R Challenge Part 1

Emmanuel Ugwuabonyi

Contents

A Explanations

Rules

- Correct results can be obtained in multiple, different ways.
- Results can depend on the data cleaning approach. Hence, it is possible that different sets of results are correct.
- It is not necessary to comment the code or results. However, commenting can be helpful to clarify why you take a particular approach, or if you note that your results are not entirely correct.
- I strongly recommend digging into packages such as stringr and lubridate for some of the data processing steps.
- Only include the answers to the posed questions into this Rmd file, not some additional analysis that you have performed.
- When you are finished, please upload both the Rmd and the knitted html file.

Data

This R challenge is about the history of Olympic Games. The following four data sets are needed for this project:

- **athletes.csv** contains metadata on athletes, e.g. their name and sex
- **games.tsv** contains metadata on the Olympic Games, e.g. start and end dates
- **results.csv** contains information on results: Which athlete participated in which Olympic Games and disciplines, achieving which position?

B Tasks

Load all packages that you need here.

```
library(tidyverse)
library(lubridate)
library(stringr)
```

Task 1 (5 points)

Read in the **results** data set. Then count the total number of observations per country, and show the top 10 countries

```
results <- read_csv(file = 'results.csv')


result_10 <- results%>%
  group_by(country)%>%
  summarise(count=n())%>%
  arrange(desc(count))%>%
  slice(1:10)

result_10[1:10,]
```

```
## # A tibble: 10 x 2
##    country       count
##    <chr>         <int>
##  1 United States 19966
##  2 France        13397
##  3 Great Britain 12699
##  4 Italy         11414
##  5 Canada        10742
##  6 Germany       10383
##  7 Japan          9412
##  8 Sweden         8811
##  9 Australia      8477
## 10 Hungary        7042
```

Count the total number of observations for Germany in all Winter Games. Note: Germany appears in the data with 3 different names (and codes), due to the fact that Germany was split into West and East.

```
Germany_count <- results %>%
  filter(str_detect(country,"Germany") & (season == "Winter"))%>%
  summarise(n = n())

Germany_count
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  3801
```

Task 2 (15 points)

Read in the data set **athletes.csv** and join it with the **results**. Briefly explain which join type you are using and why.

```
athletes <- read_csv2("athletes.csv")

athletes_results <- inner_join(athletes,results, by = 'athlete')

# left join was used because it returns all the rows from the original table even if they don't have ma
```

Using the joined data, calculate the average height and the average weight of all male participants.

```
height_weight <- athletes_results %>%
  separate(measurements, into =  c("heights", "weights"), sep ='/')%>%
  mutate(heights = as.numeric(str_extract(heights, "\\d+")),
         weights=as.numeric(str_extract(weights, "\\d+")))

male_heights_weights <- height_weight%>%
  select(sex,heights, weights) %>%
  filter(sex == 'Male')



mean(male_heights_weights$heights, na.rm = TRUE)
```

```
## [1] 177.9967
```

```
mean(male_heights_weights$weights, na.rm = TRUE)
```

```
## [1] 75.76233
```

Identify the tallest athlete (regarding variable height) and show all available information for this athlete in the joined data set.

```
height_new <- height_weight%>%
  filter(heights == max(heights, na.rm = T))

height_new
```

```
## # A tibble: 3 x 18
##   athlete name     sex   born    died  affiliations  title heights weights  year
##     <dbl> <chr>    <chr> <chr>   <chr> <chr>         <chr>   <dbl>   <dbl> <dbl>
## 1   89782 Yao•Ming Male  12 Sep~ <NA>  Shanghai Sha~ <NA>      226     141  2008
## 2   89782 Yao•Ming Male  12 Sep~ <NA>  Shanghai Sha~ <NA>      226     141  2004
## 3   89782 Yao•Ming Male  12 Sep~ <NA>  Shanghai Sha~ <NA>      226     141  2000
## # ... with 8 more variables: season <chr>, sport <chr>, discipline <chr>,
## #   pos <dbl>, medal <chr>, country <chr>, country_code <chr>, team <chr>
```

Task 3 (15 points)

