HW 4

SDS348 Spring 2021

Rose Hedderman EID: rrh2298

This homework is due on Mar 8, 2021 at 8am. Submit 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.

Question 1: (9 pts)

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.

1.1 (1 pt) Save the dataset `world_bank_pop` as `myworld` and take a look at it with `head()`. Is the data tidy? Why or why not?

```
library(tidyverse)
myworld <- world_bank_pop

head(myworld)
```

```
## # A tibble: 6 x 20
##   country indicator `2000` `2001` `2002` `2003`  `2004`  `2005`   `2006`
##   <chr>   <chr>      <dbl>  <dbl>  <dbl>  <dbl>   <dbl>   <dbl>    <dbl>
## 1 ABW     SP.URB.T~ 4.24e4 4.30e4 4.37e4 4.42e4 4.47e+4 4.49e+4  4.49e+4
## 2 ABW     SP.URB.G~ 1.18e0 1.41e0 1.43e0 1.31e0 9.51e-1 4.91e-1 -1.78e-2
## 3 ABW     SP.POP.T~ 9.09e4 9.29e4 9.50e4 9.70e4 9.87e+4 1.00e+5  1.01e+5
## 4 ABW     SP.POP.G~ 2.06e0 2.23e0 2.23e0 2.11e0 1.76e+0 1.30e+0  7.98e-1
## 5 AFG     SP.URB.T~ 4.44e6 4.65e6 4.89e6 5.16e6 5.43e+6 5.69e+6  5.93e+6
## 6 AFG     SP.URB.G~ 3.91e0 4.66e0 5.13e0 5.23e0 5.12e+0 4.77e+0  4.12e+0
## # ... with 11 more variables: `2007` <dbl>, `2008` <dbl>, `2009` <dbl>,
## #   `2010` <dbl>, `2011` <dbl>, `2012` <dbl>, `2013` <dbl>, `2014` <dbl>,
## #   `2015` <dbl>, `2016` <dbl>, `2017` <dbl>
```

The dataset is not tidy because the indicator's can be split into smaller subsets that would better represent the data.

1.2 (1 pt) Using pipes and `dplyr` functions, how many countries are there in the dataset?

```
num <- myworld %>%
  group_by(country) %>%
  summarize(count=n())
num
```

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

There are 264 countries in the dataset.

1.3 (2 pts) Use one of the `pivot` functions to create a new dataset, `myworld2`, 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`**. In this new dataset, how many lines are there per country? Why does it make sense?

```
myworld2 <- myworld %>%
  pivot_longer(cols = starts_with("20"), names_to = "year", values_to = "value")
myworld2
```

```
## # A tibble: 19,008 x 4
##    country indicator   year  value
##    <chr>   <chr>       <chr> <dbl>
##  1 ABW     SP.URB.TOTL 2000  42444
##  2 ABW     SP.URB.TOTL 2001  43048
##  3 ABW     SP.URB.TOTL 2002  43670
##  4 ABW     SP.URB.TOTL 2003  44246
##  5 ABW     SP.URB.TOTL 2004  44669
##  6 ABW     SP.URB.TOTL 2005  44889
##  7 ABW     SP.URB.TOTL 2006  44881
##  8 ABW     SP.URB.TOTL 2007  44686
##  9 ABW     SP.URB.TOTL 2008  44375
## 10 ABW     SP.URB.TOTL 2009  44052
## # ... with 18,998 more rows
```

```
perCountry <- myworld2 %>%
  group_by(country) %>%
  summarize(count=n())
perCountry
```

```
## # A tibble: 264 x 2
##    country count
##  * <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
```

