Data Management With R: Data Transformation

Matthias Haber

18 September 2017

Making sure everyone is set up

Data transformation with dplyr

Homework Exercises

Making sure everyone is set up

Packages

```r
library(tidyverse)
```

Data

336,776 flights that departed from New York City in 2013

```r
# install.packages("nycflights13")
library(nycflights13)
```

year	month	day	dep_time	sched_dep_time	dep_delay
2013	1	1	517	515	2
2013	1	1	533	529	4
2013	1	1	542	540	2
2013	1	1	544	545	-1

Data transformation with dplyr

Piping

The pipe operator %>% (Ctrl/Cmd+Shift+M) allows you to write code in sequences which has several benefits:

- serves the natural way of reading ("First this, then this, ...")
- avoids nested function calls
- minimizes the need for local variables and function definitions

Piping

dplyr is designed to work with the pipe, so this

```
df %>%
  select(x, y) %>%
  filter(year == 2017)
```

returns the sames as this

```
filter(select(df, x, y), year == 2017)
```

Variable types

- int: integers
- dbl: doubles, or real numbers
- chr: character vectors, or strings
- dttm: date-times (a date + a time)
- lgl: logical, vectors that contain only `TRUE` or `FALSE`
- fctr: factors
- date: dates

`dplyr` core functions

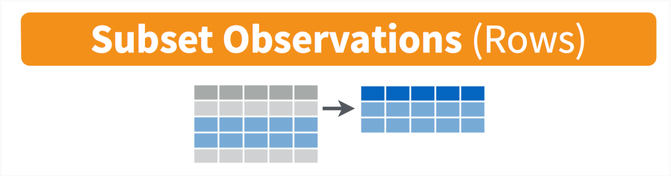
- `filter()`: select rows by their values
- `select()`: select columns by their names
- `arrange()`: order rows
- `mutate()`: create new variables
- `summarize()`: collapse many values down to a single summary
- `group_by()`: operate on it group-by-group
- `rename()`: rename columns
- `distinct()`: find distinct rows

Command structure (for all dplyr verbs):

- first argument is a data frame
- return value is a data frame
- nothing is modified in place

`filter()`

`filter()` allows to subset observations based on their values. The function takes logical expressions and returns the rows for which all are `TRUE`.



filter()

Let's select all flights on January 1st:

```
filter(flights, month == 1, day == 1)
```

year	month	day	dep_time	sched_dep_time	dep_delay
2013	1	1	517	515	2
2013	1	1	533	529	4
2013	1	1	542	540	2
2013	1	1	544	545	-1
2013	1	1	554	600	-6
2013	1	1	554	558	-4

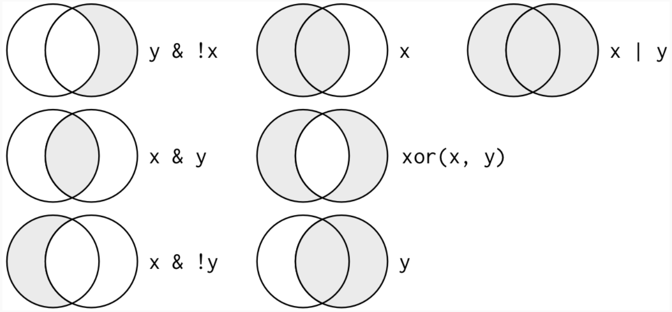
`filter()`

`filter()` revolves around using comparison operators: >, >=, <, <=, `!=` (not equal), and == (equal).

`dplyr` functions like `filter()` never modify inputs but instead return a new data frame that needs to be assigned to an object if you want to save the result.

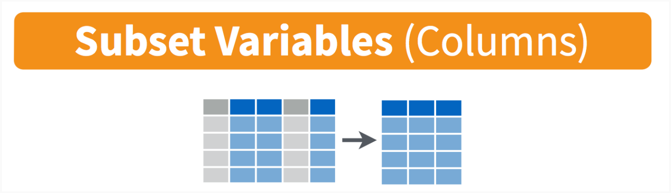
```
jan1 <- filter(flights, month == 1, day == 1)
```

Combining conditions



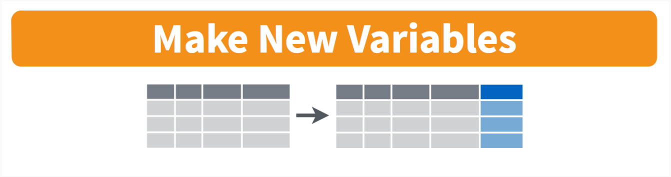
`select()`

`select()` allows to select variables.



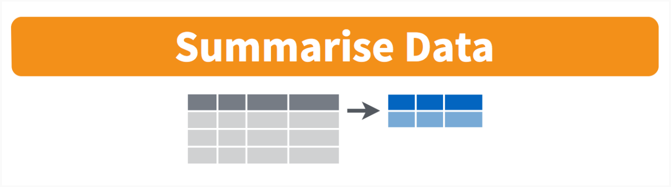
mutate()

mutate() allows to select variables.

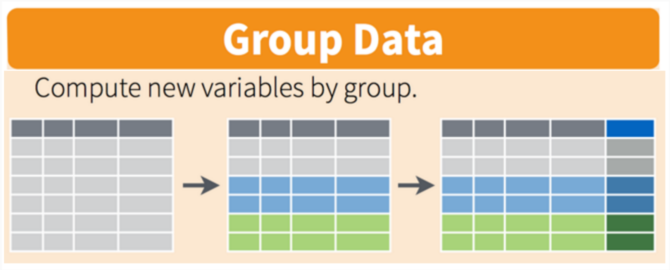


summarize()

summarize() allows to select variables.



group_by()



Homework Exercises

Homework Exercises

That’s it for today. Questions?