Movie Streaming Platform — DBMS Project

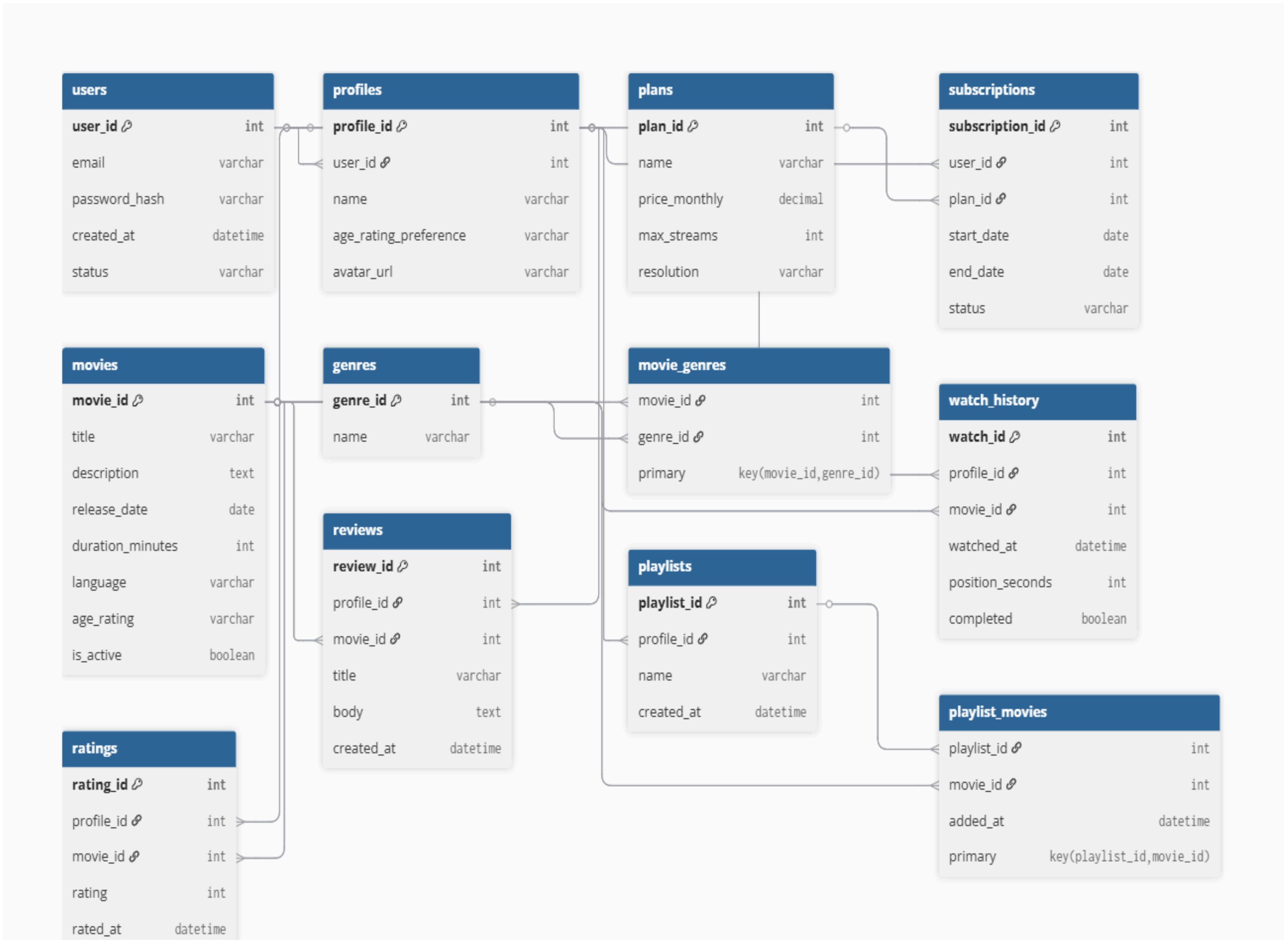
Submitted by: Yashita Bahrani (BT23CSE218)
Submitted to: Mr. Karan Potdukhe Sir

1. ER Diagram

Entities (Tables):

- users: Stores core account credentials and status.
- profiles: Manages individual user profiles under a single account (e.g., "Kids," "John's Profile").
- plans: Defines the available subscription plans (e.g., Basic, Premium 4K).
- subscriptions: Tracks the relationship between a user and their chosen plan over time.
- movies: Contains all metadata for movies, such as title, duration, and release date.
- genres: A lookup table for movie genres (e.g., Action, Romance).
- movie_genres: A junction table creating a many-to-many relationship between movies and genres.
- watch_history: Logs the viewing activity for each profile.
- ratings: Stores the 1-5 star ratings given by a profile to a movie.
- reviews: Stores detailed text reviews written by profiles for movies.
- playlists: Allows profiles to create named collections of movies.
- playlist_movies: A junction table linking playlists to the movies they contain.

BT23CSE218 - Yashita Bahrani



Cardinalities:

Relationship	Cardinality	Meaning
users → profiles	**1 → N**	One user can create multiple profiles (like Netflix).
users → subscriptions	**1 → N**	One user may have multiple historical subscriptions.
plans → subscriptions	**1 → N**	One plan can be chosen by many users.

movies ↔ genres	**M ↔ M**	Many movies can belong to many genres — via `movie_genres`.
profiles → watch_history	**1 → N**	Each profile has multiple watch history entries.
profiles → ratings	**1 → N**	Each profile can rate many movies.
profiles → reviews	**1 → N**	Each profile can review many movies.
profiles → playlists	**1 → N**	Each profile can create multiple playlists.
playlists ↔ movies	**M ↔ M**	Many movies can appear in many playlists — via `playlist_movies`.

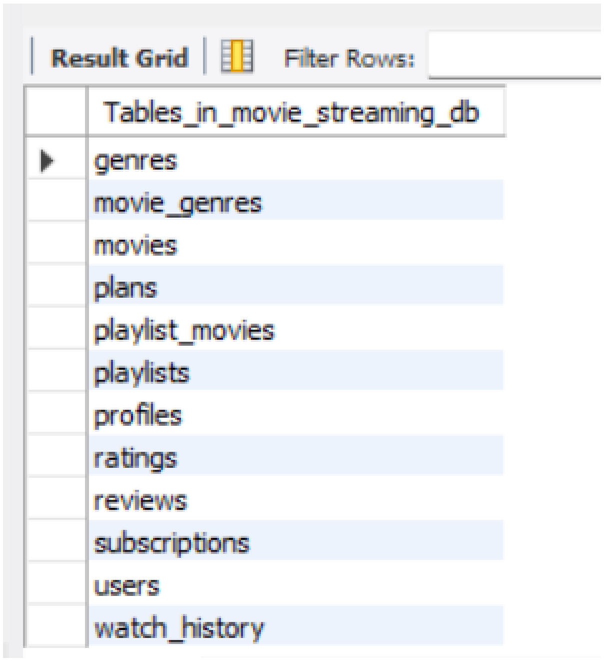
---

2. SQL Implementation

```
CREATE database movie_streaming_db;

USE movie_streaming_db;



CREATE TABLE users (

  user_id INT AUTO_INCREMENT PRIMARY KEY,

  email VARCHAR(255) UNIQUE NOT NULL,

  password_hash VARCHAR(255) NOT NULL,

  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

  status ENUM('active','suspended') DEFAULT 'active'

);


CREATE TABLE profiles (
  profile_id INT AUTO_INCREMENT PRIMARY KEY,
  user_id INT,
  name VARCHAR(100) NOT NULL,
  age_rating_preference VARCHAR(10) DEFAULT 'PG-13',
  avatar_url TEXT,
  FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE CASCADE
);

CREATE TABLE plans (
  plan_id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(50) UNIQUE NOT NULL,
  price_monthly DECIMAL(6,2) NOT NULL,
  max_streams INT NOT NULL,
  resolution VARCHAR(20)
);

CREATE TABLE subscriptions (
  subscription_id INT AUTO_INCREMENT PRIMARY KEY,
  user_id INT,
  plan_id INT,
  start_date DATE NOT NULL,
  end_date DATE,
  status ENUM('active','expired','cancelled') DEFAULT 'active',
  FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE CASCADE,
  FOREIGN KEY (plan_id) REFERENCES plans(plan_id) ON DELETE RESTRICT
);
```

