Sr. No.	Practical	Date	Sign
1	Import the legacy data from different sources such as (Excel, SqlServer, Oracle etc.) and load in the target system.	25/01/2024	
2	Perform the Extraction Transformation and Loading (ETL) process to construct the database in the Sqlserver.	25/01/2024	
3	Data Visualization From Process	01/02/2024	
5	Implementation of Classification algorithm in R Programming	11/01/2024	
6	Apply what-IF analysis for data visualization. Design and generate necessary reports.	18/01/2024	
7	Implementation of Decision Tree using R Tool	18/01/2024	
8	K-means clustering using R	18/01/2024	
9	Prediction Using Linear Regression	18/01/2024	
10	Data Analysis using Time Series Analysis	18/01/2024	
11	Import the data warehouse data in Microsoft Excel and create the Pivot table and Pivot Chart.	08/03/2024	
12	Data Analysis and Visualization using Advanced Excel	21/03/24	

Practical – 1

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

Part 1: Import Data from Excel

Steps:

Step 1 – Open Power BI.

Step 2 – Click on Get Data and select Excel Workbook.

Step 3 – Select the excel file you want to open and click open.

Step 4 – Power BI analyzes the excel file and shows you all the sheets within. Select the sheet/sheets you want to load and click on Load.

Step 5 – You can click on the Data View to see the data you've imported.

Part 2: Import Data from OData Feed

Steps:

Step 1 – Open Power BI. Click on Get Data and select Excel Workbook.

Step 2 – Paste the following link in the textbox and click OK.

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

Step 3 – The Navigator window will be opened where you can see all the tables, select the tables you want to load and then click on Load.

Step 4 – You can again click on the Data View and Select the Data you want to see from the right hand side.

EmployeeID	LastName	FirstName	Title	TitleOfCourtesy	BirthDate	HireDate	Address	City	Region	PostalCode	Countr
1	Davolio	Nancy	Sales Representative	Ms.	12/8/1948 12:00:00 AM	5/1/1992 12:00:00 AM	507 - 20th Ave. E. Apt. 2A	Seattle	WA	98122	USA
2	Fuller	Andrew	Vice President, Sales	Dr.	2/19/1952 12:00:00 AM	8/14/1992 12:00:00 AM	908 W. Capital Way	Tacoma	WA	98401	USA
3	Leverling	Janet	Sales Representative	Ms.	8/30/1963 12:00:00 AM	4/1/1992 12:00:00 AM	722 Moss Bay Blvd.	Kirkland	WA	98033	USA
4	Peacock	Margaret	Sales Representative	Mrs.	9/19/1937 12:00:00 AM	5/3/1993 12:00:00 AM	4110 Old Redmond Rd.	Redmond	WA	98052	USA
5	Buchanan	Steven	Sales Manager	Mr.	3/4/1955 12:00:00 AM	10/17/1993 12:00:00 AM	14 Garrett Hill	London		SW1 8JR	UK
6	Suyama	Michael	Sales Representative	Mr.	7/2/1963 12:00:00 AM	10/17/1993 12:00:00 AM	Coventry House Miner Rd.	London		EC2 7JR	UK
7	King	Robert	Sales Representative	Mr.	5/29/1960 12:00:00 AM	1/2/1994 12:00:00 AM	Edgeham Hollow Winchester Way	London		RG1 9SP	UK
8	Callahan	Laura	Inside Sales Coordinator	Ms.	1/9/1958 12:00:00 AM	3/5/1994 12:00:00 AM	4726 - 11th Ave. N.E.	Seattle	WA	98105	USA
9	Dodsworth	Anne	Sales Representative	Ms.	1/27/1966 12:00:00 AM	11/15/1994 12:00:00 AM	7 Houndstooth Rd.	London		WG2 7LT	UK

Practical – 2

Aim: Perform the Extraction Transformation and Loading (ETL) process to construct the database in the Sqlserver /Power Bi

Part 1: ETL in Power BI

Steps:

Step 1 – Open Power BI. Click on Get Data, select OData Feed. Enter the following URL and click OK.

