click the **OK** button.

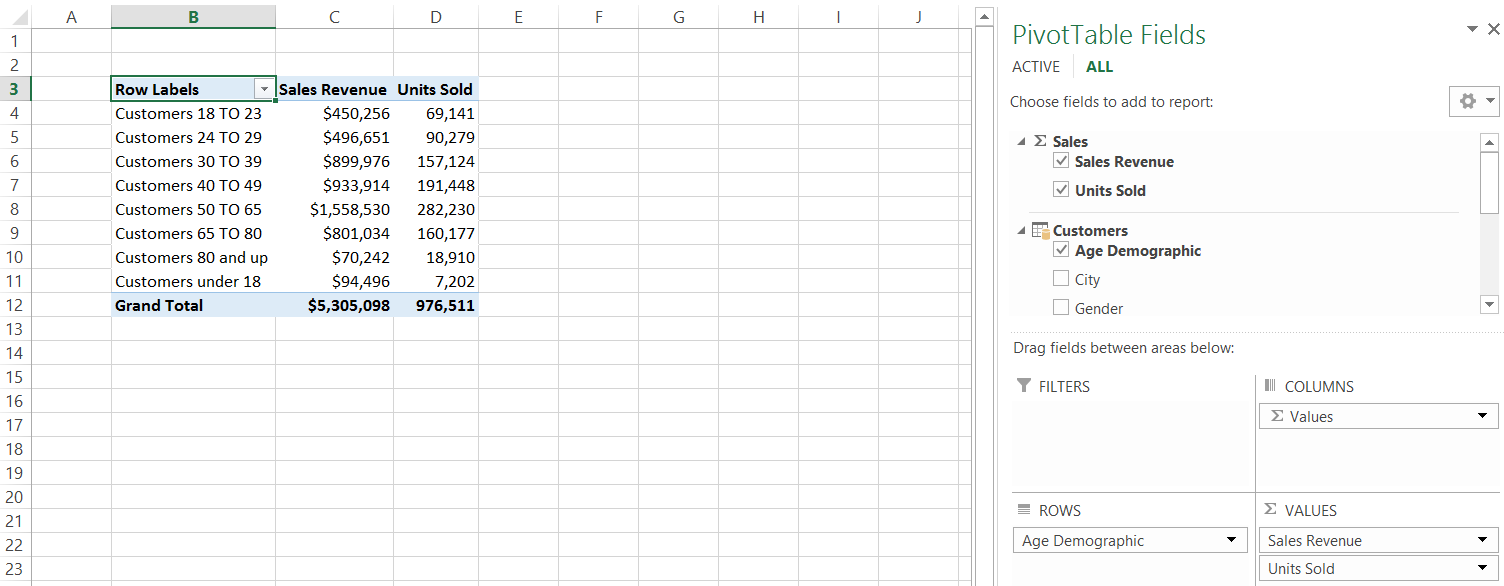


* 1. Once the new worksheet with the new PivotTable is added to the workbook, rename the worksheet **Sales by Purchase Type**.

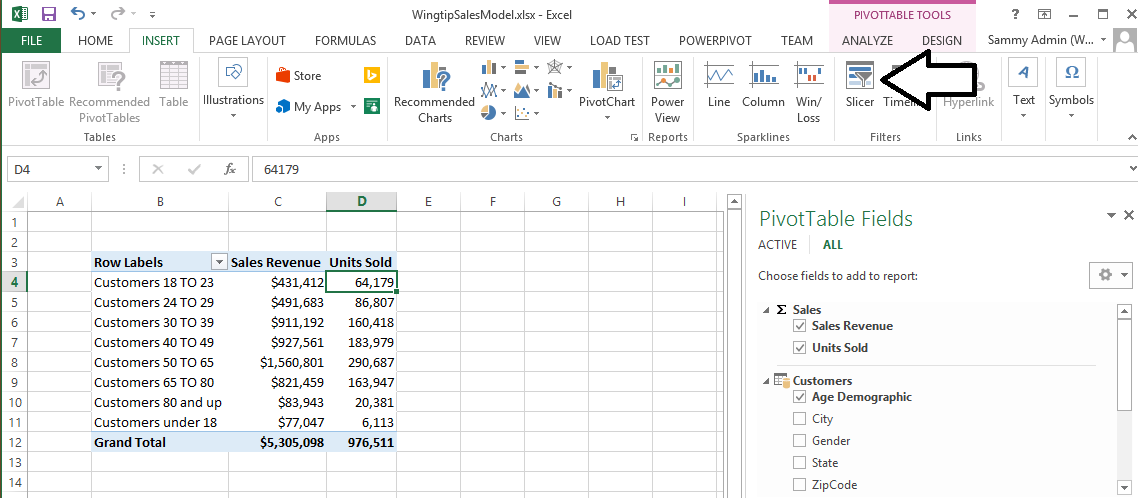
1. When you begin working on the new PivotTable, you should notice the effects of your latest changes to the data model. In particular, the **PivotTable Fields** section now displays the **Sales** table at the top with a Sigma (**Σ**) icon to signify that this table has no visible columns but only calculated fields.



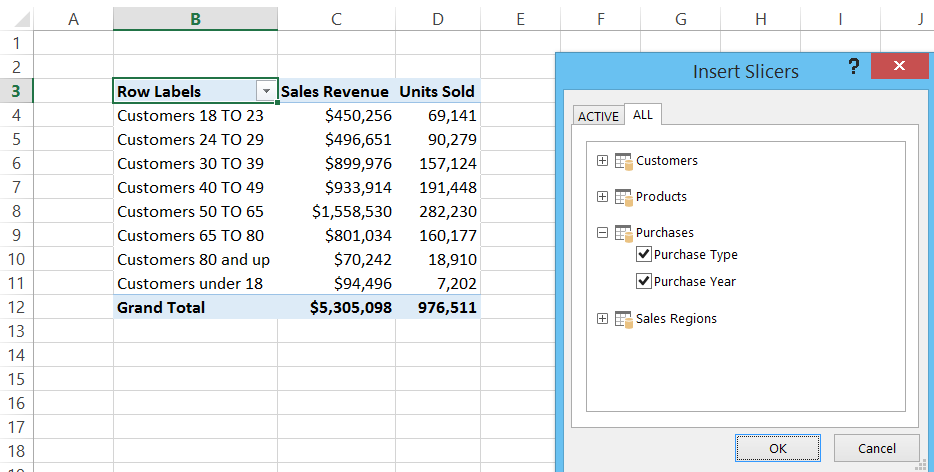
1. Modify the **PivotTable Fields** section in the pivot table as shown in the following screenshot to create a pivot table that display the calculated fields named **Sales Revenue** and **Units Sold** broken out by the **Age Demographic** column .



