* **Connect to the UniversalData\_Customers csv file**
* Name the table "**Customers**", and make sure that headers have been promoted
* Confirm that data types are accurate (**Note:** "*customer\_id*" should be whole numbers, and both "*customer\_acct\_num*" and "*customer\_postal\_code*" should be text)
* Add a new column named "*full\_name"*to merge the the "*first\_name"* and "*last\_name"* columns, separated by a space
* Create a new column named "*birth\_year"*to extract the year from the "*birthdate"*column, and format as text
* Create a **conditional column**named "*has\_children"*which equals "**N**" if "*total\_children"* = 0, otherwise "**Y**“

The set of steps that should be followed to add a .CSV file into POWER BI :-

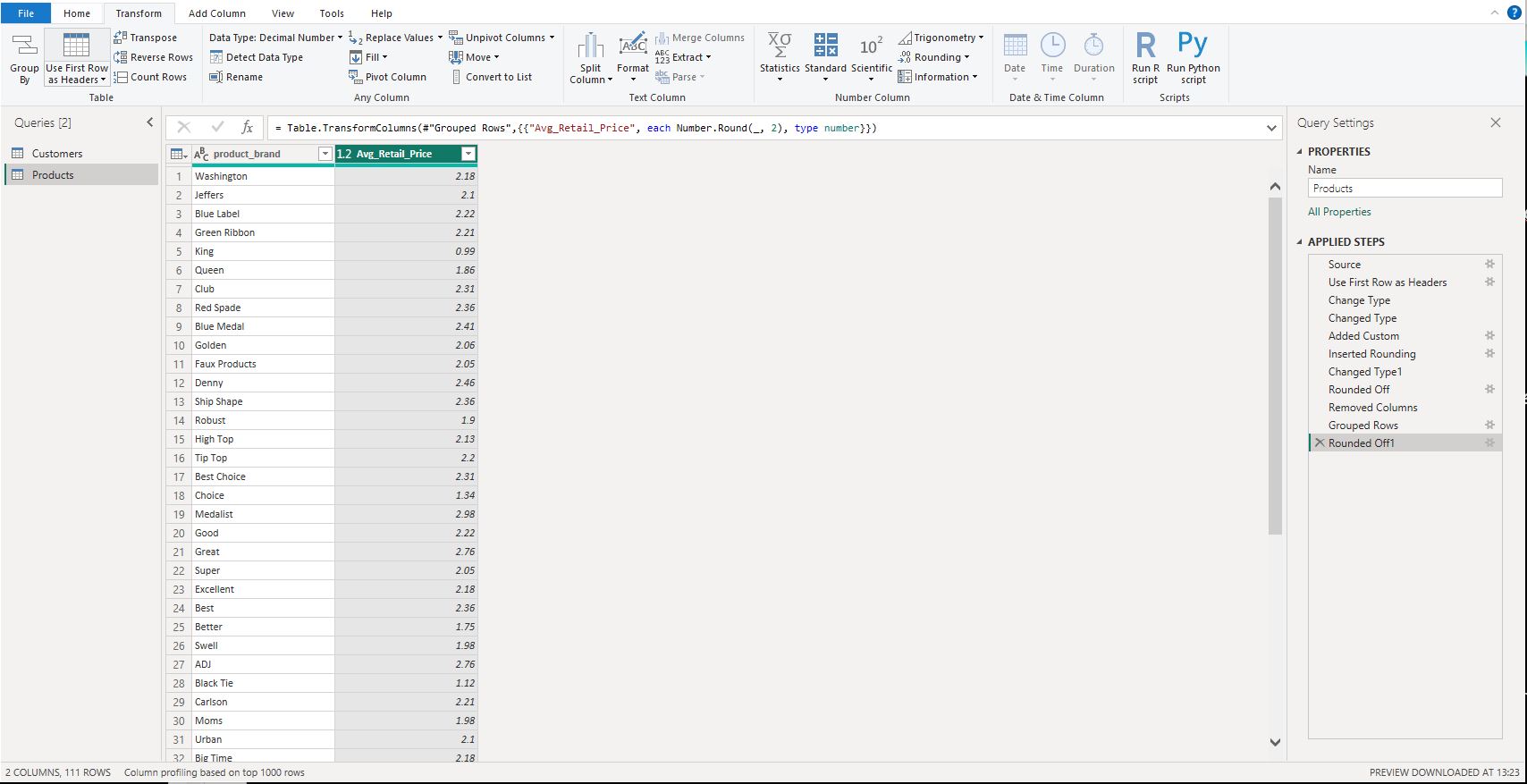
1. Go to Get Data
2. Select CSV/Text File
3. Load or Transform Data
4. After selecting transform data ,remove the null values and remove errors from every single column or If a specific column contains more than 90% of null values then remove the whole column from the database or the specific table.
5. Then load the data after transformation
6. After the “UniversalData\_Customers has been loaded into the POWER BI server, then following step by step procedures ,

