Movie Database Writeup

**Database Relations:**

Movies: Table which holds all the information about each movie that doesn’t have multiple values. This includes data like title and budget. It does not hold information on attributes that have multiple possible values such as movie producers and genres. The primary key for this relation is the movie id which is unique to each movie.

Genres: Table which contains all of the different types of genres that the movies in this database categorize as. The primary key for this table is the genre id.

Keywords: Table which contains all of the different types of keywords that the movies in this database are described as. The primary key for this table is the keyword id.

Production\_company: Table which holds all of the production companies that produced the movies in the database. The primary key for this relation is the production companies’ id.

Production\_country: Table which holds all of the countries that the movies in the database where produced in. The primary key for this relation is the countries’ iso\_3166\_1 id.

Spoken\_language: A relation that stores all of the languages that are spoken in the movies in the database. The primary key for this relation is the languages’ iso\_369\_1 id.

Movie\_has\_Genre: This is a join table that will allow the genre table to connect with the movie table. The primary key for this relation is a relationship id. It also contains two foreign keys, movie id and genre id, which will be used to connect the two tables.

Movie\_has\_ Keywords: This is a join table that will allow the keyword table to connect with the movie table. The primary key for this relation is a relationship id. It also contains two foreign keys, movie id and keyword id, which will be used to connect the two tables.

Movie\_has\_ Production\_company: This is a join table that will allow the production company table to connect with the movie table. The primary key for this relation is a relationship id. It also contains two foreign keys, movie id and production companies’ id, which will be used to connect the two tables.

Movie\_has\_ Production\_country: This is a join table that will allow the production country table to connect with the movie table. The primary key for this relation is a relationship id. It also contains two foreign keys, movie id and the countries’ iso\_3166\_1 id, which will be used to connect the two tables.

Movie\_has\_ Spoken\_language: This is a join table that will allow the spoken languages table to connect with the movie table. The primary key for this relation is a relationship id. It also contains two foreign keys, movie id and the languages’ iso\_369\_1 id, which will be used to connect the two tables.

**Normal Form of Database:**

I implemented the database to fall under second normal form. I did this by separating out all attributes that had multiple values into their own table and then creating join tables for all of the many to many relationships. It is not in third normal form or higher because data is duplicated in multiple tables. The only reason why I am able to store the two foreign keys in the join tables is because of the relationship id I created for each tuple.

**Queries:**

1. 10% What is the average budget of all movies? Your output should include just the average budget value.

Query:

SELECT AVG(budget)

FROM MovieDBAssign5.Movie

Output:

--- Average Budget ---

29045039.8753

2. 10% Show only the movies that were produced in the United States. Your output must include the movie title and the production company name.

Query:

SELECT title, prodComp\_name

FROM MovieDBAssign5.Movie

JOIN MovieDBAssign5.Movie\_has\_Production\_company

ON MovieDBAssign5.Movie.id = MovieDBAssign5.Movie\_has\_Production\_company.id

JOIN MovieDBAssign5.Production\_company

ON MovieDBAssign5.Movie\_has\_Production\_company.prodComp\_id =\ MovieDBAssign5.Production\_company.prodComp\_id

JOIN MovieDBAssign5.Movie\_has\_Production\_country

ON MovieDBAssign5.Movie.id = MovieDBAssign5.Movie\_has\_Production\_country.id

JOIN MovieDBAssign5.Production\_country

ON MovieDBAssign5.Movie\_has\_Production\_country.iso3166\_id =\ MovieDBAssign5.Production\_country.iso3166\_id

WHERE MovieDBAssign5.Production\_country.iso3166\_id = "US"

LIMIT 5

Output:

--- 5 Movies Made in the US ---

('Avatar', 'Ingenious Film Partners')

('Avatar', 'Twentieth Century Fox Film Corporation')

('Avatar', 'Dune Entertainment')

('Avatar', 'Lightstorm Entertainment')

("Pirates of the Caribbean: At World's End", 'Walt Disney Pictures')

3. 10% Show the top 5 movies that made the most revenue. Your output must include the movie title and how much revenue it brought in.

Query:

SELECT title, revenue

FROM MovieDBAssign5.Movie

ORDER BY revenue

DESC

LIMIT 5

Output:

--- Top 5 Highest Revenue ---

('Avatar', 2787965087)

('Titanic', 1845034188)

('The Avengers', 1519557910)

('Jurassic World', 1513528810)

('Furious 7', 1506249360)

4. 10% What movies have both the genre Science Fiction and Mystery. Your output must include the movie title and all genres associated with that genre.

Query:

SELECT title, MovieDBAssign5.Genre.gen\_name

FROM MovieDBAssign5.Movie

JOIN MovieDBAssign5.Movie\_has\_Genre

ON MovieDBAssign5.Movie.id = MovieDBAssign5.Movie\_has\_Genre.id

JOIN MovieDBAssign5.Genre

ON MovieDBAssign5.Movie\_has\_Genre.genre\_id = MovieDBAssign5.Genre.genre\_id

WHERE MovieDBAssign5.Movie.id = any(

SELECT MovieDBAssign5.Movie.id

FROM MovieDBAssign5.Movie

JOIN MovieDBAssign5.Movie\_has\_Genre

ON MovieDBAssign5.Movie.id = MovieDBAssign5.Movie\_has\_Genre.id

JOIN MovieDBAssign5.Genre

ON MovieDBAssign5.Movie\_has\_Genre.genre\_id = MovieDBAssign5.Genre.genre\_id

WHERE MovieDBAssign5.Genre.gen\_name = "Science Fiction")

and MovieDBAssign5.Movie.id = any(

SELECT MovieDBAssign5.Movie.id

FROM MovieDBAssign5.Movie

JOIN MovieDBAssign5.Movie\_has\_Genre

ON MovieDBAssign5.Movie.id = MovieDBAssign5.Movie\_has\_Genre.id

JOIN MovieDBAssign5.Genre

ON MovieDBAssign5.Movie\_has\_Genre.genre\_id = MovieDBAssign5.Genre.genre\_id

WHERE MovieDBAssign5.Genre.gen\_name = "Mystery")

LIMIT 5

Output:

--- 5 Movies in SciFi and Mystery Genre ---

('Tomorrowland', 'Adventure')

('Tomorrowland', 'Family')

('Tomorrowland', 'Mystery')

('Tomorrowland', 'Science Fiction')

('Inception', 'Action')

5. 10% Find the movies that have a popularity greater than the average popularity. Your output must include the movie title and their popularity.

Query:

SELECT title, popularity

FROM MovieDBAssign5.Movie

WHERE popularity > (

SELECT avg(popularity)

FROM MovieDBAssign5.Movie)

LIMIT 5

Output:

--- 5 Movies With Greater Popularity Than Average ---

('Four Rooms', 22.8762)

('Star Wars', 126.394)

('Finding Nemo', 85.6888)

('Forrest Gump', 138.133)