Data wrangling 2

1/14/2026

Quick recap

Quickly copying code

```
1 ggplot(mapping = aes(x = color, y = price), data = df.diamonds) +
2   stat_summary(fun.y = "mean", geom = "point")
```

put cursor anywhere in line 1

cmd + shift + d

```
1 ggplot(mapping = aes(x = color, y = price), data = df.diamonds) +
2 ggplot(mapping = aes(x = color, y = price), data = df.diamonds) +
3   stat_summary(fun.y = "mean", geom = "point")
```

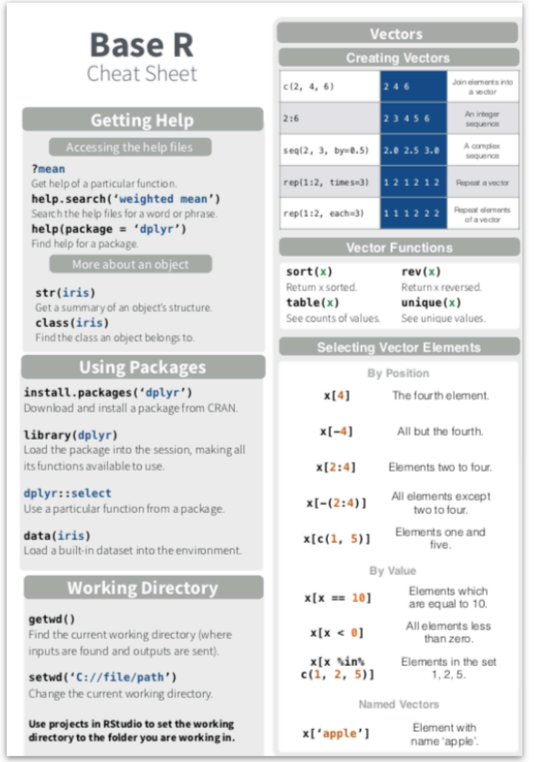
comment

cmd + shift + c

```
1 # ggplot(mapping = aes(x = color, y = price), data = df.diamonds) +
2 ggplot(mapping = aes(x = cut, y = price), data = df.diamonds) +
3   stat_summary(fun.y = "mean", geom = "point")
```

change

Two styles of coding in R



4.3.3 Operators

Table 4.3 shows the comparison operators that result in logical outputs.

Table 4.3: Table of comparison operators that result in boolean (TRUE/FALSE) outputs.

symbol	name
`==`	equal to
`!=`	not equal to
`>`, `<`	greater/less than
`>=`, `<=`	greater/less than or equal
`&`, `|`, `!`	logical operators: and, or, not
`%in%`	checks whether an element is in an object

4.3.4 Control flow

4.3.4.1 if-then

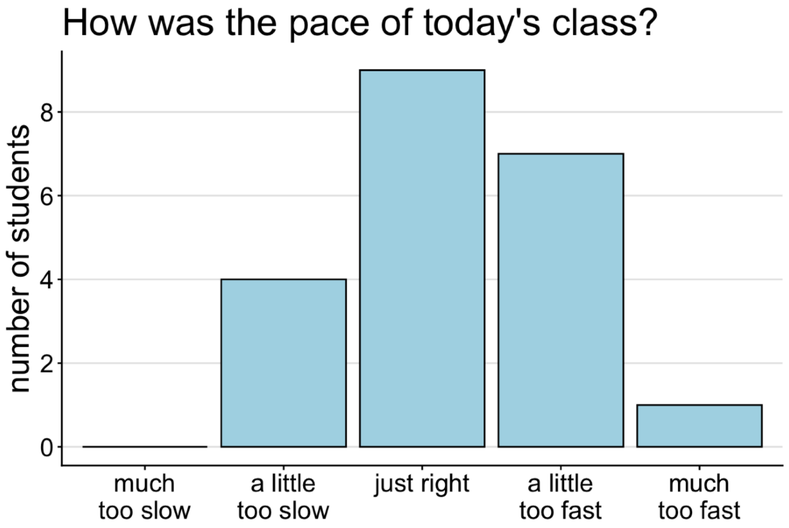
```
number = 3

if (number == 1) {
  print("The number is 1.")
} else if (number == 2) {
  print("The number is 2.")
} else {
  print("The number is neither 1 nor 2.")
}
```

