drop database imdb;

create database imdb;

use imdb;

select count(\*) from movies;

SET SQL\_SAFE\_UPDATES=0;

update movies set worlwide\_gross\_income=null where worlwide\_gross\_income='';

update movies set languages=null where languages='';

update movies set production\_company=null where production\_company='';

update movies set country=null where country='';

update movies set duration=null where duration='';

update movies set date\_published=null where date\_published='';

update movies set year=null where year='';

update movies set title=null where title='';

update movies set id=null where id='';

select count(\*) from genre;

SET SQL\_SAFE\_UPDATES=0;

update genre set movie\_id=null where movie\_id='';

update genre set genre=null where genre='';

select count(\*) from director\_mapping;

SET SQL\_SAFE\_UPDATES=0;

update director\_mapping set movie\_id=null where movie\_id='';

update director\_mapping set name\_id=null where name\_id='';

select count(\*) from role\_mapping;

SET SQL\_SAFE\_UPDATES=0;

update role\_mapping set movie\_id=null where movie\_id='';

update role\_mapping set name\_id=null where name\_id='';

update role\_mapping set category=null where category='';

select count(\*) from names;

SET SQL\_SAFE\_UPDATES=0;

update names set id=null where id='';

update names set name=null where name='';

update names set height=null where height='';

update names set date\_of\_birth=null where date\_of\_birth='';

update names set known\_for\_movies=null where known\_for\_movies='';

select count(\*) from ratings;

SET SQL\_SAFE\_UPDATES=0;

update ratings set movie\_id=null where movie\_id='';

update ratings set avg\_rating=null where avg\_rating='';

update ratings set total\_votes=null where total\_votes='';

update ratings set median\_rating=null where median\_rating='';

-- Segment 1: Database - Tables, Columns, Relationships

-- (1) What are the different tables in the database and how are they connected to each other in the database

ANS: There are 6 different tables with entity relationship diagram in the databasee which are interlinked with eachother and

they were named as movie, genre, dirctor mapping, role mapping, names and ratings

-- (2) Find the total number of rows in each table of the schema

select count(\*) from movies;

-- no of rows : 7997

select count(\*) from genre;

-- no of rows : 14662

select count(\*) from director\_mapping;

-- no of rows : 3867

select count(\*) from role\_mapping;

-- no of rows : 15615

select count(\*) from names;

-- no of rows : 25735

select count(\*) from ratings;

-- no of rows : 7997

-- (3) Identify which columns in the movie table have null values

select \* from movies;

SET SQL\_SAFE\_UPDATES=0;

update movies set worlwide\_gross\_income=null where worlwide\_gross\_income='';

update movies set languages=null where languages='';

update movies set production\_company=null where production\_company='';

update movies set country=null where country='';

update movies set duration=null where duration='';

update movies set date\_published=null where date\_published='';

update movies set year=null where year='';

update movies set title=null where title='';

update movies set id=null where id='';

select count(\*) from movies where country is null;

select

sum(case when id is null then 1 else 0 end) as id\_nulls,

sum(case when title is null then 1 else 0 end) as title\_nulls,

sum(case when year is null then 1 else 0 end) as year\_nulls,

sum(case when date\_published is null then 1 else 0 end) as date\_published\_nulls,

sum(case when duration is null then 1 else 0 end) as duration\_nulls,

sum(case when country is null then 1 else 0 end) as country\_nulls,

sum(case when worlwide\_gross\_income is null then 1 else 0 end) as worlwide\_gross\_income\_nulls,

sum(case when languages is null then 1 else 0 end) as languages\_nulls,

sum(case when production\_company is null then 1 else 0 end) as production\_company\_nulls from movies;

-- Segment 2: Movie Release Trends

-- (4) Determine the total number of movies released each year and analyse the month-wise trend

select year, substr(date\_published, 4,2) as month, count(id) as movies\_released from movies

group by year, substr(date\_published, 4,2) order by year, substr(date\_published, 4,2);

-- (5) Calculate the number of movies produced in the USA or India in the year 2019

select count(id) as movies\_released from movies

where (country = 'India' or country = 'USA') and year = 2019;

-- Segment 3: Production Statistics and Genre Analysis

-- (6) Retrieve the unique list of genres present in the dataset

select distinct genre from movies

left join genre on (movies.id = genre.movie\_id);

-- (7) Identify the genre with the highest number of movies produced overall

select genre, count(movie\_id) as movies from movies

left join genre on (movies.id = genre.movie\_id)

group by genre order by 2 desc;

-- (8) Determine the count of movies that belong to only one genre

select count(movie\_id) from

(

select movie\_id,

count(genre) as genre\_count

from genre

group by movie\_id

) A

where A.genre\_count = 1;

