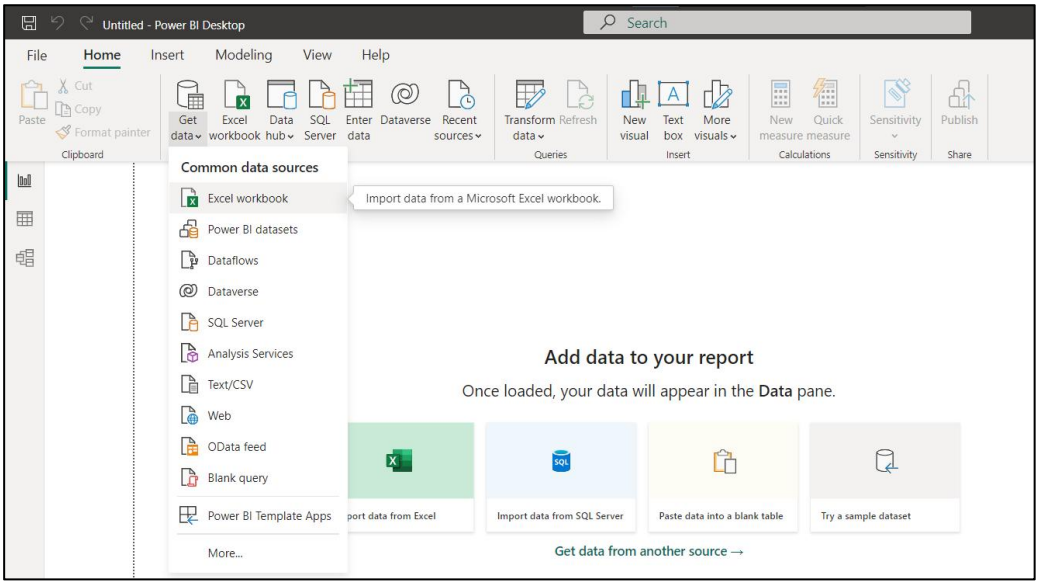
Practical No. 1

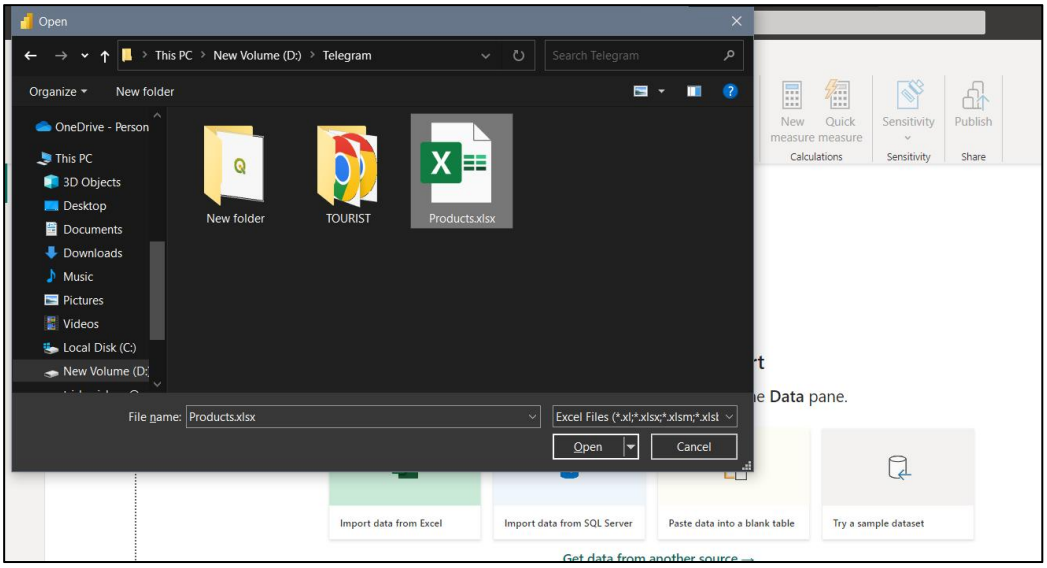
Aim: Import the legacy data from different sources such as (Excel, SQL Server, Oracle etc.) and load in the target system.

Importing Excel Data

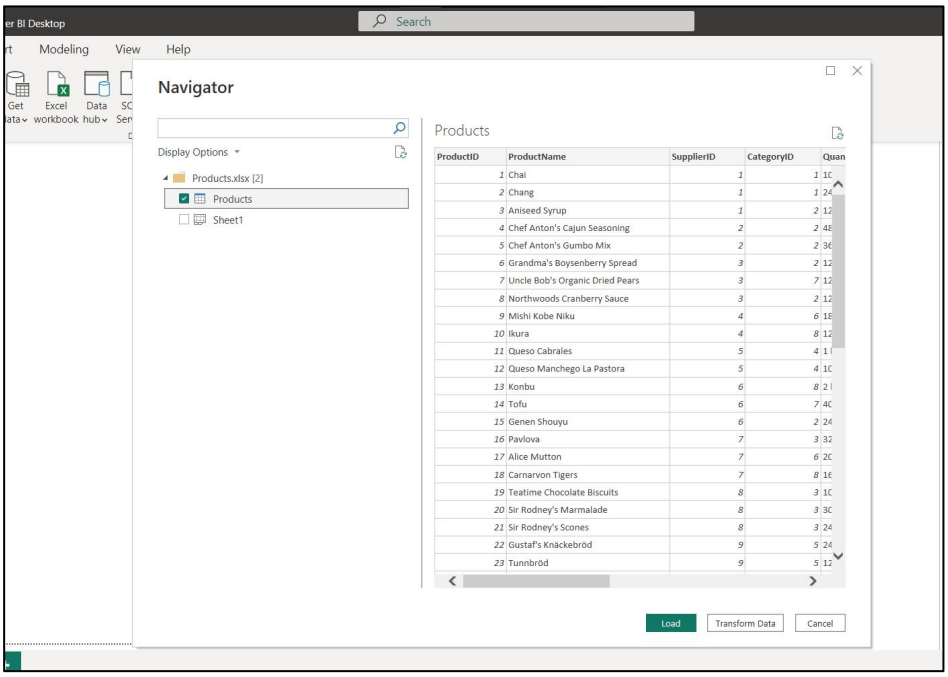
1) Launch Power BI Desktop.
2) From the Home ribbon, select Get Data. Excel is one of the Most Common data connections, so you can select it directly from the Get Data menu.



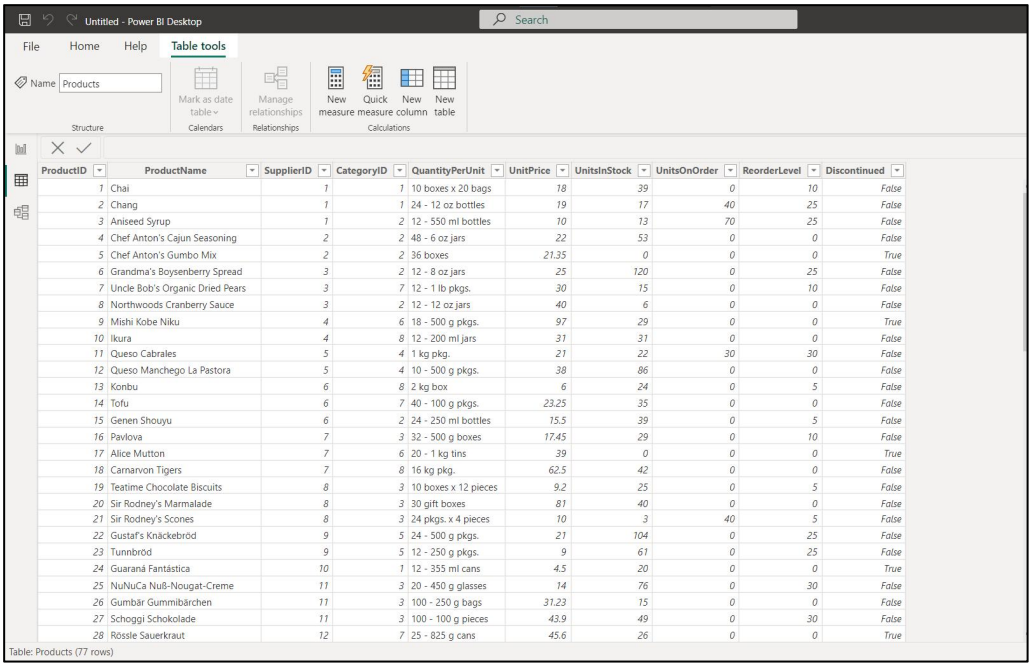
3) In the Open File dialog box, select the Products.xlsx file.



4) In the Navigator pane, select the Products table and then select Edit.



Output: -



ProductID	ProductName	SupplierID	CategoryID	QuantityPerUnit	UnitPrice	UnitsInStock	UnitsOnOrder	ReorderLevel	Discontinued
1	Chai	1	1	10 boxes x 20 bags	18	39	0	10	False
2	Chang	1	1	24 - 12 oz bottles	19	17	40	25	False
3	Aniseed Syrup	1	2	12 - 550 ml bottles	10	13	70	25	False
4	Chef Anton's Cajun Seasoning	2	2	48 - 6 oz jars	22	53	0	0	False
5	Chef Anton's Gumbo Mix	2	2	36 boxes	21.35	0	0	0	True
6	Grandma's Boysenberry Spread	3	2	12 - 8 oz jars	25	120	0	25	False
7	Uncle Bob's Organic Dried Pears	3	7	12 - 1 lb pkgs.	30	15	0	10	False
8	Northwoods Cranberry Sauce	3	2	12 - 12 oz jars	40	6	0	0	False
9	Mishi Kobe Niku	4	6	18 - 500 g pkgs.	97	29	0	0	True
10	Ikura	4	8	12 - 200 ml jars	31	31	0	0	False
11	Queso Cabrales	5	4	1 kg pkg.	21	22	30	30	False
12	Queso Manchego La Pastora	5	4	10 - 500 g pkgs.	38	86	0	0	False
13	Konbu	6	8	2 kg box	6	24	0	5	False
14	Tofu	6	7	40 - 100 g pkgs.	23.25	35	0	0	False
15	Genen Shouyu	6	2	24 - 250 ml bottles	15.5	39	0	5	False
16	Pavlova	7	3	32 - 500 g boxes	17.45	29	0	10	False
17	Alice Mutton	7	6	20 - 1 kg tins	39	0	0	0	True
18	Carnarvon Tigers	7	8	16 kg pkg.	62.5	42	0	0	False
19	Teatime Chocolate Biscuits	8	3	10 boxes x 12 pieces	9.2	25	0	5	False
20	Sir Rodney's Marmalade	8	3	30 gift boxes	81	40	0	0	False
21	Sir Rodney's Scones	8	3	24 pkgs. x 4 pieces	10	3	40	5	False
22	Gustaf's Knäckebröd	9	5	24 - 500 g pkgs.	21	104	0	25	False
23	Tunnbröd	9	5	12 - 250 g pkgs.	9	61	0	25	False
24	Guaraná Fantástica	10	1	12 - 355 ml cans	4.5	20	0	0	True
25	NuNuCa Nuß-Nougat-Creme	11	3	20 - 450 g glasses	14	76	0	30	False
26	Gumbär Gummibärchen	11	3	100 - 250 g bags	31.23	15	0	0	False
27	Schoggi Schokolade	11	3	100 - 100 g pieces	43.9	49	0	30	False
28	Rössle Sauerkraut	12	7	25 - 825 g cans	45.6	26	0	0	True

Table: Products (77 rows)

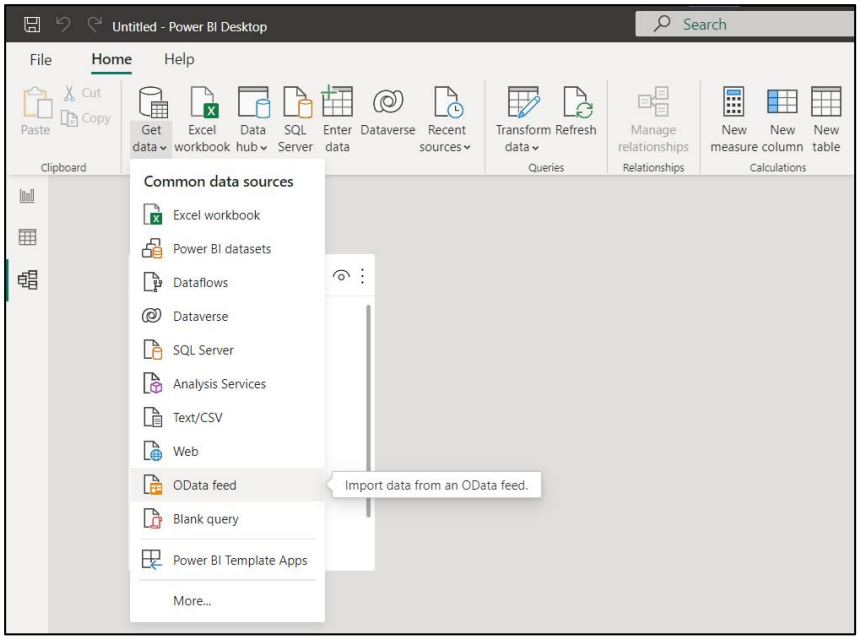
Importing Data from OData Feed

Step 1: - Download data of North wind ODATA feed from: -

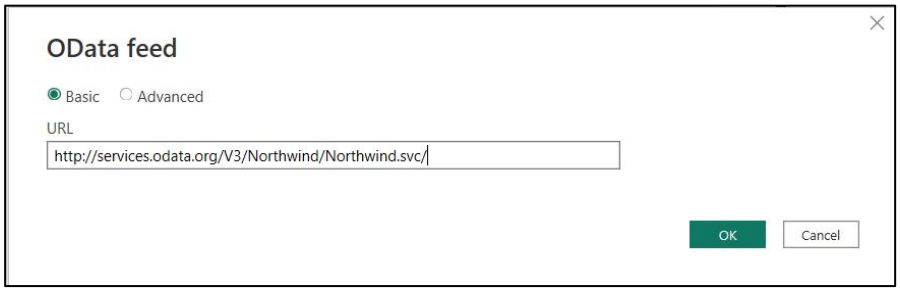
http://services.odata.org/V3/Northwind/Northwind.svc/

Connect to an OData feed:

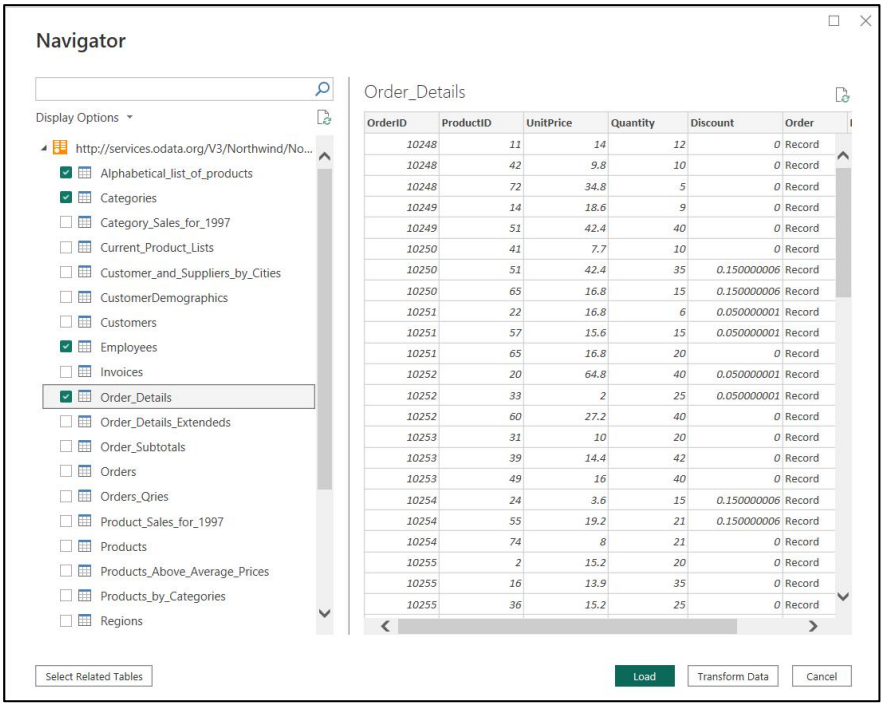
1) From the Home ribbon tab in Query Editor, select Get Data.
2) Browse to the OData Feed data source.



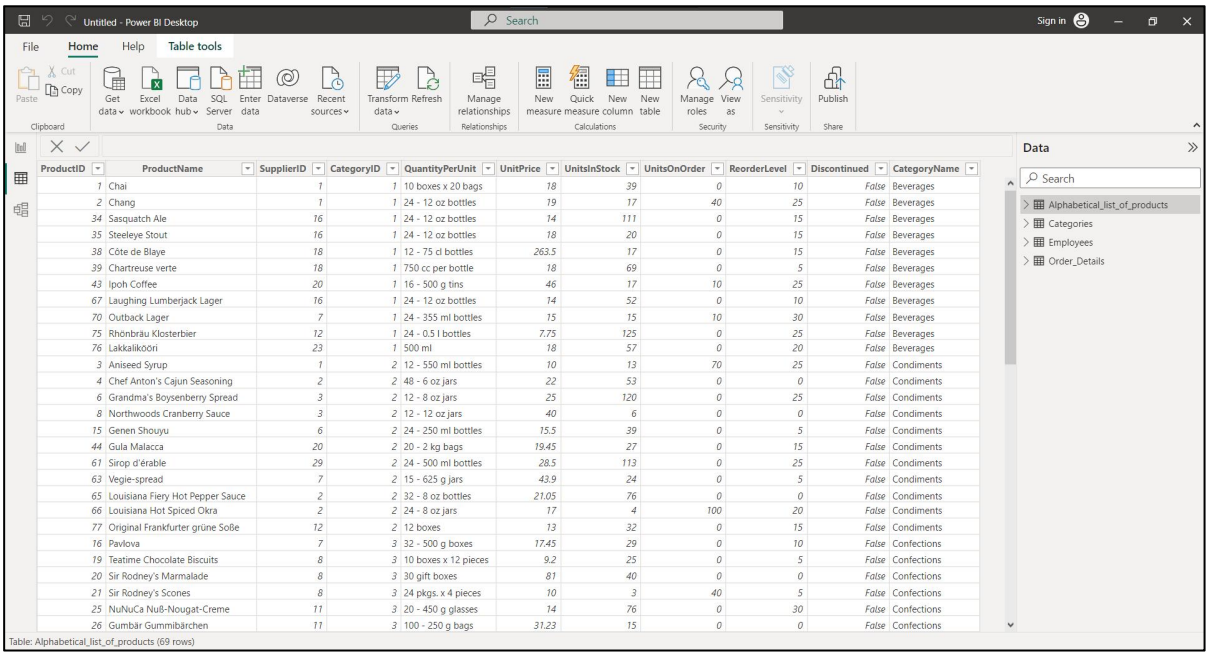
3) In the OData Feed dialog box, paste the URL for the North wind OData feed.



4) Select OK.
5) In the Navigator pane, select the Orders table, and then select Edit.


Navigator
Display Options
http://services.odata.org/V3/Northwind/No...
Alphabetical_list_of_products
Categories
Category_Sales_for_1997
Current_Product_Lists
Customer_and_Suppliers_by_Cities
CustomerDemographics
Customers
Employees
Invoices
Order_Details
Order_Details_Extendeds
Order_Subtotals
Orders
Orders_Qries
Product_Sales_for_1997
Products
Products_Above_Average_Prices
Products_by_Categories
Regions
Order_Details
OrderID ProductID UnitPrice Quantity Discount Order
10248 11 14 12 0 Record
10248 42 9.8 10 0 Record
10248 72 34.8 5 0 Record
10249 14 18.6 9 0 Record
10249 51 42.4 40 0 Record
10250 41 7.7 10 0 Record
10250 51 42.4 35 0.150000006 Record
10250 65 16.8 15 0.150000006 Record
10251 22 16.8 6 0.050000001 Record
10251 57 15.6 15 0.050000001 Record
10251 65 16.8 20 0 Record
10252 20 64.8 40 0.050000001 Record
10252 33 2 25 0.050000001 Record
10252 60 27.2 40 0 Record
10253 31 10 20 0 Record
10253 39 14.4 42 0 Record
10253 49 16 40 0 Record
10254 24 3.6 15 0.150000006 Record
10254 55 19.2 21 0.150000006 Record
10254 74 8 21 0 Record
10255 2 15.2 20 0 Record
10255 16 13.9 35 0 Record
10255 36 15.2 25 0 Record
Select Related Tables
Load
Transform Data
Cancel

Output: -


Untitled - Power BI Desktop
Search
Sign in
File Home Help Table tools
Paste Cut Copy
Get data Excel workbook Data hub SQL Server Enter data Dataverse Recent sources
Transform data Refresh
Manage relationships
New measure Quick measure New column New table
Manage roles View as
Sensitivity
Publish
Clipboard Data Queries Relationships Calculations Security Sensitivity Share
ProductID ProductName SupplierID CategoryID QuantityPerUnit UnitPrice UnitsInStock UnitsOnOrder ReorderLevel Discontinued CategoryName
1 Chai 1 1 10 boxes x 20 bags 18 39 0 10 False Beverages
2 Chang 1 1 24 - 12 oz bottles 19 17 40 25 False Beverages
34 Sasquatch Ale 16 1 24 - 12 oz bottles 14 111 0 15 False Beverages
35 Steeleye Stout 16 1 24 - 12 oz bottles 18 20 0 15 False Beverages
38 Côte de Blaye 18 1 12 - 75 cl bottles 263.5 17 0 15 False Beverages
39 Chartreuse verte 18 1 750 cc per bottle 18 69 0 5 False Beverages
43 Ipoh Coffee 20 1 16 - 500 g tins 46 17 10 25 False Beverages
67 Laughing Lumberjack Lager 16 1 24 - 12 oz bottles 14 52 0 10 False Beverages
70 Outback Lager 7 1 24 - 355 ml bottles 15 15 10 30 False Beverages
75 Rhönbräu Klosterbier 12 1 24 - 0.5 l bottles 7.75 125 0 25 False Beverages
76 Lakkalikööri 23 1 500 ml 18 57 0 20 False Beverages
3 Aniseed Syrup 1 2 12 - 550 ml bottles 10 13 70 25 False Condiments
4 Chef Anton's Cajun Seasoning 2 2 48 - 6 oz jars 22 53 0 0 False Condiments
6 Grandma's Boysenberry Spread 3 2 12 - 8 oz jars 25 120 0 25 False Condiments
8 Northwoods Cranberry Sauce 3 2 12 - 12 oz jars 40 6 0 0 False Condiments
15 Genen Shouyu 6 2 24 - 250 ml bottles 15.5 39 0 5 False Condiments
44 Gula Malacca 20 2 20 - 2 kg bags 19.45 27 0 15 False Condiments
61 Sirop d'érable 29 2 24 - 500 ml bottles 28.5 113 0 25 False Condiments
63 Vegie-spread 7 2 15 - 625 g jars 43.9 24 0 5 False Condiments
65 Louisiana Fiery Hot Pepper Sauce 2 2 32 - 8 oz bottles 21.05 76 0 0 False Condiments
66 Louisiana Hot Spiced Okra 2 2 24 - 8 oz jars 17 4 100 20 False Condiments
77 Original Frankfurter grüne Soße 12 2 12 boxes 13 32 0 15 False Condiments
16 Pavlova 7 3 32 - 500 g boxes 17.45 29 0 10 False Confections
19 Teatime Chocolate Biscuits 8 3 10 boxes x 12 pieces 9.2 25 0 5 False Confections
20 Sir Rodney's Marmalade 8 3 30 gift boxes 81 40 0 0 False Confections
21 Sir Rodney's Scones 8 3 24 pkgs. x 4 pieces 10 3 40 5 False Confections
25 NuNuCa Nuß-Nougat-Creme 11 3 20 - 450 g glasses 14 76 0 30 False Confections
26 Gumbär Gummibärchen 11 3 100 - 250 g bags 31.23 15 0 0 False Confections
Table: Alphabetical_list_of_products (69 rows)
Data
Search
Alphabetical_list_of_products
Categories
Employees
Order_Details

Practical No. 2

Aim: - Perform the Extraction Transformation and Loading (ETL) process to

construct the database in the SQL server / Power BI.

