Foundations of R Software

Prof. Shalabh
Department of Mathematics and Statistics
Indian Institute of Technology, Kanpur

Lecture - 45
Data Handling - Importing and Reading CSV and Tabular Data Files

Hello friends, welcome to the course foundations of R software. And, now from this lecture we are going to begin with a new topic. You know, that whenever you are trying to handle different types of data sets in the R software, usually the data is prepared by someone else under different types of software. Somebody is trying to prepare the data set in say Microsoft Excel, somebody is trying to combine the data in some tabular format dot txt file.

Somebody is trying to bring the data from some other type of softwares like SPSS, MATLAB etc. So, now, the question is that, when you want to handle such a data in the R software you cannot handle it directly, right. Because, if the data file is created under the software say this Microsoft Excel, then internally that has something which the R may not be able to read it. So, we need to read the data file and we need to import the data set in the R software.

And after that, whatever are the rules of the data handling we have learned or we have learned, how to handle different types of say this operation through the, through different commands they have to be implemented under the R software. So, now, in this lecture and in the next lecture, I will try to a consider a couple of some popular formats of the file, wish we would like to read them in the R software.

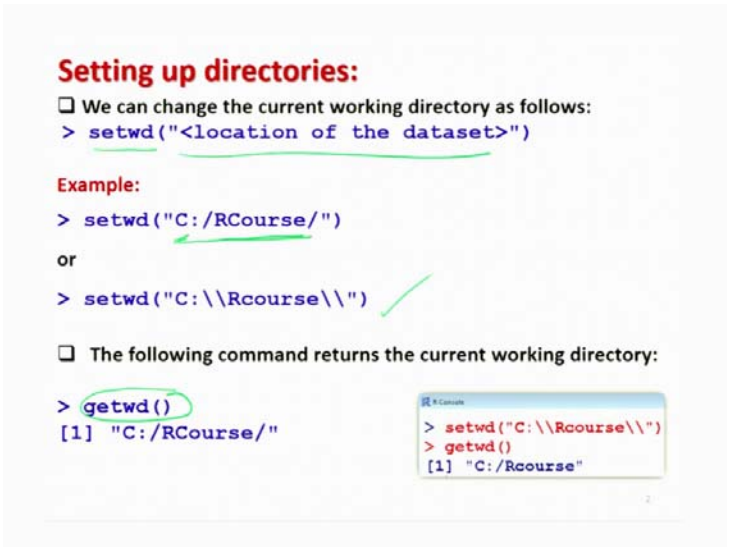
Once you can read them after that there is no issue you have learnt many many this data handling commands and you can do the suitable operation. So, in this lecture today, I am going to talk about two types of file, right. One is CSV file that is comma-separated value file and say other is tabular data file. Do you know what is this CSV file and tabular data file? CSV files are those files in which the data is written and it is separated by comma, ok.

And, then data is written on different rows or say different lines and similarly in the tabular data file, the data is separated by either by blank space or by something else, right. And, then, every line or say every row contains 1 complete set of data; that is like suppose if I try to take a very simple example of students and their marks. Suppose, I have got three subjects physics, chemistry and mathematics and I have suppose 5 students, I want to write down their marks.

So, you will see that in the first row I will try to write down the names of the student, student number 1, 2, 3, 4, 5 and after that their marks in physics, chemistry, maths for a student number 1, for a student number 2 once again marks in the physics, chemistry, maths and so on. So, every row is going to contain the same number of fields. For example, in this case there are three fields physics, chemistry and maths.

So, every student will have the marks in the in those 3 fields. So, this is your tabular data file. So, now, in this lecture, we will try to understand how we can bring the data under the dot csv format and dot txt format in the R software. So, we try to begin the lecture and we try to take here some examples. So, that I can explain you how these operations can be done, ok.

(Refer Slide Time: 04:02)



Setting up directories:

- We can change the current working directory as follows:

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

Example:

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

or

```
> setwd("C:\\Rcourse\\")
```

- The following command returns the current working directory:

```
> getwd()
[1] "C:/RCourse/"
```

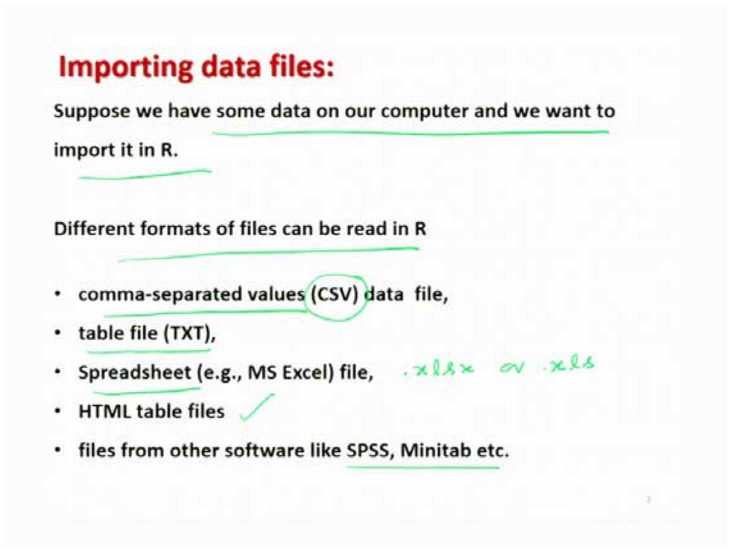
```
> setwd("C:\\Rcourse\\")
> getwd()
[1] "C:/Rcourse"
```

So, now, first of all what I have done that I have prepared some very simple examples. And, I have kept all those files in my directory which is located on the C colon in a folder whose name is R course. So, my first job is that wherever are my files located, which I want to import inside the R software that directory has to be set as the current working directory.

And we already have discussed that if you want to set the working directory the command here is setwd and then you have to give the location of the data set along with the path. So, if you try to see here, I have written here is within double quote C colon backslash R course or you can also write like this. And if you really want to know that what is your current working directory, we already have learned that we have a command here getwd that is get working directory.

So, by which you will come to know. Now, my request is that, you also try to create a working directory anywhere wherever you want I have created it on the C drive, similarly you can create it on any drive, right. So, that you can also create this type of file and you can keep all the files and their results in the same folder. So, I will be using this folder in this course, right.

(Refer Slide Time: 05:11)



Importing data files:

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

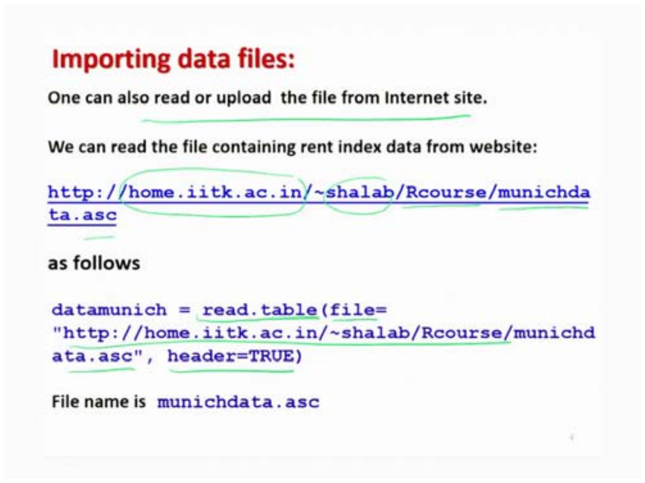
Different formats of files can be read in R

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

So, now our job is that suppose we have got some data on our computer and we want to import it in the R software. So, as we have discussed that there are different formats of the file that in which the data can be saved and those formats can be used in the R software. For example, comma separated value which are popularly known as CSV data files and tabular data file like table files or say dot txt or a spreadsheet like a for example, the files which are created in the MS Excel software and their extension is like dot xlx or dot xls and so on.

Similarly, you have HTML files the data is given in the HTML format and then there can be some other files which are imported from other softwares like as SPSS, Minitab etc. these are some statistical software. So, we try to handle the data under the software. So, they also have a storage facility and you can read the data over there and store the data over there, but in their own format, right.

