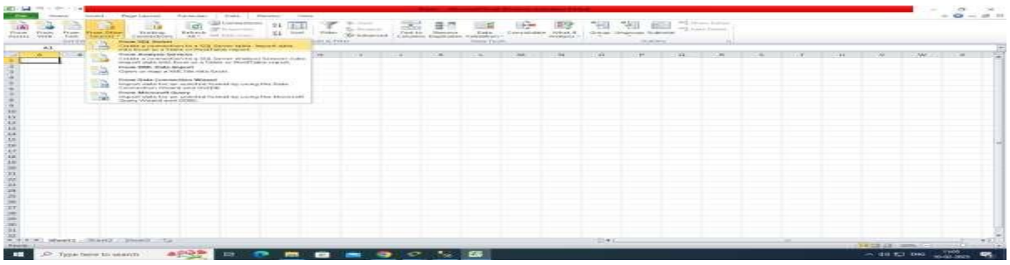
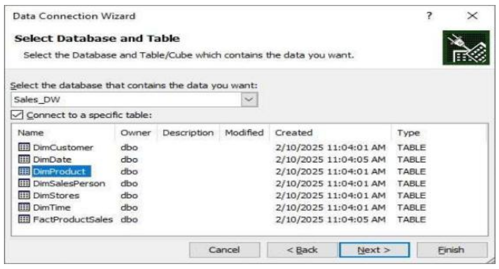
PRACTICAL NO. 01

Aim: 1a) Import the data warehouse data in Microsoft Excel and create the Pivot Table and Pivot Chart.

Step 1: In excel, click on Data Menu -> from other sources->from SQL Server



Step 2: Enter your SQL Server name and log on to window authentication and click next.
Step 3: In Data Connection Wizard, select database as Sales_DW and also select table.



Data Connection Wizard

Select Database and Table

Select the Database and Table/Cube which contains the data you want.

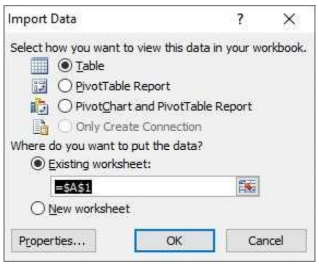
Select the database that contains the data you want:
Sales_DW

☑ Connect to a specific table:

Name	Owner	Description	Modified	Created	Type
DimCustomer	dbo			2/10/2025 11:04:01 AM	TABLE
DimDate	dbo			2/10/2025 11:04:05 AM	TABLE
DimProduct	dbo			2/10/2025 11:04:01 AM	TABLE
DimSalesPerson	dbo			2/10/2025 11:04:01 AM	TABLE
DimStores	dbo			2/10/2025 11:04:01 AM	TABLE
DimTime	dbo			2/10/2025 11:04:01 AM	TABLE
FactProductSales	dbo			2/10/2025 11:04:05 AM	TABLE

Step 4: Save file with default name and click finish.

Step 5: In this step, default Table Option and existing worksheet are already selected -> click ok

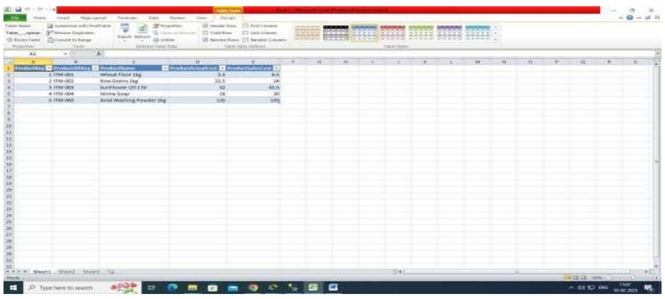


Import Data

Select how you want to view this data in your workbook.
[✓ Table]
PivotTable Report
PivotChart and PivotTable Report
Only Create Connection

Where do you want to put the data?
[✓ Existing worksheet:]
=$A$1
New worksheet

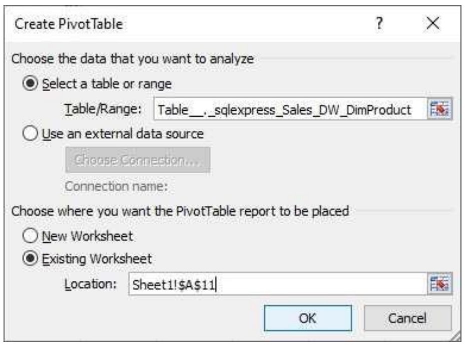
Step 6: It will open table in existing worksheet.



ProductKey	ProductAltKey	ProductName	ProductActualCost	ProductSalesCost
1	ITM-001	Wheat Flour 1kg	5.5	6.5
2	ITM-002	Rice Grains 1kg	22.5	24
3	ITM-003	SunFlower Oil 1 ltr	42	43.5
4	ITM-004	Nirma Soap	18	20
5	ITM-005	Arial Washing Powder 1kg	135	139

Step 7: Now to create pivot table click on Insert Menu -> Pivot Table

Step 8: Now in create pivot table window select existing worksheet option and also select proper location where you want to display pivot table and click ok.



Create PivotTable

Choose the data that you want to analyze

[✓ Select a table or range]

Table/Range: Table__._sqlexpress_Sales_DW_DimProduct

Use an external data source

Choose Connection...

Connection name:

Choose where you want the PivotTable report to be placed

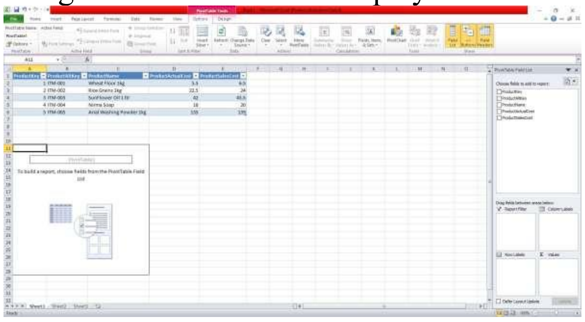
New Worksheet

[✓ Existing Worksheet]

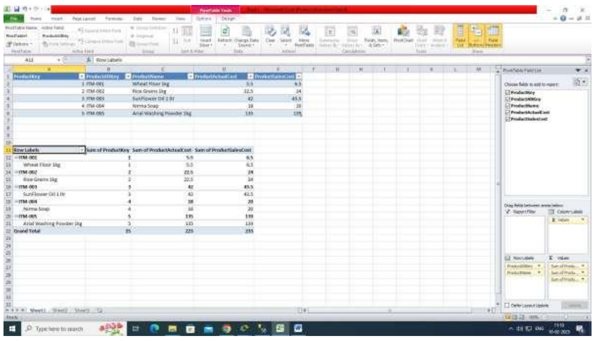
Location: Sheet1!$A$11

OK Cancel

After clicking on ok button it will display like this



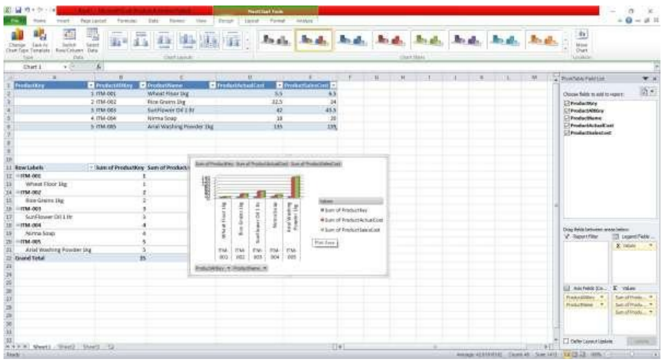
Step 9: In select pivot table fields we can choose fields to add to report -> here we are selecting all the fields.



Row Labels	Sum of ProductKey	Sum of ProductActualCost	Sum of ProductSalesCost
ITM-001	1	5.5	6.5
Wheat Floor 1kg	1	5.5	6.5
ITM-002	2	22.5	24
Rice Grains 1kg	2	22.5	24
ITM-003	3	42	43.5
SunFlower Oil 1 ltr	3	42	43.5
ITM-004	4	18	20
Nirma Soap	4	18	20
ITM-005	5	135	139
Arial Washing Powder 1kg	5	135	139
Grand Total	15	223	233

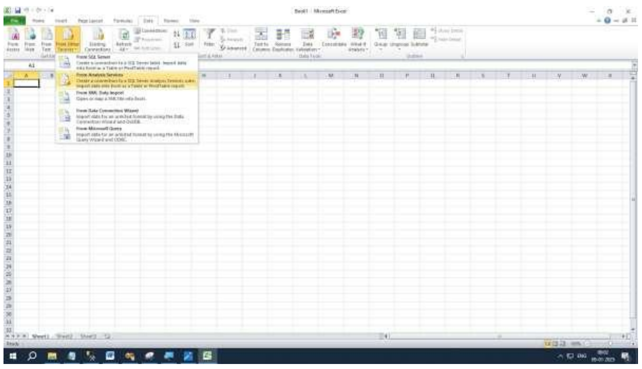
Step 10: To create Pivot Chart again go to Insert Menu-> Pivot Chart

Step 11: Here select chart type of your interest-> we are selecting default clustered column>select click ok



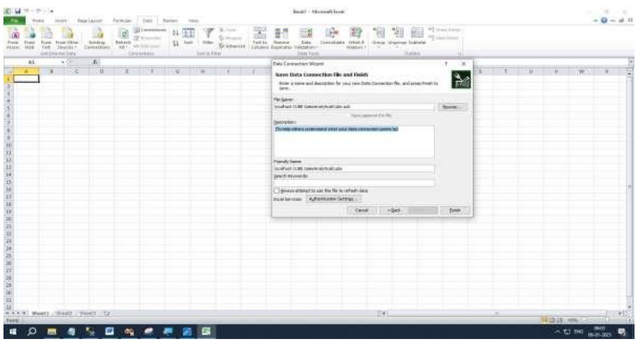
Aim: 1b) Import the cube in Microsoft Excel and create the Pivot Table and Pivot Chart to perform data analysis.

