Building a Reporting Project with Power BI Desktop

Lab Time: 45-60 minutes

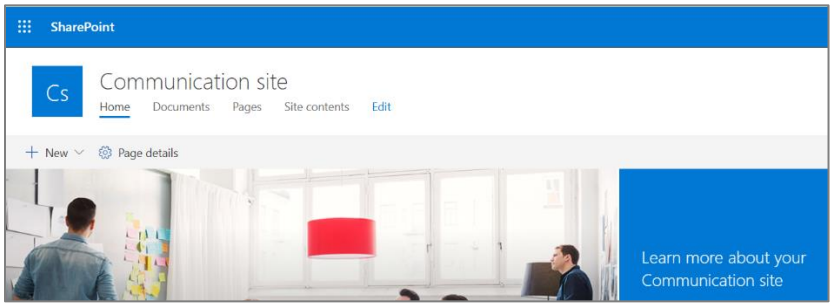
Lab Folder: C:\Student\Modules\02_PowerBiDesktop\Lab\

Lab Overview: In this lab you will begin by designing a query to extract expense data from an unstructured text file in a SharePoint Online document library. You will then convert your query into a function query to extract data from multiple files into a single table.

Exercise 1: Build a Query to Extract Data from an Unstructured Text File

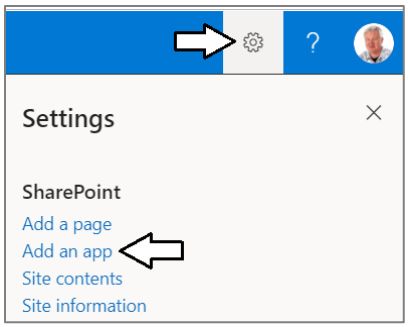
In the following exercise, you will use the **Query Editor** window to design an advanced query using a query function.

1. Navigate to the SharePoint team site at the root of your SharePoint tenancy.
   a) The SharePoint site should have a URL in the form of **https://[Your tenant name].sharepoint.com**.

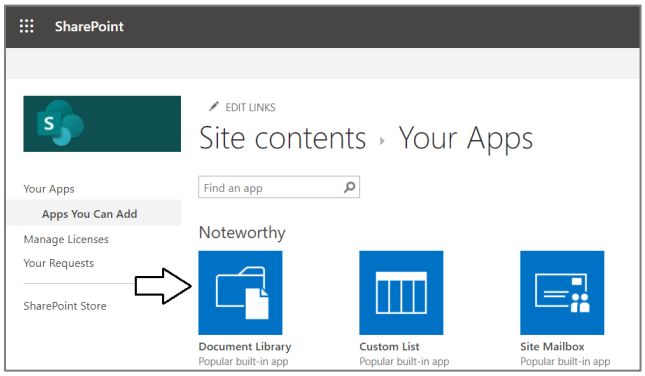


You can use any SharePoint Online team site for this exercise as long as you have permissions in the site to create new lists.

2. Create a new document library named **Data** to store the files with expense data.
   a) Drop down the Site Actions menu and select **Add an app**.

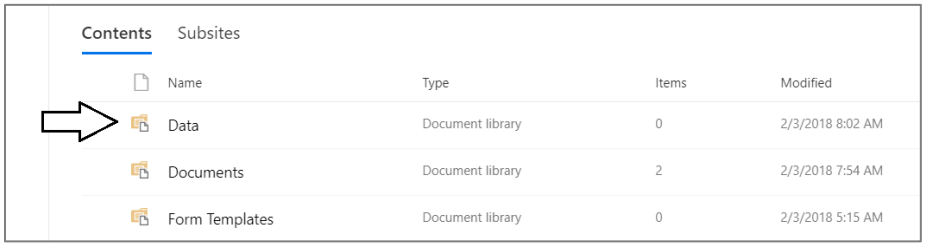


   b) Select **Document Library** as the type of list to create.

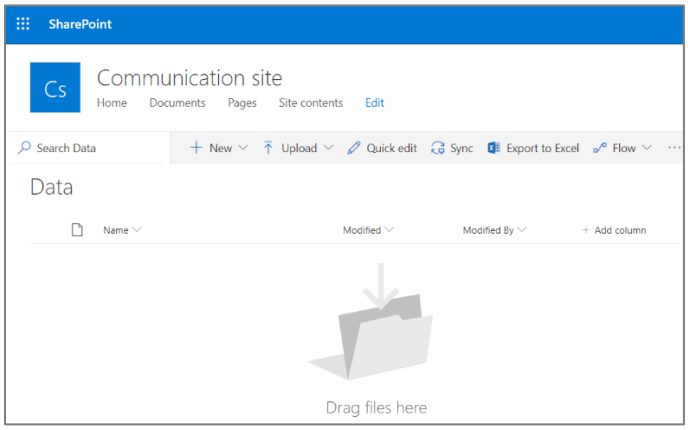


c) In the **Adding Document Library** dialog, add a **Name** of **Data** and click **Create**.

d) Once the **Data** document library has been created, navigate to its default view.



e) You should now be at the default view for the Data document library.

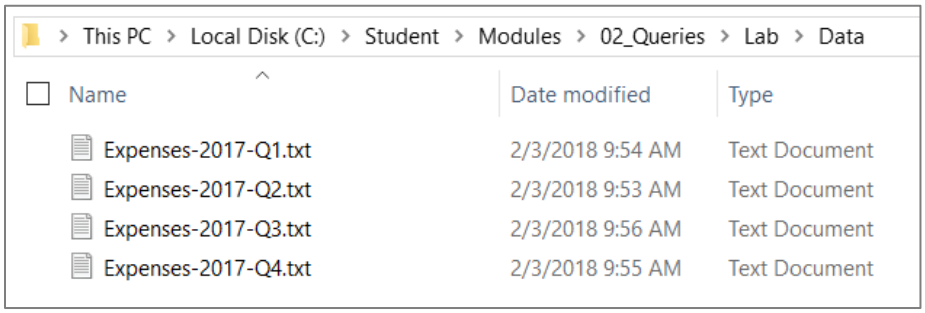


3. Upload data files to the **Data** document library.

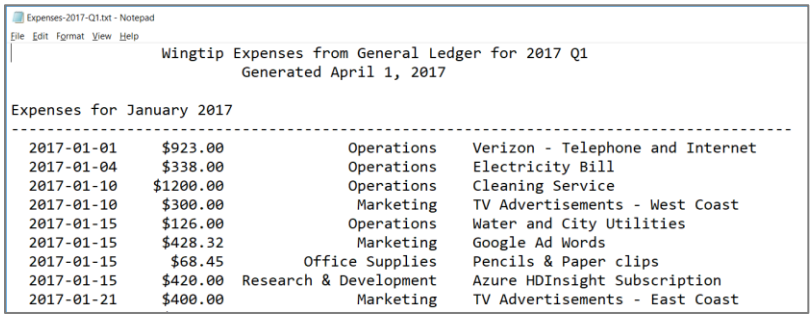
a) Using Windows Explorer, examine the data files into the follow folder.

```
C:\Student\Modules\02_Queries\Lab\Data
```

b) You should see the following four files as shown in the following screenshot.



c) Double-click on the file named **Expenses-2017-Q1.txt**. to open it in Windows Notepad and inspect its contents.



As you can see, this file is an unstructured text file with fixed-width lines which contain expense data. The other three files in the Data folder have expense data for different time periods, but the format of their contents is the same.

d) Upload the following three files to the **Data** document library.

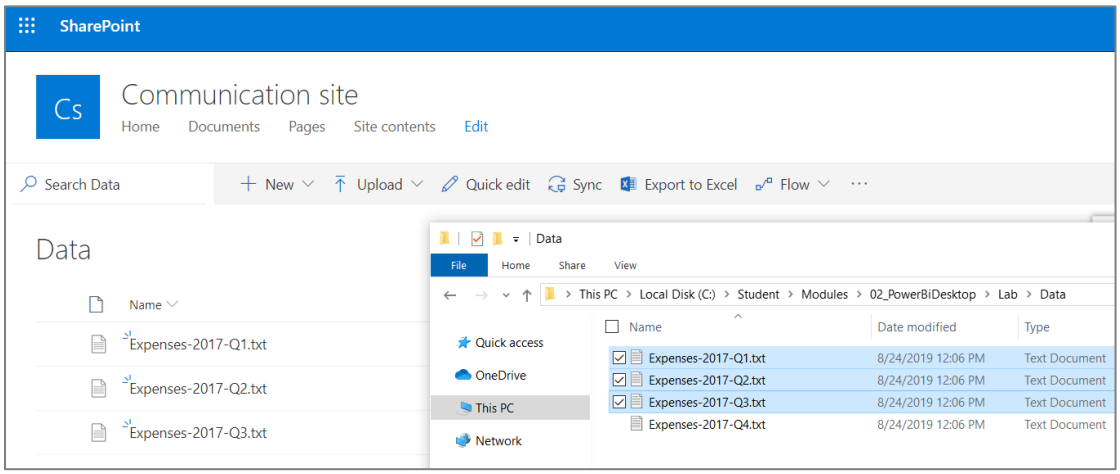
i) **Expenses-2017-Q1.txt**

ii) **Expenses-2017-Q2.txt**

iii) **Expenses-2017-Q3.txt**

Note that you should NOT upload the fourth file named **Expenses-2017-Q4.txt**. You will upload the last file later in this lab.

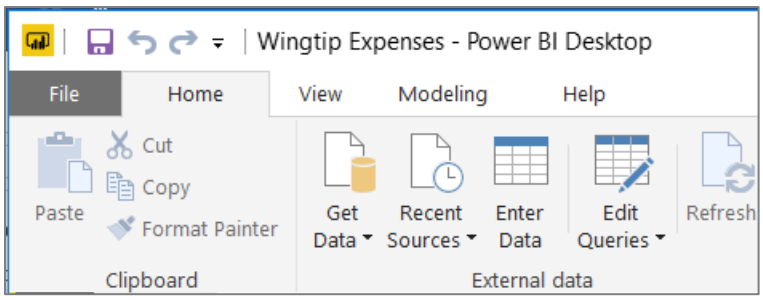
e) You should be able to verify that those three files have been upload to the **Data** document library.



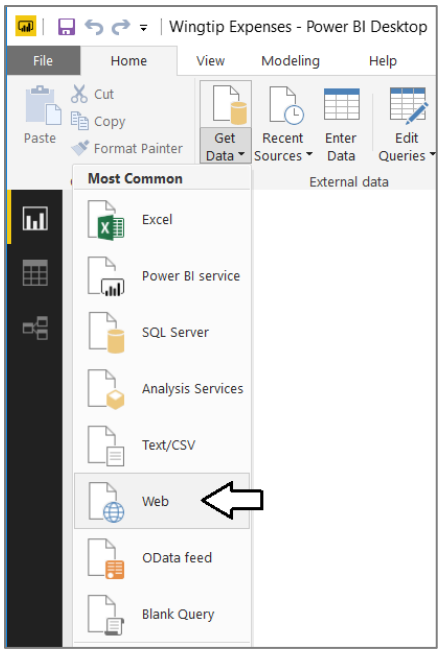
4. Create a new Power BI Desktop project named **Wingtip Expenses.pbix**.

a) Launch Power BI Desktop.

b) Begin by saving the new project and give it a name of **Wingtip Expenses.pbix**.



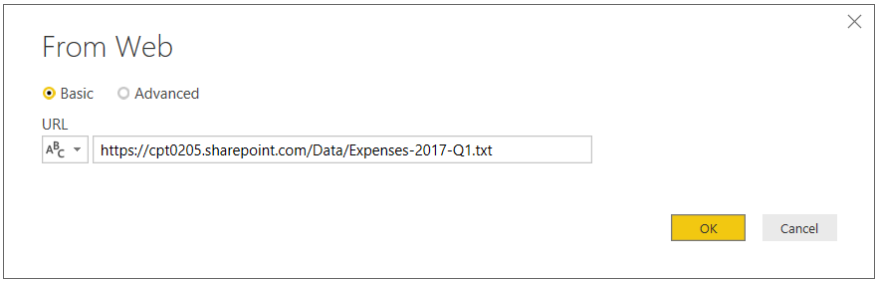
5. Create a new query to import data from the file in the **Data** document library named **Expenses-2017-Q1.txt**.
   a) Drop down the **Get Data** menu and select the **Web** command.



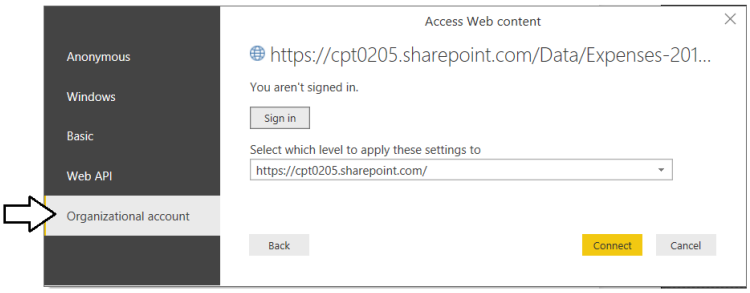
   b) In the **From Web** dialog, add the path to the file named **Expenses-2017-Q1.txt**. The path should include the base URL of your SharePoint site along with the relative file path which is **/Data/Expenses-2017-Q1.txt**.
   c) Your path should look something like the following URL

```
https://cpt0205.sharepoint.com/Data/Expenses-2017-Q1.txt
```

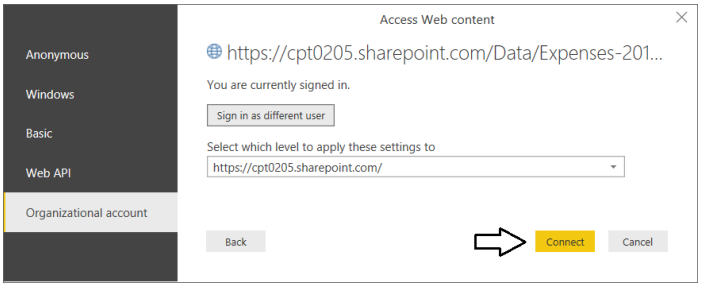
   d) Once you have added the file path to the **From Web** dialog, click **OK**.



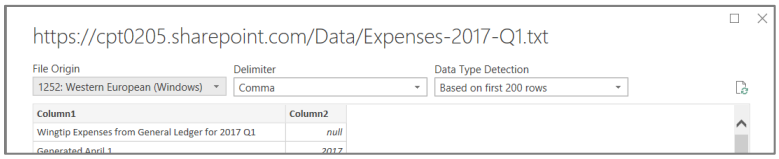
   e) If you are prompted to login with the **Access Web content** dialog, select **Organizational account** and click **Sign in**.



f) Once you have signed in on the **Access Web content** dialog, click the **Connect** button



g) You should now be prompted with the dialog shown in the following screenshot.

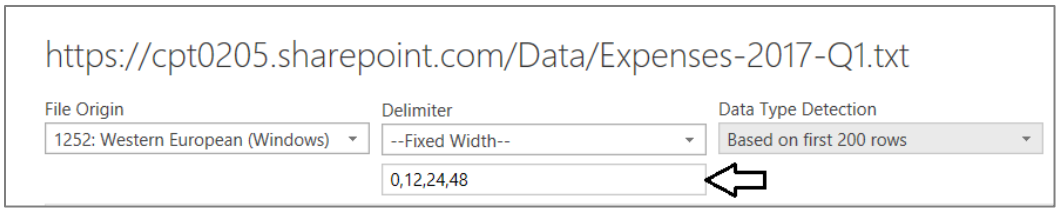


h) Set the **Delimiter** dropdown menu option to **--Fixed Width--**.

