Data Interfaces in R

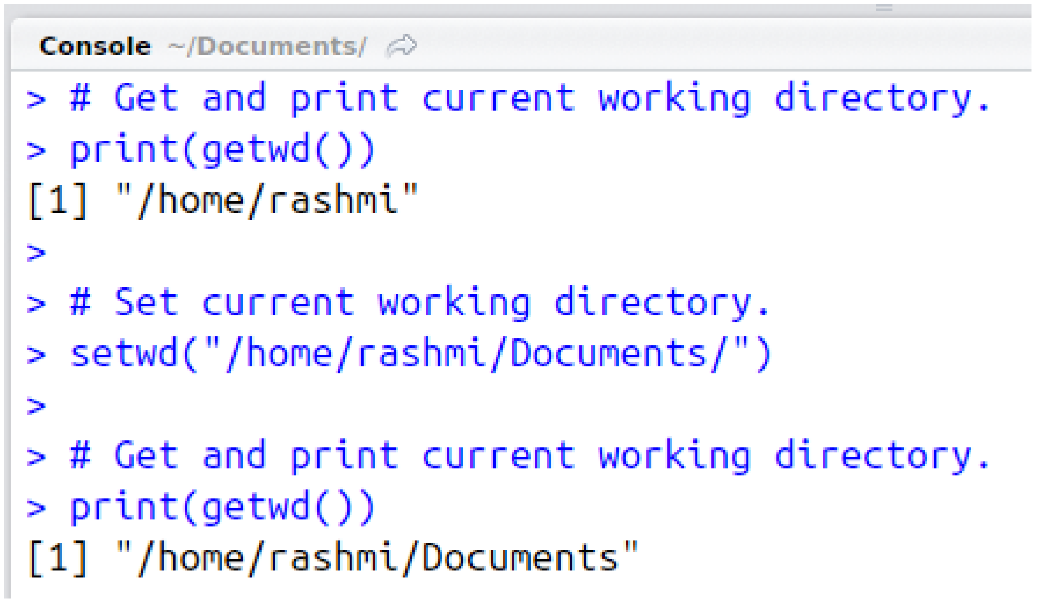
Tushar B. Kute,
http://tusharkute.com

Data Interfaces

- In R, we can read data from files stored outside the R environment.
- We can also write data into files which will be stored and accessed by the operating system.
- R can read and write into various file formats like csv, excel, xml etc.

Getting and setting working directory

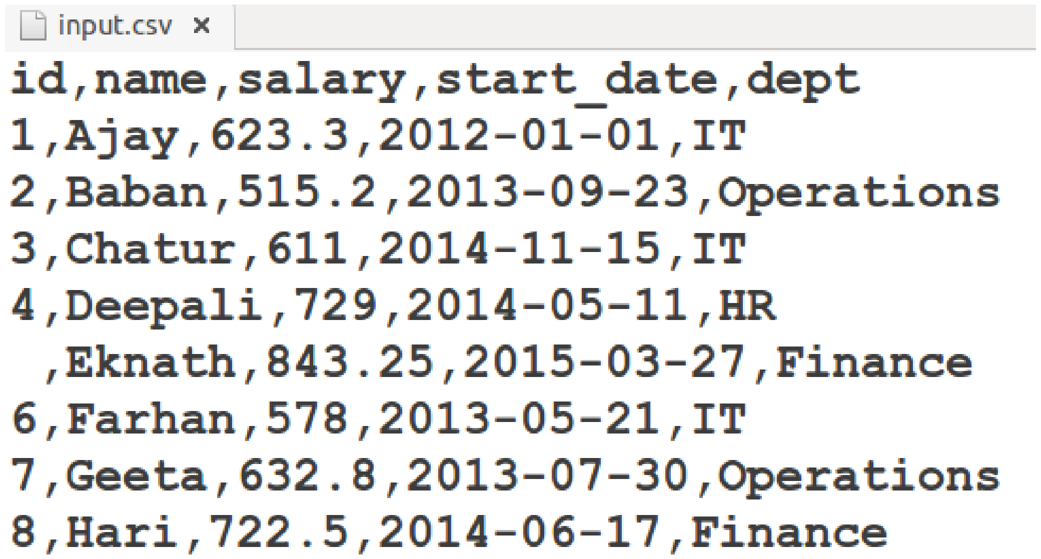
- You can check which directory the R workspace is pointing to using the getwd() function.
- You can also set a new working directory using setwd()function.



```
> # Get and print current working directory.
> print(getwd())
[1] "/home/rashmi"
>
> # Set current working directory.
> setwd("/home/rashmi/Documents/")
>
> # Get and print current working directory.
> print(getwd())
[1] "/home/rashmi/Documents"
```

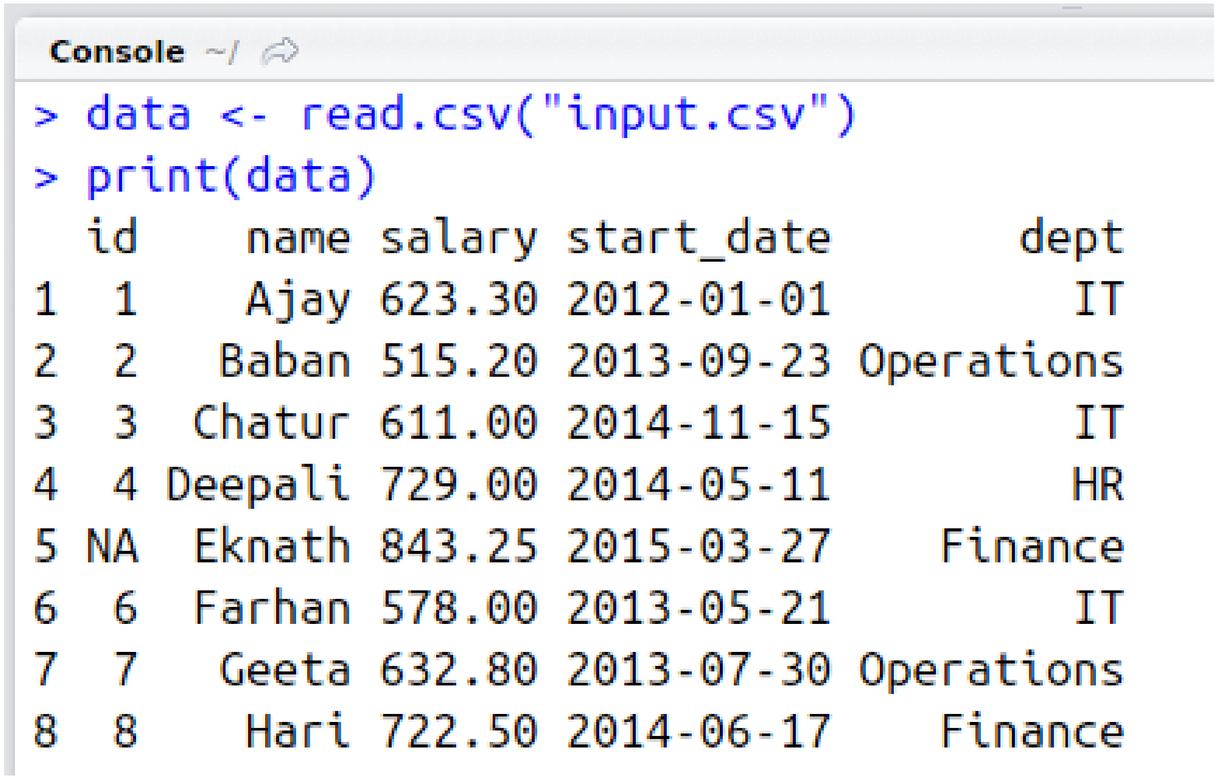
Input as CSV file

- The csv file is a text file in which the values in the columns are separated by a comma. Let's consider the following data present in the file named input.csv.
- You can create this file using windows notepad or Ubuntu gedit by copying and pasting this data. Save the file as input.csv using the save As All files(*.*) option in notepad/gedit.



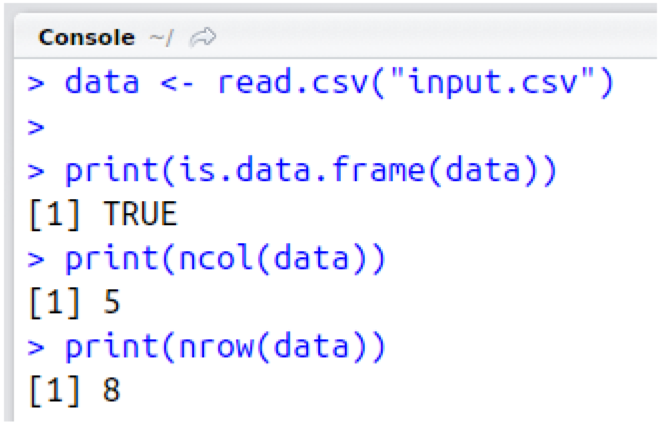
```
id,name,salary,start_date,dept
1,Ajay,623.3,2012-01-01,IT
2,Baban,515.2,2013-09-23,Operations
3,Chatur,611,2014-11-15,IT
4,Deepali,729,2014-05-11,HR
 ,Eknath,843.25,2015-03-27,Finance
6,Farhan,578,2013-05-21,IT
7,Geeta,632.8,2013-07-30,Operations
8,Hari,722.5,2014-06-17,Finance
```

Reading a CSV file

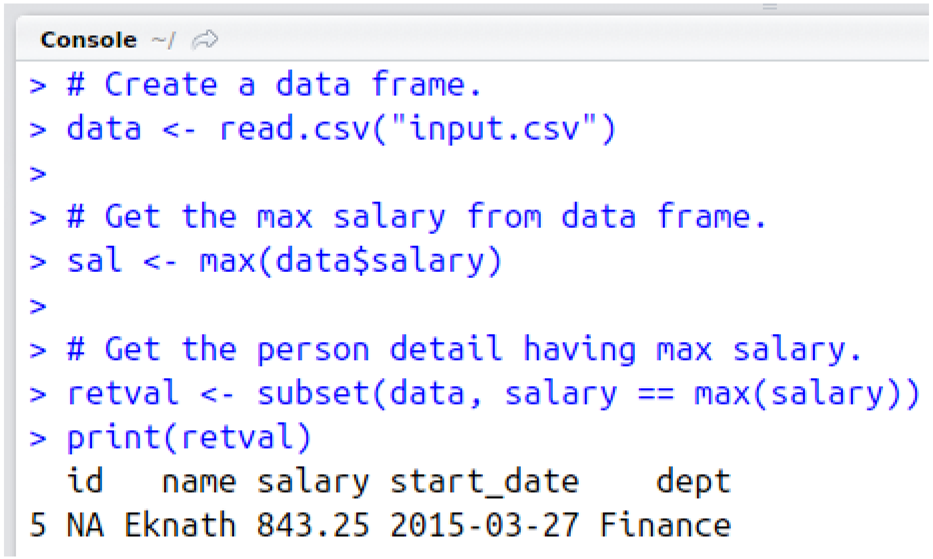

Console ~/

```
> data <- read.csv("input.csv")
> print(data)
  id    name salary start_date       dept
1  1    Ajay 623.30 2012-01-01         IT
2  2   Baban 515.20 2013-09-23 Operations
3  3  Chatur 611.00 2014-11-15         IT
4  4 Deepali 729.00 2014-05-11         HR
5 NA  Eknath 843.25 2015-03-27    Finance
6  6  Farhan 578.00 2013-05-21         IT
7  7   Geeta 632.80 2013-07-30 Operations
8  8    Hari 722.50 2014-06-17    Finance
```

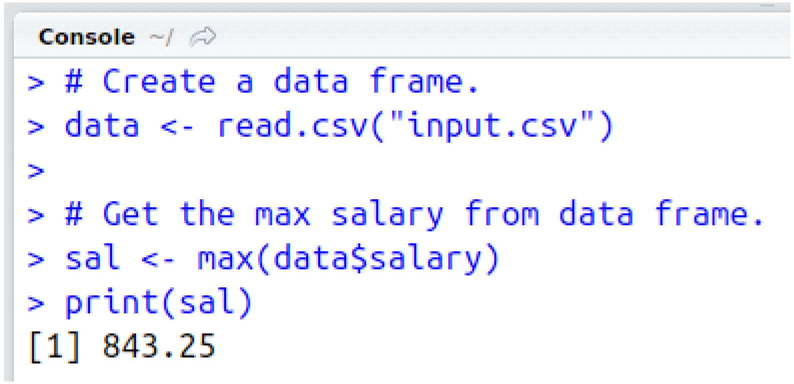
Analysing a CSV file



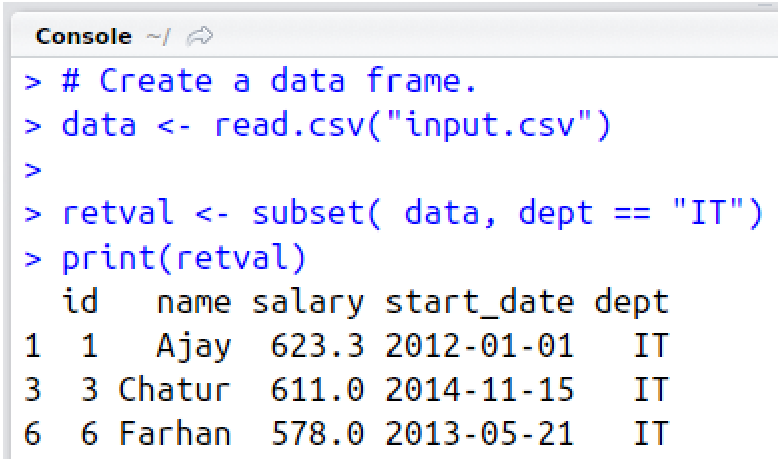
```
Console ~/
> data <- read.csv("input.csv")
>
> print(is.data.frame(data))
[1] TRUE
> print(ncol(data))
[1] 5
> print(nrow(data))
[1] 8
```



```
Console ~/
> # Create a data frame.
> data <- read.csv("input.csv")
>
> # Get the max salary from data frame.
> sal <- max(data$salary)
>
> # Get the person detail having max salary.
> retval <- subset(data, salary == max(salary))
> print(retval)
  id    name salary start_date    dept
5 NA Eknath 843.25 2015-03-27 Finance
```

