cm006: `dplyr` Exercise

Optional, but recommended startup:

1. Change the file output to both html and md *documents* (not notebook).
2. `knit` the document.
3. Stage and commit the rmd, and knitted documents.

Intro to `dplyr` syntax

Load the `gapminder` and `tidyverse` packages. Hint: `suppressPackageStartupMessages()`! - This loads `dplyr`, too.

```
# load your packages here:
library(gapminder)
library(tidyverse)
```

`select()` (8 min)

1. Make a data frame containing the columns `year`, `lifeExp`, `country` from the gapminder data, in that order.

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

```
## # A tibble: 1,704 x 2
##    lifeExp country
##      <dbl> <fct>
##  1    28.8 Afghanistan
##  2    30.3 Afghanistan
##  3    32.0 Afghanistan
##  4    34.0 Afghanistan
##  5    36.1 Afghanistan
##  6    38.4 Afghanistan
##  7    39.9 Afghanistan
##  8    40.8 Afghanistan
##  9    41.7 Afghanistan
## 10    41.8 Afghanistan
## # ... with 1,694 more rows
```

2. Select all variables, from `country` to `lifeExp`.

```
# This will work:
select(gapminder, country, continent, year, lifeExp)
```

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

```
# Better way:
## Colon tells us to select everything in between
select(gapminder, country:lifeExp)
```

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

3. Select all variables, except `lifeExp`.

```
select(gapminder, -lifeExp)
```

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

4. Put `continent` first. Hint: use the `everything()` function.

```
select(gapminder, continent, everything())
```

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

5. Rename `continent` to `cont`.

```
# compare
select(gapminder, cont = continent, everything())
```

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

```
rename(gapminder, cont = continent)
```

```
## # A tibble: 1,704 x 6
##    country     cont   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.
##  7 Afghanistan Asia   1982    39.9 12881816      978.
```

```
##  8 Afghanistan Asia   1987    40.8 13867957      852.
##  9 Afghanistan Asia   1992    41.7 16317921      649.
## 10 Afghanistan Asia   1997    41.8 22227415      635.
## # ... with 1,694 more rows
```

arrange() (8 min)

1. Order by year.

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

```
## # A tibble: 1,704 x 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Albania     Europe     1952    55.2  1282697     1601.
##  3 Algeria     Africa     1952    43.1  9279525     2449.
##  4 Angola      Africa     1952    30.0  4232095     3521.
##  5 Argentina   Americas   1952    62.5 17876956     5911.
##  6 Australia   Oceania    1952    69.1  8691212    10040.
##  7 Austria     Europe     1952    66.8  6927772     6137.
##  8 Bahrain     Asia       1952    50.9   120447     9867.
##  9 Bangladesh  Asia       1952    37.5 46886859      684.
## 10 Belgium     Europe     1952    68    8730405     8343.
## # ... with 1,694 more rows
```

2. Order by year, in descending order.

```
arrange(gapminder, desc(year))
```

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

3. Order by year, then by life expectancy.

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

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

Piping, %>% (8 min)

Note: think of `%>%` as the word "then"!

Demonstration:

Here I want to combine `select()` Task 1 with `arrange()` Task 3.

This is how I could do it by *nesting* the two function calls:

```
# Nesting function calls can be hard to read
arrange(select(gapminder, year, lifeExp, country), year, lifeExp)
```

Now using with pipes:

```
# alter the below to include 2 "pipes"
gapminder %>%
  select(year, lifeExp, country) %>%
  arrange(year, lifeExp)
```

```
## # A tibble: 1,704 x 3
##     year lifeExp country
##    <int>   <dbl> <fct>
##  1  1952    28.8 Afghanistan
##  2  1952    30   Gambia
##  3  1952    30.0 Angola
##  4  1952    30.3 Sierra Leone
##  5  1952    31.3 Mozambique
##  6  1952    32.0 Burkina Faso
##  7  1952    32.5 Guinea-Bissau
##  8  1952    32.5 Yemen, Rep.
##  9  1952    33.0 Somalia
## 10  1952    33.6 Guinea
## # ... with 1,694 more rows
```

Resume lecture

Return to guide at section 6.7.