Step 1: First in excel click on data menu -> select from other sources -> from analysis services



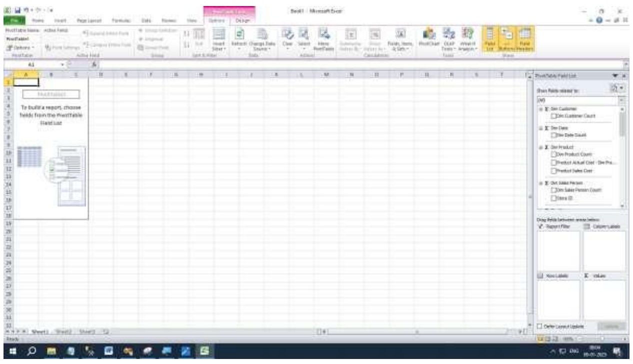
<u>Step 2</u>: Enter your SQL Server name and log on to windows authentication and click next

<u>Step 3</u>: In Data Connection Wizard, select database as cubepract4 and also select your cube name as SalesAnalyticalCube and click Next.



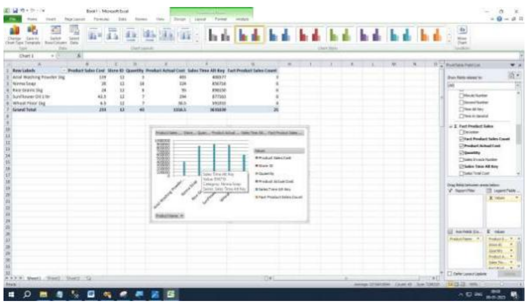
<u>Step 4</u>: Save file with default name and click finish

<u>Step 5</u>: Now click on ok



<u>Step 6</u>: In select pivot table fields we can choose fields to add to report -> here we are selecting all fields

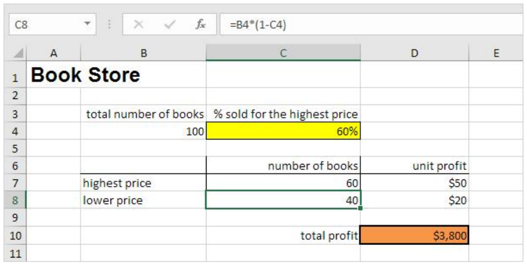
<u>Step 7</u>: To create Pivort Chart again go to Insert Menu -> Pivot Chart and select pie chart -> click ok.



PRACTICAL NO. 02

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

A book store and have 100 books in storage. You sell a certain % for the highest price of $50 and a certain % for the lower price of $20.



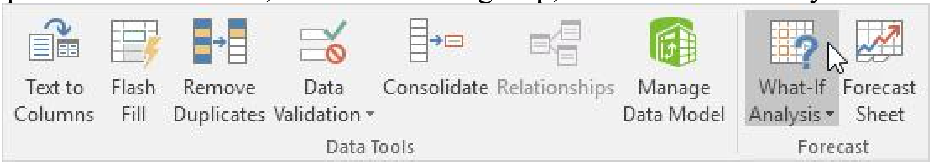
C8 =B4*(1-C4)

	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					

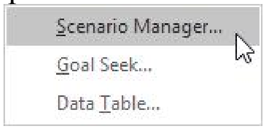
If you sell 60% for the highest price, cell D10 calculates a total profit of 60 * $50 + 40 * $20 = $3800.

Create Different Scenarios
But what if you sell 70% for the highest price? And what if you sell 80% for the highest price? Or 90%, or even 100%? Each different percentage is a different scenario. You can use the Scenario Manager to create these scenarios.
Note: You can simply type in a different percentage into cell C4 to see the corresponding result of a scenario in cell D10. However, what-if analysis enables you to easily compare the results of different scenarios. Read on.

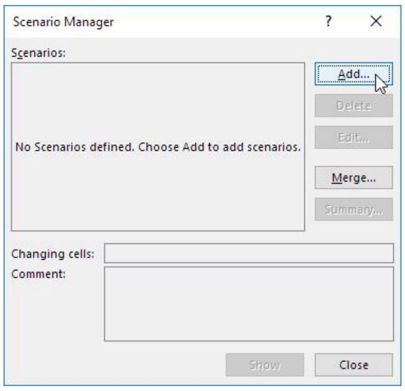
Step 1: On the Data tab, in the Forecast group, click What-If Analysis.



Step 2: Click Scenario Manager.

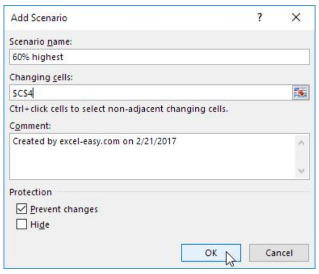


The Scenario Manager dialog box appears.
Step 3: Add a scenario by clicking on Add.

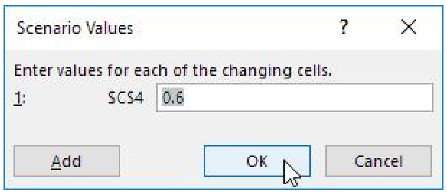


No Scenarios defined. Choose Add to add scenarios.

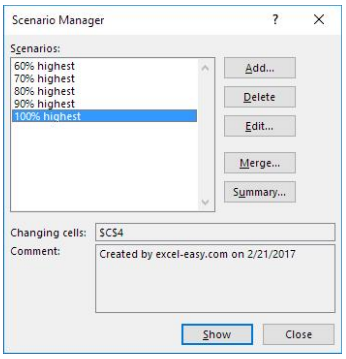
Step 4: Type a name (60% highest), select cell C4 (% sold for the highest price) for the Changing cells and click on OK.



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



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



PRACTICAL NO. 03

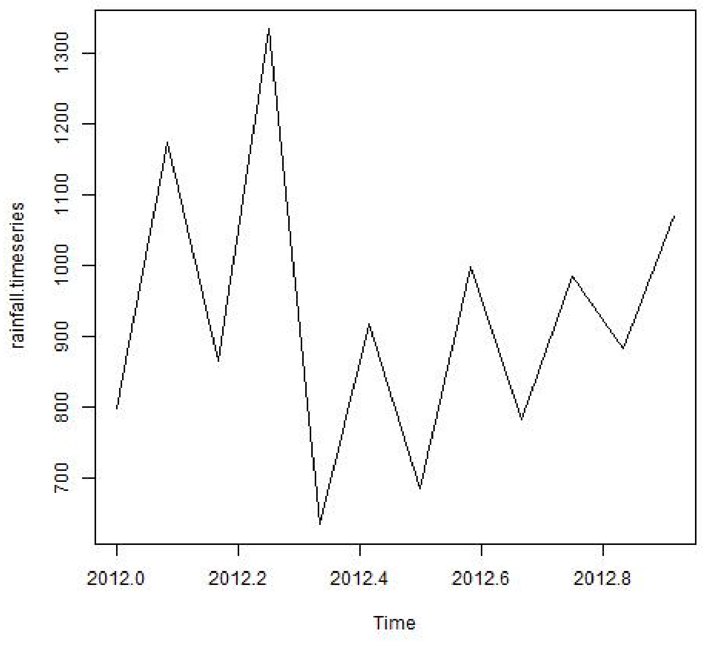
Aim: Perform the data classification using classification algorithm using R/Python.

Code:

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 to a time series 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()
```

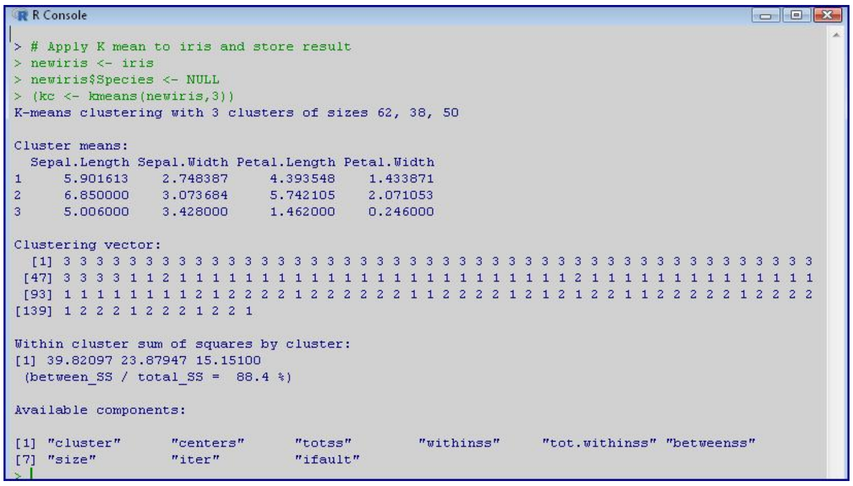
Output:



PRACTICAL NO. 04

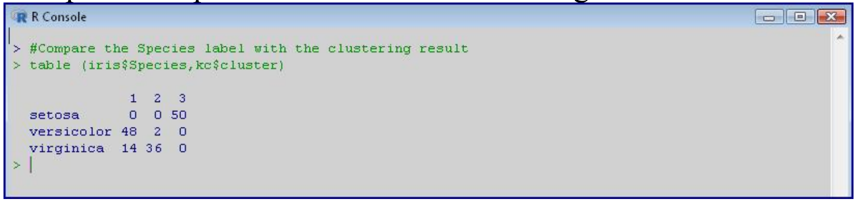
Aim: Perform the data clustering using clustering algorithm using R/Python.

Code:



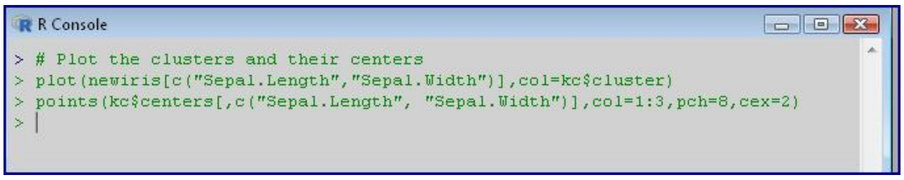
```
> # Apply K mean to iris and store result
> newiris <- iris
> newiris$Species <- NULL
> (kc <- kmeans(newiris,3))
K-means clustering with 3 clusters of sizes 62, 38, 50