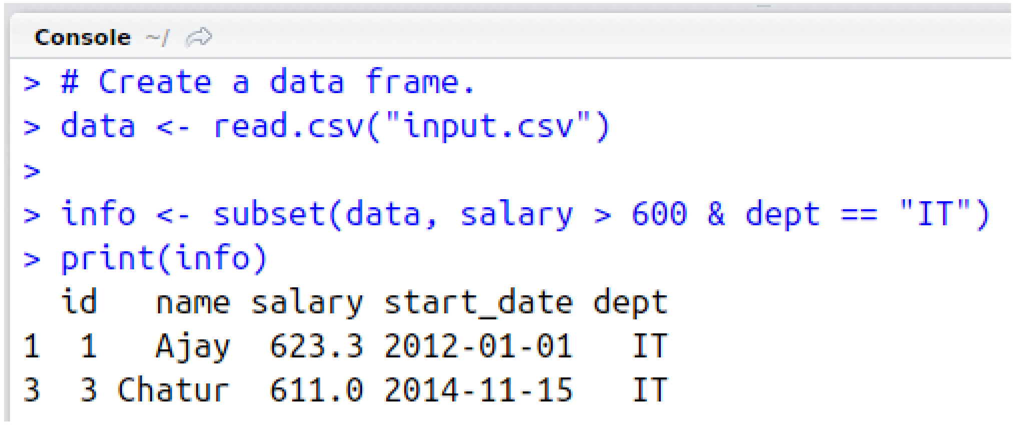


```
Console ~/
> # Create a data frame.
> data <- read.csv("input.csv")
>
> # Get the max salary from data frame.
> sal <- max(data$salary)
> print(sal)
[1] 843.25
```



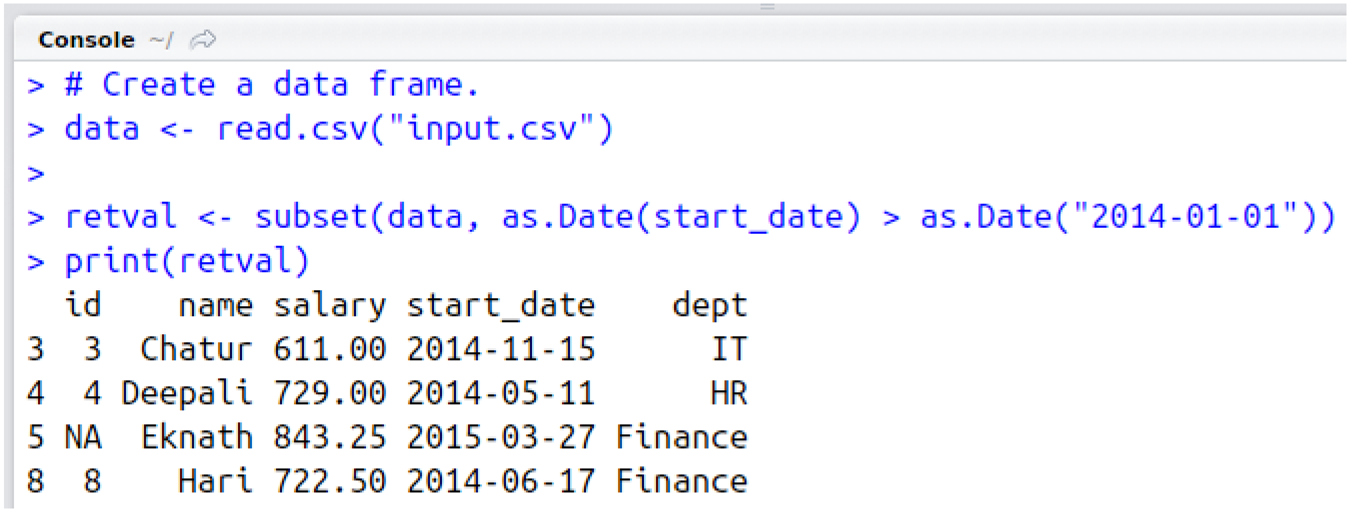
```
Console ~/
> # Create a data frame.
> data <- read.csv("input.csv")
>
> retval <- subset( data, dept == "IT")
> print(retval)
  id   name salary start_date dept
1  1   Ajay  623.3 2012-01-01   IT
3  3 Chatur  611.0 2014-11-15   IT
6  6 Farhan  578.0 2013-05-21   IT
```

Analysing a CSV file



Console ~/

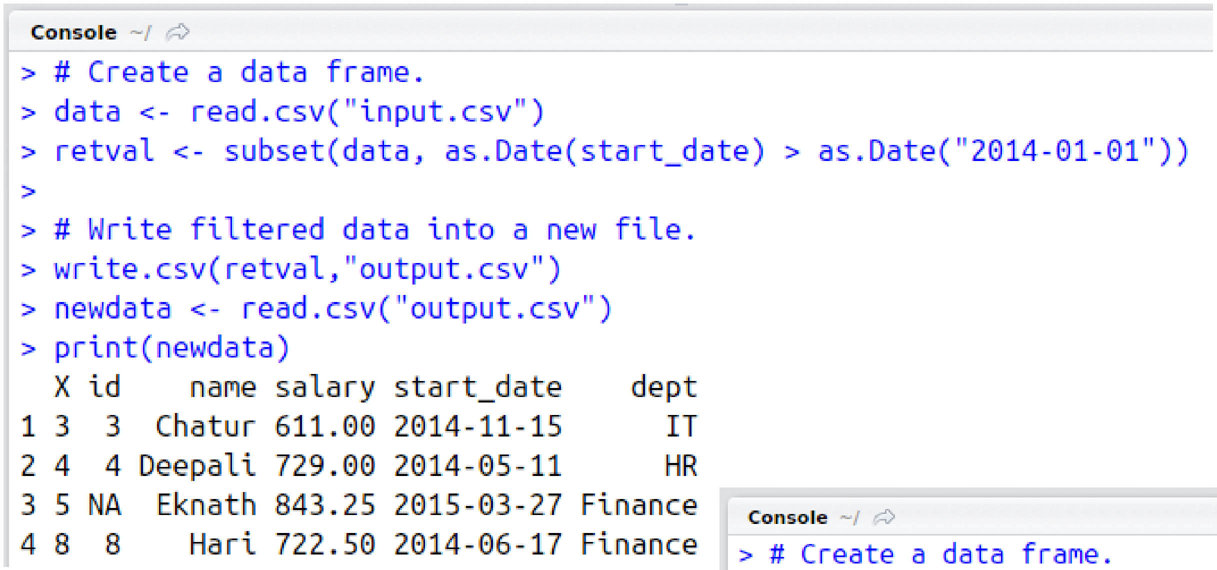
```
> # Create a data frame.
> data <- read.csv("input.csv")
>
> info <- subset(data, salary > 600 & dept == "IT")
> print(info)
  id   name salary start_date dept
1  1   Ajay  623.3 2012-01-01   IT
3  3 Chatur  611.0 2014-11-15   IT
```



Console ~/

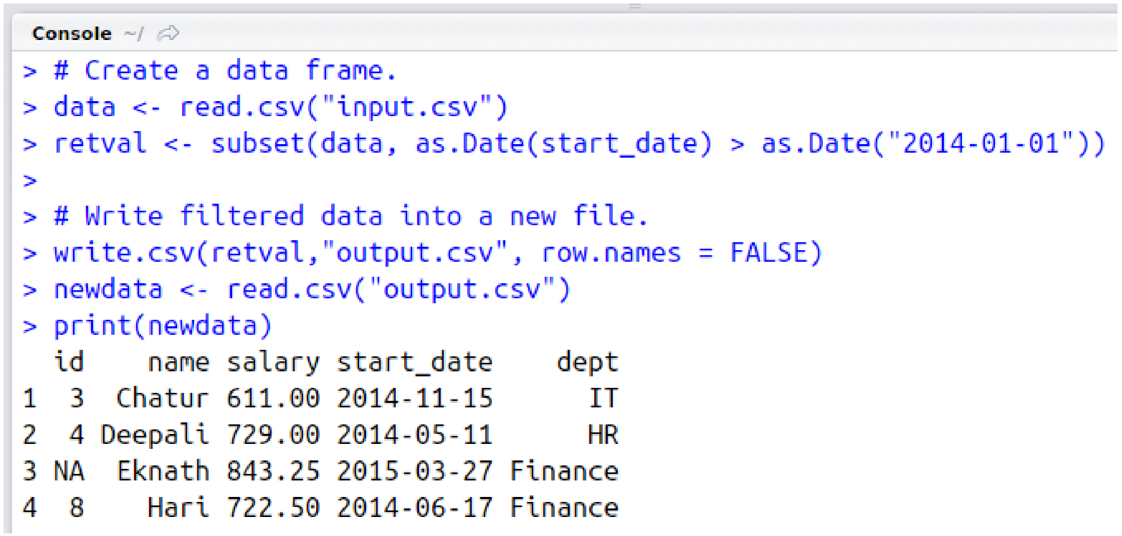
```
> # Create a data frame.
> data <- read.csv("input.csv")
>
> retval <- subset(data, as.Date(start_date) > as.Date("2014-01-01"))
> print(retval)
  id    name salary start_date    dept
3  3  Chatur 611.00 2014-11-15      IT
4  4 Deepali 729.00 2014-05-11      HR
5 NA  Eknath 843.25 2015-03-27 Finance
8  8    Hari 722.50 2014-06-17 Finance
```

Writing into a CSV file



Console ~/

```
> # Create a data frame.
> data <- read.csv("input.csv")
> retval <- subset(data, as.Date(start_date) > as.Date("2014-01-01"))
>
> # Write filtered data into a new file.
> write.csv(retval,"output.csv")
> newdata <- read.csv("output.csv")
> print(newdata)
  X id    name salary start_date    dept
1 3  3  Chatur 611.00 2014-11-15      IT
2 4  4 Deepali 729.00 2014-05-11      HR
3 5 NA  Eknath 843.25 2015-03-27 Finance
4 8  8    Hari 722.50 2014-06-17 Finance
```



Console ~/

```
> # Create a data frame.
> data <- read.csv("input.csv")
> retval <- subset(data, as.Date(start_date) > as.Date("2014-01-01"))
>
> # Write filtered data into a new file.
> write.csv(retval,"output.csv", row.names = FALSE)
> newdata <- read.csv("output.csv")
> print(newdata)
  id    name salary start_date    dept
1  3  Chatur 611.00 2014-11-15      IT
2  4 Deepali 729.00 2014-05-11      HR
3 NA  Eknath 843.25 2015-03-27 Finance
4  8    Hari 722.50 2014-06-17 Finance
```

