use imdb;

**SEGMENT 1: Database - Tables, Columns, Relationships**

Q1. What are the different tables in the database and how are they connected to each other in the database?

Ans. The database contains the following tables:

1. movie: This table stores information about movies. It has a primary key `id` and columns such as `title`, `year`, `date\_published`, `duration`, `country`, `worlwide\_gross\_income`, `languages`, and `production\_company`.

2. genre: This table represents the genres of movies. It has a composite primary key `(movie\_id, genre)` and contains the movie ID and genre for each movie.

3. director\_mapping: This table maps movies to directors. It has a composite primary key `(movie\_id, name\_id)` and contains the movie ID and director name for each movie.

4. role\_mapping: This table maps movies to actors/actresses and their roles. It has a composite primary key `(movie\_id, name\_id)` and contains the movie ID, actor/actress name, and role category for each movie.

5. names: This table stores information about people involved in movies, such as actors, actresses, and directors. It has a primary key `id` and columns like `name`, `height`, `date\_of\_birth`, and `known\_for\_movies`.

6. ratings: This table contains ratings information for movies. It has a primary key `movie\_id` and columns such as `avg\_rating`, `total\_votes`, and `median\_rating`.

These tables are connected to each other using foreign keys.

The `movie\_id` column in the genre, director\_mapping, and role\_mapping tables references the `id` column in the movie table.

The `name\_id` column in the director\_mapping and role\_mapping tables references the `id` column in the names table.

These relationships allow for the association of movies with genres, directors, and actors/actresses.

Q2. Find the total number of rows in each table of the schema.

SELECT COUNT(\*) AS total\_rows FROM movie; --- 7997 rows

SELECT COUNT(\*) AS total\_rows FROM genre; --- 14662 rows

SELECT COUNT(\*) AS total\_rows FROM director\_mapping; --- 3867 rows

SELECT COUNT(\*) AS total\_rows FROM role\_mapping; --- 15615 rows

SELECT COUNT(\*) AS total\_rows FROM names; --- 25735 rows

SELECT COUNT(\*) AS total\_rows FROM ratings; --- 7997 rows

Q3. Identify which columns in the movie table have null values.

SELECT

column\_name

FROM information\_schema.columns

WHERE table\_name = 'movie'

AND is\_nullable = 'YES';

Ans:- Column Names: languages, production\_company, title, worldwide\_gross\_income, year

**SEGMENT 2: Movie Release Trends**

Q1. Determine the total number of movies released each year and analyse the month-wise trend.

SELECT

YEAR(date\_published) AS release\_year,

MONTH(date\_published) AS release\_month,

COUNT(\*) AS movie\_count

FROM

movie

GROUP BY

release\_year, release\_month

ORDER BY

release\_year, release\_month;

Ans: Majority of the movies are released in the months of Sept, Oct, Nov in 2018 & 2017. In 2017 there were 3052 movies released which was highest in 3 years.

Q2. Calculate the number of movies produced in the USA or India in the year 2019.

SELECT

COUNT(\*) AS movie\_count

FROM

movie

WHERE

(country = 'USA' OR country = 'India')

AND year = 2019;

Ans: movie\_count: 8

**SEGMENT 3: Production Statistics and Genre Analysis**

Q1.Retrieve the unique list of genres present in the dataset.

SELECT DISTINCT genre

FROM genre;

Ans: Genre; Drama, Fantasy, Triller, Comedy, Horror, Family, Romance, Adventure, Action, Sci-fi, Crime, Mystery, Others

Q2. Identify the genre with the highest number of movies produced overall.

SELECT genre, COUNT(\*) AS movie\_count

FROM genre

GROUP BY genre

ORDER BY movie\_count DESC

LIMIT 1;

Ans: Genre - Drama || Movie Count – 4285

Q3. Determine the count of movies that belong to only one genre.

SELECT COUNT(\*) AS movie\_count

FROM (

SELECT movie\_id

FROM genre

GROUP BY movie\_id

HAVING COUNT(\*) = 1

) AS single\_genre\_movies;

Ans: Movie Count – 3289

Q4. Calculate the average duration of movies in each genre.

