Problem Set 1

Answer Key

ECON 480 — Fall 2020

Answers may be longer than I would deem sufficient on an exam. Some might vary slightly based on points of interest, examples, or personal experience. These suggested answers are designed to give you both the answer and a short explanation of *why* it is the answer.

The Popularity of Baby Names

Install and load the package `babynames`. Get help for `?babynames` to see what the data includes.

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
# install for first use
# install.packages("babynames")

# load package
library(babynames)

# explore help
# ?babynames
```

Question 1

Part A

What are the top 5 boys names for 2017, and what *percent* of overall names is each?

```
# save as a new tibble
top_5_boys_2017 <- babynames %>% # take data
  filter(sex=="M", # filter by males
         year==2017) %>% # and for 2007
  arrange(desc(n)) %>% # arrange in largest-to-smallest order of n (number)
  slice(1:5) %>% # optional, look only at first 5 rows; head(., n=5) also works
```

```
  mutate(percent = round(prop*100, 2)) # also optional, make a percent variable rounded to 2 decimals

# look at our new tibble
top_5_boys_2017
```

```
## # A tibble: 5 x 6
##    year sex   name        n    prop percent
##   <dbl> <chr> <chr>   <int>   <dbl>   <dbl>
## 1  2017 M     Liam    18728 0.00954    0.95
## 2  2017 M     Noah    18326 0.00933    0.93
## 3  2017 M     William 14904 0.00759    0.76
## 4  2017 M     James   14232 0.00725    0.72
## 5  2017 M     Logan   13974 0.00712    0.71
```

The top 5 names are

1. Liam (0.95%)
2. Noah (0.93%)
3. William (0.76%)
4. James (0.72%)
5. Logan (0.71%)

Part B

What are the top 5 *girls* names, and what *percent* of overall names is each?

```
# save as a new tibble
top_5_girls_2017 <- babynames %>% # take data
  filter(sex=="F", # filter by females
         year==2017) %>% # and for 2007
  arrange(desc(n)) %>% # arrange in largest-to-smallest order of n (number)
  slice(1:5) %>% # optional, look only at first 5 rows; head(., n=5) also works
  mutate(percent = round(prop*100, 2)) # also optional, make a percent variable rounded to 2 decimals

