|  |
| --- |
| SQL |
| **Advanced SQL-IMDB Dataset** |
| IMDB |

|  |
| --- |
| Shahin Sulaiman M  1-21-2025 |

**Introduction to the IMDB Dataset and Its Structure**

The IMDB Dataset is a comprehensive collection of data related to movies, including details about their titles, release years, genres, directors, actors, ratings, and more. This dataset is widely used for educational, research, and analytical purposes, particularly in the fields of data analysis, machine learning, and database management. Below is an overview of the dataset and its structure. The IMDb dataset contains extensive details on movies, actors, directors, and ratings. It helps analyze film industry trends, audience preferences, and production insights. The dataset is structured with multiple linked tables covering various aspects such as genres, ratings, production companies, and cast members. This document utilizes advanced SQL queries to uncover valuable insights, including identifying top-rated movies, studying voting trends, and determining key factors affecting movie success.

**Dataset Overview**

The dataset is divided into six main tables, each containing specific information about movies and the people associated with them. These tables are:

Movie Table Contains basic information about each movie, such as:

**Title**: The name of the movie.

**Release Year**: The year the movie was released.

**Duration**: The runtime of the movie in minutes.

**Country**: The country where the movie was produced.

**Income**: The worldwide gross income generated by the movie.

**Languages**: The languages in which the movie is available.

**Production Company**: The company responsible for producing the movie.

Genre Table Describes the genres associated with each movie.

Each movie can belong to one or more genres, such as Action, Comedy, Drama, etc.

**Director Mapping Table**

* Maps movies to their directors.
* This table links the movie\_id to the name\_id of the director.

**Role Mapping Table**

* Maps actors/actresses to movies and specifies their roles (e.g., actor, director, producer).
* This table links the movie\_id to the name\_id of the person and specifies their role category.

Names Table

Stores information about people associated with movies, such as actors, directors, and producers.

* **Name**: The full name of the person.
* **Height**: The height of the person (if available).
* **Date of Birth**: The birthdate of the person.
* **Known For Movies**: A list of movies the person is best known for.

Ratings Table

* Average Rating: The average rating given to the movie by viewers.
* Total Votes: The total number of votes received by the movie.
* Median Rating: The median rating of the movie.

**Dataset Structure**

The dataset is structured in a relational database format, where each table is connected through primary and foreign keys. For example:

The movie\_id in the movie table is used as a foreign key in the genre, director\_mapping, role\_mapping, and ratings tables.The name\_id in the names table is used as a foreign key in the director\_mapping and role\_mapping tables.

This structure allows for efficient querying and analysis of the data, enabling users to extract meaningful insights about movies, genres, actors, directors, and ratings.

**1. Count the total number of records in each table of the database.**

SELECT 'movie' AS table\_name, COUNT(\*) AS total\_records FROM movie

UNION ALL

SELECT 'genre', COUNT(\*) FROM genre

UNION ALL

SELECT 'director\_mapping', COUNT(\*) FROM director\_mapping

UNION ALL

SELECT 'role\_mapping', COUNT(\*) FROM role\_mapping

UNION ALL

SELECT 'ratings', COUNT(\*) FROM ratings;

**2. Identify which columns in the movie table contain null values**.

SELECT

'title' AS column\_name, COUNT(\*) AS null\_count FROM movie WHERE title IS NULL

UNION ALL

SELECT

'year', COUNT(\*) FROM movie WHERE year IS NULL

UNION ALL

SELECT

'date\_published', COUNT(\*) FROM movie WHERE date\_published IS NULL

UNION ALL

SELECT

'duration', COUNT(\*) FROM movie WHERE duration IS NULL

UNION ALL

SELECT

'country', COUNT(\*) FROM movie WHERE country IS NULL

UNION ALL

SELECT

'worlwide\_gross\_income', COUNT(\*) FROM movie WHERE worlwide\_gross\_income IS NULL

