R LAB MANUAL

IMPORTING DATA FROM LARGE DATASETS, LOADING DATA FROM DATABASES, READING FROM URL,STORING DATA USING FUNCTIONS,WRITING R SCRIPTS

1. IMPORTING DATA FROM LARGE DATASETS

```
my_crime_data_csv=read.csv("crimes.csv")
```

```
library(data.table)
my_crime_data=fread("crimes.csv")
```

The data.table is an alternative to R's default data.frame to handle tabular data. The reason it's so popular is because of the speed of execution on larger data and the terse syntax.

2. READING DATA FROM URL

```
data=read.csv("http://databank.worldbank.org/data/download/GDP.csv")
data
View(data)
```

3. STORING DATA USING FUNCTIONS

If data is imported into R, an analyst might perform selection, aggregation and manipulation. Sometimes all these operations might take several minutes or sometimes even longer, the same operations need not be repeated again and again. Hence, the dataset maybe at a stage that's pre-analyses but post-processing (where 'processing' might include cleaning, manipulating, calculating new variables, merging, selecting, aggregating and lots of other stuff).

Majorly there are 3 ways of saving and exporting the data from R

1. Saving it as an R object with the functions save()

The simplest way to save an Rfile is to save in an Rdata file. This gets stored in a binary format and can be opened only by R

```
save(df, file = "df.RData")
```

2. Saving it as a CSV file

```
write.csv(df, file = "df.csv")
```

3. Exporting it to an Excel file with WriteXLS()

```
install.packages("writexl")

df <- data.frame(Name = c("Jon", "Bill", "Maria", "Ben", "Tina"),

          Age = c(23, 41, 32, 58, 26)

          )

write_xlsx(df, "C:\\Users\\Ron\\Desktop\\Test\\people.xlsx")
```

4. WRITING R SCRIPTS

There are certainly many cases where it makes sense to type code directly into the console. For example, to open a help menu for a new function with the ? command, to take a quick look at a dataset with the `head()` function, or to do simple calculations like `1+1`, you should type directly into the console. However, the problem with writing all your code in the console is that nothing that you write will be saved. So if you make an error, or want to make a change to some earlier code, you have to type it all over again. Not very efficient. For this (and many more reasons), you'll should write any important code that you want to save as an R script. An R script is just a bunch of R code in a single file. You can write an R script in any text editor, but you should save it with the `.R` suffix to make it clear that it contains R code.} in an editor.

Steps to Create an RScript

1. Create an Rscript.
2. Store it with .R
3. Send code from an source to the console

5. WORKING WITH DATATABLES

```
library(data.table)
data("iris")
setDT(iris)
print (is.data.table(iris))
View(iris)
class(iris)
```

Basic Syntax for using Data Table

my_datatable[i,   j ,   by]

Start with my_datatable
Subset or Reorder using i
Calculate using j
Orber by ‘by’

Eg
my_dt[ORIGIN_AIRPORT == 'LAX' & DESTINATION_AIRPORT == 'MSP', mean(ARRIVAL_DELAY),by = AIRLINE]