# Dplyr Practice questions

2022-09-05

## 1 IMDB movies

```
df <- read_csv("../Datasets/movie_metadata.csv")
```

```
## Rows: 5043 Columns: 28
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (12): color, director_name, actor_2_name, genres, actor_1_name, movie_ti...
## dbl (16): num_critic_for_reviews, duration, director_facebook_likes, actor_3...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

1. *How many movies are there in the dataset?*

```
paste0("There are ", nrow(df), " movies in the dataset.")
```

```
## [1] "There are 5043 movies in the dataset."
```

2. *Each movie has a unique IMDB ID. For example, in the link http://www.imdb.com/title/tt0449088/?ref_=fn_tt_tt_1, the id is 0449088. Create a column that stores this ID.*

Multiple ways to extract the ID, each represented by a function below. Either answer is fine.

```
f1 <- function(x) {
    x %>% str_split("/") %>% sapply(function(y) {y[5]}) %>% str_remove("tt")
}
f2 <- function(x) {
    x %>% str_extract("tt\\d+/")%>% str_remove("tt") %>% str_remove("/")
}
f3 <- function(x) {
    x %>% str_extract("\\d+/") %>% str_remove("/")
}

df <- df %>%
    mutate(ID1 = movie_imdb_link %>% f1,
           ID2 = movie_imdb_link %>% f2,
           ID3 = movie_imdb_link %>% f3)
# All 3 columns have the same ID

all(df$ID1 == df$ID2)
```

```
## [1] TRUE
```

```
all(df$ID2 == df$ID3)
```

```
## [1] TRUE
```

3. *ID should be unique for each movie. Check if that is true. Hint: use group_by, summarise.*

If any movie has multiple IDs or any ID has multiple movies, then the combinations of movie names and IDs should have repeats in either ID column or movie column.

```
dfMICount <- df %>% group_by(movie_title, ID1) %>% summarise(Count = n(), .groups = "drop") %>%
    arrange(desc(Count))
dfMICount %>% head
```

```
## # A tibble: 6 x 3
##   movie_title               ID1     Count
##   <chr>                     <chr>   <int>
## 1 Ben-Hur                   2638144     3
## 2 Halloween                 0077651     3
## 3 Home                      2224026     3
## 4 King Kong                 0360717     3
## 5 Pan                       3332064     3
## 6 The Fast and the Furious  0232500     3
```

Clearly, there are repeats in the dataframe. Now, are there any duplicate IDs in the new data frame? If so, multiple movies have the same ID.

```
dfMI_ICount <- dfMICount %>% group_by(ID1) %>%summarise(Count = n()) %>%
    filter(Count > 1)
dfMI_ICount
```

```
## # A tibble: 0 x 2
## # ... with 2 variables: ID1 <chr>, Count <int>
## # i Use ‘colnames()‘ to see all variable names
```

Therefore, each movie does have a unique ID, and some movies have multiple entries. Similarly, any duplicate movies?

```
dfMI_MCount <- dfMICount %>% group_by(movie_title) %>%summarise(Count = n()) %>%
    filter(Count > 1)
dfMI_MCount
```

```
## # A tibble: 3 x 2
##   movie_title      Count
##   <chr>            <int>
## 1 Out of the Blue      2
## 2 The Dead Zone        2
## 3 The Host             2
```

Hence, each ID has a unique movie. so all ID-movie combinations are unique.

4. *Make a list of all keywords used in the dataset. Which keywords are repeated most often? (Note: this is not a dplyr based question.)*

Keywords are stored in the "plot_keyword" column. Each keyword is separated by a "|". Therefore, we write:

```
keyWords <- df$plot_keywords %>% str_split("\\|", ) %>% unlist
keyWordFrequency <- keyWords %>% table %>% sort(decreasing = T)
head(keyWordFrequency)
```

```
## .
##          love        friend        murder         death        police
##           198           166           161           132           126
## new york city
##            91
```

Another solution ignoring th regex ("\\|")

```
keyWords <- df$plot_keywords %>% str_split(fixed("|"), ) %>% unlist
keyWordFrequency <- keyWords %>% table %>% sort(decreasing = T)
head(keyWordFrequency)
```

