How can I select columns in R?

Authored by

stats writer

June 23, 2024

In order to select specific columns in R, you can use the bracket notation or the dollar sign notation. The bracket notation involves using the name of the data frame followed by the column index, while the dollar sign notation uses the name of the data frame followed by the column name. Additionally, you can also use the select() function from the dplyr package to choose columns based on their names or specific criteria. By using these methods, you can easily select and manipulate columns in R for data analysis and visualization purposes.

There are several ways to select data frame columns in R by using the R base and `dplyr` package. In this article, I will explain how to select columns by using the `select()` function from the dplyr package, R base bracket notation `df`. Using these I will cover examples like selecting a specific column/multiple columns from the data frame by name/position, and many more.

Sometimes you may need to change the column names, if so read rename data frame columns in r.

Key Points -

1. Quick Examples of Selecting Columns from the Data Frame

Following are quick examples of how to select data frame columns in R.

```
# Quick Examples of selecting columns

# Example 1: R base - Select columns by name
df

# Example 2: R base - Select columns from list
df

# Example 3: R base - Select columns by index position
df

# Example 4: Load dplyr
library('dplyr')

# Example 5: dplyr - Select columns by list of index or position
df %>% select(c(2,3))

# Example 6: Select columns by index range
df %>% select(2:3)

# Example 6: dplyr - Select columns by label name & gender
```

df %>% select('name','gender')
df %>% select(c('name','gender'))

Example 7: dplyr - Select columns except name & gender
df %>% select(-c('name','gender'))

Example 8: dplyr - Select columns between name and state
df %>% select('name':'state')

Example 9: dplyr - Select columns starts with a string
df %>% select(starts_with('gen'))

Example 10: dplyr - Select columns not start with a string
df %>% select(-starts_with('gen'))

Example 11: dplyr - Select columns that ends with a string
df %>% select(ends_with('e'))

Example 12: dplyr - Select columns that contains
df %>% select(contains('a'))

Example 13: dplyr - Select all numeric columns
df %>% select_if(is.numeric)

First, create an R DataFrame using the data.frame() function.

```
# Create DataFrame
df <- data.frame(
id = c(10,11),
name = c('sai','ram'),
gender = c('M','M'),
dob = as.Date(c('1990-10-02','1981-3-24')),
state = c('CA','NY'),
row.names=c('r1','r2')
)
df
```

Yields below output.

```
   id name gender        dob state
r1 10  sai      M 1990-10-02    CA
r2 11  ram      M 1981-03-24    NY
```

2. Get Columns using the R base

To select columns from a data frame in R, we can use the R base df bracket notation. In R, when working with a data.frame, we usually use the `$` symbol to refer to the column name along with the data frame object. However, this notation can be confusing and make the R code harder to read. Thus, the use of bracket notation is recommended as an alternative.

2.1 Select by Column Index

The `df` notation takes syntax `df`, so when using this notation to select columns in R, you can specify the column indexes/labels on the right after the comma. To select single/multiple columns by index, or range of column indexes using `starting_position:end_position` or by a list of index positions.

```
# R base - select specific column by index
df
```

Output:
"sai" "ram"

R base - by list of positions
df

R base - by range
df

Yields below output.

```
   name gender
r1  sai      M
r2  ram      M
```

2.2 Select by Name

Alternatively, to select columns by name in R you can use this notation. Simply, pass the specified column name that you want to get from a data frame, into df notation. It will return all the values of the specified column.

```
# R base - Select columns by name
df
```

Output
"sai" "ram"

2.3 Select Columns from List

Sometimes when we want to select multiple columns at a time from a data frame, you can use df notation. To specify these column names using vector within a notation. It will return the data frame with specified columns.

```
# R base - Select columns from list
df
```

Output
name gender
r1 sai M
r2 ram M

2.4 Select a column Using the `$` Operator

You can use the `$` operator to select a specific column by name. For example,

```
# Select specific column by name using $
df2 <- df$name
df2
```

Output
"sai" "ram"

3. Select Columns using the dplyr Package

You can use select() function from the dplyr package to get specified single/multiple columns of the data frame. This function allows the data frame as a first argument and the column position of single/multiple is the second argument.

