Creating Advanced Queries with Power BI Desktop

Setup Time: 60-90 minutes

Lab Folder: C:\Student\Modules\01_Queries\Lab

Overview: While the focus of this lab is advanced query design, it is also the first lab of this training course. Therefore, you will begin by setting up your computer and creating a Power BI environment which you will accomplish over the first three exercises. Once you have completed the first three exercises, you will create a Power BI Desktop project in which you will create an extensive set of queries using advanced techniques to process advanced ETL logic.

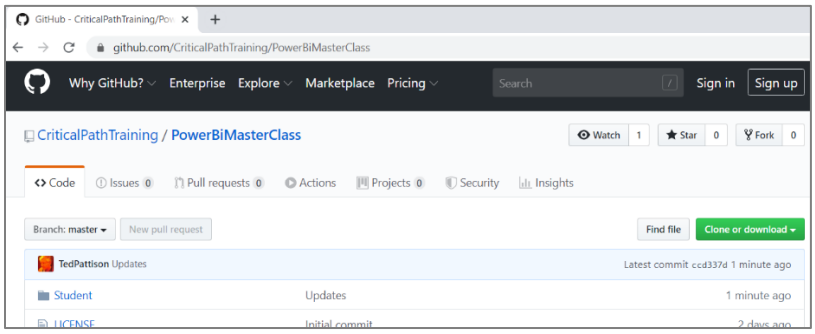
Exercise 1: Download the Student Lab Files to a Local Folder

In this exercise, you will download a local copy of the student files from the **PowerBiMasterClass** repository in GitHub.

1. Navigate to the GitHub repository named **PowerBiMasterClass** which holds the student files for this training class
   a) Launch a browser and navigate to the GitHub repository for this course at the following URL.

   https://github.com/CriticalPathTraining/PowerBiMasterClass

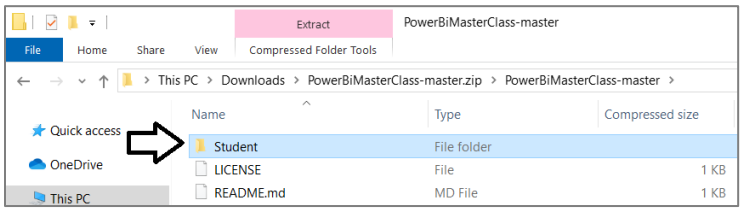
   b) You should see the home page for the repository as shown in the following screenshot.



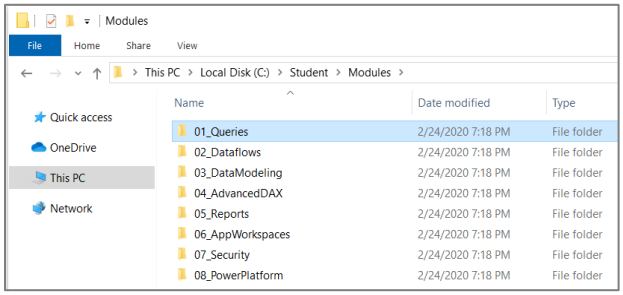
2. Download the ZIP archive with the Power BI Master Class student files to your local computer.
   a) Download the zip archive named master.zip using the following download URL.

   https://github.com/CriticalPathTraining/PowerBiMasterClass/archive/master.zip

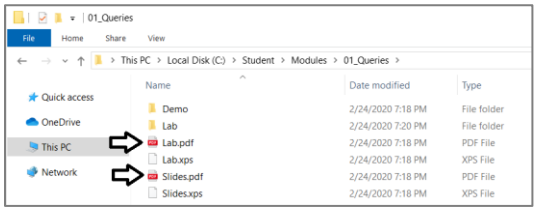
   b) Open the ZIP archive and copy the **Student** folder to the Windows clipboard.



   c) Copy the student folder to your local hard drive at a local path of **C:\Student**.



d) Drill into the **Modules** folder and look in the folder inside named **01_Queries**.



You can see that the **01_Queries** folder contains child folders named **Demo** and **Lab** as well as **Slides.pdf** and **Lab.pdf**. Note that each module in this course has its own folder in the **Models** folder with slides and lab exercise in files named **Slides.pdf** and **Lab.pdf**.

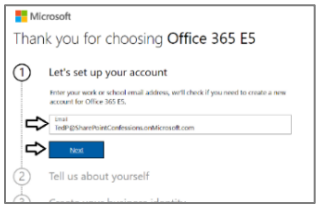
Exercise 2: Sign Up for an Office 365 E5 Trial

In this exercise you will create a new Azure AD trial tenant. This new Azure AD tenant will serve as your Power BI environment for publishing and deploying dataflows, datasets, reports, dashboards and apps.

1. Navigate to the Office 365 trial sign up web page.
   a) Launch the Chrome browser.
   b) Copy and paste the following URL into the address bar of the incognito window to navigate to the signup page.

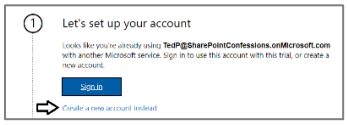
```
https://go.microsoft.com/fwlink/p/?LinkID=698279&culture=en-US&country=US
```

   c) You should now see the form to create your new **Office 365 E5** trial. Enter your email address and click **Next**.

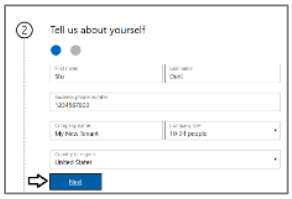


If you enter an email address for an organizational account, the form provides the option to sign in using that account. Do not click the **Sign in** button because you don't want to sign with an existing organizational account. The purpose of this exercise is to create a brand new organizational account in a brand new Microsoft 365 tenant.

   d) Click the **Create a new account instead** link.



   e) Enter your **First name** and **Last name**.
   f) Enter your mobile phone number as the **Business phone number**.
   g) Provides values for **Company size** and **Country or region** and click **Next**.

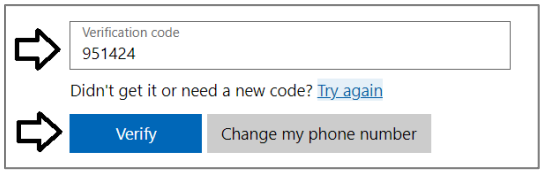


Whatever **Company name** you enter will be used as the name of the Azure AD tenant that will be created during the sign up process.

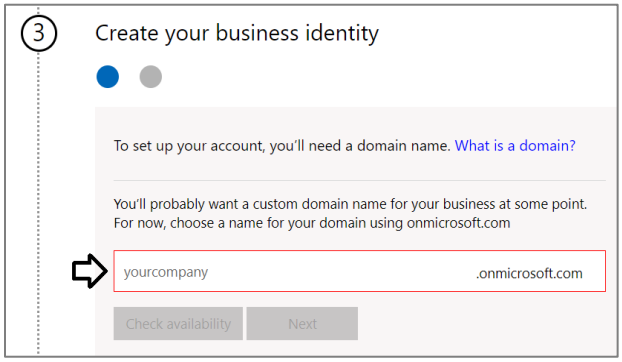
h) When prompted to prove you're not a robot, select the **Text me** option and enter the Phone number of your mobile phone.

i) Click **Send Verification Code**.

j) Retrieve the access code from your mobile device and use it to complete the validation process.

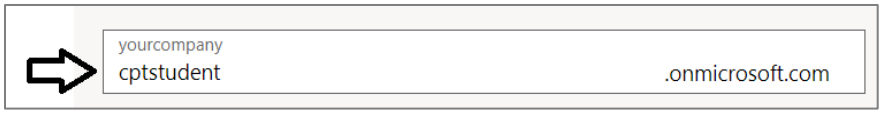


k) In the **Create your business identity** step, locate the textbox into which you will enter a domain name.



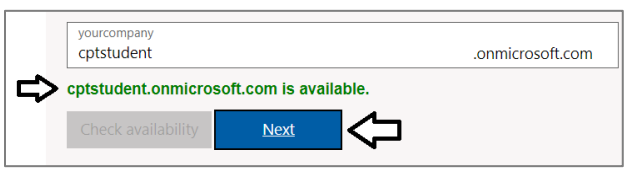
Note that the company name you enter in this textbox will be used to create an Internet domain name for a new Microsoft 365 tenant. For example, if you were to enter a company name of **cptstudent**, it would result in the creation of a new Office 365 tenant within a domain of **cptstudent.onMicrosoft.com**. The user name you enter will be used to create the first user account which will be given global admin permissions throughout the Azure AD tenant. If you enter a user name of **Student**, then the email address as well as user principal name for this account will be **student@cptstudent.onMicrosoft.com**

l) Enter a domain name for your new Microsoft 365 tenant.

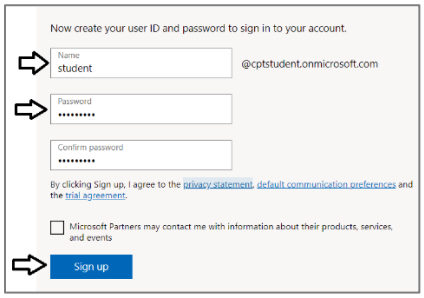


m) If the domain name you enter is not available, modify the domain name until you can verify that it is available.

n) Once you have created a domain name that is available, click **Next**.

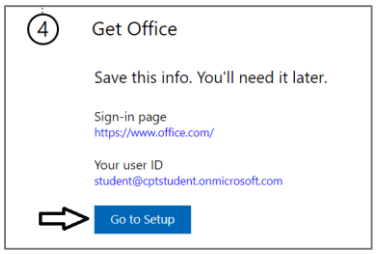


o) Enter a **Name** for your user account, a **Password** that you will remember and then click **Sign up**.



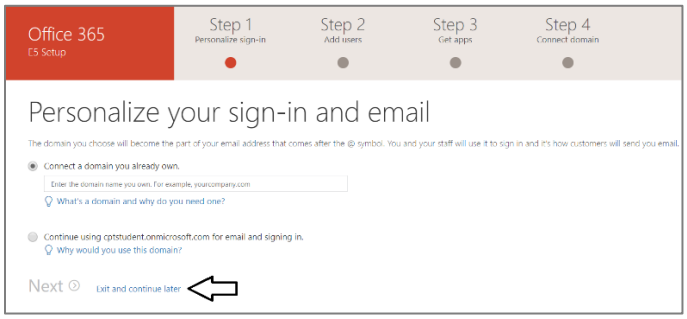
At this point, the Sign up process should begin to provision your new Microsoft 365 tenant and your new organizational account.

p) Once the provision process completes, take note of your new **user ID** and click the **Go To Setup** button.

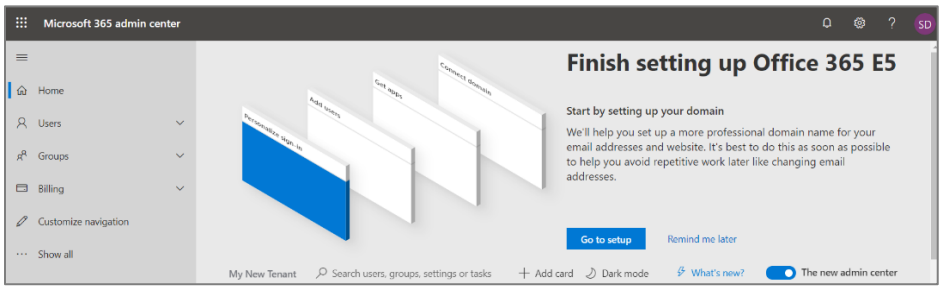


You have just created a new Microsoft 365 tenant with a 30-day trial for 25 Office 365 E5 licenses. Note that some Microsoft cloud services within your new tenant such as the Microsoft 365 admin center, Power BI, Power Apps and Power Automate can be accessed immediately. Other Office 365 services such as SharePoint Online, OneDrive for Business and your Outlook mailbox will not be ready immediately and can take some time to provision.

q) If you see the **Personalize your sign-in and email** setup page, click **Exit and continue later**.

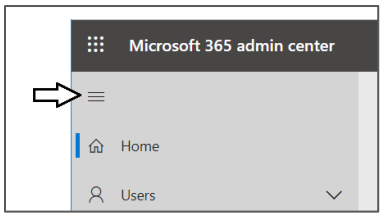


r) You should now be located at the home page of the **Microsoft 365 admin center**.

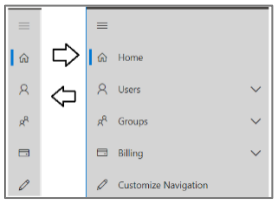


If you don't see the home page of the **Microsoft 365 admin center**, navigate to https://admin.microsoft.com/Adminportal.

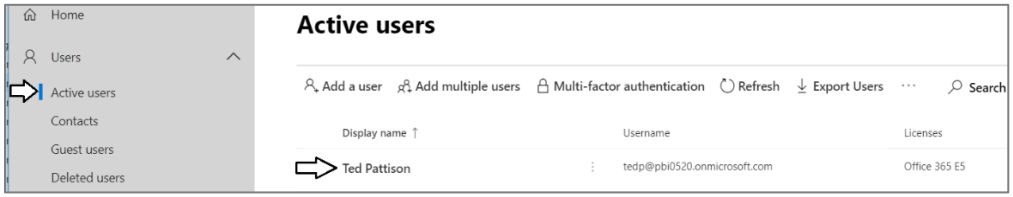
2. Inspect the set of active users in the current Azure AD tenant.
   a) Locate the top **Collapse navigation menu** with the hamburger icon just under the Microsoft 365 App Launcher menu.



   b) Toggle the **Collapse navigation menu** button to see how it collapses and expands the left navigation menu.



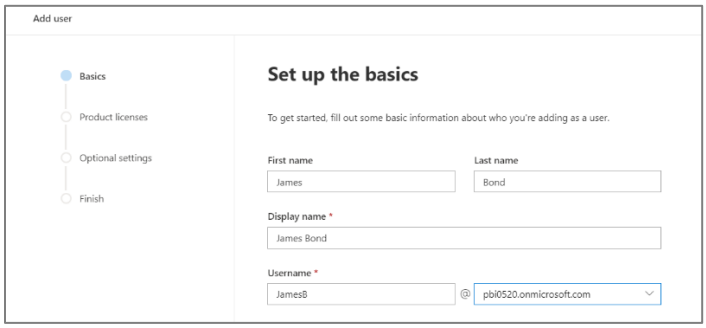
   c) Navigate to the **Active users** view where you should be able to verify that the user account you are currently logged in as is the only user account that exists in the current tenant.



Remember that your account is global tenant administrator. That means you are king of the castle for this Office 365 tenant and you have all the permissions you need to configure any tenant-level settings for Power BI.

3. Create a second Azure AD user account in your new Azure AD tenant.
   a) On the **Active Users** page, click the **Add a user** button to create a new user account

   b) Fill in the **Set up the basics** form with information for a new user account. When creating this account, you can use any name you would like. These lab instructions will demonstrate this by creating a user account for a person named **James Bond** with a user name and email of **JamesB@cptstudent.onmicrosoft.com**.



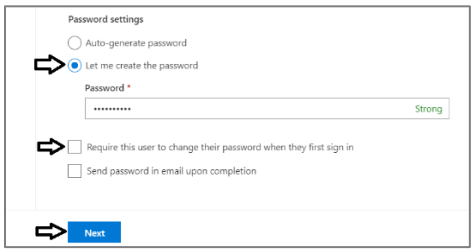
c) Move below to the **Password settings** section.

d) Select the option for **Let me create the password**.

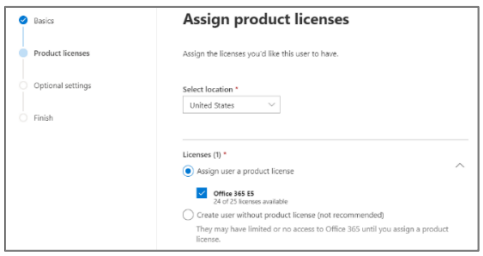
e) Enter a password of **pass@word1** into the textbox labeled **Password**.

f) Uncheck the checkbox for the **Require this user change their password when they first sign in** option.

g) Click **Next**.



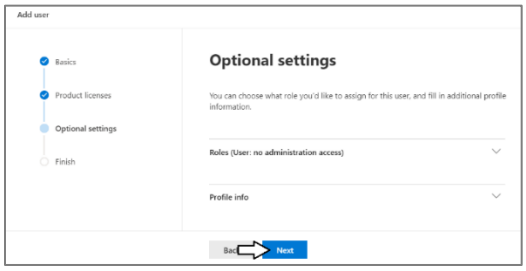
h) In the **Product licenses** section, make sure the **Office 365 E5** license is set to **On**.



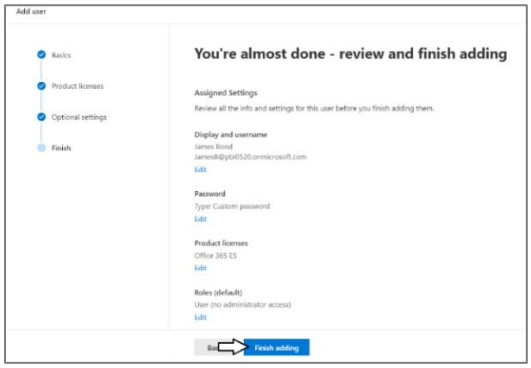
Note that the new account is usually assigned a trial license for **Office 365 E5** plan. However, it's a good practice to check and make sure the new user has been assigned a license for **Office 365 E5** which includes the **Power BI Pro** license.

i) Click the **Next** button down below.

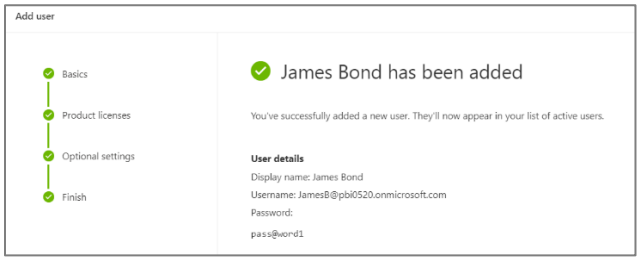
j) On the **Optional settings** view, click **Next**.



k) On the **Finish** view, Click the **Finish adding** button at the bottom to create the new user account.



l) You should see the **Finish** view with a message indicating that the new user account has been created.



m) Click the **Close** button at the bottom of the **Finish** view to close the **Add User** pane on the right.

n) Verify that the new user account has been created and is displayed along with your primary Office 365 user account.

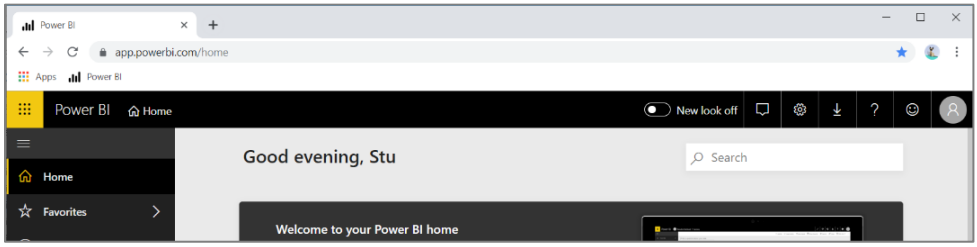
Now you have a secondary user account that does not have any administrative permissions. It's important that you test reports, dashboards and apps with standard user accounts to ensure your application doesn't require users with special permissions.

Exercise 3: Sign into the Power BI Service

In this exercise you will sign into the Power BI Service using your primary Office 365 account.

1. Log into the Power BI Service with your primary Office 365 account.

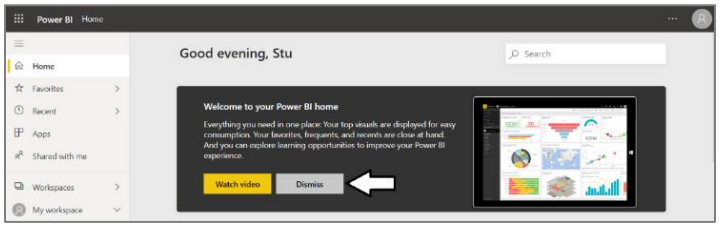
a) Navigate the Power BI portal at https://app.powerbi.com and if prompted, log in using your primary Office 365 account.



