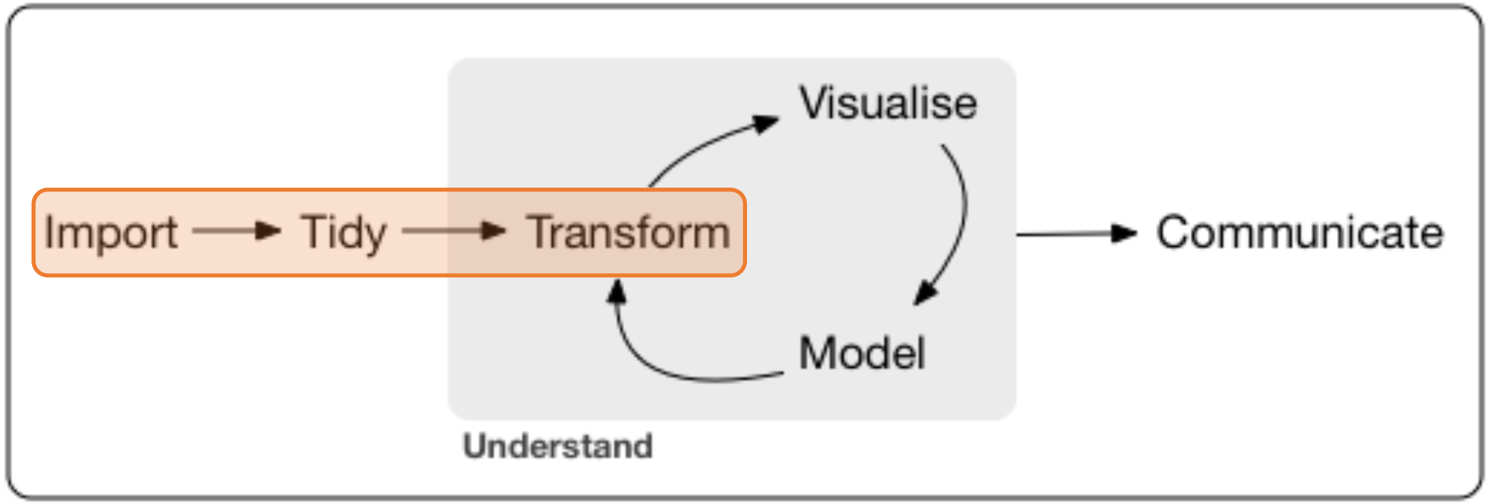
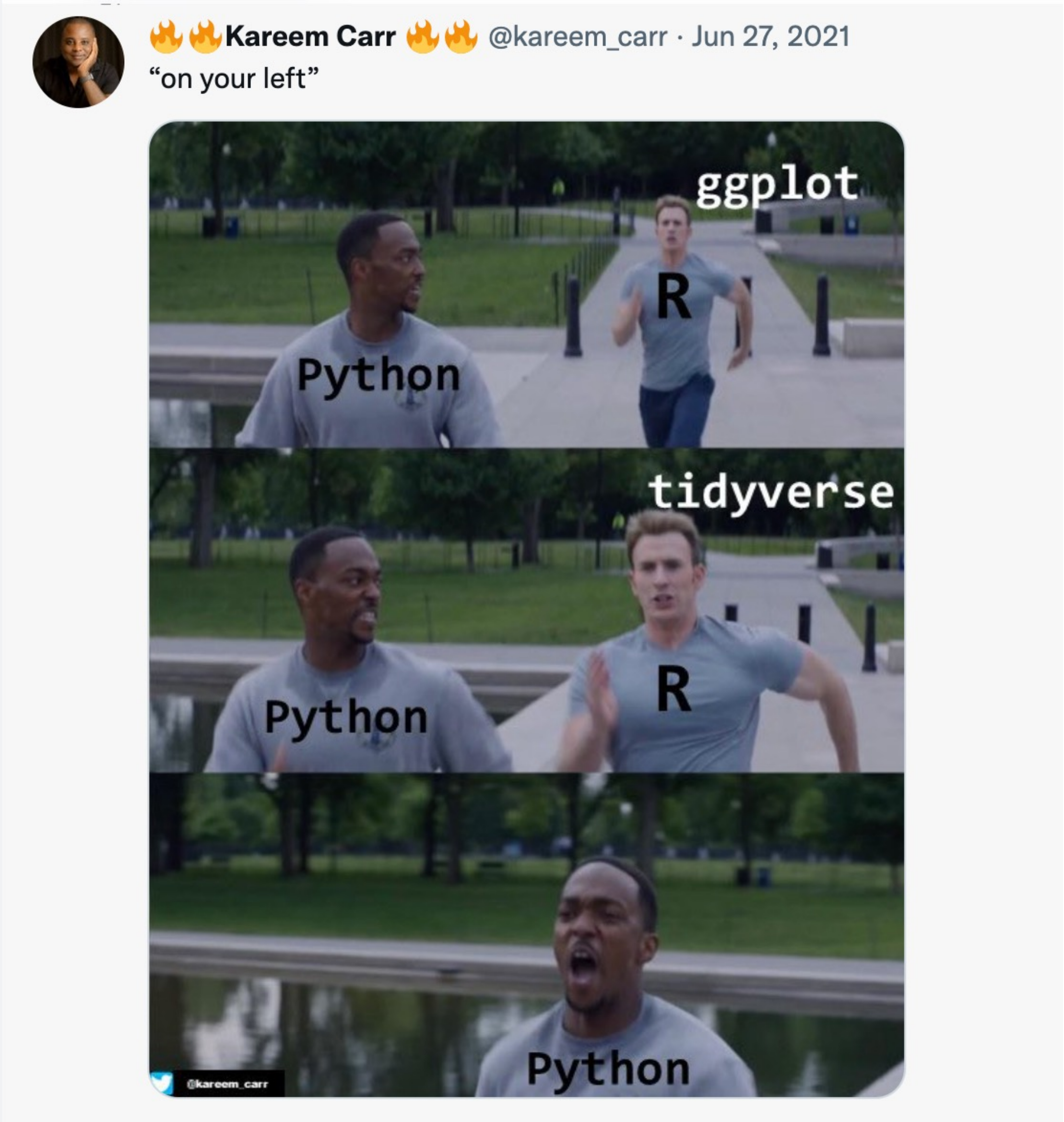
Week 6: Data transformation (II)