SELECT genre, AVG(duration) AS average\_duration

FROM movie

JOIN genre ON movie.id = genre.movie\_id

GROUP BY genre;

Ans: genre average\_duration

Drama 106.7746

Fantasy 105.1404

Thriller 101.5761

Comedy 102.6227

Horror 92.7243

Family 100.9669

Romance 109.5342

Adventure 101.8714

Action 112.8829

Sci-Fi 97.9413

Crime 107.0517

Mystery 101.8

Others 100.16

Q5. Find the rank of the 'thriller' genre among all genres in terms of the number of movies produced.

SELECT genre, movie\_count, genre\_rank

FROM (

SELECT genre, COUNT(\*) AS movie\_count,

RANK() OVER (ORDER BY COUNT(\*) DESC) AS genre\_rank

FROM genre

GROUP BY genre

) AS genre\_counts

WHERE genre = 'thriller';

--- Ans: Triller || Movie Count - 1484 || Rank - 3

**SEGMENT 4: Ratings Analysis and Crew Members**

Q1. Retrieve the minimum and maximum values in each column of the ratings table (except movie\_id).

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

MIN(total\_votes) AS min\_total\_votes, MAX(total\_votes) AS max\_total\_votes,

MIN(median\_rating) AS min\_median\_rating, MAX(median\_rating) AS max\_median\_rating

FROM ratings;

Ans. Avg\_rating : Min - 1.0 || Max - 10.0

total\_votes: Min - 100 || Max - 725138

median\_rating: Min - 1 || Max - 10

Q2. Identify the top 10 movies based on average rating.

SELECT id, title, avg\_rating

FROM movie

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

ORDER BY avg\_rating DESC

LIMIT 10;

Ans:

id title avg\_rating

tt10914342 Kirket 10

tt6735740 Love in Kilnerry 10

tt9537008 Gini Helida Kathe 9.8

tt10370434 Runam 9.7

tt10867504 Fan 9.6

tt9526826 Android Kunjappan Version 5.25 9.6

tt10869474 Safe 9.5

tt10901588 The Brighton Miracle 9.5

tt9680166 Yeh Suhaagraat Impossible 9.5

tt10405902 Shibu 9.4

Q3 Summarise the ratings table based on movie counts by median ratings.

SELECT median\_rating, COUNT(movie\_id) AS movie\_count

FROM ratings

GROUP BY median\_rating;

Ans:

median\_rating movie\_count

8 1030

7 2257

3 283

6 1975

9 429

2 119

4 479

5 985

10 346

1 94

Q4. Identify the production house that has produced the most number of hit movies (average rating > 8).

SELECT COUNT(movie.id) AS hit\_movie\_count, movie.production\_company, AVG(ratings.avg\_rating) AS average\_rating

FROM movie

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

WHERE ratings.avg\_rating > 8 AND movie.production\_company IS NOT NULL

GROUP BY movie.production\_company

ORDER BY hit\_movie\_count DESC

Limit 1;

Ans: production\_company:- Dream Warrior Picture || Hit\_movie\_count:- 3 || average\_rating:- 8.63333

Q5. Determine the number of movies released in each genre during March 2017 in the USA with more than 1,000 votes.

SELECT genre.genre, COUNT(movie.id) AS movie\_count

FROM movie

JOIN genre ON movie.id = genre.movie\_id

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

WHERE movie.country = 'USA'

AND YEAR(movie.date\_published) = 2017

AND MONTH(movie.date\_published) = 3

AND ratings.total\_votes > 1000

GROUP BY genre.genre

ORDER BY movie\_count DESC;

Ans:

genre movie\_count

Drama 16

Comedy 8

Crime 5

Horror 5

Action 4

Sci-Fi 4

Thriller 4

Romance 3

Fantasy 2

Mystery 2

Family 1

----- Q6. Retrieve movies of each genre starting with the word 'The' and having an average rating > 8.