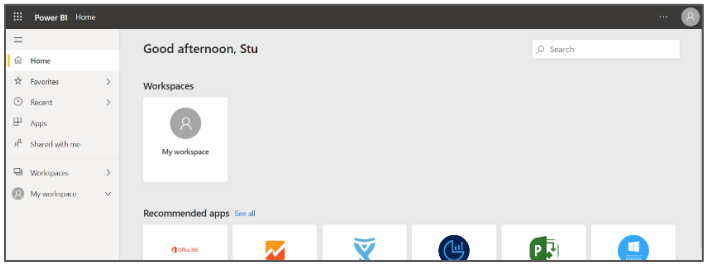
b) Drop down the User login menu in the top right corner of the screen and make sure you are logged with the new organizational account you just created and that you are not logged on using pre-existing user account such as your work account.

c) Locate the **New look** toggle and switch it from **New look off** to **New look on**.

d) Click the **Dismiss** button to remove the **Welcome to your Power BI home** panel.



e) The home page of the Power BI Service should now look something like the following screenshot.

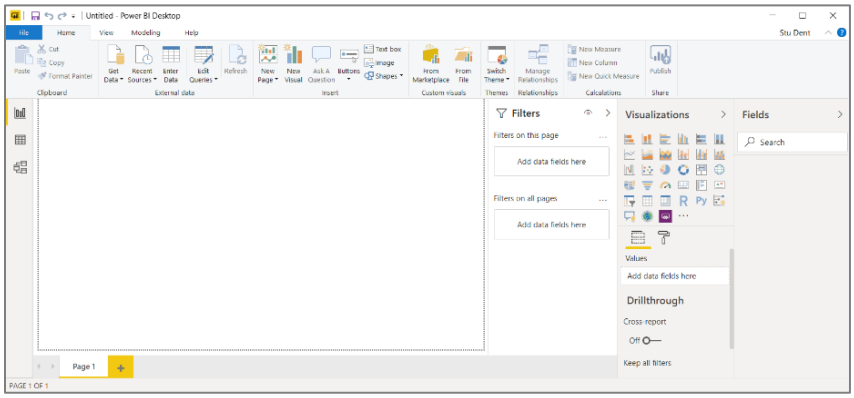


If you haven't worked with the new look of the Power BI Service yet, here's your big chance to get familiar with it.

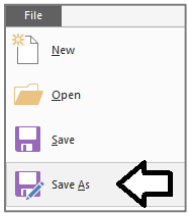
Exercise 4: Extract Customer Data using the Web Connector

It is assumed that all students have already installed the November 2019 edition of Power BI Desktop. In this exercise, you will create a new Power BI Desktop project named **CustomerSales.pbix** and begin to import customer data from a CSV file.

1. Launch Power BI Desktop.
   a) Power BI Desktop should now be running with a new, unsaved project.



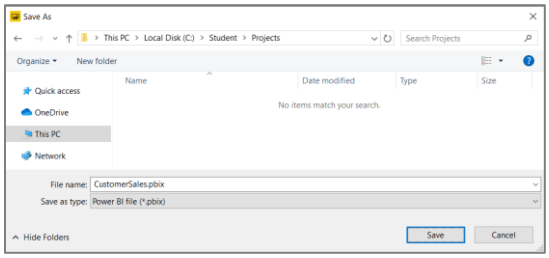
2. Before beginning your work, save the empty project as a PBIX file.
   a) Drop down the **File** menu and click the **Save As** command.



b) Save the PBIX file as **CustomerSales.pbix** using the following path location.

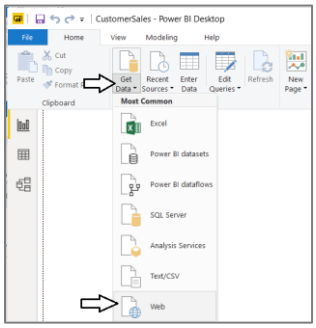
```
C:\Student\Projects\CustomerSales.pbix
```

c) Click the **Save** button on the **Save As** dialog to save the PBIX file.



3. Import customer data from as CSV file accessible over the Internet named **Customers.csv**.

a) Drop down the **Get Data** menu from the Home tab of the ribbon and select the **Web** connector.



b) When you are prompted by the **From Web** dialog, copy and paste the following URL to import data from **Customers.csv**.

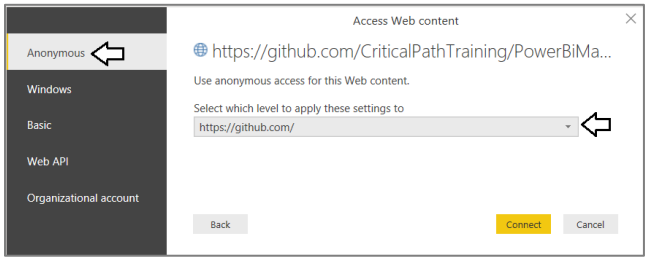
```
https://github.com/CriticalPathTraining/PowerBiMasterClass/raw/master/Student/Data/Customers.csv
```

Some students find it painful to copy and paste URLs, M code and DAX expressions from a PDF file such as this. Note that is a text file in the folder for this lab named **Data_Urls_Used_In_This_Lab.txt** if you would rather copy and paste from a text file.

c) Once you have pasted the URL into the **From Web** dialog, click the **OK** button to import the data and create a new query.

d) When prompted by the **Access Web content** dialog, make sure the authentication type is set to **Anonymous**.

e) Drop down the combo box with the caption of **Select which level to apply these settings to**.

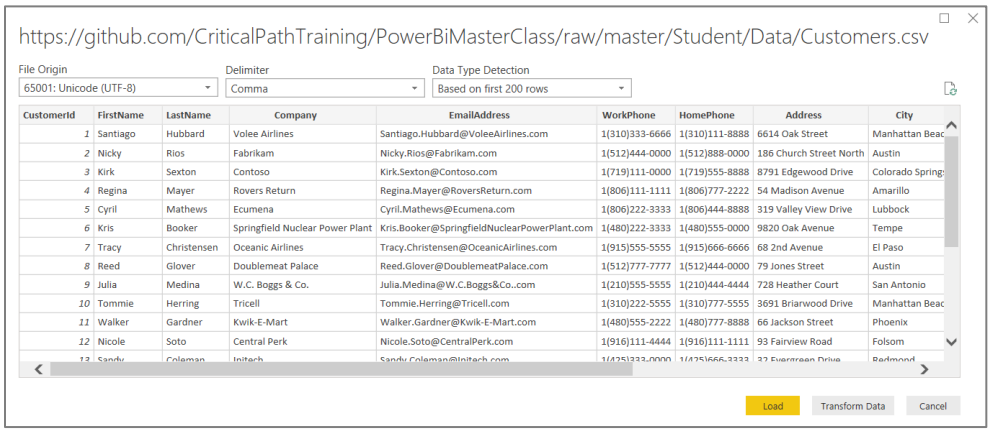


f) Select a level of **https://github.com/CriticalPathTraining/PowerBiMasterClass**.

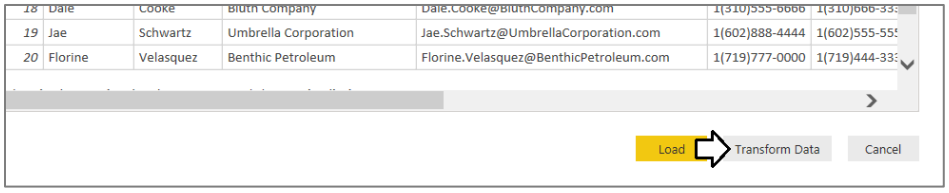
g) Click **Connect**.

At this point you're prompted with the **Customers.csv** dialog with three buttons with the captions **Load, Transform data** and **Cancel**.

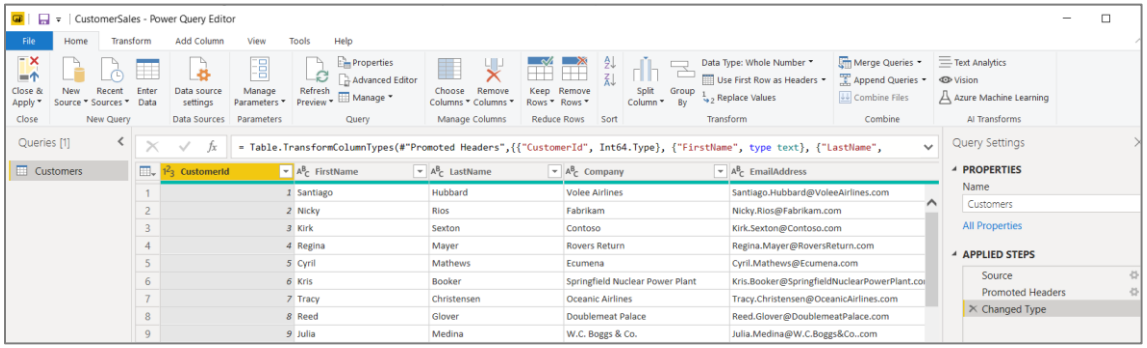
h) Inspect (*but don't change any of the settings in*) the **Customers.csv** dialog.



i) Click the **Transform data** button to view the data from **Customers.csv** in the Power Query Editor window.

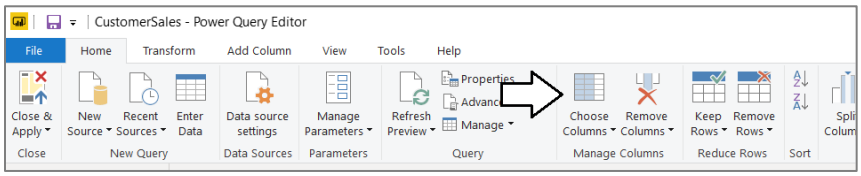


j) You should see data from **Customers.csv** in the Power Query Editor window as shown in the following screenshot.

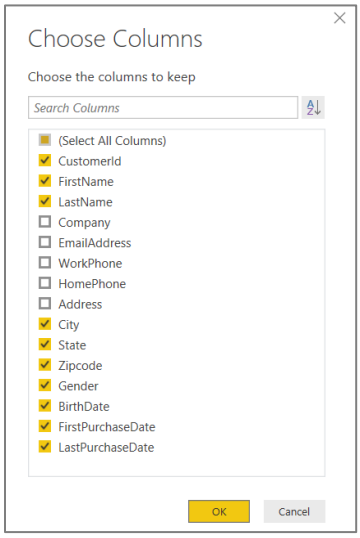


If you examine the **Query Settings** task pane on the right-hand side, you can see that the query is named **Customers** and there are three steps in the **Applied Steps** list.

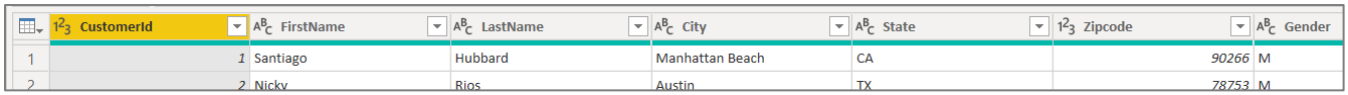
4. Select the desired set of columns from the **Customers** table.
   a) Make sure the **Customers** query is selected in the **Queries** list on the left-hand side of the Power Query Editor window.
   b) Navigate to the **Home** tab and click the **Choose Columns** button in the ribbon to display the **Choose Columns** dialog.



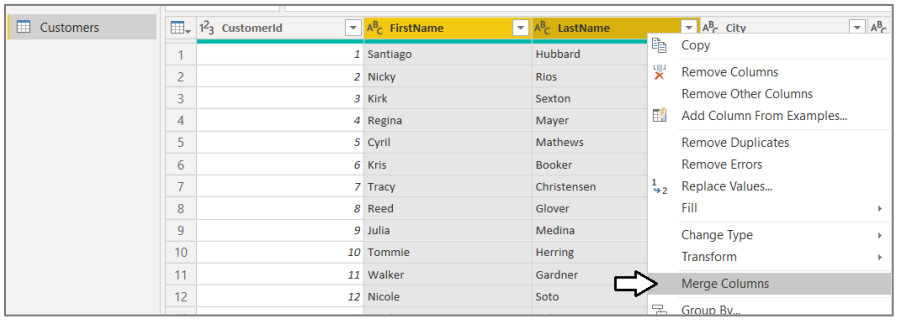
   c) In the **Choose Columns** dialog, begin by clicking on the **(Select all Columns)** checkbox at the top to unselect all column.
   d) Select **CustomerId**, **FirstName**, **LastName**. **City**, **State**, **Zipcode**, **Gender**. **BirthDate**, **FirstPurchaseDate** and **LastPurchaseDate**
   e) Once the columns you've selected match the following screenshot, click the **OK** button to close the **Choose Columns** dialog.



   f) You should be able to see that the Power Query Editor window now only shows the columns that you selected.

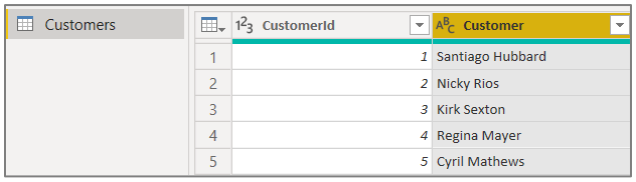


5. In this step you will merge the **FirstName** column and the **LastName** column together into a single column named **Customer**.
   a) Select the **FirstName** column by clicking on its column header.
   b) Next, hold down the **SHIFT** key and select the **LastName** column by clicking on its column header.
   c) Right-click on the selected columns and click the **Merge Columns** menu command.



d) In the **Merge Column** dialog, drop down the **Separator** control and select a value of **Space**. Add a **New column name** value of **Customer** and click the **OK** button to modify the underlying query with your changes.

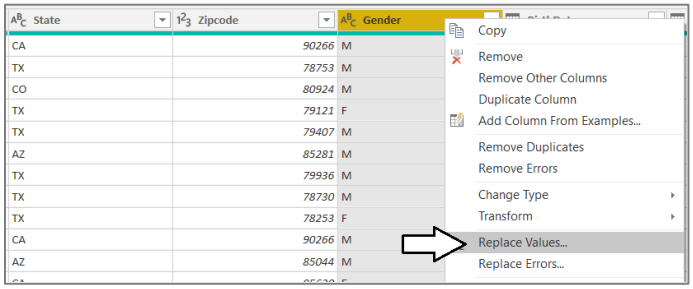
e) You should now be able to see that the **FirstName** column and the **LastName** column have been replaced with a single merged column named **Customer**.



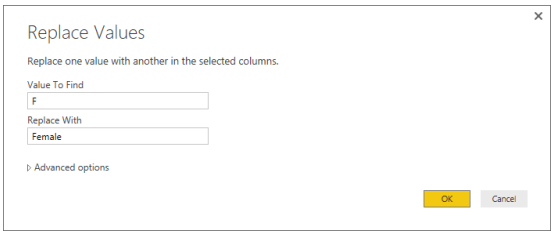
6. Modify the query so that the **Gender** column returns values of **Male** and **Female** instead of **M** and **F**.

a) Locate the **Gender** column in the **Customers** table.

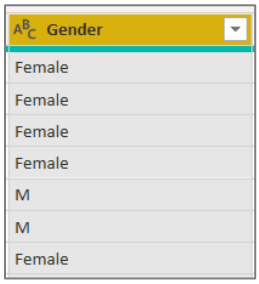
b) Right-click the header for the **Gender** column and select the **Replace Values** command to display the **Replace Values** dialog.



c) In the **Replace Value** dialog, enter a value of **F** in the **Value to Find** textbox and enter a value of **Female** in the **Replace With** textbox. Click to **OK** button add your changes to the underlying query.

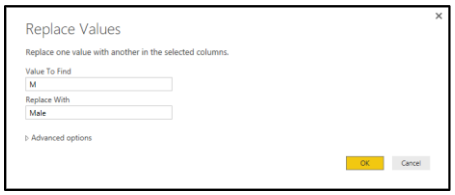


d) You should be able to see that all values of **F** in the **Gender** column have been replaced with a value of **Female**.



e) Right-click the header for the **Gender** column and select the **Replace Values** command a second time.

f) In the **Replace Value** dialog, enter a value of **M** in the **Value to Find** textbox and enter a value of **Male** in the **Replace With** textbox. Click to **OK** button add your changes to the underlying query.

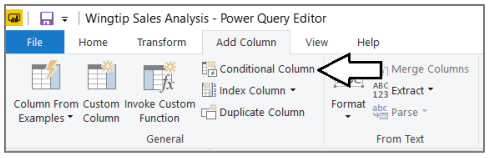


g) You should be able to confirm that all values in the **Gender** column have been replaced with a value of either **Male** or **Female**.

7. Add a new conditional column named **Customer Type** to indicate whether the customer is a repeat customer or not.

a) Activate the **Add Column** tab in the ribbon.

b) Click the **Conditional Column** button in the ribbon to display the **Add Custom Column** dialog.



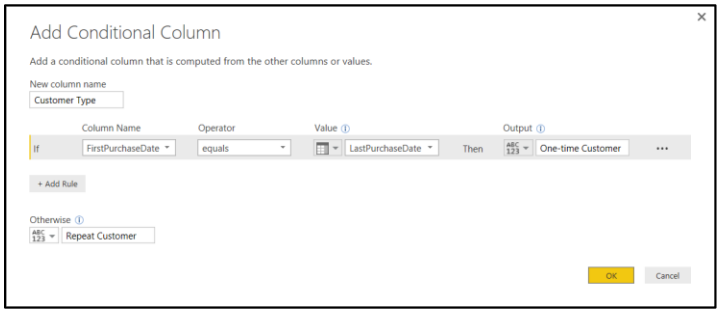
In this particular scenario, you are working under the assumption that the customer is a repeat customer when the **FirstPurchaseDate** column and the **LastPurchaseDate** column are not equal indicating the customer has made two or more purchases.

c) In the **Add Conditional Column** dialog, enter a **New column name** value of **Customer Type**.

d) Configure a rule to return a string value of “One-time Customer” if **FirstPurchaseDate** equals **LastPurchaseDate**.

e) For the **Otherwise** evaluation, return a string value of “Repeat Customer”.

f) When the **Add Conditional Column** dialog matches the screenshot below, click the **OK** button to add the new column.

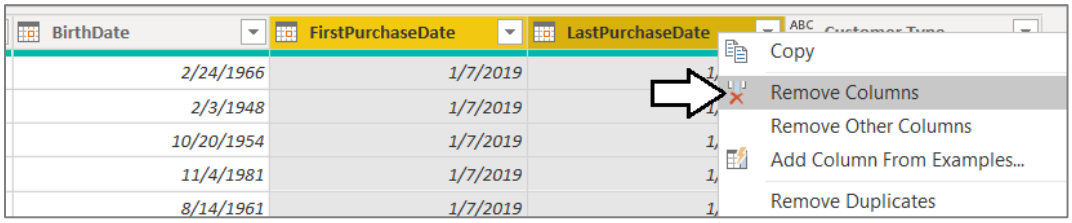


g) You should be able to verify that the new **Customer Type** column has a value of **Repeat Customer** when the current customer has a **FirstPurchaseDate** column value that is not equal to the **LastPurchaseDate** column value. When these column values are equal, the **CustomerType** column has a value of **One-time Customer**.

You might have to scroll down through several records in the **Customers** table before you begin to see **Repeat Customer**.

FirstPurchaseDate	LastPurchaseDate	Customer Type
1/7/2019	1/7/2019	One-time Customer
1/7/2019	1/7/2019	One-time Customer
1/7/2019	1/7/2019	One-time Customer
1/7/2019	1/7/2019	One-time Customer
1/7/2019	1/7/2019	One-time Customer
1/7/2019	1/7/2019	One-time Customer
1/8/2019	11/30/2019	Repeat Customer

8. Now, that you have used the **FirstPurchaseDate** column and the **LastPurchaseDate** column to calculate the value of the **Customer Type** column, you can delete them because they are no longer needed.
   a) Select the **FirstPurchaseDate** column by clicking its column header.
   b) Hold down the **SHIFT** key and click the column header for **LastPurchaseDate** so that both columns are selected.
   c) Right click the one of the selected columns and click the **Remove Columns**.



   d) You should be able to confirm that the **FirstPurchaseDate** column and the **LastPurchaseDate** columns have been removed from the query results. However, the **Customer Type** column is still there.

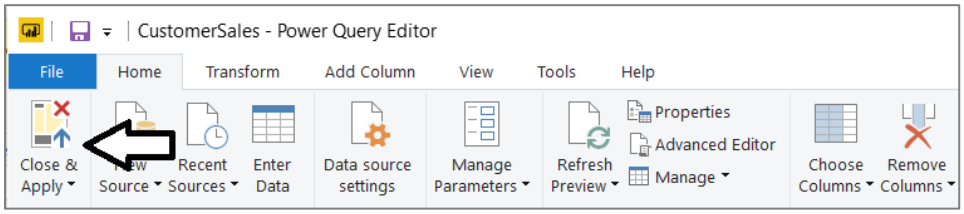
Gender	BirthDate	Customer Type
Male	2/24/1966	One-time Customer
Male	2/3/1948	One-time Customer
Male	10/20/1954	One-time Customer
Female	11/4/1981	One-time Customer
Male	8/14/1961	One-time Customer

   e) You might notice the datatype for the **Customer Type** column is not set to a specific type.