BT23CSE218 - Yashita Bahrani

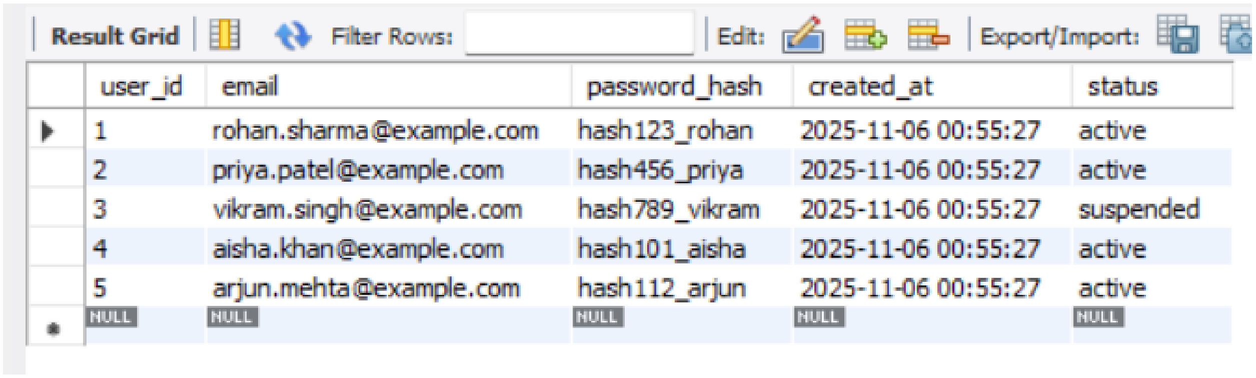
```
CREATE TABLE movies (
  movie_id INT AUTO_INCREMENT PRIMARY KEY,
  title VARCHAR(255) NOT NULL,
  description TEXT,
  release_date DATE,
  duration_minutes INT CHECK (duration_minutes > 0),
  language VARCHAR(50),
  age_rating VARCHAR(10),
  is_active BOOLEAN DEFAULT TRUE
);

CREATE TABLE genres (
  genre_id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(50) UNIQUE NOT NULL
);

CREATE TABLE movie_genres (
  movie_id INT,
  genre_id INT,
  PRIMARY KEY (movie_id, genre_id),
  FOREIGN KEY (movie_id) REFERENCES movies(movie_id) ON DELETE CASCADE,
  FOREIGN KEY (genre_id) REFERENCES genres(genre_id) ON DELETE CASCADE
);

CREATE TABLE watch_history (
  watch_id INT AUTO_INCREMENT PRIMARY KEY,
  profile_id INT,
  movie_id INT,
  watched_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  position_seconds INT,
  completed BOOLEAN DEFAULT FALSE,
  FOREIGN KEY (profile_id) REFERENCES profiles(profile_id) ON DELETE
CASCADE,
  FOREIGN KEY (movie_id) REFERENCES movies(movie_id) ON DELETE CASCADE
);

CREATE TABLE ratings (
  rating_id INT AUTO_INCREMENT PRIMARY KEY,
  profile_id INT,
  movie_id INT,
  rating INT CHECK (rating BETWEEN 1 AND 5),
  rated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  UNIQUE (profile_id, movie_id),
  FOREIGN KEY (profile_id) REFERENCES profiles(profile_id) ON DELETE
CASCADE,
  FOREIGN KEY (movie_id) REFERENCES movies(movie_id) ON DELETE CASCADE
);

CREATE TABLE reviews (
  review_id INT AUTO_INCREMENT PRIMARY KEY,
  profile_id INT,
  movie_id INT,
  title VARCHAR(100),

  body TEXT,

  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

  FOREIGN KEY (profile_id) REFERENCES profiles(profile_id) ON DELETE
CASCADE,

  FOREIGN KEY (movie_id) REFERENCES movies(movie_id) ON DELETE CASCADE

);


CREATE TABLE playlists (

  playlist_id INT AUTO_INCREMENT PRIMARY KEY,

  profile_id INT,

  name VARCHAR(100) NOT NULL,

  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

  FOREIGN KEY (profile_id) REFERENCES profiles(profile_id) ON DELETE CASCADE

);


CREATE TABLE playlist_movies (

  playlist_id INT,

  movie_id INT,

  added_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

  PRIMARY KEY (playlist_id, movie_id),

  FOREIGN KEY (playlist_id) REFERENCES playlists(playlist_id) ON DELETE
CASCADE,

  FOREIGN KEY (movie_id) REFERENCES movies(movie_id) ON DELETE CASCADE

);


SHOW TABLES;
```


Result Grid | Filter Rows:

Tables_in_movie_streaming_db
genres
movie_genres
movies
plans
playlist_movies
playlists
profiles
ratings
reviews
subscriptions
users
watch_history

4. Sample Data: INSERT statements

```
INSERT INTO users (email, password_hash, status) VALUES
('rohan.sharma@example.com', 'hash123_rohan', 'active'),
('priya.patel@example.com', 'hash456_priya', 'active'),
('vikram.singh@example.com', 'hash789_vikram', 'suspended'),
('aisha.khan@example.com', 'hash101_aisha', 'active'),
('arjun.mehta@example.com', 'hash112_arjun', 'active');
```


Result Grid | Filter Rows: | Edit: | Export/Import:

user_id	email	password_hash	created_at	status
1	rohan.sharma@example.com	hash123_rohan	2025-11-06 00:55:27	active
2	priya.patel@example.com	hash456_priya	2025-11-06 00:55:27	active
3	vikram.singh@example.com	hash789_vikram	2025-11-06 00:55:27	suspended
4	aisha.khan@example.com	hash101_aisha	2025-11-06 00:55:27	active
5	arjun.mehta@example.com	hash112_arjun	2025-11-06 00:55:27	active
NULL	NULL	NULL	NULL	NULL

```
INSERT INTO plans (name, price_monthly, max_streams, resolution) VALUES
```

BT23CSE218 - Yashita Bahrani

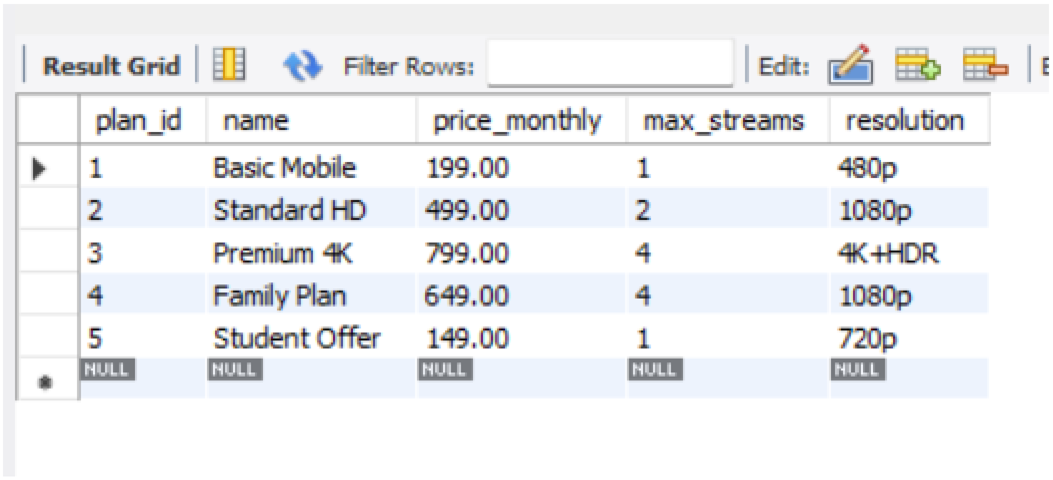
('Basic Mobile', 199.00, 1, '480p'),

('Standard HD', 499.00, 2, '1080p'),

('Premium 4K', 799.00, 4, '4K+HDR'),

('Family Plan', 649.00, 4, '1080p'),

('Student Offer', 149.00, 1, '720p');



plan_id	name	price_monthly	max_streams	resolution
1	Basic Mobile	199.00	1	480p
2	Standard HD	499.00	2	1080p
3	Premium 4K	799.00	4	4K+HDR
4	Family Plan	649.00	4	1080p
5	Student Offer	149.00	1	720p
NULL	NULL	NULL	NULL	NULL