```
## .
##          love        friend        murder         death        police
##           198           166           161           132           126
## new york city
##            91
```

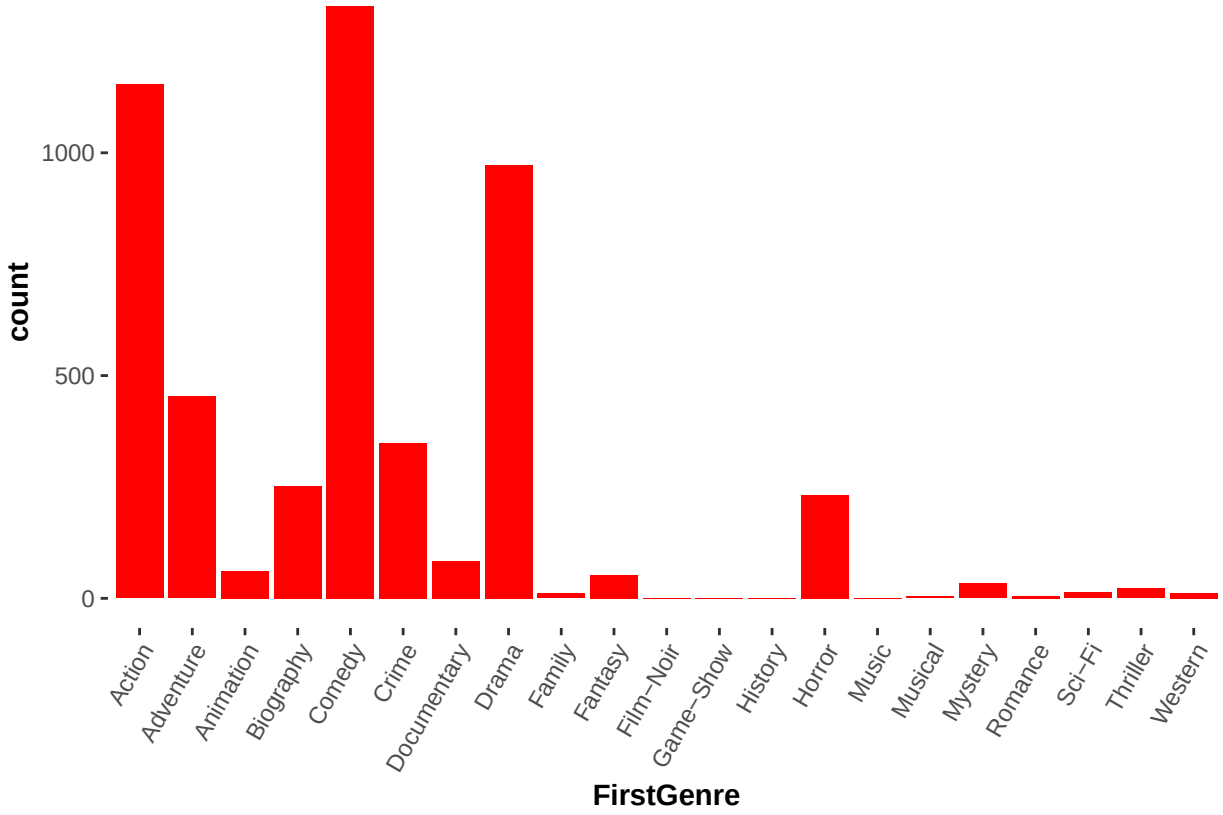
5. *Create a column that contains the first genre in the list of genres used for a given movie.*

Delete all text after the first "|"

```
df <- df %>% mutate(FirstGenre = genres %>% str_remove("\\|.*"))
```

6. *Create a barplot for the first genres, with the following conditions:*
    a. *Keep the panel background blank*
    b. *On x-axis, keep the text at an angle of 60, set the hjust and vjust arguments to 1*
    c. *Make axis titles bold*
    d. *Remove the grid.*
    e. *Fill the bars with red color*