To Rename the table name “UniversalData\_Customers” to “Customers”  
Right click on the Table Name and select the rename option from the drop list and name the table as “Customers”.

1. To check the data types of all the columns ,  
   Go to transform data and from transform data one by one check the data types of each column from the menu bar ribbons and give required data types.  
     
   Selecting Whole Numbers data type for “customer\_id”  
   Selecting Whole Numbers data type for “customer\_acc\_num”  
   Selecting Text data type for “customer\_postal\_code”
2. To connect 2 columns first name and last name to make the full\_name column concat function :-  
     
   CONCATENATE(<[Customers]firstname & “ ”>,, <[Customers]lastname>)
3. To Fetch the year from the “birthdate” column need to use Date function to fetch year from a column .  
     
   First we Transform Data from Customers table   
   Add Custom column from Add column Menu bar  
   Give Column Name as “Birth\_Year”  
   Use date function  
   Date.Year([Customers]Birthdate)
4. To add a conditional column for “has\_children” for children   
   Go to Transform Data for Customers Table  
   Go to Add column and Add a conditional column   
   and Add a condition  
     
   If (total\_children>=0)  
   then   
    has children = “N”  
   Else  
    has children = “Y”

* **Connect to the UniversalData\_Products csv file**
* Name the table "**Products**" and make sure that headers have been promoted
* Confirm that data types are accurate (**Note:** "*product\_id*" should be whole numbers, "*product\_sku*" should be text), "*product\_retail\_price*" and "*product\_cost*" should be decimal numbers)
* Use the statistics tools to return the number of distinct product brands, followed by distinct product names
  + ***Spot check:****You should see****111****brands and****1,560****product names*
* Add a calculated column named "*discount\_price*", equal to 90% of the original retail price
  + Format as a fixed decimal number, and then use the rounding tool to round to 2 digits
* Select "*product\_brand*" and use the **Group By** option to calculate the average retail price by brand, and name the new column "*Avg Retail Price*"
  + ***Spot check:****You should see an average retail price of****$2.18****for Washington products, and****$2.21****for Green Ribbon*
* Delete the last applied step to return the table to its pre-grouped state
* Replace "*null*" values with zeros in both the "*recyclable*" and "*low-fat*" columns

Solution :-  
  
The set of steps that should be followed to add a .CSV file into POWER BI :-

1. Go to Get Data
2. Select CSV/Text File
3. Load or Transform Data
4. After selecting transform data ,remove the null values and remove errors from every single column or If a specific column contains more than 90% of null values then remove the whole column from the database or the specific table.
5. Then load the data after transformation
6. After the “UniversalData\_Products has been loaded into the POWER BI server, then following step by step procedures ,
7. To Rename the table name “UniversalData\_Products” to “Products”  
   Right click on the Table Name and select the rename option from the drop list and name the table as “Products”.
8. To check the data types of all the columns ,  
   Go to transform data and from transform data one by one check the data types of each column from the menu bar ribbons and give required data types.  
     
   Selecting Whole Numbers data type for “product\_id”  
   Selecting text data type for “product\_sku”  
   Selecting Decimal Number data type for “product\_retailprice”  
   Selecting Decimal Number data type for “product\_cost”``
9. To check the distinct values from product brand in the products table  
   Go to Transform data  
   Find Statistics Ribbon and select the product brand & Product Names  
   Now select Distinct Values for both the columns 1 by 1   
     
   and it will result in a value   
     
   Product Brands :- 111  
   Product Names :- 1560
10. To add a calculated column for discount column  
      
    Go to transform data  
    Go to Add Column and select Custom Column  
    Give the column name “Discount Price “   
    Give the code   
      
    (‘Customers[retails\_price]’/100)\*90
11. To use group by on a field   
    select Group by from home menu tab with the ribbons   
    Group it by brand name and select Average of Retail Price   
    Give column name as Average Retail Price.  
      