(Refer Slide Time: 06:09)



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/munichd
ata.asc", header=TRUE)
```

File name is `munichdata.asc`

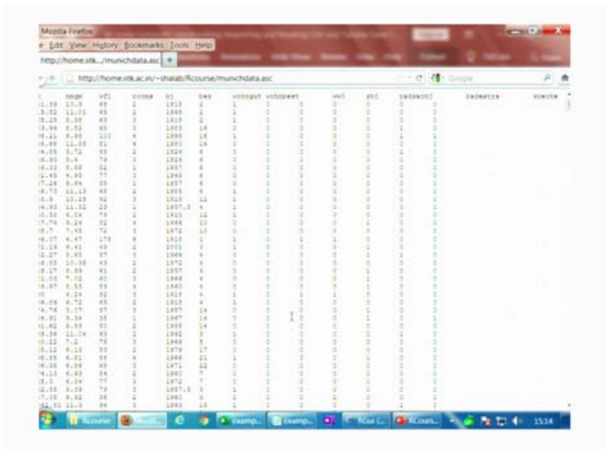
Now, before going forward, means I just want to show you that when you are trying to read the data, it is not a condition that data has to be only on your hard disk or your computer. The data can also be read from any internet site, right. For example, what I have done I have uploaded file on my home page. The address of my home page is home dot iitk dot ac dot in backslash tilde s h a l a b. And, after that I have created directory here there also R course and under which I have file munichdata dot asc.

So, actually this data is consisting some data from the Munich city in Germany about their house rent. But, anyway, my objective is that, I just want to show you if somebody has uploaded the data on the website how you can read it inside the R software directly. And after that, you can bring that file inside your R software and after that you can make such manipulations.

So, how to get it done? Well, I have not just explained you that how you can read such a data file, but I can tell you here and I will then explain you in this a lecture that, in order to read this tabular data we have a command here read dot table and after that you have to give here the file name along with the address. So, this file name is here, this address and then after that I am trying to give here one option header is equal to TRUE.

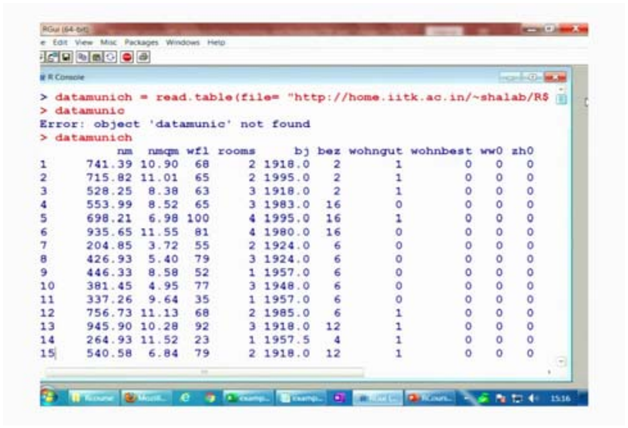
Well, what is the meaning of this read dot table header etc.? That we are going to discuss very soon, right. But, if you try to see here, this command can be used here and then this data file can be read directly in the R software, right.

(Refer Slide Time: 07:48)



So, first what I will do here that I will try to show you this location on the R internet site. So, I try to you can see here, I try to write this data and you can see here this is here the data, right, that is a long file. But I can show you here; that means, I can increase the font size here like this and if you want to see the entire file it is here like this, right, anyway. So, you can also see this data over there anyway. My objective was not to handle the data, but to show you that this data is here on this on my home page, right.

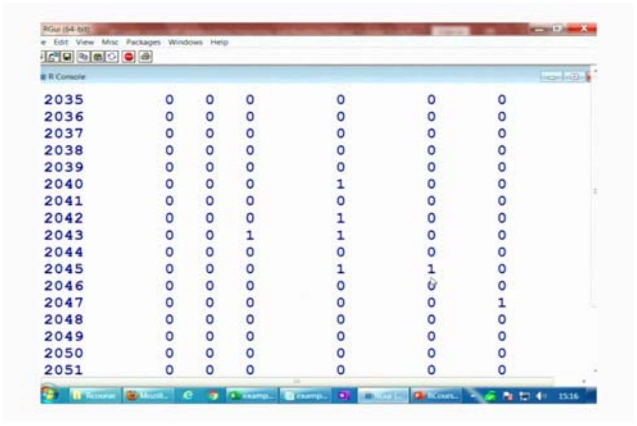
(Refer Slide Time: 08:22)



```
> datamunich = read.table(file= "http://home.iitk.ac.in/~shalab/R$
> datamunic
Error: object 'datamunic' not found
> datamunich
         nm  nmqm wfl rooms     bj bez wohngut wohnbest ww0 zh0
1    741.39 10.90  68     2 1918.0   2       1        0   0   0
2    715.82 11.01  65     2 1995.0   2       1        0   0   0
3    528.25  8.38  63     3 1918.0   2       1        0   0   0
4    553.99  8.52  65     3 1983.0  16       0        0   0   0
5    698.21  6.98 100     4 1995.0  16       1        0   0   0
6    935.65 11.55  81     4 1980.0  16       0        0   0   0
7    204.85  3.72  55     2 1924.0   6       0        0   0   0
8    426.93  5.40  79     3 1924.0   6       0        0   0   0
9    446.33  8.58  52     1 1957.0   6       0        0   0   0
10   381.45  4.95  77     3 1948.0   6       0        0   0   0
11   337.26  9.64  35     1 1957.0   6       0        0   0   0
12   756.73 11.13  68     2 1985.0   6       1        0   0   0
13   945.90 10.28  92     3 1918.0  12       1        0   0   0
14   264.93 11.52  23     1 1957.5   4       1        0   0   0
15   540.58  6.84  79     2 1918.0  12       1        0   0   0
```

And, now, what I try to do here, that I try to bring this data file in my R software and I try to give it here a name data munich. And, then I come to this R software and if you try copy and paste this command here, you can see here this data here is like data munich like this one, right and you can see here this data is here, right.

(Refer Slide Time: 08:43)

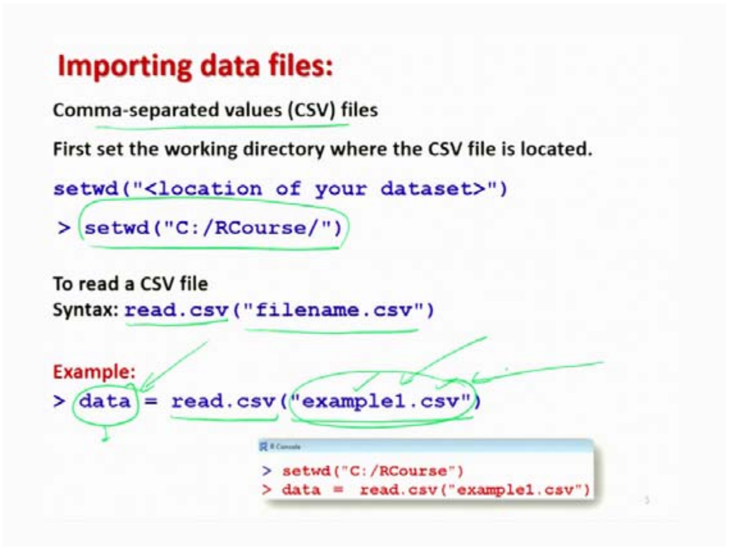