To perform sequential operations within a dplyr package you can use the infix operator `%>%` from `magrittr`. which is `%>%` is known as the pipe operator. It pipes the data frame `df` into the next function. Whatever is on the left side of `%>%` is passed as the first argument to the function on the right side.

3.1 Select columns by Column Number

The `select()` function of `dplyr` package also supports selecting columns by index from the R data frame. Use this function if you want to select the data frame columns by index or position. The following example returns columns 2 and 3 from the data frame.

```
# Load dplyr
library('dplyr')

# Select columns
df %>% select(2,3)

# Select columns by list of index or position
df %>% select(c(2,3))

# Select columns by index range
df %>% select(2:3)
```

Yields below output.

```
# Output
name gender
r1 sai M
r2 ram M
```

3.2 Select columns by Name using dplyr

You can also select data frame columns by name, select multiple columns, and all columns in the

list (contains in the list) using the dplyr package. The first example from the following selects the specified columns that are supplied to the select() function with a comma separator. The second example selects all columns from the list.

```
# Select columns by label name & gender
df %>% select('name','gender')
df %>% select(c('name','gender'))
```

Output
name gender
r1 sai M
r2 ram M

3.3. Get Columns of Not specified

To use the select() function from dplyr for column selection, simply pass the list of column names(don't want to get) specifying by negative vector. It will drop specified columns from the DataFrame by Name and return the remaining columns of the data frame.

```
# Select columns except name & gender
df %>% select(-c('name','gender'))
```

Output
id dob state
r1 10 1990-10-02 CA
r2 11 1981-03-24 NY

3.4. Select All Columns Between 2 Columns

You can also get the particular portion of columns of the data frame by using the range operator(`:`) within the `select()` function of the `dplyr` package. You can specify the range within a select() function with starting point and ending point. This will return all columns between the starting position and the ending position, including them.

```
# Select columns between name and state
df %>% select('name':'state')
```

Output

name gender dob state
r1 sai M 1990-10-02 CA
r2 ram M 1981-03-24 NY

3.5. Get Selected Columns Use starts_with()

Use `starts_with()` function within a `select()` function to get the columns based on certain criteria. In this case, it selects columns whose names start with the specified prefix. This will check for column names that start with the specified prefix.

```
# Select columns starts with a string
df %>% select(starts_with('gen'))
```

Output
gender
r1 M
r2 M

3.6. Get Selected Columns Use ends_with()

Use `ends_with()` function within a select() function to get the columns based on certain criteria. In this case, it selects columns whose names end with the specified suffix. This will check for column names that end with the specified suffix.

```
# Select columns that ends with a string
df %>% select(ends_with('e'))
```

Output
name state
r1 sai CA
r2 ram NY

3.7. Get Columns Containing character

In case you want to select all columns that contain a character or string use `contains()`. The following example selects all columns that contain a character `a`.

```
# Select columns that contains
df %>% select(contains('a'))
```

Output
name state
r1 sai CA
r2 ram NY

3.8. Select All Numeric Columns

Selecting all numeric columns is one of the most used operations. If you have a data frame with columns with strings and integers, performing certain statistical operations on the entire data frame results in error hence, first you need to select all numeric columns using `is.numeric` input to `select_if()` and operate on the result of it. Use `is.character` to select columns of character type.

```
# Select all numeric columns
df %>% select_if(is.numeric)
```

Output
id
r1 10
r2 11

Frequently Asked Questions of Select Columns in R

How do I select specific columns from a data frame?

You can use the R base `df` notation to select specific columns from the data frame by column index/column label. For example, `df` or `df`.

How do I select columns by index number?

To select columns by index number you can use the the R base df notation. for example, `df`

How can I remove columns from a data frame?

You can use negative indexing to exclude specific columns. For example, `df <- df`

Are there any functions for selecting columns more efficiently?

Use the `dplyr` package, which provides the `%>%` pipe operator and functions like `select()`to select columns from the data frame very efficiently. For example, `df %>% select(col1, col2)`

5. Conclusion

In this article, you have learned how to select single/multiple columns/range of columns using the R base bracket notation `df` and the select() method from the dplyr package, by column index/ column label with multiple examples.

Related Articles

References