INSERT INTO subscriptions (user_id, plan_id, start_date, end_date, status) VALUES

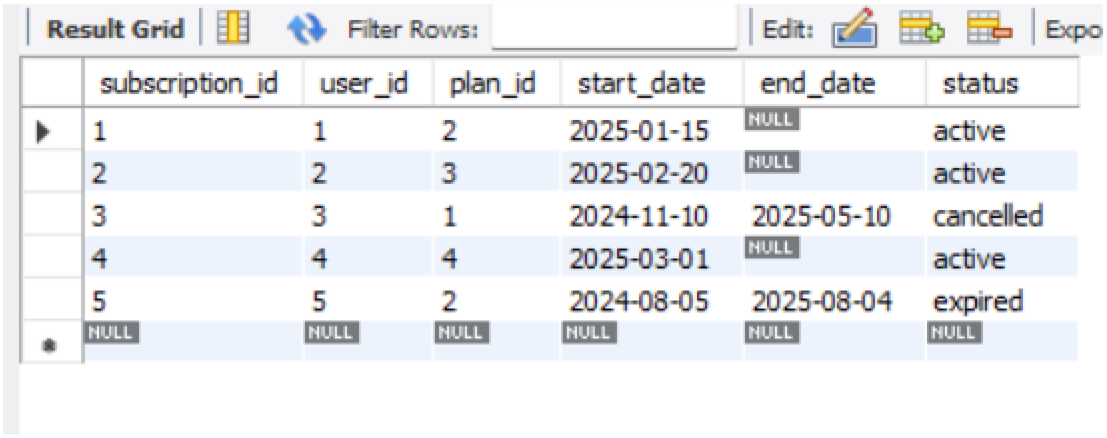
(1, 2, '2025-01-15', NULL, 'active'),

(2, 3, '2025-02-20', NULL, 'active'),

(3, 1, '2024-11-10', '2025-05-10', 'cancelled'),

(4, 4, '2025-03-01', NULL, 'active'),

(5, 2, '2024-08-05', '2025-08-04', 'expired');

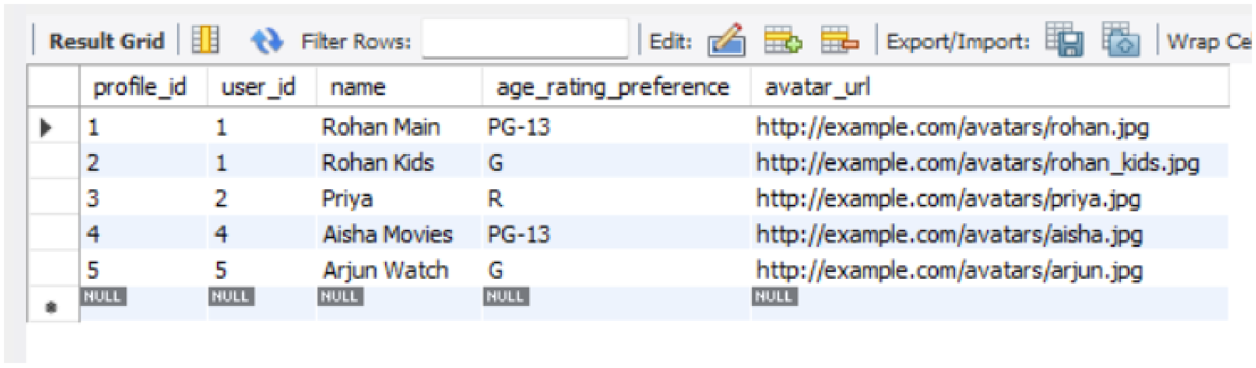


subscription_id	user_id	plan_id	start_date	end_date	status
1	1	2	2025-01-15	NULL	active
2	2	3	2025-02-20	NULL	active
3	3	1	2024-11-10	2025-05-10	cancelled
4	4	4	2025-03-01	NULL	active
5	5	2	2024-08-05	2025-08-04	expired
NULL	NULL	NULL	NULL	NULL	NULL

INSERT INTO profiles (user_id, name, age_rating_preference, avatar_url) VALUES

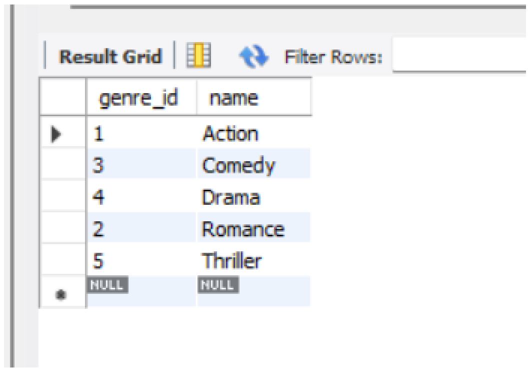
BT23CSE218 - Yashita Bahrani

(1, 'Rohan Main', 'PG-13', 'http://example.com/avatars/rohan.jpg'),
(1, 'Rohan Kids', 'G', 'http://example.com/avatars/rohan_kids.jpg'),
(2, 'Priya', 'R', 'http://example.com/avatars/priya.jpg'),
(4, 'Aisha Movies', 'PG-13', 'http://example.com/avatars/aisha.jpg'),
(5, 'Arjun Watch', 'G', 'http://example.com/avatars/arjun.jpg');



profile_id	user_id	name	age_rating_preference	avatar_url
1	1	Rohan Main	PG-13	http://example.com/avatars/rohan.jpg
2	1	Rohan Kids	G	http://example.com/avatars/rohan_kids.jpg
3	2	Priya	R	http://example.com/avatars/priya.jpg
4	4	Aisha Movies	PG-13	http://example.com/avatars/aisha.jpg
5	5	Arjun Watch	G	http://example.com/avatars/arjun.jpg
NULL	NULL	NULL	NULL	NULL

INSERT INTO genres (name) VALUES
('Action'),
('Romance'),
('Comedy'),
('Drama'),
('Thriller');

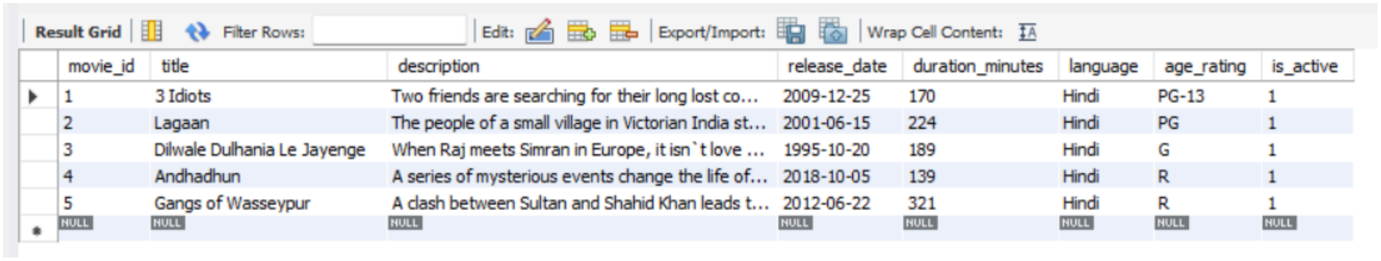


genre_id	name
1	Action
3	Comedy
4	Drama
2	Romance
5	Thriller
NULL	NULL

INSERT INTO movies (title, description, release_date, duration_minutes, language, age_rating, is_active) VALUES
('3 Idiots', 'Two friends are searching for their long lost companion. They revisit their college days and recall the memories of their friend who inspired them to think differently.', '2009-12-25', 170, 'Hindi', 'PG-13', TRUE),

