Session #3 - Exercises

Adrian C Lo

04/03/2020

INSTRUCTIONS

There are different segment with emphasis on one part of the data process, each covering questions that increase in difficulty. Fill in the answer within the code chunk. When you wish to test the code chunk, press the *green play button* on the right side of the code chunk to see your output.

Cheatsheets are provided where you can find clues for solving the questions. In more difficult cases, actual hints will be given as to which functions are required. You can access further information with *?x* where x is the function name, e.g. ?print().

Also check this page, specifically sections 8.3 and 10.2 for additional help.

We encourage to write code in the spirit of tidyverse, which improves readability. For instance, compare the two codes below that filter the mtcars dataset for automatic cars, then categorizes them per cyl capaticy, and compute the average horse power.

```
suppressPackageStartupMessages( library(dplyr) )

# base code
summarise(group_by(filter(mtcars, am == 1), cyl), hp = mean(hp))

# tidyverse code
mtcars %>%
  filter(am == 1) %>%
  group_by(cyl) %>%
  summarise(hp = mean(hp))
```

EXERCISES

data manipulations {dplyr}

```
suppressPackageStartupMessages( library(dplyr) )
suppressPackageStartupMessages( library(tidyr) )
suppressPackageStartupMessages( library(lubridate) )
```

In the following questions, you will perform exploratory analysis on the coronavirus. This dataset is contained in the like-named {coronavirus} library.

```
# devtools::install_github("RamiKrispin/coronavirus")
suppressPackageStartupMessages( library(coronavirus) )
```

1. Worldwide, how many confirmed cases of coronavirus have been found?

```
coronavirus %>%
    filter(type == "confirmed") %>%
    summarise(worldwide_confirmed = sum(cases))
```

```
## # A tibble: 1 x 1
##   worldwide_confirmed
##                 <int>
## 1               88371
```

2. Worldwide, how many people died from coronavirus?

```
coronavirus %>%
    filter(type == "death") %>%
    summarise(worldwide_confirmed = sum(cases))
```

```
## # A tibble: 1 x 1
##   worldwide_confirmed
##                 <int>
## 1                2996
```

3. Which are the top 5 countries with the most cases of confirmed coronavirus?

```
coronavirus %>%
    filter(type == "confirmed") %>%
    group_by(Country.Region) %>%
    summarise(confirmed = sum(cases)) %>%
    arrange(desc(confirmed)) %>%
    head(5)
```

```
## # A tibble: 5 x 2
##   Country.Region confirmed
##   <chr>              <int>
## 1 Mainland China     79826
## 2 South Korea         3736
## 3 Italy               1694
## 4 Iran                 978
## 5 Others               705
```

4. From which country is the last confirmed case?

```
coronavirus %>%
    filter(type == "confirmed") %>%
    arrange(desc(date)) %>%
    head(1)
```

```
## # A tibble: 1 x 7
##   Province.State Country.Region   Lat  Long date       cases type
##   <chr>          <chr>          <dbl> <dbl> <date>     <int> <chr>
## 1 ""             Armenia         40.1  45.0 2020-03-01     1 confirmed
```

5. From which country were the latest recovered cases?

```
coronavirus %>%
    filter(type == "recovered") %>%
    arrange(desc(date)) %>%
    head(1)
```

```
## # A tibble: 1 x 7
##   Province.State Country.Region   Lat  Long date       cases type
##   <chr>          <chr>          <dbl> <dbl> <date>     <int> <chr>
## 1 ""             Iran              32    53 2020-03-01    52 recovered
```

6. When and where were the most confirmed cases detected on a single day?

```
coronavirus %>%
    filter(type == "confirmed") %>%
    arrange(desc(cases)) %>%
    head(1)
```

```
## # A tibble: 1 x 7
##   Province.State Country.Region   Lat  Long date       cases type
##   <chr>          <chr>          <dbl> <dbl> <date>     <int> <chr>
## 1 Hubei          Mainland China  31.0  112. 2020-02-13 14840 confirmed
```

7. Were there any false positive confirmed cases?

```
coronavirus %>%
    filter(type == "confirmed") %>%
    filter(cases < 0)
```

```
## # A tibble: 8 x 7
##   Province.State        Country.Region   Lat   Long date       cases type
##   <chr>                 <chr>          <dbl>  <dbl> <date>     <int> <chr>
## 1 ""                    Japan           36    138   2020-01-23    -1 confi~
## 2 Queensland            Australia      -28.0  153.  2020-01-31    -1 confi~
## 3 Queensland            Australia      -28.0  153.  2020-02-02    -1 confi~
## 4 ""                    Japan           36    138   2020-02-07   -20 confi~
## 5 Lackland, TX (From D~ US              29.4  -98.6 2020-02-24    -2 confi~
## 6 Omaha, NE (From Diam~ US              41.3  -96.0 2020-02-24   -11 confi~
## 7 Travis, CA (From Dia~ US              38.3 -122.  2020-02-24    -5 confi~
## 8 From Diamond Princess Australia       35.4  140.  2020-02-29    -8 confi~
```

8. Which are the top 3 countries that have more than 20 deaths?

```
coronavirus %>%
    filter(type == "death") %>%
    group_by(Country.Region) %>%
    summarise(death = sum(cases)) %>%
    filter(death > 20) %>%
    arrange(desc(death))
```

```
## # A tibble: 3 x 2
##   Country.Region death
##   <chr>          <int>
## 1 Mainland China  2870
## 2 Iran              54
## 3 Italy             34
```

9. How many countries have a recovered-confirmed ratio of more than 0.60?

```
coronavirus %>%
    filter(type %in% c("confirmed", "recovered")) %>%
    group_by(Country.Region, type) %>%
    summarise(cases = sum(cases)) %>%

    # from {tidyr}: to have values put in separate columns
    pivot_wider(names_from = "type", values_from = "cases") %>%
    mutate(recovered = ifelse(is.na(recovered), 0, recovered)) %>%
    mutate(proportion = recovered / confirmed) %>%
    filter(proportion > 0.60) %>%
    arrange(desc(proportion))
```

```
## # A tibble: 10 x 4
## # Groups:   Country.Region [10]
##    Country.Region confirmed recovered proportion
##    <chr>              <int>     <dbl>      <dbl>
##  1 Cambodia               1         1      1    
##  2 India                  3         3      1    
##  3 Nepal                  1         1      1    
##  4 Russia                 2         2      1    
##  5 Sri Lanka              1         1      1    
##  6 Vietnam               16        16      1    
##  7 Macau                 10         8      0.8  
##  8 Singapore            106        72      0.679
##  9 Thailand              42        28      0.667
## 10 Malaysia              29        18      0.621
```

10. What is the recovery-confirmed ratio for Italy?

```
coronavirus %>%
    filter(Country.Region == "Italy") %>%
    filter(type %in% c("confirmed", "recovered")) %>%
    group_by(type) %>%
    summarise(cases = sum(cases)) %>%

    # from {tidyr}: to have values put in separate columns
    pivot_wider(names_from = "type", values_from = "cases") %>%
    mutate(proportion = recovered / confirmed)
```

```
## # A tibble: 1 x 3
##   confirmed recovered proportion
##       <int>     <int>      <dbl>
## 1      1694        83     0.0490
```

11. How many confirmed, recovered and dead cases were there in only the month February?

```
coronavirus %>%
    filter(between(date, as.Date("2020-02-01"), as.Date("2020-02-29"))) %>%
    group_by(type) %>%
    summarise(cases_in_february = sum(cases))
```

```
## # A tibble: 3 x 2
##   type      cases_in_february
##   <chr>                 <int>
## 1 confirmed             76086
## 2 death                  2728
## 3 recovered             39560
```

```
# alternatively
coronavirus %>%
    # specific date functions are in the {lubridate} package
    filter(month(date) == 2) %>%
    group_by(type) %>%
    summarise(cases_in_february = sum(cases))
```

```
## # A tibble: 3 x 2
##   type      cases_in_february
##   <chr>                 <int>
## 1 confirmed             76086
## 2 death                  2728
## 3 recovered             39560
```

In the following questions, you will explore the popularity of certain babynames. This dataset can be found in the like-named {babynames} library.

```
suppressPackageStartupMessages( library(babynames) )
```

1. What is the proportion of female babies that are called “Anna” in 1880 and 2017?

```
babynames %>%
    filter(sex == "F" & name == "Anna") %>%
    filter(year %in% c(1880,2017))
```

```
## # A tibble: 2 x 5
##    year sex   name      n    prop
##   <dbl> <chr> <chr> <int>   <dbl>
## 1  1880 F     Anna   2604 0.0267
## 2  2017 F     Anna   4520 0.00241
```

2. From 1880-1900, which was the most popular name for boys and girls?

```
babynames %>%
    filter(between(year, 1880, 1900)) %>%
    group_by(name, sex) %>%
    summarise(n = sum(n)) %>%
    arrange(desc(n)) %>%
    group_by(sex) %>%
    slice(1)
```

```
## # A tibble: 2 x 3
## # Groups:   sex [2]
##   name  sex        n
##   <chr> <chr>  <int>
## 1 Mary  F     239510
## 2 John  M     180444
```

