Lab 1: Business Analytics with Power BI

Introduction

In this lab, you will use the data analyst tool, Microsoft Power BI Desktop, to connect to data, shape it, model it, extend it with business definitions and calculations, and finally build amazing reports that will make your data fly!

Estimated time to complete this lab

2.5 hours (depends on experience)

Objectives

After completing this lab, you will be able to:

- Create a Power BI Desktop Solution.
- Create Queries based on a variety of data sources.
- Prepare a data model for reporting.
- Create an interactive dashboard layout consisting of several data visualizations.

Resources

Virtual machine (VM) Name	**Business Analytics with Power BI - Module 1**
Domain	**POWERBI-WIN10**
User name	**POWERBI-WIN10\LabUser**
Password	**P@ssw0rd1!**
Lab Files	**E:\Labs**
Asset Files	**E:\Assets**

Conditions and Terms of Use

Table of Contents

Lab: Power BI Desktop

Exercise 1: Shaping Data

In this exercise, you will use Power BI Desktop to create five queries. Four queries will source data from Microsoft SQL Server, a Microsoft Excel workbook, and a comma separated value file. A fifth query will generate a dates table without requiring a data source.

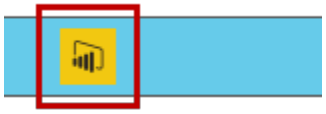
After completing this exercise, you will be able to:

- Connect to data.
- Shape data to meet business requirements.

Tasks 1: Creating a Power BI Desktop File

In this task, you will create a Power BI file.

1. To open Power BI Desktop, on the taskbar, click the **Microsoft Power BI Desktop** shortcut.



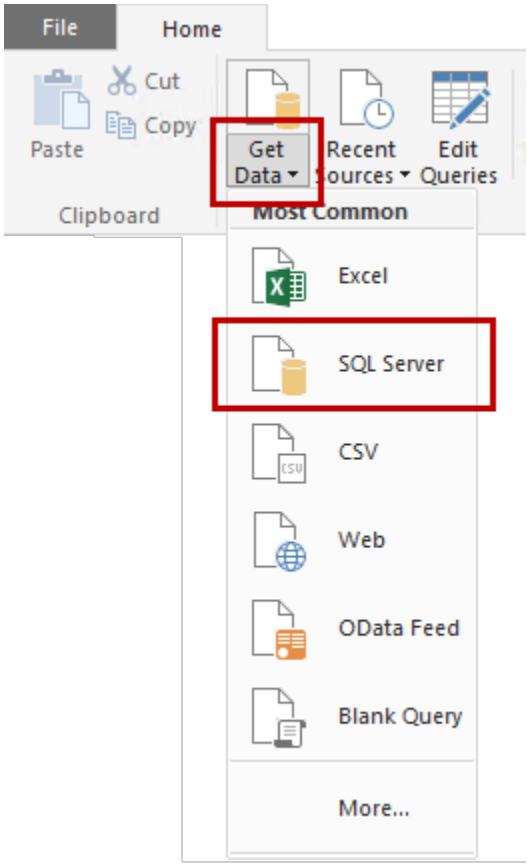
2. To save the file, click the **File** tab, and then click **Save As**.
3. In the **Save As** window, go to the **E:\Labs\Lab 1** folder.
4. In the **File Name** box, type **US Sales Analysis**.
5. Click **Save**.

NOTE: Make sure you save your progress as you progress through the lab.

Tasks 2: Importing Data from SQL Server

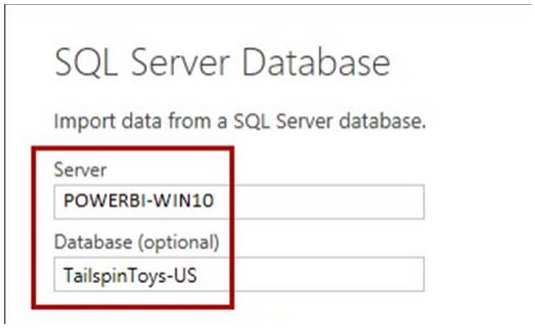
In this task, you will create a query to retrieve sales order data from the **TailspinToys-US** SQL Server database.

1. On the **Home** ribbon, in the **External Data** group, click the **Get Data** drop-down, and then click **SQL Server**.



2. In the **SQL Server Database** dialog box, in the **Server** box, type **POWERBI-WIN10**.

3. In the **Database** box, type **TailspinToys-US**.

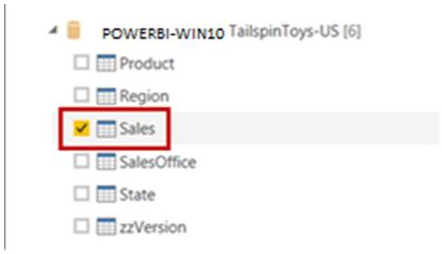


4. Leave **Import** as the connection mode, which will bring the data into Power BI Desktop. DirectQuery would allow a live connection to the database. Click **OK**.

5. If prompted to confirm the use of an unencrypted connection, click **OK**.

6. If prompted for credentials, use your Windows credentials, and then click **Connect**.

7. In the **Navigator** dialog box, select the check box for the **Sales** table.

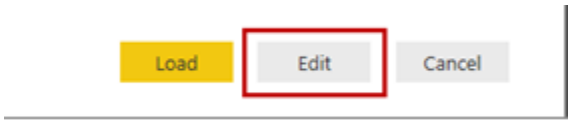


8. Review the data in the preview pane (located at the right).

   Note: The data represents the US retail sales operations, and the last two columns are foreign key columns that relate to the **Product** *and* **State** *tables.*

   In this lab, you will develop the query by introducing columns from the related tables, and by defining friendly names for the columns.

9. To develop the query, click **Edit**.



10. Notice that the **Query Editor** window opens, and that this window has its own ribbon.

    Note: This window is used to define queries that transform data.

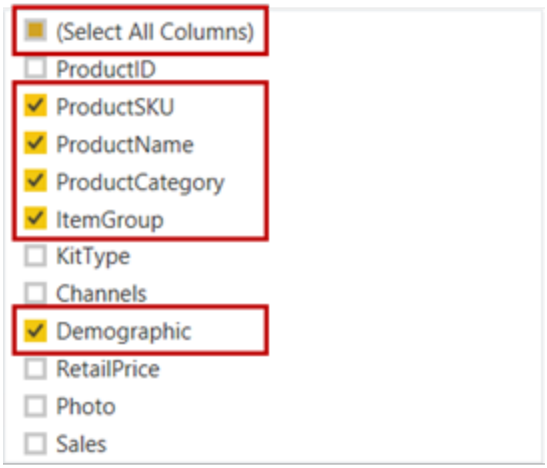
11. In the **Query Settings** pane (located at the right), in the **Name** box, notice that the query name was derived from the selected source table: **Sales**.

12. In the data pane (the large pane containing the data grid), notice that the **Product** and **State** columns (the last two columns) contain **Value** links, enabling the introduction of columns from the related tables.

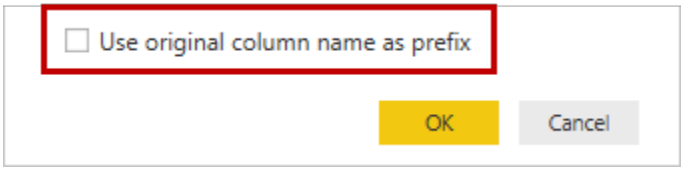
    Note: These columns were added automatically because the **CustomerStateID** *and* **ProductID** *columns in the* **Sales** *table are foreign key columns.*

13. To introduce columns from the **Product** table, in the **Product** column header, click the **Expand** icon ( ).

14. Clear the **(Select All Columns)** check box, and then select only the five columns shown in the following screenshot:



15. Clear the **Use original column name as prefix** check box.



16. Click **OK**.

17. Expand the **State** column to include only the **StateName** and **Region** columns.

Expand the **Region** column to include only the **RegionName** column.

Note: The **Region** *column represents an additional table, related to the* **State** *table.*

18. To remove unnecessary columns, first select the **OrderNumber** column header, and then while pressing the **Ctrl** key, also select the **ShipDate**, **CustomerStateID**, **ProductID**, and **PromotionCode** column headers.

19. Right-click the column selection, and then click **Remove Columns**.

20. To rename the remaining columns, first right-click the **DiscountAmount** header column, and then click **Rename**.

21. Modify the name to **Discount**, and then press Enter.

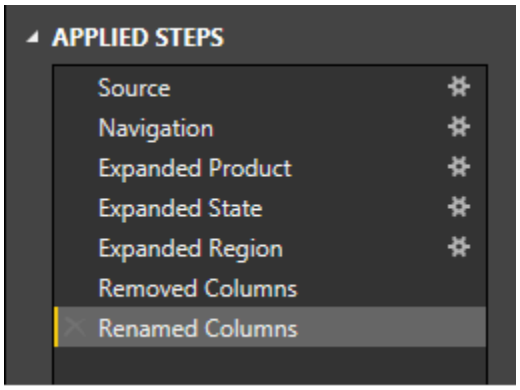
    Note: Defining concise, yet friendly names helps ensure that data visualization captions are easy to understand.

    In addition, while it is possible to give a column the same name as the table it belongs to, it is important to use column names that differ to avoid ambiguity when using Q&A (natural language querying).

22. Rename the following eight columns as shown in the following table.

Old Column Name	New Column Name
ProductSKU	Product SKU
ProductName	Product Name
ProductCategory	Product Category
ItemGroup	Product Item Group
Demographic	Product Demographic
StateName	State Name
RegionName	Region Name

23. In the **Query Settings** pane, notice the applied steps that define the logic to source and transform the query result.

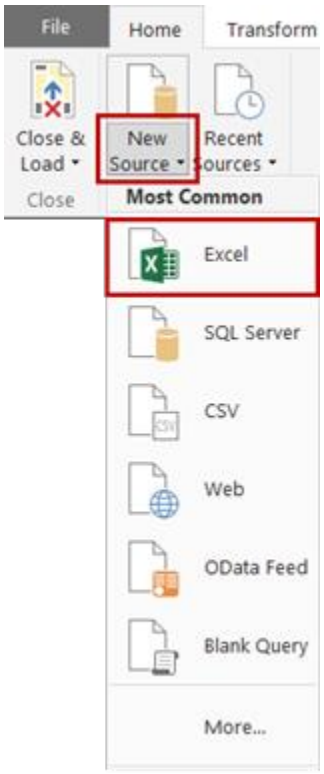


Note: You will combine the result of this query with another query to enable profit analysis.

Tasks 3: Importing Data from an Excel Workbook

In this task, you will create a second query, this time to retrieve product cost data from an Excel workbook.

1. In the **Query Editor** window, on the **Home** ribbon, in the **New Query** group, click the **New Source** drop-down, and then click **Excel**.



2. In the **Open** window, go to the **E:\Assets\Lab 1** folder.

3. Select the **ProductCost.xlsx** file, and then click **Open**.

4. In the **Navigator** dialog box, select the **ProductCost** worksheet.

5. Review the data in the preview pane.

Note: The data represents the cost of goods sold (COGS) of each product. Product cost data is stored externally from the SQL Server sales database, and will be integrated with the **Sales** *query, developed in the previous task, to enable profit analysis.*

6. To load the query, click **OK**.
7. In the **Query Editor** window, in the **Query Settings** pane, in the **Applied Steps** list, notice that four steps exist, applying default transformations of the Excel data.
8. Use the step instructions in the previous task to remove the **Product** column.

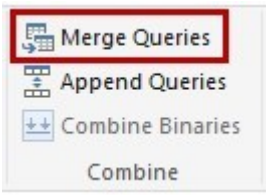
   Note: The **Product** *column is not required, because this data has already been retrieved by the* **Sales** *query.*

Tasks 4: Combining Two Queries

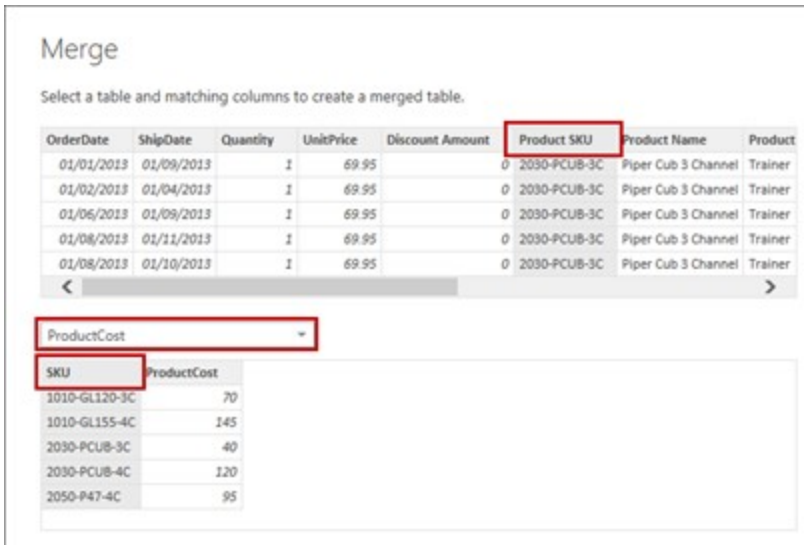
In this task, you will merge the **ProductCost** query with the **Sales** query to define the **Cost** column.