When you see **ABC** above **123**, that the column is being assigned the generic **Any** type which is bad and should be avoided.

   f) Drop down the datatype menu for the **Customer Type** column and set its value to **Text**.

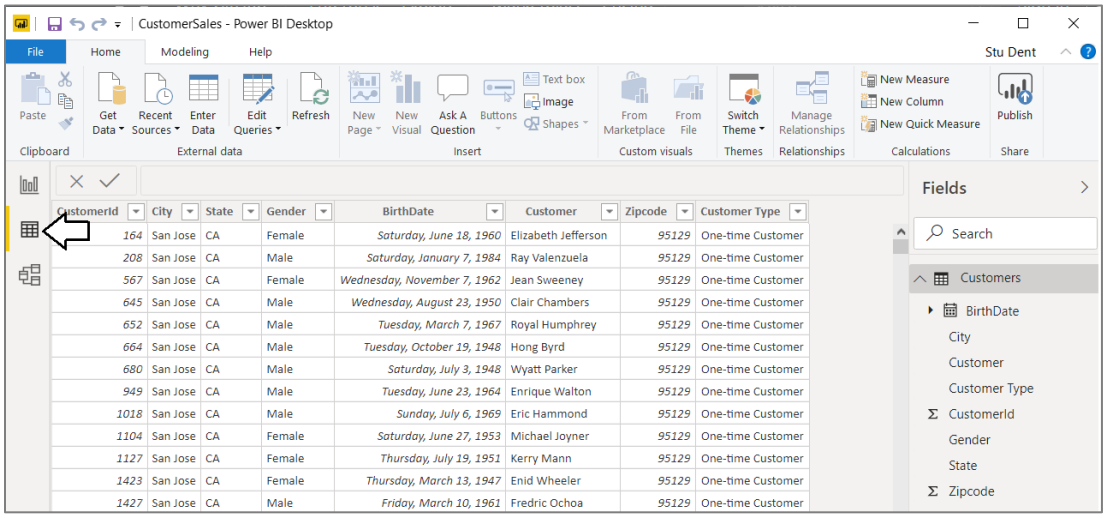
9. Execute the **Customers** query to see its output in the main Power BI Desktop window.
   a) Click the **Close and Apply** button on the **Home** tab of the ribbon to execute the updated **Customers** query.



Power BI Desktop will display the **Apply Query Changes** dialog while importing the data from the CSV file and transforming it.

b) After the **Customer** query changes have been applied, you should be able to see the main Power BI Desktop window.

c) Navigate to **Data** view to see the output of your query which is the **Customers** table.



CustomerId	City	State	Gender	BirthDate	Customer	Zipcode	Customer Type
164	San Jose	CA	Female	Saturday, June 18, 1960	Elizabeth Jefferson	95129	One-time Customer
208	San Jose	CA	Male	Saturday, January 7, 1984	Ray Valenzuela	95129	One-time Customer
567	San Jose	CA	Female	Wednesday, November 7, 1962	Jean Sweeney	95129	One-time Customer
645	San Jose	CA	Male	Wednesday, August 23, 1950	Clair Chambers	95129	One-time Customer
652	San Jose	CA	Male	Tuesday, March 7, 1967	Royal Humphrey	95129	One-time Customer
664	San Jose	CA	Male	Tuesday, October 19, 1948	Hong Byrd	95129	One-time Customer
680	San Jose	CA	Male	Saturday, July 3, 1948	Wyatt Parker	95129	One-time Customer
949	San Jose	CA	Male	Tuesday, June 23, 1964	Enrique Walton	95129	One-time Customer
1018	San Jose	CA	Male	Sunday, July 6, 1969	Eric Hammond	95129	One-time Customer
1104	San Jose	CA	Female	Saturday, June 27, 1953	Michael Joyner	95129	One-time Customer
1127	San Jose	CA	Female	Thursday, July 19, 1951	Kerry Mann	95129	One-time Customer
1423	San Jose	CA	Female	Thursday, March 13, 1947	Enid Wheeler	95129	One-time Customer
1427	San Jose	CA	Male	Friday, March 10, 1961	Fredric Ochoa	95129	One-time Customer

You might observe that the data from the **Customers** table is not sorted in an intuitive fashion. For example, you might expect the default sort order to be the same as the underlying CSV file which orders customers by **CustomerId**. Instead, the table has a seemingly random sort order which is a side effect of the way data is stored in memory given the column-based nature of the tabular database model.

You can right-click the column header for the **CustomerId** column and run a command to sort the data in the table if it makes you feel better, but this action will not really have any lasting effect because changes to the sort order are only temporary and they are not saved as part of the underlying data model. Just remember that sorting in a Power BI report is applied at the visual level.

Exercise 5: Merge Sales Region Columns into the Customers Table

In this exercise, you will merge data from a lookup query named **SalesRegions** to add new columns to the **Customers** table.

1. Import sales region data from as CSV file accessible over the Internet named **SalesRegions.csv**.

   a) Drop down the **Get Data** menu from the **Home** tab of the ribbon and select the **Web** connector.

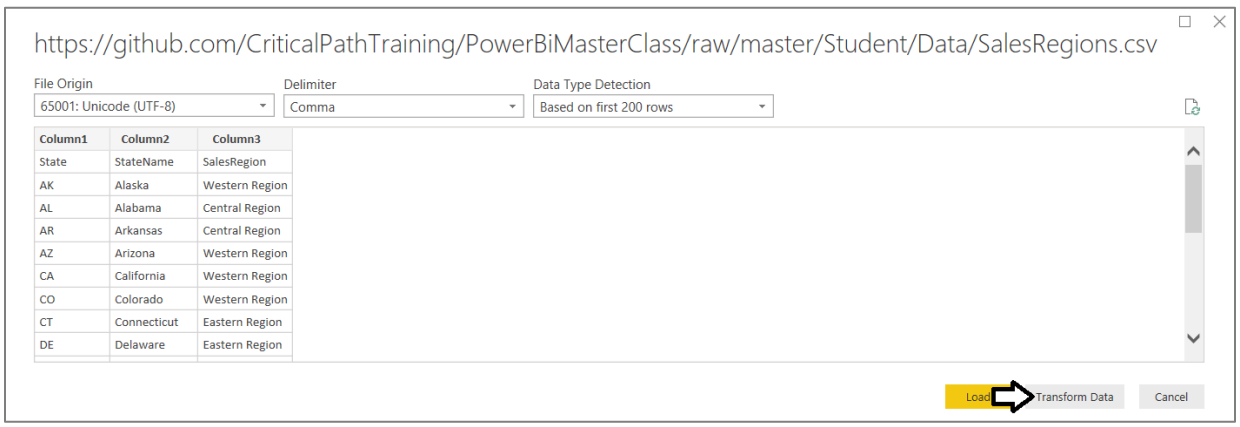
   b) When you are prompted by the **From Web** dialog, copy and paste the following URL to import data from **Customers.csv**.

   https://github.com/CriticalPathTraining/PowerBiMasterClass/raw/master/Student/Data/SalesRegions.csv

   c) Once you have pasted the URL into the **From Web** dialog, click the **OK** button to import the data and create a new query.

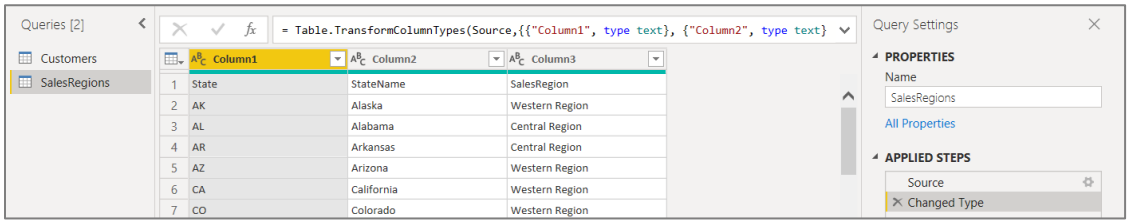
   d) Inspect (*but don't change any of the settings in*) the **SalesRegions.csv** dialog.

   e) Click the **Transform data** button to view the data from **SalesRegions.csv** in the Power Query Editor window.



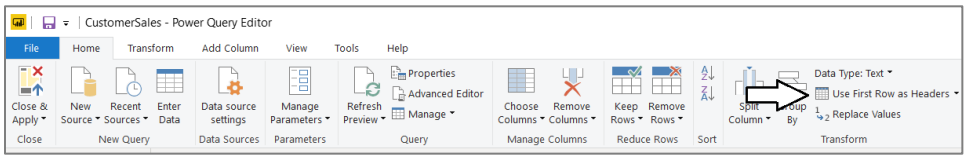
Column1	Column2	Column3
State	StateName	SalesRegion
AK	Alaska	Western Region
AL	Alabama	Central Region
AR	Arkansas	Central Region
AZ	Arizona	Western Region
CA	California	Western Region
CO	Colorado	Western Region
CT	Connecticut	Eastern Region
DE	Delaware	Eastern Region

f) You should now be looking at data imported from **SalesRegions.csv** in the Power Query Editor window.

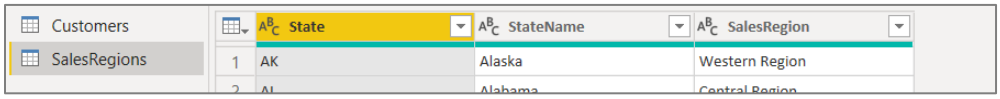


When you imported the data from **Customers.csv**, Power Query was able to automatically determine that the first row contained column header names. However, Power Query will not automatically determine that the first row contains column header names for the data files such as **SalesRegion.csv** because it only has text values, but no values based on numbers or dates.

g) Click on the **User First Row as Headers** button in the **Home** tab of ribbon to establish the correct column headers.



h) There should now be three columns returned by the **SalesRegions** query named **State**, **StateName** and **SalesRegion**.

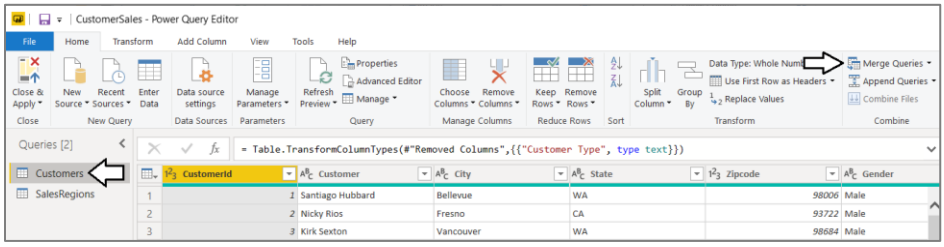


Now that you have created the **SalesRegions** query, you will merge its columns in the **Customers** query.

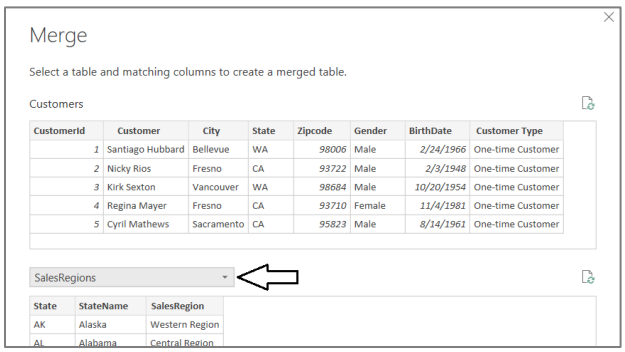
2. Merge columns values from the **SalesRegions** query for state name and sales region into the **Customers** query.

a) Select the **Customers** query in the **Queries** list on the left.

b) From the **Home** tab of the ribbon, click the **Merge Queries** button to display the **Merge** dialog.

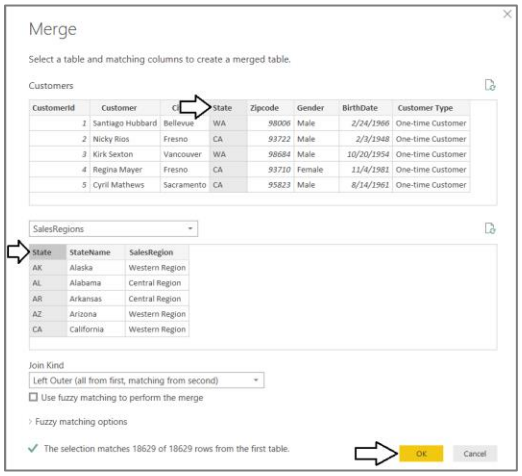


c) When you are prompted by the **Merge** dialog, use the dropdown list under **Customers** to select the **SalesRegions** query.



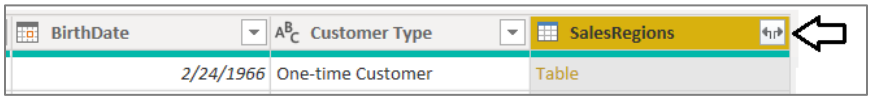
d) Select the **State** column from the **Customers** query and the **State** column from **SalesRegions** to configure a merge key.

e) Once the **Merge** dialog matches the following screenshot, click the **OK** button to complete the merge operation.



Now that you've merged the **SalesRegions** query into **Customers**, you will next add an Expand step to display the desired columns.

f) Click the **Expand** button on the right-hand side of the **SalesRegion** column to expand columns from the **SalesRegion** query.

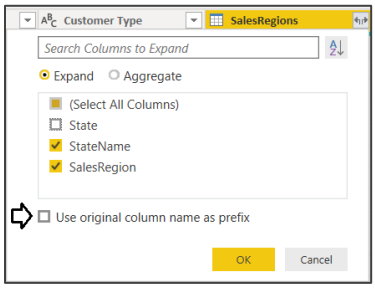


g) Click **(Select All Columns)** to unselect all columns.

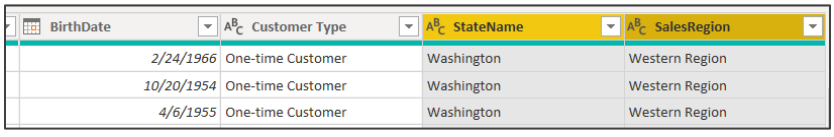
h) Select the **StateName** column and the **SalesRegion** column.

i) Uncheck the **Use original column name as prefix** checkbox.

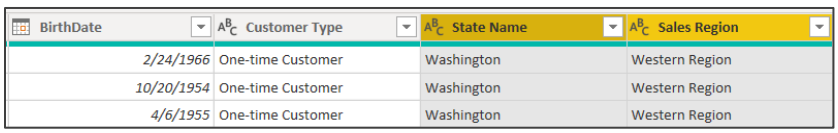
j) Click **OK** to complete the Expand operation.



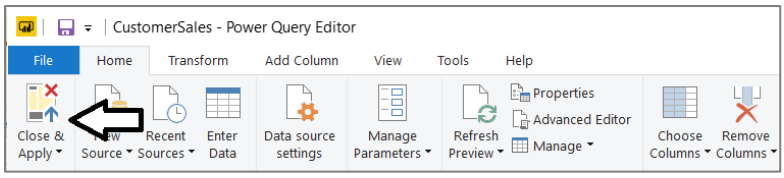
k) The **Customers** query output should now display two new columns named **StateName** and **SalesRegion**.



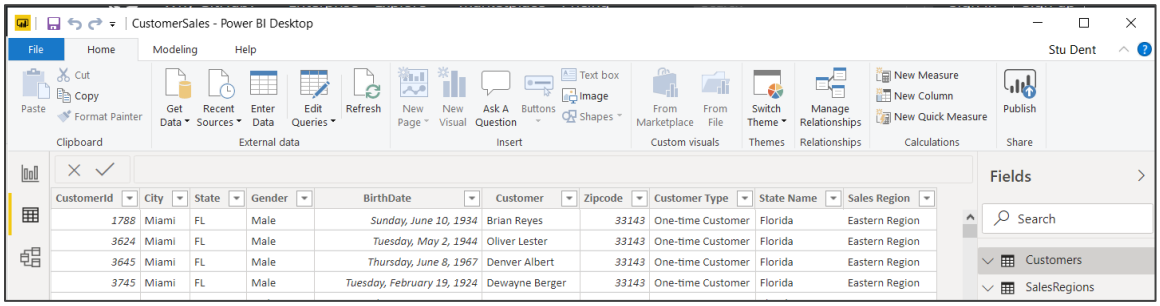
l) Rename the **StateName** and **SalesRegion** columns to **State Name** and **Sales Region**.



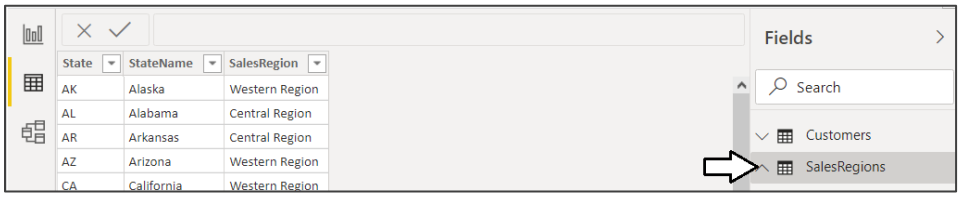
3. Execute the **Close and Apply** command on the Power Query window to see the query results in the Power BI Desktop window.
   a) Click the **Close and Apply** button on the **Home** tab of the ribbon to execute the updated queries.



   b) After the query executes, you should be able to see two tables in the main Power BI Desktop window.
   c) Navigate to **Data** view and select the **Customers** table from the **Fields** list.
   d) Verify you can see the two new columns in the **Customers** table named **State Name** and **Sales Region**.

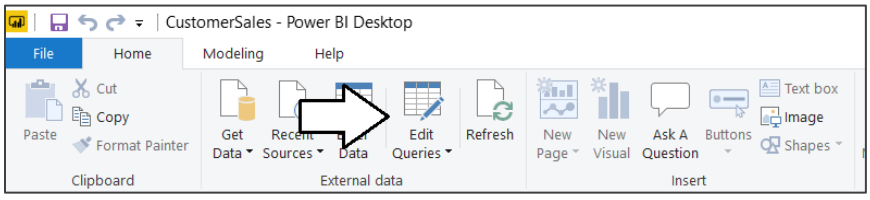


   e) You should also be able to see that a new table named **SalesRegions** has been created from the **SalesRegions** query.

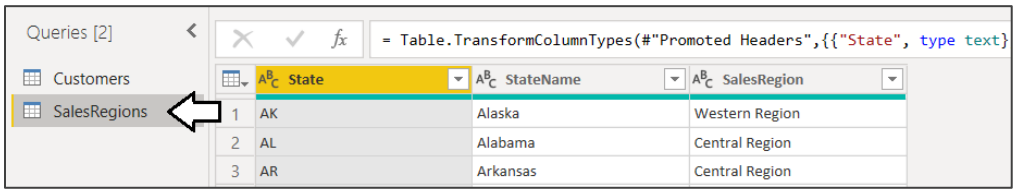


A key point to observe is that the **SalesRegions** table is not needed in the data model because its column values for **State Name** and **Sales Region** have already been merged into the **Customers** table. Next, you will disable loading for the **SalesRegions** query to make the data model smaller and less complicated**.**

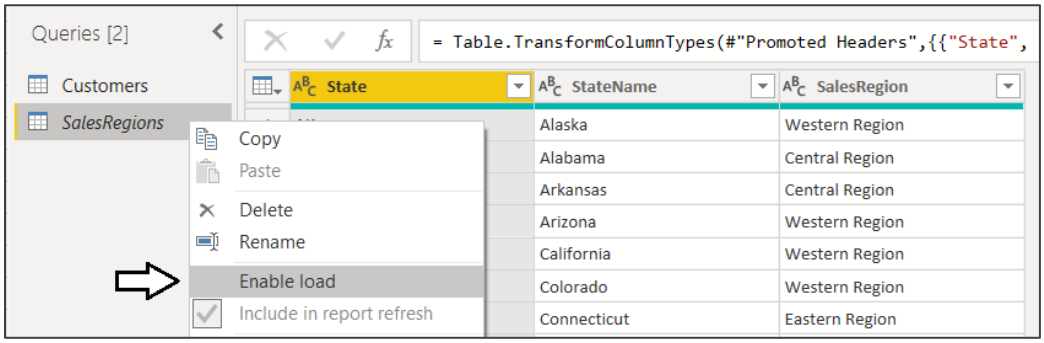
4. Disable loading for the **SalesRegions** query does not generate a table in the data model.
   a) Click the **Edit Queries** button to open the Power Query Editor window.



   b) Right-click on the **SalesRegions** query in the **Queries** list on the left to dropdown the query context menu.