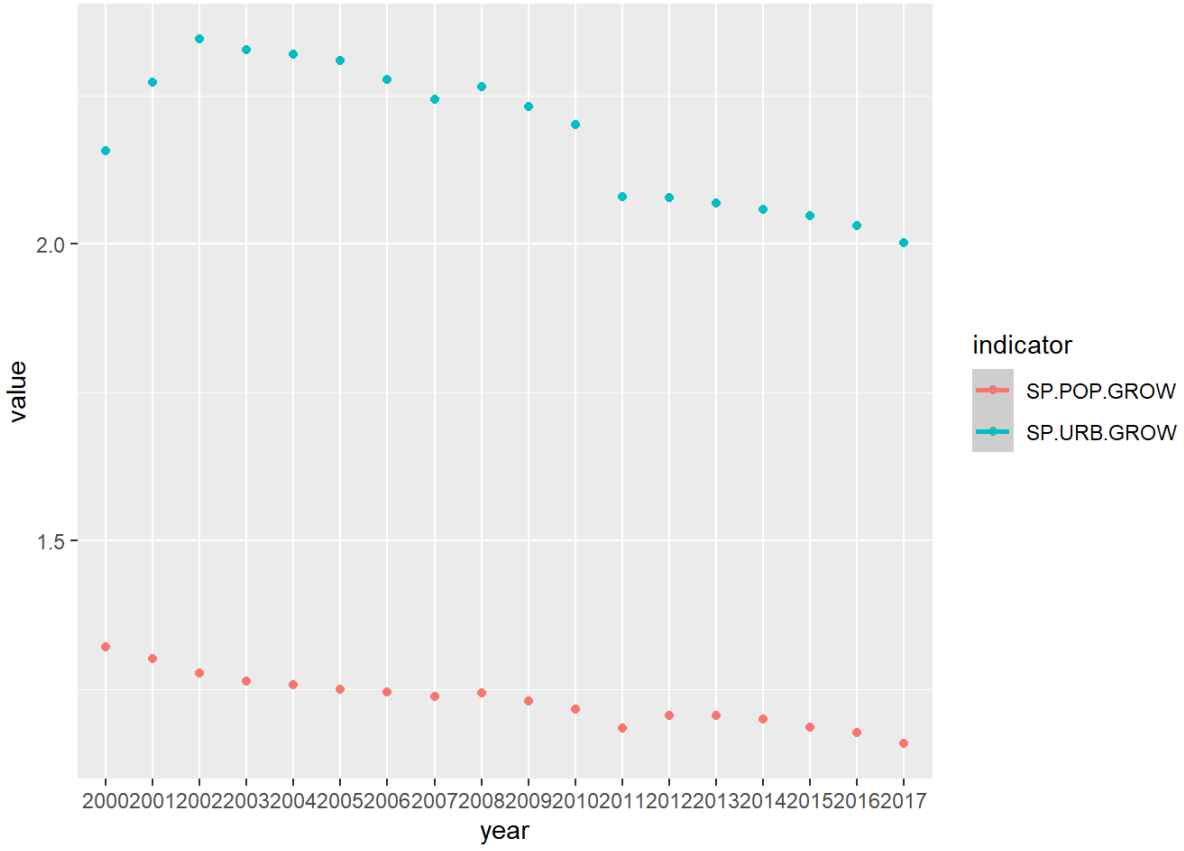
There are 72 lines per country because the dataset covers 18 years with 4 indicators per year per country.

1.4 (3 pts) Represent the total population growth and urban population growth in the world (country code is `WLD`) between 2000 and 2017. How has population growth changed over the years?

```
myworld14 <- myworld2 %>%
  filter(country == "WLD")%>%
  filter(indicator == "SP.URB.GROW" | indicator == "SP.POP.GROW")
myworld14
```

```
## # A tibble: 36 x 4
##    country indicator   year  value
##    <chr>   <chr>       <chr> <dbl>
##  1 WLD     SP.URB.GROW 2000   2.16
##  2 WLD     SP.URB.GROW 2001   2.27
##  3 WLD     SP.URB.GROW 2002   2.35
##  4 WLD     SP.URB.GROW 2003   2.33
##  5 WLD     SP.URB.GROW 2004   2.32
##  6 WLD     SP.URB.GROW 2005   2.31
##  7 WLD     SP.URB.GROW 2006   2.28
##  8 WLD     SP.URB.GROW 2007   2.24
##  9 WLD     SP.URB.GROW 2008   2.26
## 10 WLD     SP.URB.GROW 2009   2.23
## # ... with 26 more rows
```

```
ggplot(myworld14, aes(x=year, y = value, color = indicator)) + geom_point() + geom_smooth(method="lm")
```



Population growth has declined over the past 18 years. Growth has declined more on an urban scale compared to the total population

