R Programming Language is an open-source programming language that is widely used as a statistical software and data analysis tool. Data Frames in R Language are generic data objects of R that are used to store tabular data.

Data frames can also be interpreted as matrices where each column of a matrix can be of different data types. R DataFrame is made up of three principal components, the data, rows, and columns.

The data is presented in tabular form, which makes it easier to operate and understand.

**Create Dataframe in R Programming Language**

To create an R data frame use data.frame() function and then pass each of the vectors you have created as arguments to the function.

# R program to create dataframe

# creating a data frame

friend.data <- data.frame(

friend\_id = c(1:5),

friend\_name = c("Sachin", "Sourav",

"Dravid", "Sehwag",

"Dhoni"),

stringsAsFactors = FALSE

)

# print the data frame

print(friend.data)

**Get the Structure of the R Data Frame**

One can get the structure of the R data frame using str() function in R.

It can display even the internal structure of large lists which are nested. It provides one-liner output for the basic R objects letting the user know about the object and its constituents.

# R program to get the

# structure of the data frame

# creating a data frame

friend.data <- data.frame(

friend\_id = c(1:5),

friend\_name = c("Sachin", "Sourav",

"Dravid", "Sehwag",

"Dhoni"),

stringsAsFactors = FALSE

)

# using str()

print(str(friend.data))

**Summary of Data in the R data frame**

In the R data frame, the statistical summary and nature of the data can be obtained by applying summary() function.

It is a generic function used to produce result summaries of the results of various model fitting functions. The function invokes particular methods which depend on the class of the first argument.

# R program to get the

# summary of the data frame

# creating a data frame

friend.data <- data.frame(

friend\_id = c(1:5),

friend\_name = c("Sachin", "Sourav",

"Dravid", "Sehwag",

"Dhoni"),

stringsAsFactors = FALSE

)

# using summary()

print(summary(friend.data))

**Extract Data from Data Frame in R**

Extracting data from an R data frame means that to access its rows or columns. One can extract a specific column from an R data frame using its column name.

# R program to extract

# data from the data frame

# creating a data frame

friend.data <- data.frame(

friend\_id = c(1:5),

friend\_name = c("Sachin", "Sourav",

"Dravid", "Sehwag",

"Dhoni"),

stringsAsFactors = FALSE

)

# Extracting friend\_name column

result <- data.frame(friend.data$friend\_name)

print(result)

**Expand Data Frame in R Language**

A data frame in R can be expanded by adding new columns and rows to the already existing R data frame.

# R program to expand

# the data frame

# creating a data frame

friend.data <- data.frame(

friend\_id = c(1:5),

friend\_name = c("Sachin", "Sourav",

"Dravid", "Sehwag",

"Dhoni"),

stringsAsFactors = FALSE

)

# Expanding data frame

friend.data$location <- c("Kolkata", "Delhi",

"Bangalore", "Hyderabad",

"Chennai")

resultant <- friend.data

# print the modified data frame

print(resultant)

**Access Items in R Data Frame**

We can select and access any element from data frame by using single $ ,brackets [ ] or double brackets [[]] to access columns from a data frame.

# creating a data frame

friend.data <- data.frame(

friend\_id = c(1:5),

friend\_name = c("Sachin", "Sourav",

"Dravid", "Sehwag",

"Dhoni"),

stringsAsFactors = FALSE

)

# Access Items using []

friend.data[1]

# Access Items using [[]]

friend.data[['friend\_name']]

# Access Items using $

friend.data$friend\_id

**Amount of Rows and Columns**

We can find out how many rows and columns parsant in our dataframe by using dim function.

# creating a data frame

friend.data <- data.frame(

friend\_id = c(1:5),

friend\_name = c("Sachin", "Sourav",

"Dravid", "Sehwag",

"Dhoni"),

stringsAsFactors = FALSE

)

# find out the number of rows and clumns

dim(friend.data)

**Add Rows and Columns in R Data Frame**

You can easily add rows and columns in a R DataFrame. Insertion helps in expanding the already existing DataFrame, without needing a new one.

Let’s look at how to add rows and columns in a DataFrame ? with an example:

**Add Rows in R Data Frame**

To add rows in a Data Frame, you can use a built-in function rbind().

Following example demonstrate the working of rbind() in R Data Frame.

# Creating a dataframe representing products in a store

Products <- data.frame(

Product\_ID = c(101, 102, 103),

Product\_Name = c("T-Shirt", "Jeans", "Shoes"),

Price = c(15.99, 29.99, 49.99),

Stock = c(50, 30, 25)

)

# Print the existing dataframe

cat("Existing dataframe (Products):\n")

print(Products)

# Adding a new row for a new product

New\_Product <- c(104, "Sunglasses", 39.99, 40)

Products <- rbind(Products, New\_Product)

# Print the updated dataframe after adding the new product

cat("\nUpdated dataframe after adding a new product:\n")

print(Products)

**Add Columns in R Data Frame**

To add columns in a Data Frame, you can use a built-in function cbind().

Following example demonstrate the working of cbind() in R Data Frame.

# Existing dataframe representing products in a store

Products <- data.frame(

Product\_ID = c(101, 102, 103),

Product\_Name = c("T-Shirt", "Jeans", "Shoes"),

Price = c(15.99, 29.99, 49.99),

Stock = c(50, 30, 25)

)

# Print the existing dataframe

cat("Existing dataframe (Products):\n")

print(Products)

# Adding a new column for 'Discount' to the dataframe

Discount <- c(5, 10, 8) # New column values for discount

Products <- cbind(Products, Discount)

# Rename the added column

colnames(Products)[ncol(Products)] <- "Discount" # Renaming the last column

# Print the updated dataframe after adding the new column

cat("\nUpdated dataframe after adding a new column 'Discount':\n")

print(Products)