dplyr

Tidying and manipulating data

Download the section 5 .Rmd handout to `STAT240/lecture/sect05-dplyr`.

Download three data files to `STAT240/data`.

- `grocery-list.csv`
- `grocery-prices.csv`
- `madison-weather-1869-2023.csv`

Material in this section is covered by Chapters 7-8 on the notes website.

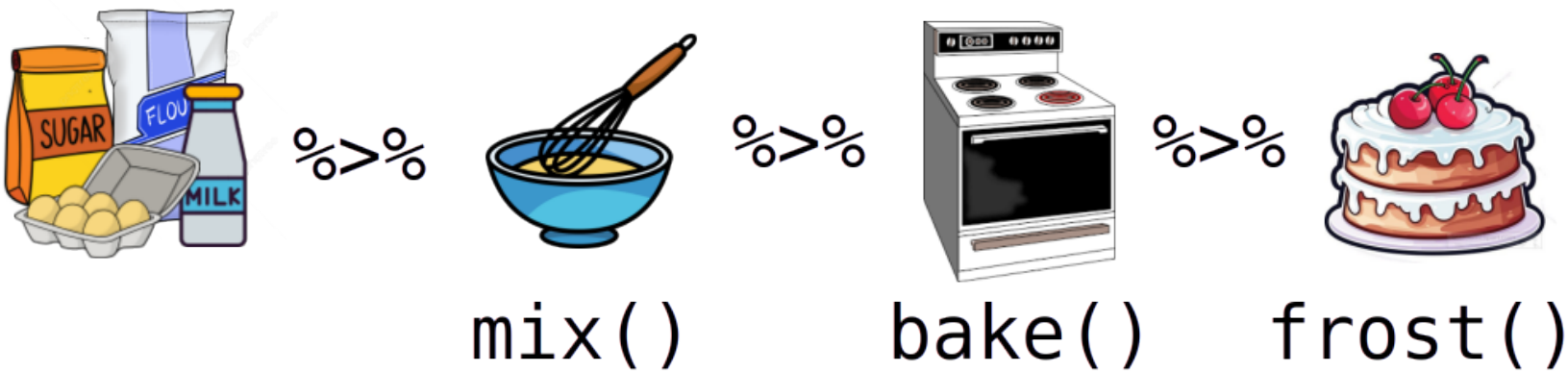
`dplyr` is a collection of command for tidying data.

- Specific "grammar" just like `ggplot`.

Here is a cheatsheet of `dplyr` commands.

The **pipe** operator %>% takes an object and passes it into a command.

- Works like the word "then"
- Useful for a sequence of functions



Basic dplyr command:

- Input a df
- Alter the df
- Output the new df

Chain commands together with the pipe.

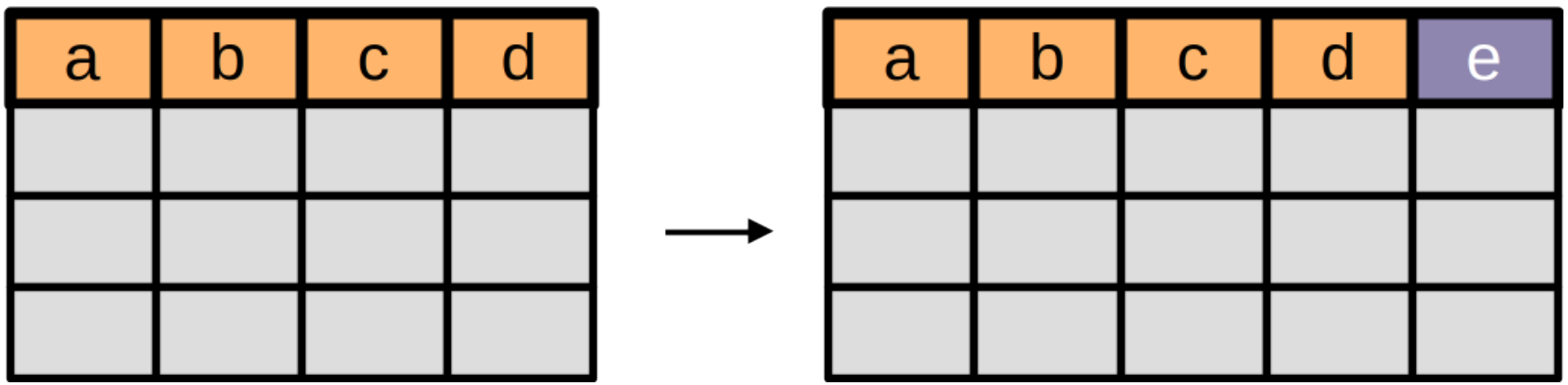
Column commands act on the columns and leave rows unchanged.

