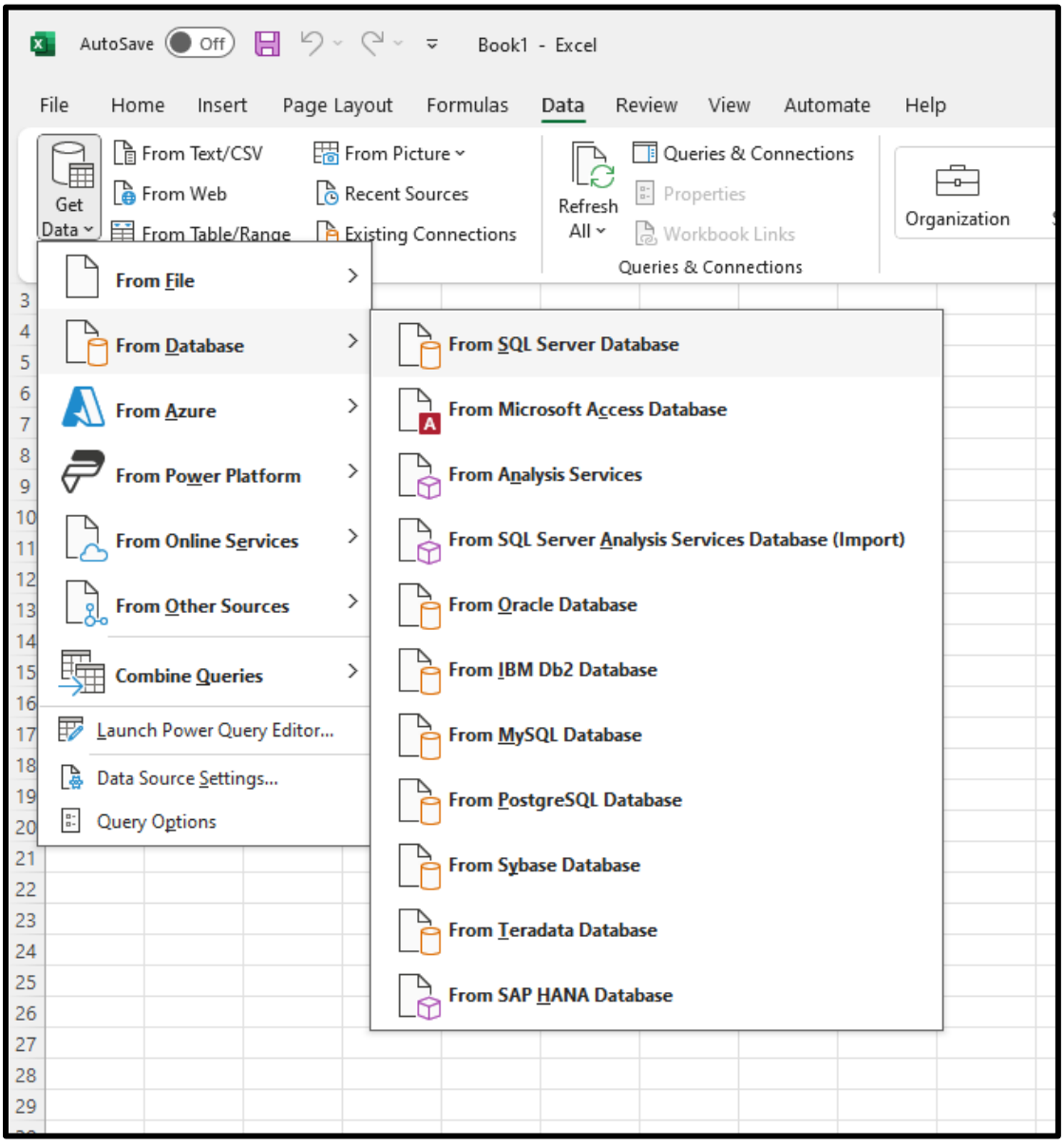
Practical 1

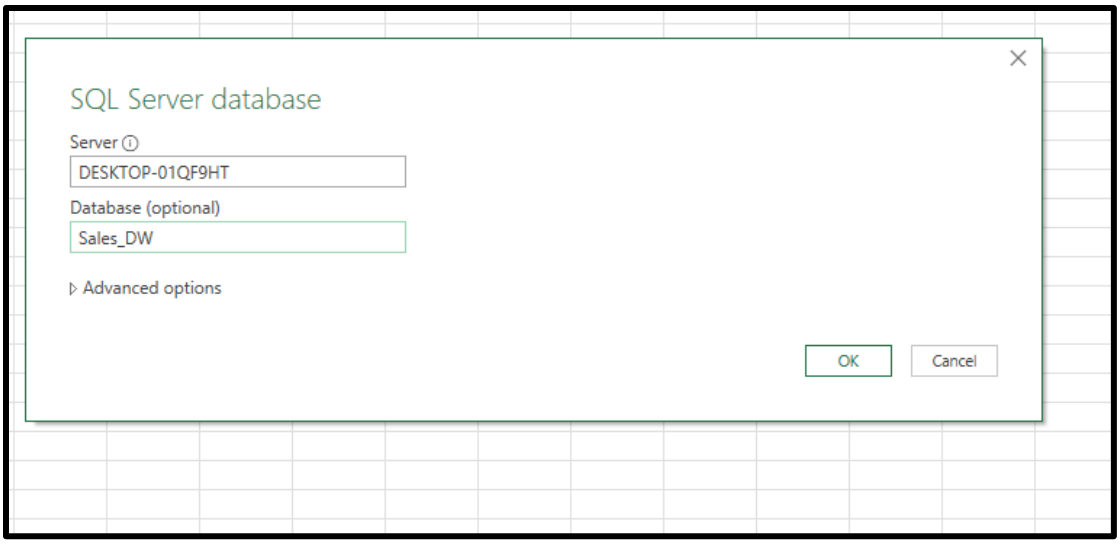
Perform the analysis for the following:

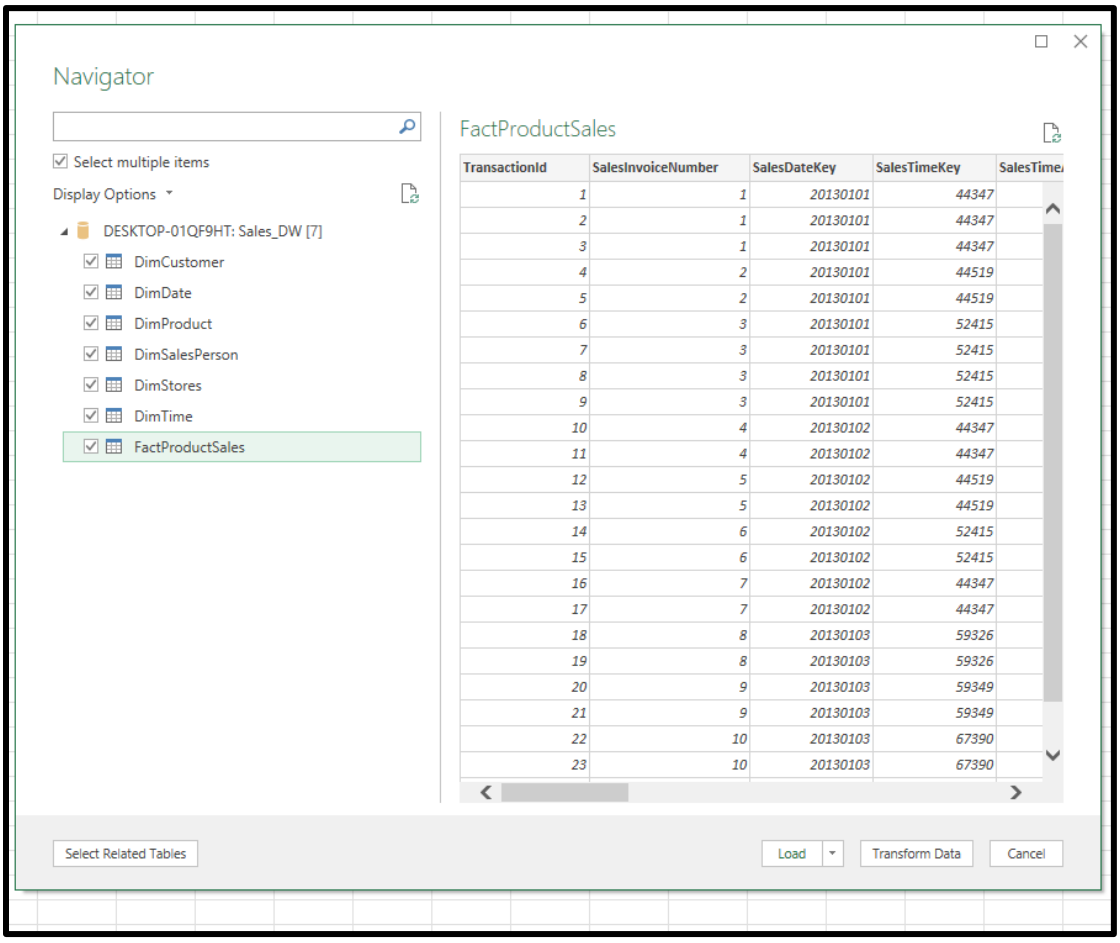
a. **Import the data warehouse data in Microsoft Excel and create the Pivot table and Pivot Chart.**

b. **Import the cube in Microsoft Excel and create the Pivot table and Pivot Chart to perform data analysis.**

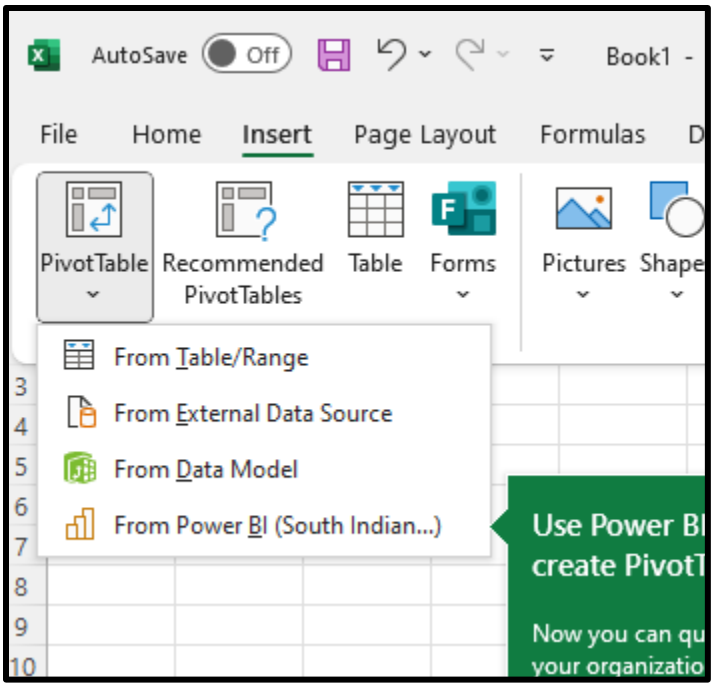
a.

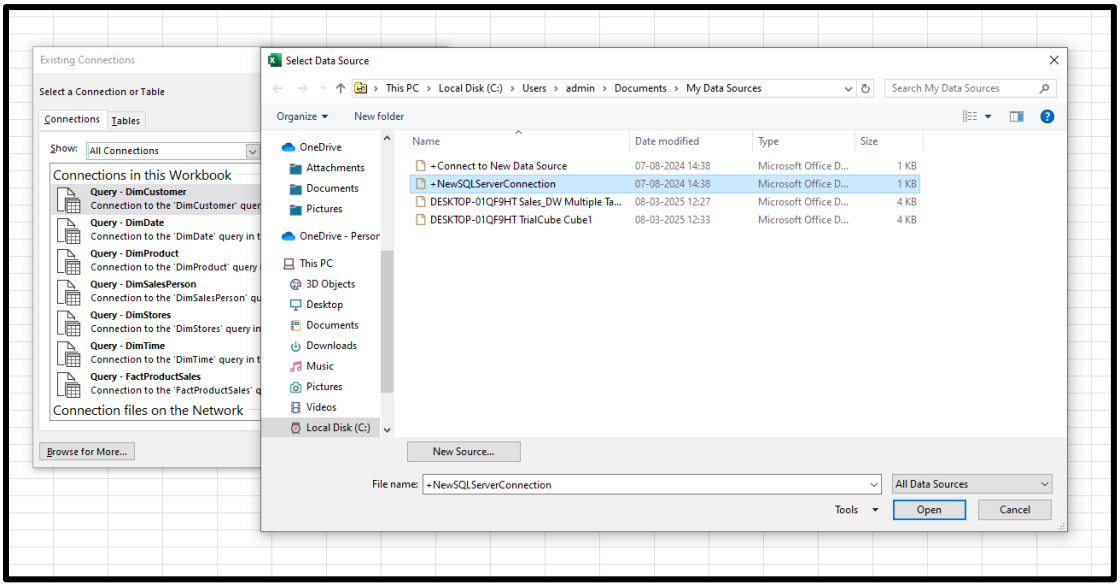


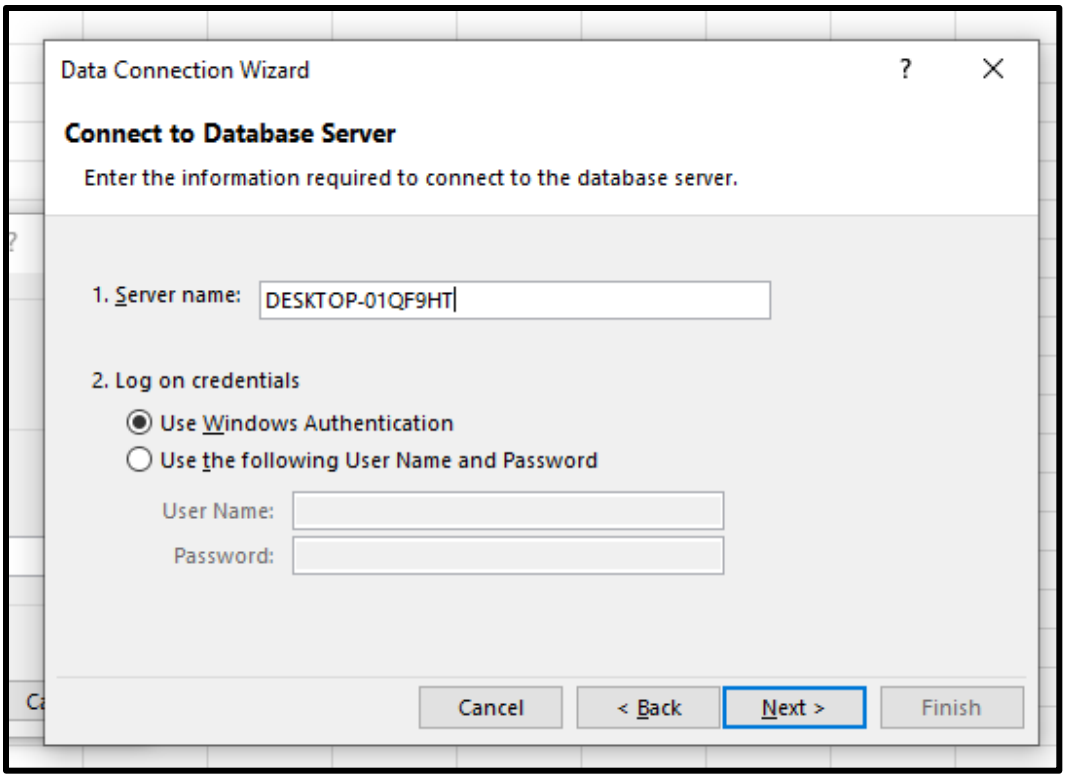


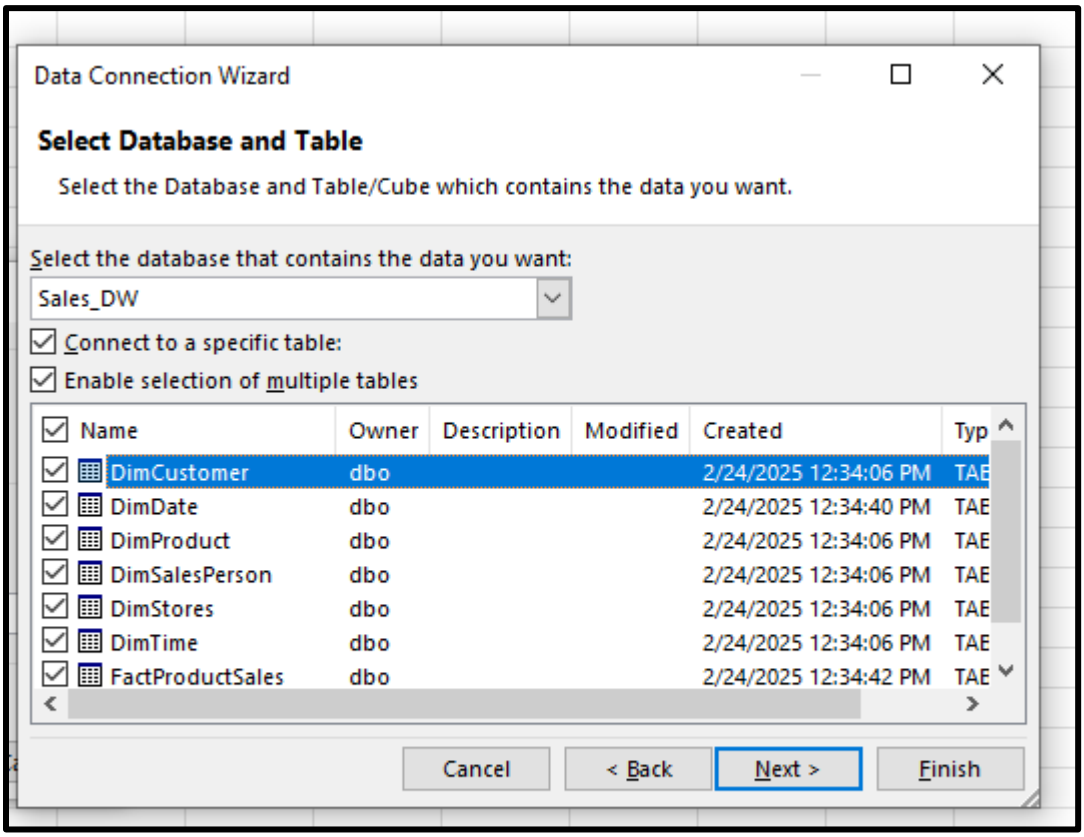


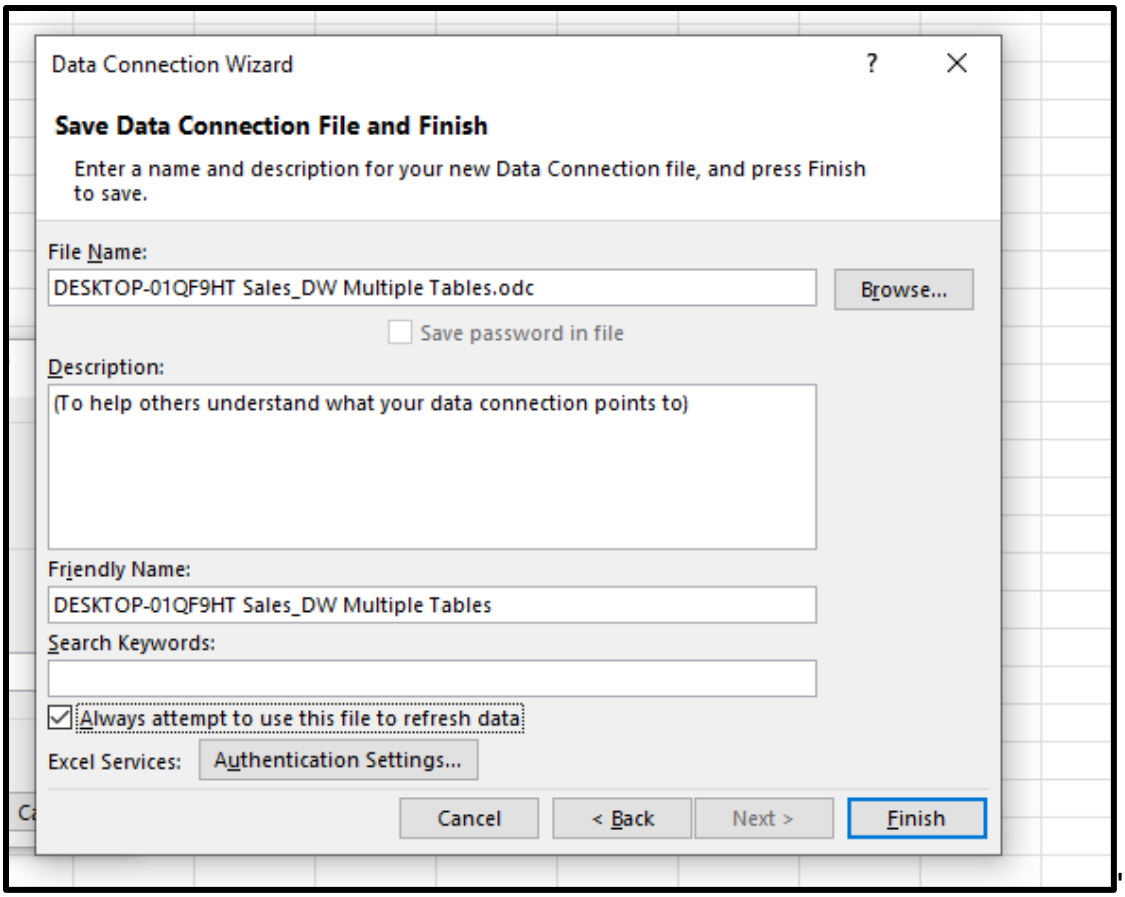
TransactionId	SalesInvoiceNumber	SalesDateKey	SalesTimeKey	SalesTime
1	1	20130101	44347	
2	1	20130101	44347	
3	1	20130101	44347	
4	2	20130101	44519	
5	2	20130101	44519	
6	3	20130101	52415	
7	3	20130101	52415	
8	3	20130101	52415	
9	3	20130101	52415	
10	4	20130102	44347	
11	4	20130102	44347	
12	5	20130102	44519	
13	5	20130102	44519	
14	6	20130102	52415	
15	6	20130102	52415	
16	7	20130102	44347	
17	7	20130102	44347	
18	8	20130103	59326	
19	8	20130103	59326	
20	9	20130103	59349	
21	9	20130103	59349	
22	10	20130103	67390	
23	10	20130103	67390	


AutoSave
Off
Book1 -
File
Home
Insert
Page Layout
Formulas
PivotTable
Recommended PivotTables
Table
Forms
Pictures
Shape
From Table/Range
From External Data Source
From Data Model
From Power BI (South Indian...)
Use Power BI
create PivotT
Now you can qu
your organizatio

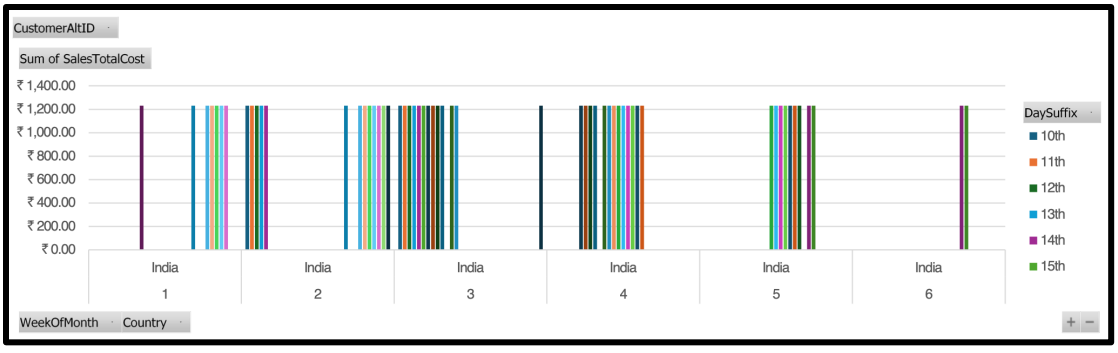

Existing Connections
Select a Connection or Table
Connections
Tables
Show:
All Connections
Connections in this Workbook
Query - DimCustomer
Query - DimDate
Query - DimProduct
Query - DimSalesPerson
Query - DimStores
Query - DimTime
Query - FactProductSales
Connection files on the Network
Browse for More...
Select Data Source
This PC
Local Disk (C:)
Users
admin
Documents
My Data Sources
Search My Data Sources
Organize
New folder
OneDrive
Attachments
Documents
Pictures
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
Name
Date modified
Type
Size
+Connect to New Data Source
+NewSQLServerConnection
DESKTOP-01QF9HT Sales_DW Multiple Ta...
DESKTOP-01QF9HT TrialCube Cube1
New Source...
File name:
+NewSQLServerConnection
All Data Sources
Tools
Open
Cancel


Data Connection Wizard
Connect to Database Server
Enter the information required to connect to the database server.
1. Server name:
DESKTOP-01QF9HT
2. Log on credentials
Use Windows Authentication
Use the following User Name and Password
User Name:
Password:
Cancel
< Back
Next >
Finish