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

Step 2 – Select the Products and Orders Table and Click on Transform Data.

Step 3 – Power Query Editor is opened now to remove a column right click on the column you want to remove and click remove.

Step 4 – To change the data type select the column whose data type you want to change and in the home tab click on the data type option to change it to your desired data type.

Step 5 – To expand the Order_Details Table go to the orders table, go to the Order_Details Column and click on the Expand Column button, select the columns you want to expand and click OK.

Step 6 – To add a custom column go to the Add Column Tab and click on Custom Column there enter the new column name and enter the custom column formula by inserting the columns you want to take data from and then finally click OK.

Step 7 – To rename a column right click on the column you want to rename select the rename option and enter your desired name.

Step 8 – To reorder a column drag the column and drop it to the place you want it.

Step 9 – Now in the Home Tab click on Close & Apply.

Step 10 – Now in the Home Tab click on Manage Relationship. Power Bi will auto detect a relationship or you can create your own by clicking on new. Once a relationship is created click on close.

Step 11 – You can click on the Model View to see the relationship.

Practical – 3

AIM : Data Visualization from ETL Process

Power BI Desktop lets you create a variety of visualizations to gain insights from your data. You can build reports with multiple pages and each page can have multiple visuals. You can interact with your visualizations to help analyze and understand your data

In this task, you create a report based on the data previously loaded. You use the Fields pane to select the columns from which you create the visualizations.

Step 1: Create charts showing Units in Stock by Product and Total Sales by Year

1. Drag UnitsInStock from the Field pane (the Fields pane is along the right of the screen) onto a blank space on the canvas. A Table visualization is created. Next, drag ProductName to the Axis box, found in the bottom half of the Visualizations pane. Then we then select Sort By > UnitsInStock using the skittles in the top right corer of the visualization.

2. Drag OrderDate to the canvas beneath the first chart, then drag LineTotal (again, from the Fields pane) onto the visual, then select Line Chart. The following visualization is created.

3. Next, drag ShipCountry to a space on the canvas in the top right. Because you selected a geographic field, a map was created automatically. Now drag LineTotal to the Values field; the circles on the map for each country are now relative in size to the LineTotal for orders shipped to that country.

Step 2: Interact with your report visuals to analyze further

Power BI Desktop lets you interact with visuals that cross-highlight and filter each other to uncover further trends.

1. Click on the light blue circle centered in Canada. Note how the other visuals are filtered to show Stock (ShipCountry) and Total Orders (LineTotal) just for Canada.

Practical – 5

Aim: Implementation of Classification algorithm in R Programming.

Consider the annual rainfall details at a place starting from January 2014. We create an R time series object for a period of 12 months and plot it.

```
#Get the data points in form of a 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(2014,1),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

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