```
df %>% ggplot(aes(x = FirstGenre)) + geom_bar(fill = "red") +
    theme(panel.background = element_blank(),
          axis.text.x = element_text(angle = 60, hjust = 1, vjust = 1),
          axis.title = element_text(face = "bold"),
          panel.grid = element_blank())
```




7. *Find the average IMDB rating for each director. Which director has the highest average IMDB rating?*

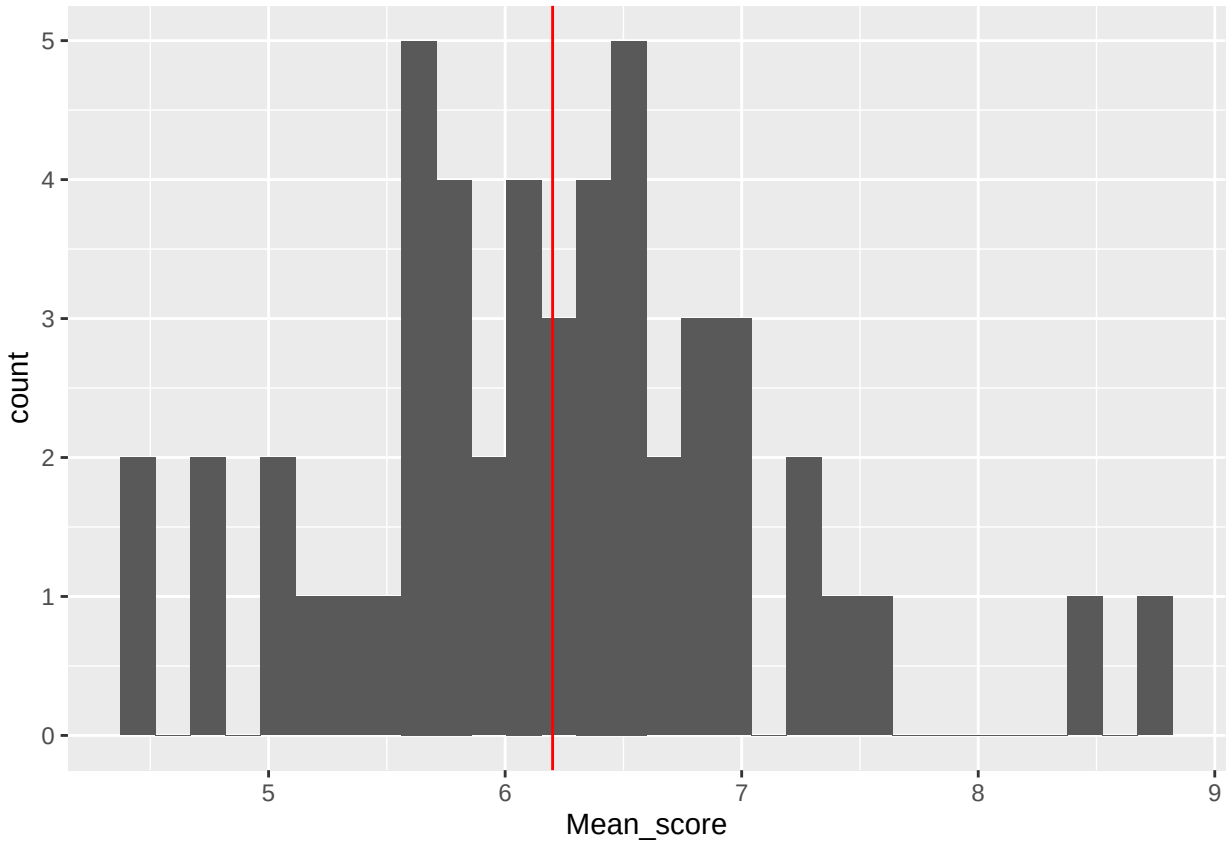
```
df %>% group_by(director_name) %>%
    summarise(Mean_score = mean(imdb_score, na.rm = T)) %>%
    arrange(desc(Mean_score)) %>%
    head(5)
```

```
## # A tibble: 5 x 2
##   director_name    Mean_score
##   <chr>                 <dbl>
## 1 John Blanchard          9.5
## 2 Cary Bell               8.7
## 3 Mitchell Altieri        8.7
## 4 Sadyk Sher-Niyaz        8.7
## 5 Charles Chaplin         8.6
```

8. *English movies are made in multiple countries. Is there a bias in terms of the IMDB score of English movies for a given country?*

```
dfCountry <- df %>% filter(language == "English") %>%
    group_by(country) %>%
    summarise(Mean_score = mean(imdb_score))
ggplot(dfCountry, aes(x = Mean_score)) + geom_histogram() +
    geom_vline(xintercept = mean(dfCountry$Mean_score), color = "red")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




Distribution of the mean scores is symmetric, so there is likely no bias.

# 2 Quality of life dataset

```
df <- read_csv("../Datasets/quality_of_life.csv")
```

```
## Rows: 240 Columns: 11
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): City
## dbl (10): Rank, Quality of Life Index, Purchasing Power Index, Safety Index,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

1. *How many cities are there in the dataset? How many quality indices?*

```
paste0("The number of cities is ", nrow(df))
```

```
## [1] "The number of cities is 240"
```

```
paste0("The number of quality indices is ", ncol(df)-2)
```

```
## [1] "The number of quality indices is 9"
```

2. *Notice that there are spaces in the column names. Having spaces in variable names is not desirable. Replace spaces with "_".*

```
colnames(df) <- colnames(df) %>% str_replace_all(" ", "_")
```

3. *Arrange the dataset based on the City.*

```
df2 <- df %>% arrange(City)
```

4. *A ranking has been assigned to each city. Which column was used to assign that? In other words, sorting by which column gives you the same dataframe?*

```
indices <- colnames(df)[-c(1,2)]
id <- sapply(1:length(indices), function(x) {
    df1 <- df
    colnames(df)[x + 2] <- "Col"
    df3 <- df %>% arrange(desc(Col))
    all(df3 == df)
})
paste0("The cities are ranked based on ", indices[id])
```