- `mutate()`, `select()`, `relocate()`, `rename()`

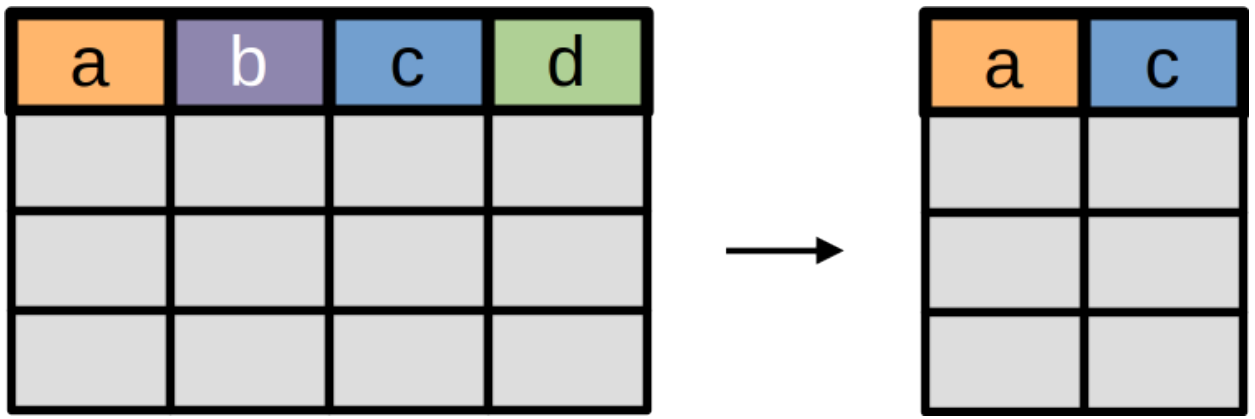
Rows commands act on the rows and leave columns unchanged.

- `arrange()`, `filter()`, `drop_na()`

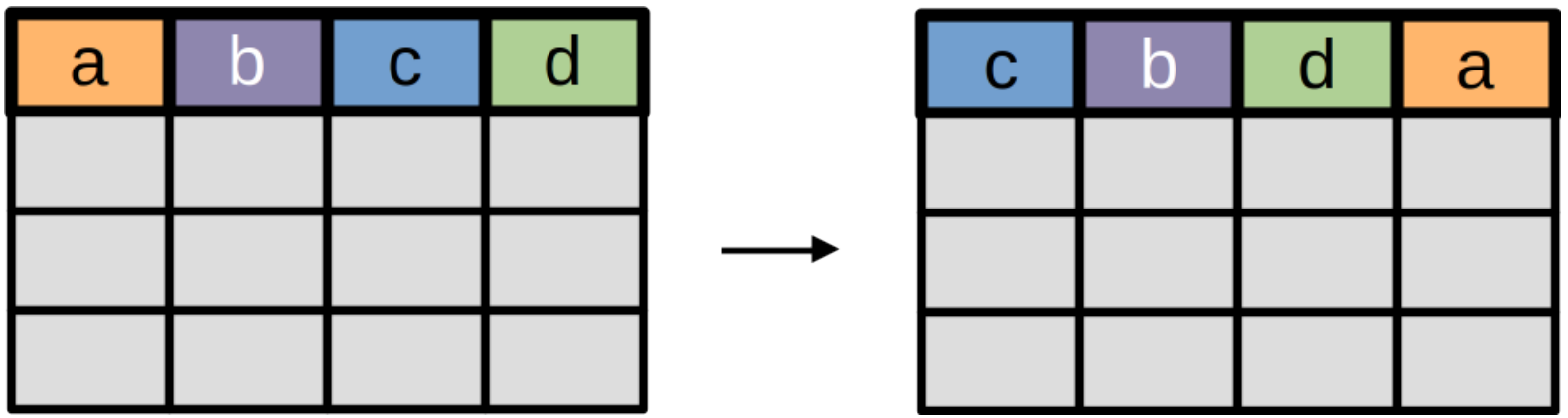
`mutate()` creates a new column.