A: Data Extraction:

The data extraction is first step of BTL, there are 2 Types of Data Extraction

1) Pull Extraction: « All the data from source systems or operational systems gets extracted to staging area. (Initial Load)
2) Partial Extraction: - Sometimes we get notification from the source system to update specific date. It is called as Delta load.

Source System Performance: The Extraction strategics should not affect source system

performance.

B: Data Transformation:

The data transformation is second step. After extracting the data there is big need to do

the transformation as per the target system. Some bullet points of Data Transformation,

- Data Extracted from source system is in to raw format, we need to transform it before loading in to target server.
- Data has to be cleaned, mapped and transformed
- There are following important steps of Data Transformation: -
    - Selection: Select data to load i in target
    - Matching: Match the data with target system
    - Data Transforming: We need to change data as per target table structures

Real life examples of Data Transformation: -

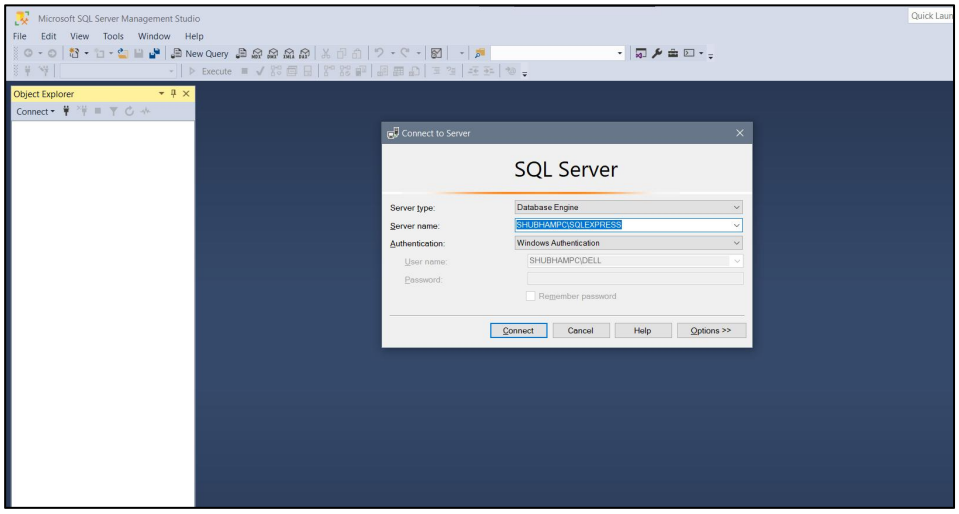
- Standardizing data: - Data is fetched from multiple sources so it needs to be standardized as per the target system.
- Character set conversion: - Need to transform the character sets as per the target systems. (First name and last name example)
- Calculated and derived values: In source system there is first value and second value and in target we need the calculation of first value and second value.
- Data Conversion in different formats: - If in source system date in in DDMMYY format and in target the date is in DDMONYYYY format then this transformation needs to be done at transformation phase.

C: Data Loading:

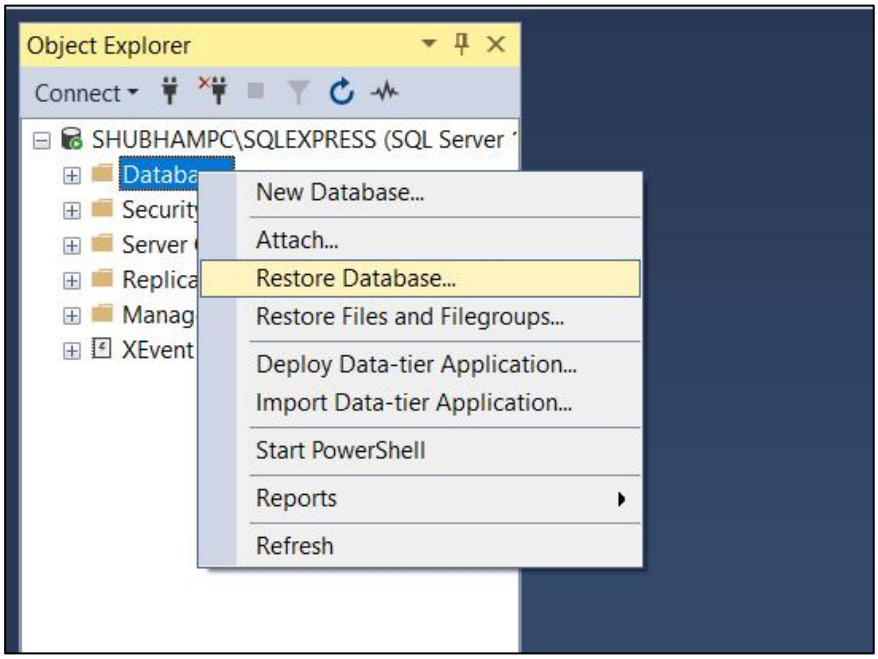
Data loading phase loads the prepared data from staging tables to main tables.

ETL process in SQL Server:

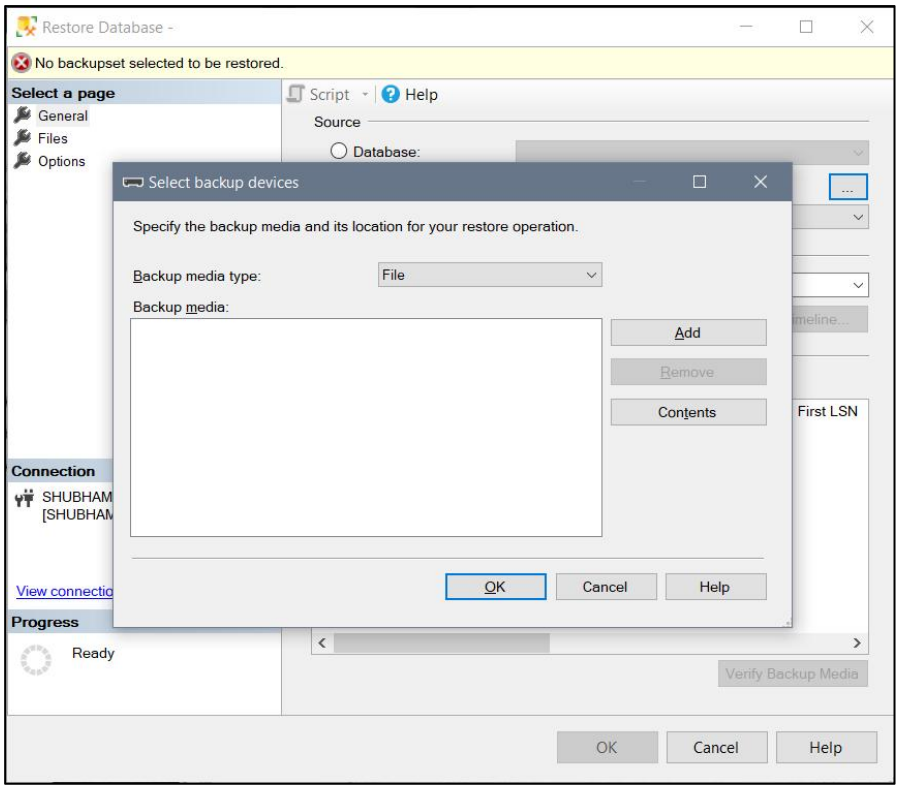
Step 1: Open SQL Server Management Studio to restore backup file



Step 2: Right click on Databases → Restore Database

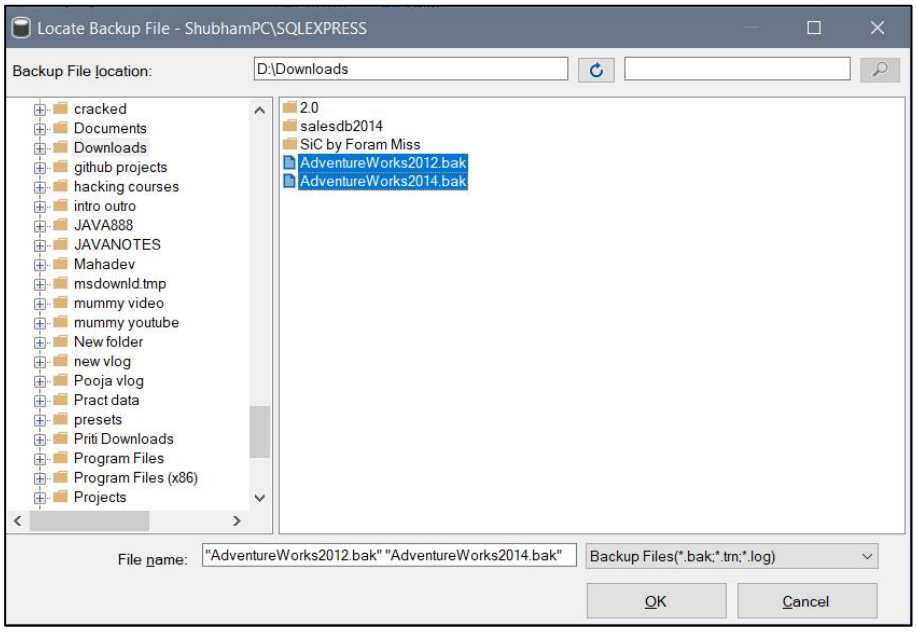


Step 3: Select Device → click on icon towards end of device box

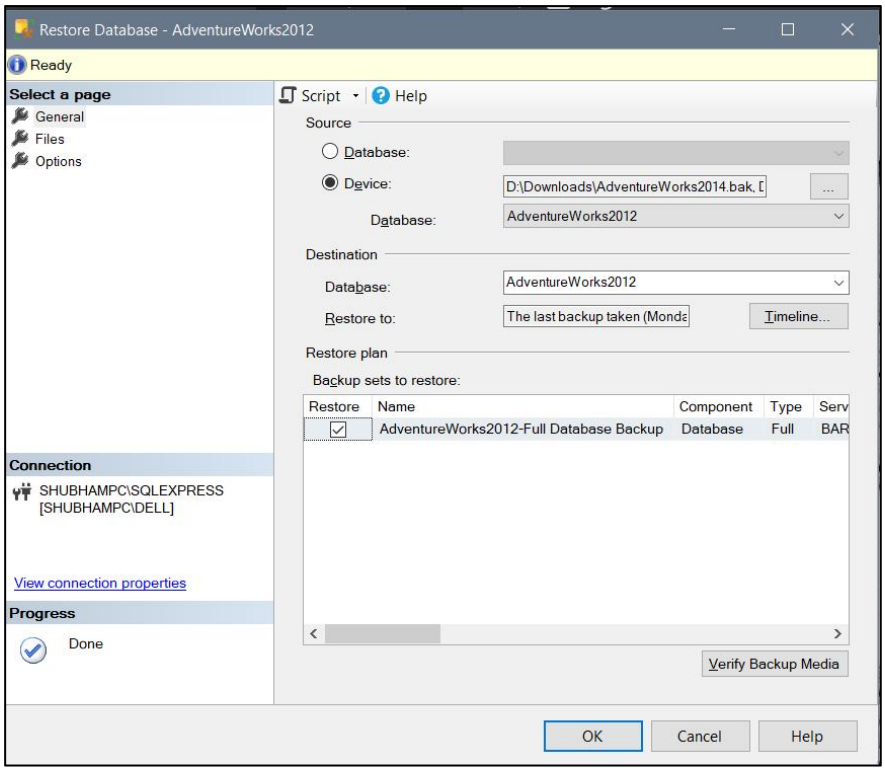


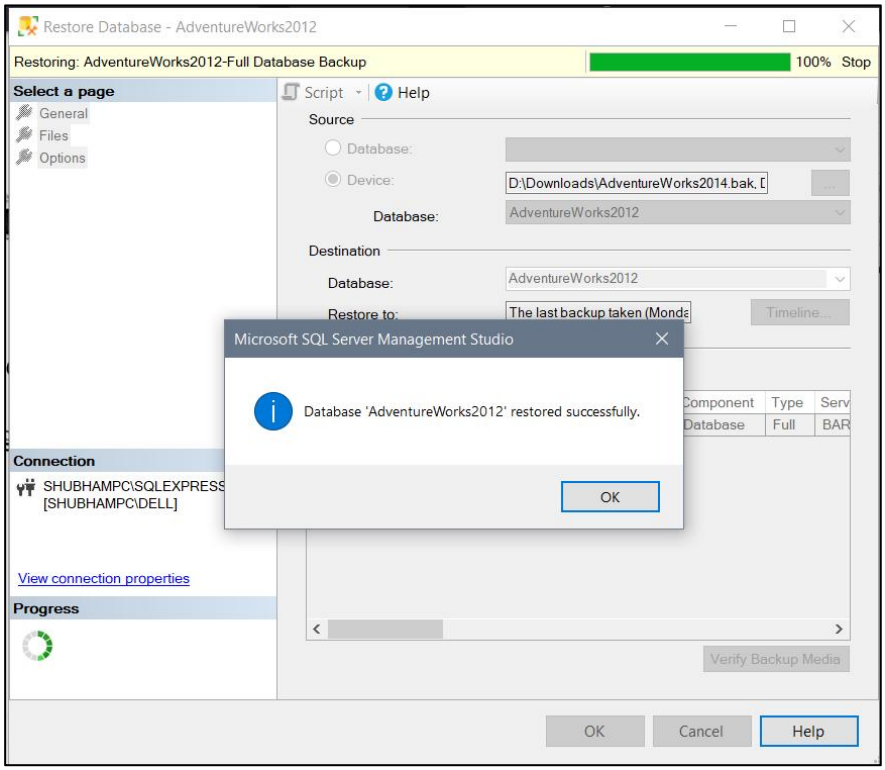
Step 4: Click on add → Select path of backup files

Step 5: Select both files at a time.



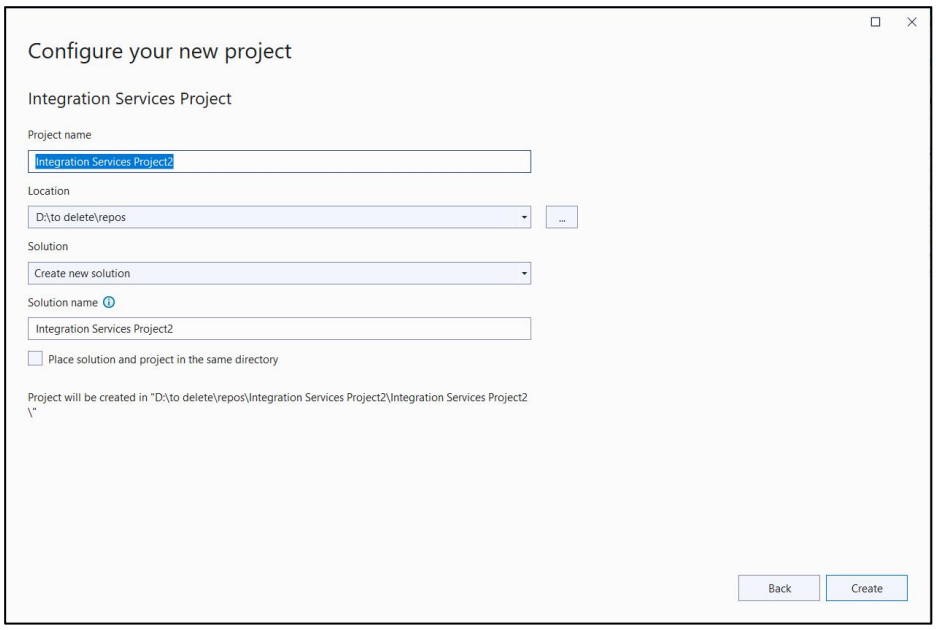
Step 6: Click ok and in select backup devices window Add both files of Adventure Works.



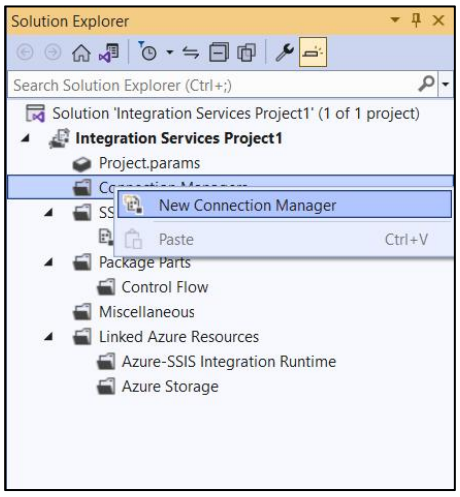


Step 7: Open SQL Server Data Tools

Select File new Project → Business Intelligence → Integration Services - Project & give appropriate project name.

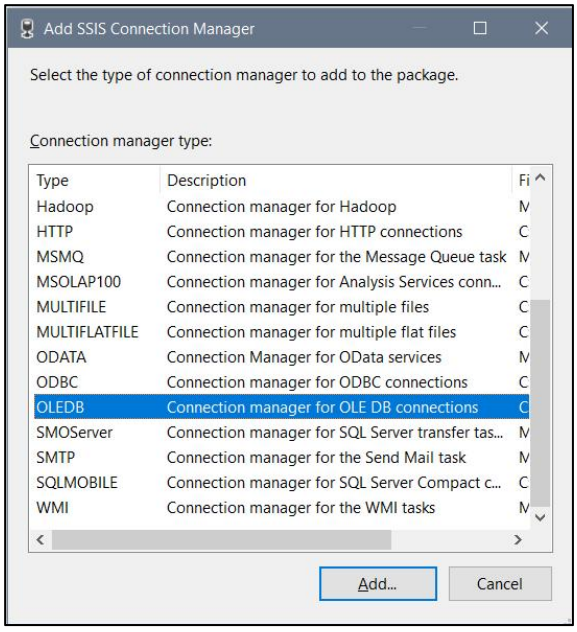


Step 8: Right click on Connection Managers in solution explorer and click on New Connection Manager.



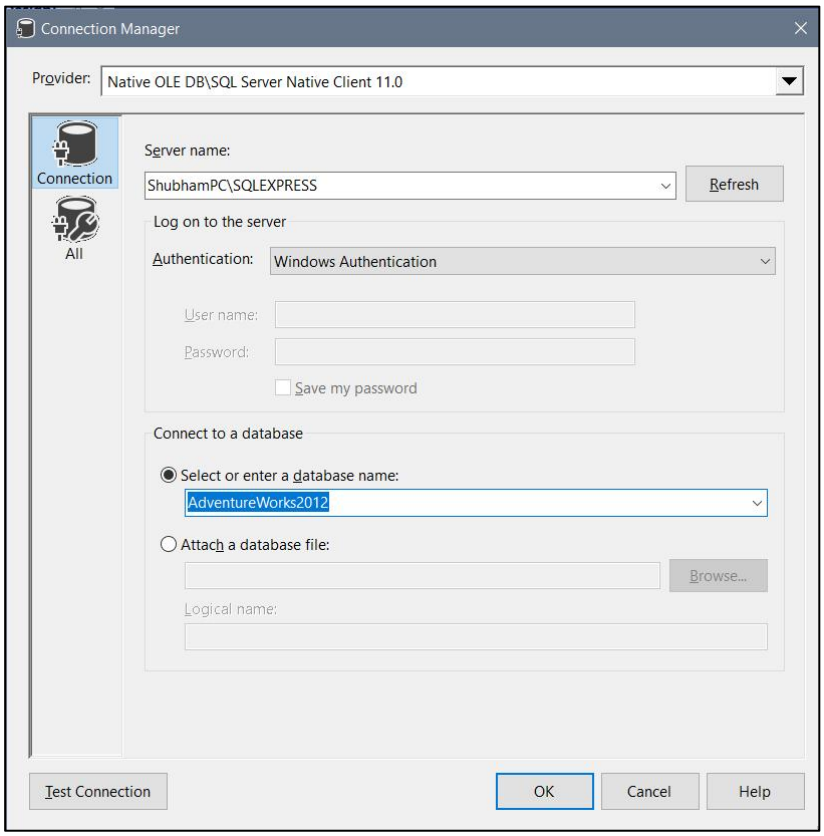
Add SSIS connection manager window appears.

Step 9: Select OLEDB Connection Manager and click on Add.



Step 10: Configure OLE DB Connection Manager Window appears → Click on New

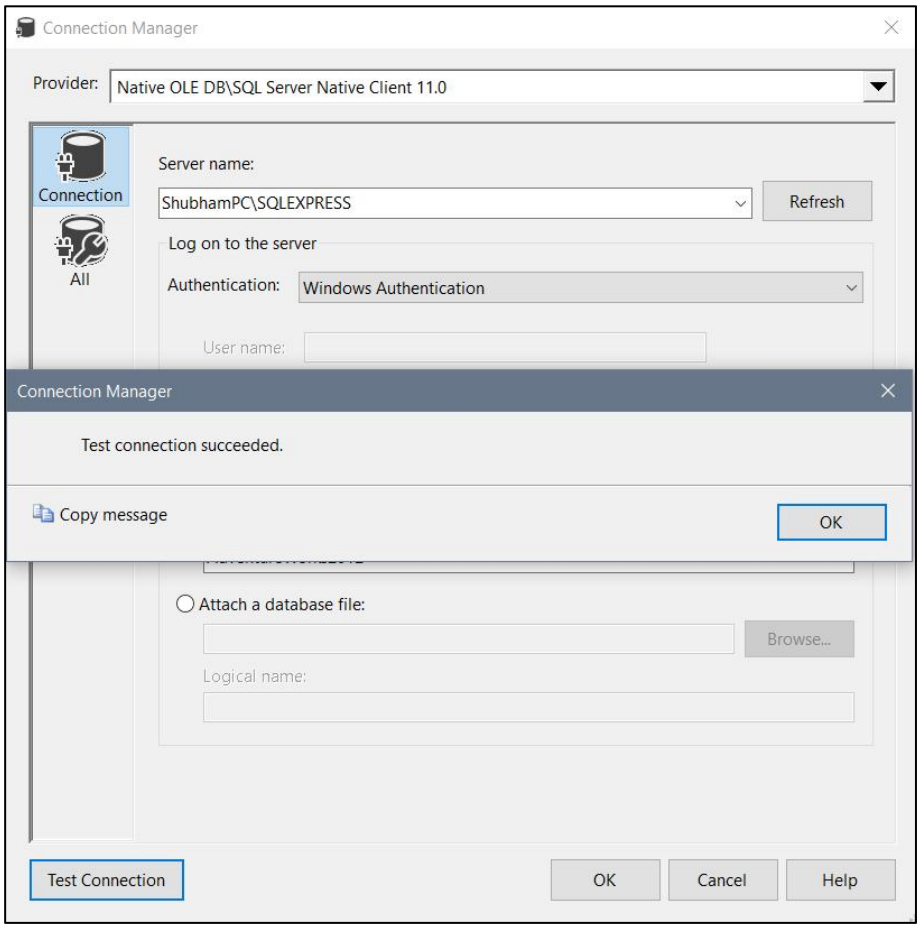
Step 11: Select Server name (as per your machine) from drop down and database name and click on Test connection.