A typical data science project :





🔥🔥Kareem Carr 🔥🔥 @kareem_carr · Jun 27, 2021

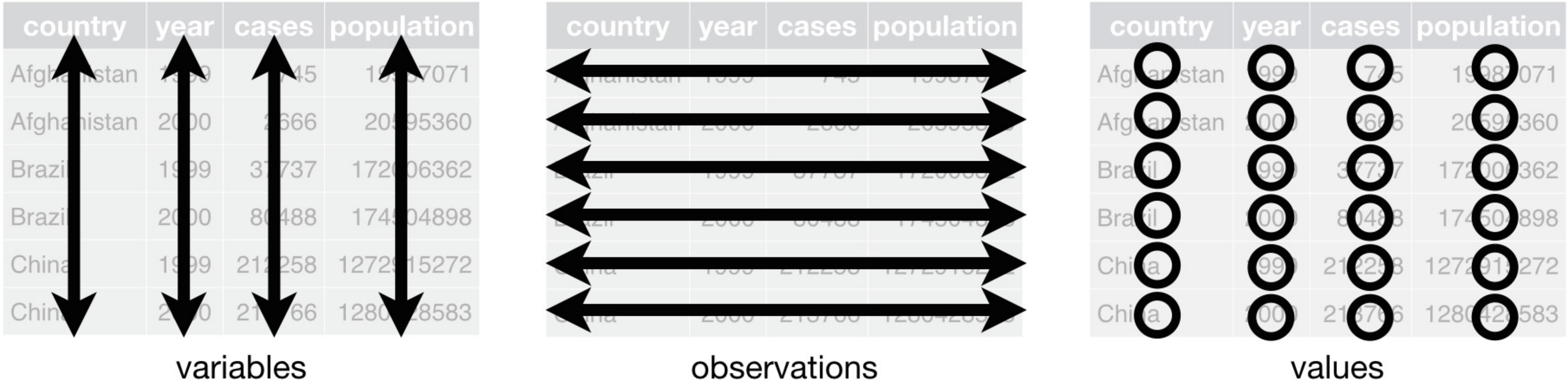
“on your left”

What makes a dataset tidy?

Each variable must have its own column.

Each observation must have its own row.