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

Clustering vector:
  [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 [47] 3 3 3 3 1 1 2 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 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [93] 1 1 1 1 1 1 1 1 2 1 2 2 2 2 1 2 2 2 2 2 2 1 1 2 2 2 2 1 2 1 2 1 2 2 1 1 2 2 2 2 2 1 2 2 2 2
[139] 1 2 2 2 1 2 2 2 1 2 2 1

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

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"
[7] "size"         "iter"         "ifault"
>
```

Compare the Species label with the clustering result



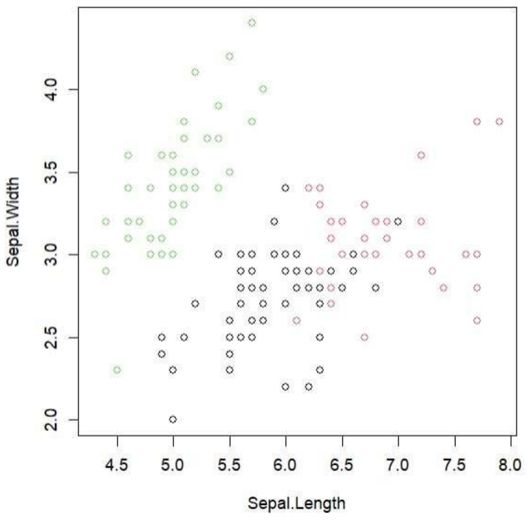
```
> #Compare the Species label with the clustering result
> table (iris$Species,kc$cluster)

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

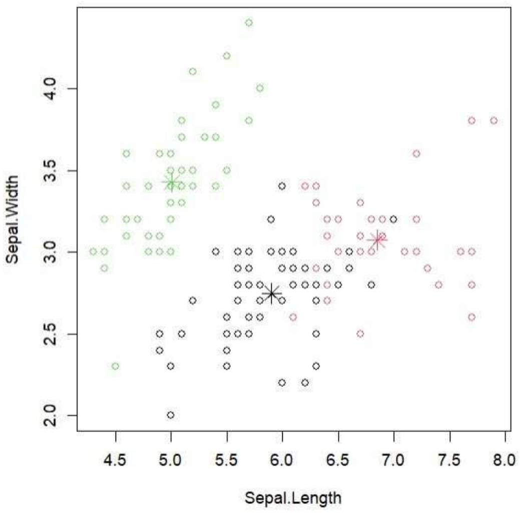
Plot the clusters and their centre



```
> # Plot the clusters and their centers
> 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:





PRACTICAL NO. 05

Aim: Perform Linear Regression on the given data warehouse data using R/Python.

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 y.
- data is the vector on which the formula will be applied.

Create Relationship Model & get the Coefficients

```
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.
relation <- lm(y~x)
print(relation)
```

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

```
Call:
lm(formula = y ~ x)
Coefficients:
(Intercept)          x
-38.4551            0.6746
```

Get the Summary of the Relationship

```
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
# Apply the lm() function.
relation <- lm(y~x)
print(summary(relation))
```

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

```
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.

Predict the weight of new persons

```
# The predictor vector.
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
# The resposne 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)
```

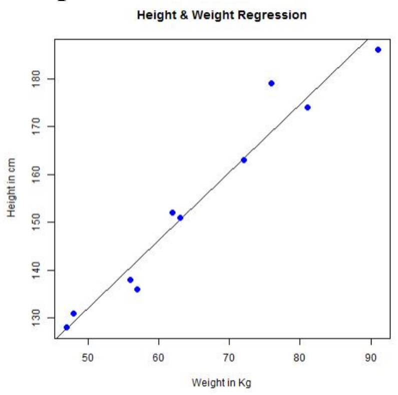
Result:

```
   1
76.22869
```

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()
```

Output:



PRACTICAL NO. 06

Aim: Perform the Logistic Regression on the given data warehouse data using R/Python.

Create Regression Model

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

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

Output:

```
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
```

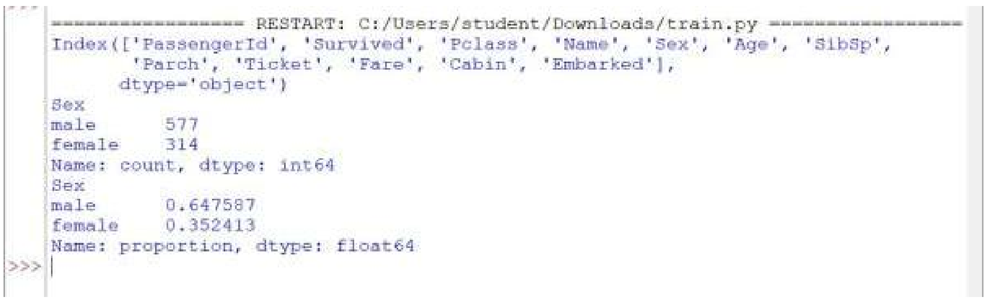
PRACTICAL NO. 07

Aim: Write a Python Program to read data from CSV file, perform sample data analysis and generate basic insights. (Use pandas as python library).

1) Male and Female ratio on the Titanic
Code:

```
import pandas as pd

#load the csv file
df = pd.read_csv('train.csv')

#Columns Names
print(df.columns)

#Count unique values in Sex column
print(df['Sex'].value_counts())

#Percentage of male and female passengers
print(df['Sex'].value_counts(normalize=True))
```

Output:



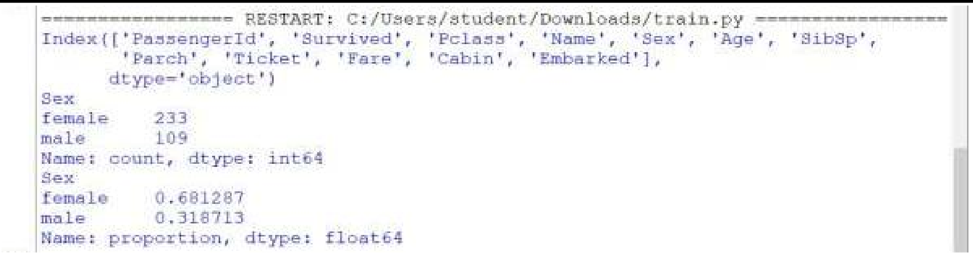
```
================== RESTART: C:/Users/student/Downloads/train.py ==================
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
Sex
male      577
female    314
Name: count, dtype: int64
Sex
male      0.647587
female    0.352413
Name: proportion, dtype: float64
>>> |
```

Insights:
The above analysis shows that 65% of people on Titanic were Male and 35% were female.

2) Surviving male to female ratio on the titanic
Code:

```
import pandas as pd
#load the csv file
df = pd.read_csv('train.csv')

#Column Names
print(df.columns)

#Count unique values in Sex column
print(df[df["Survived"] == 1]['Sex'].value_counts())

#Percentage of surviving male and female
passengers print(df[df["Survived"] == 1]['Sex'].value_counts(normalize=True))
```

Output:



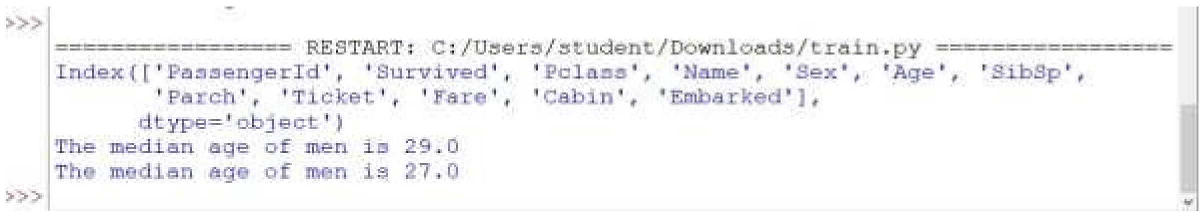
```
================= RESTART: C:/Users/student/Downloads/train.py =================
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
Sex
female    233
male      109
Name: count, dtype: int64
Sex
female    0.681287
male      0.318713
Name: proportion, dtype: float64
```

Insights:
The above analysis shows that 68% of surviving people on the Titanic were female.

3) Median age of each sex
Code:

```
import pandas as pd
#load the csv file
df = pd.read_csv('train.csv')

#median age of each sex
median_age_men=df[df['Sex']=='male']['Age'].median()
median_age_women=df[df['Sex']=='female']['Age'].median()

print(f"The median age of men is {median_age_men}")
print(f"The median age of women is {median_age_women}")
```

Output:



```
>>>
================= RESTART: C:/Users/student/Downloads/train.py =================
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
The median age of men is 29.0
The median age of men is 27.0
>>>
```

Insights:
The above analysis shows that median age of male was 29 whereas median age of female was 27.

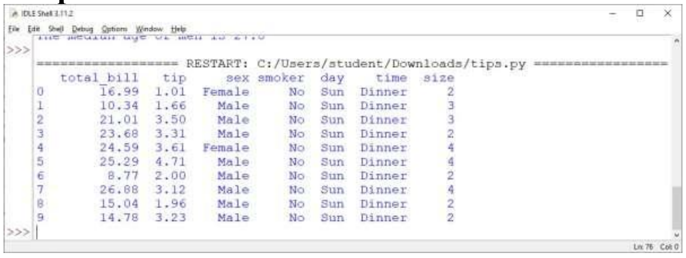
PRACTICAL NO. 08

Aim: 8a) Perform data visualization using Python on any sales data.

To install pandas library type, below command in the terminal.
pip3 install pandas

```
import pandas as pd
#reading the database
data = pd.read_csv("tips.csv")
#printing the top 10 rows.
print(data.head(10))
```

Output:

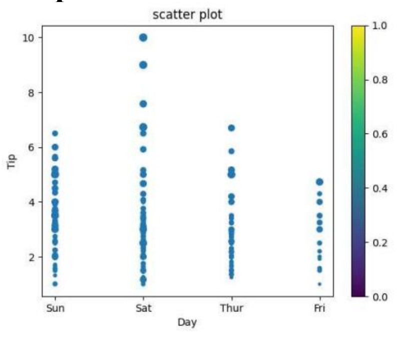


```
>>>
==================== RESTART: C:/Users/student/Downloads/tips.py ===================
   total_bill   tip     sex smoker  day    time  size
0       16.99  1.01  Female     No  Sun  Dinner     2
1       10.34  1.66    Male     No  Sun  Dinner     3
2       21.01  3.50    Male     No  Sun  Dinner     3
3       23.68  3.31    Male     No  Sun  Dinner     2
4       24.59  3.61  Female     No  Sun  Dinner     4
5       25.29  4.71    Male     No  Sun  Dinner     4
6        8.77  2.00    Male     No  Sun  Dinner     2
7       26.88  3.12    Male     No  Sun  Dinner     4
8       15.04  1.96    Male     No  Sun  Dinner     2
9       14.78  3.23    Male     No  Sun  Dinner     2
>>>
```

Scatter Plot

```
import pandas as pd
import matplotlib.pyplot as plt
#reading the database
data = pd.read_csv("tips.csv")
#Scatter plot with day against tip plt.scatter(data['day'], data['tip'],
c=data['size'], s=data['total_bill'])
#Adding Title to the plot
plt.title("Scatter plot")
#Setting the X and Y
labels plt.xlabel('Day')
plt.ylabel('Tip')
plt.colorbar() plt.show()
```

Output:

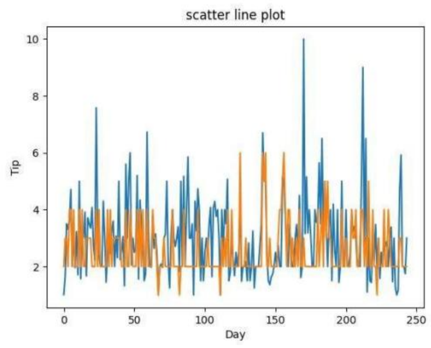


1) **Line Chart**

```
import pandas as pd
import matplotlib.pyplot as plt
#reading the database
data = pd.read_csv("tips.csv")
#Scatter plot with day against
tip plt.plot(data['tip'])
plt.plot(data['size'])
```

```
#Adding Title to the plot
plt.title("Scatter Plot")
#Setting the X and Y labels
plt.xlabel('Day')
plt.ylabel('Tip')
plt.show()
```

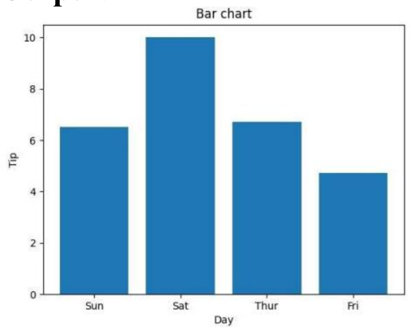
Output:



2) **Bar Chart**

```
import pandas as pd
import matplotlib.pyplot as plt
#reading the database
data = pd.read_csv("tips.csv")
#Bar chart with day against tip
plt.bar(data['day'], data['tip'])
plt.title("Bar Chart")
#Setting the X and Y labels
plt.xlabel('Day')
plt.ylabel('Tip')
#Adding the legends
plt.show()
```

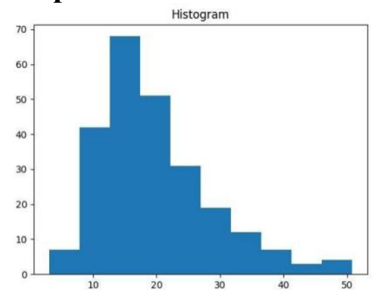
Output:



3) **Histogram**

```
import pandas as pd
import matplotlib.pyplot as plt
#reading the database
data = pd.read_csv("tips.csv")
#histogram of total_bills
plt.hist(data['total_bill'])
plt.title("Histogram")
#Adding the legends
plt.show()
```

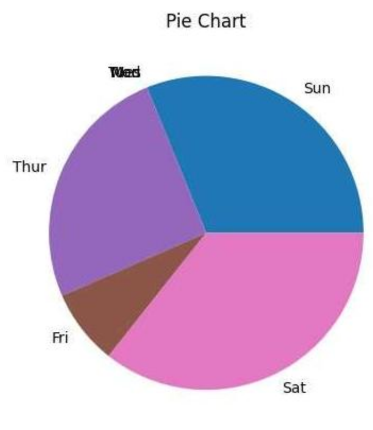
Output:



4) **Pie Chart**
5)

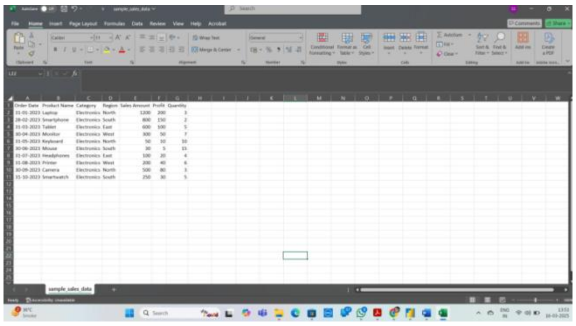
```
import pandas as pd
import matplotlib.pyplot as plt
import array
df = pd.read_csv("tips.csv")

count_Sun = df['day'].value_counts().get('Sun',0)
print("Occurrences of 'Sun':",count_Sun) count_Mon = df['day'].value_counts().get('Mon',0)

print("Occurrences of 'Mon':",count_Mon) count_Tues
= df['day'].value_counts().get('Tues',0)
print("Occurrences of 'Tues':",count_Tues) count_Wed
= df['day'].value_counts().get('Wed',0)
print("Occurrences of 'Wed':",count_Wed) count_Thu =
df['day'].value_counts().get('Thu',0)
print("Occurrences of 'Thu':",count_Thu) count_Fri =
df['day'].value_counts().get('Fri',0)
print("Occurrences of 'Fri':",count_Fri) count_Sat =
df['day'].value_counts().get('Sat',0)
print("Occurrences of 'Sat':",count_Sat)
mylabel=["Sun","Mon","Tue","Wed","Thu","Fri","Sat"]
e=(0.2, 0, 0, 0, 0, 0, 0)
arr=array.array('i',[count_Sun,count_Mon,count_Tue,count_Wed,count_Thu,count_Fri,count_Sat])
plt.pie(arr,explode=e,labels=mylabel)
#Title to the plot
plt.title("Pie Chart")
plt.show()
```

Output:



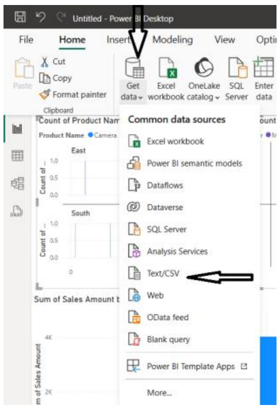
Aim: 8b) Perform data visualization using Power BI on any sales data

Step 1:- Download & Install Power BI
Step 2:- create a sample data



Order Date	Product Name	Category	Region	Sales Amount	Profit	Quantity
31-01-2023	Laptop	Electronics	North	1200	200	3
28-02-2023	Smartphone	Electronics	South	800	150	2
31-03-2023	Tablet	Electronics	East	600	100	5
30-04-2023	Monitor	Electronics	West	300	50	7
31-05-2023	Keyboard	Electronics	North	50	10	10
30-06-2023	Mouse	Electronics	South	30	5	15
31-07-2023	Headphones	Electronics	East	100	20	4
31-08-2023	Printer	Electronics	West	200	40	6
30-09-2023	Camera	Electronics	North	500	80	3
31-10-2023	Smartwatch	Electronics	South	250	30	5

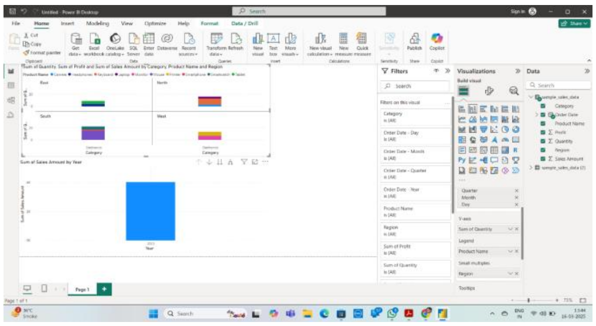
Step 3 :- Import the Sales Dataset
- Click "Home" > "Get Data".
- Choose your data source:
  - o Excel (XLSX/CSV)
  - o SQL Server
  - o Online Services (Google Sheets, SharePoint, etc.)
- Browse and select the dataset.
- Click "Load" to import.



Step 4 :- Clean & Transform Data (Power Query)
Step 5 :- Create Data Visualizations
- Click on "Line Chart" in the "Visualizations" pane.
- Drag Order Date to the X-axis.
- Drag Sales Amount to the Y-axis.
- Customize the chart (format labels, add title, etc.).

Step 6:- Add Filters & Interactivity

Output:



PRACTICAL NO. 09

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

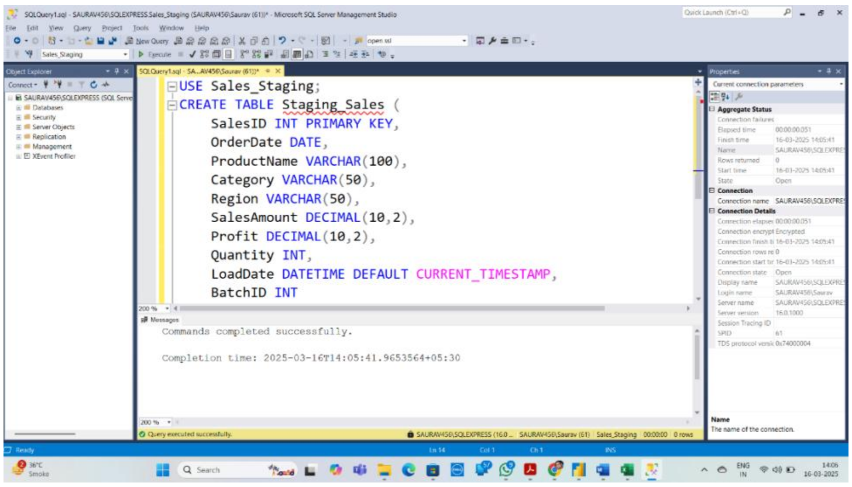
Step 1:- Create a Staging Database

First, create a staging database to store raw sales data.
CREATE DATABASE Sales_Staging;
USE Sales_Staging;

Step 2:- Create Staging Tables

Create tables that match the structure of raw sales data but include additional fields like load date and batch ID.