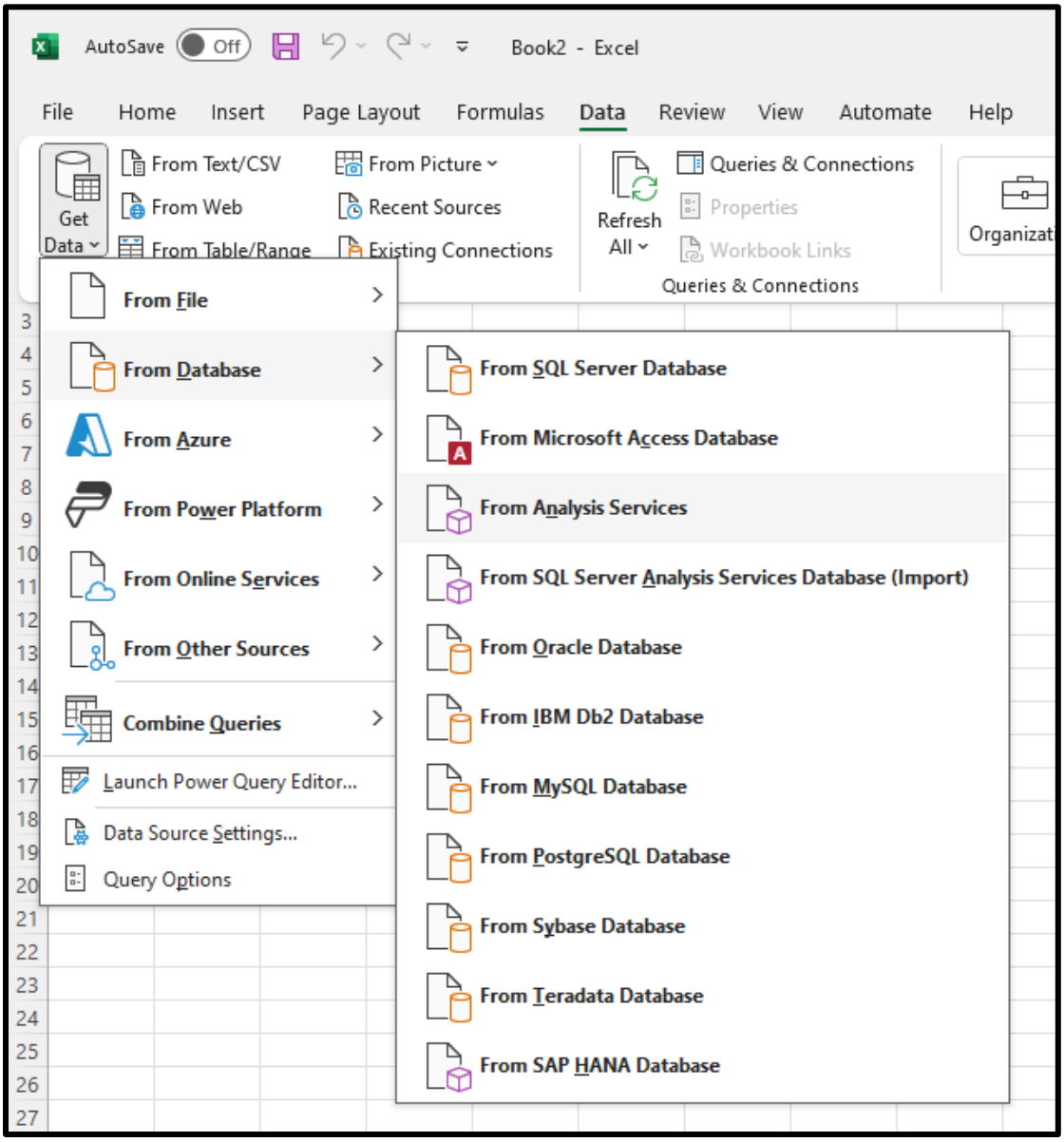

Data Connection Wizard
Select Database and Table
Select the Database and Table/Cube which contains the data you want.
Select the database that contains the data you want:
Sales_DW
Connect to a specific table:
Enable selection of multiple tables
Name
Owner
Description
Modified
Created
Typ
DimCustomer dbo 2/24/2025 12:34:06 PM TAE
DimDate dbo 2/24/2025 12:34:40 PM TAE
DimProduct dbo 2/24/2025 12:34:06 PM TAE
DimSalesPerson dbo 2/24/2025 12:34:06 PM TAE
DimStores dbo 2/24/2025 12:34:06 PM TAE
DimTime dbo 2/24/2025 12:34:06 PM TAE
FactProductSales dbo 2/24/2025 12:34:42 PM TAE
Cancel
< Back
Next >
Finish

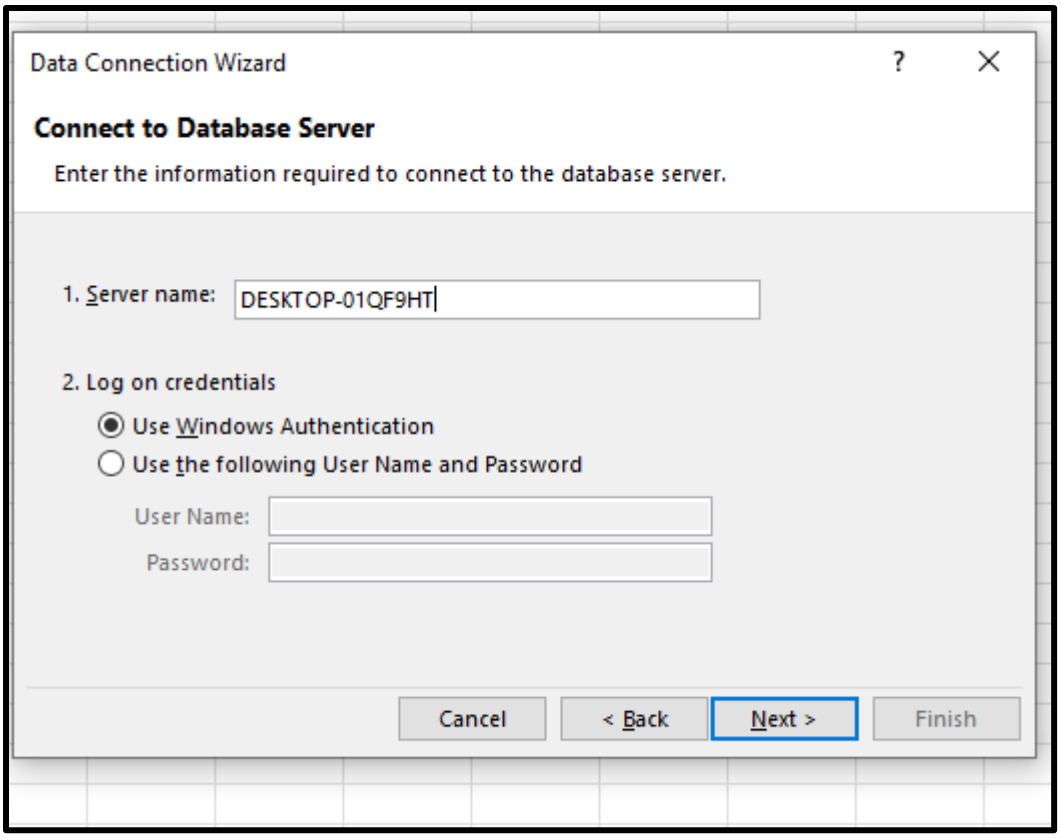

Data Connection Wizard
Save Data Connection File and Finish
Enter a name and description for your new Data Connection file, and press Finish to save.
File Name:
DESKTOP-01QF9HT Sales_DW Multiple Tables.odc
Browse...
Save password in file
Description:
(To help others understand what your data connection points to)
Friendly Name:
DESKTOP-01QF9HT Sales_DW Multiple Tables
Search Keywords:
Always attempt to use this file to refresh data
Excel Services:
Authentication Settings...
Cancel
< Back
Next >
Finish

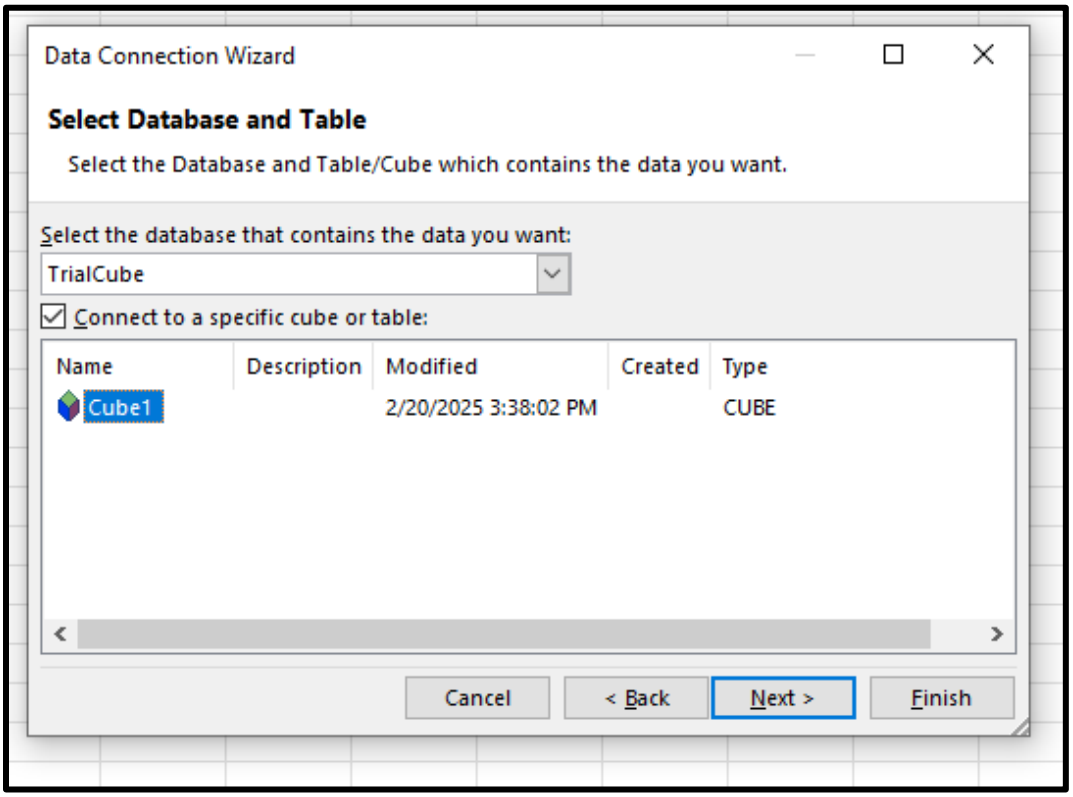
Output:


CustomerAltID
Sum of SalesTotalCost
₹ 1,400.00
₹ 1,200.00
₹ 1,000.00
₹ 800.00
₹ 600.00
₹ 400.00
₹ 200.00
₹ 0.00
India
1
India
2
India
3
India
4
India
5
India
6
DaySuffix
10th
11th
12th
13th
14th
15th
WeekOfMonth
Country

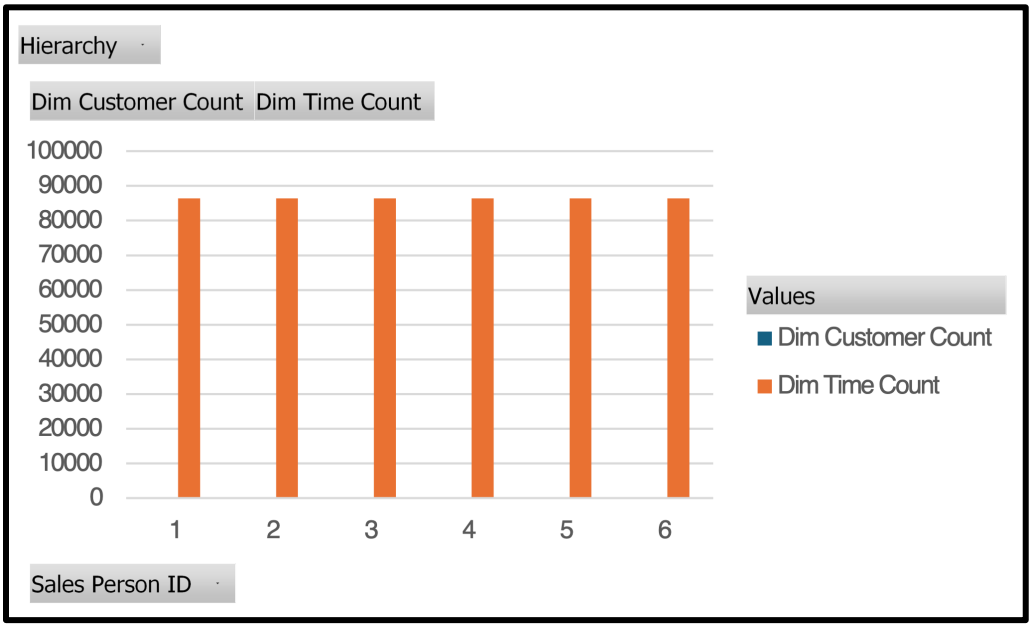
b.

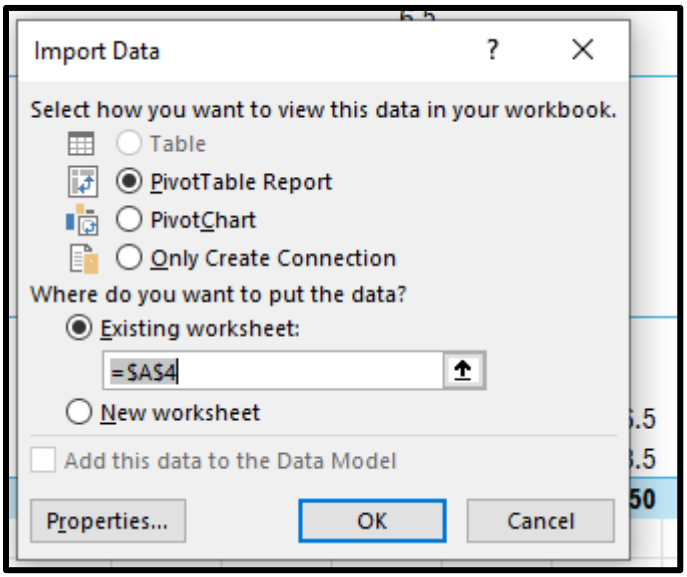

AutoSave
Off
Book2 - Excel
File
Home
Insert
Page Layout
Formulas
Data
Review
View
Automate
Help
Get Data
From Text/CSV
From Web
From Table/Range
From Picture
Recent Sources
Existing Connections
Refresh All
Queries & Connections
Properties
Workbook Links
Queries & Connections
Organizat
From File
From Database
From Azure
From Power Platform
From Online Services
From Other Sources
Combine Queries
Launch Power Query Editor...
Data Source Settings...
Query Options
From SQL Server Database
From Microsoft Access Database
From Analysis Services
From SQL Server Analysis Services Database (Import)
From Oracle Database
From IBM Db2 Database
From MySQL Database
From PostgreSQL Database
From Sybase Database
From Teradata Database
From SAP HANA Database


Data Connection Wizard
Connect to Database Server
Enter the information required to connect to the database server.
1. Server name:
DESKTOP-01QF9HT
2. Log on credentials
Use Windows Authentication
Use the following User Name and Password
User Name:
Password:
Cancel
< Back
Next >
Finish

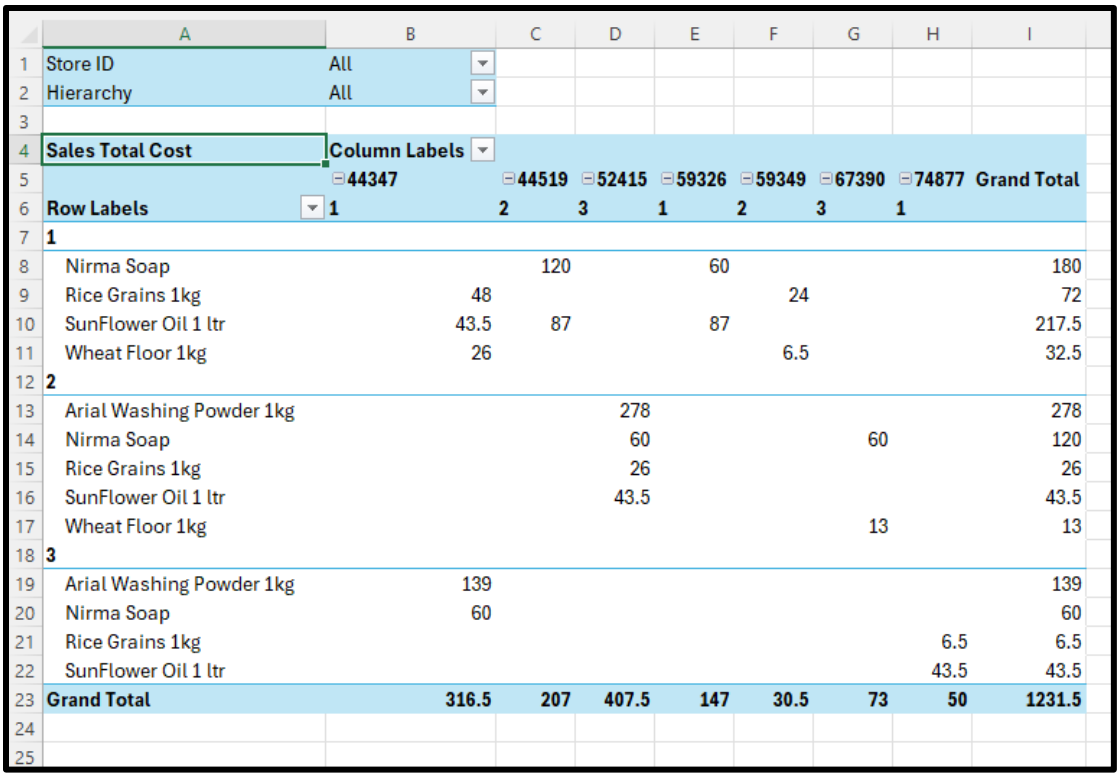

Data Connection Wizard
Select Database and Table
Select the Database and Table/Cube which contains the data you want.
Select the database that contains the data you want:
TrialCube
Connect to a specific cube or table:
Name
Description
Modified
Created
Type
Cube1
2/20/2025 3:38:02 PM
CUBE
Cancel
< Back
Next >
Finish

Output:





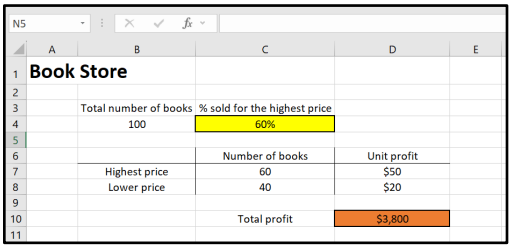
Output:



Store ID	All							
Hierarchy	All							
Sales Total Cost	**Column Labels**							
	44347		**44519**	**52415**	**59326**	**59349**	**67390**	**74877**	**Grand Total**
Row Labels	**1**	**2**	**3**	**1**	**2**	**3**	**1**	
1								
Nirma Soap		120		60				180
Rice Grains 1kg	48				24			72
SunFlower Oil 1 ltr	43.5	87		87				217.5
Wheat Floor 1kg	26				6.5			32.5
2								
Arial Washing Powder 1kg			278					278
Nirma Soap			60			60		120
Rice Grains 1kg			26					26
SunFlower Oil 1 ltr			43.5					43.5
Wheat Floor 1kg						13		13
3								
Arial Washing Powder 1kg	139							139
Nirma Soap	60							60
Rice Grains 1kg							6.5	6.5
SunFlower Oil 1 ltr							43.5	43.5
Grand Total	**316.5**	**207**	**407.5**	**147**	**30.5**	**73**	**50**	**1231.5**

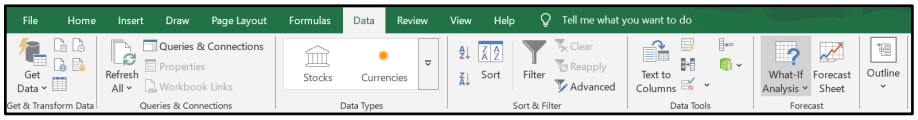
Practical 2

Apply the what – if Analysis for data visualization. Design and generate necessary reports based on the data warehouse data. Use Excel.

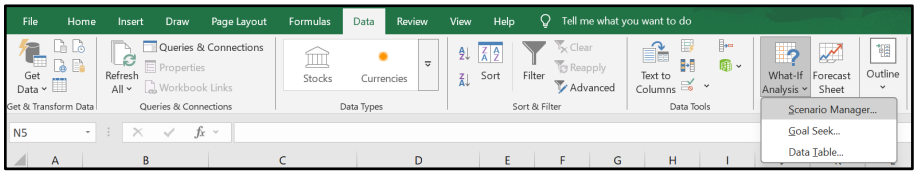


	A	B	C	D	E
1	**Book Store**				
2					
3		Total number of books	% sold for the highest price		
4		100	60%		
5					
6			Number of books	Unit profit	
7		Highest price	60	$50	
8		Lower price	40	$20	
9					
10			Total profit	$3,800	
11					

1. On the Data tab, in the Forecast group, click What-If Analysis.

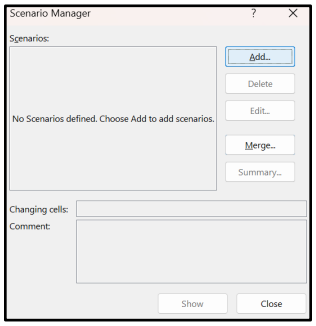


2. Click on Scenario Manager

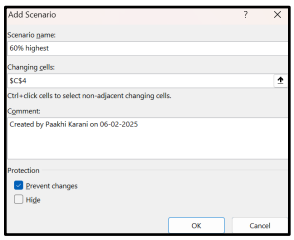


The Scenario Manager dialog box appears.

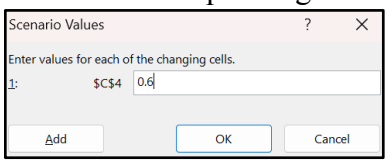
3. Add a scenario by clicking on Add.



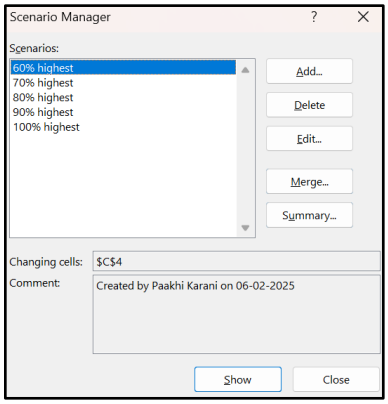
4. Type a name (60% highest), select cell C4 (% sold for the highest price) for the Changing cells and click on OK.



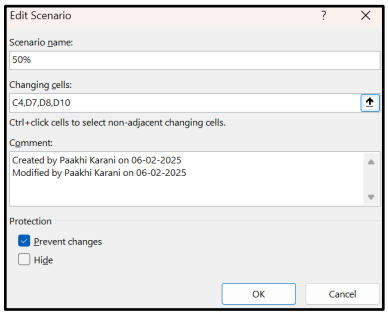
5. Enter the corresponding value 0.6 and click on OK again.

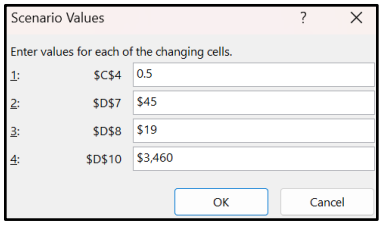


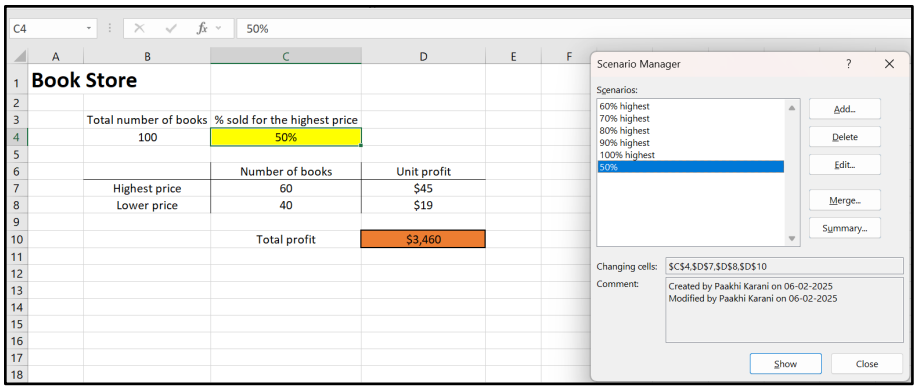
6. Next, add 4 other scenarios (70%, 80%, 90% and 100%).
   Finally, your Scenario Manager should be consistent with the picture below:



OR






C4
50%
Book Store
Total number of books
% sold for the highest price
100
50%
Number of books
Unit profit
Highest price
60
$45
Lower price
40
$19
Total profit
$3,460
Scenario Manager
Scenarios:
60% highest
70% highest
80% highest
90% highest
100% highest
50%
Add...
Delete
Edit...
Merge...
Summary...
Changing cells:
C4,D7,D8,D10
Comment:
Created by Paakhi Karani on 06-02-2025
Modified by Paakhi Karani on 06-02-2025
Show
Close

Practical 3

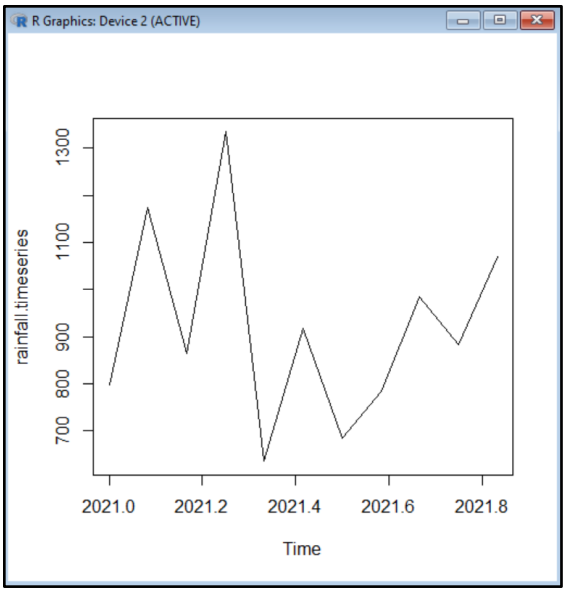
Perform the data classification using classification algorithm using R/Python.

Code:

```
> rainfall <- c(799,1174.8,865.1,1334.6,635.4,918.5,685.5,784.2,985,882.8,1071)
> rainfall.timeseries <- ts(rainfall,start = c(2021,1),frequency = 12)
> print(rainfall.timeseries)
      Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov
2021 799.0 1174.8 865.1 1334.6 635.4 918.5 685.5 784.2 985.0 882.8 1071.0
> png(file = "rainfall.png")
> plot(rainfall.timeseries)
> dev.off()
null device
          1
> plot(rainfall.timeseries)
```

Output:

```
> rainfall <- c(799,1174.8,865.1,1334.6,635.4,918.5,685.5,784.2,985,882.8,1071)
> rainfall.timeseries <- ts(rainfall,start = c(2021,1),frequency = 12)
> print(rainfall.timeseries)
        Jan    Feb    Mar    Apr    May    Jun    Jul    Aug    Sep    Oct    Nov
2021  799.0 1174.8  865.1 1334.6  635.4  918.5  685.5  784.2  985.0  882.8 1071.0
>
> png(file = "rainfall.png")
> plot(rainfall.timeseries)
> dev.off()
null device
          1
> plot(rainfall.timeseries)
>
```



Practical 4

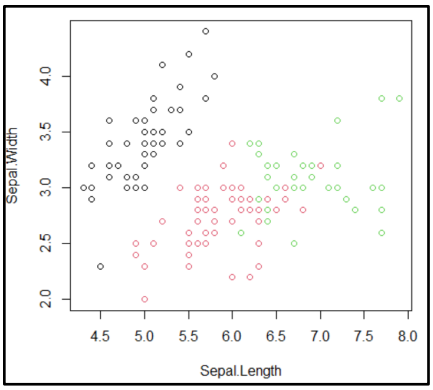
Perform the data clustering using clustering algorithm using R/Python.