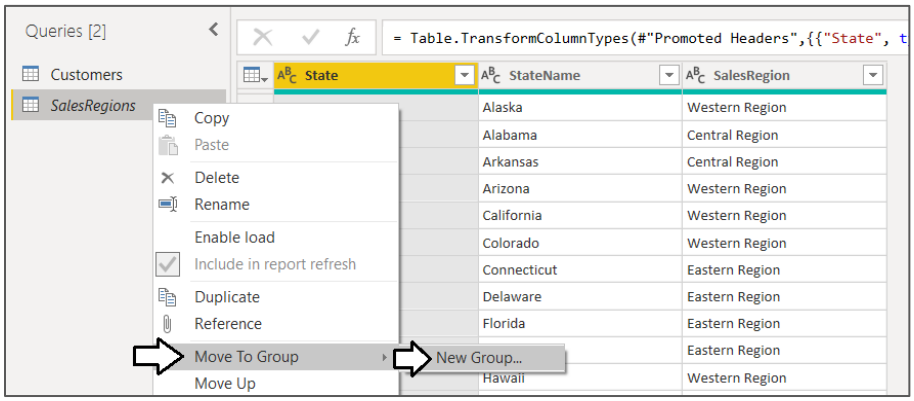
c) From the query context menu for the **SalesRegion** query, click the **Enable load** menu option to unselect this option.



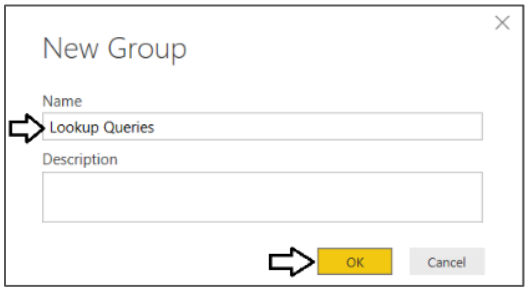
A query in the **Queries** list will be displayed in *italic font* when the **Enable load** option has been disabled.

5. Structure the queries in the **CustomerSales** project by creating two new folder groups named **Lookup Queries** and **Data Model**.

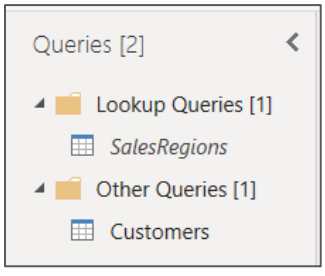
a) Right-click on the **SalesRegions** query and select the **Move To Group > New Group…** menu command.



b) In the **New Group** dialog, enter a new group **Name** of **Lookup Queries** and click **OK**.



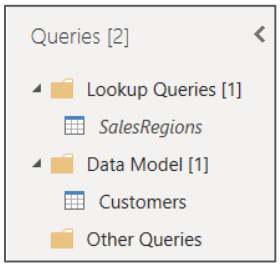
c) The **SalesRegions** query should now exist within a folder group named **Lookup Queries**.



d) Right-click on the **Customers** query and select the **Move To Group > New Group…** menu command.

e) Add a new group **Name** of **Data Model** and click **OK**.

f) The **Customers** query should now exist within a folder group named **Data Model**.

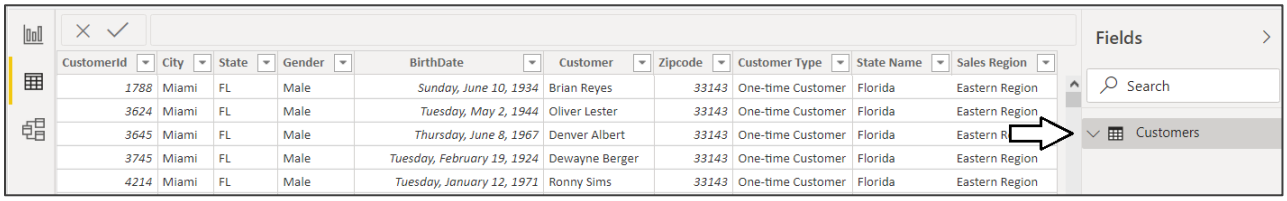


Once you add one or more group folders, Power BI Desktop automatically creates a new group folder named **Other Queries**. Don't spend time trying to delete the **Other Queries** group folder because Power BI Desktop doesn't make this possible.

6. Process the changes to your queries to see the results in the main Power BI Desktop window.

a) Click the **Close and Apply** button on the **Home** tab of the ribbon to process your changes to the **SalesRegions** query.

b) Verify you can still see the **Customers** table but that the **SalesRegions** table is no longer displayed.



Exercise 6: Appending the Output of Multiple Queries to Create the Sales Table

In this exercise, you will create a set of queries to extract data from a set of CSV files. After that, you will create another query which appends the data from multiple queries into a single output table named **Sales**.

1. Import invoice data using the **Web** connector from a CSV file named **Invoices_2019Q1.csv**.

a) Drop down the **Get Data** menu from the **Home** tab of the ribbon and select the **Web** connector.

b) When you are prompted by the **From Web** dialog, copy and paste the following URL to import data from **Customers.csv**.

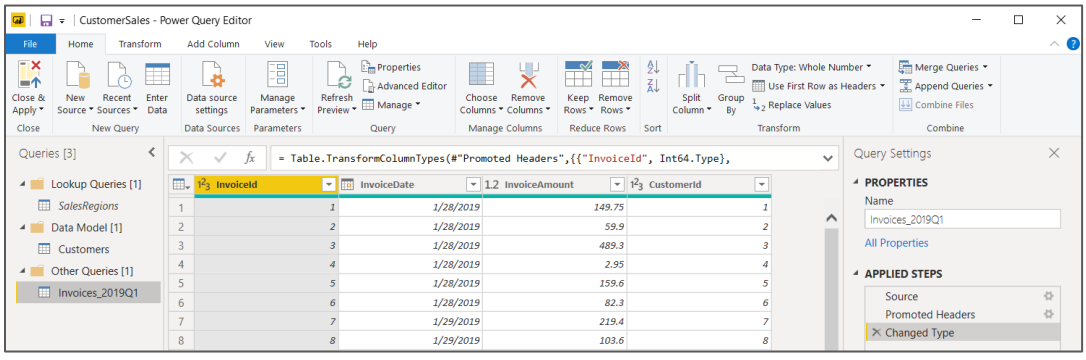
https://github.com/CriticalPathTraining/PowerBiMasterClass/raw/master/Student/Data/Invoices_2019Q1.csv

c) Once you have pasted the URL into the **From Web** dialog, click the **OK** button to import the data and create a new query.

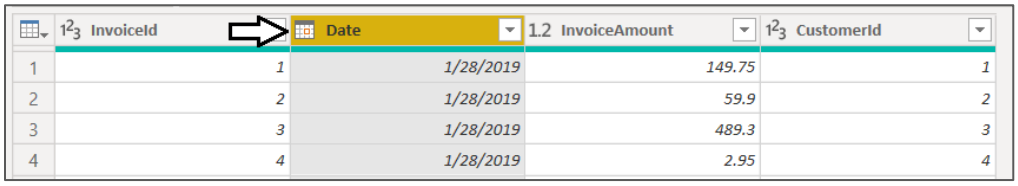
d) Inspect (*but don't change any of)* the settings in the **Invoices_2019Q1.csv** dialog.

e) Click the **Transform data** button to view the data from **Invoices_2019Q1.csv** in the Power Query Editor window.

f) You should now see query results for the **Invoices_2019Q1** query as shown in the following screenshot.

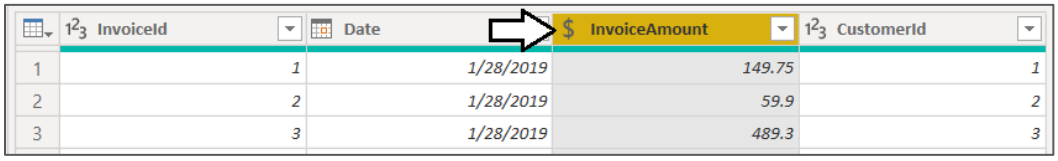


g) Change the name of the **InvoiceDate** column to **Date**.



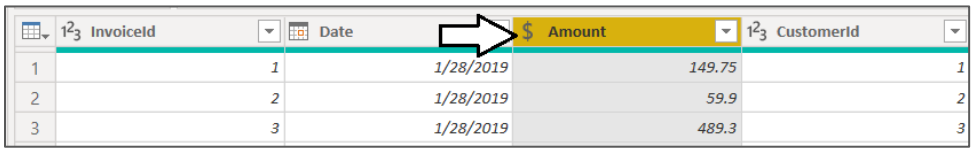
	InvoiceId	Date	InvoiceAmount	CustomerId
1	1	1/28/2019	149.75	1
2	2	1/28/2019	59.9	2
3	3	1/28/2019	489.3	3
4	4	1/28/2019	2.95	4

h) Change the type of the **InvoiceAmount** column to **Fixed decimal number**.



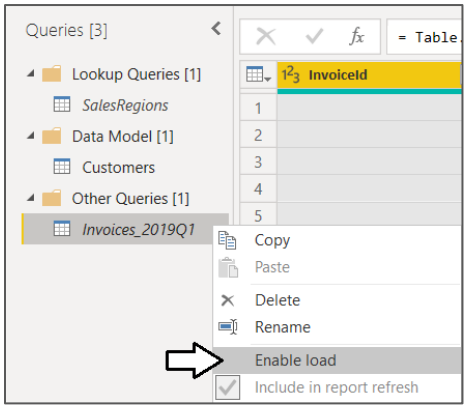
	InvoiceId	Date	InvoiceAmount	CustomerId
1	1	1/28/2019	149.75	1
2	2	1/28/2019	59.9	2
3	3	1/28/2019	489.3	3

i) Change the name of the **InvoiceAmount** column to **Amount**.

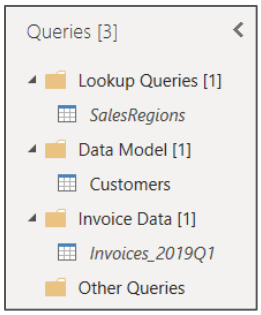


	InvoiceId	Date	Amount	CustomerId
1	1	1/28/2019	149.75	1
2	2	1/28/2019	59.9	2
3	3	1/28/2019	489.3	3

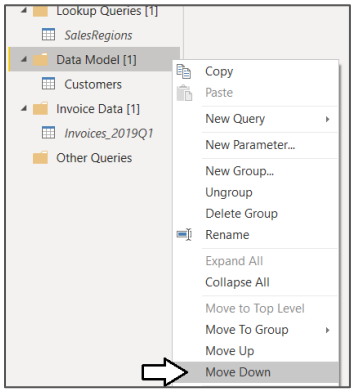
j) Disable to **Enable load** property for the **Invoices_2019Q1** query so it does not generate a table in the project's data model.



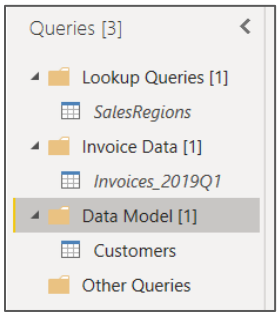
2. Move the **Invoices_2019Q1** query into a new folder group named **Invoice Data**.
   a) Right-click on the **Invoices_2019Q1** query and select the **Move To Group > New Group…** menu command.
   b) In the **New Group** dialog, enter a new group **Name** of **Invoice Data** and click **OK**.
   c) The **Queries** list in your project should now match the following screenshot.



d) Right-click the **Data Model** folder group and select the **Move Down** command to move it below the **Invoice Data** folder group.

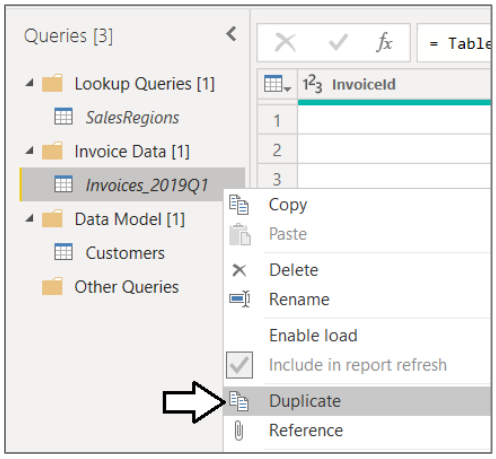


e) The **Data Model** folder group should now be located at the bottom under the other two folder groups.

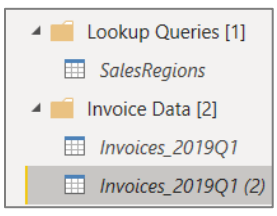


3. Create three new queries by duplicating the **Invoices_2019Q1** query.

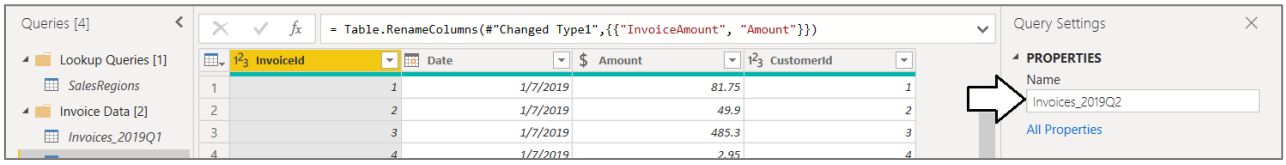
a) Right-click on the **Invoices_2019Q1** query and select the **Duplicate** menu command.



b) You should now have an identical query with a name of **Invoices_2019Q1 (2)**.

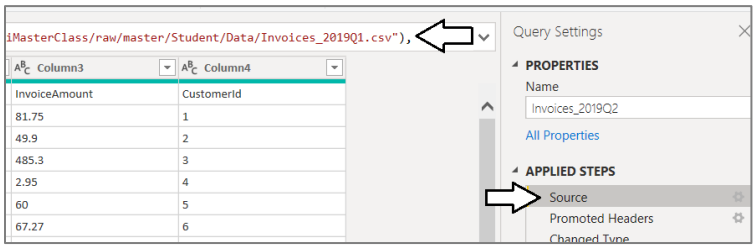


c) Update the name of the new query to **Invoices_2019Q2**.



d) In the **APPLIED STEPS** list for the **Invoices_2019Q2** query, select the top step named **Source**.

e) Inspect the M code in the Formula Bar and locate the file name **Invoices_2019Q1.csv**.



f) Using the Formula Bar, update the file name from **Invoices_2019Q1.csv** to **Invoices_2019Q2.csv**.

Now you'll create queries for **Invoices_2019Q3.csv** and **Invoices_2019Q4.csv** just like you did for **Invoices_2019Q2.csv**.

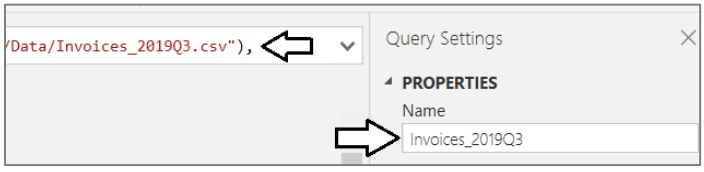
g) Right-click on the **Invoices_2019Q1** query and select the **Duplicate** menu command.

h) Rename the new query to **Invoices_2019Q3**.

i) In the **APPLIED STEPS** list for the **Invoices_2019Q3** query, select the step at the top named **Source**.

j) Inspect the M code in the Formula Bar and locate the file name **Invoices_2019Q1.csv**.

k) Using the Formula Bar, change the file from **Invoices_2019Q1.csv** to **Invoices_2019Q3.csv**.



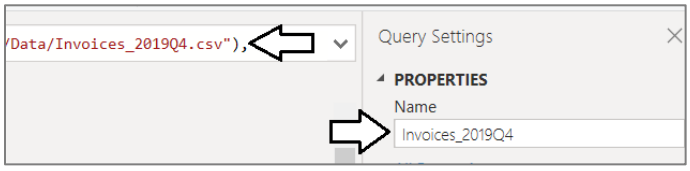
l) Right-click on the **Invoices_2019Q1** query and select the **Duplicate** menu command.

m) Rename the new query to **Invoices_2019Q4**.

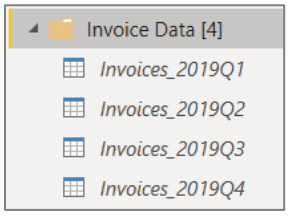
n) In the **APPLIED STEPS** list for the **Invoices_2019Q4** query, select the step at the top named **Source**.

o) Inspect the M code in the Formula Bar and locate the file name **Invoices_2019Q1.csv**.

p) Using the Formula Bar, change the file from **Invoices_2019Q1.csv** to **Invoices_2019Q4.csv**.

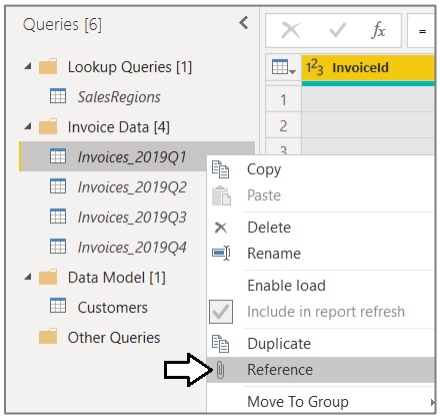


q) The **Invoice Data** folder group should now contain four queries as shown in the following screenshot.



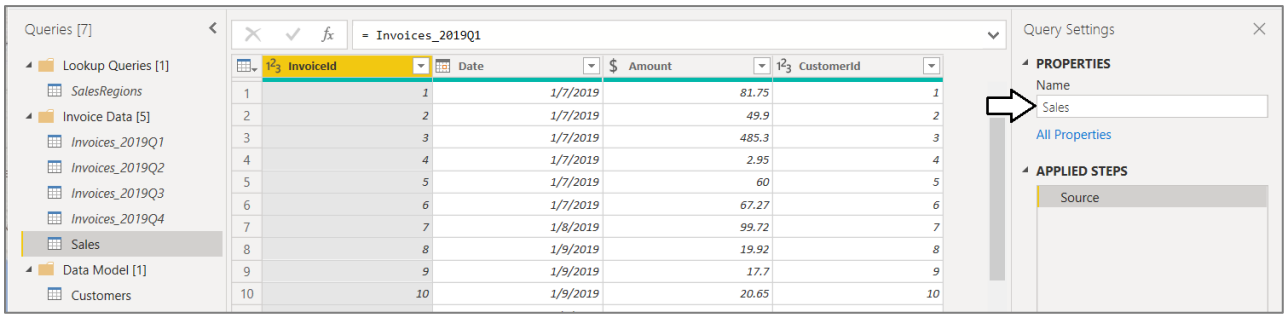
4. Create a new query name **Sales** that appends together all the rows from the 4 queries in the **Invoice Data** folder group.

a) Right-click on the **Invoices_2019Q1** query and select the **Reference** menu command.



When you create a new query in this fashion, the new query will contain a single step named **Source** which references the query on which you executed the **Reference** command. In this case, the input of the new query is the output of the **Invocies_2019Q1** query.

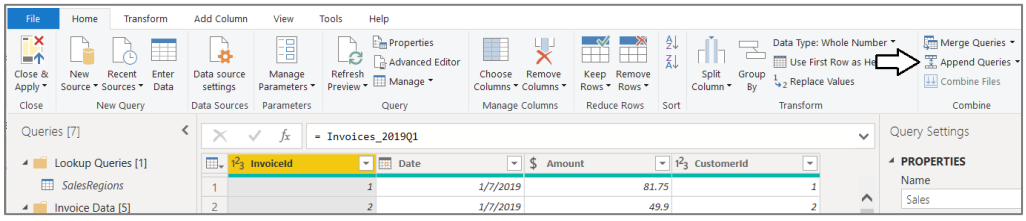
b) Rename the new query to **Sales**.