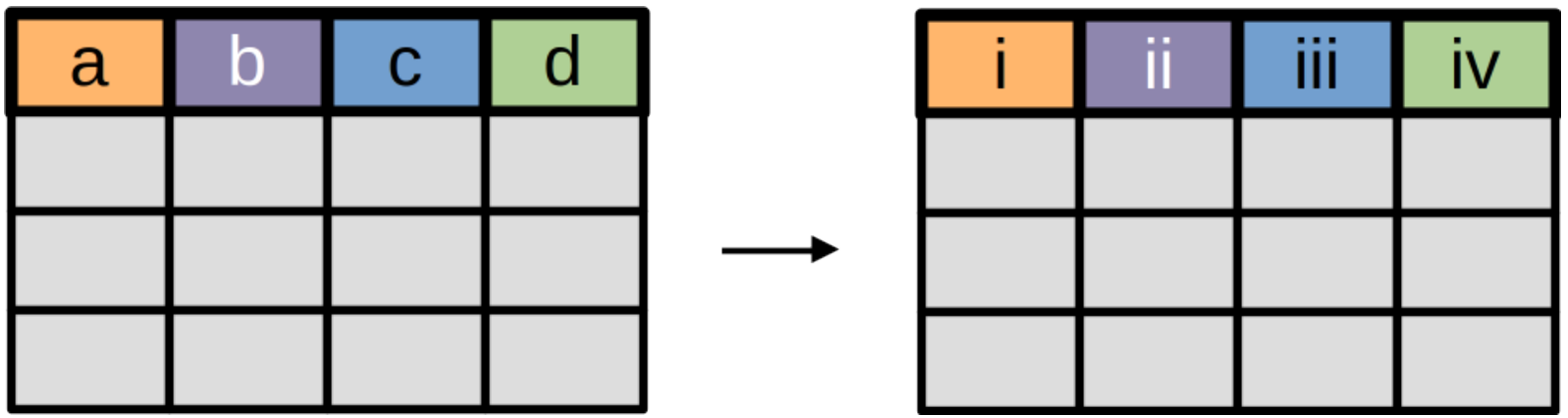
`select()` reduces to specific columns.



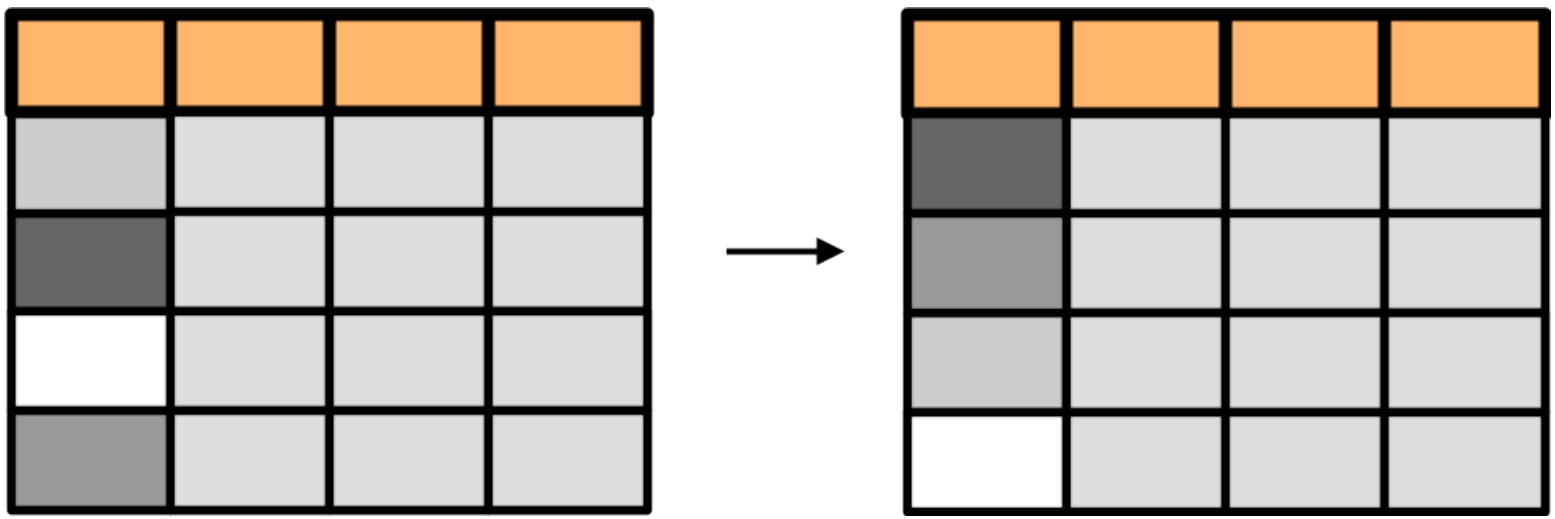
`relocate()` moves columns.