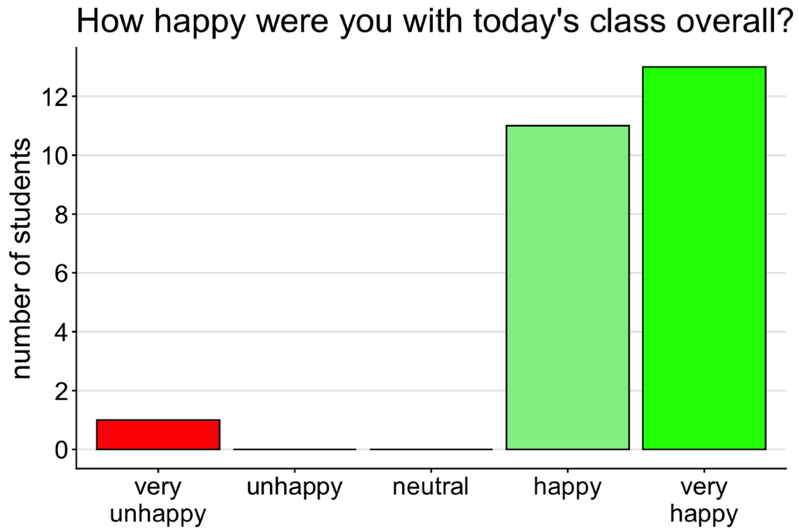

```
eat(
  slice(
    bake(
      put(
        pour(
          mix(ingredients),
          into = baking_form),
        into = oven),
      time = 30),
    pieces = 6),
  1)
```

```
ingredients %>%
  mix %>%
  pour(into = baking_form) %>%
  put(into = oven) %>%
  bake(time = 30) %>%
  slice(pieces = 6) %>%
  eat(1)
```

Your feedback

Your feedback





Really helpful! Could you please **remind us what the shortcuts are as you use them**? I'm grateful to learn them but need reminders.

I appreciate having the instructors code the examples and giving us space to try things on our own. I think **group coding might also be helpful** for examples that are a little more complicated. I am roughly familiar with R, but I think if I had less familiarity the groundwork information would have felt very fast because with my base it felt a little fast

What would be helpful is after we go through the solution for the practice, **leaving it up for a while to compare where I went wrong**.

Data wrangling time ...

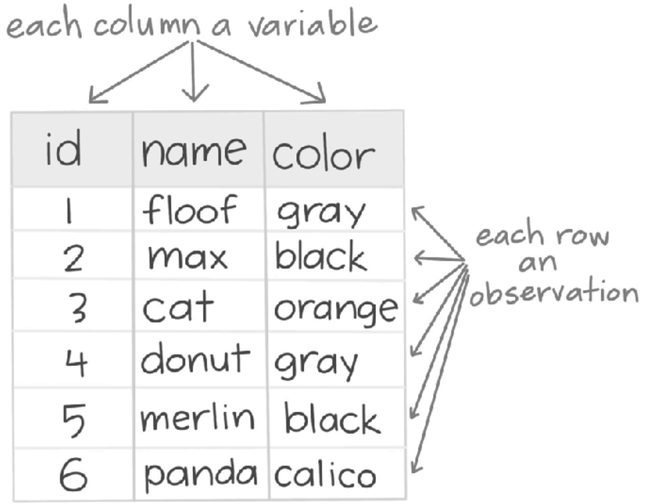
Tidy data

"TIDY DATA is a standard way of mapping the meaning of a dataset to its structure."

–HADLEY WICKHAM

In tidy data:

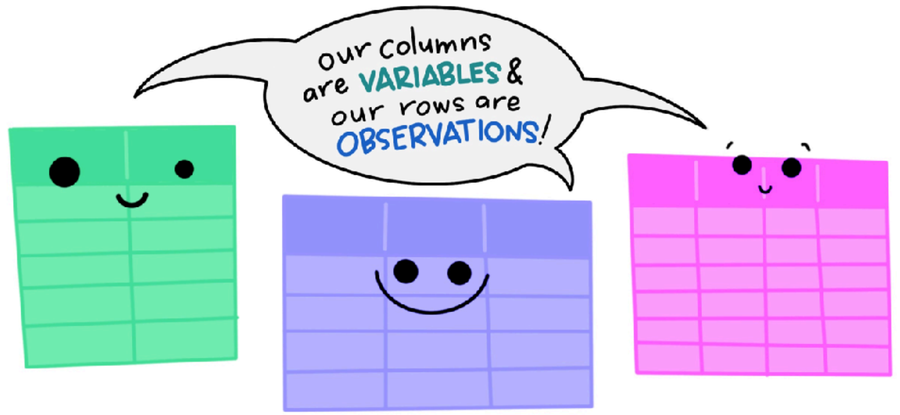
- each variable forms a column
- each observation forms a row
- each cell is a single measurement