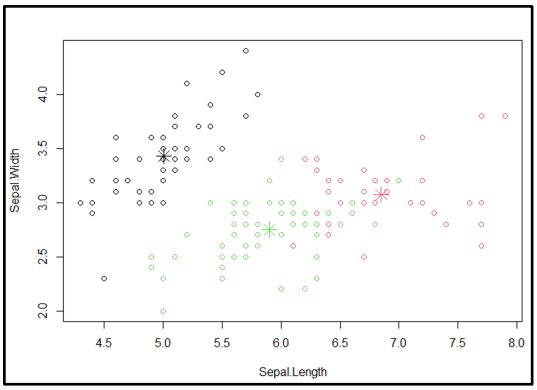
Code:

```
> newiris <- iris
> newiris$Species <- NULL
> (kc <- kmeans(newiris,3))
> table(iris$Species,kc$cluster)
> plot(newiris[c("Sepal.Length","Sepal.Width")],col=kc$cluster)
> points(kc$centers[,c("Sepal.Length","Sepal.Width")],col=1:3,pch=8,cex=2)
```

Output:

```
> newiris <- iris
> newiris$Species <- NULL
> (kc <- kmeans(newiris,3))
K-means clustering with 3 clusters of sizes 50, 62, 38

Cluster means:
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     5.006000    3.428000     1.462000    0.246000
2     5.901613    2.748387     4.393548    1.433871
3     6.850000    3.073684     5.742105    2.071053

Clustering vector:
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [76] 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 3 3 3 3 2 3 3 3 3 3 2 2 3 3 3 3 2 3 2 3 2 3 3 2 2 3 3 3 3 2 3 3 3 3 2 3 3 3 2 3 3 3 2

Within cluster sum of squares by cluster:
[1] 15.15100 39.82097 23.87947
 (between_SS / total_SS =  88.4 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"         "iter"         "ifault"
>
> table(iris$Species,kc$cluster)

              1  2  3
  setosa     50  0  0
  versicolor  0 48  2
  virginica   0 14 36
>
> plot(newiris[c("Sepal.Length","Sepal.Width")],col=kc$cluster)
>
> points(kc$centers[,c("Sepal.Length","Sepal.Width")],col=1:3,pch=8,cex=2)
```





Practical 5

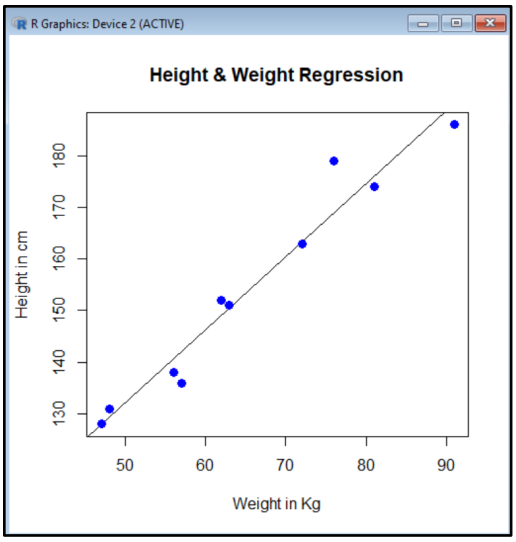
Perform the Linear regression on the given data warehouse data using R/Python.

Code:

```
> x <- c(151,174,138,186,128,136,179,163,152,131)
> y <- c(63,81,56,91,47,57,76,72,62,48)
> relation <- lm(y~x)
> png(file = "linearregression.png")
> plot(y,x,col = "blue",main = "Height & Weight Regression",abline(lm(x~y)),cex=1.3, pch=16, xlab="Weight in Kg", ylab="Height in cm")
> dev.off()
```

Output:

```
> x <- c(151,174,138,186,128,136,179,163,152,131)
> y <- c(63,81,56,91,47,57,76,72,62,48)
> relation <- lm(y~x)
> png(file = "linearregression.png")
> plot(y,x,col = "blue",main = "Height & Weight Regression",abline(lm(x~y)),cex =1.3,pch = 16,xlab = "Weight in Kg",ylab = "Height in cm")
> dev.off()
null device
          1
> plot(y,x,col = "blue",main = "Height & Weight Regression",abline(lm(x~y)),cex =1.3,pch = 16,xlab = "Weight in Kg",ylab = "Height in cm")
>
```



Practical 6

Perform the logistic regression on the given data warehouse data using R/Python.

Code:

```
> quality<-read.csv("C:/quality.csv")
> str(quality)
> table(quality$PoorCare)
> install.packages("caTools")
> library(caTools)
> qualityTrain = subset(quality,split==TRUE)
> qualityTest  =subset(quality,split==FALSE)
> nrow(qualityTrain)
> nrow(qualityTest)
> QualityLog=glm(PoorCare~OfficeVisits + Narcotics,data=qualityTrain,family = binomial) >
summary(QualityLog)
> predictTrain=predict(QualityLog,type = "response")
> summary(predictTrain)
```

```
> tapply(predictTrain, qualityTrain$PoorCare, mean)
> table(qualityTrain$PoorCare,predictTrain>0.5)
10/25
70/74
> table(qualityTrain$PoorCare,predictTrain>0.7)
8/25
73/74
> table(qualityTrain$PoorCare,predictTrain<0.2)
16/25
54/74

> install.packages("ROCR")
> library(ROCR)
> ROCRpred=prediction(predictTrain,qualityTrain$PoorCare)
> ROCRperf=performance(ROCRpred,"tpr","fpr")
> plot(ROCRperf)
> plot(ROCRperf,colorize=TRUE)
> plot(ROCRperf,colorize=TRUE,print.cutoffs.at=seq(0,1,by=0.1),text.adj=c(-0.2,10.7))
```

Practical 7

Write a Python program to read data from a CSV file, perform simple data analysis, and generate basic insights. (Use Pandas is a Python library).

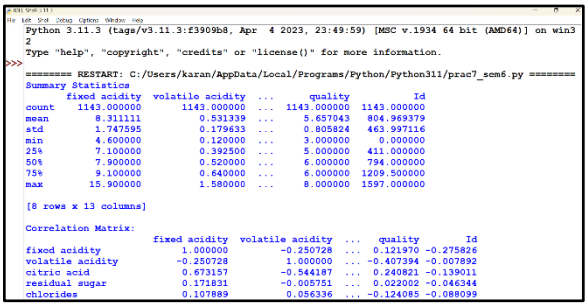
Code:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

#Load the CSV file into a DataFrame
file_path = 'C:\WineQT.csv'
df = pd.read_csv(file_path)
#df = pd.read.csv(file_path)
```

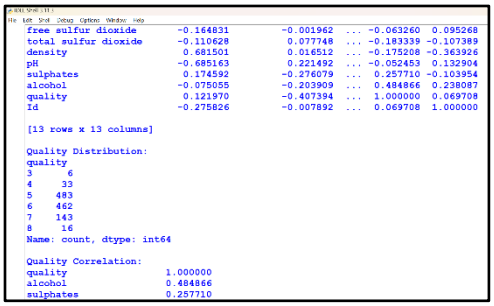
```
#Summary statistics
summary_stats = df.describe()
print("Summary Statistics")
print(summary_stats)

#Correlation matrix
correlation_matrix = df.corr()
print("\nCorrelation Matrix: ")
print(correlation_matrix)

#Quality distribution
quality_distribution = df['quality'].value_counts().sort_index()
print("\nQuality Distribution: ")
print(quality_distribution)

#Correlation with quality
quality_correlation = correlation_matrix['quality'].sort_values(ascending=False)
print("\nQuality Correlation: ")
print(quality_correlation)

# Correlation with quality
quality_correlation = correlation_matrix ['quality'].sort_values (ascending=False)
print("\nCorrelation with Quality:")
print (quality_correlation)

#Plotting
plt.figure (figsize=(10, 6))

#Quality distribution plot
plt.subplot (2, 2, 1)
sns.countplot (x='quality', data=df, palette='viridis')
plt.title('Quality Distribution')

#Heatmap of correlation matrix
plt.subplot(2, 2, 2)
sns.heatmap (correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
plt.title('Correlation Matrix Heatmap')

#Alcohol vs Quality
plt.subplot (2, 2, 3)
```

```
sns.boxplot (x='quality', y='alcohol', data=df, palette='viridis')
plt.title('Alcohol vs Quality')

#Density vs Quality
plt.subplot (2, 2, 4)
sns.boxplot (x='quality', y='density', data=df, palette='viridis')
plt.title('Density vs Quality')
plt.tight_layout ()
plt.show()
```

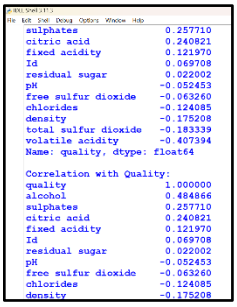
Output:



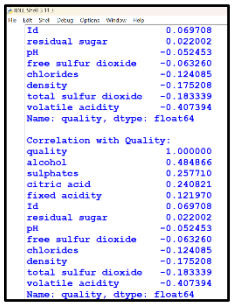
```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win3
2
Type "help", "copyright", "credits" or "license()" for more information.
>>>
======== RESTART: C:/Users/karan/AppData/Local/Programs/Python/Python311/prac7_sem6.py ========
Summary Statistics
       fixed acidity  volatile acidity  ...      quality           Id
count    1143.000000       1143.000000  ...  1143.000000  1143.000000
mean        8.311111          0.531339  ...     5.657043   804.969379
std         1.747595          0.179633  ...     0.805824   463.997116
min         4.600000          0.120000  ...     3.000000     0.000000
25%         7.100000          0.392500  ...     5.000000   411.000000
50%         7.900000          0.520000  ...     6.000000   794.000000
75%         9.100000          0.640000  ...     6.000000  1209.500000
max        15.900000          1.580000  ...     8.000000  1597.000000

[8 rows x 13 columns]

Correlation Matrix:
                      fixed acidity  volatile acidity  ...   quality        Id
fixed acidity              1.000000         -0.250728  ...  0.121970 -0.275826
volatile acidity          -0.250728          1.000000  ... -0.407394 -0.007892
citric acid                0.673157         -0.544187  ...  0.240821 -0.139011
residual sugar             0.171831         -0.005751  ...  0.022002 -0.046344
chlorides                  0.107889          0.056336  ... -0.124085 -0.088099
```



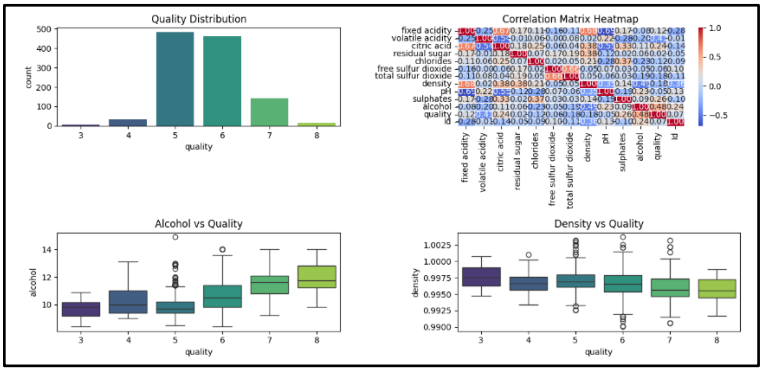
```
free sulfur dioxide      -0.164831      -0.001962  ... -0.063260  0.095268
total sulfur dioxide     -0.110628       0.077748  ... -0.183339 -0.107389
density                   0.681501       0.016512  ... -0.175208 -0.363926
pH                       -0.685163       0.221492  ... -0.052453  0.132904
sulphates                 0.174592      -0.276079  ...  0.257710 -0.103954
alcohol                  -0.075055      -0.203909  ...  0.484866  0.238087
quality                   0.121970      -0.407394  ...  1.000000  0.069708
Id                       -0.275826      -0.007892  ...  0.069708  1.000000

[13 rows x 13 columns]

Quality Distribution:
quality
3      6
4     33
5    483
6    462
7    143
8     16
Name: count, dtype: int64

Quality Correlation:
quality                1.000000
alcohol                0.484866
sulphates              0.257710
```



```
sulphates              0.257710
citric acid            0.240821
fixed acidity          0.121970
Id                     0.069708
residual sugar         0.022002
pH                    -0.052453
free sulfur dioxide   -0.063260
chlorides             -0.124085
density               -0.175208
total sulfur dioxide  -0.183339
volatile acidity      -0.407394
Name: quality, dtype: float64

Correlation with Quality:
quality                1.000000
alcohol                0.484866
sulphates              0.257710
citric acid            0.240821
fixed acidity          0.121970
Id                     0.069708
residual sugar         0.022002
pH                    -0.052453
free sulfur dioxide   -0.063260
chlorides             -0.124085
density               -0.175208
```



```
Id                     0.069708
residual sugar         0.022002
pH                    -0.052453
free sulfur dioxide   -0.063260
chlorides             -0.124085
density               -0.175208
total sulfur dioxide  -0.183339
volatile acidity      -0.407394
Name: quality, dtype: float64

Correlation with Quality:
quality                1.000000
alcohol                0.484866
sulphates              0.257710
citric acid            0.240821
fixed acidity          0.121970
Id                     0.069708
residual sugar         0.022002
pH                    -0.052453
free sulfur dioxide   -0.063260
chlorides             -0.124085
density               -0.175208
total sulfur dioxide  -0.183339
volatile acidity      -0.407394
Name: quality, dtype: float64
```



Practical 8

Perform data visualization

a. Perform data visualization using Python on any sales data.

Code:

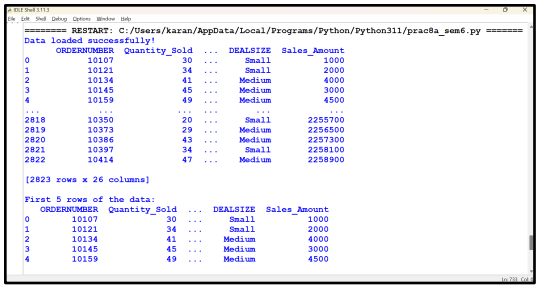
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

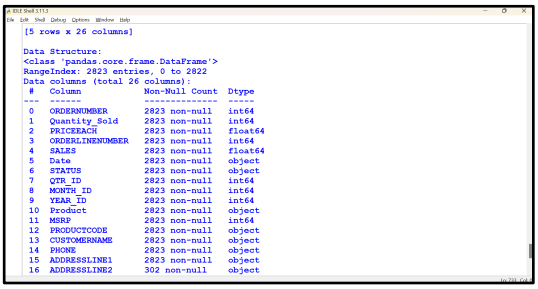
```
# Step 1: Load the CSV data into a pandas DataFrame
def load_data(csv_file):
    try:
        data = pd.read_csv('C:\sales_data.csv', encoding="ISO-8859-1")
        print("Data loaded successfully!")
        print(data)
        return data
    except Exception as e:
        print(f"Error loading data: {e}")
        return None

# Step 2: Perform basic data inspection
def data_summary(data):
    print("\nFirst 5 rows of the data:")
    print(data.head())  # Displays the first 5 rows

    print("\nData Structure:")
    print(data.info())  # Get information on the DataFrame such as column types and non-null
counts

    print("\nStatistical summary of numeric columns:")
    print(data.describe())  # Gives summary statistics for numeric columns

# Step 3: Visualize the data

# Visualization 1: Total Sales Over Time
def plot_sales_over_time(data):
    # Convert 'Date' to datetime if it's not already in the correct format
    data['Date'] = pd.to_datetime(data['Date'])

    # Aggregate data by Date and calculate the total sales for each day
    sales_by_date = data.groupby('Date')['Sales_Amount'].sum().reset_index()

    plt.figure(figsize=(10, 6))
    sns.lineplot(x='Date', y='Sales_Amount', data=sales_by_date, marker='o')
    plt.title('Total Sales Over Time')
    plt.xlabel('Date')
    plt.ylabel('Total Sales ($)')
    plt.xticks(rotation=45)
    plt.tight_layout()
```

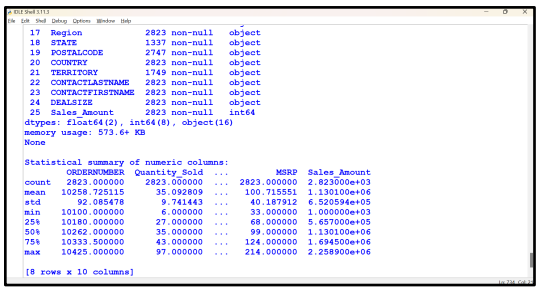
```
    plt.show()

# Visualization 2: Sales Distribution by Region
def plot_sales_by_region(data):
    plt.figure(figsize=(10, 6))
    sns.boxplot(x='Region', y='Sales_Amount', data=data)
    plt.title('Sales Distribution by Region')
    plt.xlabel('Region')
    plt.ylabel('Sales Amount ($)')
    plt.tight_layout()
    plt.show()

# Visualization 3: Sales vs Quantity Sold (Scatter Plot)
def plot_sales_vs_quantity(data):
    plt.figure(figsize=(10, 6))
    sns.scatterplot(x='Quantity_Sold', y='Sales_Amount', data=data, hue='Product',
palette='viridis')
    plt.title('Sales Amount vs Quantity Sold')
    plt.xlabel('Quantity Sold')
    plt.ylabel('Sales Amount ($)')
    plt.tight_layout()
    plt.show()

# Visualization 4: Top Products by Sales
def plot_top_products_by_sales(data):
    top_products = data.groupby('Product')['Sales_Amount'].sum().reset_index()
    top_products = top_products.sort_values('Sales_Amount', ascending=False).head(10)

    plt.figure(figsize=(10, 6))
    sns.barplot(x='Sales_Amount', y='Product', data=top_products, palette='Blues_d')
    plt.title('Top 10 Products by Sales Amount')
    plt.xlabel('Sales Amount ($)')
    plt.ylabel('Product')
    plt.tight_layout()
    plt.show()

# Main function to load data and generate visualizations
def main():
    csv_file = 'sales_data.csv'  # Replace with your actual CSV file path
    data = load_data(csv_file)
```

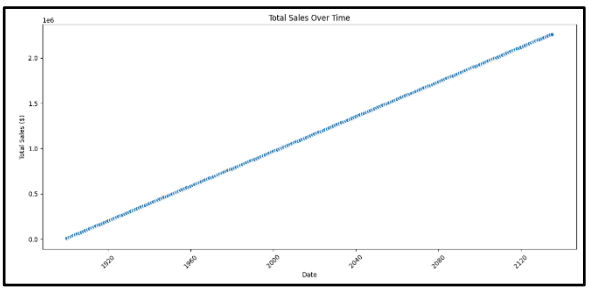
```
    if data is not None:
        # Step 2: Data summary and inspection
        data_summary(data)

        # Step 3: Data Visualizations
        plot_sales_over_time(data)
        plot_sales_by_region(data)
        plot_sales_vs_quantity(data)
        plot_top_products_by_sales(data)