3. From 1880-1900, which was the least popular name for boys and girls?

```
babynames %>%
    filter(between(year, 1880, 1900)) %>%
    group_by(name, sex) %>%
    summarise(n = sum(n)) %>%
    arrange(n) %>%
    group_by(sex) %>%
    slice(1)
```

```
## # A tibble: 2 x 3
## # Groups:   sex [2]
##   name    sex       n
##   <chr>   <chr> <int>
## 1 Abelina F         5
## 2 Abron   M         5
```

4. From 2000-2017, which was the most popular name for boys and girls?

```
babynames %>%
    filter(between(year, 2000, 2017)) %>%
    group_by(name, sex) %>%
    summarise(n = sum(n)) %>%
    arrange(desc(n)) %>%
    group_by(sex) %>%
    slice(1)
```

```
## # A tibble: 2 x 3
## # Groups:   sex [2]
##   name  sex        n
##   <chr> <chr>  <int>
## 1 Emma  F     339802
## 2 Jacob M     413884
```

5. From 2000-2017, which was the least popular name for boys and girls?

```
babynames %>%
    filter(between(year, 2000, 2017)) %>%
    group_by(name, sex) %>%
    summarise(n = sum(n)) %>%
    arrange(n) %>%
    group_by(sex) %>%
    slice(1)
```

```
## # A tibble: 2 x 3
## # Groups:   sex [2]
##   name  sex       n
##   <chr> <chr> <int>
## 1 Aada  F         5
## 2 Aabir M         5
```

6. For girls, what were the most popular name in 1880, 1917, 1943 and 2017?

```
babynames %>%
    filter(sex == "F") %>%
    filter(year %in% c(1880,1917,1943,2017)) %>%
    group_by(year,name) %>%
    summarise(n = sum(n)) %>%
    ungroup() %>% arrange(year, desc(n)) %>%
    group_by(year) %>%
    slice(1)
```

```
## # A tibble: 4 x 3
## # Groups:   year [4]
##    year name      n
##   <dbl> <chr> <int>
## 1  1880 Mary   7065
## 2  1917 Mary  64281
## 3  1943 Mary  66169
## 4  2017 Emma  19738
```

7. How many different boy names were there between 1880-1900?

```
babynames %>%
    filter(sex == "M") %>%
    filter(between(year,1880,1900)) %>%
    pull(name) %>%
    unique() %>%
    length()
```

```
## [1] 2411
```

8. How many different boy names were there between 2000-2017? Did we diversify compared to the previous era?

```
babynames %>%
    filter(sex == "M") %>%
    filter(between(year,2000,2017)) %>%
    pull(name) %>%
    unique() %>%
    length()
```

```
## [1] 30118
```

9. What is the popularity of your own name in 2017?

```
babynames %>%
    filter(name == "Adrian" & year == 2017)
```

```
## # A tibble: 2 x 5
##    year sex   name        n      prop
##   <dbl> <chr> <chr>   <int>     <dbl>
## 1  2017 F     Adrian    114 0.0000608
## 2  2017 M     Adrian   6203 0.00316
```

In the following questions, you will perform exploratory analysis on the flight schedule of airplanes arriving and departing from NYC in 2013. This dataset (flights) is contained in the {nycflights13} library.

```
suppressPackageStartupMessages( library(nycflights13) )
```

1. How many flights have an arrival delay of more than 2 hours? HINT: **tally()**

```
flights %>%
  filter(arr_delay >= 120) %>%
  tally()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1 10200
```

2. How many flights flew to Houston (IAH or HOU)?

```
flights %>%
  filter(dest %in% c("IAH", "HOU")) %>%
  tally()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  9313
```

3. How many flights were operated by UA, AA and DL separately?

```
flights %>%
  filter(carrier %in% c("UA","AA","DL")) %>%
  group_by(carrier) %>%
  summarise(n = n())
```

```
## # A tibble: 3 x 2
##   carrier     n
##   <chr>   <int>
## 1 AA      32729
## 2 DL      48110
## 3 UA      58665
```

4. How many flights departed in the months July, August and September separately?

```
flights %>%
  filter(between(month,7,9)) %>%
  tally()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1 86326
```

5. How many flights arrived with more than 2 hours delay, but left on time?

```
flights %>%
  filter(arr_delay > 120 & dep_delay == 0) %>%
  tally()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1     3
```

6. How many flights departed between midnight and 6 am (inclusive)?

```
flights %>%
  filter(between(dep_time, 0, 600)) %>%
  tally()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  9344
```

7. How many flights have a missing dep_time? What other variables are missing? What might these rows represent?

```
# example row
flights %>%
  filter(is.na(dep_time)) %>%
  head(1)
```

```
## # A tibble: 1 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>
## 1  2013     1     1       NA           1630        NA       NA
## # ... with 12 more variables: sched_arr_time <int>, arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>,
## #   time_hour <dttm>
```

```
flights %>%
  filter(is.na(dep_time)) %>%
  tally()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  8255
```

8. Sort flights to find the most delayed departure flights.

```
flights %>%
  arrange(desc(dep_delay))
```

```
## # A tibble: 336,776 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>
##  1  2013     1     9      641            900      1301     1242
##  2  2013     6    15     1432           1935      1137     1607
##  3  2013     1    10     1121           1635      1126     1239
##  4  2013     9    20     1139           1845      1014     1457
##  5  2013     7    22      845           1600      1005     1044
##  6  2013     4    10     1100           1900       960     1342
##  7  2013     3    17     2321            810       911      135
##  8  2013     6    27      959           1900       899     1236
##  9  2013     7    22     2257            759       898      121
## 10  2013    12     5      756           1700       896     1058
## # ... with 336,766 more rows, and 12 more variables: sched_arr_time <int>,
## #   arr_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>
```

9. Find the flights that flew the longest distance.

```
flights %>%
  arrange(desc(distance))
```

```
## # A tibble: 336,776 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>
##  1  2013     1     1      857            900        -3     1516
##  2  2013     1     2      909            900         9     1525
##  3  2013     1     3      914            900        14     1504
##  4  2013     1     4      900            900         0     1516
##  5  2013     1     5      858            900        -2     1519
##  6  2013     1     6     1019            900        79     1558
##  7  2013     1     7     1042            900       102     1620
##  8  2013     1     8      901            900         1     1504
##  9  2013     1     9      641            900      1301     1242
## 10  2013     1    10      859            900        -1     1449
## # ... with 336,766 more rows, and 12 more variables: sched_arr_time <int>,
## #   arr_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>
```

10. Select from flights the following columns: dep_time, dep_delay, arr_time, arr_delay

```
flights %>%
  select(dep_time, dep_delay, arr_time, arr_delay)
```

```
## # A tibble: 336,776 x 4
##    dep_time dep_delay arr_time arr_delay
##       <int>     <dbl>    <int>     <dbl>
##  1      517         2      830        11
##  2      533         4      850        20
##  3      542         2      923        33
##  4      544        -1     1004       -18
##  5      554        -6      812       -25
##  6      554        -4      740        12
##  7      555        -5      913        19
##  8      557        -3      709       -14
##  9      557        -3      838        -8
## 10      558        -2      753         8
## # ... with 336,766 more rows
```

```
# alternatively
flights %>%
  select(starts_with("dep"), starts_with("arr"))
```

```
## # A tibble: 336,776 x 4
##    dep_time dep_delay arr_time arr_delay
##       <int>     <dbl>    <int>     <dbl>
##  1      517         2      830        11
##  2      533         4      850        20
##  3      542         2      923        33
##  4      544        -1     1004       -18
##  5      554        -6      812       -25
##  6      554        -4      740        12
##  7      555        -5      913        19
##  8      557        -3      709       -14
##  9      557        -3      838        -8
## 10      558        -2      753         8
## # ... with 336,766 more rows
```

11. How many cancelled flights are there per month day?

```
flights %>%
  filter(is.na(dep_time)) %>%
  group_by(day) %>%
  tally()
```

```
## # A tibble: 31 x 2
##      day     n
##    <int> <int>
##  1     1   246
##  2     2   250
##  3     3   109
##  4     4    82
##  5     5   226
##  6     6   296
##  7     7   318
##  8     8   921
##  9     9   593
## 10    10   535
## # ... with 21 more rows
```

12. Which carrier has the most cancelled flights?

```
flights %>%
  filter(is.na(dep_time)) %>%
  group_by(carrier) %>%
  tally() %>%
  arrange(desc(n)) %>%
  head(5) # top 5
```

```
## # A tibble: 5 x 2
##   carrier     n
##   <chr>   <int>
## 1 EV       2817
## 2 MQ       1234
## 3 9E       1044
## 4 UA        686
## 5 US        663
```

