DA5020.A6.Hsiao-Yu.Peng

Hsiao-Yu Peng

2023-10-22

Bonus 1: See the attachement.

Bonus 2: What is the average runtime for the thriller movie genre.

```
db_path <- "~/Downloads/sqlite-tools-osx-x86-3430200/imdb.db"

# Connect to the database
con <- dbConnect(SQLite(), dbname = db_path)

# Define the query
query <- "SELECT Genre, AVG(Runtime) AS Avg_Runtime
          FROM movie_info
          WHERE Genre = 'Thriller';"

# Execute the query and fetch the result
result <- dbGetQuery(con, query)

# Close the database connection
dbDisconnect(con)

# print the result
print(result)
```

```
##      Genre Avg_Runtime
## 1 Thriller         108
```

The average runtime for the thriller movie genre is 108 minutes.

Q1

Q1-1 sql create the table director_info

#CREATE TABLE director_info (
# Director_ID INT AUTO_INCREMENT PRIMARY KEY,
# Director_Name VARCHAR(255) NOT NULL
#);

Q1-2 sql import csv to director_info

#.mode csv
#.import –skip 1 directors.csv director_info

- Verify the imported data

#SELECT * FROM director_info;

Q2

Q2-1. Count the number of rows in the movie_info and director_info tables.

```
db_path <- "~/Downloads/sqlite-tools-osx-x86-3430200/imdb.db"
# Connect to the database
con <- dbConnect(SQLite(), dbname = db_path)

# query
query_movie_count <- "Select COUNT(*) from movie_info;"
query_director_count <- "Select Count(*) from director_info;"

# fetch the results
movie_count <- dbGetQuery(con, query_movie_count)
director_count <- dbGetQuery(con, query_director_count)

# Close the database connection
dbDisconnect(con)


# Print out the counts
cat("Number of rows in movie_info:", movie_count[1, 1], "\n")
```

```
## Number of rows in movie_info: 1000
```

```
cat("Number of rows in director_info:", director_count[1, 1], "\n")
```

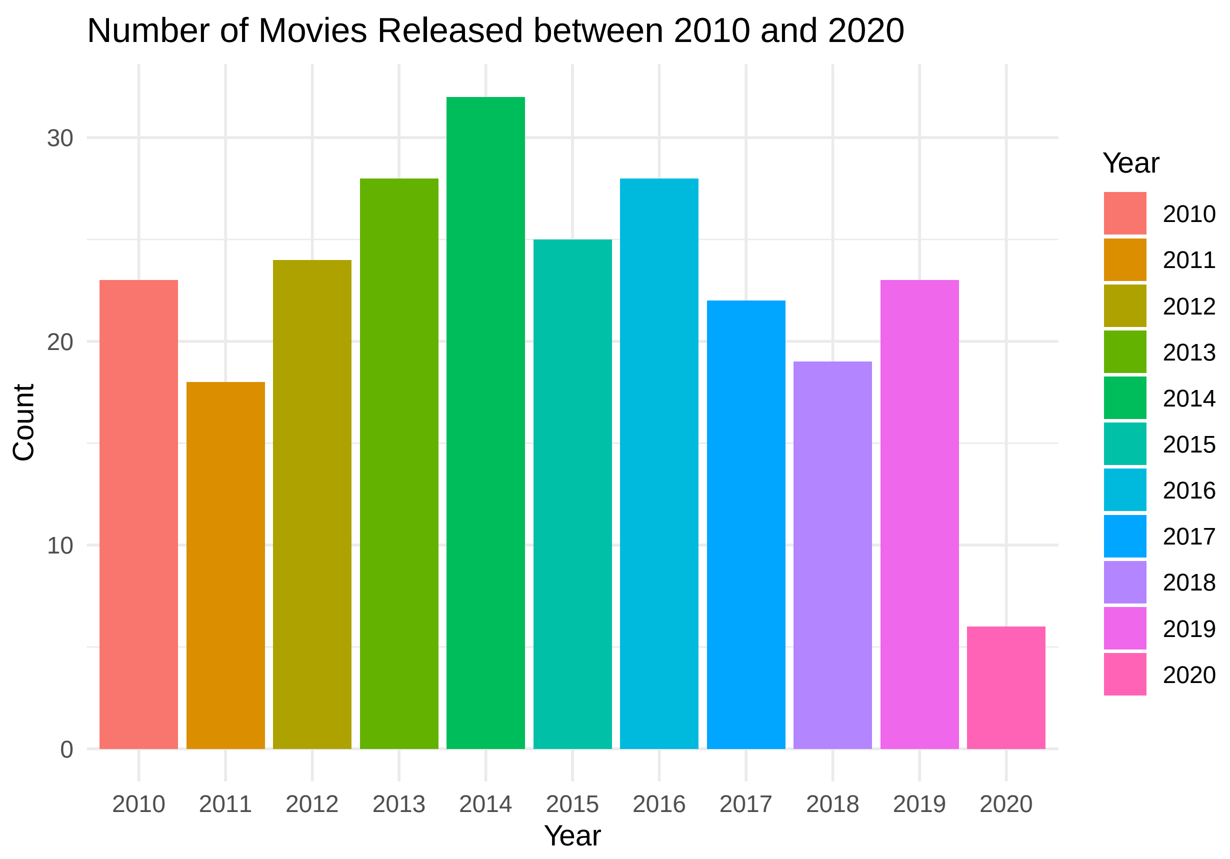
```
## Number of rows in director_info: 548
```

