Lecture 6: Working with data frames

(and tibbles)

Ingmar Sturm
UCSB

2024-07-02

Special thanks to Robin Liu for select course content used with permission.

Load the libraries

```r
library(tidyverse)
library(gapminder)
```

Recall what's in `gapminder`

```r
head(gapminder, 6)
```

```
## # A tibble: 6 x 6
##   country     continent  year lifeExp      pop gdpPercap
##   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
## 1 Afghanistan Asia       1952    28.8  8425333      779.
## 2 Afghanistan Asia       1957    30.3  9240934      821.
## 3 Afghanistan Asia       1962    32.0 10267083      853.
## 4 Afghanistan Asia       1967    34.0 11537966      836.
## 5 Afghanistan Asia       1972    36.1 13079460      740.
## 6 Afghanistan Asia       1977    38.4 14880372      786.
```

Filtering

Retrieve the data for Afghanistan for years after 1979.

```
## # A tibble: 6 x 6
##   country     continent  year lifeExp      pop gdpPercap
##   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
## 1 Afghanistan Asia       1982    39.9 12881816      978.
## 2 Afghanistan Asia       1987    40.8 13867957      852.
## 3 Afghanistan Asia       1992    41.7 16317921      649.
## 4 Afghanistan Asia       1997    41.8 22227415      635.
## 5 Afghanistan Asia       2002    42.1 25268405      727.
## 6 Afghanistan Asia       2007    43.8 31889923      975.
```

03:00

Filtering

A much better way using `dplyr::filter()` (which is part of `tidyverse`).

```r
filter(gapminder, country == "Afghanistan", year > 1979)
```

```
## # A tibble: 6 x 6
##   country     continent  year lifeExp      pop gdpPercap
##   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
## 1 Afghanistan Asia       1982    39.9 12881816      978.
## 2 Afghanistan Asia       1987    40.8 13867957      852.
## 3 Afghanistan Asia       1992    41.7 16317921      649.
## 4 Afghanistan Asia       1997    41.8 22227415      635.
## 5 Afghanistan Asia       2002    42.1 25268405      727.
## 6 Afghanistan Asia       2007    43.8 31889923      975.
```

Less repetition, easier to read.

Selecting

```
head(gapminder)
```

```
## # A tibble: 6 x 6
##   country     continent  year lifeExp      pop gdpPercap
##   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
## 1 Afghanistan Asia       1952    28.8  8425333      779.
## 2 Afghanistan Asia       1957    30.3  9240934      821.
## 3 Afghanistan Asia       1962    32.0 10267083      853.
## 4 Afghanistan Asia       1967    34.0 11537966      836.
## 5 Afghanistan Asia       1972    36.1 13079460      740.
## 6 Afghanistan Asia       1977    38.4 14880372      786.
```

Select only `country`, `year`, `lifeExp`, and `pop` variables.

02:00

Selecting

Easier way with `dplyr::select()`

```
head(select(gapminder, country, year, lifeExp, pop), 3)
```

```
## # A tibble: 3 x 4
##   country      year lifeExp      pop
##   <fct>       <int>   <dbl>    <int>
## 1 Afghanistan  1952    28.8  8425333
## 2 Afghanistan  1957    30.3  9240934
## 3 Afghanistan  1962    32.0 10267083
```

Also works with negative indices:

```
head(select(gapminder, -country, -year, -lifeExp, -pop), 3)
```

```
## # A tibble: 3 x 2
##   continent gdpPercap
##   <fct>         <dbl>
## 1 Asia           779.
## 2 Asia           821.
## 3 Asia           853.
```

Combining select and filter

```
head(gapminder)
```

```
## # A tibble: 6 x 6
##   country     continent  year lifeExp      pop gdpPercap
##   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
## 1 Afghanistan Asia       1952    28.8  8425333      779.
## 2 Afghanistan Asia       1957    30.3  9240934      821.
## 3 Afghanistan Asia       1962    32.0 10267083      853.
## 4 Afghanistan Asia       1967    34.0 11537966      836.
## 5 Afghanistan Asia       1972    36.1 13079460      740.
## 6 Afghanistan Asia       1977    38.4 14880372      786.
```

