PostgreSQL Movie Queries for Analysis

Jonathan Glasgow

Dataset Documentation:

Movies

Attribute	Description
id	Unique movie ID
imdb_id	ID for this movie on IMDB
title	Title
overview	Description
runtime	Run time, in minutes
release_date	Release date
budget	Budget
revenue	Revenue
popularity	Popularity (provenance and units unknown)
vote_average	Average user score (independent of ratings table)
vote_count	Number of users who rated movie (independent of ratings table)

Genres

Attribute	Description
id	Movie ID
genre	Genre

Ratings

Attribute	Description
userid	User ID
movieid	Movie ID (references Movies table)
rating	1–5 star rating, including half-star ratings (0.5 to 5.0)

Data for this project were assembled by Rounak Banik from The Movie Database (https://www.themoviedb.org) and MovieLens (https://movielens.org). The data includes 32,207 distinct movies released between 1878 and 2020, their genres, and nearly 8.4 million movie ratings made by more than 250,000 users.

Data for these tables were downloaded from https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset?resource=download.

Setting up the dataset:

```
CREATE TABLE Movies(id INTEGER PRIMARY KEY,
        imdb_id CHAR(9),
        title VARCHAR(150),
        overview TEXT,
        runtime INTEGER,
        release_date DATE,
        popularity REAL,
        budget BIGINT,
        revenue REAL,
        vote_average NUMERIC(3,1),
        vote_count INTEGER);
CREATE TABLE Genres(id INTEGER REFERENCES Movies,
        genre VARCHAR(20),
        CONSTRAINT genre_pk PRIMARY KEY(id, genre));

CREATE TABLE Ratings(userid INTEGER,
        movieid INTEGER REFERENCES Movies,
        rating NUMERIC(2,1),
        CONSTRAINT rating_pk PRIMARY KEY (userid, movieid));


\copy Movies FROM 'movies.csv' WITH DELIMITER ',' CSV HEADER;
\copy Genres FROM 'genres.csv' WITH DELIMITER ',' CSV HEADER;
\copy Ratings FROM 'ratings.csv' WITH DELIMITER ',' CSV HEADER;
```

Queries:

Below are 10 queries that answer important questions about the IMDb movie dataset. They call a local PostgreSQL server

1. Which movies are missing revenue information? Return all attributes.

```
SELECT * FROM Movies WHERE revenue IS NULL;
```

2. What are the movies with the 10 biggest budgets? Return the movie IDs, titles, and budgets.

SELECT id, title, budget FROM Movies WHERE budget IS NOT NULL ORDER BY budget DESC LIMIT 10;

3. What were the ranges for the popularity and vote average columns?

SELECT MIN(popularity) AS popularity_min, MAX(popularity) AS popularity_max, MIN(vote_average) AS vote_average_min, MAX(vote_average) AS vote_average_max FROM Movies;

4. What are the titles and overviews for all movies whose overview references Einstein?

SELECT title, overview FROM Movies WHERE overview LIKE '%Einstein%';

5. What was the average user rating across all movies in the database by month?

//Average based on vote average
SELECT DATE_PART('month', release_date) AS month, AVG(vote_average) FROM Movies WHERE DATE_PART('month', release_date) BETWEEN 1 AND 12 GROUP BY month ORDER BY month ASC;

//Average for each month based on Rating
SELECT DATE_PART('month', release_date) AS month, AVG(rating) FROM Movies JOIN Ratings ON Movies.id = Ratings.movieid WHERE rating IS NOT NULL AND DATE_PART('month', release_date) BETWEEN 1 AND 12 GROUP BY month ORDER BY month ASC;

6. What is the most recent movie in each genre?

SELECT * FROM (SELECT DISTINCT ON (genre) genre, title, MAX(release_date) AS release_date FROM Movies NATURAL JOIN Genres WHERE release_date IS NOT NULL GROUP BY genre, title ORDER BY genre, release_date DESC) new_table ORDER BY release_date DESC;

7. What are the genres, title, and overview for the movies that have 8 different genres?

SELECT title, genre, overview FROM Movies NATURAL RIGHT JOIN (SELECT id, COUNT(genre) FROM Genres GROUP BY id HAVING COUNT(genre) = 8) AS new_table NATURAL JOIN Genres;

8. What is the average revenue generated by movies in each genre? Don't include movies whose revenue is unknown, and movies that belong to multiple genres should contribute towards all genres.

SELECT * FROM (SELECT DISTINCT ON (genre) genre, AVG(revenue) AS average_revenue FROM Movies NATURAL JOIN Genres WHERE revenue IS NOT NULL GROUP BY genre) new_table ORDER BY average_revenue DESC;

9. How did the average revenue for movies in each genre change each year since 2015?

SELECT genre, DATE_PART('year', release_date) AS year, AVG(revenue) AS average_revenue FROM Movies NATURAL JOIN Genres WHERE revenue IS NOT NULL AND DATE_PART('year', release_date) BETWEEN 2015 AND 2020 GROUP BY genre, year ORDER BY genre, year ASC;

10. What is the average user rating generated by movies in each genre? Don't include movies that have not received any ratings, and movies that belong to multiple genres should contribute towards all genres.

```
SELECT * FROM (SELECT DISTINCT ON (genre) genre, AVG(rating) AS
average_rating FROM Movies NATURAL JOIN Genres JOIN Ratings ON Movies.id =
Ratings.movieid WHERE rating IS NOT NULL GROUP BY genre) new_table ORDER
BY average_rating DESC;
```