Q2-2. How many movies were released between 2010 and 2020 (inclusive)? Visualize the results.

```r
# Define the path to the SQLite database
db_path <- "~/Downloads/sqlite-tools-osx-x86-3430200/imdb.db"

# Connect to the database
con <- dbConnect(SQLite(), dbname = db_path)

# Define the SQL query
query <- "SELECT COUNT(*) as Count, Release_Year as Year
          FROM movie_info
          WHERE CAST(Release_Year AS INTEGER) BETWEEN 2010 AND 2020
          GROUP BY Year;"

# Execute the query and fetch the results
result <- dbGetQuery(con, query)
print(result)
```

```
##    Count Year
## 1     23 2010
## 2     18 2011
## 3     24 2012
## 4     28 2013
## 5     32 2014
## 6     25 2015
## 7     28 2016
## 8     22 2017
## 9     19 2018
## 10    23 2019
## 11     6 2020
```

```r
# Visualize the results
ggplot(result, aes(x = Year, y = Count, fill = Year)) +
  geom_bar(stat = "identity") +
  labs(title = "Number of Movies Released between 2010 and 2020",
       x = "Year", y = "Count") +
  theme_minimal()
```



```r
# Close the database connection
dbDisconnect(con)
```

Q2-3 What is the minimum, average and maximum ratings for "Action" movies. Ensure that you query the genre using wild cards.

```r
# Define the path to the SQLite database
db_path <- "~/Downloads/sqlite-tools-osx-x86-3430200/imdb.db"

# Connect to the database
con <- dbConnect(SQLite(), dbname = db_path)

# Define the SQL query
query <- "SELECT MIN(IMDB_Rating),
                 AVG(IMDB_Rating),
                 MAX(IMDB_Rating)
          FROM movie_info
          WHERE Genre LIKE '%Action%';"

# Execute the query and fetch the results
result <- dbGetQuery(con, query)
```

```
# Print the results
print(result)
```

```
##   MIN(IMDB_Rating) AVG(IMDB_Rating) MAX(IMDB_Rating)
## 1              7.6         7.948677                9
```

```
# Close the database connection
dbDisconnect(con)
```

For Action movies, the minimum rating is 7.6, the average rating is 7.94, and the maximum rating is 9.

Q2-4 What are the 25 highest-grossing movies within the dataset? Display the title, genre and gross.

```
# Define the path to the SQLite database
db_path <- "~/Downloads/sqlite-tools-osx-x86-3430200/imdb.db"

# Connect to the database
con <- dbConnect(SQLite(), dbname = db_path)

# Define the SQL query
query <- "SELECT Series_Title, Genre, CAST(Gross AS NUMERIC) AS Gross
          FROM movie_info
          WHERE Gross IS NOT NULL
          ORDER BY Gross DESC
          LIMIT 25;"

# Execute the query and fetch the results
result <- dbGetQuery(con, query)

# Print the results
print(result)
```