```
2035        0   0   0       0       0     0
2036        0   0   0       0       0     0
2037        0   0   0       0       0     0
2038        0   0   0       0       0     0
2039        0   0   0       0       0     0
2040        0   0   0       1       0     0
2041        0   0   0       0       0     0
2042        0   0   0       1       0     0
2043        0   0   1       1       0     0
2044        0   0   0       0       0     0
2045        0   0   0       1       1     0
2046        0   0   0       0       0     0
2047        0   0   0       0       0     1
2048        0   0   0       0       0     0
2049        0   0   0       0       0     0
2050        0   0   0       0       0     0
2051        0   0   0       0       0     0
```

Anyway, I am not bothered about this data. But my thing is this, ok I just wanted to show you that this data is on the internet site, but I can bring it here directly on my R console.

And now, this data is here just like here data munich and then I can handle this data for example, if you want to do anything over this data you can just do it, right. So, this is here the data you can see anyway. So, I clear the screen.

(Refer Slide Time: 09:03)



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")`

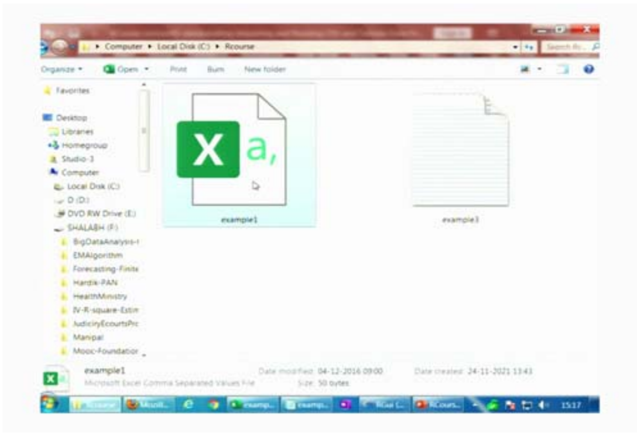
Example:

```
> data = read.csv("example1.csv")
```

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

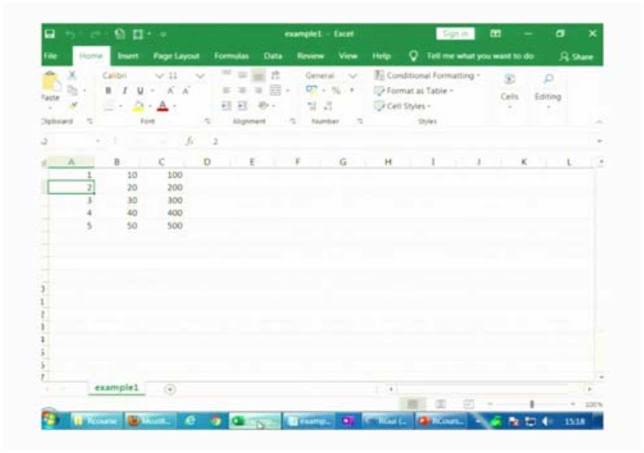
And, so, I have told you that in R software it is possible to read the data from your local directory, from your computer, as well as from some internet site, right. So, now, this has a very big advantage, that if there are more than 1 people who are staying at different locations and the data can be then shared among them and they can access this data remotely from their own places, right, ok.

(Refer Slide Time: 09:31)



So, after this I try to take here one example, right, ok. Before going further, let me try to show you here that you see this is my here this folder R course, you see here this name R course. And, then, I have here 2 files; one is example 1, example 3, I will try to use them in the lecture today. And, if you try to see this is my here example 1 and if you try to open it this comes here like this.

(Refer Slide Time: 09:45)



A	B	C
1	10	100
2	20	200
3	30	300
4	40	400
5	50	500

So, what I have done here, that there are only here these 5 rows and 3 columns in which, yeah, just for the sake of easy understanding, I have taken 5 values here 1, 2, 3, 4, 5 in the first column 10, 20, 30, 40, 50 in the second column, 100, 200, 300, 400, 500 in the third column.

So, the number of digits are indicating the location of the column also. First column has only 1 digit, second column has 2 digits and third column has 3 digits. And, similarly, the rows are here 1, 10, 100; then 2, 10, 100; 3, 30, 300; 4, 40, 400 and 5, 50, 500. So, these numbers are also indicating the row numbers like as if I say 2 so; that means, this is the second row and first column. So, that is just for the sake of easy understanding. So, that you can compare what you are trying to do and what are you getting here.

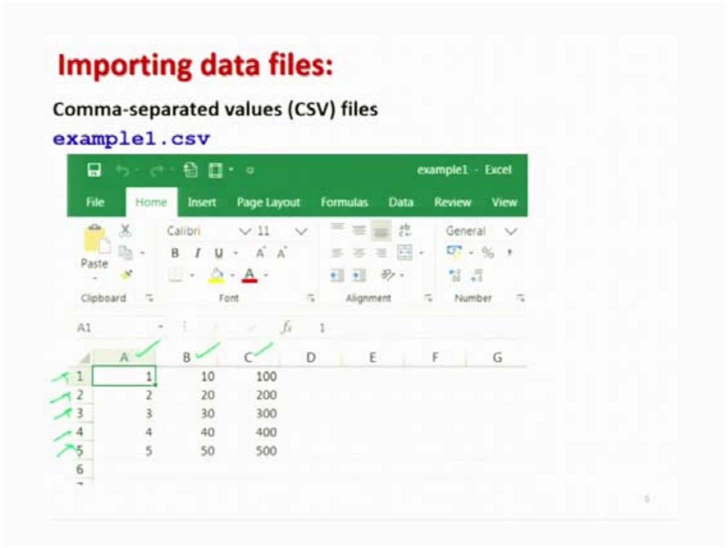
So, I want to read this file and this file is actually comma separated value file; that means, the file has been created in such a way such that the values are separated by comma. So, now, my question is how you can read the CSV files? How you can read the comma separated values files? That is all. So, for that we have a command here read dot csv and then within the parenthesis we have to give the name of the file as file name dot csv and after that there are a couple of say this options which are used depending on the nature of the file, right.

So, what I have done, because this file is located in my directory. So, I already have set my working directory and yeah to this thing so, I will cut it on my R console also. And after that, I will try to read this file using the command read dot csv and within the parenthesis, I try to give here the name example 1 dot csv and I try to store it inside a variable name d a t a data.

Now, one question comes here. When there is a file name here already, when there is already a file name like example 1 dot csv, then why are you trying to give it here say a new file name here as say data? You see, one thing you have to understand this example 1 dot csv is the file which is a comma-separated value file and that has some internal structure and that can be opened in the suitable software. But I want to bring the data values and other related information inside the R software.

So, when I am trying to import these values, the structure or the internal structure of the file is going to change. And, now, this file is capable to be read by the R software. So, this the structure of the file which is capable to be read inside the R software that is given by data, right. So, now, once I read this file, then after this, I will not be bothered about the file example 1 dot csv, but I will be working only with the variable name here data, right.

(Refer Slide Time: 13:21)



Importing data files:

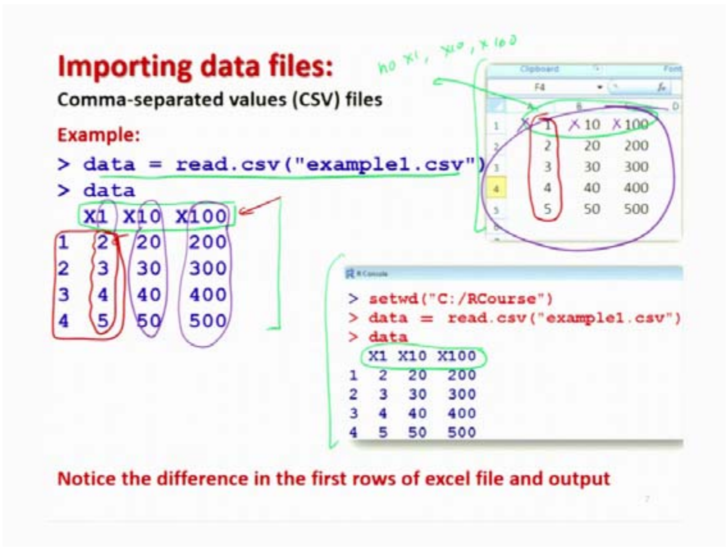
Comma-separated values (CSV) files

`example1.csv`

	A	B	C	D	E	F	G
1	1	10	100				
2	2	20	200				
3	3	30	300				
4	4	40	400				
5	5	50	500				
6							

So, if you try to see here just for the sake of your review, the example 1 dot csv file which I just shown you, it looks here like this, right. So, I have just explained you how it looks like. So, you can see here that the column names here given by are here A, B, C and row names are given by here 1, 2, 3, 4, 5, ok.

(Refer Slide Time: 12:47)



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
```