Wickham, H. (2014). Tidy Data. Journal of Statistical Software 59 (10). DOI: 10.18637/jss.v059.i10

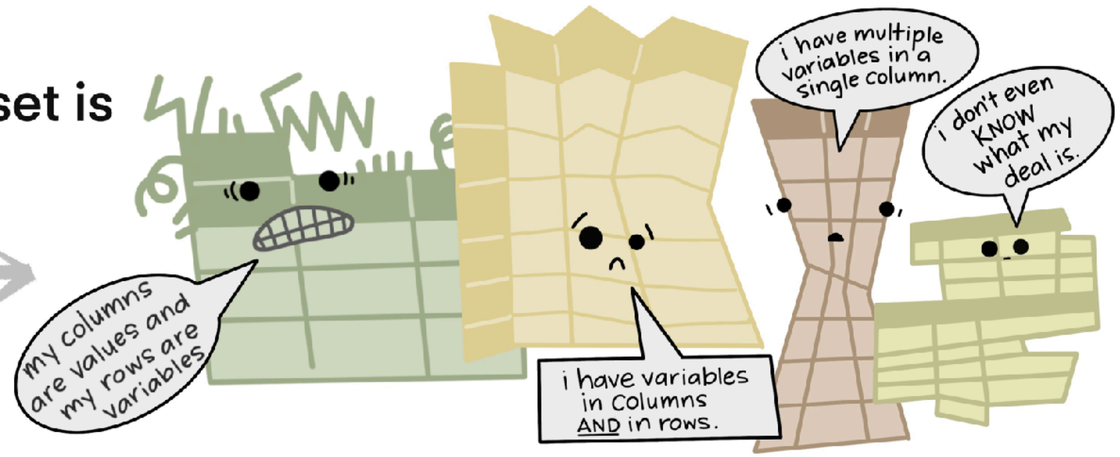
Tidy data

The standard structure of tidy data means that "tidy datasets are all alike..."



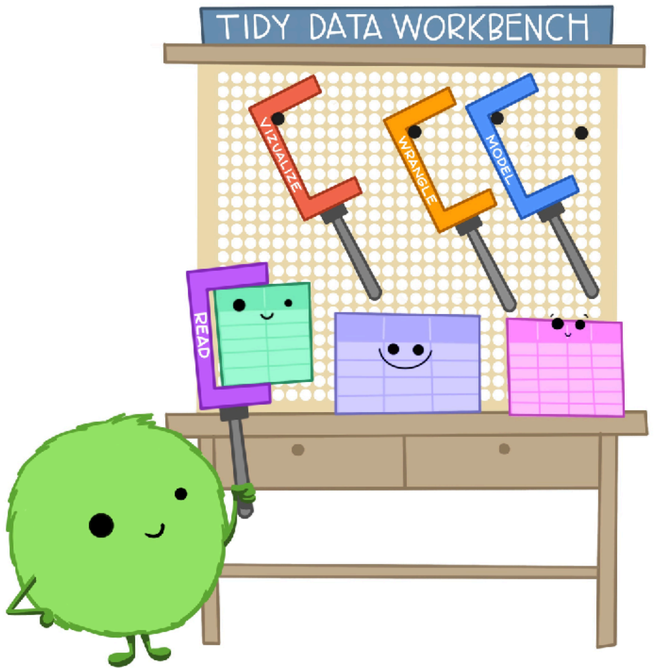
"...but every messy dataset is messy in its own way."