Step 12: If test connection succeeded

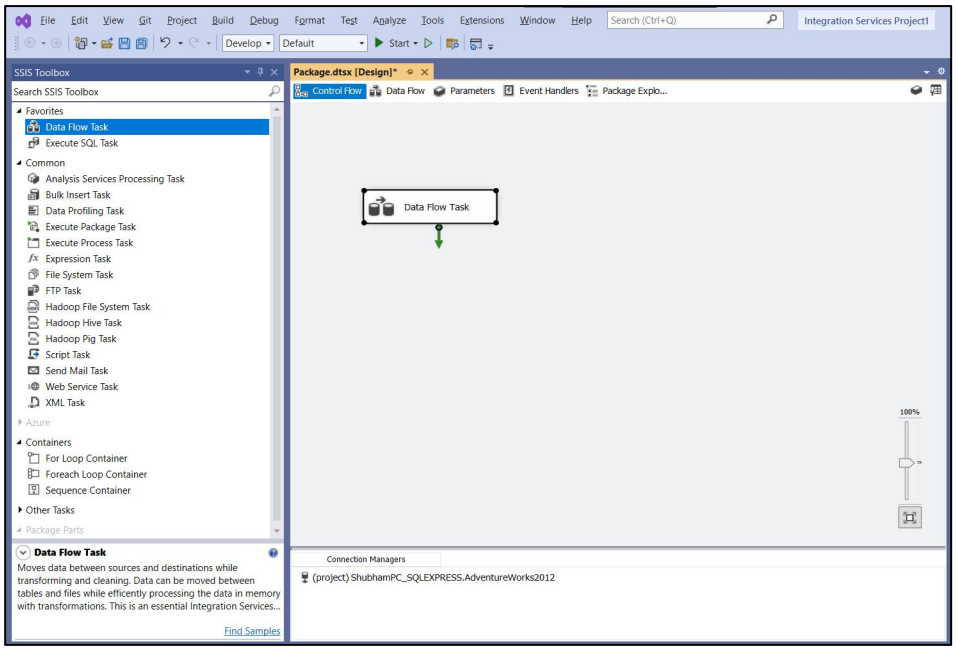
click on OK.

Step 13: Click on OK

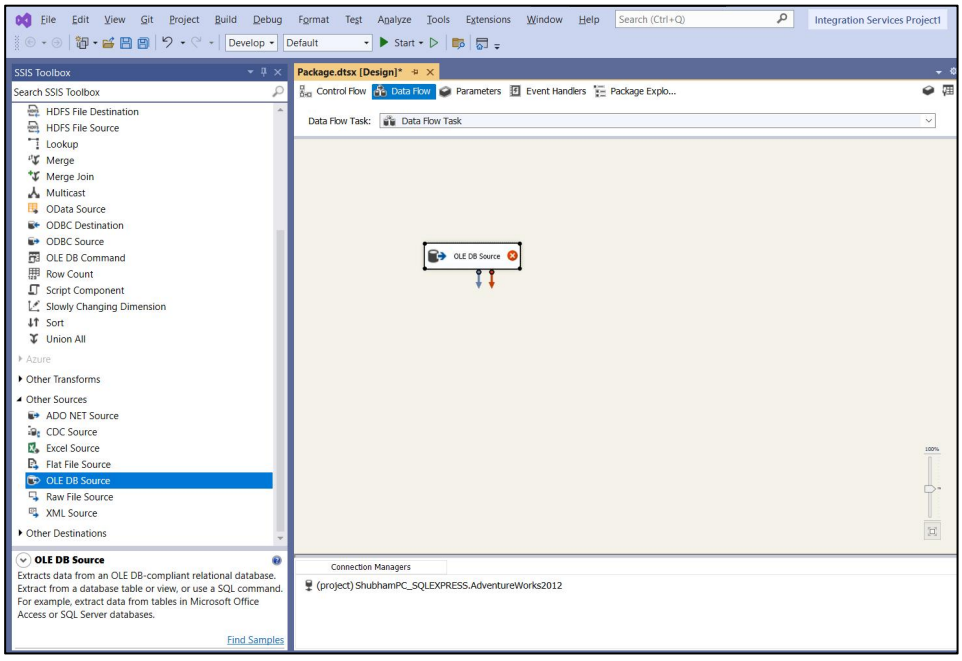


Connection is added to connection manager.

Step 14: Drag and drop Data Flow Task in Control Flow tab.

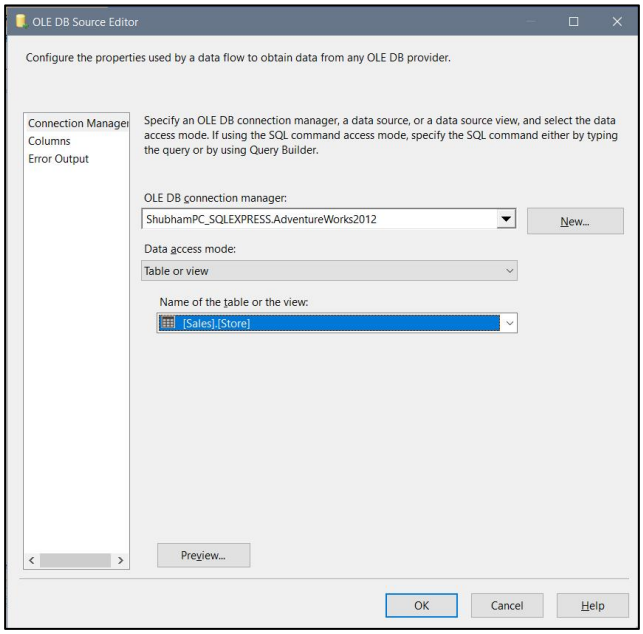


Step 15: Drag OLE DB Source from other sources and drop into data flow tab.

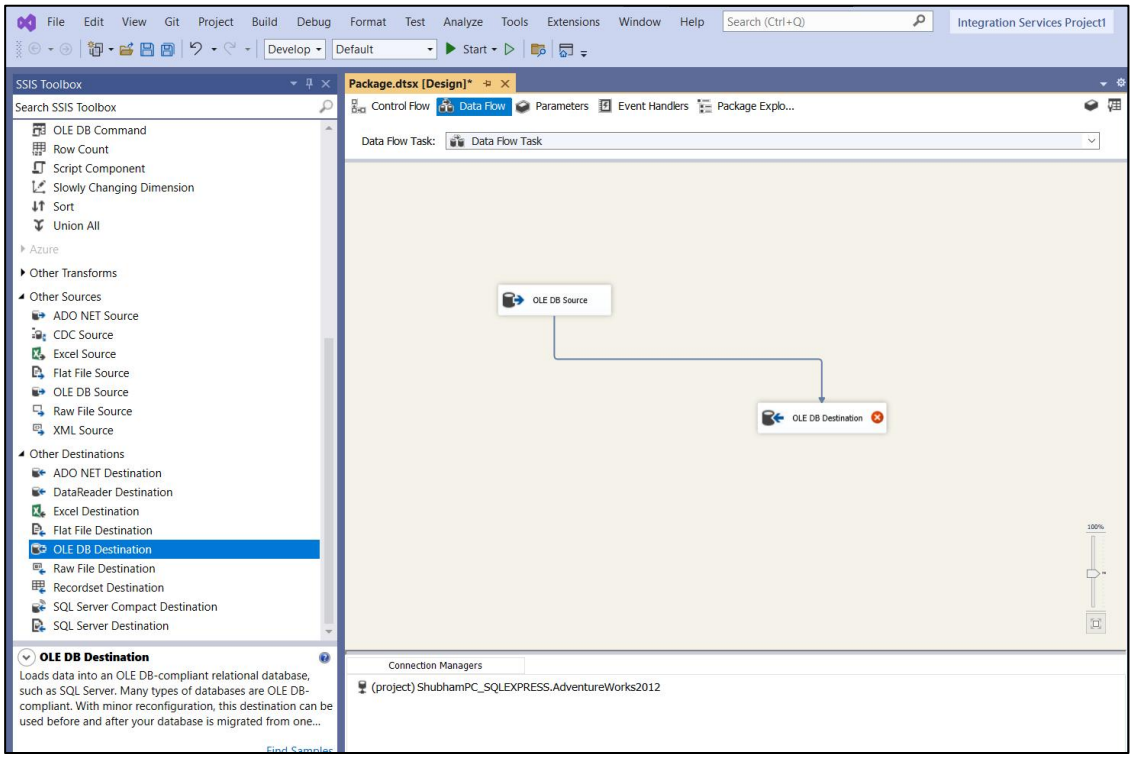


Step 16: Double click on OLE DB source → OLE DB Source Editor appears → click on New to add connection manager.

Select [Sales].[Store] table from drop down → ok.

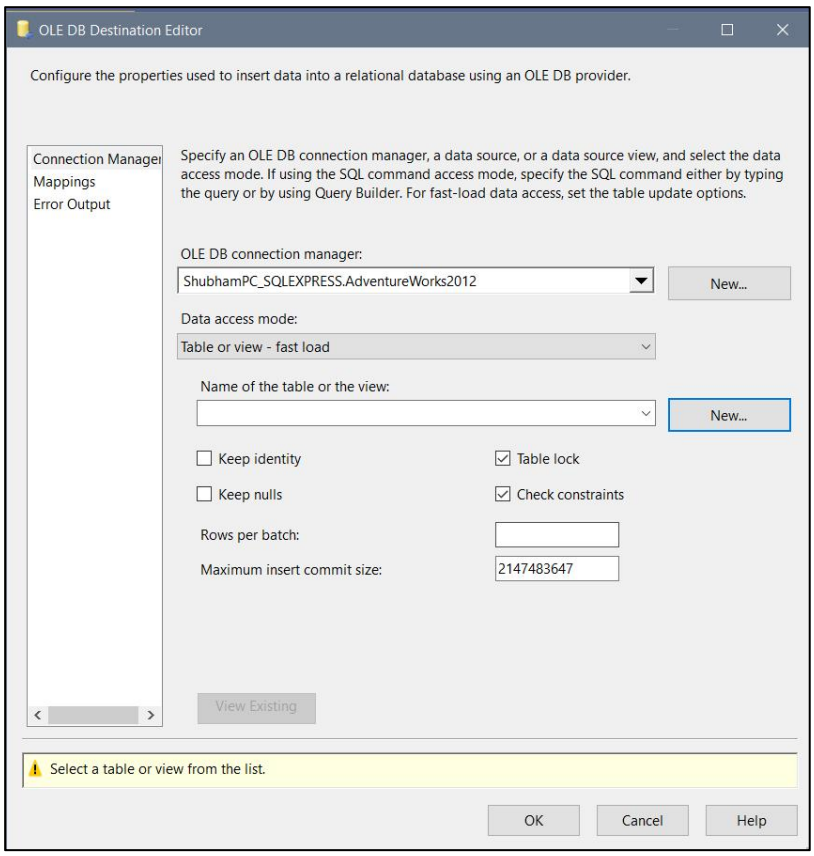


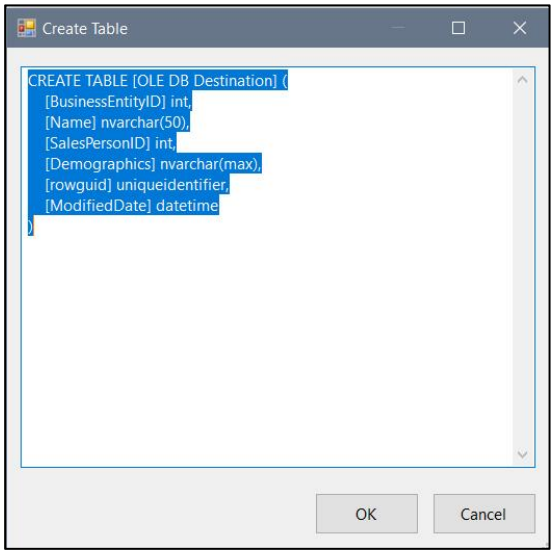
Step 17: Drag OLEDB destination in data flow tab and connect both.

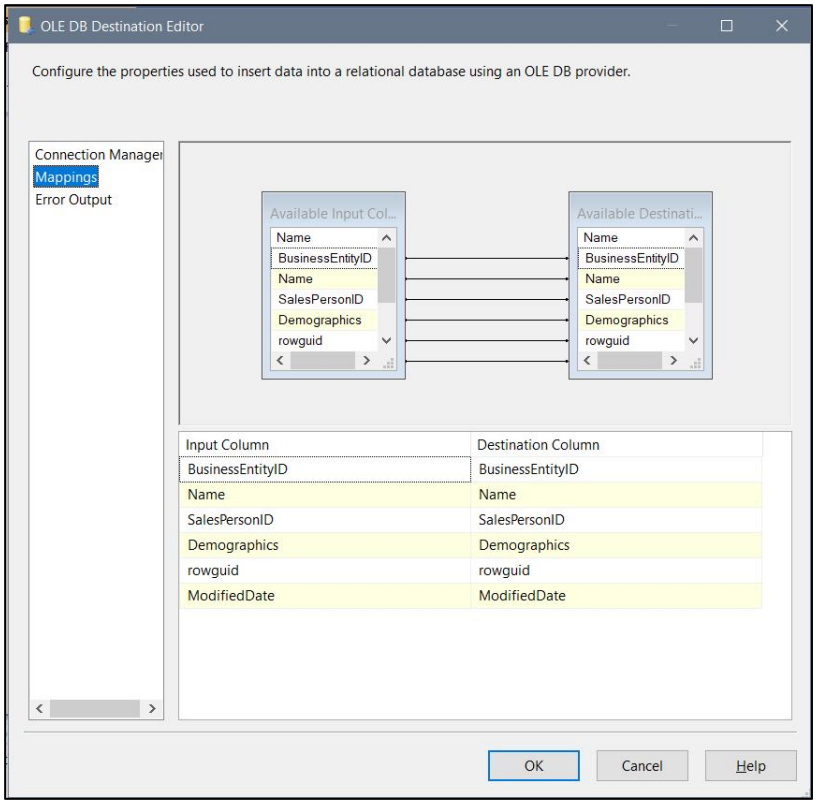


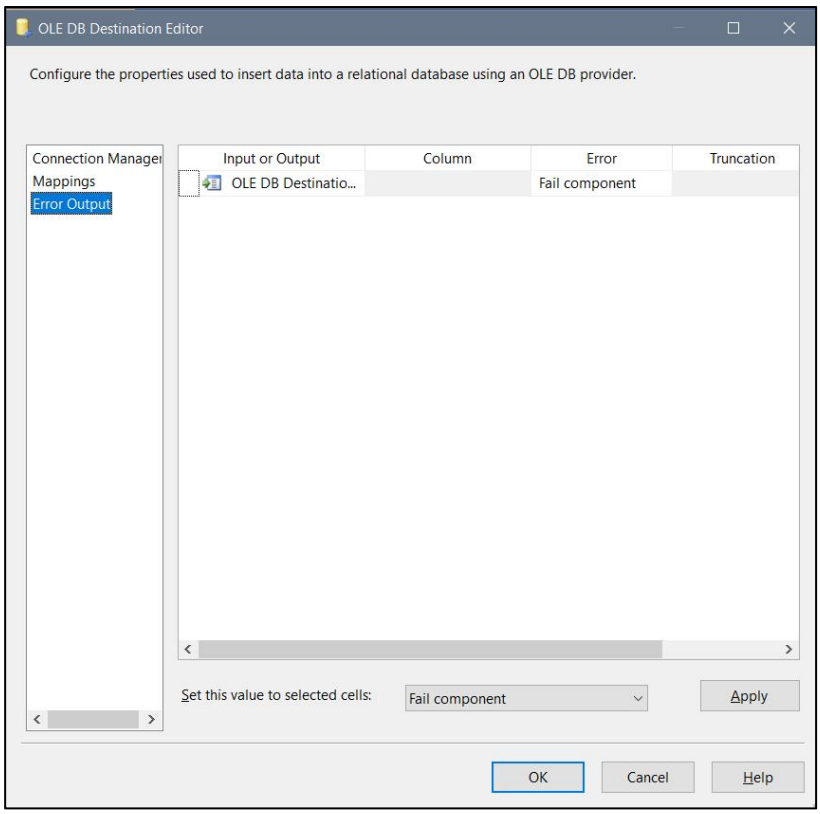
Step 18: Double click on OLE DB destination.

Step 19: Click on New to run the query to get [OLE DB Destination] in Name of the table or the view.



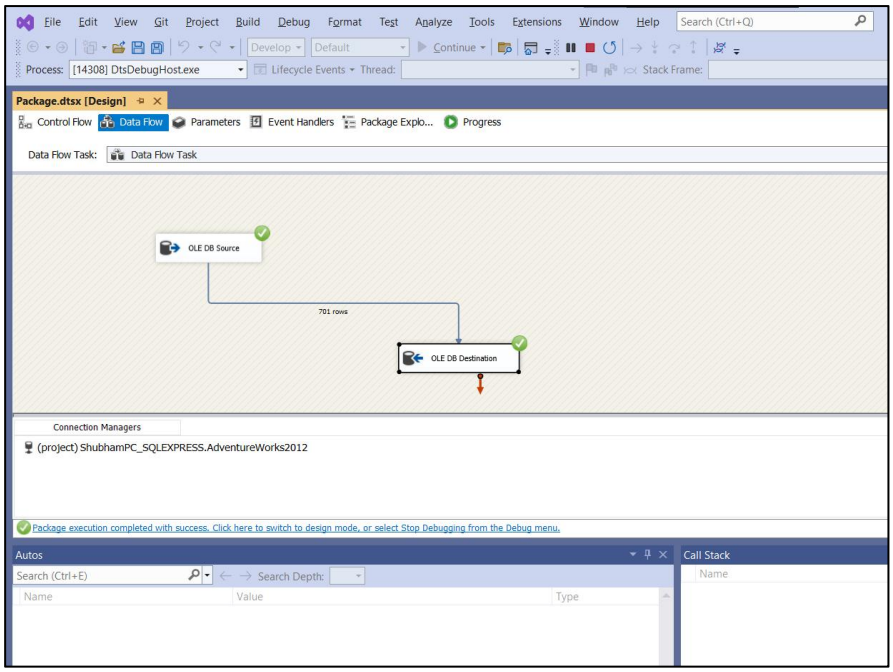



OLE DB Destination Editor
Configure the properties used to insert data into a relational database using an OLE DB provider.
Connection Manager
Mappings
Error Output
Available Input Col...
Name
BusinessEntityID
Name
SalesPersonID
Demographics
rowguid
Available Destinati...
Name
BusinessEntityID
Name
SalesPersonID
Demographics
rowguid
Input Column
Destination Column
BusinessEntityID
BusinessEntityID
Name
Name
SalesPersonID
SalesPersonID
Demographics
Demographics
rowguid
rowguid
ModifiedDate
ModifiedDate
OK
Cancel
Help


OLE DB Destination Editor
Configure the properties used to insert data into a relational database using an OLE DB provider.
Connection Manager
Mappings
Error Output
Input or Output
Column
Error
Truncation
OLE DB Destinatio...
Fail component
Set this value to selected cells:
Fail component
Apply
OK
Cancel
Help

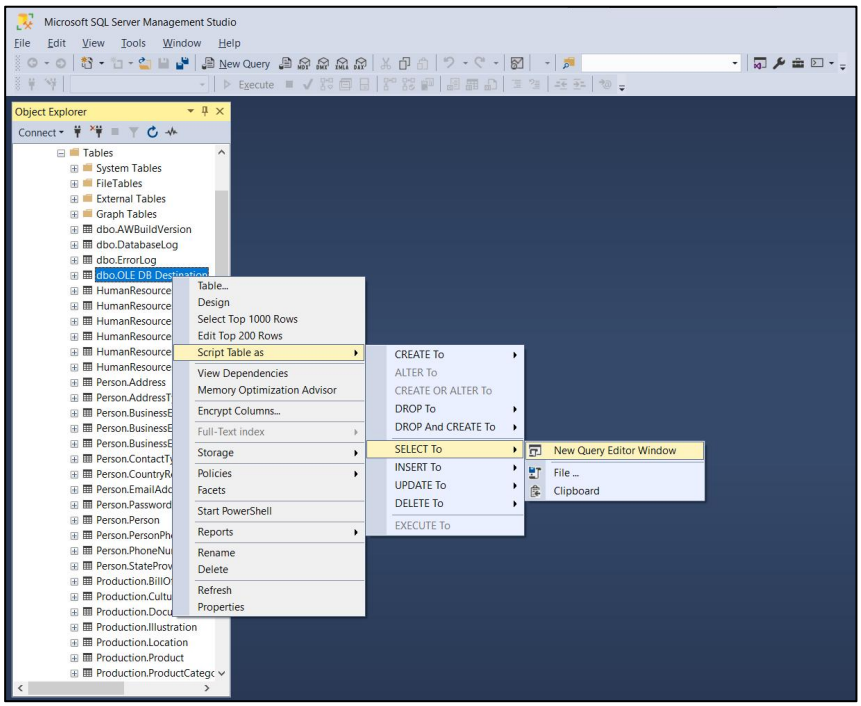
Click On OK.

Step 20: Click on Start.



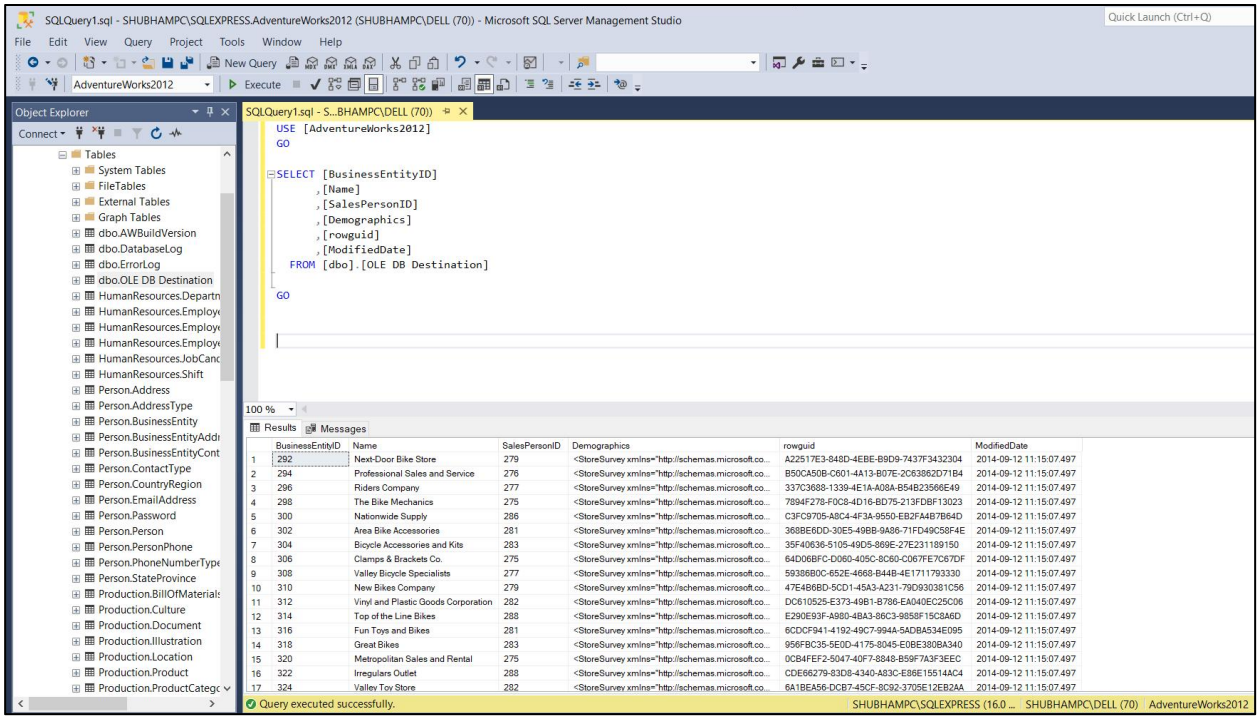
Step 21: Go to SQL Server Management Studio.

