

Introduction to R Software
Prof. Shalabh
Department of Mathematics and Statistics
Indian Institute of Technology, Kanpur

Lecture - 33
Importing CSV and Tabular Data Files

Welcome to the next lecture on the course introduction to R software. Now in this lecture, we are going to understand some concepts related to the import of data; what do we really mean by import of data. You see whenever you want to manipulate any data set; the data set has to be present on your computer and this data set may come from different sources first option is that you can create that data set yourself. For example, you open a spreadsheet and try to enter the data and even when you are trying to save the spreadsheet, there can be different forms to save that spreadsheet for example, comma separated values TXT format.

And say some other thing beside this thing it is also possible that data is available from some other source; these sources can be somebody already has entered the data in some other computer and you want to import that data on your computer where you want to do the analysis using the R software, another option can be that data is uploaded somewhere on some internet site and you want to download it or you want to work on that data set directly. So, these are different possible ways in which the data set can be available to us and in this lecture, we are going to concentrate that how to read this data set into the R software, once you can read the data set into the R software, then you can do all other manipulations which you have learnt in the earlier lectures and those which we are going to learn in the further lectures.

So, let us try to start our lecture first thing comes that whenever you want to work with a data file that data file has to be located in some directory on your computer and you have to instruct the R software to read the data from that directory only there is a default directory in the R software and it is not always possible to put the data in that default directory, but we would like to put the data at a place which is convenient to us. So, for example, if you want to read the path of the directory in the R, then how to get it done means first of all once you start the R you would like to see; where is the working directory from where or say from which location this R software is fetching the data.

(Refer Slide Time: 03:38)

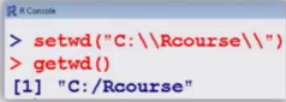
Setting up directories

❑ We can change the current working directory as follows:
> `setwd("<location of the dataset>")`
set working directory

Example:
> `setwd("C:/Rcourse/")`
or
> `setwd("C:\\Rcourse\\")`

❑ The following command returns the current working directory:
> `getwd()`
[1] "C:/Rcourse/"

getwd() working directory

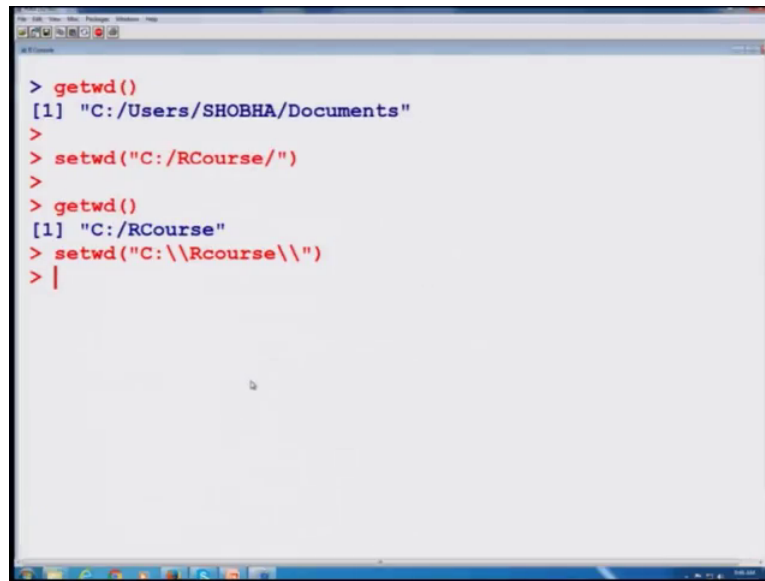


```
R Console
> setwd("C:\\Rcourse\\")
> getwd()
[1] "C:/Rcourse"
```

2

So, in order to do that thing we have say one option here which is called here get w d this means get working directory working directory and then we put an argument by this command, you can get the working directory in which the R is presently working. Now suppose I want to change this directory for that we have another command which is called as set w d; that means, set working directory and then inside the argument you have to specify the location of the data set inside the double quotes location has to be in terms of the path of the computer for example, in this course what I have done I can show you here that I have created a directory on the C drive you can see here where I am highlighting here; this I have created on the say here is C drive that you can see over here.