# look at our new tibble
top_5_girls_2017
```

```
## # A tibble: 5 x 6
##    year sex   name         n    prop percent
##   <dbl> <chr> <chr>    <int>   <dbl>   <dbl>
## 1  2017 F     Emma     19738 0.0105     1.05
## 2  2017 F     Olivia   18632 0.00994    0.99
## 3  2017 F     Ava      15902 0.00848    0.85
## 4  2017 F     Isabella 15100 0.00805    0.81
## 5  2017 F     Sophia   14831 0.00791    0.79
```

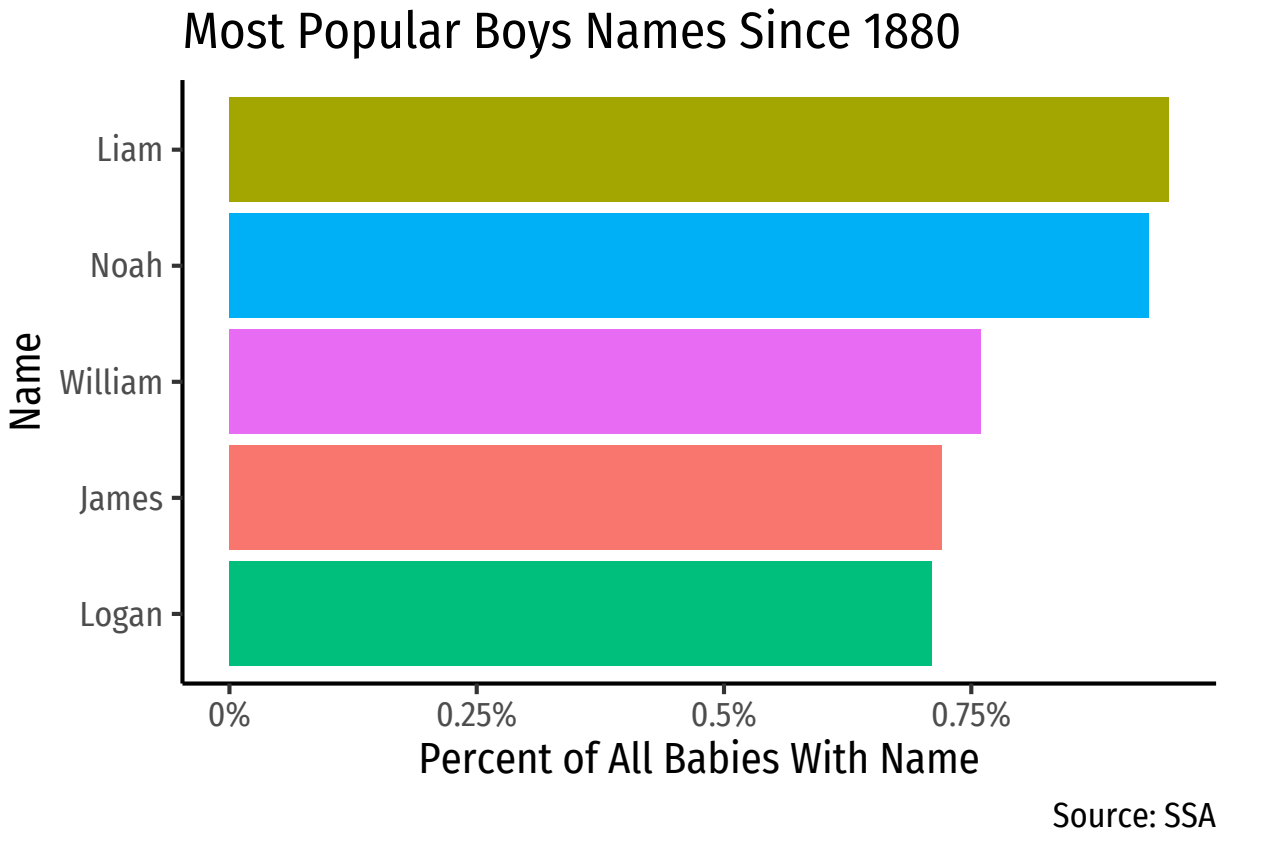
The top 5 names are

1. Emma (1.05%)
2. Olivia (0.99%)
3. Ava (0.85%)
4. Isabella (0.81%)
5. Sophia (0.79%)

Question 2

Make two barplots, of these top 5 names, one for each sex. Map aesthetics `x` to `name` and `y` to `prop`[1] and use `geom_col` (since you are declaring a specific `y`, otherwise you could just use `geom_bar()` and just an `x`.)

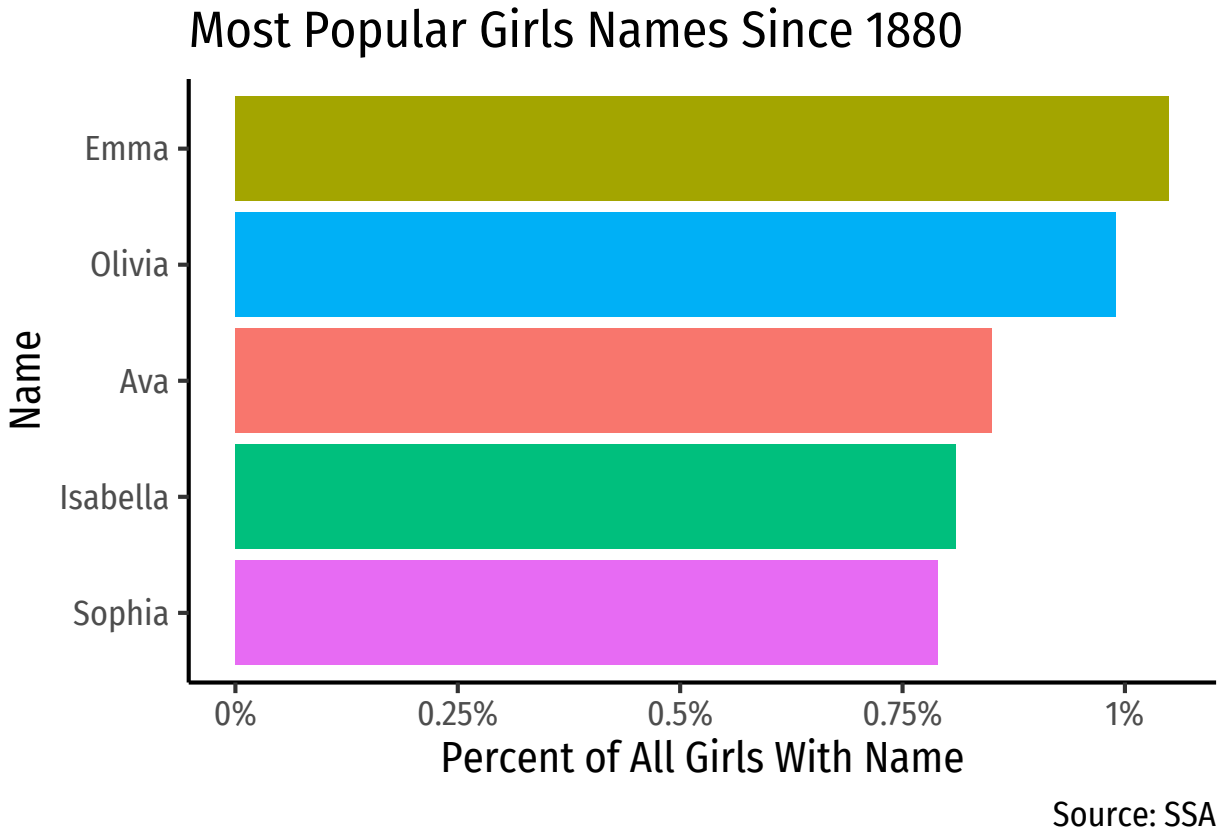
```
ggplot(data = top_5_boys_2017)+
  aes(x = reorder(name, n), #note this reorders the x variable from small to large n
      y = percent, # you can use prop if you didn't make a percent variable
      fill = name)+ # optional color!
  geom_col()+

  # now I'm just making it pretty
  scale_y_continuous(labels=function(x)paste(x,"%",sep=""))+ # optional, add percent signs
      labs(x = "Name",
         y = "Percent of All Babies With Name",
         title = "Most Popular Boys Names Since 1880",
         fill = "Boy's Name",
         caption = "Source: SSA")+
    theme_classic(base_family = "Fira Sans Condensed", base_size=16)+
  coord_flip()+ # rotate axes!
  theme(legend.position = "") # hide legend
```




```
ggplot(data = top_5_girls_2017)+
  aes(x = reorder(name, n), #note this reorders the x variable from small to large n
      y = percent, # you can use prop if you didn't make a percent variable
      fill = name)+ # optional color!
```

[1] Or `percent`, if you made that variable, as I did.