```
CREATE TABLE Staging_Sales (
        SalesID INT PRIMARY KEY,
        OrderDate DATE,
        ProductName VARCHAR(100),
        Category VARCHAR(50),
        Region VARCHAR(50),
        SalesAmount DECIMAL(10,2),
        Profit DECIMAL(10,2),
        Quantity INT,
        LoadDate DATETIME DEFAULT CURRENT_TIMESTAMP,
        BatchID INT );
```



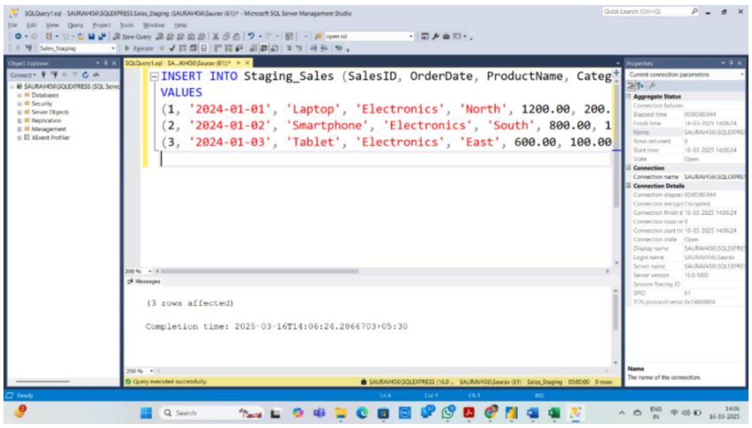
```
USE Sales_Staging;
CREATE TABLE Staging_Sales (
    SalesID INT PRIMARY KEY,
    OrderDate DATE,
    ProductName VARCHAR(100),
    Category VARCHAR(50),
    Region VARCHAR(50),
    SalesAmount DECIMAL(10,2),
    Profit DECIMAL(10,2),
    Quantity INT,
    LoadDate DATETIME DEFAULT CURRENT_TIMESTAMP,
    BatchID INT
```

```
Commands completed successfully.

Completion time: 2025-03-16T14:05:41.9653564+05:30
```

Query executed successfully.

Step 3:- Load Raw Data into the Staging Table
Simulating data load from a CSV file, API, or external source:

```
INSERT INTO Staging_Sales (SalesID, OrderDate, ProductName, Category, Region, SalesAmount, Profit, Quantity, BatchID) VALUES
(1, '2024-01-01', 'Laptop', 'Electronics', 'North', 1200.00, 200.00, 3, 101),
(2, '2024-01-02', 'Smartphone', 'Electronics', 'South', 800.00, 150.00, 2, 101),
(3, '2024-01-03', 'Tablet', 'Electronics', 'East', 600.00, 100.00, 5, 101);
```



```
INSERT INTO Staging_Sales (SalesID, OrderDate, ProductName, Categ
VALUES
(1, '2024-01-01', 'Laptop', 'Electronics', 'North', 1200.00, 200.
(2, '2024-01-02', 'Smartphone', 'Electronics', 'South', 800.00, 1
(3, '2024-01-03', 'Tablet', 'Electronics', 'East', 600.00, 100.00
```

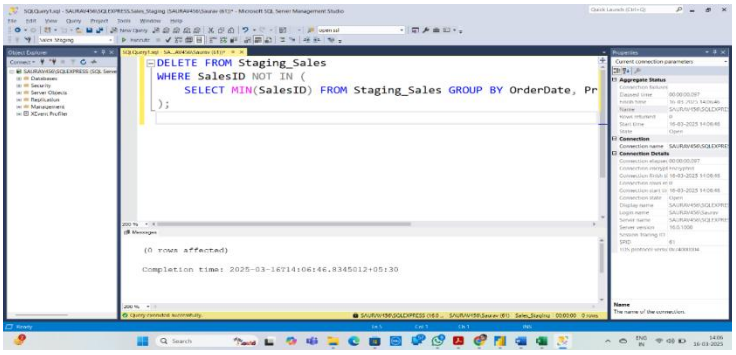
```
(3 rows affected)

Completion time: 2025-03-16T14:06:24.2866703+05:30
```

Query executed successfully.

Step 4:- Perform Data Cleansing & Transformation

• **Remove Duplicates**

```
DELETE FROM Staging_Sales
WHERE SalesID NOT IN (
        SELECT MIN(SalesID) FROM Staging_Sales GROUP BY OrderDate, ProductName,
        Region
);
```



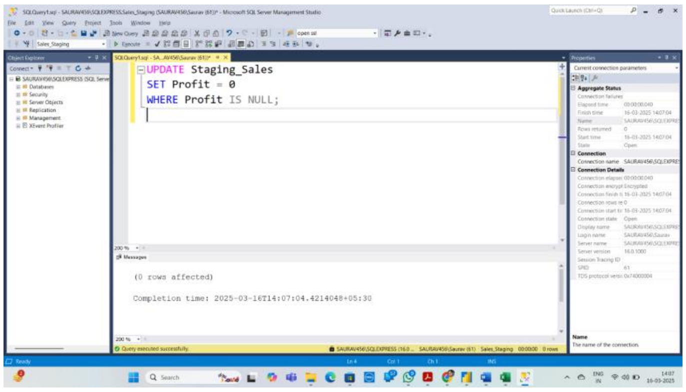
```
DELETE FROM Staging_Sales
WHERE SalesID NOT IN (
    SELECT MIN(SalesID) FROM Staging_Sales GROUP BY OrderDate, Pr
);
```

```
(0 rows affected)

Completion time: 2025-03-16T14:06:46.8345012+05:30
```

Query executed successfully.

• **Handle Null Values**

```
UPDATE Staging_Sales
SET Profit = 0
WHERE Profit IS NULL;
```



```
UPDATE Staging_Sales
SET Profit = 0
WHERE Profit IS NULL;
```

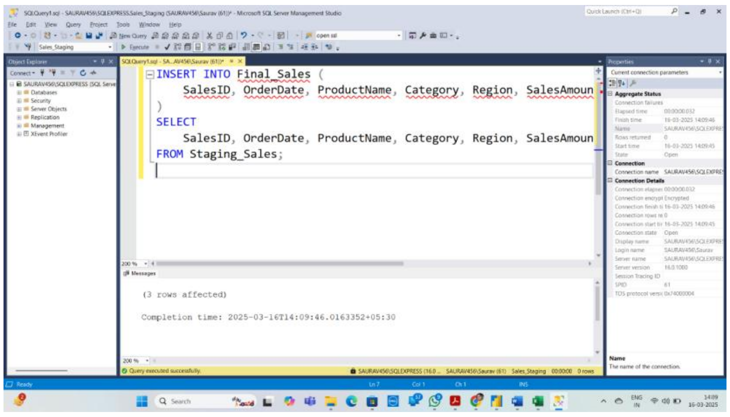
```
(0 rows affected)

Completion time: 2025-03-16T14:07:04.4214048+05:30
```

Query executed successfully.

Step 5:- Transfer Clean Data to the Final Sales Table
Move the cleaned data into the Data Warehouse (DWH).