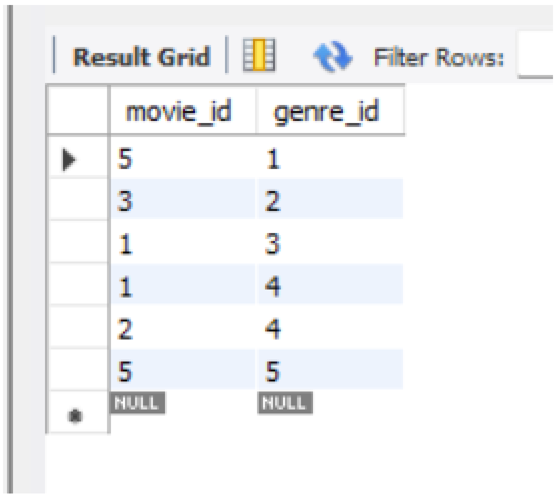
BT23CSE218 - Yashita Bahrani

```
('Lagaan', 'The people of a small village in Victorian India stake their future on a game of cricket against their ruthless British rulers.', '2001-06-15', 224, 'Hindi', 'PG', TRUE),
('Dilwale Dulhania Le Jayenge', 'When Raj meets Simran in Europe, it isn`t love at first sight but when Simran moves to India for an arranged marriage, love strikes.', '1995-10-20', 189, 'Hindi', 'G', TRUE),
('Andhadhun', 'A series of mysterious events change the life of a blind pianist, who must now report a crime that he should technically know nothing of.', '2018-10-05', 139, 'Hindi', 'R', TRUE),
('Gangs of Wasseypur', 'A clash between Sultan and Shahid Khan leads to the expulsion of Khan from Wasseypur, and ignites a deadly blood feud spanning three generations.', '2012-06-22', 321, 'Hindi', 'R', TRUE);
```



movie_id	title	description	release_date	duration_minutes	language	age_rating	is_active
1	3 Idiots	Two friends are searching for their long lost co...	2009-12-25	170	Hindi	PG-13	1
2	Lagaan	The people of a small village in Victorian India st...	2001-06-15	224	Hindi	PG	1
3	Dilwale Dulhania Le Jayenge	When Raj meets Simran in Europe, it isn`t love ...	1995-10-20	189	Hindi	G	1
4	Andhadhun	A series of mysterious events change the life of...	2018-10-05	139	Hindi	R	1
5	Gangs of Wasseypur	A clash between Sultan and Shahid Khan leads t...	2012-06-22	321	Hindi	R	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

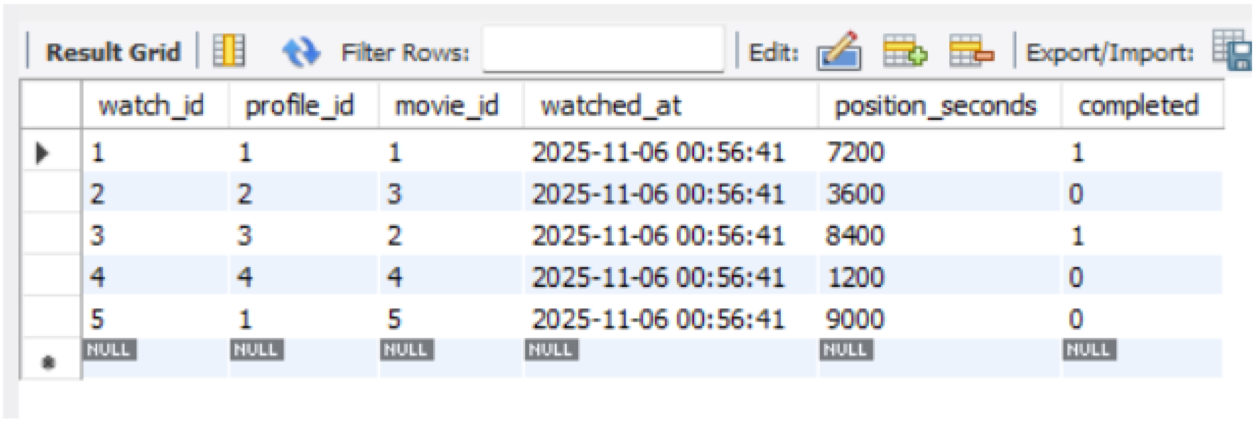
```
INSERT INTO movie_genres (movie_id, genre_id) VALUES
(1, 3), -- 3 Idiots -> Comedy
(1, 4), -- 3 Idiots -> Drama
(2, 4), -- Lagaan -> Drama
(3, 2), -- DDLJ -> Romance
(5, 1), -- Gangs of Wasseypur -> Action
(5, 5); -- Gangs of Wasseypur -> Thriller
```



movie_id	genre_id
5	1
3	2
1	3
1	4
2	4
5	5
NULL	NULL

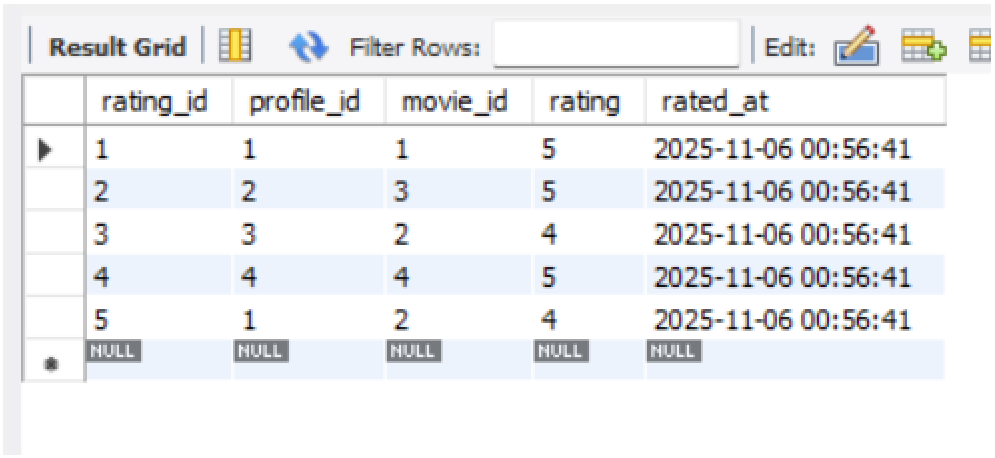
BT23CSE218 - Yashita Bahrani

```
INSERT INTO watch_history (profile_id, movie_id, position_seconds, completed)
VALUES
(1, 1, 7200, TRUE),   -- Rohan Main watched 3 Idiots
(2, 3, 3600, FALSE),  -- Priya watched DDLJ
(3, 2, 8400, TRUE),   -- Priya watched Lagaan
(4, 4, 1200, FALSE),  -- Aisha watched Andhadhun
(1, 5, 9000, FALSE);  -- Rohan Main watched Gangs of Wasseypur
```



watch_id	profile_id	movie_id	watched_at	position_seconds	completed
1	1	1	2025-11-06 00:56:41	7200	1
2	2	3	2025-11-06 00:56:41	3600	0
3	3	2	2025-11-06 00:56:41	8400	1
4	4	4	2025-11-06 00:56:41	1200	0
5	1	5	2025-11-06 00:56:41	9000	0
NULL	NULL	NULL	NULL	NULL	NULL

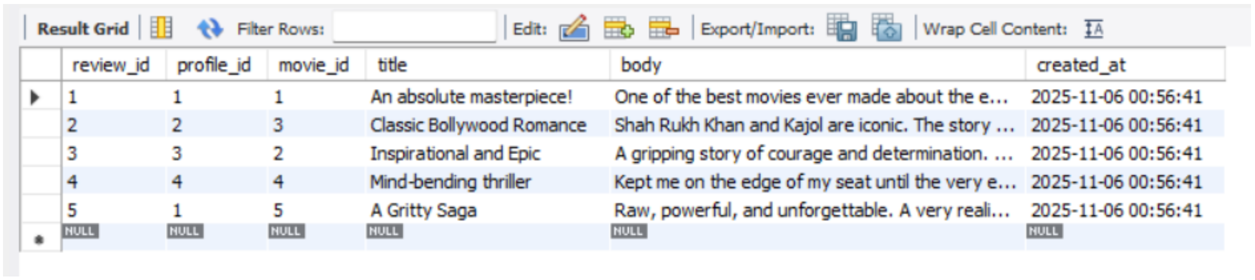
```
INSERT INTO ratings (profile_id, movie_id, rating) VALUES
(1, 1, 5),   -- Rohan Main rated 3 Idiots
(2, 3, 5),   -- Priya rated DDLJ
(3, 2, 4),   -- Priya rated Lagaan
(4, 4, 5),   -- Aisha rated Andhadhun
(1, 2, 4);   -- Rohan Main rated Lagaan
```



rating_id	profile_id	movie_id	rating	rated_at
1	1	1	5	2025-11-06 00:56:41
2	2	3	5	2025-11-06 00:56:41
3	3	2	4	2025-11-06 00:56:41
4	4	4	5	2025-11-06 00:56:41
5	1	2	4	2025-11-06 00:56:41
NULL	NULL	NULL	NULL	NULL