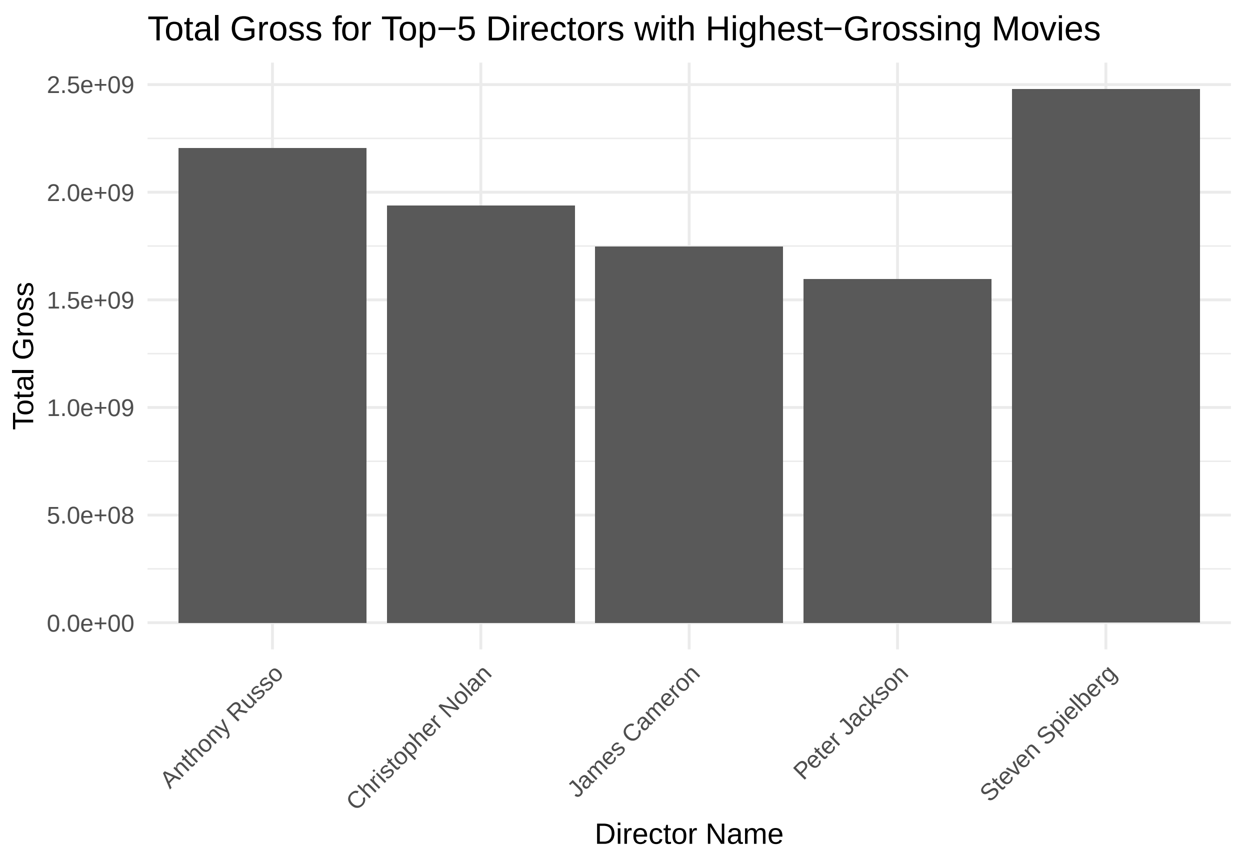
```
##                                   Series_Title                        Genre
## 1     Star Wars: Episode VII - The Force Awakens    Action, Adventure, Sci-Fi
## 2                            Avengers: Endgame     Action, Adventure, Drama
## 3                                       Avatar   Action, Adventure, Fantasy
## 4                       Avengers: Infinity War    Action, Adventure, Sci-Fi
## 5                                      Titanic              Drama, Romance
## 6                                 The Avengers    Action, Adventure, Sci-Fi
## 7                                 Incredibles 2 Animation, Action, Adventure
## 8                              The Dark Knight         Action, Crime, Drama
## 9                                    Rogue One    Action, Adventure, Sci-Fi
## 10                       The Dark Knight Rises            Action, Adventure
## 11                    E.T. the Extra-Terrestrial              Family, Sci-Fi
## 12                                  Toy Story 4 Animation, Adventure, Comedy
## 13                                The Lion King  Animation, Adventure, Drama
## 14                                  Toy Story 3 Animation, Adventure, Comedy
## 15                    Captain America: Civil War    Action, Adventure, Sci-Fi
## 16                                     Jurassic Park    Action, Adventure, Sci-Fi
## 17                  Guardians of the Galaxy Vol. 2    Action, Adventure, Comedy
## 18  Harry Potter and the Deathly Hallows: Part 2    Adventure, Drama, Fantasy
## 19                                      Finding Nemo Animation, Adventure, Comedy
## 20 The Lord of the Rings: The Return of the King     Action, Adventure, Drama
## 21                                          Deadpool    Action, Adventure, Comedy
## 22                                        Inside Out Animation, Adventure, Comedy
## 23         The Lord of the Rings: The Two Towers     Action, Adventure, Drama
## 24                                          Zootopia Animation, Adventure, Comedy
## 25                                             Joker       Crime, Drama, Thriller
##        Gross
## 1  936662225
## 2  858373000
## 3  760507625
## 4  678815482
## 5  659325379
## 6  623279547
## 7  608581744
## 8  534858444
## 9  532177324
## 10 448139099
## 11 435110554
## 12 434038008
## 13 422783777
## 14 415004880
## 15 408084349
## 16 402453882
## 17 389813101
## 18 381011219
## 19 380843261
## 20 377845905
## 21 363070709
## 22 356461711
## 23 342551365
## 24 341268248
## 25 335451311
```

```r
# Close the database connection
dbDisconnect(con)
```

