"Data verbs" cheatsheet

Gavin Band gavin.band@well.ox.ac.uk 2025

1. Keep your data in a **data frame**

`data =`

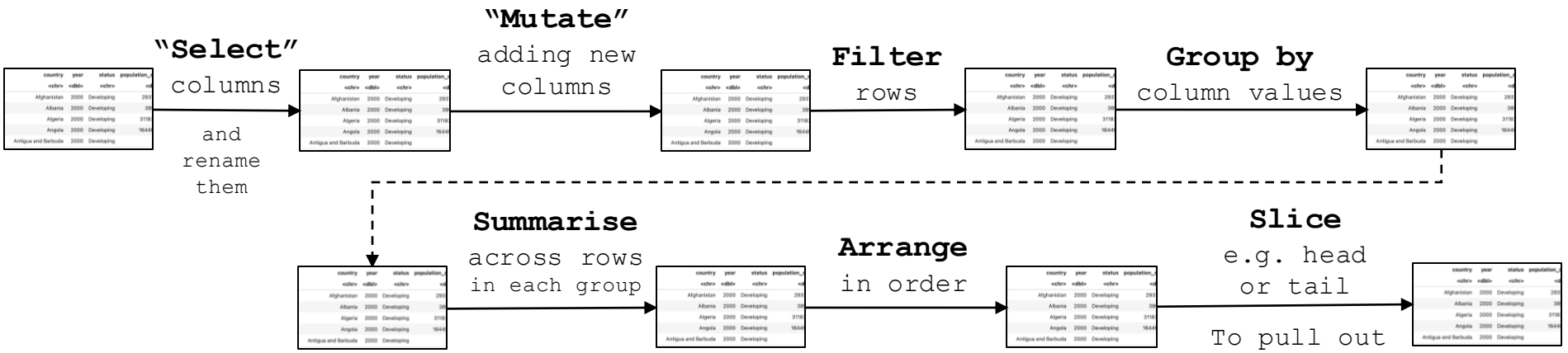
country	year	status	population_size	life_expectancy	adult_mortality	under_five_deaths
<chr>	<dbl>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>
Afghanistan	2000	Developing	293756	54.8	321	122
Albania	2000	Developing	38927	72.6	11	1
Algeria	2000	Developing	3118366	71.3	145	25
Angola	2000	Developing	1644924	45.3	48	160
Antigua and Barbuda	2000	Developing	NA	73.6	156	0
Argentina	2000	Developing	3757452	74.1	137	14
Armenia	2000	Developing	369588	72.0	142	1
Australia	2000	Developed	19153	79.5	78	2

Each row is a different record, or observation

Also known as keeping data "tidy"

Each column is a single variable

2. Manipulate your data using **data verbs**



3. Example

"What countries have the highest life expectancy, on average across time?"

```
(data
    %>% select    ( country, year, life_expectancy                          )
    %>% filter    ( !is.na( life_expectancy )                               )
    %>% group_by  ( country                                                 )
    %>% summarise( life_expectancy_average = mean( life_expectancy ))
    %>% arrange   ( desc(life_expectancy_average )                          )
    %>% head      ( n = 10 )                                                )
```

Data verb	Description	In R / dplyr	In SQL	In python (pandas or polars)
SELECT	**Selects** columns	`select()`	`SELECT`	`.select()`
MUTATE	Adds new columns	`mutate()`	`ADD COLUMN`	`.mutate()`
FILTER	**Filters** rows based on column values	`filter()`	`WHERE`	`.filter()`
SUMMARISE	**Summarises** values over rows	`summarise()`	`(aggregate functions)`	`.agg()`
GROUP BY	**Group the data** frame by column values	`group_by()`	`GROUP BY`	`.groupby()`

SORT or **ARRANGE** or **ORDER**	**Sort** rows by column values	`arrange()`
		Or in groups:
		`arrange( .by_group = TRUE )`
			`ORDER BY`	`.sort()`

JOIN	**Joins** two data frames by shared column values	`inner_join(), full_join(), left_join(), right_join()`	`INNER JOIN etc.`	`.join()`

SLICE	Takes a subset of rows	`head() and tail()`
		Or in groups:
		`slice_head(), slice_tail() slice_sample()`		