HW 5

Enter your name and EID here: Erik Mercado, emm4376

You will submit this homework assignment as a pdf file on Gradescope.

For all questions, include the R commands/functions that you used to find your answer (show R chunk). Answers without supporting code will not receive credit. Write full sentences to describe your findings.

NOTE: You must use version 1.3.0 of the `tidyr` package for this homework. If you are not sure what version you have, you can run `install.packages("tidyr")` in the console window and R will install the latest version of the package.

Question 1: (1 pt)

The dataset `world_bank_pop` is a built-in dataset in `tidyverse`. It contains information about total population and population growth, overall and more specifically in urban areas, for countries around the world. Take a look at it with `head()`. Is the data tidy? Why or why not?

```
# Call tidyr, dplyr and ggplot2 packages within tidyverse
library(tidyverse)

# Take a look!
head(world_bank_pop)
```

```
## # A tibble: 6 x 20
##   country indica~1 `2000` `2001` `2002` `2003`  `2004`  `2005`   `2006`   `2007`
##   <chr>   <chr>     <dbl>  <dbl>  <dbl>  <dbl>   <dbl>   <dbl>    <dbl>    <dbl>
## 1 ABW     SP.URB.~ 4.24e4 4.30e4 4.37e4 4.42e4 4.47e+4 4.49e+4  4.49e+4  4.47e+4
## 2 ABW     SP.URB.~ 1.18e0 1.41e0 1.43e0 1.31e0 9.51e-1 4.91e-1 -1.78e-2 -4.35e-1
## 3 ABW     SP.POP.~ 9.09e4 9.29e4 9.50e4 9.70e4 9.87e+4 1.00e+5  1.01e+5  1.01e+5
## 4 ABW     SP.POP.~ 2.06e0 2.23e0 2.23e0 2.11e0 1.76e+0 1.30e+0  7.98e-1  3.84e-1
## 5 AFG     SP.URB.~ 4.44e6 4.65e6 4.89e6 5.16e6 5.43e+6 5.69e+6  5.93e+6  6.15e+6
## 6 AFG     SP.URB.~ 3.91e0 4.66e0 5.13e0 5.23e0 5.12e+0 4.77e+0  4.12e+0  3.65e+0
## # ... with 10 more variables: `2008` <dbl>, `2009` <dbl>, `2010` <dbl>,
## #   `2011` <dbl>, `2012` <dbl>, `2013` <dbl>, `2014` <dbl>, `2015` <dbl>,
## #   `2016` <dbl>, `2017` <dbl>, and abbreviated variable name 1: indicator
```

I think that the data is not tidy because there are multiple rows per observation (country).

Question 2: (1 pt)

Using `dplyr` functions on `world_bank_pop`, count how many distinct countries there are in the dataset. Does this makes sense? Why or why not?

```
# count the number of distinct countries
world_bank_pop |>
  distinct(country) |>
  summarize(n = n())
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   264
```

There are 264 distinct countries in the dataset. This does make sense because there are that many countries in the world.

Question 3: (2 pts)

Use one of the `pivot` functions on `world_bank_pop` to create a new dataset with the years 2000 to 2017 appearing as a *numeric* variable `year`, and the different values for the indicator variable are in a variable called `value`. Save this new dataset in your environment as `myworld1`.

```
# create new variables value and year from existing data
myworld1 <- world_bank_pop |>
  pivot_longer(cols = c("2000":"2017"), names_to = "year", values_to = "value") |>
  mutate(year = as.numeric(year))
```

How many rows are there per country? Why does it make sense?

```
# count number of rows per country
myworld1 |>
  group_by(country)|>
  summarize(n=n())
```

```
## # A tibble: 264 x 2
##    country     n
##    <chr>   <int>
##  1 ABW        72
##  2 AFG        72
##  3 AGO        72
##  4 ALB        72
##  5 AND        72
##  6 ARB        72
##  7 ARE        72
##  8 ARG        72
##  9 ARM        72
## 10 ASM        72
## # ... with 254 more rows
```

There are 72 rows per country. This makes sense because now there is a row for every indicator/year combination which comes out to 72 combinations (4 indicators * 18 years).