XLSX file

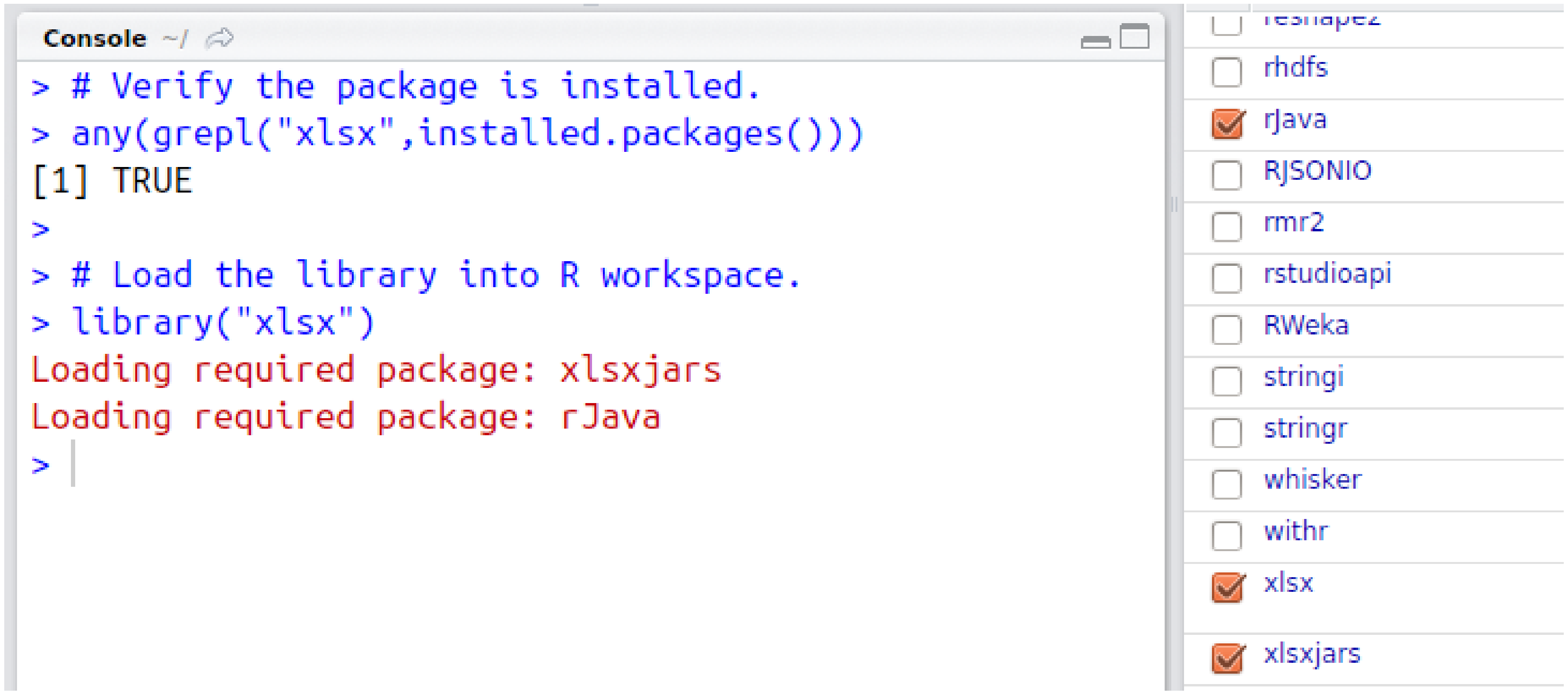
- Microsoft Excel is the most widely used spreadsheet program which stores data in the .xls or .xlsx format.
- R can read directly from these files using some excel specific packages.
- Few such packages are - XLConnect, xlsx, gdata etc.
- We will be using xlsx package. R can also write into excel file using this package.

XLSX package installation

- You can use the following command in the R console to install the "xlsx" package.
- It may ask to install some additional packages on which this package is dependent.
- Follow the same command with required package name to install the additional packages.

```
install.packages("xlsx")
```

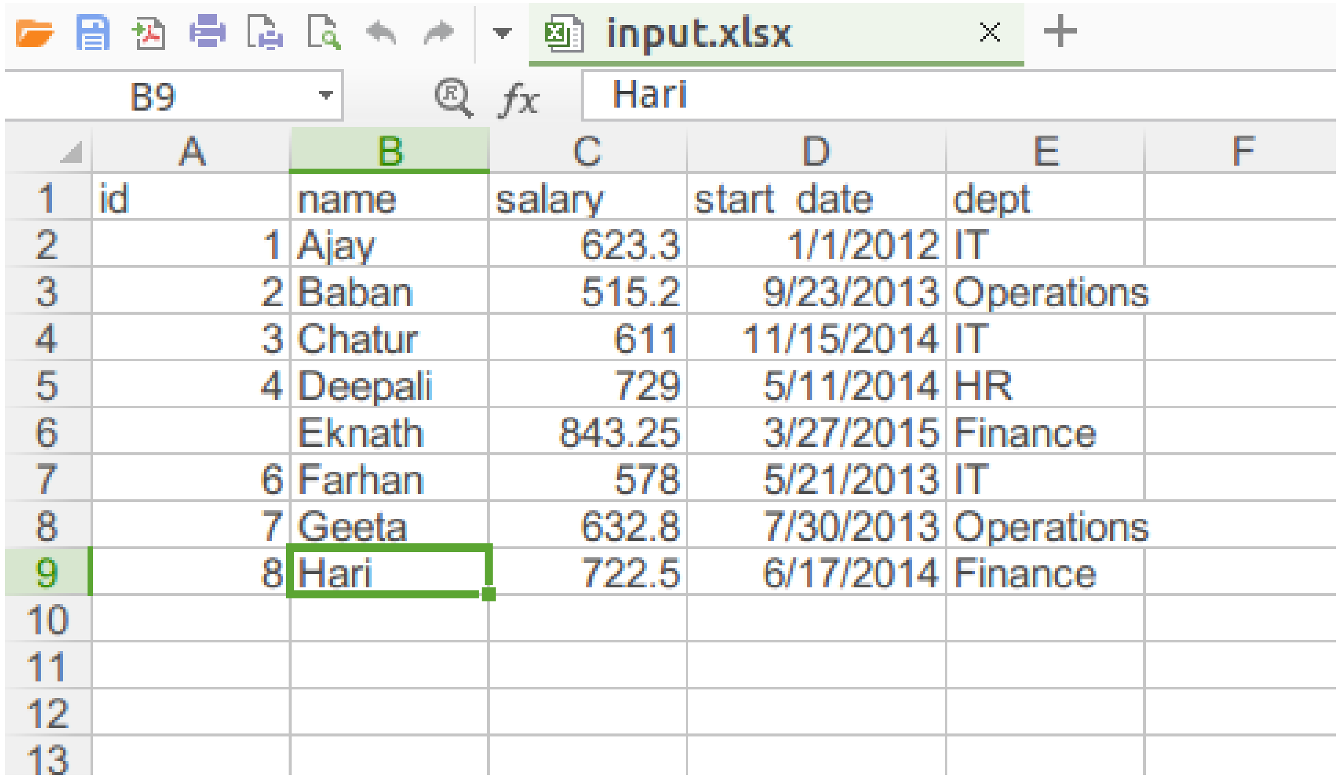
Verify installation



```
> # Verify the package is installed.
> any(grepl("xlsx",installed.packages()))
[1] TRUE
>
> # Load the library into R workspace.
> library("xlsx")
Loading required package: xlsxjars
Loading required package: rJava
>
```

☐	reshape2
☐	rhdfs
✓	rJava
☐	RJSONIO
☐	rmr2
☐	rstudioapi
☐	RWeka
☐	stringi
☐	stringr
☐	whisker
☐	withr
✓	xlsx
✓	xlsxjars

Create an xlsx file



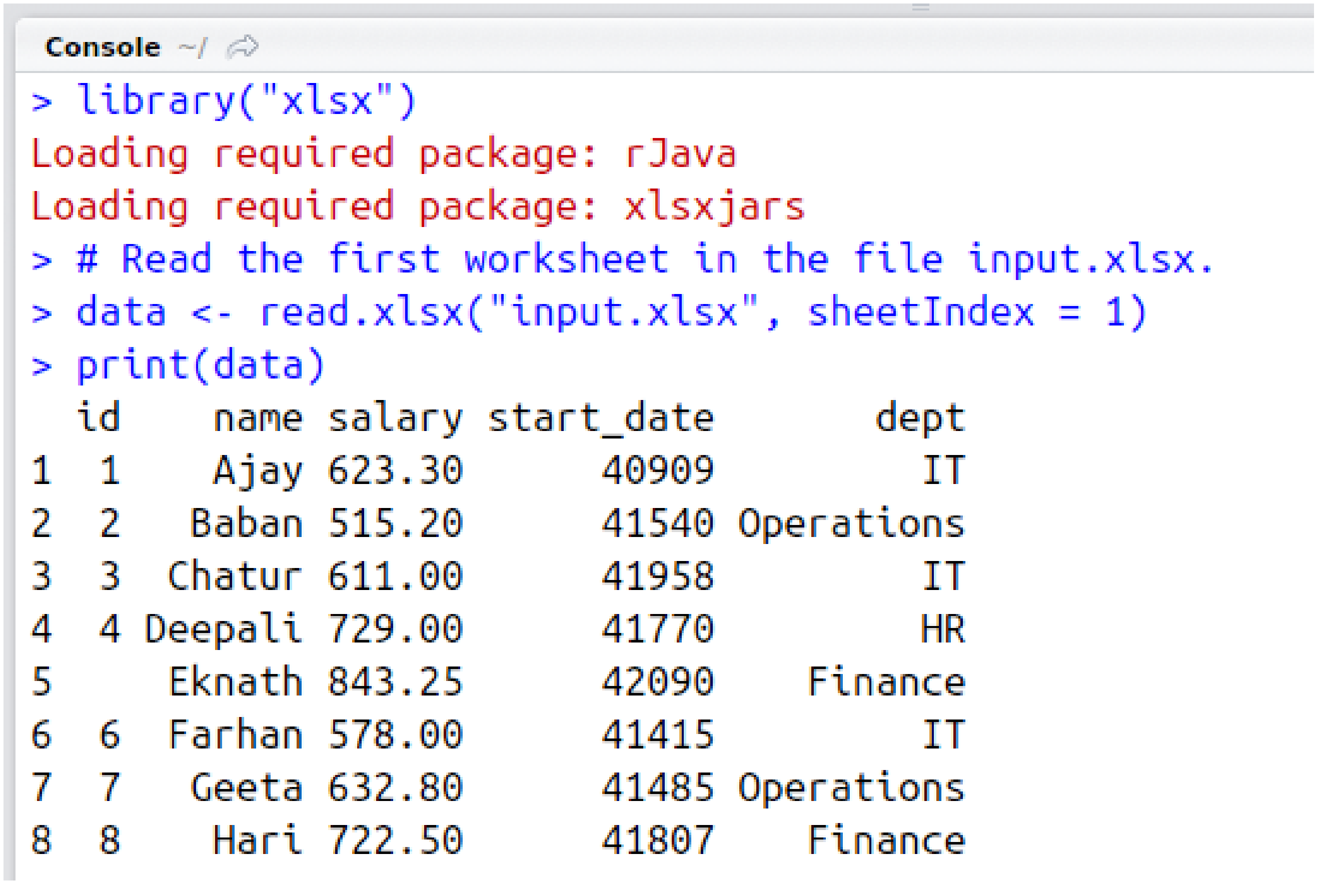
	A	B	C	D	E	F
1	id	name	salary	start date	dept	
2	1	Ajay	623.3	1/1/2012	IT	
3	2	Baban	515.2	9/23/2013	Operations	
4	3	Chatur	611	11/15/2014	IT	
5	4	Deepali	729	5/11/2014	HR	
6		Eknath	843.25	3/27/2015	Finance	
7	6	Farhan	578	5/21/2013	IT	
8	7	Geeta	632.8	7/30/2013	Operations	
9	8	Hari	722.5	6/17/2014	Finance	
10						
11						
12						
13						

Reading xlsx file

- The input.xlsx is read by using the read.xlsx() function as shown below. The result is stored as a data frame in the R environment.

```
# Read the first worksheet in the file input.xlsx.
data <- read.xlsx("input.xlsx", sheetIndex = 1)
print(data)
```

Reading xlsx file



Console ~/

```
> library("xlsx")
Loading required package: rJava
Loading required package: xlsxjars
> # Read the first worksheet in the file input.xlsx.
> data <- read.xlsx("input.xlsx", sheetIndex = 1)
> print(data)
  id     name salary start_date       dept
1  1     Ajay 623.30      40909         IT
2  2    Baban 515.20      41540 Operations
3  3   Chatur 611.00      41958         IT
4  4 Deepali 729.00      41770         HR
5      Eknath 843.25      42090    Finance
6  6   Farhan 578.00      41415         IT
7  7    Geeta 632.80      41485 Operations
8  8     Hari 722.50      41807    Finance
```

tusharkute.com

The XML file

- XML is a file format which shares both the file format and the data on the World Wide Web, intranets, and elsewhere using standard ASCII text.
- It stands for Extensible Markup Language (XML). Similar to HTML it contains markup tags. But unlike HTML where the markup tag describes structure of the page, in xml the markup tags describe the meaning of the data contained into the file.
- You can read a xml file in R using the "XML" package. This package can be installed using following command.

```
install.packages("XML")
```

Input data

- Create a XMl file by copying the below data into a text editor. Save the file with a .xml extension and choosing the file type as all files(*.*).