`filter()` (10 min)

1. Only take data with population greater than 100 million.

```
gapminder %>%
  filter(pop > 100000000)
```

```
## # A tibble: 77 x 6
##    country    continent  year lifeExp       pop gdpPercap
##    <fct>      <fct>     <int>   <dbl>     <int>     <dbl>
##  1 Bangladesh Asia       1987    52.8 103764241      752.
##  2 Bangladesh Asia       1992    56.0 113704579      838.
##  3 Bangladesh Asia       1997    59.4 123315288      973.
##  4 Bangladesh Asia       2002    62.0 135656790     1136.
##  5 Bangladesh Asia       2007    64.1 150448339     1391.
##  6 Brazil     Americas   1972    59.5 100840058     4986.
##  7 Brazil     Americas   1977    61.5 114313951     6660.
##  8 Brazil     Americas   1982    63.3 128962939     7031.
##  9 Brazil     Americas   1987    65.2 142938076     7807.
## 10 Brazil     Americas   1992    67.1 155975974     6950.
## # ... with 67 more rows
```

2. Your turn: of those rows filtered from step 1., only take data from Asia.

```
gapminder %>%
  filter(pop > 100000000, continent == "Asia")
```

```
## # A tibble: 52 x 6
##    country    continent  year lifeExp       pop gdpPercap
##    <fct>      <fct>     <int>   <dbl>     <int>     <dbl>
##  1 Bangladesh Asia       1987    52.8 103764241      752.
##  2 Bangladesh Asia       1992    56.0 113704579      838.
##  3 Bangladesh Asia       1997    59.4 123315288      973.
##  4 Bangladesh Asia       2002    62.0 135656790     1136.
##  5 Bangladesh Asia       2007    64.1 150448339     1391.
##  6 China      Asia       1952    44   556263527      400.
##  7 China      Asia       1957    50.5 637408000      576.
##  8 China      Asia       1962    44.5 665770000      488.
##  9 China      Asia       1967    58.4 754550000      613.
## 10 China      Asia       1972    63.1 862030000      677.
## # ... with 42 more rows
```

can also write code this way too but this way is longer than above

```
gapminder %>%
  filter(pop > 100000000 & continent == "Asia")
```

```
## # A tibble: 52 x 6
##    country    continent  year lifeExp       pop gdpPercap
##    <fct>      <fct>     <int>   <dbl>     <int>     <dbl>
```

```
##  1 Bangladesh Asia       1987    52.8 103764241      752.
##  2 Bangladesh Asia       1992    56.0 113704579      838.
##  3 Bangladesh Asia       1997    59.4 123315288      973.
##  4 Bangladesh Asia       2002    62.0 135656790     1136.
##  5 Bangladesh Asia       2007    64.1 150448339     1391.
##  6 China      Asia       1952    44   556263527      400.
##  7 China      Asia       1957    50.5 637408000      576.
##  8 China      Asia       1962    44.5 665770000      488.
##  9 China      Asia       1967    58.4 754550000      613.
## 10 China      Asia       1972    63.1 862030000      677.
## # ... with 42 more rows
```

3. Repeat 2, but take data from countries Brazil, and China.

```
gapminder %>%
  filter(country == "Brazil" | country == "China")
```

```
## # A tibble: 24 x 6
##    country continent  year lifeExp       pop gdpPercap
##    <fct>   <fct>     <int>   <dbl>     <int>     <dbl>
##  1 Brazil  Americas   1952    50.9  56602560     2109.
##  2 Brazil  Americas   1957    53.3  65551171     2487.
##  3 Brazil  Americas   1962    55.7  76039390     3337.
##  4 Brazil  Americas   1967    57.6  88049823     3430.
##  5 Brazil  Americas   1972    59.5 100840058     4986.
##  6 Brazil  Americas   1977    61.5 114313951     6660.
##  7 Brazil  Americas   1982    63.3 128962939     7031.
##  8 Brazil  Americas   1987    65.2 142938076     7807.
##  9 Brazil  Americas   1992    67.1 155975974     6950.
## 10 Brazil  Americas   1997    69.4 168546719     7958.
## # ... with 14 more rows
```

`mutate()` (10 min)

Let's get:

- GDP by multiplying GPD per capita with population, and
- GDP in billions, named (`gdpBill`), rounded to two decimals.

```
gapminder %>%
  mutate(gdpBill = gdpPercap*pop/1000000000)
```

```
## # A tibble: 1,704 x 7
##    country     continent  year lifeExp      pop gdpPercap gdpBill
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>   <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.    6.57
##  2 Afghanistan Asia       1957    30.3  9240934      821.    7.59
##  3 Afghanistan Asia       1962    32.0 10267083      853.    8.76
##  4 Afghanistan Asia       1967    34.0 11537966      836.    9.65
##  5 Afghanistan Asia       1972    36.1 13079460      740.    9.68
##  6 Afghanistan Asia       1977    38.4 14880372      786.   11.7
```

```
##  7 Afghanistan Asia       1982    39.9 12881816      978.   12.6
##  8 Afghanistan Asia       1987    40.8 13867957      852.   11.8
##  9 Afghanistan Asia       1992    41.7 16317921      649.   10.6
## 10 Afghanistan Asia       1997    41.8 22227415      635.   14.1
## # ... with 1,694 more rows
```

rounding digits

```
gapminder %>%
  mutate(gdpBill = (gdpPercap*pop/1000000000) %>%  round (digits = 2))
```

```
## # A tibble: 1,704 x 7
##    country     continent  year lifeExp      pop gdpPercap gdpBill
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>   <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.    6.57
##  2 Afghanistan Asia       1957    30.3  9240934      821.    7.59
##  3 Afghanistan Asia       1962    32.0 10267083      853.    8.76
##  4 Afghanistan Asia       1967    34.0 11537966      836.    9.65
##  5 Afghanistan Asia       1972    36.1 13079460      740.    9.68
##  6 Afghanistan Asia       1977    38.4 14880372      786.   11.7
##  7 Afghanistan Asia       1982    39.9 12881816      978.   12.6
##  8 Afghanistan Asia       1987    40.8 13867957      852.   11.8
##  9 Afghanistan Asia       1992    41.7 16317921      649.   10.6
## 10 Afghanistan Asia       1997    41.8 22227415      635.   14.1
## # ... with 1,694 more rows
```

Notice the backwards compatibility! No need for loops!

Try the same thing, but with `transmute` (drops all other variables).

```
gapminder %>%
  transmute(gdpBill = gdpPercap*pop/1000000000)
```

```
## # A tibble: 1,704 x 1
##    gdpBill
##      <dbl>
##  1    6.57
##  2    7.59
##  3    8.76
##  4    9.65
##  5    9.68
##  6   11.7
##  7   12.6
##  8   11.8
##  9   10.6
## 10   14.1
## # ... with 1,694 more rows
```

The `if_else` function is useful for changing certain elements in a data frame.

Example: Suppose Canada's 1952 life expectancy was mistakenly entered as 68.8 in the data frame, but is actually 70. Fix it using `if_else` and `mutate`.

```r
gapminder %>%
  mutate(lifeExp, if_else(country == "Canada" & year == 1952, 70, lifeExp))
```

```
## # A tibble: 1,704 x 7
##    country  continent  year lifeExp    pop gdpPercap `if_else(country == "~
##    <fct>    <fct>     <int>   <dbl>  <int>     <dbl>                   <dbl>
##  1 Afghani~ Asia       1952    28.8 8.43e6      779.                    28.8
##  2 Afghani~ Asia       1957    30.3 9.24e6      821.                    30.3
##  3 Afghani~ Asia       1962    32.0 1.03e7      853.                    32.0
##  4 Afghani~ Asia       1967    34.0 1.15e7      836.                    34.0
##  5 Afghani~ Asia       1972    36.1 1.31e7      740.                    36.1
##  6 Afghani~ Asia       1977    38.4 1.49e7      786.                    38.4
##  7 Afghani~ Asia       1982    39.9 1.29e7      978.                    39.9
##  8 Afghani~ Asia       1987    40.8 1.39e7      852.                    40.8
##  9 Afghani~ Asia       1992    41.7 1.63e7      649.                    41.7
## 10 Afghani~ Asia       1997    41.8 2.22e7      635.                    41.8
## # ... with 1,694 more rows
```

Your turn: Make a new column called `cc` that pastes the country name followed by the continent, separated by a comma. (Hint: use the `paste` function with the `sep=", "` argument).

These functions we've seen are called **vectorized functions**.

git stuff (Optional)

Knit, commit, push!

Bonus Exercises

If there's time remaining, we'll practice with these three exercises. I'll give you 1 minute for each, then we'll go over the answer.

1. Take all countries in Europe that have a GDP per capita greater than 10000, and select all variables except `gdpPercap`. (Hint: use `-`).
2. Take the first three columns, and extract the names.
3. Of the `iris` data frame, take all columns that start with the word "Petal".
    - Hint: take a look at the "Select helpers" documentation by running the following code: `?tidyselect::select_helpers`.
4. Convert the population to a number in billions.
5. Filter the rows of the iris dataset for Sepal.Length >= 4.6 and Petal.Width >= 0.5.

Exercises 3. and 5. are from r-exercises.