```
2014        Jan     Feb     Mar     Apr     May
        799.0   1174.8   865.1  1334.6   635.4
2014        Jun     Jul     Aug     Sep     Oct
        918.5    685.5   998.6   784.2   985.0
2014        Nov     Dec
        882.8  1071.0
```

Practical – 6

Aim: Apply the what-if analysis for data visualization. Design and generate necessary reports.

Goal Seek:

You have to sell sugar at the price of 40rs for 500kg , so its total will be 20,000rs , if we have given a target value of 30,000rs , as what will be the price of sugar if the total value is 30,000rs for 500kg. With the help of Goal Seek we will find out the price of sugar.

Step 1:

	A	B	C	D	E
1	Price	40			
2	Qty(kg)	500		Target	30000
3					
4	Total	20000			
5					

Step 2:

Output:

	A	B	C
1	Price	60	
2	Qty(kg)	500	
3			
4	Total	30000	
5			

Data Table :

In Goal seek we can check only for one price and one weight(kg), but what if we have to check the total value for different price and different weight all together at one place in the form of table. So with the help of Data Table, we can get the total value of different prices at different weight(kg).

Step 1:

	A	B	C	D	E	F	G	H	I
1	Price	40							
2	Qty(kg)	500							
3									
4	Total	20000		20000	40	42	44	47	51
5				500					
6				769					
7				985					
8				1090					
9				1372					
10									

Step 2:

	A	B	C	D	E	F	G	H	I
1	Price	40							
2	Qty(kg)	500							
3									
4	Total	20000		20000	40	42	44	47	51
5				500					
6				769					
7				985					
8				1090					
9				1372					
10									

Output :

	A	B	C	D	E	F	G	H	I	J
1	Price	40								
2	Qty(kg)	500								
3										
4	Total	20000		20000	40	42	44	47	51	
5				500	20000	21000	22000	23500	25500	
6				769	30760	32298	33836	36143	39219	
7				985	39400	41370	43340	46295	50235	
8				1090	43600	45780	47960	51230	55590	
9				1372	54880	57624	60368	64484	69972	
10										

Scenario Manager:

You have an electronic shop, where you sell electronic components, now you have to sell the components at high prices of 50rs and low prices of 50rs. To show the customer you have to make a price list for both the prices. For that we will use Scenario Manager for both price scenarios.

1. For original price:

2. For highest price:

	A	B
1	Material	Cost
2	Hard Disk	55
3	RAM	25
4	VGA Card	50
5	Video Card	20
6	Mother Board	65
7		
8	Total Cost	215

Scenario Values

Enter values for each of the changing cells.

1:	B2	55
2:	B3	25
3:	B4	50
4:	B5	20
5:	B6	65

OK Cancel

3. For Lowest price:

	A	B
1	Material	Cost
2	Hard Disk	45
3	RAM	15
4	VGA Card	40
5	Video Card	10
6	Mother Board	55
7		
8	Total Cost	165

Scenario Values

Enter values for each of the changing cells.

1:	B2	45
2:	B3	15
3:	B4	40
4:	B5	10
5:	B6	55

OK Cancel

OUTPUT (Summary) :

Scenario Summary				
	Current Values:	Original price	Lowest Price	Highest Price
Changing Cells:				
B2	50	50	45	55
B3	20	20	15	25
B4	45	45	40	50
B5	15	15	10	20
B6	60	60	55	65
Result Cells:				
B8	190	190	165	215

Notes: Current Values column represents values of changing cells at time Scenario Summary Report was created. Changing cells for each scenario are highlighted in gray.

Practical – 7

AIM : Practical Implementation of Decision Tree using R Tool.

install.packages("party")

The package "party" has the function ctree() which is used to create and analyze decison tree.

Syntax

The basic syntax for creating a decision tree in R is −

```
ctree(formula, data)
```

Input Data

We will use the R in-built data set named readingSkills to create a decision tree. It describes the score of someone's readingSkills if we know the variables "age","shoesize","score" and whether the person is a native speaker or not.

Here is the sample data.

```
# Load the party package. It will automatically load other
# dependent packages.
library(party)
# Print some records from data set readingSkills.
print(head(readingSkills))
```

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

```
nativeSpeaker age shoeSize score
1 yes 5 24.83189 32.29385
2 yes 6 25.95238 36.63105
3 no 11 30.42170 49.60593
4 yes 7 28.66450 40.28456
5 yes 11 31.88207 55.46085
6 yes 10 30.07843 52.83124
Loading required package: methods
Loading required package: grid
.............................
.............................
```

We will use the ctree() function to create the decision tree and see its graph.

```
# 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()
```

Output:

```
null device
Loading required package: grid
Loading required package: mvtnorm
Loading required package: modeltools
Loading required package: stats4
Loading required package: strucchange
Loading required package: zoo

Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric

Loading required package: sandwich
```

Practical – 8

Aim: K -means clustering using R

```
Console   Terminal ×   Background Jobs ×
R 4.3.2 · ~/
> # 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 50, 38, 62

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

Clustering vector:
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [21] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [41] 1 1 1 1 1 1 1 1 1 1 3 3 2 3 3 3 3 3 3 3
 [61] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3
 [81] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
[101] 2 3 2 2 2 2 3 2 2 2 2 2 2 3 3 2 2 2 2 3
[121] 2 3 2 3 2 2 3 3 2 2 2 2 2 3 2 2 2 2 3 2
[141] 2 2 3 2 2 2 3 2 2 3

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

Available components:

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

Compare the Species label with the clustering result.

```
Console   Terminal ×   Background Jobs ×
R 4.3.2 · ~/
> # Compare the Species label with the clustering result
> table (iris$Species,kc$cluster)

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

Plot the clusters and their center.