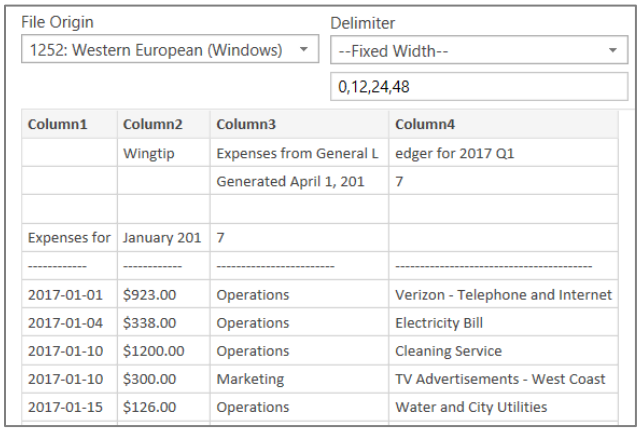
i) In the textbox under the **Delimiter** dropdown menu, add the following column positions.

```
0,12,24,48
```

j) Make sure your fixed-width column positions match the following screenshot.

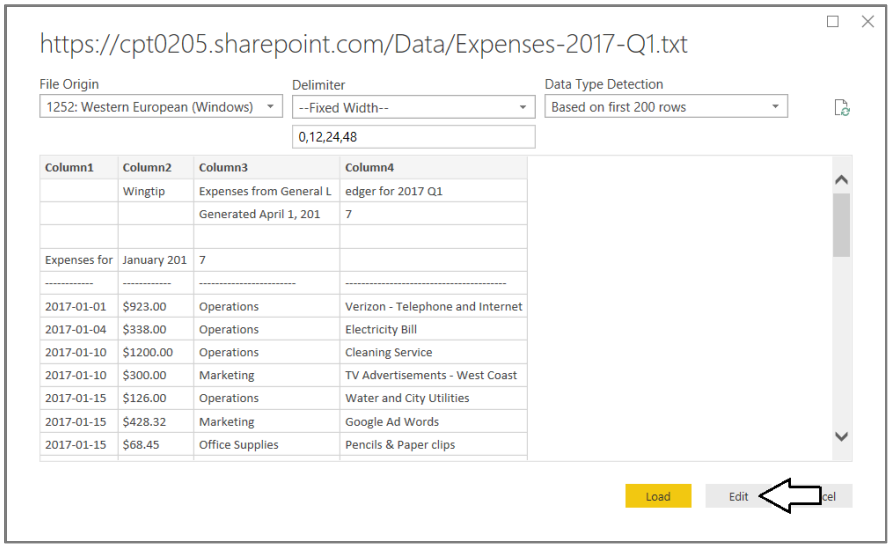


k) At this point, you should be able to see data in some of the rows conforming to the fixed-width column scheme.

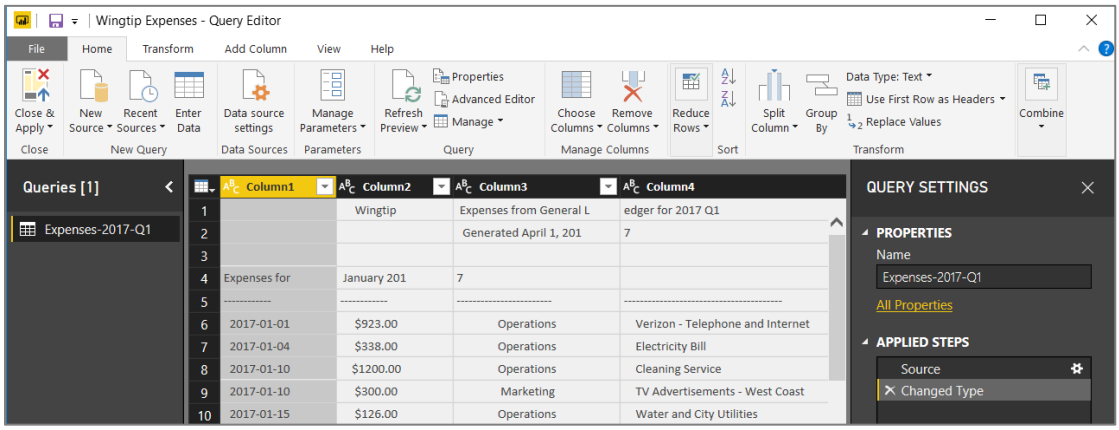


Column1	Column2	Column3	Column4
	Wingtip	Expenses from General L	edger for 2017 Q1
		Generated April 1, 201	7
Expenses for	January 201	7	
---	---	---	---
2017-01-01	$923.00	Operations	Verizon - Telephone and Internet
2017-01-04	$338.00	Operations	Electricity Bill
2017-01-10	$1200.00	Operations	Cleaning Service
2017-01-10	$300.00	Marketing	TV Advertisements - West Coast
2017-01-15	$126.00	Operations	Water and City Utilities

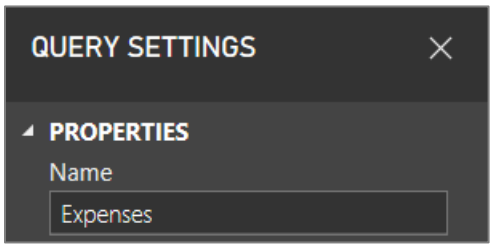
l) Click the **Edit** button at the bottom right of the dialog to open the new query in the Query Editor window.



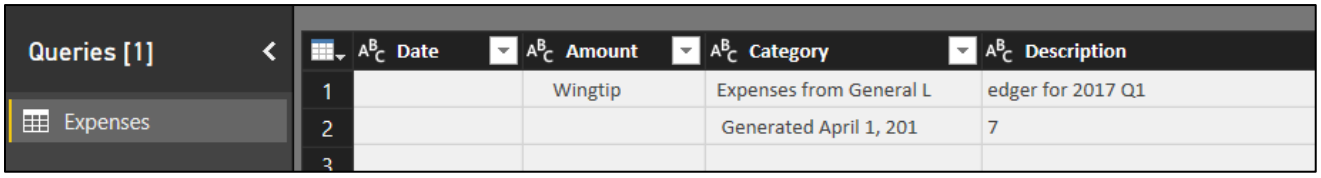
m) Your new query should appear in the Query Editor window.



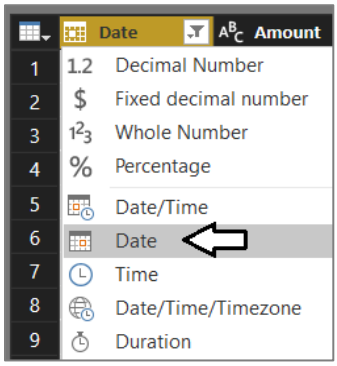
n) In the **QUERY SETTING** pane in the upper right, modify the name of the query to **Expenses**.



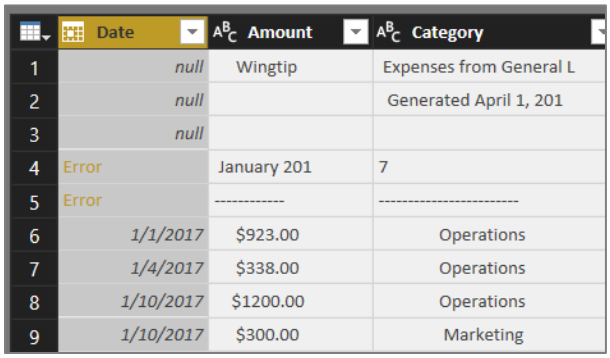
o) Update the 4 column names in the query to **Date**, **Amount**, **Category** and **Description** as shown in the following screenshot.



p) Use the Data type dropdown menu at the top left of the **Date** column to change its type to **Date**.

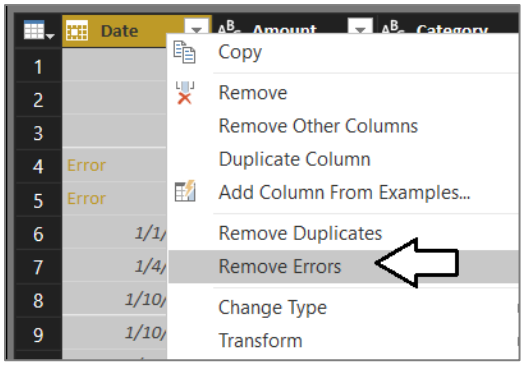


q) You should observe that rows that contain valid data are able to convert their **Date** column values to actual dates while the rows that do not contain valid data show either errors or null values in the **Date** column.

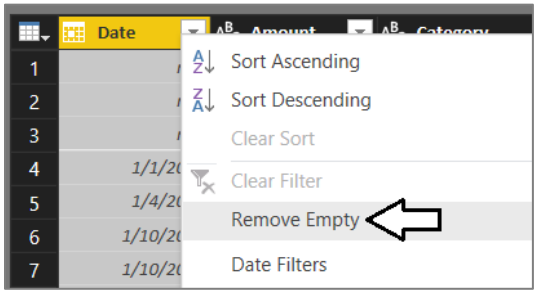


	Date	Amount	Category
1	null	Wingtip	Expenses from General L
2	null		Generated April 1, 201
3	null		
4	Error	January 201	7
5	Error	----------	----------
6	1/1/2017	$923.00	Operations
7	1/4/2017	$338.00	Operations
8	1/10/2017	$1200.00	Operations
9	1/10/2017	$300.00	Marketing

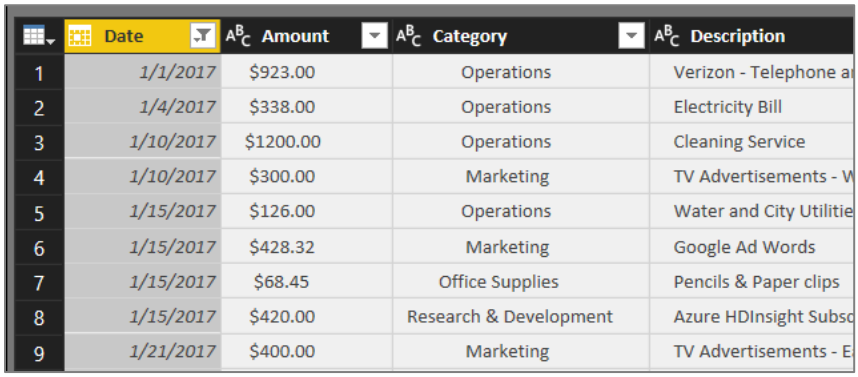
r) Right-click on the **Date** column header and select the **Remove Errors** command.



s) Click the dropdown menu on the right-side of the **Date** column and select the **Remove Empty** command.

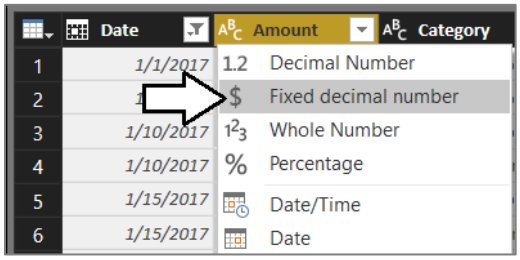


t) The query output should now only contain rows with a valid date value in the **Date** column.

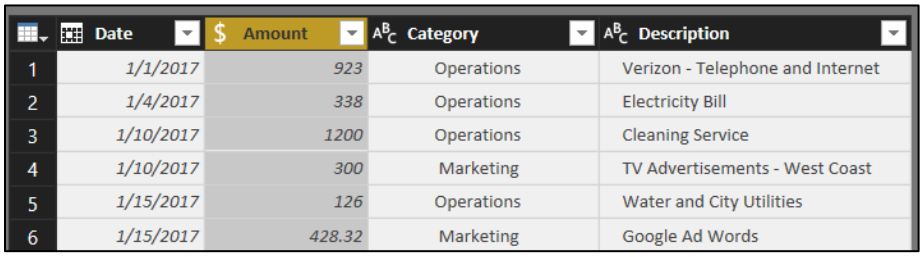


	Date	Amount	Category	Description
1	1/1/2017	$923.00	Operations	Verizon - Telephone a
2	1/4/2017	$338.00	Operations	Electricity Bill
3	1/10/2017	$1200.00	Operations	Cleaning Service
4	1/10/2017	$300.00	Marketing	TV Advertisements - W
5	1/15/2017	$126.00	Operations	Water and City Utilitie
6	1/15/2017	$428.32	Marketing	Google Ad Words
7	1/15/2017	$68.45	Office Supplies	Pencils & Paper clips
8	1/15/2017	$420.00	Research & Development	Azure HDInsight Subsc
9	1/21/2017	$400.00	Marketing	TV Advertisements - E

u) Modify the data type of the **Amount** column to be **Fixed decimal number**.

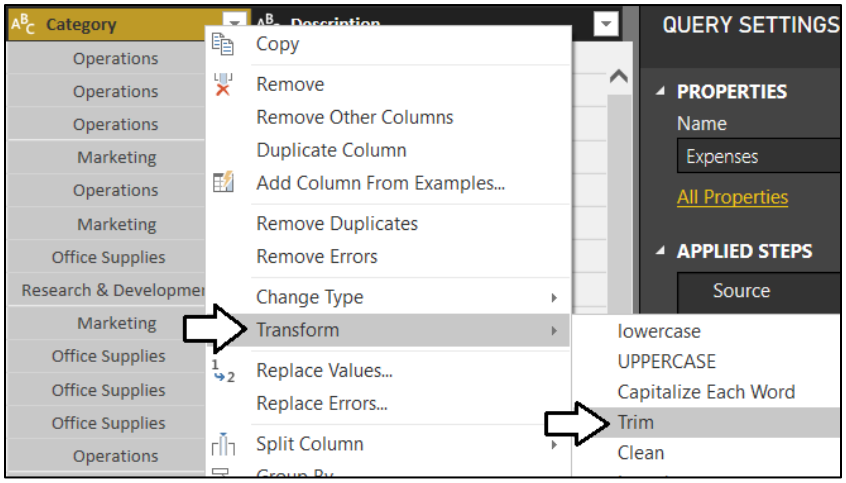


v) The data type for the **Amount** column should now show a dollar sign to indicate its type is Fixed decimal number.



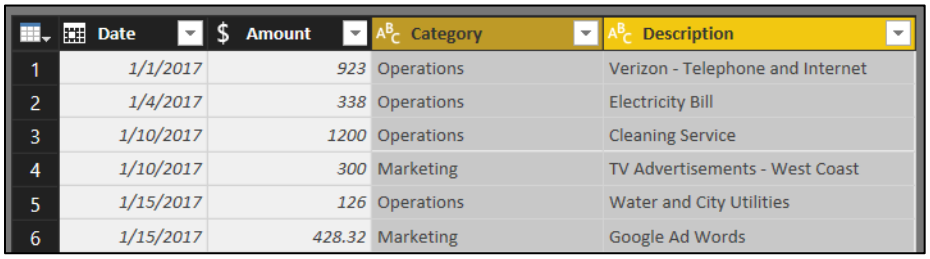
	Date	Amount	Category	Description
1	1/1/2017	923	Operations	Verizon - Telephone and Internet
2	1/4/2017	338	Operations	Electricity Bill
3	1/10/2017	1200	Operations	Cleaning Service
4	1/10/2017	300	Marketing	TV Advertisements - West Coast
5	1/15/2017	126	Operations	Water and City Utilities
6	1/15/2017	428.32	Marketing	Google Ad Words

w) Right click on the **Category** column header and select the **Transform > Trim** command.



