IMT 573: lab-dplyr

Alexander Davis

Tuesday, October 13, 2020

Obectives

In this demo we will practice working with data. We will employ the `dplyr` verbs to manipulate a dataset in various ways.

```r
# Load some helpful libraries for this course
library(tidyverse)
```

Import and inspect the data We'll be using Gapminder data, which represents the health and wealth of nations. It was pioneered by Hans Rosling, who is famous for describing the prosperity of nations over time through famines, wars and other historic events with this beautiful data visualization in his 2006 TED Talk: The best stats you've ever seen:

Let's import this data into R and see what it looks like.

```r
gapminder <- read_csv('https://raw.githubusercontent.com/OHI-Science/data-science-training/master/data/g
```

```
##
## -- Column specification ----------------------------------------------------------------------------
## cols(
##   country = col_character(),
##   year = col_double(),
##   pop = col_double(),
##   continent = col_character(),
##   lifeExp = col_double(),
##   gdpPercap = col_double()
## )
```

What is the size of this dataset?

```r
# Find and print the number of row and columns in this dataset
dim(gapminder)
```

```
## [1] 1704    6
```

Consider looking at the raw data. What variables are listed? What data types are used for this data? Do you spot any immediate concerns?

```r
# Use RStudio utils to view the raw dataset
# View(gapminder)
# Get variable summaries
summary(gapminder)
```

```
##    country               year           pop               continent        
##  Length:1704        Min.   :1952   Min.   :6.001e+04   Length:1704       
##  Class :character   1st Qu.:1966   1st Qu.:2.794e+06   Class :character  
##  Mode  :character   Median :1980   Median :7.024e+06   Mode  :character  
##                     Mean   :1980   Mean   :2.960e+07                     
##                     3rd Qu.:1993   3rd Qu.:1.959e+07                     
##                     Max.   :2007   Max.   :1.319e+09                     
##     lifeExp        gdpPercap       
##  Min.   :23.60   Min.   :   241.2  
##  1st Qu.:48.20   1st Qu.:  1202.1  
##  Median :60.71   Median :  3531.8  
##  Mean   :59.47   Mean   :  7215.3  
##  3rd Qu.:70.85   3rd Qu.:  9325.5  
##  Max.   :82.60   Max.   :113523.1  
```

dplyr Verbs for Data Manipulation

dplyr is a grammar of data manipulation, providing a consistent set of verbs that help you solve the most common data manipulation challenges:

- mutate() adds new variables that are functions of existing variables
- select() picks variables based on their names.
- filter() picks cases based on their values.
- summarise() reduces multiple values down to a single summary.
- arrange() changes the ordering of the rows.

Let's practice our data manipulation skills with the gapminder data that you just loaded.

Filter: keep rows matching criteria

Q1: Filter the gapminder data for life expectancy less than 29

```r
filter(gapminder, lifeExp < 29)
```

```
## # A tibble: 2 x 6
##   country      year     pop continent lifeExp gdpPercap
##   <chr>       <dbl>   <dbl> <chr>       <dbl>     <dbl>
## 1 Afghanistan  1952 8425333 Asia         28.8      779.
## 2 Rwanda       1992 7290203 Africa       23.6      737.
```

Q2: "Filter the gapminder data for the country Mexico"

```r
## Your turn
filter(gapminder, country == "Mexico")
```

```
## # A tibble: 12 x 6
##    country  year       pop continent lifeExp gdpPercap
##    <chr>   <dbl>     <dbl> <chr>       <dbl>     <dbl>
##  1 Mexico   1952  30144317 Americas     50.8     3478.
##  2 Mexico   1957  35015548 Americas     55.2     4132.
##  3 Mexico   1962  41121485 Americas     58.3     4582.
##  4 Mexico   1967  47995559 Americas     60.1     5755.
##  5 Mexico   1972  55984294 Americas     62.4     6809.
##  6 Mexico   1977  63759976 Americas     65.0     7675.
##  7 Mexico   1982  71640904 Americas     67.4     9611.
##  8 Mexico   1987  80122492 Americas     69.5     8688.
##  9 Mexico   1992  88111030 Americas     71.5     9472.
## 10 Mexico   1997  95895146 Americas     73.7     9767.
## 11 Mexico   2002 102479927 Americas     74.9    10742.
## 12 Mexico   2007 108700891 Americas     76.2    11978.
```

