```
## To read and import csv file in R

Import1 <- read.table("C:\\Users\\Swapnil bandekar\\Downloads\\Swapnil\\Data
Analytics\\My Work\\R\\Datasets\\sample.csv",sep=",",header=TRUE)
View(Import1)
dim(Import1)

Import2 <- read.csv("C:\\Users\\Swapnil bandekar\\Downloads\\Swapnil\\Data
Analytics\\My Work\\R\\Datasets\\sample.csv",sep=",",header=TRUE)
View(Import2)
dim(Import2)


# "read.table" and "read.csv" : both commands do the same work
# "C:\\Users\\Swapnil bandekar\\Downloads\\Swapnil\\Data Analytics\\My Work\\R\\"
=> file path
# While Specifying the path for importing the file we can use "Double
Blackslash(\\) or single frontflash(/)"
# sample.csv => filename
# Sep="," => separator
# header=TRUE => To keep header

## What if I remove header=TRUE and sep="," argument??

Import3 <- read.table("C:\\Users\\Swapnil bandekar\\Downloads\\Swapnil\\Data
Analytics\\My Work\\R\\sample.csv",sep = ",")
View(Import3)
dim(Import3)

Import4 <- read.table("C:\\Users\\Swapnil bandekar\\Downloads\\Swapnil\\Data
Analytics\\My Work\\R\\sample.csv")
View(Import4)
dim(Import4)

# We can't see the header name that are present in csv file and we will find it
harder to read the data
# same is the case if we don't use sep=","
# these 2 are imp arguments while reading csv file

## to read text file

Import5 <- read.table("C:\\Users\\Swapnil bandekar\\Downloads\\Swapnil\\Data
Analytics\\My Work\\R\\Datasets\\sample.txt", sep="\t",header=TRUE)
View(Import5)

# here separator is sep="\t"


## To access json file
```

```
library(jsonlite)

Web1 <-
fromJSON("http://api.glassdoor.com/api/api.htm?v=1&format=json&t.p=47699&t.k=g9GdVH
lQ1eM&action=employers&q=pharmaceuticals&userip=192.168.43.42&useragent=Mozilla/%2F
4.0")
Web1
class(Web1)
length(Web1)
str(Web1)

installed.packages(lib.loc = NULL)
View(installed.packages(lib.loc = NULL))


# to read json file it is important to load jsonlite library
# JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is
easy for humans to read and write. It is easy for machines to parse and generate


## To import and read HTML table in R

# We have to load XML package first

library(XML)

Text <-
readLines(url("https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_by_
population"))

# readlines will read the entire HTML url
# we will give the HTML webpage's url in the "url" command

M1 <- readHTMLTable(Text,asText=TRUE)

# readHTMLTable will read all the HTML tables present in the specified webpage's
url
# asText=TRUE => Everything should be read as text

class(M1)

# class = List

length(M1)

# length =2 that means 2 HTML tables are present in this webpage

M1[[1]]

# to access the 1st table
```

```r
M2[[2]]

# to access the 2nd table

class(M1[[1]])
class(M1[[2]])

# class of both tables is data.frame

View(M1[[1]])
View(M1[[2]])

# If there are 20 tables in webpage then all tables are stored in an object as
"List" and length of that object will be 20

# If we want to access data from secure webpage then authentication needs to be
provided

M2 <- readHTMLTable(Text)

class(M2)

View(M2[[1]])

# asText=TRUE is an optional argument


## Accessing the database

# ORACLE - RODBC
# MYSQL - RmySql
# SQLITE - RSQLite
# Hadoop - RHadoop(RMR,RHBASE,RHDFS,RHIVE)

library(DBI)

# database interface
# common for all databases

library(RSQLite)

# database specific

# while accessing the database we need to provide some details as argument
# Database server name , database name , user id , password , db_driver
# Database server name => Server and IP address
# db_driver => type of database
# If we want to access SQLite database then driver will be RSQLite
```

```
driver <- dbDriver("SQLite")
#Type of database

db_file <- "database file path\\database.sqlite"
# generally an IP address in corporate setup

conn <- dbConnect(driver,db_file)

# connecting R with the database
# for local database

# conn <- dBconnect(driver,db_file,userid,pwd)

dbListTables(conn)