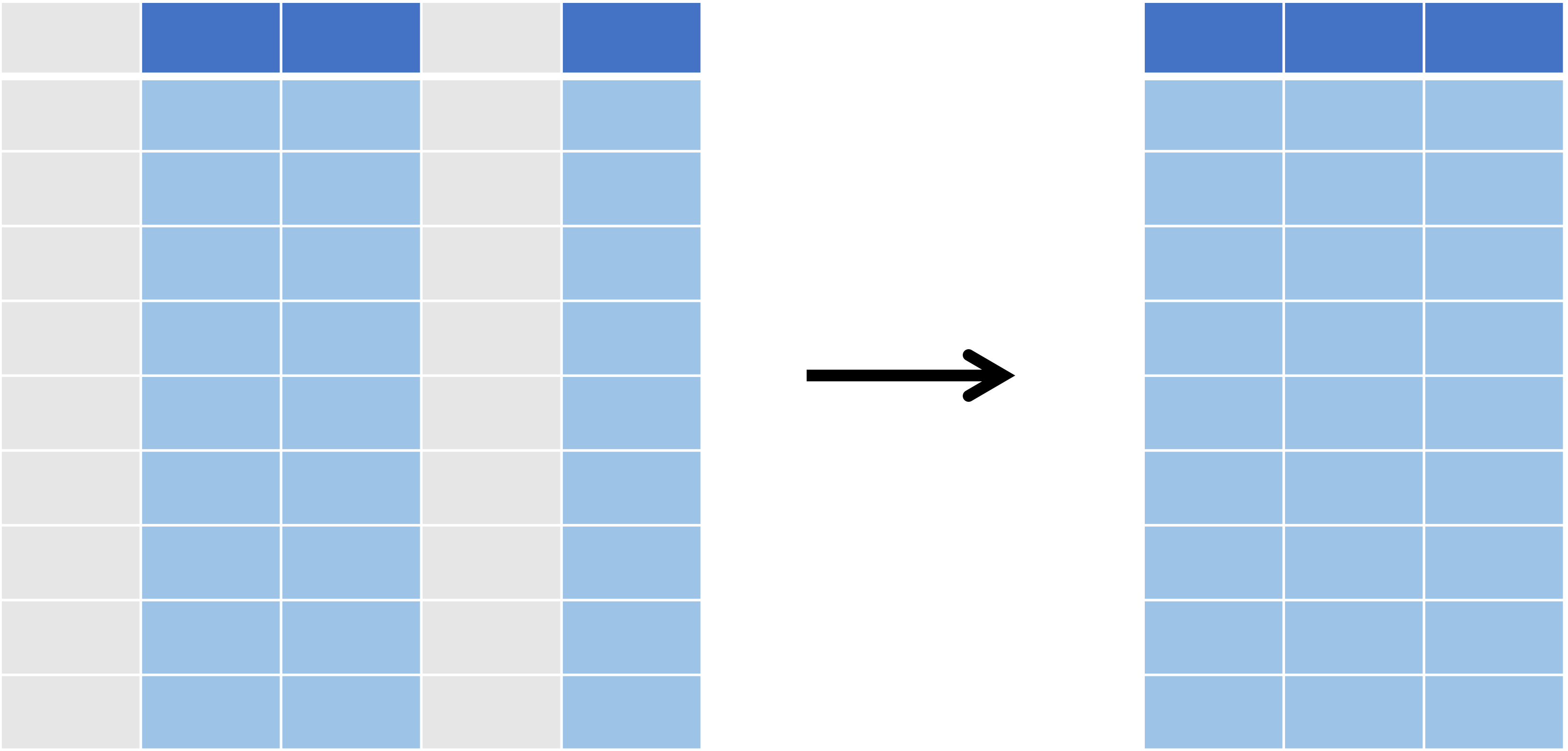
Each value must have its own cell.