Read in the data set **games.tsv** and join it with your existing data set. Briefly explain: For which Olympic Games do you have metadata, but no results? What is the reason for the missing results?

```
games <- read_tsv("games.tsv")

games_athletes_results <-left_join(athletes_results, games,
                                   by =c('season', 'year'))
games_anti_joint <- anti_join(games, athletes_results, by =c('season','year'))

games_anti_joint
```

```
## # A tibble: 10 x 7
##     year season games_city  games_country games_opened games_closed games_remark
##    <dbl> <chr>  <chr>       <chr>         <date>       <date>       <chr>
##  1  1916 Summer Berlin      Germany       NA           NA           Not held du~
##  2  1940 Winter Garmisch-P~ Germany       NA           NA           Not held du~
##  3  1940 Summer Helsinki    Finland       NA           NA           Not held du~
##  4  1944 Winter Cortina d'~ Italy         NA           NA           Not held du~
##  5  1944 Summer London      Great Britain NA           NA           Not held du~
##  6  2022 Winter Beijing     People's Rep~ NA           NA           <NA>
##  7  2024 Summer Paris       France        NA           NA           <NA>
##  8  2026 Winter Milano-Cor~ Italy         NA           NA           <NA>
##  9  2028 Summer Los Angeles United States NA           NA           <NA>
## 10  2032 Summer Brisbane    Australia     NA           NA           <NA>
```

Create a new variable that contains the athletes' `birthday`, formatted as a date column. Then create another variable that holds the age of athletes in years at the opening of the Olympic Games.

```
df_bday_age <- games_athletes_results%>%
  separate(born, c("birthday", "birth_city"), "in")%>%
  mutate(birthday = dmy(birthday))

df_bday_age <- df_bday_age%>%
  mutate (age = round(interval(birthday, games_opened)/dyears(1)))
df_bday_age
```

```
## # A tibble: 291,710 x 24
##    athlete name    sex   birthday   birth_city    died        affiliations title
##      <dbl> <chr>   <chr> <date>     <chr>         <chr>       <chr>        <chr>
##  1       1 Jean-F~ Male  1886-12-12 " Bordeaux, ~ 2 October ~ <NA>         <NA>
##  2       1 Jean-F~ Male  1886-12-12 " Bordeaux, ~ 2 October ~ <NA>         <NA>
##  3       1 Jean-F~ Male  1886-12-12 " Bordeaux, ~ 2 October ~ <NA>         <NA>
##  4       1 Jean-F~ Male  1886-12-12 " Bordeaux, ~ 2 October ~ <NA>         <NA>
##  5       1 Jean-F~ Male  1886-12-12 " Bordeaux, ~ 2 October ~ <NA>         <NA>
##  6       2 Arnaud~ Male  1969-04-01 " Meulan, Yv~ <NA>        Racing CF, ~ <NA>
##  7       2 Arnaud~ Male  1969-04-01 " Meulan, Yv~ <NA>        Racing CF, ~ <NA>
##  8       3 Jean•B~ Male  1898-08-13 " Biarritz, ~ 17 July 19~ TCP, Paris ~ <NA>
##  9       3 Jean•B~ Male  1898-08-13 " Biarritz, ~ 17 July 19~ TCP, Paris ~ <NA>
## 10       3 Jean•B~ Male  1898-08-13 " Biarritz, ~ 17 July 19~ TCP, Paris ~ <NA>
## # ... with 291,700 more rows, and 16 more variables: measurements <chr>,
## #   year <dbl>, season <chr>, sport <chr>, discipline <chr>, pos <dbl>,
```

```
## #   medal <chr>, country <chr>, country_code <chr>, team <chr>,
## #   games_city <chr>, games_country <chr>, games_opened <date>,
## #   games_closed <date>, games_remark <chr>, age <dbl>
```

Calculate the average age per sport of the female participants. Then print out a ranking of the 10 sports with the lowest average age

```
df_female_age <- df_bday_age %>%
  group_by(sport)%>%
  filter(sex == 'Female'& !is.na(age))%>%
  summarize(avg_age = mean(age),n = n())%>%
  arrange(avg_age)
df_female_age[1:10,]
```

```
## # A tibble: 10 x 3
##    sport                      avg_age     n
##    <chr>                        <dbl> <int>
##  1 Rhythmic Gymnastics           19.5   747
##  2 Artistic Gymnastics           19.8  9453
##  3 Swimming                      20.1 10734
##  4 Skateboarding                 21.2    40
##  5 Figure Skating                21.4  1303
##  6 Diving                        22.2  1278
##  7 Ski Jumping                   22.5    65
##  8 Alpine Skiing                 23.0  3782
##  9 Short Track Speed Skating     23.0   918
## 10 Artistic Swimming             23.1  1029
```