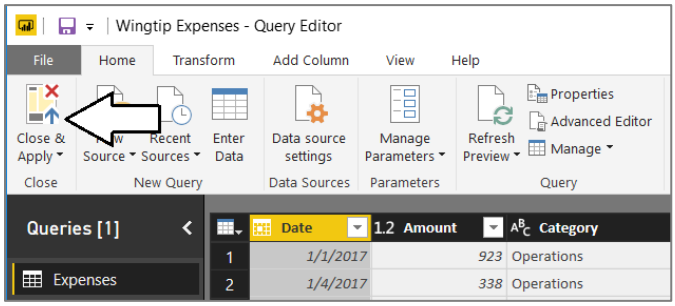
x) Right click on the **Description** column header and select the **Transform > Trim** command.

y) The **Category** and the **Description** columns should no longer have any extra whitespace.



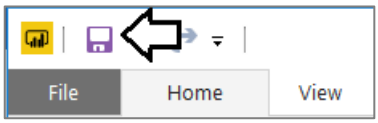
	Date	Amount	Category	Description
1	1/1/2017	923	Operations	Verizon - Telephone and Internet
2	1/4/2017	338	Operations	Electricity Bill
3	1/10/2017	1200	Operations	Cleaning Service
4	1/10/2017	300	Marketing	TV Advertisements - West Coast
5	1/15/2017	126	Operations	Water and City Utilities
6	1/15/2017	428.32	Marketing	Google Ad Words

z) You are done designing this query. Click the **Close and Apply** button to run the query and close the Query Editor window.



This would be a good time to save the work you have completed so far.

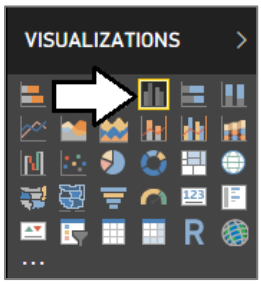
6. Save your work in the Wingtip Expenses project by clicking the **Save** icon in the upper left of the Power BI Desktop window.



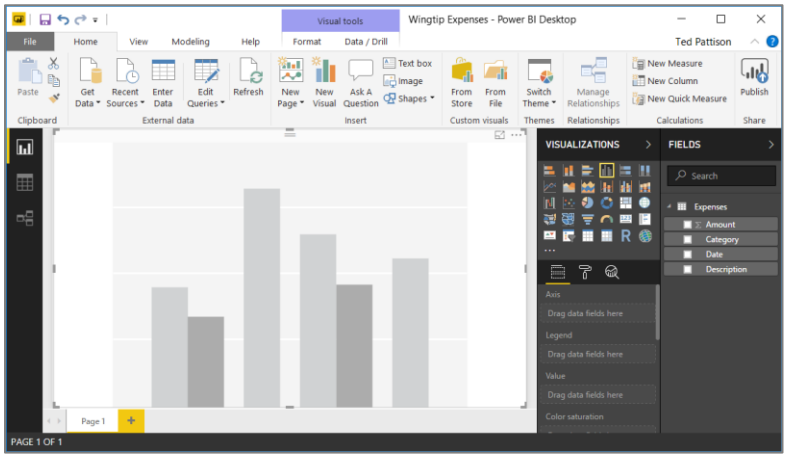
7. Create a report to display expense data.

a) In the Power BI Desktop application window, navigate to report design view.

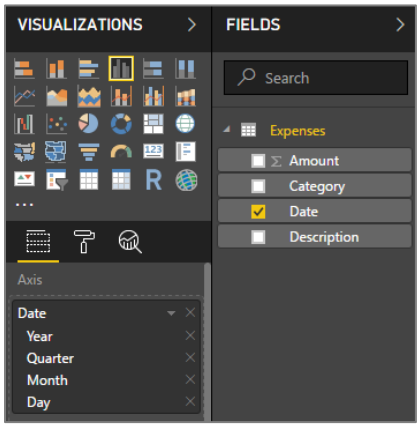
b) Add a new **Clustered column chart** visual.



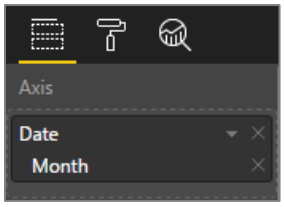
c) Using the mouse, reposition the column chart visual so it takes up the entire page in the report.



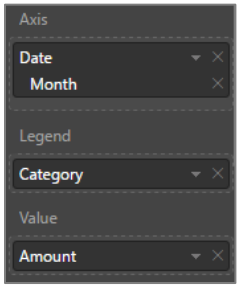
d) Click on the checkbox for the **Date** column in the **FIELDS** list on the right. When you do this, you should see that a date hierarchy is automatically added to the **Axis** well.



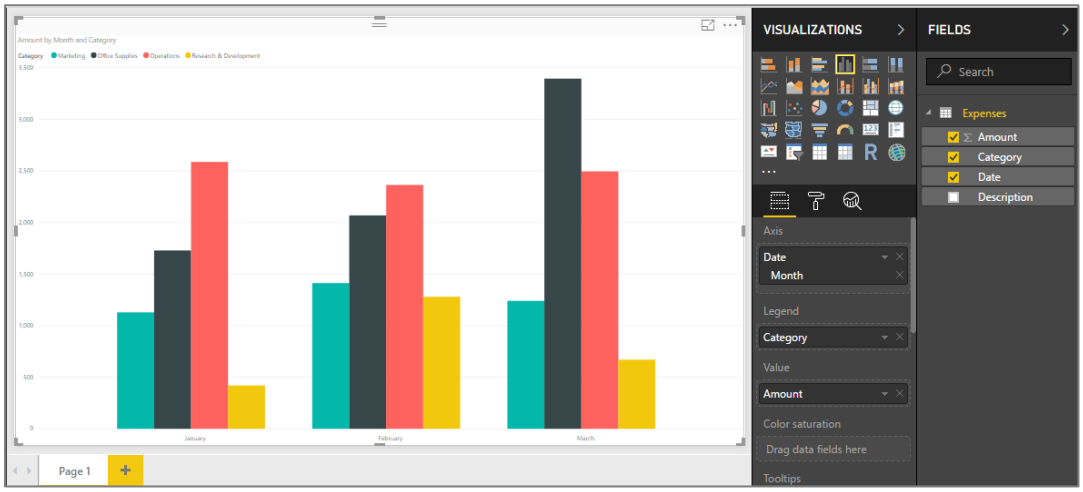
e) Remove **Year**, **Quarter** and **Day** from the date hierarchy so that only **Month** remains as shown in the following screenshot.



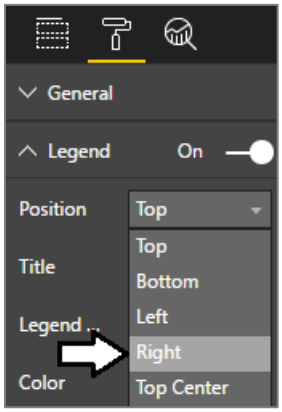
f) Add the **Category** field to the **Legend** well and add the **Amount** field to the **Value** well.



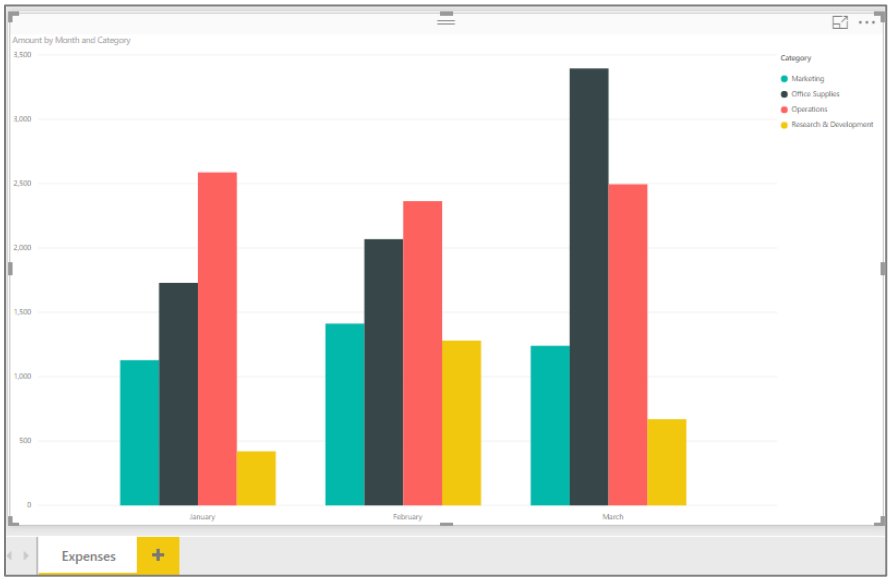
g) You report should now match the following screenshot.



h) Modify the **Position** property in the **Legend** section of the **Format** pane so that column chart displays its legend on the right.

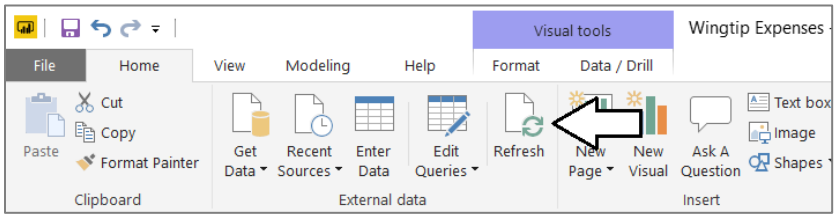


i) In the lower left corner of the report page, update the page title to **Expenses**.

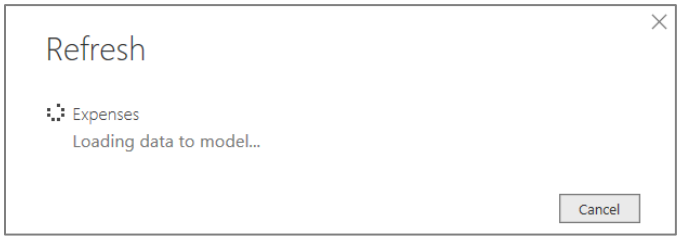


Now that you have created the query and the report, you should be able to refresh the data at any time. In the next step, you will test this by refreshing the report. Note that nothing will change because the data will remain the same. However, the key point is that refreshing the report will import the most recent data into your project without having to make any updates to your query or report.

8. Refresh the data in the Wingtip Expenses project.
   a) Click on the **Refresh** button in the **Home** tab of the ribbon.



   b) Wait while Power BI Desktop imports the current expenses data.

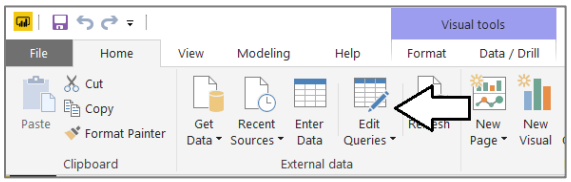


9. Save your work clicking the **Save** icon in the upper, left-hand side of the Power BI Desktop application window.

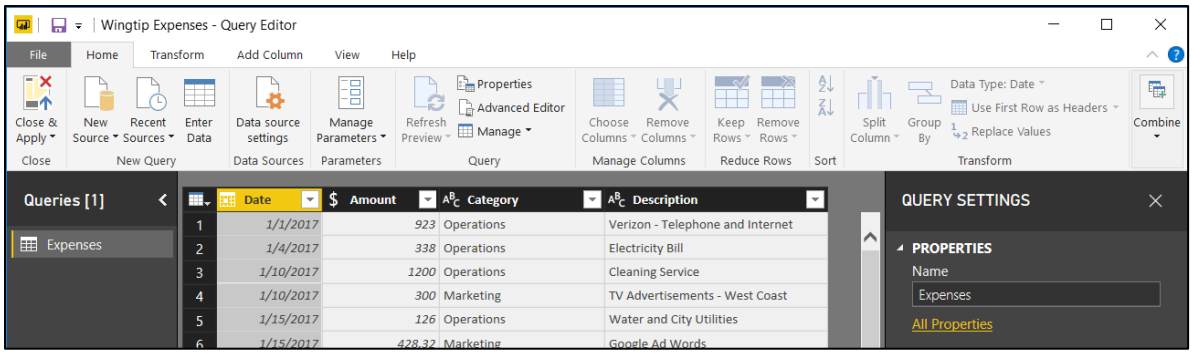
Exercise 2: Designing a Function Query to Extract Data from Multiple Files

In the following exercise, you will continue to work on the Power BI Desktop project named **Wingtip Expenses.pbix** to design a function query to implement an advanced query design where the data from multiple files can be imported into a single table.

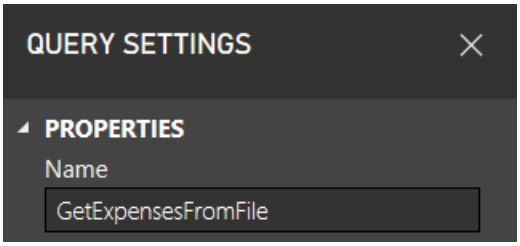
1. Convert the **Expenses** query into a function query.
   a) In the Power BI Desktop application window, click the **Edit Queries** button to navigate to the Query Editor window.



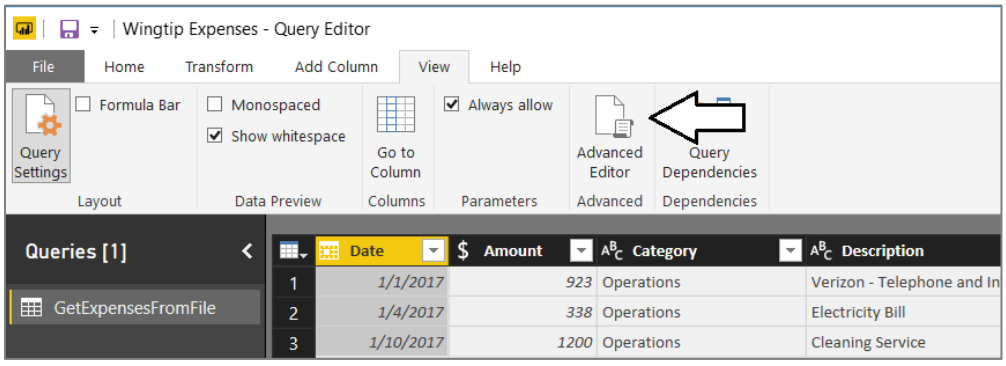
   b) At this point, your project should contain a single query named **Expenses**.



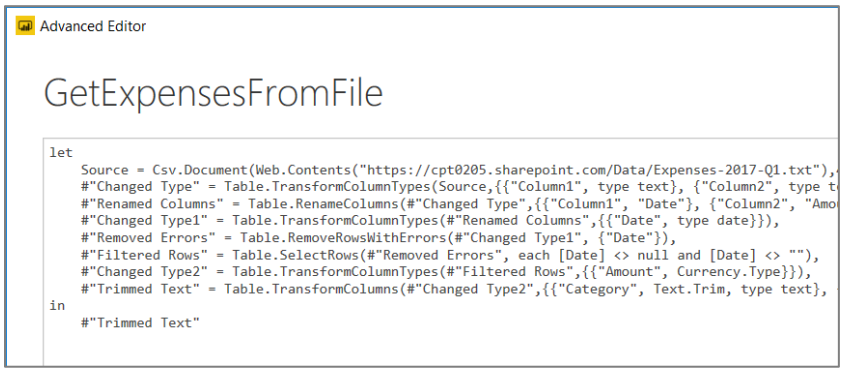
c) Rename the query from **Expenses** to **GetExpensesFromFile**.



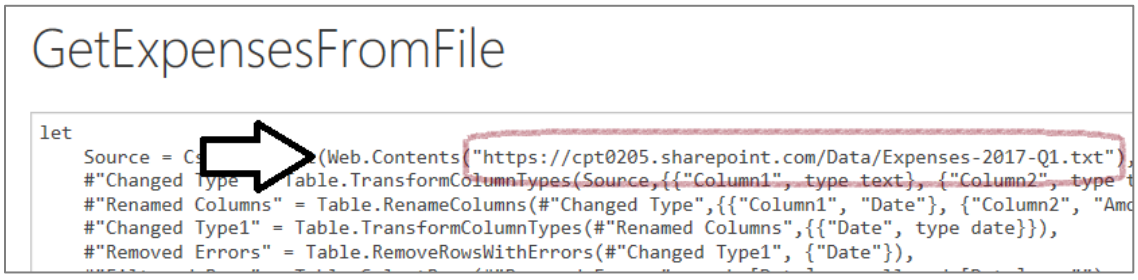
d) Click the **Advanced Editor** button from the **View** tab to display the **Advanced Editor** dialog.



e) You should see an editable view of the M code from the **GetExpensesFromFile** query in the Advanced Editor window.



f) You should also be able to see a function call to **Web.Contents** with a path to the file named **Expenses-2017-Q1.txt**.