    

F. In specific columns recyclable and low fat  
  
Go to transform data and Select both columns   
and one by one remove the null values from drop down list .

* Connect to the **UniversalData\_Stores** csv file
* Name the table "**Stores**" and make sure that headers have been promoted
* Confirm that data types are accurate (**Note:** "*store\_id*" and "*region\_id*" should be whole numbers)
* Add a calculated column named "*full\_address*", by merging "*store\_city*", "*store\_state*", and "*store\_country*", separated by a comma and space (***hint:****use a custom separator*)
* Add a calculated column named "*area\_code*", by extracting the characters before the dash ("-") in the "*store\_phone*" field

The set of steps that should be followed to add a .CSV file into POWER BI :-

1. Go to Get Data
2. Select CSV/Text File
3. Load or Transform Data
4. After selecting transform data ,remove the null values and remove errors from every single column or If a specific column contains more than 90% of null values then remove the whole column from the database or the specific table.
5. Then load the data after transformation

After Loading the data into POWER BI , go to transform data and Promote the Table headers by right clicking the overall change drop down and Promote the headers and give required data types.

1. To Rename the table name “UniversalData\_Stores” to “Stores”  
   Right click on the Table Name and select the rename option from the drop list and name the table as “Stores”.
2. To check the data types of all the columns ,  
   Go to transform data and from transform data one by one check the data types of each column from the menu bar ribbons and give required data types.  
     
   Selecting Whole Numbers data type for “store\_id”  
   Selecting whole number data type for “region\_id”
3. To concatenate three columns we use a DAX code with the concatenate function  
     
   Full Address = CONCATENATE(Store[City] CONCATENATE("Store[State] ", Store[Country].
4. To extract area code from the phone number column we use the Delimiter in the Text Function to extract the code  
     
   Go to transform data  
   Select Custom column from Add Column Menu Tab  
   Give the Column Name Area Code  
   After giving column name use Text function with Before Delimiter  
     
   Text.BeforeDelimiter(‘Stores’[PhoneNumber,’-‘)

Connect to the **UniversalData\_Regions** csv file

1. Name the table "**Regions**" and make sure that headers have been promoted
2. Confirm that data types are accurate (**Note:** "*region\_id*" should be whole numbers)

The set of steps that should be followed to add a .CSV file into POWER BI :-

1. Go to Get Data
2. Select CSV/Text File
3. Load or Transform Data
4. After selecting transform data ,remove the null values and remove errors from every single column or If a specific column contains more than 90% of null values then remove the whole column from the database or the specific table.
5. Then load the data after transformation

After Loading the data into POWER BI , go to transform data and Promote the Table headers by right clicking the overall change drop down and Promote the headers and give required data types.

1. To Rename the table name “UniversalData\_Regions” to “Regions”  
   Right click on the Table Name and select the rename option from the drop list and name the table as “Regions”.
2. To check the data types of all the columns ,  
   Go to transform data and from transform data one by one check the data types of each column from the menu bar ribbons and give required data types.  
     
   Selecting Whole Numbers data type for “region\_id”

 Connect to the **UniversalData\_Calendar** csv file

* Name the table "**Calendar**" and make sure that headers have been promoted
* Use the date tools in the query editor to add the following columns:
  + *Start of Week (starting Sunday*
  + *Name of Day*
  + *Start of Month*
  + *Name of Month*
  + *Quarter of Year*
  + *Year*

The set of steps that should be followed to add a .CSV file into POWER BI :-

1. Go to Get Data
2. Select CSV/Text File
3. Load or Transform Data
4. After selecting transform data ,remove the null values and remove errors from every single column or If a specific column contains more than 90% of null values then remove the whole column from the database or the specific table.

Then load the data after transformation  
  
  
After Loading the data into POWER BI , go to transform data and Promote the Table headers by right clicking the overall change drop down and Promote the headers and give required data types.

1. To Rename the table name “UniversalData\_Calendar” to “Calendar”  
   Right click on the Table Name and select the rename option from the drop list and name the table as “Calendar”.
2. Now using the Date Tools from the Query Editor we can directly apply the functions on the column to find dates into new columns.  
     
   a. Start of the Week  
   b. Name of the day  
   c. Start of the Month  
   d. Name of the month  
   e. Quarter of the month

***Connect to the UniversalData\_Returns csv file***

* Name the table "Return\_Data" and make sure that headers have been promoted
* Confirm that data types are accurate (all ID columns and quantity should be whole numbers)

The set of steps that should be followed to add a .CSV file into POWER BI :-

1. Go to Get Data
2. Select CSV/Text File
3. Load or Transform Data
4. After selecting transform data ,remove the null values and remove errors from every single column or If a specific column contains more than 90% of null values then remove the whole column from the database or the specific table.

