# SQLite

Rebecca Weiss

10/19/2021

---

```
#load all necessary libraries
library(RSQLite)
library(DBI)
library(tidyverse)
```

Clear the workspace:

```
rm(list = ls())
```

## 1. This question should be done completely in the SQLite Console, not in R. Start by loading the imdb.db file using the console and download the directors.csv file (from Canvas). Perform the following tasks:

1. (5pts) Create a table named director_info using SQLite; the columns are: Director_ID, and Director_Name. The Director_ID should be the primary key.
2. (5pts) Import the entire data from the CSV file into the director_info table using the SQLite .import command (see helpful resources below). Verify that your data was imported correctly. Copy the queries above into a comment chunk in your Rmd file

```
# /Applications/sqlite3 imdb.db
# .open "imdb.db"
# .tables

# create table director_info(
#   Director_ID TEXT PRIMARY KEY,
#   Director_Name TEXT);
# .separator ','
# .import directors.csv director_info

## verify it loaded correctly
# select * from director_info;
```

**2. This question should be done in RStudio. Connect to the database, using R, and write queries to answer the questions below (answer each question in a separate R chunk). Do not load the entire database or its tables in your R environment.**

```
db <- dbConnect(SQLite(), db="imdb.db")

# make sure tables are there
dbListTables(db)
```

```
## [1] "director_info" "movie_info"
```

1. (5 pts) Count the number of rows in the movie_info and director_info tables.

```
# rows from movie_info
n_movie <- dbFetch(dbSendQuery(db, "SELECT COUNT(*) FROM movie_info;"))
print(paste("The number of rows in movie_info =", n_movie))
```

```
## [1] "The number of rows in movie_info = 1000"
```

```
# rows from director_info
n_director <- dbFetch(dbSendQuery(db, "SELECT COUNT(*) FROM director_info;"))
print(paste("The number of rows in director_info =", n_director))
```

```
## [1] "The number of rows in director_info = 549"
```

2. (5 pts) How many movies were released between 2010 and 2020 (inclusive)? Visualize the results.

```
# first, make sure we know the movie column names
dbGetQuery(db, "PRAGMA table_info(movie_info)")$name
```

```
## [1] "ID"           "Series_Title" "Release_Year" "Runtime"      "Genre"
## [6] "IMDB_Rating"  "Director_ID"  "Gross"
```