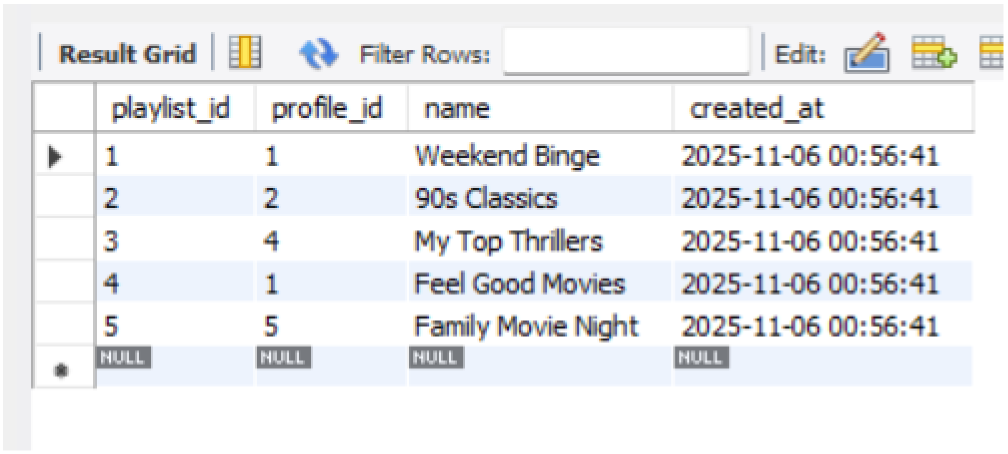
BT23CSE218 - Yashita Bahrani

```
INSERT INTO reviews (profile_id, movie_id, title, body) VALUES
(1, 1, 'An absolute masterpiece!', 'One of the best movies ever made about the education system. A must watch!'),
(2, 3, 'Classic Bollywood Romance', 'Shah Rukh Khan and Kajol are iconic. The story is timeless.'),
(3, 2, 'Inspirational and Epic', 'A gripping story of courage and determination. The cricket match was legendary.'),
(4, 4, 'Mind-bending thriller', 'Kept me on the edge of my seat until the very end. The plot twists are incredible.'),
(1, 5, 'A Gritty Saga', 'Raw, powerful, and unforgettable. A very realistic portrayal of crime and power struggles.');
```



review_id	profile_id	movie_id	title	body	created_at
1	1	1	An absolute masterpiece!	One of the best movies ever made about the e...	2025-11-06 00:56:41
2	2	3	Classic Bollywood Romance	Shah Rukh Khan and Kajol are iconic. The story ...	2025-11-06 00:56:41
3	3	2	Inspirational and Epic	A gripping story of courage and determination. ...	2025-11-06 00:56:41
4	4	4	Mind-bending thriller	Kept me on the edge of my seat until the very e...	2025-11-06 00:56:41
5	1	5	A Gritty Saga	Raw, powerful, and unforgettable. A very reali...	2025-11-06 00:56:41
NULL	NULL	NULL	NULL	NULL	NULL

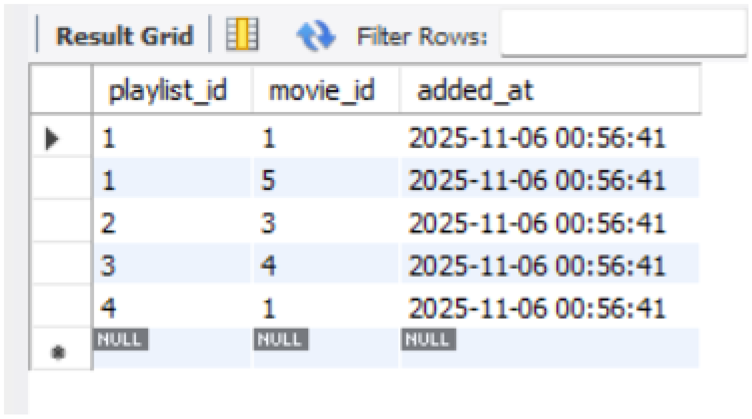
```
INSERT INTO playlists (profile_id, name) VALUES
(1, 'Weekend Binge'),
(2, '90s Classics'),
(4, 'My Top Thrillers'),
(1, 'Feel Good Movies'),
(5, 'Family Movie Night');
```



playlist_id	profile_id	name	created_at
1	1	Weekend Binge	2025-11-06 00:56:41
2	2	90s Classics	2025-11-06 00:56:41
3	4	My Top Thrillers	2025-11-06 00:56:41
4	1	Feel Good Movies	2025-11-06 00:56:41
5	5	Family Movie Night	2025-11-06 00:56:41
NULL	NULL	NULL	NULL

```
INSERT INTO playlist_movies (playlist_id, movie_id) VALUES
```

(1, 1), -- Add '3 Idiots' to 'Weekend Binge'
(1, 5), -- Add 'Gangs of Wasseypur' to 'Weekend Binge'
(2, 3), -- Add 'DDLJ' to '90s Classics'
(3, 4), -- Add 'Andhadhun' to 'My Top Thrillers'
(4, 1); -- Add '3 Idiots' to 'Feel Good Movies'



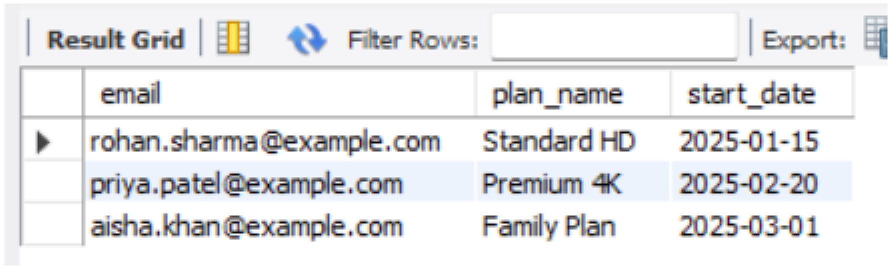
playlist_id	movie_id	added_at
1	1	2025-11-06 00:56:41
1	5	2025-11-06 00:56:41
2	3	2025-11-06 00:56:41
3	4	2025-11-06 00:56:41
4	1	2025-11-06 00:56:41
NULL	NULL	NULL

5. Advanced SQL Queries (30)

a. Joins (INNER, LEFT, RIGHT, SELF, CROSS)

1. **Get a list of all users who have an active subscription and show their email and the name of their subscription plan.**

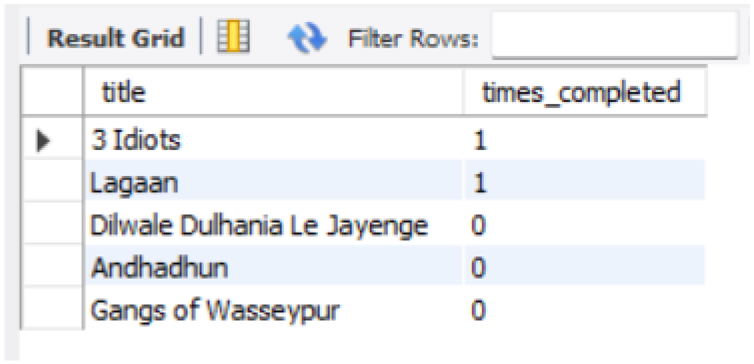
```
SELECT

    u.email,

    p.name AS plan_name,

    s.start_date

FROM users u

INNER JOIN subscriptions s ON u.user_id = s.user_id

INNER JOIN plans p ON s.plan_id = p.plan_id

WHERE s.status = 'active';
```



email	plan_name	start_date
rohan.sharma@example.com	Standard HD	2025-01-15
priya.patel@example.com	Premium 4K	2025-02-20
aisha.khan@example.com	Family Plan	2025-03-01