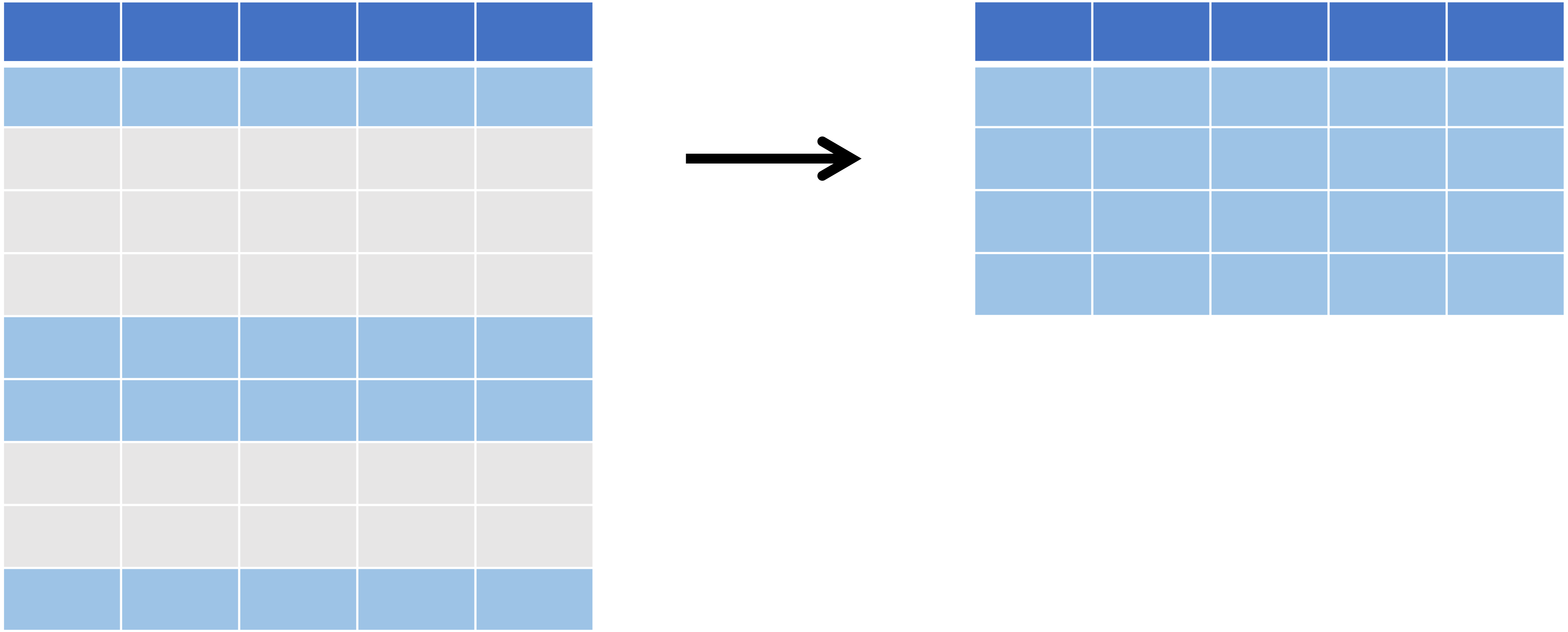
country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

dplyr, ggplot2, and all the other packages in the tidyverse are designed to work with tidy data.

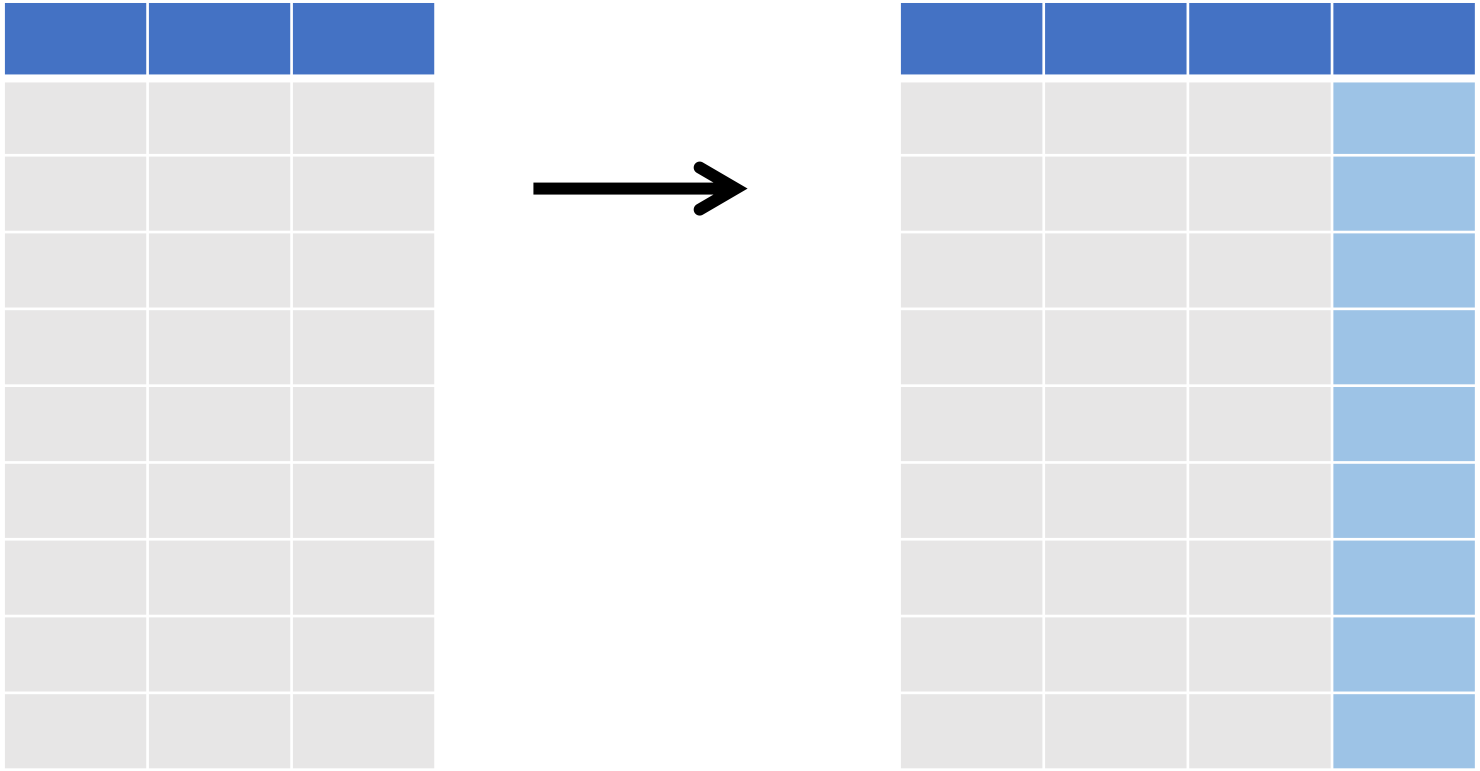
Which function?