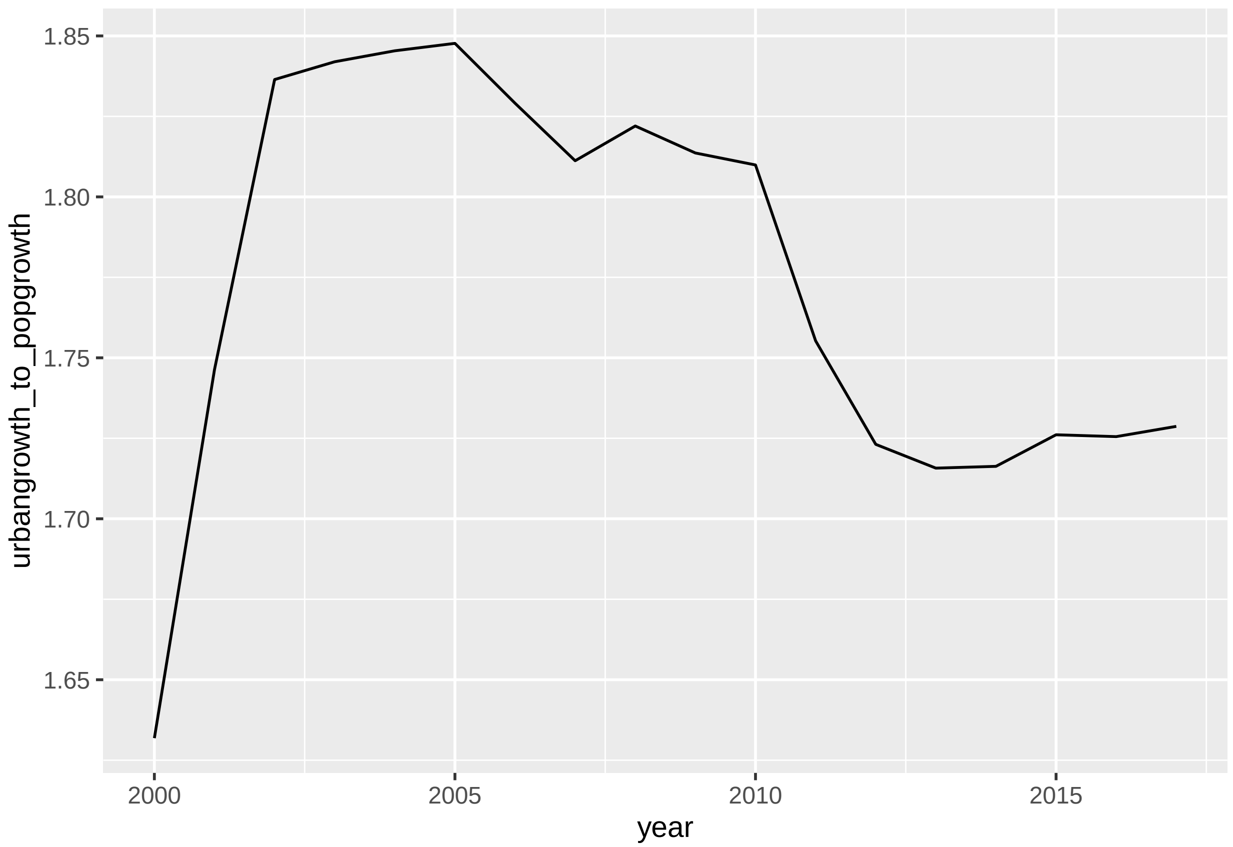
Question 4: (3 pts)

Use another `pivot` function on `myworld1` to create a new dataset, `myworld2`, with the different categories for the indicator variable appearing as their own variables. Use `dplyr` functions to rename `SP.POP.GROW` and `SP.URB.GROW`, as `pop_growth` and `pop_urb_growth` respectively.

```
# create vriable for each indicator and rename two of them
myworld2 <- myworld1 |>
  pivot_wider(names_from = "indicator", values_from = "value") |>
  rename(pop_growth = "SP.POP.GROW", pop_urb_growth = "SP.URB.GROW")
```