2. **List all movies and the number of times each has been watched to completion. Include movies that have never been watched.**

```
SELECT

    m.title,

    COUNT(w.watch_id) AS times_completed

FROM movies m

LEFT JOIN watch_history w ON m.movie_id = w.movie_id AND w.completed = TRUE

GROUP BY m.title

ORDER BY times_completed DESC;
```

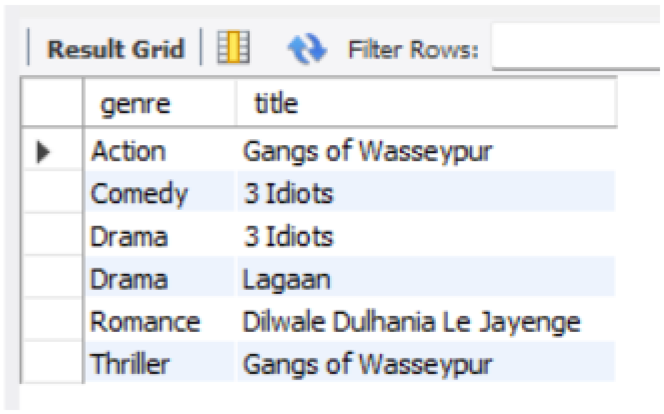


title	times_completed
3 Idiots	1
Lagaan	1
Dilwale Dulhania Le Jayenge	0
Andhadhun	0
Gangs of Wasseypur	0

3. **Show all genres and the titles of movies associated with them. Ensure that genres with no movies are still listed.**

```
SELECT

    g.name AS genre,

    m.title

FROM movie_genres mg

RIGHT JOIN genres g ON mg.genre_id = g.genre_id
```

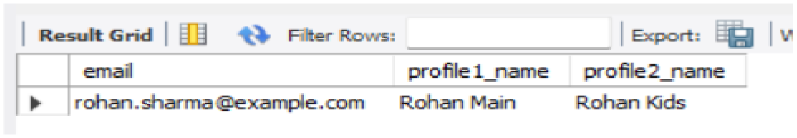
```
LEFT JOIN movies m ON mg.movie_id = m.movie_id

ORDER BY g.name;
```



genre	title
Action	Gangs of Wasseypur
Comedy	3 Idiots
Drama	3 Idiots
Drama	Lagaan
Romance	Dilwale Dulhania Le Jayenge
Thriller	Gangs of Wasseypur

4. **Find pairs of profiles that belong to the same user account.**

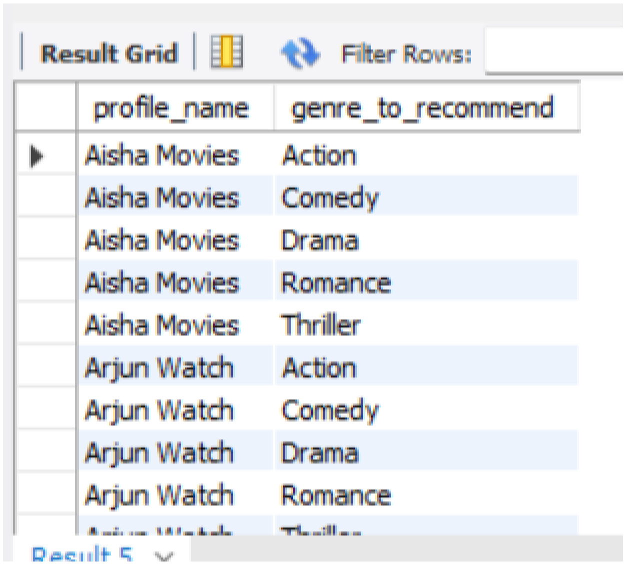
```
SELECT

    u.email,

    p1.name AS profile1_name,

    p2.name AS profile2_name

FROM profiles p1

INNER JOIN profiles p2 ON p1.user_id = p2.user_id AND p1.profile_id <
p2.profile_id

INNER JOIN users u ON p1.user_id = u.user_id;
```



email	profile1_name	profile2_name
rohan.sharma@example.com	Rohan Main	Rohan Kids

5. **Generate a report that shows every possible combination of a user profile and a genre, to help create a recommendation matrix.**

```
SELECT

    p.name AS profile_name,

    g.name AS genre_to_recommend

FROM profiles p

CROSS JOIN genres g

ORDER BY p.name, g.name;
```



Result Grid | Filter Rows:

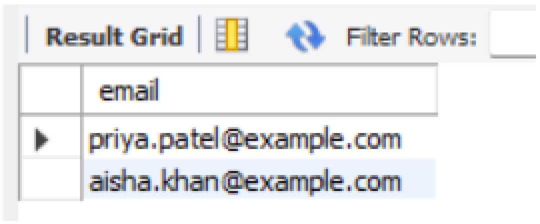
profile_name	genre_to_recommend
Aisha Movies	Action
Aisha Movies	Comedy
Aisha Movies	Drama
Aisha Movies	Romance
Aisha Movies	Thriller
Arjun Watch	Action
Arjun Watch	Comedy
Arjun Watch	Drama
Arjun Watch	Romance
Arjun Watch	Thriller

Result 5

b. Subqueries (IN, EXISTS, ANY, ALL)

6. **Find the email addresses of all users who have subscribed to a plan that costs more than 500 per month.**

```
SELECT email

FROM users

WHERE user_id IN (

    SELECT user_id

    FROM subscriptions

    WHERE plan_id IN (

        SELECT plan_id

        FROM plans

        WHERE price_monthly > 500.00

    )

);
```

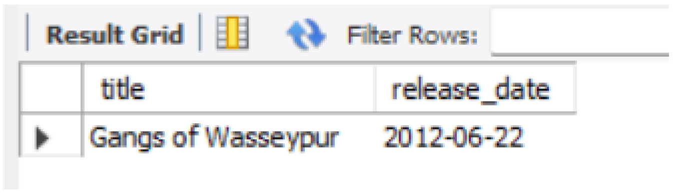


Result Grid | Filter Rows:

email
priya.patel@example.com
aisha.khan@example.com

7. **List all movies that have not received any ratings.**

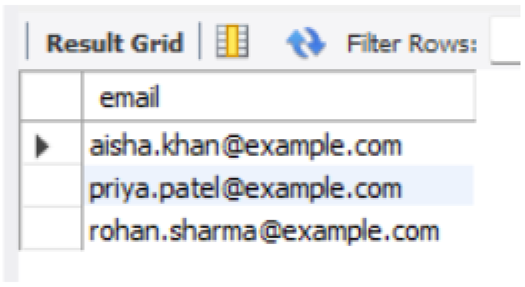
```
SELECT title, release_date

FROM movies

WHERE movie_id NOT IN (

    SELECT DISTINCT movie_id

    FROM ratings

);
```



```
title               release_date
Gangs of Wasseypur  2012-06-22
```

8. **Find all users who have at least one profile that has written a review.**

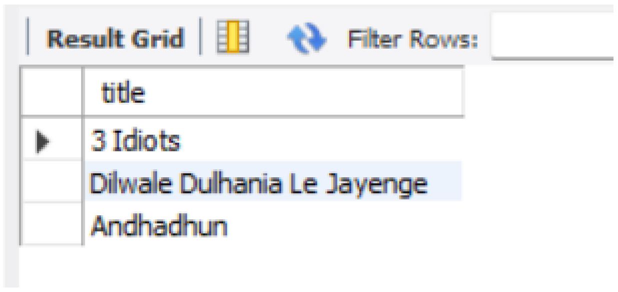
```
SELECT email

FROM users u

WHERE EXISTS (

    SELECT 1

    FROM profiles p

    JOIN reviews r ON p.profile_id = r.profile_id

    WHERE p.user_id = u.user_id

);
```



```
email
aisha.khan@example.com
priya.patel@example.com
rohan.sharma@example.com
```