1.5 (2 pts) Use one of the `pivot` functions to create a new dataset, `myworld3`, 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. What is the country code that had the highest population growth in 2017?

```
myworld3 <- myworld2 %>%
  pivot_wider(names_from = indicator, values_from = value) %>%
  rename("pop_growth"=SP.POP.GROW, "pop_urb_growth"= SP.URB.GROW)
myworld3
```

```
## # A tibble: 4,752 x 6
##    country year  SP.URB.TOTL pop_urb_growth SP.POP.TOTL pop_growth
##    <chr>   <chr>       <dbl>          <dbl>       <dbl>      <dbl>
##  1 ABW     2000        42444         1.18         90853     2.06
##  2 ABW     2001        43048         1.41         92898     2.23
##  3 ABW     2002        43670         1.43         94992     2.23
##  4 ABW     2003        44246         1.31         97017     2.11
##  5 ABW     2004        44669         0.951        98737     1.76
##  6 ABW     2005        44889         0.491       100031     1.30
##  7 ABW     2006        44881        -0.0178      100832     0.798
##  8 ABW     2007        44686        -0.435       101220     0.384
##  9 ABW     2008        44375        -0.698       101353     0.131
## 10 ABW     2009        44052        -0.731       101453     0.0986
## # ... with 4,742 more rows
```

```
myworld4 <- myworld3 %>%
  filter(year == 2017) %>%
  arrange(desc(pop_growth))
myworld4
```

```
## # A tibble: 264 x 6
##    country year  SP.URB.TOTL pop_urb_growth SP.POP.TOTL pop_growth
##    <chr>   <chr>       <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 with the code OMN has the highest population growth in 2017.

Question 2: (10 pts)

From answering the previous question, we have no idea what actual countries are represented by the codes. We will now use a package that has information about the coding system used by the World bank.

2.1 (2 pts) Install the package `countrycode`. We will use a built-in dataset called `codelist`. Make sure to upload the library and save this dataset as `mycodes`. Using `dplyr` functions, modify `mycodes` to: 1. select only the variables continent, `wb` (World Bank code), and `country.name.en` (country name in English); 2. filter to keep countries in Europe only; 3. remove countries with missing `wb` code. How many countries are there in Europe with a World Bank code?

```
# install.packages("countrycode")

library(countrycode)
# dataset saved as mycodes
mycodes <- codelist
# mycodes modified to select/filter/modify according to the requirements of the question
mycodes <- mycodes %>%
  select(continent, wb, country.name.en) %>%
  filter(continent == "Europe") %>%
  filter(!is.na(wb))
mycodes
```

```
## # A tibble: 46 x 3
##    continent wb    country.name.en
##    <chr>     <chr> <chr>
##  1 Europe    ALB   Albania
##  2 Europe    AND   Andorra
##  3 Europe    AUT   Austria
##  4 Europe    BLR   Belarus
##  5 Europe    BEL   Belgium
##  6 Europe    BIH   Bosnia & Herzegovina
##  7 Europe    BGR   Bulgaria
##  8 Europe    HRV   Croatia
##  9 Europe    CZE   Czechia
## 10 Europe    DNK   Denmark
## # ... with 36 more rows
```

There are 46 countries in Eurpoe with a World Bank code.

2.2 (2 pts) Use a `left_join()` function to create a new dataset, `myeurope`, to add data to the countries in `mycodes` dataset from `myworld3` dataset. Match the two datasets based on the World Bank code. Using `dplyr` functions, change the name of the variable containing the World Bank code to `country`.

```
# create new dataset myeurope
myworld3 <- myworld3 %>%
  rename('wb' = country)
myworld3
```

```
## # A tibble: 4,752 x 6
##    wb    year  SP.URB.TOTL pop_urb_growth SP.POP.TOTL pop_growth
##    <chr> <chr>       <dbl>          <dbl>       <dbl>      <dbl>
##  1 ABW   2000        42444         1.18         90853     2.06
##  2 ABW   2001        43048         1.41         92898     2.23
##  3 ABW   2002        43670         1.43         94992     2.23
##  4 ABW   2003        44246         1.31         97017     2.11
##  5 ABW   2004        44669         0.951        98737     1.76
##  6 ABW   2005        44889         0.491       100031     1.30
##  7 ABW   2006        44881        -0.0178      100832     0.798
##  8 ABW   2007        44686        -0.435       101220     0.384
##  9 ABW   2008        44375        -0.698       101353     0.131
## 10 ABW   2009        44052        -0.731       101453     0.0986
## # ... with 4,742 more rows
```

