Working with tables in R

Alla Tambovtseva

Working with tables: how to filter rows and columns

Today we will continue working with the data set on Chilean referendum. Let's upload the file:

```
df <- read.csv("http://math-info.hse.ru/f/2017-18/ps-ms/Chile.csv")
```

And recall how it looks like:

```
View(df)
```

We have already learnt that a certain column can be chosen via adding the dollar sign `$`. We can save a column as a vector as well:

```
educ <- df$education  # respondents' education
head(educ)  # some first values
```

```
## [1] P  PS P  P  S  P
## Levels: P PS S
```

To select rows from a data frame based on a certain condition we need the `subset()` function. At first we write the name of our data frame and then we indicate the condition. For instance, we can choose people older than 30 years and save the results to the data frame `old`:

```
old <- subset(df, df$age > 30)
head(old)
```

```
##     X region population sex age education income statusquo vote
## 1   1      N     175000   M  65         P  35000   1.00820    Y
## 3   3      N     175000   F  38         P  15000   1.23072    Y
## 4   4      N     175000   F  49         P  35000  -1.03163    N
## 9   9      N     175000   F  41         P  15000  -1.01292    U
## 10 10      N     175000   M  41         P  15000  -1.29617    N
## 11 11      N     175000   M  64         P  15000   1.36566    Y
```

For non-strict inequalities we can use the same operators as in mathematics (note that there is no spaces between `>` and `=`):

```
old2 <- subset(df, df$age >= 30)  # not younger than 30
head(old2)
```

```
##     X region population sex age education income statusquo vote
## 1   1      N     175000   M  65         P  35000   1.00820    Y
## 3   3      N     175000   F  38         P  15000   1.23072    Y
## 4   4      N     175000   F  49         P  35000  -1.03163    N
## 9   9      N     175000   F  41         P  15000  -1.01292    U
## 10 10      N     175000   M  41         P  15000  -1.29617    N
## 11 11      N     175000   M  64         P  15000   1.36566    Y
```

Of course, conditions can be combined. If two conditions should hold true at the same time (recall the intersection of sets), they are joined with `&` operator. Suppose we want to choose males older than 50 years old. To do this, we should add two filters, one for the variable `age` and another for the variable `sex`:

```
old_males <- subset(df, df$age > 50 & df$sex == "M")
head(old_males)
```

```
##     X region population sex age education income statusquo vote
## 1   1      N     175000   M  65         P  35000   1.00820    Y
## 11 11      N     175000   M  64         P  15000   1.36566    Y
## 19 19      N     175000   M  67         P  75000   1.32279    Y
## 25 25      N     175000   M  58         P  35000   1.52601    Y
## 30 30      N     175000   M  53        PS  35000  -1.29617    A
## 44 44      N     125000   M  51         P  15000   0.52166    Y
```

Important: note that in the example above we used `==` for equality testing. In R and in programming in general, a single `=` is used for value assignment and a double `==` is used for equality testing. If we put only one `=`, we will get an error.

If we want R to choose rows that suit to at least one condition (only the first condition is true, only the second condition is true, both conditions are true), we need the `or` operator that is `|`. Let's select respondents that are either younger than 30 or older than 60:

```
young_old <- subset(df, df$age < 30 | df$age > 60)
head(young_old)
```

```
##   X region population sex age education income statusquo vote
## 1 1      N     175000   M  65         P  35000   1.00820    Y
## 2 2      N     175000   M  29        PS   7500  -1.29617    N
## 5 5      N     175000   F  23         S  35000  -1.10496    N
## 6 6      N     175000   F  28         P   7500  -1.04685    N
## 7 7      N     175000   M  26        PS  35000  -0.78626    N
## 8 8      N     175000   F  24         S  15000  -1.11348    N
```

Note that it is not an exclusive `or` that does not allow both statements to be true simultaneously. However, in our case above only one condition can hold.

If you forgot the `subset()` function, it is usually easy to select rows and columns using only square brackets. All we should remember is the order of values in square brackets. The row index (condition on rows) usually goes first, and the column index (condition on columns) goes second. For example, choose the value that is stored in the first row in the second column:

```
df[1, 2]
```

```
## [1] N
## Levels: C M N S SA
```

Or in the second row in the third column:

```
df[2, 3]
```

```
## [1] 175000
```

If we need exactly one row and all the columns, we can leave the second position blank (it means *all* by default):

```
df[2, ]  # all indicators for the 2nd respondent
```

```
##   X region population sex age education income statusquo vote
## 2 2      N     175000   M  29        PS   7500  -1.29617    N
```