SELECT COUNT(g.movie\_id) AS movie\_count, g.genre

FROM genre g

INNER JOIN (

SELECT m.id

FROM movie m

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

WHERE r.avg\_rating > 8 AND m.title LIKE 'The%'

) AS sub ON g.movie\_id = sub.id

GROUP BY g.genre;

Ans:

movie\_count genre

7 Drama

1 Horror

1 Mystery

3 Crime

1 Action

1 Thriller

1 Romance

**SEGMENT 5: Crew Analysis**

Q1. Identify the columns in the names table that have null values.

SELECT COLUMN\_NAME

FROM INFORMATION\_SCHEMA.COLUMNS

WHERE TABLE\_NAME = 'names'

AND IS\_NULLABLE = 'YES';

Ans: Columns with Null Values are:- date\_of\_birth, height, known\_for\_movies, name

Q2. Determine the top three directors in the top three genres with movies having an average rating > 8.

SELECT genre.genre AS top\_genre, AVG(ratings.avg\_rating) AS highest\_rated, names.name AS director\_name

FROM movie

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

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

INNER JOIN director\_mapping ON movie.id = director\_mapping.movie\_id

INNER JOIN names ON names.id = director\_mapping.name\_id

WHERE ratings.avg\_rating > 8

GROUP BY top\_genre, director\_name

ORDER BY highest\_rated DESC

LIMIT 3;

Ans:

top\_genre highest\_rated director\_name

Romance 9.7 Srinivas Gundareddy

Drama 9.6 Balavalli Darshith Bhat

Action 9.5 Pradeep Kalipurayath

Q3. Find the top two actors whose movies have a median rating >= 8.

SELECT names.name AS actor\_name, AVG(ratings.median\_rating) AS average\_median\_rating

FROM names

INNER JOIN role\_mapping ON names.id = role\_mapping.name\_id

INNER JOIN ratings ON role\_mapping.movie\_id = ratings.movie\_id

GROUP BY actor\_name

HAVING AVG(ratings.median\_rating) >= 8

ORDER BY average\_median\_rating DESC

LIMIT 2;

Ans:

rating name

10 Aamir Qureshi

10 Aarav Mavi

Q4. Identify the top three production houses based on the number of votes received by their movies.

SELECT movie.production\_company, SUM(ratings.total\_votes) AS total\_votes

FROM movie

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

GROUP BY movie.production\_company

ORDER BY total\_votes DESC

LIMIT 3;

**Ans**:

production\_company total\_votes

Marvel Studios 2656967

Twentieth Century Fox 411163

Warner Bros. 2396057

Q5. Rank actors based on their average ratings in Indian movies released in India.

SELECT names.name AS actor\_name, AVG(ratings.avg\_rating) AS average\_rating

FROM movie

INNER JOIN role\_mapping ON movie.id = role\_mapping.movie\_id

INNER JOIN names ON role\_mapping.name\_id = names.id

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

WHERE movie.country = 'India' AND role\_mapping.category = 'actor'

GROUP BY names.name

ORDER BY average\_rating DESC;

Q6. Identify the top five actresses in Hindi movies released in India based on their average ratings.

SELECT names.name AS actress\_name, AVG(ratings.avg\_rating) AS average\_rating

FROM movie

INNER JOIN role\_mapping ON movie.id = role\_mapping.movie\_id

INNER JOIN names ON role\_mapping.name\_id = names.id

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

WHERE movie.country = 'India' AND movie.languages LIKE '%Hindi%' AND role\_mapping.category = 'actress'

GROUP BY names.name

ORDER BY average\_rating DESC

LIMIT 5;

Ans:

actress\_name average\_rating

Pranati Rai Prakash 9.4

Leera Kaljai 9.2

Puneet Sikka 8.7

Bhairavi Athavle 8.4

Radhika Apte 8.4

**SEGMENT 6: Broader Understanding of Data**

Q1. Classify thriller movies based on average ratings into different categories.

