# ROI of an Oscar

#### November 30, 2017 | Runjini Murthy | Sprint 2

#### Description

The goal of this sprint is to demonstrate the effect of an Oscar win on immediate box office performance following the win. This will be achieved by determining the necessary data, setting up data models, creating a database of the relevant tables, and then querying for results.

## Skill Backlog User Story

As a budding data analyst, I need to be able to construct data models so I can structure the format of the database I'd like to query.

## Project Proposal

My project has three main components: One, I need to construct a data model to represent databases for movies and award categories. Two, I need to build these databases in SQLite. Three, I would like to be able to query the database(s) using SQL.

## Key Questions

- What does my data model look like?

- What are the basic commands I need to understand in R? (SKIP THIS; NOT FOCUSING ON R THIS WEEK)

- What are the libraries/packages I need to install in R? (SKIP THIS; NOT FOCUSING ON R THIS WEEK)

- How do I install packages in R? (SKIP THIS; NOT FOCUSING ON R THIS WEEK)

- What is the script to scrape the web? (SKIP THIS; NOT FOCUSING ON SCRAPING THIS WEEK)

- How do I output a database/table based on the data model I want to create?

- How do I translate this table into SQLite or something I can access with SQL?

- What exactly is SQL, and what environment/program/format do I need to use it?

- What are basic SQL commands?

- My query is: What is the box office performance of a movie resulting in an Oscar win?

Project questions:

- Why is it when I ask for the rank, I only see 6 instead of the full 100? Is there a limit?

## Key Findings

- SQL = standard query language

- Relational database example with books, authors, genres (i.e. Harry Potter and Stephen King)

- SQLite works well in applications not needing human expert support: TVs, game consoles, internet of things

- SelectorGadget: Chrome extension to inspect CSS elements on a webpage; helpful for scraping task

## Gameplan

Here is my overall approach:

1. Determine how to get SQL/SQLite on my computer.

2. Determine if I'm working with a CSV database or need to parse from a website.

3. Scrap parsing; not the goal of the sprint this week.

4. Focus work on constructing models/tables in order to perform functions and analysis.

5. Focus on data from the week before Oscar win and the week after Oscar win (from BoxOfficeMojo.com - copied and pasted as CSV in Excel)

6. Map out data model(s) for project.

7. Determine data types for each attribute in data model.

8. Create primary and foreign keys based off of movie\_ID; ID is primary key for the Movies table, but serves as the foreign key for the two box office return tables.

9. Figure out how to create a join. Use "inner join" function. An "inner join" is used to pull matching data from tables.

10. Filter off of column values.

11. Bonus step: Determine the percent changes for those films where at least one Oscar was one. This seems to involve inserting a column with a calculation.

---

## Project Ideas

1. SQLite database from a spreadsheet: load an existing database into SQL (one line of SQL code that is hard)

2. Querying from a CSV using command line.

3. Experimenting with joins

4. Connecting to a SQL database from R

## Pitches

1. Create SQLite database to demonstrate the three tables: award show data, movie data, financial data.

2. Connect to SQLite database (as built in step 1) using R.

3. Join award show and movie databases using SQL.

4. Scrape data from oscars.com to build SQLite database.

5. Completely unrelated project, but mine text data from Facebook using API

## Data Model Structure

I decided to construct three tables to perform the SQL functions I wanted to test. The structure of these data models is as follows:

Movies

- Movie ID (primary key)

- Movie name (text)

- Budget (numeric)

- Studio (text)

- Won an Oscar? (boolean)

- Oscar category (text)

Pre-Oscar Box Office Data

- Movie ID (foreign key)

- Theater count (integer)

- Gross (numeric)

- Per screen average (numeric)

Post-Oscar Box Office Data

- Movie ID (foreign key)

- Theater count (integer)

- Gross (numeric)

- Per screen average (numeric)

## Key SQL Commands Used

1. SELECT - Choose, pick

2. INNER JOIN - Join function to combine two tables off of a key

3. WHERE - conditional value; used in this case to filter off of a Boolean value

4. FROM - selects which table the command is working off of

5. Syntax note - Pre\_Oscars.Weekend\_Gross when used with SELECT translates to: Select the Weekend\_Gross column from the Pre.Oscars table

## Here are some overall notes on the skills I learned.

1. Learned some basic SQL commands and syntax: select, where, from, inner join; for the column selection, the format is table.column

2. Learned how to think about structuring a data model.

3. Learned how to use SQLite Studio; the GUI interface is helpful to visualize changes.

4. Learned about the importance of data types (i.e. integer vs. text; string vs. text).

5. Punctuation or spaces will change how the column headers are formatted in SQL queries. At one point, I had an extra space after Theater\_Count, and so the column header pulled in as: "Theater\_Count " (i.e. with the punctuation marks). Similarly, "Oscar\_Winner?" pulled in with the punctuation marks and question mark.

6. SQL Studio will indicate when column headers are correct by turning blue and/or giving you a dropdown of the available column headers. This came in handy to validate my syntax was correct because when I was mistyping the information, it didn't show up as a dropdown value.

## Resources/Links

1. Web scraping from IMDB: https://www.analyticsvidhya.com/blog/2017/03/beginners-guide-on-web-scraping-in-r-using-rvest-with-hands-on-knowledge/

2. How to install packages in R: http://jtleek.com/modules/01\_DataScientistToolbox/02\_09\_installingRPackages/#1

3. Basics of R: https://www.analyticsvidhya.com/learning-paths-data-science-business-analytics-business-intelligence-big-data/learning-path-r-data-science/

4. Exporting from R to a CSV: http://www.instantr.com/2012/12/11/exporting-a-dataset-from-r/

5. SQLite training: https://www.quackit.com/sqlite/tutorial/

6. More SQLite training: http://www.sqlitetutorial.net/

7. SQL Studio training: https://wiki.sqlitestudio.pl/index.php/

8. And even more SQLite training: https://www.tutorialspoint.com/sqlite/

9. Book recommendation from Nat: https://www.barnesandnoble.com/p/sams-teach-yourself-sql-in-10-minutes-ben-forta/1100070678/2677684175041?st=PLA&sid=BNB\_DRS\_New+Marketplace+Shopping+Textbooks\_00000000&2sid=Google\_&sourceId=PLGoP164984&gclid=EAIaIQobChMIoNiLnIn31wIVkmV-Ch3PVgf4EAQYASABEgI5e\_D\_BwE

10. Inner join vs. outer join: https://www.diffen.com/difference/Inner\_Join\_vs\_Outer\_Join

## Notes for Future Projects

1. Using web scraping to gather data from Amazon customer reviews

# SQL query

# SELECT Pre\_Oscars.Weekend\_Gross, Pre\_Oscars.Theater\_Count, Pre\_Oscars.Average, Post\_Oscars.Weekend\_Gross, Post\_Oscars."Theater\_Count ", Post\_Oscars.Average, Movies.Name, Movies."Oscar Winner?"

# FROM Pre\_Oscars

# INNER JOIN Post\_Oscars on Post\_Oscars.Movie\_ID = Pre\_Oscars.Movie\_ID

# INNER JOIN Movies on Movies.ID = Pre\_Oscars.Movie\_ID

# WHERE Movies."Oscar Winner?" = 1