```
  geom_col()+
  # now I'm just making it pretty
  scale_y_continuous(labels=function(x)paste(x,"%",sep=""))+ # optional, add percent signs
      labs(x = "Name",
         y = "Percent of All Girls With Name",
         title = "Most Popular Girls Names Since 1880",
         fill = "Girl's Name",
         caption = "Source: SSA")+
    theme_classic(base_family = "Fira Sans Condensed", base_size=16)+
  coord_flip()+ # rotate axes!
  theme(legend.position = "") # hide legend
```



Question 3

Find your name.[2] `count` by `sex` how many babies since 1880 were named your name.[3] Also add a variable for the percent of each sex.

```
babynames %>%
  filter(name == "Ryan") %>%
  count(sex, wt=n) %>%
  mutate(percent = round((n/sum(n)*100),2))
```

```
## # A tibble: 2 x 3
##   sex        n percent
```

[2] If your name isn't in there :(, pick a random name.

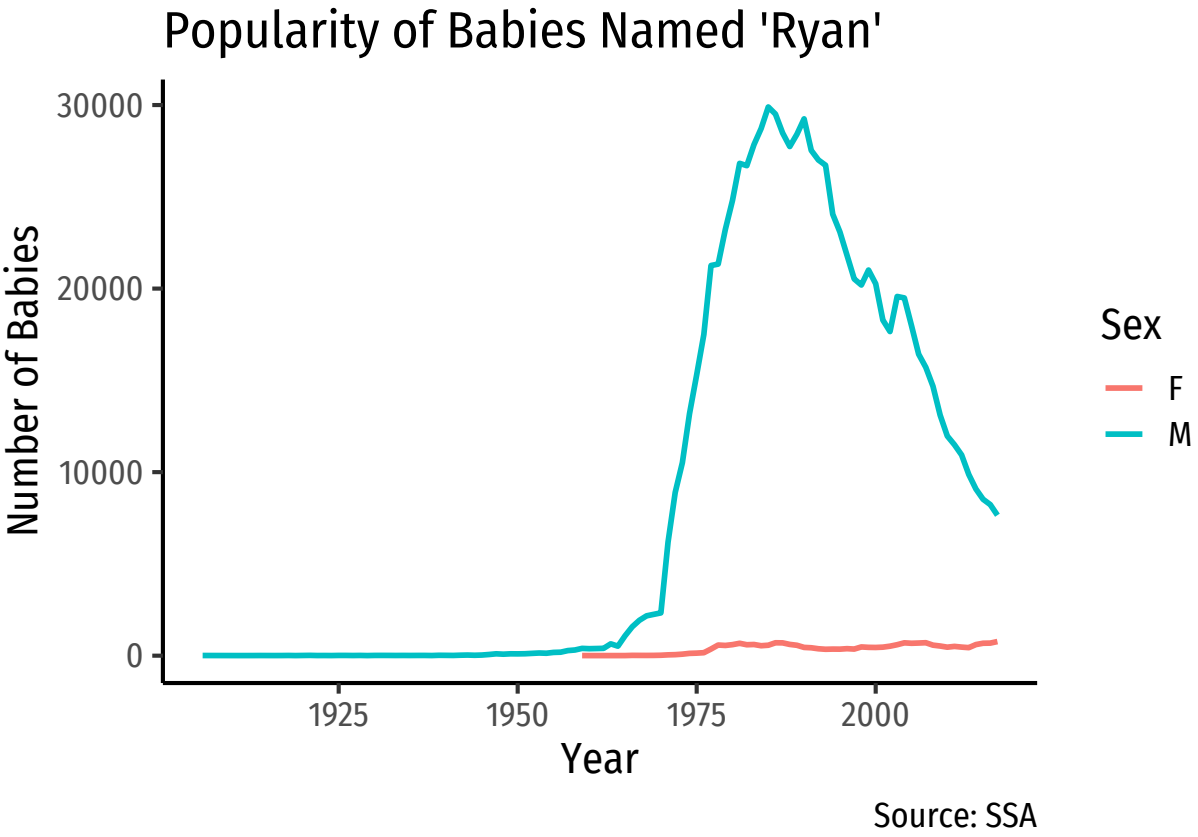
[3] Hint: if you do this, you'll get the number of *rows* (years) there are in the data. You want to add the number of babies in each row (`n`), so inside `count`, add `wt=n` to weight the count by `n`.

```
##   <chr>  <int>   <dbl>
## 1 F      22910    2.42
## 2 M     924877   97.6
```

Question 4

Make a line graph of the number of babies with your name over time, colored by sex.