```
myeurope <- mycodes %>%
  left_join(myworld3, by = 'wb')
myeurope
```

```
## # A tibble: 828 x 8
##    continent wb    country.name.en year  SP.URB.TOTL pop_urb_growth SP.POP.TOTL
##    <chr>     <chr> <chr>           <chr>       <dbl>          <dbl>       <dbl>
##  1 Europe    ALB   Albania         2000      1289391          0.742     3089027
##  2 Europe    ALB   Albania         2001      1298584          0.710     3060173
##  3 Europe    ALB   Albania         2002      1327220          2.18      3051010
##  4 Europe    ALB   Albania         2003      1354848          2.06      3039616
##  5 Europe    ALB   Albania         2004      1381828          1.97      3026939
##  6 Europe    ALB   Albania         2005      1407298          1.83      3011487
##  7 Europe    ALB   Albania         2006      1430886          1.66      2992547
##  8 Europe    ALB   Albania         2007      1452398          1.49      2970017
##  9 Europe    ALB   Albania         2008      1473392          1.44      2947314
## 10 Europe    ALB   Albania         2009      1495260          1.47      2927519
## # ... with 818 more rows, and 1 more variable: pop_growth <dbl>
```

2.3 (1 pt) Using `dplyr` functions, what was the total population in European countries in 2017? Give your answer in million (round to the next million).

```
# filter function filters for only numbers for 2017 and summarize(sum()) calculates the sum of the intended variable
myeurope
```

```
## # A tibble: 828 x 8
##    continent wb    country.name.en year  SP.URB.TOTL pop_urb_growth SP.POP.TOTL
##    <chr>     <chr> <chr>           <chr>       <dbl>          <dbl>       <dbl>
##  1 Europe    ALB   Albania         2000      1289391          0.742     3089027
##  2 Europe    ALB   Albania         2001      1298584          0.710     3060173
##  3 Europe    ALB   Albania         2002      1327220          2.18      3051010
##  4 Europe    ALB   Albania         2003      1354848          2.06      3039616
##  5 Europe    ALB   Albania         2004      1381828          1.97      3026939
##  6 Europe    ALB   Albania         2005      1407298          1.83      3011487
##  7 Europe    ALB   Albania         2006      1430886          1.66      2992547
##  8 Europe    ALB   Albania         2007      1452398          1.49      2970017
##  9 Europe    ALB   Albania         2008      1473392          1.44      2947314
## 10 Europe    ALB   Albania         2009      1495260          1.47      2927519
## # ... with 818 more rows, and 1 more variable: pop_growth <dbl>
```

```
myeurope23 <- myeurope %>%
  filter(year == 2017)
myeurope23
```

```
## # A tibble: 46 x 8
##    continent wb    country.name.en year  SP.URB.TOTL pop_urb_growth SP.POP.TOTL
##    <chr>     <chr> <chr>           <chr>       <dbl>          <dbl>       <dbl>
##  1 Europe    ALB   Albania         2017      1706345          1.54      2873457
##  2 Europe    AND   Andorra         2017        67845         -0.520       76965
##  3 Europe    AUT   Austria         2017      5117624          1.15      8809212
##  4 Europe    BLR   Belarus         2017      7428883          0.674     9507875
##  5 Europe    BEL   Belgium         2017     11140192          0.401    11372068
##  6 Europe    BIH   Bosnia & Herze~ 2017      1679019          0.472     3507017
##  7 Europe    BGR   Bulgaria        2017      5283572         -0.273     7075991
##  8 Europe    HRV   Croatia         2017      2337910         -0.705     4125700
##  9 Europe    CZE   Czechia         2017      7803157          0.379    10591323
## 10 Europe    DNK   Denmark         2017      5063231          0.855     5769603
## # ... with 36 more rows, and 1 more variable: pop_growth <dbl>
```