And, vice versa, if we need exactly one column, we can skip the first value in square brackets:

```
df[, 4] # 4th column,
```

```
##   [1] M M F F F F M F F M M M F F M M F M M F F F M F M F F F M M F F M M
##  [35] M F M M M F F M F M F F F M M F F F M M M M M F F M M F F F M F F M
##  [69] F M M M F F M F F M M M M M M F M F F F M M F M F F M F F F M M F M
```

```
##  [103] M M F F M F F M M F F M M M M M F F F F M M F F F M F M M M F F F F
##  [137] M F F M M M F F F F F F M M F F F M M F M F M F M F M F F F M M F
##  [171] M F M F F F F F M M F F F M M M F F M F M F M F F F M M F M F F M M F F
##  [205] M M M F F M M F F M M M F M F M F M F M F F M F F F F M M F M M F F
##  [239] M F M M F F M M F M F M M F F M F F F M F M M M F M F M F M F F F M
##  [273] M F M F F M M M F F M M F F M M F M M M F M F M F M F F M F M F F M
##  [307] F M M F F F F M M M F M F F F M F F M F M M M M F F F M M F F M M M
##  [341] F F M M M F M F F M F M M F M M F F F M F M F F M F M M F F M M F F
##  [375] F M M F F M M M F F M M F M F M F M F M F F F M M M M M M F F M M M
##  [409] F F M M M F F M M F F F F F M M F M F F M M F F M F M F F M F M F F
##  [443] M M F M F F F M F F F M M M F F M M F M M M F F M F M F F M F F M M F
##  [477] M M F F M F F F M F F M M M M F F M M M F F F M M M M F F F M M F F
##  [511] F F M F M F F M M M M F F F M M F F F M M M F M F M F F M F F F M F
##  [545] M M M M F F M F M M F M M F F M F M M M F M F F M F F F M M F M F F
##  [579] M F F F M F M M F M F M M M F F F M M F M F F F M F M F M M F F M F
##  [613] F M M F F M M M M F M F F M F F F M F M F F M F M F M M F F M M M M
##  [647] F F F M F F M M M F F M M F F M F M F F M F M M F M F M M F M F M M
##  [681] F F M F M M M M F F F M F M M M F F M M F M F M M M M F F F M F M M M
##  [715] F F M M F F F F F M M F F M M M F F F M M F F M M M F F M M F M M M F
##  [749] F M F F F M M F M F M M F M F M F F M M M F M F F F M F F M F M F M
##  [783] F M M F F F M M F M F F M F F M M M M F F F M F F M F M F M M M F F
##  [817] F F M M F F M F M M F M M F M F F M M M F F M M F M F M F M F F M M
##  [851] F F M M F F F M M M M F F F M F F M M M M M M F F M M F F M F M M M
##  [885] M F F M F M M F M M F F M F M F F F M F M F F M M M F F F M M M M F
##  [919] F M M M F F M F F F M M M M F F F M M F M F M F M F M F M F F M M M F F
##  [953] M M F F F F F M M F M F M F F M M M M M F F F M F M F M M M M F F M
##  [987] F M F M F M F M M M F F M F M F F M M M M M M F F F M M F F M M F F F
## [1021] F F M M M F F M M M M F M F M F F M F M F M M M F F F M M M F F F M
## [1055] M F M F F M M F M M M F F F M M M F F M F M F F M M F F F M F M F F M
## [1089] M F M M F F M F F F M M M F M F F F F M F M F F M M M F M F M F F M
## [1123] M F M M M F M F M F M M F F F M M M F M F M F F M F F M M F F M M M
## [1157] M M F F F F M M F F F F F F M F M F F M M F F F F F M M M F F M M M
## [1191] F M F M F M F M F F F F M M M F M F F F F M F M M F F M M M M F M M
## [1225] F F F M M M M F F M F F F M F M F M F F M M F F F M M M F F M F F M
## [1259] M M M M F F M F F M M M M M F F M F M F F M F M F M M F F F M M F F
## [1293] M F M M F F F M F M M F F F M F F M F M M F M M F M F M M M F F M M
## [1327] F F M F F M M M F F M M F F M M F F F F M F F M F M F M F M M F F M
## [1361] F F F M M F M F F M F F M M M F F F M M F F M M M F M M F M F M F M
## [1395] F M M F M F M M F M M M M F M F F M M M F M M M F F F M F M F F F F
## [1429] M M M F M M F F F M M F M F F M M M F F M M M F F M F F M M M F M M
## [1463] F M F F M M F F F F M M M F M M F F F F M M M F M F M F F M M F F F
## [1497] F M M M F F M M M M F M F F F F M M M M F F F M M F F F M F F F M M
## [1531] F M M M F F M F F M F F F M M M M M F F M F M F M M F F M F F F M M
## [1565] M F M F F M M M F M F F M F M F F F F M M F M M M F F M M F F F F F
## [1599] M M F F F M M M M M F F M F F F M M M F F F F F M M M M F M F F M F
## [1633] M M F M M F F F M M F F M M M F F M F M F F M M M F F M F F M F M F
## [1667] F F M M M F F F M M F F F M F F M F M M F F M F M F F F M M M F F F
## [1701] F M M M F F F M M M M M F F M M F F M F F M M M M M F F M M M F M
## [1735] F M M M F M F F M M F M M M F M F F M M M F F F F M F M F M F M M M F
## [1769] M F F F M F M F M M F F F M F M M M M F F F M M F F F F M F F M F F
## [1803] M F M M F F F M F M M F M F M F F M M F F F M M M F M F M F M F F M
## [1837] F M F M M F F M M F M F M F M M F F F F F M M M F F M M M M F F F M
## [1871] M F M F F M F F M M M M F F F M M M F F F M F F M M F M F M F F F M
## [1905] M M M M F F F F F M M F M M M F F M F F M M M F F M F F F M M M M F
```

```
## [1939] F F M F F M F M M F F M F F F M F M M F F M F M F F M M M M F F M F
## [1973] F M M F F M M F F M F M M M F F M F F F M M M F F M F M F F M F M F
## [2007] M M M F F F M M F M M M F F F M F M M M F M F M F F M M F M F M F M
## [2041] F F M F M F M F M M M M F F F M F F F M F F M M M M F M F F M M F F
## [2075] M F M F F M F M F F M M F M M M F F F M M M M F M F F F M F M M F M M
## [2109] F F M F F M F M M F F F M F M M F F M F F M F M M M M F F F F M M F F
## [2143] M M M F F M F M F M F M M F F M F M F F M M M F F M M M M M F M F F
## [2177] M F M M F F M M M F F M M M M F F M M M F F F M M F F M M F F F M M
## [2211] F F F M M F M F F M M M M F F F F F M M M M F F M M F F M F F F M M
## [2245] M F F M F M F M F M M F F F M M F F F M M M F M F F M M F F F F F M
## [2279] M F F M F M M M F F M F M M M F F M F F F M M F M M F F F M
## [2313] F M M F F M F M F M M F F M M F F M M M F M F F F M F M F M F M F F
## [2347] M M M F F F M F M M F M F F F F M M M M M F F F M F F F M M F F M M
## [2381] F F M M F M F M F M M F F F M M M M F F F F M M M M F F F M F M M F
## [2415] M M M F F F F F F M M M F F F M M M F F F F M F F M F F M F F F F F
## [2449] M M F F M F F M F M F F F M M F F F F M F F M M M F F M F F M F F M
## [2483] M F F M F F F M M F F M F M F F F M M M F F F M M F F F F F M M F F
## [2517] M F F M F M M F M M M F F F M F M M F M F M F M F F F M M F M F F M
## [2551] M F F F M M F M F F M F F F M M M F F M F F M F M F M F M M M F F M F
## [2585] M F M M M F M M F F F M F F M M M F F F M M F F F M M M F M F M M F
## [2619] M F F M F M F M F M F M M F M M F F F F M M M M M F F F F F M M F M
## [2653] F M M F F F M M F M F M F M M F F M M F F F M M M F F F M M F F M F
## [2687] M F M F M F F F M M F F M M
## Levels: F M
```