```
Console   Terminal ×   Background Jobs ×
R 4.3.2 · ~/
> # Plot the cluster 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)
>
```

```
> plot(newiris[c("Petal.Length","Petal.Width")],col=kc$cluster)
> points(kc$centers[,c("Petal.Length","Petal.Width")],col=1:3,pch=17,cex=2)
> dev.off()
null device
          1
```

Practical – 9

AIM : Prediction Using Linear Regression.

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.

y = ax + b is an equation for linear regression. Where, y is the response variable, x isthe predictor variable and a and b are constants which are called the coefficients.

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 responsevariable.

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))
```

```
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()
```

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

Practical – 10

AIM : Data Analysis using Time Series Analysis.

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 oftime 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 timeseries data. The data for the time series is stored in an R object called time-series object. It isalso 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.

Example

Consider the annual rainfall details at a place starting from January 2014. 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(2014,1),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:

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

```
2014          Jan       Feb       Mar       Apr       May
           799.0     1174.8     865.1    1334.6     635.4
2014          Jun       Jul       Aug       Sep       Oct
           918.5      685.5     998.6     784.2     985.0
2014          Nov       Dec
           882.8     1071.0
```

Practical – 11

AIM : Data Modelling and Analytics with Pivot Table in Excel.

Data Model is used for building a model where data from various sources can be combined by creating relationships among the data sources. A Data Model integrates the tables, enabling extensive analysis using PivotTables, Power Pivot, and Power View.
A **Data Model** is created automatically when you import two or more tables simultaneously from a database. The existing database relationships between those tables is used to create the Data Model in Excel.

Step 1 − Open a new blank Workbook in Excel.
Step 2 − Click on the **DATA** tab.
Step 3 − In the **Get External Data** group, click on the option **From Access**. The **Select Data** Source dialog box opens.
Step 4 − **Select Events.accdb**, Events Access Database file.

Step 5 − The **Select Table** window, displaying all the tables found in the database, appears.
Step 6 − Tables in a database are similar to the tables in Excel. Check the **'Enable selection of multiple** tables' box, and select all the tables. Then click **OK.**

Step 7 – The Import Data window appears. Select the **PivotTable Report** option. This option imports the tables into Excel and prepares a PivotTable for analyzing the imported tables. Notice that the checkbox at the bottom of the window - **'Add this data to the Data Model'** is selected and disabled.

Step 8 – The data is imported, and a **PivotTable** is created using the imported tables.

Explore Data Using PivotTable

Step 1 − You know how to add fields to PivotTable and drag fields across areas. Even if you are not sure of the final report that you want, you can play with the data and choose the best suited report.
In **PivotTable Fields,** click on the arrow beside the table - Medals to expand it to show the fields in that table. Drag the **NOC_CountryRegion** field in the Medals table to the COLUMNS area.
Step 2 − Drag Discipline from the Disciplines table to the ROWS area.
Step 3 − Filter Discipline to display only five sports: Archery, Diving, Fencing, Figure Skating, and Speed Skating. This can be done either in PivotTable Fields area, or from the Row Labels filter in the PivotTable itself.
Step 4 − In PivotTable Fields, from the Medals table, drag Medal to the VALUES area.
Step 5 − From the Medals table, select Medal again and drag it into the FILTERS area.

Step 6 − Click the dropdown list button to the right of the Column labels.
Step 7 − Select Value Filters and then select Greater Than…
Step 8 − Click OK

The Value **Filters dialog** box for the count of Medals **is greater than** appears.

Step 9 – Type 80 in the Right Field.
Step 10 – Click OK.