Return the `country`, `year`, and `lifeExp` for Rwanda in years between 1960 and 1970.

```
## # A tibble: 2 x 3
##   country  year lifeExp
##   <fct>   <int>   <dbl>
## 1 Rwanda   1962    43
## 2 Rwanda   1967    44.1
```

03:00

The Pipe

The operator `|>` *pipes* the left-hand side as **the first** argument to the right-hand side, and returns the result.

```r
25 |> sqrt() # no argument passed in explicitly
```

```
## [1] 5
```

```r
increment_power <- function(x, pwr = 2) {
  x <- x + 1
  return(x^pwr)
}
1 |> increment_power() # equiv. increment_power(1)
```

```
## [1] 4
```

```r
1 |> increment_power(3) # equiv. increment_power(1, 3)
```

```
## [1] 8
```

The Pipe

Use the pipe `|>` to simplify common operations.

```
select(filter(gapminder, country == "Rwanda", year > 1960, year < 1970),
       country, year, lifeExp)
```

```
## # A tibble: 2 x 3
##   country  year lifeExp
##   <fct>   <int>   <dbl>
## 1 Rwanda   1962    43
## 2 Rwanda   1967    44.1
```

```
gapminder |>
  filter(country == "Rwanda", year > 1960, year < 1970) |>
  select(country, year, lifeExp)
```

```
## # A tibble: 2 x 3
##   country  year lifeExp
##   <fct>   <int>   <dbl>
## 1 Rwanda   1962    43
## 2 Rwanda   1967    44.1
```

The Pipe

Another approach you may have tried:

```
filtered_gap <- filter(gapminder, country == "Rwanda", year > 1960, year < 1970)
select(filtered_gap, country, year, lifeExp)
```

```
## # A tibble: 2 x 3
##   country  year lifeExp
##   <fct>   <int>   <dbl>
## 1 Rwanda   1962    43
## 2 Rwanda   1967    44.1
```

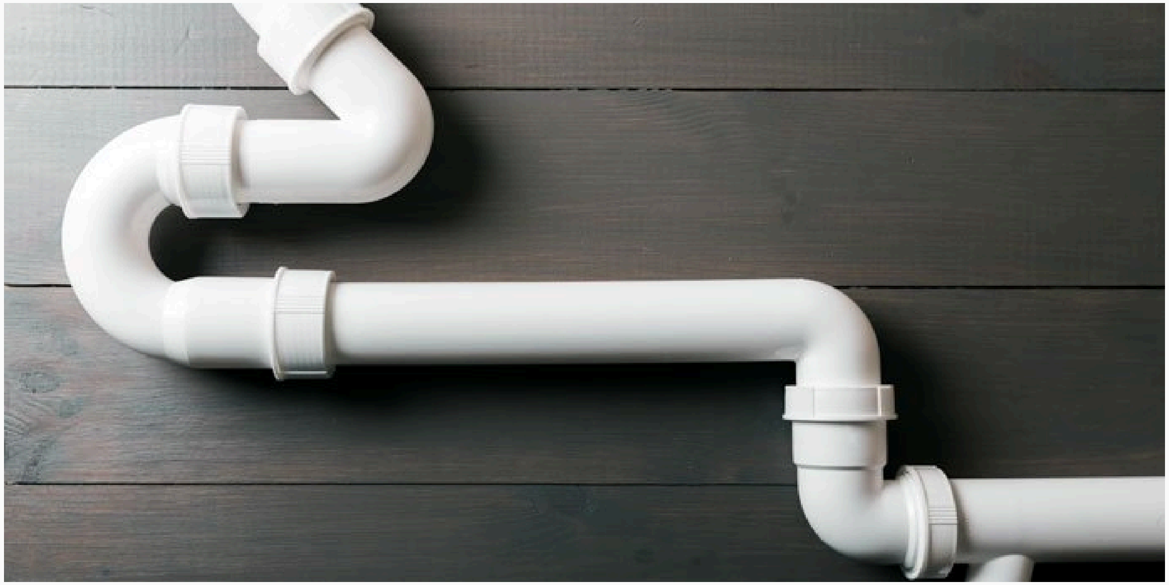
```
gapminder |>
  filter(country == "Rwanda", year > 1960, year < 1970) |>
  select(country, year, lifeExp)
```

```
## # A tibble: 2 x 3
##   country  year lifeExp
##   <fct>   <int>   <dbl>
## 1 Rwanda   1962    43
## 2 Rwanda   1967    44.1
```