UNION ALL

SELECT

'languages', COUNT(\*) FROM movie WHERE languages IS NULL

UNION ALL

SELECT

'production\_company', COUNT(\*) FROM movie WHERE production\_company IS NULL;

**3. Determine the total number of movies released each year, and analyze how the trend changes monthwise**.

SELECT

YEAR(date\_published) AS release\_year,

MONTH(date\_published) AS release\_month,

COUNT(\*) AS total\_movies

FROM

movie

GROUP BY

YEAR(date\_published),

MONTH(date\_published)

ORDER BY

release\_year,

release\_month;

**4. How many movies were produced in either the USA or India in the year 2019?**

SELECT DISTINCT genre

FROM genre;

SELECT

COUNT(\*) AS movies\_with\_one\_genre

FROM (

SELECT

movie\_id

FROM

genre

GROUP BY

movie\_id

HAVING COUNT(genre) = 1

) AS single\_genre\_movies;

**5. List the unique genres in the dataset, and count how many movies belong exclusively to onegenre**

SELECT country,COUNT(\*) AS movie\_produced

FROM movie

WHERE year= 2019 and (country ='USA' OR country ='India')

GROUP BY country;

**6. Which genre has the highest total number of movies produced?**

SELECT

genre,

COUNT(\*) AS total\_movies

FROM

genre

GROUP BY

genre

ORDER BY

total\_movies DESC

LIMIT 1;

**7. Calculate the average movie duration for each genre**.

SELECT

g.genre,

AVG(m.duration) AS avg\_duration

FROM

genre g

JOIN

movie m

ON

g.movie\_id = m.id

GROUP BY

g.genre

ORDER BY

avg\_duration DESC;

**8. Identify actors or actresses who have appeared in more than three movies with an average rating below 5**.

SELECT n.name, COUNT(\*) AS low\_rated\_movies

FROM role\_mapping rm

JOIN ratings r ON rm.movie\_id = r.movie\_id

JOIN names n ON rm.name\_id = n.id

WHERE r.avg\_rating < 5

GROUP BY n.name

HAVING COUNT(\*) > 3

order by low\_rated\_movies desc;

**9. Find the minimum and maximum values for each column in the ratings table, excluding the movie\_id column.**

SELECT MIN(avg\_rating) AS min\_avg\_rating,

MAX(avg\_rating) AS max\_avg\_rating FROM ratings;

SELECT MIN(total\_votes) AS min\_total\_votes,

MAX(total\_votes) AS max\_total\_votes FROM ratings;

SELECT MIN(median\_rating) AS min\_median\_rating,

MAX(median\_rating) AS max\_median\_rating FROM ratings;

**10. Which are the top 10 movies based on their average rating?**

SELECT m.title, r.avg\_rating

FROM movie m

JOIN ratings r ON m.id = r.movie\_id

ORDER BY r.avg\_rating DESC LIMIT 10;

**11. Summarize the ratings table by grouping movies based on their median ratings**

select median\_rating, count(movie\_id) as movie\_counts

from ratings

group by median\_rating

order by median\_rating;

**12. How many movies, released in March 2017 in the USA within a specific genre, had more than 1,000 votes?**

select \* from movie;

select \* from ratings;

SELECT

g.genre, COUNT(\*) AS totalmovies

FROM

genre g

JOIN

movie m ON g.movie\_id = m.id

JOIN

ratings r ON m.id =r.movie\_id

WHERE

m.date\_published LIKE '2017-03-%'

AND m.country = 'USA'

AND r.total\_votes > 1000

GROUP BY g.genre

ORDER BY totalmovies DESC;

**13. Find movies from each genre that begin with the word “The” and have an average rating greater than 8.**

SELECT g.genre,m.title,r.avg\_rating

FROM genre g

JOIN movie m ON g.movie\_id = m.id

JOIN ratings r ON m.id = r.movie\_id

WHERE m.title LIKE 'The %'

AND r.avg\_rating > 8

ORDER BY g.genre, r.avg\_rating DESC;