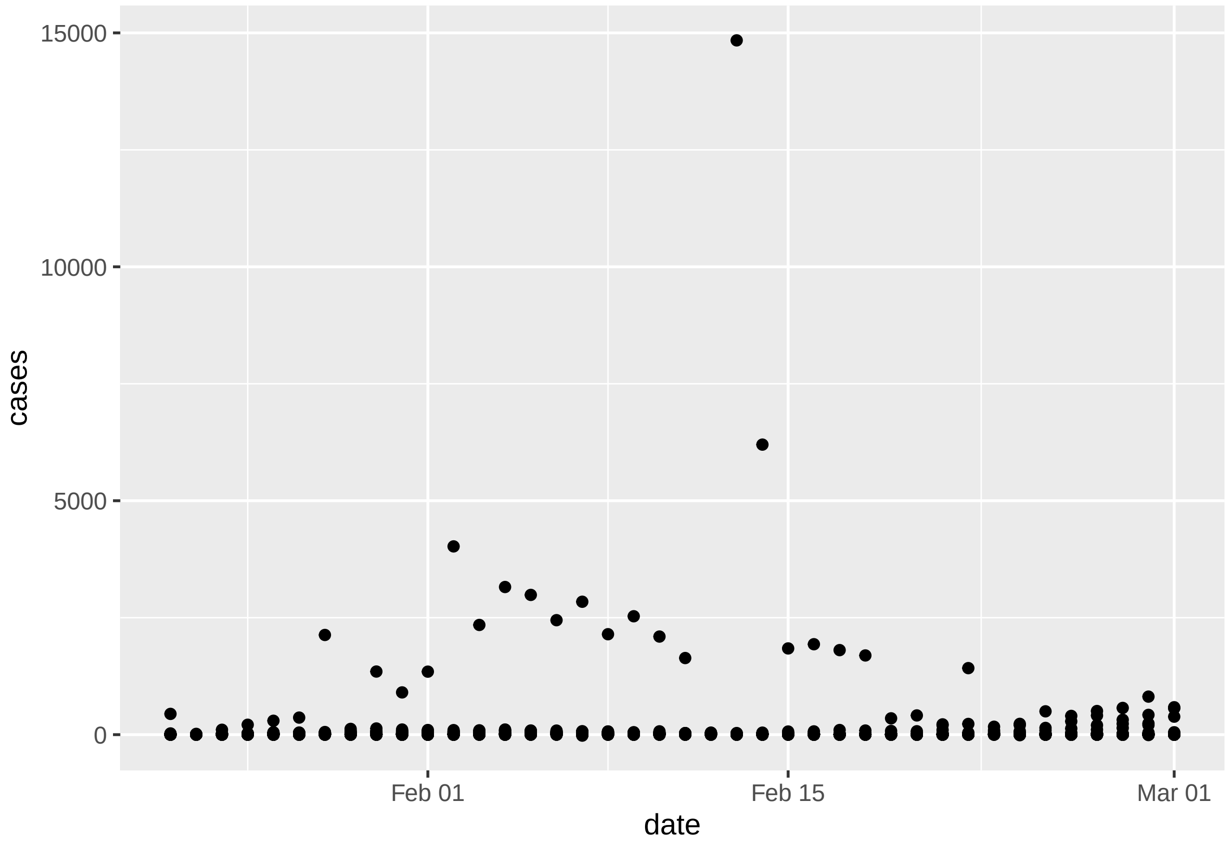
data visualizations {ggplot2}

```
suppressPackageStartupMessages( library(ggplot2) )
suppressPackageStartupMessages( library(forcats) )
```

In the following questions, you will perform exploratory analysis on the coronavirus. This dataset is contained in the like-named {coronavirus} library.

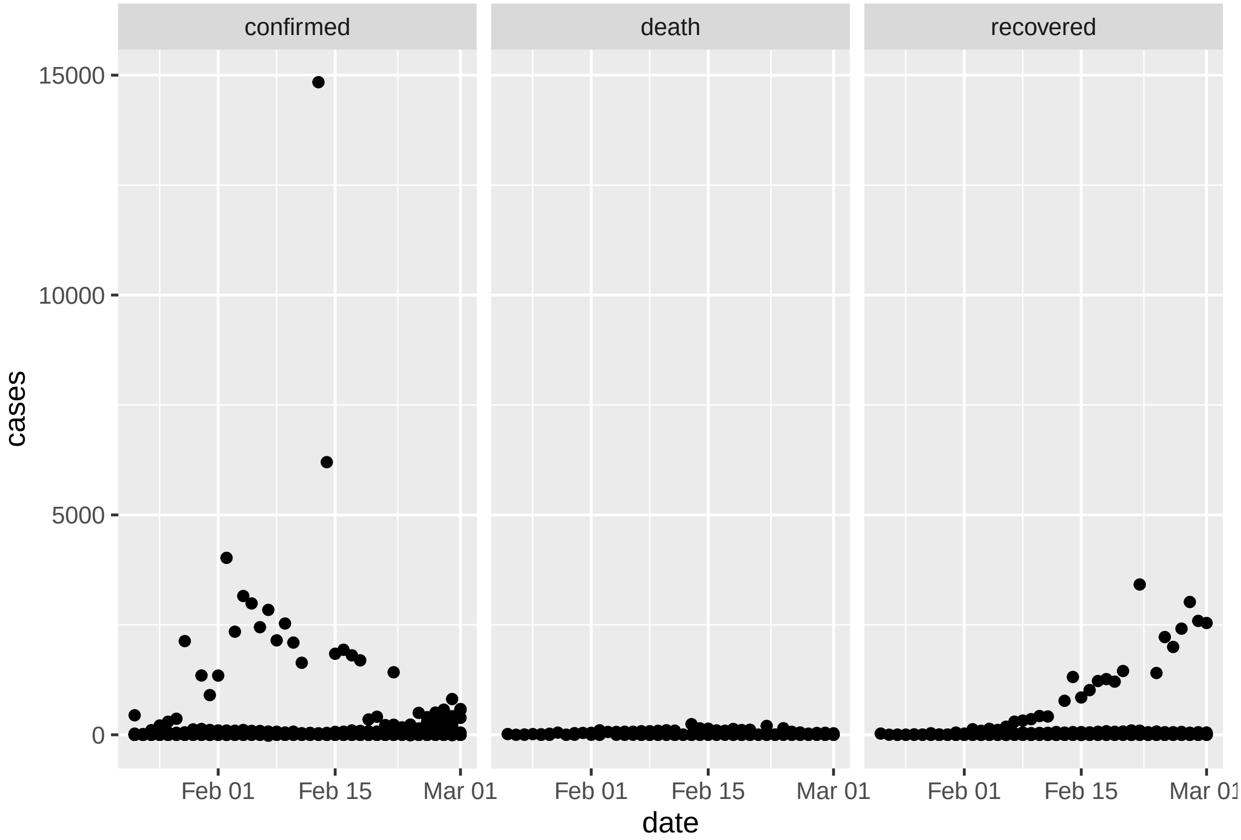
1. Plot the confirmed coronavirus cases over time.

```
coronavirus %>%
  filter(type == "confirmed") %>%
  ggplot(., aes(date, cases)) +
  geom_point()
```



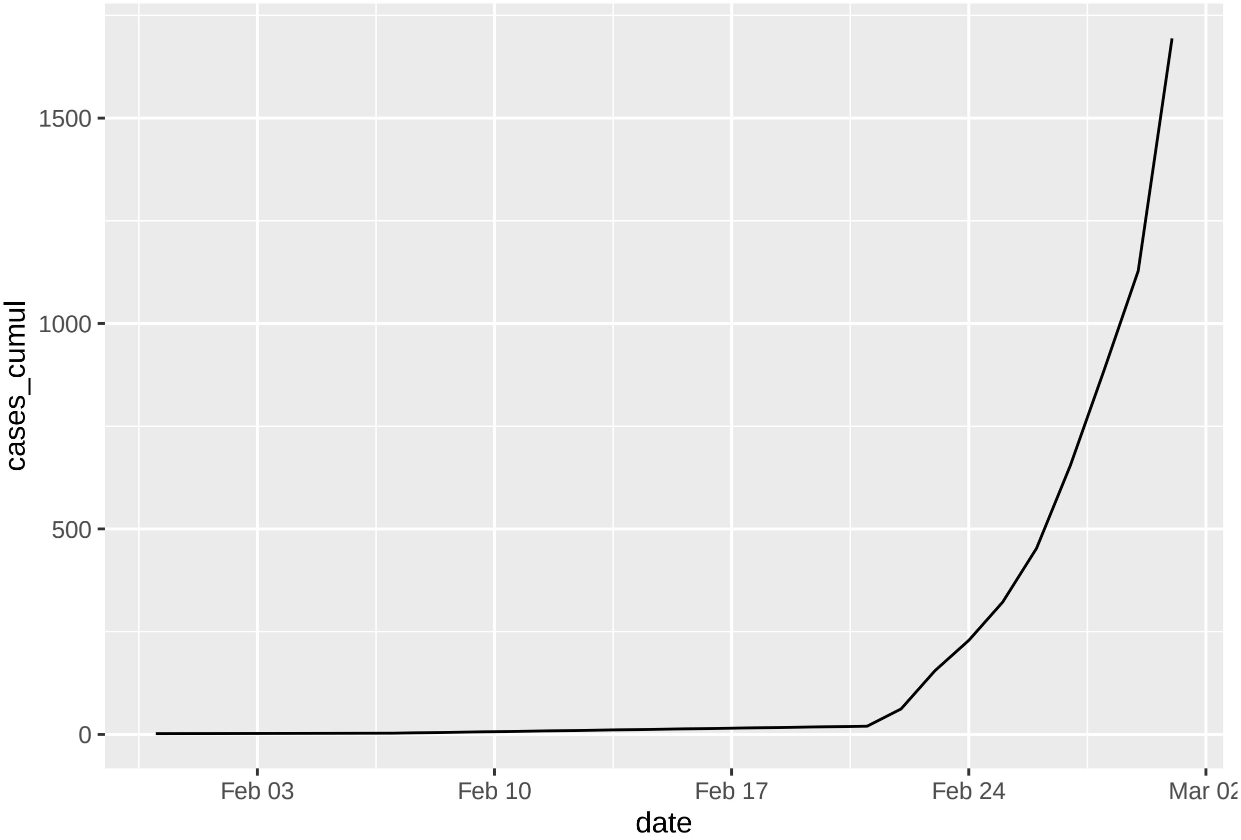
2. Plot the confirmed cases over time, as well as recovered from and death by coronavirus in side-by-side plots

```
coronavirus %>%
  ggplot(., aes(date, cases)) +
  geom_point() +
  facet_wrap(~ type)
```