Using `dplyr` functions, find the ratio of urban growth compared to the population growth in the world for each year. *Hint: the country code `WLD` represents the entire world.* Create a `ggplot` to display how the percentage of urban population growth has changed over the years. Why does your graph not contradict the fact that the urban population worldwide is increasing over the years?

```
# create graph showing urban growth to population growth ratio of the world over the years
myworld2 |>
  filter(country == "WLD") |>
  group_by(year) |>
  summarize(urbangrowth_to_popgrowth = pop_urb_growth/pop_growth) |>
  ggplot(aes(x = year, y = urbangrowth_to_popgrowth)) +
  geom_line()
```



My graph does not contradict the fact that the urban population worldwide is increasing over the years because while the rate is decreasing over the years, the total population is still increasing.

Question 5: (1 pt)

In `myworld2`, which country code had the highest population growth in 2017? *Hint: Use the `arrange()` function here.*

```
# arrange countries by population growth and filter for 2017
myworld2 |>
  filter(year == "2017") |>
  arrange(desc(pop_growth))
```

```
## # A tibble: 264 x 6
##    country  year SP.URB.TOTL pop_urb_growth SP.POP.TOTL pop_growth
##    <chr>   <dbl>       <dbl>          <dbl>       <dbl>      <dbl>
##  1 OMN      2017     3874061           5.95     4636262       4.67
##  2 BHR      2017     1331176           4.73     1492584       4.62
##  3 NRU      2017       13649           4.50       13649       4.50
##  4 NER      2017     3511546           4.18    21477348       3.82
##  5 GNQ      2017      908248           4.42     1267689       3.71
##  6 AGO      2017    19311773           4.38    29784193        3.31
##  7 UGA      2017     9942492           5.76    42862958        3.26
##  8 COD      2017    35691987           4.57    81339988        3.25
##  9 BDI      2017     1380411           5.72    10864245        3.18
## 10 TZA      2017    18942681           5.28    57310019        3.08
## # ... with 254 more rows
```

The country code OMN has the highest population growth.

Question 6: (1 pt)

When answering the previous, we only reported the three-letter code and (probably) have no idea what the actual country is. We will now use the package `countrycode` with a built-in dataset called `codelist` that has information about the coding system used by the World bank:

Using `dplyr` functions, modify `mycodes` above to only keep the variables `continent`, `wb` (World Bank code), and `country.name.en` (country name in English). Then remove countries with missing `wb` code.

```
# Paste and run the following into your console (NOT HERE): install.packages("countrycode")

# Call the countrycode package
library(countrycode)

# Create a list of codes with matching country names
mycodes <- codelist |>
  select(continent, wb, country.name.en) |>
  filter(!is.na(wb))
```

How many countries are there in `mycodes`?

```
# count number of countries in dataset
mycodes |>
  distinct(country.name.en) |>
  summarize(n=n())
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   218
```

There are 218 countries in mycodes.

Question 7: (1 pt)

Use a `left_join()` function to add the information of the country codes **to** `myworld2` dataset. Match the two datasets based on the World Bank code. *Note: the column containing the World Bank code does not have the same name in each dataset.* Using `dplyr` functions, only keep the data available for Europe and for the year 2017. Save this new dataset as `myeurope`.

```r
# merge datastes and filter for continent and year, create new object
myeurope <- myworld2 |>
  left_join(mycodes, by = c("country" = "wb")) |>
  filter(continent == "Europe", year == "2017")
```

How many rows are there in this new dataset `myeurope`? What does each row represent?

```r
# count number fo rows in dataset
myeurope |>
  summarise(n=n())
```