Q2-5. Which directors have the highest-grossing movies. Display the director name and the total gross. Ensure that you join the necessary tables. Visualize the results using a Bar chart.

```r
# Define the path to the SQLite database
db_path <- "~/Downloads/sqlite-tools-osx-x86-3430200/imdb.db"

# Connect to the database
con <- dbConnect(SQLite(), dbname = db_path)
```

```r
query <- "SELECT d.Director_Name, SUM(CAST(Gross AS NUMERIC)) AS Total_Gross
          FROM director_info d
          JOIN movie_info m ON d.Director_ID = m.Director_ID
          WHERE m.Gross IS NOT NULL
          GROUP BY Director_Name
          ORDER BY Total_Gross DESC
          LIMIT 5;"

# Execute the query and fetch the results
result <- dbGetQuery(con, query)

# Close the database connection
dbDisconnect(con)

# Visualize the results with a bar chart
ggplot(result, aes(x = Director_Name, y = Total_Gross)) +
  geom_bar(stat = "identity") +
  labs(title = "Total Gross for Top-5 Directors with Highest-Grossing Movies",
       x = "Director Name", y = "Total Gross") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Steven Spielberg has the highest-grossing movies.

Q2-6 Create a function called verifyDirector() that takes a director name as its argument, and queries the database to check if the director exists. Your function should display a message to notify the user if the director was found or not.

```r
verifyDirector <- function(director_name) {
  # Define the path to the SQLite database
  db_path <- "~/Downloads/sqlite-tools-osx-x86-3430200/imdb.db"

  # Connect to the database
  con <- dbConnect(SQLite(), dbname = db_path)

  # Define the SQL query
  query <- sprintf("SELECT 1
                    FROM director_info
                    WHERE Director_Name = '%s'
                    LIMIT 1;", director_name)

  # Execute the query and fetch the result
  result <- dbGetQuery(con, query)

  # Close the database connection
  dbDisconnect(con)

  # Check the result and display a message
  if (nrow(result) > 0) {
    cat(sprintf("Director '%s' found in the database.\n", director_name))
  } else {
    cat(sprintf("Director '%s' not found in the database.\n", director_name))
  }
}

# Example for the function
verifyDirector("Steven Spielberg")
```

```
## Director 'Steven Spielberg' found in the database.
```

```r
verifyDirector("Ben Tasker")
```

```
## Director 'Ben Tasker' not found in the database.
```