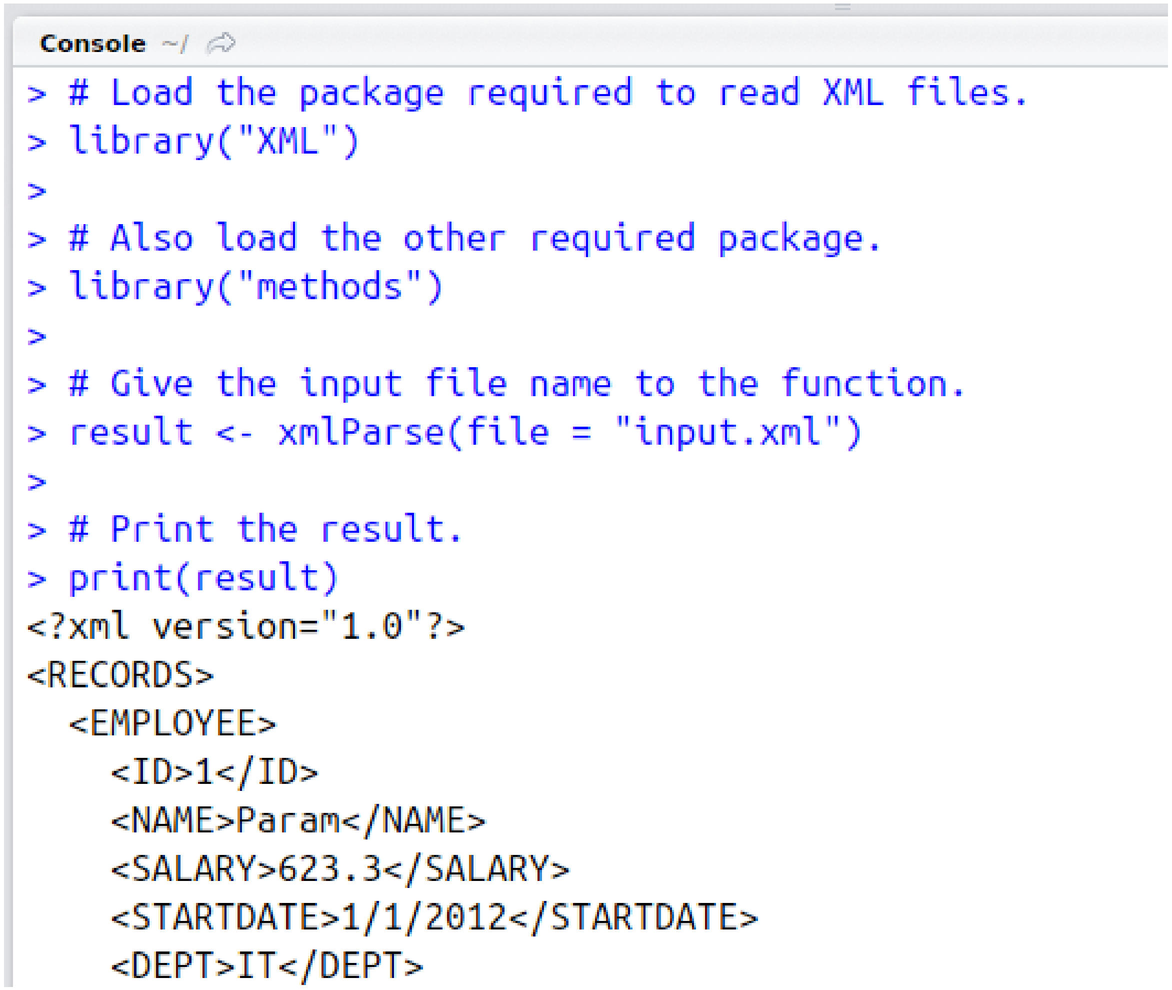
```
<RECORDS>
   <EMPLOYEE>
      <ID>1</ID>
      <NAME>Param</NAME>
      <SALARY>623.3</SALARY>
      <STARTDATE>1/1/2012</STARTDATE>
      <DEPT>IT</DEPT>
   </EMPLOYEE>            …….
```

Reading XML file

- The xml file is read by R using the function xmlParse(). It is stored as a list in R.

```
# Load the package required to read XML files.
library("XML")

# Also load the other required package.
library("methods")

# Give the input file name to the function.
result <- xmlParse(file = "input.xml")

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

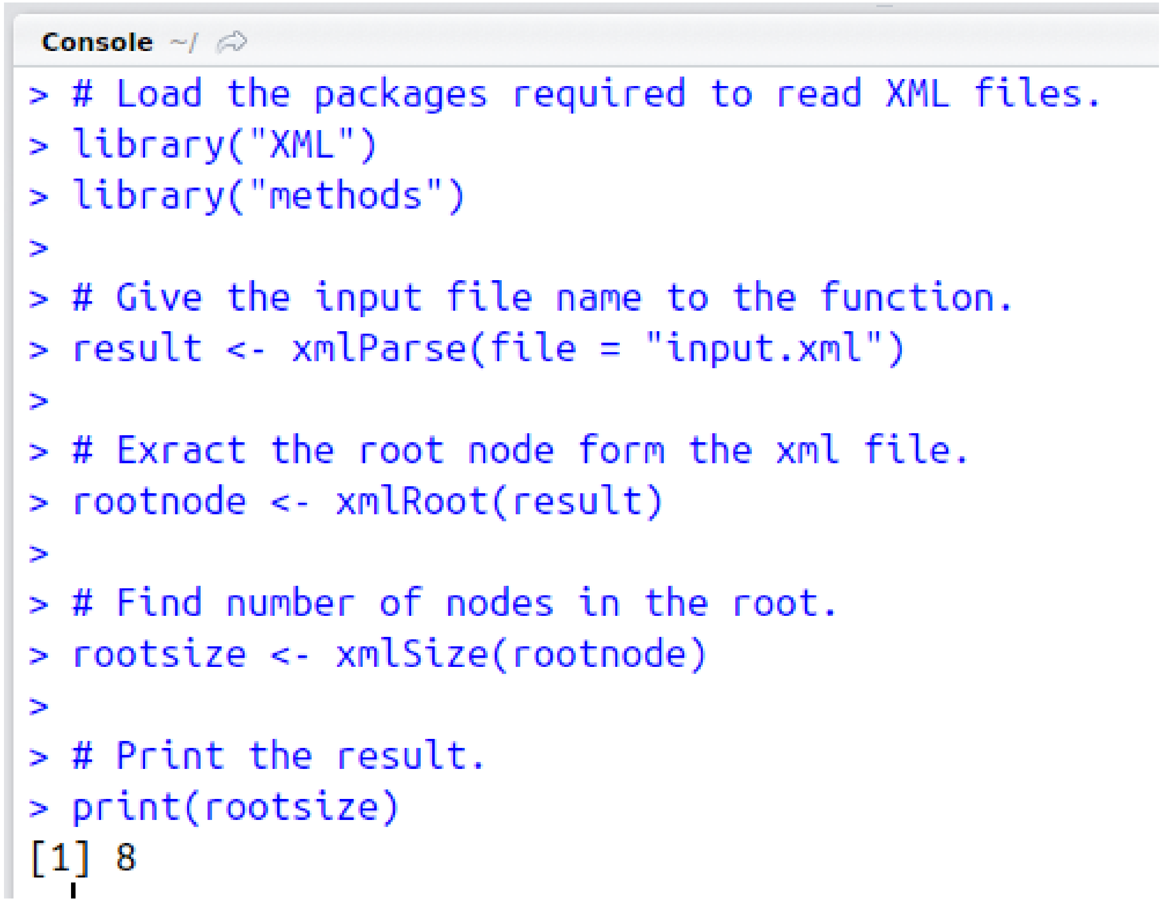
Reading XML file



Console ~/

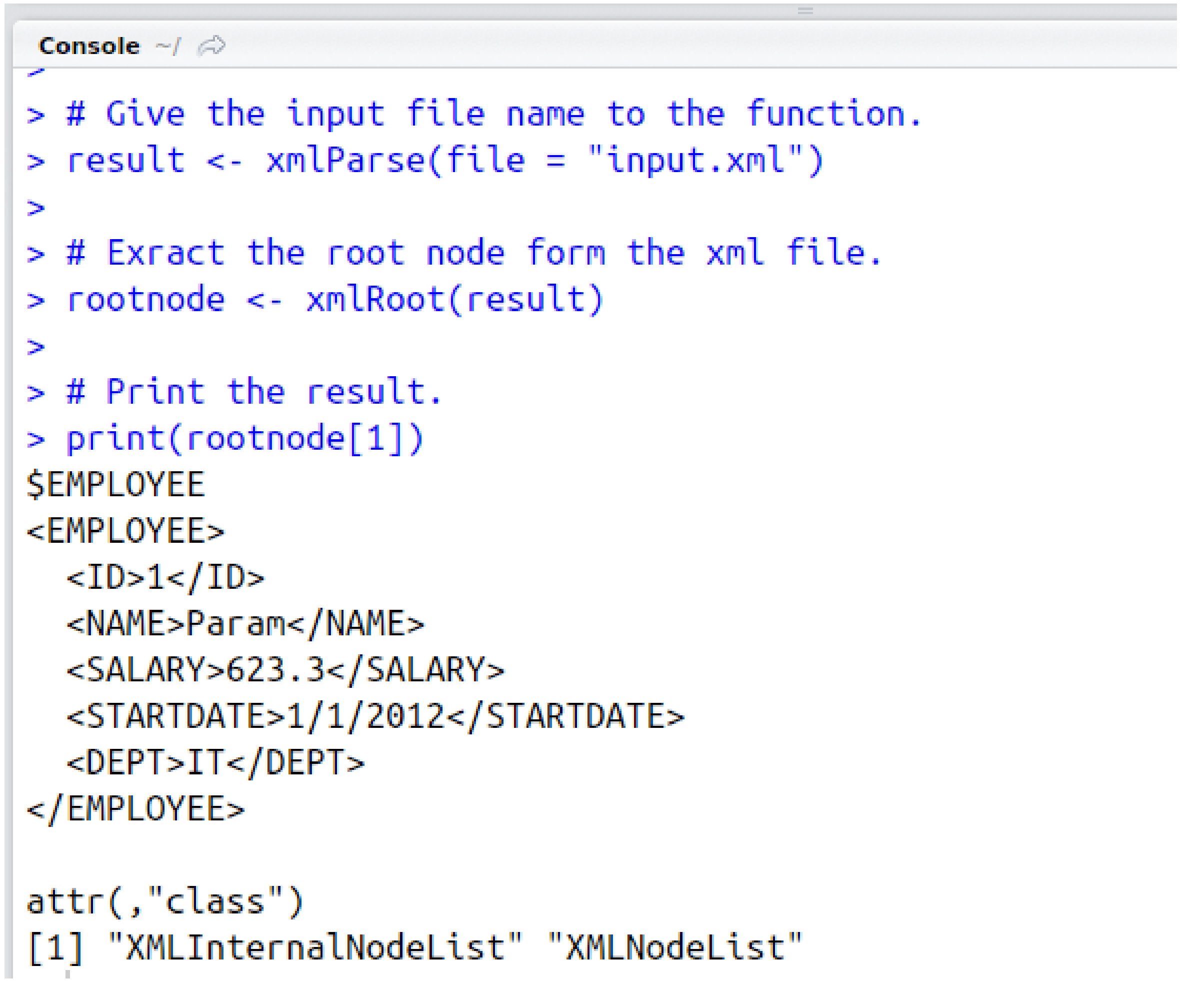
```
> # Load the package required to read XML files.
> library("XML")
>
> # Also load the other required package.
> library("methods")
>
> # Give the input file name to the function.
> result <- xmlParse(file = "input.xml")
>
> # Print the result.
> print(result)
<?xml version="1.0"?>
<RECORDS>
  <EMPLOYEE>
    <ID>1</ID>
    <NAME>Param</NAME>
    <SALARY>623.3</SALARY>
    <STARTDATE>1/1/2012</STARTDATE>
    <DEPT>IT</DEPT>
```

Get number of nodes