```
sum(myeurope23$SP.POP.TOTL)
```

```
## [1] 744218594
```

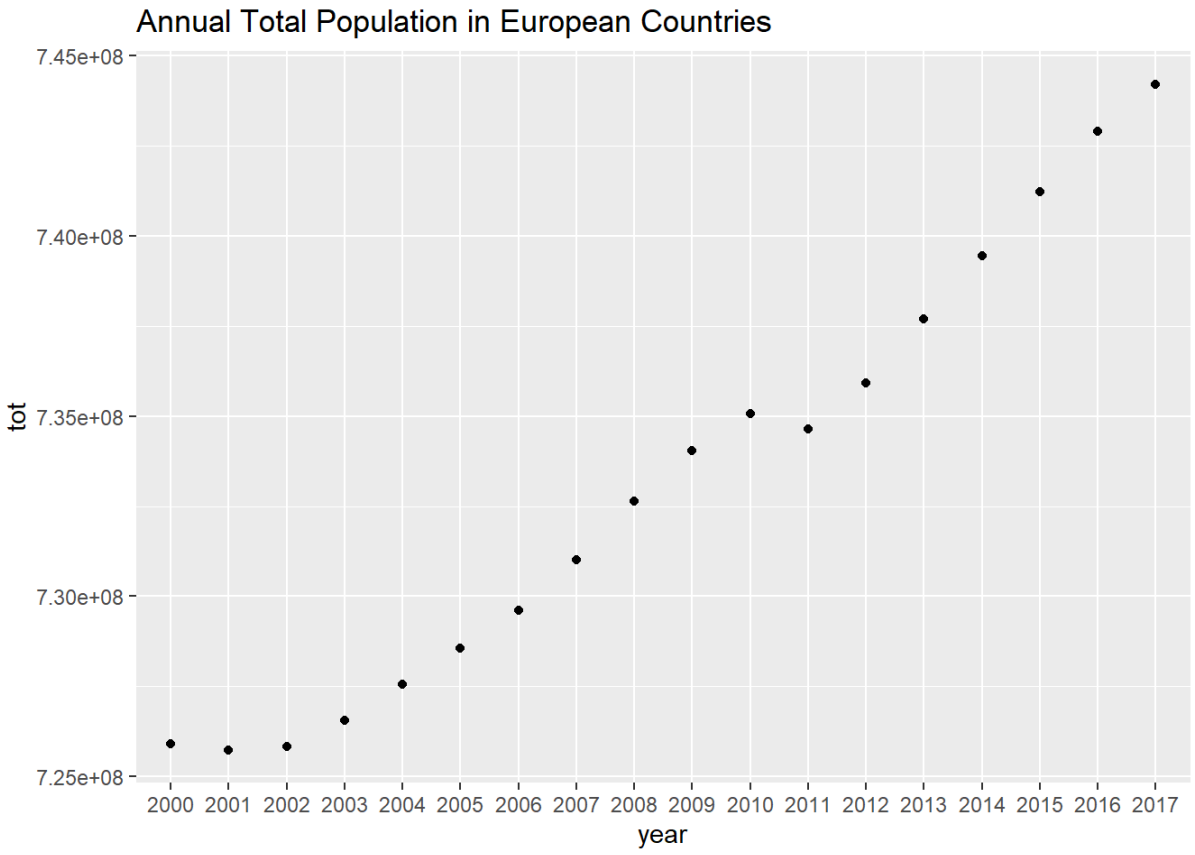
The total population in European countries in 2017 was 744 million poeple total.

2.4 (2 pts) Represent the annual total population in European countries between 2000 and 2017. Express the total population in million. How has European population changed over the years?

```
# total annual population in European countries
myeurope24 <- myeurope %>%
  group_by(year) %>%
  summarize(tot = sum(SP.POP.TOTL))
myeurope24
```

```
## # A tibble: 18 x 2
##    year        tot
##  * <chr>     <dbl>
##  1 2000  725904591
##  2 2001  725742262
##  3 2002  725820971
##  4 2003  726559119
##  5 2004  727548670
##  6 2005  728568822
##  7 2006  729603462
##  8 2007  731025779
##  9 2008  732649697
## 10 2009  734052512
## 11 2010  735080673
## 12 2011  734647571
## 13 2012  735932303
## 14 2013  737704677
## 15 2014  739454947
## 16 2015  741221225
## 17 2016  742901319
## 18 2017  744218594
```

```
ggplot(myeurope24, aes(x=year, y = tot)) + geom_point() + geom_smooth(method="lm") + ggtitle("Annual Total
 Population in European Countries")
```



The total population of Europe has steadily increased from 2000 to 2017 where in 2017 the total population was jsut under 744 million.

2.5 (2 pts) Create a new dataset `myeurope2017` by filtering the data for the `year` 2017, dropping the variable `year` , and creating a new variable `prop_urb` which is the proportion of urban population for each country. Which European country had the lowest proportion of urban population in 2017?