Task 4 (10 points)

Calculate the medal table of the Olympic Summer Games 2016 in Rio de Janeiro and display the top 10 countries (ordered by Gold, Silver, and then Bronze medals). Your final table should look like this: https://en.wikipedia.org/wiki/2016_Summer_Olympics_medal_table. Hint 1: In team sports such as Handball, many players receive a gold medal, but for the countries' medal table it only counts as one gold medal. You can recognize team sports by the fact that the variable `team` is not missing.

```
summer_games_2016 <- games_athletes_results%>%
  filter(year == 2016 & season == "Summer" )%>%
  pivot_wider(names_from = medal, values_from = medal)

summer_games_2016_individual <- summer_games_2016 %>%
  filter(is.na(team)) %>%
  group_by(country)%>%
  summarise(Gold   = sum(!is.na(Gold)),
            Silver = sum(!is.na(Silver)),
            Bronze = sum(!is.na(Bronze)))

summer_games_2016_team <- summer_games_2016 %>%
  filter(!is.na(team)) %>%
  select(sport,discipline,country, Gold, Silver, Bronze)%>%
  distinct(.keep_all = TRUE)%>%
  group_by(country)%>%
```

```
  summarise(Gold   = sum(!is.na(Gold)),
            Silver = sum(!is.na(Silver)),
            Bronze = sum(!is.na(Bronze)))

medal_table_summer_2016 <- summer_games_2016_individual%>%
  left_join(summer_games_2016_team, by ='country')

# replace all NA with 0 in order to calculate the sum of the medals
medal_table_summer_2016[is.na(medal_table_summer_2016)] = 0

medal_table_summer_2016 <- medal_table_summer_2016 %>%
  mutate(Gold = Gold.x + Gold.y,
         Silver = Silver.x + Silver.y,
         Bronze = Bronze.x + Bronze.y)%>%
  select(country, Gold, Silver, Bronze)%>%
  mutate(Total = Gold + Silver + Bronze) %>%
  arrange(desc(Gold),desc(Silver), desc(Bronze))


medal_table_summer_2016[1:10,]
```

```
## # A tibble: 10 x 5
##    country                     Gold Silver Bronze Total
##    <chr>                      <int>  <int>  <int> <int>
##  1 United States                 46     37     38   121
##  2 Great Britain                 27     23     17    67
##  3 People's Republic of China    26     18     26    70
##  4 Russian Federation            19     17     20    56
##  5 Germany                       17     10     15    42
##  6 Japan                         12      8     21    41
##  7 France                        10     18     14    42
##  8 Republic of Korea              9      3      9    21
##  9 Italy                          8     12      8    28
## 10 Australia                      8     11     10    29
```

Task 5 (5 points)

Some of the athletes have started for multiple countries (e.g. due to migration or other reasons). Show all athletes that have started for at least 4 countries. Then choose one of these athletes and display the 4 countries for which he or she started.

```
games_athletes_results%>%
  mutate(UID = paste(name, athlete, country)) %>%
  distinct(UID, .keep_all=TRUE) %>%
  group_by(name,athlete) %>%
  summarise(name = first(name), countries = n()) %>%
  arrange(countries) %>%
  filter(countries >= 4)
```

```
## # A tibble: 5 x 3
## # Groups:   name [5]
##   name                athlete countries
```

```
##   <chr>                  <dbl>     <int>
## 1 Ilija•Lupulesku         2821         4
## 2 Irina•Lashko           50029         4
## 3 Jasna•Šekaric          44849         4
## 4 Makharbek•Khadartsev   59866         4
## 5 Michal•Sliwinski       10927         4
```

```
games_athletes_results%>%
    filter(athlete == 59866)%>%
    select(country)
```

```
## # A tibble: 4 x 1
##   country
##   <chr>
## 1 Uzbekistan
## 2 Russian Federation
## 3 Unified Team
## 4 Soviet Union
```