```
> # Load the packages required to read XML files.
> library("XML")
> library("methods")
>
> # Give the input file name to the function.
> result <- xmlParse(file = "input.xml")
>
> # Exract the root node form the xml file.
> rootnode <- xmlRoot(result)
>
> # Find number of nodes in the root.
> rootsize <- xmlSize(rootnode)
>
> # Print the result.
> print(rootsize)
[1] 8
```

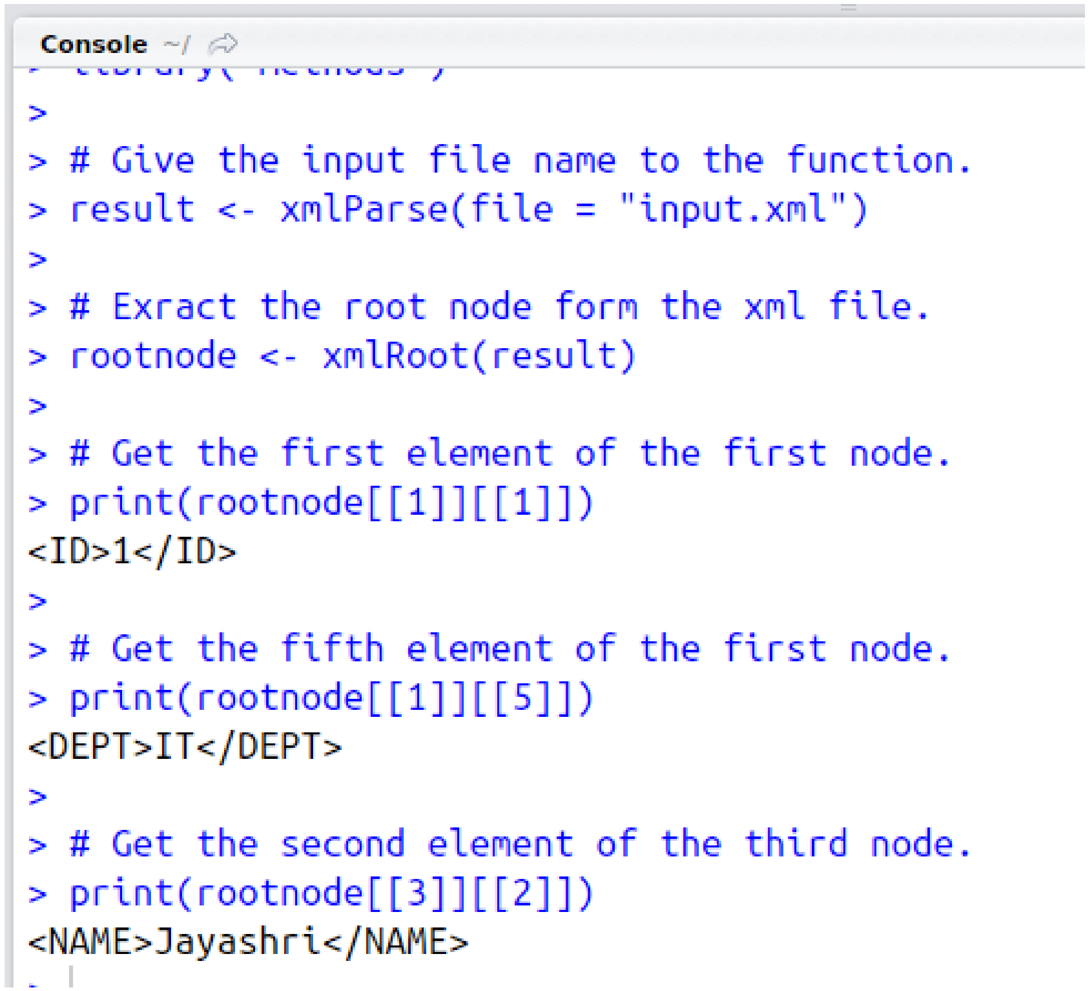
Get details of first node



Console ~/

```
> # Give the input file name to the function.
> result <- xmlParse(file = "input.xml")
>
> # Exract the root node form the xml file.
> rootnode <- xmlRoot(result)
>
> # Print the result.
> print(rootnode[1])
$EMPLOYEE
<EMPLOYEE>
  <ID>1</ID>
  <NAME>Param</NAME>
  <SALARY>623.3</SALARY>
  <STARTDATE>1/1/2012</STARTDATE>
  <DEPT>IT</DEPT>
</EMPLOYEE>

attr(,"class")
[1] "XMLInternalNodeList" "XMLNodeList"
```

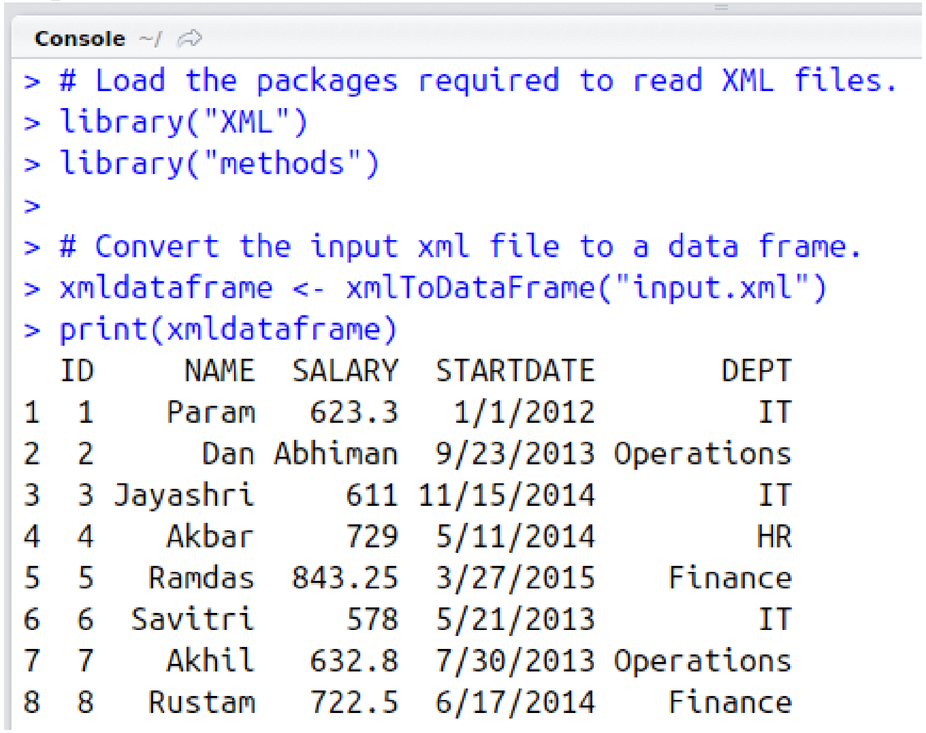
Get different elements of the node



```
>
> # Give the input file name to the function.
> result <- xmlParse(file = "input.xml")
>
> # Exract the root node form the xml file.
> rootnode <- xmlRoot(result)
>
> # Get the first element of the first node.
> print(rootnode[[1]][[1]])
<ID>1</ID>
>
> # Get the fifth element of the first node.
> print(rootnode[[1]][[5]])
<DEPT>IT</DEPT>
>
> # Get the second element of the third node.
> print(rootnode[[3]][[2]])
<NAME>Jayashri</NAME>
```

XML to data frame

```
# Load the packages required to read XML files.
library("XML")
library("methods")
# Convert the input xml file to a data frame.
xmldataframe <- xmlToDataFrame("input.xml")
print(xmldataframe)
```



```
> # Load the packages required to read XML files.
> library("XML")
> library("methods")
>
> # Convert the input xml file to a data frame.
> xmldataframe <- xmlToDataFrame("input.xml")
> print(xmldataframe)
  ID     NAME  SALARY  STARTDATE       DEPT
1  1    Param   623.3   1/1/2012         IT
2  2      Dan Abhiman  9/23/2013 Operations
3  3 Jayashri     611 11/15/2014         IT
4  4    Akbar     729  5/11/2014         HR
5  5   Ramdas  843.25  3/27/2015    Finance
6  6  Savitri     578  5/21/2013         IT
7  7    Akhil   632.8  7/30/2013 Operations
8  8   Rustam   722.5  6/17/2014    Finance
```

Web data

- Many websites provide data for consumption by its users.
- For example the World Health Organization(WHO) provides reports on health and medical information in the form of CSV, txt and XML files.
- Using R programs, we can programmatically extract specific data from such websites.
- Some packages in R which are used to scrap data from the web are – "Rcurl", “XML", and "stringr".
- They are used to connect to the URL’s, identify required links for the files and download them to the local environment.

Install and input

- **Install R Packages**
  - The following packages are required for processing the URL’s and links to the files. If they are not available in your R Environment, you can install them using following commands.

```
install.packages("RCurl")
install.packages("XML")
install.packages("stringr")
install.packages("plyr")
```

- **Input Data**
  - We will visit the URL <http://www.geos.ed.ac.uk/~weather/jcmb_ws/> weather data and download the CSV files using R for the year 2015.

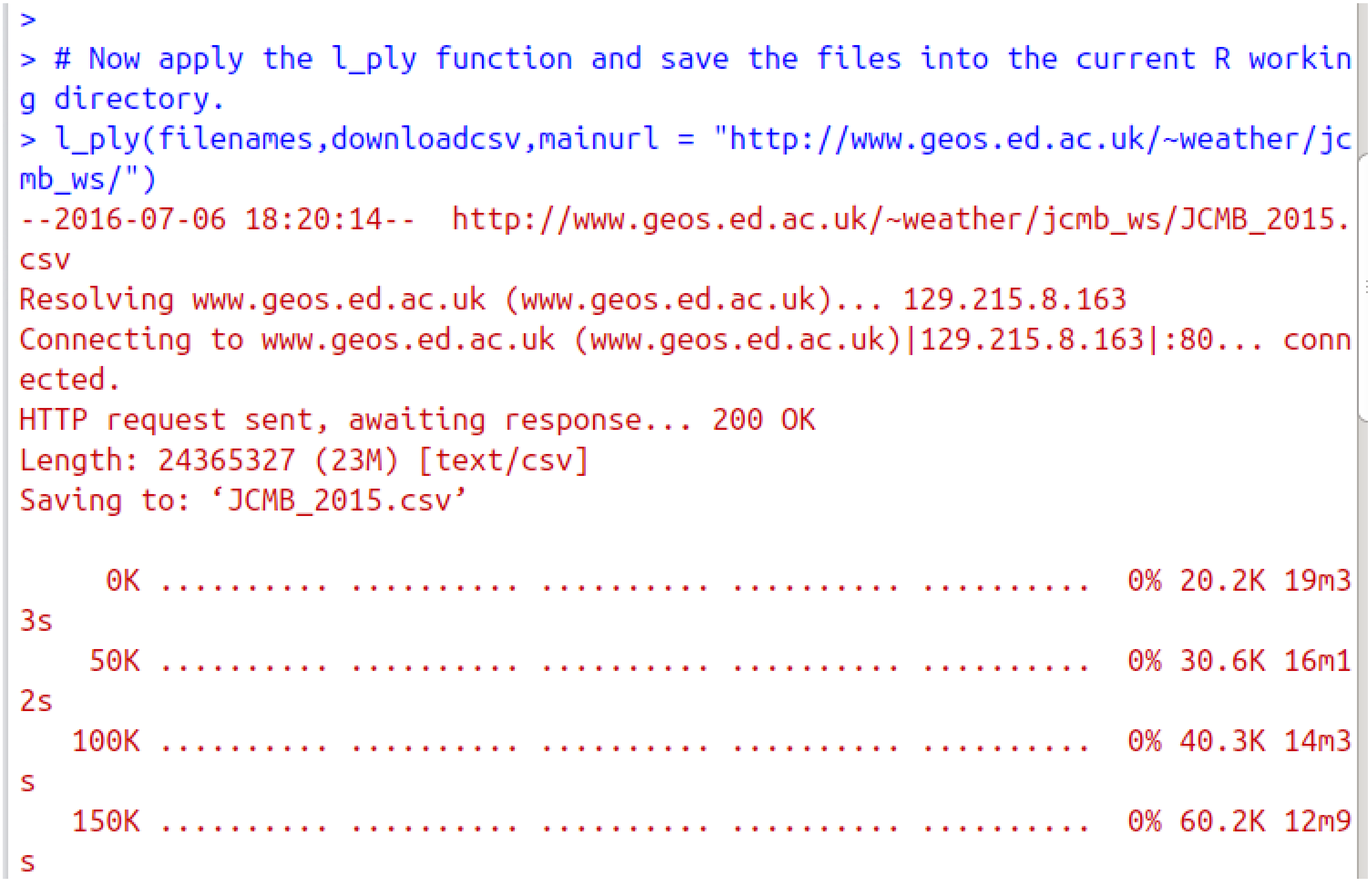
Example:

- We will use the function getHTMLLinks() to gather the URLs of the files.
- Then we will use the function download.file() to save the files to the local system.
- As we will be applying the same code again and again for multiple files, we will create a function to be called multiple times.
- The filenames are passed as parameters in form of a R list object to this function.

Example:

```
# Read the URL.
url <- "http://www.geos.ed.ac.uk/~weather/jcmb_ws/"

# Gather the html links present in the webpage.
links <- getHTMLLinks(url)

# Identify only the links which point to the JCMB 2015 files.
filenames <- links[str_detect(links, "JCMB_2015")]

# Store the file names as a list.
filenames_list <- as.list(filenames)

# Create a function to download the files by passing the URL and filename list.
downloadcsv <- function (mainurl,filename) {
   filedetails <- str_c(mainurl,filename)
   download.file(filedetails,filename)
}

# Now apply the l_ply function and save the files into the current R working
directory.
l_ply(filenames,downloadcsv,mainurl =
"http://www.geos.ed.ac.uk/~weather/jcmb_ws/")
```

Web data download



```
>
> # Now apply the l_ply function and save the files into the current R workin
g directory.
> l_ply(filenames,downloadcsv,mainurl = "http://www.geos.ed.ac.uk/~weather/jc
mb_ws/")
--2016-07-06 18:20:14--  http://www.geos.ed.ac.uk/~weather/jcmb_ws/JCMB_2015.
csv
Resolving www.geos.ed.ac.uk (www.geos.ed.ac.uk)... 129.215.8.163
Connecting to www.geos.ed.ac.uk (www.geos.ed.ac.uk)|129.215.8.163|:80... conn
ected.
HTTP request sent, awaiting response... 200 OK
Length: 24365327 (23M) [text/csv]
Saving to: ‘JCMB_2015.csv’