–HADLEY WICKHAM

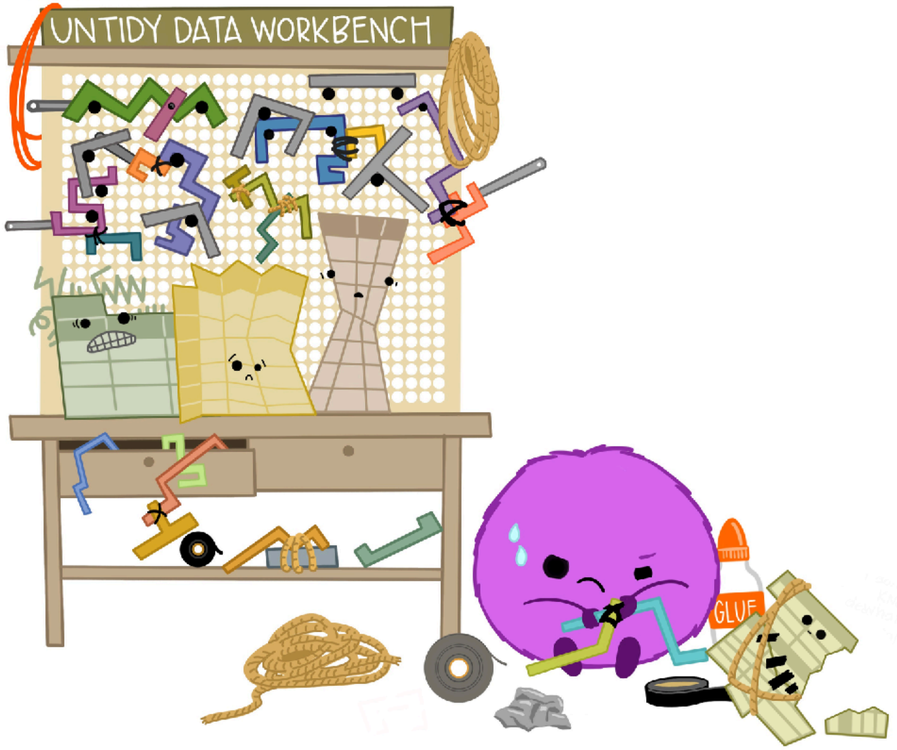


Tidy data

When working with tidy data, we can use the same tools in similar ways for different datasets...



...but working with untidy data often means reinventing the wheel with one-time approaches that are hard to iterate or reuse.



Using functions with `tidyverse` verbs

```
1 df.starwars %>%
2   select(where(fn = is.numeric))
```

my recommendation

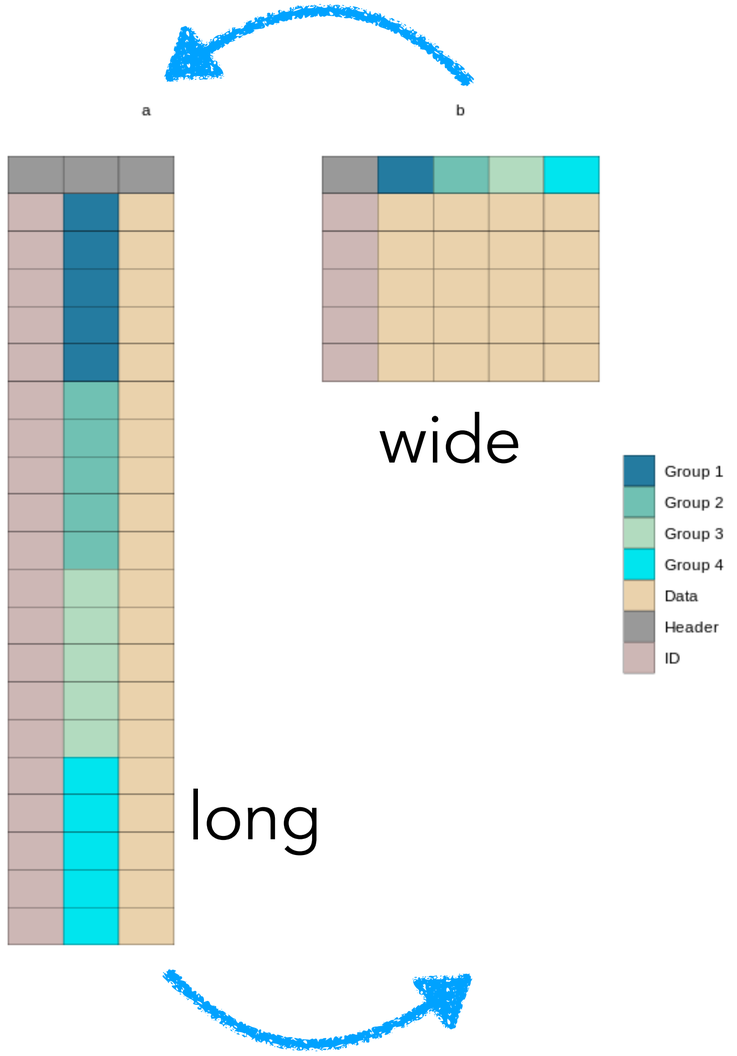
- `fn = is.numeric`
- `fn = "is.numeric"`
- `fn = function(x){is.numeric(x)}`
- `fn = ~ is.numeric(.)`