The PivotTable displays only those regions, which has more than total 80 medals.

Create Relationship between Tables

Relationships let you analyze your collections of the data in Excel, and create interesting and aesthetic reports from the data you import.

Step 1 – Insert a new Worksheet.
Step 2 – Create a new table with new data. Name the new table as Sports.

Step 3 – Now you can create relationship between this new table and the other tables that already exist in the Data Model in Excel. Rename the Sheet1 as **Medals** and Sheet2 as **Sports**.
On the **Medals sheet**, in the **PivotTable Fields List,** click All. A complete list of available tables will be displayed. The newly added table - Sports will also be displayed.

Step 4 – Click on Sports. In the expanded list of fields, select Sports. Excel messages you to create a relationship between tables.

Step 5 – Click on CREATE.
The Create Relationship dialog box opens.

Step 6 – To create the relationship, one of the tables must have a column of unique, non-repeated, values. In the Disciplines table, SportID column has such values. The table Sports that we have created also has the SportID column. In Table, select Disciplines.
Step 7 – In Column (Foreign), select SportID.
Step 8 – In Related Table, select Sports.
Step 9 – In Related Column (Primary), SportID gets selected automatically. Click OK.
Step 10 – The PivotTable is modified to reflect the addition of the new Data Field Sport. Adjust the order of the fields in the Rows area to maintain the Hierarchy. In this case, Sport should be first and **Discipline** should be the next, as **Discipline** will be nested in Sport as a sub-category.

Practical – 12

AIM : Data Analysis and Visualization using Advanced Excel.

Power View is a feature of Microsoft Excel 2013 that enables **interactive** data exploration, visualization, and presentation encouraging intuitive ad-hoc reporting.

Create a Power View Sheet

Make sure **Power View** add-in is enabled in Excel 2013.

Step 1 − Click on the **File** menu and then Click on **Options**.

The **Excel Options** window appears.

Step 2 − Click on **Add-Ins**.

Step 3 − In the **Manage** box, click the drop-down arrow and select **Excel Add-ins**.

Step 4 − All the available **Add-ins** will be displayed. If **Power View** Add-in is

enabled, itappears in Active Application Add-ins.

Now, you are ready to create the **Power View sheet**.

Step 1 – Click on the **Data Table**.

Step 2 – Click on **Insert** tab.

Step 3 – Click on **Power View** in **Reports** Group.

Event	Discipline	Sport	Participation
Combined 15 + 15km mass start	Cross Country S	Skiing	duo
individual	Figure skating	Skating	individual
giant slalom	Alpine Skiing	Skiing	individual
500m	Speed Skating	Skating	individual
5000m	Speed Skating	Skating	individual
10km pursuit	Biathlon	Biathlon	individual
15km	Biathlon	Biathlon	individual
Half-pipe	Snowboard	Skiing	individual
1500m	Short Track S.	Skating	individual
curling	Curling	Curling	team
Team	Nordic Combined	Skiing	team
15km	Cross Country S	Skiing	individual
ice dancing	Figure skating	Skating	duo
4x7.5km relay	Biathlon	Biathlon	team
aerials	Freestyle Ski.	Skiing	individual
four-man	Bobsleigh	Bobsleigh	team
K90 individual (70m)	Ski Jumping	Skiing	individual
downhill	Alpine Skiing	Skiing	individual
Alpine combined	Alpine Skiing	Skiing	individual
K120 team (90m)	Ski Jumping	Skiing	team
10km	Biathlon	Biathlon	individual
Snowboard Cross	Snowboard	Skiing	individual

An **Opening Power View** window opens, showing the progress of Working on opening Power Viewsheet.

The **Power View sheet** is created for you and added to your Workbook with the **Power View**. On the Right-side of the **Power View**, you find the **Power View Fields**. Under the **Power View Fields** you will find **Areas**.

In the Ribbon, if you click on **Design tab**, you will find various **Visualization** options.

Create Charts and other Visualizations

For every visualization you want to create, you start on a Power View sheet by creating a table, which you then easily convert to other visualizations, to find one that best illustrates your Data.