     0K .......... .......... .......... .......... ..........  0% 20.2K 19m3
3s
    50K .......... .......... .......... .......... ..........  0% 30.6K 16m1
2s
   100K .......... .......... .......... .......... ..........  0% 40.3K 14m3
s
   150K .......... .......... .......... .......... ..........  0% 60.2K 12m9
s
```

Verify file download

- After running the above code, you can locate the following files in the current R working directory.
  - "JCMB_2015.csv"
  - "JCMB_2015_Apr.csv"
  - "JCMB_2015_Feb.csv"
  - "JCMB_2015_Jan.csv"
  - "JCMB_2015_Mar.csv"

Connecting to MySQL

- The data is Relational database systems are stored in a normalized format.
- So, to carry out statistical computing we will need very advanced and complex Sql queries.
- But R can connect easily to many relational databases like MySql, Oracle, Sql server etc. and fetch records from them as a data frame.
- Once the data is available in the R environment, it becomes a normal R data set and can be manipulated or analyzed using all the powerful packages and functions.

The RMySQL package

- R has a built-in package named "RMySQL" which provides native connectivity between with MySql database.
- You can install this package in the R environment using the following command.

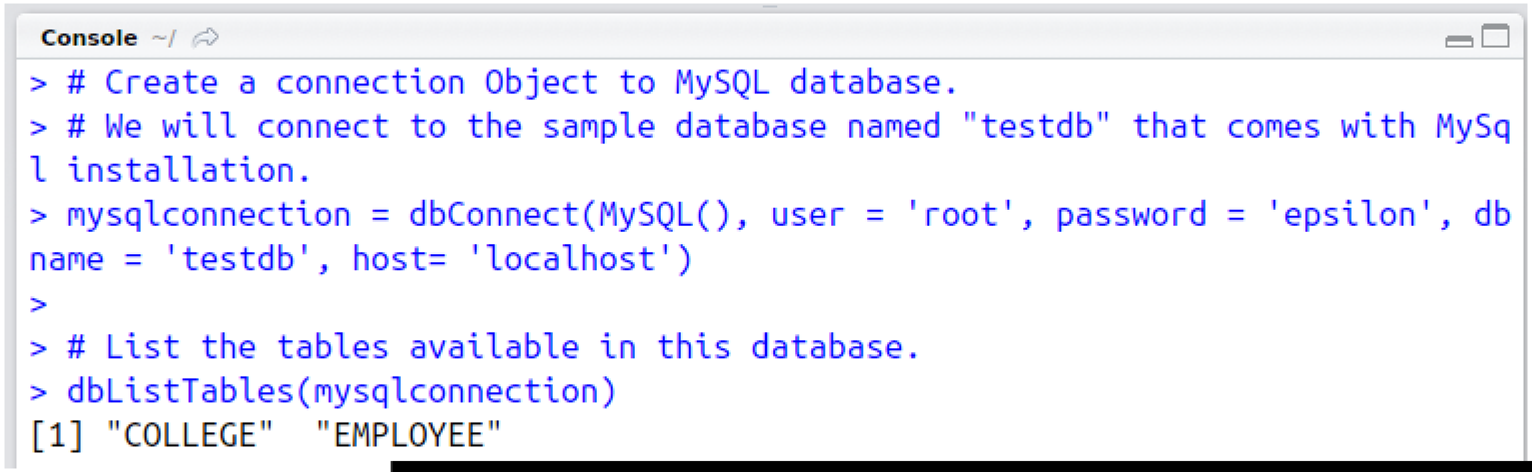
```
install.packages("RMySQL")
```

Connecting R to MySQL

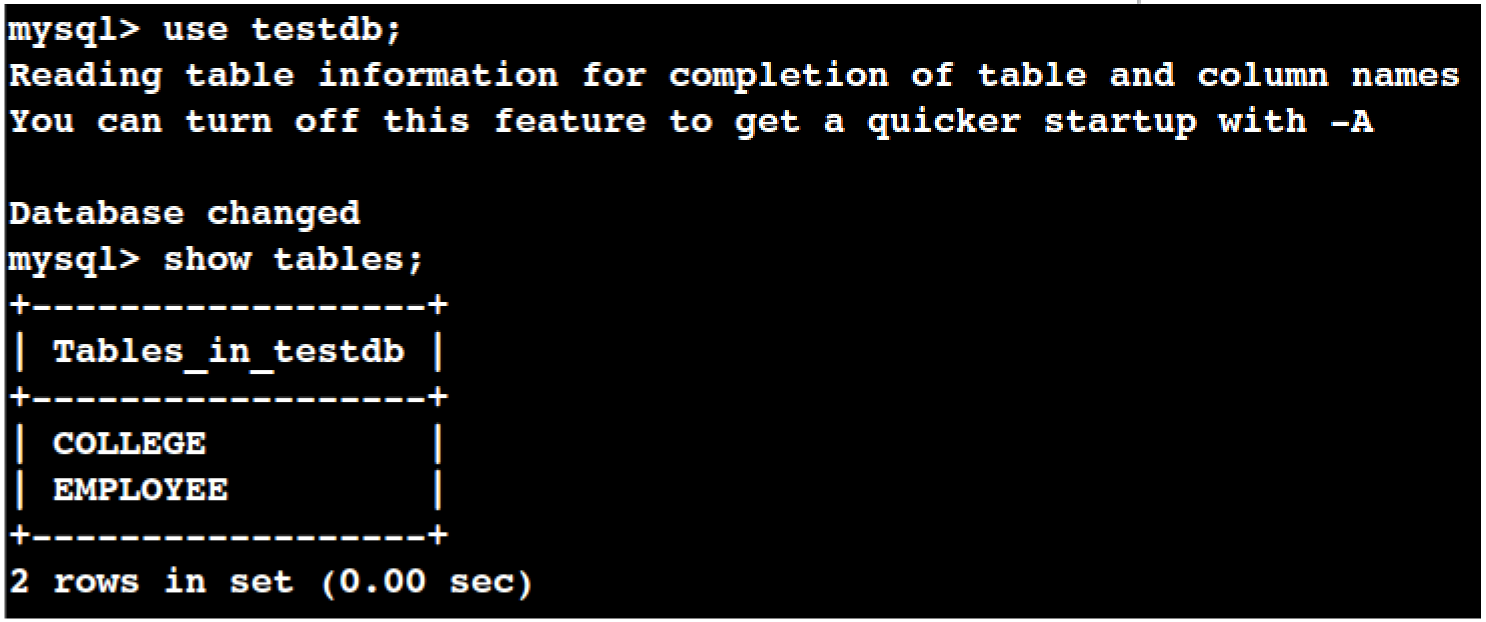
- Once the package is installed we create a connection object in R to connect to the database.
- It takes the username, password, database name and host name as input.

```
# Create a connection Object to MySQL database.
# We will connect to the sample database named "testdb"
that comes with MySql installation.
mysqlconnection = dbConnect(MySQL(), user = 'root',
password = 'epsilon', dbname = 'testdb', host=
'localhost')


# List the tables available in this database.
dbListTables(mysqlconnection)
```

Connecting R to MySQL



```
> # Create a connection Object to MySQL database.
> # We will connect to the sample database named "testdb" that comes with MySq
l installation.
> mysqlconnection = dbConnect(MySQL(), user = 'root', password = 'epsilon', db
name = 'testdb', host= 'localhost')
>
> # List the tables available in this database.
> dbListTables(mysqlconnection)
[1] "COLLEGE"  "EMPLOYEE"
```

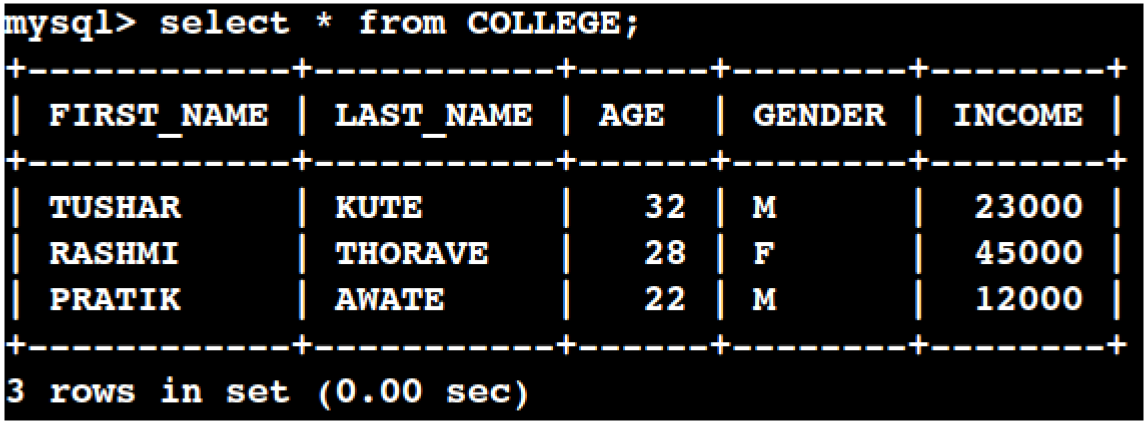


```
mysql> use testdb;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-------------------+
| Tables_in_testdb |
+-------------------+
| COLLEGE          |
| EMPLOYEE         |
+-------------------+
2 rows in set (0.00 sec)
```

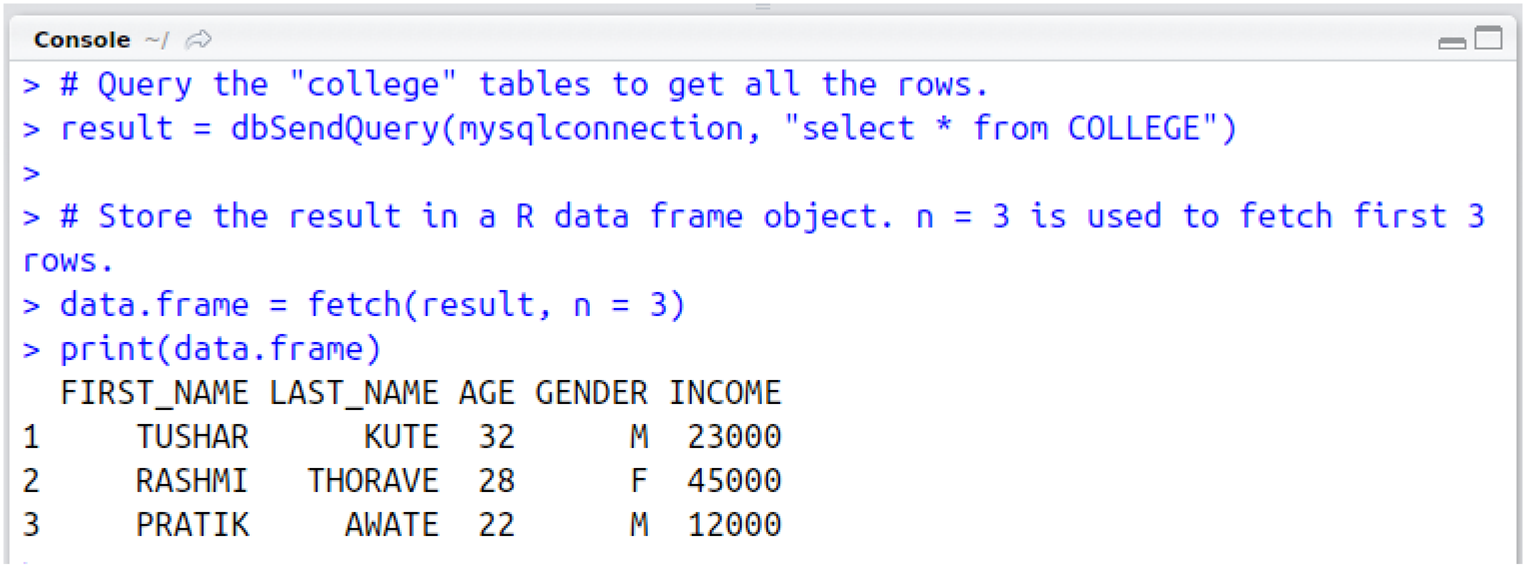
Querying the tables

```
# Query the "college" tables to get all the rows.
result = dbSendQuery(mysqlconnection, "select * 
from COLLEGE")


# Store the result in a R data frame object. n = 3 
is used to fetch first 3 rows.
data.frame = fetch(result, n = 3)
print(data.frame)
```

Querying the tables