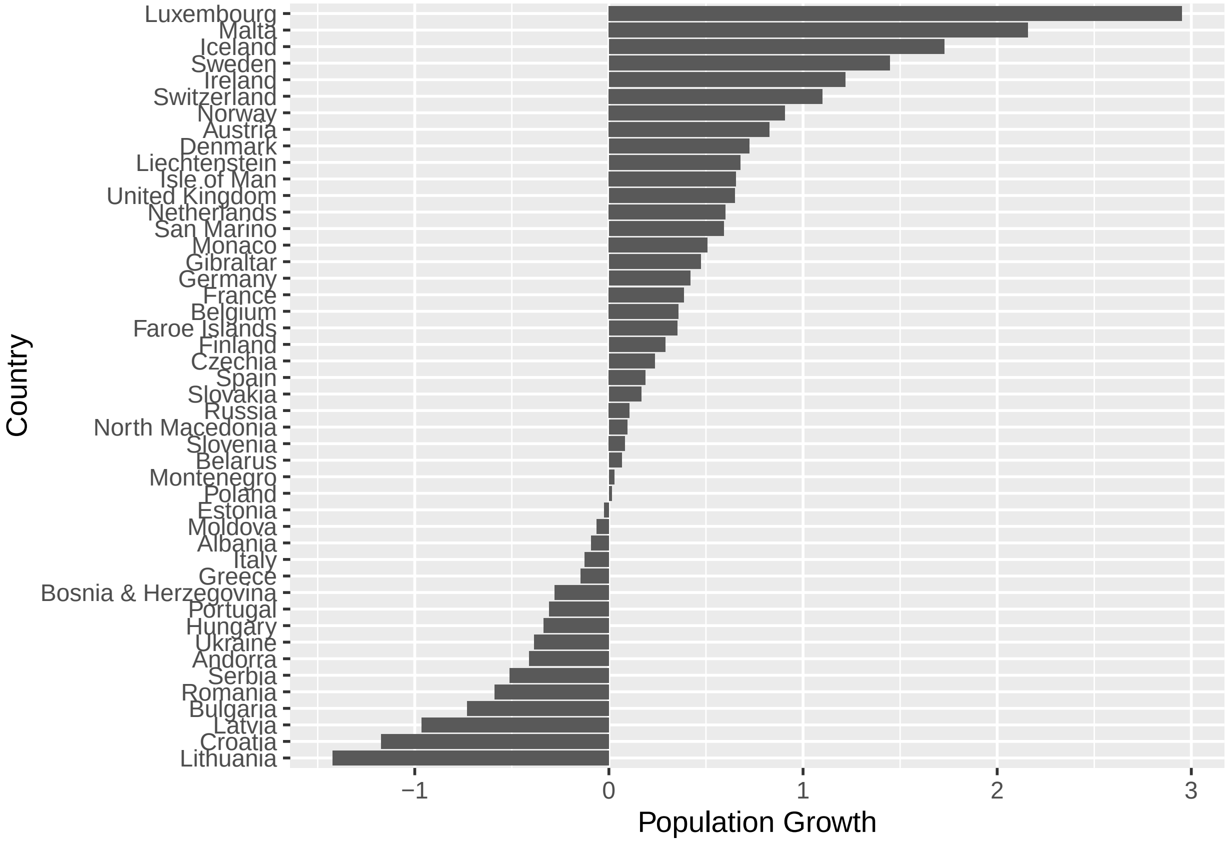
```
## # A tibble: 1 x 1
##       n
##   <int>
## 1    46
```

There are 46 rows in myeurope. Each row represents a country.

Question 8: (2 pts)

Using `dplyr` functions on `myeurope`, only keep information for the population growth in 2017 then compare the population growth per country with `ggplot` using `geom_bar()`. Use the `reorder()` function to order countries in order of population growth. Which country in Europe had the lowest population growth in 2017?

```r
# create bar graph of population growth rates
myeurope |>
  filter(year == 2017) |>
  arrange(desc(pop_growth)) |>
  ggplot(aes(y = reorder(country.name.en, pop_growth), x = pop_growth)) +
  geom_bar(stat = "identity") +
  labs(x = "Population Growth",  y = "Country")
```



The lowest popultion growth was from Lithuania.

Question 9: (1 pt)

When dealing with location data, we can actually visualize information on a map if we have geographic information such as latitude and longitude. Next, we will use a built-in function called `map_data()` to get geographic coordinates about countries in the world (see below). Take a look at the dataset `mapWorld`. What variables could we use to join `mapWorld` and `myeurope`? *Note: the variables do not have the same name in each dataset but they contain the same information.*

```
# Geographic coordinates about countries in the world
mapWorld <- map_data("world") %>%
        as_tibble()
```

I could use the "region" column in mapWorld and the "country.name.en" column in myeurope to join the two datasets.