```
myeurope2017 <- myeurope %>%
  filter(year == 2017) %>%
  mutate(prop_urb = SP.URB.TOTL/SP.POP.TOTL) %>%
  arrange(desc(prop_urb))
myeurope2017
```

```
## # A tibble: 46 x 9
##    continent wb    country.name.en year  SP.URB.TOTL pop_urb_growth SP.POP.TOTL
##    <chr>     <chr> <chr>           <chr>       <dbl>          <dbl>       <dbl>
##  1 Europe    GIB   Gibraltar       2017        34571          0.473       34571
##  2 Europe    MCO   Monaco          2017        38695          0.508       38695
##  3 Europe    BEL   Belgium         2017     11140192          0.401    11372068
##  4 Europe    SMR   San Marino      2017        32422          0.759       33400
##  5 Europe    MLT   Malta           2017       439915          2.23       465292
##  6 Europe    ISL   Iceland         2017       320032          1.77       341284
##  7 Europe    NLD   Netherlands     2017     15604089          1.09     17132854
##  8 Europe    LUX   Luxembourg      2017       543862          3.25       599449
##  9 Europe    AND   Andorra         2017        67845         -0.520       76965
## 10 Europe    DNK   Denmark         2017      5063231          0.855     5769603
## # ... with 36 more rows, and 2 more variables: pop_growth <dbl>, prop_urb <dbl>
```

The European country Liechtenstein has the lowest proportion of urban population in 2017.

2.6 (1 pt) Using `dplyr` functions, find the top 3 countries in terms of their total population in 2017.

```
myeurope2017 %>%
  top_n(3,SP.POP.TOTL)
```

```
## # A tibble: 3 x 9
##   continent wb    country.name.en year  SP.URB.TOTL pop_urb_growth SP.POP.TOTL
##   <chr>     <chr> <chr>           <chr>       <dbl>          <dbl>       <dbl>
## 1 Europe    FRA   France          2017     53815732          0.715    67118648
## 2 Europe    DEU   Germany         2017     63890984          0.468    82695000
## 3 Europe    RUS   Russia          2017    107348258          0.278   144495044
## # ... with 2 more variables: pop_growth <dbl>, prop_urb <dbl>
```

The top 3 countries in terms of their total population in 2017 are Russia, France, and Germany.

Question 3: (6 pts)

When dealing with location data, we can actually visualize information on a map if we have geographic information such as latitude and longitude.

3.1 (1 pt) 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` with `glimpse()`. What variable could we use to join this dataset with `myeurope2017` dataset?

```
# geographic coordinates about countries in the world
# install.packages("maps")
mapWorld <- map_data("world")
mapWorld
```

```
##         long      lat group order      region subregion
## 1  -69.89912 12.45200     1     1       Aruba      <NA>
## 2  -69.89571 12.42300     1     2       Aruba      <NA>
## 3  -69.94219 12.43853     1     3       Aruba      <NA>
## 4  -70.00415 12.50049     1     4       Aruba      <NA>
## 5  -70.06612 12.54697     1     5       Aruba      <NA>
## 6  -70.05088 12.59707     1     6       Aruba      <NA>
## 7  -70.03511 12.61411     1     7       Aruba      <NA>
## 8  -69.97314 12.56763     1     8       Aruba      <NA>
## 9  -69.91181 12.48047     1     9       Aruba      <NA>
## 10 -69.89912 12.45200     1    10       Aruba      <NA>
## 12  74.89131 37.23164     2    12 Afghanistan      <NA>
## 13  74.84023 37.22505     2    13 Afghanistan      <NA>
## 14  74.76738 37.24917     2    14 Afghanistan      <NA>
## 15  74.73896 37.28564     2    15 Afghanistan      <NA>
## 16  74.72666 37.29072     2    16 Afghanistan      <NA>
## 17  74.66895 37.26670     2    17 Afghanistan      <NA>
##  [ reached 'max' / getOption("max.print") -- omitted 99322 rows ]
```

You could join the mapworld and myeurope2017 datasets based on country name.

3.2 (1 pt) We want to use a `left_join()` function to create a new dataset, `mymap`, to add data to the countries in `myeurope2017` dataset from `mapWorld` dataset, matching the two datasets based on the country name. If we then use `dplyr` functions, we can identify some missing values for `lat` and `long` in the new dataset. Indeed, some countries such as United Kingdom did not have a match. Why do you think this happened?