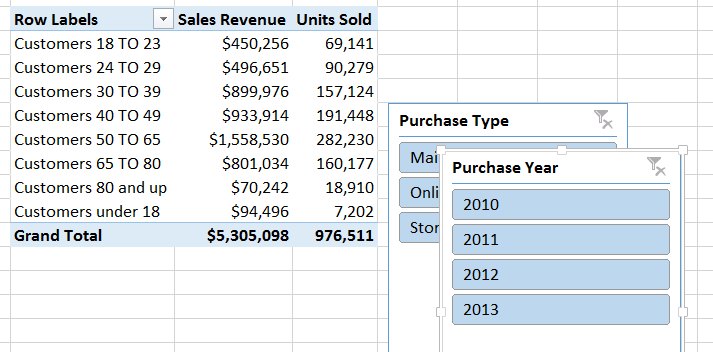
1. Add two slicers based to the **Sales by Purchase Type** worksheet to filter the PivotTable.
   1. Make sure you have one of the cells of the PivotTable selected.
   2. Click the **Insert** tab in the ribbon.
   3. In the **Filters** ribbon group, locate and click the **Slicers** button to display the **Insert Slicers** dialog.



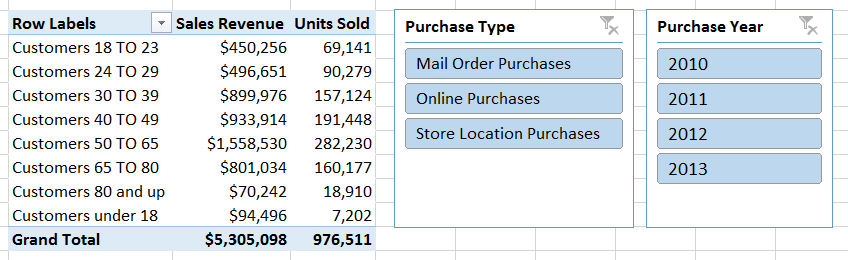
* 1. In the **Insert Slicers** dialog, click **ALL** to move from the **ACTIVE** tab to the **ALL** tab.
  2. From the **Purchases** table, select both **Purchase Type** and **Purchase Year**.



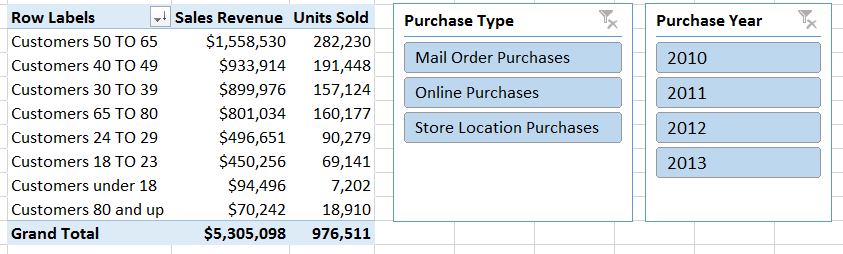
* 1. Excel should add the slicer elements onto the worksheet. However, you will need to move the slicers to produce a better layout for interacting with this PivotTable.



* 1. Use your mouse to move the **Purchase Type** slicer and the **Purchase Year** slicer so that our worksheet appears like the one shown in the following screenshot.



* 1. Take a moment to test the behaviors of the slicers. The content and totals within the PivotTable should change as you apply and remove filters using the two slicers.
  2. Sort the PivotTable so that the age groups with highest sales revenues are sorted to the top.



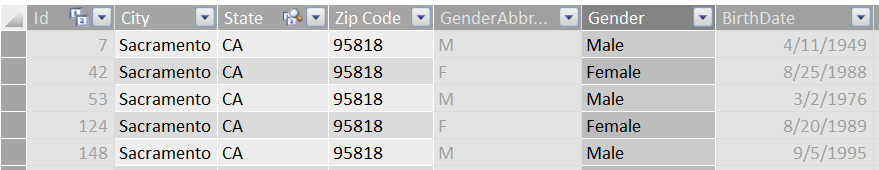
1. Save our changes to **WingtipSalesModel.xslx**.

### Exercise 4: Working with Hierarchies

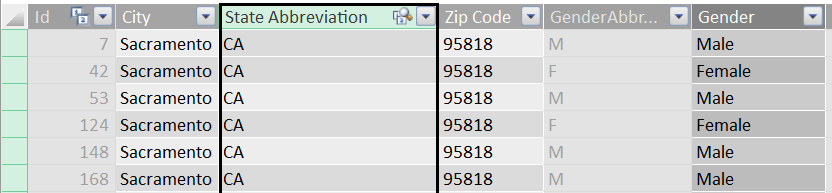
In this exercise you will create additional calculated columns in order to create two different dimensional hierarchies.

#### Add Calculated Columns to the Customers Table

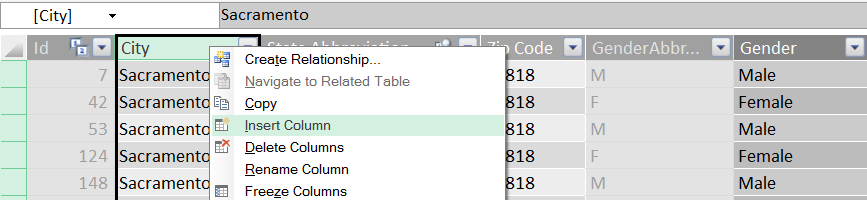
1. Navigate back to the PowerPivot window for **WingtipSalesModel.xslx**.
2. In the PowerPivot window, switch from **Diagram View** back to **Data View**.
3. Click on the **Customers** table to display its columns and data.