```
INSERT INTO Final_Sales (
        SalesID, OrderDate, ProductName, Category, Region, SalesAmount, Profit, Quantity
)
```

```
SELECT
        SalesID, OrderDate, ProductName, Category, Region, SalesAmount, Profit, Quantity
FROM Staging_Sales;
```

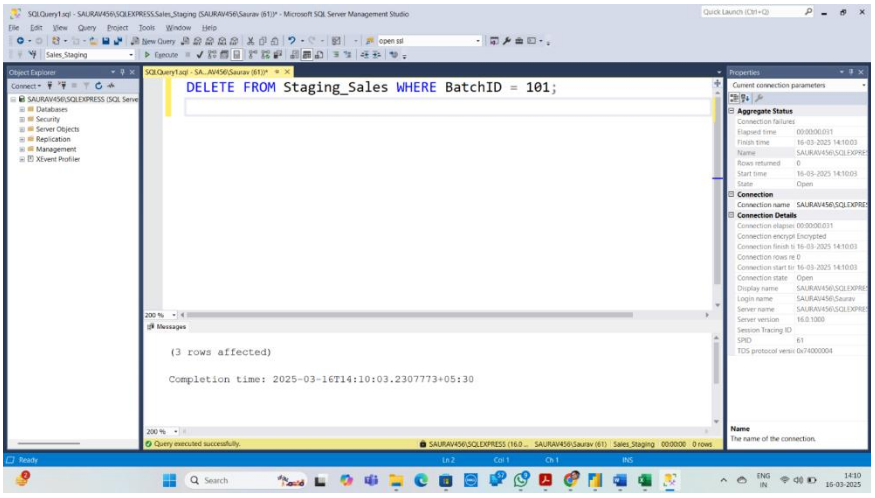


```
INSERT INTO Final_Sales (
    SalesID, OrderDate, ProductName, Category, Region, SalesAmoun
)
SELECT
    SalesID, OrderDate, ProductName, Category, Region, SalesAmoun
FROM Staging_Sales;
```

```
(3 rows affected)

Completion time: 2025-03-16T14:09:46.0163352+05:30
```

<u>Step 6</u>:- Archive or Delete Processed Data
Once data is loaded, either archive it or delete it from the staging area.

```
DELETE FROM Staging_Sales WHERE BatchID = 101;
```



```
DELETE FROM Staging_Sales WHERE BatchID = 101;
```

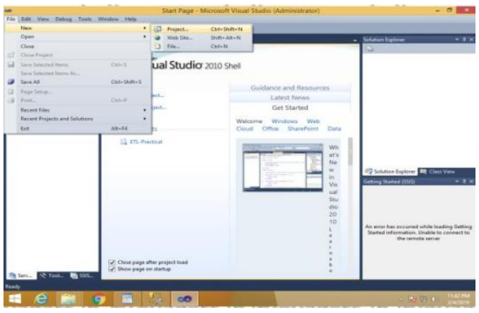
```
(3 rows affected)