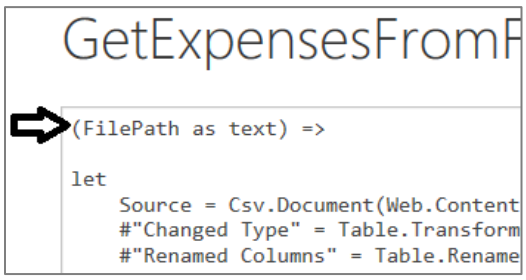
g) Place your cursor at the very beginning of the M code in front of the let statement.

h) Add the following M code to the top of the query.

```
(FilePath as text) =>
```

The way that you convert a query into a function query is by adding parentheses and the arrow operator. In this scenario, you are defining your query function to accept a single text parameter named **FilePath**.

i) At this point, your Advanced Editor dialog should match the following screenshot.

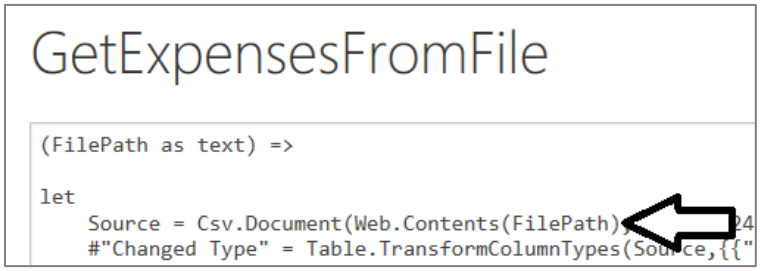


j) Move down in the M code to the line which assign a value to **Source**.

k) Update the call to **Web.Contents** to replace the hard-coded file path to the parameter named **FilePath**.

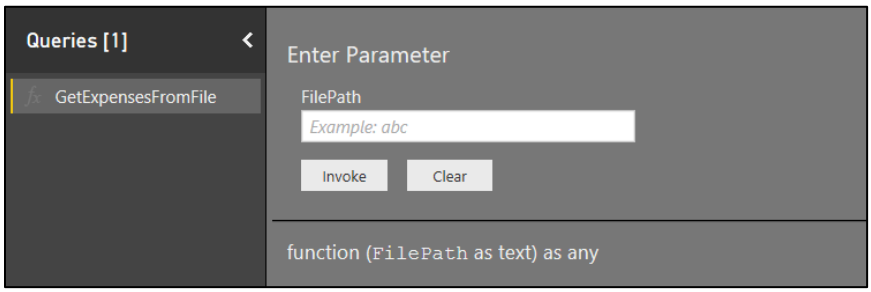
```
Source = Csv.Document(Web.Contents(FilePath),4,{0,12,24,48},null,1252),
```

l) Your Advanced Editor dialog should match the following screenshot.



m) Click the **Done** button in the Advanced Editor dialog to close it and save the changes to your M code.

n) Note that the Query Editor window now displays the **GetExpensesFromFile** query differently.

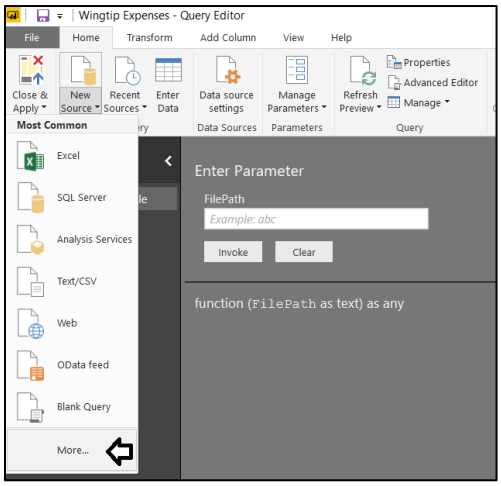


Once you convert a query into a function query, you can no longer edit it in the standard fashion. Instead, you must make any additional changes to this query by modifying its M code in the Advanced Editor window. This should not be a problem in this scenario because you did all the work to design the query logic before you converted it into a function query.

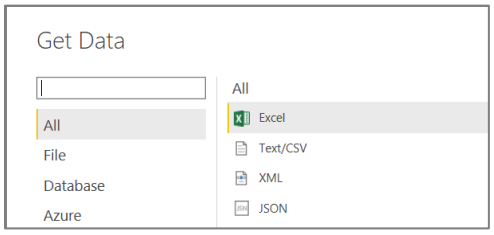
o) If you inspect the query in the **Queries** list on the left, you can see it now has an ***fx*** icon indicating the query returns a function.

p) Save your work clicking the **Save** icon in the upper, left-hand side of the Query Editor window.

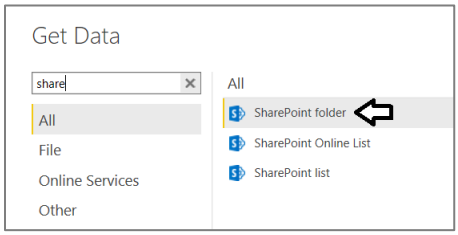
2. Create a new query named **Expenses** to call the **GetExpensesFromFile** function query.
   a) Drop down the **New Source** menu and select **More…**.



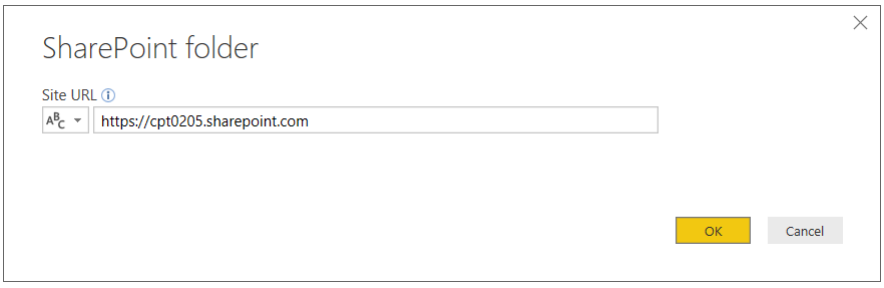
   b) When the **Get Data** dialog appears, place your cursor in the search textbox.



   c) Type in “share” to see the available SharePoint datasources.
   d) Select the **SharePoint folder** datasource.

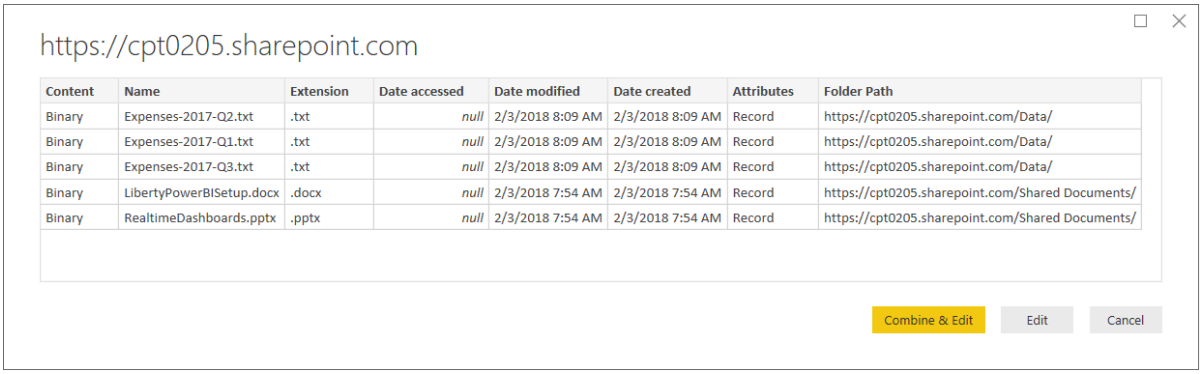


   e) Click the **Connect** button at the bottom right of the **Get Data** dialog.
   f) When promoted by the **SharePoint folder** dialog, enter the base URL to your SharePoint site and click **OK**.



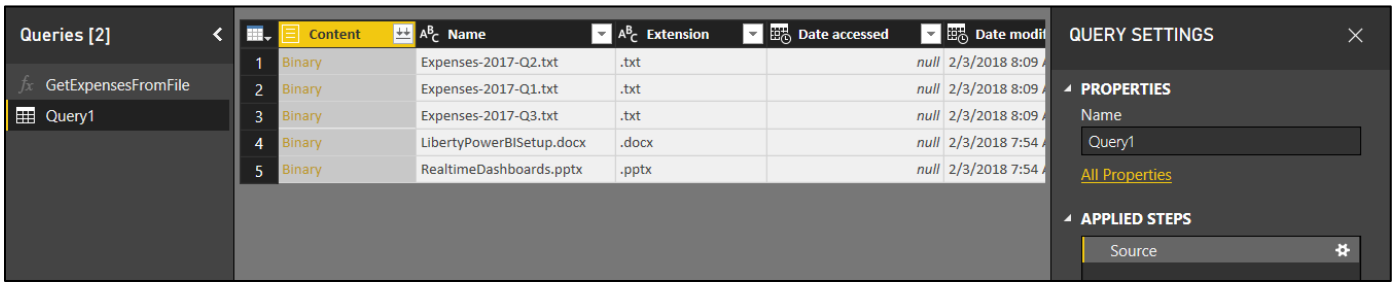
It’s a bit counter-intuitive with the **SharePoint folder** datasource. But, yes you pass the base URL to the site not the path to the library.

g) You should now see a dialog with a list of files from all document libraries in the site.

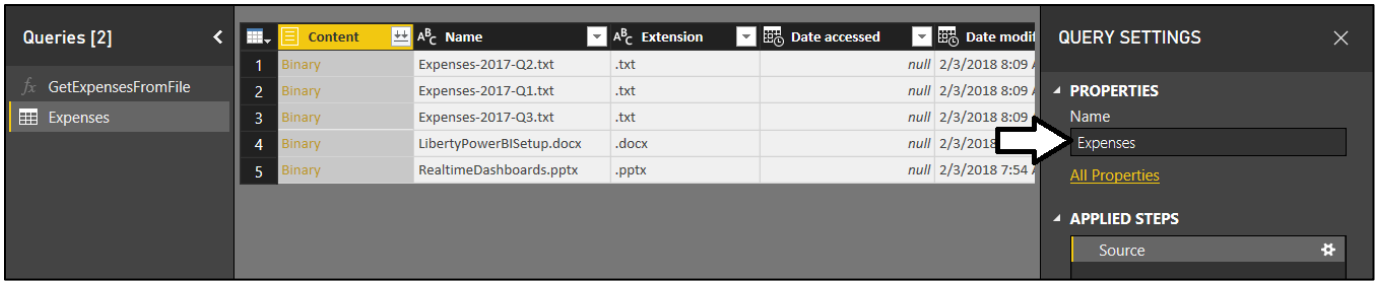


h) Click the **Edit** button to open the new query in the Query Editor window.

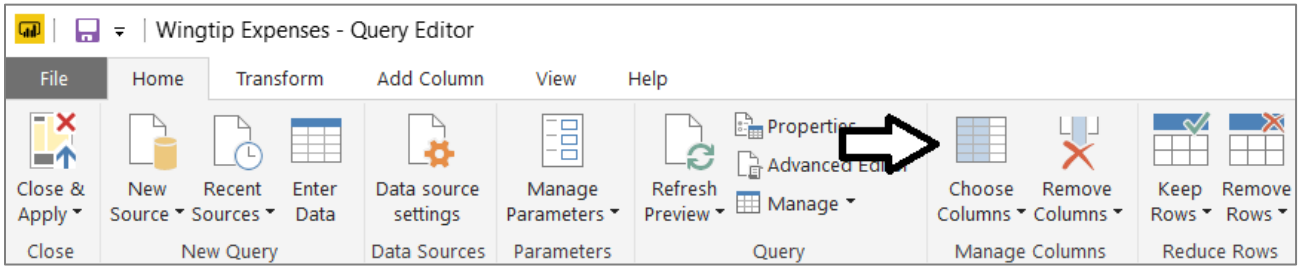
i) When the Query Editor window opens, should see a query named **Query1** as shown in the following screenshot.



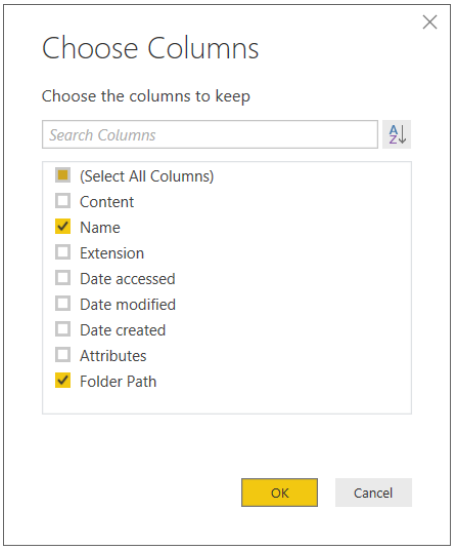
j) Rename the query from **Query1** to **Expenses**.



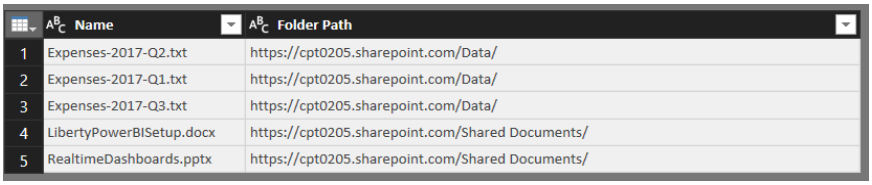
k) Click the **Choose Columns** button in the **Home** tab of the Query Editor window.



l) Unselect all columns and then select just the **Name** column and the **Folder Path** column then click **OK**.

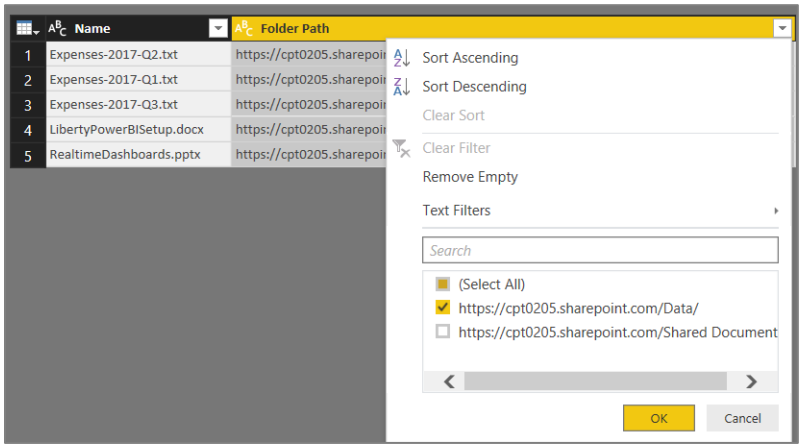


m) At this point, your query output should match the following screenshot.

