How can I read specific rows from a CSV file into R?

Authored by

stats writer

June 25, 2024

Reading specific rows from a CSV (Comma Separated Values) file into R can be achieved by using the "read.csv" function. This function allows the user to specify the file path and additional parameters such as row indices or column names to be read. By providing the desired row indices, the user can extract only the specific rows they need from the CSV file into R. This feature is particularly useful when dealing with large datasets, as it allows for efficient data manipulation and analysis. Overall, using the "read.csv" function is a simple yet effective way to read specific rows from a CSV file into R.

Read Specific Rows from CSV File into R

You can use the following methods to read specific rows from a CSV file into R:

Method 1: Import CSV File Starting from Specific Row

df <- read.csv("my_data.csv", skip=2)

This particular example will skip the first two rows in the CSV file and import all other rows in the file starting at the third row.

Method 2: Import CSV File where Rows Meet Condition

library(sqldf)

df <- read.csv.sql("my_data.csv",
sql = "select * from file where `points` > 90", eol = "n")

This particular example will only import the rows in the CSV file where the value in the 'points' column is greater than 90.

The following examples show how to use each of these methods in practice with the following CSV file called my_data.csv:

```
"team","points","assists","rebounds"
"A",99,33,30
"B",90,28,28
"C",86,31,24
"D",88,39,24
"E",95,34,28
```


Example 1: Import CSV File Starting from Specific Row

The following code shows how to import the CSV file and skip the first two rows in the file:

```
#import data frame and skip first two rows
df <- read.csv('my_data.csv', skip=2)

#view data frame
df

B X90 X28 X28.1
1 C 86 31 24
2 D 88 39 24
3 E 95 34 28
```

Notice that the first two rows (with teams A and B) have been skipped when importing the CSV file.

By default, R attempts to use the values in the next available row as the column names.

To rename the columns, you can use the names() function as follows:

```
#rename columns
names(df) <- c('team', 'points', 'assists', 'rebounds')

#view updated data frame
df
```

team points assists rebounds
1 C 86 31 24
2 D 88 39 24
3 E 95 34 28

Example 2: Import CSV File where Rows Meet Condition

We can use the read.csv.sql function from the sqldf package to do so:

```
library(sqldf)

#only import rows where points > 90
df <- read.csv.sql("my_data.csv",
sql = "select * from file where `points` > 90", eol = "n")

#view data frame
df
```

team points assists rebounds
1 "A" 99 33 30
2 "E" 95 34 28

Notice that only the two rows in the CSV file where the value in the 'points' column is greater than 90 have

been imported.

Note #1: In this example, we used the eol argument to specify that the "end of line" in the file is indicated by n, which represents a line break.

Note #2: In this example, we used a simple SQL query but you can write more complex queries to filter rows by even more conditions.

How to Read a CSV from a URL in R