Completion time: 2025-03-16T14:10:03.2307773+05:30
```

PRACTICAL NO. 10

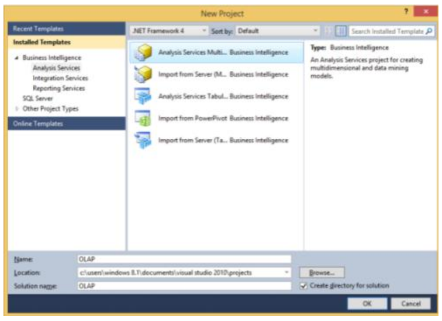
Aim: Create the cube with suitable dimension and fact tables based on ROLAP, MOLAP and HOLAP model.

Step 1: Creating Data Warehouse
CREATED IN PRACTICAL 9

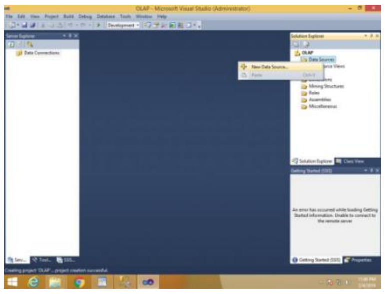
Step 2: Start SSDT environment and create New Data Source
Go to Sql Server Data Tools --> Right click and run as administrator
Click on File → New → Project



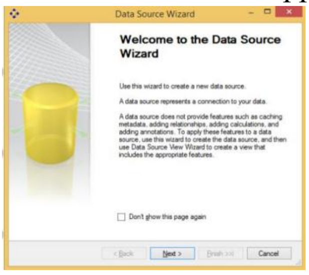
In Business Intelligence → Analysis Services Multidimensional and Data Mining models → appropriate project name → click OK


New Project
Recent Templates
Installed Templates
Business Intelligence
Analysis Services
Integration Services
Reporting Services
SQL Server
Other Project Types
Online Templates
.NET Framework 4
Sort by: Default
Analysis Services Multi... Business Intelligence
Import from Server (M... Business Intelligence
Analysis Services Tabul... Business Intelligence
Import from PowerPivot Business Intelligence
Import from Server (Ta... Business Intelligence
Type: Business Intelligence
An Analysis Services project for creating multidimensional and data mining models.
Name: OLAP
Location: c:\users\windows 8.1\documents\visual studio 2010\projects
Solution name: OLAP
Browse...
Create directory for solution
OK
Cancel

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



Data Source Wizard appears



Data Source Wizard

Welcome to the Data Source Wizard

Use this wizard to create a new data source.

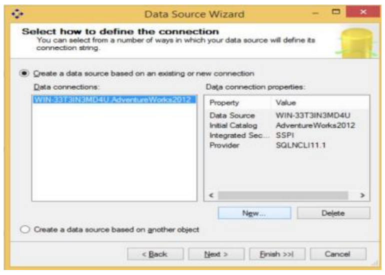
A data source represents a connection to your data.

A data source does not provide features such as caching metadata, adding relationships, adding calculations, and adding annotations. To apply these features to a data source, use this wizard to create the data source, and then use Data Source View Wizard to create a view that includes the appropriate features.

☐ Don't show this page again

< Back | Next > | Finish >>| | Cancel

Click on New



Data Source Wizard

Select how to define the connection
You can select from a number of ways in which your data source will define its connection string.

● Create a data source based on an existing or new connection

Data connections:
WIN-33T3IN3MD4U.AdventureWorks2012

Data connection properties:

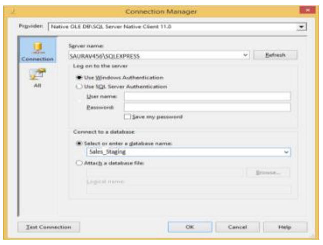
Property	Value
Data Source	WIN-33T3IN3MD4U
Initial Catalog	AdventureWorks2012
Integrated Sec...	SSPI
Provider	SQLNCLI11.1

New... | Delete

○ Create a data source based on another object

< Back | Next > | Finish >>| | Cancel

Select Server Name → select Use SQL Server Authentication → Select or enter a database name (Sales_Staging)

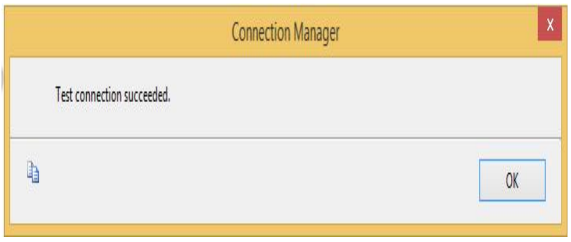

Connection Manager

Provider: Native OLE DB\SQL Server Native Client 11.0

Server name: SAURAV456\SQLEXPRESS

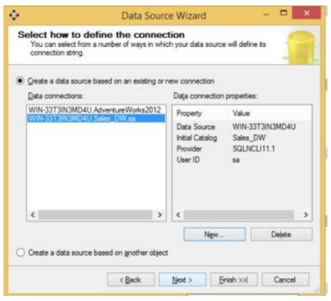
Log on to the server
- (●) Use Windows Authentication
- ( ) Use SQL Server Authentication
- User name:
- Password:
- [ ] Save my password

Connect to a database
- (●) Select or enter a database name: Sales_Staging
- ( ) Attach a database file:
- Logical name:


Connection Manager

Test connection succeeded.

Click ok


Data Source Wizard

Select how to define the connection
You can select from a number of ways in which your data source will define its connection string.

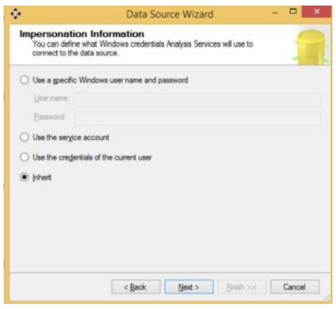
- (●) Create a data source based on an existing or new connection
- ( ) Create a data source based on another object

Data connections:
- WIN-33T3IN3MD4U AdventureWorks2012
- WIN-33T3IN3MD4U Sales_DW.sa

Data connection properties:

Property	Value
Data Source	WIN-33T3IN3MD4U
Initial Catalog	Sales_DW
Provider	SQLNCLI11.1
User ID	sa

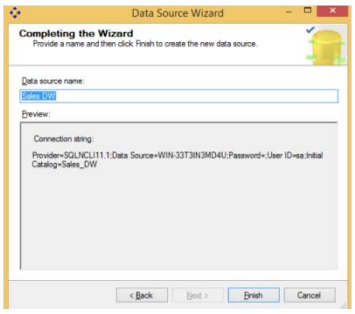
Select Inherit → Next


Data Source Wizard

Impersonation Information
You can define what Windows credentials Analysis Services will use to connect to the data source.

- ( ) Use a specific Windows user name and password
- User name:
- Password:
- ( ) Use the service account
- ( ) Use the credentials of the current user
- (●) Inherit

Click Finish


Data Source Wizard

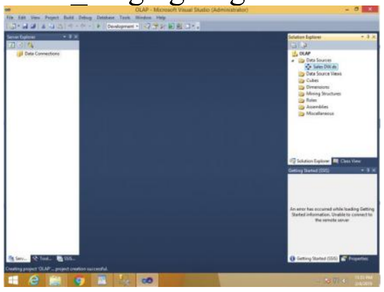
Completing the Wizard
Provide a name and then click Finish to create the new data source.

Data source name: Sales DW

Preview:

Connection string:
Provider=SQLNCLI11.1;Data Source=WIN-33T3IN3MD4U;Password=;User ID=sa;Initial Catalog=Sales_DW

Sales_Staging.ds gets created under Data Sources in Solution Explorer

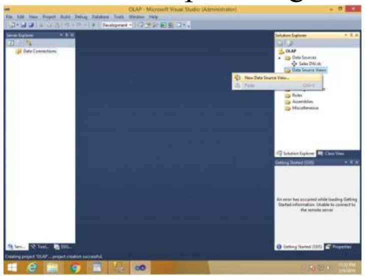


An error has occurred while loading Getting Started information. Unable to connect to the remote server

Creating project 'OLAP'... project creation successful.

<u>Step 3</u>: Creating New Data Source View

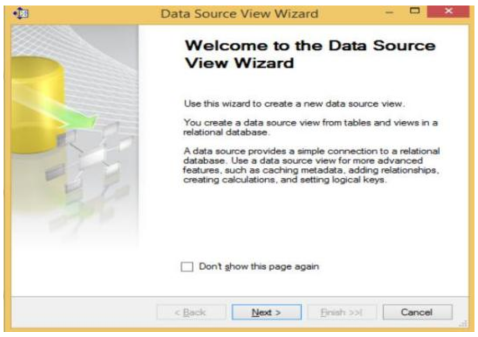
In Solution explorer right click on Data Source View → Select New Data Source View



An error has occurred while loading Getting Started information. Unable to connect to the remote server

Creating project 'OLAP'... project creation successful.

Click Next


Data Source View Wizard

Welcome to the Data Source View Wizard

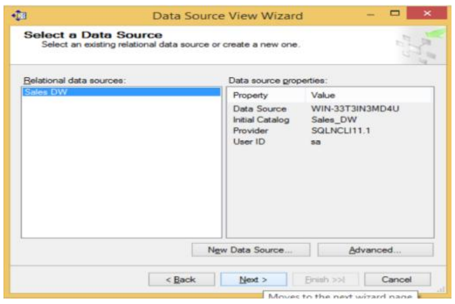
Use this wizard to create a new data source view.

You create a data source view from tables and views in a relational database.

A data source provides a simple connection to a relational database. Use a data source view for more advanced features, such as caching metadata, adding relationships, creating calculations, and setting logical keys.

[ ] Don't show this page again

Click Next


Data Source View Wizard

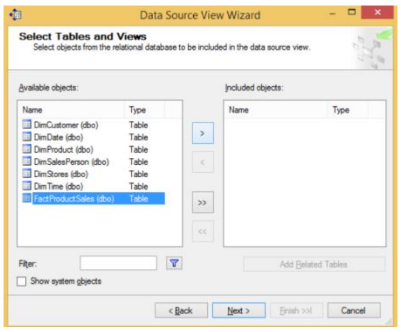
Select a Data Source
Select an existing relational data source or create a new one.

Relational data sources:
- Sales DW

Data source properties:

Property	Value
Data Source	WIN-33T3IN3MD4U
Initial Catalog	Sales_DW
Provider	SQLNCLI11.1
User ID	sa

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


Data Source View Wizard

Select Tables and Views
Select objects from the relational database to be included in the data source view.

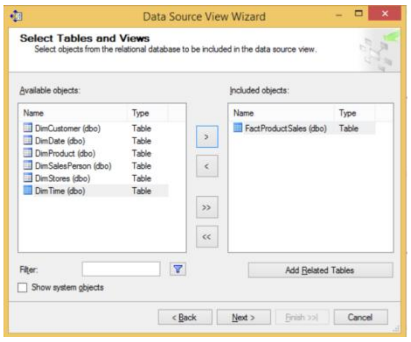
Available objects:

Name	Type
DimCustomer (dbo)	Table
DimDate (dbo)	Table
DimProduct (dbo)	Table
DimSalesPerson (dbo)	Table
DimStores (dbo)	Table
DimTime (dbo)	Table
FactProductSales (dbo)	Table

Included objects:

Name	Type

Filter:
Show system objects


Data Source View Wizard

Select Tables and Views
Select objects from the relational database to be included in the data source view.

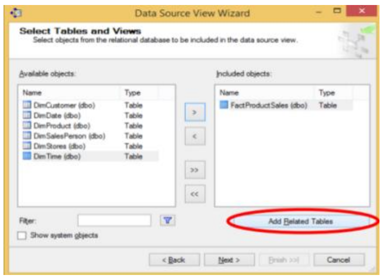
Available objects:

Name	Type
DimCustomer (dbo)	Table
DimDate (dbo)	Table
DimProduct (dbo)	Table
DimSalesPerson (dbo)	Table
DimStores (dbo)	Table
DimTime (dbo)	Table

Included objects:

Name	Type
FactProductSales (dbo)	Table

Click on Add Related Tables


Data Source View Wizard

Select Tables and Views
Select objects from the relational database to be included in the data source view.