1. In the **Queries** pane (located on the left), select the **Sales** query.

2. On the **Home** ribbon, in the **Combine** group, click **Merge Queries**.



3. In the **Merge** dialog box, for the **Sales** table, select the **Product SKU** column header.
4. In the drop-down list for the table to join to, select the **ProductCost** query.
5. Then, in the lower grid, select the **SKU** column header.



6. If the **Privacy Levels** dialog box appears, for the **microsoftbi** data connection, in the adjacent drop-down list, select **Organizational**.

Note: A privacy level specifies an isolation level that defines the degree to which one data source will be isolated from other data sources. An **Organizational** *data source limits the visibility of a data source to a trusted group of people, and includes other organizational data sources.*

7. Configure the privacy level for the **e:** file location to **Organizational** as well.

Note: As it now stands, the **Sales** *query can be combined with the* **ProductCost** *query.*

8. Click **Save**.

9. For the **Join Kind** leave the default (Left Outer), which means we want all sales, even if they do not have a matching product cost.

10. Located at the bottom-left corner of the **Merge** dialog box, notice that all 269 112 rows have matched.

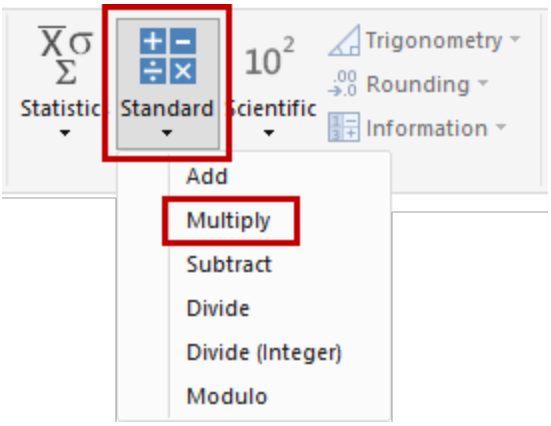
11. Click **OK**.

12. In the data pane, at the end of the columns, notice the addition of a new column named **NewColumn**.

13. Expand the **NewColumn** column to include only the **ProductCost** column.

14. To calculate sales cost, first select the **Quantity** column header, and while pressing the **Control** key, select the **ProductCost** column header.

15. On the **Add Column** ribbon, in the **From Number** group, click the **Standard** drop-down, and then click **Multiply**.



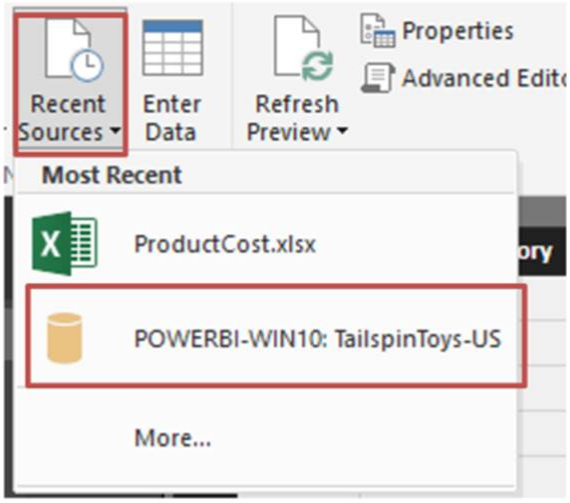
16. Rename the **Inserted Multiplication** column as **Cost**.

17. Remove the **ProductCost** column.

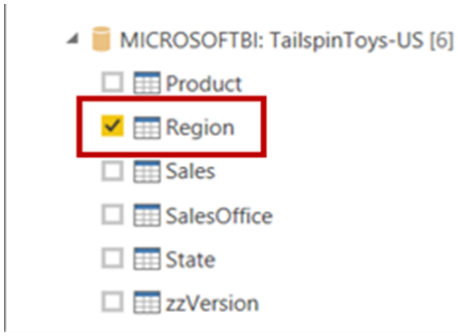
Tasks 5: Importing Additional Data from SQL Server

In this task, you will create an additional query from the **TailspinToys-US** SQL Server database to retrieve **Region** data.

1. On the **Home** ribbon, in the **New Query** group, click the **Recent Sources** drop-down, and then click **POWERBI-WIN10: TailspinToys-US**.



2. In the **Navigator** dialog box, select the **Region** table.



3. Review the data in the preview pane.

   Note: The data represents all regions, and this data will be used to add a query that will enable a many-to-many relationship between sales managers (who can be jointly responsible for multiple regions) and sales.

   The manager data will be loaded in the next task.

4. To load the query, click **OK**.

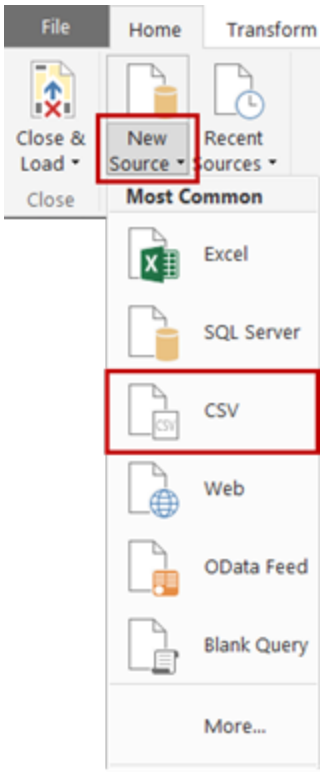
5. Right-click the **RegionName** column header, and then click **Remove Other Columns**.

6. Rename the **RegionName** column as **Region Name**.

Tasks 6: Importing Data from a CSV File

In this task, you will create a query from a CSV file to retrieve manager data.

1. In the **Query Editor** window, on the **Home** ribbon, in the **New Query** group, click the **New Source** drop-down, and then click **CSV**.

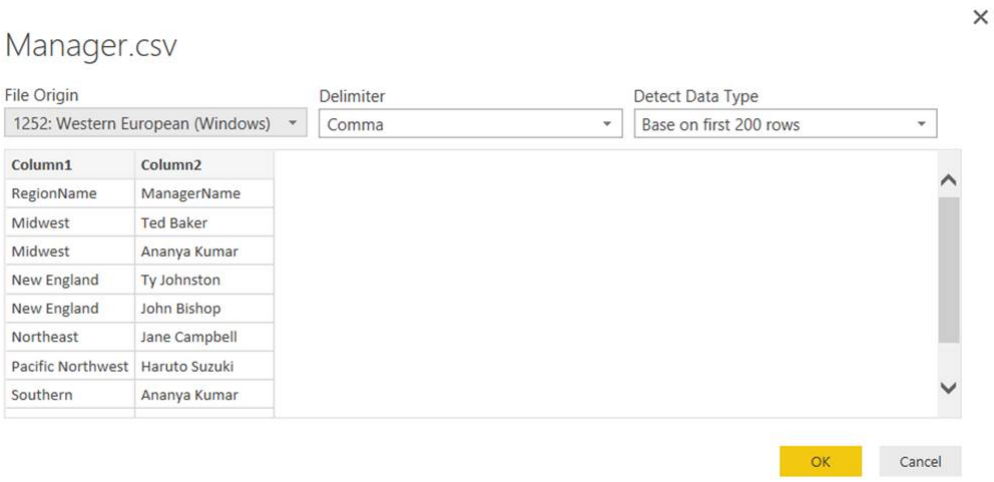


2. In the **Open** window, go to the **E:\Assets\Lab 1** folder.

3. Select the **Manager.csv** file, and then click **Open**.

   Note: The data represents sales managers and their assignment to one, or possibly more, sales regions.

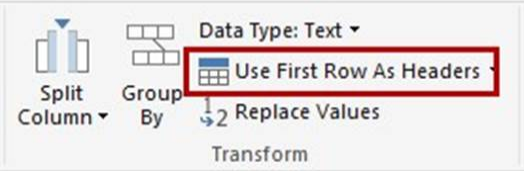
4. Leave the defaults for the file configuration window:



5. Click **OK**.

6. Notice that the first row contains the column names.

7. To promote the first row values as column headers, on the **Home** ribbon, in the **Transform** group, click **Use First Row as Headers**.



8. Rename the columns as **Region Name** and **Manager Name**.

9. Notice that some sales regions have two managers assigned, for example **Ananya Kumar** is assigned to both the **Midwest** and **Southern** sales regions.

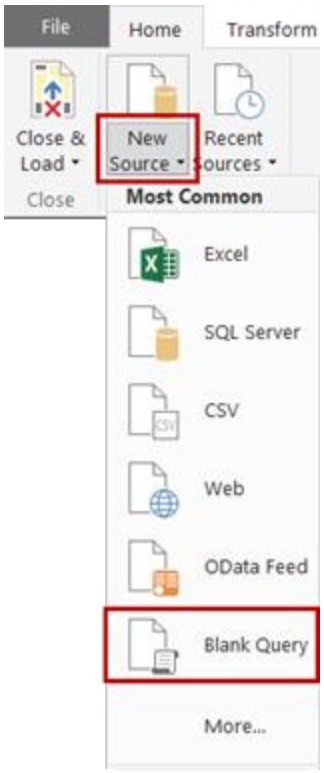
	Region Name	Manager Name
1	Midwest	Ted Baker
2	Midwest	Ananya Kumar
3	New England	Ty Johnston
4	New England	John Bishop
5	Northeast	Jane Campbell
6	Pacific Northwest	Haruto Suzuki
7	Southern	Ananya Kumar
8	Southwest	Ty Johnston
9	Southwest	Carmen Carrington

Note: Analyzing sales statistics by sales manager will be achieved by a many-to-many relationship that you will configure in the next exercise.

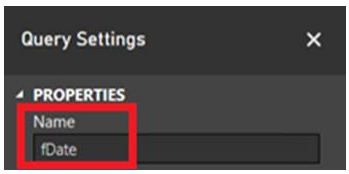
Tasks 7: Creating Data with the Advanced Editor

In this task, you will create a query by using a predefined script to generate date data.

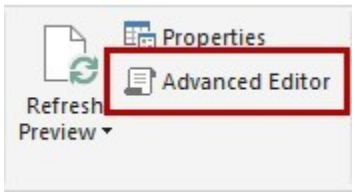
1. In the **Query Editor** window, on the **Home** ribbon, in the **New Query** group, click the **New Source** drop-down, and then click **Blank Query**.



2. In the **Query Editor** window, in the **Query Settings** pane, in the **Name** box, replace the text with **fDate**, and then press Enter.



3. To define the query, on the **Home** ribbon, click **Advanced Editor**.



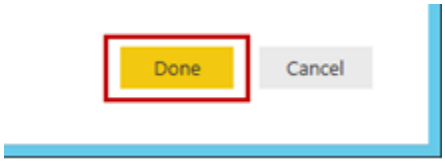
4. In the **Advanced Editor** window, remove all text from inside the query box.

5. Use **Notepad** to open the **E:\Assets\Lab 1\DateGenerationFunction.txt** file. Copy the entire content of the file, and then paste it into the query box.

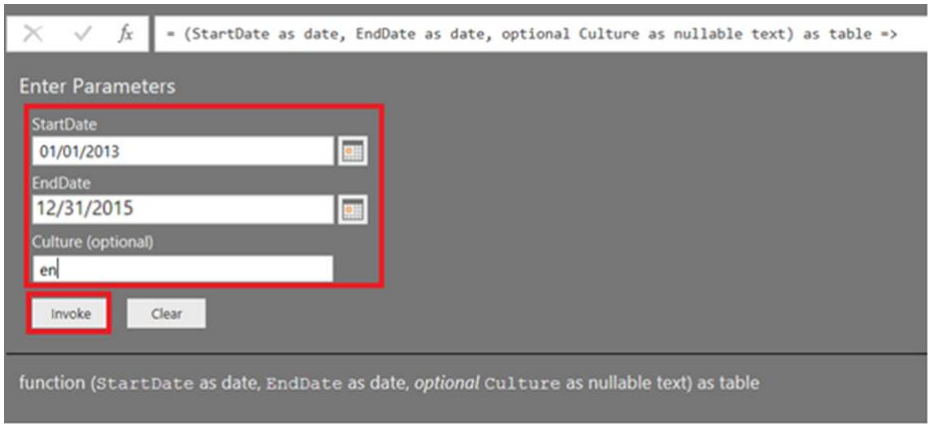
Note: It is not important to understand the detail of this script. In essence, the script defines a function that accepts start and end dates, and an optional culture. When invoked, this function will produce a query result defining date data.

The script has been adapted from a blog post published by Matt Masson titled "Creating a Date Dimension with a Power Query Script".

