Practical 1: Import the legacy data from different sources such as (Excel, SqlServer, Oracle etc.) and load in the target system.

Importing Excel Data

1) Launch Power BI Desktop.

2) From the Home ribbon, select Get Data - O Data Feed

3) Paste the link

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

4) Select orders table and click load

Practical 2: Perform the Extraction Transformation and Loading (ETL) process to construct the database in the Sqlserver / Power BI.

Step1:

Home- OData Feed

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

Select Orders and Products and click transform data

Step2:

Remove all the columns and keep **Productid, productname, quantityperunit, unitinstock**

Step3:

Change the data type of unitsinstock column to whole number

Step4:

Select order_details column

Click on the right icon

Click on expand

Ok

Step5:

Add column – Custom column - give column name (linetotal) – enter formula

[Order_Details.UnitPrice]*[Order_Details.Quantity]

Step6:

Rename the column name as Total

Step7:

Left click and drag the Total column beside the shipcountry column

Step8:

Close and apply

Step9:

Select data, the tables will be displayed

Step10:

Click manage relationships

Step11:

Click edit and click ok

Step12:

Click model and the ER diagram will be displayed

Practical 3: Data Visualization from ETL Process

Step1:

From Fields drag and drop the products_name on the canvas

Drop Unitsinstock in the same table

Step2:

Select the table on the canvas and click on the clustered column chart

Step3:

Place ProductName in axis

Place Unitsinstock in value

Step4:

Go to the settings – security – check the last box

Step5:

Right click on Unitsinstock and click sum

Step6:

Drag and drop ordersdate and total

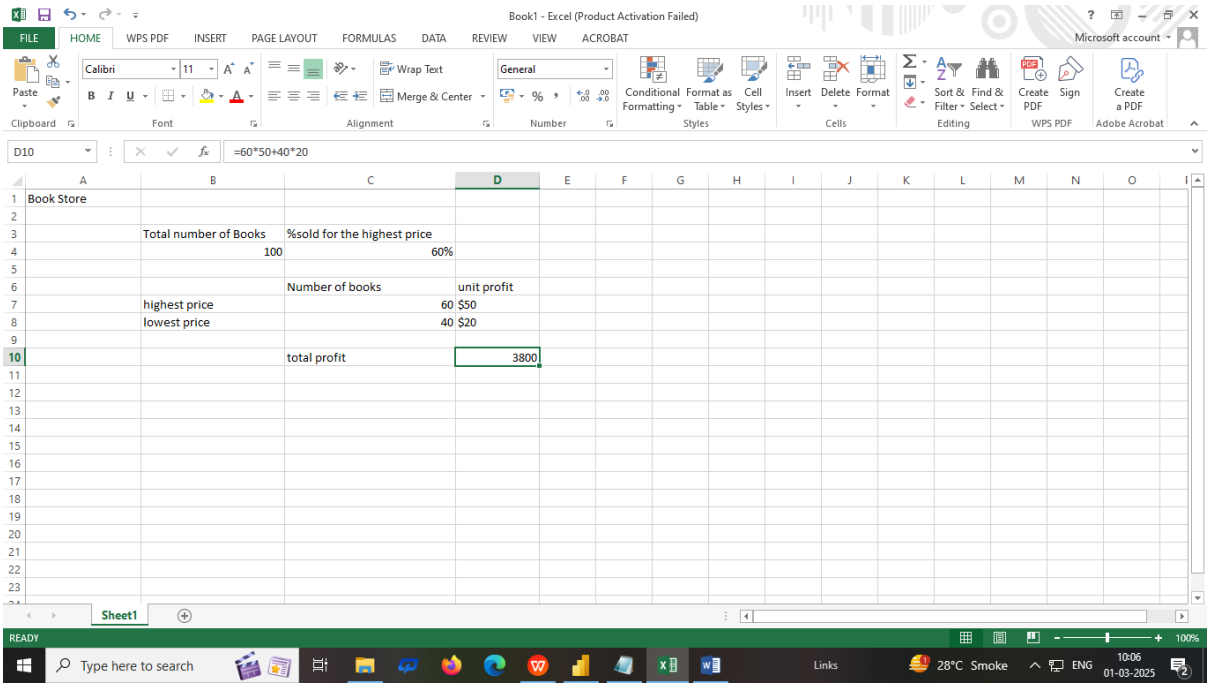
Select line chart

Drag and drop the total from legend to values and the line chart will be generated

Step7:

Drag and drop shipcountry on the canvas

Practical 4: Apply the what – if Analysis for data visualization. Design and generate necessary reports based on the data warehouse data.



1. Create the Excel workbook.
2. Add formula in formula bar of D10
   =60*50+40*20
3. On the Data tab, in the Forecast group, click What-If Analysis.
4. Click Scenario Manager.
5. The Scenario Manager dialog box appears.
6. Add a scenario by clicking on Add.
7. Type a scenario name as 60% highest and changing place to C4
8. Enter the corresponding value 0.6 and click on OK again.
9. Next, add 4 other scenarios (70%, 80%, 90% and 100%).

Practical 5: Implementation of Classification algorithm in R Programming.

```
# 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()
```

Practical 6: Practical Implementation of Decision Tree using R Tool

```
Step1: install.packages("party")

Step2:

# Load the party package. It will automatically load other dependent
packages.

library(party)

# Print some records from data set readingSkills.
print(head(readingSkills))

Step3:

# Load the party package. It will automatically load other dependent
packages.
library(party)

# Create the input data frame.
input.dat <- readingSkills[c(1:105),]

# Give the chart file a name.
png(file = "decision_tree.png")

# Create the tree.
output.tree <- ctree(
nativeSpeaker ~ age + shoeSize + score,

data = input.dat)

# Plot the tree.
plot(output.tree)

# Save the file.
dev.off()
```

Practical 7: Perform the data clustering using clustering algorithm.

Step 1:

```
newiris <- iris
newiris$Species <- NULL
(kc <- kmeans(newiris,3))
```

Step2:

```
table (iris$Species,kc$cluster)
```

Step3:

```
plot(newiris[c("Sepal.Length","Sepal.Width")],col=kc$cluster)
```

points(kc$centers[,c("Sepal.Length","Sepal.Width")],col=1:3,pch=8,cex=2)

Practical 8: Prediction Using Linear Regression

Step1:

```
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)
```

Step2:

```
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))
```

Step3:

```
# 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)
```

Step4:

```
# 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()
```

Practical 9: Data Analysis using Time Series Analysis

```
# Get the data points in form of a R vector.
rainfall <-
c(799,1174.8,865.1,1334.6,635.4,918.5,685.5,998.6,784.2,985,88
2.8,1071)

# 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()
```