9. **Find a movie that is rated higher than ANY movie in the 'Comedy' genre.**

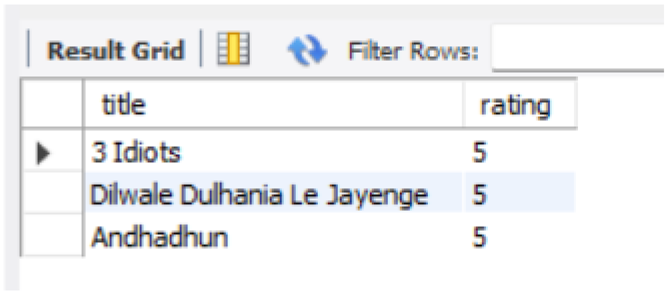
```
SELECT DISTINCT m.title

FROM movies m

JOIN ratings r ON m.movie_id = r.movie_id

WHERE r.rating > ANY (

    SELECT r_inner.rating

    FROM ratings r_inner

    JOIN movie_genres mg ON r_inner.movie_id = mg.movie_id

    JOIN genres g ON mg.genre_id = g.genre_id

    WHERE g.name = 'Drama'

);
```



```
title
3 Idiots
Dilwale Dulhania Le Jayenge
Andhadhun
```

10. **Find the movie(s) with the highest rating, better than or equal to ALL other ratings.**

```
SELECT m.title, r.rating

FROM movies m

JOIN ratings r ON m.movie_id = r.movie_id

WHERE r.rating >= ALL (

    SELECT rating FROM ratings

);
```



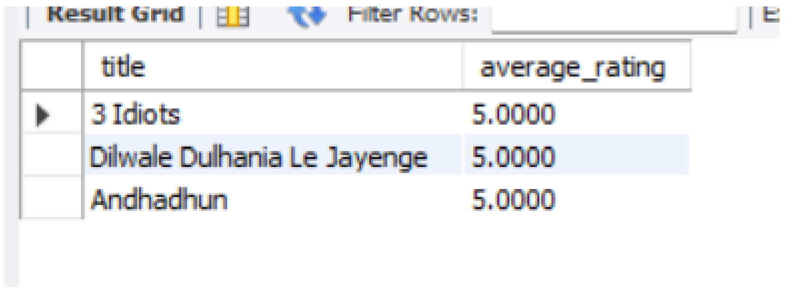
Result Grid | Filter Rows:

	title	rating
▶	3 Idiots	5
	Dilwale Dulhania Le Jayenge	5
	Andhadhun	5

c. Aggregate Functions, GROUP BY + HAVING

11. **Calculate the average rating for each movie and only show movies with an average rating of 4.5 or higher.**

```
SELECT

    m.title,

    AVG(r.rating) AS average_rating

FROM movies m

JOIN ratings r ON m.movie_id = r.movie_id

GROUP BY m.title

HAVING AVG(r.rating) >= 4.5

ORDER BY average_rating DESC;
```



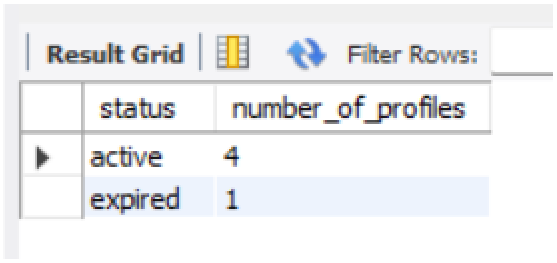
Result Grid | Filter Rows:

	title	average_rating
▶	3 Idiots	5.0000
	Dilwale Dulhania Le Jayenge	5.0000
	Andhadhun	5.0000

12. **Find the total number of profiles associated with each user subscription status ('active', 'expired', 'cancelled').**

```
SELECT

    s.status,

    COUNT(p.profile_id) AS number_of_profiles

FROM subscriptions s
```

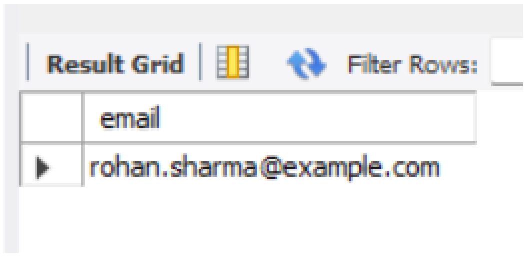
```
JOIN users u ON s.user_id = u.user_id

JOIN profiles p ON u.user_id = p.user_id

GROUP BY s.status;
```



```
status   number_of_profiles
active   4
expired  1
```

13. **Identify users who have more than one active profile and are on a 'Premium 4K' plan.**

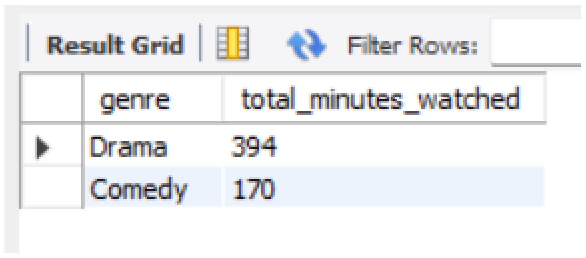
```
SELECT

    u.email

FROM users u

JOIN profiles p ON u.user_id = p.user_id

JOIN subscriptions s ON u.user_id = s.user_id

JOIN plans pl ON s.plan_id = pl.plan_id

WHERE s.status = 'active' AND pl.name = 'Standard HD'

GROUP BY u.email

HAVING COUNT(p.profile_id) > 1;
```



```
email
rohan.sharma@example.com
```

14. **Find the total watch time (in minutes) for each genre.**

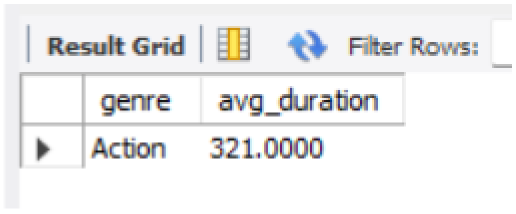
```
SELECT

    g.name AS genre,
    SUM(m.duration_minutes) AS total_minutes_watched
FROM genres g
JOIN movie_genres mg ON g.genre_id = mg.genre_id
JOIN movies m ON mg.movie_id = m.movie_id
JOIN watch_history wh ON m.movie_id = wh.movie_id
WHERE wh.completed = TRUE
GROUP BY g.name
ORDER BY total_minutes_watched DESC;
```



genre	total_minutes_watched
Drama	394
Comedy	170

15. **Find the genre with the highest average movie duration.**

```
SELECT
    g.name AS genre,
    AVG(m.duration_minutes) as avg_duration
FROM genres g
JOIN movie_genres mg ON g.genre_id = mg.genre_id
JOIN movies m ON mg.movie_id = m.movie_id
GROUP BY g.name
ORDER BY avg_duration DESC
LIMIT 1;
```

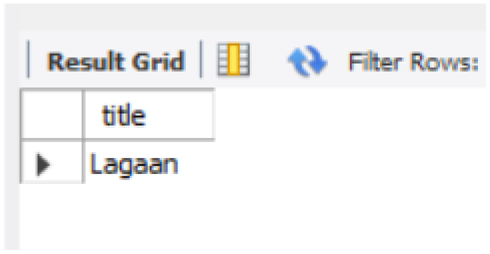


genre	avg_duration
Action	321.0000

d. Nested queries and correlated subqueries

16. **List all movies watched by profiles belonging to the user 'priya.patel@example.com'.**

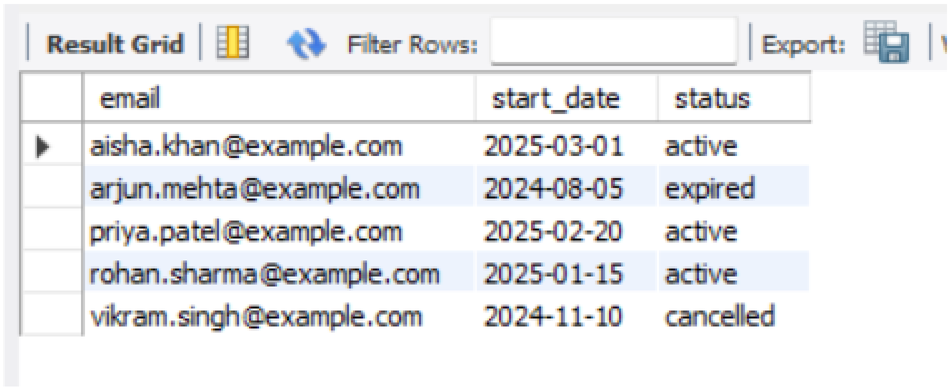
```
SELECT DISTINCT m.title

FROM movies m

WHERE m.movie_id IN (

    SELECT wh.movie_id

    FROM watch_history wh

    WHERE wh.profile_id IN (

        SELECT p.profile_id

        FROM profiles p

        WHERE p.user_id = (

            SELECT u.user_id

            FROM users u

            WHERE u.email = 'priya.patel@example.com'

        )

    )

);
```



```
title
Lagaan
```