```
mymap <-  myeurope2017 %>%
  rename("region" = country.name.en) %>%
  left_join(mapWorld, by = 'region')
mymap
```

```
## # A tibble: 19,748 x 14
##    continent wb    region year  SP.URB.TOTL pop_urb_growth SP.POP.TOTL
##    <chr>     <chr> <chr>  <chr>       <dbl>          <dbl>       <dbl>
##  1 Europe    GIB   Gibra~ 2017        34571          0.473       34571
##  2 Europe    MCO   Monaco 2017        38695          0.508       38695
##  3 Europe    MCO   Monaco 2017        38695          0.508       38695
##  4 Europe    MCO   Monaco 2017        38695          0.508       38695
##  5 Europe    MCO   Monaco 2017        38695          0.508       38695
##  6 Europe    MCO   Monaco 2017        38695          0.508       38695
##  7 Europe    MCO   Monaco 2017        38695          0.508       38695
##  8 Europe    MCO   Monaco 2017        38695          0.508       38695
##  9 Europe    BEL   Belgi~ 2017     11140192          0.401    11372068
## 10 Europe    BEL   Belgi~ 2017     11140192          0.401    11372068
## # ... with 19,738 more rows, and 7 more variables: pop_growth <dbl>,
## #   prop_urb <dbl>, long <dbl>, lat <dbl>, group <dbl>, order <int>,
## #   subregion <chr>
```

Some countries don't have values for lat and long because their data does not match up to the mapWorld dataset.

3.3 (1 pt) To identify all countries that did not have an exact match, do an `anti_join()` and display only distinct country names. How many countries did not have an exact match? *Note: using anti_join() is a very usedul function to identify differences between datasets.*

```
trouble <- myeurope2017 %>%
  rename("region" = country.name.en) %>%
  anti_join(mapWorld, by = 'region')
trouble
```

```
## # A tibble: 5 x 9
##   continent wb    region year  SP.URB.TOTL pop_urb_growth SP.POP.TOTL pop_growth
##   <chr>     <chr> <chr>  <chr>       <dbl>          <dbl>       <dbl>      <dbl>
## 1 Europe    GIB   Gibra~ 2017        34571          0.473       34571     0.473
## 2 Europe    GBR   Unite~ 2017     54892898          0.958    66022273     0.648
## 3 Europe    CZE   Czech~ 2017      7803157          0.379    10591323     0.236
## 4 Europe    MKD   North~ 2017      1202983          0.415     2083160     0.0938
## 5 Europe    BIH   Bosni~ 2017      1679019          0.472     3507017    -0.279
## # ... with 1 more variable: prop_urb <dbl>
```

After using anti_join, it was discovered that 5 countries did not have an exact match.

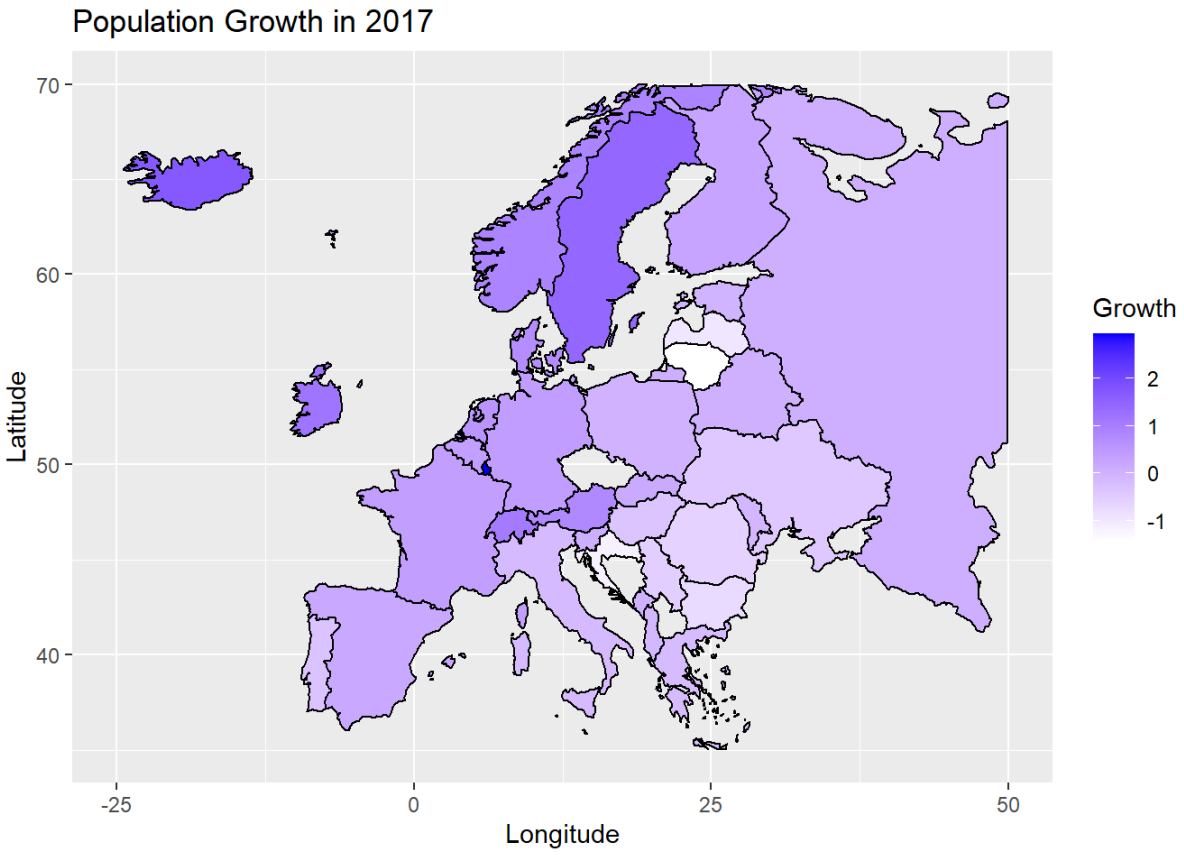
3.4 (1 pt) Joining datasets by variables containing names often leads to a mismatch because spelling can vary from one dataset to another. Sometimes we need to manually fix spelling in order to be able to match values. Consider the code given below. Replace the name of United Kingdom so that its name in `myeurope2017` dataset corresponds to the name given in `mapWorld` dataset. Following this code, add a pipe and use a `left_join()` function to create the new dataset, `mymap` , adding data to the countries in `myeurope` dataset from `mapWorld` dataset.