In database tab→ Adventure works→ Right click on [dbo].[OLE DB Destination] Scrip Table as→ SELECT To New Query Editor Window.



Step 22: Execute following query to get output.

```
USE [AdventureWorks2012]
GO
SELECT[BusinessEntityID]
,[Name]
,[SalesPersonID]
,[Demographics]
,[rowguid]
,[ModifiedDate]
FROM [dbo].[OLE DB Destination]
GO
```



Practical No. 3

Aim: Create the cube with suitable dimension and fact tables based on OLAP.

Step 1: Creating Data Warehouse

Let us execute our T-SQL Script to create data warehouse with fact tables, dimensions and populate them with appropriate test values.

Download T-SQL script attached with this article for creation of Sales Data Warehouse or download from this article "Create First Data Warehouse" and run in your SQL Server. Downloading "Data Warehouse SQLScript.zip" from the article.

https://www.codeproject.com/Articles/652108/Create-First-Data-WareHouse

After downloading extract file in folder.

Follow the given steps to run the query in SSMS (SQL Server Management Studio).

Open SQL Server Management Studio 2012

Connect Database Engine

Password for sa: admin123 (as given during installation)

Click Connect.

Open New Query editor

4. Copy paste Scripts given below in various steps in new query editor window one by one

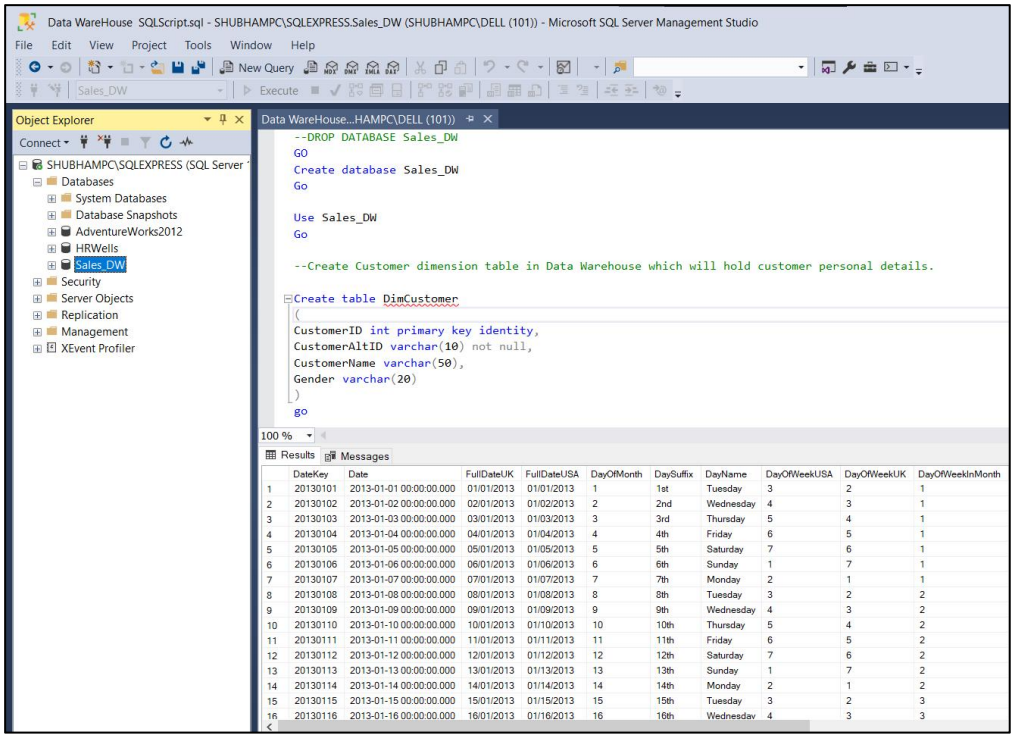
5. To run the given SQL Script, press F5

6. It will create and populate "Sales DW" database on your SQL Server

OR

1. Go to the extracted SQL file and double click on it.

2. New SQL Query Editor will be opened containing Sales DW Database.

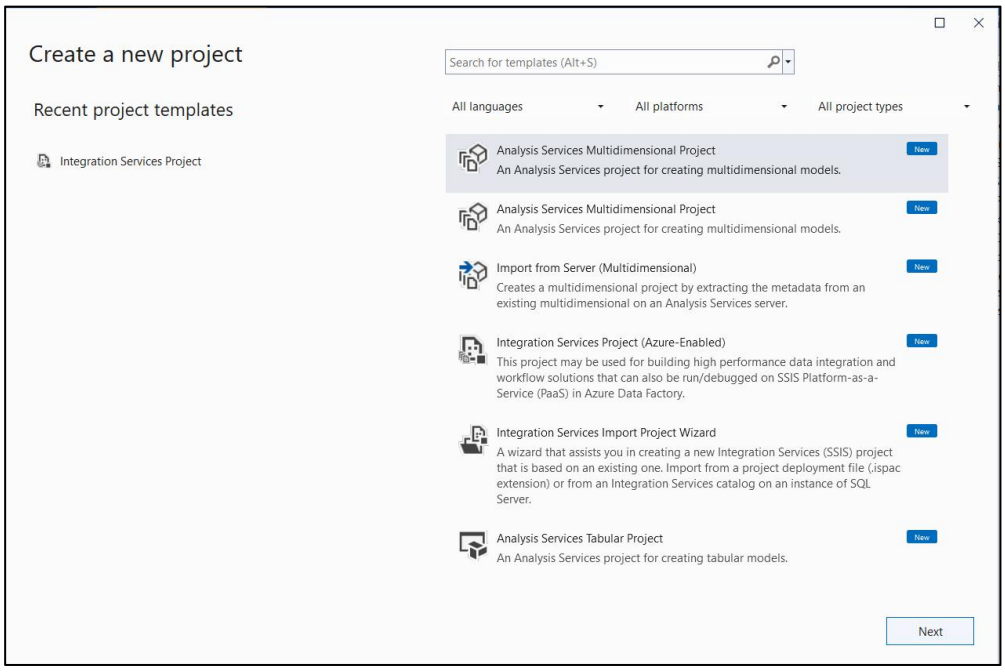


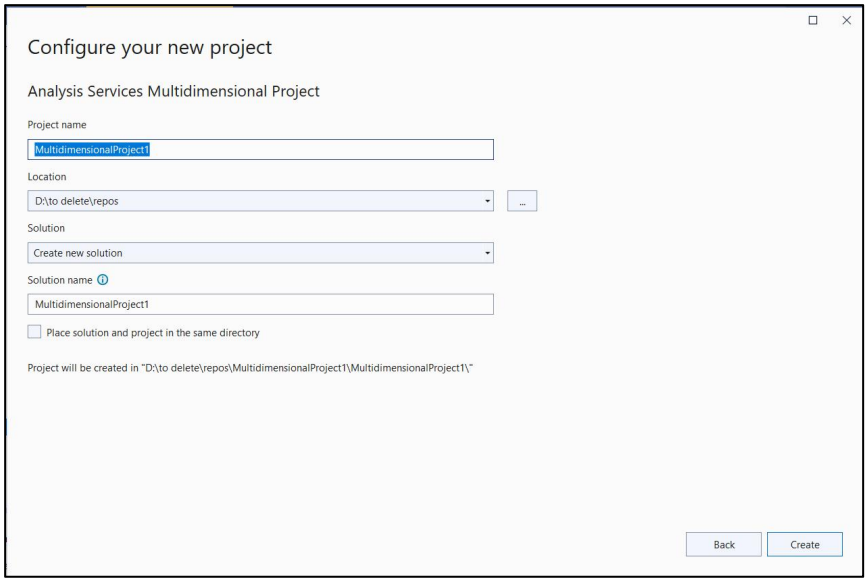
Step 2: Start SSDT environment and create New Data Source.

Go to SQL Server Data Tools → Right click and run as administrator

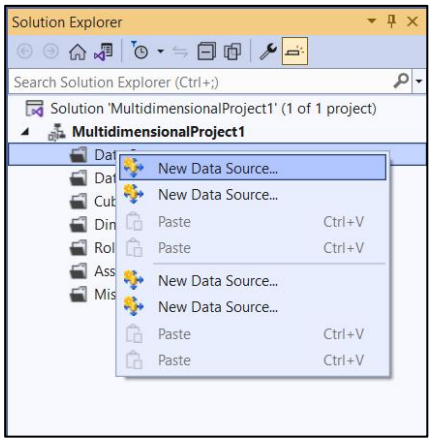
Click on File → New Project

In Business Intelligence → Analysis Services Multidimensional Project → appropriate project name → click OK.



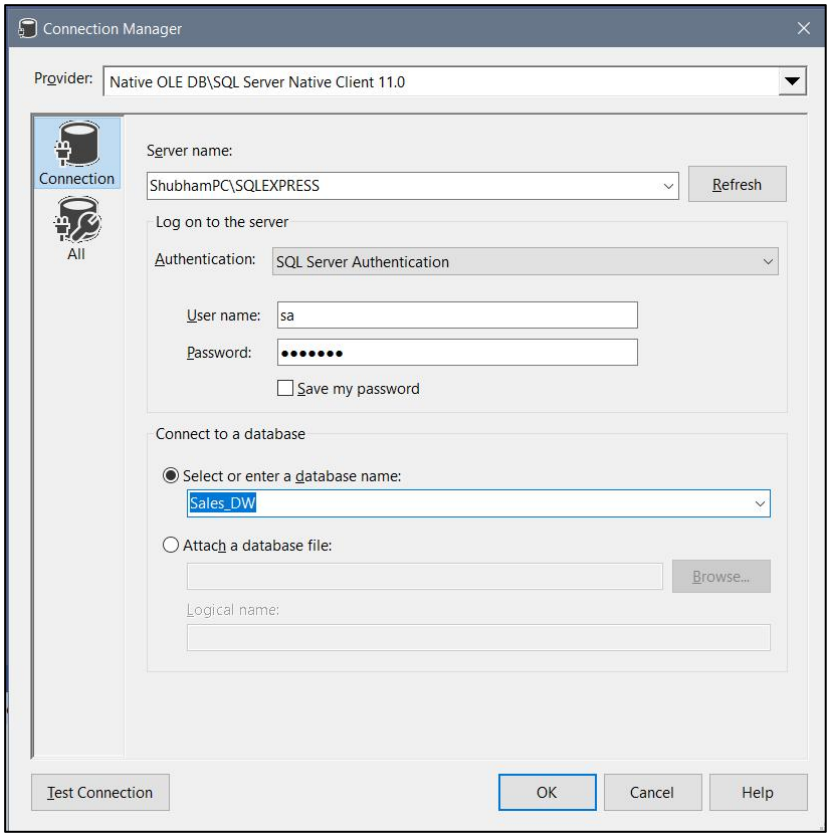


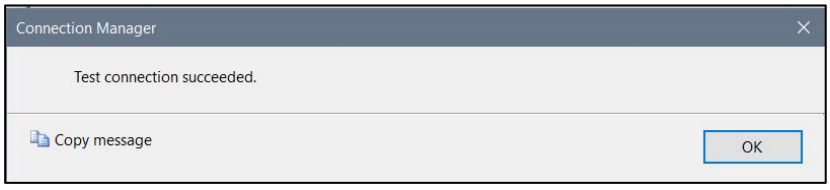
Right click on Data Sources in solution explorer → New Data Source.



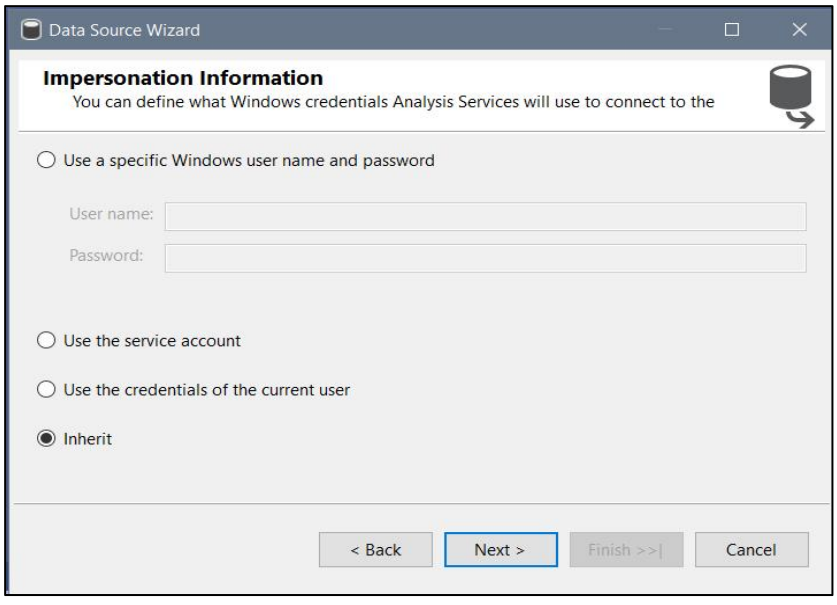
Select Server → Name select Use SQL Server Authentication → Select or enter a database name (Sales DW).

Click Next

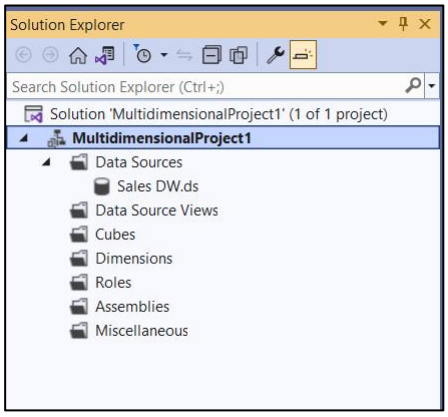




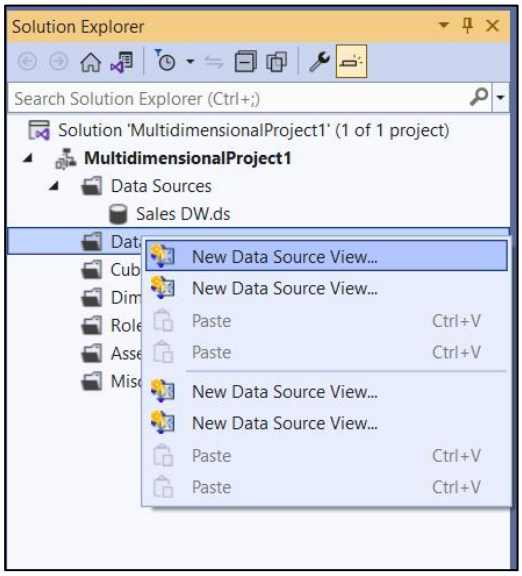
Select Inherit → Next



Sales_DW.ds gets created under Data Sources in Solution Explorer

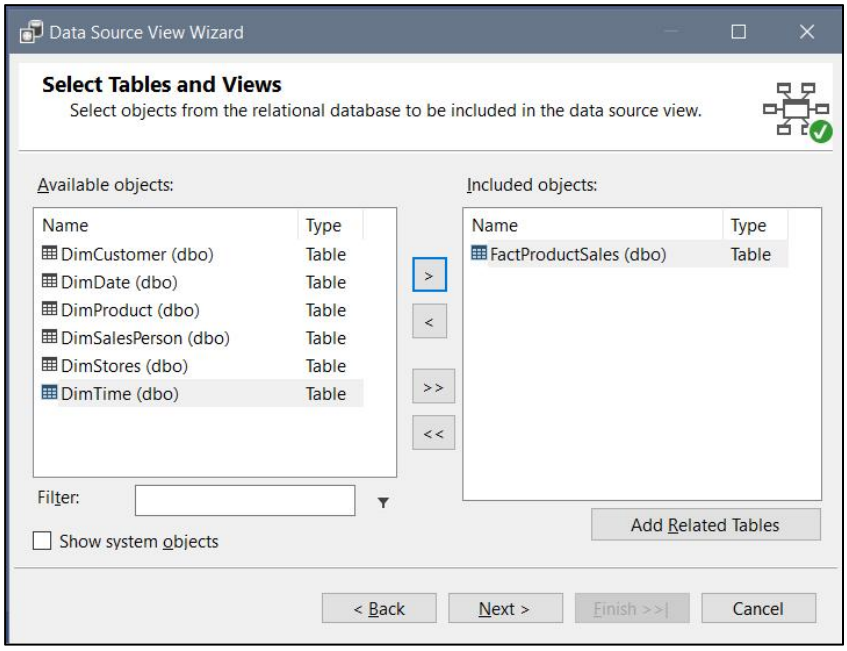


Step 3: Creating New Data Source View in Solution explorer right click on Data Source View Select New Data Source View

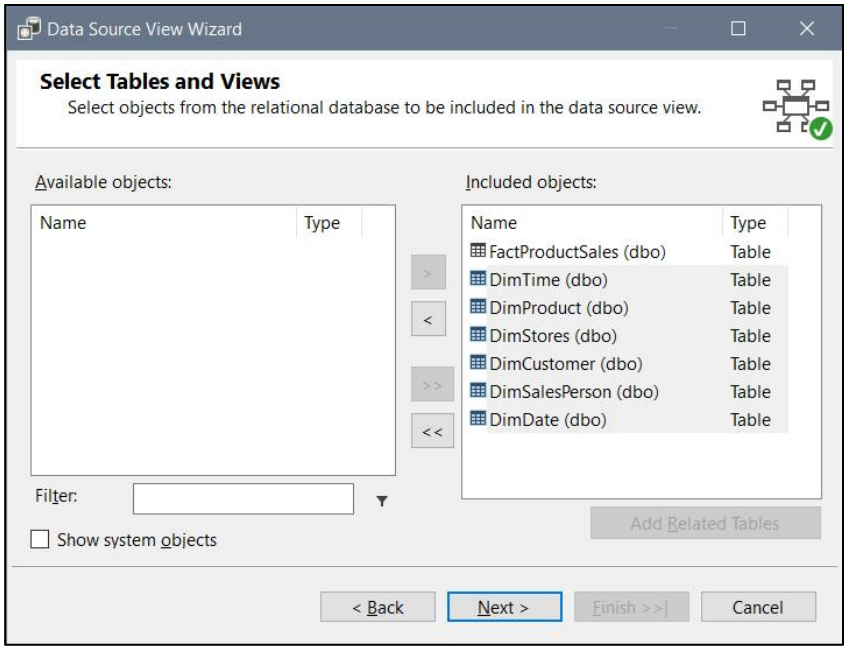


Click Next

Select FactProductSales(dbo) from Available objects and put in Includes Objects by clicking on

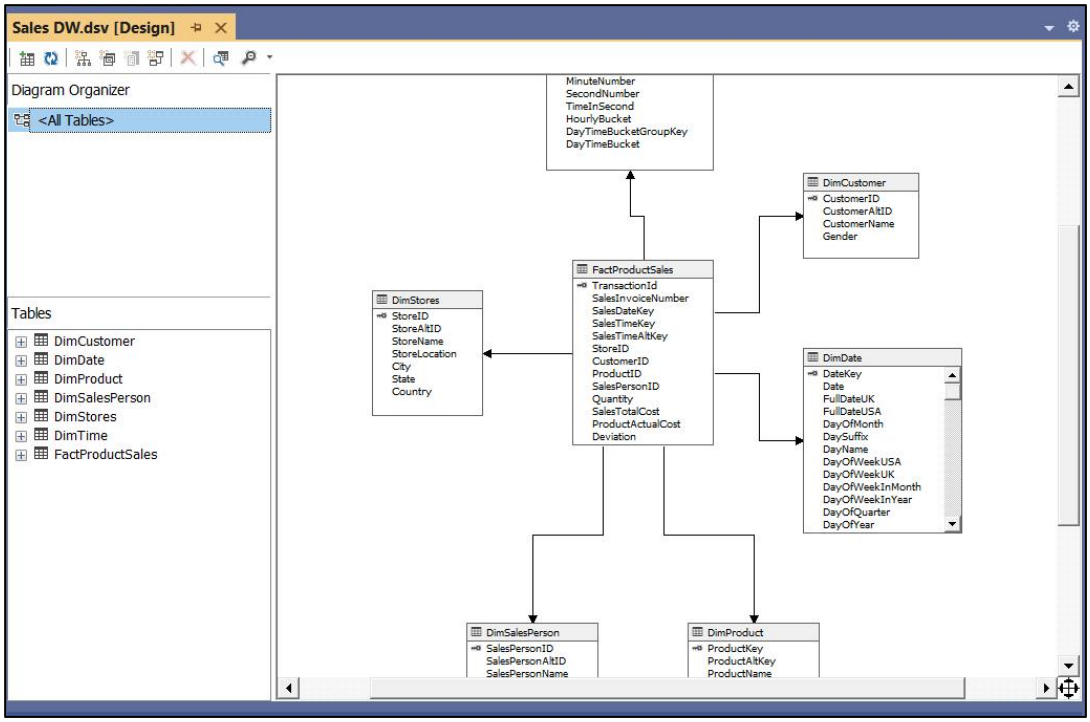


Click on Add Related Tables



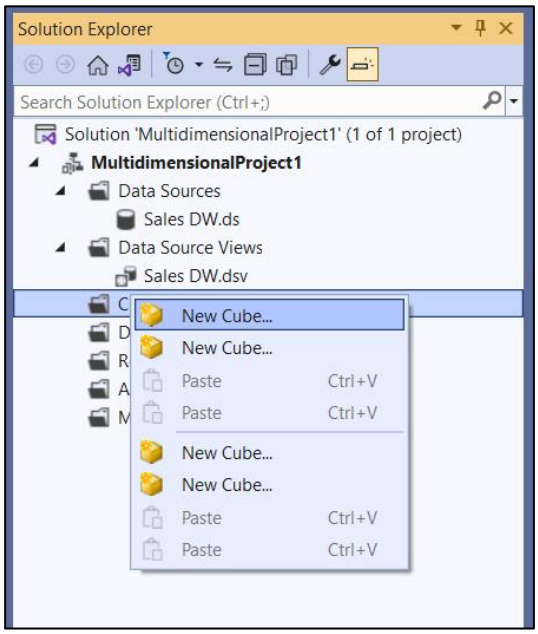
Click Finish

Sales DW.dsv appears in Data Source Views in Solution Explorer.