# Entry point of the program
if __name__ == "__main__":
    main()
```

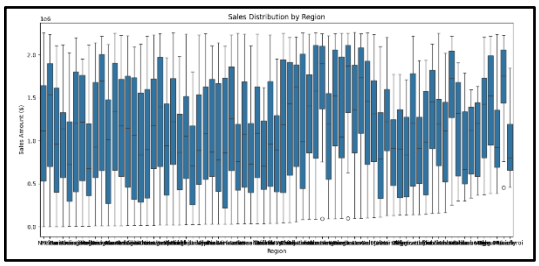
Output:

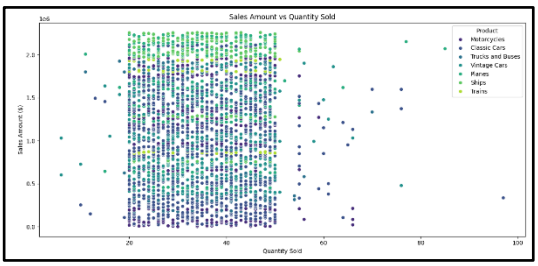


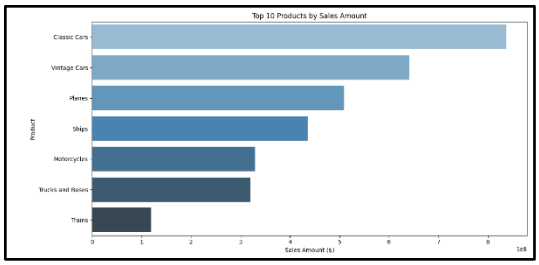












b. Perform data visualization using PowerBI on any sales data.

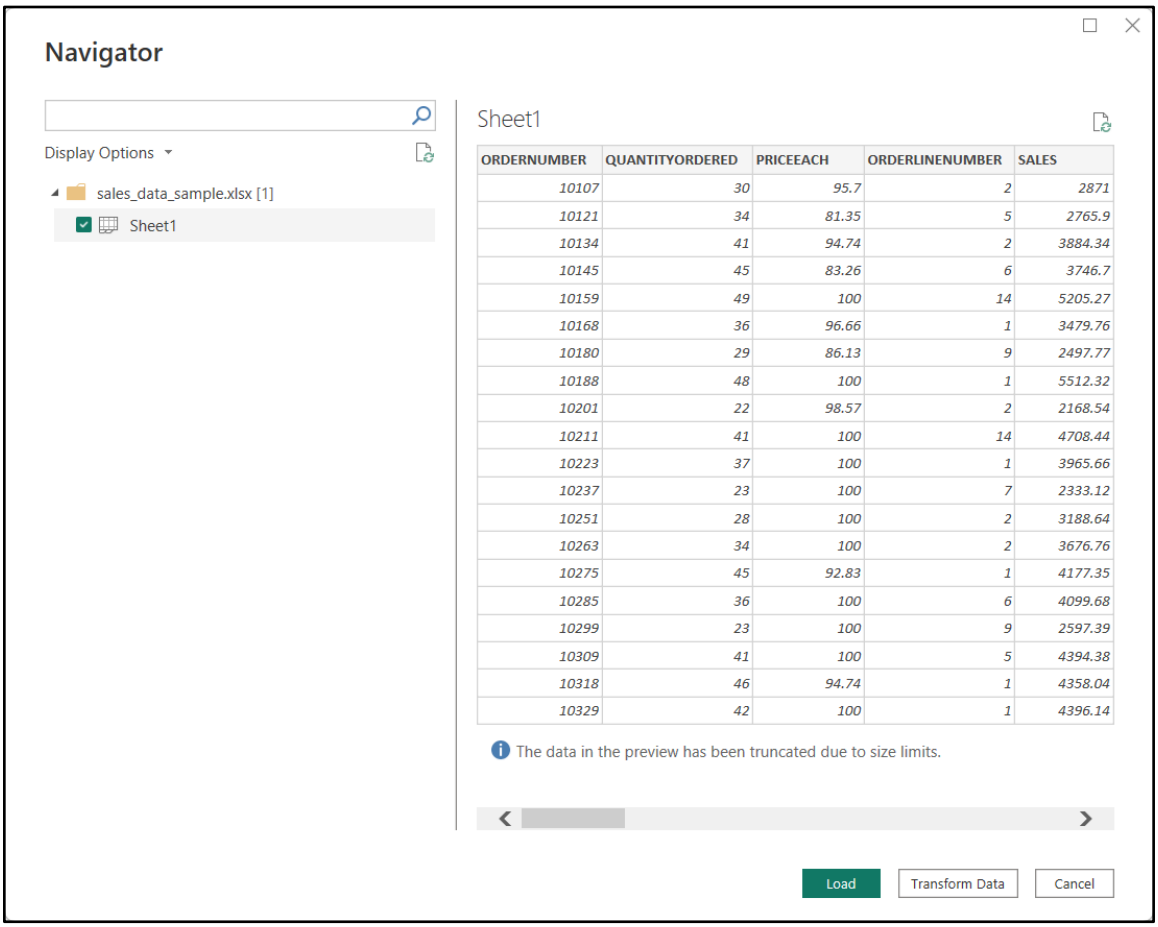
Step 1: Install and Open Power BI

1. **Download & Install Power BI Desktop**

- If you haven't installed it yet, download it from Power BI Download.
- Install and open **Power BI Desktop**.

Step 2: Load the Sales Data

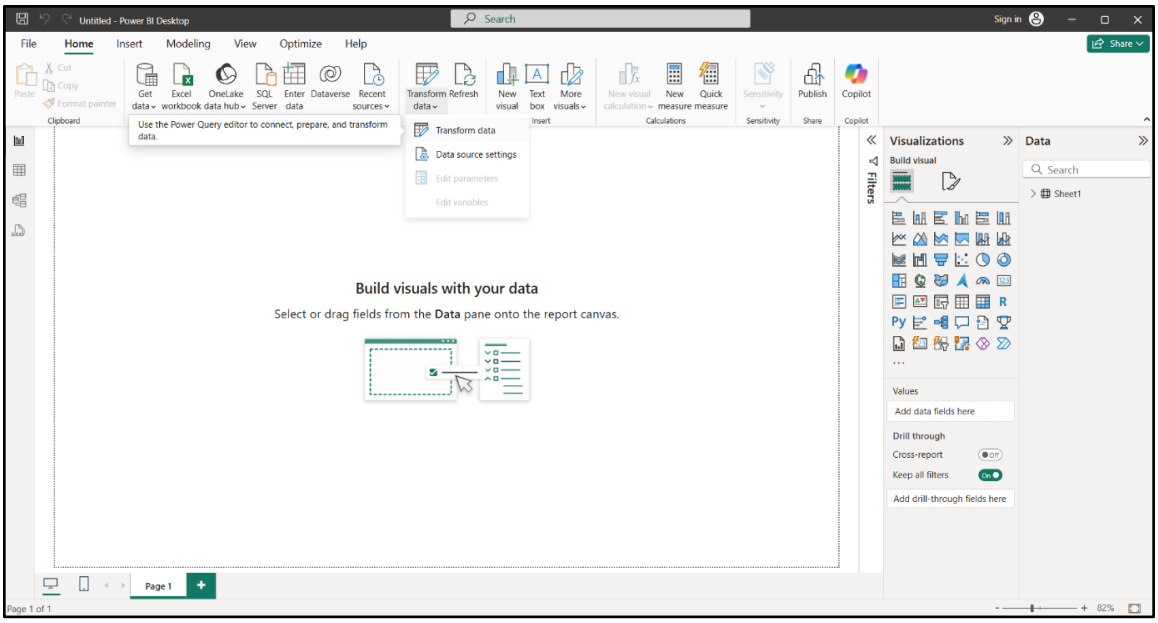
1. Open **Power BI Desktop**.
2. Click on **"Home" > "Get Data" > "Text/CSV"**.
3. Browse and select the **sales_data_sample.csv** file you uploaded.
4. Click **Open**, then click **Load** (you can check the data preview before loading).



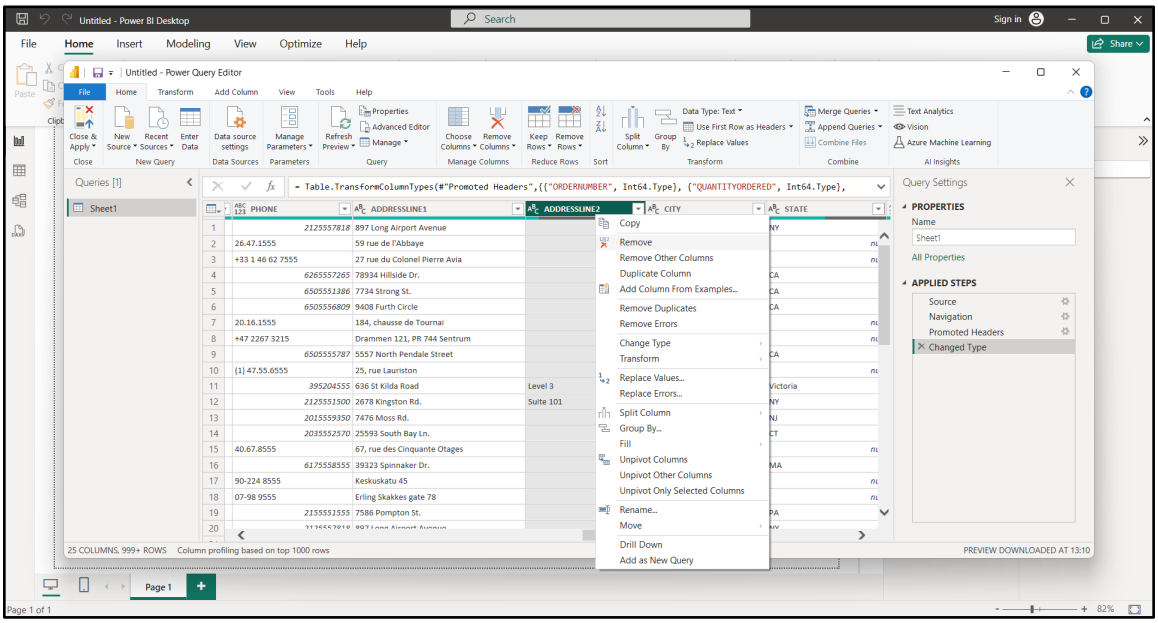
Step 3: Transform Data in Power Query Editor

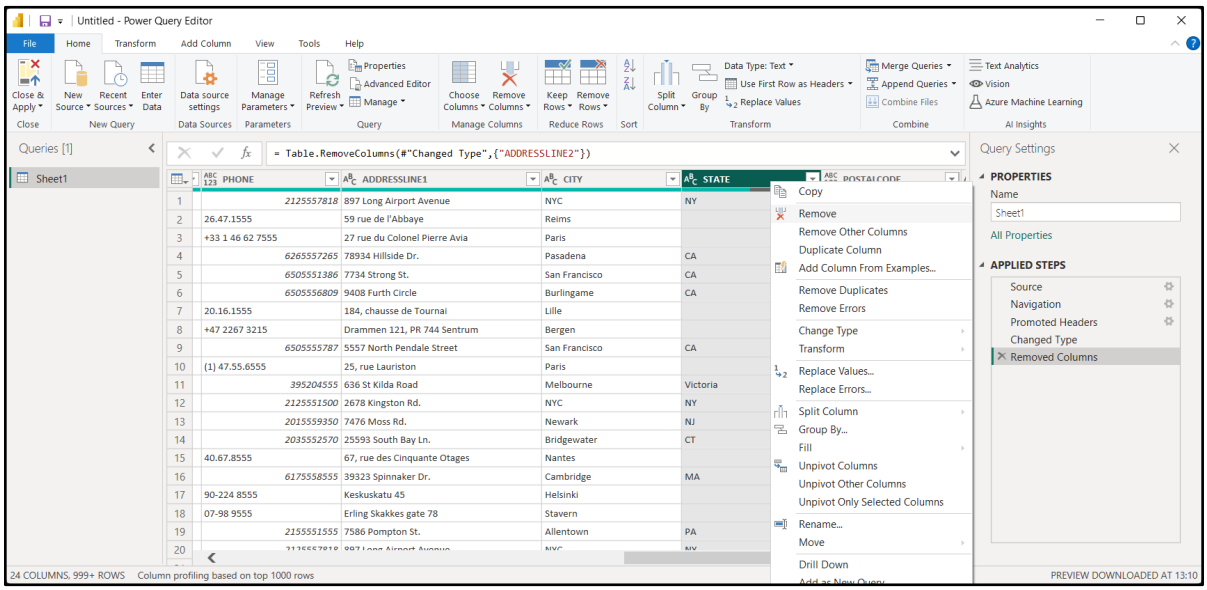
Before building reports, let's clean and format the data:

1. Click on **"Transform Data"** to open **Power Query Editor**.

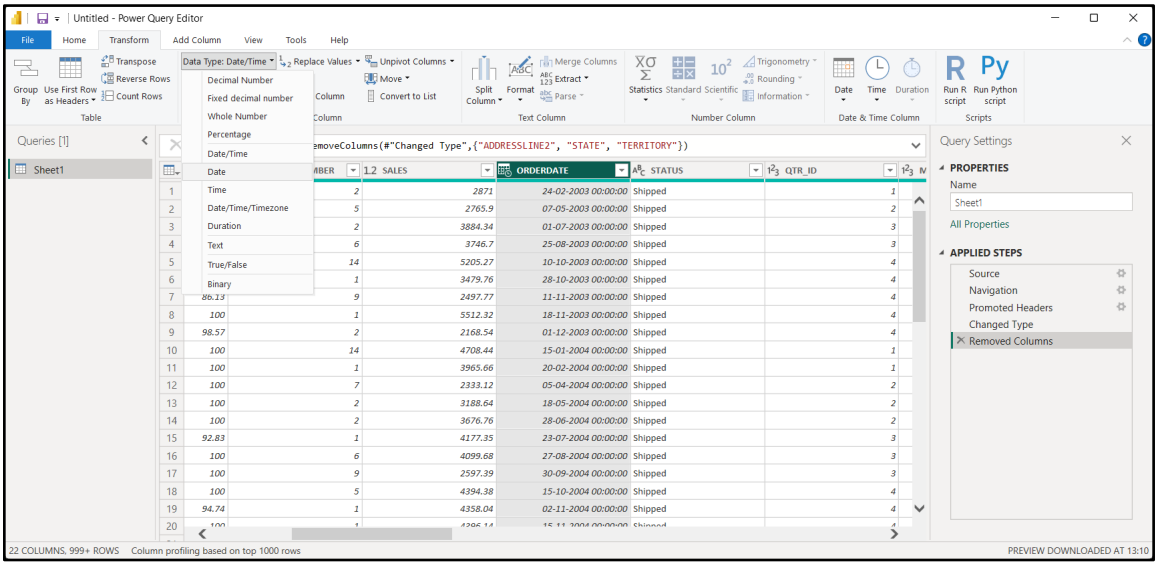


2. Review columns and check for missing values:
    - **ADDRESSLINE2, STATE, TERRITORY** have missing values, so you can remove them or fill in with "Unknown".
    - To remove a column: **Right-click** on the column header > Click **Remove**.

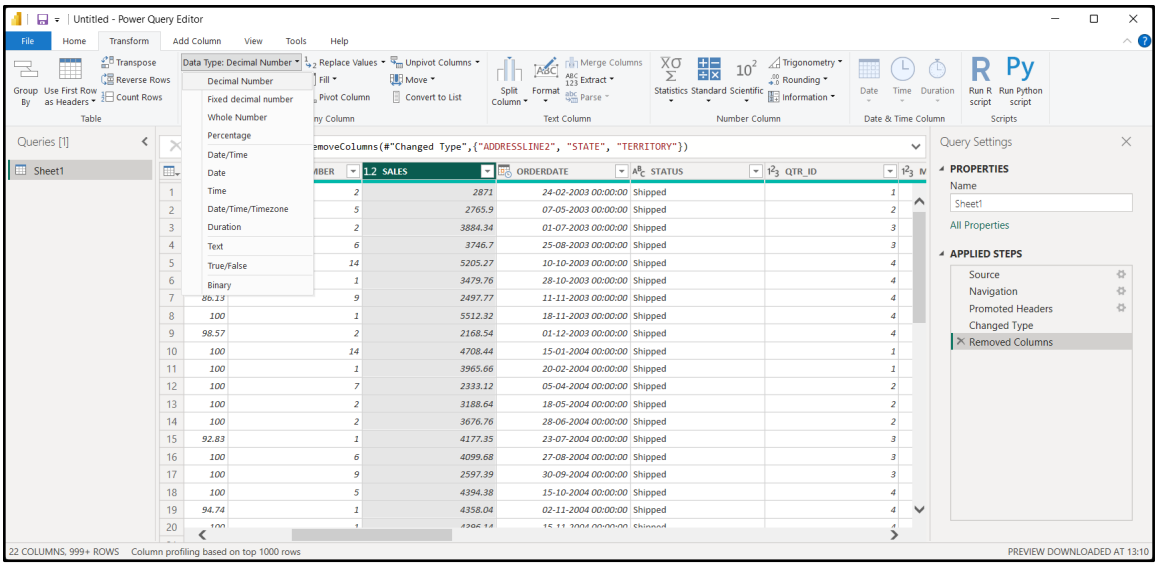




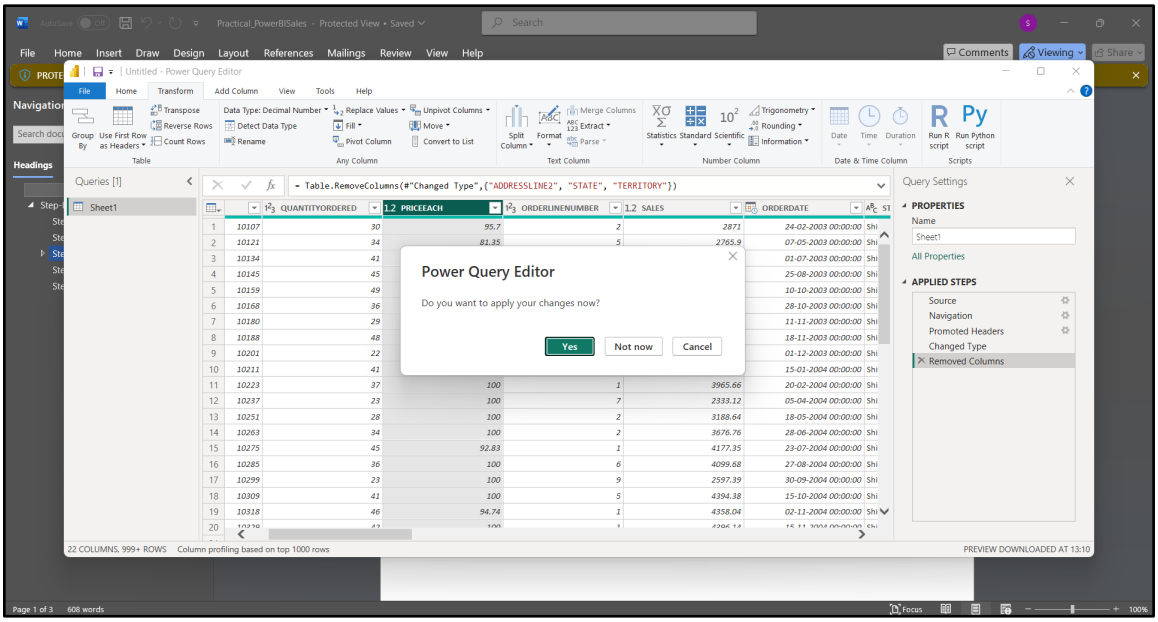
3. Convert **ORDERDATE** column to **Date** format:
    - Select **ORDERDATE** column.
    - In the top menu, click **Transform > Data Type > Date**.



4. Ensure **SALES, PRICEEACH** are in **Decimal Number** format.



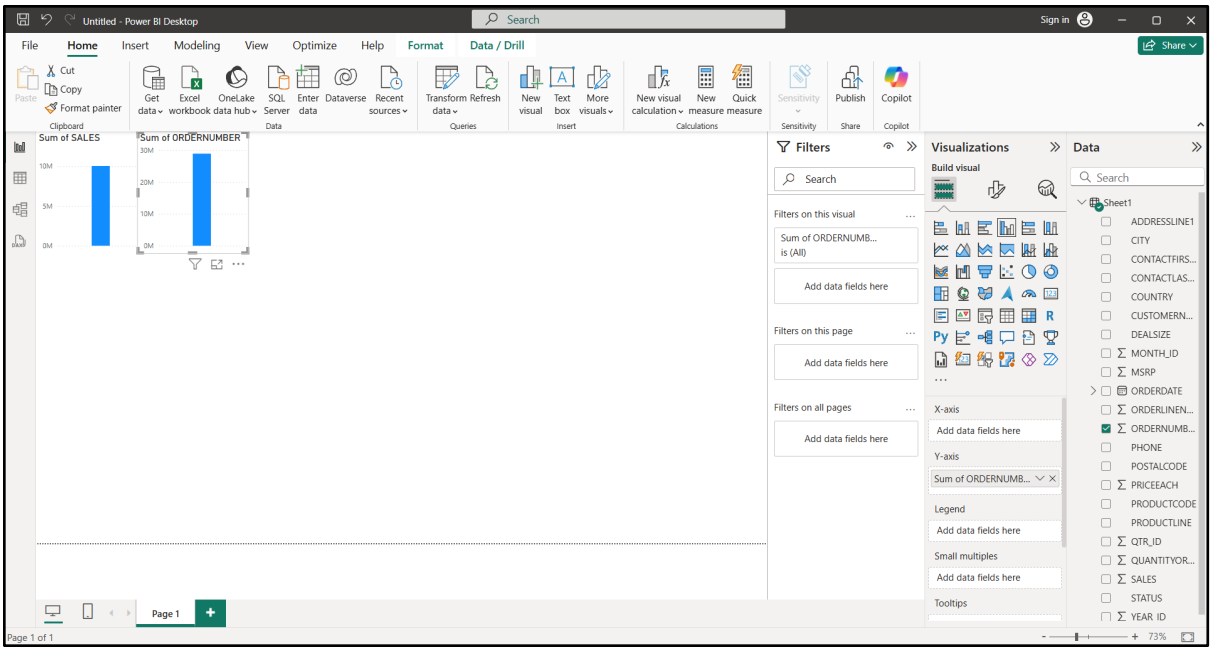
5. Click **"Close & Apply"**.



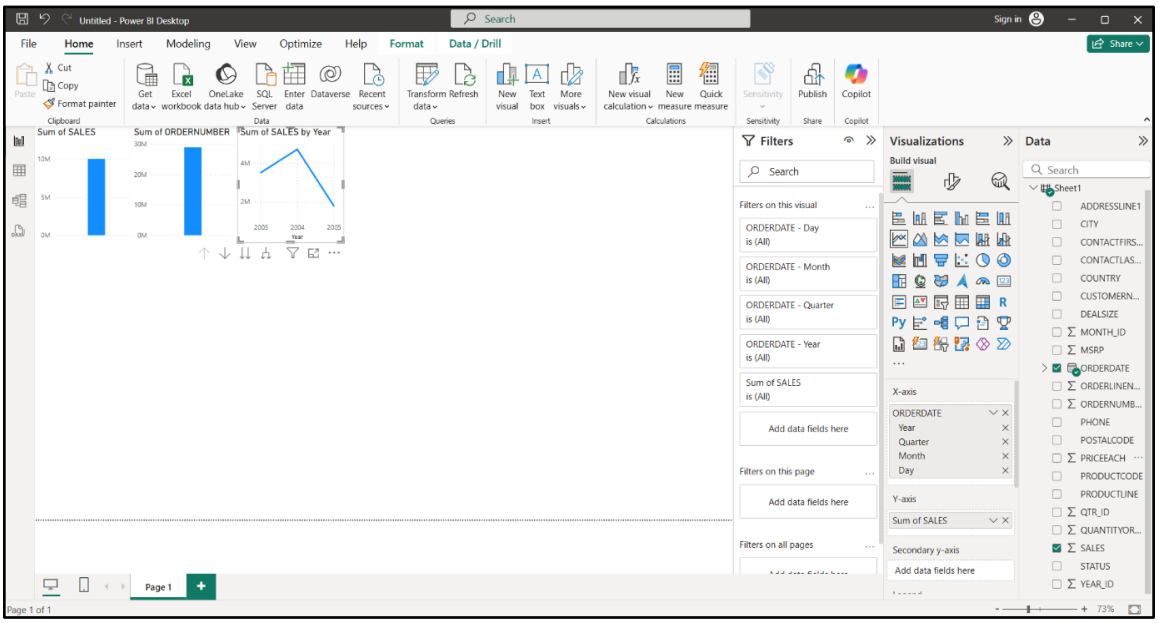
Step 4: Create Basic Visualizations

1. Sales Performance Overview

- **Total Sales & Orders**
    1. Go to **Report View** (bottom left panel).
    2. Drag **SALES** to a **Card Visual** (this shows total revenue).
    3. Drag **ORDERNUMBER** to another **Card Visual** (shows total orders).

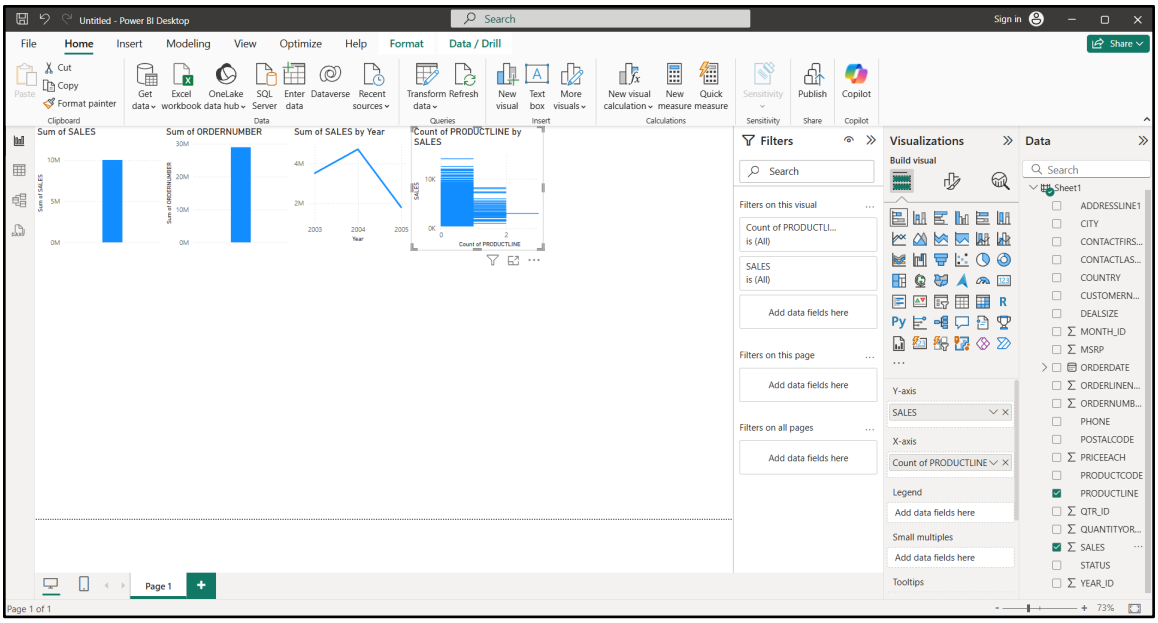
- 

- **Sales Trend Over Time**
    4. Drag **ORDERDATE** to **X-Axis** in a **Line Chart**.
    5. Drag **SALES** to **Y-Axis** (shows sales over time).



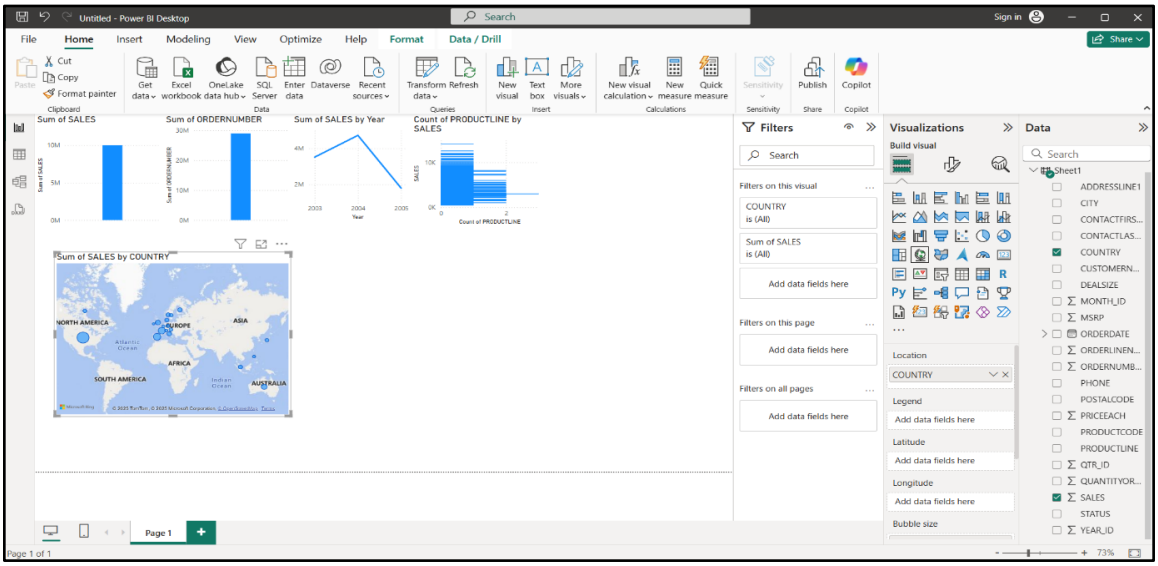
2. Sales by Product Line

1. Insert a **Bar Chart**.
2. Drag **PRODUCTLINE** to **X-Axis**.
3. Drag **SALES** to **Y-Axis** (shows revenue per product category).