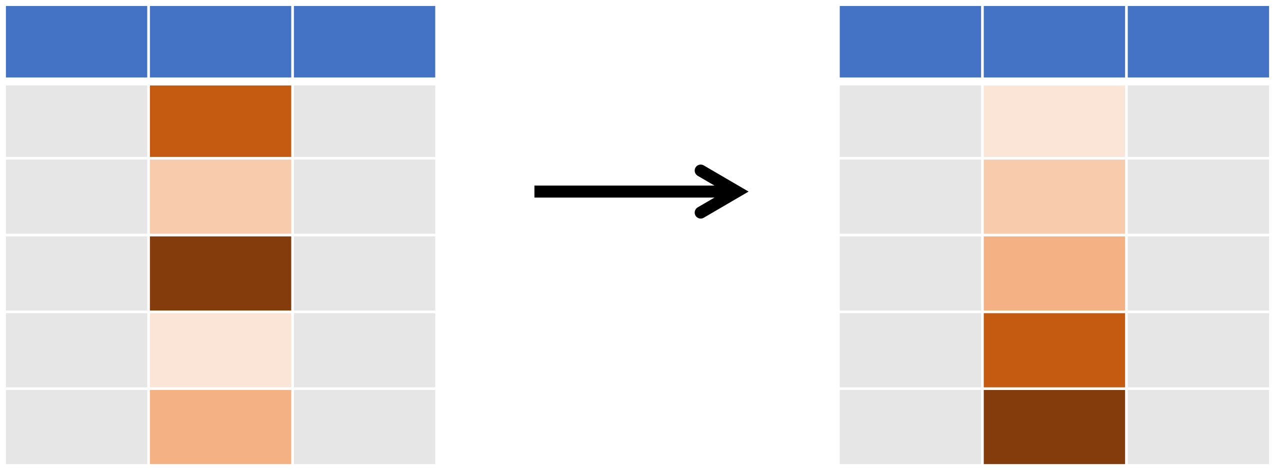
Which function?