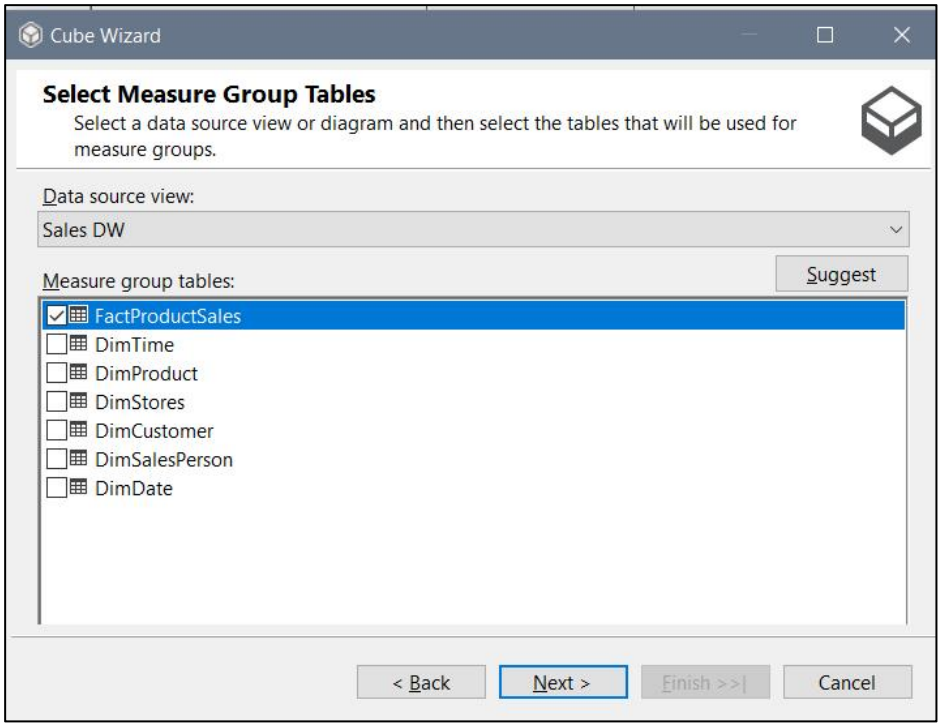
Step 4: Creating new cube

Right click on cubes → New Cube

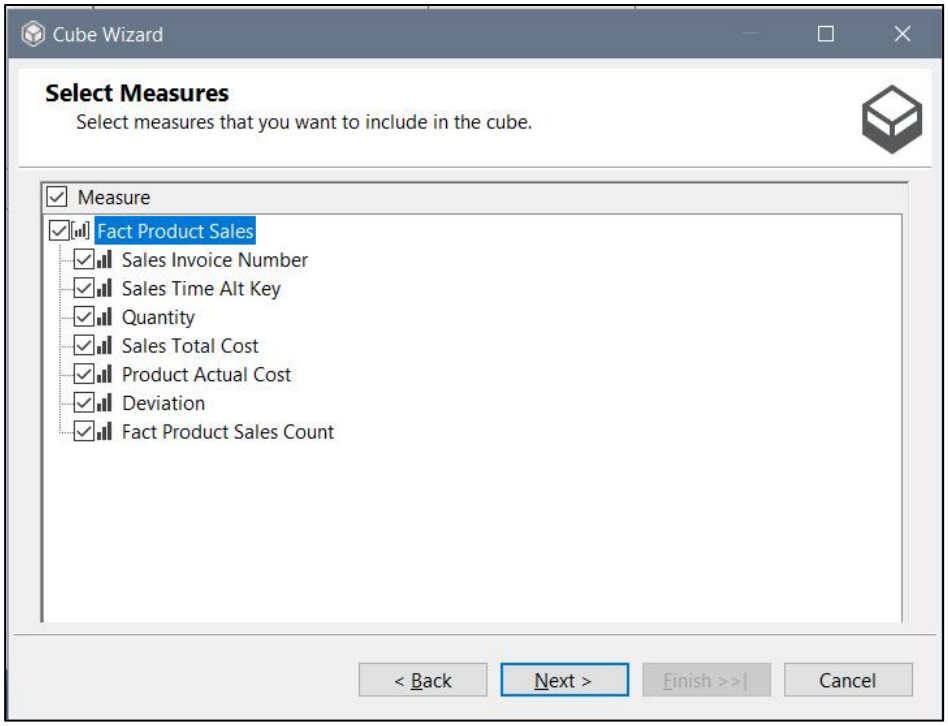


Select Use existing tables in Select Creation Method → Next

In Select Measure Group Tables → Select FactProductSales → Click Next



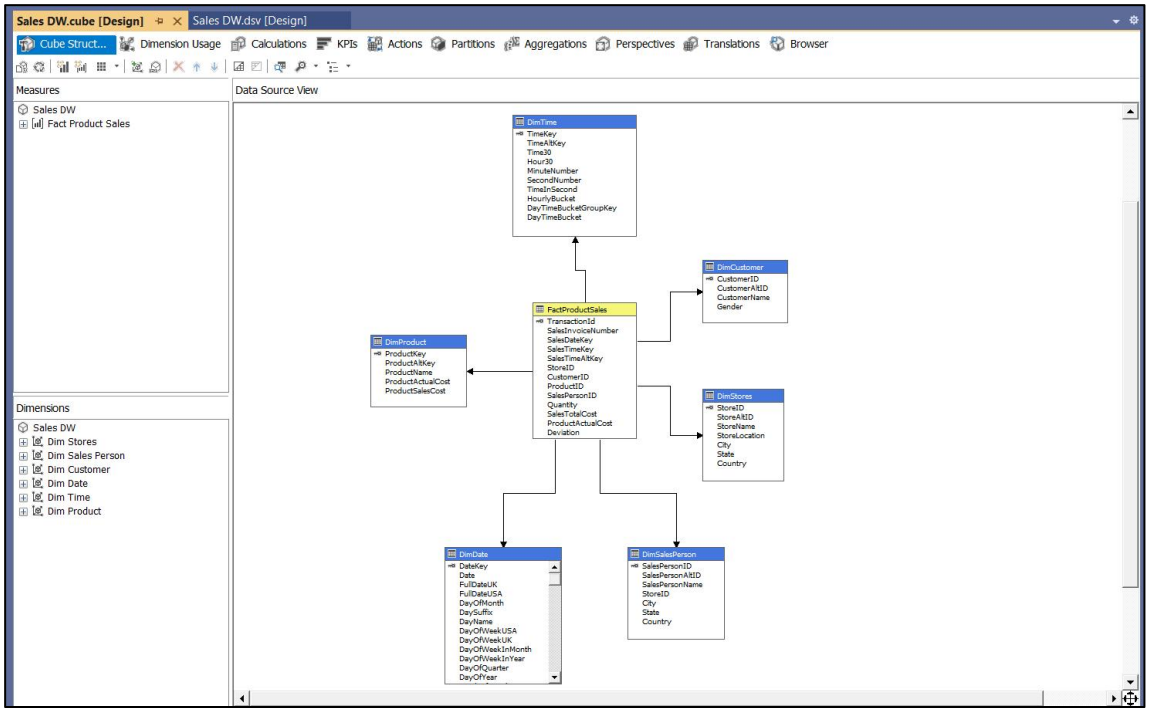
In Select Measures → check all measures → Next



In Select New Dimensions → Check all Dimensions→ Next.

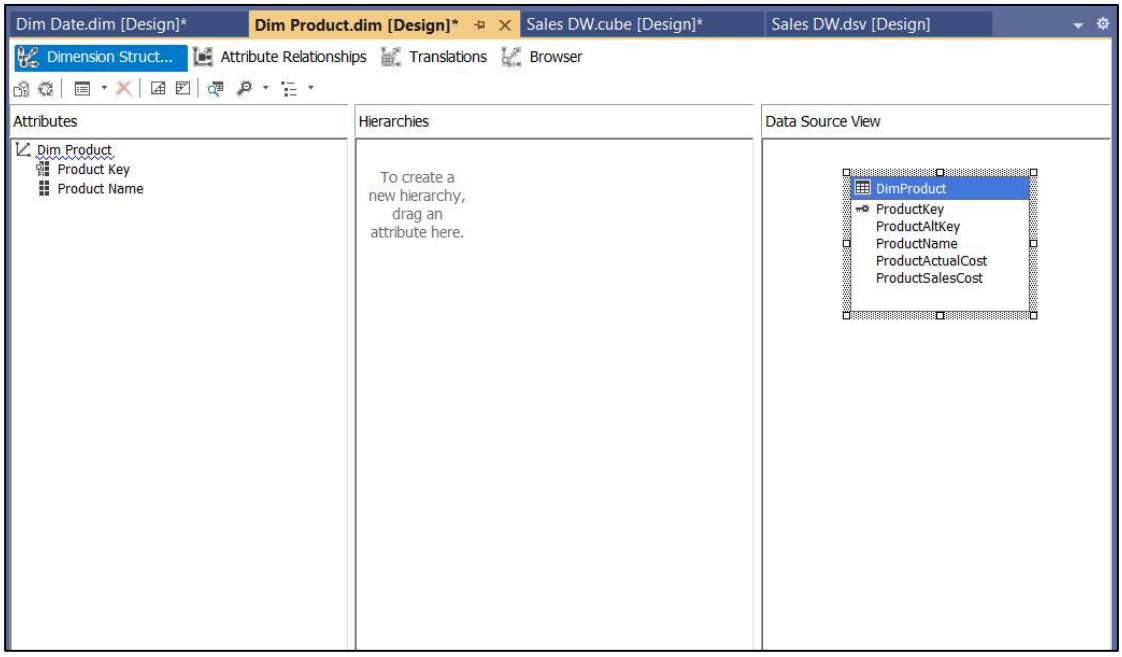
Click on Finish.

Sales_DW.cube is created.



Step 5: Dimension Modification In dimension tab → Double Click Dim Product.dim

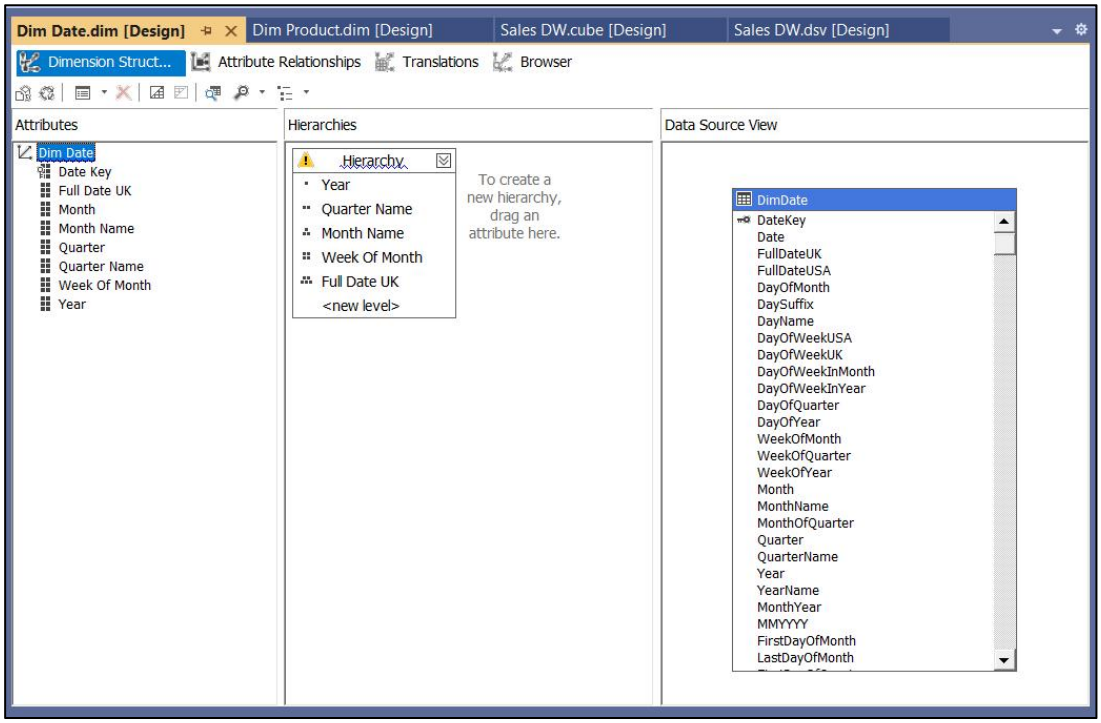
Step 6: Drag and Drop Product Name from Table in Data Source View and Add in Attribute Pane at left side.



Step 7: Creating Attribute Hierarchy in Date Dimension.

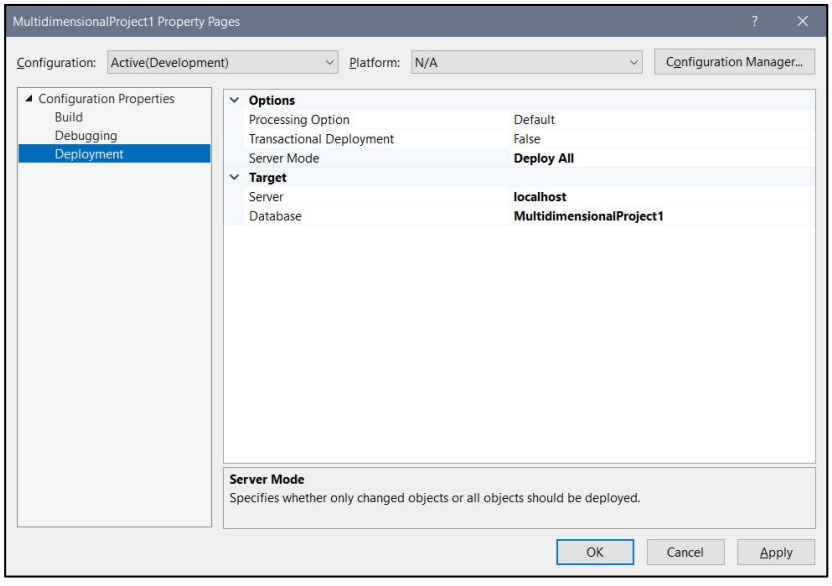
Double click on Dim Date dimension → Drag and Drop Fields from Table shown in Data Source View to Attributes → Drag and Drop attributes from leftmost pane of attributes to middle pane of Hierarchy.

Drag fields in sequence from Attributes to Hierarchy window (Year, Quarter Name, Month, Name, Week of the Month, Full Date UK).

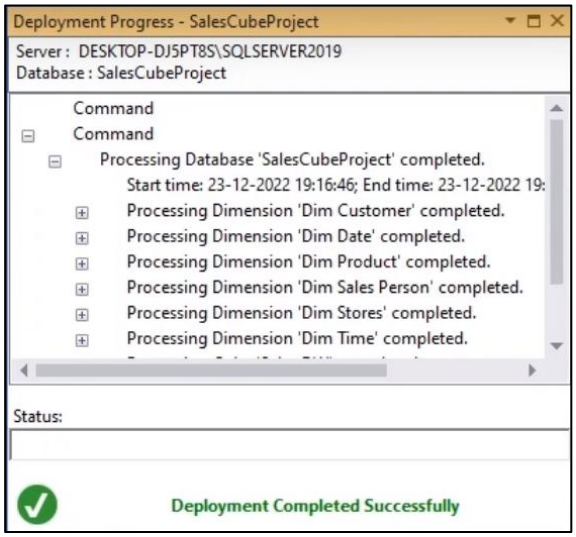


Step 8: Deploy Cube Right click on Project name → Properties

Do following changes and click on Apply & OK.



Right click on project name → Deploy.



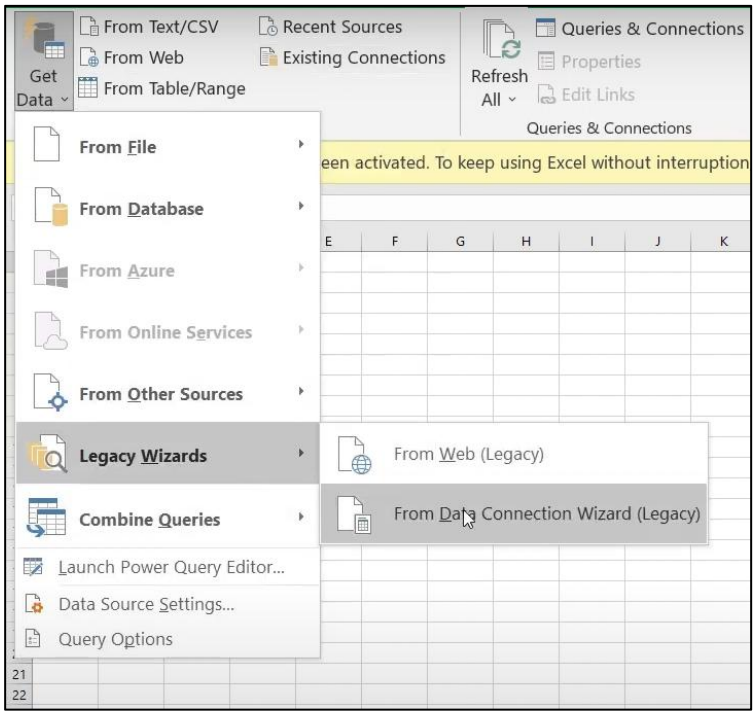
Practical No. 4

A) Aim: Import the data warehouse data in Microsoft Excel and create the Pivot table and Pivot Chart

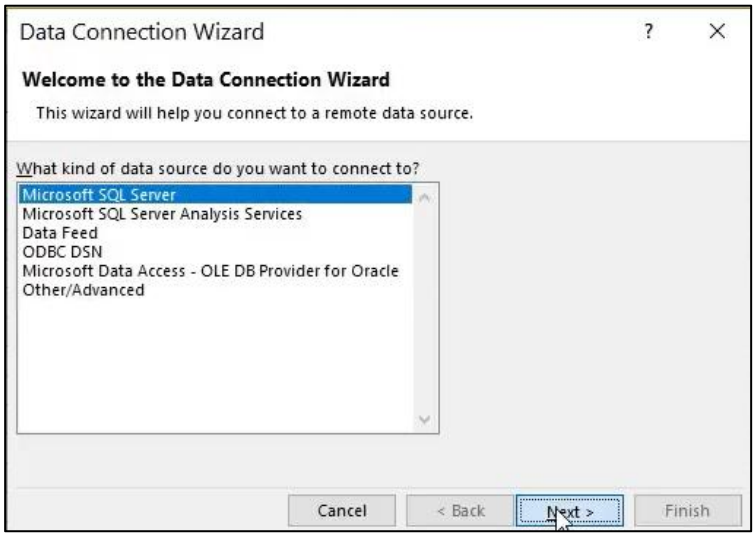
(MS Office Professional is used to make sure Power View is enabled for visualization.)

Step 1: Open Excel 2013 (Professional)

Go to Data tab→ Get External Data → From Other Sources →From Data Connection Wizard



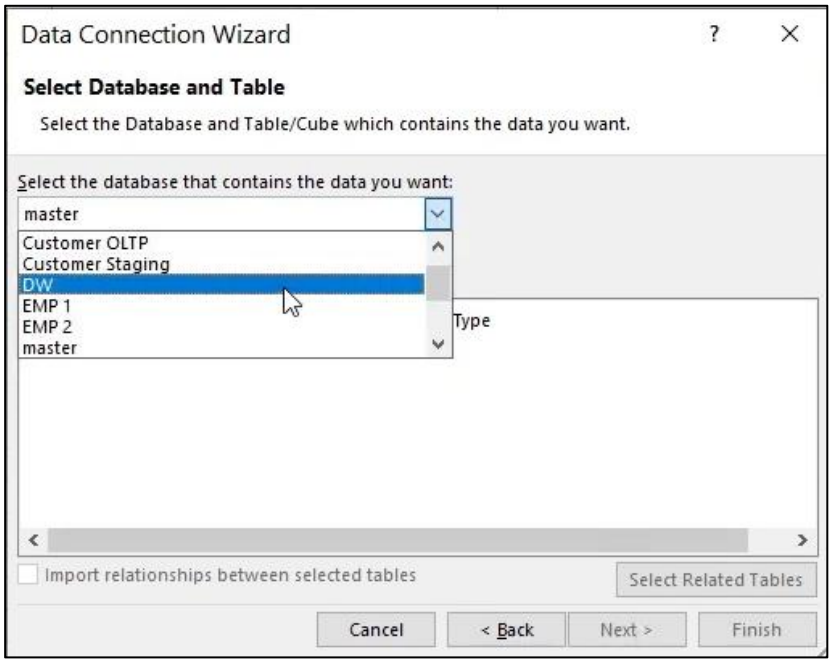
Step 2: In Data Connection Wizard → Select Microsoft SQL Server → Click on next



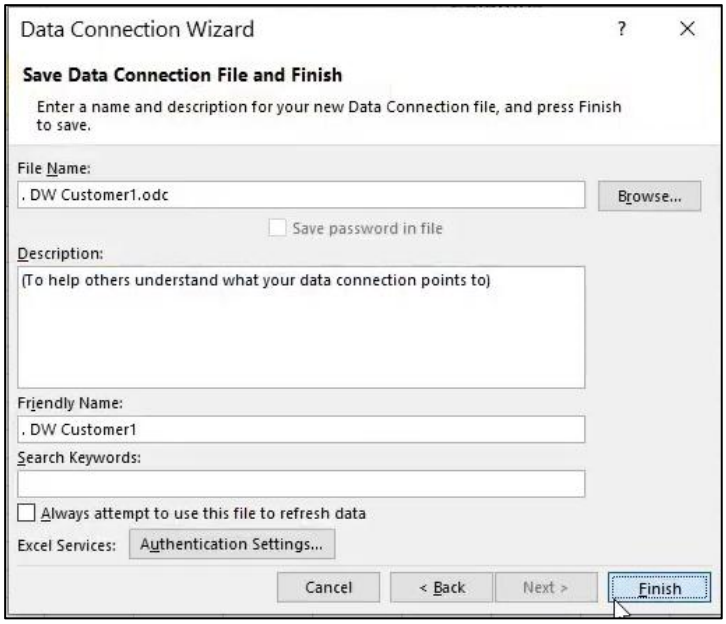
Step 3: In connect to Database Server provide Server name (Microsoft SQL Server Name)

Click on OK

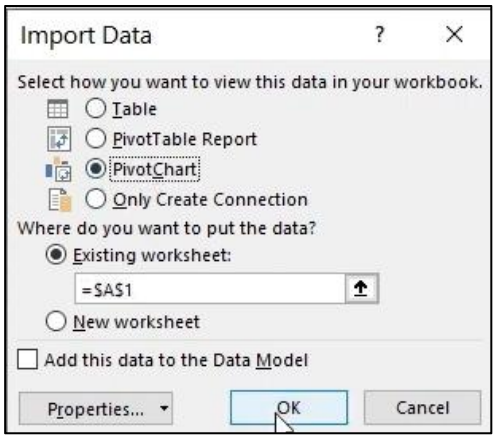
Step 4: In Select Database and Table Select → Sales_DW (already created in SQL) → check all dimensions and import relationships between selected tables



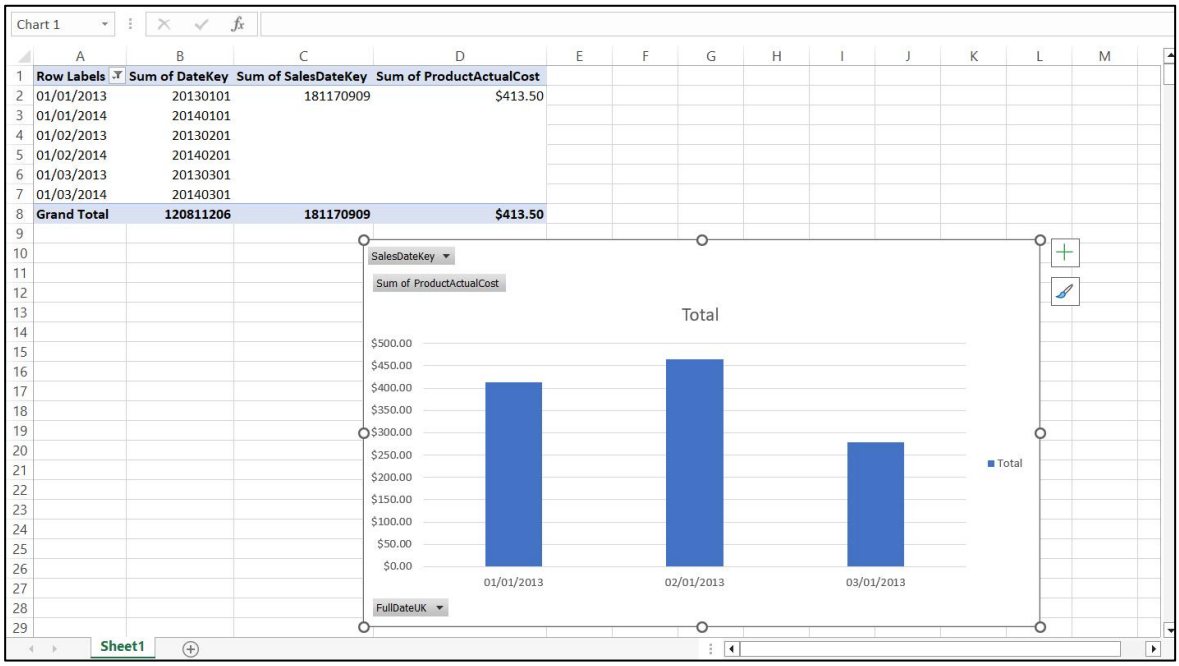
Step 5: In save data connection files browse path and click on Finish



Step 6: In import data select Pivot Chart and click on OK



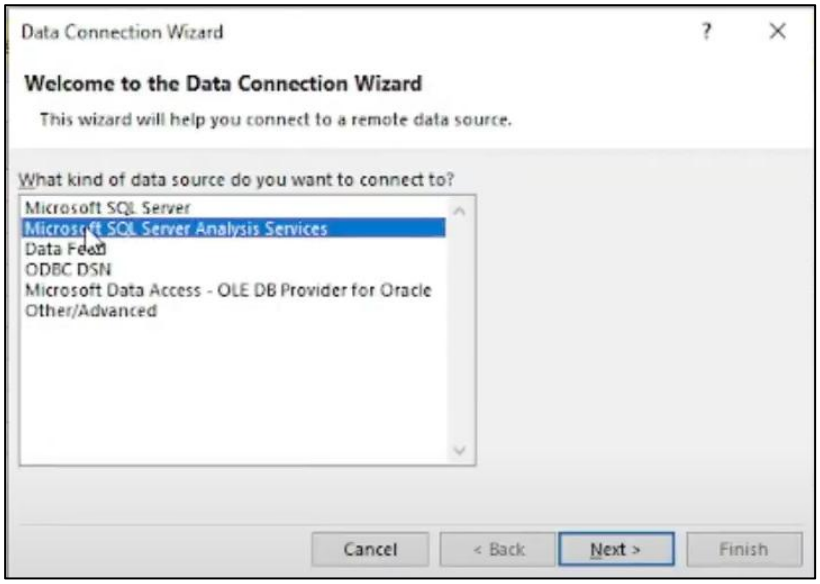
Step 7: In fields put SalesDateKey in filters, FullDateUK in axis and Sum of ProductActualCost in values



B) Aim: -Import the cube in Microsoft Excel and create the Pivot table and Pivot Chart to perform data analysis.