```
## [1] "The cities are ranked based on Quality_of_Life_Index"
```

5. *Create column containing the country of each city.*

```
df <- df %>% mutate(Country = City %>% str_split(",") %>%
                        sapply(function(x) {x[2]}))
```

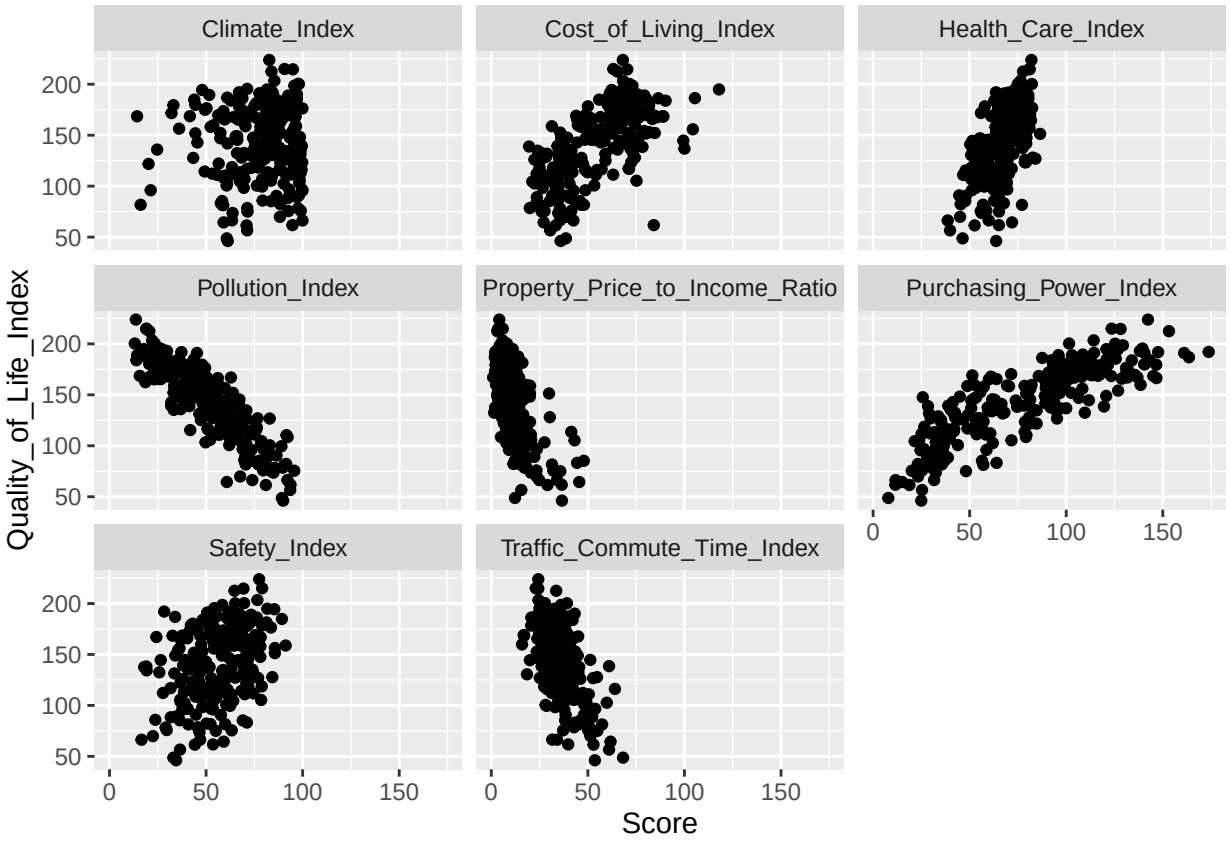
6. *Find the average Quality of life index for each country. Which country is better to live in?*

```
df %>% group_by(Country) %>%
    summarise(Mean_index = mean(Quality_of_Life_Index)) %>%
    arrange(desc(Mean_index)) %>%
    head
```

```
## # A tibble: 6 x 2
##   Country        Mean_index
##   <chr>               <dbl>
## 1 " NC"                203.
## 2 " Australia"         202.
## 3 " Netherlands"       194.
## 4 " NM"                192.
## 5 " OK"                191.
## 6 " Switzerland"       191.
```

7. *Make scatter plot of the Quality of life index (y axis) against the other indices (x-axis, one plot for each index). Use the same conditions as above. Which index can explain the quality of life best?*

```
df3 <- df %>% gather(key = "Index", value = "Score", -Rank, -Quality_of_Life_Index, -City, -Country)
ggplot(df3, aes(x = Score, y = Quality_of_Life_Index)) +
    geom_point() +
    facet_wrap(~Index, ncol = 3)
```




Purchasing power index has the best relationship with the quality of life index.