6. Click **Done**.



7. In the **Enter Parameters** dialog box, type the following values:



Enter Parameters

StartDate

01/01/2013

EndDate

12/31/2015

Culture (optional)

en

Invoke

Clear

function (StartDate as date, EndDate as date, *optional* Culture as nullable text) as table

*Note: The date values must be entered in US format (mm/dd/yyyy), because the lab virtual machine regional settings are set to **United States**.*

The culture value can be set to any valid culture (such as fr, fr-FR or esES) and is used to produce localized month names. You may enter the culture code for your region.

Culture names are documented at MSDN within the national language support (NLS) API Reference.

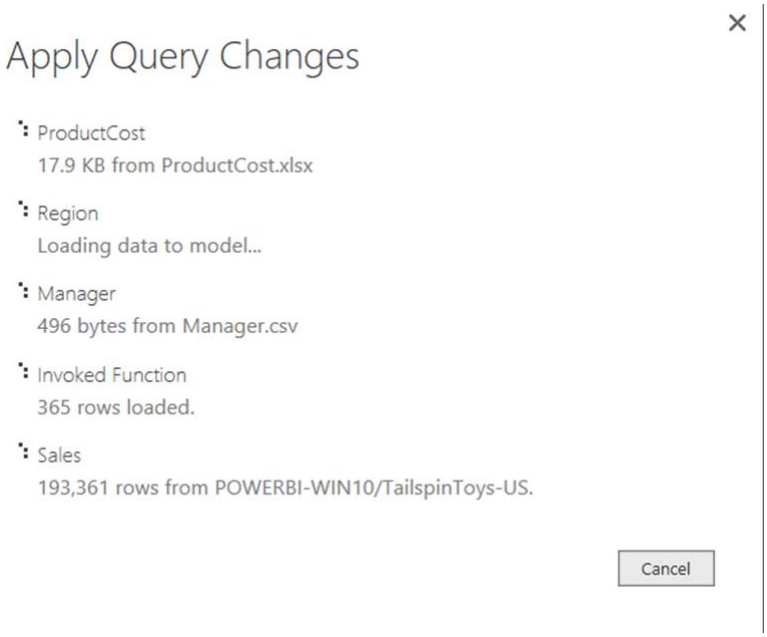
8. Click **Invoke**.

9. Change the resulting query's name from **Invoked Function** to **Date**.

10. To close the **Query Editor** window, on the **File** ribbon tab, select **Close**.

11. Notice the warning that notifies you that the new queries have not been applied.

12. Click **Apply Changes** and wait for the completion of the data load.



Exercise 2: Creating the Data Model

In this exercise, you will manage the model relationships to connect all five queries, hide tables and columns that are not appropriate to use in reports, sort columns, and create columns and measures to enable profitability analysis.

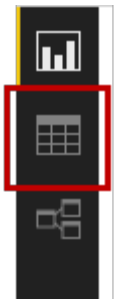
After completing this exercise, you will be able to:

- Create a data model by properly establishing relationships between tables;
- Extending your model with additional metadata such as data categorization, custom attribute sorting and hierarchies;
- Add calculations to your model.

Tasks 1: Managing Relationships

In this task, you will manage the model relationships to connect all queries.

1. To toggle to Data view, on the left side, click **Data**.



2. Notice the **Fields** pane (located on the right), which allows selecting a table to view its data.

3. On the **Data Tools | Modeling** contextual ribbon, in the **Relationships** group, click **Manage Relationships**.

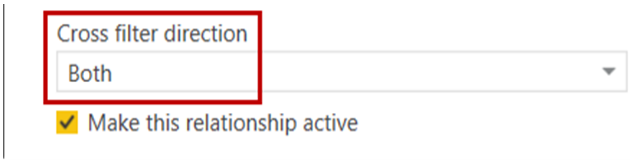
4. In the **Manage Relationships** dialog box, notice that two relationships were automatically detected and created.

Note: Relationships can be automatically detected when column names match and their data types are compatible.

The two relationships are required to enable many-to-many analysis between sales managers and sales.

5. Select the relationship between the **Manager** and **Region** tables, and then click **Edit**.

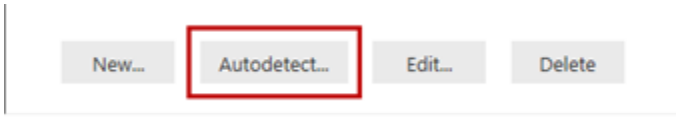
6. In the **Cross filter direction** drop-down list, ensure that **Both** is selected.



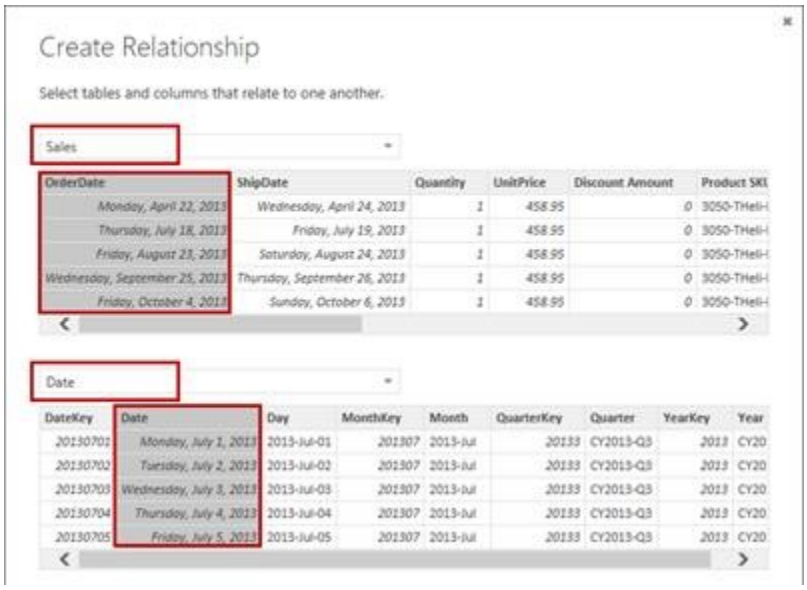
7. Click **OK**.

8. Repeat the last steps to similarly configure the relationship between the **Sales** and **Region** tables.

9. To discover any additional relationships, click **Autodetect**.



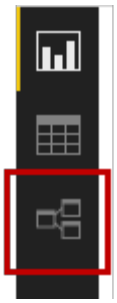
10. When informed that a new relationship was found between Sales and ProductCost, click **Close**.
11. To create a relationship, click **New**.
12. In the **Create Relationship** dialog box, in the first drop-down list, select the **Sales** table.
13. In the second drop-down list, select the **Date** table.
14. In the **Sales** table, select the **OrderDate** column.
15. In the **Date** table, select the **Date** column header.



16. Click **OK**.
17. Verify that four relationships are defined (the order might be different).

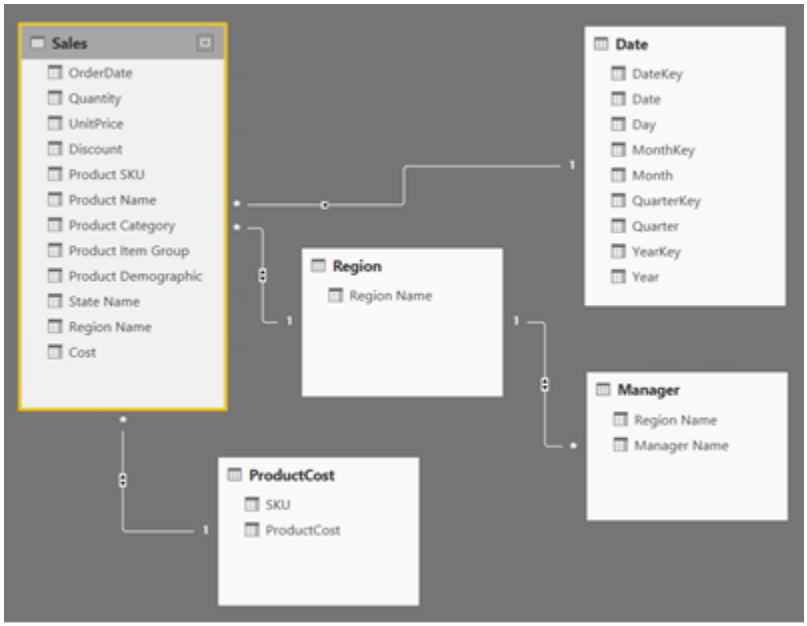
Active	From: Table (Column)	To: Table (Column)
☑	Manager (Region Name)	Region (Region Name)
☑	Sales (Region Name)	Region (Region Name)
☑	Sales (Product SKU)	ProductCost (SKU)
☑	Sales (OrderDate)	Date (Date)

18. Click **Close**.
19. To toggle to the **Relationships** view, at the left side, click **Relationships**.



20. Review the relationships in the diagram, noticing the annotations for each relationship indicating type and direction (1 and *).

21. Resize and reposition the tables to enable a clear understanding of the data model.



Tasks 2: Hiding Tables and Fields

In this task, you will hide tables and fields that are not appropriate to use in reports.

1. Toggle to the **Data** view.

2. In the **Fields** pane (located at the right), expand the **Date** table.

3. In the **Date** table, notice that four fields have the sigma symbol (Σ) next to them.

Note: Fields adorned with the sigma symbol can be aggregated in a report. It is not appropriate for these fields to be aggregated. Therefore, each will be hidden.

4. From inside the **Date** table, right-click the **DateKey** field, and then click **Hide in Report View**.
5. Repeat the previous step to hide the remaining three visible key fields in the **Date** table.
6. In the data grid, notice that hidden columns are displayed differently.
7. Expand the **Sales** table, and then hide the **OrderDate** and **Region Name** fields.
8. To hide the **ProductCost** table, right-click the table, and then click **Hide in Report View**.
9. Hide the **Region** table as well.
10. Toggle to the **Relationships** view, and notice that hidden tables and fields are grayed out.

Tasks 3: Sorting Columns

In this task, you will create new columns to enable reporting on profit.

1. In the **Data** view, in the **Fields** pane, from inside the **Date** table, select the **Day** field.
2. On the **Data Tools | Modeling** contextual ribbon, in the **Sort** group, click the **Sort by Column**, and then click **DateKey**.

3. Use the steps in this task to sort the **Month** column by the **MonthKey** column.

Note: It is not necessary to configure the sort column for the **Quarter** *and* **Year** *columns, because the text sort order is the same as the chronological sort order.*

Tasks 4: Categorizing a Column

In this task, you will categorize the **State Name** column.

1. In the **Fields** pane, from inside the **Sales** table, select the **State Name** field.
2. On the **Data Tools | Modeling** contextual ribbon, in the **Properties** group, in the **Data Category** drop-down list, click **State or Province**.

Note: This will enable default spatial reporting by using maps.

Tasks 5: Creating Calculated Columns

In this task, you will create new columns to enable profit reporting.

1. In the **Fields** pane, right-click the **Sales** table, and then click **New Column**.
2. Notice that the focus is set to the formula bar, and that the column name is selected.

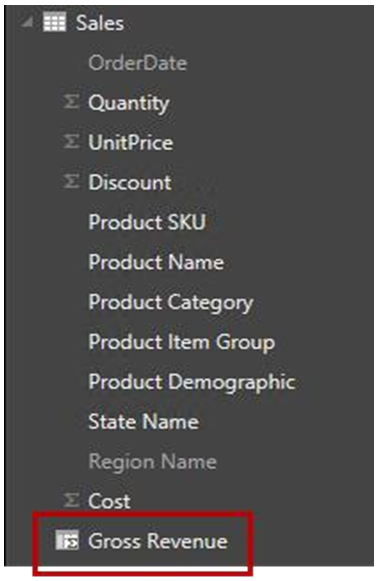
3. Replace the word **Column** with **Gross Revenue**.
4. On the right side of the equals sign, use IntelliSense to type the following formula:

DAX

```
[Quantity] * [UnitPrice]
```

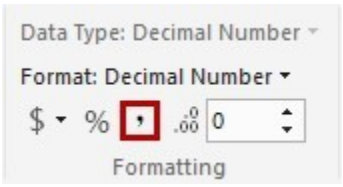
Note: This column could also have been implemented as a new column in a query, because either approach produces the same outcome. The appropriate approach will often be determined by whether the M language (used by queries) or DAX (used for calculated columns) best achieves the requirement.

5. Press Enter.
6. In the **Fields** pane, notice the addition of the column.



7. To format the column, in the **Fields** pane, select the **Gross Revenue** field.

8. On the **Data Tools | Modeling** contextual ribbon, in the **Formatting** group, click the **Thousands Separator**.



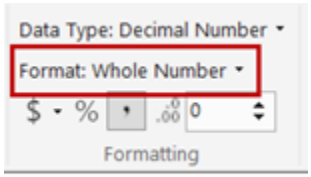
9. In the **Sales** table, create an additional column named **Net Revenue**, formatted with the thousands separator, and based on the following formula:

DAX

```
[Gross Revenue] - [Discount]
```

10. Format the **Quantity**, **Discount** and **Cost** columns with the thousands separator.

11. Set the format for the **Discount** and **Cost** columns to **Whole Number**.



Tasks 6: Creating Measures

In this task, you will create three measures to report on profitability.