(Refer Slide Time: 05:24)

A screenshot of an R console window. The window title is "R Console". The console shows the following commands and their outputs:

```
> getwd()
[1] "C:/Users/SHOBHA/Documents"
>
> setwd("C:/RCourse/")
>
> getwd()
[1] "C:/RCourse"
> setwd("C:\\Rcourse\\")
> |
```

And inside this directory I have put some files which we are going to use in this course and before we try to go further, we try to see here in the R console that; what is the working directory. So, for that I start here say get w d and you can see here that it is trying to give us some directory here something like say C colon say backslash users SHOBHA backslash documents and now I want to change this directory. So, I have to give the name of the folder and the location for example, here I can write it here something like C colon slash front slash rather R course front slash and this has to be enclosed inside this double quotes. So, I try to use this path and I want to now change the working directory.

So, you can see here once I do this thing, now my working directory is set and now in case if I try to get here the working directory this will come out to be like this. So, you can see here in the highlighted part that this is the current working directory which I have changed and earlier the working directory was here right another option is that I can also use say another command to change the path which is the same thing, but in a different format means every system computer system has its own path for example, if you are working in windows or in Unix or in Linux or say in Macintosh you have to see how the path is described. So, I am not doing anything new, but I am simply trying to use the basic fundamental of that computer system to define the path.

(Refer Slide Time: 07:17)

Importing Data Files

Suppose we have some data on our computer and we want to import it in R.

Different formats of files can be read in R

- comma-separated values (CSV) data file,
- table file (TXT),
- Spreadsheet (e.g., MS Excel) file,
- files from other software like SPSS, Minitab etc.

↑ ↑

3

And then you have to just write down the path word there, right. So, now, this is how you can change the directory. Now I come to another aspect you want to import the data files in the R. So, you have 2 options either you try to create your data set and put this data file inside the working directory or you try to import the data set from other sources and then try to put it in the working directory. So, now, our question is very simple that we have got some data on our computer and we want to import it in the R software in R, there are different options in which the data can be read different types of formats of the file can be read directly into the R. So, what we are going to do here that we are going to discuss some of the popular formats of the file and we will see that how to read them in the R package.

So, some of the popular formats in which the R can read the data are say comma separated values CSV data file, we call it or table file they are they have an extension dot TXT some time, we have a file update of the data from some spreadsheet one of the popular source for spreadsheet is the software Microsoft Excel and the extension of the file is dot x l s or say dot x l s x right. So, those especially files can be read and beside those thing if we have a data files from others a software for example, software like SPSS software like Minitab, they have their own structure for entering the data. So, we can also say import the data files which are created in say Microsoft Excel SPSS Minitab or say any other software.

But you need to check it with the help menu that what type of sources can be read in the R software before you try to read any new type of data all right, there are certain ways to read the data that we will try to illustrate it here another option is that suppose the data is not available on your computer, but it is available somewhere on the website. So, in R it is also possible to read the data directly from the website right.

(Refer Slide Time: 09:58)

Importing Data Files

One can also read or upload the file from Internet site.

We can read the file containing rent index data from website:

<http://home.iitk.ac.in/~shalab/Rcourse/munichdata.asc>

as follows

```
> datamunich <- read.table(file=  
"http://home.iitk.ac.in/~shalab/Rcourse/munichdata.asc", header=TRUE)
```

File name is `munichdata.asc`

(Handwritten green notes: "file name" with address, header)

For example, I have kept here a data on my web page and the address of my web page is here home dot i i t k dot a c dot i n backslash tilde s h a l a b and inside that I have created a directory R course and then inside that I have kept a data file which is here munich data dot a s c.

So, the file name here is munich data dot a s c actually this file is containing the data on the rents in the Munich city Munich is the city in Germany where they have collected the data on the rents of the of accommodation. For example, of a apartment has 1 room, 2 room, 3 rooms; what is the area; what is the rent and so on. So, this data file is simply containing that data set well my objective is not to analyze the data contained in this file, but my objective here is simply to read a file from the internet source. So, I have kept this file over this address, I now in order to read such a data you will see there are different types of function read table file and so on, but here we are going to use the function here read dot table and then inside the arguments first of all I would write the

file name file name if this file name is not in the working directory then I have to write down the address also here.

And then separated by a comma I have to declare whether the file has a header or not. So, what I do here that I try to write this read table and inside the arguments I try to write the file name file is equal to this munich data dot a c and its address the address is here and I try to enclose it with inverted comma and then this file has got a header. So, I am trying to write down here header equal to true well at this moment you need not to get confused very shortly we are going to discuss that what are these commands like as read table or say header is true or header is false header is present header is absent and all those things we are going to discuss very soon.