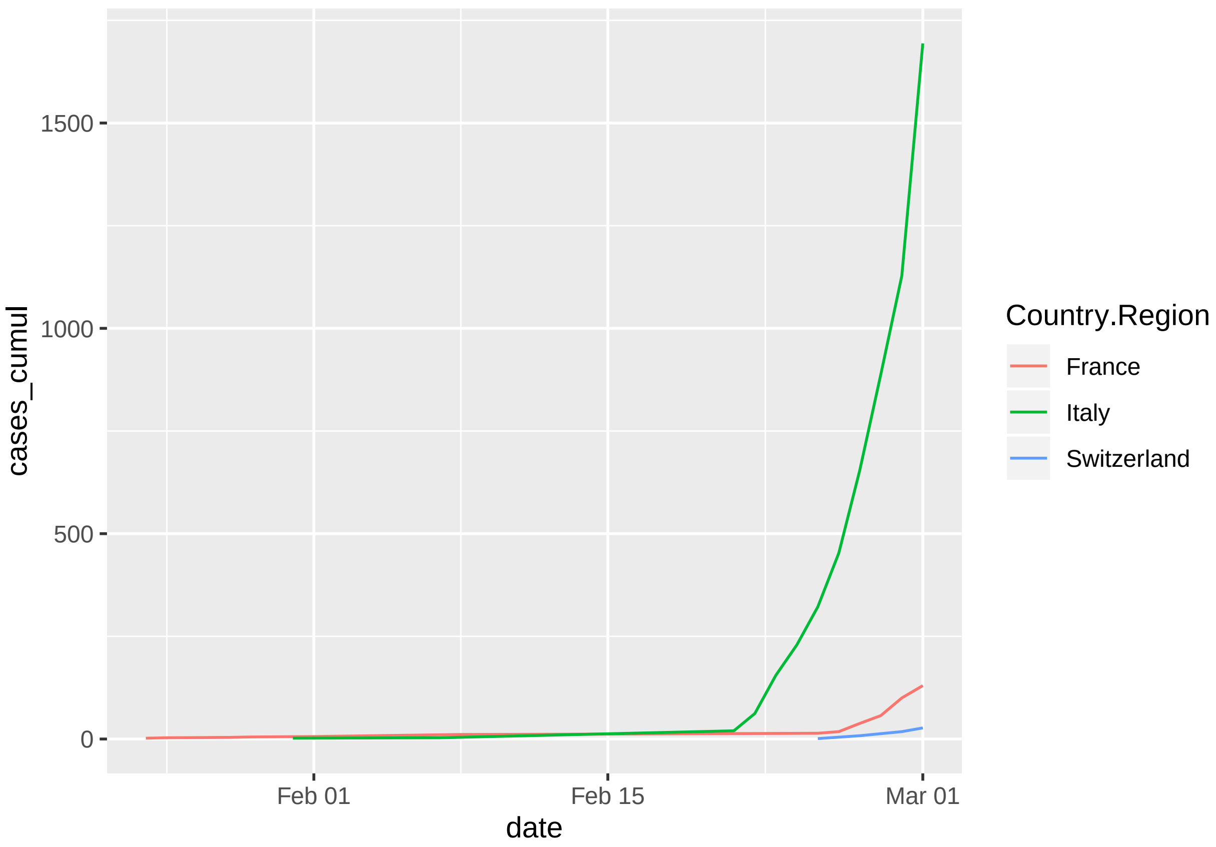
3. Plot the cumulative frequency of confirmed cases in Italy. HINT: **cumsum()**

```
coronavirus %>%
  filter(Country.Region == "Italy") %>%
  filter(type == "confirmed") %>%
  mutate(cases_cumul = cumsum(cases)) %>%
  ggplot(., aes(date, cases_cumul)) +
  geom_line()
```



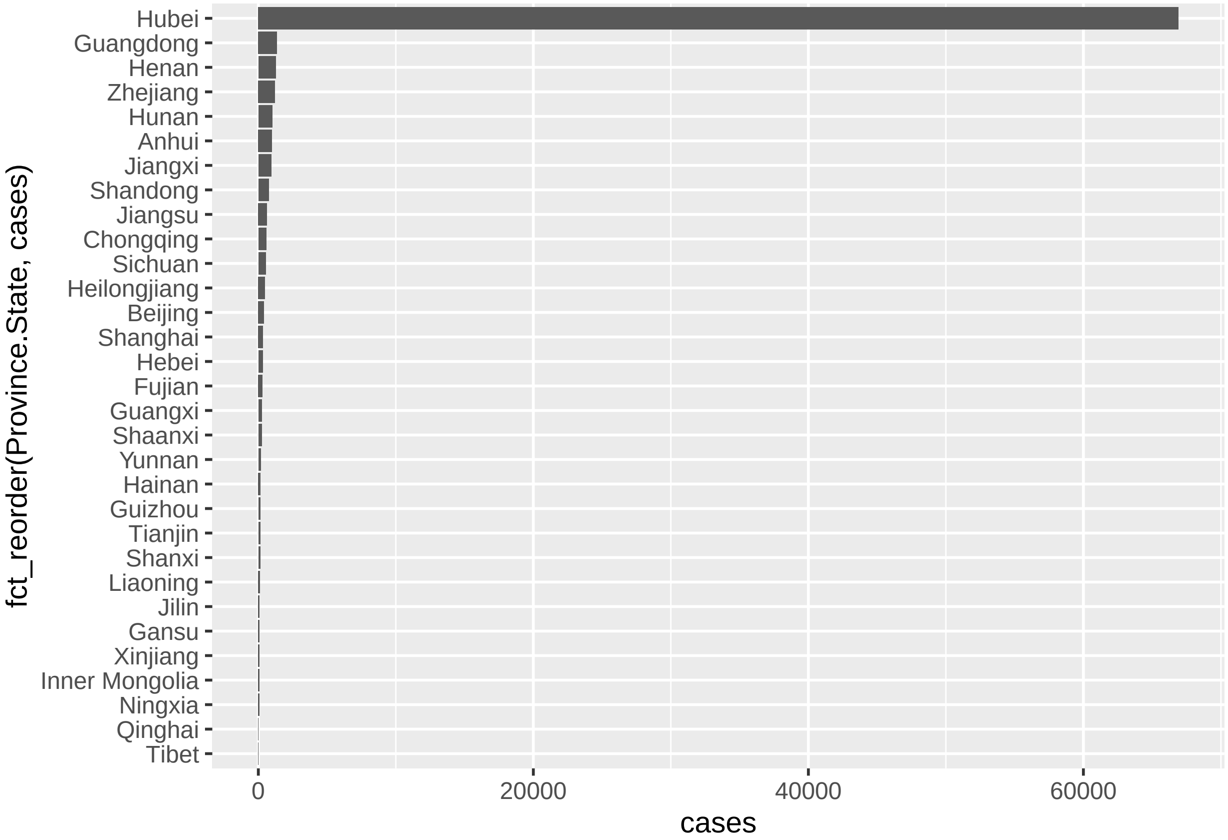
4. Plot the cumulative frequency of confirmed cases in Italy as well as its neighbouring countries (France and Switzerland). Plot the progression in one plot, but with differently colored lines. HINT: **cumsum()**

```
coronavirus %>%
  filter(Country.Region %in% c("Italy","Switzerland","France")) %>%
  filter(type == "confirmed") %>%
  group_by(Country.Region) %>%
  mutate(cases_cumul = cumsum(cases)) %>%
  ggplot(., aes(date, cases_cumul, color = Country.Region)) +
  geom_line()
```