```
mysql> select * from COLLEGE;
+------------+-----------+------+--------+--------+
| FIRST_NAME | LAST_NAME | AGE  | GENDER | INCOME |
+------------+-----------+------+--------+--------+
| TUSHAR     | KUTE      |   32 | M      |  23000 |
| RASHMI     | THORAVE   |   28 | F      |  45000 |
| PRATIK     | AWATE     |   22 | M      |  12000 |
+------------+-----------+------+--------+--------+
3 rows in set (0.00 sec)
```

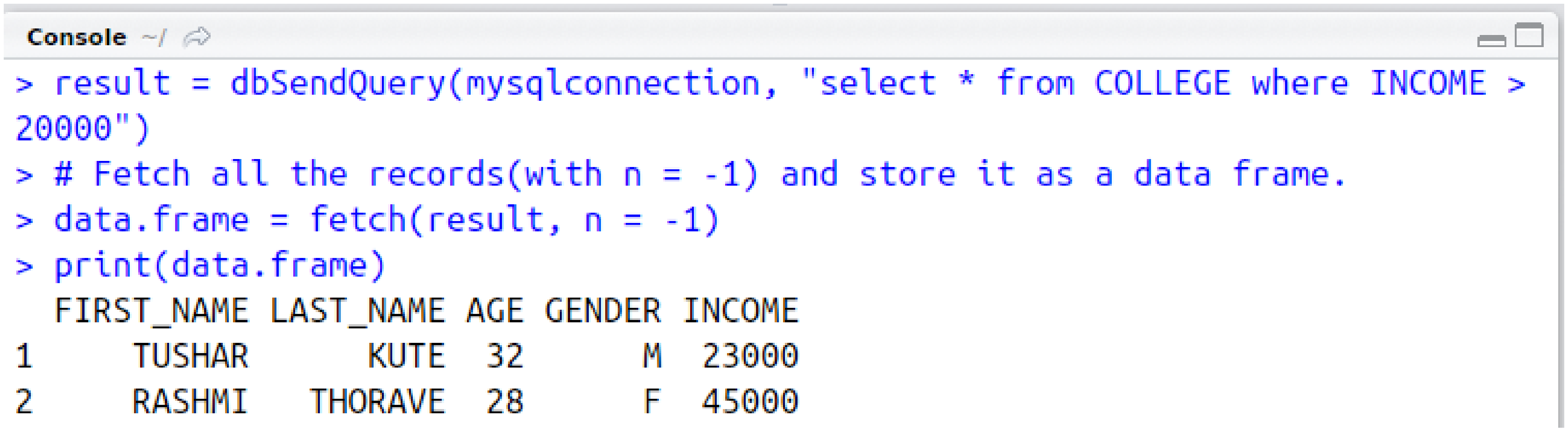


```
> # Query the "college" tables to get all the rows.
> result = dbSendQuery(mysqlconnection, "select * from COLLEGE")
>
> # Store the result in a R data frame object. n = 3 is used to fetch first 3
rows.
> data.frame = fetch(result, n = 3)
> print(data.frame)
  FIRST_NAME LAST_NAME AGE GENDER INCOME
1     TUSHAR      KUTE  32      M  23000
2     RASHMI   THORAVE  28      F  45000
3     PRATIK     AWATE  22      M  12000
```

Querying the filters

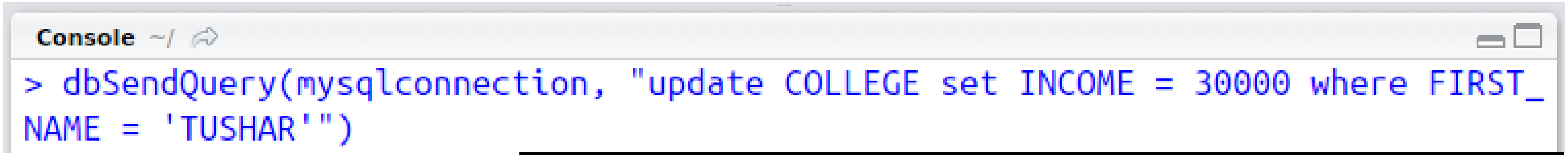
```
result = dbSendQuery(mysqlconnection, "select
* from COLLEGE where INCOME > 20000")


# Fetch all the records(with n = -1) and store
it as a data frame.

data.frame = fetch(result, n = -1)

print(data.frame)
```

Querying the filters

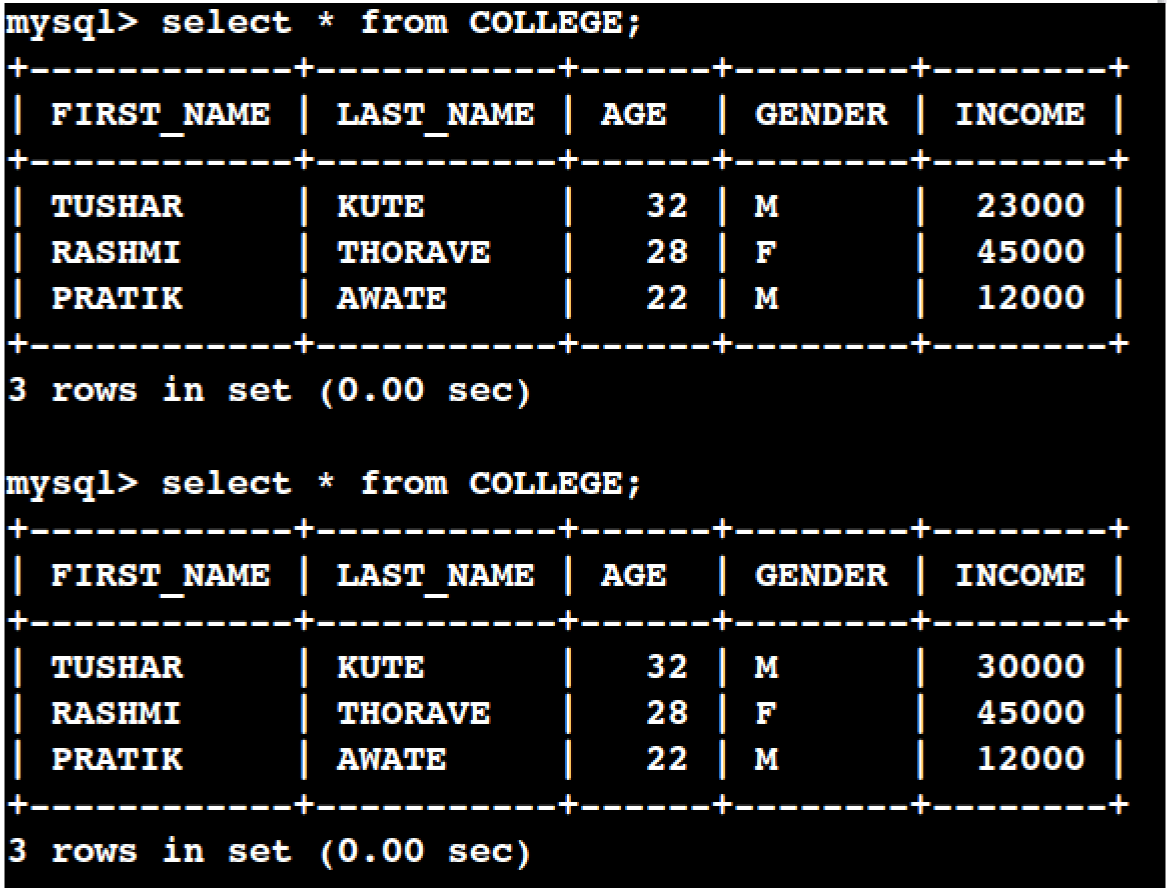

Console ~/

```
> result = dbSendQuery(mysqlconnection, "select * from COLLEGE where INCOME >
20000")
> # Fetch all the records(with n = -1) and store it as a data frame.
> data.frame = fetch(result, n = -1)
> print(data.frame)
  FIRST_NAME LAST_NAME AGE GENDER INCOME
1     TUSHAR      KUTE  32      M  23000
2     RASHMI   THORAVE  28      F  45000
```

Update

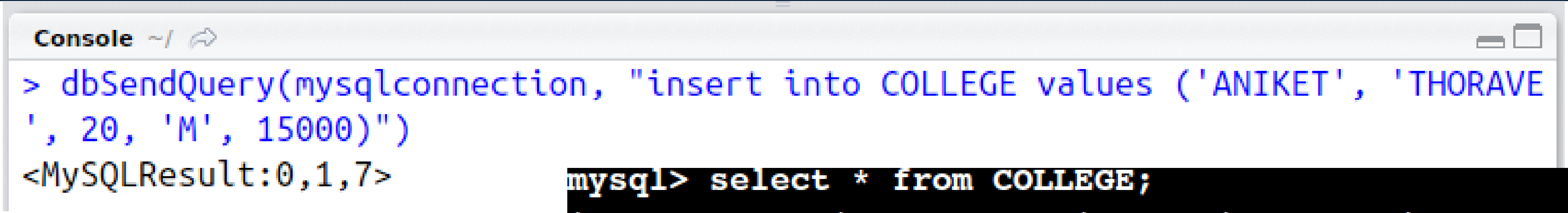


```
> dbSendQuery(mysqlconnection, "update COLLEGE set INCOME = 30000 where FIRST_
NAME = 'TUSHAR'")
```

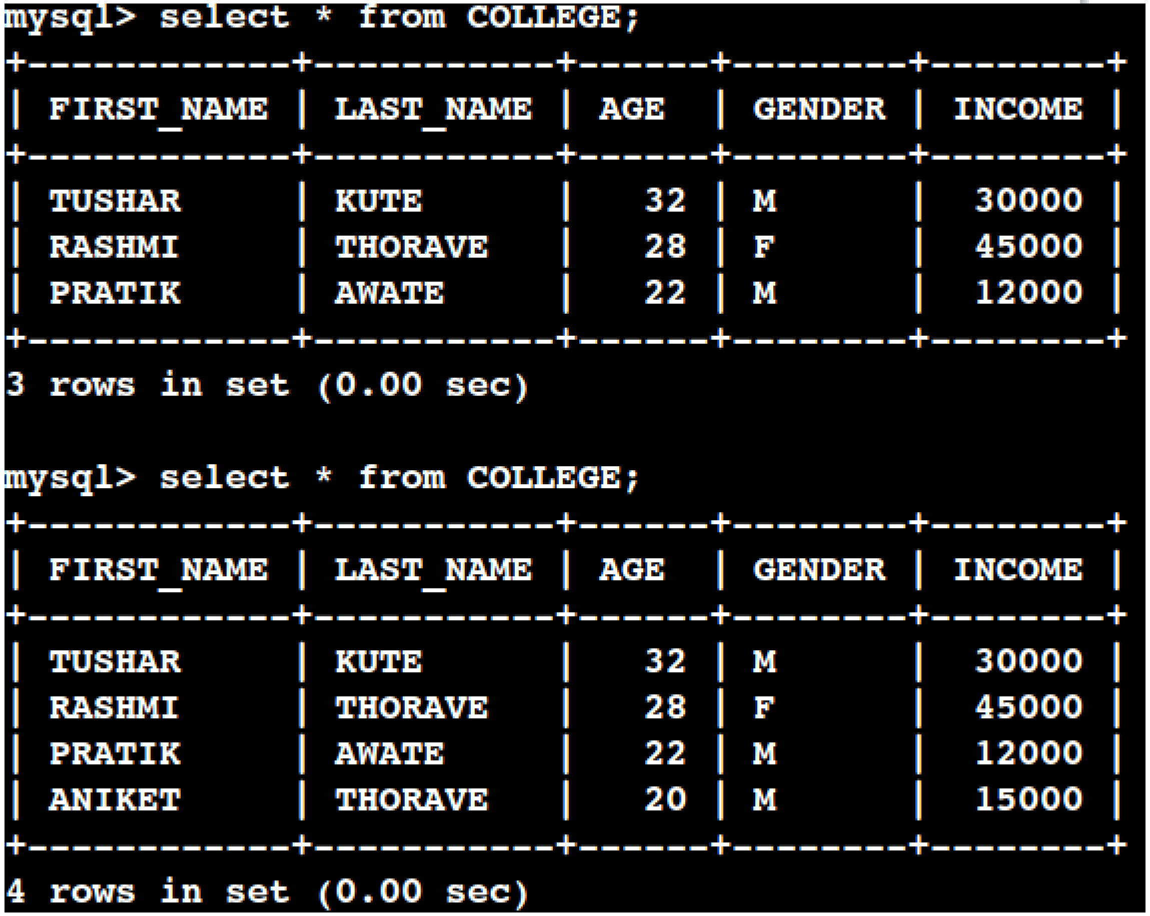


