LING 0700 LAB: WEEK 3

TA: Wesley Mark Lincoln

COURSE ADMIN

REMINDERS

- Slides on `facet_wrap()` vs. `facet_grid()` available

LECTURE REVIEW

PIPING

```
# option 1
ratings %>% select(Word, Frequency) %>% glimpse()
```

```
select(rating, Word, Frequency) %>% glimpse()
```

```
glimpse(select(rating, Word, Frequency))
```

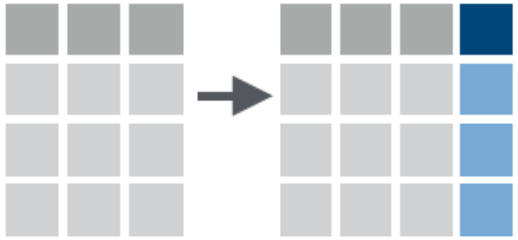
WHY?

dplyr::select()

select(.data, …**)** Extract columns as a table.

mtcars |> select(mpg, wt)

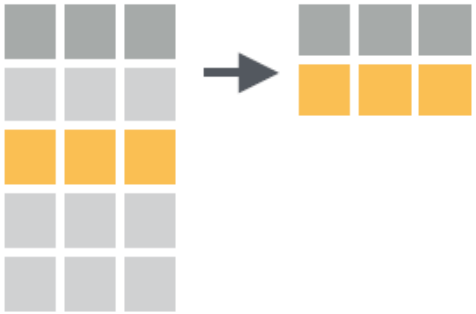
dplyr::mutate()



mutate(.data, …, .keep = "all", .before = NULL, .after = NULL**)** Compute new column(s). Also **add_column()**.

mtcars |> mutate(gpm = 1 / mpg)

dplyr::filter()



filter(.data, …, .preserve = FALSE**)** Extract rows that meet logical criteria.

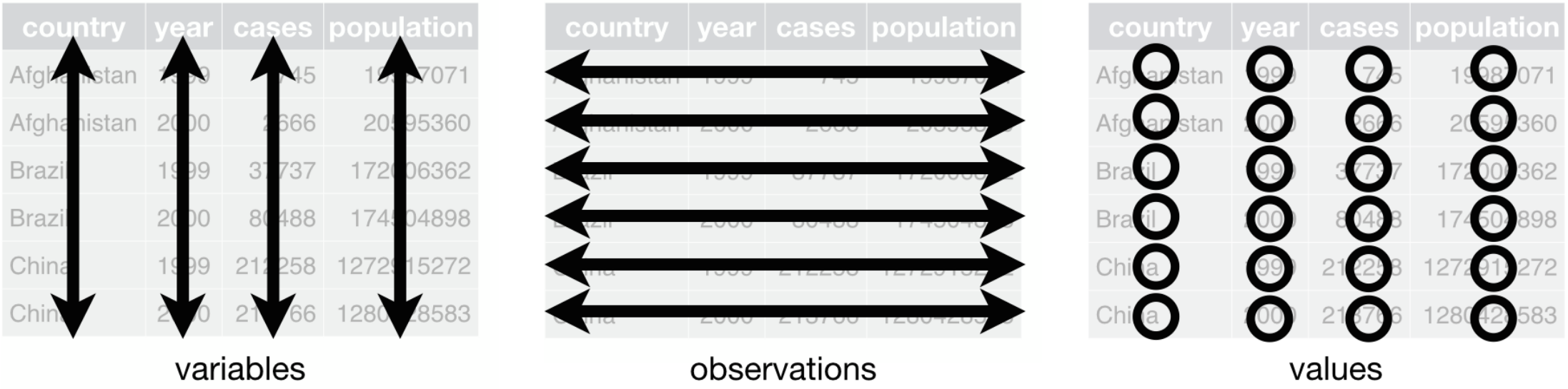
```
mtcars |> filter(mpg > 20)
```

dplyr::select()

rename(.data, …**)** Rename columns. Use **rename_with()** to rename with a function.

mtcars |> rename(miles_per_gallon = mpg)

WHAT IS TIDY DATA?



WHAT IS TIDY DATA?

These examples use tuberculosis data available in base R. Access the datasets by simply calling `table1`, `table2`, etc.

This dataset (`table1`) is **not tidy** because columns 2 and 3 do not correspond to variables; 1999 and 2000 are values of a variable `year`.

A tibble: 3 × 3

country	1999	2000
<chr>	<dbl>	<dbl>
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

WHAT IS TIDY DATA?

A way to distinguish values from variables:

- variables can be thought of as questions
- values are (possible) answers to those questions

A tibble: 3 × 3

country	1999	2000
<chr>	<dbl>	<dbl>
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

WHAT IS TIDY DATA?

Q: Country?
- A: Afghanistan
- A: Brazil
- A: China

Q: Year?
- A: 1999
- A: 2000

Q: Cases?
- A: 745
- A: 37737
- … etc.

A tibble: 3 × 3

country	1999	2000
<chr>	<dbl>	<dbl>
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

Year is a variable and ought to have its own column.
All the numerical values (cases) ought to be in **one** column.

WHAT IS TIDY DATA?

This dataset (`table2`) is not tidy because `cases` and `population` are variables and ought to have a column each.

The contents of the pink rectangle constitute one observation and should occupy one row, not two.

A tibble: 12 × 4

country	year	type	count
<chr>	<dbl>	<chr>	<dbl>
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272

WHAT IS TIDY DATA?

We could make the case that this data *is* tidy, and that `type` is a variable with values `cases` and `population`.