flexible, short, works well with other verbs we'll learn about later

- `fn = ~ !is.numeric(.)`

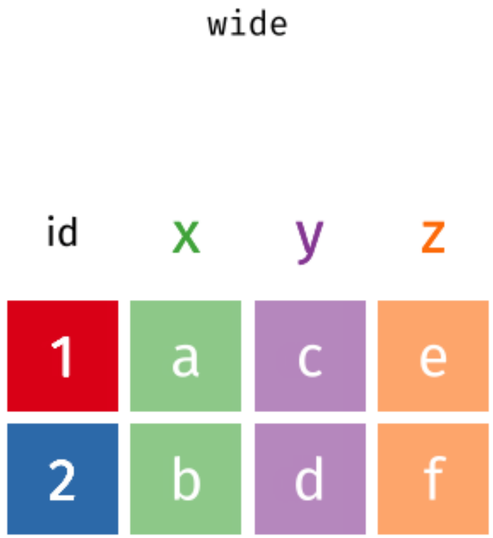
select all variables that are not numeric

pivot_longer()



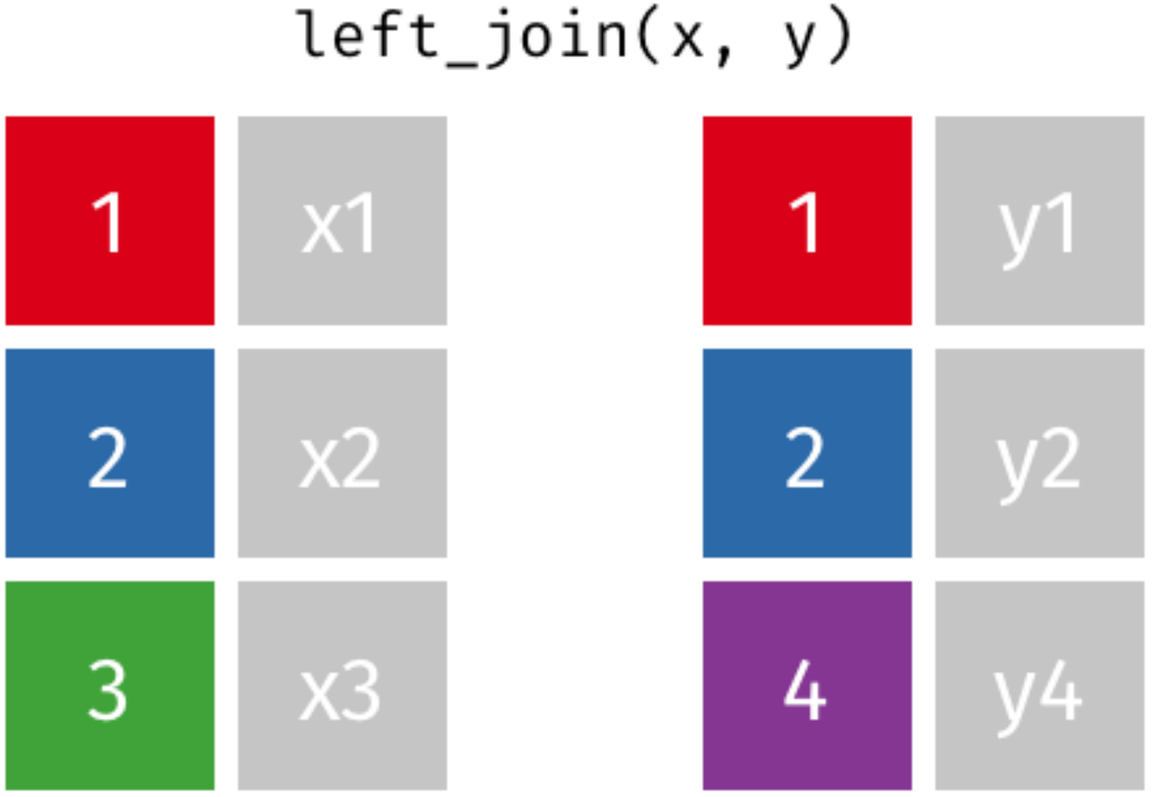
pivot_wider()

pivot_longer() & pivot_wider()