# I want to list all the tables of the database

dbListFields(conn,"Player")

dbListFields(conn,"Country")

# to check different cols in DB table
# gives the col names of Player and Country table respectively

# db_file => location of the database
# driver => type of database i.e. SQLite
# Sequence doesn't matter as long as all the arguments all mentioned in the syntax

# mydb<-
dBConnect(MySQL(),user='ab',password='abc123',dbname='XXX_mycustomerdb',host='11.12
.12.13')

temp <- dbReadTable(conn,"Player")

# I am reading the contents of Player table

View(temp)
class(temp)
str(temp)

# class= data.frame (always)

League_DF <- dbReadTable(conn,"League")
View(League_DF)
str(League_DF)


dbGetQuery(conn,"Select * from player")

# very powerful command , can be used to write any possible Query
# limitation of R => Stores data in memory
# so we can load only the required data in R instead of all the data for analysis
# Typical SQL Query

Temp1 <- dbGetQuery(conn,"Select * from player where player_name='Aaron Mooy'")
View(Temp1)

Player_220 <- dbGetQuery(conn, "select * from Player where weight > 220")
View(Player_220)
dbDisconnect(conn)

# databases can be available online as Open Source

dbWriteTable()

# dataframe can be written as table
# Replacing the table , adding rows to the table and deleting the table is possible
# working with multiple databases is possible


#### Data Manipulation in R


getwd() # gives cureent working directory

setwd("C:\\Users\\Swapnil bandekar\\Downloads\\Swapnil\\Data Analytics\\My
Work\\R\\Datasets") # To set new working directory

# helps to read the file from current directory ( no need to specify the location)

Retail <- read.csv("22 Sep - retail_sales.csv")

class(Retail)

View(Retail)

dim(Retail)

# Want to read the 2nd column from data.frame
# DATFRAME(ROW,COLUMN)

Retail[,2] # reads the 2nd column

Retail[2,] # reads the 2nd Row


## A new DF with cols 3 to 7 from original DF

Retail_1 <- Retail[,3:7]

View(Retail_1)
```

```r
class(Retail_1)

## A new dataframe with col 2,3,7
Retail_2 <- Retail[,c(2,3,7)]
View(Retail_2)

## A new datatframe with Row 2,5,8,20,22 and col 1,4
Retail_3 <- Retail[c(2,5,8,20,22),c(1,4)]
View(Retail_3)

## New dataframe should have all columns except 4th col
Retail_4 <- Retail[,-4]
View(Retail_4)

## Create dataframe with col names "cost" and "revenue"
Retail_5 <- Retail[,c("Cost","Revenue")]
View(Retail_5)

## To access the particular col
# Method 1
Retail[,3]
# Method 2
Retail$Month
# Retail = DF name , Month = col name
head(Retail)
# Gives the Top 6 rows from the dataset by default
head(Retail,8)
```

```r
# We can parse the no of rows if we want to access extra no of rows

head(Retail,2)

# if we want to aceess less no of rows

tail(Retail)

# Gives the Bottom 6 rows from the dataset by default

tail(Retail,8)

# We can parse the no of rows if we want to access extra no of rows

tail(Retail,2)

# if we want to aceess less no of rows


## How many rows and cols are present in DF

nrow(Retail)

ncol(Retail)

dim(Retail)

# Dimension command gives the no of rows and cols

str(Retail)

# str command specifies the datatype of each col


## How can I change the datatype of col (Type casting)

class(Retail$Supplier)

Retail$Supplier <- as.character(Retail$Supplier)

str(Retail)

# changes the datatype of Supplier col

# Other commands to change the datatype are as.numeric , as.logical , as.array ,
as.data.frame


## How to print the col names
```

```
colnames(Retail) #1

names(Retail) #2

names(Retail[4]) # gives the colname for 4th col


## How to change the col name : "City" => "Region"

Retail_6 <- Retail

View(Retail_6)

Retail_6$City <- "Region"

# Can't use this command as it will change all the values of City col to "Region"

# Changing col name is similar as "How to replace value in a vector"

class(names(Retail))

names(Retail)[2] <- "Region"

names(Retail)


# col name is changed from "City" to "Region"