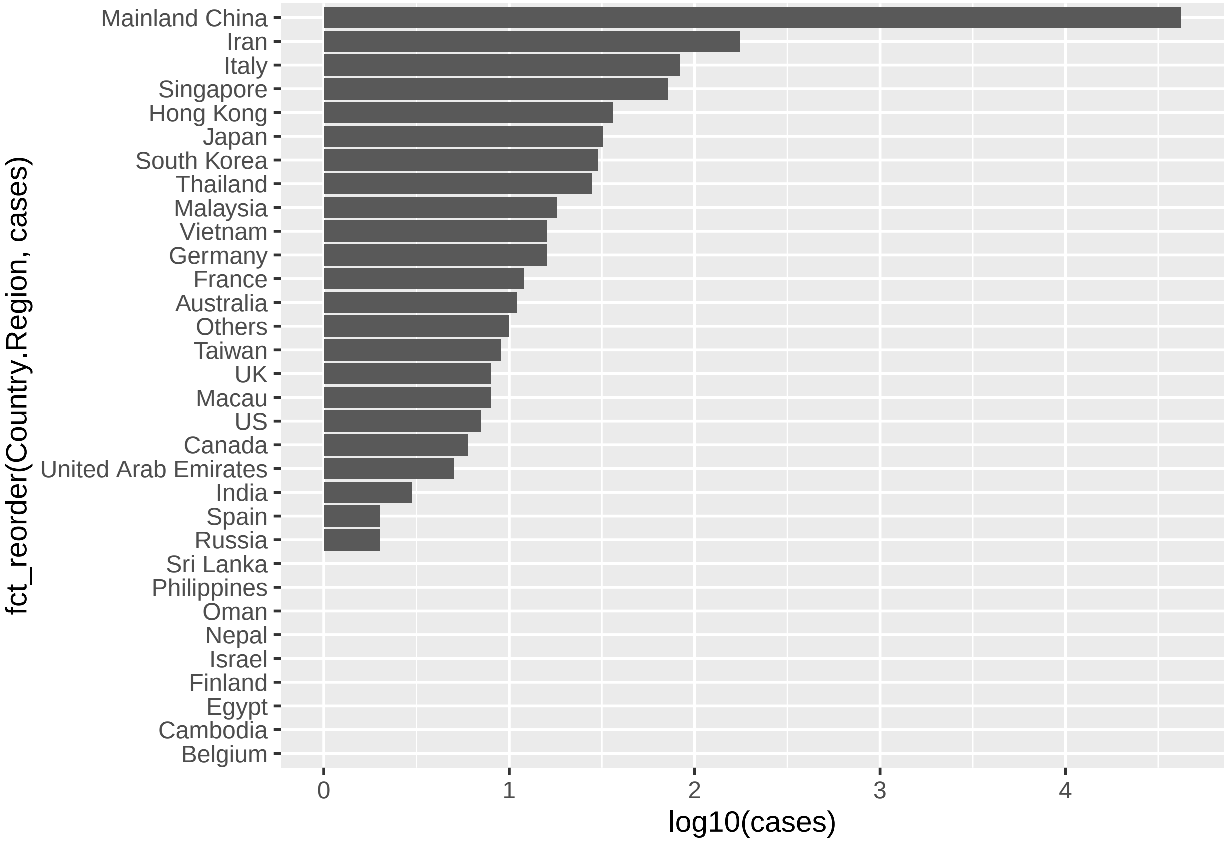
5. Selectively for Mainland China, which province has shown the most confirmed cases? Plot with bars, and sorted in decreasing severity. HINT: **fct_reorder()** from the {forcats} library and **coord_flip()**

```
coronavirus %>%
  filter(Country.Region == "Mainland China") %>%
  filter(type == "confirmed") %>%
  group_by(Province.State) %>%
  summarise(cases = sum(cases)) %>%
  ggplot(., aes(fct_reorder(Province.State, cases), cases)) +
  geom_col() +
  coord_flip()
```



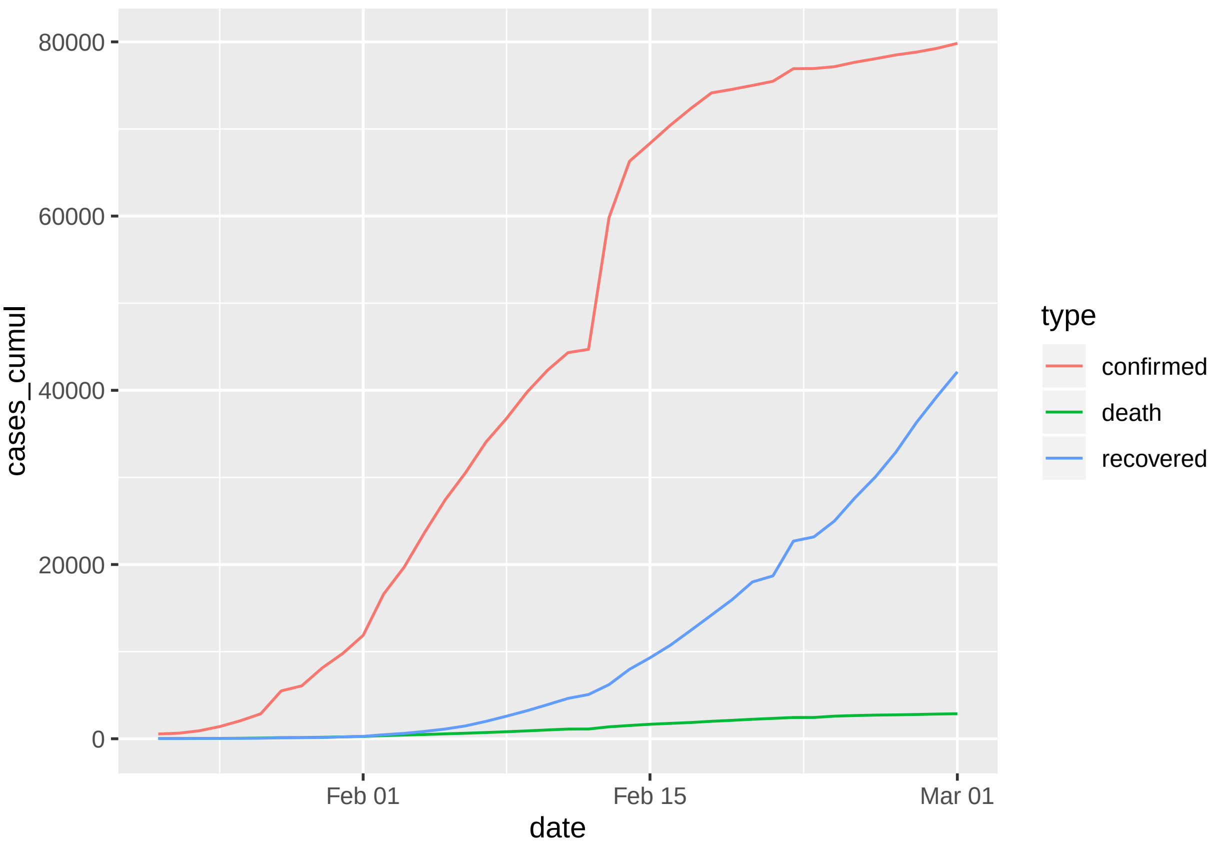
6. In the coronavirus dataset, filter for recovered cases, and show per country the total amount of recovered cases. Plot with bars, and sorted in decreasing severity. Also, normalize the number of cases with log10 (to reduce the saturation from China).

```
coronavirus %>%
  filter(type == "recovered") %>%
  group_by(Country.Region) %>%
  summarise(cases = sum(cases)) %>%
  ggplot(., aes(fct_reorder(Country.Region, cases), log10(cases))) +
  geom_col() +
  coord_flip()
```



7. Selectively for Mainland China, plot the cumulative frequency of confirmed cases, as well as recovered and dead cases. Place in one plot, but with differently colored lines.

```
coronavirus %>%
  filter(Country.Region == "Mainland China") %>%
  group_by(type, date) %>%
  summarise(cases = sum(cases)) %>%
  ungroup() %>% group_by(type) %>%
  mutate(cases_cumul = cumsum(cases)) %>%
  ggplot(., aes(date, cases_cumul, color = type)) +
  geom_line()
```

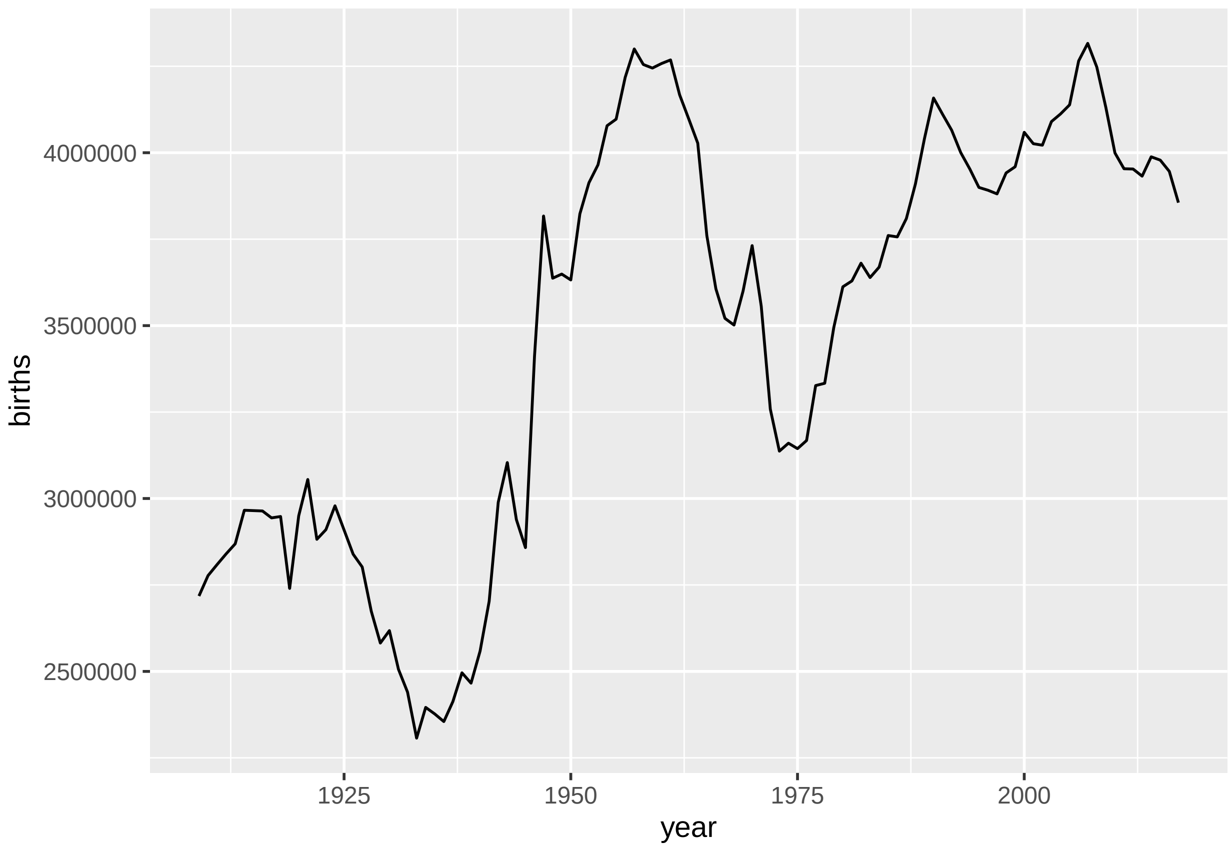


In the following questions, you will visualize the popularity of certain babynames as well as the number of births. This dataset can be found in the like-named {babynames} library.

```
suppressPackageStartupMessages( library(babynames) )
```