1. In the **Fields** pane**,** right-click the **Sales** table, and then click **New Measure**.

2. In the formula bar, replace the word **Measure** with **Gross Profit**.

3. On the right side of the equals sign, type the following formula:

DAX

```
SUM([Gross Revenue]) - SUM([Cost])
```

Note: Unlike calculated columns, measures typically involve the aggregation of values from many rows. Also, unlike calculated columns, measures do not store additional data.

4. Format the **Gross Profit** measure as a whole number, and with the thousands separator.

5. Repeat the previous steps in this task to create and format an additional measure named **Net Profit**, by using the following formula:

DAX

```
SUM([Net Revenue]) - SUM([Cost])
```

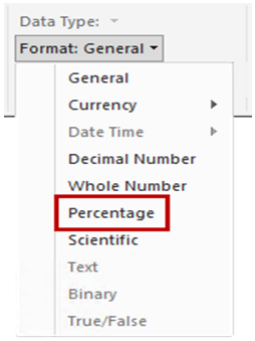
6. Create a third measure named **Profitability**, by using the following formula.

DAX

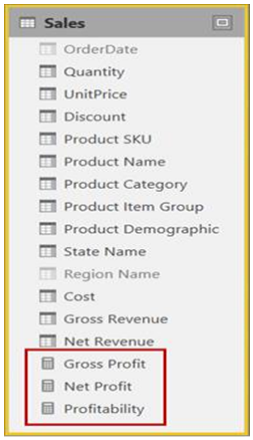
```
DIVIDE(Sales[Net Profit], SUM(Sales[Net Revenue]))
```

Note: The **DIVIDE** *function divides two sum expressions, provided the second argument results in a non-zero number. If the second argument results in zero or blank (missing), then the function will return a blank.*

7. Format the measure as **Percentage**.



8. Toggle to the **Relationships** view, and in the **Sales** table, notice that the three measures are adorned with a different icon.



Tasks 7: Creating Hierarchies

In this task, you will create some hierarchies that will be used later, for reporting.

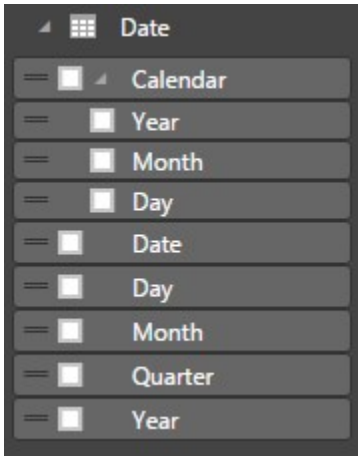
1. In the **Fields** pane, in the **Reporting View,** expand the **Date** table, right-click **Year**, and then select **New Hierarchy**

2. Right-click the new hierarchy called **Year Hierarchy** and select **Rename**. Rename **Year Hierarchy** to **Calendar**.

3. Right-click the **Month** attribute in the **Date** table, and then click **Add to Calendar**.

4. Do the same for the **Day** attribute or drag it to the hierarchy**.**

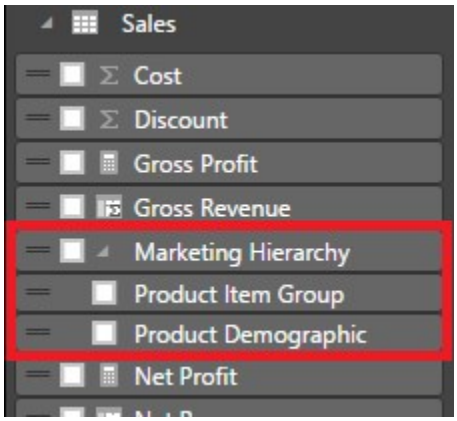
5. Expand the hierarchy and inspect its contents. It should look like the following screenshot:



6. Notice that if you right-click a **Level** in the hierarchy, you can control its placement. You can move an attribute up or down in the hierarchy.

7. Create a new hierarchy in the **Sales** table with **Product Category** and **Product Name**. Name it **Product Categories**.

8. Create a second new hierarchy in the **Sales** table with **Product Item Group** and **Product Demographic**. Name it **Marketing Hierarchy**. It should look like the following screenshot:



Exercise 3: Extending the Data Model

In this exercise, you will practice what you have learned about DAX calculations. After completing this exercise, you will be able to:

- Create a Data Table.
- Create DAX Advanced Calculations.
- Use Quick Measures feature.
- Use Disconnected Tables.
- Use What If parameter.

Tasks 1: Creating a Data Table

In this task, you will create a data table in DAX, which will be a subset of an existing table. This will generate a new lookup table for **Products**.

1. In the **Modeling** tab, select the option **New Table**. This table will be used to generate a new lookup table for **Products**.

2. In the **Formula bar**, type the following expression:

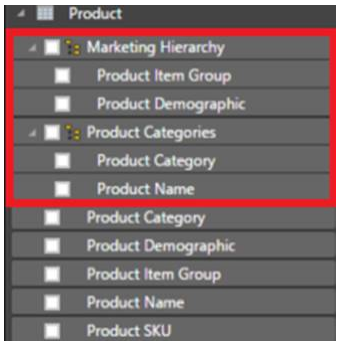
DAX

```
Product =
DISTINCT(SELECTCOLUMNS(
Sales,"Product SKU",Sales[Product SKU],
"Product Name",Sales[Product Name],
"Product Category",Sales[Product Category],
"Product Item Group",Sales[Product Item Group],
"Product Demographic",Sales[Product Demographic]))
```

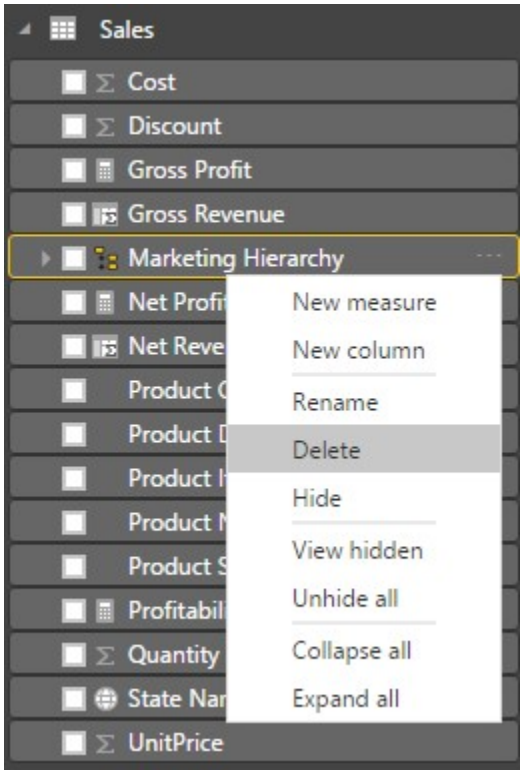
3. Notice that a new table called **Product** is being generated, by selecting all of the distinct combinations of **Product** attributes that exist in the **Sales** table.

4. Recreate the hierarchies you previously created for product attributes in the **Sales** table, but this time, do it in the **Product** table.

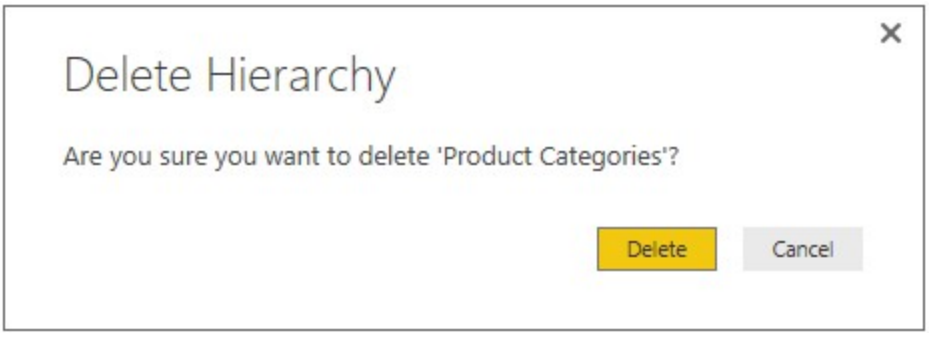
5. In the end, you should have the following in the Product table:



6. Delete the previous **Product** related hierarchies in the **Sales** table by right-clicking each one and then clicking **Delete**.



You will receive a warning before the deletion. It is OK to remove hierarchies, because they do not result in removal of the underlying attributes.



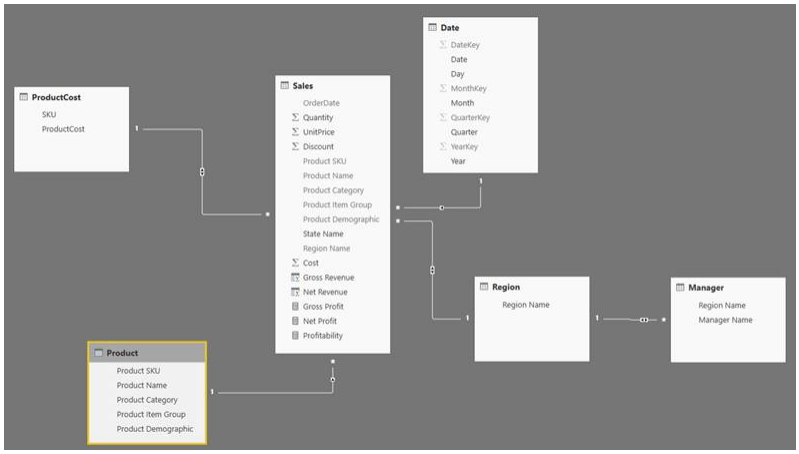
7. In the **Sales** table, make sure to also hide all of the **Product**-related attributes, namely:
    - Product SKU
    - Product Name
    - Product Category
    - Product Item Group
    - Product Demographic

8. To hide an attribute, you can right-click it in the **Report View**, and then click **Hide**. Alternatively, in **Data View**, you can select **Hide in Report View**.

9. Finally, we are going to relate the new **Product** table with the **Sales** table, but we will let Power BI do the hard work.

10. Click **Manage Relationships** in the **Modeling** tab, and then click **Autodetect**. PBI Desktop will find a new relationship that will connect **Sales** to **Product**, via the **Product SKU**.

11. The data model should now look similar to the following screenshot:



Tasks 2: Creating Advanced Calculations

In this task, you will create some more calculations in the model, with DAX, using what you have learned so far. Although you have not learned how to create reports yet, we advise you to try out your formulas, with simple tables, as you progress.

1. In the **Report View**, expand the **Sales** table. We are going to create a new calculation for the sum of Net Revenue for Sales that had a discount.

CHALLENGE: Do you think that this should be a Measure or a Column? Why?

A:_____

_____

_____

2. We are going to create a new measure for the new requirement. The expression you should use is:

DAX

```
Net Revenue Discount = CALCULATE(SUM(Sales[Net Revenue]),Sales[Discount]>0)
```

If you do not remember how to create a calculation, please go to section **Creating Measures** on **Exercise 2.**

3. We now have a new requirement from our **Midwest** users. They want a measure that will give them their **Net Revenue** immediately, without having to filter anything.

CHALLENGE: Can you come up with a solution?

Note: Please, do not call the instructor until you have given it some serious thought. He or she will ask you some questions too.

Here is a possible result:

Product Category	Midwest Net Revenue	Net Revenue
Collective pitch	12,517,969.10	44,916,866.20
Trainer	4,182,401.65	16,917,746.45
Warbird	2,942,886.95	12,008,713.70
Co-Axial	1,298,421.50	5,279,725.50
Glider	441,233.85	1,811,700.75
Fixed pitch	90,518.50	354,862.90
Total	**21,473,431.55**	**81,289,615.50**

4. These users are really enjoying exploring their data, and now they want a measure that will allow them to see the contribution of each **Product Name** to the total of what they are selecting; for instance, Category, but it can be used with anything, really.