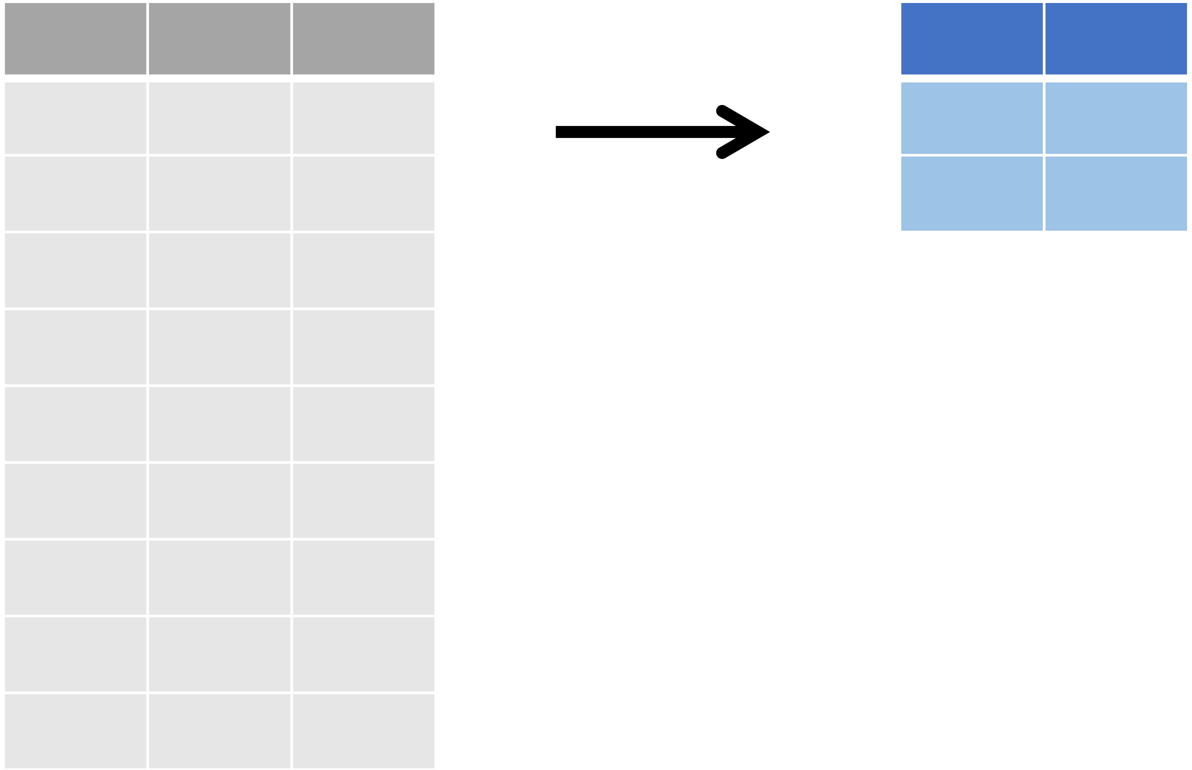
Which function?



Which function?



Which function?



Let’s recap

- What is the arrange() fuction?
- What is the filter() fuction?
- What is the select() fuction?
- What is the mutate() fuction?
- What is the summarise() fuction?
- What is the group_by() fuction?
- What do we use the pipe (%>%) for?

Today

- **count()**
- **Combine functions using the pipe (%>%)**
- Pivoting
- Separating
- Other useful functions: rename(), recode(), glimpse(), clean_names(), as.character/ as.numeric/ as.Date

Let's get set

- Create an R project for this session and name it “week_6”
- Open the script file and rename it
- Load the tidyverse package
- Import the Sinai covid dayaset data

count()

- Count the unique values of one or more variables
    - count(sinai_covid, facility)
    - count(sinai_covid, facility, sex)

Combining functions using the pipe %>%

- The point of the pipe is to help you write code in a way that is easier to read and understand.
- Let’s explore again the age of patients in each facility, using the pipe:

```
sinai_covid %>%
        count (facility)
```

Your turn! Exercise 1

- Use the pipe to:
- Extract all patients from THE MOUNT SINAI HOSPITAL and MOUNT SINAI BI BROOKLYN
  Select facility, id, sex, ethnicity and bmi
- Count ethnicity by facility

Your turn! Exercise 2

- What is the mean age of patients per facility?

Use the pipe!

Your turn! Exercise 3

- How many patients in each facility are older than 50 years?

Use the pipe!

Your turn! Exercise 4

- What is the median BMI by patient outcome (deceased_indicator)?

Use the pipe!