SELECT

movie.title AS movie\_title,

ratings.avg\_rating AS average\_rating,

CASE

WHEN ratings.avg\_rating >= 8.5 THEN 'Excellent'

WHEN ratings.avg\_rating >= 7.5 THEN 'Very Good'

WHEN ratings.avg\_rating >= 6.5 THEN 'Good'

ELSE 'Average or Below'

END AS rating\_category

FROM

movie

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

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

WHERE

genre.genre = 'Thriller'

ORDER BY

ratings.avg\_rating DESC;

Q2. Analyse the genre-wise running total and moving average of the average movie duration.

SELECT

genre.genre AS movie\_genre,

movie.duration AS movie\_duration,

SUM(movie.duration) OVER (PARTITION BY genre.genre ORDER BY movie.year) AS running\_total,

AVG(movie.duration) OVER (PARTITION BY genre.genre ORDER BY movie.year ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS moving\_average

FROM

movie

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

GROUP BY

genre.genre, movie.duration, movie.year

ORDER BY

genre.genre, movie.year;

Q3. Identify the five highest-grossing movies of each year that belong to the top three genres.

WITH top\_three\_genres AS (

SELECT genre, COUNT(\*) AS movie\_count

FROM genre

GROUP BY genre

ORDER BY movie\_count DESC

LIMIT 3

),

highest\_grossing\_movies AS (

SELECT m.year, m.title, m.worlwide\_gross\_income, g.genre,

ROW\_NUMBER() OVER (PARTITION BY m.year, g.genre ORDER BY m.worlwide\_gross\_income DESC) AS `rank`

FROM movie m

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

INNER JOIN top\_three\_genres t ON g.genre = t.genre

)

SELECT year, genre, title, worlwide\_gross\_income

FROM highest\_grossing\_movies

WHERE `rank` <= 5

ORDER BY year, genre, `rank`;

Q4. Determine the top two production houses that have produced the highest number of hits among multilingual movies.

WITH hit\_movies AS (

SELECT m.production\_company, COUNT(\*) AS hit\_count

FROM movie m

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

WHERE r.avg\_rating >= 7.0

AND m.production\_company IS NOT NULL

GROUP BY m.production\_company

),

top\_production\_houses AS (

SELECT production\_company, hit\_count

FROM hit\_movies

ORDER BY hit\_count DESC

LIMIT 2

)

SELECT production\_company, hit\_count

FROM top\_production\_houses;

**Ans:** A24:- 7 || Warner Bros. :- 6

Q5. Identify the top three actresses based on the number of Super Hit movies (average rating > 8) in the drama genre.

WITH super\_hit\_drama\_movies AS (

SELECT m.id AS movie\_id, m.title, r.avg\_rating

FROM movie m

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

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

WHERE r.avg\_rating > 8.0

AND g.genre = 'drama'

),

actresses AS (

SELECT m.id AS movie\_id, nm.name

FROM movie m

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

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

WHERE rm.category = 'actress'

),

actress\_movie\_count AS (

SELECT a.name, COUNT(\*) AS movie\_count

FROM super\_hit\_drama\_movies s

INNER JOIN actresses a ON s.movie\_id = a.movie\_id

GROUP BY a.name

),

ranked\_actresses AS (

SELECT name, movie\_count, ROW\_NUMBER() OVER (ORDER BY movie\_count DESC) AS `rank`

FROM actress\_movie\_count

)

SELECT name, movie\_count

FROM ranked\_actresses

WHERE `rank` <= 3;

Option 2

Query to identify the top three actresses based on the number of Super Hit movies (average rating > 8) in the drama genre

SELECT a.name, COUNT(\*) AS movie\_count

FROM movie m

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

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

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

INNER JOIN names a ON rm.name\_id = a.id

WHERE r.avg\_rating > 8.0

AND g.genre = 'drama'

AND rm.category = 'actress'

GROUP BY a.name

ORDER BY movie\_count DESC

LIMIT 3;

Ans:

name movie\_count

Parvathy Thiruvothu 2

Susan Brown 2

Amanda Lawrence 2

Q6. Retrieve details for the top nine directors based on the number of movies, including average inter-movie duration, ratings, and more.