```
# note here I'm going to wrangle the data and then pipe it directly into ggplot
# you can wrangle the data and save it as a different tibble, then use THAT tibble
# for your (data = ...) command in ggplot

# first wrangle data
babynames %>%
  filter(name == "Ryan") %>%

  # now we pipe into ggplot
  ggplot(data = .)+ # the "." is a placeholder for the stuff above!
  aes(x = year,
      y = n,
      color = sex)+
  geom_line(size=1)+
  labs(x = "Year",
       y = "Number of Babies",
       title = "Popularity of Babies Named 'Ryan'",
       color = "Sex",
       caption = "Source: SSA")+
    theme_classic(base_family = "Fira Sans Condensed", base_size=16)
```



Question 5

Part A

Make a table of the most common name for boys by year between 1980-2017.[4]

```
babynames %>%
  group_by(year) %>% # we want one observation per year
  filter(sex == "M",
         year>1979) %>% # or >==1980
  arrange(desc(n))%>% # start with largest n first
  slice(1) # take first row only
```

```
## # A tibble: 38 x 5
## # Groups:   year [38]
##     year sex   name        n   prop
##    <dbl> <chr> <chr>   <int>  <dbl>
##  1  1980 M     Michael 68693 0.0370
##  2  1981 M     Michael 68765 0.0369
##  3  1982 M     Michael 68228 0.0362
##  4  1983 M     Michael 67995 0.0365
##  5  1984 M     Michael 67736 0.0361
##  6  1985 M     Michael 64906 0.0337
##  7  1986 M     Michael 64205 0.0334
##  8  1987 M     Michael 63647 0.0326
```

[4]Hint: once you've got all the right conditions, you'll get a table with a lot of data. You only want to `slice` the `1`st row for each table.

```
##  9  1988 M     Michael 64133 0.0320
## 10  1989 M     Michael 65382 0.0312
## # ... with 28 more rows
```

Part B

Now do the same for girls.

```
babynames %>%
  group_by(year) %>% # we want one observation per year
  filter(sex == "F",
         year>1979) %>% # or >==1980
  arrange(desc(n))%>% # start with largest n first
  slice(1) # take first row only
```

```
## # A tibble: 38 x 5
## # Groups:   year [38]
##     year sex   name         n   prop
##    <dbl> <chr> <chr>    <int>  <dbl>
##  1  1980 F     Jennifer 58376 0.0328
##  2  1981 F     Jennifer 57049 0.0319
##  3  1982 F     Jennifer 57115 0.0315
##  4  1983 F     Jennifer 54342 0.0304
##  5  1984 F     Jennifer 50561 0.0280
##  6  1985 F     Jessica  48346 0.0262
##  7  1986 F     Jessica  52674 0.0285
##  8  1987 F     Jessica  55991 0.0299
##  9  1988 F     Jessica  51538 0.0268
## 10  1989 F     Jessica  47885 0.0240
## # ... with 28 more rows
```

Question 6

Now let's graph the evolution of the most common names since 1880.

Part A

First, find out what are the top 10 *overall* most popular names for boys and for girls. You may want to create two vectors, each with these top 5 names.

```
babynames %>%
  group_by(name) %>% # we want one row per name
  filter(sex=="M") %>%
  summarize(total=sum(n)) %>% # add upp all of the n's for all years for each name
  arrange(desc(total)) %>% # list largest total first
  slice(1:5)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 5 x 2
##   name      total
##   <chr>     <int>
## 1 James   5150472
## 2 John    5115466
```

```
## 3 Robert  4814815
## 4 Michael 4350824
## 5 William 4102604
```

```
# make a vector of the names (we'll need this for our graph below)
top_boys_names<-c("James", "John", "Robert", "Michael", "William")

# you could alternatively add a command,
# %>% pull(name) to the first chunk of code,
# and it would do the same thing, but we'd want to save it,
# for example:

babynames %>%
  group_by(name) %>% # we want one row per name
  filter(sex=="M") %>%
  summarize(total=sum(n)) %>% # add upp all of the n's for all years for each name
  arrange(desc(total)) %>% # list largest total first
  slice(1:5) %>%
  pull(name)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## [1] "James"   "John"    "Robert"  "Michael" "William"
```