So, here I simply want to illustrate that yes it is possible to read the data. So, what I try to do here that I try to read this data from the internet site. So, we try to copy this address and, but before I can show you that whether this data is here available or not here. So, you can see here, I have this copied this address over here and you can see here that this data is available here this is some values that you do not have to worry for this thing I can increase the font size also if you wish, but you can see that yes this data is available over here and now I try to read this data inside the data vector say data munich on the R console.

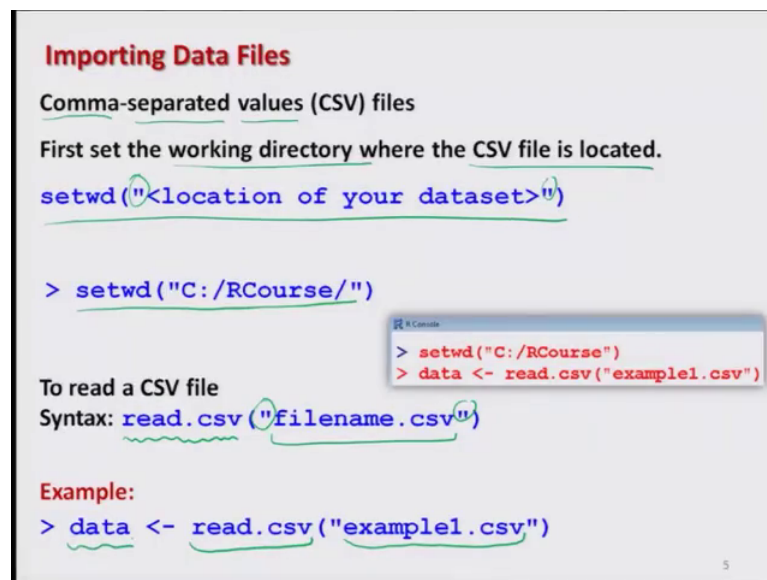
(Refer Slide Time: 13:49)

```
> datamunich <- read.table(file= "http://home.iitk.ac.in/~$
> datamunich
      nm  nmqm  wfl rooms      bj bez  wohngut wohnbest
1  741.39 10.90  68      2 1918.0  2      1      0
2  715.82 11.01  65      2 1995.0  2      1      0
3  528.25  8.38  63      3 1918.0  2      1      0
4  553.99  8.52  65      3 1983.0 16      0      0
5  698.21  6.98 100      4 1995.0 16      1      0
6  935.65 11.55  81      4 1980.0 16      0      0
7  204.85  3.72  55      2 1924.0  6      0      0
8  426.93  5.40  79      3 1924.0  6      0      0
9  446.33  8.58  52      1 1957.0  6      0      0
10 381.45  4.95  77      3 1948.0  6      0      0
11 337.26  9.64  35      1 1957.0  6      0      0
12 756.73 11.13  68      2 1985.0  6      1      0
13 945.90 10.28  92      3 1918.0 12      1      0
14 264.93 11.52  23      1 1957.5  4      1      0
15 540.58  6.84  79      2 1918.0 12      1      0
16 757.74  9.24  82      4 1966.0 10      0      0
17 538.70  7.48  72      3 1972.0 10      0      0
```

So, you can see here I have simply copied and pasted this thing.

And it is actually done now if you try to write down here datamunich the name of the file you can see here that this data is now available here right. So, you can see here this entire huge file is available over here right. So, that will give you a confidence yes this can be done.

(Refer Slide Time: 14:29)



Importing Data Files

Comma-separated values (CSV) files

First set the working directory where the CSV file is located.

```
setwd("<location of your dataset>")
```

```
> setwd("C:/RCourse/")
```

To read a CSV file

Syntax: `read.csv("filename.csv")`

Example:

```
> data <- read.csv("example1.csv")
```

R Console

```
> setwd("C:/RCourse")  
> data <- read.csv("example1.csv")
```

5

So, can now we come back to our slides, next we try to understand that how to read a comma separated values files there are different formats in which the data file can be saved and one popular source CSV file. CSV is the short form of comma separated values and in this case the values are separated by comma. So, in order to read any file first rule which you always have to follow that please set the working directory where the CSV file is located.

So, for that you have to use the command as we discussed earlier set working directory set w d and inside the arguments within the inverted commas you have to write down the location of your data set. For example, here I am doing set w d c colon front slash R course and then contain inside the inverted comma now the syntax to read a CSV file is read dot CSV and then inside the double quotes you have to give the name of the file which you want to read, right. For example, here I have created simple file say called as example one dot CSV and I try to read this CSV file over here, right and I try to store all the values inside the vector here data. So, you can see here what I get over here. So, and

you can also see here that example one dot file is here that you can see here I am highlighting it; this is in my working directory right and here if I try to do.