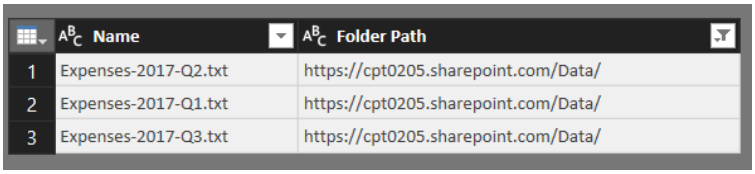


	Name	Folder Path
1	Expenses-2017-Q2.txt	https://cpt0205.sharepoint.com/Data/
2	Expenses-2017-Q1.txt	https://cpt0205.sharepoint.com/Data/
3	Expenses-2017-Q3.txt	https://cpt0205.sharepoint.com/Data/
4	LibertyPowerBISetup.docx	https://cpt0205.sharepoint.com/Shared Documents/
5	RealtimeDashboards.pptx	https://cpt0205.sharepoint.com/Shared Documents/

n) Using the dropdown menu on the right side of the **Folder Path** column, select just the folder path that ends with **/Data/**.



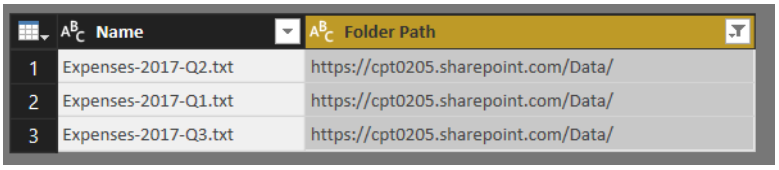
o) Now your query results should only display the three data files from the **Data** document library.



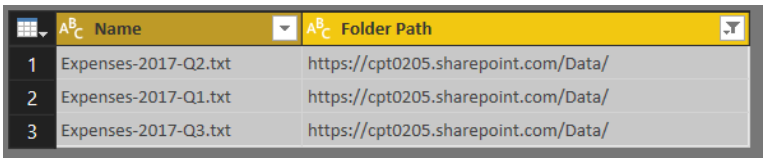
	Name	Folder Path
1	Expenses-2017-Q2.txt	https://cpt0205.sharepoint.com/Data/
2	Expenses-2017-Q1.txt	https://cpt0205.sharepoint.com/Data/
3	Expenses-2017-Q3.txt	https://cpt0205.sharepoint.com/Data/

Over the next few steps you will combine the two columns into a single column named **FilePath** by adding a **Merge Column** step. Note that you must select the column on the right first and then the column on the left second to merge the columns correctly.

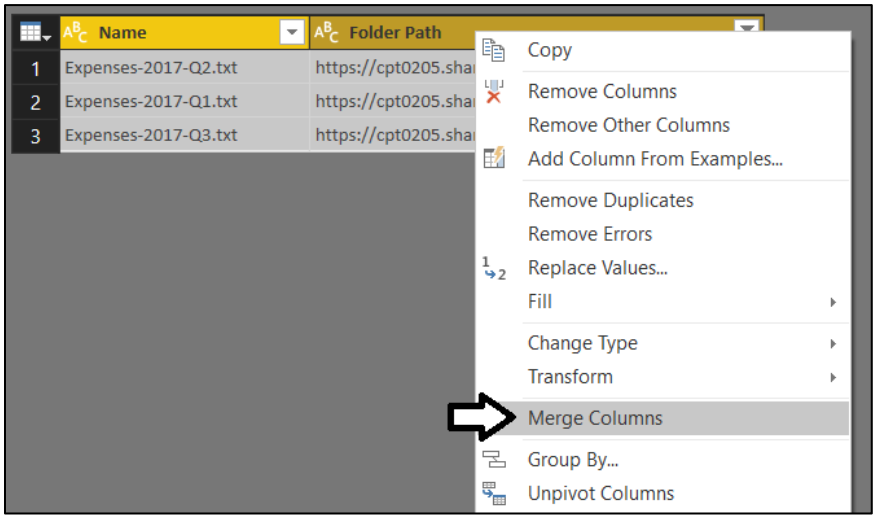
3. Add a **Merge Columns** step to create a new column named **FilePath**.
   a) Using the mouse, select the **Folder Path** column by clicking its column header.



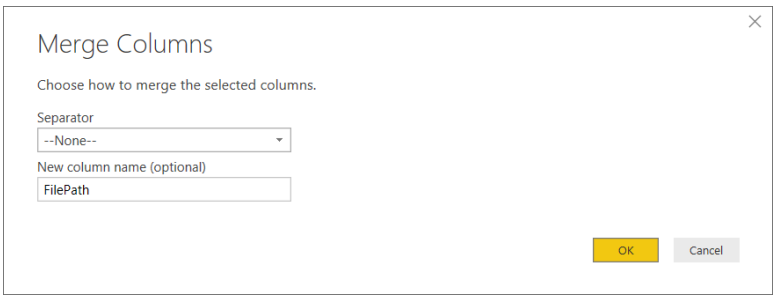
   b) With the **Folder Path** column selected, hold down the **Ctrl** key and click the **Name** column so both columns are selected.



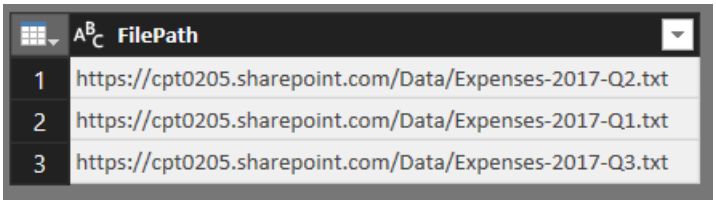
   c) Right-click the **Folder Path** column header and select the **Merge Columns** command.



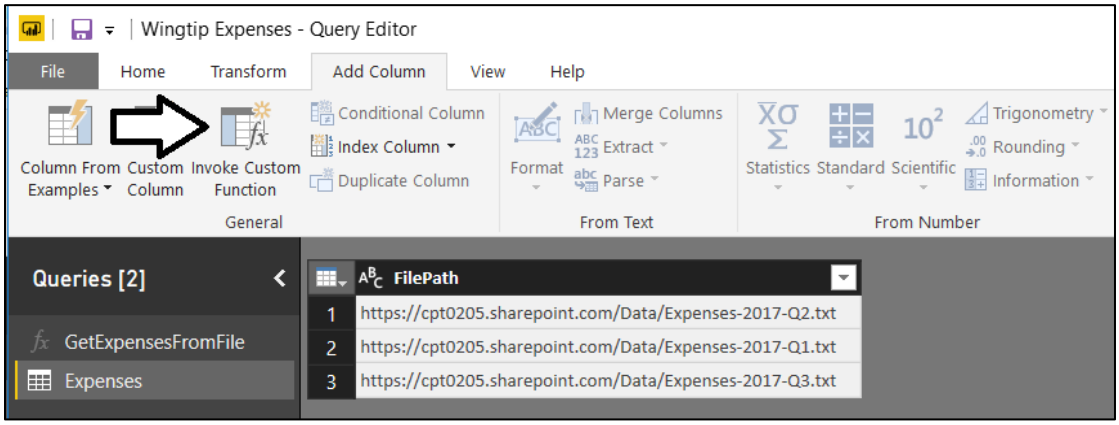
   d) In the **Merge Columns** dialog…
      i) Leave the **Separator** set to **--None--**.
      ii) Add a **New column name** of **FilePath**.
      iii) Click **OK**.



e) The query results should now show a single column named FilePath as shown in the following screenshot.

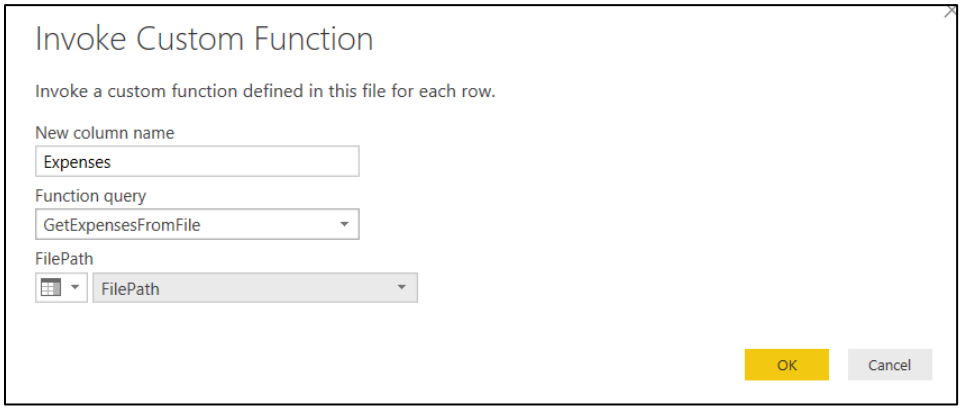


4. Modify the **Expenses** query to call the function query named **GetExpensesFromFile**.
   a) In the Query Editor window, make sure the **Expenses** query is the selected in the **Queries** list on the left.
   b) Navigate to the **Add Columns** tab.
   c) Click the **Invoke Custom Function** button in the ribbon.

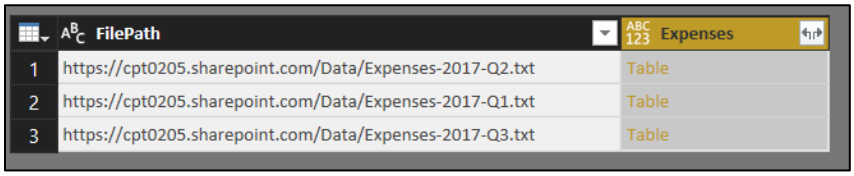


   d) In the **Invoke Custom Function** dialog…
      i) Add a **New column name** of **Expenses**.
      ii) Drop down the **Function query** menu and select **GetExpensesFromFile**.
      iii) Use the dropdown menu under the **FilePath** parameter and select **Column Name**.

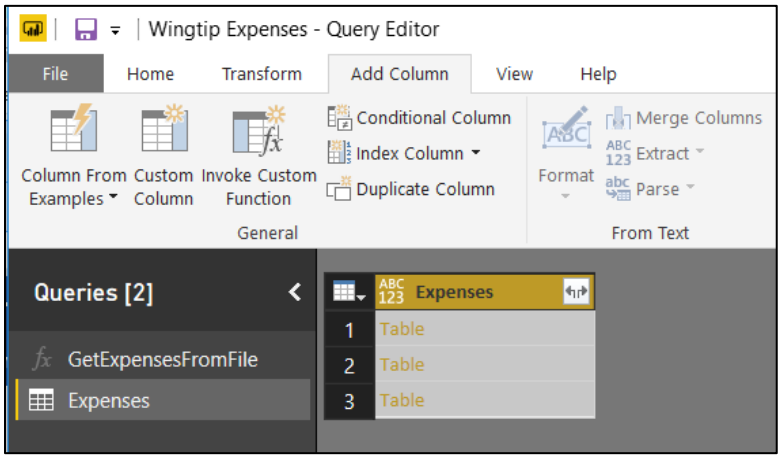
      iv) Use the dropdown to configure the **FilePath** parameter with the **FilePath** column and then click **OK**.



e) You should see a new column named **Expenses** whose values contains **Table** objects as shown in the following screenshot.



f) Remove the column named **FilePath** from the query so the query results only show the new column named **Expenses**.



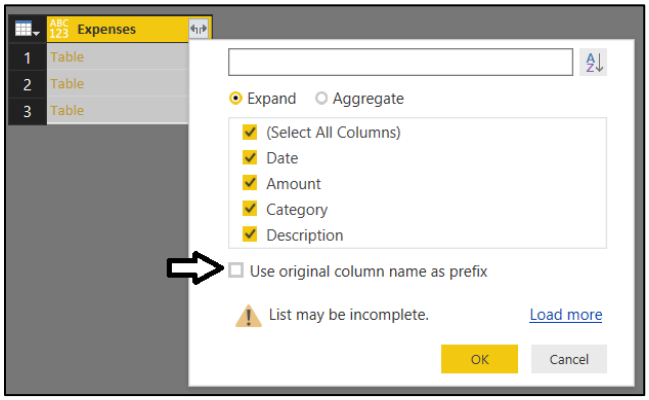
5. Expand the **Table** objects in the **Expenses** column into rows.

a) Click the **Expand** button on the right side of the **Expenses** column header.

b) Select the columns named **Date**, **Amount**, **Category** and **Description**.

c) Make sure the **Use original column name as prefix** checkbox is not selected.

d) Click **OK** to expand the rows for each **Table** object.



e) You query should now return a separate row for each expense.

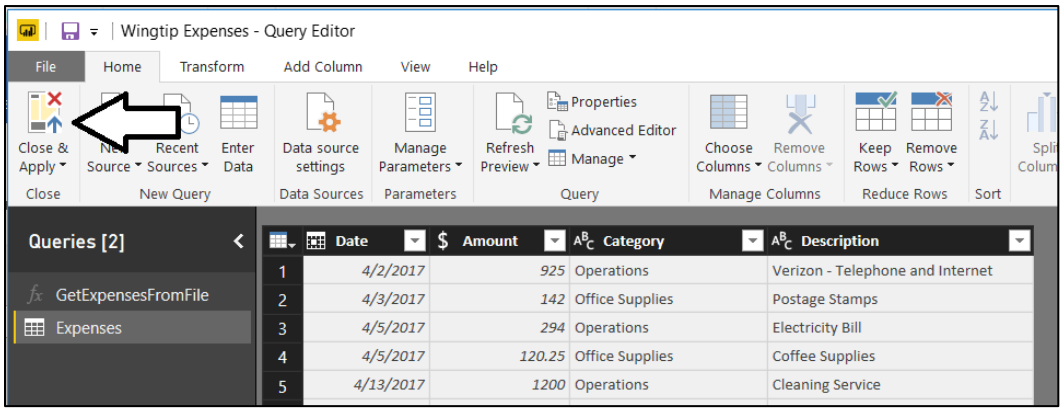
	Date	Amount	Category	Description
1	4/2/2017	925	Operations	Verizon - Telephone and Internet
2	4/3/2017	142	Office Supplies	Postage Stamps
3	4/5/2017	294	Operations	Electricity Bill
4	4/5/2017	120.25	Office Supplies	Coffee Supplies
5	4/13/2017	1200	Operations	Cleaning Service
6	4/15/2017	126	Operations	Water and City Utilities

You will notice the datatype for the columns in the table are not set to the correct types. Your next task is to fix that.

6. Update the datatypes for the columns in the **Expenses** query.
   a) Update the datatype of the **Date** column to **Date**.
   b) Update the datatype of the **Amount** column to **Fixed Decimal number**.
   c) Update the datatype of the **Category** column to **Text**.
   d) Update the datatype of the **Description** column to **Text**.

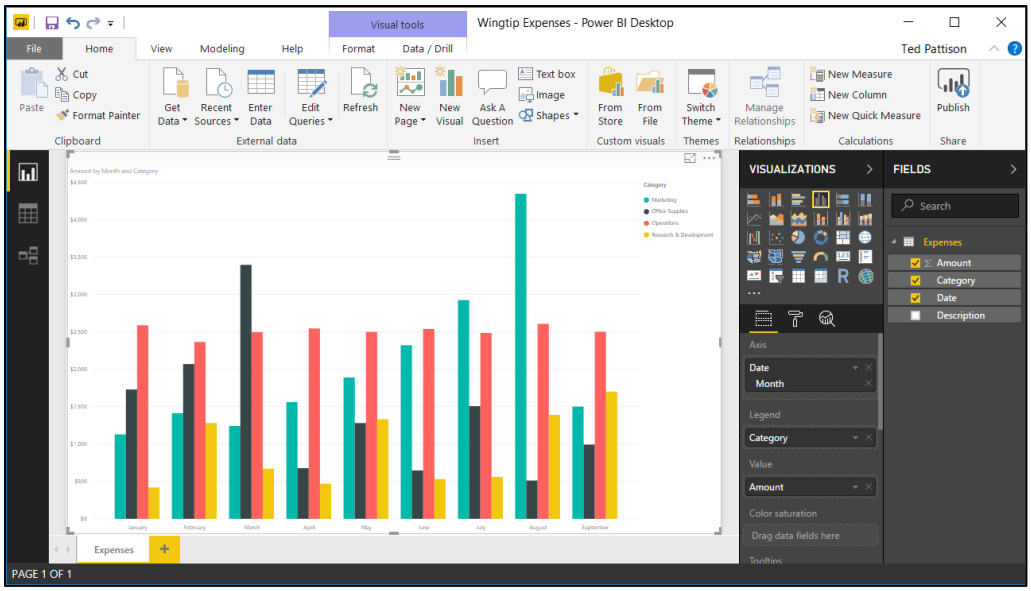
	Date	Amount	Category	Description
1	4/2/2017	925	Operations	Verizon - Telephone and Internet
2	4/3/2017	142	Office Supplies	Postage Stamps
3	4/5/2017	294	Operations	Electricity Bill
4	4/5/2017	120.25	Office Supplies	Coffee Supplies
5	4/13/2017	1200	Operations	Cleaning Service
6	4/15/2017	126	Operations	Water and City Utilities

7. Test out the query to make sure it works properly.
   a) Click the **Close and Apply** button in the ribbon to close the Query Editor window and to execute the **Expenses** query.