Question 10: (2 pts)

Use a joining function to check if any information from `myeurope` is not contained in `mapWorld`, matching the two datasets based on the country name.

```
# merge datsats and then see which entries didn't get merged data from second dataset
myeurope |>
  left_join(mapWorld, by = c("country.name.en" = "region")) |>
  filter(is.na(long))
```

```
## # A tibble: 4 x 13
##   country  year SP.URB.TOTL pop_ur~1 SP.PO~2 pop_g~3 conti~4 count~5  long   lat
##   <chr>   <dbl>       <dbl>    <dbl>   <dbl>   <dbl> <chr>   <chr>   <dbl> <dbl>
## 1 BIH      2017     1679019    0.472  3.51e6  -0.279 Europe  Bosnia~    NA    NA
## 2 CZE      2017     7803157    0.379  1.06e7   0.236 Europe  Czechia    NA    NA
## 3 GBR      2017    54892898    0.958  6.60e7   0.648 Europe  United~    NA    NA
## 4 GIB      2017       34571    0.473  3.46e4   0.473 Europe  Gibral~    NA    NA
## # ... with 3 more variables: group <dbl>, order <int>, subregion <chr>, and
## #   abbreviated variable names 1: pop_urb_growth, 2: SP.POP.TOTL,
## #   3: pop_growth, 4: continent, 5: country.name.en
```

Some countries such as United Kingdom did not have a match. Why do you think this happened? *Hint: find the distinct country names in `mapWorld`, arrange them in alphabetical order, and scroll through the names. Can you find any of these countries with no match in a slightly different form?* If you need to print more output from a tibble, you can use `print(n = X)` where `X` is the number of lines to print out.

```
# arrange country names alphabetically
mapWorld |>
  distinct(region) |>
  arrange(region)
```

```
## # A tibble: 252 x 1
##    region
##    <chr>
##  1 Afghanistan
##  2 Albania
##  3 Algeria
##  4 American Samoa
##  5 Andorra
##  6 Angola
##  7 Anguilla
##  8 Antarctica
##  9 Antigua
## 10 Argentina
## # ... with 242 more rows
```

This most likely happened because the names of the countries were written differently in both datasets.

Question 11: (1 pt)

Consider the `myeurope` dataset. Recode some of the country names so that the countries with no match from the previous question (with the exception of Gibraltar which is not technically a country anyway) will have a match.