WITH director\_movie\_count AS (

SELECT dm.name\_id, nm.name, COUNT(\*) AS movie\_count

FROM director\_mapping dm

INNER JOIN names nm ON dm.name\_id = nm.id

GROUP BY dm.name\_id, nm.name

),

director\_average\_duration AS (

SELECT dm.name\_id, AVG(m.duration) AS average\_duration

FROM director\_mapping dm

INNER JOIN movie m ON dm.movie\_id = m.id

GROUP BY dm.name\_id

),

director\_total\_ratings AS (

SELECT dm.name\_id, SUM(r.total\_votes) AS total\_votes

FROM director\_mapping dm

INNER JOIN ratings r ON dm.movie\_id = r.movie\_id

GROUP BY dm.name\_id

),

ranked\_directors AS (

SELECT dmc.name\_id, dmc.name, dmc.movie\_count, ad.average\_duration, tr.total\_votes,

ROW\_NUMBER() OVER (ORDER BY dmc.movie\_count DESC) AS `rank`

FROM director\_movie\_count dmc

LEFT JOIN director\_average\_duration ad ON dmc.name\_id = ad.name\_id

LEFT JOIN director\_total\_ratings tr ON dmc.name\_id = tr.name\_id

)

SELECT name, movie\_count, average\_duration, total\_votes

FROM ranked\_directors

WHERE `rank` <= 9;

2nd option

SELECT nm.name, COUNT(\*) AS movie\_count, AVG(m.duration) AS average\_duration, SUM(r.total\_votes) AS total\_votes

FROM director\_mapping dm

INNER JOIN names nm ON dm.name\_id = nm.id

INNER JOIN movie m ON dm.movie\_id = m.id

INNER JOIN ratings r ON dm.movie\_id = r.movie\_id

GROUP BY dm.name\_id, nm.name

ORDER BY movie\_count DESC

LIMIT 9;

Ans:

name movie\_count average\_duration total\_votes

A.L. Vijay 5 122.6 1754

Andrew Jones 5 86.4 1989

Chris Stokes 4 88 3664

Justin Price 4 86.5 5343

Jesse V. Johnson 4 95.75 14778

Steven Soderbergh 4 100.25 171684

Sion Sono 4 125.5 2972

Ã–zgÃ¼r Bakar 4 93.5 1092

Sam Liu 4 78 28557

**Segment 7: Recommendations**

Based on the analysis, provide recommendations for the types of content Bolly Movies should focus on producing.

Ans: Based on the Analysis of the IMDB Movies, the recommendations for the types of content Bolly Movies should focus on producing is:-

1. The 'Triller' genre has caught the highest attention and interest amongst the audience as the amount of 'Thriller' movies watched is good, so the Bollywood movie production houses should keep their interest towards producing more 'Thriller' genre movies.

2. The 'Drama' genre has gained the overall average highest IMDB rating by the audience, so the Bollywood movies production houses should focus more on producing quality content movies in the 'Drama' genre as they have been doing.

3. The Bollywood movie production houses should also focus on producing good quality movies in other genres as well for the growth of the Bollywood movie industry.

**Extra Questions:**

Q1. Determine the average duration of movies released by Bolly Movies compared to the industry average.

WITH hindi\_movies\_average\_duration AS (

SELECT AVG(duration) AS hindi\_average

FROM movie

WHERE languages LIKE '%Hindi%'

),

other\_languages\_average\_duration AS (

SELECT AVG(duration) AS other\_languages\_average

FROM movie

WHERE languages NOT LIKE '%Hindi%'

)

SELECT hindi\_average, other\_languages\_average

FROM hindi\_movies\_average\_duration, other\_languages\_average\_duration;

option 2

SELECT

AVG(CASE WHEN languages LIKE '%Hindi%' THEN duration END) AS hindi\_average,

AVG(CASE WHEN languages NOT LIKE '%Hindi%' THEN duration END) AS other\_languages\_average

