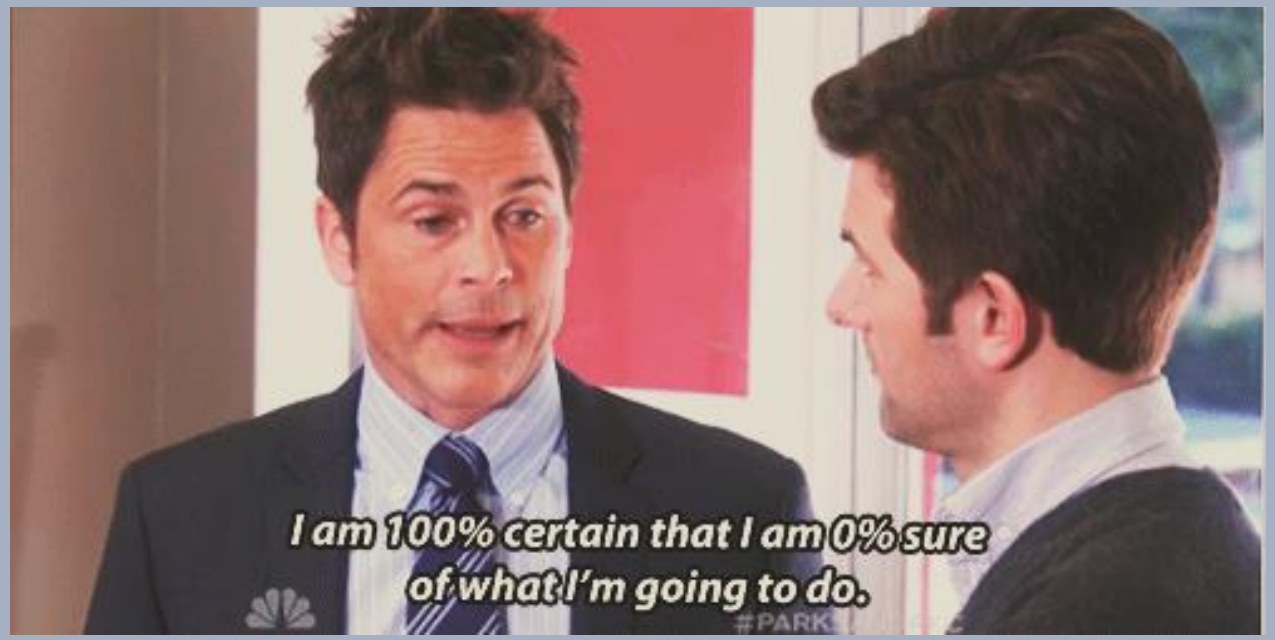
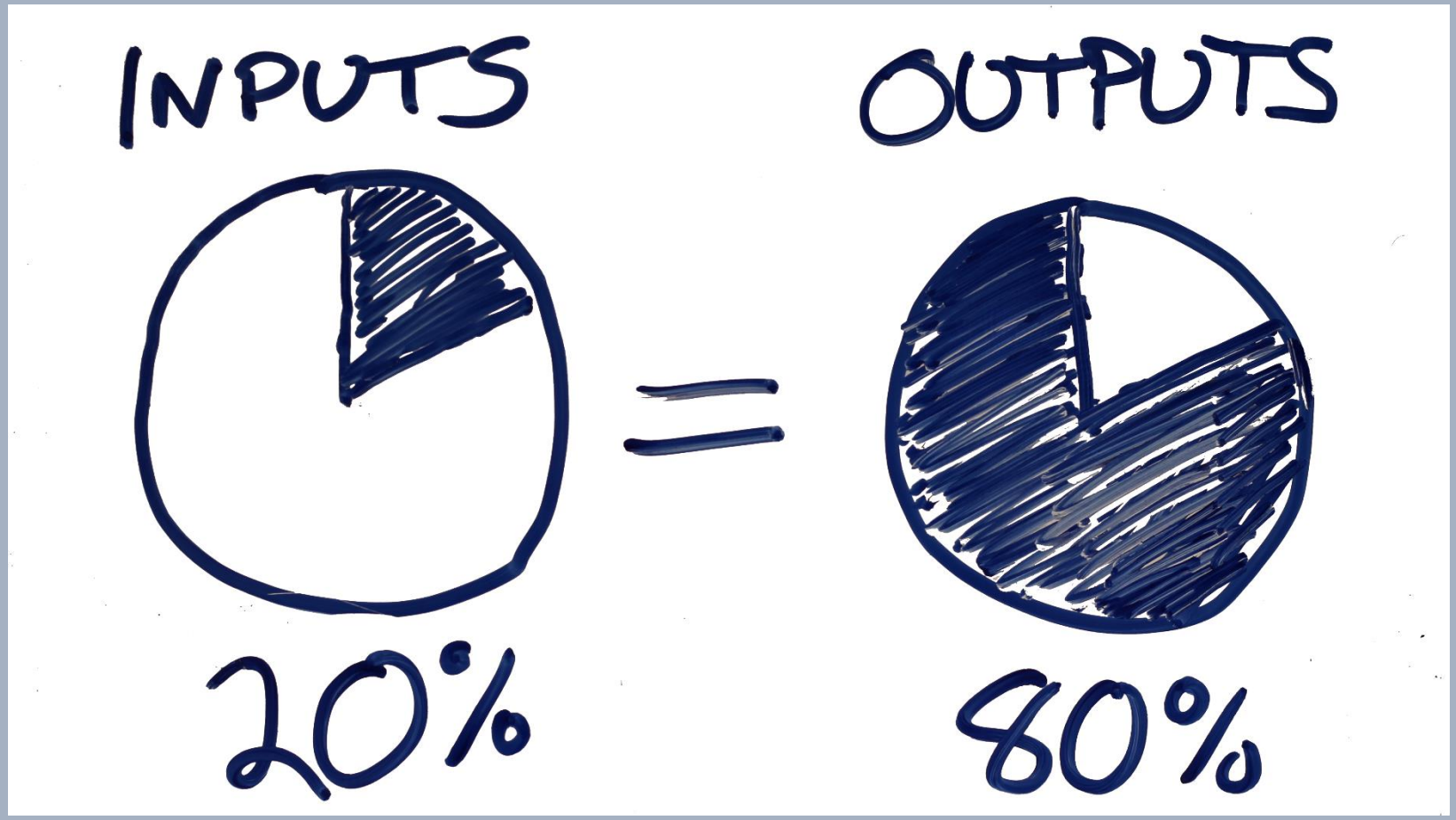
> R You Ready