## How to change col name dynamically??

# We have to find What is the index of "Region" col and replace it with "City"

which(names(Retail)=="Region")

# Gives the index position of "Region" col i.e. 2
# Double equal is for comparision

names(Retail)[which(names(Retail)=="Region")] <- "City"

names(Retail)

length(Retail)

# gives the no of cols in DF , same as ncol


length(Retail$Supplier)

# gives the no of rows in DF , same as nrow
```

```
## Methamatical functions

mean(Retail$Cost)
min(Retail$Cost)
max(Retail$Cost)
sd(Retail$Cost) # Standard deviation
var(Retail$Cost) # Variance

## How to find unique values in col??

unique(Retail$Item_Category)

## Count of unique values

length(unique(Retail$Item_Category))




## How to filter dataset

# A new DF with only the records of Category "Arts & Architecture"

Retail_ANA <- subset(Retail,Retail$Item_Category=="Art & Architecture")

View(Retail_ANA)

# A new DF with only the records of Category "Arts & Architecture" and Month "Feb"

Retail_ANA1 <- subset(Retail,Retail$Item_Category=="Art & Architecture"&
Retail$Month=="Feb")

View(Retail_ANA1)


# A new DF with only the records of Category "Arts & Architecture" OR Month "Feb"

Retail_ANA2 <- subset(Retail,Retail$Item_Category=="Art & Architecture" |
Retail$Month=="Feb")

View(Retail_ANA2)

# For "OR" condition we use Pipe Operator(|) ( Shift + "\" key )


## How to add new col in DF
```

```
Retail$Cost_USD <- Retail$Cost/70

# New col is added as the end by default



## To reorder the dataset ( To change the index of a cell)

# I want to shift the Cost_USD col next to Cost Col

Retail <- Retail[,c(1:5,10,6:9)]


## How to limit the Cost_USD col to 2 digits

Retail$Cost_USD <- round(Retail$Cost_USD,2)



## How to do mathematical calculation on 2 cols

# I want to calculate profit by substracting cost from revenue

Retail$Profit <- Retail$Revenue-Retail$Cost


## How to sort the data

Temp1 <- c(10,15,20,68,26,9,2)

Temp1


sort(Temp1)


sort(Temp1,decreasing=TRUE)


order(Temp1)

# order command tells the index where you have the smallest value
# smallest value is at 7th position , 2nd smallest is at 6th postion , 3rd smallest
is at 1st position and so on...

Temp1[order(Temp1)]


Temp1[order(-Temp1)]
```

```
# order command can be used to sort the data


## I want to sort the the Retail DF according to the Revenue col

# We can't use sort command as it will sort Revenue col only and all the other cols will remain as it is
# We can use order command

Retail_N <- Retail[order(Retail$Revenue),]

View(Retail_N)

# Sort command on DF

Retail_BKP <- Retail


Retail_BKP$Revenue <- sort(Retail_BKP$Revenue)

View(Retail_BKP)



## Functions to clean the dataset

## String Manipulation

## How to concatenate the stings??

paste("Sumita","Karamakar",sep = "_")

paste("Sumita","Karamakar","_")

# "Sep=" is a must

Retail$City_Month <- paste(Retail$City,Retail$Month,sep = "_")


## How to extract a particular portion from a string

Vec11 <- c("SWAPNILBANDEKAR","OMKARACHAREKAR","AAKASHSAWANT","ADITYAJADHAV")

Vec11



substr(Vec11,start = 5,stop = 9)

# "substr" command is used to extract a particular portion from string
```

```
substr(Vec11,5,9)

# Start and Stop are optinal argument