Then load the data after transformation  
  
  
After Loading the data into POWER BI , go to transform data and Promote the Table headers by right clicking the overall change drop down and Promote the headers and give required data types.

1. To Rename the table name “UniversalData\_Return” to “Return”  
   Right click on the Table Name and select the rename option from the drop list and name the table as “Return”.
2. To check the data types of all the columns ,  
   Go to transform data and from transform data one by one check the data types of each column from the menu bar ribbons and give required data types.  
     
   Selecting Whole Numbers data type for all the table columns in this table.

Add a new folder on your desktop (or in your documents) named "**UniversalData\_Transactions**", containing both the **UniversalData\_Transactions\_1997** and **UniversalData\_Transactions\_1998** csv files

* Connect to the folder path, and choose "Edit" (*vs. Combine and Edit*)
* Click the "*Content*" column header (double arrow icon) to combine the files, then remove the "*Source.Name*" column
* Name the table "**Transaction\_Data**", and confirm that headers have been promoted
* Confirm that data types are accurate (all ID columns and *quantity* should be whole numbers)
  + ***Spot check:****You should see data from 1/1/1997 through 12/30/1998 in the "transaction\_date" column*

The set of steps that should be followed to add a .CSV file into POWER BI :-

1. Bringing both the data files into 1 custom folder from Transactions 1997 & 1998
2. Rename the folder as UniversalData\_Transactions
3. Go to Get Data
4. Select Folder option
5. Load & combine or Transform and combine data
6. After Combining data , select Content and check on both the table and load the data
7. After selecting transform data ,remove the null values and remove errors from every single column or If a specific column contains more than 90% of null values then remove the whole column from the database or the specific table.
8. Then load the data after transformation

1. After Loading the data into POWER BI , go to transform data and Promote the Table headers by right clicking the overall change drop down and Promote the headers and give required data types.
2. To check the data types of all the columns ,  
   Go to transform data and from transform data one by one check the data types of each column from the menu bar ribbons and give required data types.  
     
   Selecting Whole Numbers data type for all the table columns in this table.

* DAX MEASURES :-  
    
    
  1. In the **Calendar** table, or Sundays (otherwise add a column named "***Weekend***"

Equals "***Y***" for Saturdays "***N***")

To Add a Weekend Column  
Go to transform data in the Calendar Table  
Select a Conditional Column from Add column Menu Tab Ribbon  
Give the column name ‘Weekend’  
  
IF(Day.Name(‘Calendar’[Date]=”Sunday”)  
then ‘Calendar’[Weekend]=’Y’  
  
else if (Day.Name(‘Calendar’[Date]=”Saturday”)  
then ‘Calendar’[Weekend]=’N’  
  
else  
then ‘Calendar’[Weekend]=’NA’

2. In the **Calendar** table, add a column named "***End of Month***"

* + Returns the last date of the current month for each row

To Add a End of Month Column  
Go to transform data in the Calendar Table  
Select a Custom Column from Add column Menu Tab Ribbon  
Give the column name ‘End of Month’  
Use the Date Function  
  
Date.EndOfMonth’Calendar’[Date]

1. In the **Customers** table, add a column named "***Current Age***"

Calculates current customer ages using the "*birthdate*" column and the TODAY() function

To Add a Current Age Column  
 Select New column from the data menu bar   
 Select Add New Column  
 Rename the new column as ‘Current Age ’  
 Use Date Function  
  
 DATEDIFF(‘Calendar’[Date],TODAY(),YEAR)

1. In the **Customers** table, add a column named "***Priority***"

Equals "***High***" for customers who own homes and have Golden membership cards (otherwise "***Standard***")    
  
To Add a Current Age Column  
 Select New column from the data menu bar   
 Select Add New Column  
 Rename the new column as ‘Priority ’  
Use IF ELSE Condition  
  
  
IF (‘Customers[homeowner]=”Y” & ‘Customers’[member\_card] =”Golden”,”Priority”,”Standard”)