-- (9) Calculate the average duration of movies in each genre

select g.genre, round(avg(m.duration),2) as avg\_duration

from genre g

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

group by g.genre;

-- (10) Find the rank of the 'thriller' genre among all genres in terms of the number of movies produced

with cte as(

select genre, count(movie\_id) as movies from movies m

left join genre g on (m.id = g.movie\_id)

group by genre order by 2 desc

)

select \*, rank() over (order by movies desc) as genre\_rank from cte;

-- Segment 4: Ratings Analysis and Crew Members

-- (11) Retrieve the minimum and maximum values in each column of the ratings table (except movie\_id

SELECT MAX(avg\_rating) AS max\_avg\_rating,

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

FROM ratings;

SELECT MAX(total\_votes) AS max\_total\_votes,

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

FROM ratings;

select max(median\_rating) as max\_median\_rating,

min(median\_rating) as min\_median\_rating

from ratings;

-- (12) Identify the top 10 movies based on average rating

select movie\_id from ratings order by avg\_rating desc limit 10;

-- (13) Summarise the ratings table based on movie counts by median ratings

select \* from ratings;

SELECT median\_rating, COUNT(\*) AS movie\_counts

FROM ratings

GROUP BY median\_rating;

-- (14) Identify the production house that has produced the most number of hit movies (average rating > 8)

SELECT production\_company, COUNT(movies.id) AS movies

FROM movies

LEFT JOIN ratings ON (movies.id = ratings.movie\_id)

WHERE avg\_rating > 8

GROUP BY production\_company

ORDER BY movies DESC;

-- (15) Determine the number of movies released in each genre during March 2017 in the USA with more than 1,000 votes

SELECT genre, COUNT(id) AS movies\_released

FROM movies

LEFT JOIN genre ON (movies.id = genre.movie\_id)

LEFT JOIN ratings ON (movies.id = ratings.movie\_id)

WHERE total\_votes > 1000 and year = 2017

GROUP BY genre;

-- (16) Retrieve movies of each genre starting with the word 'The' and having an average rating > 8

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

FROM movies m

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

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

WHERE m.title LIKE 'The%'

AND r.avg\_rating > 8;

-- Segment 5: Crew Analysis

-- (17) Identify the columns in the names table that have null values

select \* from names;

update names set id=null where id='';

update names set name=null where name='';

update names set height=null where height='';

update names set date\_of\_birth=null where date\_of\_birth='';

update names set known\_for\_movies=null where known\_for\_movies='';

select

sum(case when id is null then 1 else 0 end) as id\_nulls,

sum(case when name is null then 1 else 0 end) as name\_nulls,

sum(case when height is null then 1 else 0 end) as height\_nulls,

sum(case when date\_of\_birth is null then 1 else 0 end) as date\_of\_birth\_nulls,

sum(case when known\_for\_movies is null then 1 else 0 end) as known\_for\_movies\_nulls

from names;

-- (18) Determine the top three directors in the top three genres with movies having an average rating > 8

WITH cte AS (

SELECT

genre,

name\_id AS director\_id,

COUNT(id) AS movies

FROM movies

LEFT JOIN genre ON (movies.id = genre.movie\_id)

LEFT JOIN director\_mapping ON (movies.id = director\_mapping.movie\_id)

GROUP BY name\_id, genre

ORDER BY genre, movies DESC)

,

cte2 as

(SELECT \* ,

ROW\_NUMBER() OVER (partition by genre ORDER BY movies DESC)ranking

FROM cte

WHERE director\_id IS NOT NULL

),

cte3 as

(SELECT genre, COUNT(id) AS movies

FROM movies

LEFT JOIN genre ON (movies.id = genre.movie\_id)

GROUP BY genre

order by movies desc limit 3)

SELECT director\_id, name , genre from cte2

left join names on (cte2.director\_id = names.id)

WHERE ranking <= 3

and genre in (select genre from cte3);

-- (19) Find the top two actors whose movies have a median rating >= 8

SELECT

rm.name\_id AS actor\_id,

nm.name AS actor\_name,

COUNT(DISTINCT m.id) AS movie\_count,

ROUND(

IF(COUNT(DISTINCT r.movie\_id) % 2 = 0,

(MAX(r.median\_rating) + MIN(r.median\_rating)) / 2,

MAX(r.median\_rating)

), 2

) AS median\_rating

FROM

role\_mapping rm

JOIN movies m ON rm.movie\_id = m.id

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

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

WHERE

rm.category = 'actor' AND

r.median\_rating >= 8

GROUP BY

actor\_id, actor\_name

ORDER BY

median\_rating DESC

LIMIT 2;

-- (20) Identify the top three production houses based on the number of votes received by their movies