	A	B	C
1	1	10	100
2	2	20	200
3	3	30	300
4	4	40	400
5	5	50	500

```
> 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
```

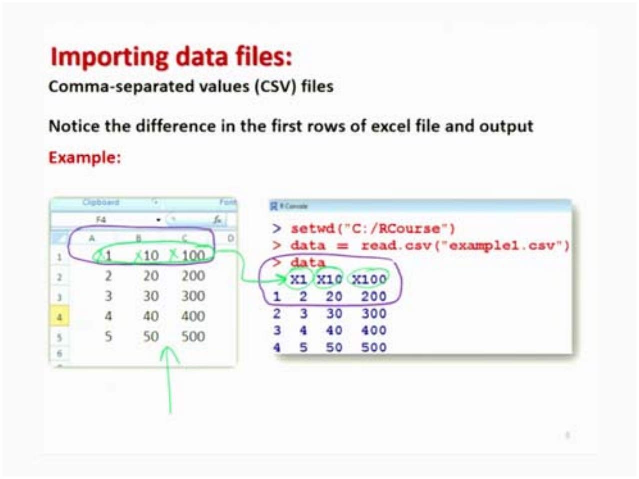
Notice the difference in the first rows of excel file and output

Now, what we try to do here, that we try to read this file in the R software. So, first I try to show you actually through this screenshot, what is going to happen and then I will try to explain you inside the R software. So, once you try to read this file and you name it under the data it looks like this. And this is here the screen shot. And this is here the example 1 dot csv file.

Now, can you compare it what is really happening? Can you see some changes in the structure? Ok. If you try to see here, what is this here? X 1 X 10 X 100 and you can see here in the screen shot also it is here and there was no X 1 X 10 X 100 here no X 1 X 10 X 100 here. What is happening here? Number 2, if you try to see here the means another difference; in this original file your data values are starting from 1, 2, 3, 4, 5. But, in this file the values are starting from here 2 3 4 and 5.

So, the first question comes here, how this X 1 X 10 X 100 entered automatically and you have not given any such command, right. And, number 2, why this data is beginning from here 2 instead of here 1. Now, before I move forward, let me try to have your attention here. Can you see here what is here 1? What is here 10? And, what is here this 100? Do not you think that something is happening that when this structure is imported in the R software, then in the first row this X 1 X 10 X 100 is entered? And this happened automatically.

(Refer Slide Time: 14:26)



Importing data files:

Comma-separated values (CSV) files

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

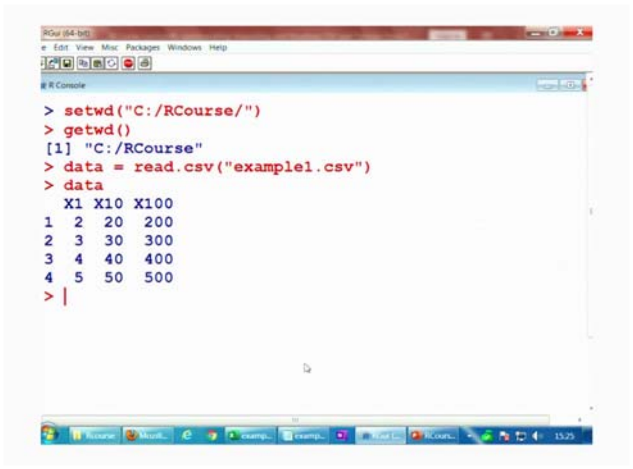
Example:

	A	B	C	D
1	1	10	100	
2	2	20	200	
3	3	30	300	
4	4	40	400	
5	5	50	500	
6				

```
> 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
```

So, now we try to understand, why this is happening? And you have not done it, but still this is happening, right. If you try to see here more clearly here, that original data has like this column names A, B, C and data is starting from 1 to 100, sorry, like here 1, 10, 100 in the columns. And, but, when you are trying to read it inside the R software it is becoming like this. So, let us first try to understand this thing and try to see whether this is happening in the R software or not? So, first I need to set my working directory, right.

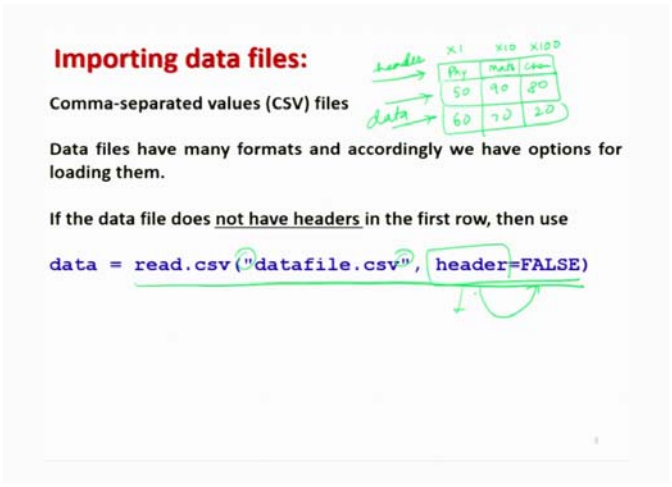
(Refer Slide Time: 14:57)



```
> setwd("C:/RCourse/")
> getwd()
[1] "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
> |
```

So, I click here. So, now, you can see here, my working directory here is like this C colon backlash R course. But, yeah, that is not a very big deal for you also to do such a thing. And, then, I try to read here this example 1 and I try to store it under the data, right. And, now, if you try to see what is happening here with the data is giving you here X 1 X 10 X 100. So, the question is what is this happening? And, then the data values are starting from 2 20 200 rather than 1 10 100, right.

(Refer Slide Time: 15:30)



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 <u>not have headers</u> in the first row, then use

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

header
data
X1 X10 X100
Phy Math Chem
50 90 80
60 70 20

So, now let us try to find out the answers one by one. You know, that when you are trying to handle the data files there is something like called as header; means header is something like, I try to write down the data values here and I try to write down here the see here subjects say physics, maths and say chemistry. And here I try to write down the value there 50, 60, 90, 70, 80, 20.

So, now, when you are trying to give the data values, these two rows they are your data values. This is only a header means this is only the name this is not the data. But it is a header which has to be read only as a name and this does not contain the data. So, what is happening in this example here, that whatever is your here first row 1, 10, 100; the R software is trying to read the first row as header and it is doing it automatically.

And then it is trying to give these headers a name like as X 1 X 10 and X 100. So, essentially what it is trying to do it is trying to fix the alphabet here X X 10 and X 100 like this. So, now, it is something like this now physics becomes here something like X 1 maths becomes X 10 and chemistry becomes here X 100 like this. So, you have not given it, but on the other hand you also have not informed the R software that there is no header. So, that is why the default here is that R is trying to assume that the data file has got a header and that is why the first row is being considered as header.

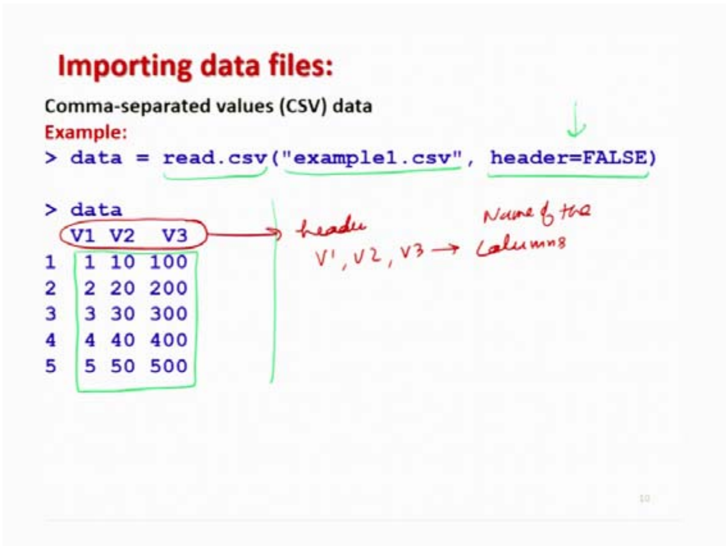
Now, in case if header is not there, then we need to have one option which can inform the R software that in this file there is no header and similarly this will give you a control also, that if you are trying to read a file and if there is no header you should inform the R software that there is no header. And, if there is a header then you must inform the R software that that there is no header, right.