```
babynames %>%
  group_by(name) %>% # we want one row per name
  filter(sex=="F") %>%
  summarize(total=sum(n)) %>% # add upp all of the n's for all years for each name
  arrange(desc(total)) %>% # list largest total first
  slice(1:5)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 5 x 2
##   name        total
##   <chr>       <int>
## 1 Mary      4123200
## 2 Elizabeth 1629679
## 3 Patricia  1571692
## 4 Jennifer  1466281
## 5 Linda     1452249
```

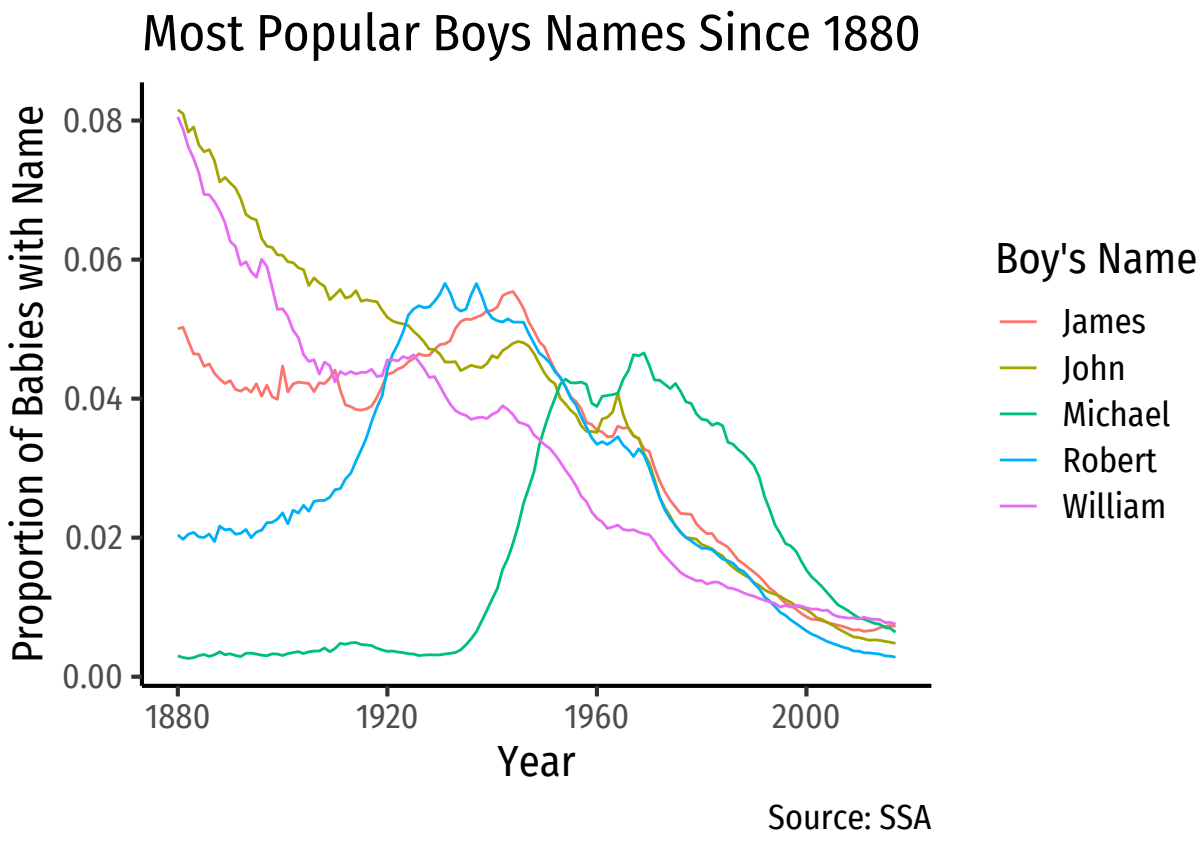
```
# make a vector of the names (we'll need this for our graph below)
top_girls_names<-c("Mary", "Elizabeth", "Patricia", "Jennifer", "Linda")
```

Part B

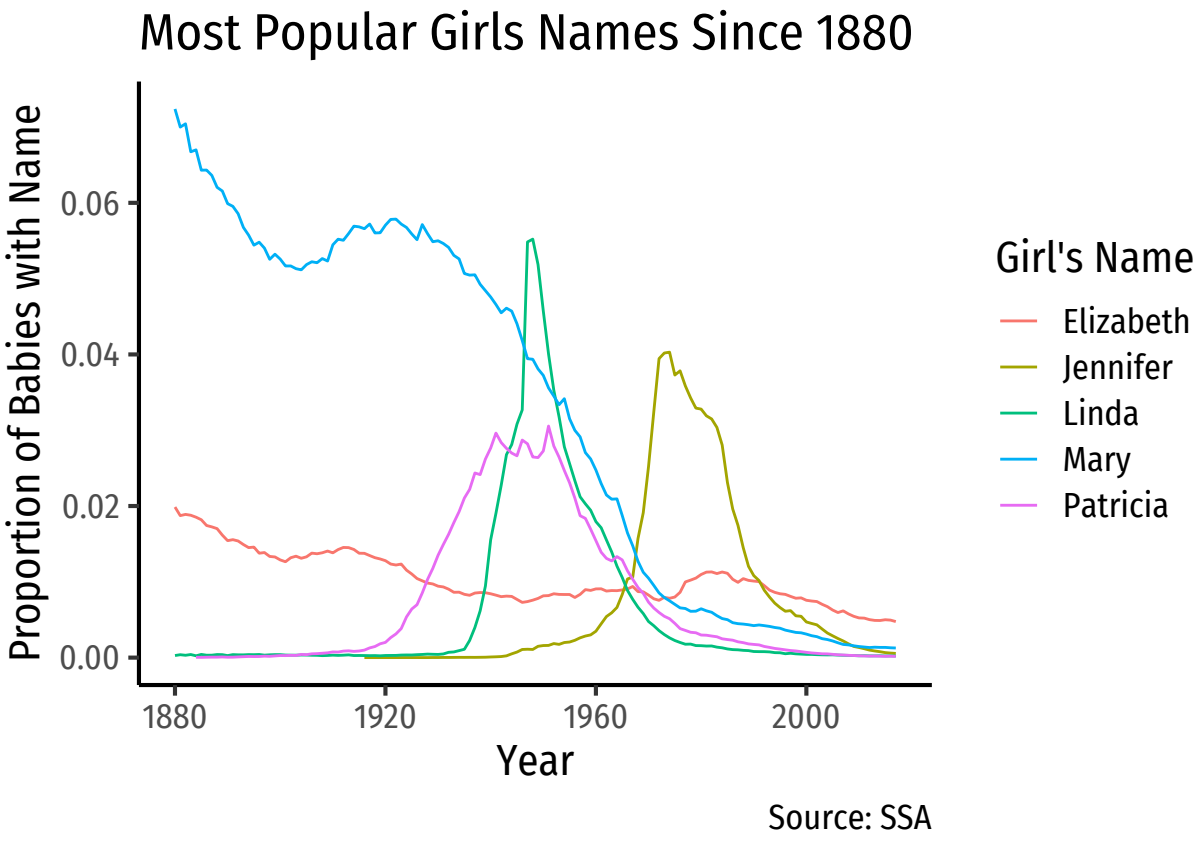
Now make two linegraphs of these 5 names over time, one for boys, and one for girls.