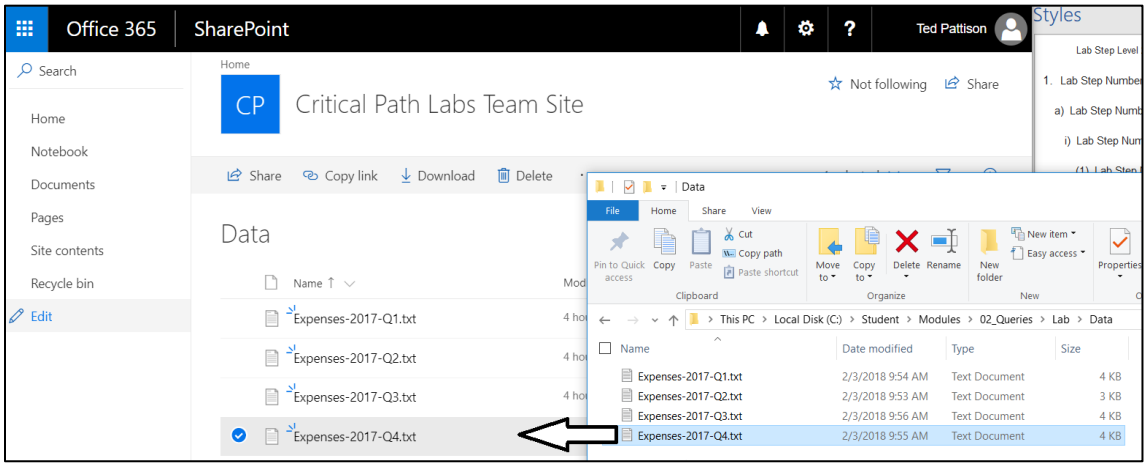


When the **Expenses** query executes, it should call the **GetExpensesFromFile** passing the name of each of the three data files you uploaded to the **Data** document library in SharePoint Online. The query should also append the rows of data from each of these three data files into a single table named **Expenses**. The report with the clustered column chart should still display correctly, but now it should show data combined from three different files to include expenses for **Q1**, **Q2** and **Q3**.

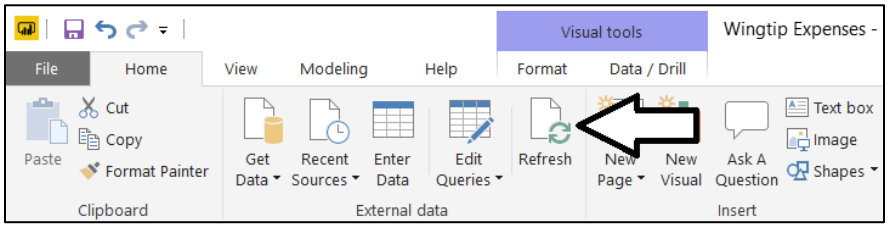
8. Inspect the **Expenses** page of the report you created in the previous exercise.
   a) The clustered column chart should now show data from January through September.



9. Add another data file to the **Data** document library in SharePoint Online.
   a) Using the browser, return to the **Data** document library in your SharePoint site.
   b) Upload the one remaining file named **Expenses-2017-Q4.txt**.

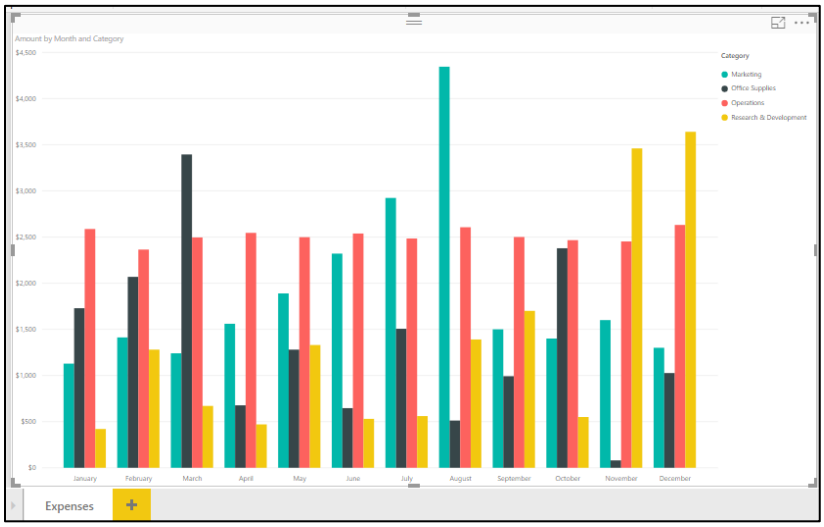


10. Refresh the data in the Power BI Desktop project named **Wingtip expenses.pbix**.
    a) Return to the **Wingtip expenses.pbix** project in Power BI Desktop.
    b) Click the **Refresh** button on the **Home** tab to re-execute the **Expenses** query.



c) When the **Expenses** query runs, the **Refresh** dialog indicates that it is also executing **GetExpensesFromFile**.

d) Once the query has completed, you should see that the column chart now displays data through December.



11. Save your work clicking the **Save** icon in the upper, left-hand side of the Power BI Desktop application window.

Congratulations. You have now designed a complex query using a function query. Keep in mind that you will be continuing to work on the Power BI Desktop project named **Wingtip Expenses.pbix** in the next lab on data modeling.

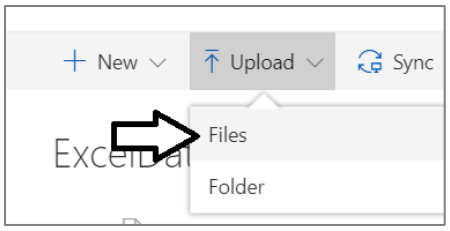
Exercise 3: Upload the Budgets.xslx Workbook File to SharePoint

In the following exercise, you will upload an Excel workbook file named **Budgets.xlsx** to your SharePoint site.

1. Navigate to your SharePoint site.
   a) You should use the same site that you used in the previous lab when you created the document library named **Data**.
2. Create new document library named **ExcelData**.
   a) Drop down the Site Actions menu and select **Add an app**.
   b) Select **Document Library** as the type of list to create.
   c) In the **Adding Document Library** dialog, add a name of **ExcelData** and click **Create**.

   d) Once the **ExcelData** document library has been created, navigate to its default view.

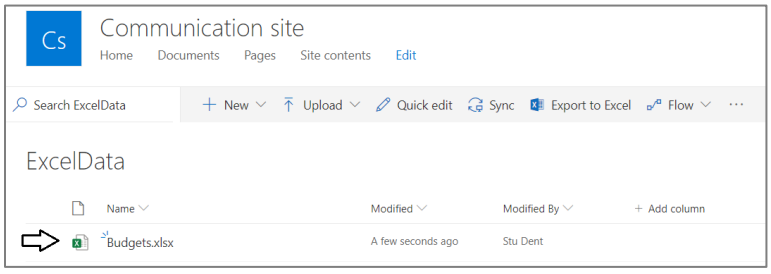
3. Upload the workbook file named **Budgets.xlsx** to the **ExcelData** document library.
   a) Click the **Upload > File** command from the SharePoint ribbon of the **ExcelData** document library to upload a document.



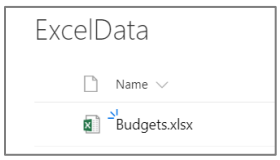
   b) Upload the workbook file named **Budgets.xlsx** which is located in the **Student** folder at the following path.

```
C:\Student\Modules\02_PowerBiDesktop\Lab\Budgets.xlsx
```

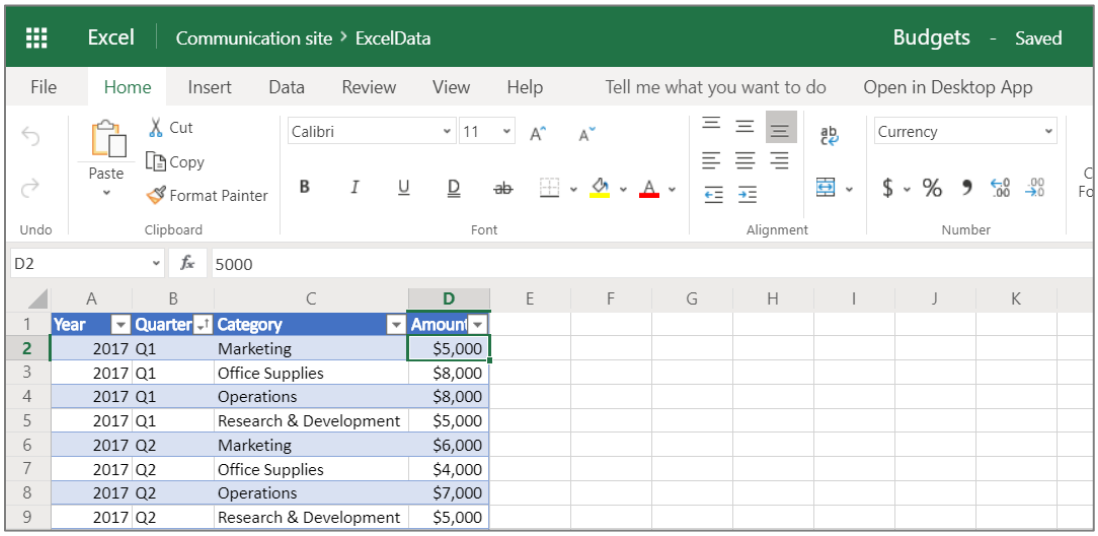
   c) Once the **Budgets.xlsx** workbook file has uploaded, you should see it in the default view of the **ExcelData** library.



4. Open the **Budgets.xlsx** workbook in Excel Online to inspect its contents.
   a) Click on the file link for **Budgets.xlsx** to open it in Excel Online.



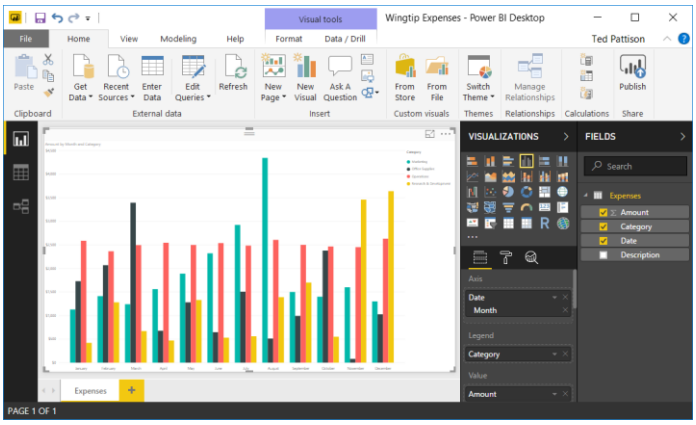
   b) You should see the workbook contains a single table with data for expense budget amounts by year, quarter and category.



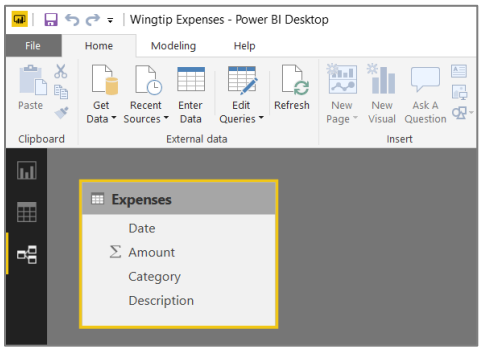
Exercise 4: Import the Data from Budgets.xslx into the Wingtip Expenses Project

In this exercise you will import budget data from the Excel workbook file named **Budgets.xlsx** into your Power BI desktop project.

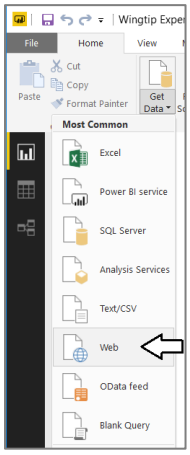
1. Open the Power BI Desktop project named **Wingtip Expenses.pbix**.
   a) Launch Power BI Desktop if it's not already running.
   b) Open the Power BI Desktop named **Wingtip Expenses.pbix** located in **Student** folder at **C:\Student\Projects**.
   c) Your project should be at the point where you finished in the lab exercises on query design.



   d) Examine the **Wingtip Expenses** project in Relationship view to confirm the project contains a single table named **Expenses**.



2. Create a new query to import the **Budgets** table from **Budgets.xlsx**.
   a) Drop down the **Get Data** menu and select the **Web** command.

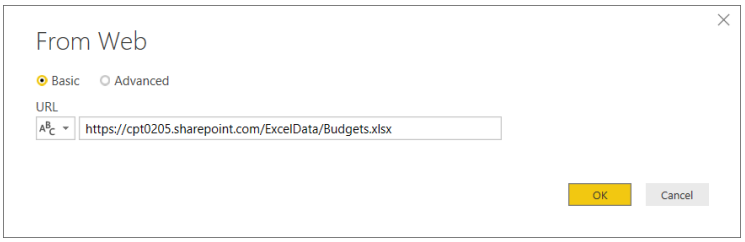


b) In the **From Web** dialog, add the path to **Budgets.xlsx**. The path should include the base URL of your SharePoint site along with the relative file path which is **/ExcelData/Budgets.xlsx**.

c) Your path should look something like the following URL

```
https://cpt0205.sharepoint.com/ExcelData/Budgets.xlsx
```

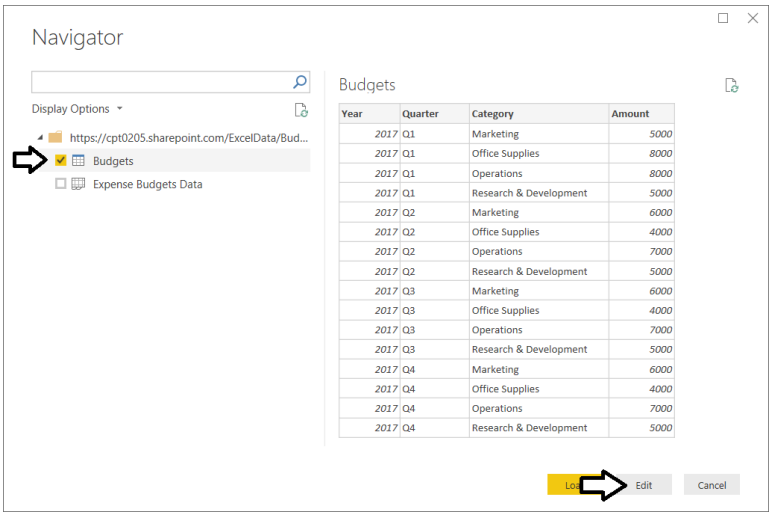
d) Once you have added the file path in the **From Web** dialog, click **OK**.