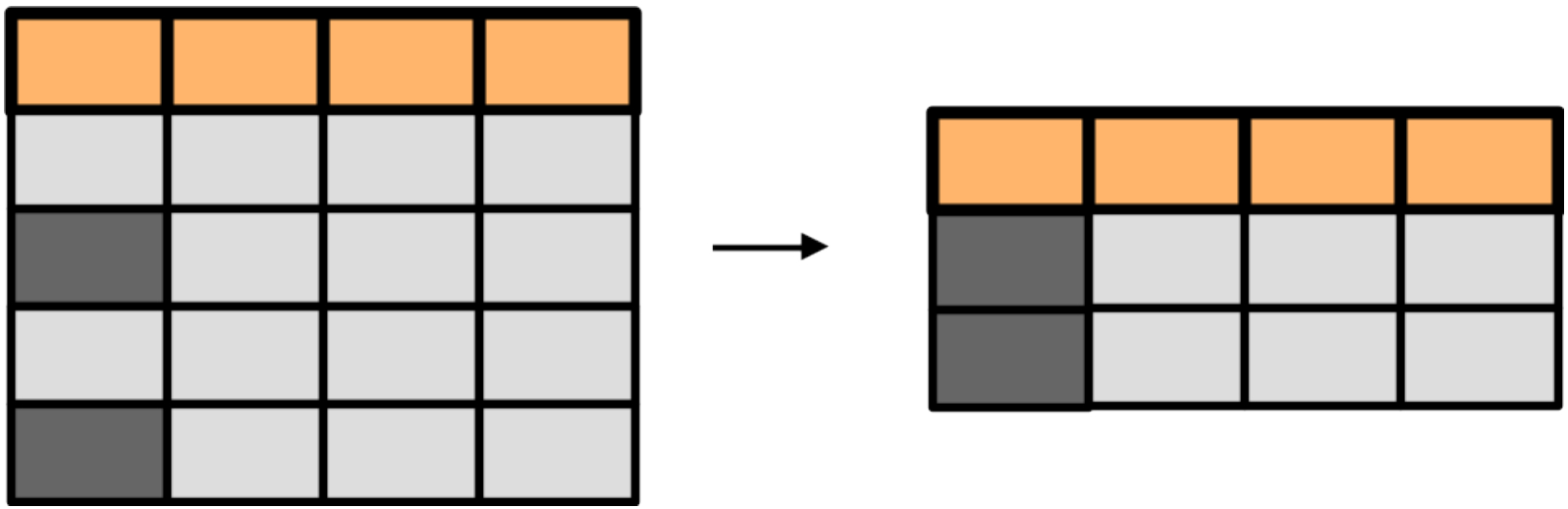
`rename()` renames columns.