Step 1 – Under the **Power View Fields**, select the fields you want to visualize.

Step 2 – By default, the **Table** View will be displayed. As you move across the Table, on the top-right corner, you find two symbols – Filters and Pop out.

Step 3 – Click on the **Filters** symbol. The filters will be displayed on the right side. **Filters** has two tabs. View tab to filter all visualizations in this **View** and **Table** tab to filter the specific values in this table only.

Visualization – Matrix

A **Matrix** is made up of rows and columns like a **Table**. However, a Matrix has the following capabilities that a Table does not have −

- Display data without repeating values.
- Display totals and subtotals by row and column.
- With a hierarchy, you can drill up/drill down.

Collapse and Expand the Display

Step 1 − Click on the **DESIGN** tab.

Step 2 − Click on **Table** in the **Switch Visualization** Group.

Step 3 − Click on **Matrix**.

The **Matrix Visualization** appears.

Visualization – Card

You can convert a **Table** to a series of **Cards** that display the data from **each row** in the table laid out in a **Card format**, like an **index Card**.

Step 1 – Click on the **DESIGN** tab.

Step 2 – Click on **Table** in the **Switch Visualization** Group.

Step 3 – Click on **Card**.

The **Card Visualization** appears.

Visualization – Charts

In **Power View**, you have a number of Chart options: Pie, Column, Bar, Line, Scatter, and Bubble. You can use several design options in a chart such as showing and hiding labels, legends, and titles.

Charts are interactive. If you click on a Value in one Chart −

- The **Value** in that chart is highlighted.
- All the Tables, Matrices, and Tiles in the report are filtered to that Value.
- That **Value** in all the other Charts in the report is highlighted.

The charts are interactive in a presentation setting also.

Step 1 − Create a **Table Visualization** from **Medals** data.

You can use Line, Bar and Column Charts for comparing data points in one or more data series. In these Charts, the x-axis displays one field and the y-axis displays another, making it easy to see the relationship between the two values for all the items in the Chart.

Line Charts distribute category data evenly along a horizontal (category) axis, and all numerical value data along a vertical (value) axis.

Step 2 − Create a Table Visualization for two Columns, **NOC_CountryRegion** and **Count of Medal**.

Step 3 − Create the same Table Visualization below.

Step 4 – Click on the **Table Visualization** below.

Step 5 – Click on **Other Chart** in the **Switch Visualization** group.

Step 6 – Click on **Line**.

The **Table Visualization** converts into **Line Chart Visualization**.

In a **Bar Chart**, categories are organized along the vertical axis and values along the horizontal axis. In **Power View**, there are three subtypes of the **Bar Chart: Stacked, 100%stacked**, and **Clustered**.

Step 7 − Click on the **Line Chart Visualization**.

Step 8 − Click on **Bar Chart** in the **Switch Visualization** Group.

Step 9 − Click on the **Stacked Bar** option.

The **Line Chart Visualization** converts into **Stacked Bar Chart Visualization**.

Step 10 − In the **Power View Fields**, in the **Medals** Table, select the **Field Gender** also.

Step 11 − Click on one of the bars. That portion of the bar is highlighted. Only the row containing the Data specific to the selected bar is displayed in the table above.

You can use the column charts for showing data changes over a period of time or for illustrating comparison among different items. In a Column Chart, the categories are along the horizontal axis and values are along the vertical axis.

In Power View, there are three **Column Chart** subtypes: **Stacked, 100% stacked**, and **Clustered.**

Step 12 − Click on the **Stacked Bar Chart Visualization**.

Step 13 − Click on Column Chart in the **Switch Visualization** group.

Step 14 − Click on **Stacked Column**.

The **Stacked Bar Chart** Visualization converts into **Stacked Column Chart** Visualization.

You can have simple **Pie Chart Visualizations** in Power View.

Step 1 − Click on the **Table Visualization** as shown below.

Step 2 − Click on **Other Chart** in the **Switch Visualization** group.

Step 3 − Click on **Pie** as shown in the image given below.