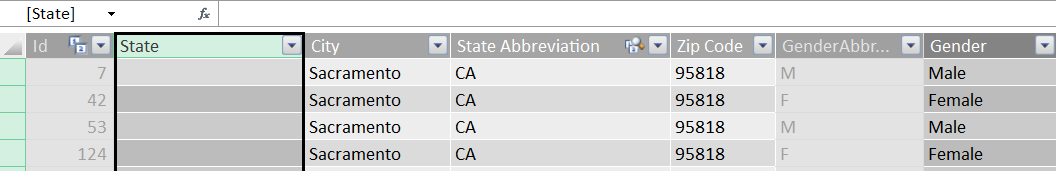
1. Change the name of the **State** column to **State Abbreviation**.



1. Create a new calculated column named **State** to display the full state name.
   1. Right-click on **City** column and then click **Insert Column** to create a new calculated column to the left of the **City** column.

.

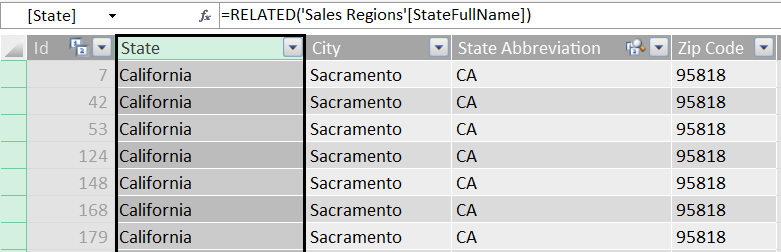
* 1. Rename the new column to **State**.



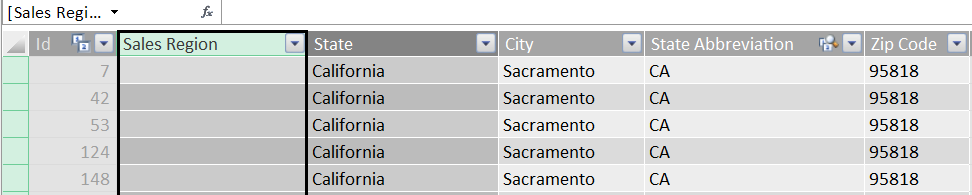
* 1. Enter the following DAX expression for the new calculated column named **State**.

=RELATED('Sales Regions'[StateFullName])

* 1. The new calculated column named **State** should look like the one in the following screenshot.



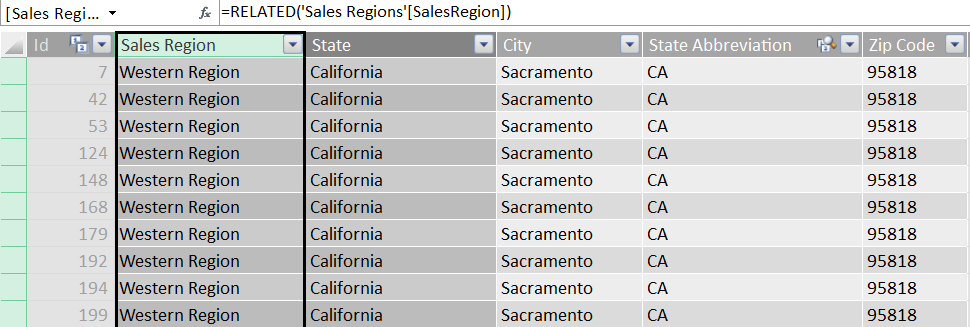
1. Create a calculated column named **Sales Region**.
   1. Right-click on **State** column and then click **Insert Column** to create a new calculated column to the left of the **State** column. Rename the new column to **Sales Region**.

.

* 1. Enter the following DAX expression for this calculated column.

=RELATED('Sales Regions'[SalesRegion])

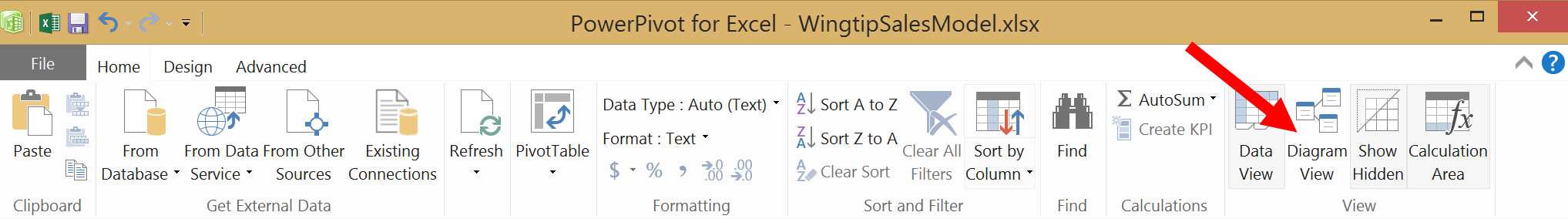
* 1. Verify that the **Sales Region** columns displays a value associated with the state.



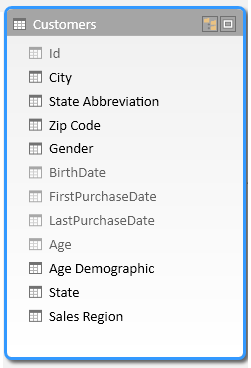
You have now created the required calculated columns in the **Customers** table so that you can now move ahead and create a new hierarchy based on the geographical location of each **Customer**.