(Refer Slide Time: 16:45)

Importing Data Files
Comma-separated values (CSV) files

Example:

```
> data <- read.csv("example1.csv")  
> data
```

	X1	X10	X100
1	2	20	200
2	3	30	300
3	4	40	400
4	5	50	500

No information on header

```
> setwd("C:/RCourse")  
> data <- read.csv("example1.csv")  
> data
```

	X1	X10	X100
1	2	20	200
2	3	30	300
3	4	40	400
4	5	50	500

1, 10, 100 is disappearing

Notice the difference in the first rows of excel file and output

So, you see it has read and now it is trying to give it here say data like this one right now after that we try to understand what it is really trying to do and what are the different features that we need to understand. So, here I have done the same thing which you just saw on the R console this is the file which I have given a screenshot over here; it has just 5 values just for the sake of understanding 1, 2, 3, 4, 5 followed by 10, 20, 30, 40, 50 followed by 100, 200, 300, 400, 500 means intentionally I have kept these values at 1, 10, 100. So, that you can remember easily that one means first row 2 means second row and similarly 5 means fifth row.

And then unit in once they are the first column unit in tens in the second column and units in hundred in the third column. So, this is the CSV file which I am trying to read here for that I give here command; this read dot CSV and inside the argument example one dot CSV within the inverted commas and this gives you here this thing. So, this is the screenshot of whatever we did just now what you need to observe here; here is the following, right if you try to compare means I would say I remove this highlighting if you try to concentrate over this thing these are 5 values, but when it is reading here this is giving us only here 4 values; what is really this happening the first row which is here one ten and hundred one ten and hundred is disappearing.

Why this is. So, so this gives us an hint that there is something wrong because whenever we are trying to write a data file usually the first row gives the name of the variables and that is called as header. So, in this case I am not telling for example, when I try to write down this command over here read dot CSV there is no information on header means I am not telling the R software; whether this file has header or not. So, what is happening by default the R software through the command read dot CSV is taking first row which is your here 1, 10 and 100 as header and that is why it is not printing it here, right. So, now, let us try to improve it.

(Refer Slide Time: 20:10)

Importing Data Files

Comma-separated values (CSV) files

Data files have many formats and accordingly we have options for loading them.

If the data file does not have headers in the first row, then use

```
data <- read.csv("datafile.csv", header=FALSE)
```

TRUE ✓
FALSE ✓

7

So, now whenever you are trying to read a CSV file using the read dot CSV command you have to keep in mind whether there is a header or not and this part is the same command that we just used and after this we also have to write what is header h e a d e r; header is equal to what this can be true or this can be false right and; obviously, as we talked earlier, this true and false; they are my logical variables, right. So, we try to use the command header equal to true or false to inform the R software that when you are trying to use the read dot CSV file then please try to understand that first row is header or not right. So, let us now try to take the same example.

And we try to do it here with the header equal to false command. For example, I try to write down the same command here, but now I am adding here header equal to false.

(Refer Slide Time: 21:16)

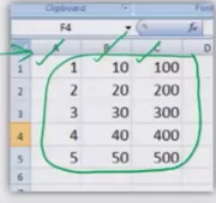
Importing Data Files
Comma-separated values (CSV) data
Example:

```
> data <- read.csv("example1.csv", header=FALSE)
```

> data

	V1	V2	V3
1	1	10	100
2	2	20	200
3	3	30	300
4	4	40	400
5	5	50	500

No header

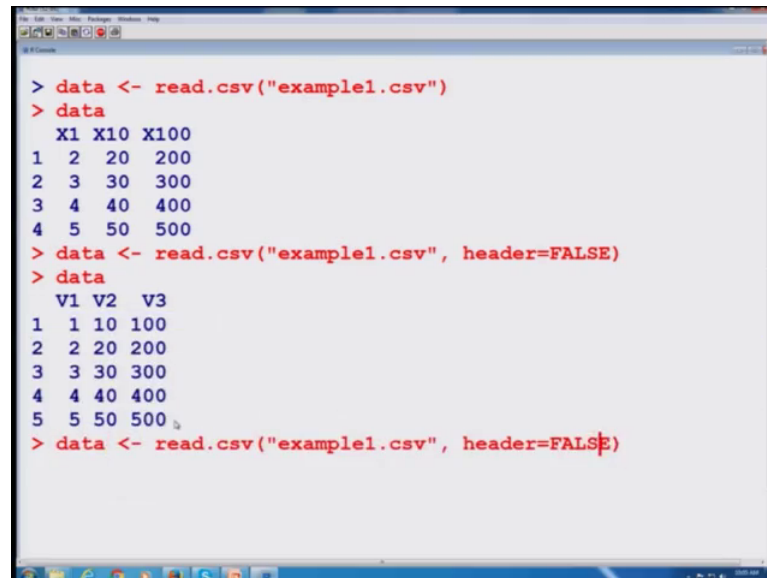


```
> data <- read.csv("example1.csv", header=FALSE)
> data
  V1 V2 V3
1  1 10 100
2  2 20 200
3  3 30 300
4  4 40 400
5  5 50 500
```