SELECT

m.production\_company,

SUM(r.total\_votes) AS total\_votes\_received

FROM

movies m

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

GROUP BY

m.production\_company

HAVING

total\_votes\_received IS NOT NULL

ORDER BY

total\_votes\_received DESC

LIMIT 3;

-- (21) Rank actors based on their average ratings in Indian movies released in India

SELECT

rm.name\_id AS actor\_id,

nm.name AS actor\_name,

AVG(r.avg\_rating) AS average\_rating,

RANK() OVER (ORDER BY AVG(r.avg\_rating) DESC) AS actor\_rank

FROM

movies m

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

LEFT JOIN role\_mapping rm ON m.id = rm.movie\_id

LEFT JOIN names nm ON rm.name\_id = nm.id

WHERE

m.country = 'India' AND

rm.category = 'actor'

GROUP BY

actor\_id, actor\_name

ORDER BY

average\_rating DESC;

-- (22) Identify the top five actresses in Hindi movies released in India based on their average ratings

SELECT

rm.name\_id AS actress\_id,

nm.name AS actress\_name,

AVG(r.avg\_rating) AS average\_rating

FROM

movies m

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

LEFT JOIN role\_mapping rm ON m.id = rm.movie\_id

LEFT JOIN names nm ON rm.name\_id = nm.id

WHERE

m.country = 'India' AND

m.languages LIKE '%Hindi%' AND

rm.category = 'actress'

GROUP BY

actress\_id, actress\_name

ORDER BY

average\_rating DESC

LIMIT 5;

-- Segment 6: Broader Understanding of Data

-- (26) Classify thriller movies based on average ratings into different categories

select id , avg\_rating ,

case when avg\_rating > 5 then 'Hit Movie'

when avg\_rating < 5 then 'Flop Movie'

else 'Avg Movie' end as Movie\_category

from movies m left join genre g on (m.id = g.movie\_id)

left join ratings r on (m.id = r.movie\_id)

where genre = 'Thriller';

-- (24) analyse the genre-wise running total and moving average of the average movie duration

select id , genre , duration ,

sum(duration) over(partition by genre order by id asc) cum\_sum,

avg(duration) over(partition by genre order by id asc) moving\_average

from movies

left join genre on (movies.id = genre.movie\_id) order by genre , id;

-- (25) Identify the five highest-grossing movies of each year that belong to the top three genres

WITH cte AS (

SELECT

year, title, worlwide\_gross\_income, genre

FROM movies m

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

)

SELECT

year,

genre,

worlwide\_gross\_income

FROM cte

ORDER BY year, worlwide\_gross\_income DESC

LIMIT 5;

-- (26) Determine the top two production houses that have produced the highest number of hits among multilingual movies

WITH cte AS (

SELECT

m.id, m.production\_company, m.languages, r.avg\_rating

FROM movies m

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

)

SELECT production\_company

FROM cte

WHERE languages LIKE '%,%'

ORDER BY production\_company DESC

LIMIT 2;

-- (27) Identify the top three actresses based on the number of Super Hit movies (average rating > 8) in the drama genre

WITH cte AS (

SELECT nm.name, r.avg\_rating

FROM names nm

INNER JOIN ratings r ON nm.id = r.movie\_id

)

SELECT name

FROM cte

WHERE avg\_rating > 8;

-- (27) Retrieve details for the top nine directors based on the number of movies, including average inter-movie duration, ratings, and more

SELECT

director\_mapping.name\_id AS director\_id,

COUNT(movies.id) AS Num\_of\_movies,

AVG(movies.duration) AS avg\_movie\_duration,

AVG(ratings.avg\_rating) AS avg\_rating

FROM

movies

LEFT JOIN genre ON (movies.id = genre.movie\_id)

LEFT JOIN ratings ON (movies.id = ratings.movie\_id)

LEFT JOIN director\_mapping ON (movies.id = director\_mapping.movie\_id)

WHERE

director\_mapping.name\_id IS NOT NULL

GROUP BY

director\_mapping.name\_id

ORDER BY

Num\_of\_movies DESC;

-- Segment 7: Recommendations

-- (29) Based on the analysis, provide recommendations for the types of content Bolly movies should focus on producing

According to the analysis, Bolly movies are consider as most preferd content based on receive higher median ratings and revenue. Audiences appreciate movies that make them feel emotionally connected to the characters and the plot. High production quality can significantly enhance the overall viewing experience. Consider conducting test screenings to gather feedback from a diverse group of viewers. This can help you identify areas for improvement before the official release. While it's important to learn from successful films, avoid replicating them too closely. Audiences appreciate originality and fresh ideas. Strive to bring a unique perspective or storytelling style to your movies.