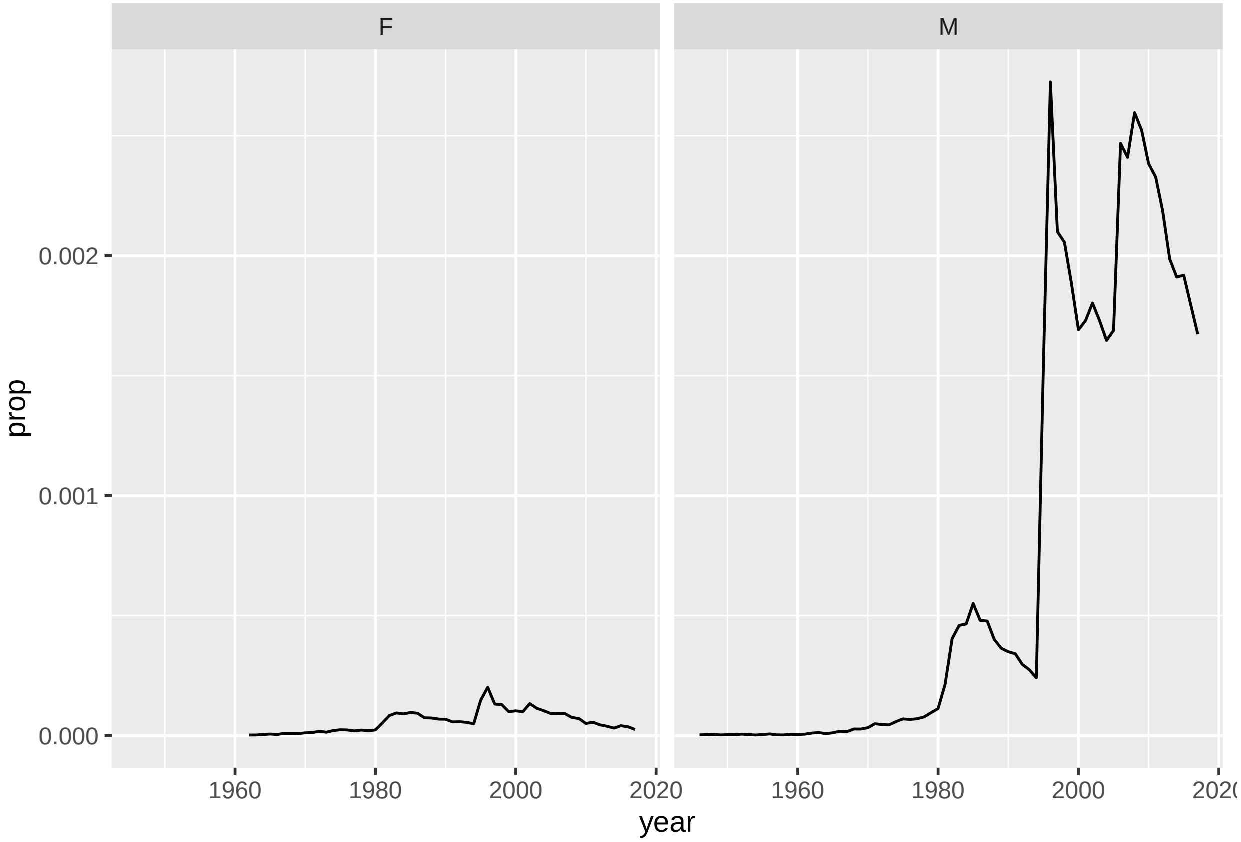
1. Plot the number of birth over time from the births dataset.

```
births %>%
    ggplot(., aes(year, births)) + geom_line()
```



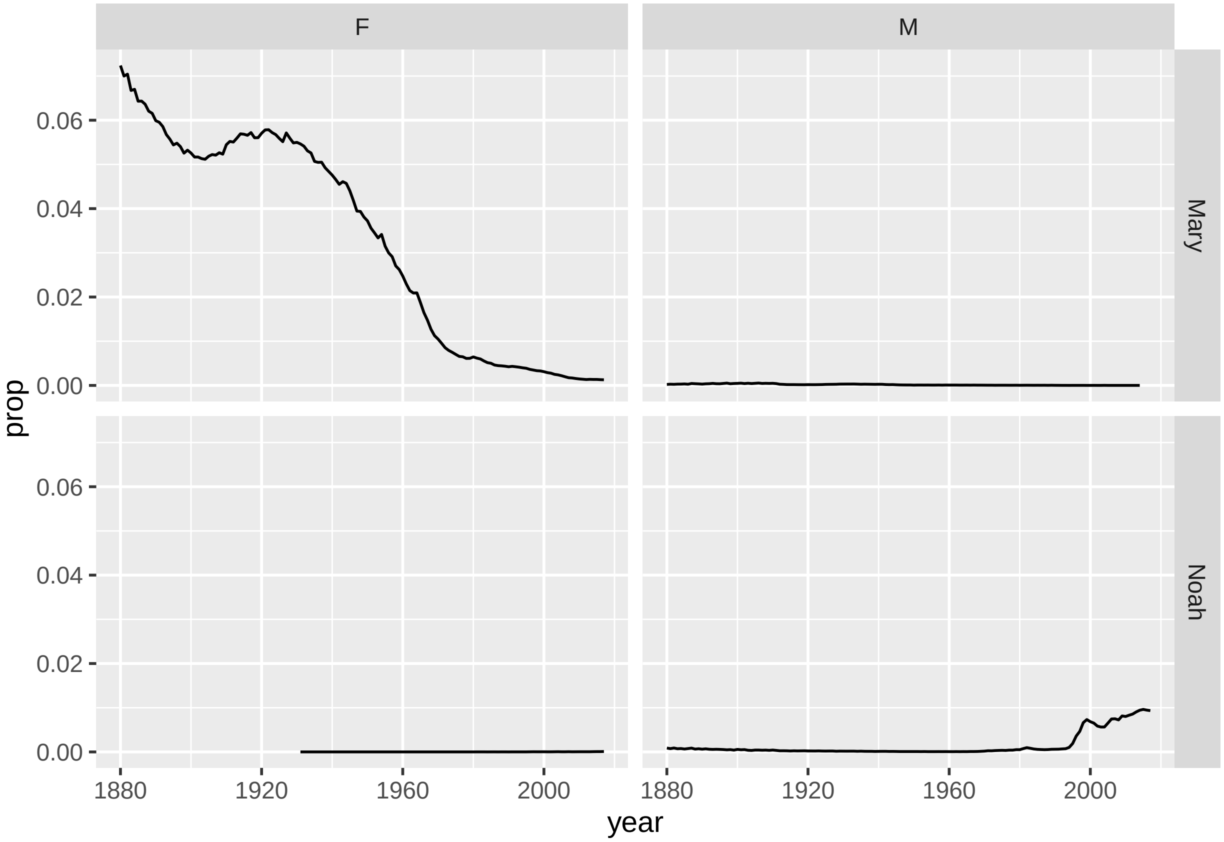
2. Plot the development of the name "Tristan" over time for each sex from the babynames dataset.

```
babynames %>%
    filter(name == "Tristan") %>%
    ggplot(., aes(year, prop)) +
    geom_line() +
    facet_wrap(~ sex)
```