So, the absence and presence of header in the file has to be notified to the R software. And, for that, what we try to do here, we try to add here one option header h e a d e r which is all in lower case alphabets and this is a logical value. So, it will try to take the options in logical TRUE and logical FALSE. And so, as soon as you try to say here header is equal to FALSE; that means, there is no header and so the data in the R software will be read from the first row.

And, if you see here header is equal to TRUE; that means, the header is there and so the data will not be read from the first row. So, now, you can see here, now gradually we can increase here many many options which are required, but I will try to make this explanation simple. So, I will be considering here only couple of important options in the whole lecture.

So, now my this new command becomes here read dot csv and inside the double quotes you have to give the file name and then after that you have to specify whether header is present or absent by saying this logical FALSE or TRUE.

(Refer Slide Time: 18:45)



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
```

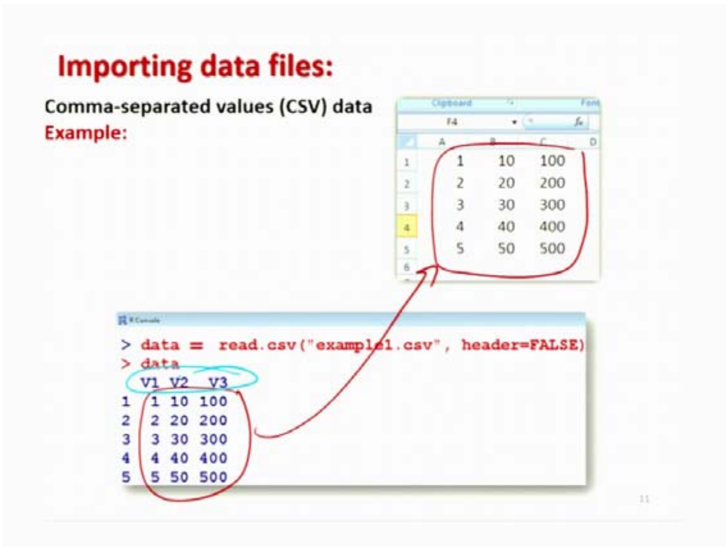
header
V1, V2, V3 → Name of the Columns

So, now, in this case if you try to see what the data which you have taken here, do you think that you have specified the header? Answer is No. So, that is why, I would like to add here that header is equal to FALSE. So, my command becomes here read dot csv and then within the double quotes I try to give here this double quote and the file name. So, the file name is the same example 1 dot csv, but now I have added here header is equal to FALSE.

So, now, you can see here this type of outcome. Now, if you try to see here whatever was your data that is now intact. And, R software has inserted a row here, what is this row? V 1 V 2 V 3. Actually, R has inserted the header and it has given the names of the columns automatically as V 1 V 2 and V 3. So, V 1, V 2, V 3 they are the columns or the name of the column, right, so which has been given by the R software automatically.

So, now when R is trying to give these names automatically, then there is a possibility that you may like to give a name which makes sense or which is needed according to your data, right.

(Refer Slide Time: 19:58)



Importing data files:

Comma-separated values (CSV) data

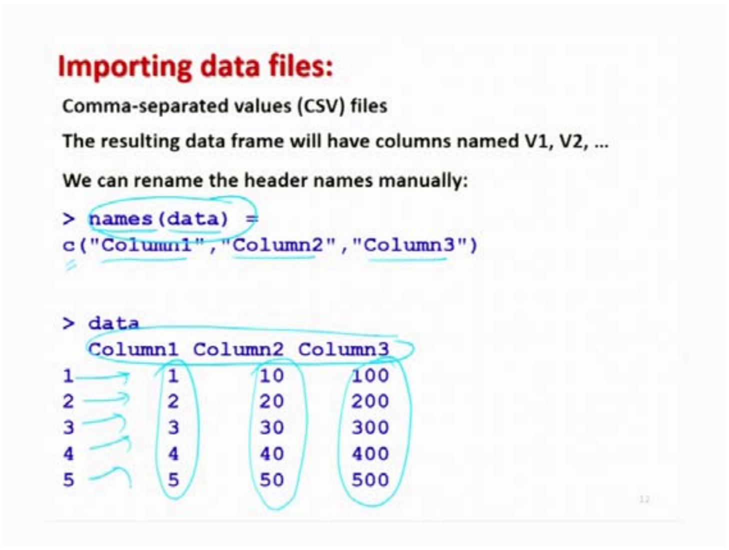
Example:

	A	B	C
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)
> 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 now in this screenshot, this was your data. And, now this is your here this data which is matching. Now, the data is read correctly, but now there is a header also.

(Refer Slide Time: 20:11)



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:

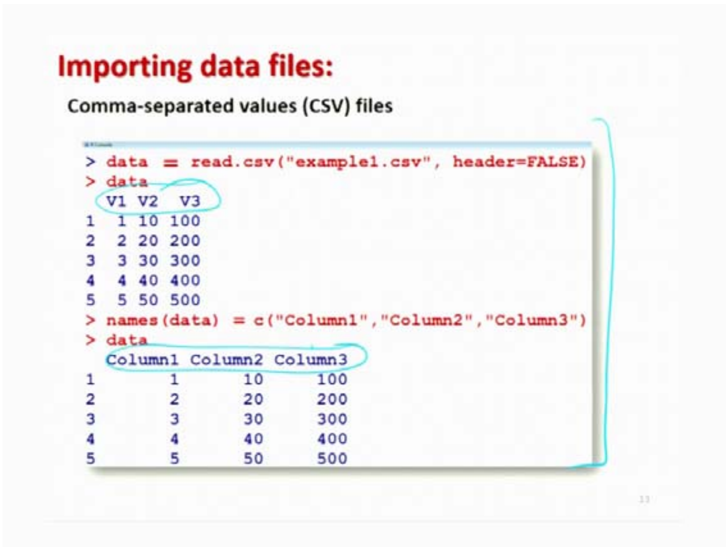
```
> names(data) =
c("Column1","Column2","Column3")

> 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, now, in case if you want to change the names of your columns. So, you already have learned this command which is here name n a m e s and inside the parenthesis you try to give the name of this data which is d a t a and then you try to give the names of the columns that you want to give, suppose, I want to give Column 1, Column 2 and Column 3.

So, I try to create a data vector of these three values and all these names are going to be written inside the double quotes because they are the characters. And, if you try to now use this command here names and inside the parenthesis data, then the names of these columns are going to be changed and earlier you had the names here like as V 1, V 2, V 3. Now, they will become here Column 1, Column 2, Column 3.

And, you can see here this is your first column consisting of the values 1 2 3 4 5; second column consisting of the values 10 20 30 40 50 and the third column consisting of the values 100 200 300 400 500. And similar are your rows. So, exactly the same data has been obtained here.