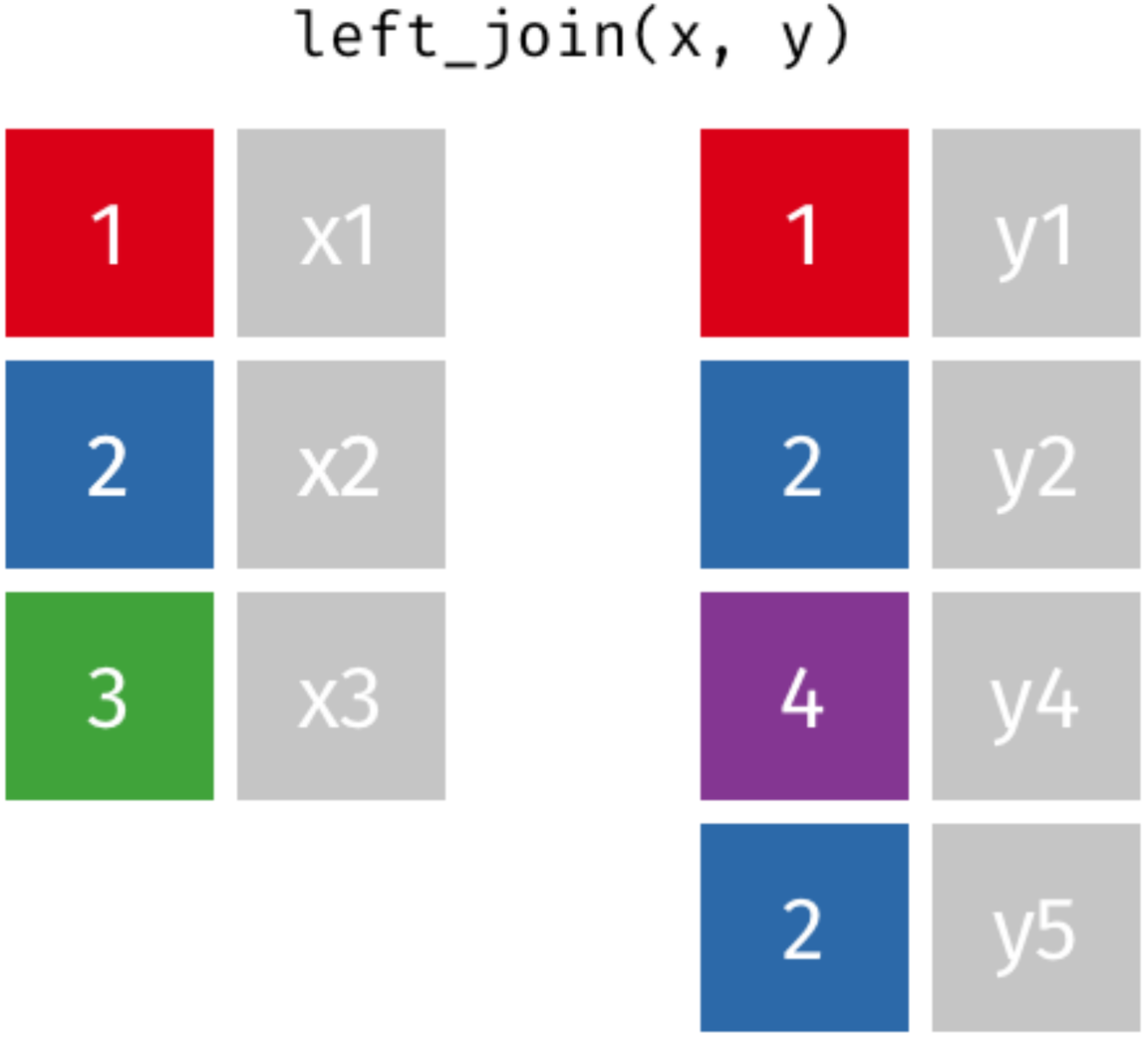
https://github.com/gadenbuie/tidyexplain