Here is a glimpse:

Product Category	Product Name	Percentage of Product Net Revenue
Co-Axial	3CAX-B Helicopter	19.60 %
	4CAX-B Helicopter	25.32 %
	Tailspin Heli - Co-Ax Pro Mk I - 4ch	55.08 %
	Total	**100.00 %**
Collective pitch	6CCP-A Helicopter	19.29 %
	Tailspin Heli - Max Pro Flight - 6ch	80.71 %
	Total	**100.00 %**
Fixed pitch	3CFP-I Helicopter	56.05 %
	4CFP-I Helicopter	21.79 %
	Tailspin Heli - Pro Mk III - 5ch	22.17 %
	Total	**100.00 %**
Glider	Tailspin Aviator Mk2-11	40.48 %
	Trainer - Tailspin GL-120	24.60 %
	Trainer - Tailspin GL-155	34.92 %
	Total	**100.00 %**
Trainer	Piper Cub 3 Channel	6.07 %
	Piper Cub 4 Channel	35.32 %
	SkyTrainer	11.01 %
	Tailspin Aviator Mk2-12	16.33 %
	Tailspin Aviator Mk2-15	31.27 %
	Total	**100.00 %**
Warbird	P47 4 Channel	7.56 %
	P47 5 Channel	52.52 %
	P51	16.57 %
	Tailspin Warbird BM32	23.35 %
	Total	**100.00 %**
Total		**100.00 %**

The following screenshot shows another one, using an attribute that is not even from the **Product** table:

Region Name	Product Name	Percentage of Product Net Revenue
Midwest	3CAX-B Helicopter	1.21 %
	3CFP-I Helicopter	0.23 %
	4CAX-B Helicopter	1.54 %
	4CFP-I Helicopter	0.09 %
	6CCP-A Helicopter	11.22 %
	P47 4 Channel	1.04 %
	P47 5 Channel	7.20 %
	P51	2.26 %
	Piper Cub 3 Channel	1.16 %
	Piper Cub 4 Channel	7.12 %
	SkyTrainer	2.11 %
	Tailspin Aviator Mk2-11	0.84 %
	Tailspin Aviator Mk2-12	3.10 %
	Tailspin Aviator Mk2-15	5.98 %
	Tailspin Heli - Co-Ax Pro Mk I - 4ch	3.30 %
	Tailspin Heli - Max Pro Flight - 6ch	47.08 %
	Tailspin Heli - Pro Mk III - 5ch	0.10 %
	Tailspin Warbird BM32	3.21 %
	Trainer - Tailspin GL-120	0.51 %
	Trainer - Tailspin GL-155	0.71 %
	Total	**100.00 %**
New England	3CAX-B Helicopter	1.06 %
	3CFP-I Helicopter	0.25 %
	4CAX-B Helicopter	1.18 %
	4CFP-I Helicopter	0.08 %
	6CCP-A Helicopter	9.33 %
	P47 4 Channel	0.51 %
	P47 5 Channel	3.48 %

CHALLENGE: Give it some thought. It is not that difficult. A hint: Take a second look at the **ALL function**.

5. Before they ask us, we are now going to create some time intelligence measures for them.

6. The first Measure will be for obtaining the **Net Revenue** for previous month. We are going to use the **DATEADD** function for this one:

DAX

```
Net Revenue LM = CALCULATE(SUM(Sales[Net Revenue]),
DATEADD('Date'[Date],-1,MONTH))
```

Please, take a minute to understand how it is built.

7. Now we need to build a new Measure, also for **Net Revenue**, which will return the **percentage growth, between current month and last month**.

CHALLENGE: We rely on you for getting the job done!

Here is a possible result:

Month	Net Revenue Growth between Months	Net Revenue LM	Net Revenue
2013-Jan			5,751.10
2013-Feb	2440.34 %	5,751.10	146,097.45
2013-Mar	221.17 %	146,097.45	469,220.75
2013-Apr	27.93 %	469,220.75	600,276.25
2013-May	-3.37 %	600,276.25	580,073.05
2013-Jun	-20.59 %	580,073.05	460,653.65
2013-Jul	22.79 %	460,653.65	565,648.35
2013-Aug	5.31 %	565,648.35	595,686.75
2013-Sep	279.02 %	595,686.75	2,257,755.05
2013-Oct	-20.15 %	2,257,755.05	1,802,762.30
2013-Nov	-49.36 %	1,802,762.30	912,986.35
2013-Dec	-29.84 %	912,986.35	640,507.55
2014-Jan	38.09 %	640,507.55	884,487.50
2014-Feb	-5.99 %	884,487.50	831,485.85
2014-Mar	7.59 %	831,485.85	894,572.50
2014-Apr	604.26 %	894,572.50	6,300,075.35
2014-May	-19.69 %	6,300,075.35	5,059,630.65
2014-Jun	-25.61 %	5,059,630.65	3,763,965.50
2014-Jul	-14.86 %	3,763,965.50	3,204,464.95
2014-Aug	-16.43 %	3,204,464.95	2,678,074.90
2014-Sep	1.14 %	2,678,074.90	2,708,734.75
Total	**-0.00 %**	**81,289,615.50**	**81,289,615.50**

8. Next, we will build one for the **same period last year**. Here is the DAX expression for this:

DAX

```
Net Revenue PP = CALCULATE(SUM(Sales[Net
Revenue]),SAMEPERIODLASTYEAR('Date'[Date]))
```

9. Now, we need to build one, also for **Net Revenue**, which will return the **YTD**. Create one for **QTD** too.

   CHALLENGE: Shall you try?

   A possible result:

Month	Net Revenue	Net Revenue YTD	Net Revenue QTD
2013-Jan	5,751.10	5,751.10	5,751.10
2013-Feb	146,097.45	151,848.55	151,848.55
2013-Mar	469,220.75	621,069.30	621,069.30
2013-Apr	600,276.25	1,221,345.55	600,276.25
2013-May	580,073.05	1,801,418.60	1,180,349.30
2013-Jun	460,653.65	2,262,072.25	1,641,002.95
2013-Jul	565,648.35	2,827,720.60	565,648.35
2013-Aug	595,686.75	3,423,407.35	1,161,335.10
2013-Sep	2,257,755.05	5,681,162.40	3,419,090.15
2013-Oct	1,802,762.30	7,483,924.70	1,802,762.30
2013-Nov	912,986.35	8,396,911.05	2,715,748.65
2013-Dec	640,507.55	9,037,418.60	3,356,256.20
2014-Jan	884,487.50	884,487.50	884,487.50
2014-Feb	831,485.85	1,715,973.35	1,715,973.35
2014-Mar	894,572.50	2,610,545.85	2,610,545.85
2014-Apr	6,300,075.35	8,910,621.20	6,300,075.35
2014-May	5,059,630.65	13,970,251.85	11,359,706.00
2014-Jun	3,763,965.50	17,734,217.35	15,123,671.50
2014-Jul	3,204,464.95	20,938,682.30	3,204,464.95
2014-Aug	2,678,074.90	23,616,757.20	5,882,539.85
2014-Sep	2,708,734.75	26,325,491.95	8,591,274.60
2014-Oct	[illegible]	[illegible]	[illegible]
Total	**81,289,615.50**	**30,388,941.15**	

10. Let us create a final advanced calculation to obtain our top 10 **Product Names** by **Net Profit**.

    CHALLENGE: It is the final challenge in this Exercise, we promise.

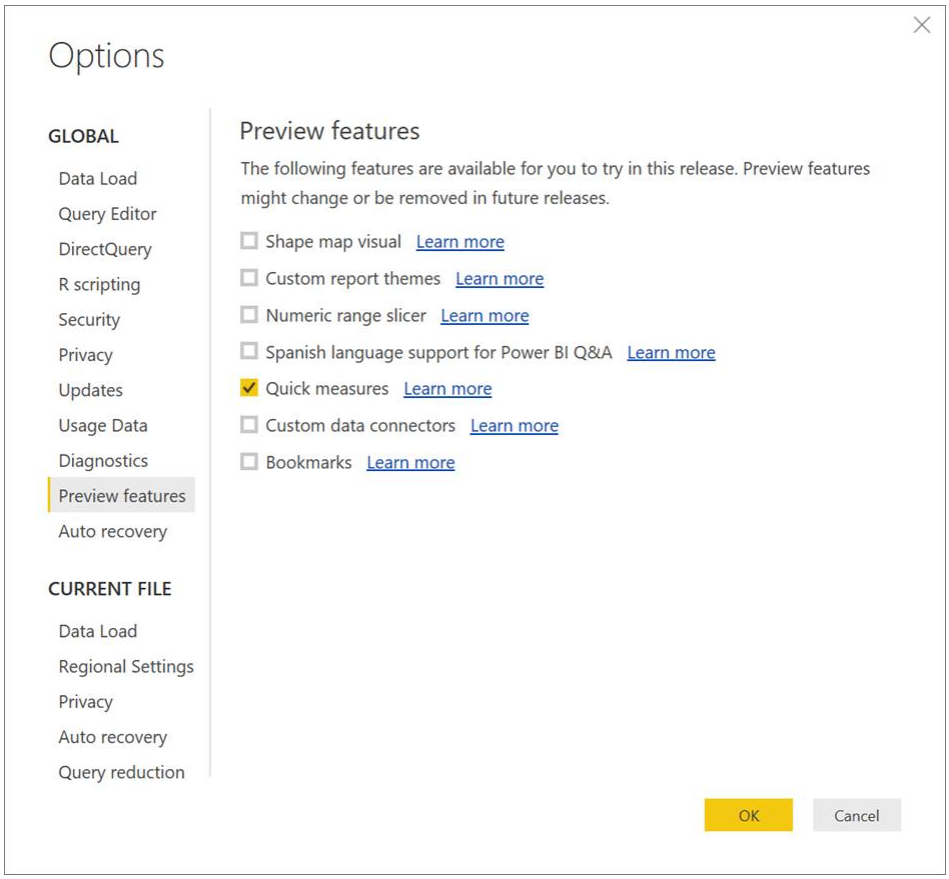
    A glimpse of what we should get:

Product Name	Product Rank	Net Profit
Tailspin Heli - Max Pro Flight - 6ch	1	8,353,028
6CCP-A Helicopter	2	2,094,360
Piper Cub 4 Channel	3	1,245,153
Tailspin Aviator Mk2-15	4	1,026,410
P47 5 Channel	5	1,010,973
Tailspin Warbird BM32	6	658,274
Tailspin Heli - Co-Ax Pro Mk I - 4ch	7	631,918
Tailspin Aviator Mk2-12	8	615,820
P51	9	332,761
4CAX-B Helicopter	10	270,166
3CAX-B Helicopter	11	256,033
Piper Cub 3 Channel	12	201,687
SkyTrainer	13	193,035
P47 4 Channel	14	130,678
Tailspin Aviator Mk2-11	15	113,469
Trainer - Tailspin GL-155	16	90,878
Trainer - Tailspin GL-120	17	63,774
3CFP-I Helicopter	18	45,866
Tailspin Heli - Pro Mk III - 5ch	19	15,392
4CFP-I Helicopter	20	14,610
Total	**1**	**17,364,285**

Tasks 3: Using Quick Measure

In this task, you will create an additional calculation in the model with the Quick Measures feature. If you are not familiar DAX, Quick Measures can help you to create measures and to learn about DAX.

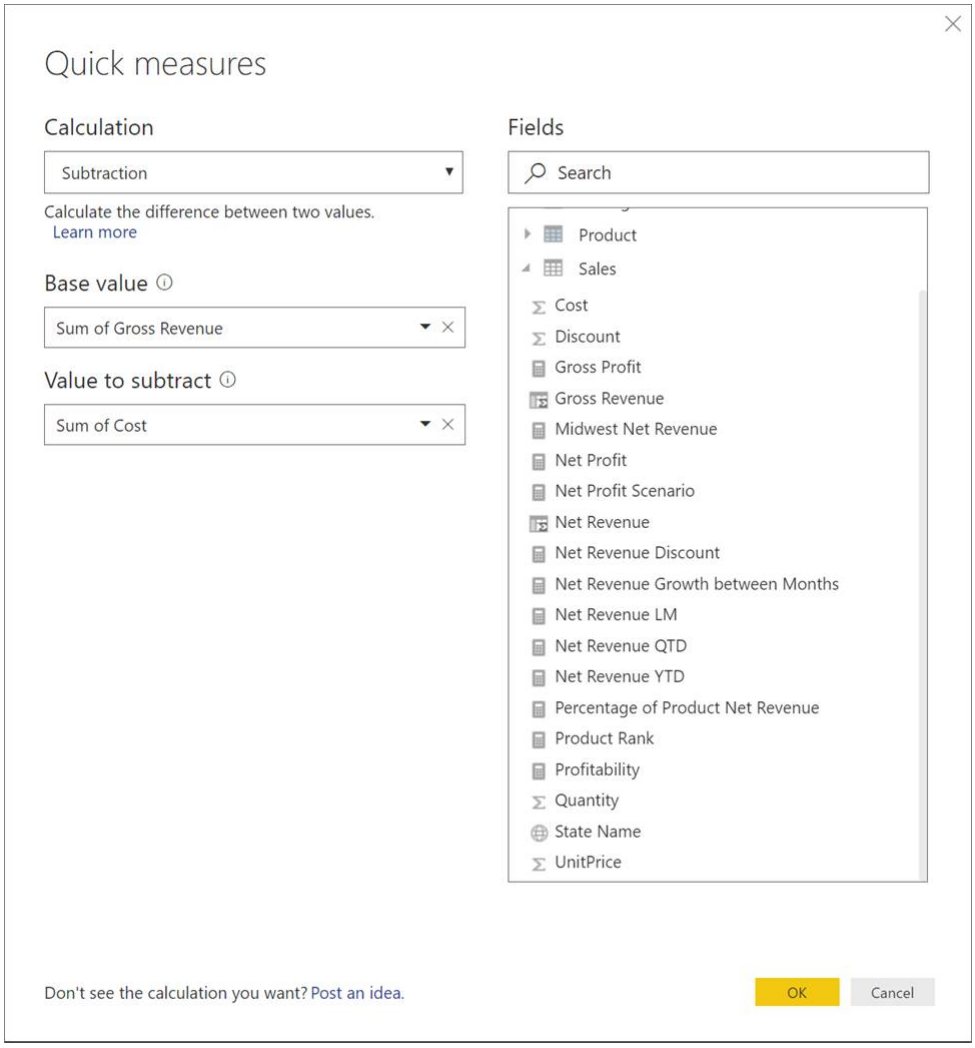
1. In the **File** tab, in the **Option and settings** category, click **Options** menu.
2. In the Preview features category, check **quick measure** option and Click **OK** and then restart Power BI Desktop. If you already enable quick measure feature, you can skip this step.



