

# SOC - Security Analyst Interview Exercise

## Documentation of results

By Mariana de San Juan

### Understanding the assignment

From two files of datasets in CSV format is necessary to answer the following questions by SQL Queries.

- According to the provided dataset, how many 2018 films were categorized as a Comedy?
- According to the provided dataset, how many 2018 films got a score of 8.0 or higher? (Note that this will require joining the two datasets together)
- What was the best film of 2018?
- Do audiences prefer longer films, or shorter films? You may choose to simply outline your methodology to approach this problem.

### The plan

For this assignment, we are going to use HeidiSQL as the database management system (DBMS) to analyze datasets and obtain results through SQL queries.

### Hands-on step by step

The database model was created under the name 'cinematrends' to proceed with the creation of tables with the following column and data type:

title_basics_2018.csv		
Field	datatype	description
tconst	string	alphanumeric unique identifier of the title
primaryTitle	string	the more popular title / the title used by the filmmakers on promotional materials at the point of release
originalTitle	string	original title, in the original language
year	YYYY	represents the release year of a title
runtimeMinutes	int	primary runtime of the title, in minutes
genres	string array	includes up to three genres associated with the title

title_ratings.csv		
Field	datatype	description
tconst	string	alphanumeric unique identifier of the title
averageRating	float	weighted average of all the individual user ratings
numVotes	int	number of votes the title has received

The next step is importing the datasets from the title\_ratings.csv and title\_basics\_2018.csv files in order to populate the database.

cinematrends\cinematrends\title\_ratings\ - HeidiSQL 12.10.0.7000

Database filter: cinematrends (240.0 KiB)  
 Table filter: cinematrends (32.0 KiB)  
 title\_basics\_2018 (16.0 KiB)  
 title\_ratings (16.0 KiB)  
 information\_schema (208.0 KiB)

cinematrends.title\_ratings: Next Show all Sorting Columns (3/3) Filter

#	tconst	averageRating	numVotes
1	tt0000001	5.6	1,543
2	tt0000002	6.1	186
3	tt0000003	6.5	1,201
4	tt0000004	6.2	114
5	tt0000005	6.1	1,921
6	tt0000006	5.2	102
7	tt0000007	5.4	613
8	tt0000008	5.4	1,651
9	tt0000009	5.4	81
10	tt0000010	6.9	5,515
11	tt0000011	5.2	235
12	tt0000012	7.4	9,381
13	tt0000013	5.7	1,437
14	tt0000014	7.1	4,095
15	tt0000015	6.1	737
16	tt0000016	5.9	1,075
17	tt0000017	4.7	213

```

235 /* 12,815 rows imported in 0.188 seconds. */
236 SELECT * FROM `cinematrends`.`title_basics_2018` LIMIT 1000;
237 SHOW CREATE TABLE `cinematrends`.`title_ratings`;
238 SELECT * FROM `cinematrends`.`title_ratings` LIMIT 1000;
239 SHOW TABLE STATUS LIKE 'title_ratings';

```

Screenshot 1. Evidence of dataset loaded into the title\_ratings table.

cinematrends\cinematrends\title\_basics\_2018\ - HeidiSQL 12.10.0.7000

Database filter: cinematrends (240.0 KiB)  
 Table filter: cinematrends (32.0 KiB)  
 title\_basics\_2018 (16.0 KiB)  
 title\_ratings (16.0 KiB)  
 information\_schema (208.0 KiB)