Q3: if we want two country names? We can't use the == operator here, because it can only operate on one thing at a time. We will use the `%in%` operator:

```
filter(gapminder, country %in% c("Mexico", "Peru"))
```

```
## # A tibble: 24 x 6
##    country  year      pop continent lifeExp gdpPercap
##    <chr>   <dbl>    <dbl> <chr>       <dbl>     <dbl>
##  1 Mexico   1952 30144317 Americas     50.8     3478.
##  2 Mexico   1957 35015548 Americas     55.2     4132.
##  3 Mexico   1962 41121485 Americas     58.3     4582.
##  4 Mexico   1967 47995559 Americas     60.1     5755.
##  5 Mexico   1972 55984294 Americas     62.4     6809.
##  6 Mexico   1977 63759976 Americas     65.0     7675.
##  7 Mexico   1982 71640904 Americas     67.4     9611.
##  8 Mexico   1987 80122492 Americas     69.5     8688.
##  9 Mexico   1992 88111030 Americas     71.5     9472.
## 10 Mexico   1997 95895146 Americas     73.7     9767.
## # ... with 14 more rows
```

Q4: "We want data for Mexico in 2002?"

```
## Your turn
filter(gapminder, country == "Mexico" & year == 2002)
```

```
## # A tibble: 1 x 6
##   country  year       pop continent lifeExp gdpPercap
##   <chr>   <dbl>     <dbl> <chr>       <dbl>     <dbl>
## 1 Mexico   2002 102479927 Americas     74.9    10742.
```

Select: pick columns by name

```
select(gapminder, year, country, lifeExp)
```

```
## # A tibble: 1,704 x 3
##     year country     lifeExp
##    <dbl> <chr>         <dbl>
##  1  1952 Afghanistan    28.8
##  2  1957 Afghanistan    30.3
##  3  1962 Afghanistan    32.0
##  4  1967 Afghanistan    34.0
##  5  1972 Afghanistan    36.1
##  6  1977 Afghanistan    38.4
##  7  1982 Afghanistan    39.9
##  8  1987 Afghanistan    40.8
##  9  1992 Afghanistan    41.7
## 10  1997 Afghanistan    41.8
## # ... with 1,694 more rows
```

We can also use - to deselect columns

```
select(gapminder, -continent, -lifeExp) # you can use - to deselect columns
```

```
## # A tibble: 1,704 x 4
##    country      year      pop gdpPercap
##    <chr>       <dbl>    <dbl>     <dbl>
##  1 Afghanistan  1952  8425333      779.
##  2 Afghanistan  1957  9240934      821.
##  3 Afghanistan  1962 10267083      853.
##  4 Afghanistan  1967 11537966      836.
##  5 Afghanistan  1972 13079460      740.
##  6 Afghanistan  1977 14880372      786.
##  7 Afghanistan  1982 12881816      978.
##  8 Afghanistan  1987 13867957      852.
##  9 Afghanistan  1992 16317921      649.
## 10 Afghanistan  1997 22227415      635.
## # ... with 1,694 more rows
```

Arrange: reorder rows

Q: Sorted by year and then life-expectancy

```
arrange(gapminder, year, lifeExp)
```

```
## # A tibble: 1,704 x 6
##    country        year     pop continent lifeExp gdpPercap
##    <chr>         <dbl>   <dbl> <chr>       <dbl>     <dbl>
##  1 Afghanistan    1952 8425333 Asia         28.8      779.
##  2 Gambia         1952  284320 Africa       30        485.
##  3 Angola         1952 4232095 Africa       30.0     3521.
##  4 Sierra Leone   1952 2143249 Africa       30.3      880.
##  5 Mozambique     1952 6446316 Africa       31.3      469.
##  6 Burkina Faso   1952 4469979 Africa       32.0      543.
##  7 Guinea-Bissau  1952  580653 Africa       32.5      300.
##  8 Yemen Rep.     1952 4963829 Asia         32.5      782.
##  9 Somalia        1952 2526994 Africa       33.0     1136.
## 10 Guinea         1952 2664249 Africa       33.6      510.
## # ... with 1,694 more rows
```

Q: But your boss wants to see the data sorted in reverse chronological order.

```r
## Your turn
arrange(gapminder, -year)
```

```
## # A tibble: 1,704 x 6
##    country      year       pop continent lifeExp gdpPercap
##    <chr>       <dbl>     <dbl> <chr>       <dbl>     <dbl>
##  1 Afghanistan  2007  31889923 Asia         43.8      975.
##  2 Albania      2007   3600523 Europe       76.4     5937.
##  3 Algeria      2007  33333216 Africa       72.3     6223.
##  4 Angola       2007  12420476 Africa       42.7     4797.
##  5 Argentina    2007  40301927 Americas     75.3    12779.
##  6 Australia    2007  20434176 Oceania      81.2    34435.
##  7 Austria      2007   8199783 Europe       79.8    36126.
##  8 Bahrain      2007    708573 Asia         75.6    29796.
##  9 Bangladesh   2007 150448339 Asia         64.1     1391.
## 10 Belgium      2007  10392226 Europe       79.4    33693.
## # ... with 1,694 more rows
```