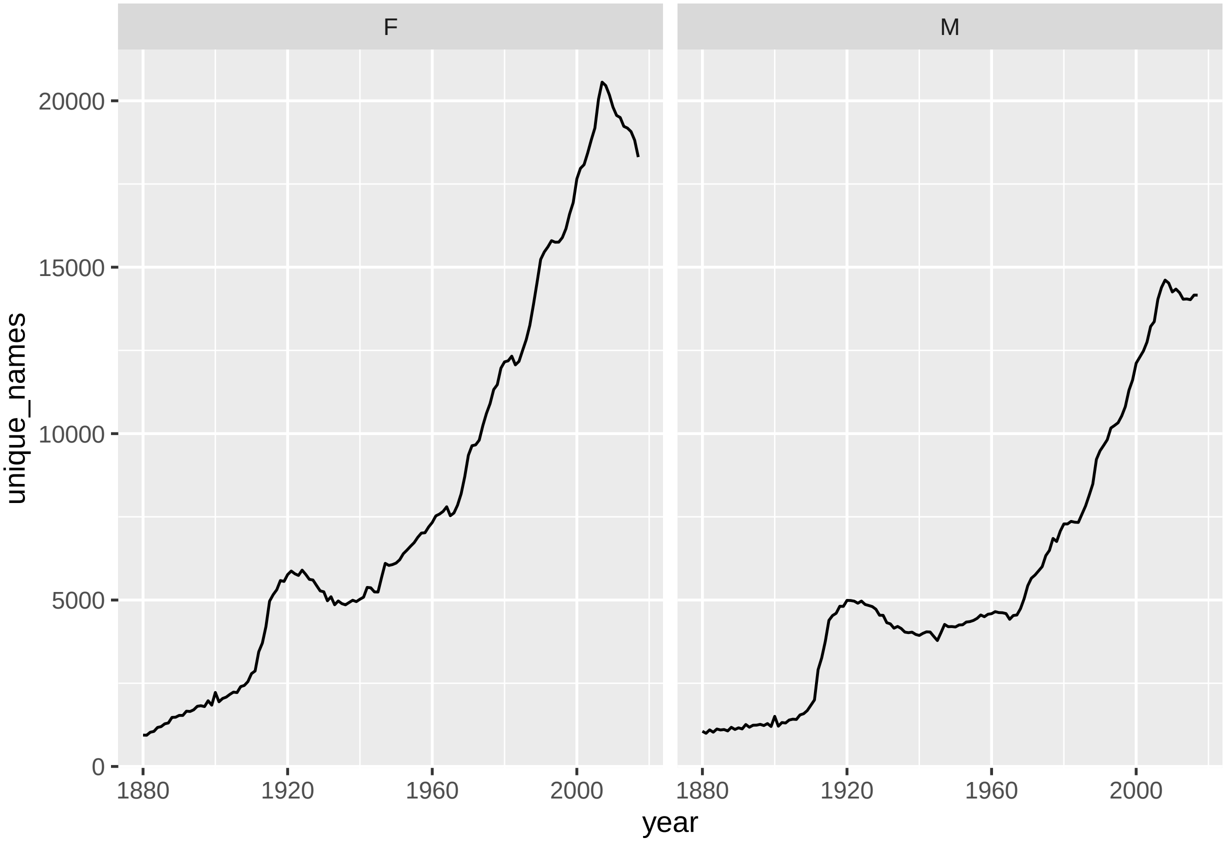
3. Plot the development of the name “Mary” and “Noah” over time for each sex. What can you say about the development of these names?

```
babynames %>%
    filter(name %in% c("Mary","Noah")) %>%
    ggplot(., aes(year, prop)) +
    geom_line() +
    facet_grid(name ~ sex)
```



4. Extract from the babynames dataset for each year, the number of unique names for boys and girls. Plot these over time side-by-side. What is the trend? Which sex has more diversified names? Any peculiar trends?

```
babynames %>%
    group_by(year, sex) %>%
    summarise(unique_names = n()) %>%
    ggplot(., aes(year, unique_names)) +
    geom_line() +
    facet_grid(. ~ sex)
```

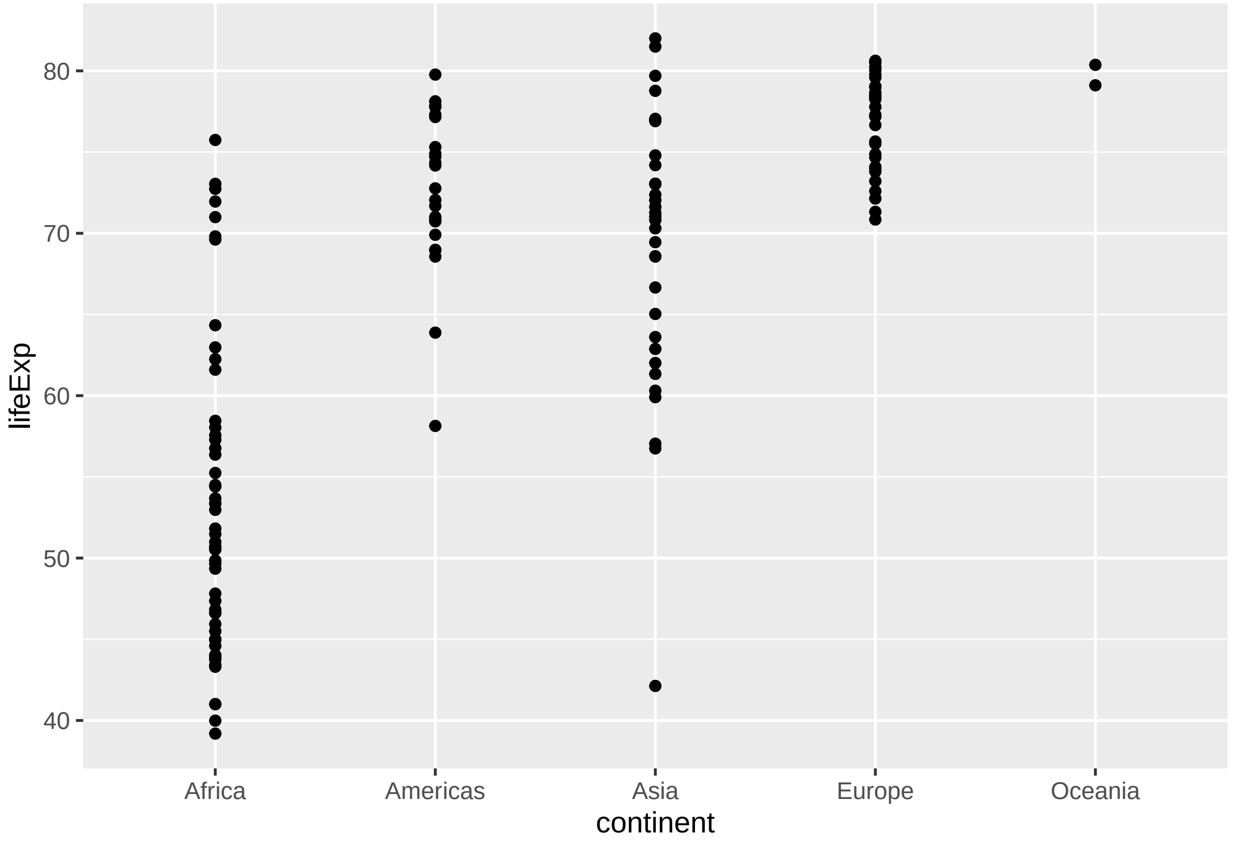


In the following questions, you will visualize the economic parameteres from the gapminder dataset. This dataset can be found in the like-named {gapminder} library.

```
suppressPackageStartupMessages( library(gapminder) )
```

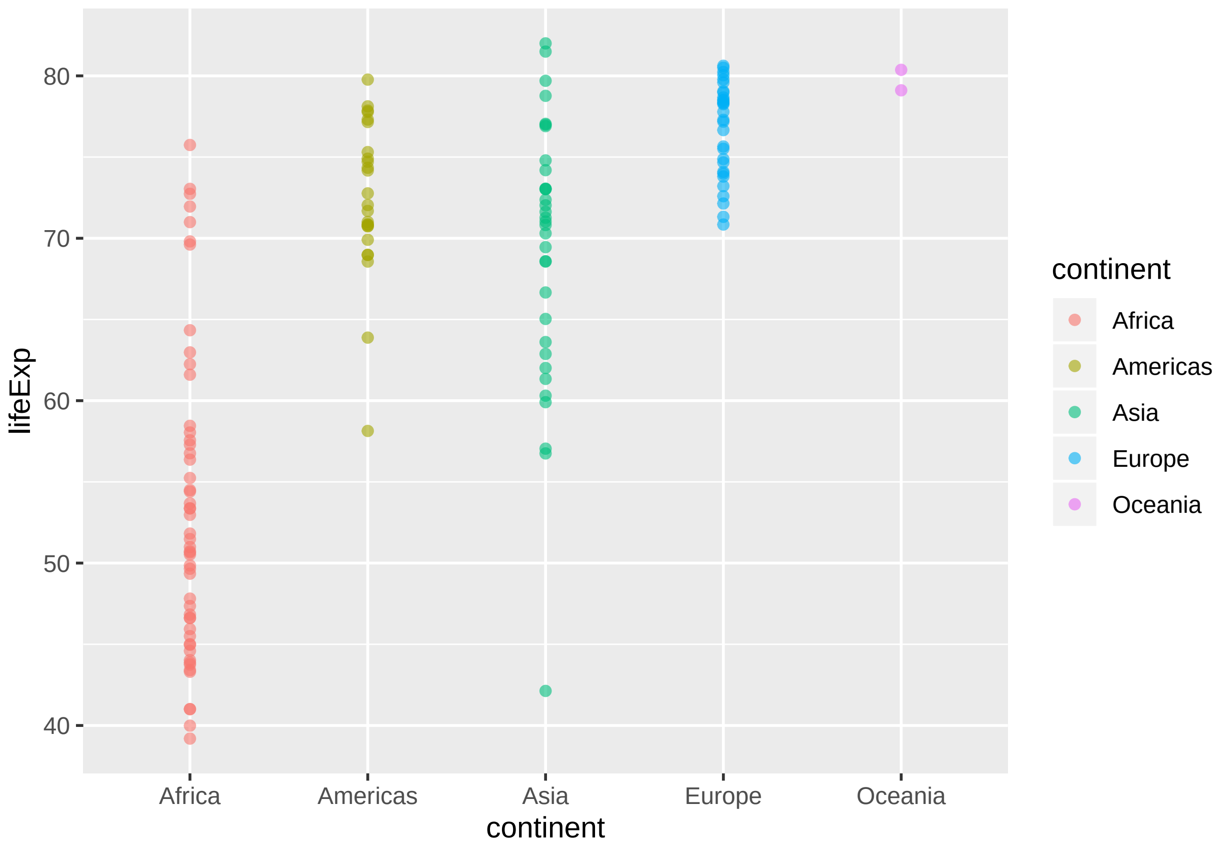
1. Plot the life expectancy for each continent from the year 2002 as individual points

```
gapminder %>%
    filter(year == 2002) %>%
    ggplot(., aes(continent, lifeExp)) +
    geom_point()
```