(Refer Slide Time: 21:15)



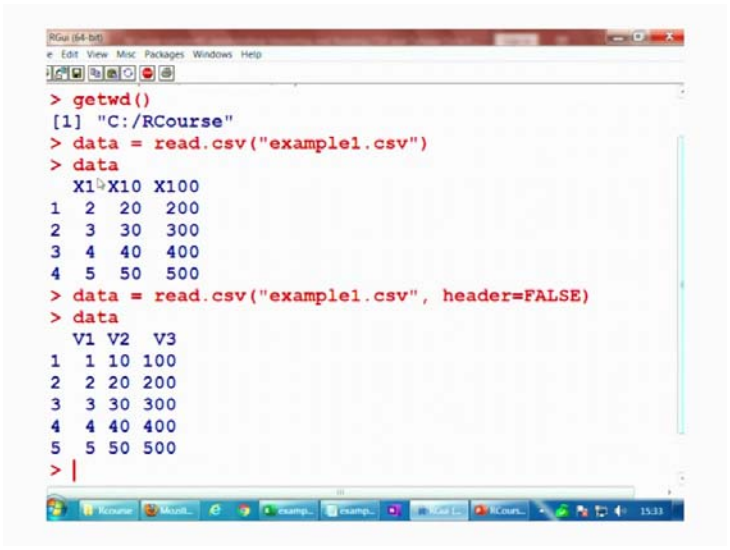
Importing data files:

Comma-separated values (CSV) files

```
> 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
```

And you can see here this is the screenshot of the same operation, that here are the names V 1 V 2 V 3. But now the changed names are Column 1 Column 2 Column 3. So, now, let me try to show you this operation on the R console also, right.

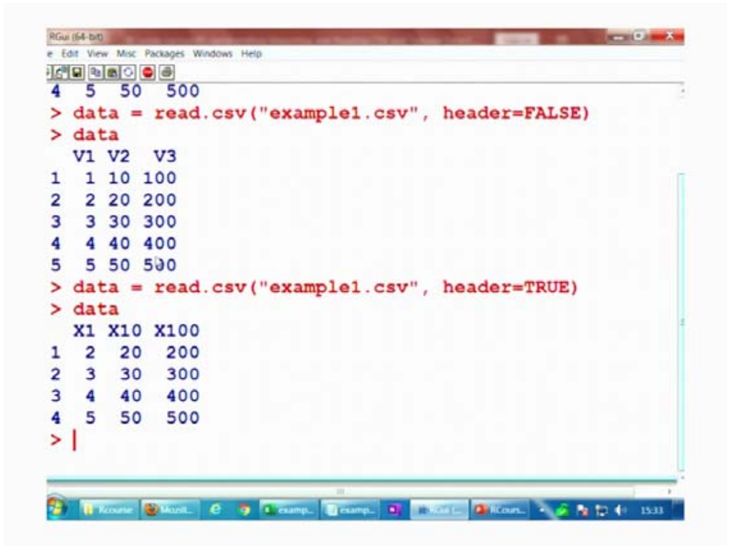
(Refer Slide Time: 21:31)



```
> getwd()
[1] "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
> 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
> 
```

So, now you can see here this was your here data with example 1 without any option. So, if you try to give it now your header is equal to here FALSE; that means, there is no header, then you can see here the data here is like this, right.

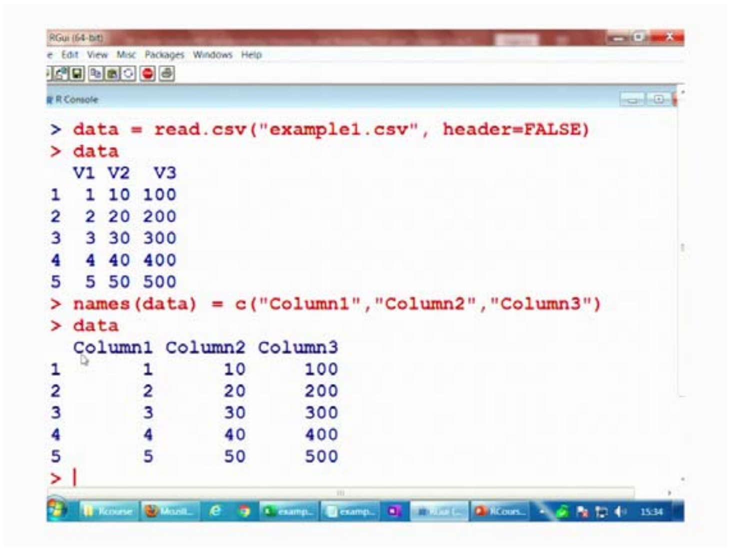
(Refer Slide Time: 21:44)



```
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=TRUE)
> data
  X1 X10 X100
1  2  20  200
2  3  30  300
3  4  40  400
4  5  50  500
> 
```

Suppose, if you try to give here header is equal to here TRUE; then now you can see what is really happening, this is the default and that was happening when you were trying to not use this option header then it was automatically taking header is equal to here TRUE, right. So, that is why there was a name like X 1 X 10 X 100 and your data values were starting from X 2 20 200.

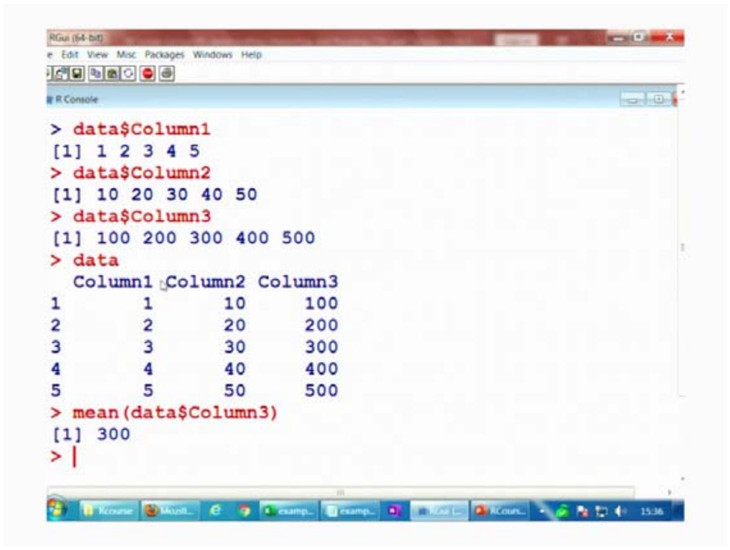
(Refer Slide Time: 22:05)



```
> 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
> |
```

But anyway, let me try to have here this here FALSE. So, that the data here is like this with the name V 1 V 2 V 3. And, now, I would like to change the name. So, I try to use here the option here say names of the data vector here like this. And, once I try to operate it here, you can see now data becomes here this Column 1 Column 2 Column 3. So, this is how you can read these variables here.

(Refer Slide Time: 22:29)



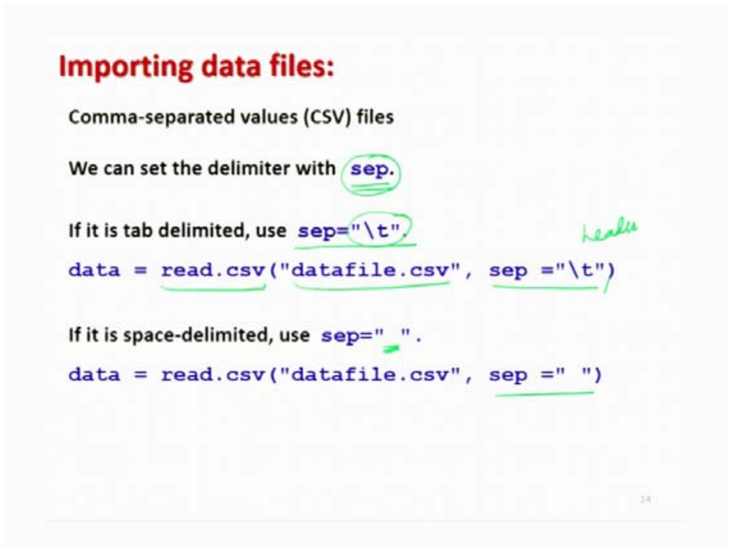
```
> data$Column1
[1] 1 2 3 4 5
> data$Column2
[1] 10 20 30 40 50
> data$Column3
[1] 100 200 300 400 500
> data
  Column1 Column2 Column3
1       1      10     100
2       2      20     200
3       3      30     300
4       4      40     400
5       5      50     500
> mean(data$Column3)
[1] 300
> |
```

And, now if you try to see here if you try to write down here like this, what do you get here? Data dollar Column 1 is your 1 2 3 4 5; data dollar Column 2 is your head 10 20 30 40 50; and data Column 3 is your here 100 200 300 400 500. So that means, what are you getting? You already have learned that if you want to read a particular column of a data set then the rule here is name of that variable data, then dollar and then that name of the variables. So, now, you can see here the name of your variables in the data are Column 1 Column 2 Column 3.