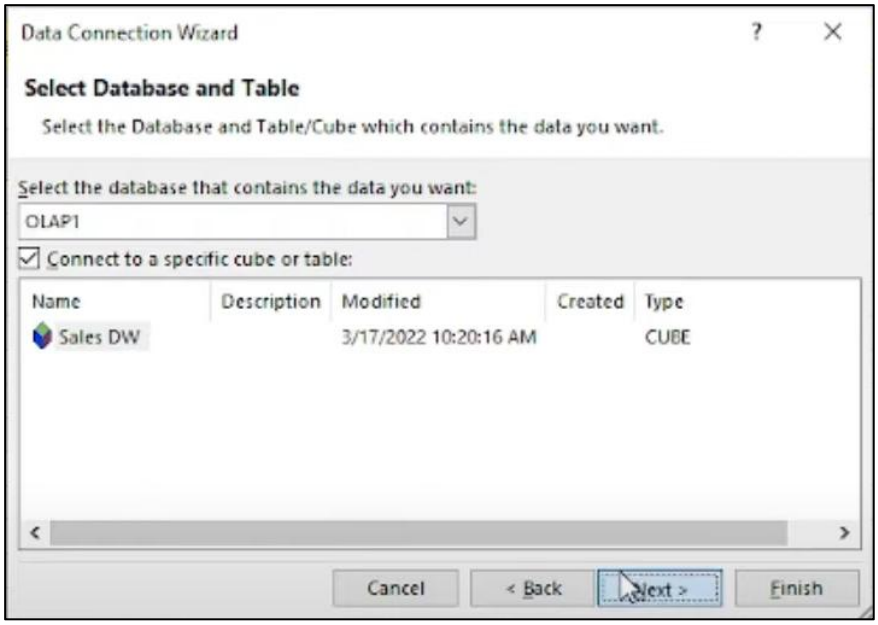
Step 1: Open Excel 2013 (Professional)

Go to Data tab → Get External Data From Other Sources → From Analysis Services

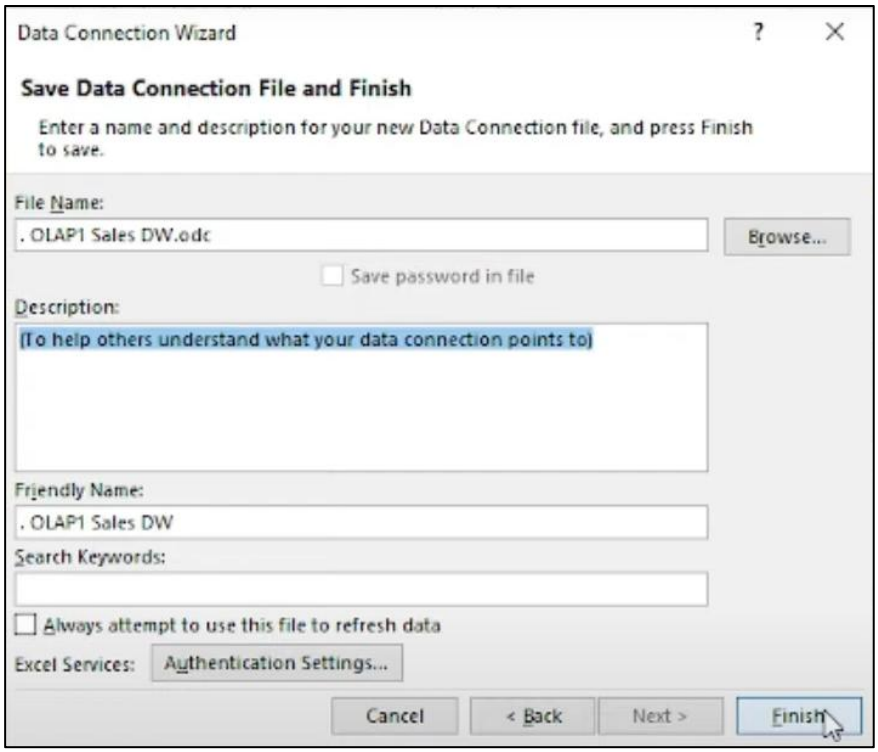


Step 2: Select Server name and Windows Authentication and click on Next

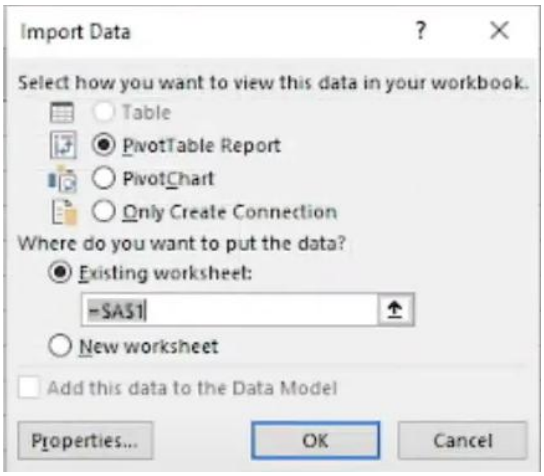
Step 3: Select OLAP (as per created before) click on Next



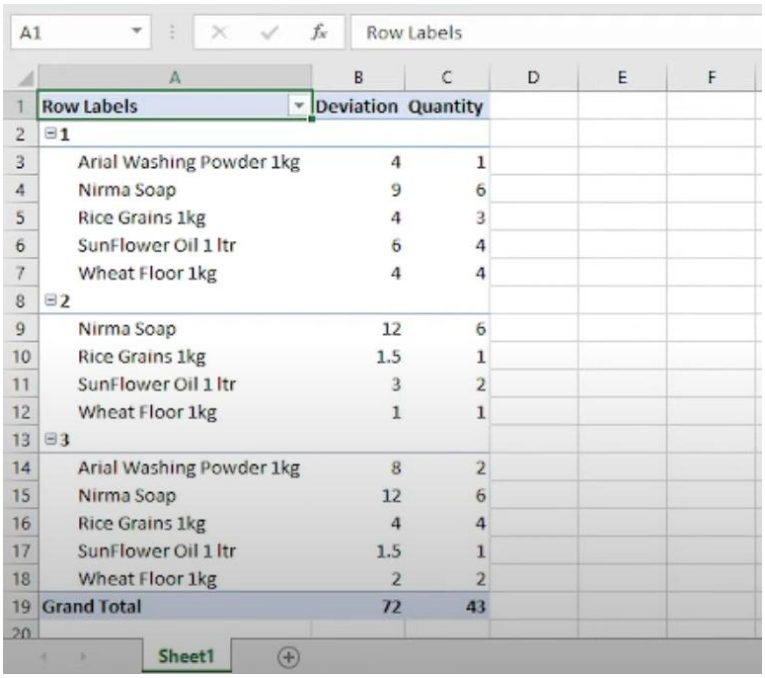
Step 4: Browse and select path name and click on Finish



Step 5: Select PivotTableReport OK Import Data

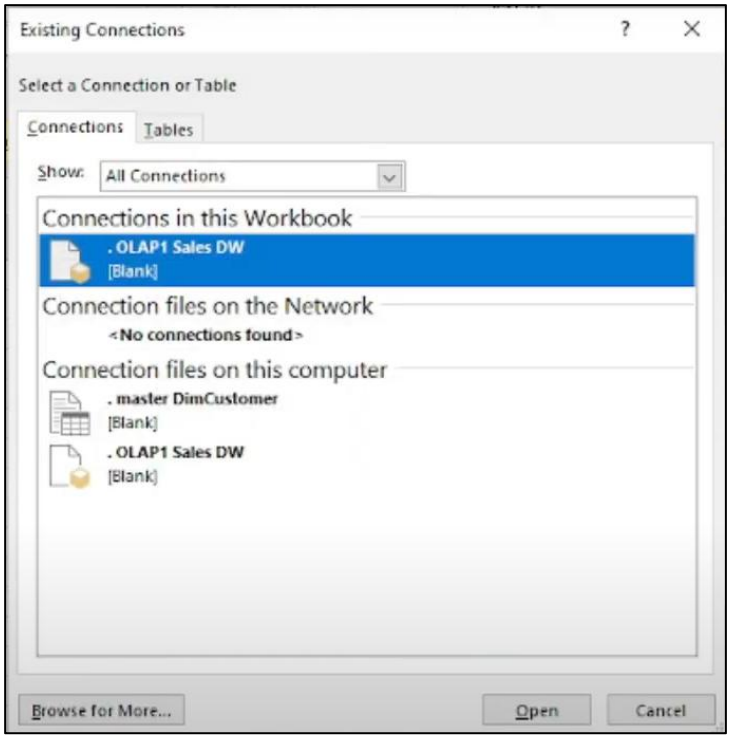


Step 6: Drag and Drop Fields in rows column and values

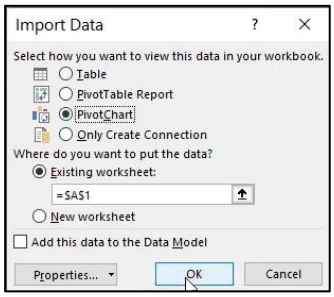


	A	B	C	D	E	F
1	Row Labels	Deviation	Quantity			
2	⊟1					
3	Arial Washing Powder 1kg	4	1			
4	Nirma Soap	9	6			
5	Rice Grains 1kg	4	3			
6	SunFlower Oil 1 ltr	6	4			
7	Wheat Floor 1kg	4	4			
8	⊟2					
9	Nirma Soap	12	6			
10	Rice Grains 1kg	1.5	1			
11	SunFlower Oil 1 ltr	3	2			
12	Wheat Floor 1kg	1	1			
13	⊟3					
14	Arial Washing Powder 1kg	8	2			
15	Nirma Soap	12	6			
16	Rice Grains 1kg	4	4			
17	SunFlower Oil 1 ltr	1.5	1			
18	Wheat Floor 1kg	2	2			
19	Grand Total	72	43			
20						

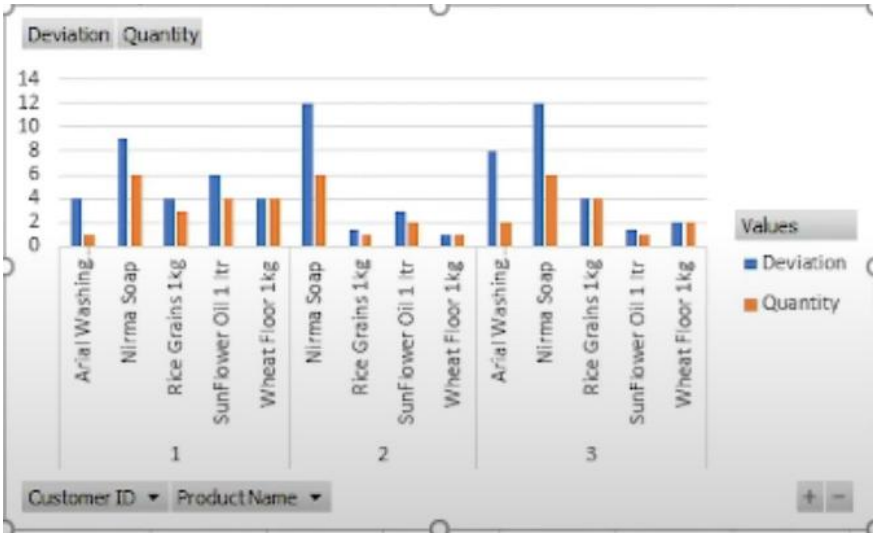
Step 7: Go to Data tab existing connections → Select existing connection OLAP Sales DW and click on Open



Step 8: Select PivotChart OK Import Data



Step 9: Click on OK

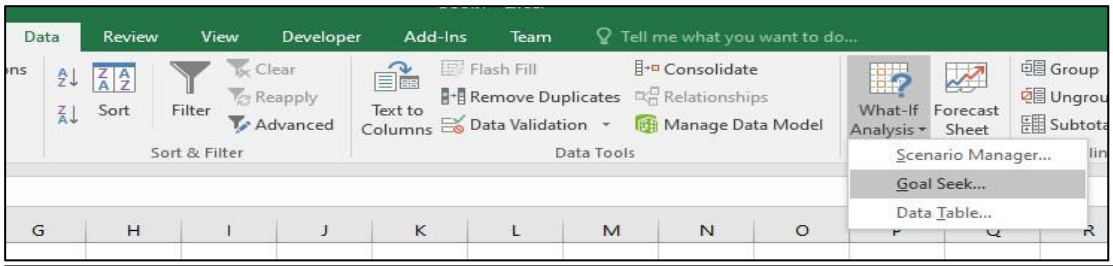

Deviation
Quantity
14
12
10
8
6
4
2
0
Arial Washing
Nirma Soap
Rice Grains 1kg
SunFlower Oil 1 ltr
Wheat Floor 1kg
Nirma Soap
Rice Grains 1kg
SunFlower Oil 1 ltr
Wheat Floor 1kg
Arial Washing
Nirma Soap
Rice Grains 1kg
SunFlower Oil 1 ltr
Wheat Floor 1kg
1
2
3
Values
Deviation
Quantity
Customer ID
Product Name

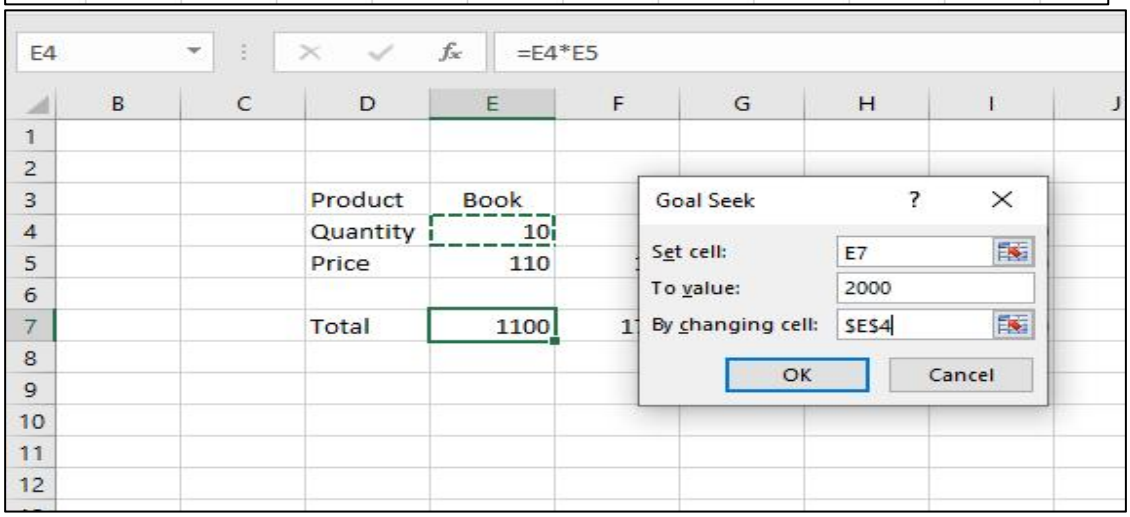
Practical No. 5

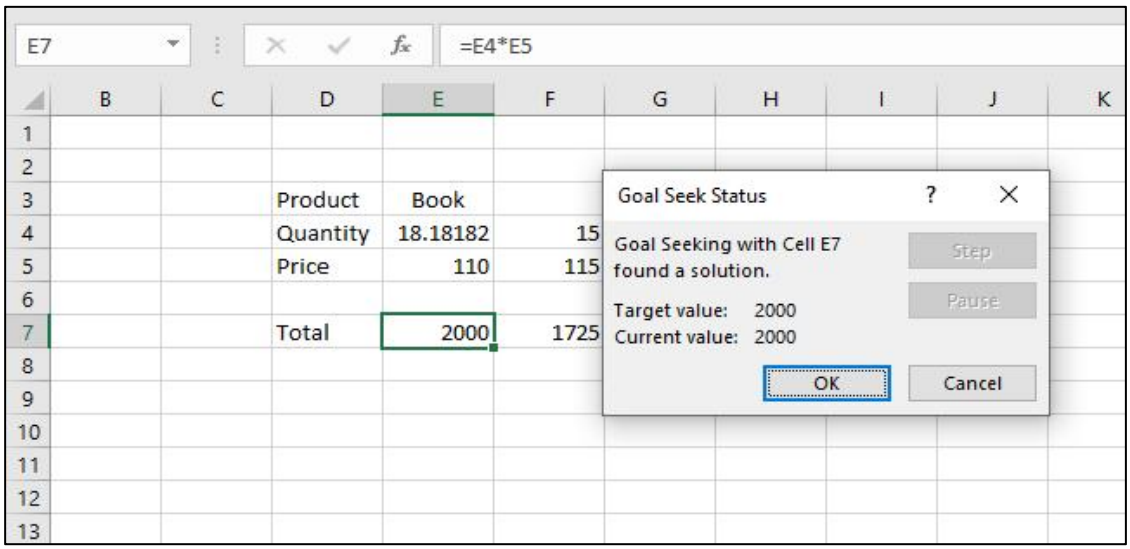
Aim: - Apply What if analysis for data visualization. Design and generate necessary reports based on the data warehouse data.

Goal Seek Method:

Step 1: Go to Data Tab → Select the What- if Analysis → click on goal seek

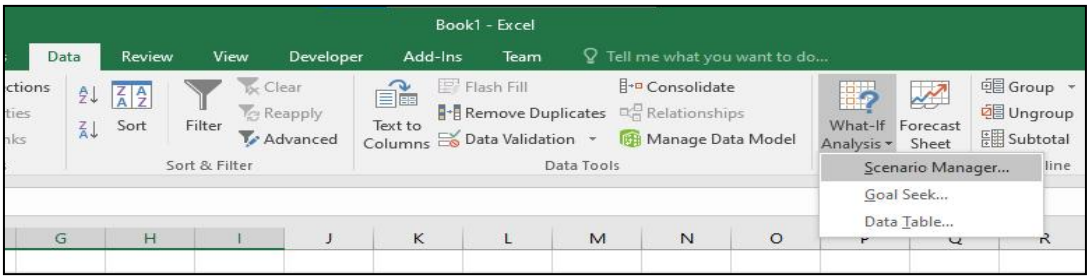




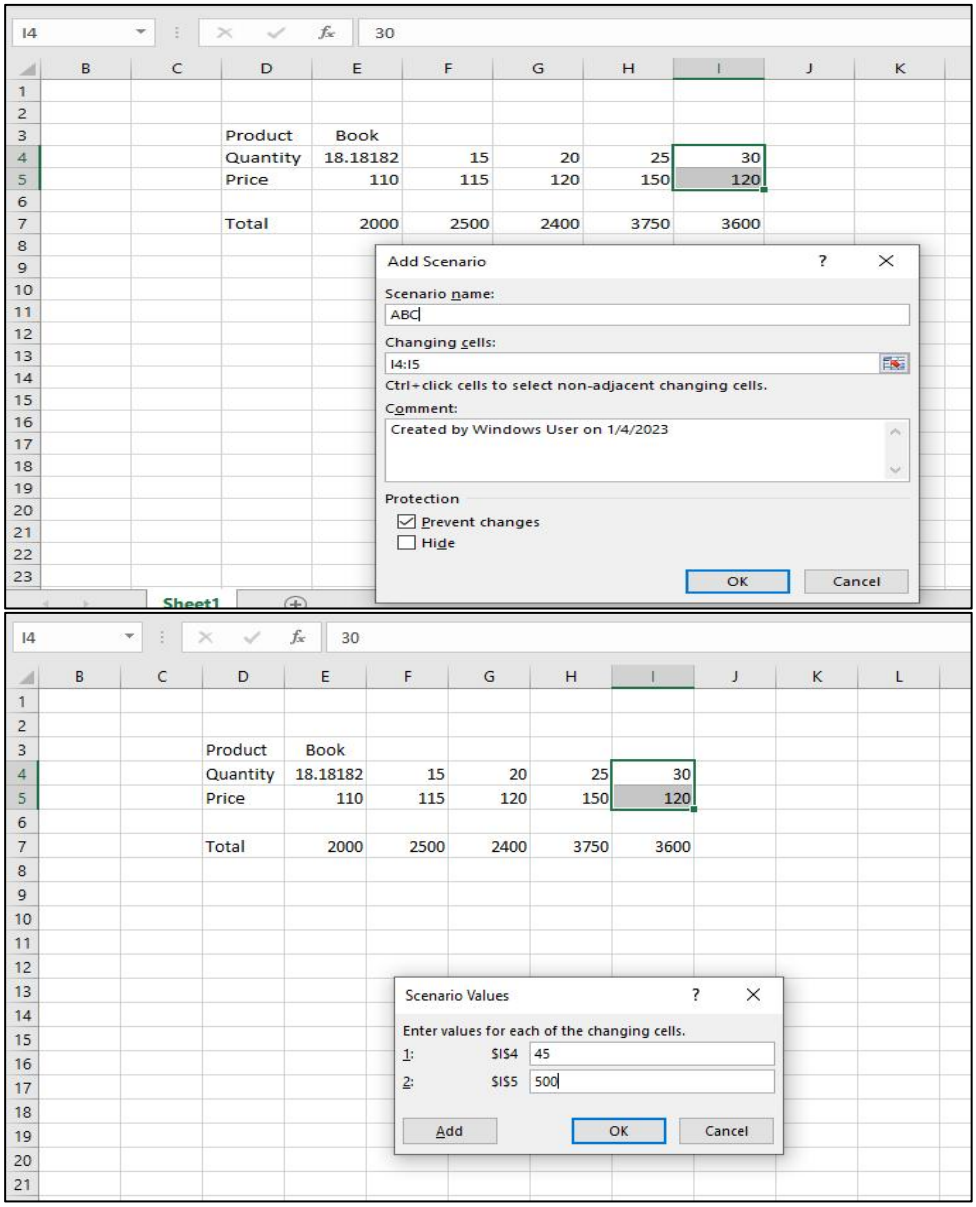


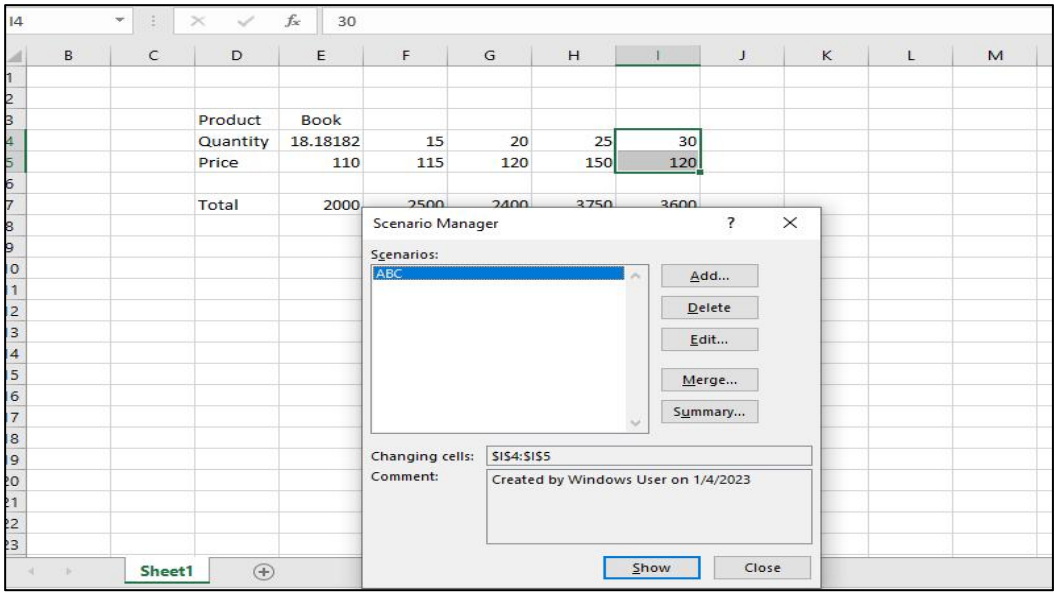
Scenario Manager Method

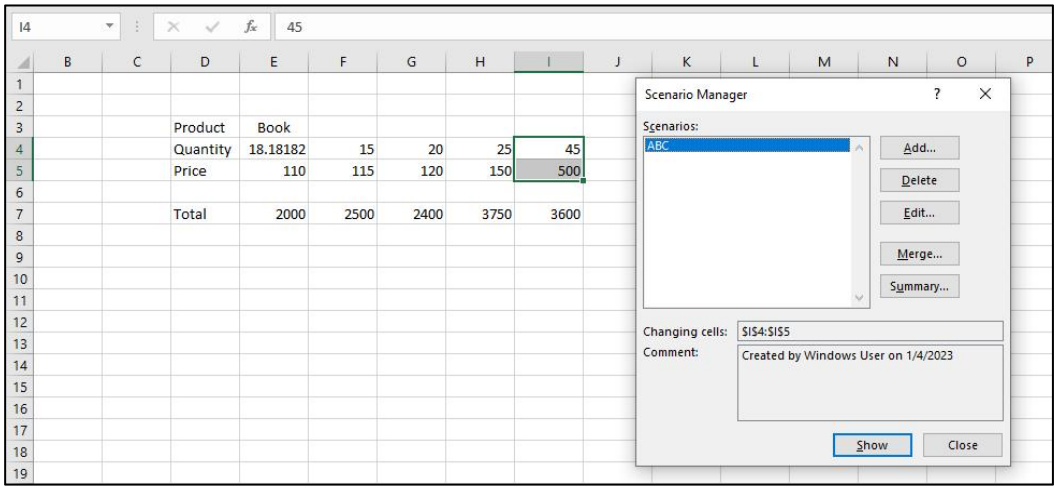
Step 1: Go to Data Tab → Select the What- if Analysis → click on Scenario Manager



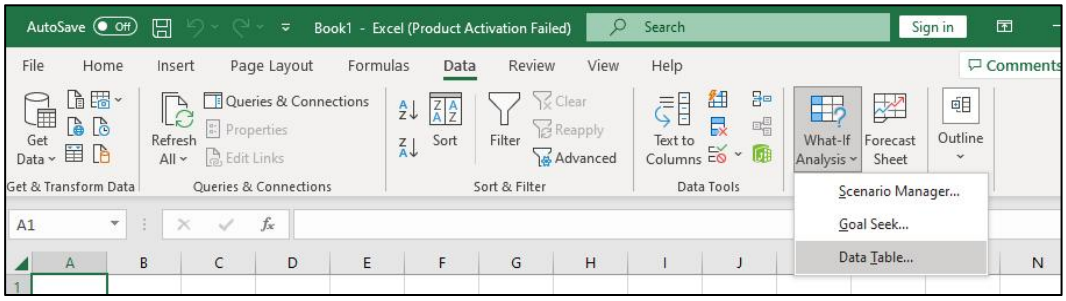
Step 2: Add Scenario values and then click on ok.

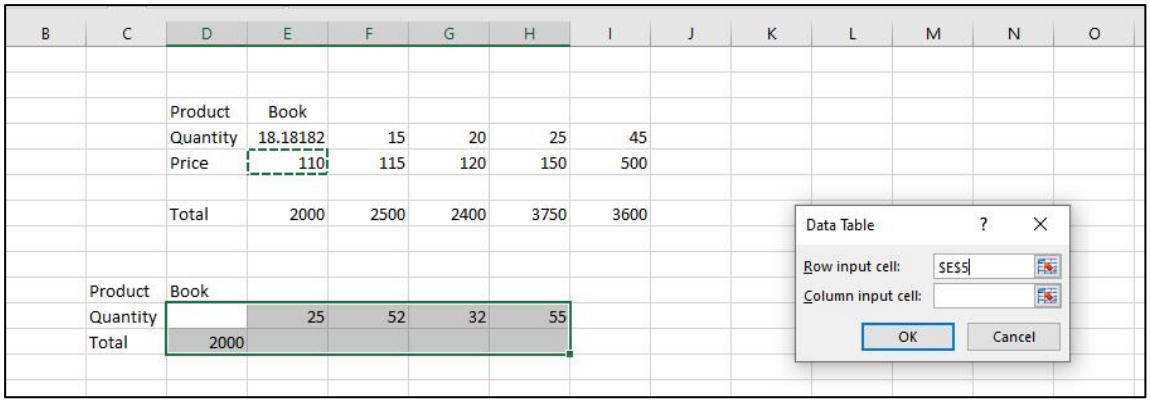


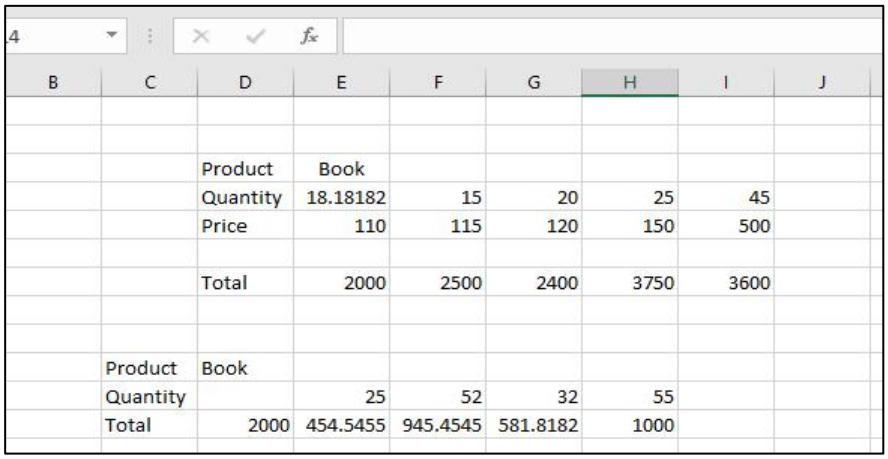




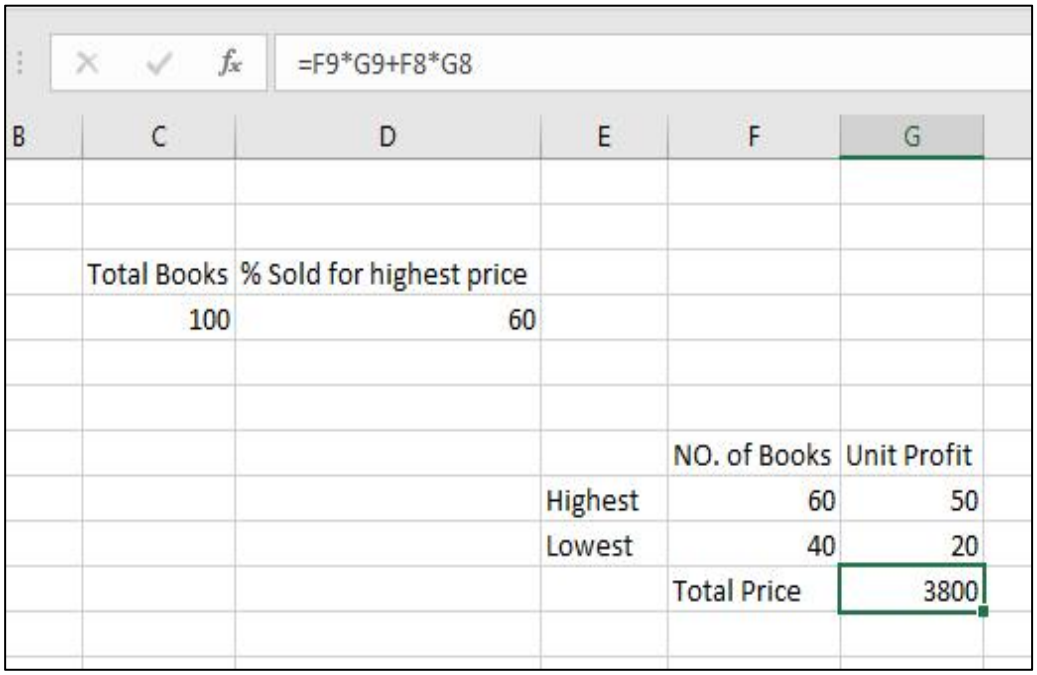
Data Table Method

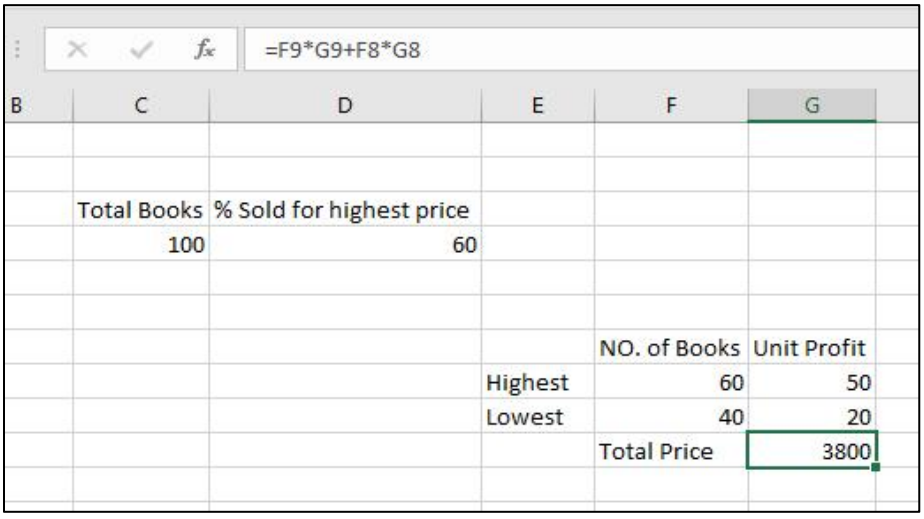



Product Book
Quantity 18.18182 15 20 25 45
Price 110 115 120 150 500
Total 2000 2500 2400 3750 3600
Product Book
Quantity 25 52 32 55
Total 2000
Data Table
Row input cell: E5
Column input cell:
OK
Cancel


Product Book
Quantity 18.18182 15 20 25 45
Price 110 115 120 150 500
Total 2000 2500 2400 3750 3600
Product Book
Quantity 25 52 32 55
Total 2000 454.5455 945.4545 581.8182 1000

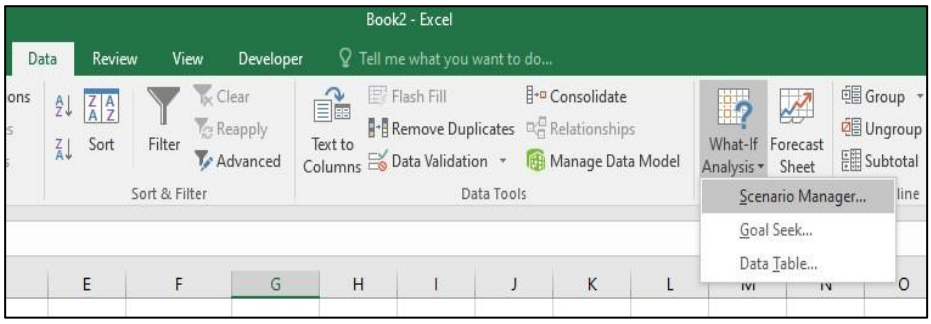
Percentage


=F9*G9+F8*G8
Total Books % Sold for highest price
100 60
NO. of Books Unit Profit
Highest 60 50
Lowest 40 20
Total Price 3800

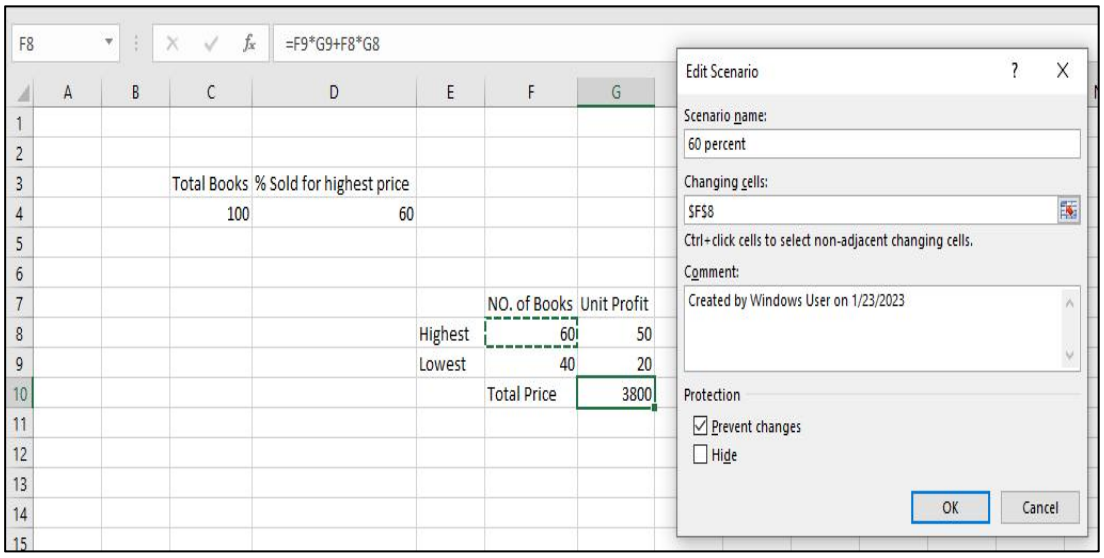


Step1: In Excel, On the Data tab, in the Data tools group, click What-If Analysis

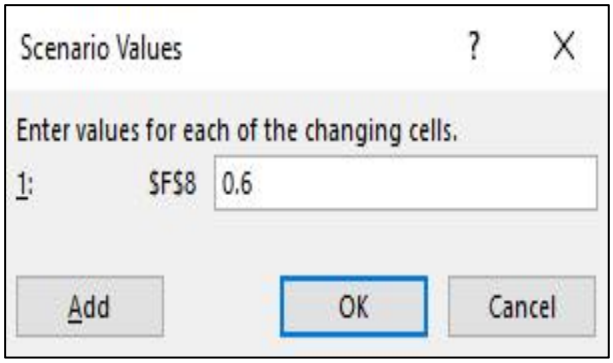
Step 2: Click on What –if-Analysis and select scenario manager.

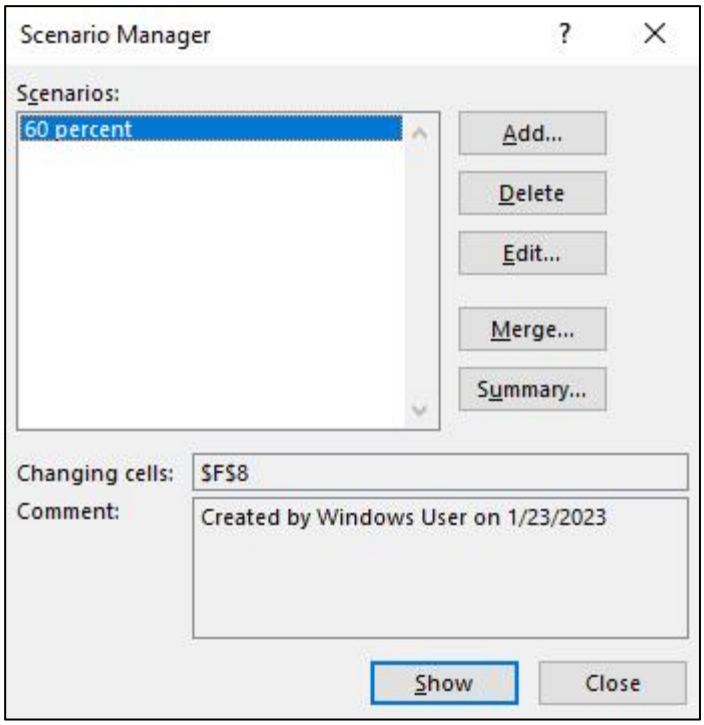


Step 3: Add a scenario by clicking on Add.

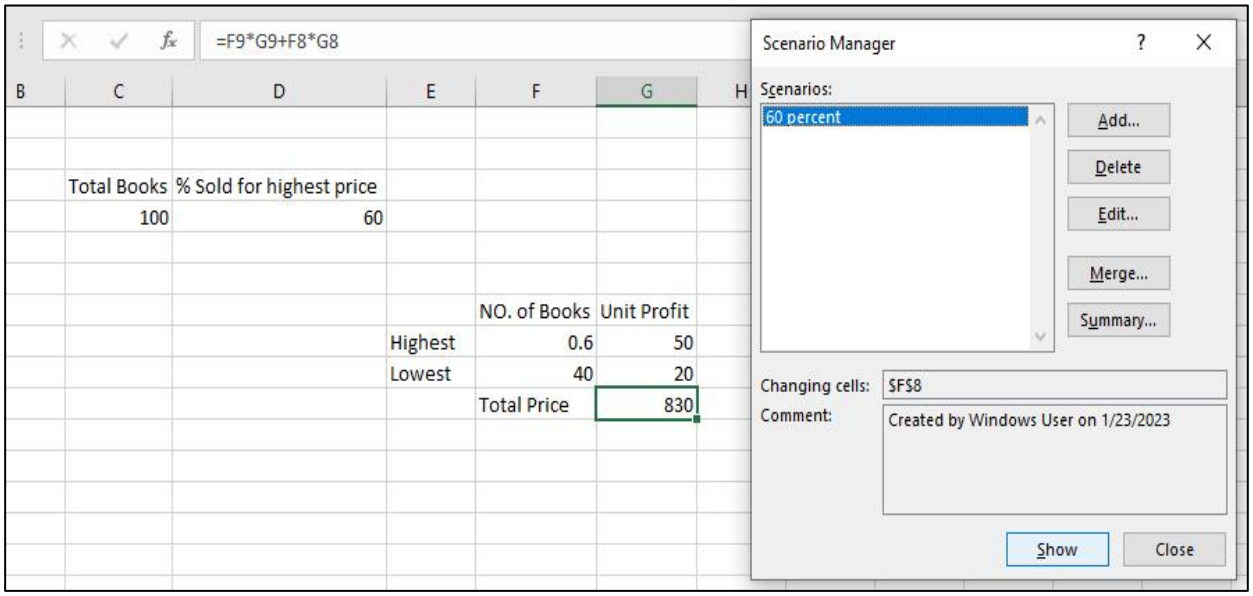


Step 4: Enter the corresponding value 0.6 and click on OK again.


Scenario Values
Enter values for each of the changing cells.
1:
F8
0.6
Add
OK
Cancel


Scenario Manager
Scenarios:
60 percent
Add...
Delete
Edit...
Merge...
Summary...
Changing cells:
F8
Comment:
Created by Windows User on 1/23/2023
Show
Close

FINAL OUTPUT:



Practical No. 6

Aim: Perform data classification using classification algorithm.

Time series is a series of data points in which each data point is associated with a timestamp. A simple example is the price of a stock in the stock market at different points of time on a given day. Another example is the amount of rainfall in a region at different months of the year. R language uses many functions to create, manipulate and plot the time series data. The data for the time series is stored in an R object called time-series object. It is also a R data object like a vector or data frame.

The time series object is created by using the ts() function.

SYNTAX

The basic syntax for ts() function in time series analysis is –

```
timeseries.object.name <-  ts(data, start, end, frequency)
```

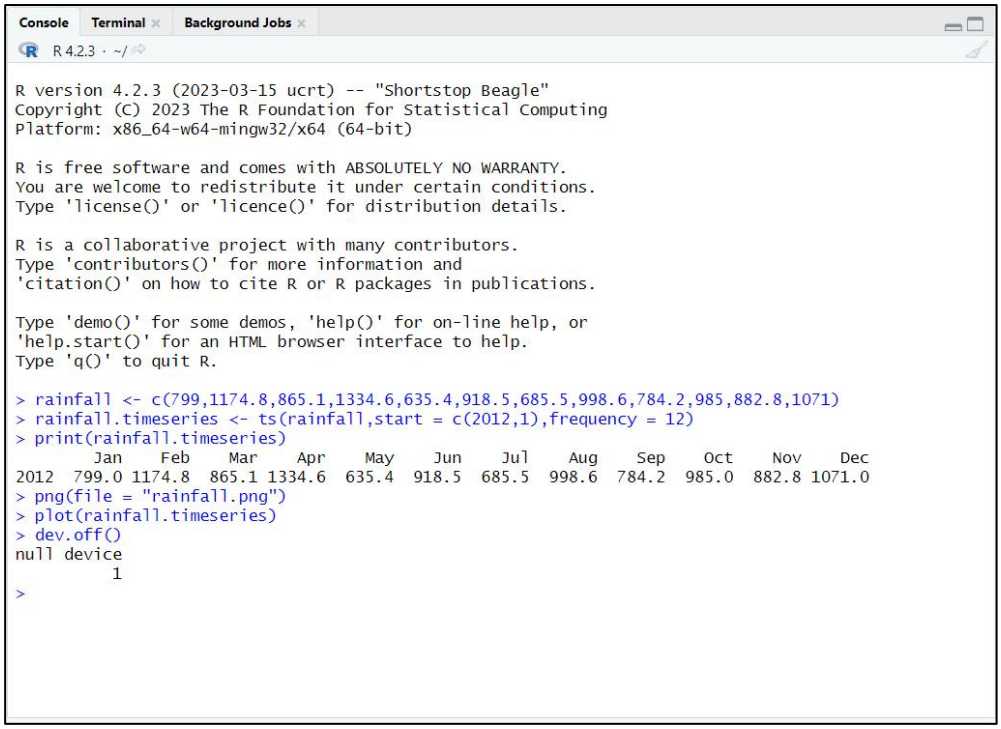
Following is the description of the parameters used −

- data is a vector or matrix containing the values used in the time series.
- start specifies the start time for the first observation in time series.
- end specifies the end time for the last observation in time series.
- frequency specifies the number of observations per unit time.