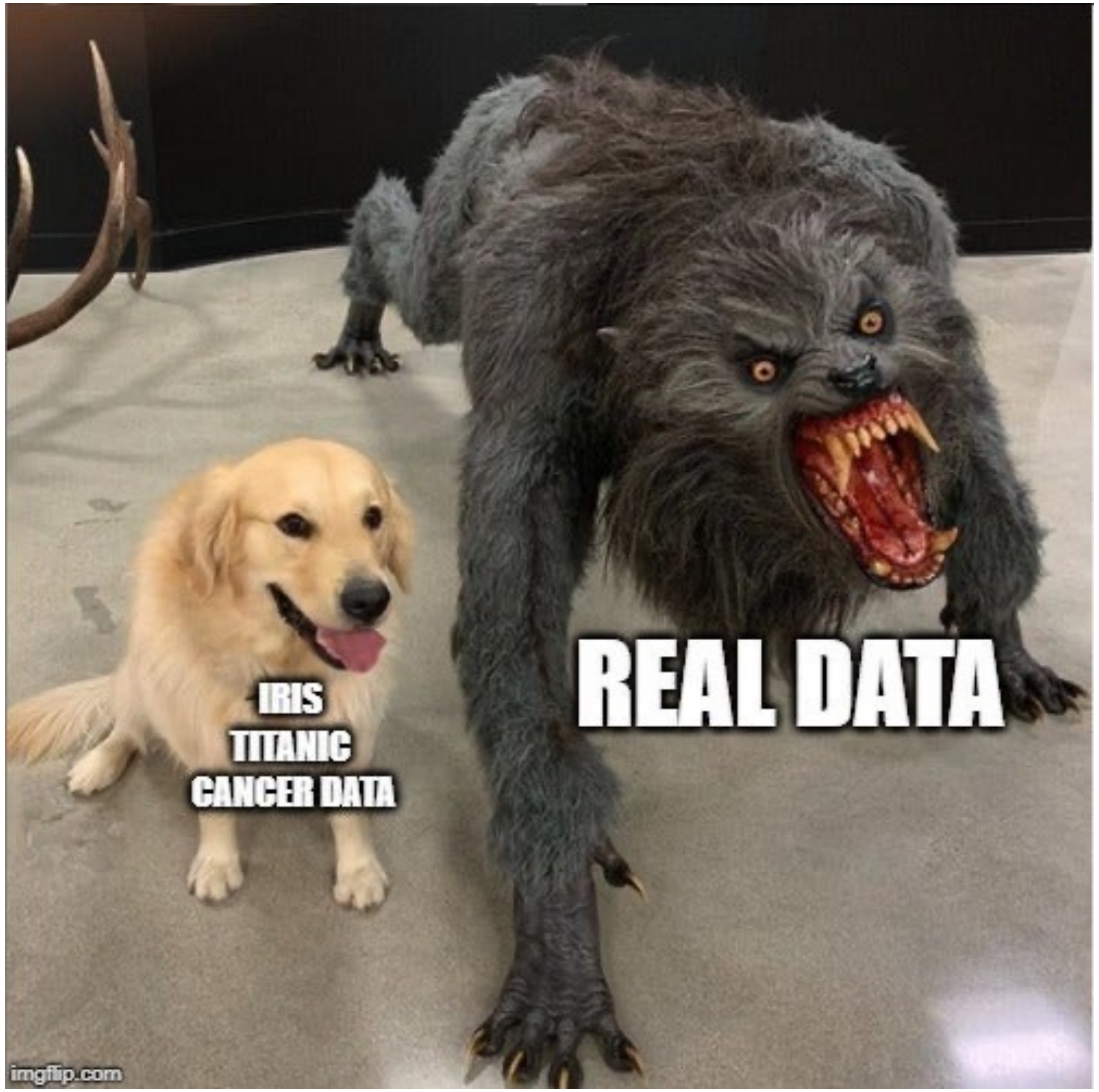
Today

- count()
- Combine functions using the pipe (%>%)
- **Pivoting**
- **Separating**
- Other useful functions: rename(), recode(), glimpse(), clean_names(), as.character/ as.numeric/ as.Date

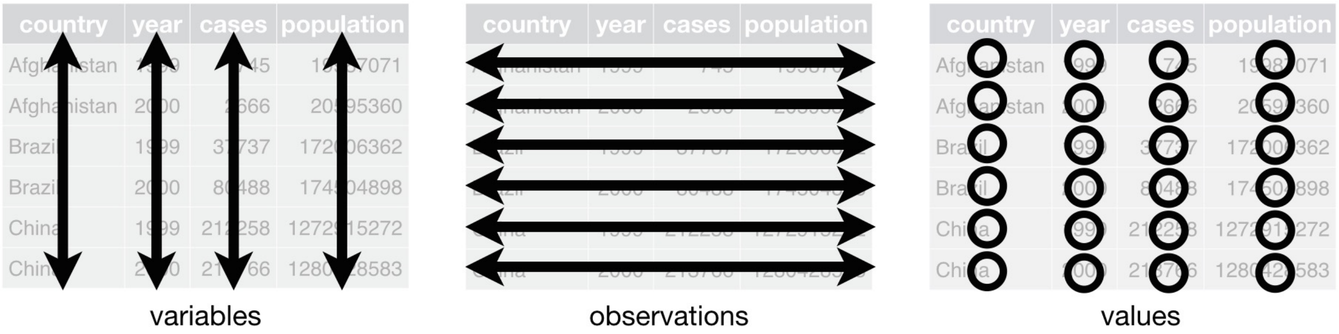
Tidy data

“Happy families are all alike; every unhappy family is unhappy in its own way.” –– Leo Tolstoy

“Tidy datasets are all alike, but every messy dataset is messy in its own way.” –– Hadley Wickham



Tidy dataset



Two common problems:

1. One variable might be spread across multiple columns.
2. One observation might be scattered across multiple rows.

Pivoting

Step 1: Figure out what the variables and observations are

Step 2: Identify the problem

Step 3: Fix the problem

- To fix most problems, you’ll need two important functions in tidyr:

  pivot_longer()

  pivot_wider()

pivot_longer()

- A common problem is a dataset where some of the column names are not names of variables, but *values* of a variable.
- Take table4a: the column names 1999 and 2000 represent values of the year variable, and each row represents two observations.
- To tidy a dataset like this, we need to **pivot** using pivot_longer

Your turn! Exercise 5

- Print table4b
- What is the problem?
- How would you fix it?

