Exercise 4 : Importing and Exporting Data And Descriptive Statistics

In this session of exercise, the following things should be implemented in R

- Importing files of various formats from the local drive
- Exporting R data to the local drive
- Reshaping and Merging datasets

Note the text shaded (which you will found as you go through the exercises) with grey are commands.

```
This is command
```

Practice Session

1. Follow the given commands to create a table

```
trial <- matrix(c(34,11,9,32), ncol=2)
colnames(trial) <- c('sick', 'healthy')
rownames(trial) <- c('risk', 'no_risk')
trial.table <- as.table(trial)
trial.table
```

With the above code, you do the following:

I. Create a matrix with the number of cases for every combination of sick/healthy and risk/no risk behavior.

II. Add column names to point out which category the counts are for.

III. Convert that matrix to a table.

2. Extracting data from table in R

```
trial.table['risk', 'sick']
```

Output:

```
[1] 34
```

Assignments

Section I : Importing Files to R

1. Import the ClinicalTrail dataset (data is shared) and find the following details.

After importing go through the dataset and see all column names.

i. How many subjects (patients) are enrolled at each center in a clinical trial.

ii. How many subjects (patients) are under the age of 60 in a clinical trial.

iii. Which center has the most subjects with a missing value for age in the clinical trial?

The above questions can be answered by using table() command. For question (iii), use "is.na" argument to find the missing values in the dataset.

Example of is.na function

Let's say, we have created a dataframe by using following command

```
a<-
data.frame(First=c(1:4),Second=c("A","B",NA,"D"),Third=c("D",NA,"E","
F"))
```

Ouput is as follows

```
 First Second Third
1     1      A     D
2     2      B  <NA>
3     3   <NA>     E
4     4      D     F
```

I am using is.na function inside the table function to see how many null values are there

```
table(is.na(a$Second)) ## Checking in column "Second"
```

Output

```
FALSE  TRUE
    3     1
```

```
table(is.na(a$Third)) ## Checking in column "Third"
FALSE  TRUE
    3     1
```

2. Import dataset access_log file (file is shared) and in to R. Import only and make Date, Month, Year, Hours, Min and Seconds columns

3. Import dataset error_log file (file is shared) into R and add suitable column names to it.

4. Import input.json file (file is shared) into R and convert it into dataframe and store it in object.
   i. Extract the names starting with alphabet "R" and whose salary is less than 500.
   ii. Extract employee information whose salary is less than 800 and working in IT and Operations Dept.

Practice Session : Importing XML files into R

5. Consider sample.xml file (file is shared). Do not paste the commands, type the commands and understand the output. (try this at home)

```
install.packages("XML")
library("XML")  # Load the package required to read XML files.
library("methods")      # Also load the other required package.
result <- xmlParse(file = "input.xml") # Print the result.
print(result) # Print the result.
```

To get number of nodes present in XML file

```
rootnode <- xmlRoot(result)  # Extract the root node form the xml file.
rootsize <- xmlSize(rootnode)  # Find number of nodes in the root.
```

```
print(rootsize)          # Print the result.
```

To get details of first node

```
rootnode <- xmlRoot(result)       # Exract the root node form the xml file.
print(rootnode[1])    # Print the result.
```

To get different elements of a Node

```
rootnode <- xmlRoot(result)       # Exract the root node form the xml file.
print(rootnode[[1]][[1]])         # Get the first element of the first node.
print(rootnode[[1]][[5]])         # Get the fifth element of the first node.
print(rootnode[[3]][[2]])         # Get the second element of the third node
```

Converting XML to dataframe

```
xmldataframe <- xmlToDataFrame("input.xml")
print(xmldataframe)
```

For more functions in XML package see help page (?xml)

6. Consider a dataset **iris** (in-built dataset) and store it in different object.
   Find the following details.
   - i. How many variety of species are there and at frequency those species are appearing the dataset.
   - ii. Find the species name whose "Sepal.Length" is equal to 6.9 and at what frequency those species are appearing in the dataset.

7. Consider a dataset **mtcars** (in-built dataset) and store it in different object. Second column in the dataset is "mpg" which is Miles/(US) gallon
   Add an extra column called "Condition" to the mtcars and fill the values as follows (Example is shown below)
   - i. Multiply mpg and cyl

8. **Example**
   Look at example for adding column in the existing dataset. I am considering the dataframe (a) which is created above. The command goes as follows

```
a$Fourth<-a$First+1
```

The output will be as follows, with an extra column "Fourth"

```
First Second Third Fourth
1     1       A     D      2
2     2       B  <NA>      3
3     3    <NA>     E      4
4     4       D     F      5
```

9. After adding an extra column, export it to local drive in both csv and text files

10. Import dataset bnames2 (dataset is shared), which consists of top 1000 male and female baby names in US from 1800 to 2008. Dataset is 258000 x5.

This dataset is in ZIP file. Read the ZIP data into R. You can use read.csv function to read zip file also. This data is tidy, but we can do few analysis on this data. For example

i. How many times name "Greg" has appeared in the given dataset.
ii. How many times name "John" has appeared from years 1880-2000.
iii. In which year was your name most popular? Least popular?