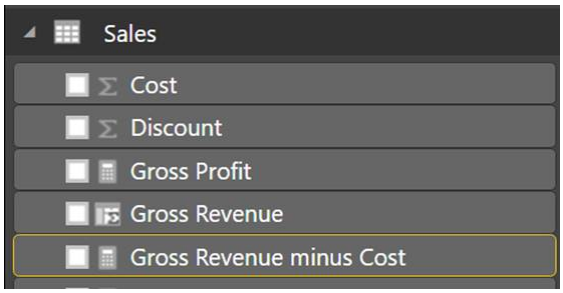
3. After restart Power BI Desktop, open **US Sales Analysis.pbix** file.

4. In the **Fields** pane, right-click the Sales table, and then click **New Quick Measure** and configure a new Quick measure with **Calculation** as Subtraction, **Base value** as Sum of Gross Revenue from Sales table and **Value to subtract** as Sum of Cost from Sales table, as shown in the following

screenshot:



5. Click **OK** to create measure into the model.



6. Select measure named **Gross Revenue minus Cost** in the Fields pane and you can find the DAX formula like below in the formula bar.

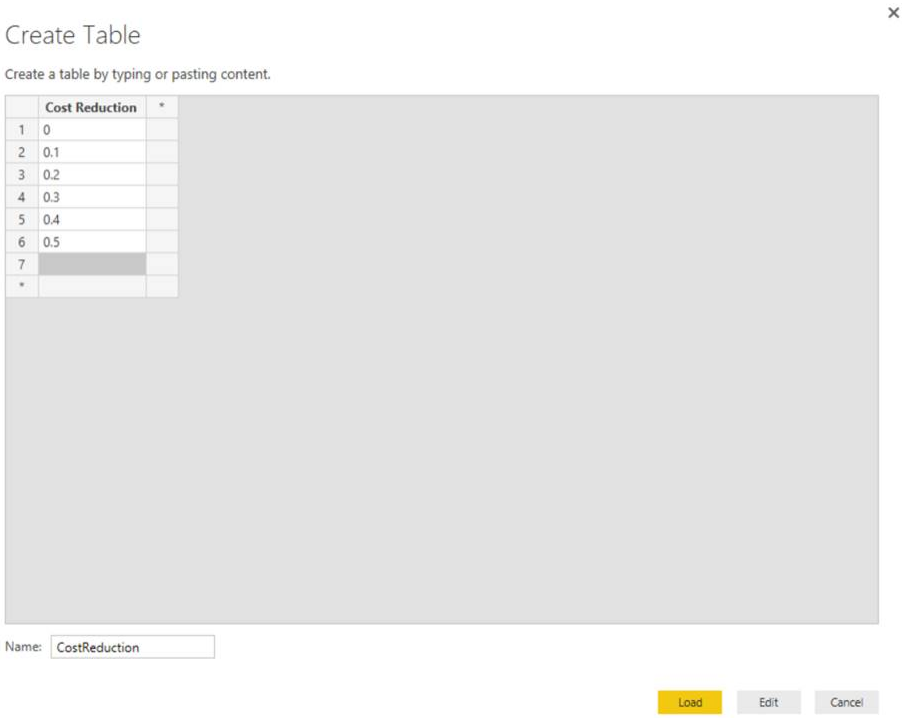
DAX

```
SUM('Sales'[Gross Revenue]) - SUM('Sales'[Cost])
```

Tasks 4: Using Disconnected Tables

In this task, you will use disconnected tables to create a scenario around lowering product costs.

1. Create a table that will have the percentage we are going to apply to cost reduction. In the **Home** tab, in the **External Data** group, click **Enter Data** and configure a new table named **CostReduction** with a column named **Cost Reduction** with values between 0 and 0.5, as shown in the following screenshot:



2. Click **Load** to load the table into the model.

3. This table will not have any relationships with other tables, as have learned during the Disconnected Tables demo.

4. Create a new **measure** in the **CostReduction** table with the following expression:

DAX

```
CostRedux = MIN(CostReduction[Cost Reduction])
```

5. Hide that measure, because it is only an auxiliary measure that will be used to **harvest** a **Cost Reduction value** from its table.

6. Format the **Cost Reduction** field as **Percentage**.

7. In a previous exercise, we have created a measure to calculate the Net Profit:

```DAX
Net Profit = SUM([Net Revenue]) - SUM([Cost])
```

We are now going to create a new measure, which will use the **harvest measure** to apply a percentage decrease to **Cost**. Create the measure with the following expression:

```DAX
Net Profit Scenario = SUM([Net Revenue]) - ((1-
CostReduction[CostRedux])*SUM([Cost]))
```

Take a minute to inspect the expression and understand what it is doing.

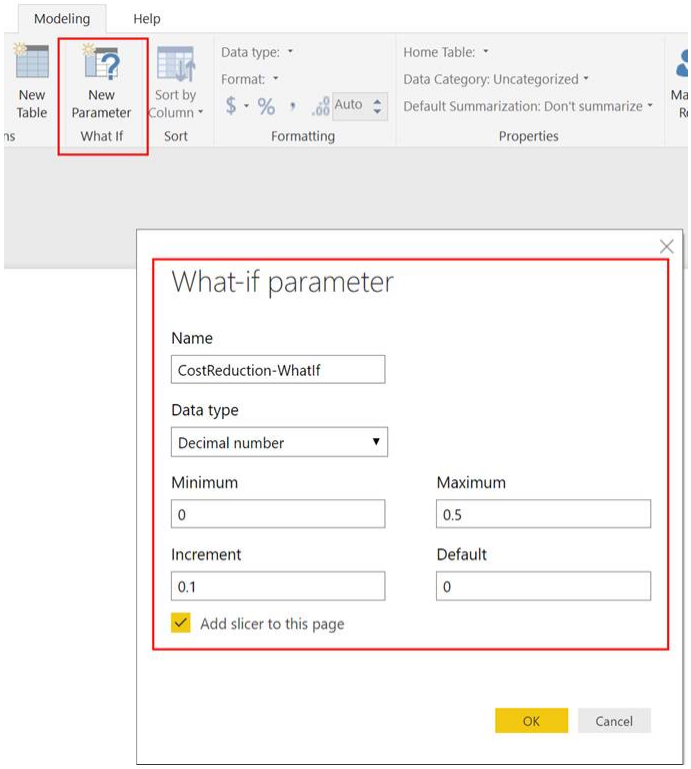
8. With a 10% reduction in costs, how much would we increase Net Profit?

A:_____

Tasks 5: Using What-If parameter

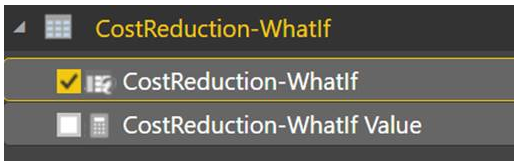
In this task, you will use What If parameter to create a scenario around lowering product costs. the What-If parameter is similar disconnected table scenario.

1. In the **Modeling** Tab, in the **What If** group, click **New Parameter** and configure a new parameter named **CostReduction-WhatIf** with Data type as Decimal number, 0 as Minimum value, 0.5 as Maximum value, 0.1 as Increment and 0 as Default value, as shown in the following screenshot :



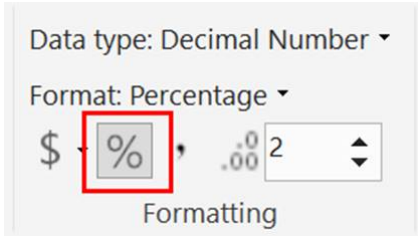
2. Click to OK to create that parameter as table into the model.

3. This table have one attribute and one calculated measure **CostReduction-WhatIf Value**. And this table does not have any relationships with other tables.

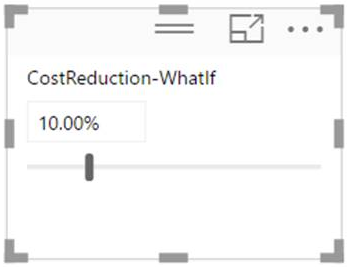


4. In addition, a new slicer with What-If parameter is created on the report page.

5. Select **CostReduction-WhatIf** attribute in the **Fields** pane and format this attribute as **Percentage**.



After this formatting, The slicer for parameter will present a value with percentage.



6. In a previous exercise, we have created a measure to calculate the Net Profit:

DAX

```
Net Profit = SUM([Net Revenue]) - SUM([Cost])
```

We are now going to create a new measure, which will use the **measure from the What If table** to apply a percentage decrease to **Cost**. Create the measure with the following expression:

DAX

```
Net Profit Scenario (what-if) = SUM([Net Revenue]) - ((1-'CostReduction-
WhatIf'[CostReduction-WhatIf Value])*SUM([Cost]))
```

Take a minute to inspect the expression and understand what it is doing.

7. With a 10% reduction in costs, how much would we increase Net Profit?

A:_____

Exercise 4: Exploring and Reporting

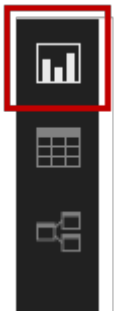
In this exercise, you will build visualizations that take advantage of all of the work you have done defining and enhancing the data model. After completing this exercise, you will be able to:

- Explore a data model.
- Select appropriate visualizations to convey your message.
- Control the interaction between visualizations in a report.
- Define filtering options.

Tasks 1: Adding Report Decoration

In this task, you will add a text box and an image to the report page.

1. To toggle to **Report** view, on the left side, click **Report**.



Note: The blank area is the report page canvas.

2. At the bottom-left corner, notice the page navigation control.

Note: A report can consist of multiple pages.

3. To insert an image, on the **Home** ribbon, in the **Insert** group, click **Image**.

4. In the **Open** window, go to the **D:\Assets\Lab 1** folder.

5. Select the **TailspinToysLogo.png** file, and then click **Open**.

6. To resize the image, ensure that it is selected to reveal the border guides.

7. Drag the border guides to resize the image to create a smaller sized tile, and then reposition it as follows:

8. To insert a text box, on the **Home** ribbon, in the **Insert** group, click **Text Box**.

9. Inside the text box, type **US Sales Analysis**.

10. Select the entire text, and then use the text format bar to increase the font size to **60**.

11. Resize the text box to a smaller size, and then reposition it as follows:

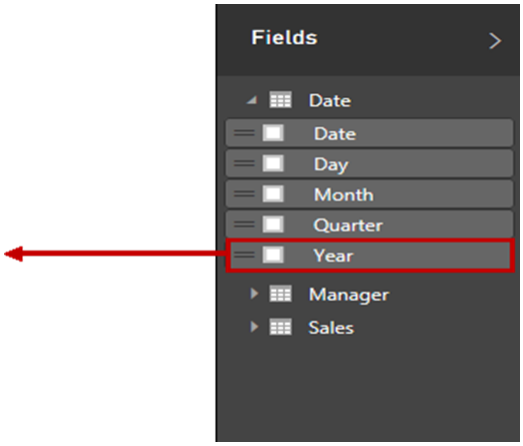
Tasks 2: Creating a Slicer

In this task, you will create a slicer to enable the report user to interact and filter by year.

1. In the **Fields** pane, notice that only five tables are available.

   Note: The remaining tables are hidden.

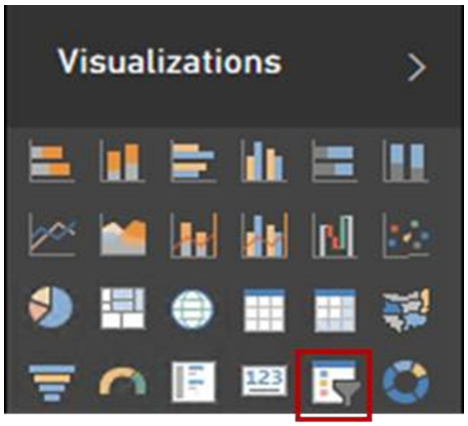
2. To create a visualization based on a field, in the **Fields** pane, expand the **Date** table, and then drag the **Year** field and drop it on a blank area of the canvas.