cinematrends.title\_basics: Next Show all Sorting Columns (6/6) Filter

#	tconst	primaryTitle	originalTitle
1	tt0069049	The Other Side of the Wind	The Other Side of the Wind
2	tt0111414	A Thin Life	A Thin Life
3	tt0170651	T.G.M. - osvoboditel	T.G.M. - osvoboditel
4	tt0192528	Heaven & Hell	Reverse Heaven
5	tt0253093	Gangavataran	Gangavataran
6	tt0262759	Seven Jews from My Class	Siedmiu Żydów z mojej klasy
7	tt0276568	To Chase a Million	To Chase a Million
8	tt0328810	My Land	Aamaar Bhuvan
9	tt0337981	A Documentary on the Making of 'Gore Vidal's Caligula'	A Documentary on the Making of 'Gor
10	tt0347307	The Stone Cross	Kaminnyy khrest
11	tt0360556	Fahrenheit 451	Fahrenheit 451
12	tt0365545	Nappily Ever After	Nappily Ever After
13	tt0427543	A Million Little Pieces	A Million Little Pieces
14	tt0432010	The Queen of Sheba Meets the Atom Man	The Queen of Sheba Meets the Atom
15	tt0825334	Caravaggio and My Mother the Pope	Caravaggio and My Mother the Pope
16	tt0850247	Amsteroid	Amsteroid

```

232 /* Converting filename to 8.3 format: C:\Users\desjm\Downloads\title_basics_2018.csv => C:\Users\desjm\DOWNLO~1\TITLE_~1.CS
233 LOAD DATA LOW_PRIORITY LOCAL INFILE 'C:\\Users\\desjm\\DOWNLO~1\\TITLE_~1.CSV' REPLACE INTO TABLE `cinematrends`.`title_bas
234 /* Info: Records: 12815 Deleted: 0 Skipped: 0 Warnings: 0 */
235 /* 12,815 rows imported in 0.188 seconds. */
236 SELECT * FROM `cinematrends`.`title_basics_2018` LIMIT 1000;

```

Screenshot 2. Evidence of dataset loaded into the title\_basics\_2018 table.

## Results

The following step is opening a Query tab and writing the transactions to retrieve the necessary information to answer the questions of this assignment. Now that the datasets have been successfully loaded into our database, we will proceed with crafting SQL queries to address each of our research questions.

According to the provided dataset, how many 2018 films were categorized as a Comedy?

The screenshot shows the HeidiSQL interface with a query window open. The query is as follows:

```
1 USE cinematrends;
2 SELECT COUNT(*) AS total_comedias
3 FROM title_basics_2018
4 WHERE year = 2018 AND genres = 'comedy';
5
```

The results pane shows a single row with the value 800 for the column 'total\_comedias'.

The status bar at the bottom indicates: DE German (Germany), r15: c1 (120 B), Connected: 00:06 h, MariaDB 11.8.2, Uptime: 01:46 h, Server time: 4:27 PM, Idle.

2. According to the results of the Query 800 films from 2018 were categorized as a Comedy.

According to the provided dataset, how many 2018 films got a score of 8.0 or higher? (Note that this will require joining the two datasets together)

The screenshot shows the HeidiSQL interface with a query window open. The query is as follows:

```
1 USE cinematrends;
2 SELECT COUNT(*) AS total_comedias
3 FROM title_basics_2018
4 WHERE year = 2018 AND genres = 'comedy';
5 SELECT b.primaryTitle AS best_film_2018
6 FROM title_basics_2018 b
7 JOIN title_ratings r ON b.tconst = r.tconst
8 WHERE b.year = 2018
9 ORDER BY r.averageRating DESC, r.numVotes DESC
10 LIMIT 1;
11 SELECT COUNT(*) AS total_películas_8_o_mas
12 FROM title_basics_2018 b
13 JOIN title_ratings r ON b.tconst = r.tconst
14 WHERE b.year = 2018 AND r.averageRating >= 8.0;
15
```

The results pane shows a single row with the value 780 for the column 'total\_películas\_8\_o\_mas'.

The status bar at the bottom indicates: EN English (United States), r15: c1 (475 B), Connected: 00:39 h, MariaDB 11.8.2, Uptime: 02:39 h, Server time: 5:20 PM, Idle.