1. In the **Customers** table, add a column named "***Short Country***"

Returns the first three characters of the customer country, and converts to all uppercase

To Add a Short Country Column  
 Select New column from the data menu bar   
 Select Add New Column  
 Rename the new column as ‘Short Country’  
Use the UPPER FUNCTION  
  
  
Short Country = UPPER(LEFT(Customers[customer\_country],3))

1. In the **Customers** table, add a column named "***House Number***"

Extracts all characters/numbers before the first space in the "*customer\_address*" column (***hint:****use SEARCH*)

To Add a House Number Column  
 Select New column from the data menu bar   
 Select Add New Column  
 Rename the new column as ‘House Number  
Use the SEARCH FUNCTION  
  
  
House Number = LEFT(Customers[customer\_address], SEARCH(" ", Customers[customer\_address], 1) - 1)

1. In the **Products** table, add a column named "***Price Tier***"

Equals "***High***" if the retail price is >**$3**, "***Mid***" if the retail price is >**$1**, and "***Low***" otherwise

To Add a Price Tier Column  
 Select New column from the data menu bar   
 Select Add New Column  
 Rename the new column as ‘House Number  
Use the IF ELSE CONDITION  
  
  
Price Tier = IF([product\_retail\_price] > 3, "High", IF([product\_retail\_price] > 1, "Mid", "Low"))

1. In the **Stores** table, add a column named "***Years\_Since\_Remodel***"

Calculates the number of years between the current date (TODAY()) and the last remodel date

To Add a Years Since Remodel Column  
 Select New column from the data menu bar   
 Select Add New Column  
 Rename the new column as ‘Years Since Remodel’  
Use the DATEDIFF CONDITION  
  
  
Years\_Since Remodel = DATEDIFF(Stores[last\_remodel\_date],TODAY(),YEAR)

1. Create new measures named "**Quantity Sold**" and "**Quantity Returned**" to calculate the sum of quantity from each data table

***Spot check:****You should see total Quantity Sold =****833,489****and total Quantity Returned =****8,289***

To create measures named “Quantity Sold” & “Quantity Returned”  
We create measures of these names   
  
Measure 1 in Transactions Data Table  
  
Quantity Sold = CALCULATE(SUM('Transaction Data'[quantity]))  
  
Measure 2 in Return Table  
  
Quantity Returned = SUM('Return Data'[quantity])

1. Create new measures named "**Total Transactions**" and "**Total Returns**" to calculate the count of rows from each data table

***Spot check:****You should see****269,720****transactions and****7,087****returns*

To create measures named “Total Transactions” & “Total Returns”  
We create measures of these names   
  
Measure 1 in Transactions Data Table  
  
Total Transactions = COUNT('Transaction Data'[quantity])  
  
Measure 2 in Return Table  
  
Total Returns = COUNT('Return Data'[quantity])

1. Create a new measure named "**Return Rate**" to calculate the ratio of quantity returned to quantity sold (format as %)

***Spot check:****You should see an overall return rate of****0.99%***

To create measure named “Return Rate”   
  
Measure in Returns Data Table  
  
Return Rate = (DIVIDE([Quantity Returned],[Quantity Sold]))

1. Create a new measure named "**Weekend Transactions**" to calculate transactions on weekends

***Spot check:****You should see****76,608****total weekend transactions*

To create measure named “Return Rate”   
  
Measure in Transactions Data Table  
  
  
Return Rate = (DIVIDE([Quantity Returned],[Quantity Sold]))

1. Create a new measure named "**% Weekend Transactions**" to calculate weekend transactions as a percentage of total transactions (format as %)

***Spot check:****You should see****28.4%****weekend transactions*

To create measure named “% Weekend Transactions”   
  
 Measure in Transactions Data Table  
   
 % Weekend Transactions = CALCULATE(DIVIDE([Weekend Transactions],[Total Transactions])\*100)

1. Create new measures named "**All Transactions**" and "**All Returns**" to calculate grand total transactions and returns (regardless of filter context)

***Spot check:****You should see****269,720****transactions and****7,087****returns across all rows (test with product\_brand on rows)*

To create measures named “All Transactions” & “All Returns”  
We create measures of these names   
  
  
Measure 1 in Transactions Data Table  
  
Total Transactions = SUM('Transaction Data'[quantity])  
  
Measure 2 in Return Table  
  
Total Returns = SUM('Return Data'[quantity])