Mutate: add new variables

Q: Imagine we want to know each country's annual GDP. We can multiply pop by gdpPercap to create a new column named gdp.

```r
gapminder %>%
  mutate(gdp = pop * gdpPercap)
```

```
## # A tibble: 1,704 x 7
##    country      year      pop continent lifeExp gdpPercap          gdp
##    <chr>       <dbl>    <dbl> <chr>       <dbl>     <dbl>        <dbl>
##  1 Afghanistan  1952  8425333 Asia         28.8      779.  6567086330.
##  2 Afghanistan  1957  9240934 Asia         30.3      821.  7585448670.
##  3 Afghanistan  1962 10267083 Asia         32.0      853.  8758855797.
##  4 Afghanistan  1967 11537966 Asia         34.0      836.  9648014150.
##  5 Afghanistan  1972 13079460 Asia         36.1      740.  9678553274.
##  6 Afghanistan  1977 14880372 Asia         38.4      786. 11697659231.
##  7 Afghanistan  1982 12881816 Asia         39.9      978. 12598563401.
##  8 Afghanistan  1987 13867957 Asia         40.8      852. 11820990309.
##  9 Afghanistan  1992 16317921 Asia         41.7      649. 10595901589.
## 10 Afghanistan  1997 22227415 Asia         41.8      635. 14121995875.
## # ... with 1,694 more rows
```

Q. Now we want to calculate the annual GDP for all Asian countries in the year 2007 and add it as a new column. How can you do it?

```r
## Your turn
gapminder %>%
  filter(continent == "Asia" & year == 2007) %>%
  mutate(gdp = pop * gdpPercap)
```

```
## # A tibble: 33 x 7
##    country          year        pop continent lifeExp gdpPercap     gdp
##    <chr>           <dbl>      <dbl> <chr>       <dbl>     <dbl>   <dbl>
##  1 Afghanistan      2007   31889923 Asia         43.8      975. 3.11e10
##  2 Bahrain          2007     708573 Asia         75.6    29796. 2.11e10
##  3 Bangladesh       2007  150448339 Asia         64.1     1391. 2.09e11
##  4 Cambodia         2007   14131858 Asia         59.7     1714. 2.42e10
##  5 China            2007 1318683096 Asia         73.0     4959. 6.54e12
##  6 Hong Kong China  2007    6980412 Asia         82.2    39725. 2.77e11
##  7 India            2007 1110396331 Asia         64.7     2452. 2.72e12
##  8 Indonesia        2007  223547000 Asia         70.6     3541. 7.92e11
##  9 Iran             2007   69453570 Asia         71.0    11606. 8.06e11
## 10 Iraq             2007   27499638 Asia         59.5     4471. 1.23e11
## # ... with 23 more rows
```

Q. Now we want to calculate the population in thousands for all Asian countries in the year 2007 and add it as a new column. How can you do it? *Hint: You will use the same logic as the previous question, just with gdp calculation replaced with pop/1000 calculation*

```
## Your turn
gapminder %>%
  filter(continent == "Asia" & year == 2007) %>%
  mutate(popThousands = pop/1000)
```

```
## # A tibble: 33 x 7
##    country          year        pop continent lifeExp gdpPercap popThousands
##    <chr>           <dbl>      <dbl> <chr>       <dbl>     <dbl>        <dbl>
##  1 Afghanistan      2007   31889923 Asia         43.8      975.       31890.
##  2 Bahrain          2007     708573 Asia         75.6    29796.         709.
##  3 Bangladesh       2007  150448339 Asia         64.1     1391.      150448.
##  4 Cambodia         2007   14131858 Asia         59.7     1714.       14132.
##  5 China            2007 1318683096 Asia         73.0     4959.     1318683.
##  6 Hong Kong China  2007    6980412 Asia         82.2    39725.        6980.
##  7 India            2007 1110396331 Asia         64.7     2452.     1110396.
##  8 Indonesia        2007  223547000 Asia         70.6     3541.      223547
##  9 Iran             2007   69453570 Asia         71.0    11606.       69454.
## 10 Iraq             2007   27499638 Asia         59.5     4471.       27500.
## # ... with 23 more rows
```