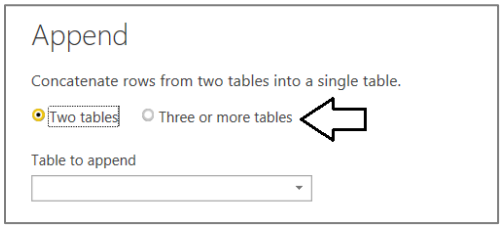
At this point, the **Sales** query only returns rows from the **Invoices_2019Q1** query. In the next step, you will update the **Sales** query so that it appends together all the rows from all four invoice queries in the **Invoice Data** folder group.

c) Make sure the **Sales** query is selected in the **Queries** list on the left.

d) Click the **Append Queries** button on the **Home** tab in the ribbon.



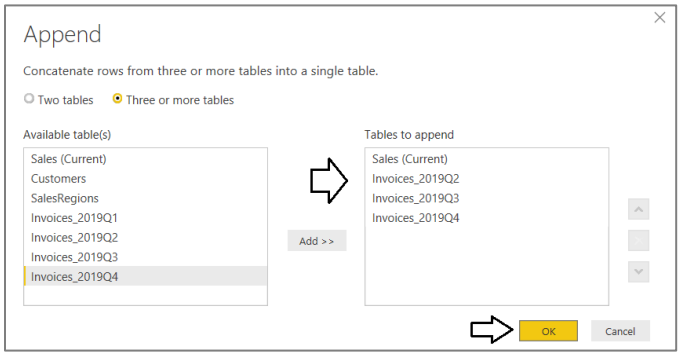
e) When the **Append** dialog appears, select the option for **Three or more tables**.



f) Selecting the **Three or more tables** option will automatically add **Sales** to the **Tables to append** list.

g) Add the queries **Invoices_2019Q2**, **Invoices_2019Q3 and Invoices_2019Q4** to the **Tables to append** list.

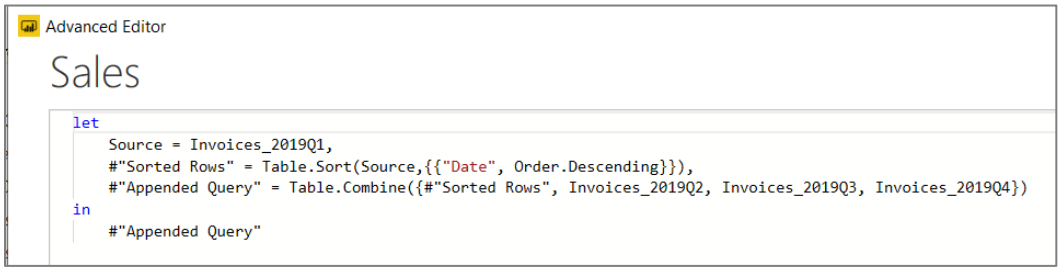
h) Click **OK** to complete the **Append** operation.



At this point, the query has been updated with the logic it needs. In the next step you will inspect the M code behind this query,

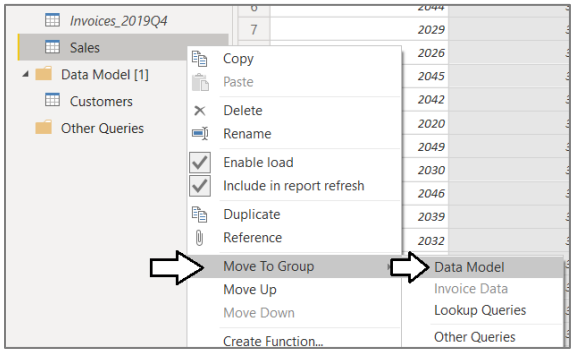
i) With the **Sales** query selected in the **Queries** list on the left, click the **Advanced Editor** button to inspect the query's M code.

j) You should see that the M code for this query is pretty simple.

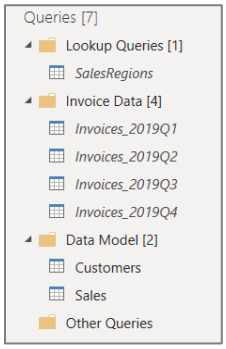


5. Moves the **Sales** query into the **Data Model** folder group.

a) Right-click the **Sales** query and select the **Move To Group > Data Model** menu command.

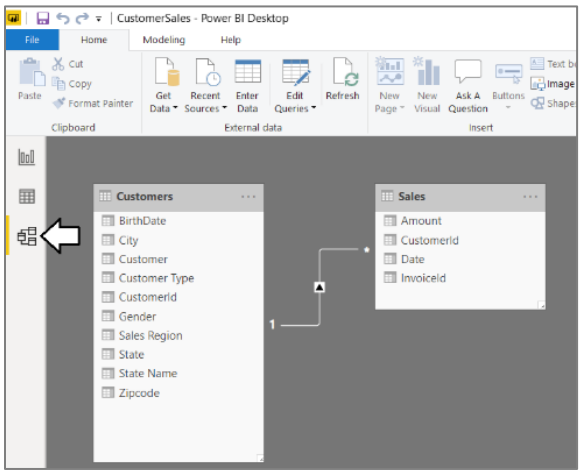


b) The structure of folder groups and queries in your project should now match the following screenshot.

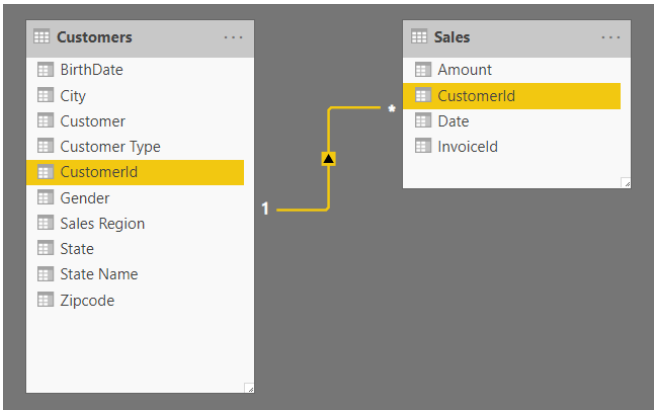


As projects get larger, it can be really helpful to use folder groups to make your projects more maintainable and easier to understand.

6. Run the **Close and Apply** command to apply the query changes you have made.
   a) Click the **Close and Apply** command from the **Home** tab to close the Power Query window and apply the query changes.
   b) Once the query changes have been applied, navigate to **Model** view in the primary Power BI Desktop widow.
   c) You should see that the project's data model now contains the **Customers** table and the **Sales** table.

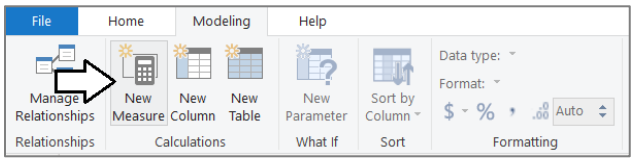


d) Note that Power BI Desktop has automatically create a relationship between the two tables based on the **CustomerId** column.



Power BI Desktop will automatically create a relationship between two tables like this when both tables have a column with the same name of the same type and at least one of the tables has all unique values for that column so that it can serve as a primary key.

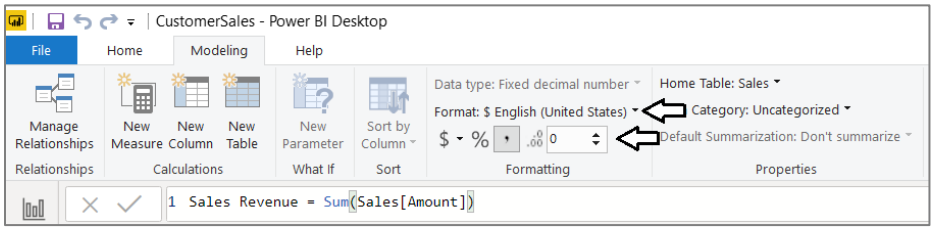
7. Create a measure in the **Sales** table named **Sales Revenue** to perform a sum aggregation on the **Amount** column.
   a) Navigate to **Data** view and select the **Sales** table from the **Fields** list.
   b) Click the **New Measure** button in the **Modeling** tab of the ribbon to add a new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



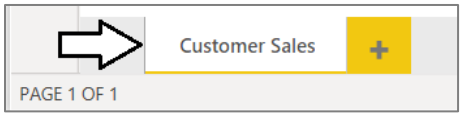
   c) Enter the following DAX expression into the formula bar to create the measure named **Sales Revenue**.

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

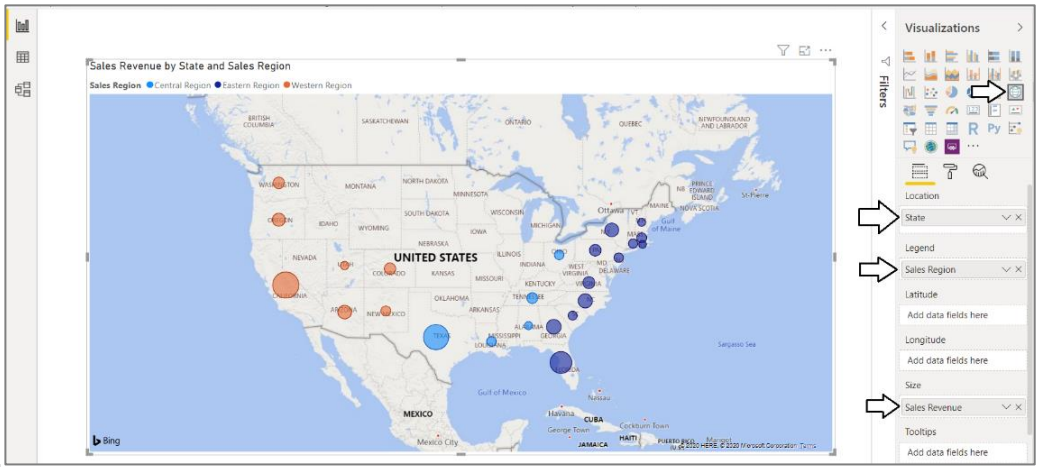
   d) Press the **ENTER** key to add the measure to data model.
   e) Modify the formatting by dropping down the **Format** menu on the ribbon and selecting **Currency > $ English (United States)**.
   f) Use the spinner control below the **Format** menu to set the number of decimal places shown to zero.



8. Add a Map visual to the project's report to surface insights from the customer and sales data in the project's data model.
   a) Navigate to Report view.
   b) Change name of page to **Customer Sales**.



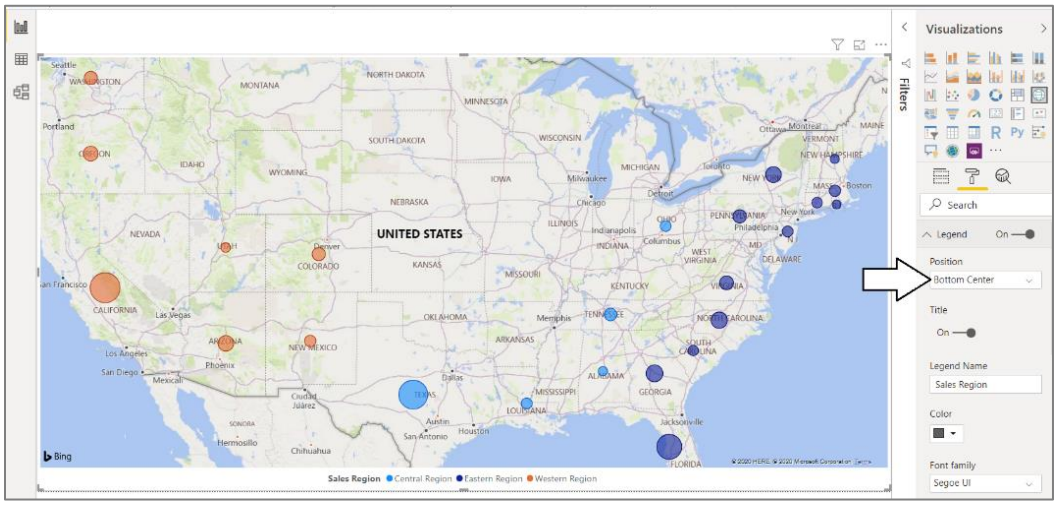
   c) Add a new Map visual to the page
   d) Add the **State** column from the **Customers** table into the **Location** data role.
   e) Add the **Sales Region** column from the **Customers** table into the **Legend** data role.
   f) Add the **Sales Revenue** measure from the **Sales** table into the **Size** data role.



g) With the **Map** visual selected, navigate to the Format pane

h) In the **Legend** section, update the **Position** property to **Bottom Center**.

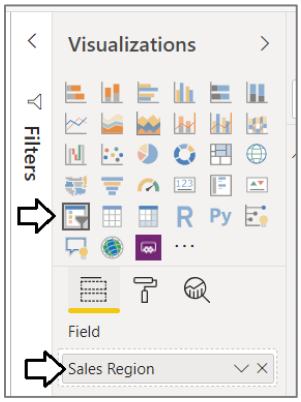
i) Using the mouse, position the Map visual to take up the entire width and height of the report leaving a little room at the top.



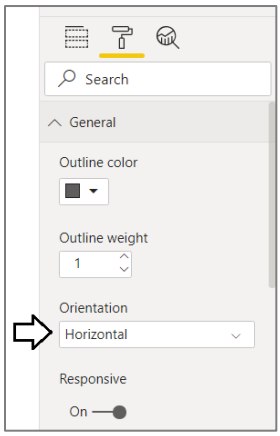
9. Add slicer visual so that report consumers can filter by sales region.

a) Add a slicer visual to the page.

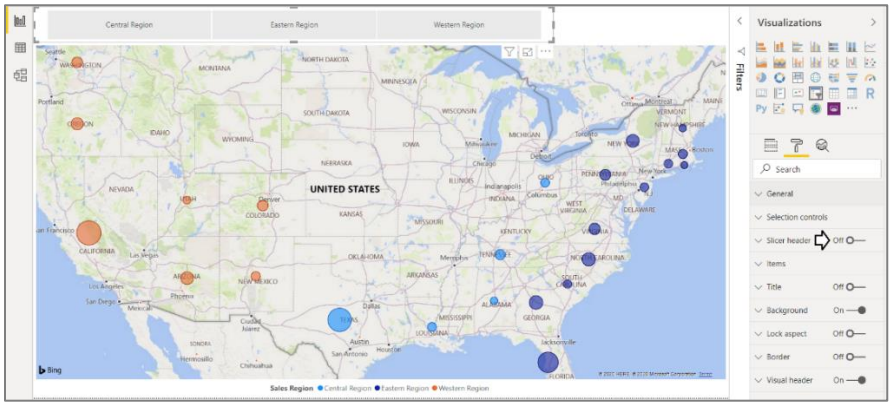
b) Add the **Sales Region** column from the **Customers** table in the **Field** data role for the slicer visual.



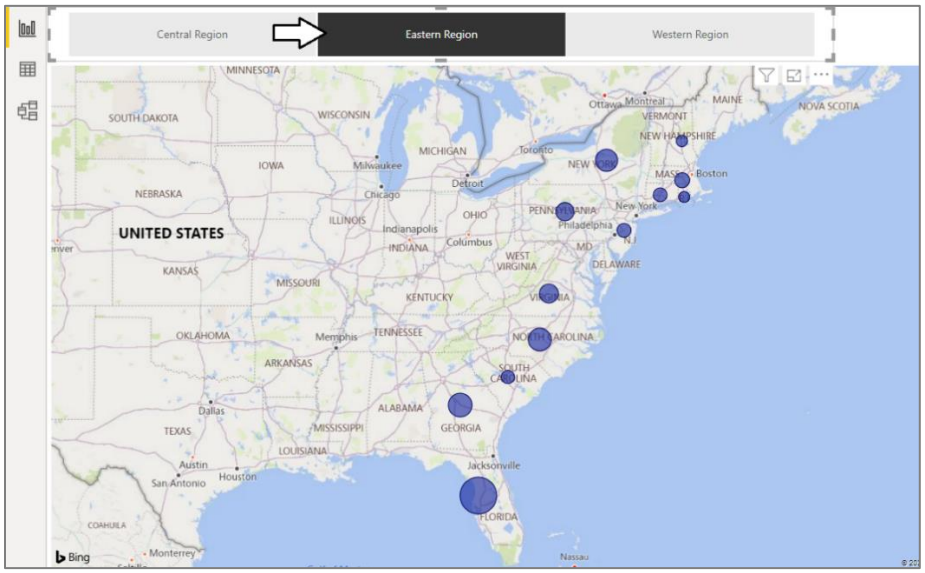
c) Navigate to the **General** section in the Format pane and update the **Orientation** property in the to **Horizontal**.



d) Position the slicer visual at the top of the report as shown in the following screenshot,



e) Test the slider visual and verify it filters the states shown by the sales region selected.

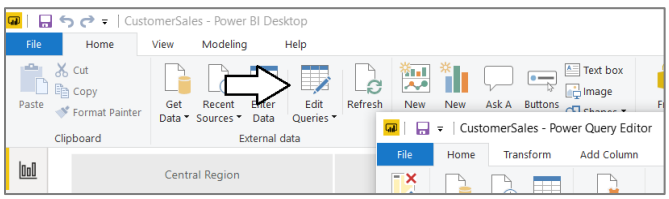


You now integrated sales region data into the project by performing the merge operation between the **Customers** query and the **SalesRegions** query. However, the is still a problem because Power BI is sorting sales regions alphabetically. As you can see, the slicer shows the sequence of regions as **Central Region**, **Eastern Region** and **Western Region**. In the next exercise, you will move through the steps to implement a custom sales region sort order. The first part of the solution will be to create a query which generates data for sorting sales region data.

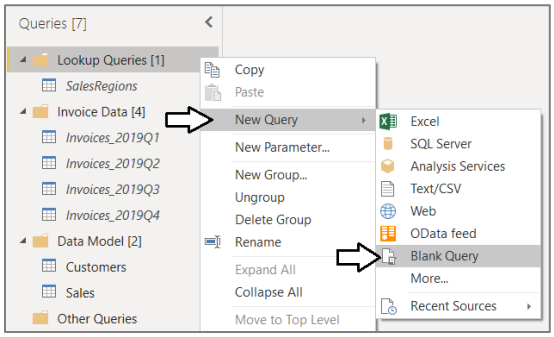
Exercise 7: Creating a Table using M Code to Sort Sales Regions

In this exercise, you will create a query named **SalesRegionSort** using an M expression with hard-coded row values.

1. Create a new query named **SalesRegionSort** with hard-coded rows of data which will be used to implement a custom sort order.
   a) Click the **Edit Queries** button to open the Power Query Editor window.



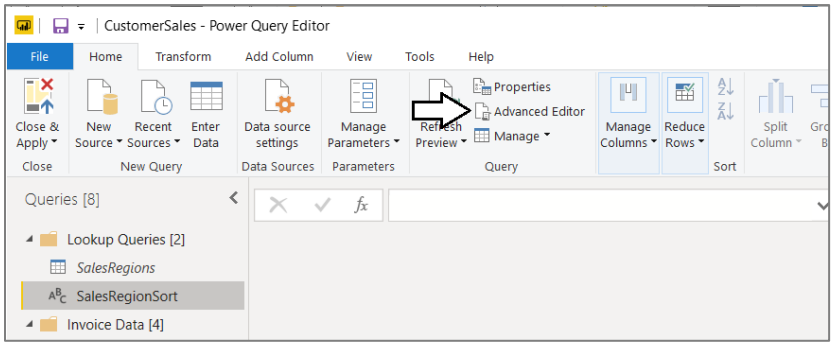
b) Right-click on the **Lookup Queries** folder group and select the **New Query > Blank Query** menu command.



c) You should see that a new query has been created named **Query1**.

d) Change the name of the query from **Query1** to **SalesRegionSort**.

e) With the **SalesRegionSort** query select in the **Queries** list, click the **Advanced Editor** button on the **Home** tab of the ribbon.

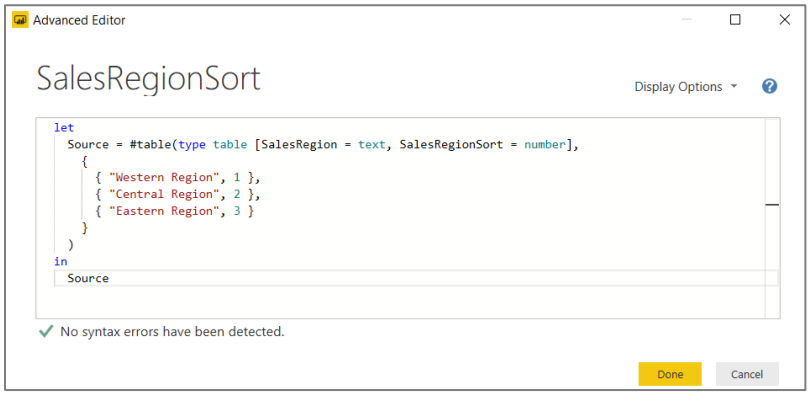


f) Copy and paste the following M code into the Advanced Editor window for the **SalesRegionSort** query.