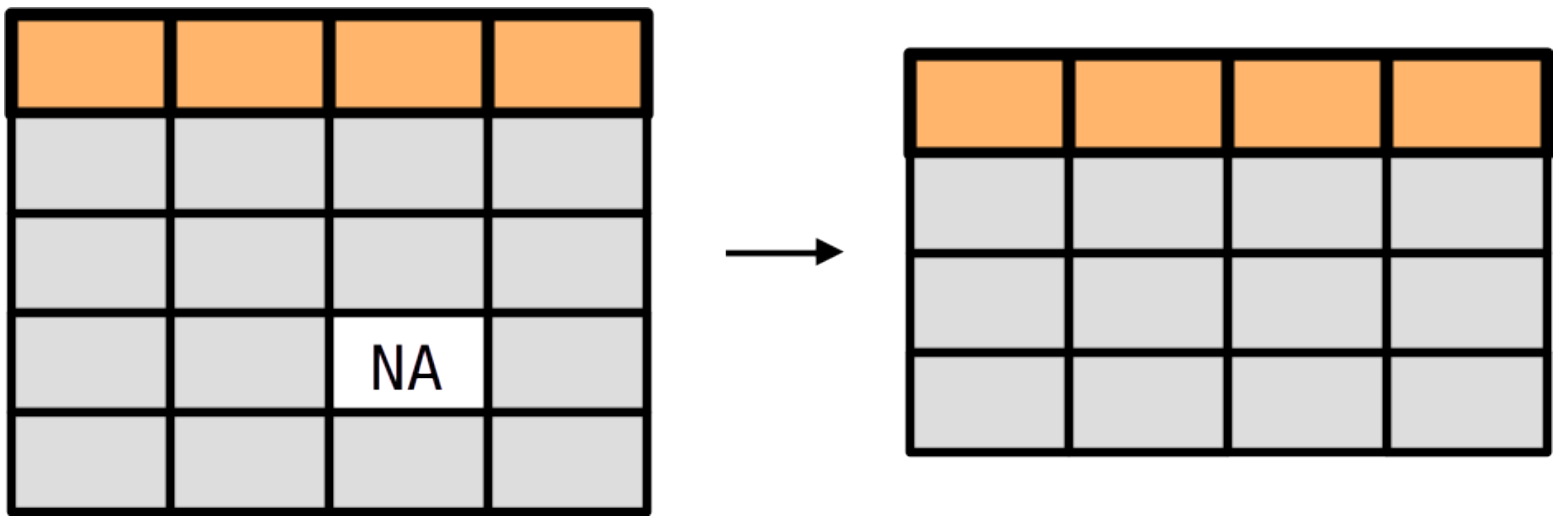
`arrange()` sorts rows.



`filter()` reduces to specific rows.



`drop_na()` removes rows with missing values.



An important skill is to translate ordinary language into `dplyr` verbs.

Write your own code to respond to five requests.

`summarize()` reduces a column to a single value.

- For example: `sum()`, `mean()`

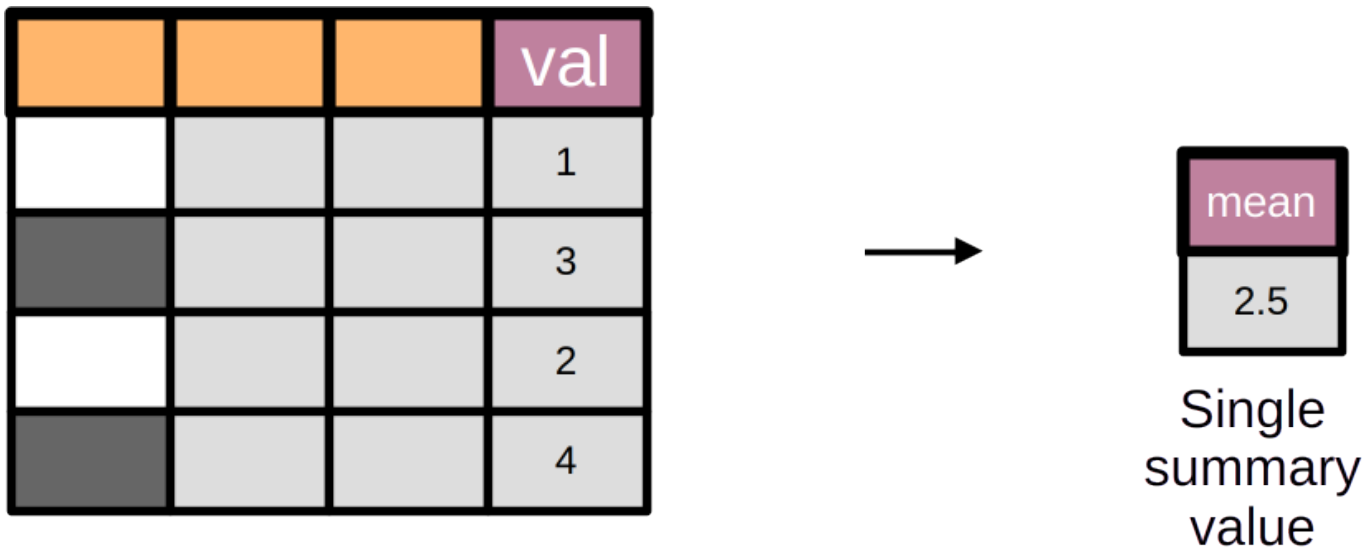
Often combined with `group_by()`.