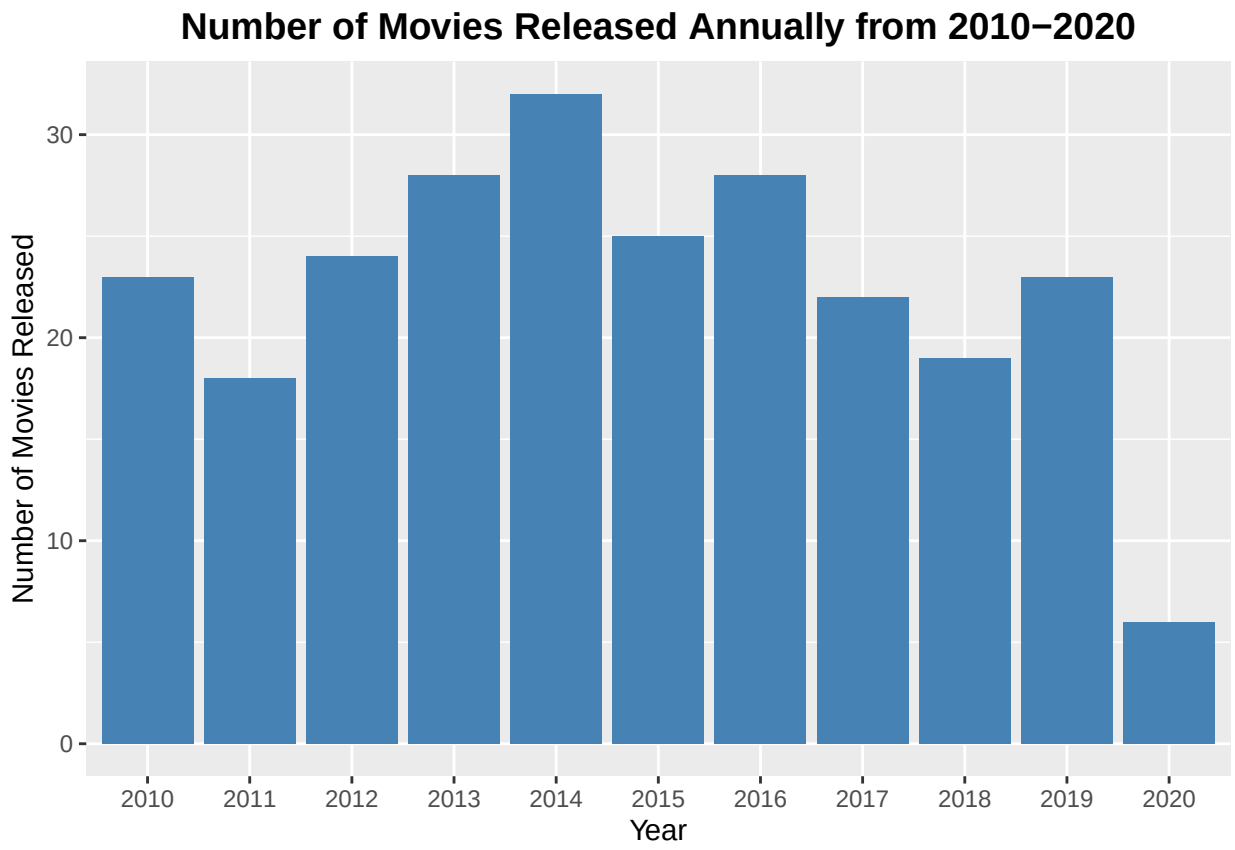
```
# know column name "Release_Year", now can filter + visualize
number_movie <- dbGetQuery(db, 'SELECT "Release_Year",  count(*)
FROM movie_info
WHERE "Release_Year" > 2009 AND "Release_Year" < 2021
GROUP BY "Release_Year"')

(number_movie <- number_movie %>%
    rename(movies =`count(*)`))
```

```
##    Release_Year movies
## 1          2010     23
## 2          2011     18
## 3          2012     24
## 4          2013     28
## 5          2014     32
```

```
## 6           2015      25
## 7           2016      28
## 8           2017      22
## 9           2018      19
## 10          2019      23
## 11          2020       6
```

```
ggplot(number_movie, aes(x=Release_Year, y=movies)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(title = "Number of Movies Released Annually from 2010-2020",
       x = "Year",
       y = "Number of Movies Released") +
    theme(plot.title = element_text(size = 14, hjust = 0.5, face = "bold"))
```




3. (5 pts) What is the minimum, average and maximum ratings for "Action" movies. Ensure that you query the genre using wild cards.

```
dbGetQuery(db, 'SELECT max(IMDB_Rating), min(IMDB_Rating), avg(IMDB_Rating)
FROM movie_info
WHERE Genre LIKE "%action%";')
```

```
##   max(IMDB_Rating) min(IMDB_Rating) avg(IMDB_Rating)
## 1                9              7.6         7.948677
```

4. (5 pts) What are the 25 highest-grossing movies within the dataset? Display the title, genre and gross.

```
dbGetQuery(db, 'SELECT Series_Title, Genre, Gross
FROM movie_info
WHERE Gross != "NA"
ORDER BY Gross DESC LIMIT 25;')
```

```
##                                     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
```