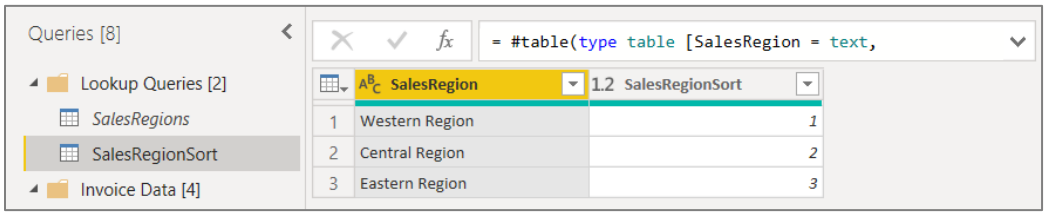
```
let
  Source = #table(type table [SalesRegion = text, SalesRegionSort = number],
    {
      { "Western Region", 1 },
      { "Central Region", 2 },
      { "Eastern Region", 3 }
    }
  )
in
  Source
```

If it's tricky to copy and paste from the PDF file, you can copy this M code from a file in the **Lab** folder named **SalesRegionSort.m.txt**.

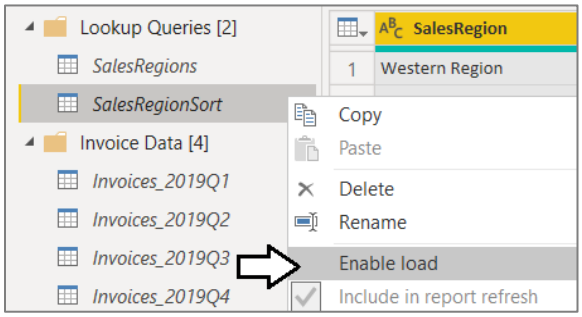
g) Once you have copied the M code into the **Advanced Editor** window, click the **OK** button to save your changes.



h) The **SalesRegionSort** query should return a table with three rows as shown in the following screenshot.

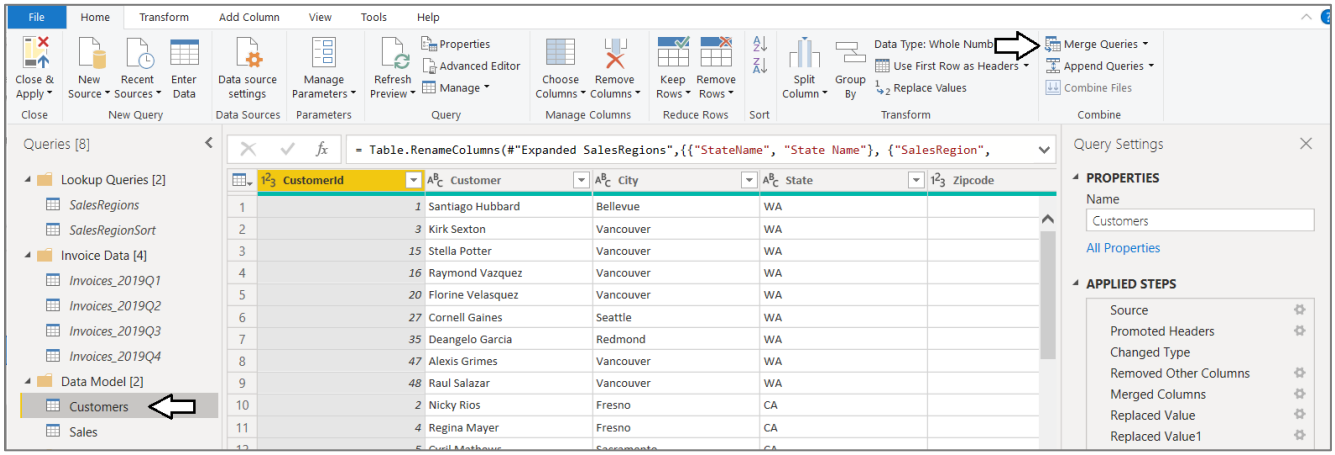


i) Disable the **Enable load** property for the **SalesRegionSort** query so it does not generate a table in the data model.



2. Merge the **SalesRegionSort** column with the **Customers** query.

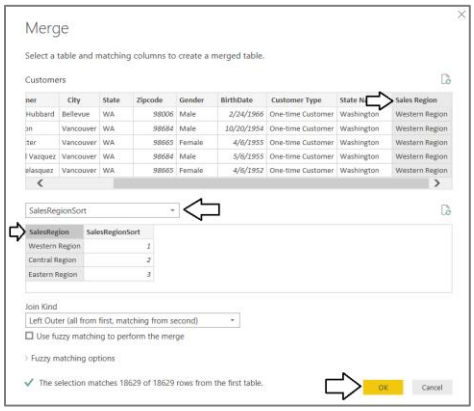
a) Select the **Customers** query in the **Queries** list and then click the **Merge Queries** button on the **Home** tab of the ribbon.



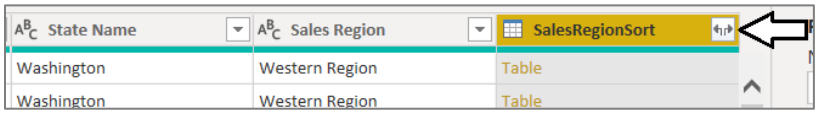
b) In the **Merge** dialog, select the **SalesRegionSort** query from the dropdown list.

c) Select **Sales Region** column in the **Customers** query and the **SalesRegion** column in the **SalesRegionSort** query.

d) Click **OK** to complete the merge operation.



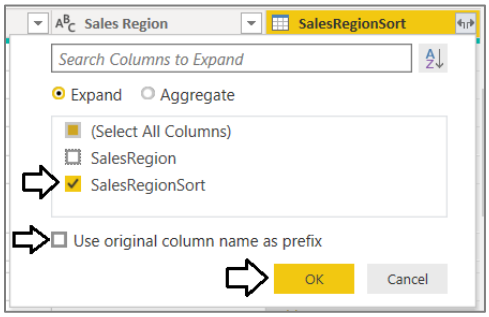
e) Click the Expand button on right right-hand side of the **SalesRegionSort** column header.



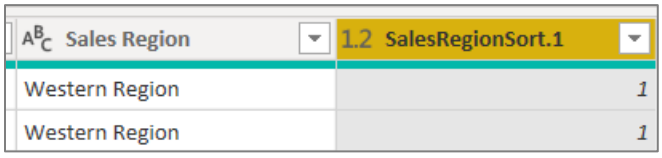
f) In the Expand dropdown menu, select the column named **SalesRegionSort** and unselect the **SalesRegion** column.

g) Uncheck the **Use original column name as prefix** checkbox.

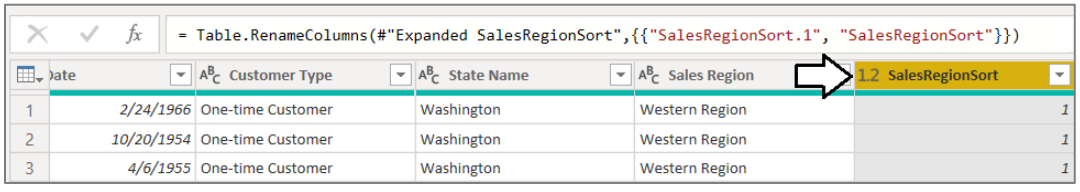
h) Click **OK**.



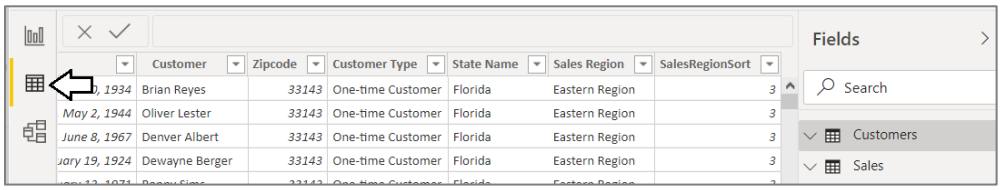
i) You should now see a new column named **SalesRegionSort.1**.



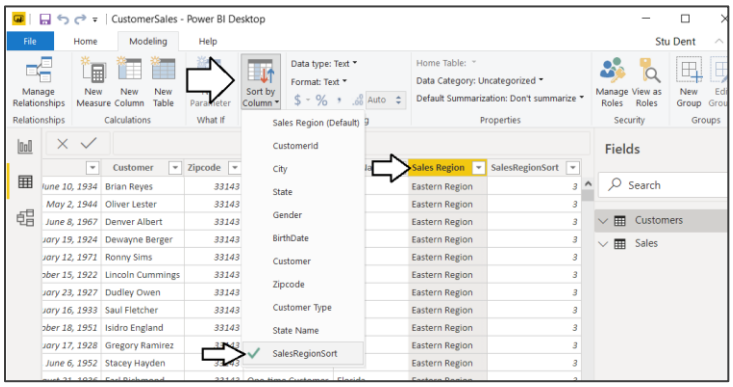
j) Rename the **SalesRegionSort.1** column to **SalesRegionSort**.



3. Run the **Close and Apply** command to see the query changes in the main Power BI Desktop window.
   a) Click the **Close and Apply** button in the ribbon to close the Power Query Editor window and apply query changes.
   b) Navigate to **Data** view and verify you can see the **SalesRegionSort** column in the **Customers** table.



   c) Click on the header of the **Sales Region** column to select it.
   d) With the **Sales Region** column selected, activate the **Modeling** tab in the ribbon.
   e) Drop down the **Sort by Column** menu and set **SalesRegionSort** as the sorting column for the **Sales Region** column.

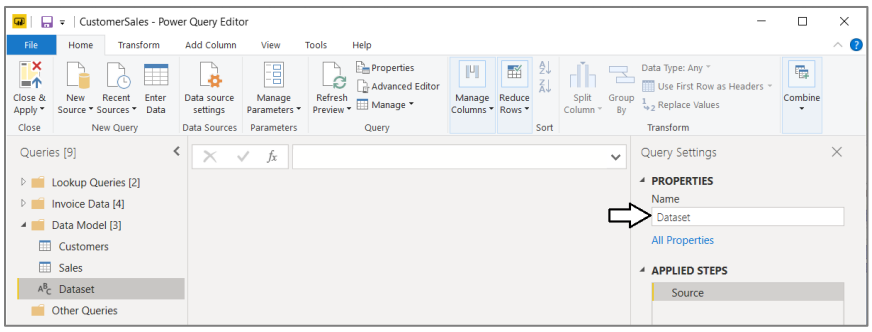


4. See the effects of changing the sort order of the **Sales Region** column.
   a) Navigate to Report view and verify the slicer displays regions as **Western Region**, **Central Region** and **Eastern Region**.

Exercise 8: Creating a Table with M Code to Track the Last Dataset Update

In this exercise, you will create a new query using M code to record the last refresh time for the current dataset.

1. Create a new query named **Datsaet**.
   a) Click the **Edit Queries** button to open the Power Query Editor window.
   b) Right-click on the **Data Model** folder group and select the **New Query > Blank Query** menu command.
   c) Rename the new query to **Dataset**.

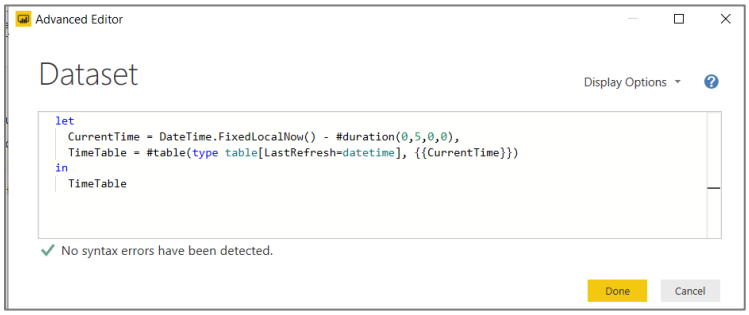


d) Copy and paste the following M code into the Advanced Editor window for the **Dataset** query.

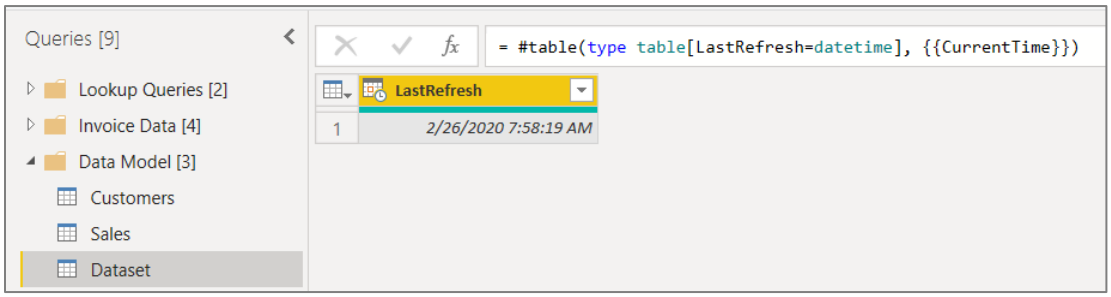
```
let
  CurrentTime = DateTime.FixedLocalNow() - #duration(0,5,0,0),
  TimeTable = #table(type table[LastRefresh=datetime], {{CurrentTime}})
in
  TimeTable
```

If it's tricky to copy and paste from the PDF file, you can copy this M code from a file in the **Lab** folder named **Dataset.m.txt**.

e) Once you have copied the M code into the **Advanced Editor** window, click the **OK** button to save your changes.



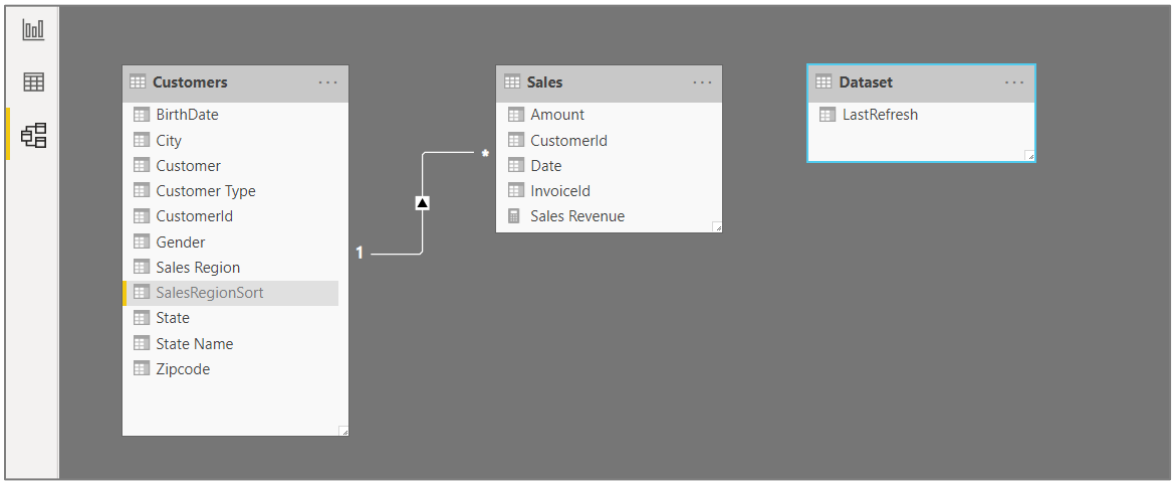
f) The **Datset** query should return a table with one row and one column as shown in the following screenshot.



2. See your changes to the project's data model.

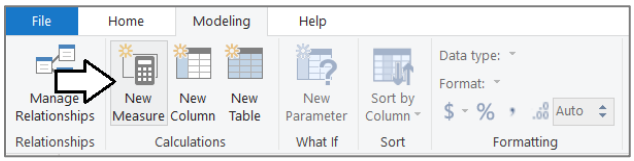
a) Click the **Close and Apply** button in the ribbon to close the Power Query Editor window and apply query changes.

b) Navigate to **Model** view and verify you can see the **Dataset** table.



Note that there is no need to establish a relationship between the **Dataset** table and any other table in the data model.

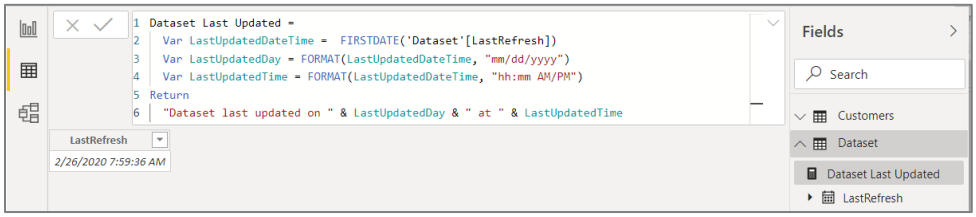
3. Create a measure in the **Dataset** table named **Dataset Last Updated** to display the date and time of the last dataset refresh.
   a) Navigate to **Data** view and select the **Dataset** table from the **Fields** list.
   b) Click the **New Measure** button in the **Modeling** tab of the ribbon to add a new measure to the **Dataset** table.



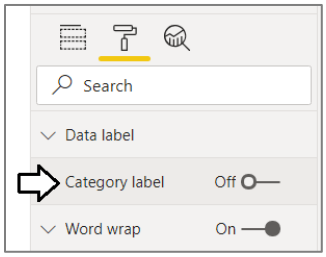
   c) Enter the following DAX expression into the formula bar to create the measure named **Dataset Last Updated**.

```
Dataset Last Updated =
  Var LastUpdatedDateTime = FIRSTDATE('Dataset'[LastRefresh])
  Var LastUpdatedDay = FORMAT(LastUpdatedDateTime, "mm/dd/yyyy")
  Var LastUpdatedTime = FORMAT(LastUpdatedDateTime, "hh:mm AM/PM")
Return
  "Dataset last updated on " & LastUpdatedDay & " at " & LastUpdatedTime
```

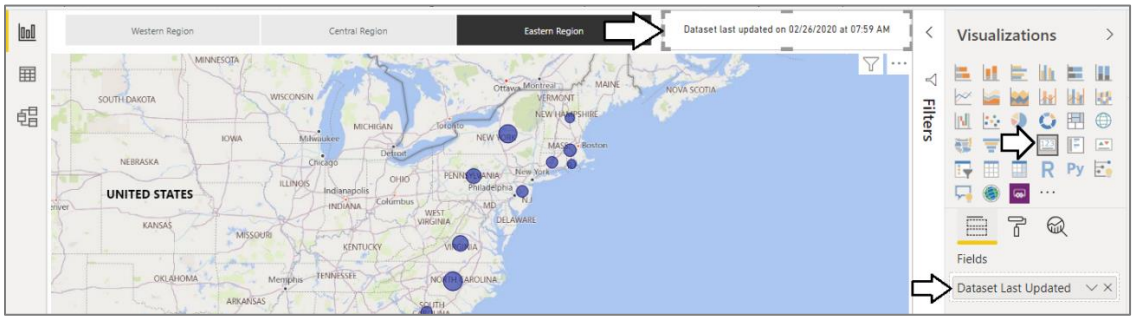
   d) Once you have entered the DAX expression, press the **ENTER** key to add the measure to the data model.



4. Add a Card visual to the report to display the date and time of the last dataset refresh.
   a) Navigate to **Report** view and add a new **Card** visual to the top right corner of the report.
   b) Add the **Dataset Last Updated**. measure to the Fields data role of the **Card** visual.
   c) Change the font size to **12**.
   d) Turn off the **Category label** property for the Card visual.



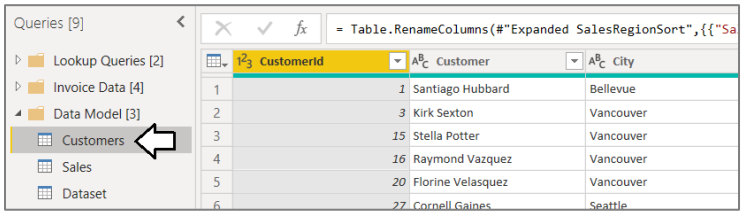
   e) The text displayed in the Card visual should now match the following screenshot.



Exercise 9: Creating a Custom Column to Calculate Customer Age

This is the first ***EXTRA CREDIT*** lab exercise in which you will create a custom column with M code to calculate customer age.

1. Create a new custom column named **Age**.
   a) Click the **Edit Queries** button to open the Power Query Editor window.
   b) Navigate to the **Customers** query.
   c) Navigate to the **Add Column** tab in the ribbon and click **Custom Column**.