```
mapWorld <- mapWorld %>%
  rename('country.name.en' = region)

mymap <- myeurope2017 %>%
  mutate(country_clean = recode(country.name.en,
                                'United Kingdom' = 'UK',
                                'Bosnia & Herzegovina' = 'Bosnia and Herzegovina',
                                'Czechia' = 'Czech Republic',
                                'North Macedonia' = 'Macedonia')) %>%
    left_join(mapWorld, by = 'country.name.en')
mymap
```

```
## # A tibble: 19,748 x 15
##    continent wb    country.name.en year  SP.URB.TOTL pop_urb_growth SP.POP.TOTL
##    <chr>     <chr> <chr>           <chr>       <dbl>          <dbl>       <dbl>
##  1 Europe    GIB   Gibraltar       2017        34571          0.473       34571
##  2 Europe    MCO   Monaco          2017        38695          0.508       38695
##  3 Europe    MCO   Monaco          2017        38695          0.508       38695
##  4 Europe    MCO   Monaco          2017        38695          0.508       38695
##  5 Europe    MCO   Monaco          2017        38695          0.508       38695
##  6 Europe    MCO   Monaco          2017        38695          0.508       38695
##  7 Europe    MCO   Monaco          2017        38695          0.508       38695
##  8 Europe    MCO   Monaco          2017        38695          0.508       38695
##  9 Europe    BEL   Belgium         2017     11140192          0.401    11372068
## 10 Europe    BEL   Belgium         2017     11140192          0.401    11372068
## # ... with 19,738 more rows, and 8 more variables: pop_growth <dbl>,
## #   prop_urb <dbl>, country_clean <chr>, long <dbl>, lat <dbl>, group <dbl>,
## #   order <int>, subregion <chr>
```

3.5 (2 pts) Let's visualize how population growth varies across European countries. Install the package `ggmap`, call the corresponding library, and use the R code provided below. Try to identify what each component of the graph does by completing the code with comments.

```
# install.packages("ggmap")
# library(ggmap)

mymap %>%
  ggplot(aes(x=long, y=lat, group = group, fill = pop_growth)) +
  # the map is displayed on the grid with a color gradient saying dark blue is less and light blue is more
 growth
  geom_polygon(colour = "black") +
  # the color of the map changes from the default gradient of dark navy to light blue to a gradient of ligh
t purple to
  # bright blue respectively aka the gradient color was changed
  scale_fill_gradient(low = "white", high = "blue", guide="colorbar") +
  # Graph titles and labels are added for the legend, title, x and y axis
  labs(fill = "Growth", title = "Population Growth in 2017", x="Longitude", y="Latitude") +
  # the grid's coordinates are resized so the map on shows Europe
  xlim(-25,50) + ylim(35,70)
```




```
##        sysname        release        version       nodename        machine
##      "Windows"      "10 x64"  "build 18363"     "ROSE-XPS"      "x86-64"
##          login           user effective_user
##        "roseh"        "roseh"        "roseh"
```