Your turn! Exercise 6

- Import the “experiment1” dataset
- What is the problem?
- How would you fix it?

pivot_Wider()

- pivot_wider() is the opposite of pivot_longer().
- Take table2: an observation is a country in a year, but each observation is spread across two rows.
- To tidy this up, we only need two parameters:
  1. The column to take variable names from (type).
  2. The column to take values from (count).

Your turn! Exercise 7

- Import the “experiment2” dataset
- What is the problem?
- How will you fix it?

Separating

- table3 has a different problem: we have one column (rate) that contains two variables (cases and population).
- To fix this problem, we’ll need the separate() function.

Case Study

- The tidyr::who dataset contains tuberculosis (TB) cases broken down by year, country, age, gender, and diagnosis method.
- This is a very typical real-life example dataset.
- Let’s tidy it

Case Study

1. country, iso2, and iso3 are three variables that redundantly specify the country.
2. year is a variable.
3. new_sp_m014 - new_rel_f65 are counts of new TB cases recorded by group. Column names encode three variables that describe the group - these are values, not variables.
4. We need to gather all the columns (group). We know the cells represent the count of cases, so we’ll use the variable cases.
5. There are a lot of missing values in the current representation, so we’ll use values_drop_na

Case Study – data dictionary

- The first three letters of each column denote whether the column contains new or old cases of TB. In this dataset, each column contains new cases.
- The next two letters describe the type of TB:
    - rel stands for cases of relapse
    - ep stands for cases of extrapulmonary TB
    - sn stands for cases of pulmonary TB that could not be diagnosed by a pulmonary smear (smear negative)
    - sp stands for cases of pulmonary TB that could be diagnosed by a pulmonary smear (smear positive)
- The sixth letter gives the sex of TB patients. The dataset groups cases by males (m) and females (f).
- The remaining numbers gives the age group. The dataset groups cases into seven age groups:
    - 014 = 0 – 14 years old
    - 1524 = 15 – 24 years old
    - 2534 = 25 – 34 years old
    - 3544 = 35 – 44 years old
    - 4554 = 45 – 54 years old
    - 5564 = 55 – 64 years old
    - 65 = 65 or older

- What do you think should be done to tidy this data?

Case Study – solution using the pipe

```
who_clean <- who %>%
        pivot_longer( cols = new_sp_m014:newrel_f65,
                names_to = "key",
                values_to = "cases",
                values_drop_na = TRUE) %>%
        mutate(key = stringr::str_replace(key, "newrel", "new_rel")) %>%
        separate(key, c("new", "var", "sexage")) %>%
        select(-new, -iso2, -iso3) %>%
        separate(sexage, c("sex", "age"), sep = 1)
```

Today

- count()
- Combine functions using the pipe (%>%)
- Pivoting
- Separating
- **Other useful functions: rename(), recode(), glimpse(), clean_names(), as.character/ as.numeric/ as.Date**

Manual column names cleaning

- rename() uses the style NEW = OLD (the new column name is given before the old column name)

Automatic column names cleaning

- The function clean_names() converts all names to consist of only underscores, numbers, and letters

as.character/ as.numeric/ as.Date

```
sinai_covid <- sinai_covid %>%
        mutate (copd = as.factor(copd))
```

```
sinai_covid <- sinai_covid %>%
mutate_at (vars (sex:cancer_flag), as.factor)
```

Recode()

```
sinai_covid %>%
        mutate(asthma = fct_recode(asthma, "Yes" = "1", "No" = "0"))
```

glimpse()

- glimpse (sinai_covid)

Your turn! Exercise 8

Using the pipe, create a new dataset from sinai_covid in which:

- The sex variable is named gender and the levels are woman/man
- deceased_indicator is a factor

Your turn! Exercise 9

- Observe the columns indicating health status, is there a way to make this dataset more tidy?

Final project

- Groups of 2-3
- 10 minutes presentation + 5 minutes for questions
- Import, tidy, transform, visualize, and model
- Judges