```
mysql> select * from COLLEGE;
+------------+-----------+------+--------+--------+
| FIRST_NAME | LAST_NAME | AGE  | GENDER | INCOME |
+------------+-----------+------+--------+--------+
| TUSHAR     | KUTE      |   32 | M      |  23000 |
| RASHMI     | THORAVE   |   28 | F      |  45000 |
| PRATIK     | AWATE     |   22 | M      |  12000 |
+------------+-----------+------+--------+--------+
3 rows in set (0.00 sec)

mysql> select * from COLLEGE;
+------------+-----------+------+--------+--------+
| FIRST_NAME | LAST_NAME | AGE  | GENDER | INCOME |
+------------+-----------+------+--------+--------+
| TUSHAR     | KUTE      |   32 | M      |  30000 |
| RASHMI     | THORAVE   |   28 | F      |  45000 |
| PRATIK     | AWATE     |   22 | M      |  12000 |
+------------+-----------+------+--------+--------+
3 rows in set (0.00 sec)
```

Insert

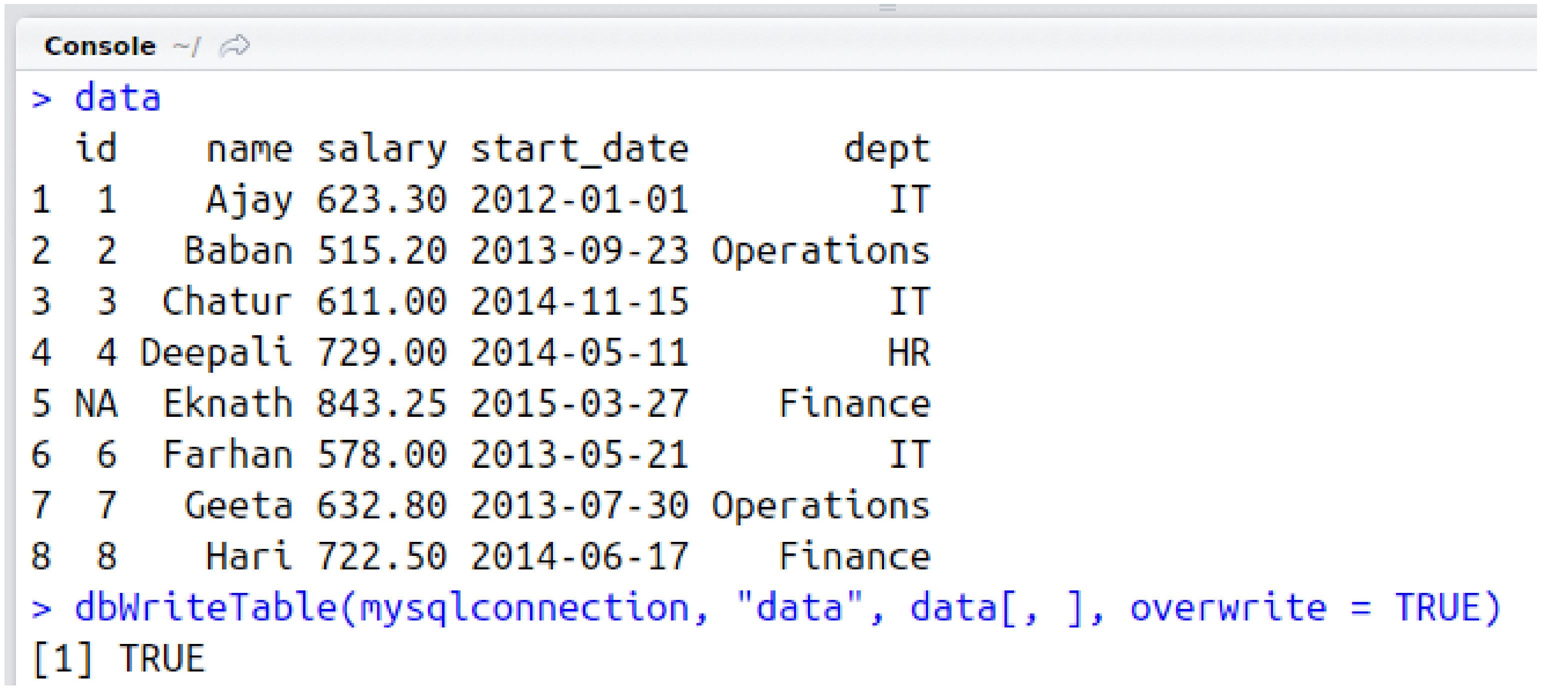


```
Console ~/
> dbSendQuery(mysqlconnection, "insert into COLLEGE values ('ANIKET', 'THORAVE
', 20, 'M', 15000)")
<MySQLResult:0,1,7>
```



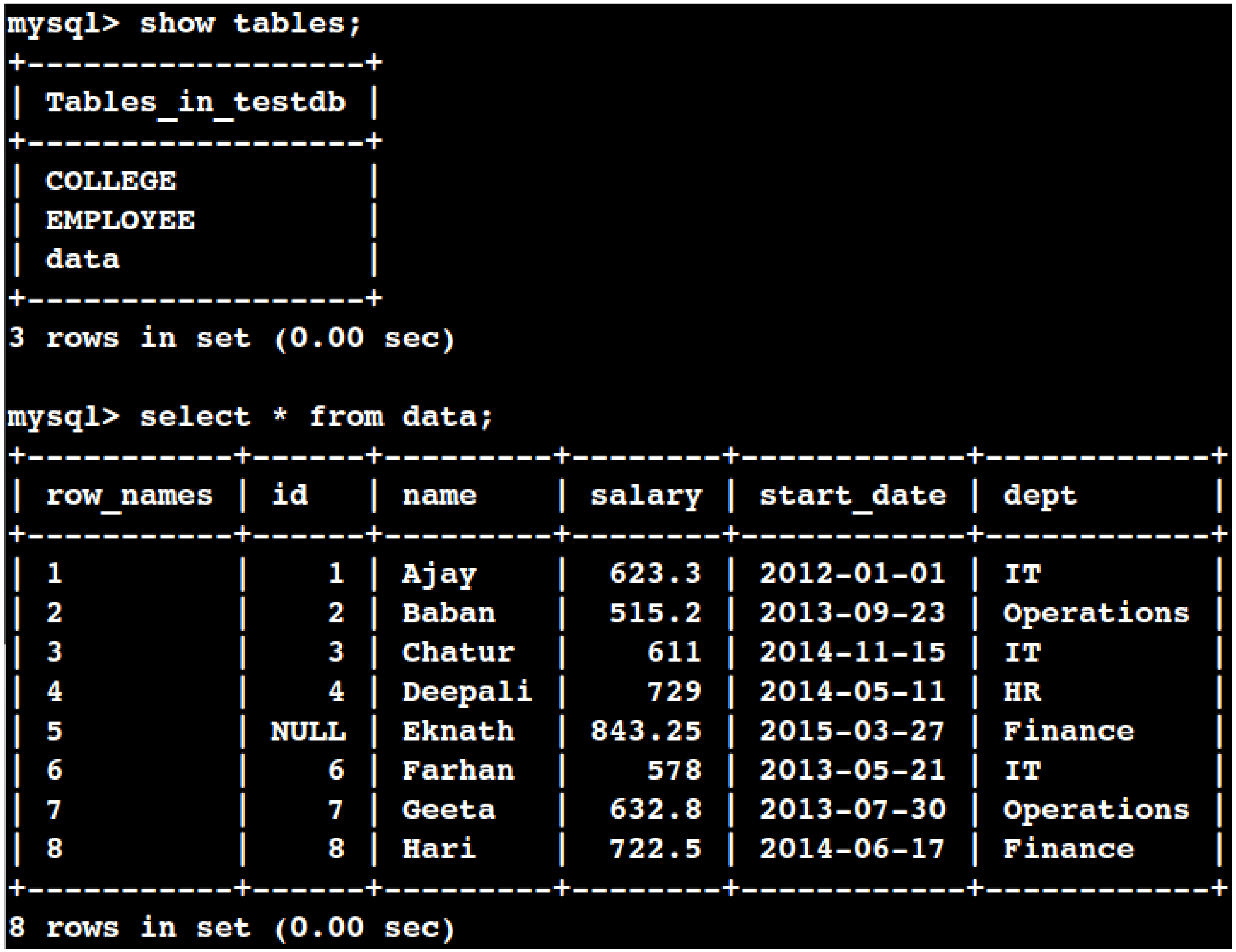
```
mysql> select * from COLLEGE;
+------------+-----------+-----+--------+--------+
| FIRST_NAME | LAST_NAME | AGE | GENDER | INCOME |
+------------+-----------+-----+--------+--------+
| TUSHAR     | KUTE      |  32 | M      |  30000 |
| RASHMI     | THORAVE   |  28 | F      |  45000 |
| PRATIK     | AWATE     |  22 | M      |  12000 |
+------------+-----------+-----+--------+--------+
3 rows in set (0.00 sec)

mysql> select * from COLLEGE;
+------------+-----------+-----+--------+--------+
| FIRST_NAME | LAST_NAME | AGE | GENDER | INCOME |
+------------+-----------+-----+--------+--------+
| TUSHAR     | KUTE      |  32 | M      |  30000 |
| RASHMI     | THORAVE   |  28 | F      |  45000 |
| PRATIK     | AWATE     |  22 | M      |  12000 |
| ANIKET     | THORAVE   |  20 | M      |  15000 |
+------------+-----------+-----+--------+--------+
4 rows in set (0.00 sec)
```

Insert



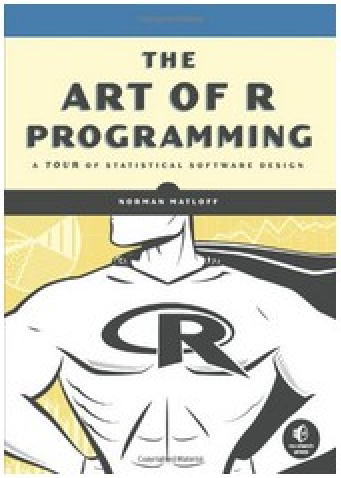
```
> data
  id    name salary start_date       dept
1  1    Ajay 623.30 2012-01-01         IT
2  2   Baban 515.20 2013-09-23 Operations
3  3  Chatur 611.00 2014-11-15         IT
4  4 Deepali 729.00 2014-05-11         HR
5 NA  Eknath 843.25 2015-03-27    Finance
6  6  Farhan 578.00 2013-05-21         IT
7  7   Geeta 632.80 2013-07-30 Operations
8  8    Hari 722.50 2014-06-17    Finance
> dbWriteTable(mysqlconnection, "data", data[, ], overwrite = TRUE)
[1] TRUE
```

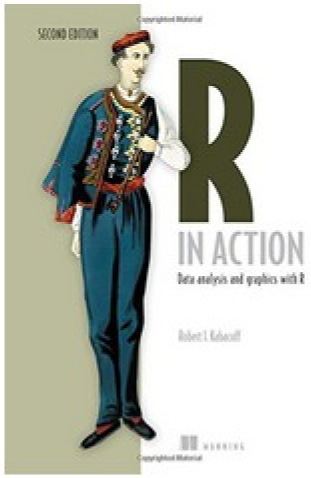
Insert

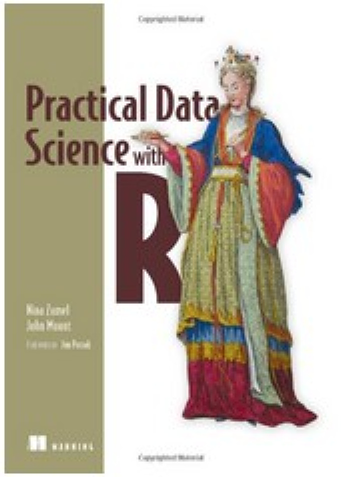


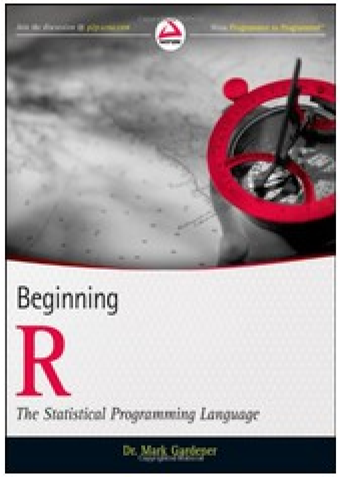
```
mysql> show tables;
+-------------------+
| Tables_in_testdb |
+-------------------+
| COLLEGE           |
| EMPLOYEE          |
| data              |
+-------------------+
3 rows in set (0.00 sec)

mysql> select * from data;
+-----------+------+---------+--------+------------+------------+
| row_names | id   | name    | salary | start_date | dept       |
+-----------+------+---------+--------+------------+------------+
| 1         |    1 | Ajay    |  623.3 | 2012-01-01 | IT         |
| 2         |    2 | Baban   |  515.2 | 2013-09-23 | Operations |
| 3         |    3 | Chatur  |    611 | 2014-11-15 | IT         |
| 4         |    4 | Deepali |    729 | 2014-05-11 | HR         |
| 5         | NULL | Eknath  | 843.25 | 2015-03-27 | Finance    |
| 6         |    6 | Farhan  |    578 | 2013-05-21 | IT         |
| 7         |    7 | Geeta   |  632.8 | 2013-07-30 | Operations |
| 8         |    8 | Hari    |  722.5 | 2014-06-17 | Finance    |
+-----------+------+---------+--------+------------+------------+
8 rows in set (0.00 sec)
```

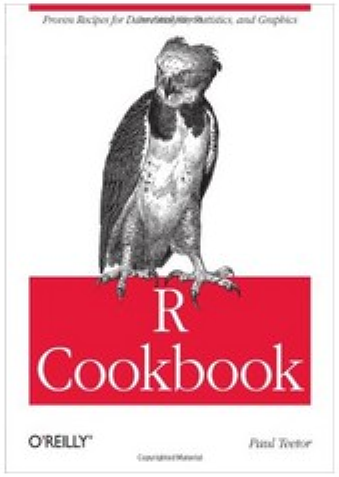
Useful resources











Thank you

Web Resources
http://mitu.co.in
http://tusharkute.com

Blogs
http://digitallocha.blogspot.in
http://kyamputar.blogspot.in

tushar@tusharkute.com