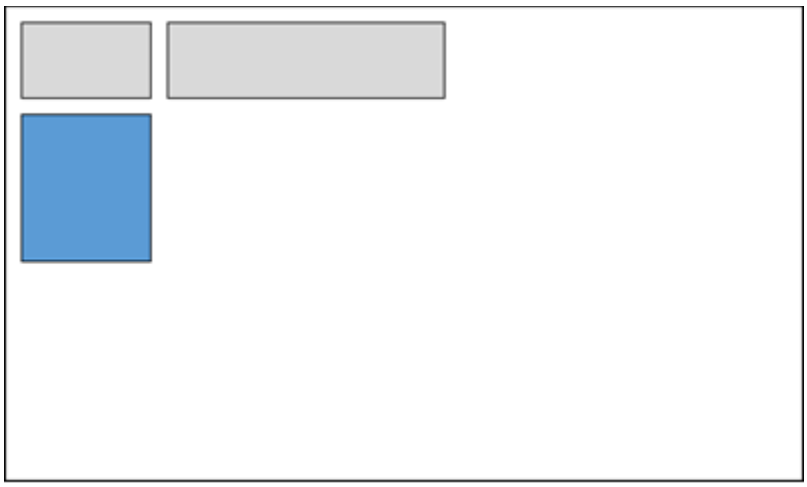
3. Notice that the field was used to create a new table visualization.

4. To switch the visualization to a slicer, in the **Visualizations** pane, click the **Slicer** icon.

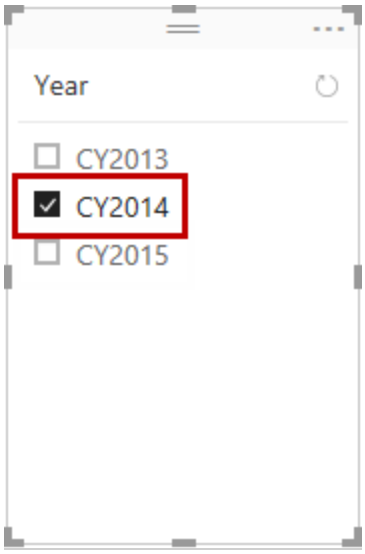
   Tip: Hovering over a visualization type icon will reveal a tooltip that describes the visualization type.



5. Resize the slicer, and then reposition it as follows:



6. In the slicer, select the **CY2014** check box.



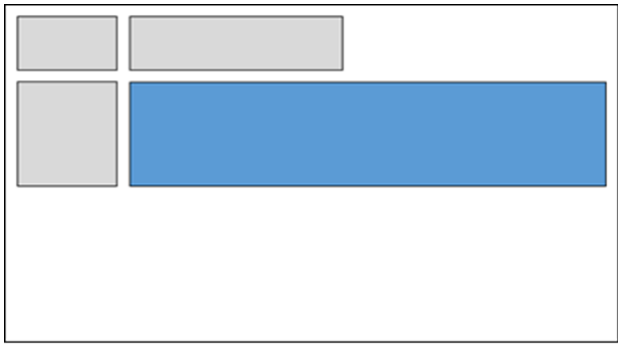
Note: All data visualizations on the page will now be filtered by year CY2014.

Tasks 3: Creating a Combo Chart

In this task, you will create a combo chart visualization to display monthly revenue, cost, and profitability.

1. In the **Fields** pane, from inside the **Sales** table, drag the **Net Revenue** field and drop it on a blank area of the canvas.

2. Resize and reposition the visualization as follows:

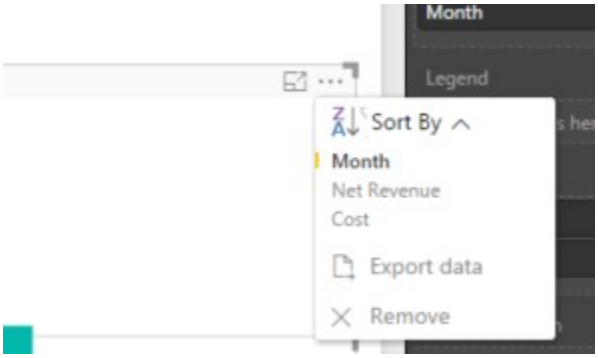


3. Drag the following fields and drop them inside the chart:

Table	Field
Sales	Cost

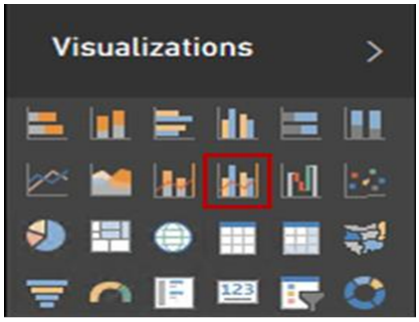
Date	Month

4. On the horizontal axis, verify if the months are chronologically sorted. If not, change the sorting options on the top-right of your visualization

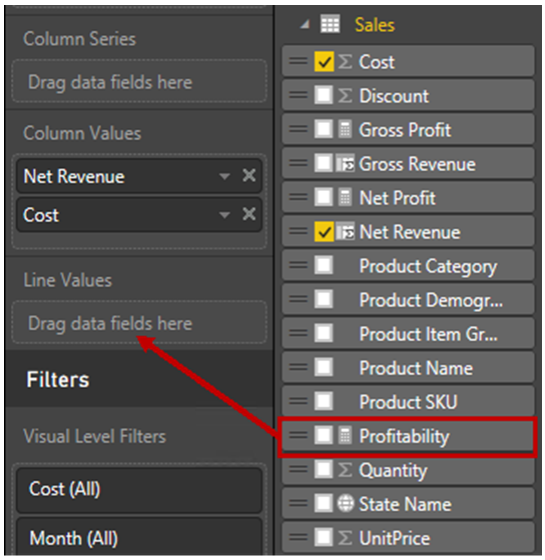


Note: This is because you configured the sort column in the previous exercise.

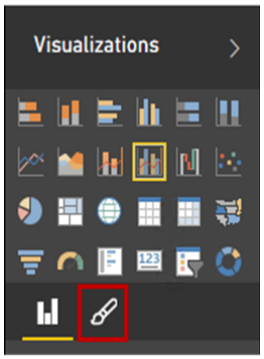
5. Modify the visualization to **Line and Clustered Column Chart**.



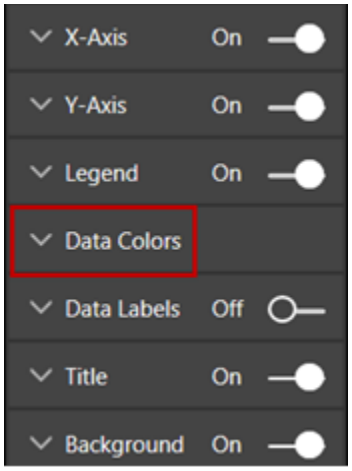
6. In the **Fields** pane, from inside the **Sales** table, drag the **Profitability** field and drop it in the **Visualizations** pane, inside the **Line Values** well.



7. To modify the visualization style, toggle to **Format**.

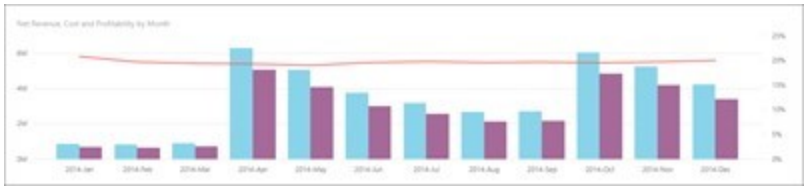


8. Expand **Data Colors**.



9. Select appropriate colors for the three fields (suggesting blue for **Net Revenue**, purple for **Cost**, and orange for **Profitability**).

10. Verify that the visualization matches the following screenshot:

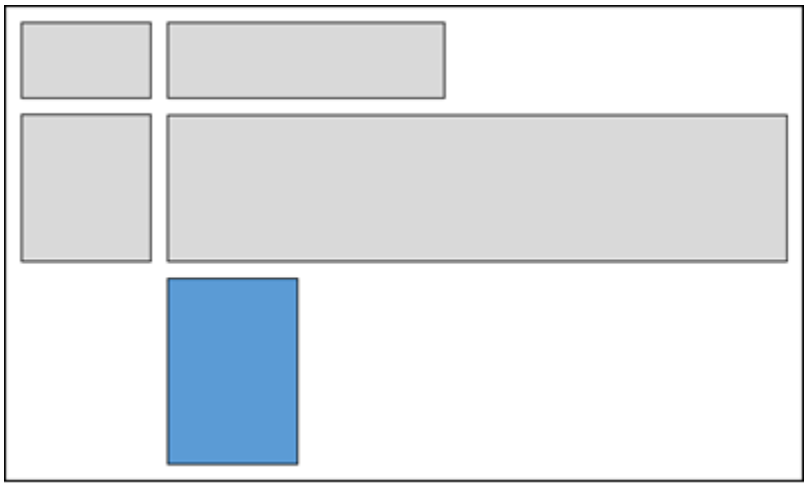


11. Notice the different vertical axis scales, on the left and right sides of the chart.

Tasks 4: Creating Tables

In this task, you will create two tables. The first table will display regional sales and the second table will display manager sales. The tables will help you to understand the many-to-many relationship defined between managers and sales.

1. In the **Fields** pane, from inside the **Manager** table, drag the **Region Name** field and drop it on a blank area of the canvas.
2. Add the **Net Revenue** field to the new visualization.
3. Resize and reposition the visualization as follows:

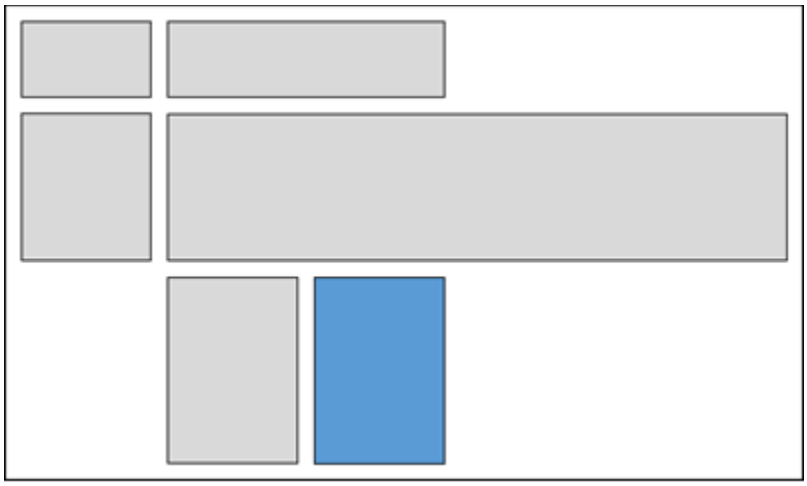


4. To sort the table rows by descending net revenue, click the **Net Revenue** column header.
5. Verify that the visualization matches the following:

Region Name	Net Revenue
Southern	12,216,301
Midwest	11,206,199
Pacific Northwest	5,725,437
Northeast	5,218,621
Southwest	3,969,162
New England	3,527,536
Total	**41,863,256**

6. Repeat the steps in this task to create a second table to display revenue, but this time by using the **Manager Name** field.

7. Resize and reposition the visualization as follows:



8. Sort the table by descending net revenue.

9. Verify that the visualization matches the following:

Manager Name	Net Revenue
Ananya Kumar	23,422,500
Ted Baker	11,206,199
Ty Johnston	7,496,698
Haruto Suzuki	5,725,437
Jane Campbell	5,218,621
Carmen Carrington	3,969,162
John Bishop	3,527,536
Total	**41,863,256**

10. At a glance, notice that the sum of the individual manager revenue values exceeds the $41.8 million table total.

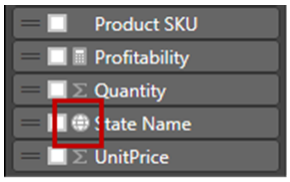
Note: The many-to-many relationship results in duplication of revenue values, because multiple managers are assigned to a single region. The total of the second table is correct, as can be verified against the total of the first table.

Tasks 5: Creating a Map

In this task, you will create a map to display US state profitability by product item type.

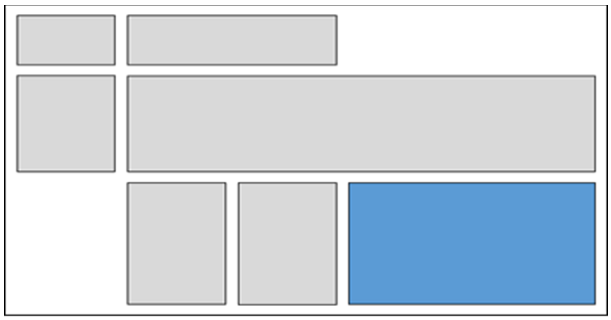
1. In the **Fields** pane, from inside the **Sales** table, notice that the **State Name** field is decorated with a spatial icon.