#### Create the Customer Geography Hierarchy

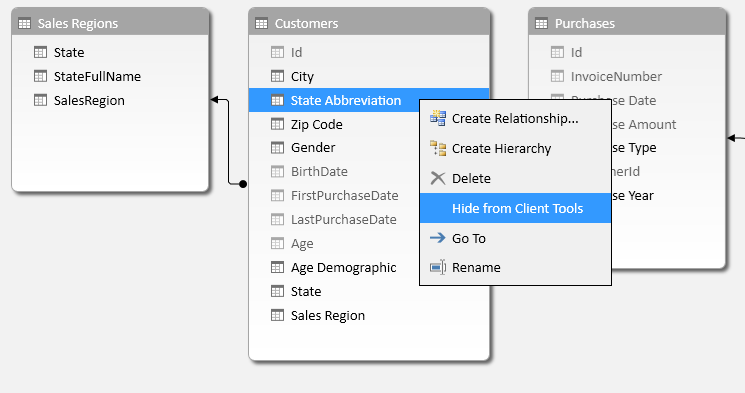
1. From the Ribbon in the **View** group, click on the **Diagram View** button.



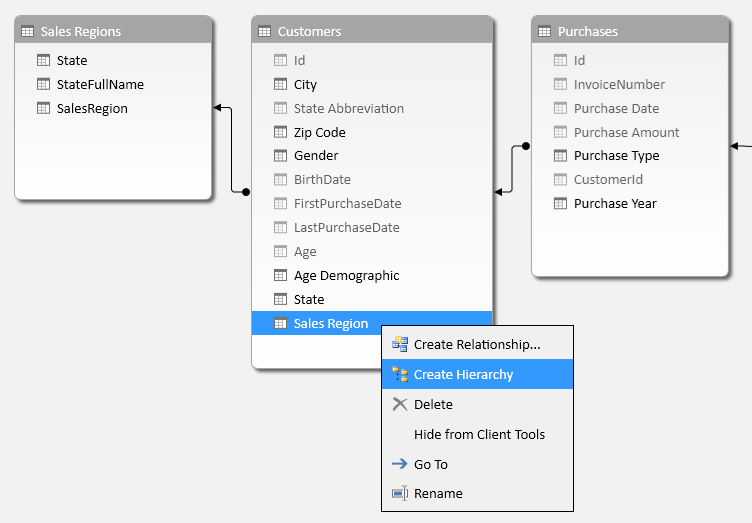
1. Inspect the current state of the **Customers** table in **Diagram View**.



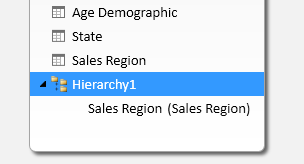
1. Hide the column named State Abbreviation.



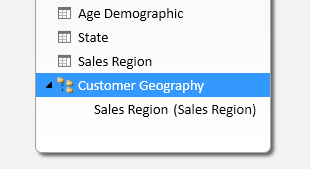
1. Right-click on the **Sales Region** column and then click the **Create Hierarchy** command.



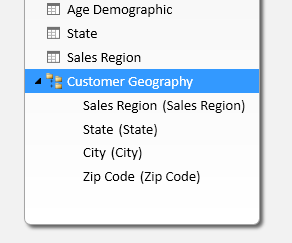
1. When you execute the **Create Hierarchy** command, you should see that the new hierarchy is created at the bottom of the **Customer** table and given a default name of **Hierarchy1**.



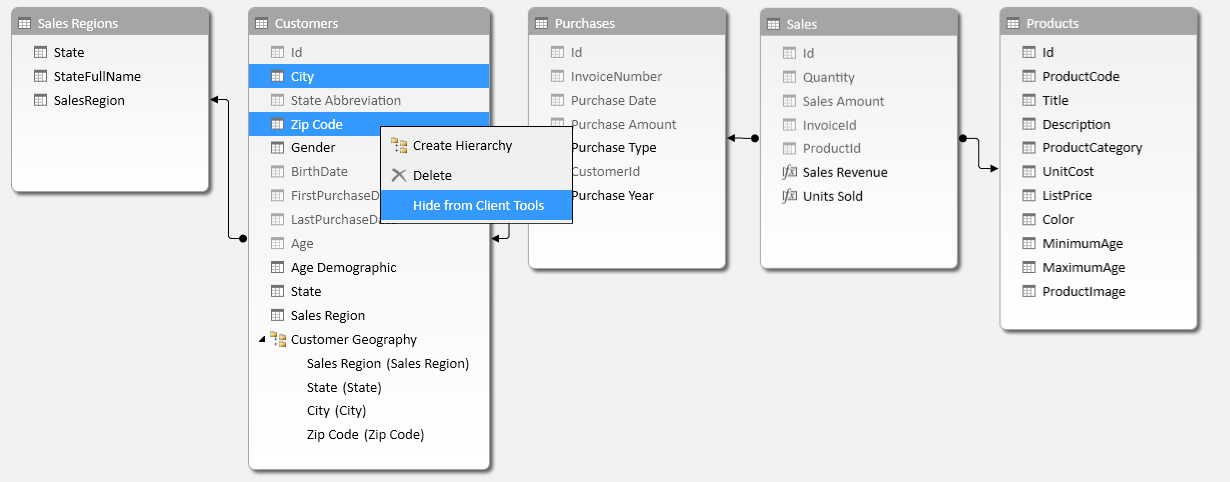
1. Rename the new hierarchy from **Hierarchy1** to **Customer** **Geography**.



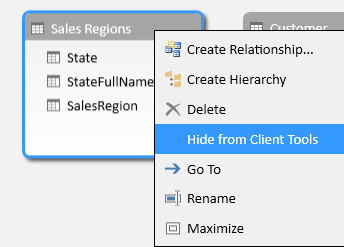
1. Add the columns named **State**, **City** and **Zip Code** to the hierarchy by dragging and dropping them with your mouse.



1. In the **Customers** table above the hierarchy, hide the columns named **City** and **Zip Code** so those columns can only be accessed through the **Customer Geography** hierarchy.