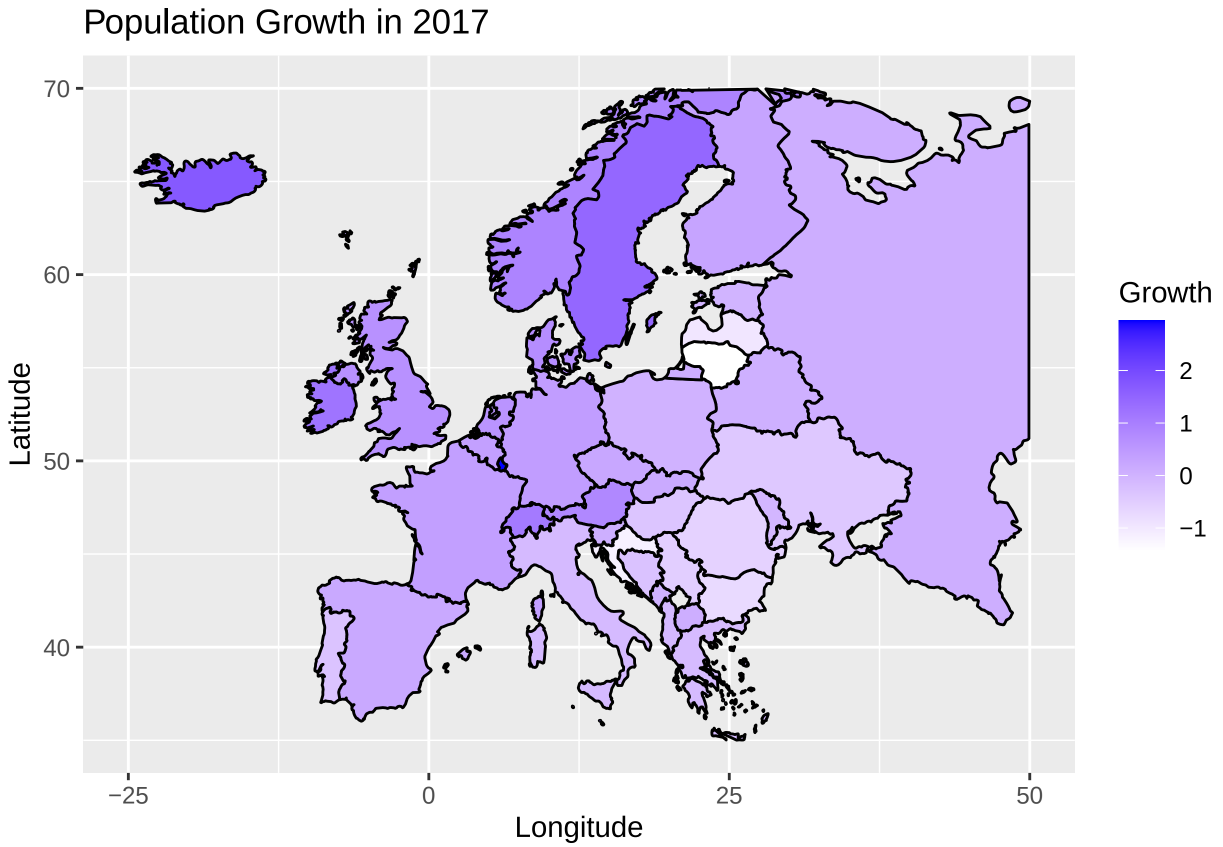
Hint: use ***recode()*** *inside* ***mutate()*** *as described in this article https://www.statology.org/recode-dplyr/.* Then add a pipe and use a `left_join()` function to add the geographic information in `mapWorld` to the countries in `myeurope`. Save this new dataset as `mymap`.

```
# recode country names to allow merge to happen
mymap <-myeurope |>
  mutate(country_name = recode(country.name.en,
                              "Bosnia & Herzegovina" = "Bosnia and Herzegovina",
                              "Czechia" = "Czech Republic",
                              "United Kingdom" = "UK")) |>
  left_join(mapWorld, by = c("country_name" = "region"))
```

Question 12: (2 pts)

Let's visualize how population growth varies across European countries in 2017 with a map. Use the R code provided below. Add a comment after each `#` to explain what each component of this code does. *Note: it would be a good idea to run the code piece by piece to see what each layer adds to the plot.*

```
# Build a map!
mymap %>%
  # loads the data into ggplot
  ggplot(aes(x = long, y = lat, group = group, fill = pop_growth)) +
  # specifies the type of graph
  geom_polygon(colour = "black") +
  # specifies the colors for coloring at different levels
  scale_fill_gradient(low = "white", high = "blue") +
  # gives the text for each of the axis and the title for the graph
  labs(fill = "Growth" ,title = "Population Growth in 2017",
       x ="Longitude", y ="Latitude") +
  # set the range for the axis
  xlim(-25,50) + ylim(35,70)
```



Which country had the highest population growth in Europe in 2017? *Hint: it's very tiny! You can refer to this map for European geography: https://www.wpmap.org/europe-map-hd-with-countries/*

From the graph, it looks like Luxembourg had the highest population growth.

Formatting: (2 pts)

Comment your code, write full sentences, and knit your file!

```
##                                                                       sysname
##                                                                      "Darwin"
##                                                                       release
##                                                                      "21.6.0"
##                                                                       version
## "Darwin Kernel Version 21.6.0: Sun Nov  6 23:31:16 PST 2022; root:xnu-8020.240.14~1/RELEASE_X86_64"
##                                                                      nodename
##                                         "wireless-10-145-38-150.public.utexas.edu"
##                                                                       machine
##                                                                      "x86_64"
##                                                                         login
##                                                                        "root"
##                                                                                                  user
##                                                                                                "erik"
##                                                                                        effective_user
##                                                                                                "erik"
```