Now, you can even access the individual value that after that now you can do whatever operations you want. For example, if you want to find out here the mean of the values in the column number 3 you can simply write down here like this. So, if you try to see what you have done, the data was in CSV format. The data was an external drive that was outside the R software. Now, you have brought that data object into the R software. Now, this file can be read under the R software.

All the values are being accessed by the R software and after that you can do all sorts of data manipulation by using the command that you have learned and you can handled it very easily. So, that was my objective to show you here in this lecture.

(Refer Slide Time: 23:48)



Importing data files:

Comma-separated values (CSV) files

We can set the delimiter with sep.

If it is tab delimited, use sep="\t".

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

If it is space-delimited, use sep=" ".

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

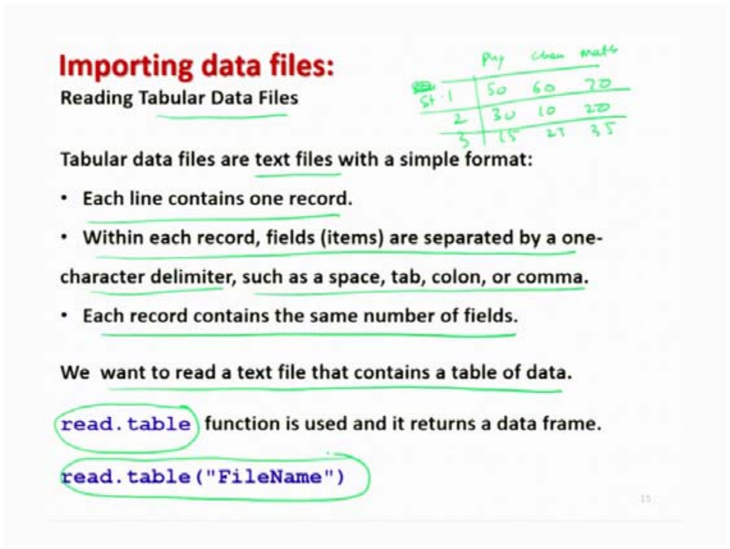
Now, in the first case, I have given you all the things in quite detail. Now, after this you will see the similar type of operations will be there when I try to consider different types of files, right. So, now, in this comma-separated value file, there is one option here the option here is that instead of using the comma you can also use another separator and this is controlled by the option here sep, right.

For example, you also have heard about the tab delimited files in which the data is separated by the tab. So, in that case you have to give here the option here s e p that is separator is equal to within double quotes backlash t, right. For example, I have taken here only the CSV file, but if the values in the files are separated by the tab, then you have to use the same command read dot csv. And then you have to give here the name of the files inside the double quotes and yeah, do not forget to write the complete name, right.

For example, data file name dot csv and then you have to use here the option here sep s e p and then if this is a tab del limited file you have to give it here back slash t. And, similarly, means there are many other options, for example, if you try to see another option is the blank space the different data values are separated by the blank space in that case you simply try to use sep is equal to blank space.

And, you try to add here sep is equal to blank space, yeah, along with this if you want to use the option here header is equal to TRUE or FALSE that also you can use and gradually you can increase this complexity, right, ok.

(Refer Slide Time: 25:13)



Importing data files:

Reading Tabular Data Files

	Phy	Chem	Math
St 1	50	60	70
2	30	10	20
3	15	27	35

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")`

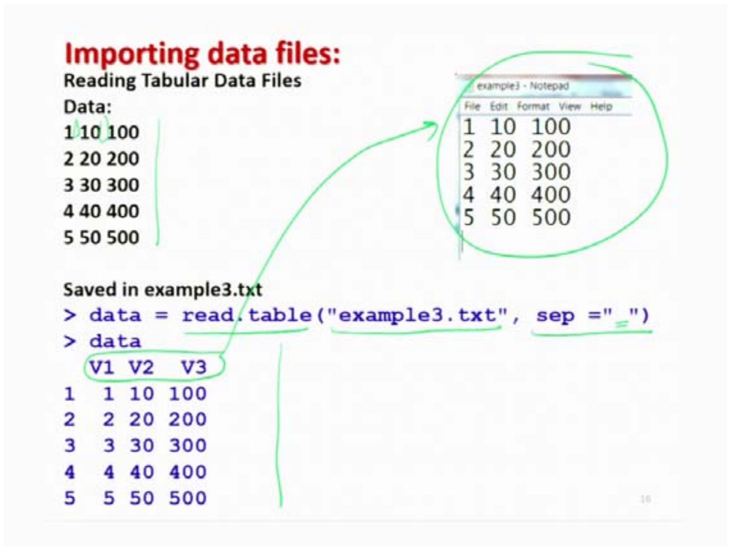
15

So, now after the CSV file, let us try to understand how we can read the tabular data file? And, how we can import the tabular data file inside the R software, right. This tabular data files are simply the text file, right. For example, they can be created in the notepad, they can be opened in the notepad and they have a very simple format. The each line contains 1 record and within each record the field separated by a 1 character delimiter such as space, tab, colon or comma that is all.

For example, if I have to write down the marks of my student, I can write down here like, suppose I will write down here 50, 60, 70, then 30, then 10 and 20, then 15, then 25, 35. So, I know that for example, this would not be there, but I will write down here physics, this is chemistry, this is here maths and then here student 1, student 2, student 3 like this. So, every here row, that is trying to indicate a value and every value has got the same fields. So, each record contains the same number of fields.

For example, every student has got the values in three different field that is physics, chemistry and maths. And, now we want to read such a text file that contains a table of data. For that the command here is read dot table r e a d dot t a b l e. And, after, that whatever you have learned in the case of read dot csv everything will be valid here. So, you will simply write down here read dot table and within double quotes inside the parenthesis you have to write down the file name, right.

(Refer Slide Time: 26:44)



Importing data files:

Reading Tabular Data Files

Data:

```
1 10 100
2 20 200
3 30 300
4 40 400
5 50 500
```

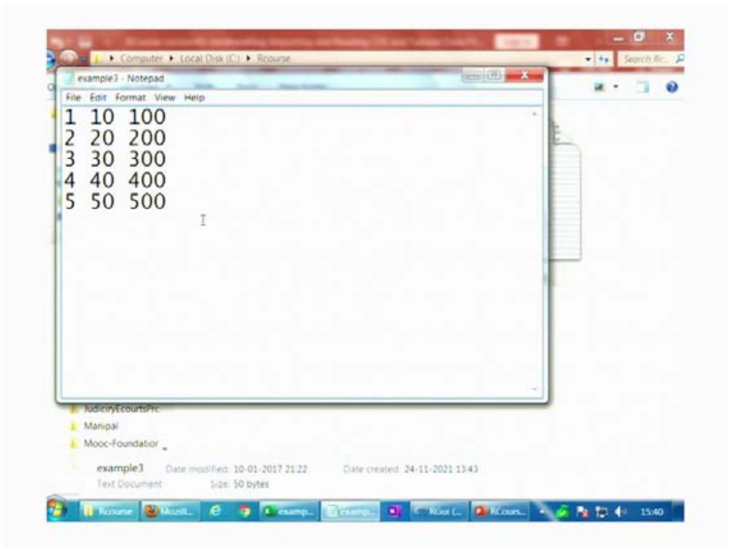
Saved in example3.txt

```
> data = read.table("example3.txt", sep =" ")
> data
  V1 V2  V3
1  1 10 100
2  2 20 200
3  3 30 300
4  4 40 400
5  5 50 500
```

```
1  10  100
2  20  200
3  30  300
4  40  400
5  50  500
```

So, for example, I can show you here, that I have created here a small file to show you in the same folder here, whose name is example 3. Yeah, you can ask that why I have not taken here example 2? Actually, example 2 is something else I will try to show you. But, somehow at the time of presentation I thought that, ok, this is a better option to first give you about this thing.