1. Create a new measure to calcul ate "**Total Revenue**" based on transaction quantity and product retail price, and format as $ (***hint:****you'll need an iterator*)

***Spot check:****You should see a total revenue of****$1,764,546***

To create measure named Total Revenue”   
  
Measure in Transactions Data Table  
  
Total Revenue = SUMX('Transaction Data', 'Transaction Data'[quantity] \* RELATED(Products[product\_retail\_price]))

1. Create a new measure to calculate "**Total Cost**" based on transaction quantity and product cost, and format as $ (***hint:****you'll need an iterator*)

***Spot check:****You should see a total cost of****$711,728***

To create measure named Total Cost”   
  
Measure in Transactions Data Table

Total Cost = SUMX('Transaction Data', 'Transaction Data'[quantity] \* RELATED(Products[product\_cost]))

1. Create a new measure named "**Total Profit**" to calculate total revenue minus total cost, and format as $

***Spot check:****You should see a total profit of****$1,052,819***

To create measure named Total Profit”   
  
Measure in Transactions Data Table

Total Profit = CALCULATE([Total Revenue]-[Total Cost])

1. Create a new measure to calculate "**Profit Margin**" by dividing total profit by total revenue calculate total revenue (format as %)

***Spot check:****You should see an overall profit margin of****59.67%***

To create measure named “Profit Margin”   
  
Measure in Transactions Data Table

Profit Margin = CALCULATE([Total Profit]/[Total Revenue])

1. Create a new measure named "**Unique Products**" to calculate the number of unique product names in the **Products** table

***Spot check:****You should see****1,560****unique products*To create measure named Unique Products”   
  
Measure in Products Data Table

Unique Products = DISTINCTCOUNT(Products[product\_name])

1. Create a new measure named "**YTD Revenue**" to calculate year-to-date total revenue, and format as $

***Spot check:****Create a matrix with "****Start of Month****" on rows; you should see****$872,924****in YTD Revenue in September 1998*

To create measure named “YTD Revenue”   
  
Measure in Transactions Data Table

YTD Revenue = TOTALYTD(SUM('Transaction Data'[Revenue]),'Calendar'[date])

1. Create a new measure named "**60-Day Revenue**" to calculate a running revenue total over a 60-day period, and format as $

***Spot check:****Create a matrix with "****date****" on rows; you should see****$97,570****in 60-Day Revenue on 4/14/1997*

To create measure named “60-Day Revenue”   
  
Measure in Transactions Data Table

60-Day Revenue = CALCULATE(SUM('Transaction Data'[Revenue]),DATESINPERIOD('Calendar'[date].[Date],MIN('Calendar'[date]),60,DAY))

1. Create new measures named “**Last Month Transactions**", "**Last Month Revenue**", "**Last Month Profit**", and "**Last Month Returns**"

***Spot check:****Create a matrix with "****Start of Month****" on rows to confirm accuracy*

To create measures named “Last Month Transactions”, “Last Month Revenue”, “Last Month Profit” &” Last Month Returns”  
  
We create measures of these names   
  
Measure 1 in Transactions Data Table  
  
Last Month Transactions = CALCULATE(COUNT('Transaction Data'[Revenue]),DATESINPERIOD('Calendar'[date].[Date],MIN('Calendar'[date]),30,DAY))

Measure 2 in Transactions Data Table  
  
Last Month Revenue = CALCULATE(SUM('Transaction Data'[Revenue]),DATESINPERIOD('Calendar'[date].[Date],MIN('Calendar'[date]),30,DAY))  
  
Measure 3 in Transaction Data Table  
  
Last Month Profit = CALCULATE(SUM('Transaction Data'[Profit]),DATESINPERIOD('Calendar'[date].[Date],MIN('Calendar'[date]),30,DAY))

Measure 4 in Returns Data Table  
  
Last Month Returns = CALCULATE(COUNT('Return Data'[quantity]),DATESINPERIOD('Calendar'[date].[Date],MIN('Calendar'[date]),30,DAY))

1. Create a new measure named "**Revenue Target**" based on a 5% lift over the previous month revenue, and format as $

***Spot check:****You should see a Revenue Target of****$99,223****in March 1998*

Target Revenue = CALCULATE('Transaction Data'[Total Revenue]+('Transaction Data'[Last Month Revenue]\*0.05))

For Building the Report , After applying all the Visualisations in the Report View the Final PBIX report will look like this