DATA MANIPULATION WITH R


I am 100% certain that I am 0% sure
of what I'm going to do.

Pareto Principle



To take the mean within groups of a variable in a data frame

```r
library(dplyr)
group_by(airquality, Month) %>%
  summarise(o3 = mean(Ozone, na.rm = TRUE))
```

- Self-explanatory
- Easier to understand

```r
aggregate(airquality[, "Ozone"],
          list(Month = airquality[, "Month"]),
          mean, na.rm = TRUE)
```

- What are the square brackets [ ] for?
- What is list()?
- Why is the mean() function just sitting there all by itself?
- Is na.rm = TRUE an argument to aggregate()?

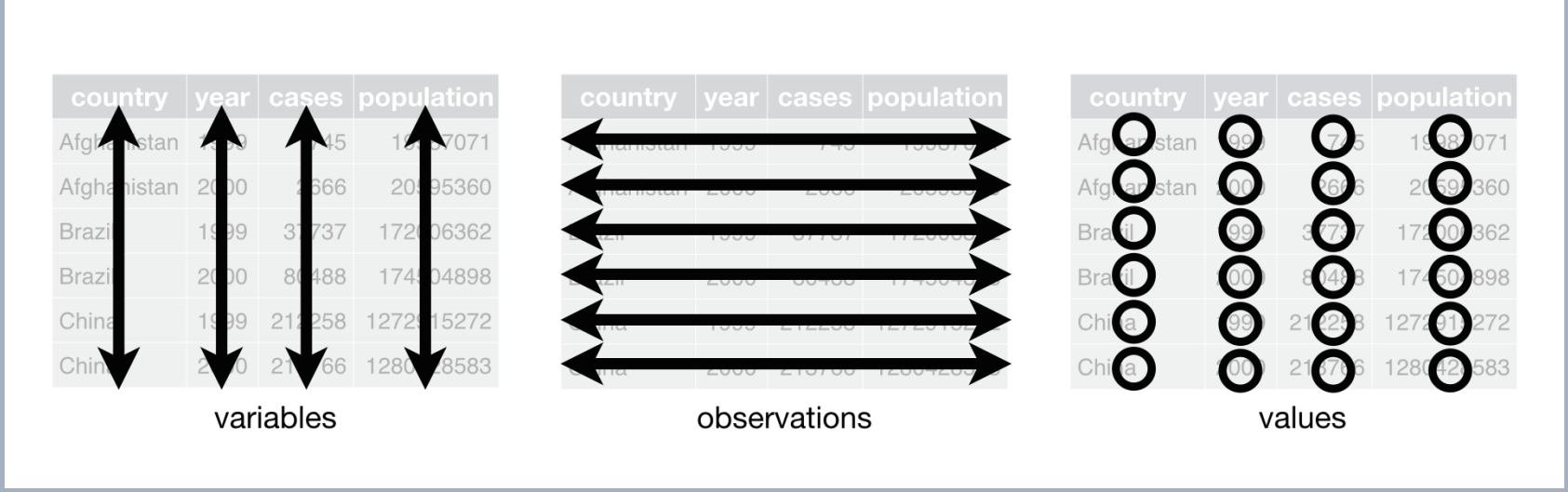
Tidyverse

Collection of R packages designed to work together.
All packages share an underlying design philosophy, grammar, and data structures.
An easier way to code!