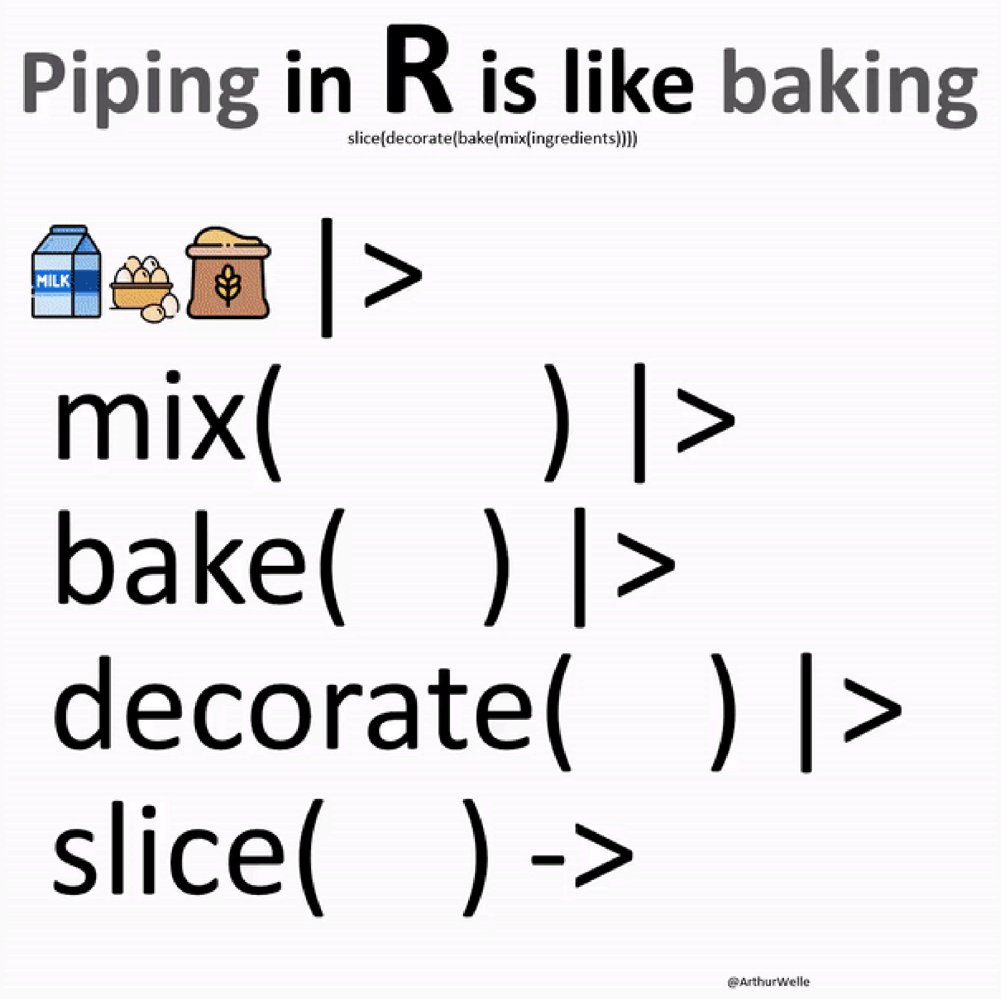
The Pipe

```r
gapminder |>
  filter(country == "Rwanda", year > 1960, year < 1970) |>
  select(country, year, lifeExp)
```

```
## # A tibble: 2 x 3
##   country  year lifeExp
##   <fct>   <int>   <dbl>
## 1 Rwanda   1962    43
## 2 Rwanda   1967    44.1
```