But this treatment is an unlikely one; it is more probably that `cases` and `population` are variables which will be used to calculate rate (`cases/population`).

A tibble: 12 × 4

country	year	type	count
<chr>	<dbl>	<chr>	<dbl>
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272

WHAT IS TIDY DATA?

This data **is tidy**. Each column is a variable and vice versa.

Each country name occurs twice, but notice that each occurrence is a single observation: the researchers measured `cases` and `population` once in 1999 and once in 2000. Thus, each row **is** an observation and vice versa.

A tibble: 6 × 4

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

LAB EXERCISES

LAB EXERCISE Q2.2

2. Suppose we attempt to import the csv file given above with the code below. What will be the result?

```
data <- read_csv("https://pos.it/r4ds-students-csv",
    col_types = list(AGE = col_double())
)
```

- ○ imports with no errors or warnings
- ○ fails to import, throws error
- ◉ ✅ imports, but with a warning that there are parsing issues
- ○ imports, but changes the column name to `age`

LAB EXERCISE Q2.2

2. Suppose we attempt to import the csv file given above with the code below. What will be the result?

```
data <- read_csv("https://pos.it/r4ds-students-csv",
    col_types = list(AGE = col_double())
)
```

Here, we are reading a csv file in from a given URL…

… and specificying that the AGE column contains doubles

- ○ imports with no errors or warnings
- ○ fails to import, throws error
- ◉ ✅ imports, but with a warning that there are parsing issues
- ○ imports, but changes the column name to `age`

LAB EXERCISE Q2.2

```
[28] data <- read_csv("https://pos.it/r4ds-students-csv",
         col_types = list(AGE = col_double())
     )
     data
```

The warning message helpfully includes a way to find out what went wrong.

```
Warning message:
"One or more parsing issues, call `problems()` on your data frame for details,
e.g.:
  dat <- vroom(...)
  problems(dat)"
```

A spec_tbl_df: 6 × 5

Student ID	Full Name	favourite.food	mealPlan	AGE
<dbl>	<chr>	<chr>	<chr>	<dbl>
1	Sunil Huffmann	Strawberry yoghurt	Lunch only	4
2	Barclay Lynn	French fries	Lunch only	5
3	Jayendra Lyne	N/A	Breakfast and lunch	7
4	Leon Rossini	Anchovies	Lunch only	NA
5	Chidiegwu Dunkel	Pizza	Breakfast and lunch	NA
6	Güvenç Attila	Ice cream	Lunch only	6

LAB EXERCISE Q2.2

```
[ ] problems(data)
```

A tibble: 1 × 5

row	col	expected	actual	file
<int>	<int>	<chr>	<chr>	<chr>
6	5	a double	five	

problems() shows that there was an issue in `row 6, col 5`: we told R to expect a double, but it found the string "`five`", so it replaced that with `NA` (pink box).

LAB EXERCISE Q2.2

```
Student ID,Full Name,favourite.food,mealPlan,AGE
1,Sunil Huffmann,Strawberry yoghurt,Lunch only,4
2,Barclay Lynn,French fries,Lunch only,5
3,Jayendra Lyne,N/A,Breakfast and lunch,7
4,Leon Rossini,Anchovies,Lunch only,
5,Chidiegwu Dunkel,Pizza,Breakfast and lunch,five
6,Güvenç Attila,Ice cream,Lunch only,6
```

Student ID	Full Name	favourite.food	mealPlan	AGE
<dbl>	<chr>	<chr>	<chr>	<dbl>
1	Sunil Huffmann	Strawberry yoghurt	Lunch only	4
2	Barclay Lynn	French fries	Lunch only	5
3	Jayendra Lyne	N/A	Breakfast and lunch	7
4	Leon Rossini	Anchovies	Lunch only	NA
5	Chidiegwu Dunkel	Pizza	Breakfast and lunch	NA
6	Güvenç Attila	Ice cream	Lunch only	6

Visually, here's what happened.

The value "five" in the original .csv file (left) was replaced with NA in the tibble created by R (right).

LAB EXERCISE Q2.2

Notice that Jayendra's favourite food was N/A in the .csv file already (`row 3, col 3`).

```
Student ID,Full Name,favourite.food,mealPlan,AGE
1,Sunil Huffmann,Strawberry yoghurt,Lunch only,4
2,Barclay Lynn,French fries,Lunch only,5
3,Jayendra Lyne,N/A,Breakfast and lunch,7
4,Leon Rossini,Anchovies,Lunch only,
5,Chidiegwu Dunkel,Pizza,Breakfast and lunch,five
6,Güvenç Attila,Ice cream,Lunch only,6
```

Student ID	Full Name	favourite.food	mealPlan	AGE
<dbl>	<chr>	<chr>	<chr>	<dbl>
1	Sunil Huffmann	Strawberry yoghurt	Lunch only	4
2	Barclay Lynn	French fries	Lunch only	5
3	Jayendra Lyne	N/A	Breakfast and lunch	7
4	Leon Rossini	Anchovies	Lunch only	NA
5	Chidiegwu Dunkel	Pizza	Breakfast and lunch	NA
6	Güvenç Attila	Ice cream	Lunch only	6

LAB EXERCISE Q2.3

3. Suppose we import the dataset given above and name it `data`. What will `is.na(data[3,3])` return?

- ○ True
- ◉ ✅ False

When we run `is.na()` on `row 3`, `col 3`, however, the output is `FALSE` – because "`N/A`" is a string, distinct from `NA`, a special entity of type logical. The function `is.na()` specifically tests for this logical `NA`.