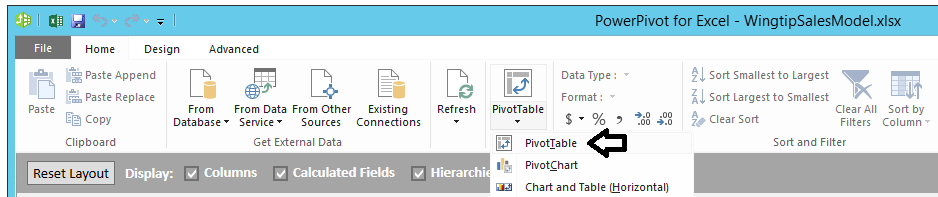


1. Hide the **Sales Regions** table.

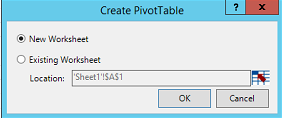


#### Create a PivotTable to Break Down Sales Revenue by Customer Geography

1. Create a new PivotTable that makes use of the hierarchy you just created.
   1. From the **Home** tab in the ribbon of the **POWERPIVOT** window, drop down the **PivotTable** menu control and click **PivotTable** to create a new pivot table in in the workbook **WingtipSalesModel.xslx**.

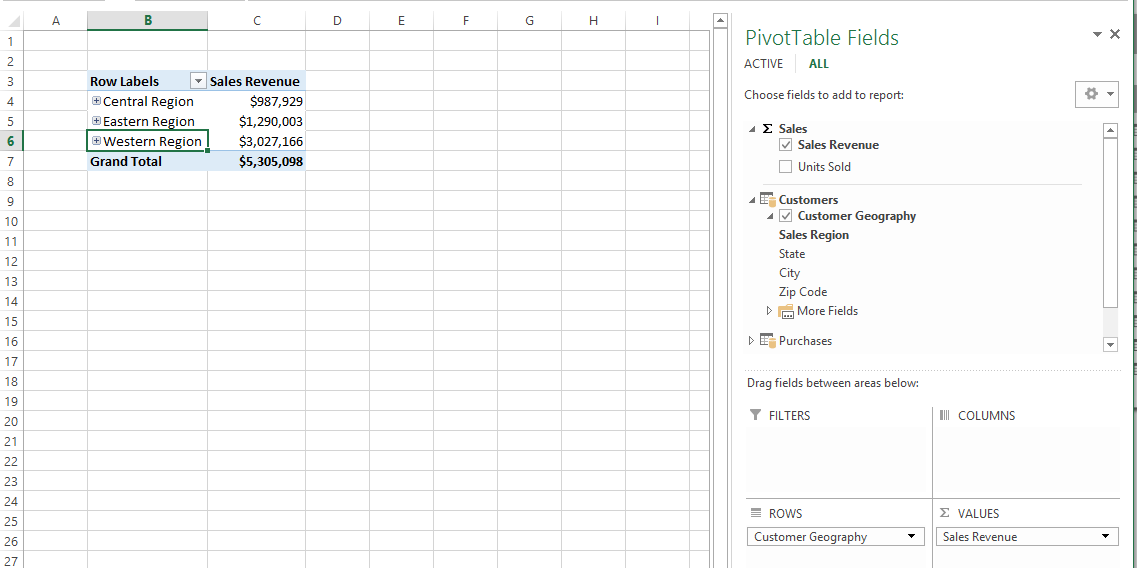


* 1. When the **Create PivotTable** dialog appears, select **New Worksheet** and click the **OK** button.

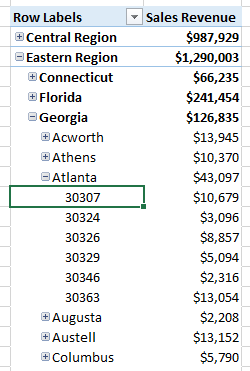


* 1. Once the worksheet with the PivotTable is added to the workbook, rename the worksheet **Sales by Customer Geography**.

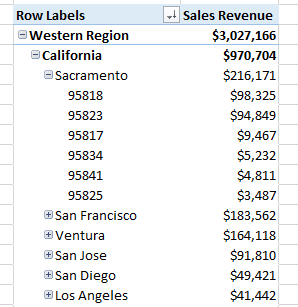
1. Create the layout of the PivotTable by adding the **Sales Revenue** calculated field to **VALUES** section and then adding the **Customer Geography** hierarchy to the ROWS section as shown in the following screenshot.



1. Experiment by drilling down into all levels of the **Customer Geography** hierarchy so you can see sales revenue totals at the granularity of state, city and zip code.



1. ***Student challenge***: sort all four levels of **Customer Geography** hierarchy in descending order to show higher sales revenue at the top. When you are done, your pivot table should look like the one in the following screenshot.

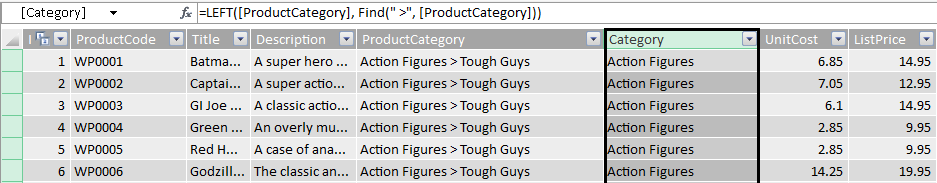


#### Create Calculated Columns for the Product Category Hierarchy

1. Return to the PowerPivot window.
2. Navigate to **Products** table in Data View.
3. Create a new calculated column named **Category**.
   1. Create a new calculated in between **ProductCategory** and **UnitCost**.
   2. Rename the new column **Category**.
   3. Type in the following DAX expression. Alternatively, you can copy-and-paste the text for this DAX expression from the file in the student folder at **C:\Student\Modules\ExcelPowerPivot\Lab\** **ProductCategoryDAX.txt**.