It is possible to add conditions in square brackets as well. Again, if we want to filter rows, the condition should go first, and then we can specify the columns needed:

```
d1 <- df[df$age > 30, 1:5]  # from 1st to 5th column
head(d1)
```

```
##     X region population sex age
## 1   1      N     175000   M  65
## 3   3      N     175000   F  38
## 4   4      N     175000   F  49
## 9   9      N     175000   F  41
## 10 10      N     175000   M  41
## 11 11      N     175000   M  64
```

If a sequence of indices is used (like in the code above), mind that both endpoints are included. Here the 1st and the 5th columns are not left aside.

If columns that we need are not side by side, their indices should be passed as a vector:

```
d2 <- df[df$age > 30, c(1, 3, 5)]  # vector c(1, 3, 5)
head(d2)
```

```
##     X population age
## 1   1     175000  65
## 3   3     175000  38
## 4   4     175000  49
## 9   9     175000  41
## 10 10     175000  41
## 11 11     175000  64
```

At last, one more helpful thing. So as to select columns by their names, the `subset()` function can be used, and columns needed should be again written as a vector:

```
d <- subset(df, select = c(age, sex, vote))
head(d)
```

```
##   age sex vote
## 1  65   M    Y
## 2  29   M    N
## 3  38   F    Y
## 4  49   F    N
## 5  23   F    N
## 6  28   F    N
```

If we want to choose a lot of variables and it is easier to exclude some columns rather than list others to be left, we can put a minus before the vector in `select`:

```
data <- subset(df, select = -c(statusquo)) # take all except statusquo
head(data)
```

```
##   X region population sex age education income vote
## 1 1      N     175000   M  65         P  35000    Y
## 2 2      N     175000   M  29        PS   7500    N
## 3 3      N     175000   F  38         P  15000    Y
## 4 4      N     175000   F  49         P  35000    N
## 5 5      N     175000   F  23         S  35000    N
## 6 6      N     175000   F  28         P   7500    N
```

Sometimes it can be more convenient, especially if remember that this operation can be done with the help of the same `subset()` function.