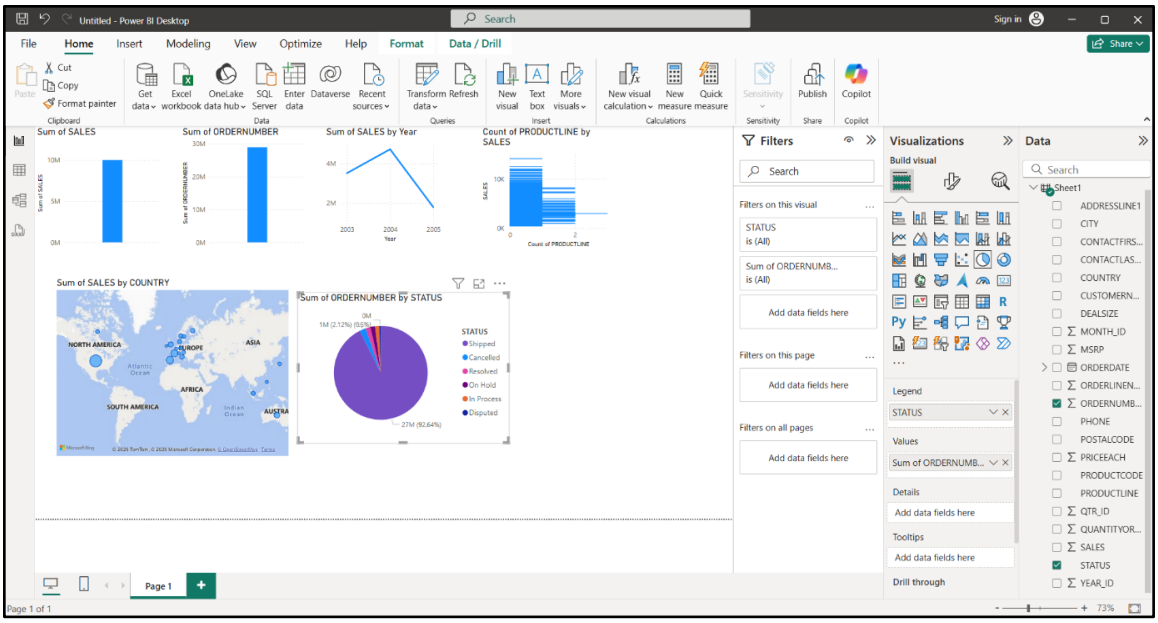
3. Sales by Country

1. Insert a **Map Visual** (Globe icon).
2. Drag **COUNTRY** to **Location** field.
3. Drag **SALES** to **Values** field (shows sales by country).



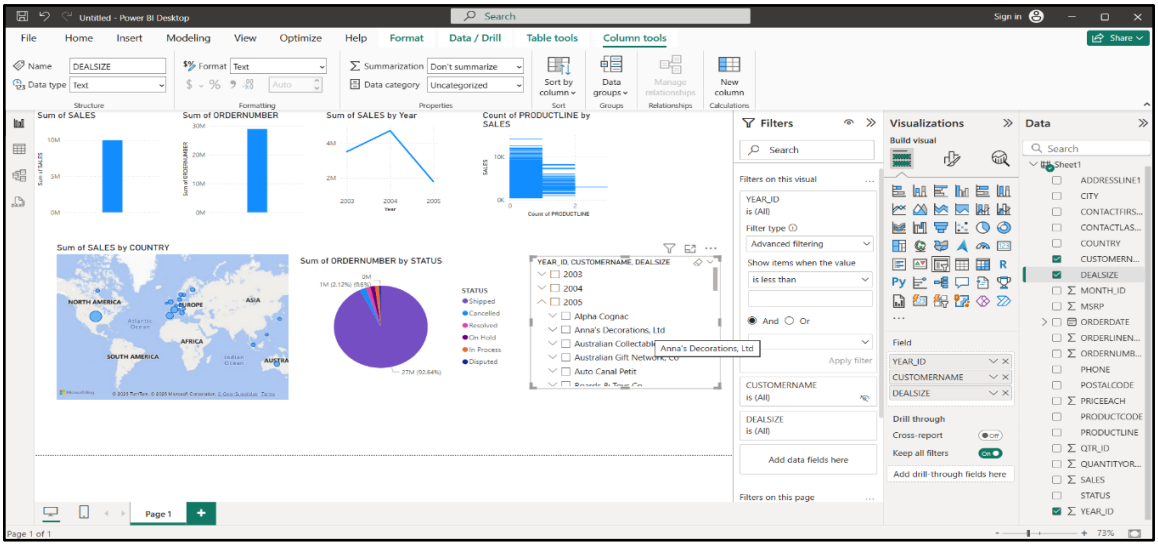
4. Order Status Breakdown

1. Insert a **Pie Chart**.
2. Drag **STATUS** to **Legend**.
3. Drag **ORDERNUMBER** to **Values** (shows percentage of orders in each status).



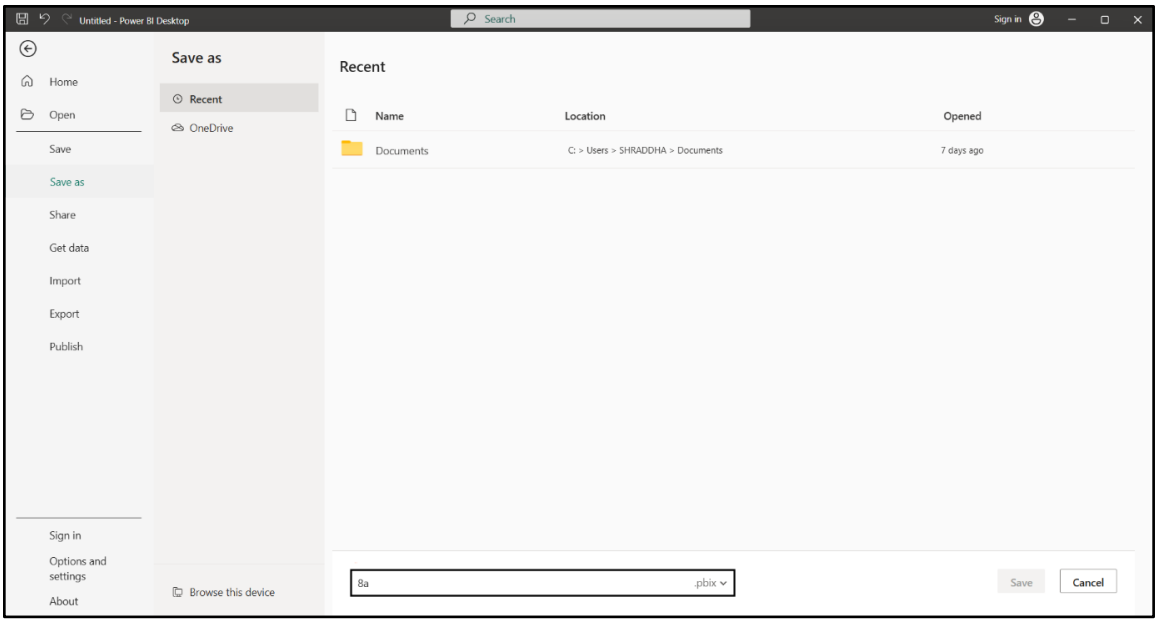
Step 5: Add Filters and Slicers

- Click on **"Slicer"** and add **YEAR_ID** to filter by year.
- Add **CUSTOMERNAME** to filter by customer.
- Add **DEALSIZE** to filter by deal size.



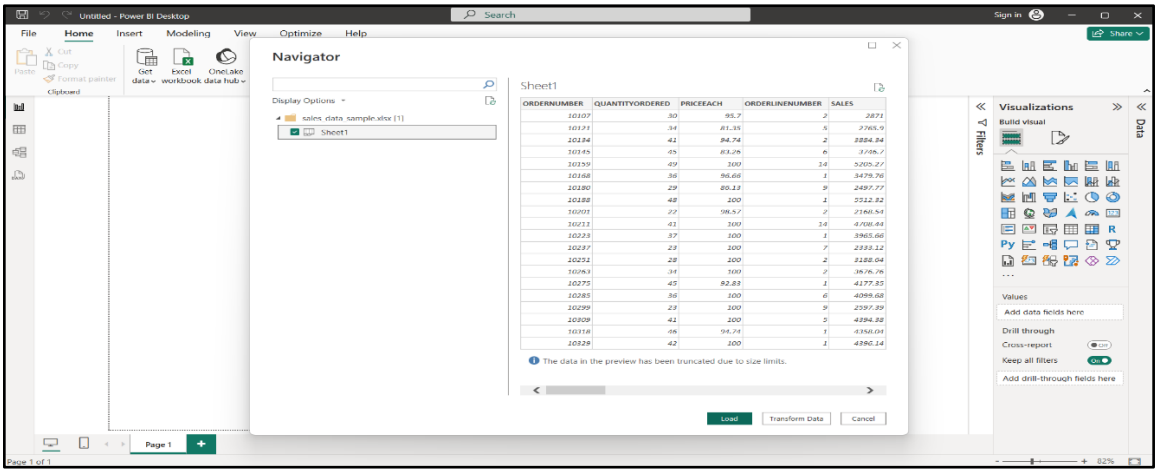
Step 6: Save & Publish Report

1. Click **File > Save As** and save the Power BI report.
2. Click **Publish** to share on **Power BI Service** (if needed).



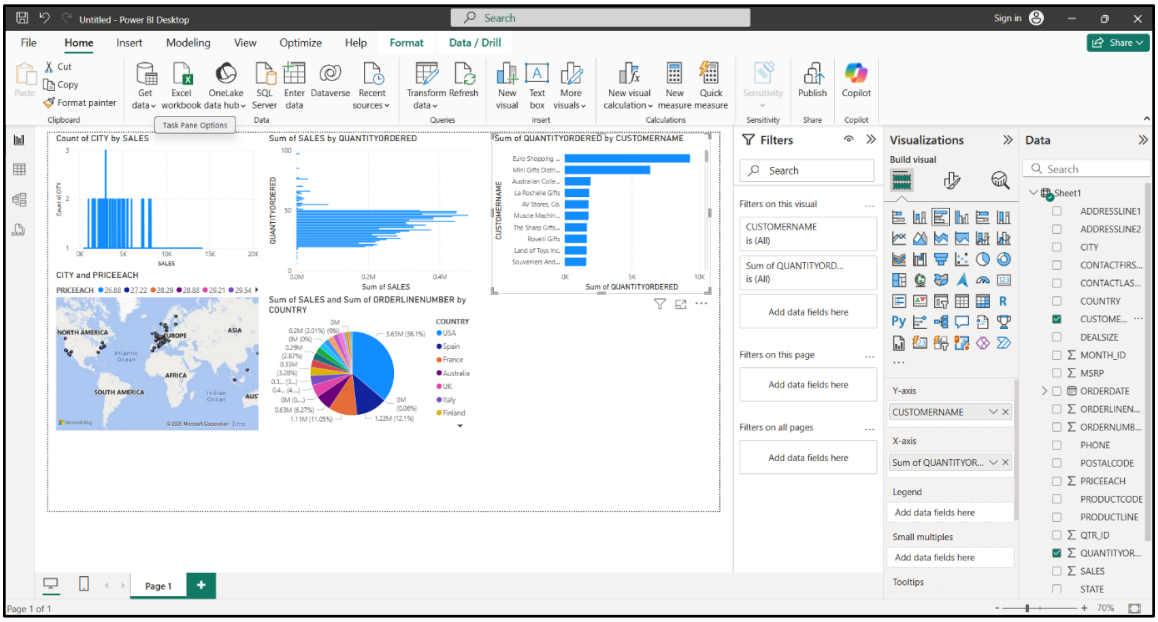
Steps to Perform Data Visualization in Power BI

1. **Import the Data:**
   - Open **Power BI Desktop**.
   - Click on **"Get Data" > "Excel"** and select the uploaded file.
   - Load the relevant sheets into Power BI.

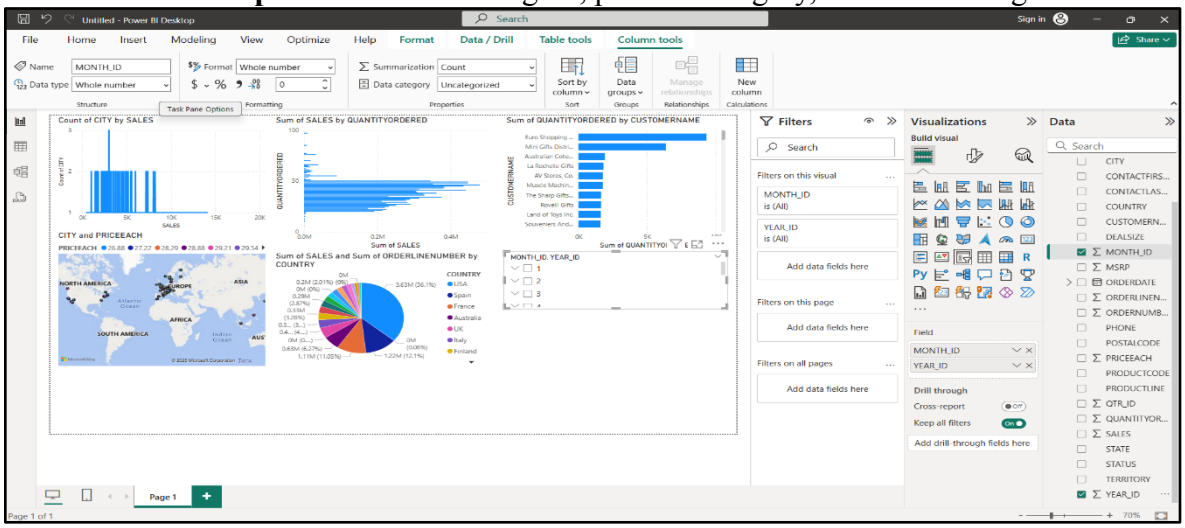


2. **Data Preparation:**
   - Check for missing values or inconsistent data.
   - Apply transformations (if needed) using **Power Query Editor**.
3. **Create Key Visuals:**
   - **Sales Trends:** Use a **Line Chart** to show sales performance over time.
   - **Regional Sales Analysis:** Use a **Map Visual** to display sales by region.
   - **Top Products Sold:** Use a **Bar Chart** to highlight best-selling products.

- **Revenue Breakdown:** Use a **Pie Chart** or **Treemap** to show sales by category.
- **Customer Segmentation:** Use **Clustered Bar Charts** to group customers based on sales.

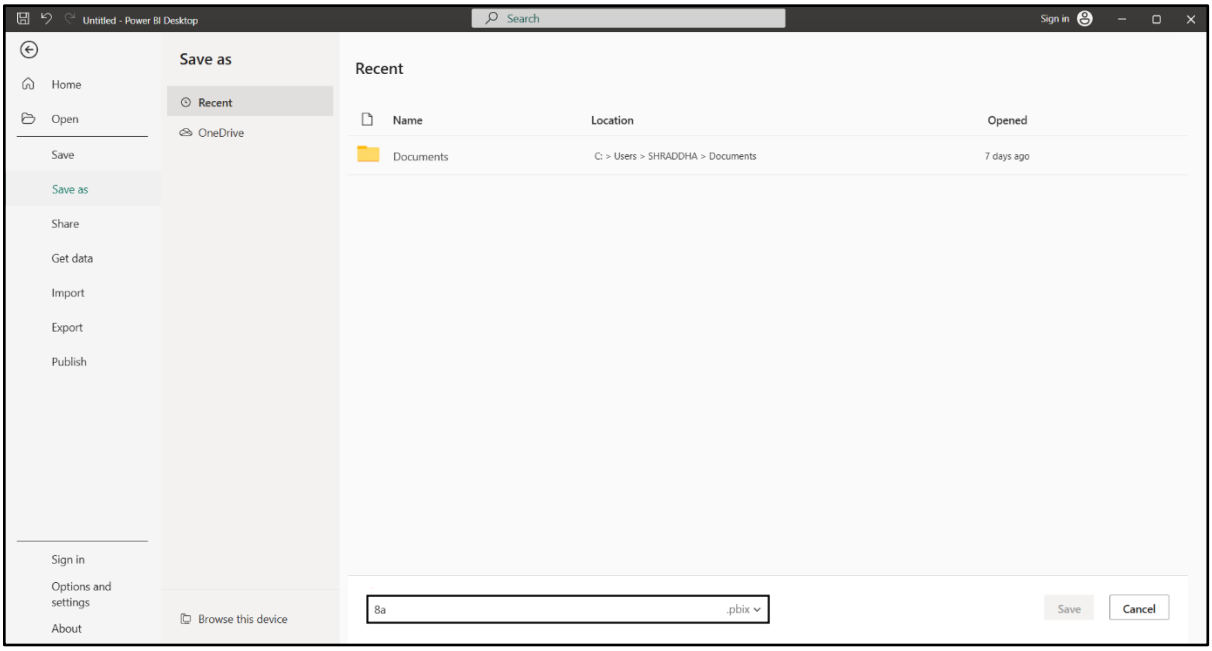


4. **Add Filters and Slicers:**
   - Include **Date Slicers** to filter data by month, quarter, or year.
   - Use **Dropdown Filters** for region, product category, or customer segment.



5. **Enhance with DAX Measures:**
   - Create calculated measures such as:
   - `Total Sales = SUM(Sales[Revenue])`
   - `Sales Growth = ( [Total Sales] - PREVIOUSMONTH([Total Sales]) ) / PREVIOUSMONTH([Total Sales])`
   - Use **KPIs** for profit margin and sales targets.

6. **Publish and Share:**
    - Save the report and publish it to **Power BI Service**.
    - Share with stakeholders via dashboards.

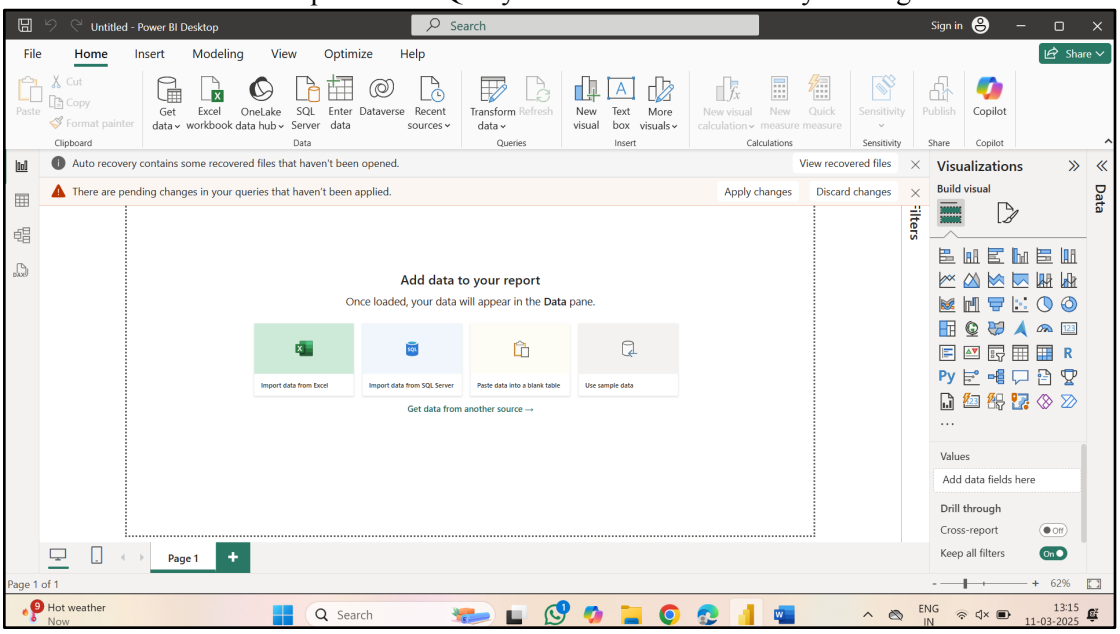


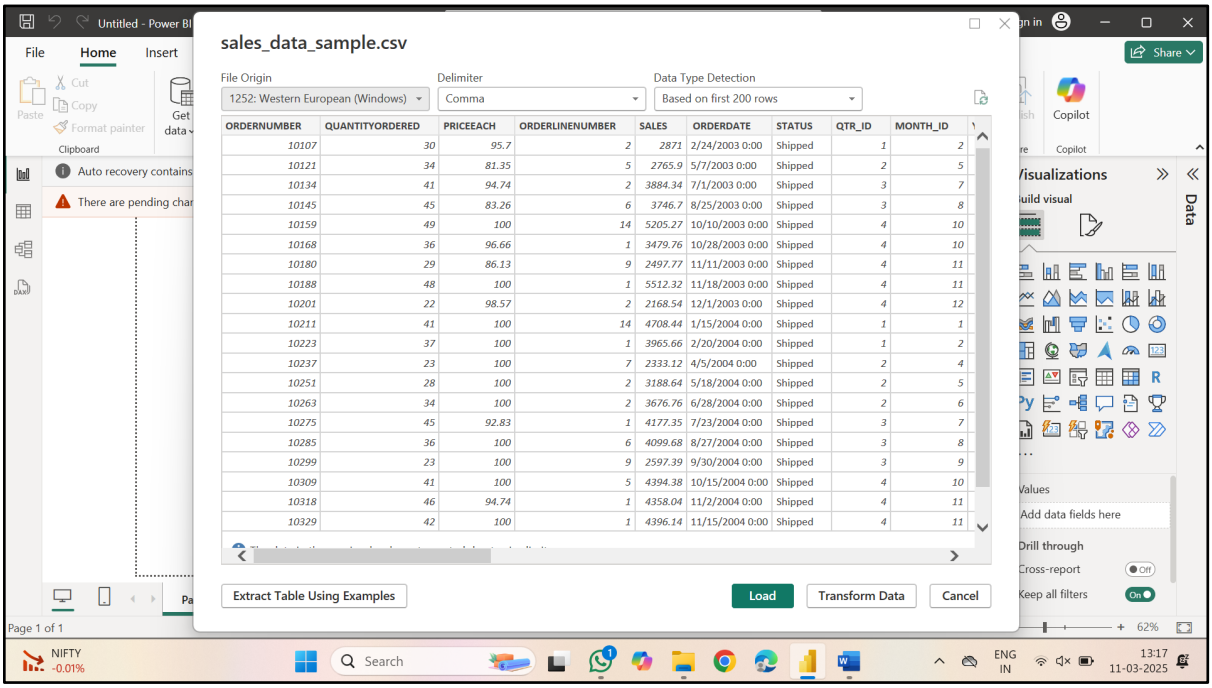
Practical 9