The Pipe


Piping in R is like baking
slice(decorate(bake(mix(ingredients))))
|>
mix( ) |>
bake( ) |>
decorate( ) |>
slice( ) ->
@ArthurWelle

img credit

The Pipe

Return the *first 3 rows* of `continent`, `year`, and `lifeExp` of the "Americas" continent by rewriting the following code using pipes.

```
head(select(filter(gapminder, country == "Chile"), continent, year, lifeExp), 3)
```

```
## # A tibble: 3 x 3
##   continent  year lifeExp
##   <fct>     <int>   <dbl>
## 1 Americas   1952    54.7
## 2 Americas   1957    56.1
## 3 Americas   1962    57.9
```

03:00

dplyr and tidyverse

Reminder that `select()` and `filter()` are functions in the `dplyr` package which is part of the `tidyverse`. You must call `library(dplyr)` or `library(tidyverse)` to use it. `library(tidyverse)` automatically loads `dplyr`; it is sometimes called a *meta-package*.

Interestingly `|>` is a base R pipe.

`tidyverse` has its own pipe which looks like `%>%`; you might see this in the wild. It became so popular that `|>` was recently added to base R.

Which one should you use? It's a matter of preference but, in short, use `%>%` if you want greater flexibility and you want your code to work without the `tidyverse`. Use `|>` if you want speed. comparison

```r
25 |> sqrt()
```

```
## [1] 5
```

```r
25 %>% sqrt()
```

```
## [1] 5
```

Comparing dplyr and base R

```
gapminder |>
  filter(country == "Chile") |>
  select(continent, year, lifeExp) |>
  head(3)
```

```
## # A tibble: 3 x 3
##   continent  year lifeExp
##   <fct>     <int>   <dbl>
## 1 Americas   1952    54.7
## 2 Americas   1957    56.1
## 3 Americas   1962    57.9
```

```
head(gapminder[gapminder$country == "Chile", c("continent", "year", "lifeExp")], 3)
```

```
## # A tibble: 3 x 3
##   continent  year lifeExp
##   <fct>     <int>   <dbl>
## 1 Americas   1952    54.7
## 2 Americas   1957    56.1
## 3 Americas   1962    57.9
```

More practice

Return a tibble containing `country`, `year`, and `gdpPercap` for countries with GDP per cap less than 300 for years before 2007.

```
## # A tibble: 3 x 3
##   country           year gdpPercap
##   <fct>            <int>     <dbl>
## 1 Congo, Dem. Rep.  2002      241.
## 2 Guinea-Bissau     1952      300.
## 3 Lesotho           1952      299.
```

05:00

Creating a tibble (or a data frame)

```r
(dogs <- tibble(name = c("Ralph", "Sully", "Capsule"),
                age = c(3, 12, 1)))
```

```
## # A tibble: 3 x 2
##   name      age
##   <chr>   <dbl>
## 1 Ralph       3
## 2 Sully      12
## 3 Capsule     1
```

```r
is_rescue <- c(F, F, T)
(dogs <- tibble(name = c("Ralph", "Sully", "Capsule"),
                age = c(3, 12, 1),
                rescue = is_rescue))
```

```
## # A tibble: 3 x 3
##   name      age rescue
##   <chr>   <dbl> <lgl>
## 1 Ralph       3 FALSE
## 2 Sully      12 FALSE
## 3 Capsule     1 TRUE
```

Adding variables to an existing tibble

```
(dogs <- tibble(name = c("Ralph", "Sully", "Capsule"),
                age = c(3, 12, 1)))
```

```
## # A tibble: 3 x 2
##   name      age
##   <chr>   <dbl>
## 1 Ralph       3
## 2 Sully      12
## 3 Capsule     1
```

```
is_rescue <- c(F, F, T)
dogs$rescue <- is_rescue
dogs
```

```
## # A tibble: 3 x 3
##   name      age rescue
##   <chr>   <dbl> <lgl>
## 1 Ralph       3 FALSE
## 2 Sully      12 FALSE
## 3 Capsule     1 TRUE
```

Adding variables to an existing tibble

```
head(gapminder, 1)
```

```
## # A tibble: 1 x 6
##   country     continent  year lifeExp     pop gdpPercap
##   <fct>       <fct>     <int>   <dbl>   <int>     <dbl>
## 1 Afghanistan Asia       1952    28.8 8425333      779.
```