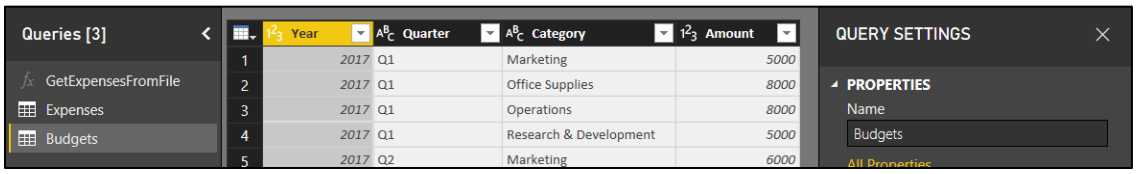
If you are prompted to login with the **Access Web content** dialog, select **Organizational account** and click **Sign in** to sign in with your credentials. Once you have signed in, click the **Connect** button.

e) You should now be prompted by the **Navigator** dialog as shown in the following screenshot.

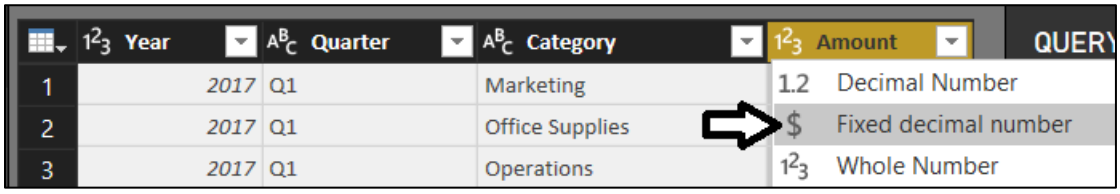
f) Select the **Budgets** table on the left and then click the **Edit** button to open the new query in the Query Editor window.



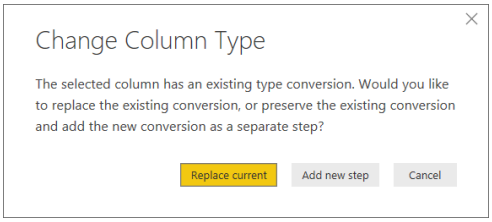
g) The query output for the new **Budgets** query should have four columns named **Year**, **Quarter**, **Category** and **Amount**.



h) Change the datatype of the **Amount** column to **Fixed Decimal number**.



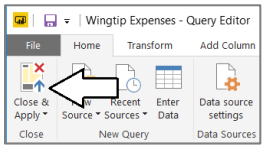
i) If you are promoted by the **Change Column Type** dialog, click the **Replace current** button to continue.



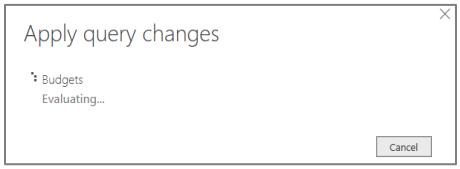
j) Verify that the **Amount** column shows a dollar sign indicating its type is **Fixed Decimal number**.

3. Execute the **Budgets** query to import the **Budgets** table into the project's data model.

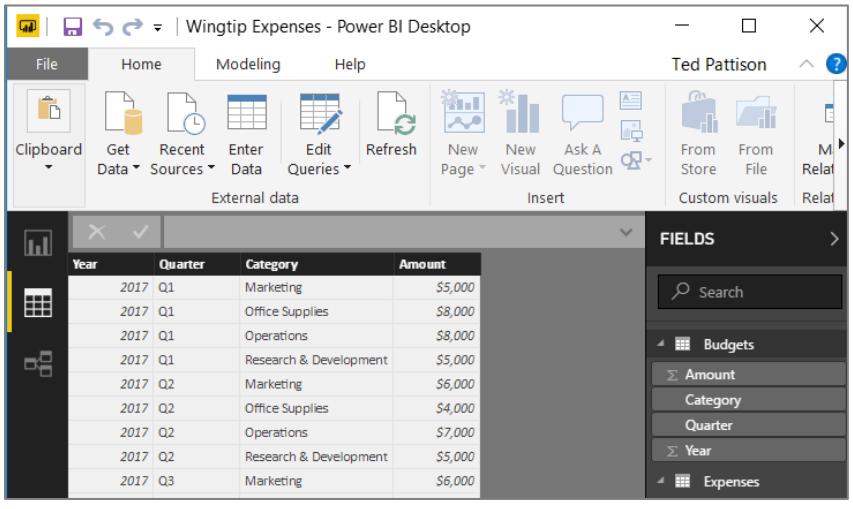
a) Click the **Close and Apply** button to close the Query Editor window and to execute the Budgets query.



b) Wait for the **Budgets** query to complete.



c) After the **Budgets** query executes successfully, you should be able to navigate to Data View and see the **Budgets** table data,

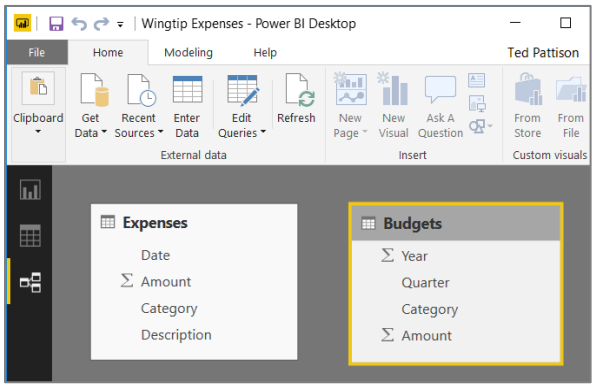


4. Save your work by clicking the **Save** icon in the upper, left-hand side of the Power BI Desktop application window.

Exercise 5: Creating a Relationship Between the Expenses Table and the Budgets Table

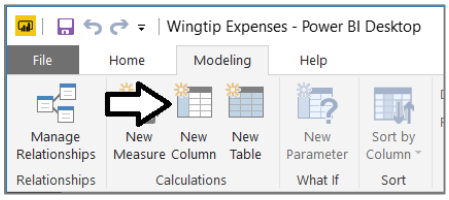
In this exercise you will create calculated columns that allow you to create a relationship between the two tables in the data model.

1. Examine the data model for the **Wingtip Expenses** project in Relationship view.
   a) Navigate to Relationship view.
   b) You should see the **Expenses** table and the **Budgets** table without any relationship between them.



Now that you have two tables in the data model, you must create a design which makes it possible to add a relationship between the **Expenses** table and the **Budgets** table. This will involve creating calculated columns in each of these tables to create key fields.

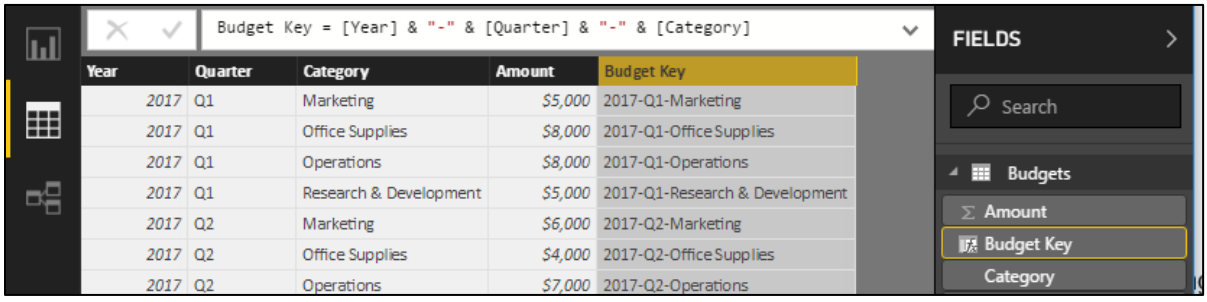
2. Extend the **Budgets** table by adding a new calculated column named **Budget Key**.
   a) Navigate to Data View and select the **Budgets** table in the **Fields** list on the right.
   b) Click the **New Column** button to create a new calculated column.



   c) Type in the following DAX expression to create a new calculated named of **Budget Year**.

```
Budget Key = [Year] & "-" & [Quarter] & "-" & [Category]
```

   d) Press the ENTER key to add the **Budget Key** calculated column to the **Budgets** table.



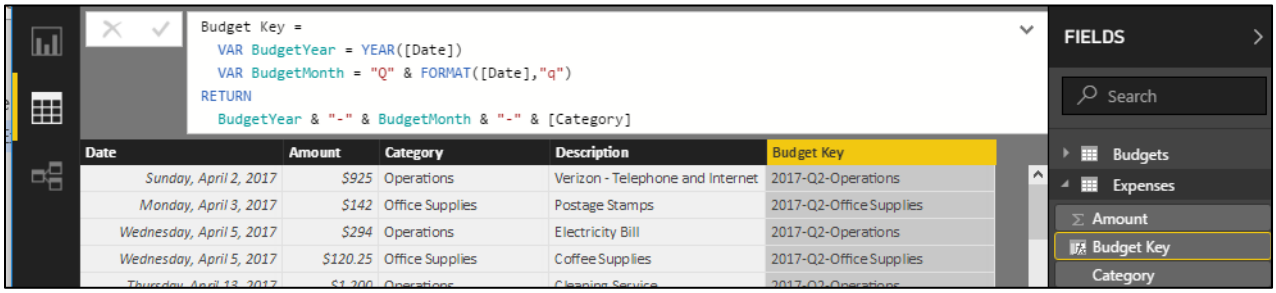
Year	Quarter	Category	Amount	Budget Key
2017	Q1	Marketing	$5,000	2017-Q1-Marketing
2017	Q1	Office Supplies	$8,000	2017-Q1-Office Supplies
2017	Q1	Operations	$8,000	2017-Q1-Operations
2017	Q1	Research & Development	$5,000	2017-Q1-Research & Development
2017	Q2	Marketing	$6,000	2017-Q2-Marketing
2017	Q2	Office Supplies	$4,000	2017-Q2-Office Supplies
2017	Q2	Operations	$7,000	2017-Q2-Operations

Now you have created a calculated column in the **Budgets** table named **Budget Key**. Your next step is to create a complimentary calculated column in the **Expenses** table which will also have the name **Budget Key**. However, you are going to get a little more involved with DAX by writing an expression that includes variables.

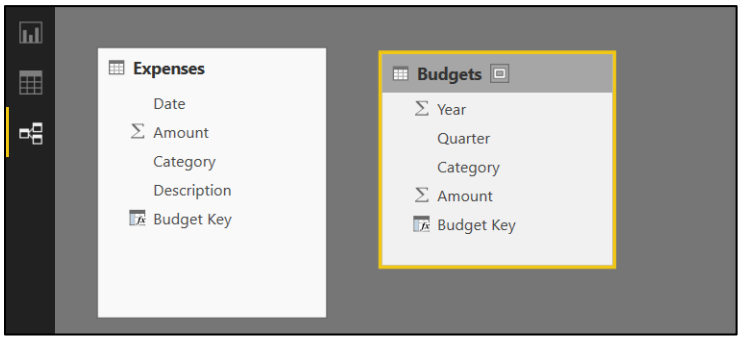
3. Create a calculated column in the **Expenses** table named **Budget Key**.
   a) Navigate to Data view and select the **Expense** table in the **Fields** list on the right.
   b) Click the **New Column** button to create a new calculated column.
   c) Enter the following DAX expression to create a new calculated column named **Budget Key**.

```
Budget Key =
  VAR BudgetYear = YEAR([Date])
  VAR BudgetMonth = "Q" & FORMAT([Date],"q")
RETURN
  BudgetYear & "-" & BudgetMonth & "-" & [Category]
```

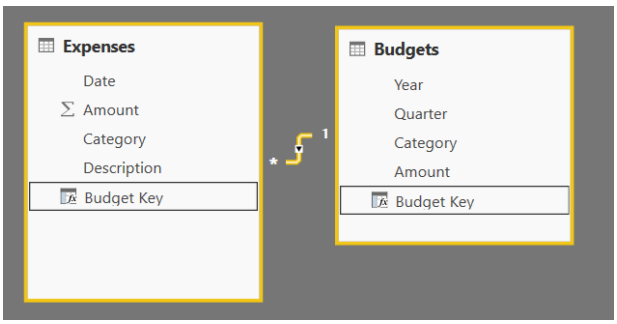
   d) Press the ENTER key to add the **Budget Key** calculated column to the **Expenses** table.



4. Create a relationship between the **Expenses** table and the **Budgets** table.
   a) Navigate to Relationship view.
   b) You should see that the **Expenses** table and the **Budgets** table now each contain a column named **Budget Key**.

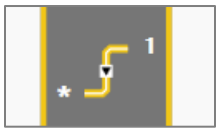


   c) Using the mouse, drag and drop the **Budget Key** column in **Expenses** on top of the **Budget Key** column in **Budgets**.
   d) You should see that you have created a relationship between these two tables as shown in the following screenshot.

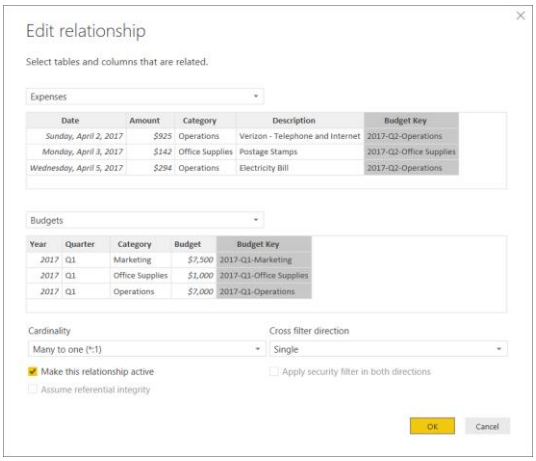


It actually doesn't matter from which table you choose to drag and drop the **Budget Key** field. It will work just fine if you do the reverse and drag and drop the **Budget Key** column in **Budgets** on top of the **Budget Key** column in **Expenses**.

5. Inspect the properties of the new relationship.
   a) Double click on the relationship line connecting the two tables to display the **Edit Relationship** dialog.



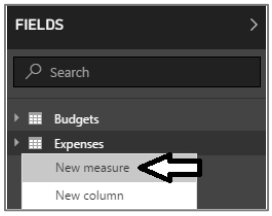
   b) Inspect the relationship properties by examining what's inside the **Edit Relationship**.



   c) There's no need to modify the relationship properties. Click **OK** close the **Edit Relationship** dialog.

Save your work clicking the **Save** icon in the upper, left-hand side of the Power BI Desktop application window.

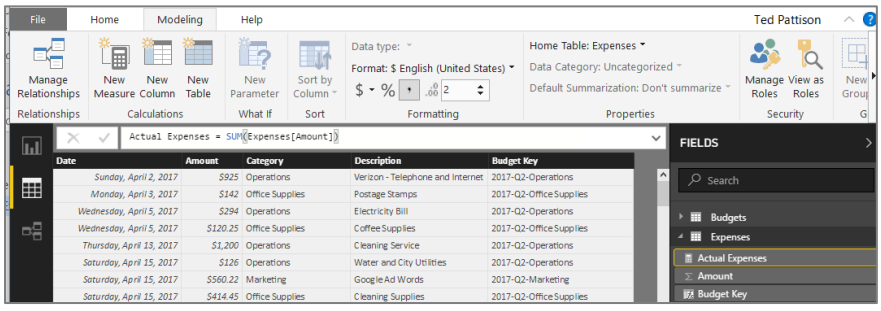
6. Add new measures to the **Expenses** table for calculating sums for expense and budget amounts.
   a) Add a new measure by right-clicking in the **Expenses** table in the **FIELDS** list and clicking **New measure**.



   b) Add a new measure named **Actual Expenses** using the following DAX expression.

```
Actual Expenses = SUM(Expenses[Amount])
```

   c) Once created, set the formatting for the named **Actual Expenses** measure for currency with 2 places after the decimal point.