left_join()



https://github.com/gadenbuie/tidyexplain

left_join()



https://github.com/gadenbuie/tidyexplain

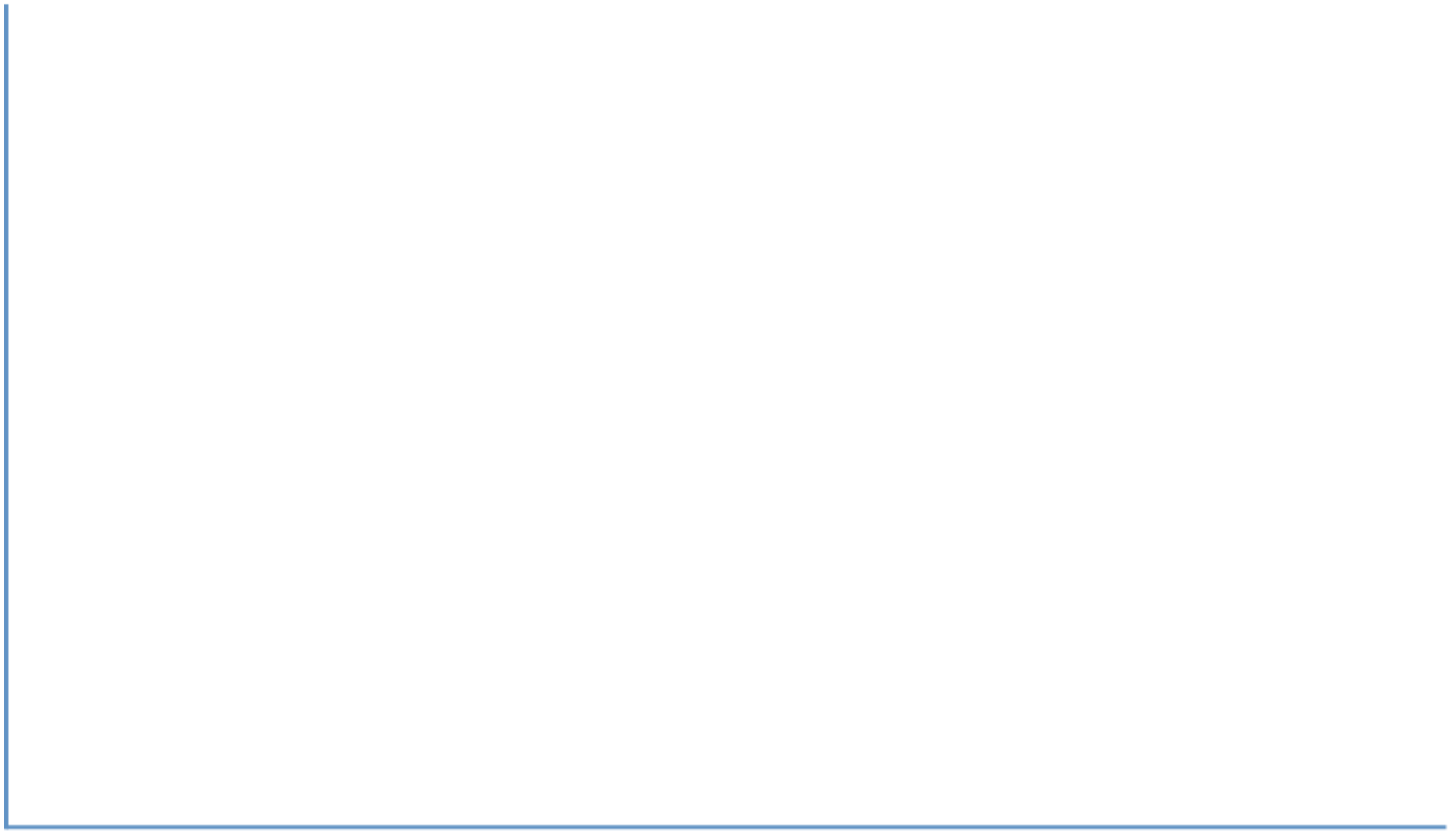
Timer

blue

pink

Feedback

How was the pace of today's class?



How happy were you with today's class overall?

What did you like about today's class? What could be improved next time?

Thank you!