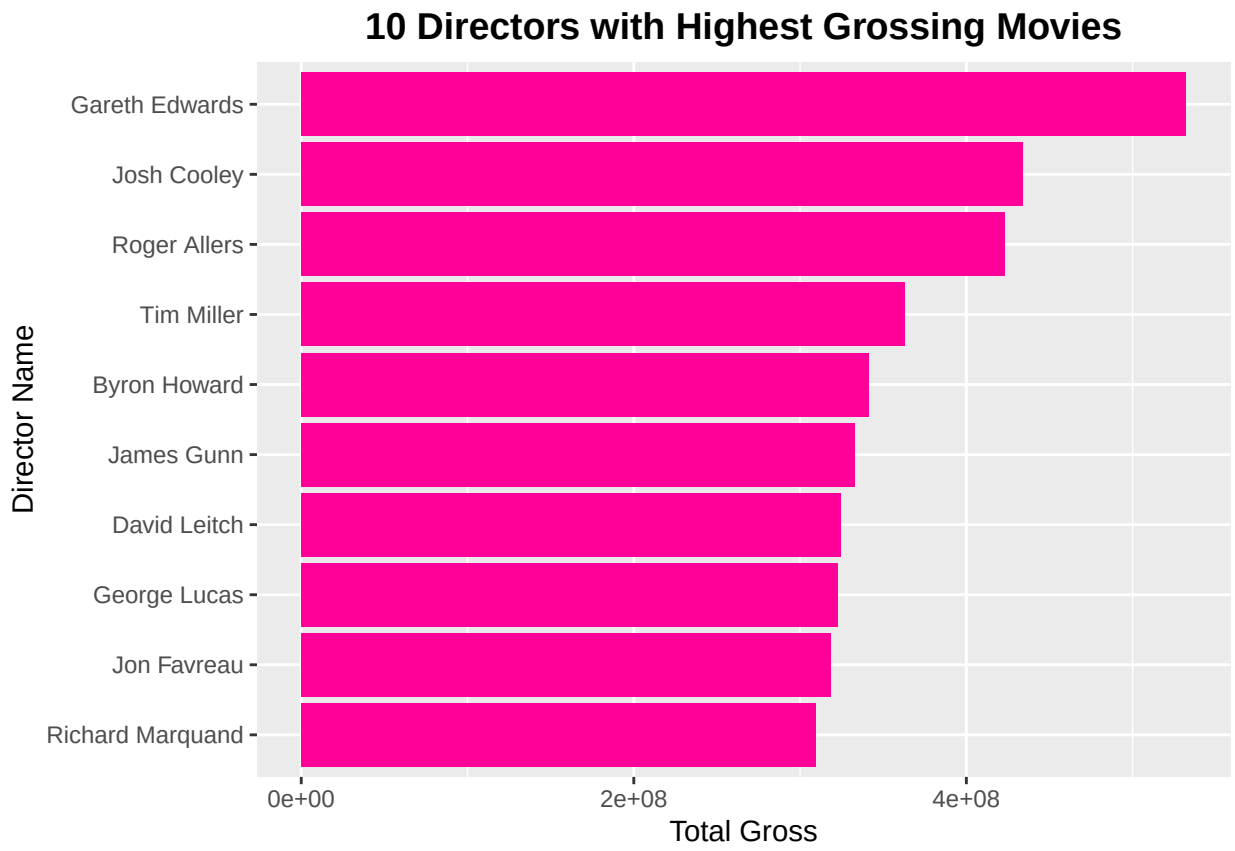
```
## 22 356461711
## 23 342551365
## 24 341268248
## 25 335451311
```

5. (10 pts) 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.

```
# get data for top 10 directors with highest grossing movies
(top_directors <- dbGetQuery(db, 'SELECT Director_Name, Gross
FROM movie_info
INNER JOIN director_info on
movie_info.Director_ID=director_info.Director_ID
WHERE Gross != "NA"
GROUP BY Director_Name
ORDER BY Gross DESC LIMIT 10;'))
```

```
##       Director_Name     Gross
## 1    Gareth Edwards 532177324
## 2       Josh Cooley 434038008
## 3      Roger Allers 422783777
## 4        Tim Miller 363070709
## 5      Byron Howard 341268248
## 6        James Gunn 333176600
## 7      David Leitch 324591735
## 8      George Lucas 322740140
## 9       Jon Favreau 318412101
## 10 Richard Marquand 309125409
```

```
ggplot(top_directors, aes(x=reorder(Director_Name, Gross), y=Gross)) +
  geom_bar(stat = "identity", fill = "#FF0099FF") +
  labs(title = "10 Directors with Highest Grossing Movies",
       x = "Director Name",
       y = "Total Gross") +
  theme(plot.title = element_text(size = 14, hjust = 0.5, face = "bold")) +
  coord_flip()
```




6. (10 pts) 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.

```
verifyDirector <- function(director_name)  {
  directors <- dbGetQuery(db, 'SELECT Director_Name
FROM director_info;')

  if (any(directors$Director_Name == director_name)) {
    print("Director name entered is in database")
  } else  {
    print("Director name is not in database")
  }
}
```

```
# verify it works:
verifyDirector("Peter Jackson") # should exist
```

```
## [1] "Director name entered is in database"
```

```
verifyDirector("Willy Wonka") # should not exist
```

```
## [1] "Director name is not in database"
```

### 3. What is the average runtime for the thriller movie genre.

```
dbGetQuery(db, 'SELECT  avg(Runtime)
FROM movie_info
WHERE Genre LIKE "%thriller%";')
```

```
##   avg(Runtime)
## 1     119.6423
```

From the output, the average run time of the thriller genre is ~119 minutes.