Create the Data staging area for the selected database using SQL.

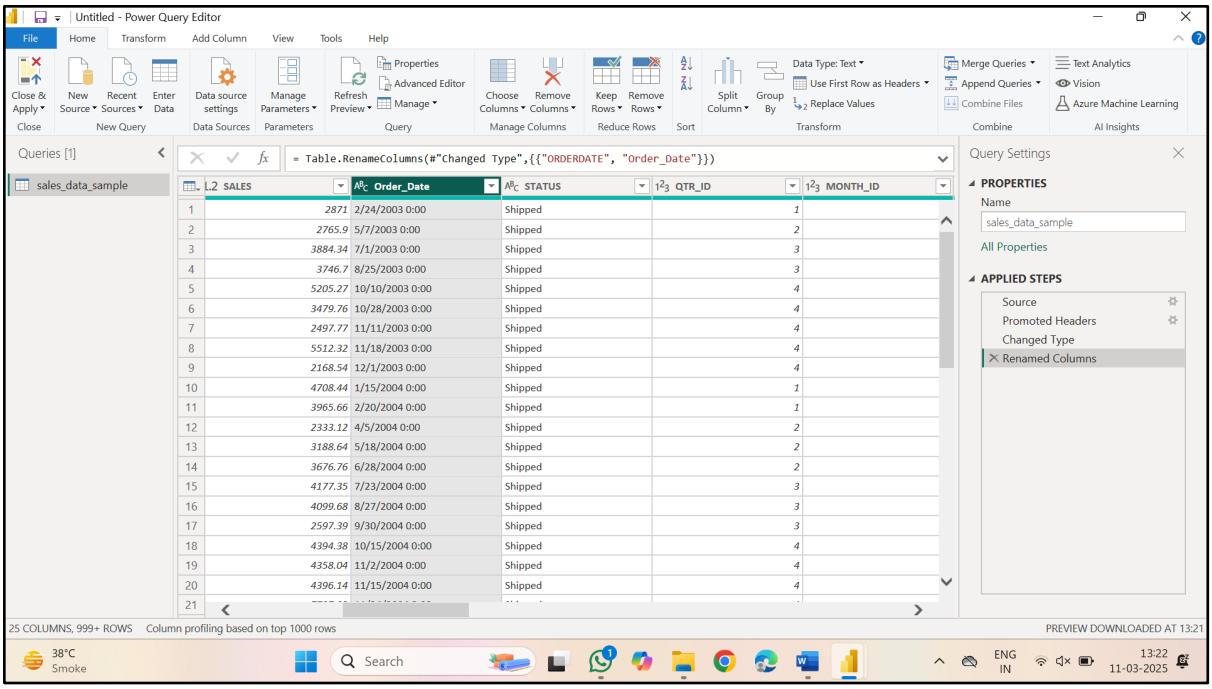
1. Load Data into Power BI

Open Power BI Desktop.
Click Home > Get Data > Excel Workbook.
Browse and select sales_data_sample.xlsx.
Click Transform Data to open Power Query Editor instead of directly loading the data.



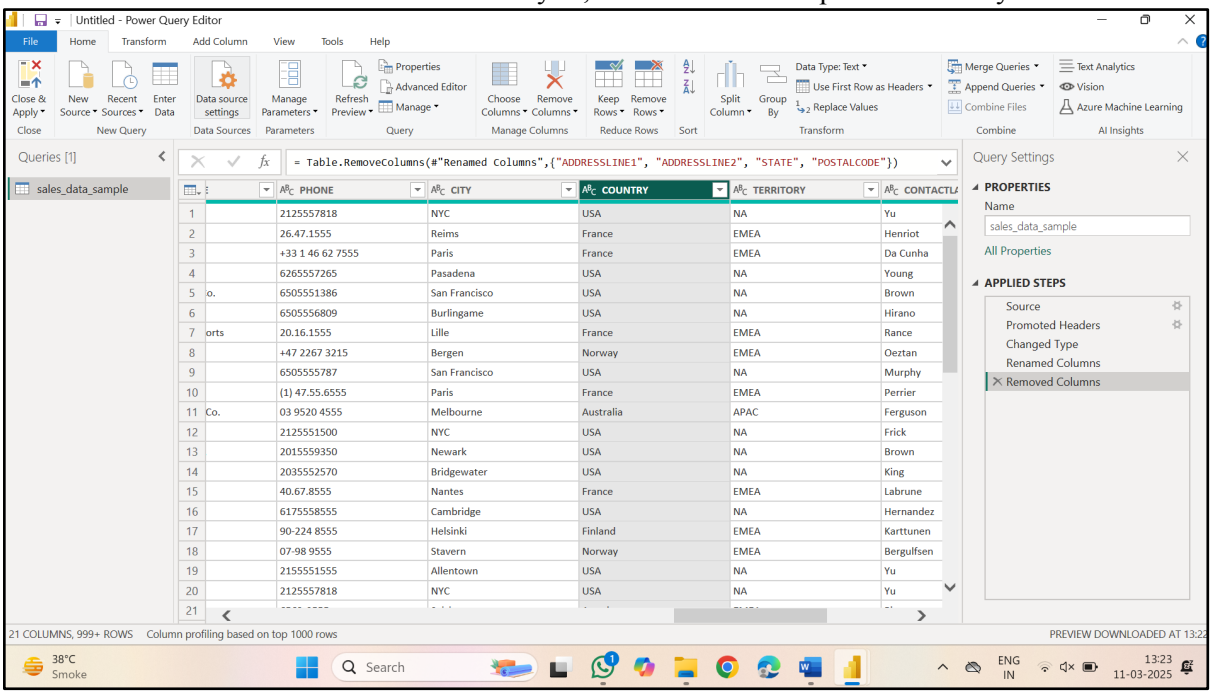


2. Perform Data Transformations in Power Query
A. Inspect and Rename Columns
Check all column names and rename them to follow a clear naming convention (e.g., Order_Date instead of order date).



B. Remove Unnecessary Columns

If there are extra columns not needed for analysis, remove them to improve efficiency.

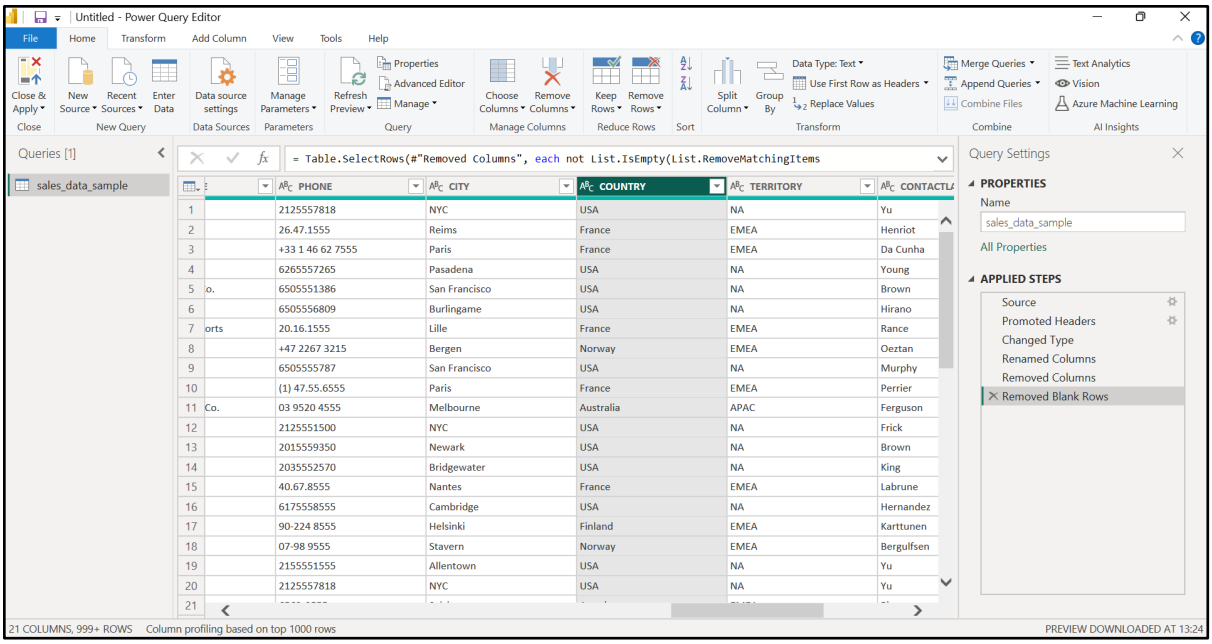


C. Handle Missing Data

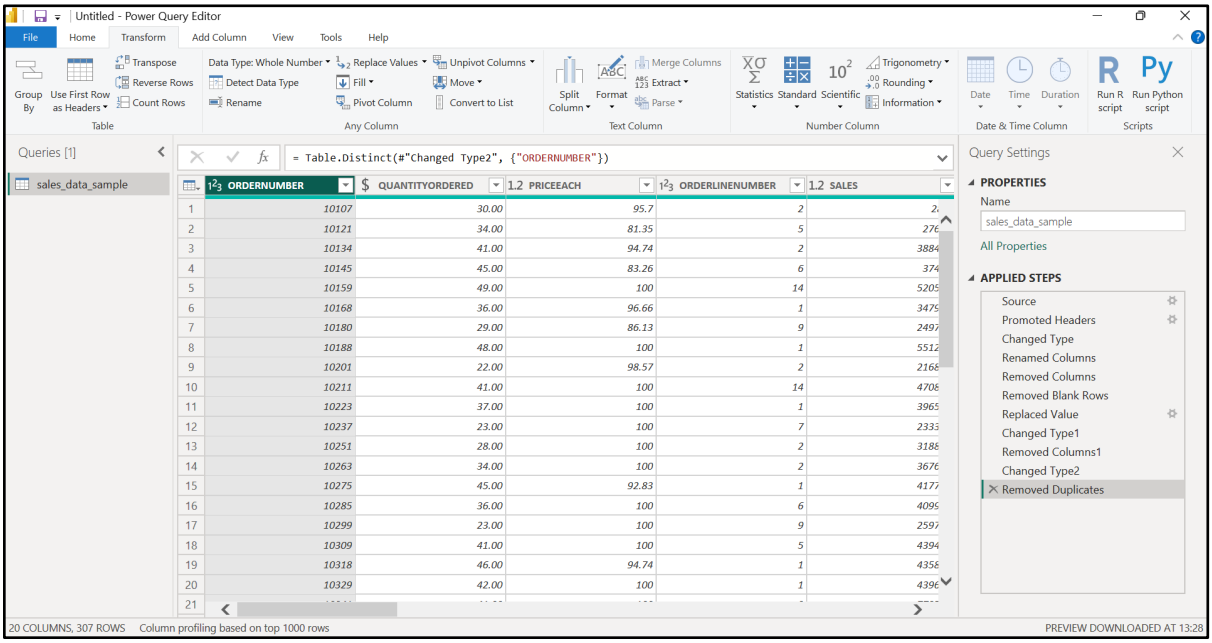
Check for null values in key fields such as Order_ID, Customer_Name, Product_Code, etc.

Apply one of the following:

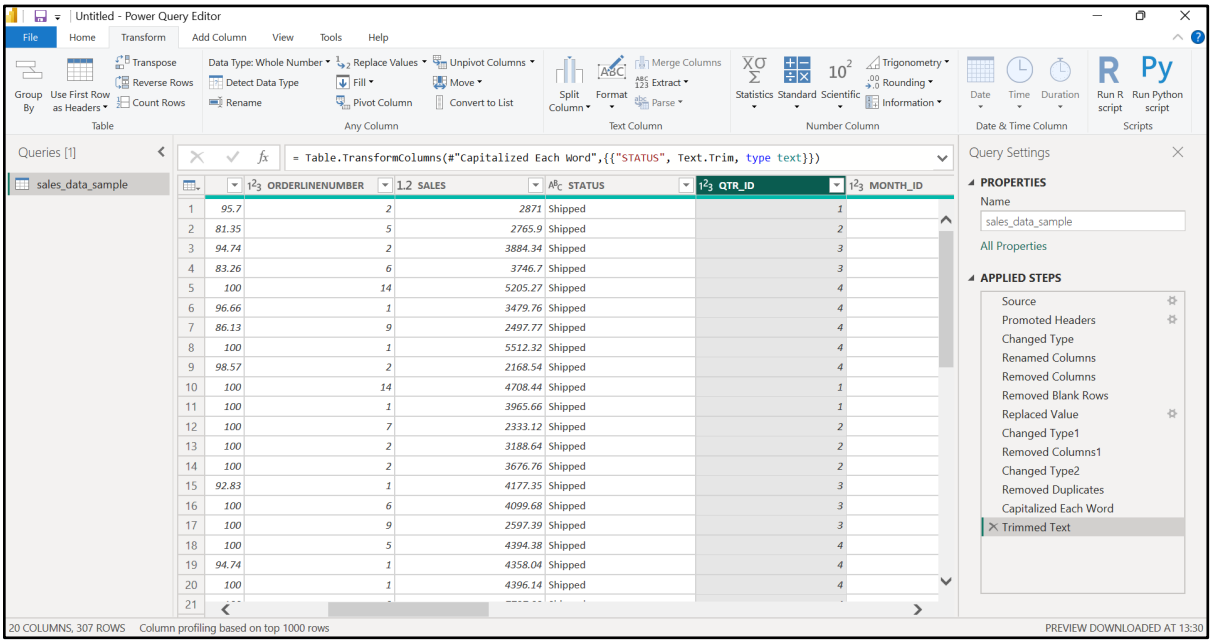
Use Remove Rows > Remove Blank Rows for completely empty records.



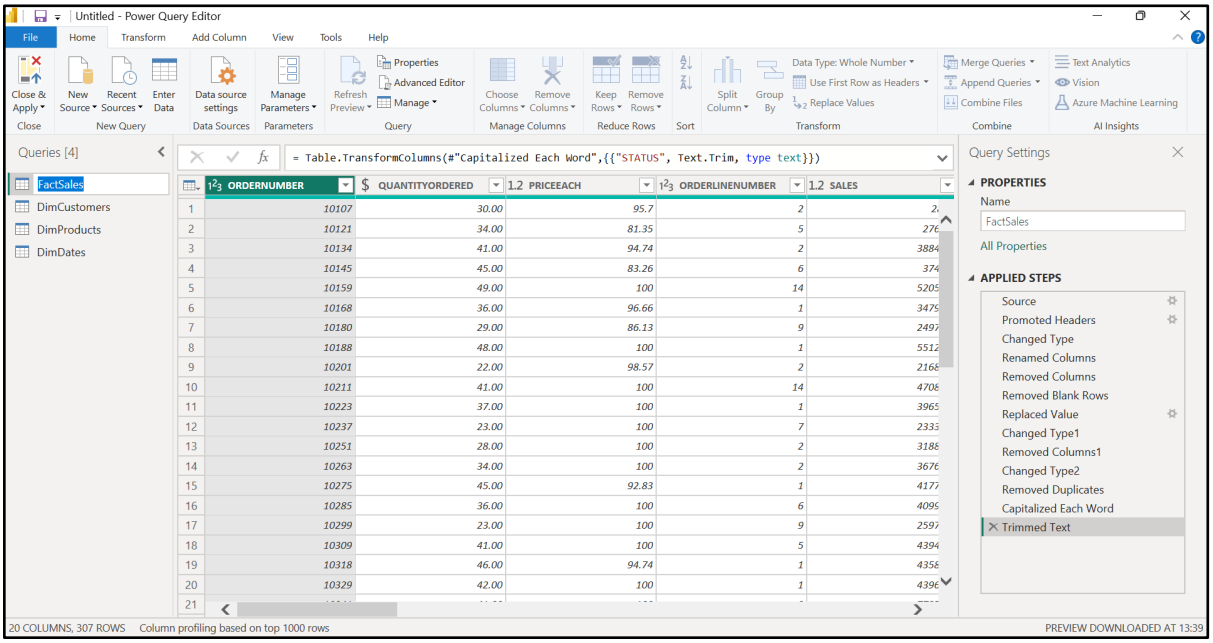
Use Replace Values to fill nulls with "Unknown" (for text) or 0 (for numeric fields).
Use Fill Down (for fields where previous values should be carried forward).
D. Change Data Types
Ensure that each column has the correct data type:
Dates: Convert to Date type (Order_Date, Ship_Date).
Numbers: Convert Quantity, Price, Total_Amount to Decimal Number or Whole Number.
Text: Convert Customer_Name, Product_Name, Region to Text.
E. Remove Duplicates
Identify and remove duplicates based on Order_ID or Invoice_ID (whichever is the unique identifier).



F. Standardize Text Formatting
Convert text fields to Proper Case (e.g., Product Name, Customer Name):
Text.Proper([Column_Name])
Trim spaces from text fields to avoid mismatches:
Text.Trim([Column_Name])

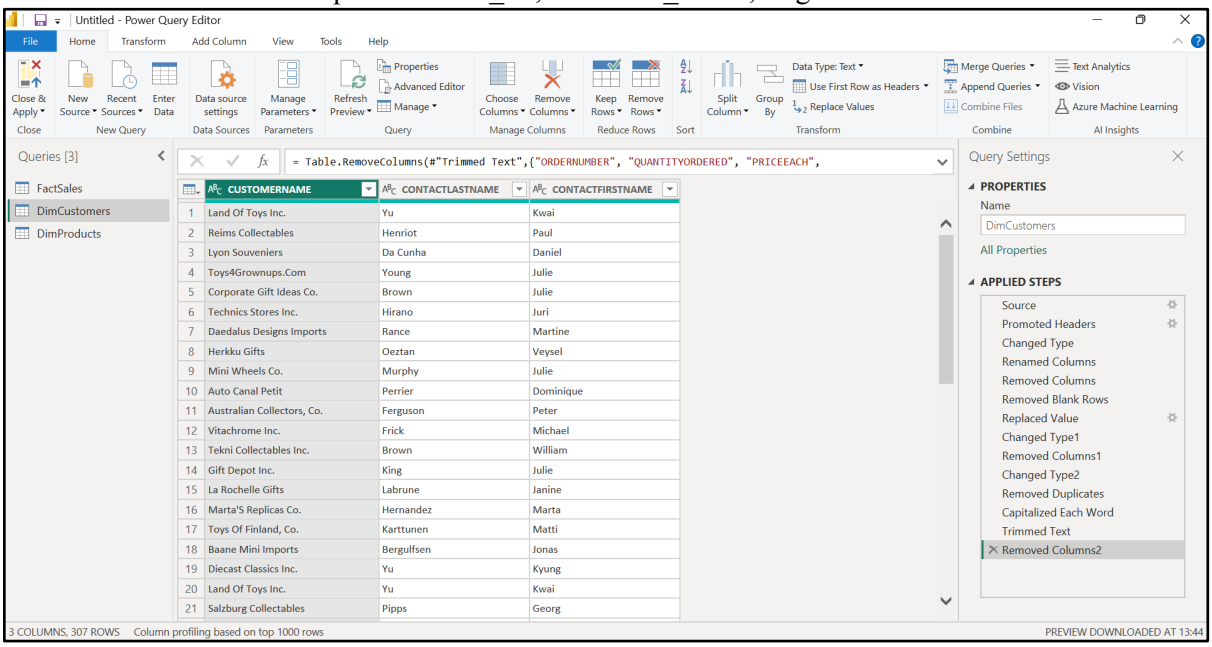


3. Create Staging Tables
Fact Table (FactSales):
Keep transactional data such as Order_ID, Date, Product_Code, Quantity, Total_Sales.

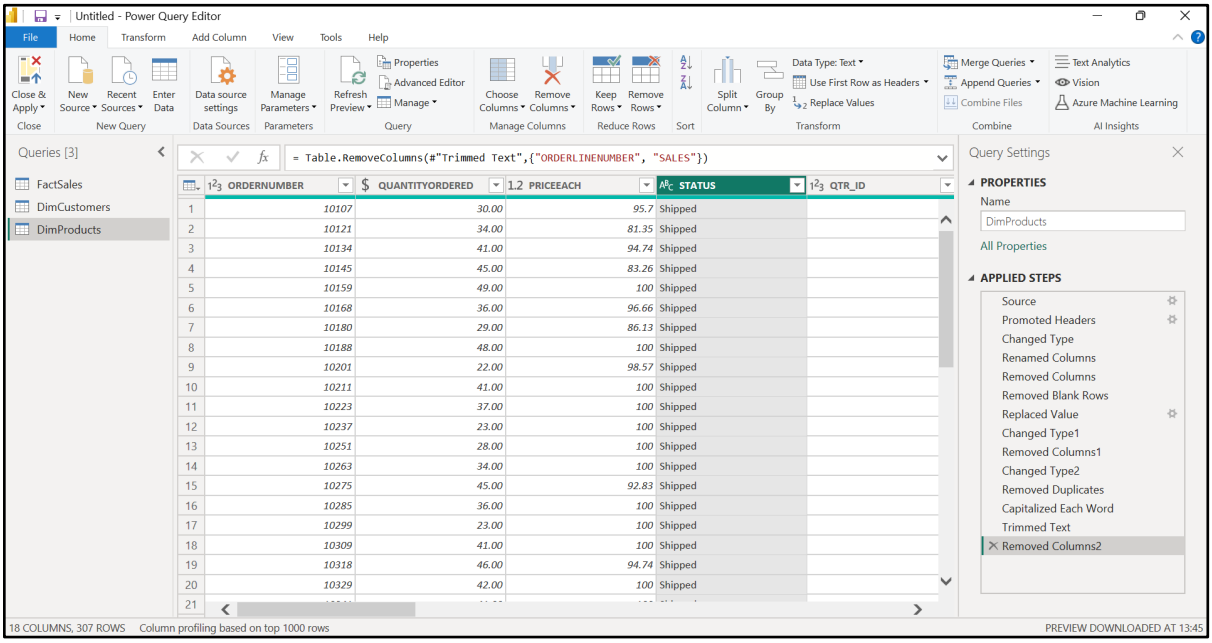


Dimension Tables:

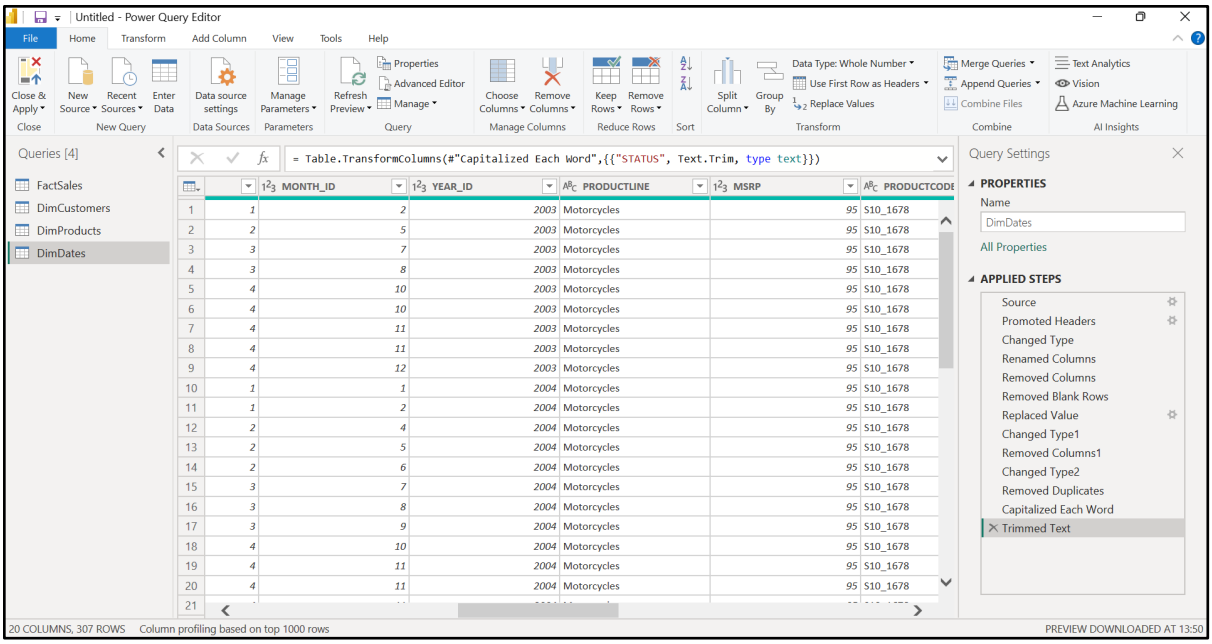
DimCustomers: Extract unique Customer_ID, Customer_Name, Region.



DimProducts: Extract unique Product_Code, Product_Name, Category.



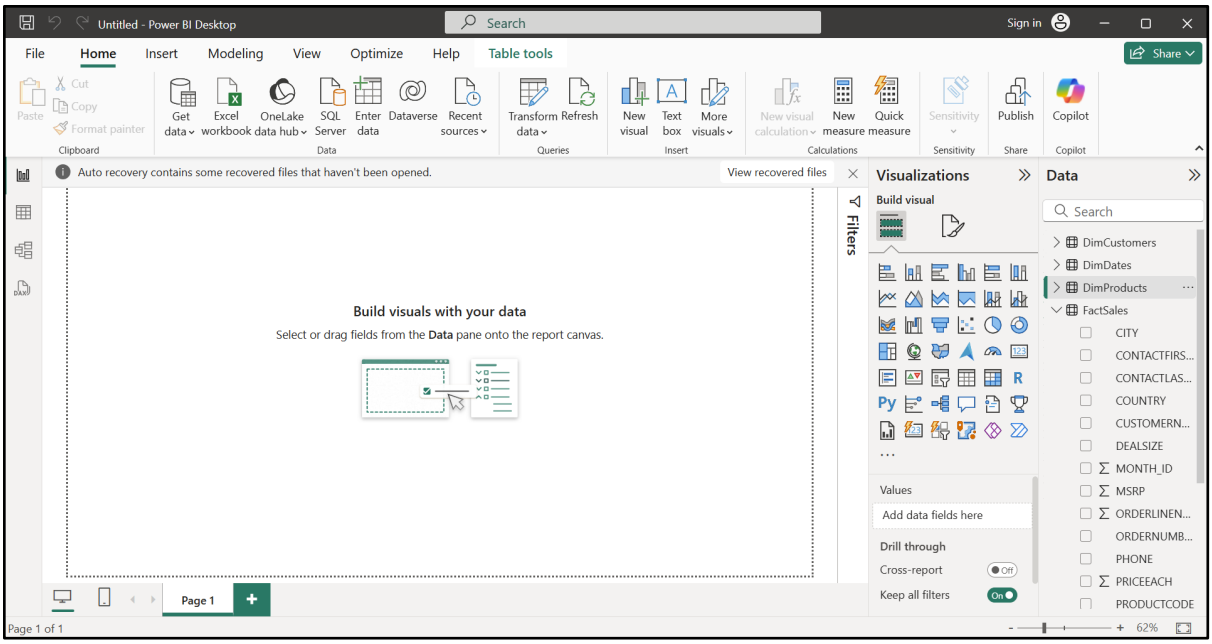
DimDates: Create a Date Table if not present (Order_Date, Year, Month, Quarter).



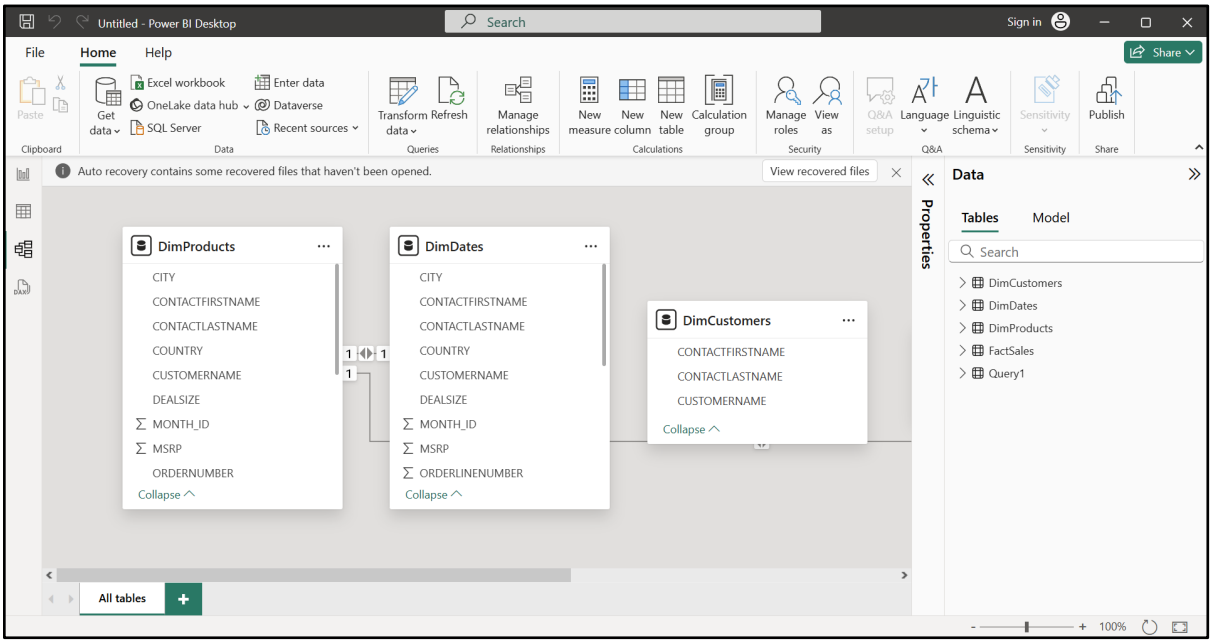
To create a Date Table in Power Query:
Go to New Source > Blank Query.
Open Advanced Editor and paste:

```
let
    StartDate = #date(2020, 1, 1),
    EndDate = #date(2030, 12, 31),
    DateList = List.Dates(StartDate, Number.From(EndDate - StartDate) + 1, #duration(1, 0, 0, 0)),
    DateTable = Table.FromList(DateList, Splitter.SplitByNothing(), {"Date"}),
    ChangedType = Table.TransformColumnTypes(DateTable, {{"Date", type date}})
in
    ChangedType
```

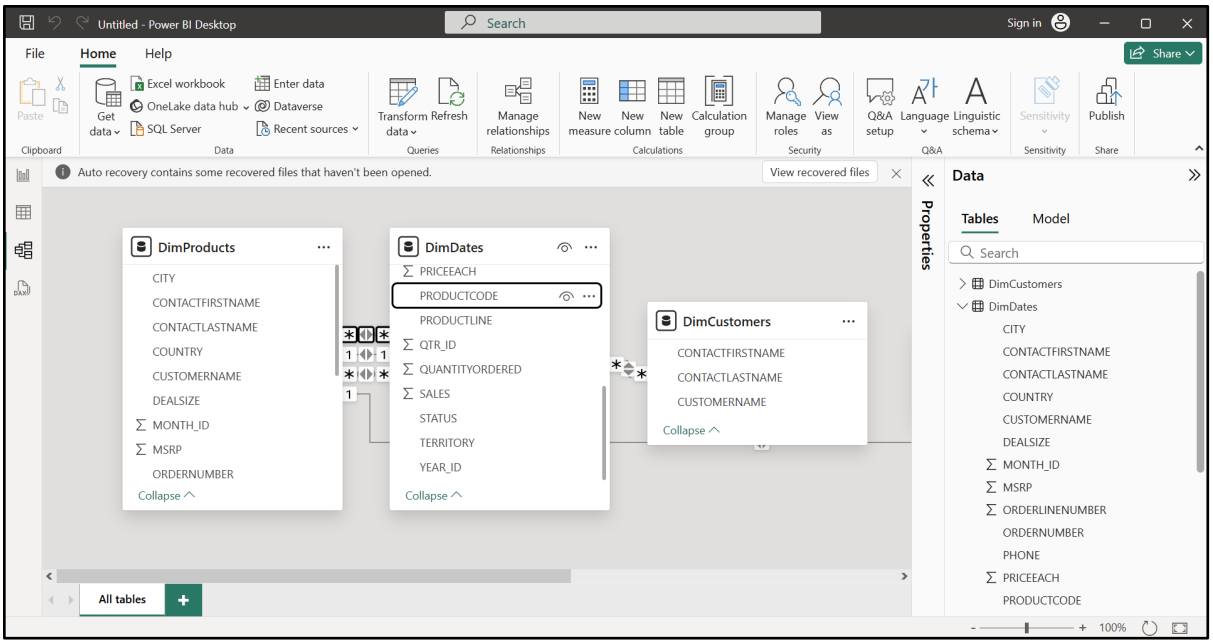
Click Close & Apply.



4. Define Relationships in Power BI Model

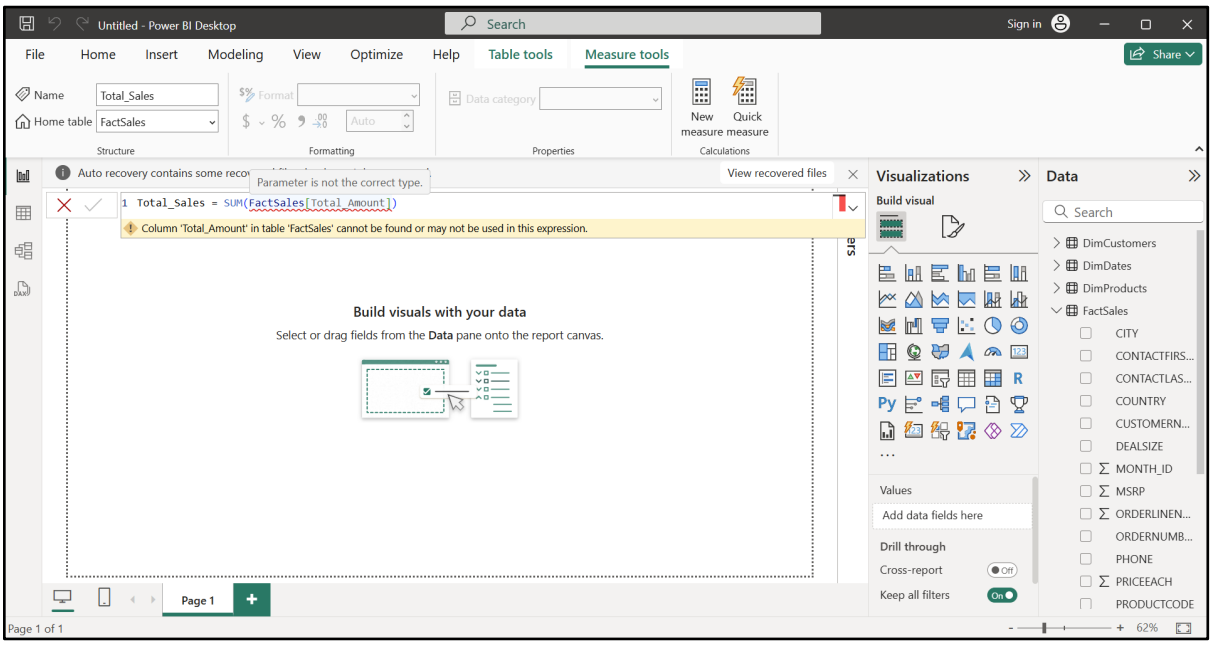


Go to Model View and establish relationships:
FactSales[Customer_ID] → DimCustomers[Customer_ID]
FactSales[Product_Code] → DimProducts[Product_Code]
FactSales[Order_Date] → DimDates[Date]

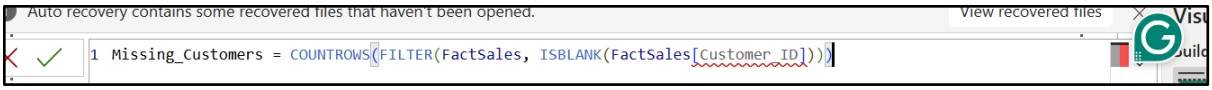


5. Create Data Validation Reports

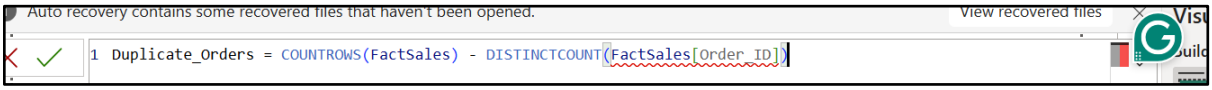
Use DAX Measures to check data quality:
Total Sales:
Total_Sales = SUM(FactSales[Total_Amount])



Count of Missing Customers:
Missing_Customers = COUNTROWS(FILTER(FactSales, ISBLANK(FactSales[Customer_ID])))

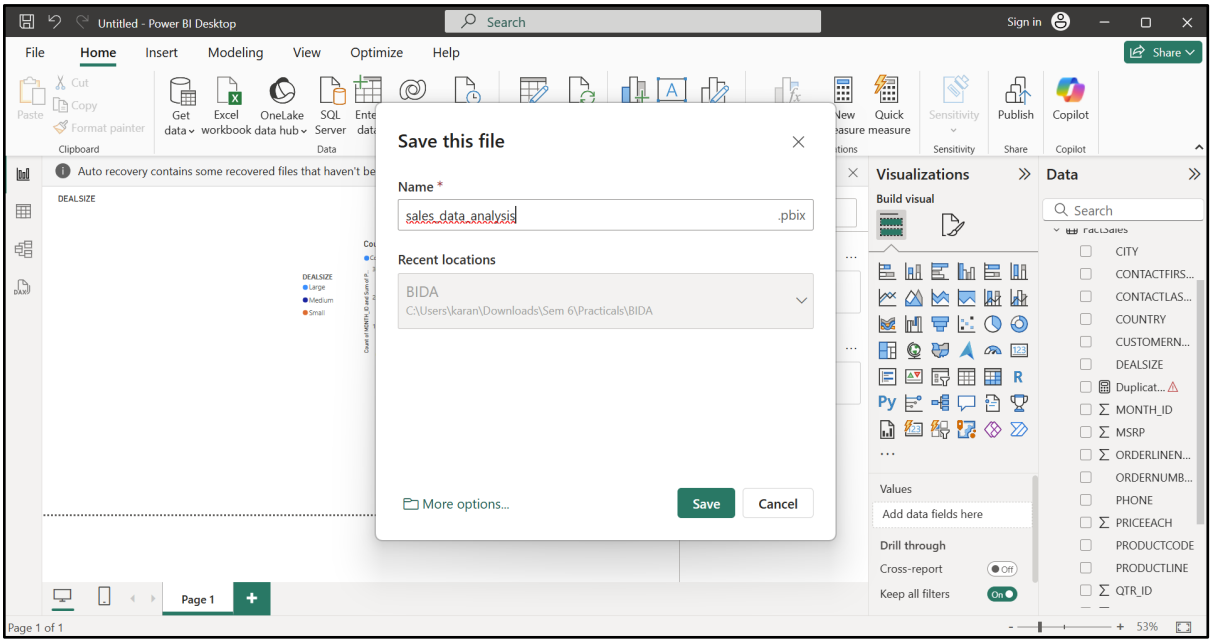


Duplicate Orders Check:
Duplicate_Orders = COUNTROWS(FactSales) - DISTINCTCOUNT(FactSales[Order_ID])



Create a Table Visual for missing data analysis.

6. Save & Publish

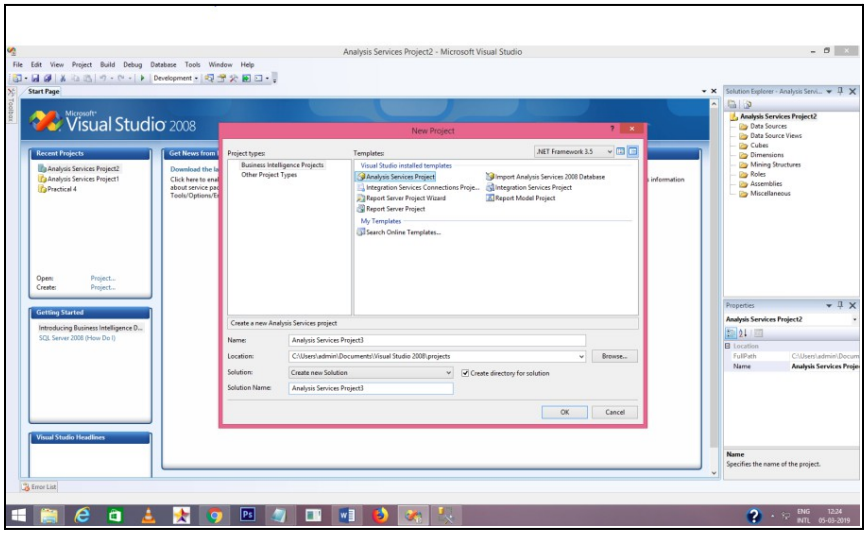


Click Close & Apply to load the transformed data into Power BI.

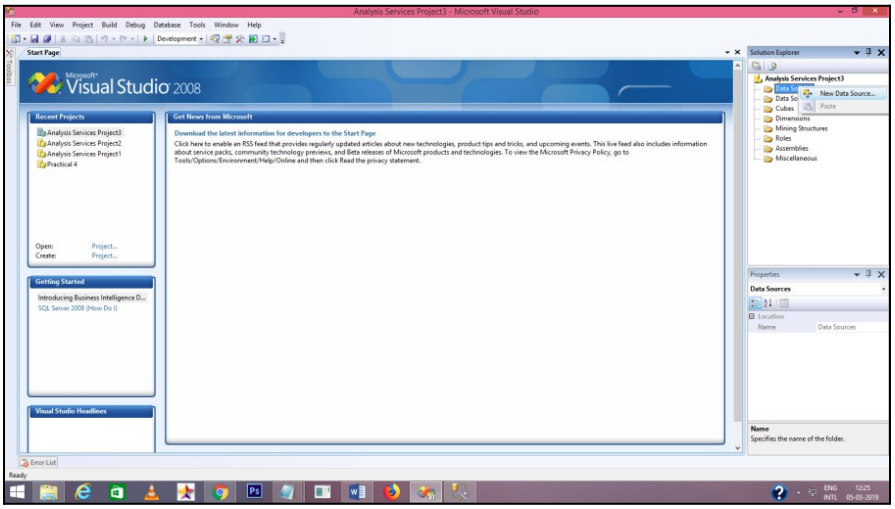
Practical 10

Create the cube with suitable dimension and fact tables based on ROLAP, MOLAP and HOLAP model.

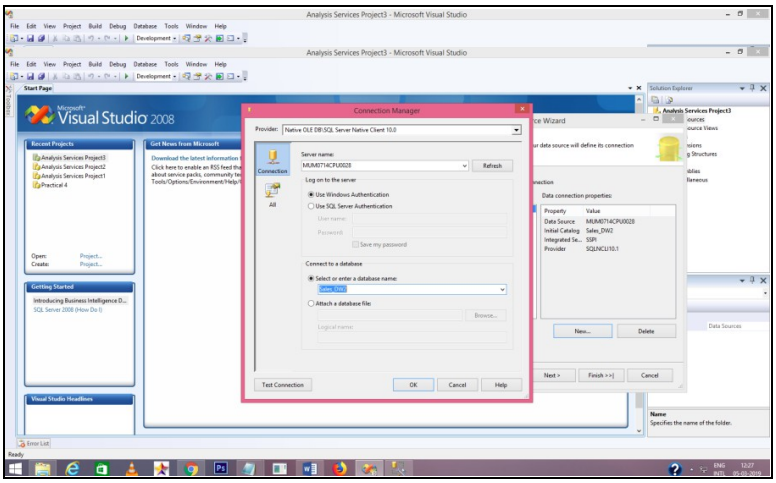
Step 1.Click File -> New -> Project ->Business Intelligence Projects ->select Analysis Services Project-> Assign Project Name -> Click OK.



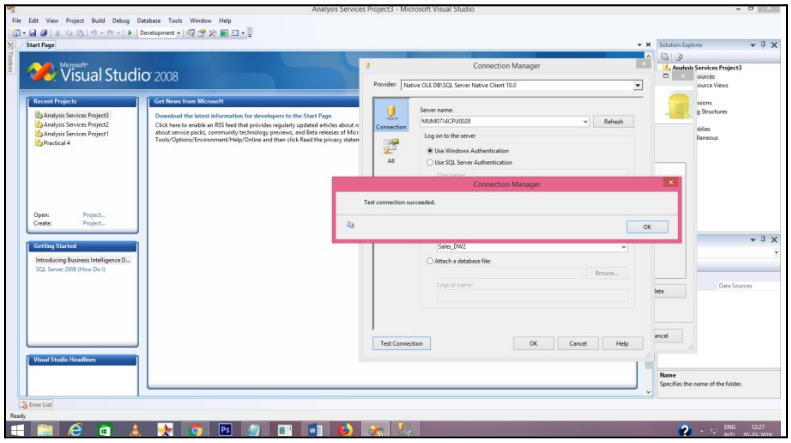
Step 2. In Solution Explorer, Right click on Data Source -> Click New Data Source



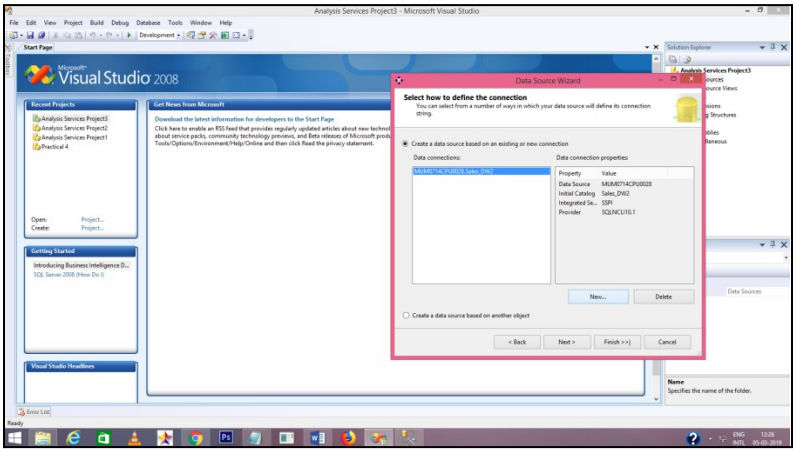
Step 3. Click on New



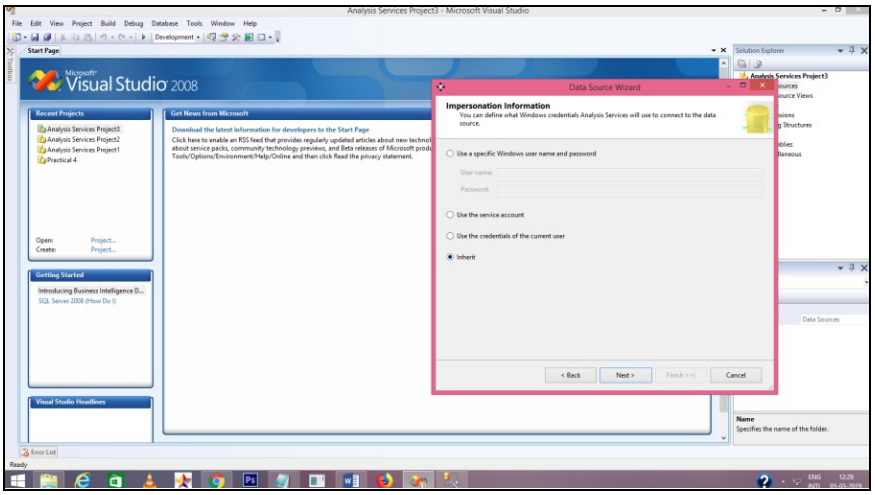
Step 4. Creating New Connection.
Step 5. Click on Test Connection and verify for its success.



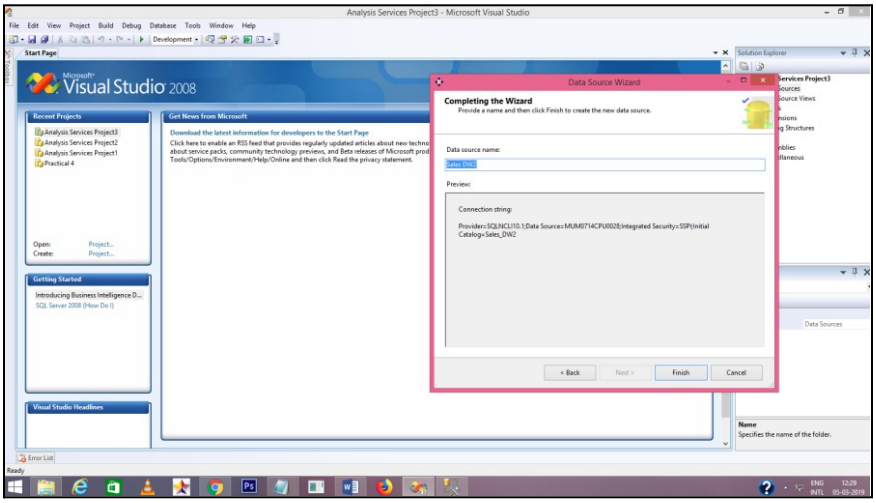
Step 6. Select Connection created in Data Connections-> Click Next.



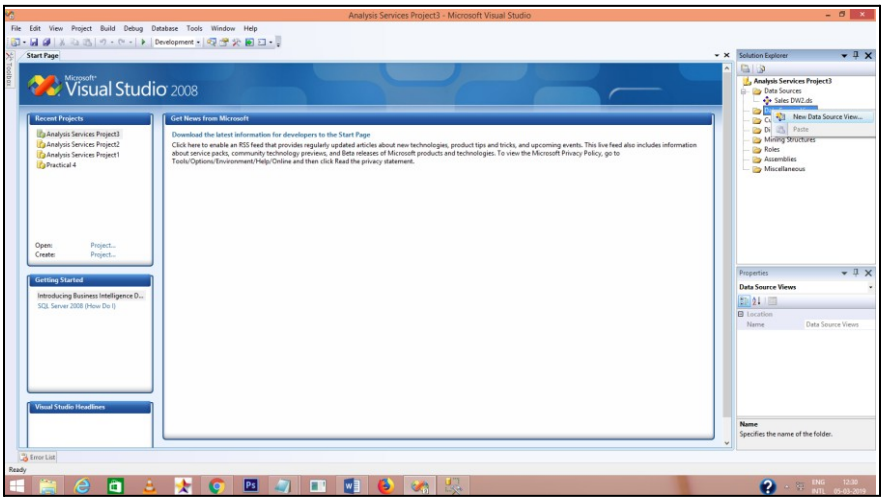
Step 7. Select Option Inherit.



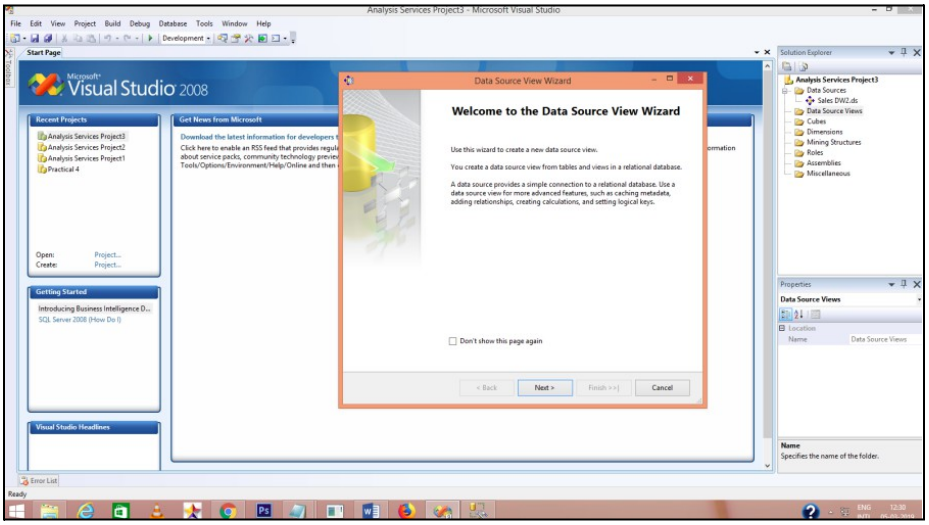
Step 8. Assign Data Source Name -> Click Finish



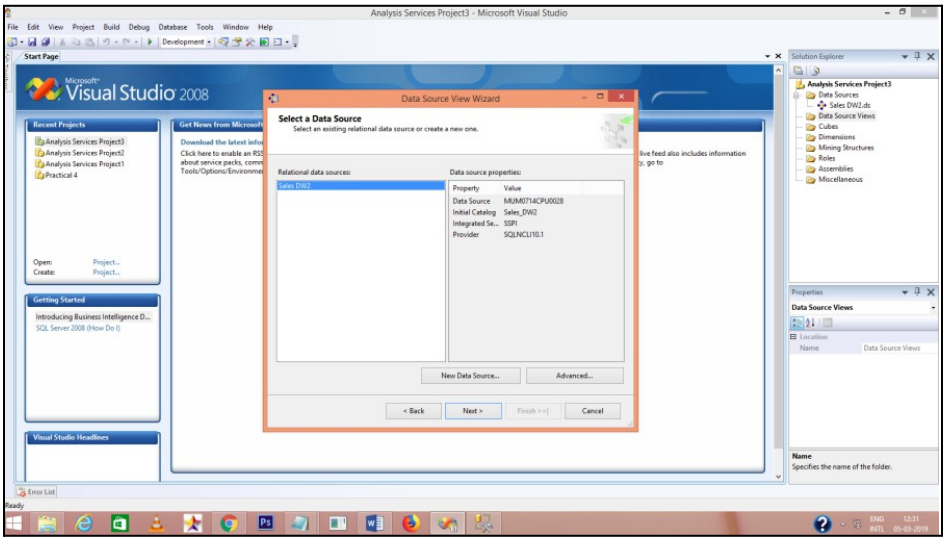
Step 9. In the Solution Explorer, Right Click on Data Source View -> Click on New Data Source View.



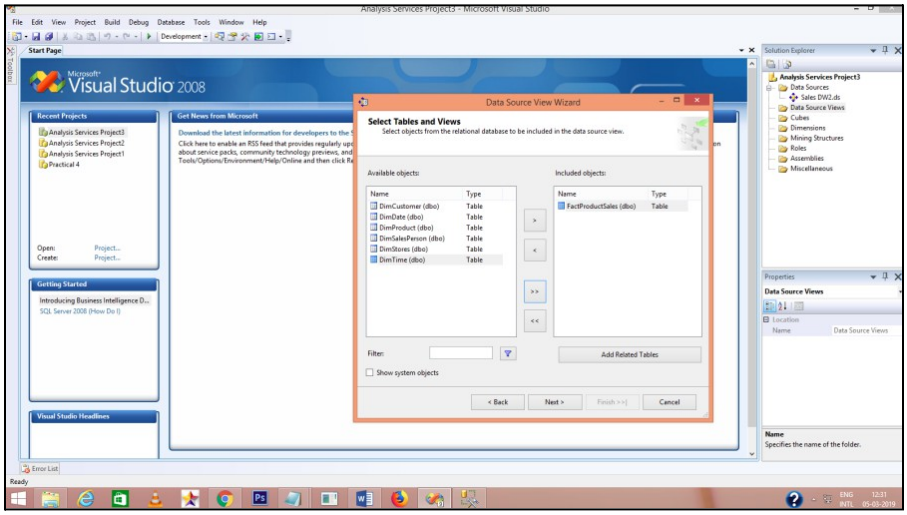
Step 10. Click Next.



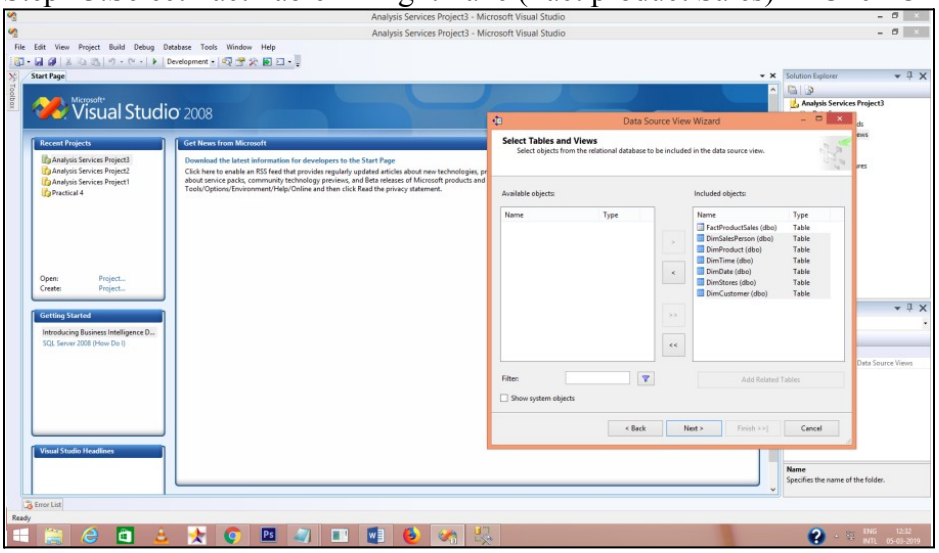
Step 11. Select Relational Data Source we have created previously (Sales_DW)-> Click Next.



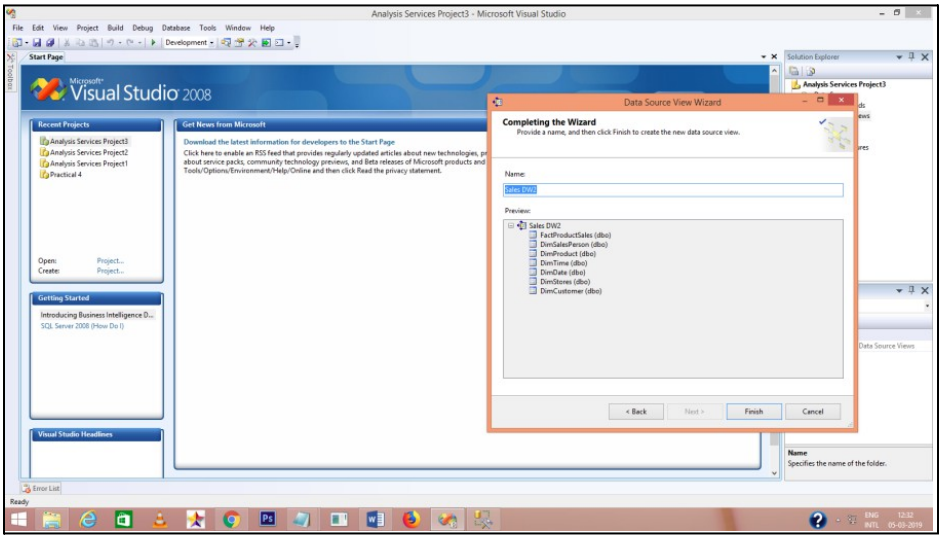
Step 12. First move your Fact Table to the right side to include in object list.



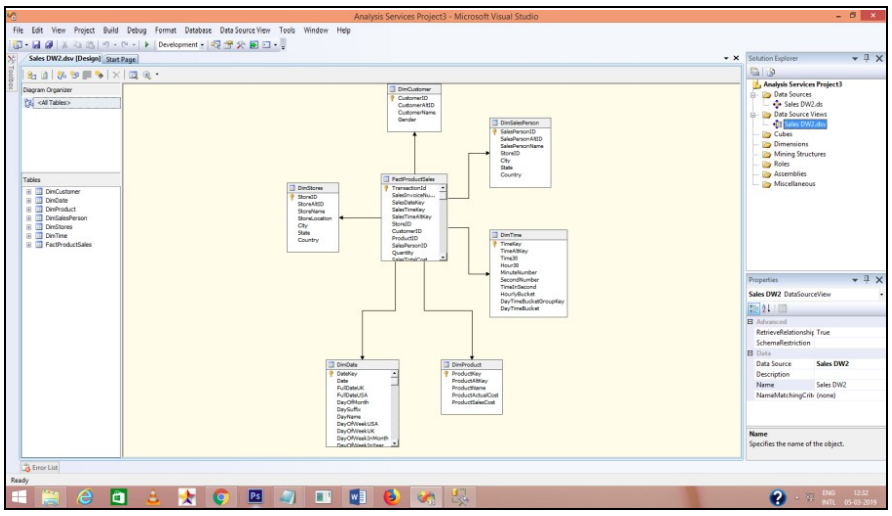