**14. Of the movies released between April 1, 2018, and April 1, 2019, how many received a median rating of 8?**

SELECT COUNT(\*) AS movie\_count

FROM movie m

JOIN ratings r ON m.id = r.movie\_id

WHERE m.date\_published BETWEEN '2018-04-01' AND '2019-04-01'

AND r.median\_rating = 8;

**15. Do German movies receive more votes on average than Italian movies?**

SELECT country, AVG(r.total\_votes) AS average\_votes

FROM movie m

JOIN ratings r ON m.id = r.movie\_id

WHERE m.country IN ('Germany','Italy')

GROUP BY m.country;

**16. Identify the columns in the names table that contain null values.**

SELECT

column\_name

FROM

information\_schema.columns

WHERE

table\_name = 'names' AND

is\_nullable = 'YES';

**17. Who are the top two actors whose movies have a median rating of 8 or higher?**

SELECT n.name AS actor\_name, COUNT(r.movie\_id) AS movie\_count

FROM role\_mapping rm

JOIN names n ON rm.name\_id= n.id

JOIN ratings r ON rm.movie\_id = r.movie\_id

WHERE rm.category= 'actor'

AND r.median\_rating >= 8

GROUP BY n.name

ORDER BY movie\_count DESC

LIMIT 2;

**18. Which are the top three production companies based on the total number of votes their movies received?**

select m.production\_company as production\_company, r.total\_votes as Total\_votes

From movie m

Join ratings r on r.movie\_id = m.id

order by Total\_votes desc

limit 3;

**19. How many directors have worked on more than three movies?**

SELECT COUNT(\*) AS director\_count

FROM (

SELECT dm.name\_id

FROM director\_mapping dm

GROUP BY dm.name\_id

HAVING COUNT(dm.movie\_id) > 3

) AS directors;

**20. Calculate the average height of actors and actresses separately**

SELECT rm.category,

AVG(n.height) AS average\_height

FROM role\_mapping rm

JOIN names n ON rm.name\_id = n.id

GROUP BY rm.category;

**21. List the 10 oldest movies in the dataset along with their title, country, and director.**

SELECT m.title, m.country, n.name AS director, m.date\_published

FROM movie m

JOIN director\_mapping dm ON m.id = dm.movie\_id

JOIN names n ON dm.name\_id= n.id

ORDER BY m.date\_published ASC

LIMIT 10;

**22. List the top 5 movies with the highest total votes, along with their genres.**

select m.title, sum(r.total\_votes) as total\_votes, g.genre

from movie m

join ratings r on m.id = r.movie\_id

join genre g on m.id = g.movie\_id

group by m.title, g.genre

order by total\_votes desc

limit 5;

**23. Identify the movie with the longest duration, along with its genre and production company.**

SELECT movie.title, g.genre, movie.production\_company, movie.duration

FROM

movie

WHERE

movie.duration = (SELECT MAX(duration) FROM movie);

describe movie;

SELECT m.title, g.genre, m.production\_company,m.duration

FROM movie m

JOIN genre g ON m.id = g.movie\_id;

**24. Determine the total number of votes for each movie released in 2018.**

select m.title, sum(r.total\_votes) as totalvotes

from movie m

join ratings r

on m.id = movie\_id

where m.year = 2018

group by m.title

order by totalvotes desc;

**25. What is the most common language in which movies were produced?**

SELECT languages, COUNT(\*) AS movie\_count FROM movie

GROUP BY languages

ORDER BY movie\_count DESC LIMIT 1;

**Key Features and Application of the Dataset**

* Scalability: The dataset is designed to handle a large number of records, making it suitable for large-scale analysis.
* Flexibility: It supports multiple genres, roles, and languages, providing a comprehensive view of the movie industry.
* Normalization: The tables are normalized to reduce redundancy and improve data integrity.
* Educational Use: Teaching SQL, database management, and data analysis.
* Research: Analyzing trends in movie production, genres, and audience preferences.
* Development: Building applications that require movie-related data, such as recommendation systems or movie databases.