Now, you can see here there are 5 rows and 3 columns starting from 1, 2, 3, 4, 5, but in this case, we are getting all the values here. So, here you are saying that there is no header and that you can see here in this; while there is no header also had it been like that here you try to write some name here, you try to write some name here, you write here; you try to write some name, then possibly you have to give header is equal to true and then it will leave the first row the R software will not read the first row.

But it will start reading the data from the second row. So, that is the difference between using the option header is equal to true or header is equal to false now after this we try to do this analysis on the R console also.

(Refer Slide Time: 22:31)



```
> data <- read.csv("example1.csv")
> data
  X1 X10 X100
1  2  20  200
2  3  30  300
3  4  40  400
4  5  50  500
> data <- read.csv("example1.csv", header=FALSE)
> data
  V1 V2 V3
1  1 10 100
2  2 20 200
3  3 30 300
4  4 40 400
5  5 50 500
> data <- read.csv("example1.csv", header=FALSE)
```

So, you can see here that nobody notice at that time, but here the values are starting from 2, but now I am trying to take a header equal to false and then I try to take here the data this is giving me here like this all the values over here all the 5 values and similarly if I try to make it here header equal to say here true. For example, you can see here the data comes out to be changed. So, you can see here when the header is false then I have here 5 values, but when header is true; that means, the first row is trying to indicate the names of the variables.

So, I am instructing the R software to start the reading data from the second row and it is here only 4 row the first row is taken as header right now we come back to our slides now you can see here when you are trying to use this command then R is automatically giving some names here it is trying to give the names here V 1, V 2, V 3, but suppose you want to change these names well V 1, V 2, V 3 they are the default names which are given by r, but definitely you would like to give them some logical names by which you can identify the variable.

(Refer Slide Time: 24:04)

Importing Data Files

Comma-separated values (CSV) files

The resulting data frame will have columns named V1, V2, ...

We can rename the header names manually:

```
> names(data) <-c("Column1", "Column2", "Column3")
```

(Handwritten annotations: 'data' points to the object, 'Column1', 'Column2', 'Column3' are underlined, and 'V1', 'V2', 'V3' are written above with arrows pointing to the respective column names in the code.)

```
> data
```

	Column1	Column2	Column3
1	1	10	100
2	2	20	200
3	3	30	300
4	4	40	400
5	5	50	500

(Handwritten annotations: 'V1', 'V2', 'V3' are written above the column headers with arrows pointing to them. A bracket on the right side of the table indicates the data rows.)

So, in case if you want to change these names then it is also possible to do it R. So, the question is now how to rename the header because we header is the place where you are giving all the names of the columns or say all the names to the variables.

So, in order to change the name we have a format here that we use the syntax `names n a m e s` and inside the argument we write the name of the data file and then I says see here something like this here name of the first column; let us take it; suppose here column 1, then name of the second column name of the third column inside the double quotes and separated by commas, right. So, for example, here I am saying that I had here 3 names V 1, V 2 and V 3 and now I am saying that please change the name of V 1 as column one name of V 2 as column 2 and name of V 3 as column 3, right and when I do. So, you can see here the names are change here means earlier this was V 1; this was V 2 and this was V 3. So, let us try to do here in the R console also.

And you can see here that here I have got all the 5 values there is no problem of header here.

(Refer Slide Time: 25:46)

```
> data <- read.csv("example1.csv", header=FALSE)
> data
  V1 V2 V3
1  1 10 100
2  2 20 200
3  3 30 300
4  4 40 400
5  5 50 500
> names(data) <-c("Column1", "Column2", "Column3")
> data
  Column1 Column2 Column3
1         1         10    100
2         2         20    200
3         3         30    300
4         4         40    400
5         5         50    500
> |
```

So, if you try to see; earlier we had this data which has the name here V 1, V 2, V 3 which are given over here. Now I am trying to change the name of this data. So, you can see here now once I say in data these names; names are here changed; column 1, column 2, column 3. Now we come back to our slides and we see in the next slide. So, this is only the screenshot of the operation that we just did and after this we come to another issue whenever we are trying to use a comma separated file; that means, the data in a row they are separated by comma, but it is also possible that somebody has used some other option to separate the values.