Summarize with group_by

Q. Find the total population on each continent in 2005

```
gapminder %>%
  filter(year == 2002) %>%
  group_by(continent) %>%
  mutate(cont_pop = sum(pop))
```

```
## # A tibble: 142 x 7
## # Groups:   continent [5]
##    country      year        pop continent lifeExp gdpPercap   cont_pop
##    <chr>       <dbl>      <dbl> <chr>       <dbl>     <dbl>      <dbl>
##  1 Afghanistan  2002  25268405 Asia         42.1      727. 3601802203
##  2 Albania      2002   3508512 Europe       75.7     4604.  578223869
##  3 Algeria      2002  31287142 Africa       71.0     5288.  833723916
##  4 Angola       2002  10866106 Africa       41.0     2773.  833723916
##  5 Argentina    2002  38331121 Americas     74.3     8798.  849772762
##  6 Australia    2002  19546792 Oceania      80.4    30688.   23454829
##  7 Austria      2002   8148312 Europe       79.0    32418.  578223869
##  8 Bahrain      2002    656397 Asia         74.8    23404. 3601802203
##  9 Bangladesh   2002 135656790 Asia         62.0     1136. 3601802203
## 10 Belgium      2002  10311970 Europe       78.3    30486.  578223869
## # ... with 132 more rows
```

Q. Find the median population on each continent in 2002

```
## Your Turn
gapminder %>%
  select(continent, country, year, pop) %>%
  filter(year == 2002) %>%
  group_by(continent, year) %>%
  summarise(median = median(pop))
```

```
## ‘summarise()‘ regrouping output by ’continent’ (override with ‘.groups‘ argument)
```

```
## # A tibble: 5 x 3
## # Groups:   continent [5]
##   continent  year    median
##   <chr>     <dbl>     <dbl>
## 1 Africa     2002  8821778.
## 2 Americas   2002  8650322
## 3 Asia       2002 22662365
## 4 Europe     2002  9518744
## 5 Oceania    2002 11727414.
```

Summarize with group_by

We can use more than one grouping variable. Let’s get total populations by continent and year.

```
gapminder %>%
  group_by(continent, year) %>%
  summarize(cont_pop = sum(pop))
```

```
## ‘summarise()‘ regrouping output by ’continent’ (override with ‘.groups‘ argument)
```

```
## # A tibble: 60 x 3
## # Groups:   continent [5]
##    continent  year  cont_pop
##    <chr>     <dbl>     <dbl>
##  1 Africa     1952 237640501
##  2 Africa     1957 264837738
##  3 Africa     1962 296516865
##  4 Africa     1967 335289489
##  5 Africa     1972 379879541
##  6 Africa     1977 433061021
##  7 Africa     1982 499348587
##  8 Africa     1987 574834110
##  9 Africa     1992 659081517
## 10 Africa     1997 743832984
## # ... with 50 more rows
```

Let’s chain many of these verbs. What is the maximum GDP per continent across all years?

```
## Your turn
gapminder %>%
  group_by(continent, year) %>%
  summarize(max(gdpPercap))
```

```
## ‘summarise()‘ regrouping output by ’continent’ (override with ‘.groups‘ argument)
```

```
## # A tibble: 60 x 3
## # Groups:   continent [5]
##    continent  year ‘max(gdpPercap)‘
##    <chr>     <dbl>            <dbl>
##  1 Africa     1952            4725.
##  2 Africa     1957            5487.
##  3 Africa     1962            6757.
##  4 Africa     1967           18773.
##  5 Africa     1972           21011.
##  6 Africa     1977           21951.
##  7 Africa     1982           17364.
##  8 Africa     1987           11864.
##  9 Africa     1992           13522.
## 10 Africa     1997           14723.
## # ... with 50 more rows
```

Find the maximum life expectancy for countries in Asia.

```
## Your turn
gapminder %>%
  filter(continent == "Asia") %>%
  group_by(continent, country) %>%
  summarize(max(lifeExp))
```

```
## ‘summarise()‘ regrouping output by ’continent’ (override with ‘.groups‘ argument)
```

```
## # A tibble: 33 x 3
## # Groups:   continent [1]
##    continent country          ‘max(lifeExp)‘
##    <chr>     <chr>                     <dbl>
##  1 Asia      Afghanistan                43.8
##  2 Asia      Bahrain                    75.6
##  3 Asia      Bangladesh                 64.1
##  4 Asia      Cambodia                   59.7
##  5 Asia      China                      73.0
##  6 Asia      Hong Kong China            82.2
##  7 Asia      India                      64.7
##  8 Asia      Indonesia                  70.6
##  9 Asia      Iran                       71.0
## 10 Asia      Iraq                       65.0
## # ... with 23 more rows
```

Additional Data wrangling questions Q. What other questions can you ask from the gapminder dataset (questions that you can answer by wrangling the data)? *Growth of life expentancy along with growth in life expectancy?*

Data wrangling questions with another dataset Q. PS1 was on nycflight data. Assuming that the problem set and data is fresh in your mind, what other questions can you ask from the flight dataset (questions that you can answer by wrangling the data)? And what dplyr functions would you use to answer that question? (*Example question: What is the typical delay of flights?*) *Are flights delayed more in the A.M. or P.M? Are there better times during the day? Functions to use are probably group_by, max or min, and probably a filter*