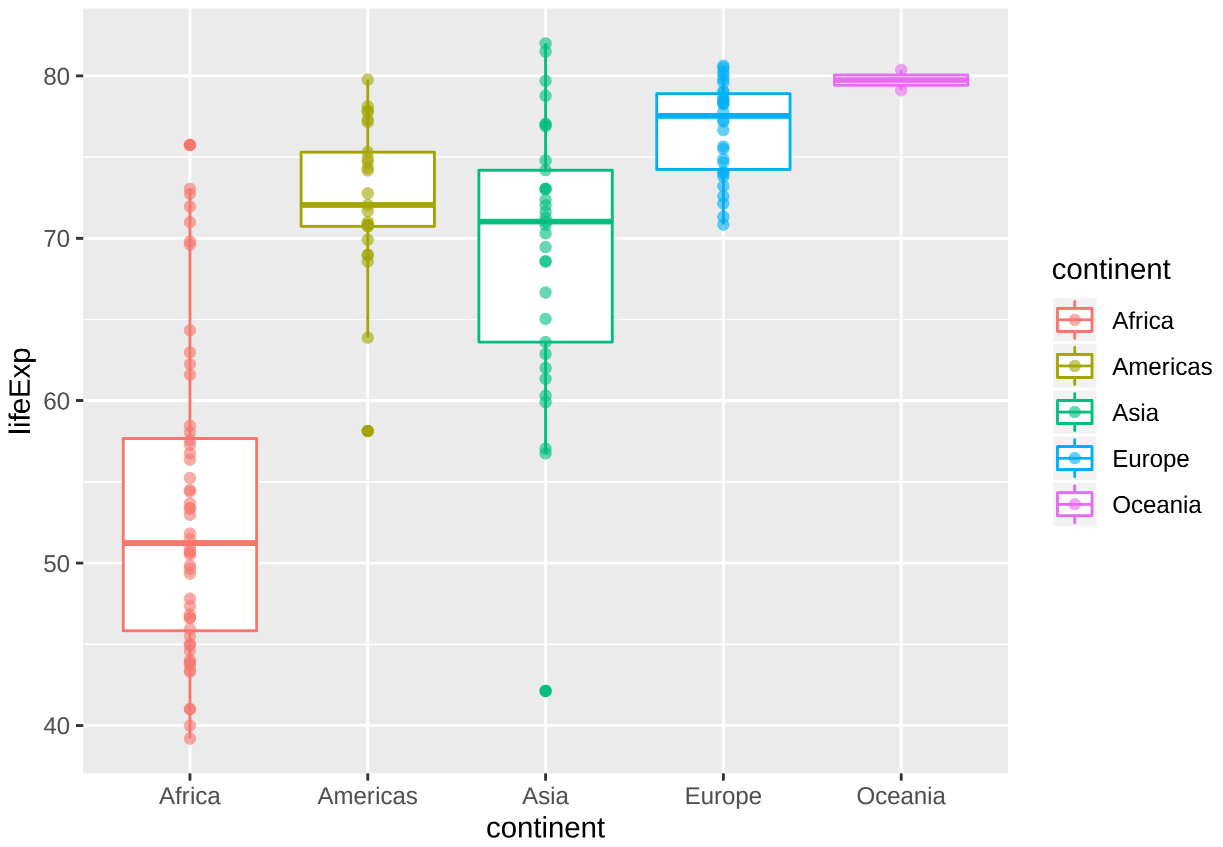
2. Continue from the previous question: add some transparency in the dots and give each continent a different color.

```
gapminder %>%
    filter(year == 2002) %>%
    ggplot(., aes(continent, lifeExp, color = continent)) +
    geom_point(alpha = 0.60)
```



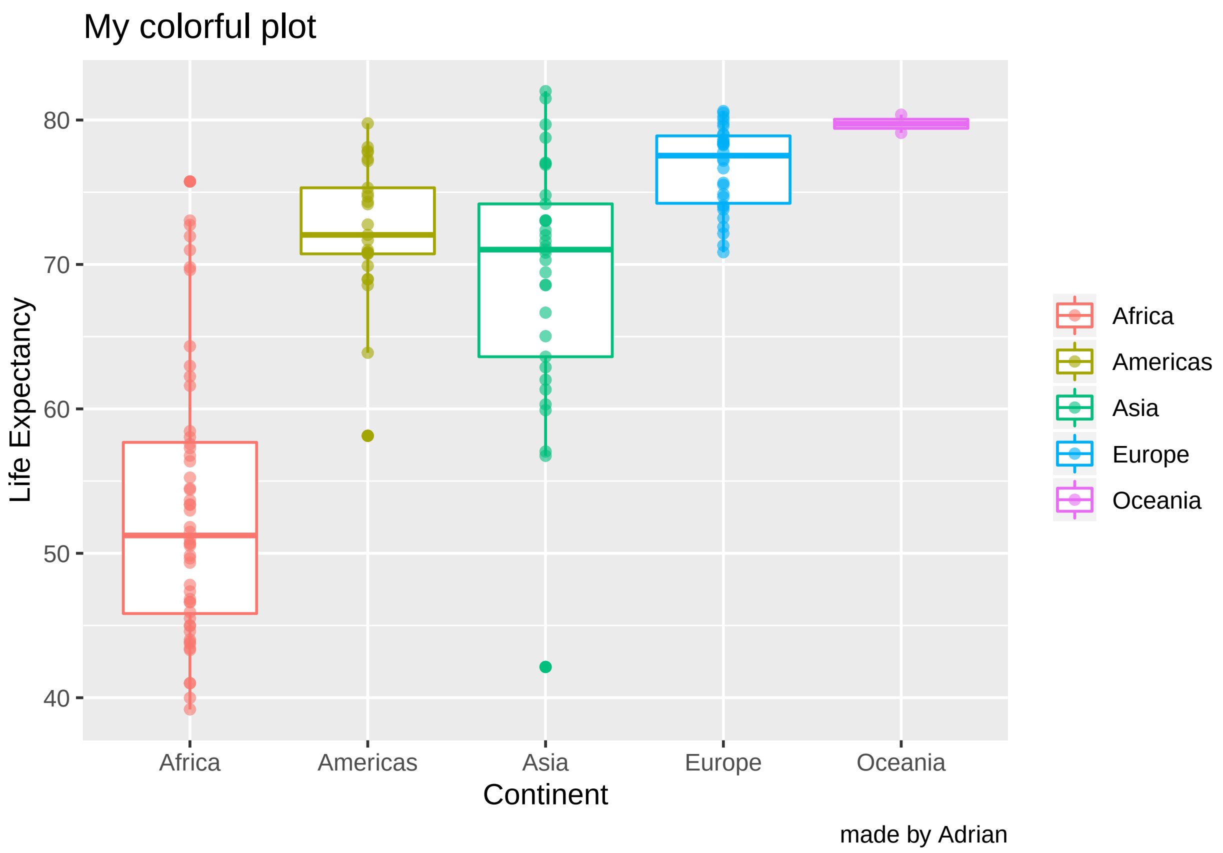
3. Continue from the previous question: add a boxplot to the graph. Add it in such a way that the individual points will still be visible.

```
gapminder %>%
    filter(year == 2002) %>%
    ggplot(., aes(continent, lifeExp, color = continent)) +
    geom_boxplot() +
    geom_point(alpha = 0.60)
```



4. Continue from the previous question: Adjust the labels so they are more presentable. Change the title on the y-axis to “Life Expectancy”, and the title on the x-axis to “Continent”. Add a title “My colorful plot”. Also add a caption with “made by {your name}”. Finally, remove the legend title.

```
gapminder %>%
    filter(year == 2002) %>%
    ggplot(., aes(continent, lifeExp, color = continent)) +
    geom_boxplot() +
    geom_point(alpha = 0.60) +
    labs(x = "Continent", y = "Life Expectancy",
         title = "My colorful plot", caption = "made by Adrian") +
    theme(legend.title = element_blank())
```



My colorful plot
Life Expectancy
80
70
60
50
40
Africa
Americas
Asia
Europe
Oceania
Continent
Africa
Americas
Asia
Europe
Oceania
made by Adrian