Section II : Merging data frames

Using merge function to combine two data frames based on column names

11. Create the following dataframes (Dataframe 1, Dataframe 2) as shown below. You can copy and paste the commands on R console

Dataframe 1:

```
Died.At <- c(22,40,72,41)
Writer.At <- c(16, 18, 36, 36)
First.Name <- c("John", "Edgar", "Walt", "Jane")
Second.Name <- c("Doe", "Poe", "Whitman", "Austen")
Sex <- c("MALE", "MALE", "MALE", "FEMALE")
Date.Of.Death <- c("2015-05-10", "1849-10-07", "1892-03-26","1817-07-18")
writers_df <- data.frame(Died.At, Writer.At, First.Name, Second.Name, Sex, Date.Of.Death)
```

Dataframe 2:

```
data2 <- data.frame(Age.At.Death=c(22,40,72,41), Location=5:8)
```

12. Extract the names of Dataframe 1 and Dataframe 2 created in the previous question. Output will be as follows

```
[1] "Died.At"  "Writer.At"     "First.Name"    "Second.Name"
[5] "Sex"            "Date.Of.Death"
```

13. Change the names of Dataframe 1. Output should be as follows. (Use indexing). Here we are changing the column name in order to make column names same in both the vector.

```
[1] "Age.At.Death"  "Writer.At"     "First.Name"    "Surname"
[5] "Gender"        "Date.Of.Death"
```

14. We can also merge the dataframes without changing the column name also. Command is shown as follows

```
merge(writers_df, data2, by.x="Age.At.Death", by.y="Age")
```

15. Write a command to merge the above two dataframes. Use by.x and by.y arguments separately and check the output.

16. Reduce the age by 1 in each row of Dataframe 1

17. Add the extra column to the Dataframe 1. Column name should be ID and the values are 1:4

18. Change any one value of Age.At.Death in one of the dataframe and trying to merge command and check the output.

19. Try the merge command on the datasets created with the argument `all.x=TRUE` and `all.y=TRUE` and check the output

Using merge function to combine two data frames based on rownames

20. Create the following dataframe.

```
Address <- c("50 West 10th", "77 St. Marks Place", "778 Park Avenue")
Maried <- c("YES", "NO", "YES")
limited_writers_df <- data.frame(Address, Maried)
limited_writers_df
```

Apply the merge function on "writers_df" and "limited_writers_df" datasets and check the output. To merge rows, use by=0 argument.

Section III : Transforming data frames

21. Create the following dataframe. You can copy paste the commands on R console

```
fy <- c(2010,2011,2012,2010,2011,2012,2010,2011,2012)
company <-
c("Apple","Apple","Apple","Google","Google","Google","Microsoft","Microsoft","Microsoft")
revenue <- c(65225,108249,156508,29321,37905,50175,62484,69943,73723)
profit <- c(14013,25922,41733,8505,9737,10737,18760,23150,16978)
companiesData <- data.frame(fy, company, revenue, profit)
```

Find the following:

i. Transform the data by adding an extra column Margin (You have to frame a formula how margin is calculated)
ii. Find which company is having highest margin
iii. Find which company is having lowest margin

Section IV: Reshaping the Data

22. Create the following data frame. You can copy and paste the commands

```
Subject <- c(1,2,1,2,2,1)
Gender <- c("M", "F", "M", "F", "F","M")
Test <- c("Read", "Write", "Write", "Listen", "Read", "Listen")
Result <- c(10, 4, 8, 6, 7, 7)
observations_long <- data.frame(Subject, Gender, Test, Result)
```

The command gives the output as follows

```
 Subject Gender   Test Result
       1      M   Read     10
       2      F  Write      4
       1      M  Write      8
       2      F Listen      6
       2      F   Read      7
       1      M Listen      7
```

Convert the above data as shown below

```
 Subject Gender Listen Read Write
       1      M      7   10     8
       2      F      6    7     4
```

23. Import the Pew dataset (It is shared) in table format and go through the data. This dataset is not ready for analysis, columns are of same variables ("Income").
Note: After reshaping the dataset, save the output in different object

We have 11 columns and their names are as follows

```
[1] "religion"          "X.10k"              "X.10.20k"            "X.20.30k"          "X.30.40k"
[6] "X.40.50k"          "X.50.75k"          "X.75.100k"          "X.100.150k"        "X.150k"
[11] "Don.t.know.refused"
```

Change the column names from second column to tenth column as shown below

```
[1] "religion"          "10k"               "10-20k"             "20-30k"             "30-40k"
[6] "40-50k"            "50-75k"            "75-100k"            "100-150k"           "150k"
[11] "Don.t.know.refused"
```

Now, put all the columns[2:10] in one column (can be named as Income and it corresponding values).

24. Import GDP data (Data is shared). The dataset is having 14 rows and 13 columns.
Reshape the data in such a way that it should have four columns: Country, Variable, Year, Var1.
After reshaping store the new data in different object