FROM movie

WHERE duration IS NOT NULL;

Ans: Hindi Average:- 125.9795 || Other Language average: 103.1308

Q2. Query to analyze the correlation between the number of votes and the average rating for movies produced in Hindi

WITH hindi\_movie\_stats AS (

SELECT

AVG(r.total\_votes) AS avg\_votes,

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

FROM movie m

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

WHERE m.languages LIKE '%Hindi%'

AND r.total\_votes IS NOT NULL

AND r.avg\_rating IS NOT NULL

)

SELECT

avg\_votes AS average\_votes,

avg\_rating AS average\_rating,

(SUM((r.total\_votes - avg\_votes) \* (r.avg\_rating - avg\_rating)) / COUNT(\*)) /

(SQRT(SUM(POW(r.total\_votes - avg\_votes, 2)) / COUNT(\*)) \* SQRT(SUM(POW(r.avg\_rating - avg\_rating, 2)) / COUNT(\*))) AS correlation

FROM movie m

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

CROSS JOIN hindi\_movie\_stats;

Q3. Find the production house that has consistently produced movies with high ratings over the past three years.

WITH high\_ratings\_movies AS (

SELECT m.production\_company, r.avg\_rating

FROM movie m

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

WHERE m.date\_published >= '2017-01-01' AND m.date\_published <= '2019-12-31'

AND r.avg\_rating >= 8.0

AND m.production\_company IS NOT NULL

AND r.avg\_rating IS NOT NULL

),

production\_house\_ratings AS (

SELECT production\_company, COUNT(\*) AS num\_high\_ratings

FROM high\_ratings\_movies

WHERE production\_company IS NOT NULL

GROUP BY production\_company

),

consistent\_high\_ratings AS (

SELECT production\_company

FROM production\_house\_ratings

WHERE num\_high\_ratings = 3

AND production\_company IS NOT NULL

)

SELECT production\_company

FROM consistent\_high\_ratings;

Option 2:

SELECT m.production\_company

FROM movie m

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

WHERE m.date\_published >= '2017-01-01' AND m.date\_published <= '2019-12-31'

AND r.avg\_rating >= 8.0

AND m.production\_company IS NOT NULL

AND r.avg\_rating IS NOT NULL

GROUP BY m.production\_company

HAVING COUNT(\*) = 3;

Ans: Dream Warrior Pictures || National Theatre Live

Q4. Identify the top three directors who have successfully delivered commercially successful movies with high ratings.

WITH commercially\_successful\_movies AS (

SELECT m.id, m.production\_company, r.avg\_rating, m.worlwide\_gross\_income

FROM movie m

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

WHERE m.worlwide\_gross\_income IS NOT NULL

AND r.avg\_rating >= 8.0

AND m.production\_company IS NOT NULL

),

director\_success\_counts AS (

SELECT dm.name\_id, COUNT(\*) AS success\_count

FROM commercially\_successful\_movies csm

INNER JOIN director\_mapping dm ON csm.id = dm.movie\_id

GROUP BY dm.name\_id

),

directors\_commercial\_ratings AS (

SELECT dm.name\_id, COUNT(\*) AS total\_movies, MAX(success\_count) AS max\_success\_count

FROM commercially\_successful\_movies csm

INNER JOIN director\_mapping dm ON csm.id = dm.movie\_id

INNER JOIN director\_success\_counts dsc ON dm.name\_id = dsc.name\_id

GROUP BY dm.name\_id

HAVING COUNT(\*) >= 1

AND MAX(success\_count) >= 1

ORDER BY MAX(success\_count) DESC

LIMIT 3

)

SELECT n.name AS director\_name, dcr.total\_movies, dcr.max\_success\_count

FROM directors\_commercial\_ratings dcr

INNER JOIN names n ON dcr.name\_id = n.id

ORDER BY dcr.max\_success\_count DESC;

Ans:

director\_name total\_movies max\_success\_count

Joe Russo 2 2

Anthony Russo 2 2

James Mangold 2 2

Segment 7: Recommendations