`group_by()` sets the "grouping" property of the df.

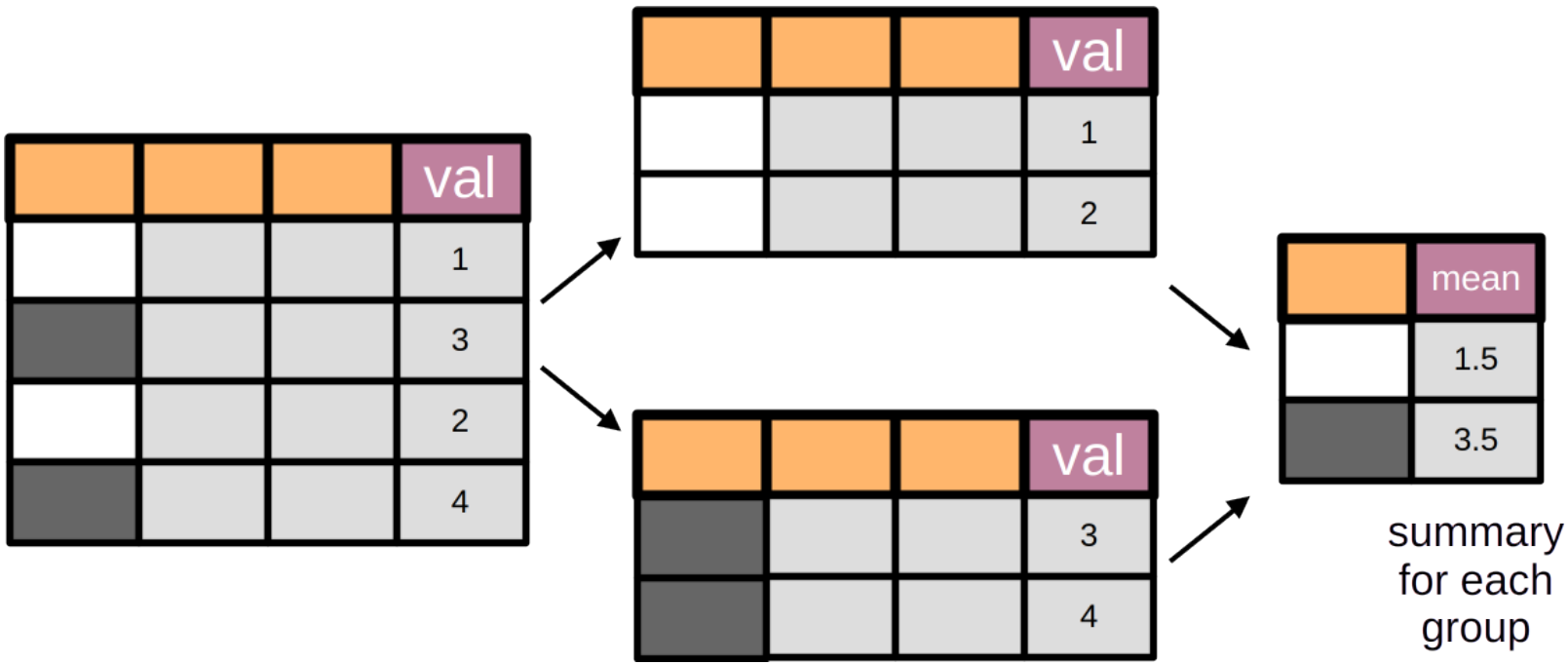
- Gives instructions for future commands
- Changes "level" of calculation

Let's find the average price of fruits and vegetables.

`summarize()`:



`group_by() %>% summarize()`:



Other functions to use with or without `group_by()`:

- `slice_min()` for the lowest values
- `slice_max()` for the highest values
- `n()` for the number of rows

Can use `count()` as a shortcut for `n()`.

`group_by()` has useful applications to `mutate()`.

- Resulting column value depends on group.

Which fruits and vegetables make up the highest % cost?

Remove the grouping instruction with `ungroup`.

We can perform a grouping operation at first, then go back to looking at the entire df at once.

Let's see how it works.

- Run each code chunk as-is
- Predict what happens if you include `group_by`
- Uncomment the line and see what happens

We can create a new column with `mutate()`, such as the product of two other columns.

We can create a column based on a condition with `case_when`.