   Note: This is because the column was categorized as State or Province, earlier in this lab.



2. Create a new visualization based on the **State Name** field.

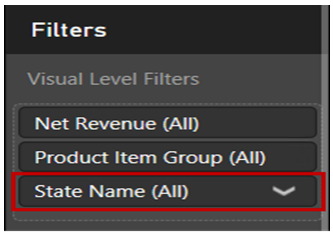
3. Resize and reposition the visualization as follows.



4. Drag the **Net Revenue** field and drop it into the map, and notice that it is added to **Values** well.

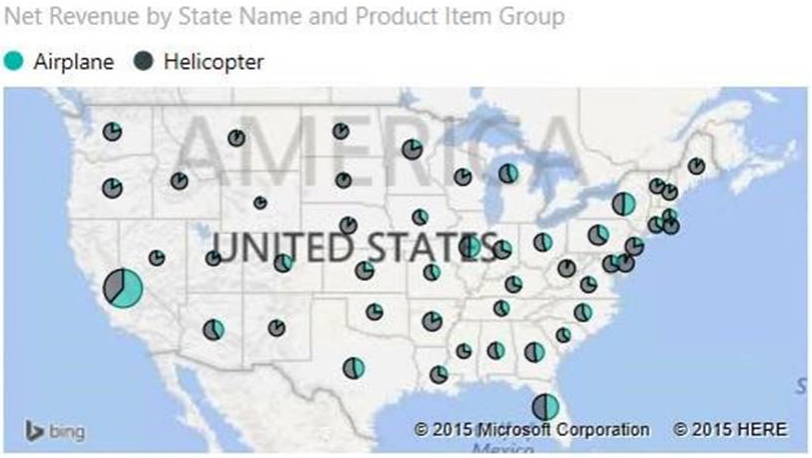
5. To display pie charts, drag the **Product Item Group** field from the **Product** table, and then drop it into the map. Notice that it is added to the **Legend** well.

6. To display only the contiguous US states, in the **Filters** section, click **State Name (All)** to expand the filter list.



7. Select **(All)**, and then clear **Alaska** and **Hawaii**.

8. Verify that the visualization matches the following screenshot:



9. Verify that the report page matches the following screenshot:

Tasks 6: Interacting with the Report Page

In this task, you will interact with the report page and change the way the visualizations interact.

1. In the slicer, clear **CY2014** and notice that all years are represented in the report.

2. Check **CY2015**.

3. To filter by a particular month, in the combo chart, hover the cursor over any one of the **Net Revenue** columns, and then take note of the value.

4. Click the column, and notice that all of the report page visualizations filter by that month. (You can verify this by comparing the table totals to the noted value).

5. To clear the filter, click in a blank area of the combo chart.

6. To determine the net profit for a particular state and product item group, hover the cursor over any state pie section, and then take note of the value.

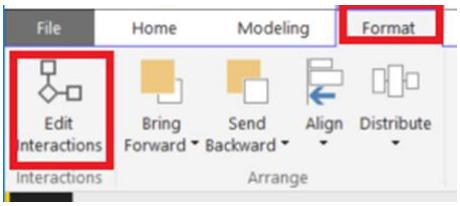
7. Click the state pie slice, and notice that all of the report page visualizations filter by that state and product item group.

8. To clear the filter, click in a blank area of the map.

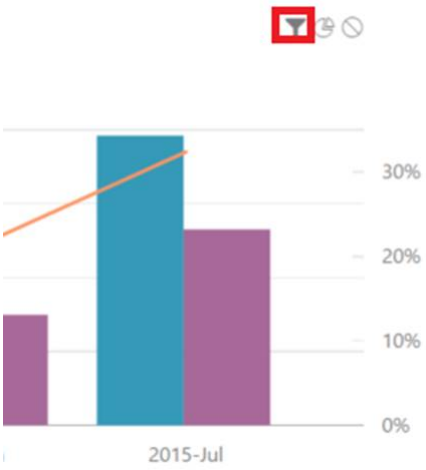
9. Notice how it does not make sense to filter the Clustered Bar chart based on a selection in the pie chart on the map, because the impact is barely visible. Let us change that interaction.

10. Click the map's border to make sure it is selected.

11. Click the **Format** tab, and then click the **Edit Interactions** option in the **Interactions** group



12. Change the interaction to **Filter** by selecting the appropriate option in the clustered bar chart.



13. Just to try a different interaction, remove the filter between the slicer and the two tables (region and manager net profit).

    Note: No instructions given, deliberately.

14. Disable **Edit Iteractions** and test your changes.

15. Select a **Product Item Group** in a pie chart and notice that the clustered bar chart gets filtered and not highlighted.

16. Change the year in the slicer, and notice that the tables do not change.

17. Change the Page name to **US Sales**.

Tasks 7: Using Custom Visualizations

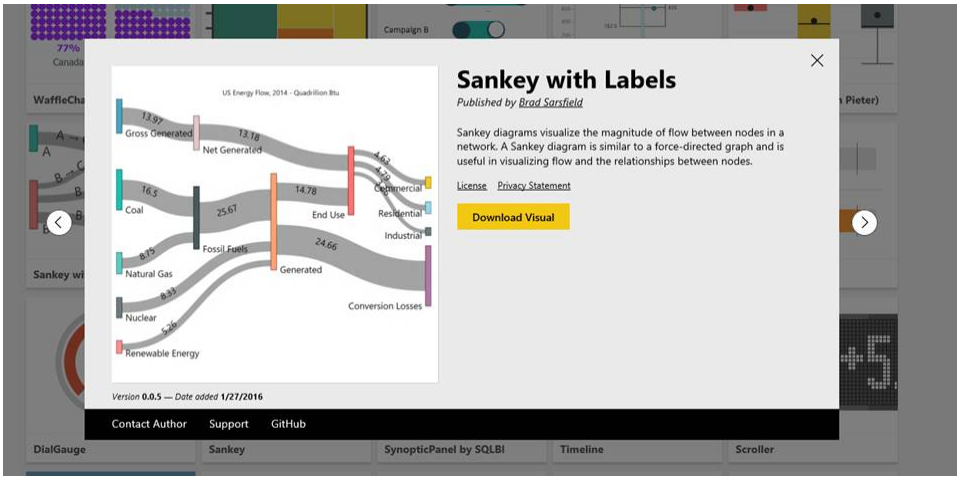
In this task, you will create one more report by using custom visualizations.

Note: Lab will be deliberately less guided.

1. Create a **new page** called **US Sales Scenario**. We are going to use our disconnected table in this report.

2. We are going to use custom visualizations in this report. Go to http://app.powerbi.com/visuals for more information on them.

3. You can click a certain visualization to obtain more information about it.

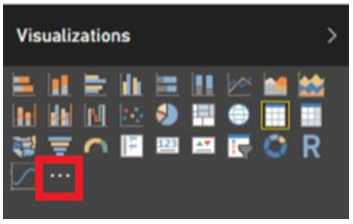


4. We are not going to download any of them for now, because the ones we need are already in the **E:\Assets\Lab 1** folder (the .pbiviz files).

5. We are going to use the following:

   - Timeline
   - Sparkline
   - Stars
   - Waffle Chart

6. Let us start by importing them into Power BI desktop. In the **Visualizations** area, click the ellipsis (…) icon.



7. Ignore the warning.

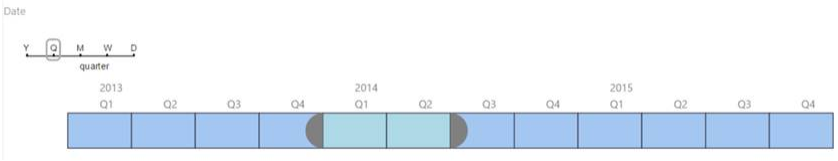
8. Select one of the visualization files and import. Do the same for the remaining three.

9. In the end, you should have the following report:

10. These are the general steps you need to take:

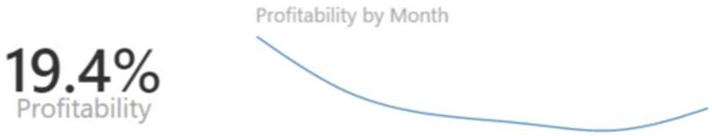
    a Place the logo of the company on the top left.

    b Place a **Timeline** associated to the **Date** field next to the company logo. It should display the quarters, as shown:

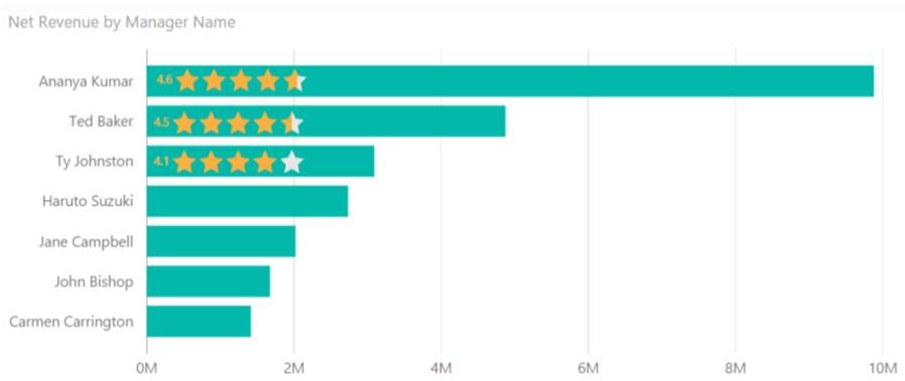


    c Below the logo, we have a **Card** and a **Sparkline** for Net Revenue.

    d Below it, we have another set dedicated to **Profitability**



    e Under the last set, we have a **Clustered bar chart** that displays **Net Revenue** by **Manager** and also overlays three **Star** charts for the top three Managers:



For generating the stars, you can filter any measure for the manager you want and use the **Mod** function to always generate a number from 0 to 5.

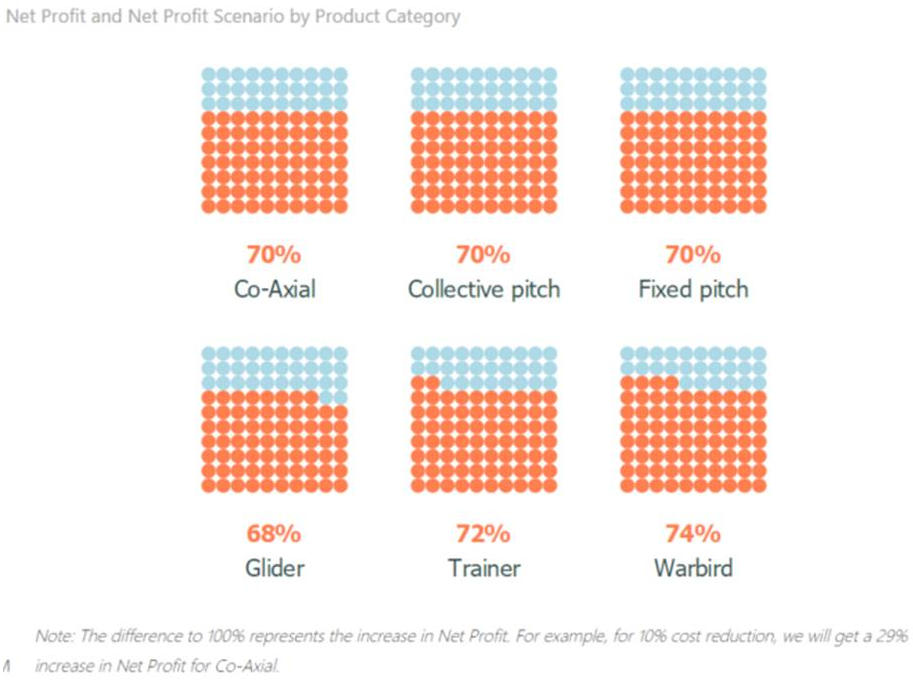
Sample:

DAX

```
AnanyaKumarNetRevenueRating = MOD(CALCULATE(SUM(Sales[Net Revenue]),
Manager[Manager Name]="Ananya Kumar"),5)
```

f Finally, on the right side, under the Timeline, we have a horizontally oriented **Slicer** for **Cost Reduction**.

g And finally we have a **Waffle Chart** that displays **Net Profit for values** and **Net Profit Scenario** for **Maximum value**, grouped by **Product Category**.



11. Good luck!

Finishing Up

In this task, you will finish up by saving the Power BI Desktop file, and then closing the application.

1. To save the Power BI Desktop file, on the **File** tab, select **Save**. Make sure the final file is saved under **E:\Labs\Lab 1\US Sales Analysis.pbix**.
2. To close the application, on the **File** tab, select **Exit**.

In this lab, you created a Power BI Desktop solution to enable the reporting and analysis of regional sales activity. This involved creating Power BI Desktop queries that sourced data from SQL Server, an Excel workbook and a CSV file.

You then created a Power BI Desktop report with several visualizations that will allow you to monitor the business.