=LEFT([ProductCategory], Find(" >", [ProductCategory]))

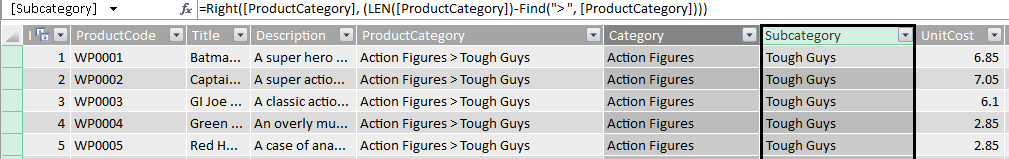
* 1. When you are done, you screen should match the following screenshot.



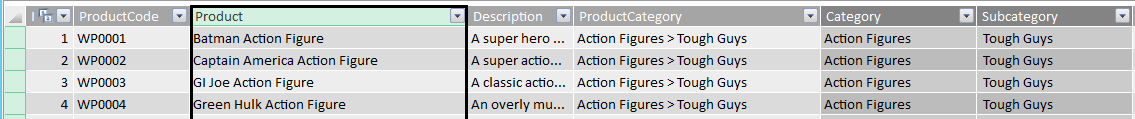
1. Create a new calculated column named **Subcategory**.
   1. Create a new calculated in between **Category** and **UnitCost**.
   2. Rename the new column **Subcategory**.
   3. Type in the following DAX expression. Alternatively, you can copy-and-paste the text for this DAX expression from the file in the student folder at **C:\Student\Modules\ExcelPowerPivot\Lab\** **ProductSubcategoryDAX.txt**.

=Right([ProductCategory], (LEN([ProductCategory])-Find("> ", [ProductCategory])))

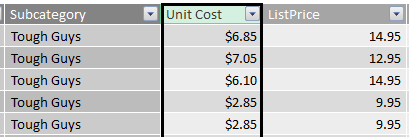
* 1. When you are done, you screen should match the following screenshot.



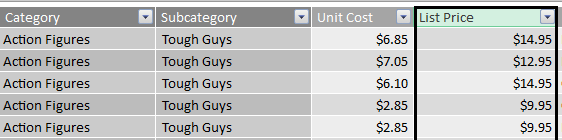
1. Change name of **Title** column to **Product**.



1. Modify the **UnitCost** column.
   1. Change its name from **UnitCost** to **Unit Cost**.
   2. Change its formatting to Currency.

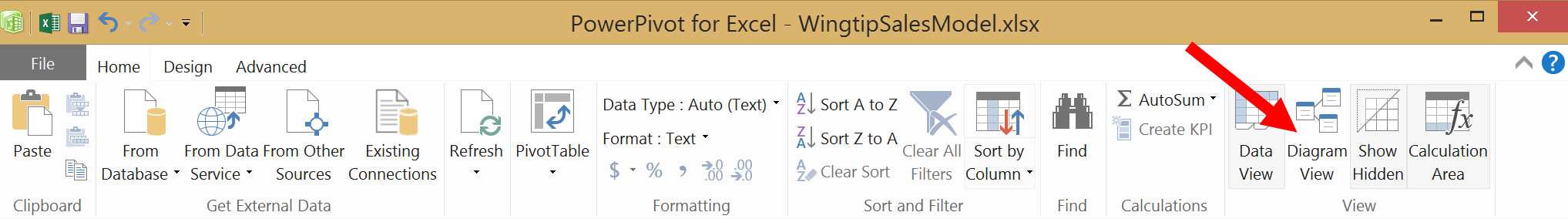


1. Modify the **ListPrice** column.
   1. Change its name from **ListPrice** to **List Price**.
   2. Change its formatting to Currency.

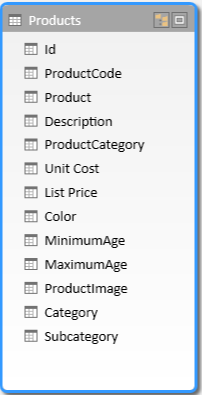


#### Create the Product Category Hierarchy

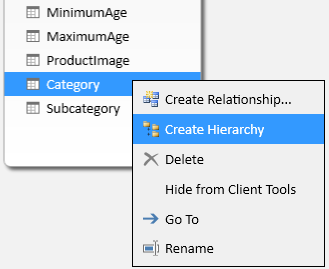
1. From the Ribbon in the **View** group, click on the **Diagram View** button.



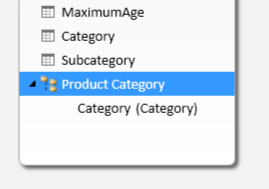
1. Inspect the current state of the **Products** table in **Diagram View**.



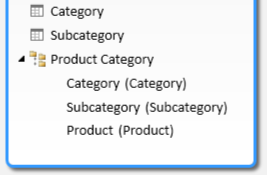
1. Right-click on the **Category** column and then click the **Create Hierarchy** command.



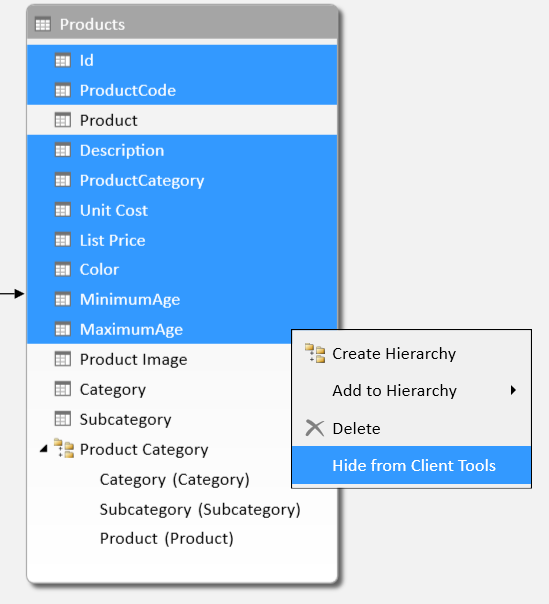
1. When you execute the **Create Hierarchy** command, you should see that the new hierarchy is created at the bottom of the **Customer** table and given a default name of **Hierarchy1**. Rename the new hierarchy from **Hierarchy1** to **Product Category**.