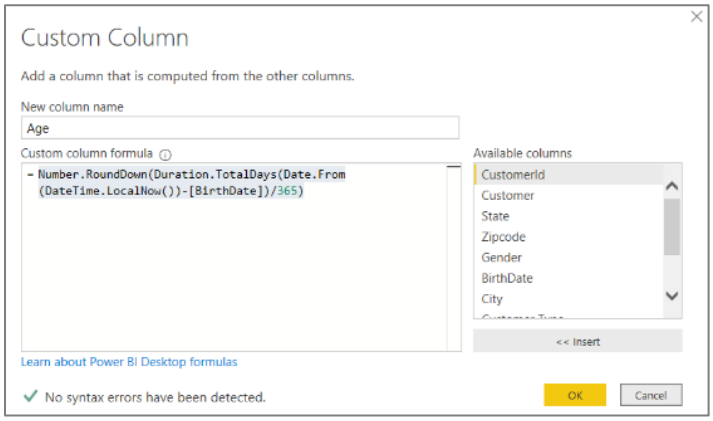


   d) When prompted by the **Custom Column** dialog enter a **New column name** of **Age**.
   e) Copy and paste the following M code into the **Custom column formula** textbox.

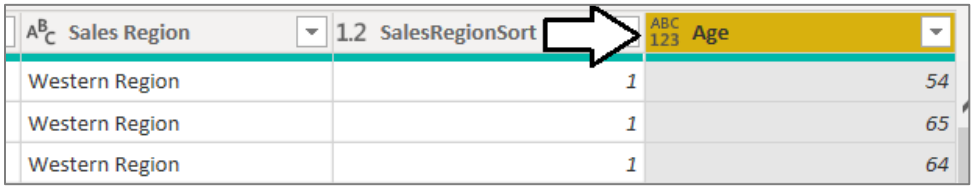
```
Number.RoundDown(Duration.TotalDays(Date.From(DateTime.LocalNow())-[BirthDate])/365)
```

You can also copy and paste this M expression from a text file in the **Lab** folder named **CustomerAge.m.txt**.

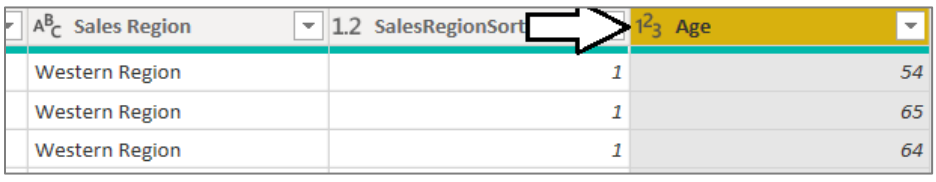
   f) Once the **Custom Column** dialog matches the following screenshot, click the **OK** button to add the new custom column.



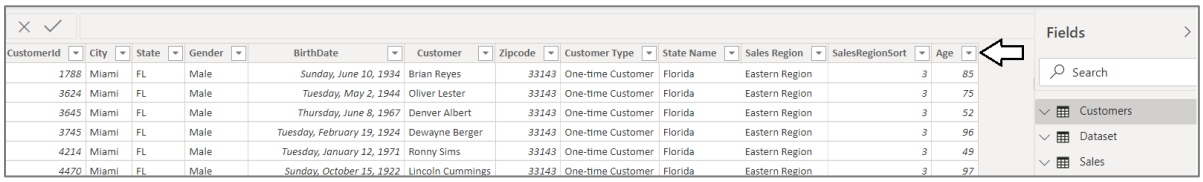
   g) You should see that the **Age** column has been created with a datatype of **Any**.



   h) Change the datatype of the **Age** column to **Whole Number**.



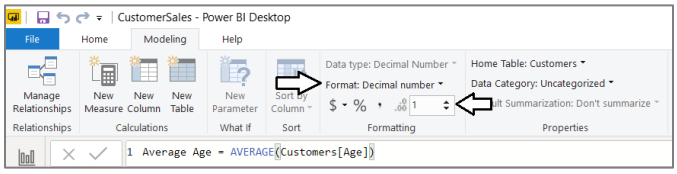
2. Process the changes you made to the **Customers** query.
   a) Click **Close & Apply** in the Power Query Editor window to process your query changes.
   b) Inspect the **Customers** table in the main Power BI Desktop window and verify you can see the new **Age** column.



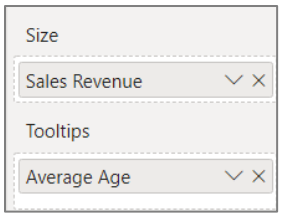
3. Add a new measure to the **Customers** table named **Average Age**.
   a) Navigate to **Data** view and select the **Customers** table from the **Fields** list.
   b) Click the **New Measure** button in the **Modeling** tab of the ribbon to add a new measure to the **Customers** table.
   c) Enter the following DAX expression into the formula bar to create the measure named **Sales Revenue**.

```
Average Age = AVERAGE(Customers[Age])
```

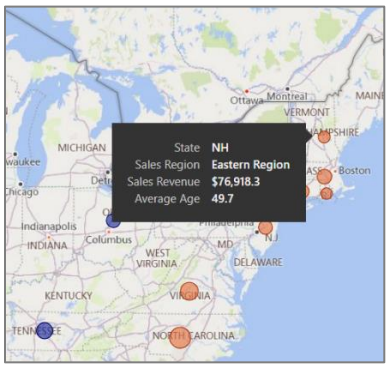
   d) Press the **ENTER** key to add the measure to data model.
   e) Modify the formatting by setting the **Format** menu to **Whole number** and the precision spinner to **2** places.



4. Add the **Average Age** measure to the **Tooltips** data role of the Map visual.
   a) Navigate to Report View and select the Map visual.
   b) Add the **Average Age** measure to the **Tooltips** data role.



   c) Hover the mouse over one of the bubbles in the map and inspect the tooltip to see the average customer age for a state.



Can you guess which state has the highest average age for its customers? Find the answer by inspecting the tooltip for each state.

Exercise 10: Creating a Reusable Function Query to Clean Text Values

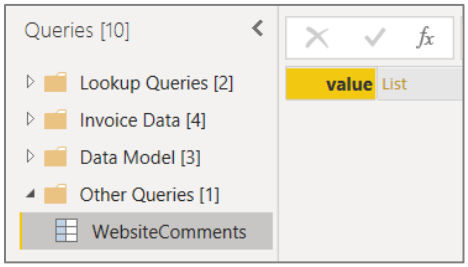
This is the second ***EXTRA CREDIT*** exercise in which you will begin by importing customer comments from a JSON file and surfacing those comments in a new Map visual. After that, you will create and use a query function named **CleanText** that strips unwanted characters from the customer comments so that the comments can be displayed more reliably in a Power BI report.

1. Import customer website comments from a JSON file accessible over the Internet named **WebsiteComments.json**.
   a) Drop down the **Get Data** menu from the Home tab of the ribbon and select the **Web** connector.
   b) When prompted by the **From Web** dialog, copy and paste the following URL to import data from **WebsiteComments.json**.

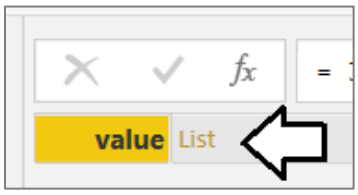
```
https://github.com/CriticalPathTraining/PowerBiMasterClass/raw/master/Student/Data/WebsiteComments.json
```

   c) Once you have pasted the URL into the **From Web** dialog, click the **OK** button to import the data and create a new query.

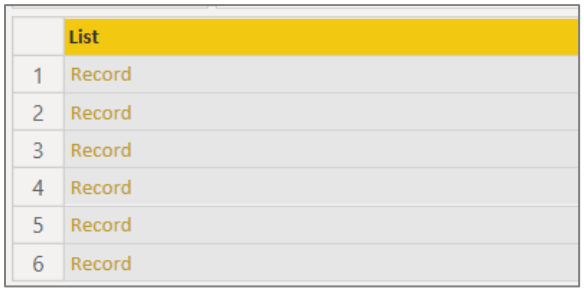
   d) Inspect (*but don't change any of the settings in*) the **WebsiteComments.json** dialog.
   e) Click the **Transform data** button to create a new query named **WebsiteComments** from **WebsiteComments.json**.



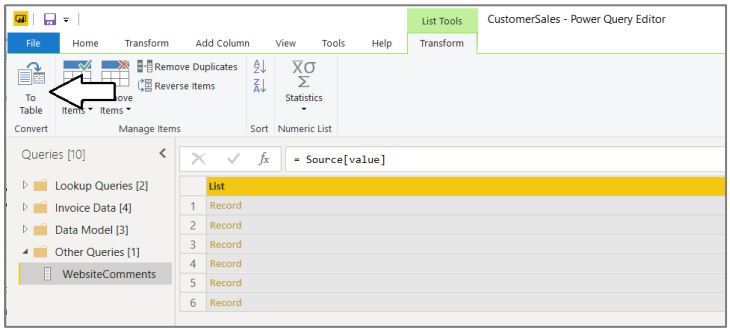
   f) In the query results pane, click on the **List** link to drill down into the list held by the **value** property.



   g) You should now see a list of records as shown in the following screenshot.

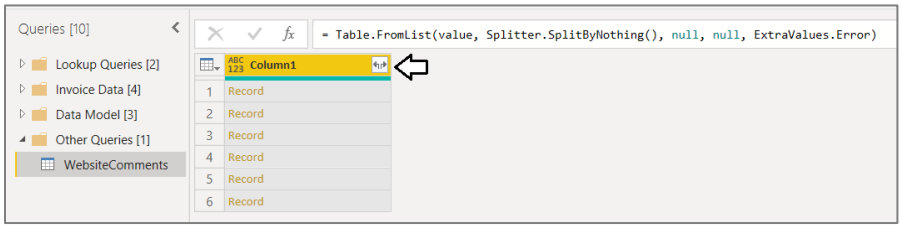


h) Click the **To Table** button on the **Transform** tab in the ribbon to display the **To Table** dialog.

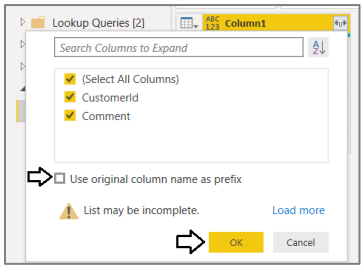


i) When prompted by the **To Table** dialog, click the **OK** button to convert the list into a table of records.

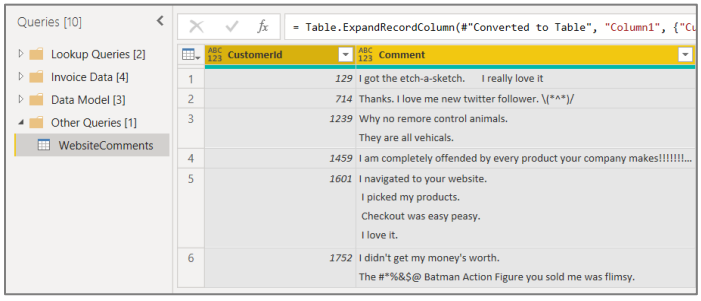
j) Click the Expand button for **Column1**.



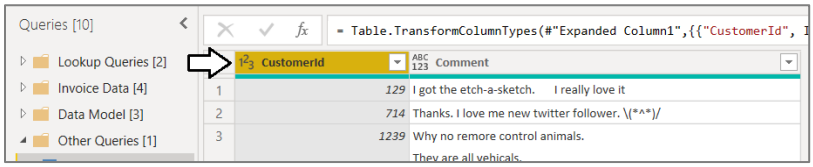
k) Uncheck the **Use original column name as prefix** checkbox and click **OK**.



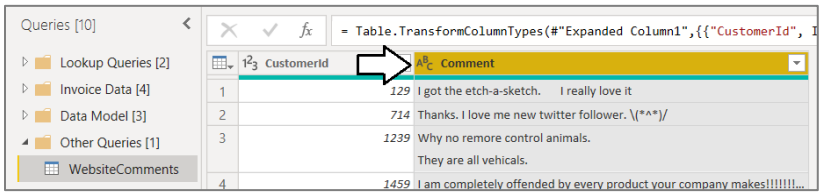
l) You should now see two columns named **CustomerId** and **Comment**.



m) Convert the datatype of the **CustomerId** column to **Whole Number**.

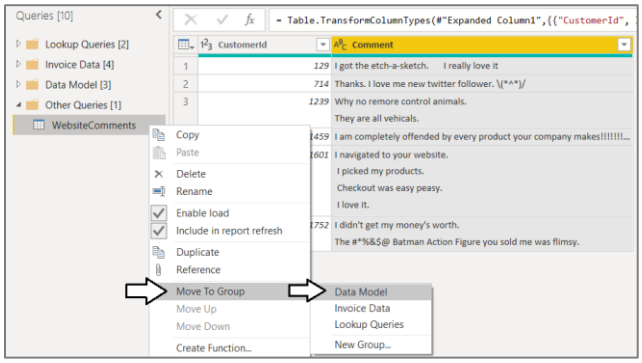


n) Convert the datatype of the **Comment** column to Text.

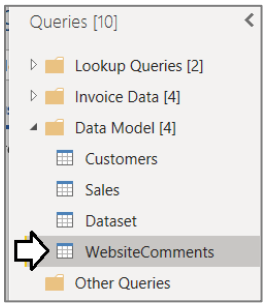


2. Add the **WebsiteComments** query to the **Data Model** folder group.

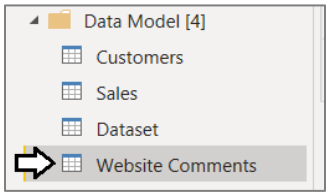
a) Right-click the **WebsiteComments** query and select the **Move To Group > Data Model** command.



b) The **WebsiteComments** query should now appear in the **Data Model** folder group.

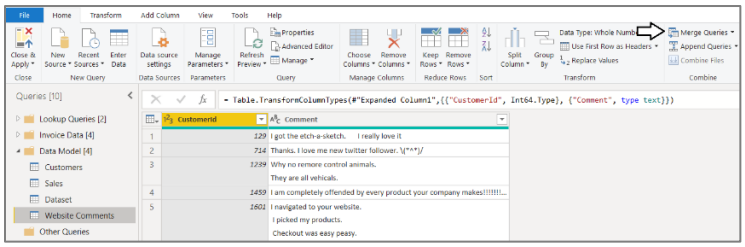


c) Rename the **WebsiteComments** query to The **Website Comments**.

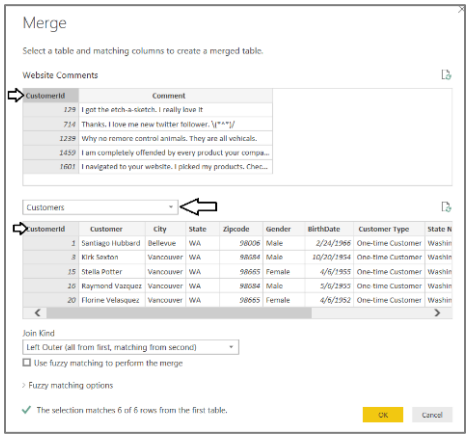


Since this query will create a table in the data model visible to report builders, a space is added to its name to make it more readable.

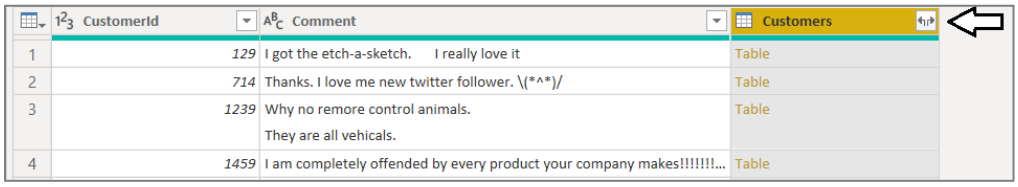
3. Merge data from the **Customers** query into the **Website Comments** query.
   a) With the **Website Comments** query selected in the **Queries** list, click the **Merge Query** link on the **Home** tab in the ribbon.



   b) On the **Merge** dialog, use the dropdown list to select the **Customers** table.
   c) Select the **CutomerId** for both queries to configure the **Merge** key.
   d) Click the **OK** button to complete the merge operation.

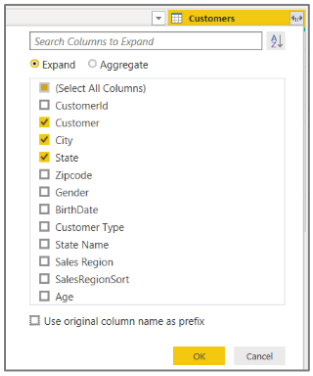


   e) Click the Expand button for the **Customers** column.

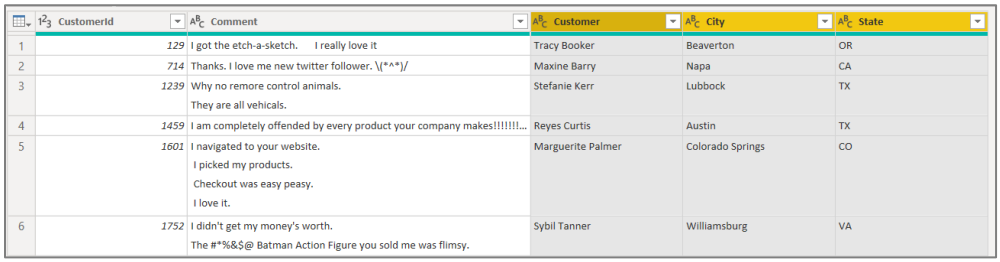


	CustomerId	Comment	Customers
1	129	I got the etch-a-sketch. I really love it	Table
2	714	Thanks. I love me new twitter follower. \(*^*)/	Table
3	1239	Why no remore control animals. They are all vehicals.	Table
4	1459	I am completely offended by every product your company makes!!!!!!!...	Table

   f) Click **(Select All Columns)** to unselect all columns.
   g) Select the columns **Customer**, **City** and **State**.
   h) Uncheck the **Use original column name as prefix** checkbox and click **OK**.



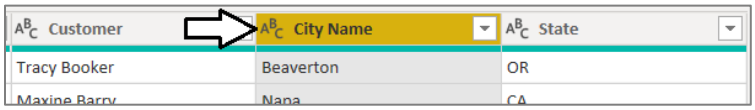
i) The **Website Comments** query output should now contain the columns **Customer**, **City** and **State**.



	CustomerId	Comment	Customer	City	State
1	129	I got the etch-a-sketch. I really love it	Tracy Booker	Beaverton	OR
2	714	Thanks. I love me new twitter follower. \(*^*)/	Maxine Barry	Napa	CA
3	1239	Why no remore control animals. They are all vehicals.	Stefanie Kerr	Lubbock	TX
4	1459	I am completely offended by every product your company makes!!!!!!!...	Reyes Curtis	Austin	TX
5	1601	I navigated to your website. I picked my products. Checkout was easy peasy. I love it.	Marguerite Palmer	Colorado Springs	CO
6	1752	I didn't get my money's worth. The #*%&$@ Batman Action Figure you sold me was flimsy.	Sybil Tanner	Williamsburg	VA

4. Create a merged column that parses together the city and state to support geographic mapping of cities.

a) Change the name of the **City** column to **City Name**.

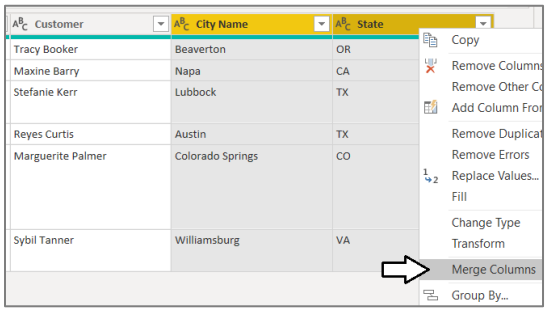


Customer	City Name	State
Tracy Booker	Beaverton	OR
Maxine Barry	Napa	CA

b) Select the **City Name** column.

c) Hold down the shift key and select the **State** column so both columns are selected.

d) Right-click the header of one of the selected columns and then select the **Merge Columns** command.

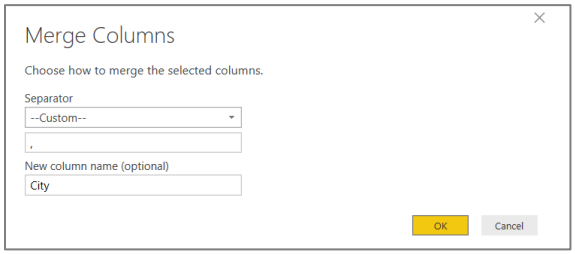


e) In the **Merge Columns** dialog, select a Separator value of **--Custom--**.