## How to extract last 5 character from Vec11??

length(Vec11)

nchar(Vec11)

# "nchar" gives length of each element in the vector


substr(Vec11,(nchar(Vec11)-4),nchar(Vec11))


# gives me the last 5 characters from each string


### How to find and replace value in a vec/col

X1 <- c("You are good I am good","You are good I am good We are good","You are
good","I am good","You are good I am good")

X1

## I want to change "good" with "great"


gsub("good","great",X1)

# gsub is used to find and replace value in a string

sub('good', 'great', X1)


# sub() will replace only the first occurance of a pattern while gsub() will
replace all the occurances





## I want to change "Art & Architecture" with "Business & Architecture"

Retail$Item_Category_N <- gsub("Art & Architecture","Business &
Architecture",Retail$Item_Category)
```

```
## How to split a string??

X2 <- "Split the words in the sentence it is a major requirement in text analytics
project"

X2


strsplit(X2," ")


X3 <- "Split the words in the sentence , it is a major requirement in text
analytics project"

X3

strsplit(X3,",")



X4 <- "Split the words in the sentence_it is a major requirement in text analytics
project"

X4

strsplit(X4,"_")



# How to find if the particular character/pattern in present in a string
# will use grep command

X5 <- c('abc_dcd', 'asad/css', 'asas_cdsd')

grep('_',X5)

# '_' is present in 1st and 3rd element


grepl('_', X5)

# grepl will do the logical comparison




#### How to handle Dates in R
```

```
Date_Var <- c("2016/02/12","2016/03/10","2017/01/15")

Date_Var

class(Date_Var)

#class = character

# By default dates will be stored as character

Date_Var1 <- as.Date(Date_Var,"%Y/%m/%d")

Date_Var1

class(Date_Var1)

#class = Date

Date_Var1+10


Date_Var2 <- c("16/March/12","16/March/10","17/April/15")

Date_Var2

Date_Var3 <- as.Date(Date_Var2,"%y/%B/%d")

Date_Var3

class(Date_Var2)
class(Date_Var3)

Date_Var3+100

# If year is in 4 digit : %Y
# If year is in 2 digit : %y
# If month is in digits : %m
# If month is in string and full : %B
# If month is in string and short : %b
# For Date : %d

Date_Var4 <- c("2016-Mar-12","2016-Jan-10","2017-Apr-15")

Date_Var4

Date_Var5 <- as.Date(Date_Var4,"%Y-%b-%d")

Date_Var5

months(Date_Var5)
```

```
weekdays(Date_Var5)



date_var6 <- Sys.time()
date_var6

class(date_var6)
typeof(date_var6)


months(date_var6)
weekdays(date_var6)


date_var7 <- as.POSIXlt(date_var6)
date_var7

class(date_var7)
typeof(date_var7)


date_var7$wday
date_var7$hour
date_var7$min




# Finding Time Interval


Date_Var5[3] - Date_Var5[1]


# difftime() fn can be used to find the time difference wrt different units


difftime(Date_Var5[3], Date_Var5[1], units = 'days')

difftime(Date_Var5[3], Date_Var5[1], units = 'weeks')

difftime(Date_Var5[3], Date_Var5[1], units = 'hours')



## We can use "lubridate" package for Date Manipulation

library(lubridate)
```

```
Date1 <- "20-12-2019"
Date1
class(Date1)

dmy(Date1)
class(dmy(Date1))


Date2 <- "20-JAN-2019"
dmy(Date2)
class(dmy(Date2))

Date3 <- "Jan-20-2019"
mdy(Date3)
class(mdy(Date3))


# Some date types

ymd("20170131")
ydm("20173101")
mdy("January 31st,2017")
dmy("31st of January '17")


yq("2001:Q3")


X6 <- dmy_hms("12-JAN-19 11:46:20")
X6

date(X6)
month(X6)
year(X6)
day(X6)
wday(X6) # weekday
hour(X6)
minute(X6)
second(X6)
week(X6)
semester(X6)
am(X6) # True
pm(X6) # False
leap_year(X6) # False
```