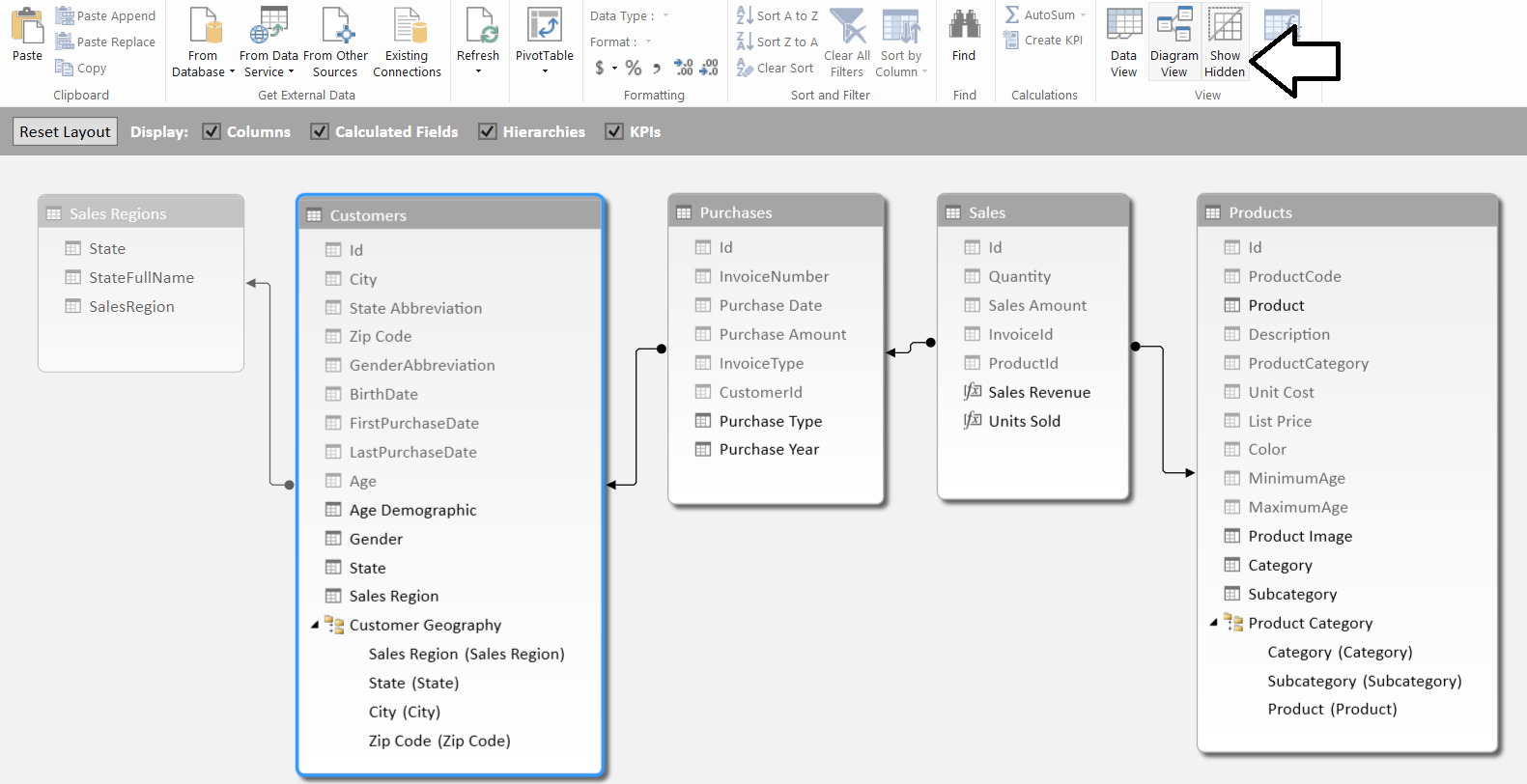
1. Add the **Subcategory** column and **Product** column to the **Product Category** hierarchy underneath the **Category** column by dragging and dropping them with the mouse.



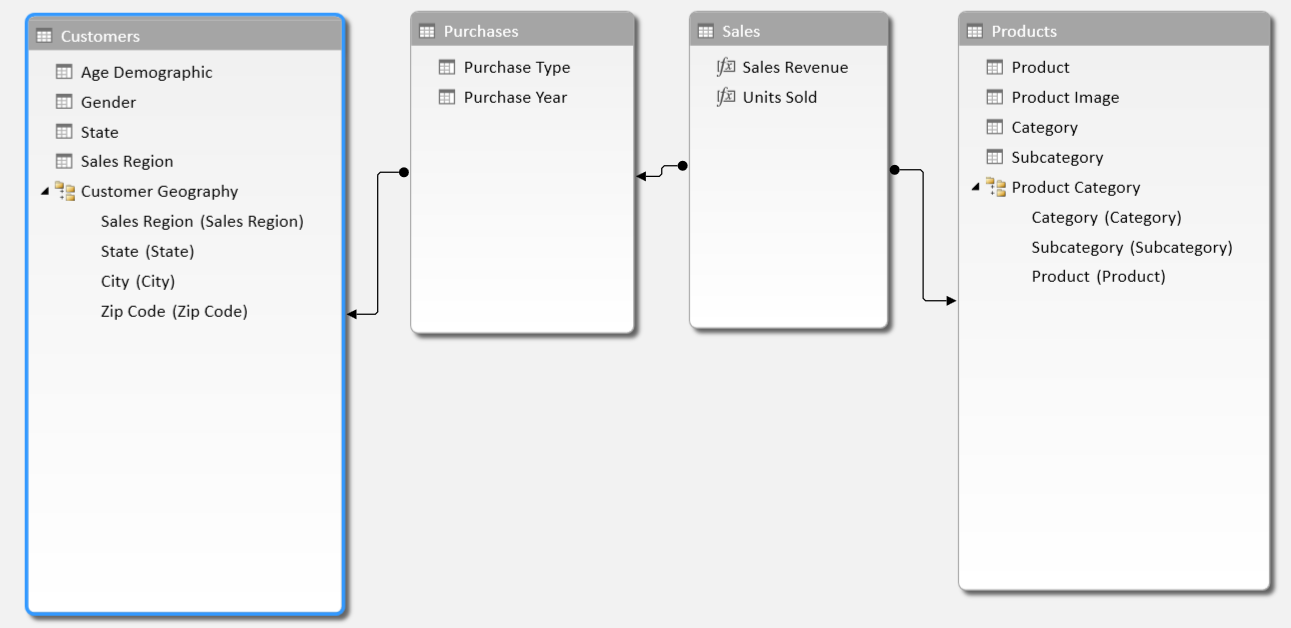
1. Locate the **ProductImage** column in the **Products** table and rename this column to **Product Image**.
2. Above the hierarchy, hide all the columns in the **Products** table except for **Product**, **Product Image**, **Category** and **Subcategory**.