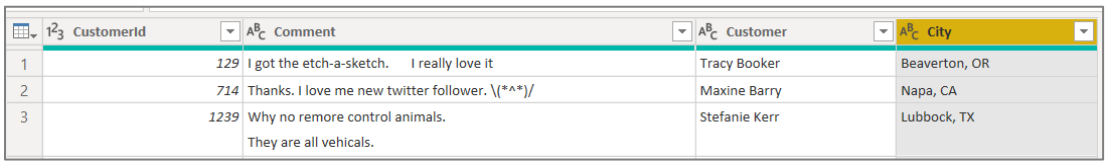
f) Enter a custom separator of a comma followed by a space.

g) Enter **City** for the **New column name**.

h) Click **OK** to complete the merge operation.

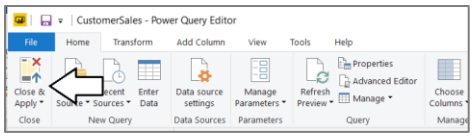


i) The **City** column should now contain both the city name and the state.



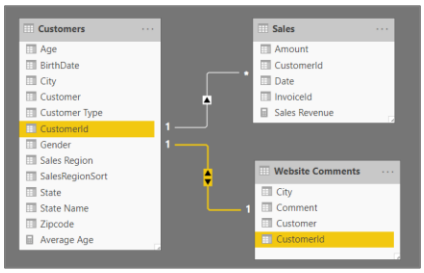
	CustomerId	Comment	Customer	City
1	129	I got the etch-a-sketch. I really love it	Tracy Booker	Beaverton, OR
2	714	Thanks. I love me new twitter follower. \(*^*)/	Maxine Barry	Napa, CA
3	1239	Why no remore control animals. They are all vehicals.	Stefanie Kerr	Lubbock, TX

5. Process your query changes.
   a) Click **Close and Apply** to process the query changes and close the Power Query Editor window,



   b) Navigate to **Model** view in the main Power BI Desktop windows and inspect the **Website Comments** table.

   c) Power BI Desktop has created a relationship between **Customers** and **Website Comments** based on **CustomerId**.



You will now add two new measures to the **Website Comments** table to surface website comments in tooltips for a Map visual.

6. Create a measure in the **Website Comments** table named **Customer Name** to display the first customer.
   a) Navigate to **Data** view and select the **Website Comments** 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 **Website Comments** table.
   c) Enter the following DAX expression into the formula bar to create the measure named **Customer Name**.

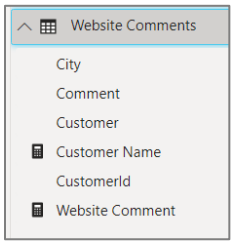
```
Customer Name = FIRSTNONBLANK('Website Comments'[Customer], 'Website Comments'[Customer])
```

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

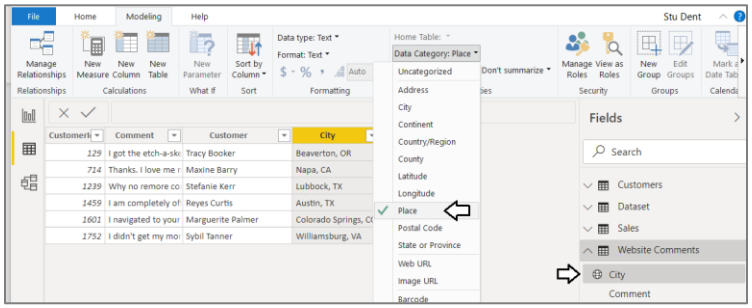
7. Create a measure in the **Website Comments** table named **Website Comment** to display the first customer.
   a) Click the **New Measure** button in the **Modeling** tab of the ribbon to add a new measure to the **Website Comments** table.
   b) Enter the following DAX expression into the formula bar to create the measure named **Customer Name**.

```
Website Comment = FIRSTNONBLANK('Website Comments'[Comment], 'Website Comments'[Comment])
```

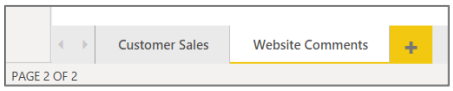
   c) Press the **ENTER** key to add the measure to data model.
   d) The **Website Comments** table should now contain two new measures named **Customer Name** and **Website Comment**.



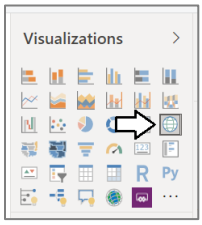
8. Set the **Data Category** for the **City** column to support geographic mapping for US cities.
   a) Navigate to **Data** view and select the **City** column from the **Website Comments** table in the **Fields** list.
   b) Drop down the **Data Category** menu and assign the **City** column a data category of **Place**.



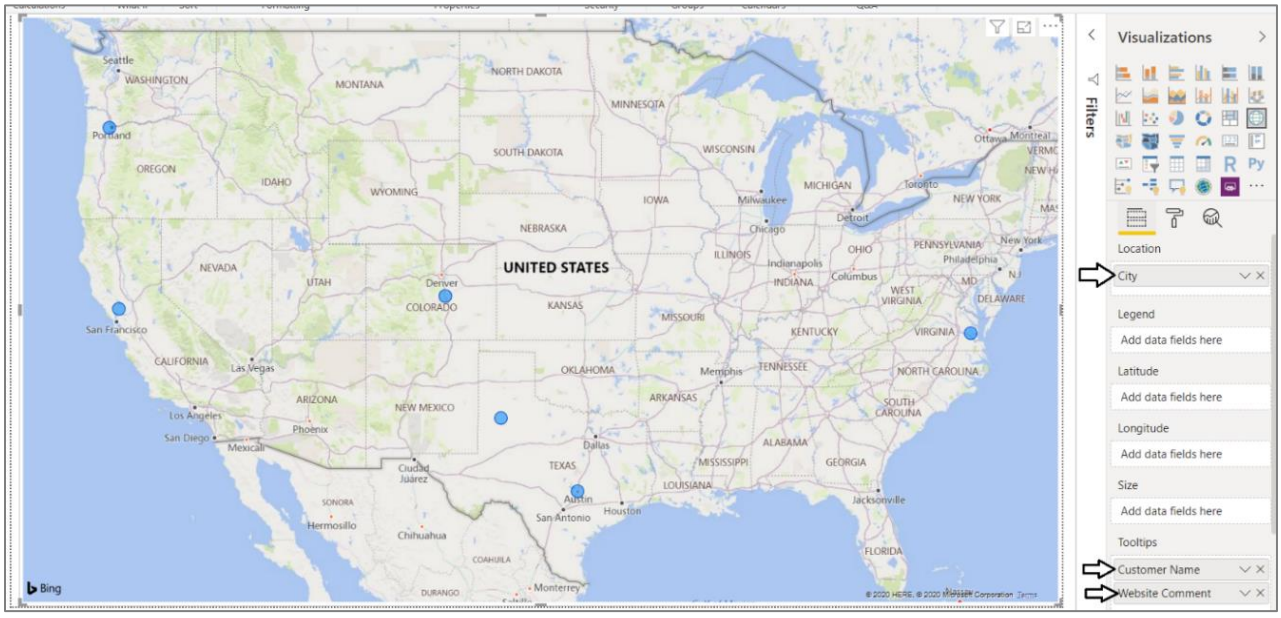
9. Create a new report page named **Website Comments** and add a Map visual.
   a) Navigate to **Report** view.
   b) Add a new report page and rename it to **Website Comments.**



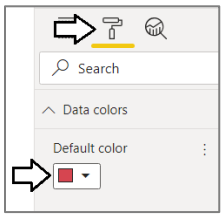
   c) Add a new Map visual and reposition it so it takes up the entire width and height of the report page.



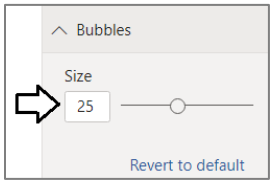
   d) Add the **City** field from the **Website Comments** table to the **Location** data role of the Map visual.
   e) Add the **Customer Name** and **Website Comment** measures from the **Website Comments** table to the **Tooltips** data role.



10. Make the map bubbles for the website comments easier to see.
    a) With the Map visual selected, navigate to the Format pane.
    b) Expand the **Data** color section and select a bright color such as red so the bubbles stand out more.



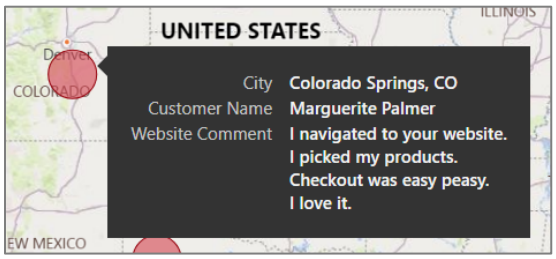
    c) Expand the **Bubble** section and increase the bubble **Size** to **25**.



    d) The map bubbles should now be easier to see.

    e) Set the **Title** property of the Map visual to **Off**.

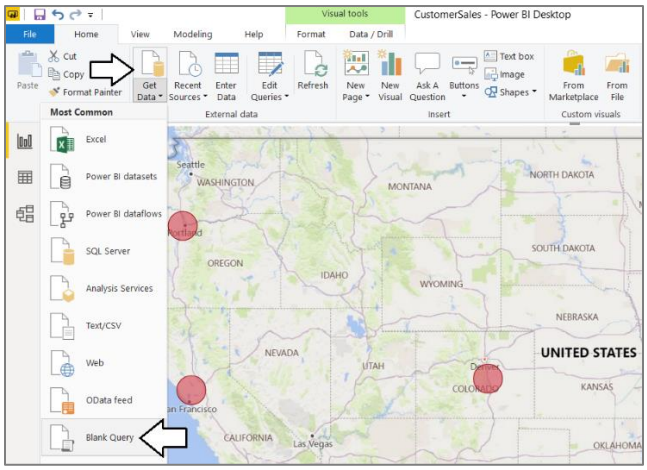
    f) Hover the mouse over a bubble and inspect the tooltip which should display the customer name and website comments.



At this point, the Map visual is displaying website comments. However, some website comments contain unwanted characters such as line breaks and tabs. In the final steps of this lab, you will create a function query to strip out these unwanted charactors.

11. Create a new function query named **CleanText**.

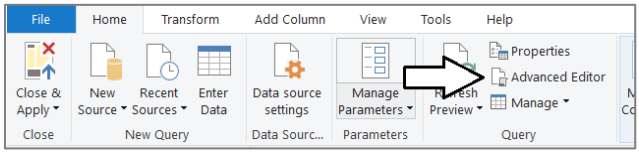
a) Drop down the **Get Data** menu and select the **Blank Query** command.



b) When you execute the **Blank Query** command, Power Query Editor window creates a new query named **Query1**.

c) Rename the new query from **Query1** to **CleanText**.

d) Click on the **Advanced Editor** button in the ribbon so you can directly edit the M code for the **CleanText** query.



e) Delete all the existing M code from the Advanced Editor window and replace it with the following M expression.

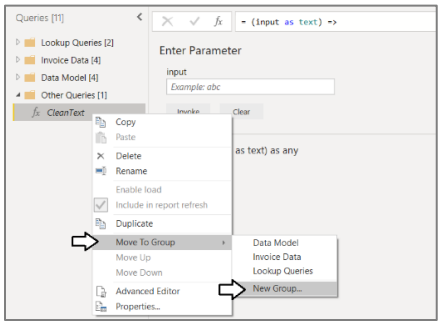
```
 (input as text) =>
let
    set1 = {"A".."Z"},
    set2 = {"a".."z"},
    set3 =  List.Transform({0..9}, each Number.ToText(_) ),
    set4 = {" ", "-", "_", ".", "'"},
    allowedChars = set1 & set2 & set3 & set4,
    output = Text.Select(input, allowedChars)
in
    output
```

If it's a problem to copy and paste M code from this PDF file, you can also copy this M code from **CleanText.m.txt** in the **Lab** folder.

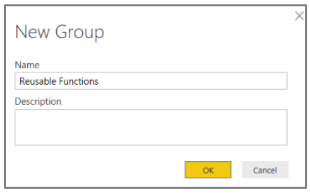
f) Once you have added the M expression into the **Advanced Editor** dialog for the **CleanText** query, click **OK**.

12. Move the **CleanText** query function into a new folder group named **Reusable Functions**.

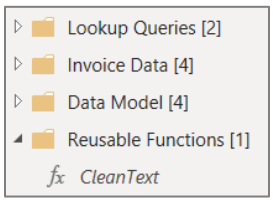
a) Right-click the **CleanText** query in the **Queries** list and select the **Move To Group > New Group** command.



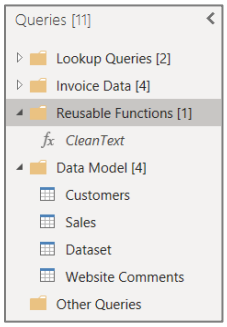
b) In the **New Group** dialog, enter a new of **Reusable Functions** and click **OK**.



c) The **CleanText** query should now be in the **Reusable Functions** folder group.



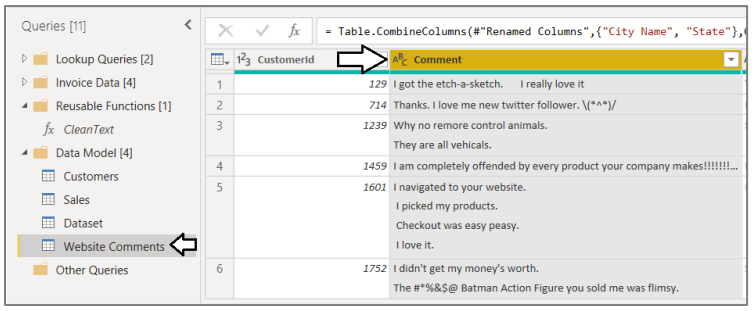
d) Move the **Reusable Functions** folder group up so it appears above the **Data Model** folder group.



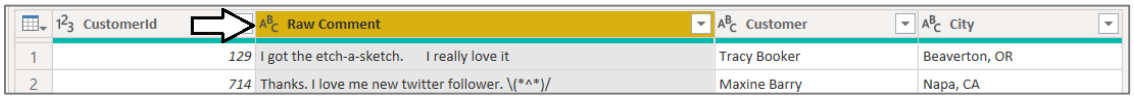
The **Data Model** folder group is placed at the bottom because it contains the only queries that will generate tables in the data model.

13. Call the **CleanText** query function from the **Website Comments** query to clean the text in customer comments.

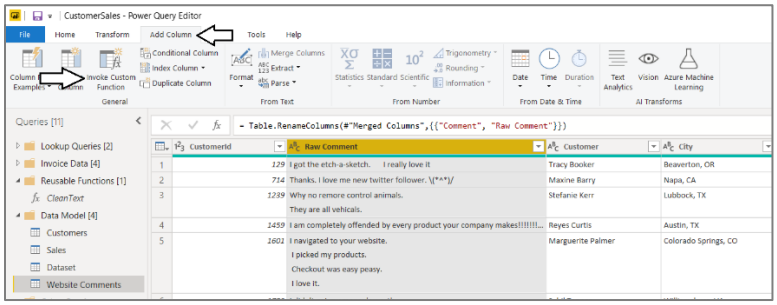
a) Select the **Website Comments** query in the **Queries** list on the left and locate the **Comment** column.



b) Change the name of the **Comment** column to **Raw Comment**.



c) Navigate to the **Add Column** tab in the ribbon and click the **Invoke Custom Function** button.

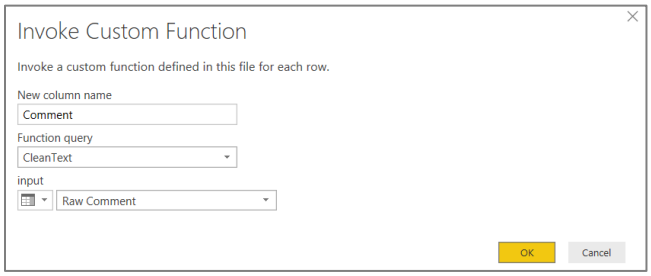


d) In the **Invoke Custom Function** dialog, enter a **New column name** of **Comment**.

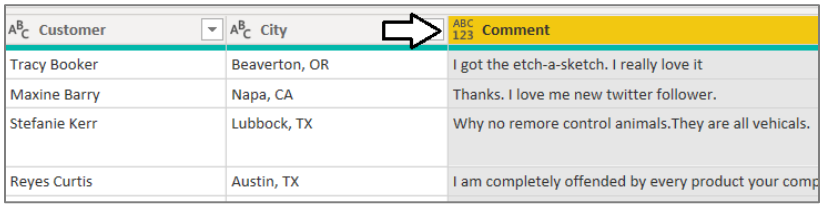
e) Select **CleanText** from the **Function query** dropdown list.

f) Select the **Raw Comment** column as the function **input**.

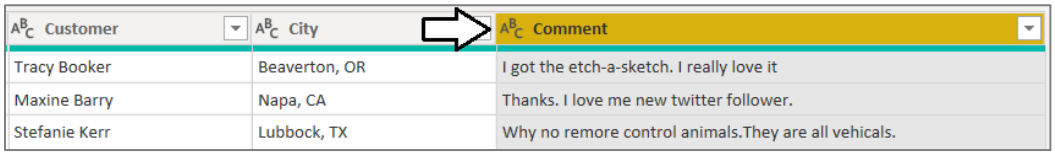
g) When the **Invoke Custom Function** dialog matches the following screenshot, click **OK** to complete the operation.



h) You should now see a new **Comment** column whose text value has been processed by the **CleanText** query function.

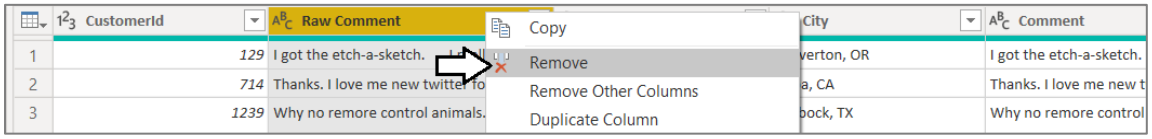


i) Convert the datatype of the **Comment** column to **Text**.

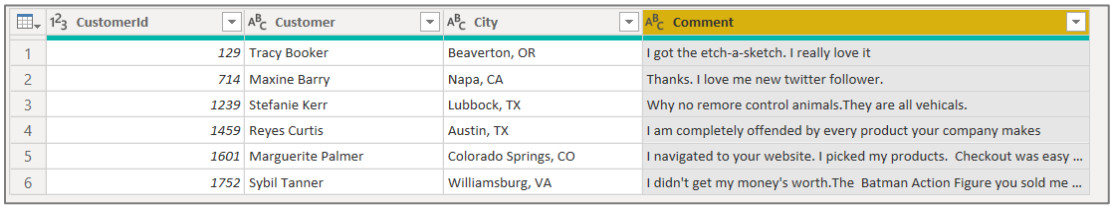


Customer	City	Comment
Tracy Booker	Beaverton, OR	I got the etch-a-sketch. I really love it
Maxine Barry	Napa, CA	Thanks. I love me new twitter follower.
Stefanie Kerr	Lubbock, TX	Why no remore control animals.They are all vehicals.

j) Remove the **Raw Comment** column which is no longer needed.



k) The output of the **Website Comments** column should now match the following screenshot.



	CustomerId	Customer	City	Comment
1	129	Tracy Booker	Beaverton, OR	I got the etch-a-sketch. I really love it
2	714	Maxine Barry	Napa, CA	Thanks. I love me new twitter follower.
3	1239	Stefanie Kerr	Lubbock, TX	Why no remore control animals.They are all vehicals.
4	1459	Reyes Curtis	Austin, TX	I am completely offended by every product your company makes
5	1601	Marguerite Palmer	Colorado Springs, CO	I navigated to your website. I picked my products. Checkout was easy ...
6	1752	Sybil Tanner	Williamsburg, VA	I didn't get my money's worth.The Batman Action Figure you sold me ...

14. Process changes to your queries to see the result of cleaning website comments.
    a) Click **Close and Apply** to process the query changes and close the Power Query Editor window,
    b) Inspect the comments in the Map visual on the **Website Comments** page.
    c) All website comments in tooltips should be displayed without line breaks or other unwanted characters..

Congratulations. You have now completed all the exercises in the lab including the extra credit.