3. According to the datasets there are 780 films from 2018 with a score of 8.0 or higher.

## What was the best film of 2018?

The screenshot shows the HeidiSQL interface with a query executed against the 'cinematrends' database. The query selects the top film of 2018 based on average rating and number of votes. The result shows the film 'Exteriores: Mulheres Brasileiras na Diplomacia'.

```

1  USE cinematrends;
2  SELECT COUNT(*) AS total_comedias
3  FROM title_basics_2018
4  WHERE year = 2018 AND genres = 'comedy';
5  SELECT b.primaryTitle AS best_film_2018
6  FROM title_basics_2018 b
7  JOIN title_ratings r ON b.tconst = r.tconst
8  WHERE b.year = 2018
9  ORDER BY r.averageRating DESC, r.numVotes DESC
10 LIMIT 1;

```

The result set shows:

#	best_film_2018
1	Exteriores: Mulheres Brasileiras na Diplomacia

4. IMDb crowns the top film by balancing the highest ratings with the greatest number of votes ensuring quality meets popularity. Under this method, the best film of 2018 is: Exteriores: Mulheres Brasileiras na Diplomacia

*Do audiences prefer longer films, or shorter films? You may choose to simply outline your methodology to approach this problem.*

I wanted to see if people prefer longer or shorter films, so I compared the duration of each movie with the number of audience votes. More votes usually mean more interest. By grouping movies by length and checking the average votes, I got a better idea of what kind of runtime tends to attract the most attention.

The screenshot shows the HeidiSQL interface with a query executed against the 'cinematrends' database. The query groups films by length (Short, Medium, Long) and calculates the average number of votes. The result shows that longer films tend to have higher average votes.

```

1  USE cinematrends;
2  SELECT
3  CASE
4  WHEN b.runtimeMinutes < 90 THEN 'Short (<90 min)'
5  WHEN b.runtimeMinutes BETWEEN 90 AND 120 THEN 'Medium (90-120 min)'
6  WHEN b.runtimeMinutes > 120 THEN 'Long (>120 min)'
7  ELSE 'Unknown'
8  END AS film_length,
9  COUNT(*) AS film_count,
10 AVG(r.numVotes) AS avg_votes
11 FROM title_basics_2018 b
12 JOIN title_ratings r ON b.tconst = r.tconst
13 WHERE b.runtimeMinutes IS NOT NULL
14 GROUP BY film_length
15 ORDER BY avg_votes DESC;

```

The result set shows:

#	film_length	film_count	avg_votes
1	Long (>120 min)	876	8,726.6632
2	Medium (90-120 min)	3,622	2,821.1469
3	Short (<90 min)	2,731	312.1571

5. My analysis suggests that audiences tend to prefer longer films, as these titles show higher engagement through a greater number of votes.

## My SQL Script

```
SELECT COUNT(*) AS total_películas_8_o_mas
FROM title_basics_2018 b
JOIN title_ratings r ON b.tconst = r.tconst
WHERE b.year = 2018 AND r.averageRating >= 8.0;
SELECT
  CASE
    WHEN b.runtimeMinutes < 90 THEN 'Short (<90 min)'
    WHEN b.runtimeMinutes BETWEEN 90 AND 120 THEN 'Medium (90-120 min)'
    WHEN b.runtimeMinutes > 120 THEN 'Long (>120 min)'
    ELSE 'Unknown'
  END AS film_length,
  COUNT(*) AS film_count,
  AVG(r.numVotes) AS avg_votes
FROM title_basics_2018 b
JOIN title_ratings r ON b.tconst = r.tconst
WHERE b.runtimeMinutes IS NOT NULL
GROUP BY film_length
ORDER BY avg_votes DESC;
```

## Conclusion

SQL is a versatile language used to analyze and extract insights from large datasets, tailored to the specific needs of data analysis. With just a few queries, we can uncover meaningful answers. However, the ability of engineers and analysts to solve problems effectively plays a crucial role in minimizing bias and ensuring reliable conclusions. In fact my SQL script has a mix of english and spanish but this is a test, in the workplace I can stick to a single language english or spanish.