```
babynames %>%
  group_by(year) %>%
  filter(sex == "M",
         name %in% top_boys_names) %>%
  ggplot(data = .,
         aes(x = year,
             y = prop,
```

```
            color = name))+
  geom_line()+
      labs(x = "Year",
         y = "Proportion of Babies with Name",
         title = "Most Popular Boys Names Since 1880",
         color = "Boy's Name",
         caption = "Source: SSA")+
    theme_classic(base_family = "Fira Sans Condensed", base_size=16)
```




```
babynames %>%
  group_by(year) %>%
  filter(sex == "F",
         name %in% top_girls_names) %>%
  ggplot(data = .,
         aes(x = year,
             y = prop,
             color = name))+
  geom_line()+
    labs(x = "Year",
         y = "Proportion of Babies with Name",
         title = "Most Popular Girls Names Since 1880",
         color = "Girl's Name",
         caption = "Source: SSA")+
    theme_classic(base_family = "Fira Sans Condensed", base_size=16)
```



Question 7

***Bonus (hard!)*: What are the 10 most common "gender-neutral" names?**[5]

There's a lot to this, so I'll break this up step by step and show you what happens at each major step.

We want to find the names where 48% to 52% of the babies with the name are male, as I defined in the footnote. First let's `mutate` a variable to figure out how many babies with a particular name are male.

To do this, we'll need to make a two variables to count the number of `male`s and `female`s of each name each year. We'll use the `ifelse()` function for each:

1. Make a `male` variable where, for each name in each year, if `sex=="M"`, then count the number of males (`n`) that year, otherwise set it equal to `0`.
2. Make a `female` variable where, for each name in each year, if `sex=="F"`, then count the number of females (`n`) that year, otherwise set it equal to `0`.

```
babynames %>%
  mutate(male = ifelse(sex == "M", n, 0),
         female = ifelse(sex == "F", n, 0))
```

```
## # A tibble: 1,924,665 x 7
##     year sex   name           n   prop  male female
##    <dbl> <chr> <chr>      <int>  <dbl> <dbl>  <dbl>
##  1  1880 F     Mary        7065 0.0724     0   7065
##  2  1880 F     Anna        2604 0.0267     0   2604
##  3  1880 F     Emma        2003 0.0205     0   2003
```

[5] This is hard to define. For our purposes, let's define this as names where between 48 and 52% of the babies with the name are Male.

```
##  4  1880 F     Elizabeth  1939 0.0199     0   1939
##  5  1880 F     Minnie     1746 0.0179     0   1746
##  6  1880 F     Margaret   1578 0.0162     0   1578
##  7  1880 F     Ida        1472 0.0151     0   1472
##  8  1880 F     Alice      1414 0.0145     0   1414
##  9  1880 F     Bertha     1320 0.0135     0   1320
## 10  1880 F     Sarah      1288 0.0132     0   1288
## # ... with 1,924,655 more rows
```

Now with this variable, we want to count the total number of males and females with each name over the entire dataset. Let's first `group_by(name)` so we'll get one row for every name. We will `summarize()` and take the `sum` of our `male` and of our `female` variables.

```
babynames %>%
  mutate(male = ifelse(sex == "M", n, 0),
         female = ifelse(sex == "F", n, 0)) %>%
  group_by(name) %>%
    summarize(Male = sum(male),
              Female = sum(female))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 97,310 x 3
##    name       Male Female
##    <chr>     <dbl>  <dbl>
##  1 Aaban       107      0
##  2 Aabha         0     35
##  3 Aabid        10      0
##  4 Aabir         5      0
##  5 Aabriella     0     32
##  6 Aada          0      5
##  7 Aadam       254      0
##  8 Aadan       130      0
##  9 Aadarsh     199      0
## 10 Aaden      4653      5
## # ... with 97,300 more rows
```

Now, we want to figure out what *fraction* of each name is Male or Female. It doesn't matter which we do here, I'll do Male. `mutate()` a new variable I'll call `perc_male` for the percent of the name being for Male babies. It takes the summed variables we made before, and takes the fraction that are Male, multiplying by 100 to get percents (which isn't necessary, but is easy to read).