Step 13.Select Fact Table in Right Pane (Fact product Sales) -> Click On Add Related Tables.



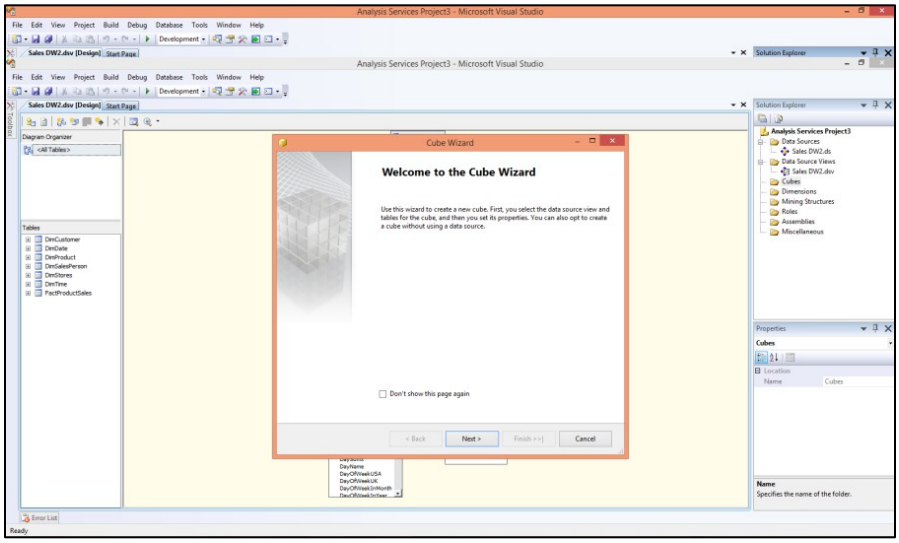
Step 14. Assign Name (SalesDW DSV)-> Click Finish



Step 15. Now Data Source View is ready to use.

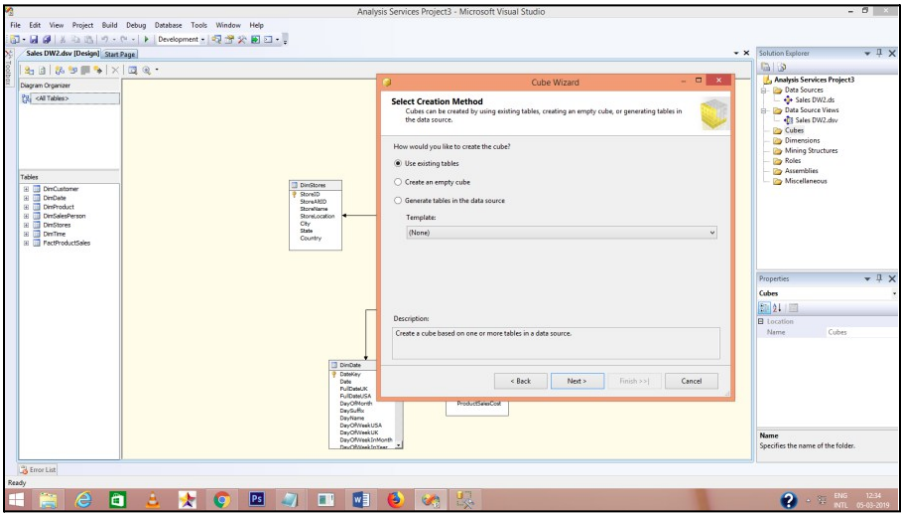


Step 16. In Solution Explorer -> Right Click on Cube-> Click New Cube.

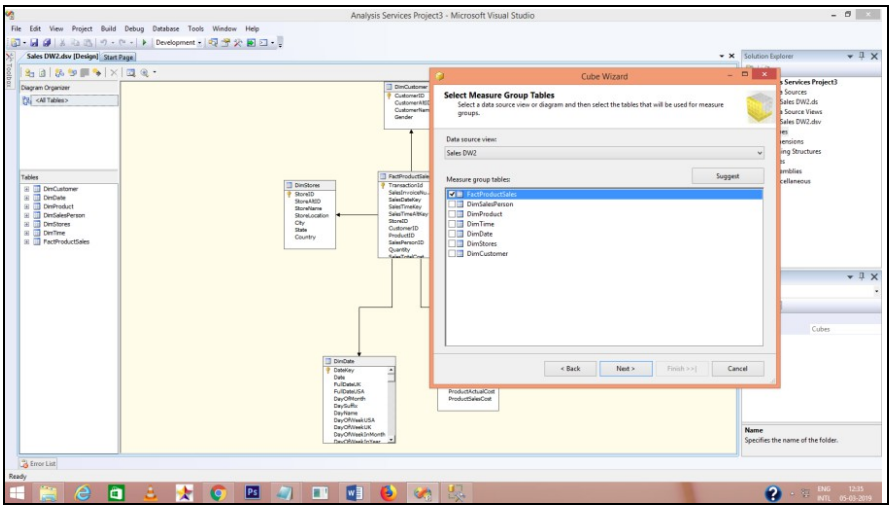


Step 17. Click Next.

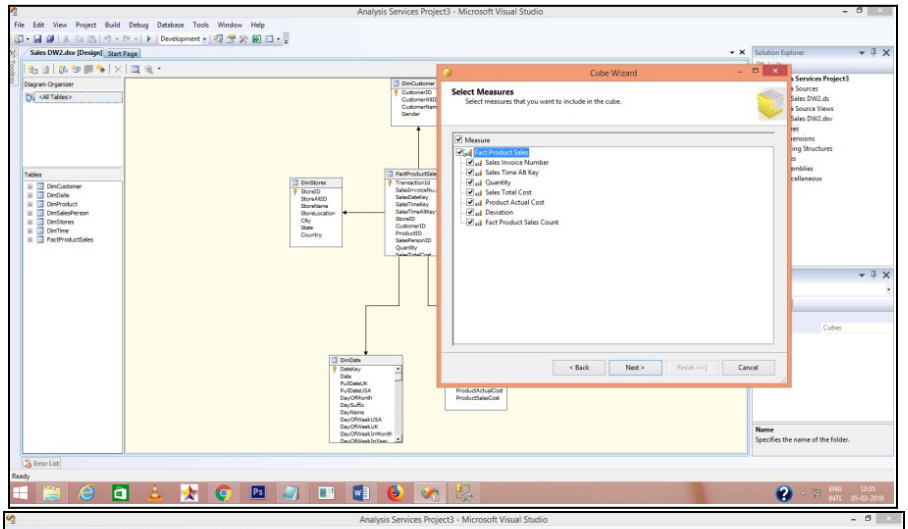
Step 18. Select Option Use existing Tables -> Click Next.

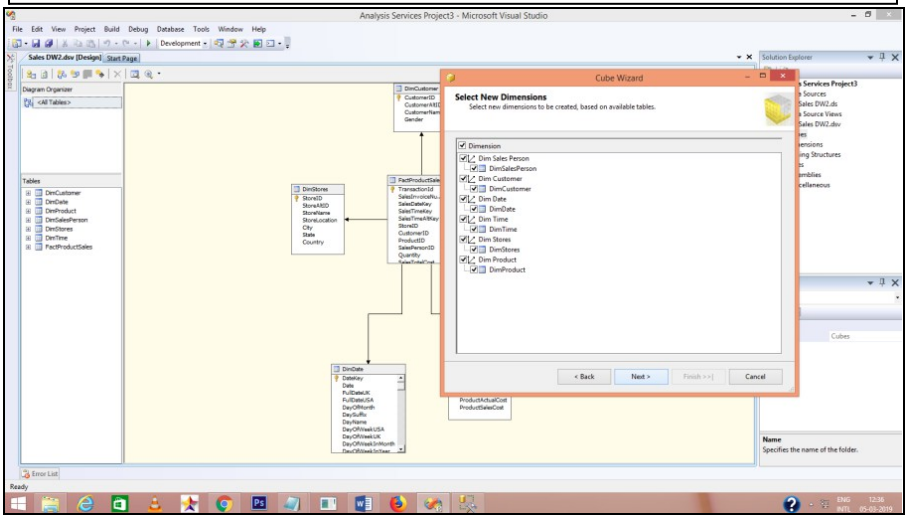


Step 19. Select Fact Table Name from Measure Group Tables (FactProductSales) -> Click Next.

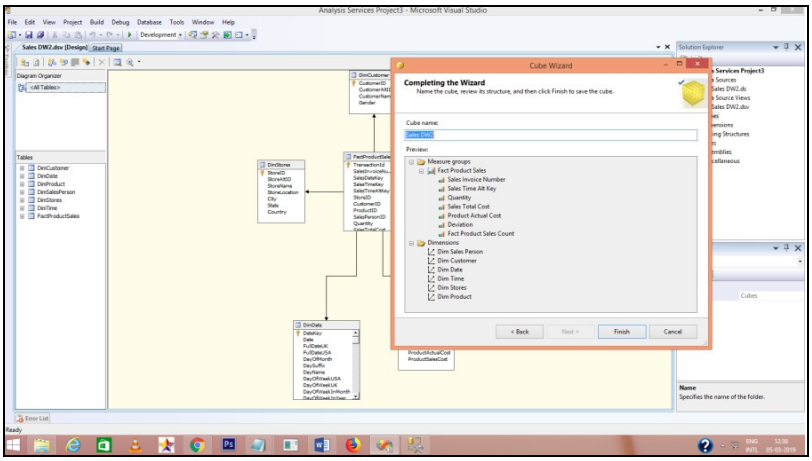


Step 20. Choose Measures from the List which you want to place in your Cube --> Click Next.
Step 21. Select All Dimensions here which are associated with your Fact Table-> Click Next

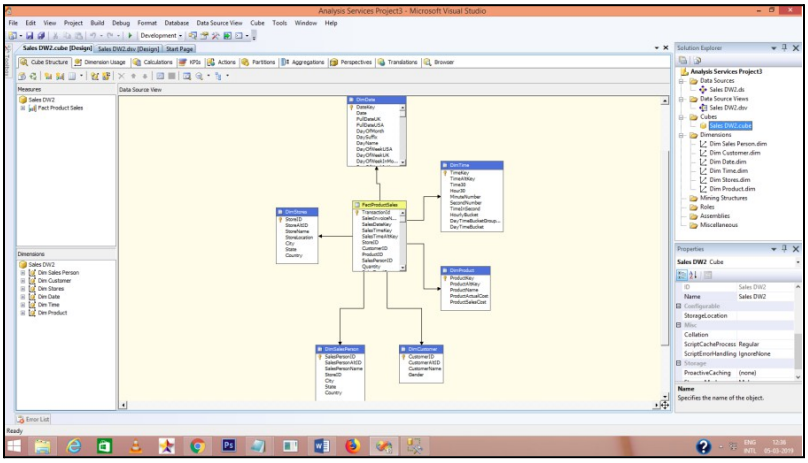




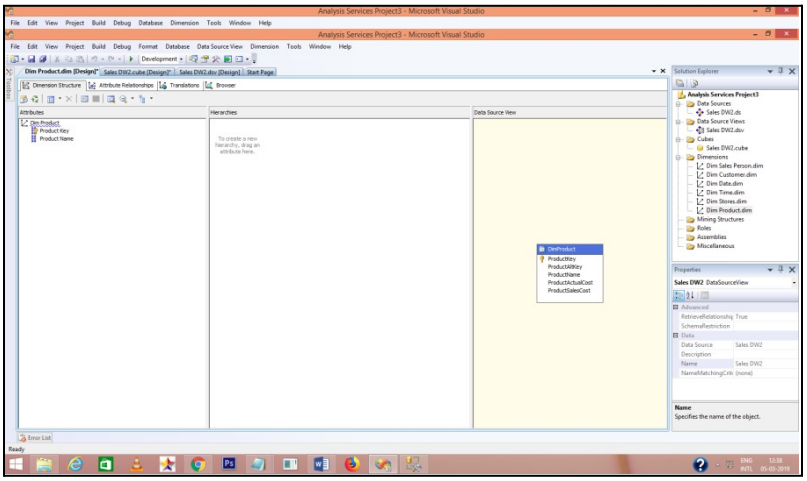
Step 22. Assign Cube Name (SalesDW2) -> Click Finish.



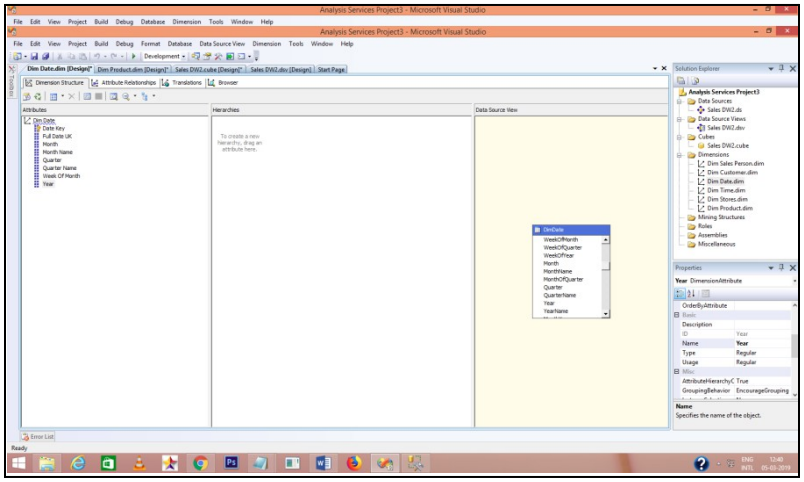
Step 23. Now your Cube is ready, you can see the newly created cube and dimensions added in your solution explorer.



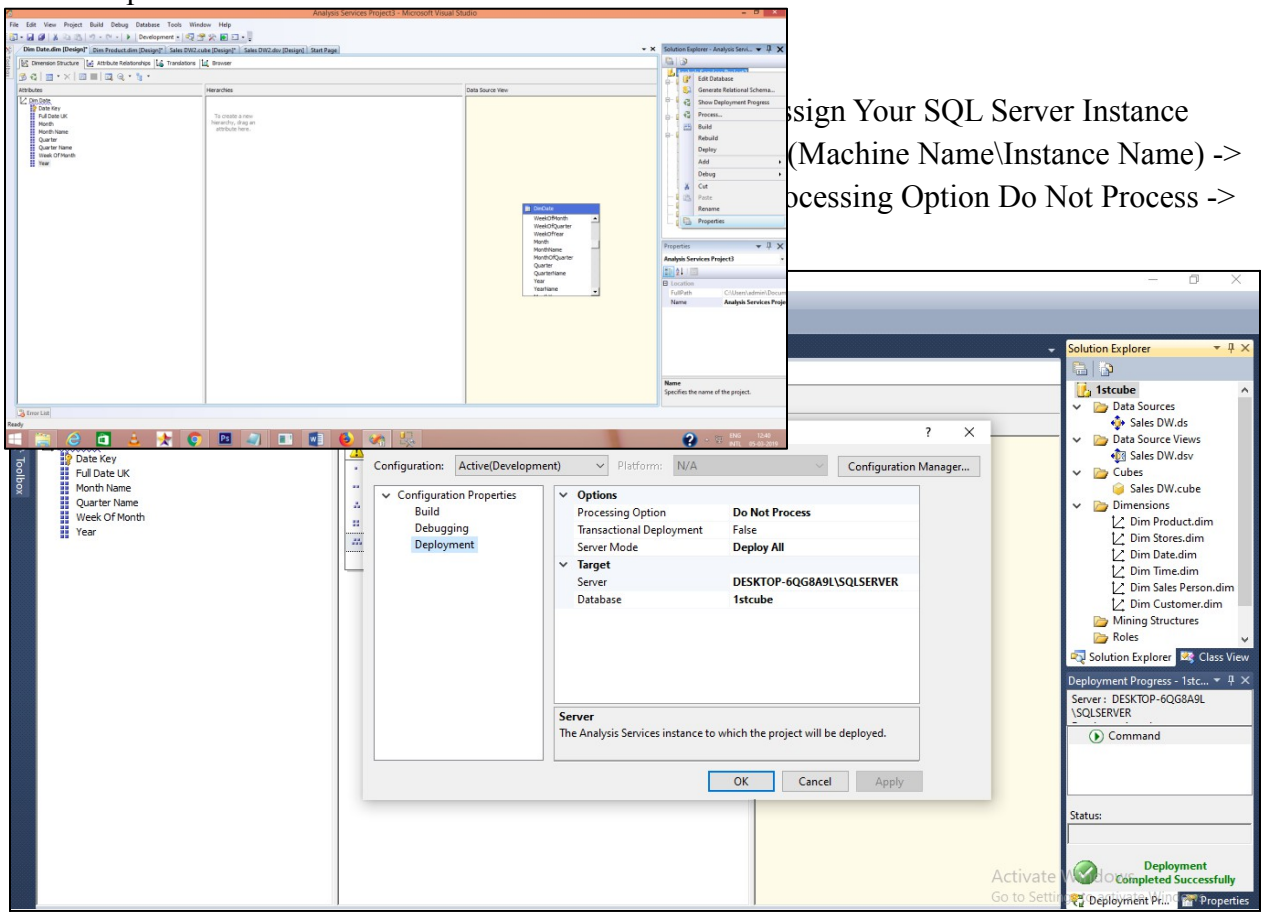
Step 24.In Solution Explorer, double click on dimension Dim Product -> Drag and Drop Product Name from Table in Data Source View and Add in Attribute Pane at left side.



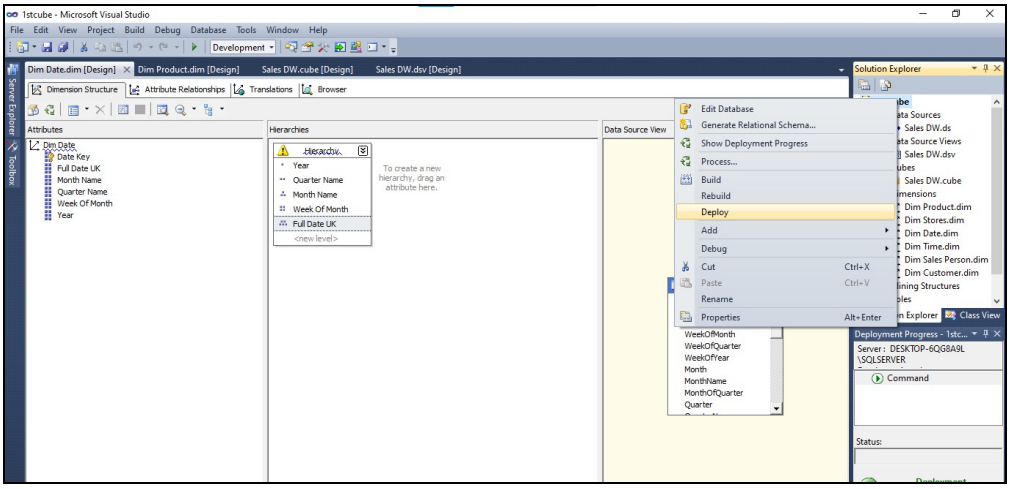
Step 25.Double click On Dim Date dimension -> Drag and Drop Fields from Table shown in Data Source View to Attributes-> Drag and Drop attributes from leftmost pane of attributes to middle pane of Hierarchy.



Step 26. In Solution Explorer, right click on Project Name (Analysis Services Project3) Click Properties.

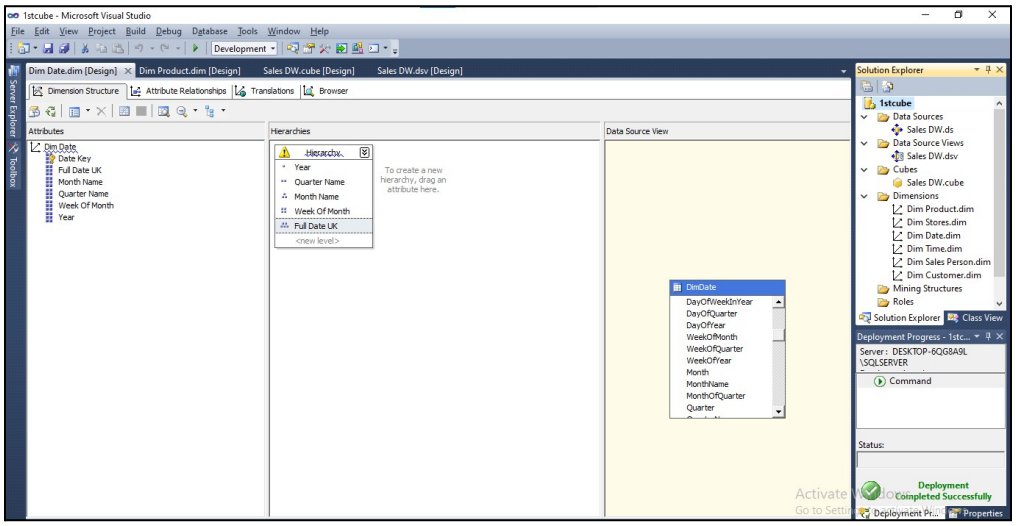


Step 28. In Solution Explorer, right click on Project Name (AnalysisServicesProject)

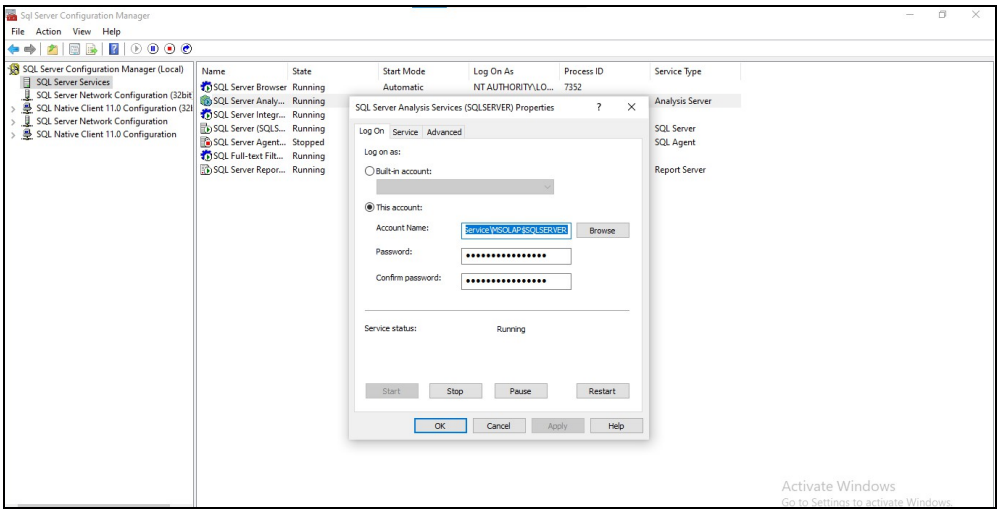


Click Deploy.

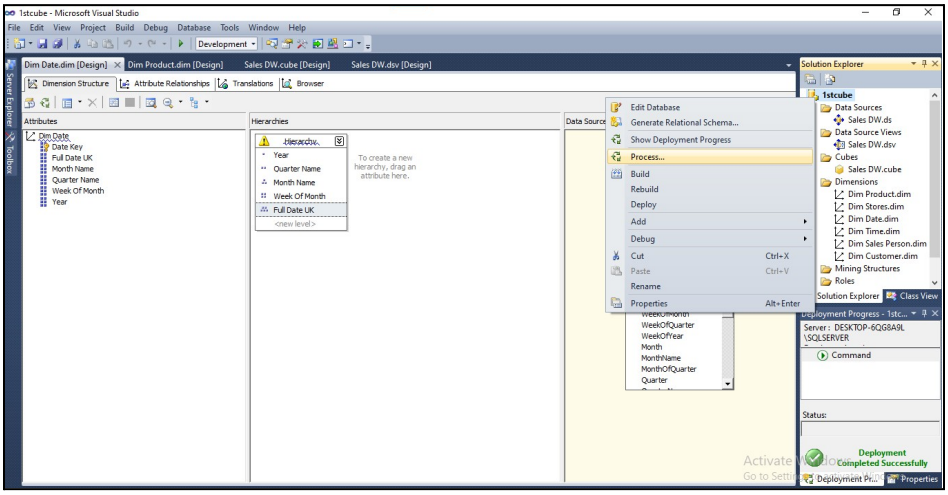
Step 29. Once Deployment will finish, you can see the message Deployment Completed in deployment Properties.



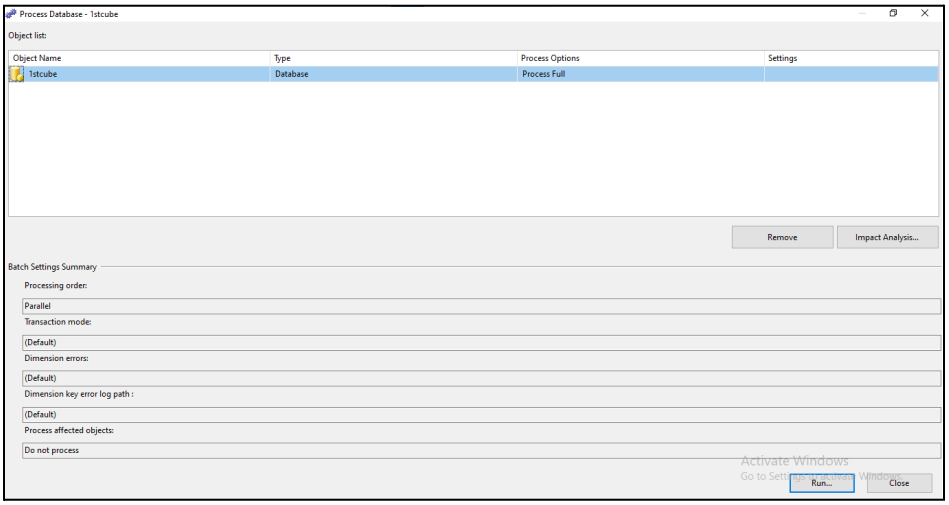
Step 30. Open SQL Server Configuration Server→Click on SQL Analysis Server→ Copy the account name and close the window.



Step 31. In Solution Explorer, right click on Project Name (AnalysisServicesProject3) →Click Process.

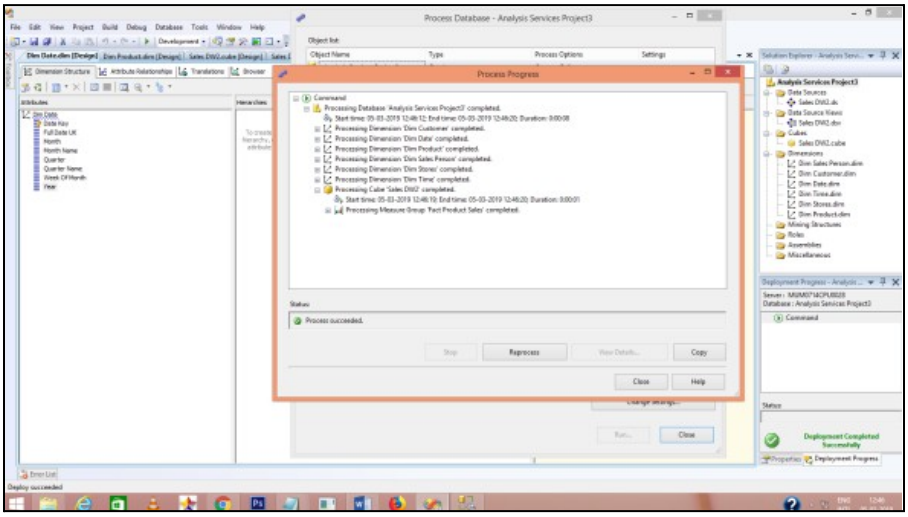


Step 32. Click on Run Button to process the 1stcube.



Step 33. Once processing is complete, you can see Status as Process Succeeded

-->Click Close to close both the open windows for processing one after the other.



Step 34. Now go to SQL Server Management Studio, disconnect the existing database and connect to Analysis Services→ in the database folder you will see the 1stcube has been successfully created.