(Refer Slide Time: 27:09)



```
1 10 100
2 20 200
3 30 300
4 40 400
5 50 500
```

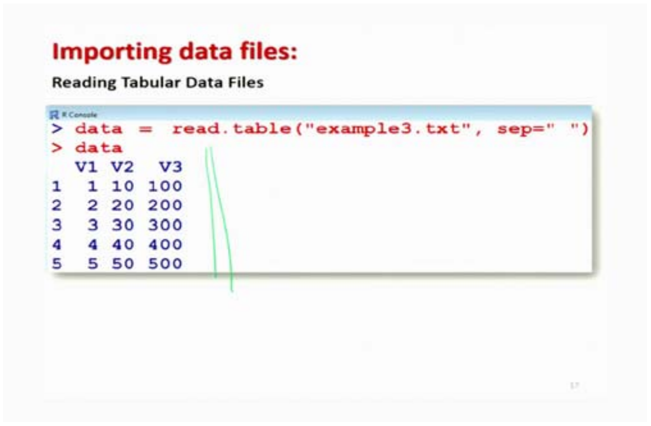
So, this is your here example 3 and if you open it this will be here this file. So, it is opened in the note pad and you can see here these are the values. So, if you try to see just to make the illustration and explanation simple. Once again, I have taken the same values. And, the first column 1, 2, 3, 4 and 5; second column 10, 20, 30, 40, 50 and in the third column 100, 200, 300, 400, 500 just like what I did in the case of this example 1. That is just to make the life simple and easy to understand. And, if you try to see here, the values here are separated only by a blank space single blank space, right.

That can be comma also, that can be some other character also, right. So, if you try to now see here, I have given you now the screen shot also. So, that you can understand it very easily. So, this is the same thing which I just shown you, right. So, now, and this is your here data and this is separated by here this blank spaces. Now, I try to read this data from this example 3 dot txt. So, this is your here read dot table I will use the command.

And, then I will try to use here within double quotes example 3 dot txt. So, you have to give the complete name. And, after that, because these values are separated by a blank space. So, I am trying to write down here sep is equal to within double quotes a blank space. And once you try to read it here, it will give here this type of outcome. Now, you will understand what is really happening. What is this here? V 1 V 2 V 3 this is not here if you try to see, right. So, what it is trying to do?

It is trying to automatically create a header, whose names are V 1 V 2 V 3 given to the three Column 1 Column 2 Column 3, respectively.

(Refer Slide Time: 28:41)



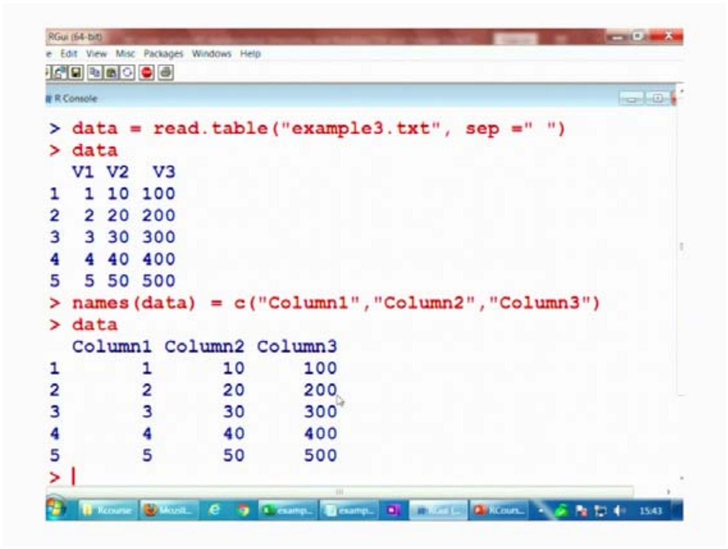
Importing data files:

Reading Tabular Data Files

```
> data  =  read.table("example3.txt",  sep=" ")
> data
  V1  V2   V3
1   1  10  100
2   2  20  200
3   3  30  300
4   4  40  400
5   5  50  500
```

And you can see here, this is here the screen shot. And, after that, the in the column names here that also you can do.

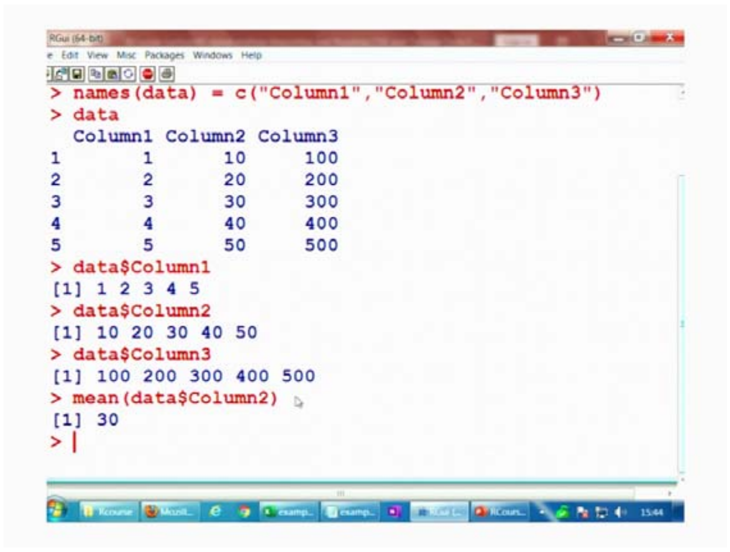
(Refer Slide Time: 28:54)



```
> data = read.table("example3.txt", sep =" ")
> 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, let me try to show you these operations in the R software also. And, so, you can see here. Now, the data here is like this, right. And, if you want to change here the names here that also you can do. So, now, if you try to see here, this will become here like this Column 1 Column 2 Column 3 exactly the same way as you have done in the case of this earlier dot csv file. And, similarly, if you now try to write down here say data say dollar here say column 1.

(Refer Slide Time: 29:15)



```
> 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
> data$Column1
[1] 1 2 3 4 5
> data$Column2
[1] 10 20 30 40 50
> data$Column3
[1] 100 200 300 400 500
> mean(data$Column2)
[1] 30
> |
```

So, you can see here, this will give you here this value. So, 1 2 3 4 5. Then, similarly, if you try to take here Column 2, this will give you here the values in the second column. And, similarly, if you try to take here the data dollar Column 3, this is here the values in the third file. And, then similarly, if you try to suppose want to find out the mean of the values in the Column 2, you can simply find out here mean and then the name of the variable data dollar Column 2 and this comes out to be here 30.

So, you can see here the way you are going to handle this file this is going to have a similar structure and commands are also the same and most of the commands you already have done in the earlier lectures and the rule is the same. So, now, it is up to you that how do you want to use it, right. So, now, we come to an end to lecture and you can see here that was a pretty interesting lecture at least to me, that you have learnt that how to read two different structures of file and how to call them in the inside the R software.

And, the best part is that, now you have learned many things. And, once I tell you that how you can read the file; after that you can handle all the commands very easily, right. So, as you can see that, as we are moving further towards the end of this course, your responsibility is increasing day by day. And, now it is up to you how much you want to think, how much you want to execute and, how much you want to experiment with the R software.

So, now this is your turn. Now, you know how to read a CSV file, how to read a txt file why do not you take up some file and try to read it and then do not stop there. But, try to do some operation which you are doing on the CSV or the txt file, which you were doing before learning the R software. So, we saved as dot txt file or dot csv file. So, you can get the CSV or txt file directly from your Excel file also.

Well, in the next lecture, I will try to show you how you are going to read the Excel file also. After, that there is no problem you can read any file you can handle, any data set in the R software. So, you try to practice it and I will see you in the next lecture, till then goodbye.