Return the *total* GDP for each row. This is pop $\times$ gdpPercap.

```
my_gap <- gapminder # create a copy so we don't overwrite gapminder
my_gap$totalGdp <- my_gap$pop * my_gap$gdpPercap
head(my_gap, 2)
```

```
## # A tibble: 2 x 7
##   country     continent  year lifeExp     pop gdpPercap    totalGdp
##   <fct>       <fct>     <int>   <dbl>   <int>     <dbl>       <dbl>
## 1 Afghanistan Asia       1952    28.8 8425333      779. 6567086330.
## 2 Afghanistan Asia       1957    30.3 9240934      821. 7585448670.
```

Remember `my_gap$pop` and `my_gap$gdpPercap` are vectors.

dplyr::Mutate()

```
(dogs <- tibble(name = c("Ralph", "Sully", "Capsule"),
                age = c(3, 12, 1)))
```

```
## # A tibble: 3 x 2
##   name      age
##   <chr>   <dbl>
## 1 Ralph       3
## 2 Sully      12
## 3 Capsule     1
```

```
dogs <- tibble(name = c("Ralph", "Sully", "Capsule"),
               age = c(3, 12, 1))
dogs |> mutate(rescue = c(F, F, T))
```

```
## # A tibble: 3 x 3
##   name      age rescue
##   <chr>   <dbl> <lgl>
## 1 Ralph       3 FALSE
## 2 Sully      12 FALSE
## 3 Capsule     1 TRUE
```

Adding Variables

Return a tibble containing `country`, `year`, and *total GDP* for countries with GDP per cap less than 300 for years before 2007.

```
gapminder |>
  mutate(totalGdp = pop * gdpPercap) |>
  filter(gdpPercap < 300, year < 2007) |>
  select(country, year, totalGdp)
```

```
## # A tibble: 3 x 3
##   country           year     totalGdp
##   <fct>            <int>        <dbl>
## 1 Congo, Dem. Rep.  2002 13355730548.
## 2 Guinea-Bissau     1952   174108987.
## 3 Lesotho           1952   223760205.
```

```
head(gapminder, 1) # Remember the original tibble is unmodified!!
```

```
## # A tibble: 1 x 6
##   country     continent  year lifeExp     pop gdpPercap
##   <fct>       <fct>     <int>   <dbl>   <int>     <dbl>
## 1 Afghanistan Asia       1952    28.8 8425333      779.
```

Saving the result to a new tibble

```
my_gap <- gapminder |>
            mutate(totalGdp = pop * gdpPercap) |>
            filter(gdpPercap < 300, year < 2007) |>
            select(country, year, totalGdp)
my_gap
```

```
## # A tibble: 3 x 3
##   country           year    totalGdp
##   <fct>            <int>       <dbl>
## 1 Congo, Dem. Rep.  2002 13355730548.
## 2 Guinea-Bissau     1952   174108987.
## 3 Lesotho           1952   223760205.
```

Summary

- It is common to manipulate tibbles and data frames
- `filter`, `select`, and `mutate` are useful functions in `dplyr` which is part of the `tidyverse` meta-package
- the pipe `|>` operator simplifies a lot of operations, but don't go overboard!!

$$\frac{5^7 - 2\sqrt{4}}{\log_2(100)}$$

```
(5^7 - 2*sqrt(4)) / log(100, base = 2)
```

```
## [1] 11758.38
```

```
5 %>% # I'm using %>% here because this is not allowed with |>
  '^'(7) %>%
  '-'(2 %>% '*'(4 %>% sqrt())) %>%
  '/'(100 %>% log(2))
```

```
## [1] 11758.38
```

More on dplyr

`dplyr` contains many more functions for *data wrangling* (aka *data munging*).

See the dplyr cheatsheet

Most important things I left out: `group_by` and `summarize`.

```
gapminder |>
  group_by(continent) |>
  summarize(avg_life_exp = mean(lifeExp), avg_gdpPercap = mean(gdpPercap))
```

```
## # A tibble: 5 x 3
##   continent avg_life_exp avg_gdpPercap
##   <fct>            <dbl>         <dbl>
## 1 Africa            48.9         2194.
## 2 Americas          64.7         7136.
## 3 Asia              60.1         7902.
## 4 Europe            71.9        14469.
## 5 Oceania           74.3        18622.
```