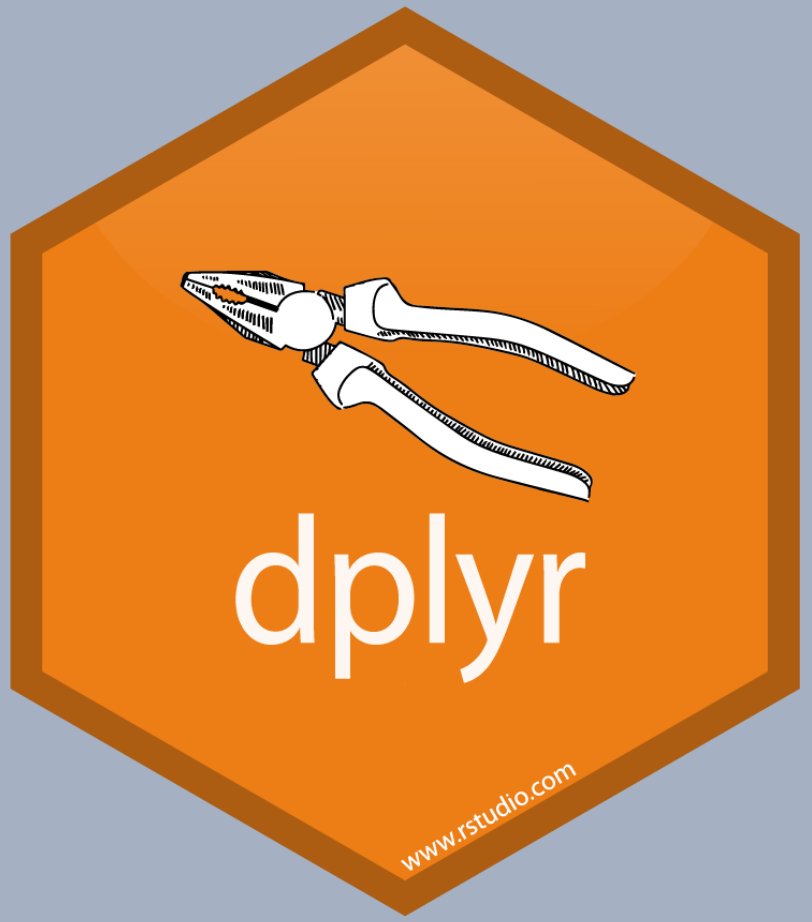
- **dplyr:** Grammar of data manipulation
- **ggplot2:** System for declaratively creating graphics, based on The Grammar of Graphics
- **tibble:** Modern reimagining of the data.frame, keeping what time has proven to be effective, and throwing out what is not
- **readr:** Fast and friendly way to read rectangular data (like csv, tsv, and fwf)
- **tidyr:** Tools to help you create tidy data where each variable is in a column, each observation is a row, and rach value is a cell
- **purrr:** Functional programming toolkit
- Learn more at *tidyverse.org/*

Tidy Data

- Variables must be in columns
- Observations must be in rows
- Values must be in cells



What is dplyr?



Grammar of data manipulation

+ mutate() to create new variables from existing ones
+ select() picks variables on their names
+ filter() allows pointed selection based on given criteria
+ summarise() reduces multiple values down to a single summary
+ arrange() changes the ordering of rows
+ group_by() performs any of the above on a group-by-group basis

dplyr syntax:

All calls to dplyr verbs follow the same format:

1. The first argument is a dataframe.
2. The subsequent arguments describe what to do to that dataframe using unquoted variable names.

Each call returns a new dataframe rather than overwriting the old one.

%>% allows you to pipe the output from one function to the input of another function to reduce redundancy.
Instead of nesting functions (reading from the inside to the outside), the idea of of piping is to read the functions from left to right.

%>%

Before

```
summarise(
  group_by((filter
            (team, name == "Anagha")
            ),
            year
            ),
  total = sum(n))
```

After

```
team %>%
  filter(name == "Anagha") %>%
  group_by(year) %>%
  summarise(total = sum(n))
```


I'm ready.

Further Resources

- Tidyverse Cookbook - https://rstudio-education.github.io/tidyverse-cookbook/
- Introduction to dplyr - https://dplyr.tidyverse.org/articles/dplyr.html
- Introduction to the Tidyverse - https://www.datacamp.com/courses/introduction-to-the-tidyverse

THANK YOU