```
babynames %>%
  mutate(male = ifelse(sex == "M", n, 0),
         female = ifelse(sex == "F", n, 0)) %>%
  group_by(name) %>%
    summarize(Male = sum(male),
              Female = sum(female))%>%
  mutate(perc_male = (Male/(Male+Female)*100))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 97,310 x 4
##    name       Male Female perc_male
##    <chr>     <dbl>  <dbl>     <dbl>
##  1 Aaban       107      0     100
##  2 Aabha         0     35       0
```

```
##  3 Aabid        10      0     100
##  4 Aabir         5      0     100
##  5 Aabriella     0     32       0
##  6 Aada          0      5       0
##  7 Aadam       254      0     100
##  8 Aadan       130      0     100
##  9 Aadarsh     199      0     100
## 10 Aaden      4653      5      99.9
## # ... with 97,300 more rows
```

Right now, it's still in alphabetical order. We want to arrange it by `perc_male`, and more importantly, we want `perc_male` to be between 48 and 52, so let's `filter` accordingly:

```
babynames %>%
  mutate(male = ifelse(sex == "M", n, 0),
         female = ifelse(sex == "F", n, 0)) %>%
  group_by(name) %>%
    summarize(Male = sum(male),
              Female = sum(female))%>%
  mutate(perc_male = (Male/(Male+Female)*100)) %>%
  arrange(perc_male) %>%
  filter(perc_male > 48,
         perc_male < 52)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 266 x 4
##    name        Male Female perc_male
##    <chr>      <dbl>  <dbl>     <dbl>
##  1 Demetrice   1623   1754      48.1
##  2 Shenan        25     27      48.1
##  3 Yael        3162   3414      48.1
##  4 Harlo        164    177      48.1
##  5 Daylyn       202    218      48.1
##  6 Oluwatosin   139    150      48.1
##  7 Chaning       13     14      48.1
##  8 Kirin        351    378      48.1
##  9 Odera         13     14      48.1
## 10 Jireh        644    693      48.2
## # ... with 256 more rows
```

This gives us a lot of names, all falling between 48% and 52% male. But we want the most popular names that are in this range. So let's finally `mutate` a new variable called `total` that simply adds the number of `Male` and `Female` babies with a name. Then let's `arrange` our results by `desc(total)` to get the largest first, and then `slice(1:10)` to get the top 10 only.

```
babynames %>%
  mutate(male = ifelse(sex == "M", n, 0),
         female = ifelse(sex == "F", n, 0)) %>%
  group_by(name) %>%
    summarize(Male = sum(male),
              Female = sum(female))%>%
  mutate(perc_male = (Male/(Male+Female)*100)) %>%
  arrange(perc_male) %>%
  filter(perc_male > 48,
         perc_male < 52) %>%
  mutate(total = Male+Female) %>%
```

```
  arrange(desc(total)) %>%
  slice(1:10)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 10 x 5
##    name      Male Female perc_male total
##    <chr>    <dbl>  <dbl>     <dbl> <dbl>
##  1 Kerry    49596  48534      50.5 98130
##  2 Robbie   20863  22264      48.4 43127
##  3 Justice  17080  15782      52.0 32862
##  4 Blair    14470  14195      50.5 28665
##  5 Kris     13982  13490      50.9 27472
##  6 Elisha   13330  13599      49.5 26929
##  7 Unknown   9307   9416      49.7 18723
##  8 Mckinley  9389   8955      51.2 18344
##  9 Baby      6078   5871      50.9 11949
## 10 Santana   4651   4952      48.4  9603
```

Political and Economic Freedom Around the World

For the remaining questions, we'll look at the relationship between Economic Freedom and Political Freedom in countries around the world today. Our data for economic freedom comes from the Fraser Institute, and our data for political freedom comes from Freedom House.

Question 8

Download these two datasets that I've cleaned up a bit:[6]

- `econfreedom.csv`
- `freedomhouse2018.csv`

Load them with `df<-read_csv("name_of_the_file.csv")` and save one as `econfreedom` and the other as `polfreedom`. Look at each `tibble` you've created.

I am creating this document for/from the website, so these are all stored in a folder called `data`, one folder up from my current folder, `homeworks`. To get there, I go up one folder (`..`) and move to `data`, where these `csv` files are stored.

I suggest you either keep the data in the same folder as your `R` working directory (always check with `getwd()`), or create an R Project and store the data files in that same folder.

```
# import data with read_csv from readr

# note these file paths will be different for you
polfreedom<-read_csv("../data/freedomhouse2018.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Country/Territory` = col_character(),
##   Status = col_character()
## )
```

[6] If you want, try downloading them from the websites yourself!

```
## See spec(...) for full column specifications.
```

```
econfreedom<-read_csv("../data/econfreedom.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   X1 = col_double(),
##   Country = col_character(),
##   ISO = col_character(),
##   ef = col_double(),
##   gdp = col_double(),
##   continent = col_character()
## )
```