(Refer Slide Time: 26:21)

Importing Data Files

Comma-separated values (CSV) files

We can set the delimiter with sep.

If it is tab delimited, use sep="\t". *1,2,3*
1,2,3
one blank space

```
data <- read.csv("datafile.csv", sep="\t")
```

If it is space-delimited, use sep=" ". *66* *22* *one blank space*

```
data <- read.csv("datafile.csv", sep=" ")
```

11

For example, one popular choice is that the values are separated by blank space for example, in case if I have suppose here 1, 2, 3; 3 values and these values can be written like this one; no space 2 no space and say a 3 or another option is I can write here one then some space blank space and then I can write down here 3 after a blank space. So, there is here blank space and this is actually one unit of blank space that you just press the spacebar once, right. So, these files can have different types of structure and we want to read that data in the R software. So, for that we have to inform the R software that what is the separator in the data file or what is the delimiter in the data file for that we used a command here or syntax here `sep`.

Which mean separator and suppose in case if the values are delimited by a tab; tab is a key on your keyboard by which you try to move forward, right. So, if the values are delimited by a tab then we use `sep = "\t"` inside the double quotes and we write here backslash t; t means tab. So, the this is trying to show here that the values are tab delimited the values are separated by tab, right. Similarly in case if the values are separated only by blank space then we simply try to write down this double quotes and then I leave here one blank space one blank space means you press the spacebar on your keyboard just once and then the syntax to read the file is the following `read.csv` then inside the double quotes you have to write down the file name.

Say for example, data file dot CSV and then you have to specify the separator what is your separator is it tab delimited or in the second case whether it is say space delimited and based on that you can read your file without any problem.

(Refer Slide Time: 29:35)

Importing Data Files

Reading Tabular Data Files

Tabular data files are text files with a simple format:

- Each line contains one record.
- Within each record, fields (items) are separated by a one-character delimiter, such as a space, tab, colon, or comma.
- Each record contains the same number of fields.

We want to read a text file that contains a table of data.

`read.table` function is used and it returns a data frame.

```
read.table("FileName")
```

Handwritten annotations on the slide include: 'Age weight' above the sample data, circled values for 'name 1 | 45 | 60', 'name 2 | 60 | 50', and 'name 3 | 75 | 70', and an arrow pointing to the space between fields labeled 'space'.

So, in case if you any such file and then there are other options also you can explore depending on your need. Now the next option is that how to read a tabular data files; that means, the data has been given in the form of a table in a very simple format. So, this is essentially a text format. So, now, we are going to answer the next question that how to read the data in a text format.

So, whenever we use the word tabular data files; this means they are simply the text files in which the data has been entered as a text and they have a very simple format the format is like this that each line contains one record and within each of the record the fields are separated by one character delimiter and this delimiter can be a space tab colon comma like this and each or record contains the same number of fields does it look very difficult to understand possibly yes take an example. Suppose, I try to write down here say 3 values; say name 1, name 2, name 3 and suppose I write their age 45 years; name to has age say 60 years and name 3 has age 75 years and suppose they are sub weight 45 years old person has suppose weight 60, 60 years old person has weight 50.

And 75 years old has weight 70 kgs; what you can see over here there I have got here 3 records record number 1, record number 2 and record number here 3; this is what I meant by this statement. Now you can see here within a record these values are separated by say see here, here, here, here, here, here, here by space. This can be comma also, I can put here comma also. So, this is what I meant by this statement that each value is separated by a

space tab colon or comma and each and every record has got only here 2 fields one is here age another is a hear weight say age and weight. So, this data set this data set and third data set; that means, the all the 3 data set have got the same number of field that is age and weight.

So, now in case, if you want to reach such TXT file, then the syntax here is read dot table and inside this read dot table, you can simply write down the name of the file inside the double quotes for example, the file can be exemplified one dot TXT, right. So, this is the screenshot of this operation. So, now, we stop here we in this lecture and I would request all of you to just have a quick look over the entire lecture and try to experiment this command on your own computer; on the next turn, we will have more discussion on importing the file from different types of software; till then goodbye.