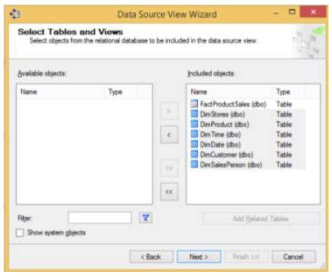
Available objects:

Name	Type
DimCustomer (dbo)	Table
DimDate (dbo)	Table
DimProduct (dbo)	Table
DimSalesPerson (dbo)	Table
DimStores (dbo)	Table
DimTime (dbo)	Table

Included objects:

Name	Type
FactProductSales (dbo)	Table

Click Next


Data Source View Wizard

Select Tables and Views
Select objects from the relational database to be included in the data source view.

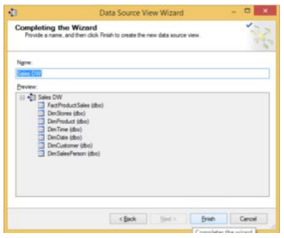
Available objects:

Name	Type

Included objects:

Name	Type
FactProductSales (dbo)	Table
DimStores (dbo)	Table
DimProduct (dbo)	Table
DimTime (dbo)	Table
DimDate (dbo)	Table
DimCustomer (dbo)	Table
DimSalesPerson (dbo)	Table

Click Finish

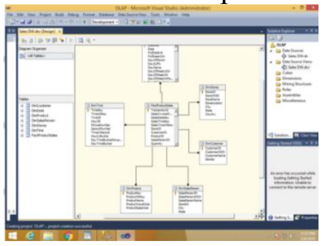

Data Source View Wizard

Completing the Wizard
Provide a name, and then click Finish to create the new data source view.

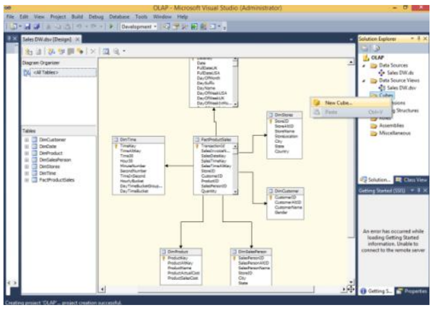
Name:
Sales DW

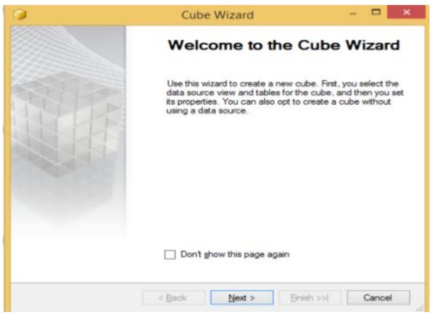
Preview:
- Sales DW
  - FactProductSales (dbo)
  - DimStores (dbo)
  - DimProduct (dbo)
  - DimTime (dbo)
  - DimDate (dbo)
  - DimCustomer (dbo)
  - DimSalesPerson (dbo)

Sales_Staging.dsv appears in Data Source Views in Solution Explorer.



Step 4: Creating new cube
Right click on Cubes → New Cube



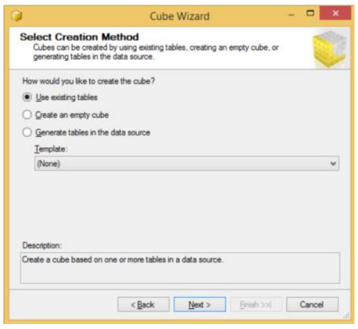

Cube Wizard

Welcome to the Cube Wizard

Use this wizard to create a new cube. First, you select the data source view and tables for the cube, and then you set its properties. You can also opt to create a cube without using a data source.

Don't show this page again

Select Use existing tables in Select Creation Method → Next


Cube Wizard

Select Creation Method
Cubes can be created by using existing tables, creating an empty cube, or generating tables in the data source.

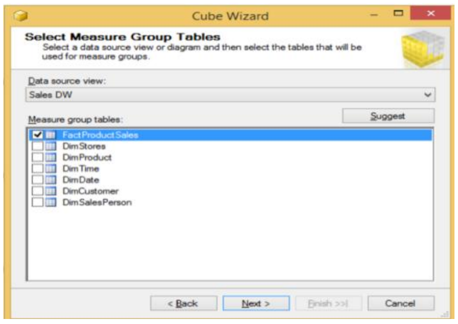
How would you like to create the cube?

- [✓ Use existing tables]
- Create an empty cube
- Generate tables in the data source

Template:
(None)

Description:
Create a cube based on one or more tables in a data source.

In Select Measure Group Tables → Select FactProductSales → Click Next


Cube Wizard

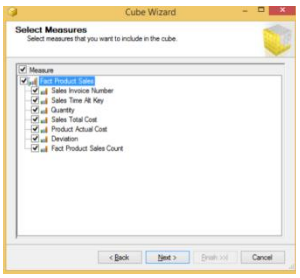
Select Measure Group Tables
Select a data source view or diagram and then select the tables that will be used for measure groups.

Data source view:
Sales DW

Measure group tables:

- [✓] FactProductSales
- [ ] DimStores
- [ ] DimProduct
- [ ] DimTime
- [ ] DimDate
- [ ] DimCustomer
- [ ] DimSalesPerson

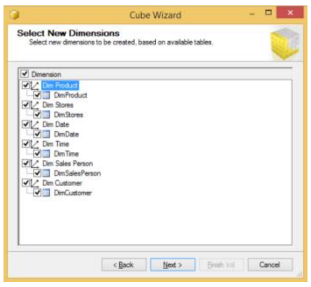
In Select Measures →
check all measures → Next


Cube Wizard

Select Measures
Select measures that you want to include in the cube.

- [✓] Measure
  - [✓] Fact Product Sales
    - [✓] Sales Invoice Number
    - [✓] Sales Time Alt Key
    - [✓] Quantity
    - [✓] Sales Total Cost
    - [✓] Product Actual Cost
    - [✓] Deviation
    - [✓] Fact Product Sales Count

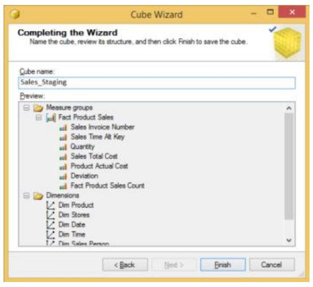
In Select New Dimensions → Check all
Dimensions → Next


Cube Wizard

Select New Dimensions
Select new dimensions to be created, based on available tables.

- [✓] Dimension
  - [✓] Dim Product
    - [✓] DimProduct
  - [✓] Dim Stores
    - [✓] DimStores
  - [✓] Dim Date
    - [✓] DimDate
  - [✓] Dim Time
    - [✓] DimTime
  - [✓] Dim Sales Person
    - [✓] DimSalesPerson
  - [✓] Dim Customer
    - [✓] DimCustomer

Click on Finish


Cube Wizard

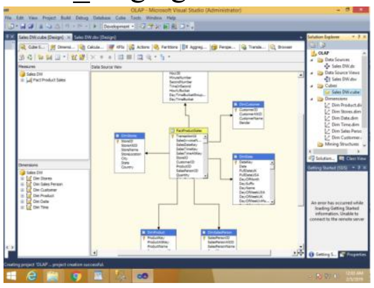
Completing the Wizard
Name the cube, review its structure, and then click Finish to save the cube.

Cube name: Sales_Staging

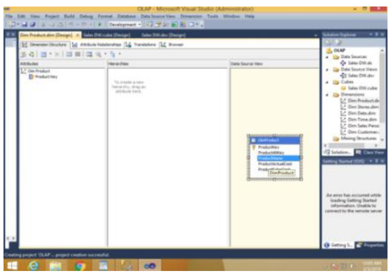
Preview:

- Measure groups
  - Fact Product Sales
    - Sales Invoice Number
    - Sales Time Alt Key
    - Quantity
    - Sales Total Cost
    - Product Actual Cost
    - Deviation
    - Fact Product Sales Count
- Dimensions
  - Dim Product
  - Dim Stores
  - Dim Date
  - Dim Time
  - Dim Sales Person

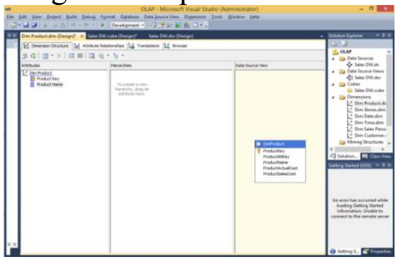
Sales_Staging.cube is created



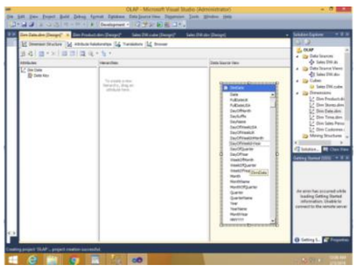
<u>Step 5</u>: Dimension Modification
In dimension tab → Double Click Dim Product.dim

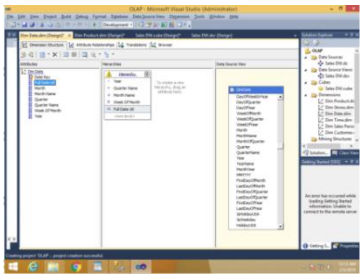


Drag and Drop Product Name from Table in Data Source View and Add in Attribute Pane at left side



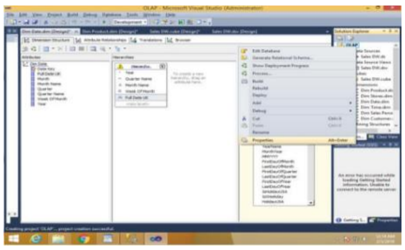
<u>Step 6</u>: Creating Attribute Hierarchy in Date Dimension
Double click On Dim Date dimension -> Drag and Drop Fields from Table shown in Data Source View to Attributes-> Drag and Drop attributes from leftmost pane of attributes to middle pane of Hierarchy.
Drag fields in sequence from Attributes to Hierarchy window (Year, Quarter Name, Month Name, Week of the Month, Full Date UK)



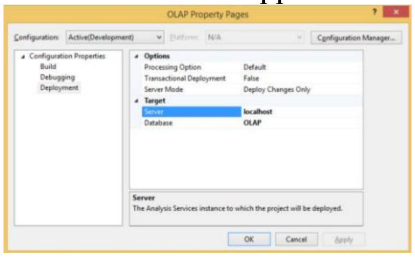


Step 7: Deploy Cube .

Right click on Project name → Properties



This window appears



OLAP Property Pages

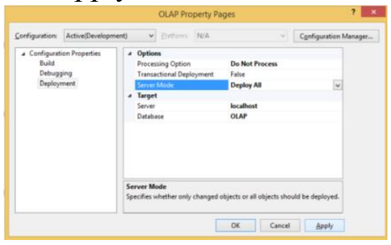
Configuration: Active(Development) Platform: N/A Configuration Manager...

- Configuration Properties
  - Build
  - Debugging
  - Deployment

Options	
Processing Option	Default
Transactional Deployment	False
Server Mode	Deploy Changes Only
Target	
Server	**localhost**
Database	**OLAP**

Server
The Analysis Services instance to which the project will be deployed.

Do following changes and click
 on Apply & ok



OLAP Property Pages

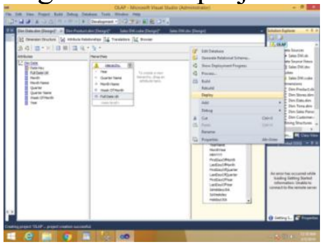
Configuration: Active(Development) Platform: N/A Configuration Manager...

- Configuration Properties
  - Build
  - Debugging
  - Deployment

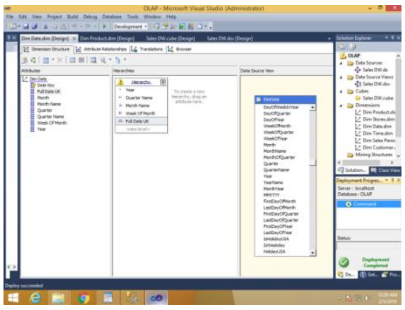
Options	
Processing Option	**Do Not Process**
Transactional Deployment	False
Server Mode	**Deploy All**
Target	
Server	**localhost**
Database	**OLAP**

Server Mode
Specifies whether only changed objects or all objects should be deployed.

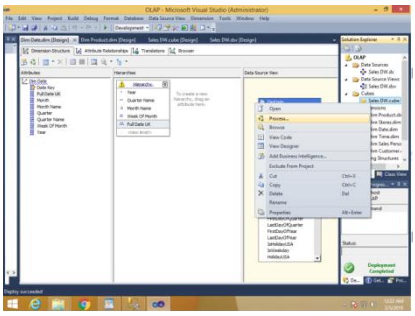
Right click on project name →Deploy



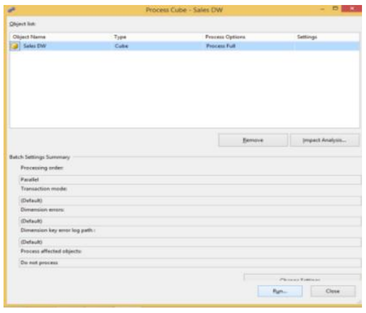
Deployment successful



To process cube right click on Sales_DW.cube
→ Process



Click run



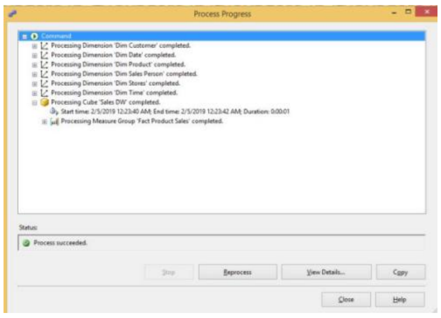
Process Cube - Sales DW

Object list:

Object Name	Type	Process Options	Settings
Sales DW	Cube	Process Full	

Batch Settings Summary

Processing order:
Parallel
Transaction mode:
(Default)
Dimension errors:
(Default)
Dimension key error log path :
(Default)
Process affected objects:
Do not process



Process Progress

```
Command
  Processing Dimension 'Dim Customer' completed.
  Processing Dimension 'Dim Date' completed.
  Processing Dimension 'Dim Product' completed.
  Processing Dimension 'Dim Sales Person' completed.
  Processing Dimension 'Dim Stores' completed.
  Processing Dimension 'Dim Time' completed.
  Processing Cube 'Sales DW' completed.
    Start time: 2/5/2019 12:23:40 AM; End time: 2/5/2019 12:23:42 AM; Duration: 0:00:01
    Processing Measure Group 'Fact Product Sales' completed.
```

Status:
Process succeeded.

Browse the cube for analysis in solution explorer