```
# look at each dataframe
polfreedom
```

```
## # A tibble: 209 x 40
##    `Country/Territ~ Status `PR Rating` `CL Rating`    A1    A2    A3     A    B1
##    <chr>            <chr>         <dbl>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 Abkhazia         PF                4           5     3     2     1     6     2
##  2 Afghanistan      NF                5           6     1     0     1     2     2
##  3 Albania          PF                3           3     3     3     2     8     3
##  4 Algeria          NF                6           5     1     1     1     3     1
##  5 Andorra          F                 1           1     4     4     4    12     4
##  6 Angola           NF                6           6     0     2     1     3     2
##  7 Antigua and Bar~ F                 2           2     4     4     4    12     3
##  8 Argentina        F                 2           2     4     4     3    11     4
##  9 Armenia          PF                5           4     1     1     2     4     2
## 10 Australia        F                 1           1     4     4     4    12     4
## # ... with 199 more rows, and 31 more variables: B2 <dbl>, B3 <dbl>, B4 <dbl>,
## #   B <dbl>, C1 <dbl>, C2 <dbl>, C3 <dbl>, C <dbl>, `Add Q` <dbl>, PR <dbl>,
## #   D1 <dbl>, D2 <dbl>, D3 <dbl>, D4 <dbl>, D <dbl>, E1 <dbl>, E2 <dbl>,
## #   E3 <dbl>, E <dbl>, F1 <dbl>, F2 <dbl>, F3 <dbl>, F4 <dbl>, F <dbl>,
## #   G1 <dbl>, G2 <dbl>, G3 <dbl>, G4 <dbl>, G <dbl>, CL <dbl>, Total <dbl>
```

```
econfreedom
```

```
## # A tibble: 112 x 6
##       X1 Country    ISO      ef    gdp continent
##    <dbl> <chr>      <chr> <dbl>  <dbl> <chr>
##  1     1 Albania    ALB    7.4   4543. Europe
##  2     2 Algeria    DZA    5.15  4784. Africa
##  3     3 Angola     AGO    5.08  4153. Africa
##  4     4 Argentina  ARG    4.81 10502. Americas
##  5     5 Australia  AUS    7.93 54688. Oceania
##  6     6 Austria    AUT    7.56 47604. Europe
##  7     7 Bahrain    BHR    7.6  22348. Asia
##  8     8 Bangladesh BGD    6.35   973. Asia
##  9     9 Belgium    BEL    7.51 45181. Europe
## 10    10 Benin      BEN    6.22   805. Africa
## # ... with 102 more rows
```

Question 9

The `polfreedom` dataset is still a bit messy. Let's overwrite it (or assign to something like `polfreedom2`) and select `Country/Territory` and `Total` (total freedom score) and rename `Country.Territory` to `Country`.

```
polfreedom<-polfreedom %>%
  select(`Country/Territory`, Total) %>%
  rename(Country=`Country/Territory`)
```

Question 10

Now we can try to merge these two datasets into one. Since they both have `Country` as a variable, we can merge these tibbles using `left_join(econfreedom, polfreedom, by="Country")`[7] and save this as a new tibble (something like `freedom`).

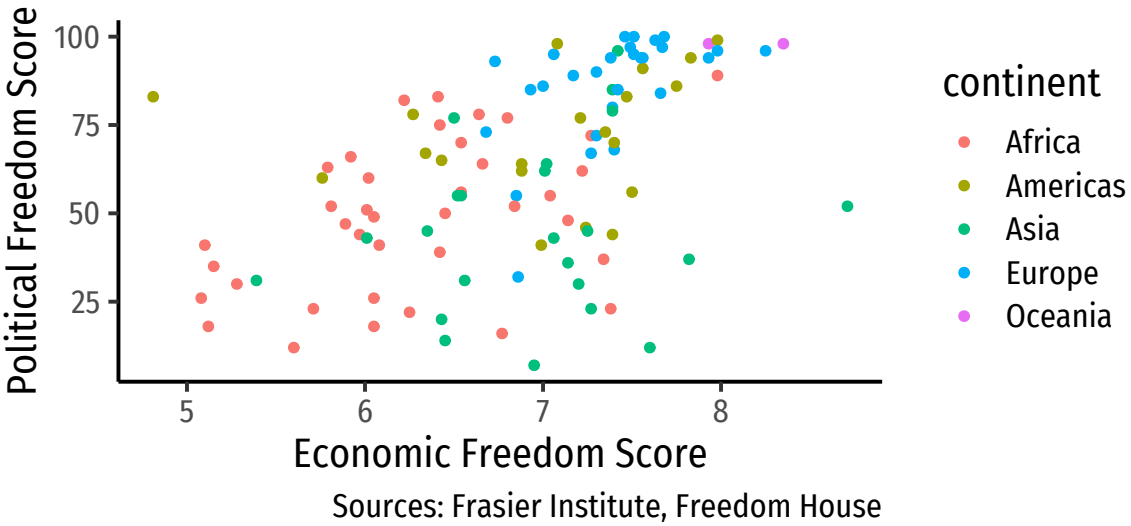
This one is a bit advanced to explain (but see the last few slides of 1.5 for more), so just copy what I gave you!

```
freedom<-left_join(econfreedom, polfreedom, by="Country")
```

Question 11

Now make a scatterplot of Political Freedom (`total`)[8] as `y` on Economic Freedom (`ef`) as `x` and `color` by `continent`.

```
## Warning: Removed 1 rows containing missing values (geom_point).
```



Question 12

Let's do this again, but highlight some key countries. Pick three countries, and make a new tibble from `freedom` that is only the observations of those countries. Additionally, *install* and *load* a packaged called `ggrepel`[9] Next, redo your plot from question 11, but now add a layer:

[7] Note, if you saved as something else in question 9., use that instead of `polfreedom`!
[8] Feel free to `rename` these!
[9] This automatically adjusts labels so they don't cover points on a plot!