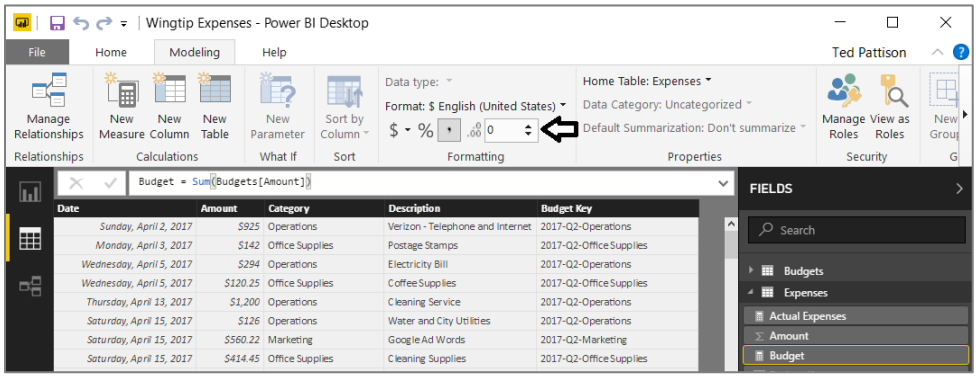


d) Add a second measure by right-clicking in the **Expenses** table in the **FIELDS** list and clicking **New measure**.

e) Create the new measure named **Budget** using the following DAX expression.

```
Budget = Sum(Budgets[Amount])
```

f) Once created, set the formatting for the named **Budget** measure for currency with **0** places after the decimal point.

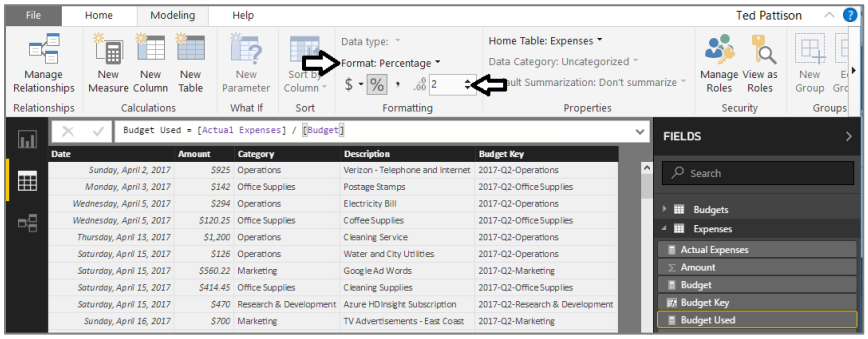


g) Add a third measure by right-clicking in the **Expenses** table in the **FIELDS** list and clicking **New measure**.

h) Create the new measure named **Budget Used** using the following DAX expression.

```
Budget Used = [Actual Expenses] / [Budget]
```

i) Once created, set the formatting for the named **Budget Used** measure for **Percentage** with **2** places of precision.



j) Add a fourth measure by right-clicking in the **Expenses** table in the **FIELDS** list and clicking **New measure**.

k) Create the new measure named **Status** using the following DAX expression.

```
Status =
IF(
    [Budget Used] > 1,
    UNICHAR(9940),
    UNICHAR(9989)
)
```

l) Press the ENTER key to add the **Status** measure to the **Expenses** table.

The **Status** measure returns a text value which means that, unlike the other measures, there's no need to format it.

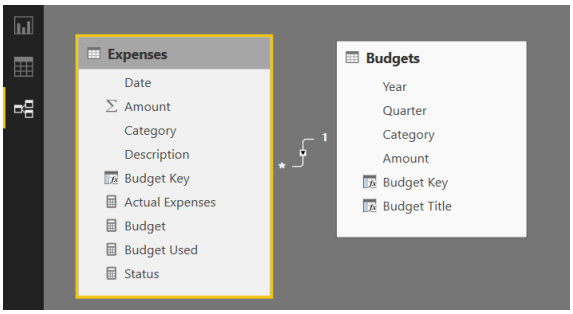
7. Add a new calculated column named **Budget Title** to the **Budgets** table.
   a) Add a new measure by right-clicking in the **Budgets** table in the **FIELDS** list and clicking **New measure**.
   b) Create a new calculated column named **Budget Title** using the following DAX expression.

```
Budget Title = [Category] & " Budget for " & [Quarter] & " of " & [Year]
```

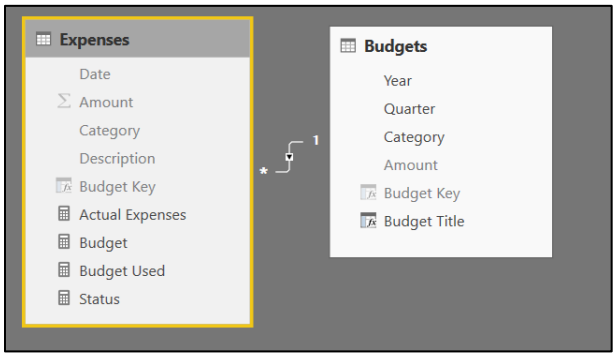
   c) Press the ENTER key to add the **Budget Title** column to the **Budgets** table.

Year	Quarter	Category	Amount	Budget Key	Budget Title
2017	Q1	Marketing	$5,000	2017-Q1-Marketing	Marketing Budget for Q1 of 2017
2017	Q1	Office Supplies	$8,000	2017-Q1-Office Supplies	Office Supplies Budget for Q1 of 2017
2017	Q1	Operations	$8,000	2017-Q1-Operations	Operations Budget for Q1 of 2017
2017	Q1	Research & Development	$5,000	2017-Q1-Research & Development	Research & Development Budget for Q1 of 2017
2017	Q2	Marketing	$6,000	2017-Q2-Marketing	Marketing Budget for Q2 of 2017
2017	Q2	Office Supplies	$4,000	2017-Q2-Office Supplies	Office Supplies Budget for Q2 of 2017
2017	Q2	Operations	$7,000	2017-Q2-Operations	Operations Budget for Q2 of 2017

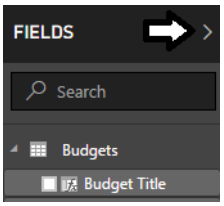
8. Hide the fields in the data model that do not need to be shown in Report view.
   a) Navigate to relationship view.
   b) Note that all the fields in both tables are visible in Report view.



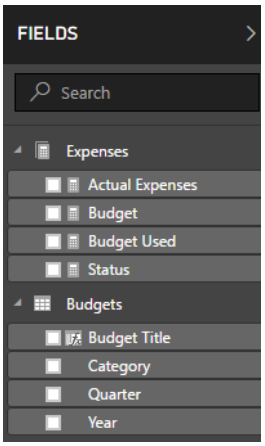
   c) Right-click on each of the following fields in the **Expenses** table and enable the **Hide in report view** setting.
      i) **Date**
      ii) **Amount**
      iii) **Category**
      iv) **Description**
      v) **Budget Key**
   d) Right-click on each of the following fields in the **Budgets** table and enable the **Hide in report view** setting.
      i) **Amount**
      ii) **Budget Key**
   e) You should be able to verify the fields that are not visible in Report view because they are greyed out in Relationship view..



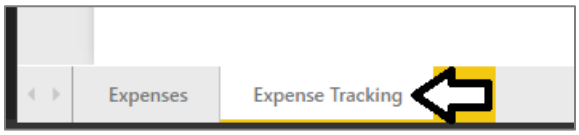
9. Inspect view of the data model in Report view.
   a) Navigate to Report view and inspect the **FIELDS** list.
   b) Refresh the view of the **FIELDS** list by clicking the button on the right with the arrow icon twice to toggle the view off and on.



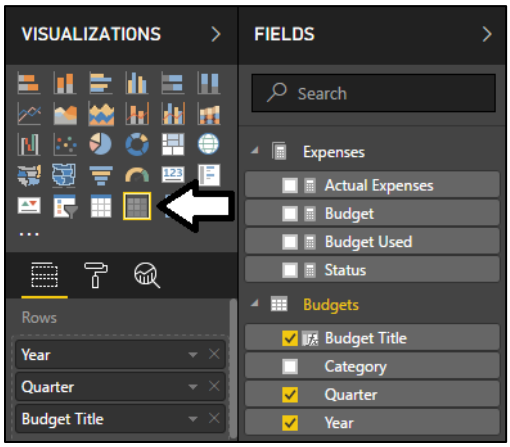
   c) After the **FIELDS** list has been refreshed, the **Expenses** table is on top because it is recognized as a Fact table.



10. Create a new report page named **Expense Tracking** that shows actual expenses compared to expense budgets.
   a) Add a new page to the report and name it **Expense Tracking**.



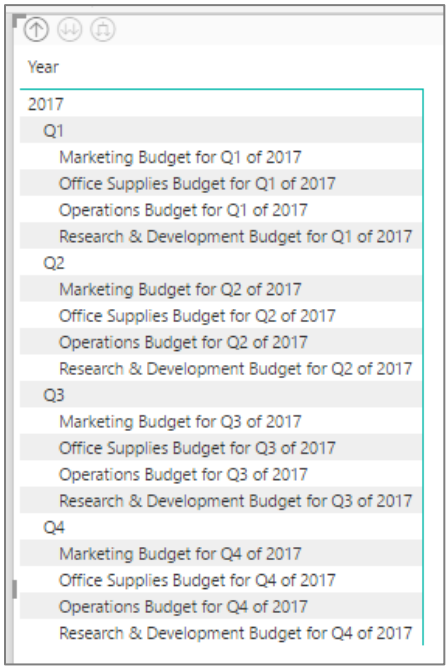
   b) Add a new **Matrix** visual to the page.
   c) Add the columns from **Budgets** table named **Year**, **Quarter** and **Budget Title** into the **Rows** well.



d) By default, the Matrix visual will only show rows for the field at the top of the **Rows** well named **Year**.

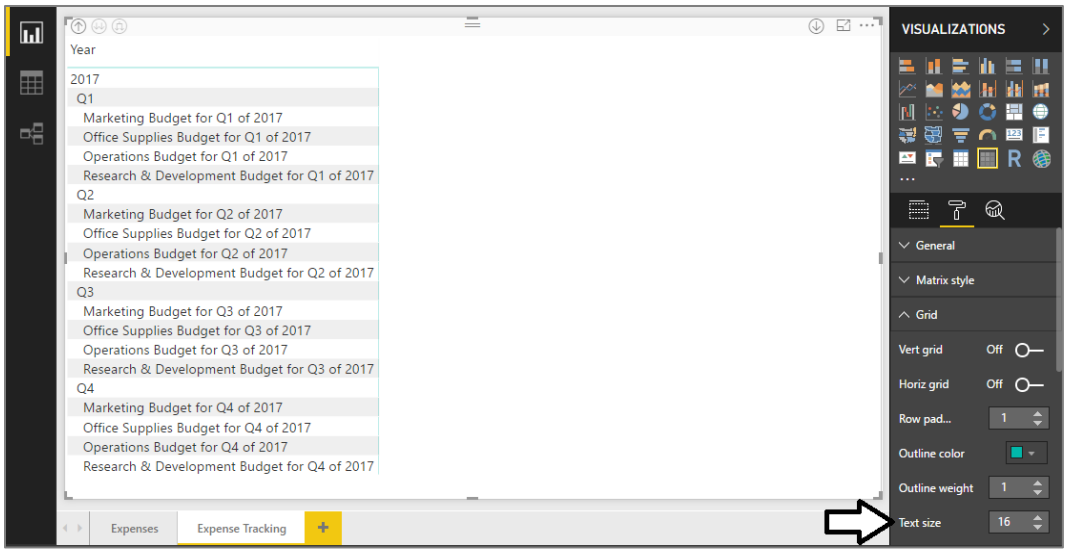
e) Click the **Expand Down** in the toolbar of the matric visual button twice to display rows for **Quarter** and **Budget Title**.

f) Your Matrix visual should now match the following screenshot.



g) Resize the Matrix visual so it takes up the entire page.

h) Update the **Text size** property in the **Grid** section to change the font size value of **16**.



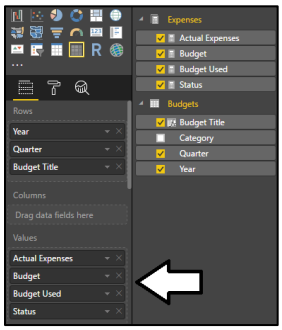
i) Add the following four fields from the **Expenses** table into the **Values** well of the **Matrix** visual.

i) **Actual Expenses**

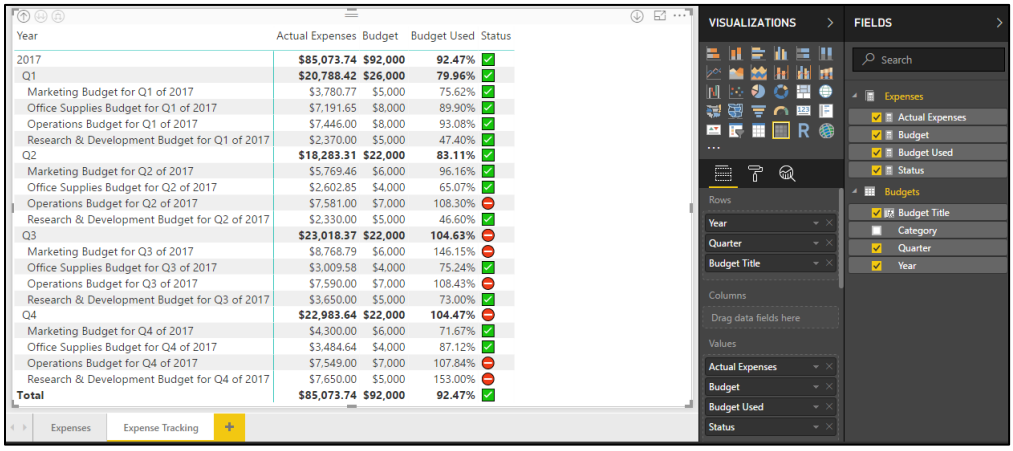
ii) **Budget**

iii) **Budget Used**

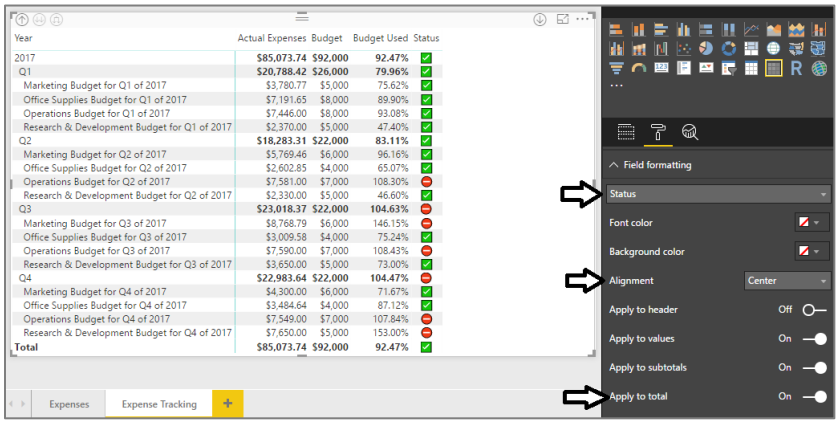
iv) **Status**



j) Your report should now match the following screenshot.



k) Adjust the **Field formatting** of the **Status** column to the UNICHAR symbol character is centered.



11. Save your work clicking the **Save** icon in the upper, left-hand side of the Power BI Desktop application window.

Congratulations. You have now finished this lab.