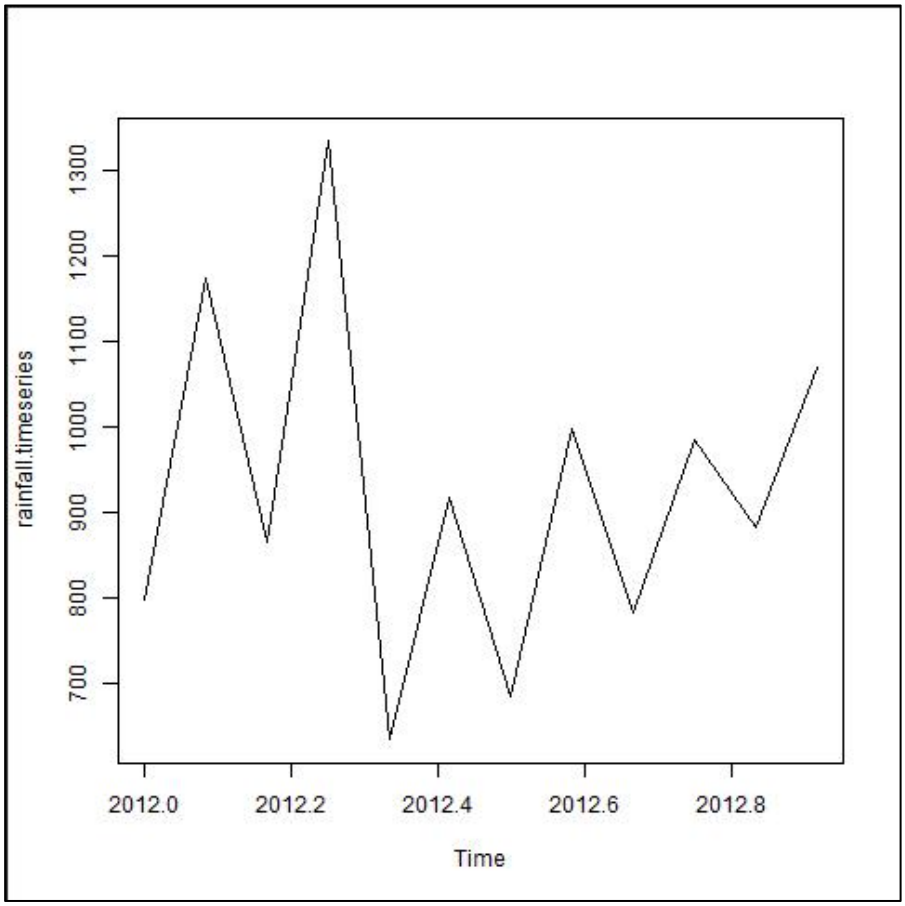
Except the parameter "data" all other parameters are optional.

Consider the annual rainfall details at a place starting from January 2012. We create an R time series object for a period of 12 months and plot it.

```
#Get the data points in form of a R vector.
rainfall <- c(799,1174.8,865.1,1334.6,635.4,918.5,685.5,998.6,784.2,985,882.8,1071)
#Convert it ti a timeseries object
rainfall.timeseries <- ts(rainfall,start = c(2012,1),frequency=12)
#print the timeseries data
print(rainfall.timeseries)
#give the chart file a name
png(file = "rainfall.png")
#plot a graph of the time series
plot(rainfall.timeseries)
#save the file
dev.off()
```



Output:


rainfall.timeseries
700
800
900
1000
1100
1200
1300
2012.0
2012.2
2012.4
2012.6
2012.8
Time

Practical No. 7

Aim: -Perform Data clustering using clustering algorithm.

K Means Clustering in R Programming is an Unsupervised Non-linear algorithm that cluster data based on similarity or similar groups. It seeks to partition the observations into a pre-specified number of clusters. Segmentation of data takes place to assign each training example to a segment called a cluster. In the unsupervised algorithm, high reliance on raw data is given with large expenditure on manual review for review of relevance is given. It is used in a variety of fields like Banking, healthcare, retail, Media, etc.

Theory

K-Means clustering groups the data on similar groups. The algorithm is as follows:

- Choose the number K clusters.
- Select at random K points, the centroids(Not necessarily from the given data).
- Assign each data point to closest centroid that forms K clusters.
- Compute and place the new centroid of each centroid.
- Reassign each data point to new cluster.
- After final reassignment, name the cluster as Final cluster.

The Dataset

Iris dataset consists of 50 samples from each of 3 species of Iris(Iris setosa, Iris virginica, Iris versicolor) and a multivariate dataset introduced by British statistician and biologist Ronald Fisher in his 1936 paper The use of multiple measurements in taxonomic problems. Four features were measured from each sample i.e length and width of the sepals and petals and based on the combination of these four features, Fisher developed a linear discriminant model to distinguish the species from each other.

```
# Installing Packages
install.packages("ClusterR")
install.packages("cluster")

# Loading package
library(ClusterR)
```

```
library(cluster)

# Removing initial label of
# Species from original dataset
iris_1 <- iris[, -5]

# Fitting K-Means clustering Model
# to training dataset
set.seed(240) # Setting seed
kmeans.re <- kmeans(iris_1, centers = 3, nstart = 20)
kmeans.re

# Cluster identification for
# each observation
kmeans.re$cluster

# Confusion Matrix
cm <- table(iris$Species, kmeans.re$cluster)
cm

# Model Evaluation and visualization
plot(iris_1[c("Sepal.Length", "Sepal.Width")])
plot(iris_1[c("Sepal.Length", "Sepal.Width")],
     col = kmeans.re$cluster)
plot(iris_1[c("Sepal.Length", "Sepal.Width")],
     col = kmeans.re$cluster,
     main = "K-means with 3 clusters")
```

```
## Plotting cluster centers
kmeans.re$centers
kmeans.re$centers[, c("Sepal.Length", "Sepal.Width")]

# cex is font size, pch is symbol
points(kmeans.re$centers[, c("Sepal.Length", "Sepal.Width")],
       col = 1:3, pch = 8, cex = 3)

## Visualizing clusters
y_kmeans <- kmeans.re$cluster
clusplot(iris_1[, c("Sepal.Length", "Sepal.Width")],
                 y_kmeans,
                 lines = 0,
                 shade = TRUE,
                 color = TRUE,
                 labels = 2,
                 plotchar = FALSE,
                 span = TRUE,
                 main = paste("Cluster iris"),
                 xlab = 'Sepal.Length',
                 ylab = 'Sepal.Width')
```

Output:

Model kmeans_re:

```
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
 [46] 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 3 2 2 2 2 2 2 2 2 2 2
 [91] 2 2 2 2 2 2 2 2 2 2 3 2 3 3 3 3 3 3 3 2 2 3 3 3 2 3 2 3 2 3 2 2 3 3 3 3 2 3
[136] 3 3 3 2 3 3 3 2 3 3 3 2 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"
[7] "size"         "iter"         "ifault"
>
```

Cluster identification:

```
> # Cluster identification for
> # each observation
> kmeans.re$cluster
  [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
 [46] 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 3 2 2 2 2 2 2 2 2 2 2
 [91] 2 2 2 2 2 2 2 2 2 2 3 2 3 3 3 3 3 3 3 2 2 3 3 3 2 3 2 3 2 3 2 2 3 3 3 3 2 3
[136] 3 3 3 2 3 3 3 2 3 3 3 2 3 3 2
>
```

Confusion Matrix:

```
> cm

           1  2  3
  setosa  50  0  0
  versicolor  0 48  2
  virginica   0 14 36
>
```

Practical No. 8

Aim: Perform the Linear regression on the given data warehouse data.

Regression analysis is a very widely used statistical tool to establish a relationship model between two variables. One of these variables is called predictor variable whose value is gathered through experiments. The other variable is called response variable whose value is derived from the predictor variable.

In Linear Regression these two variables are related through an equation, where exponent (power) of both these variables is 1. Mathematically a linear relationship represents a straight line when plotted as a graph. A non-linear relationship where the exponent of any variable is not equal to 1 creates a curve.

Following is the description of the parameters used −

- y is the response variable.
- x is the predictor variable.
- a and b are constants which are called the coefficients.

Steps to Establish a Regression

A simple example of regression is predicting weight of a person when his height is known. To do this we need to have the relationship between height and weight of a person.

The steps to create the relationship is −

- Carry out the experiment of gathering a sample of observed values of height and corresponding weight.
- Create a relationship model using the lm() functions in R.
- Find the coefficients from the model created and create the mathematical equation using these
- Get a summary of the relationship model to know the average error in prediction. Also called residuals.
- To predict the weight of new persons, use the predict() function in R.

Input Data

Below is the sample data representing the observations -

#Values of height

151, 174, 138, 186, 128, 136, 179, 163, 152, 131

#Values of weight.

63, 81, 56, 91, 47, 57, 76, 72, 62, 48

lm() Function:

This function creates the relationship model between the predictor and the response variable.

Syntax:

The basic syntax for lm() function in linear regression is-

lm(formula,data)

Following is the description of the parameters used: -

- formula is a symbol presenting the relation between x and
- data is the vector on which the formula will be applied.

A. Create Relationship Model & get the Coefficients

```
#Values of height

x<-c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

#Values of width

y<-c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

#Apply the lm() function.

relation <- lm(y~x)

print(relation)
```

```
Console   Terminal   Background Jobs
R 4.2.3 · ~/
> 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)
> print(relation)

Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)            x
   -38.4551       0.6746

>
```

B. Get the Summary of the Relationship

```
# Values of height

x<-c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

#Values of width

y<- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

#Apply the lm() function.

relation <- lm(y~x)

print(summary(relation))
```

```
Console   Terminal   Background Jobs
R 4.2.3 · ~/
> 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)
> print(summary(relation))

Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-6.3002 -1.6629  0.0412  1.8944  3.9775

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -38.45509    8.04901  -4.778  0.00139 **
x             0.67461    0.05191  12.997 1.16e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.253 on 8 degrees of freedom
Multiple R-squared:  0.9548,    Adjusted R-squared:  0.9491
F-statistic: 168.9 on 1 and 8 DF,  p-value: 1.164e-06
```

predict() Function

Syntax

The basic syntax for predict() in linear regression is-

predict(object, newdata)

Following is the description of the parameters used-

- object is the formula which is already created using the lm() function.
- newdata is the vector containing the new value for predictor variable.

C. Predict the weight of new persons

#The predictor vector.

x<- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

#The response vector.

y<-c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

Apply the lm() function.

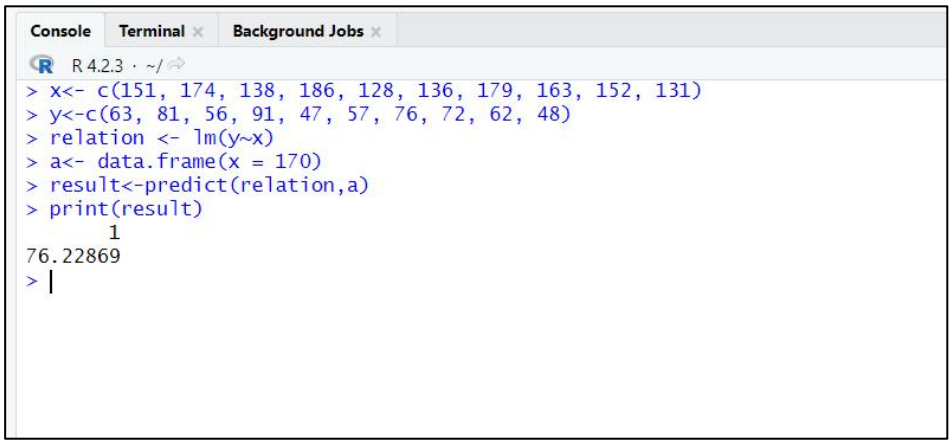
relation <- lm(y~x)

#Find weight of a person with height 170.

a<- data.frame(x = 170)

result predict(relation,a)

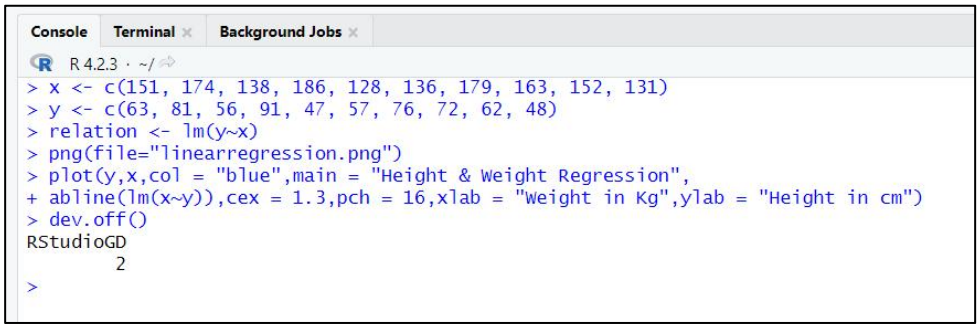
print(result)



D. Visualize the Regression Graphically

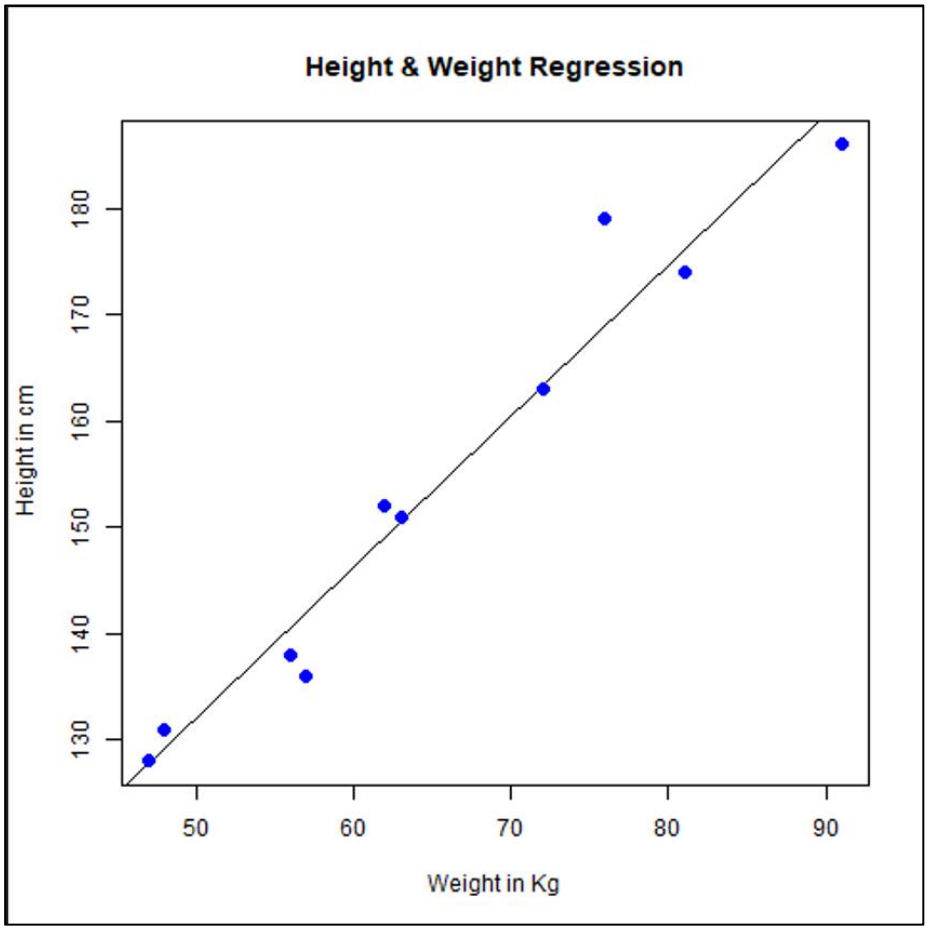
```
#Create the predictor and response variable.
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
relation <- lm(y~x)
#Give the chart file a name.
png(file="linearregression.png")

# Plot the chart
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")
# Save the file.
dev.off()
```




```
#Plot the chart.
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")
```

Output:


Height & Weight Regression
Height in cm
130
140
150
160
170
180
50
60
70
80
90
Weight in Kg

Practical No. 9

Aim: Perform the Logical regression on the given data warehouse data.

The Logistic Regression is a regression model in which the response variable (dependent variable) has categorical values such as True/False or 0/1. It actually measures the probability of a binary response as the value of response variable based on the mathematical equation relating it with the predictor variables.

The general mathematical equation for logistic regression is –

y = 1/(1+e^-(a+b1x1+b2x2+b3x3+...))

Following is the description of the parameters used −

- y is the response variable.
- x is the predictor variable.
- a and b are the coefficients which are numeric constants.

The function used to create the regression model is the glm() function.

Syntax:

The basic syntax for glm() function in logistic regression is −

```
#glm(formula,data,family)
```

Following is the description of the parameters used −

- formula is the symbol presenting the relationship between the variables.
- data is the data set giving the values of these variables.
- family is R object to specify the details of the model. It's value is binomial for logistic regression.

The in-built data set "mtcars" describes different models of a car with their various engine specifications. In "mtcars" data set, the transmission mode (automatic or manual) is described by the column am which is a binary value (0 or 1). We can create a logistic regression model between the columns "am" and 3 other columns - hp, wt and cyl.

```
Console  Terminal  Background Jobs
R 4.2.3 · ~/
> mtcars
                   mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
Merc 280C         17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
Merc 450SE        16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
Merc 450SL        17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
Merc 450SLC       15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
Fiat 128          32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic       30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla    33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
Toyota Corona     21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
Dodge Challenger  15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
AMC Javelin       15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
Camaro Z28        13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
Pontiac Firebird  19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
Fiat X1-9         27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Porsche 914-2     26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Lotus Europa      30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Ford Pantera L    15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Ferrari Dino      19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Maserati Bora     15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Volvo 142E        21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
>
```

```
# Select some columns form mtcars.

input <- mtcars[,c("am","cyl","hp","wt")]

print(head(input))
```

When we execute the above code, it produces the following result −

```
Console  Terminal  Background Jobs
R 4.2.3 · ~/
> input <- mtcars[,c("am","cyl","hp","wt")]
>
> print(head(input))
                  am cyl  hp    wt
Mazda RX4          1   6 110 2.620
Mazda RX4 Wag      1   6 110 2.875
Datsun 710         1   4  93 2.320
Hornet 4 Drive     0   6 110 3.215
Hornet Sportabout  0   8 175 3.440
Valiant            0   6 105 3.460
>
```

Create Regression Model

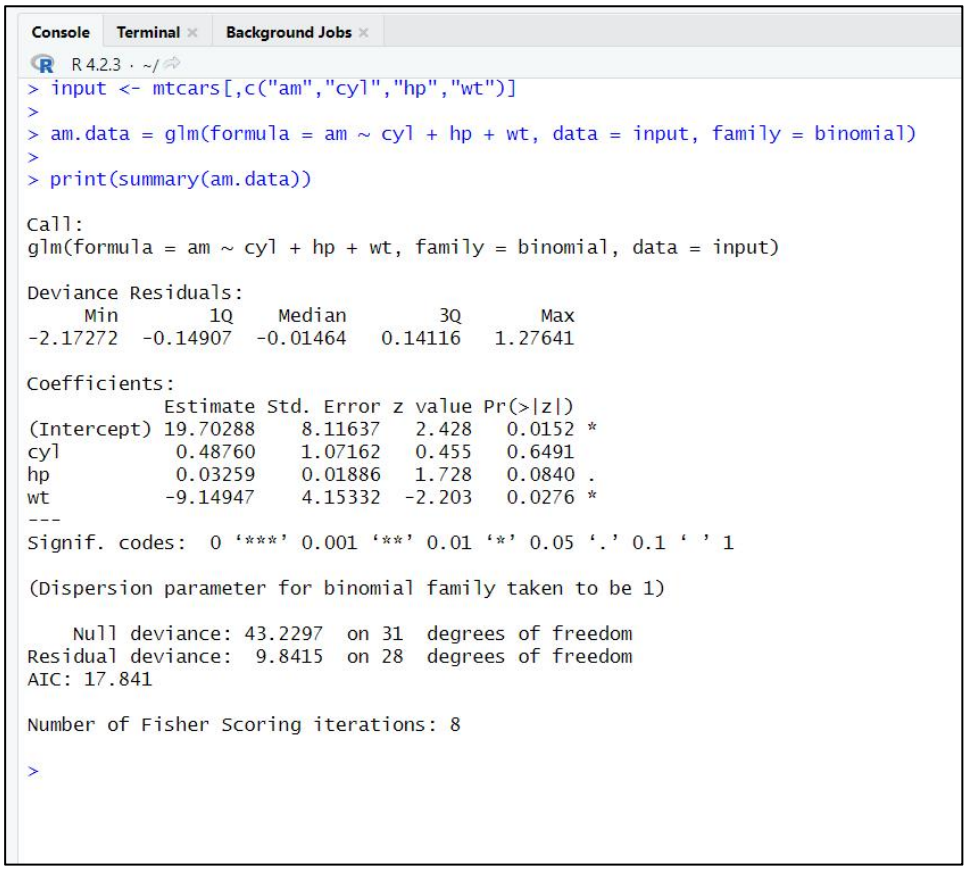
We use the glm() function to create the regression model and get its summary for analysis.

```
input <- mtcars[,c("am","cyl","hp","wt")]

am.data = glm(formula = am ~ cyl + hp + wt, data = input, family = binomial)

print(summary(am.data))
```

When we execute the above code, it produces the following result –



```
Console   Terminal   Background Jobs
R 4.2.3 · ~/
> input <- mtcars[,c("am","cyl","hp","wt")]
>
> am.data = glm(formula = am ~ cyl + hp + wt, data = input, family = binomial)
>
> print(summary(am.data))

Call:
glm(formula = am ~ cyl + hp + wt, family = binomial, data = input)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.17272  -0.14907  -0.01464   0.14116   1.27641

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 19.70288    8.11637   2.428   0.0152 *
cyl          0.48760    1.07162   0.455   0.6491
hp           0.03259    0.01886   1.728   0.0840 .
wt          -9.14947    4.15332  -2.203   0.0276 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 43.2297  on 31  degrees of freedom
Residual deviance:  9.8415  on 28  degrees of freedom
AIC: 17.841

Number of Fisher Scoring iterations: 8

>
```

Conclusion:

In the summary as the p-value in the last column is more than 0.05 for the variables "cyl" and "hp", we consider them to be insignificant in contributing to the value of the variable "am". Only weight (wt) impacts the "am" value in this regression model.