1. Examine what the current state of the Data Model looks like to a client tool such as Excel.
   1. Locate the **Show Hidden** button in the **View** ribbon group of the **Home** tab.



* 1. Click the **Show Hidden** button to toggle back and forth between showing and hiding the columns that you have hidden from client tools. When you hide the hidden members, you can see that the data model is now much easier to understand.

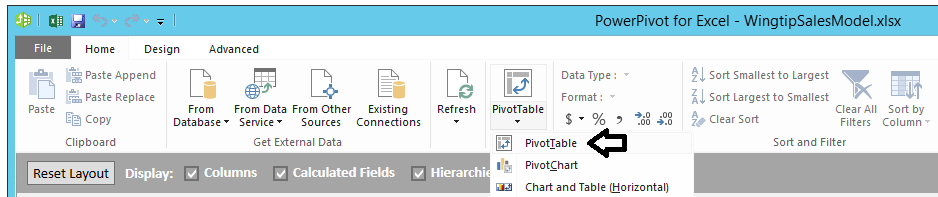


* 1. Click to **Show Hidden** button once more to shown the hidden columns in the PowerPivot window.

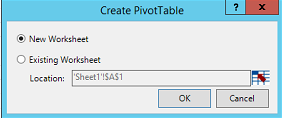
Typically, you should leave the PowerPivot window configured to show the hidden columns instead of hiding them because certain tasks working with the Data Model require them to be visible.

#### Create a New PivotTable to Break Down Sales Revenue by the Product Category Hierarchy

1. Create a new PivotTable that makes use of the Product Category hierarchy you just created.
   1. From the **Home** tab in the ribbon of the **POWERPIVOT** window, drop down the **PivotTable** menu control and click **PivotTable** to create a new pivot table in in the workbook **WingtipSalesModel.xslx**.

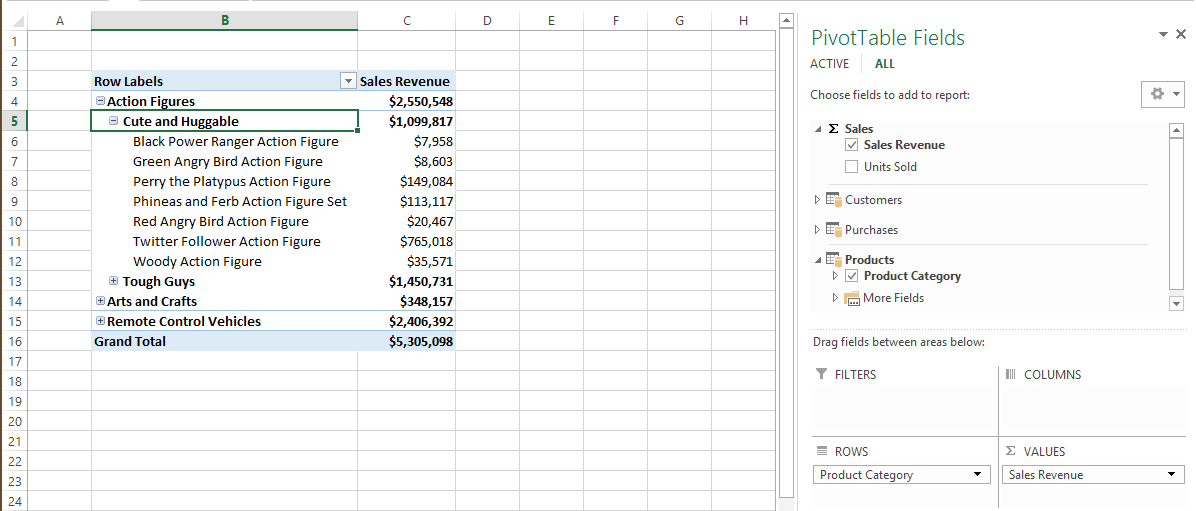


* 1. When the **Create PivotTable** dialog appears, select **New Worksheet** and click the **OK** button.

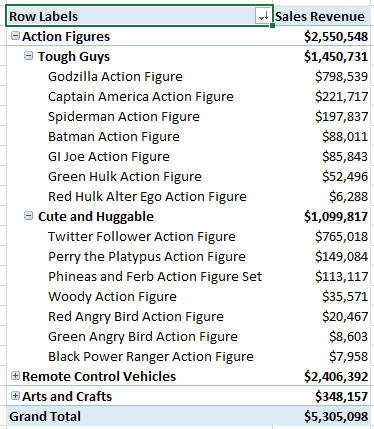


* 1. Once the worksheet with the PivotTable is added to the workbook, rename the worksheet **Sales by Product Category**.

1. Create the layout of the PivotTable by adding the **Sales Revenue** calculated field to **VALUES** section and then adding the **Product Category** hierarchy to the **ROWS** section as shown in the following screenshot.



1. ***Student challenge***: sort all three levels of **Product Category** hierarchy in descending order in the PivotTable to show higher sales revenue at the top. When you are done, your pivot table should look like the one in the following screenshot.



1. Save your changes to **WingtipSalesModel.xlsx**.

Great work. You have now completed this lab.