17. **For each user, find the date of their most recent subscription.**

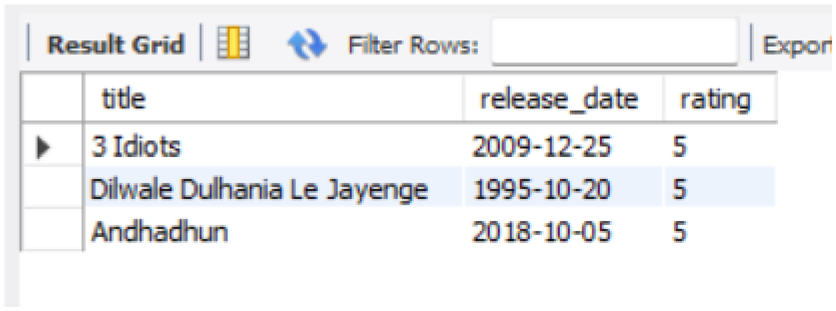
```
SELECT u.email, s1.start_date, s1.status

FROM users u

JOIN subscriptions s1 ON u.user_id = s1.user_id

WHERE s1.start_date = (

    SELECT MAX(s2.start_date)
    FROM subscriptions s2
    WHERE s2.user_id = s1.user_id
);
```



email	start_date	status
aisha.khan@example.com	2025-03-01	active
arjun.mehta@example.com	2024-08-05	expired
priya.patel@example.com	2025-02-20	active
rohan.sharma@example.com	2025-01-15	active
vikram.singh@example.com	2024-11-10	cancelled

18. **Find movies that have a rating higher than the average rating of all movies released in the same year.**

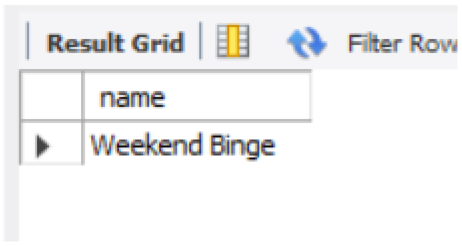
```
SELECT m1.title, m1.release_date, r.rating
FROM movies m1
JOIN ratings r ON m1.movie_id = r.movie_id
WHERE r.rating > (
    SELECT AVG(r2.rating)
    FROM ratings r2
    JOIN movies m2 ON r2.movie_id = m2.movie_id
    WHERE YEAR(m2.release_date) != YEAR(m1.release_date)
);
```



title	release_date	rating
3 Idiots	2009-12-25	5
Dilwale Dulhania Le Jayenge	1995-10-20	5
Andhadhun	2018-10-05	5

19. **List all playlists that contain at least one movie from the 'Action' genre.**

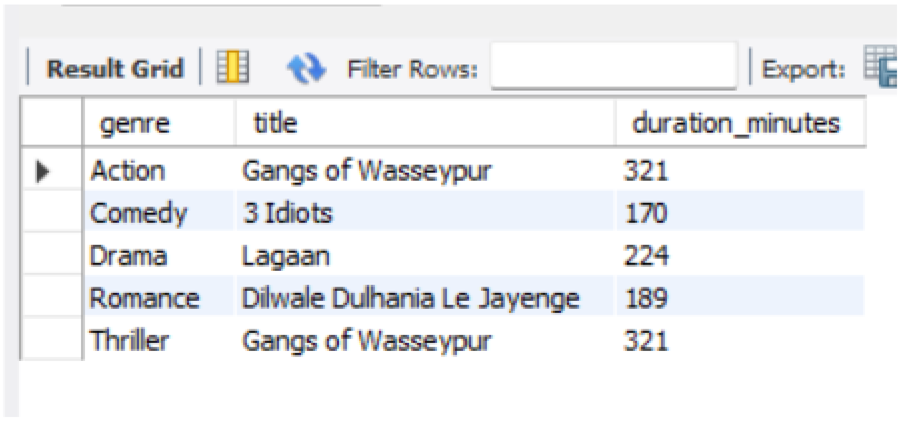
```
SELECT p.name

FROM playlists p

WHERE EXISTS (

    SELECT 1

    FROM playlist_movies pm

    JOIN movie_genres mg ON pm.movie_id = mg.movie_id

    JOIN genres g ON mg.genre_id = g.genre_id

    WHERE pm.playlist_id = p.playlist_id AND g.name = 'Action'

);
```



```
name
Weekend Binge
```

20. **For each genre, find the title of its longest movie.**

```
SELECT g.name AS genre, m.title, m.duration_minutes

FROM genres g

JOIN movie_genres mg ON g.genre_id = mg.genre_id

JOIN movies m ON mg.movie_id = m.movie_id

WHERE m.duration_minutes = (

    SELECT MAX(m2.duration_minutes)

    FROM movies m2

    JOIN movie_genres mg2 ON m2.movie_id = mg2.movie_id

    WHERE mg2.genre_id = g.genre_id

);
```



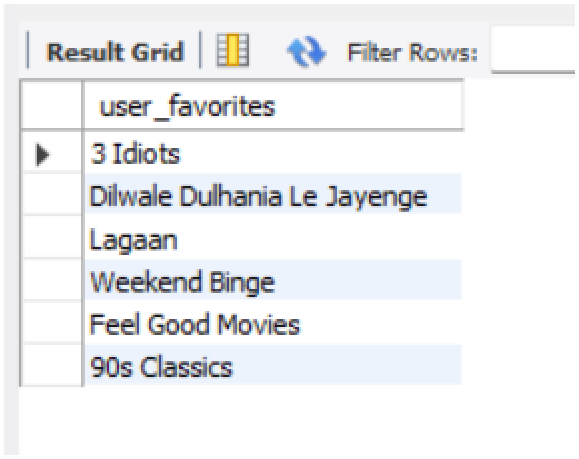
Result Grid | Filter Rows: | Export:

genre	title	duration_minutes
Action	Gangs of Wasseypur	321
Comedy	3 Idiots	170
Drama	Lagaan	224
Romance	Dilwale Dulhania Le Jayenge	189
Thriller	Gangs of Wasseypur	321

`e. Set operations (UNION, INTERSECT, EXCEPT)`

21. **Create a consolidated list of a specific user's favorite content, showing both their highly-rated movies (4 stars or more) and the playlists they created.**

```
SELECT

    m.title AS user_favorites

FROM movies m

WHERE m.movie_id IN (

    SELECT r.movie_id

    FROM ratings r

    WHERE r.rating >= 4 AND r.profile_id IN (

        SELECT p.profile_id

        FROM profiles p

        WHERE p.user_id = 1

    )

)

UNION

SELECT

    pl.name AS user_favorites

FROM playlists pl

WHERE pl.profile_id IN (

    SELECT p.profile_id
    FROM profiles p

    WHERE p.user_id = 1

);
```

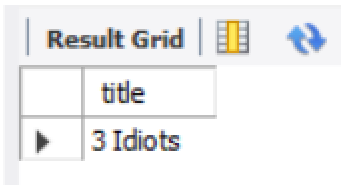


Result Grid | Filter Rows:

```
user_favorites
3 Idiots
Dilwale Dulhania Le Jayenge
Lagaan
Weekend Binge
Feel Good Movies
90s Classics
```

22. **Find movies that are classified as both 'Drama' and 'Comedy'.**

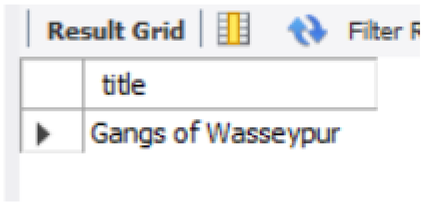
```
SELECT m.title

FROM movies m

WHERE m.movie_id IN (

    SELECT mg.movie_id

    FROM movie_genres mg

    JOIN genres g ON mg.genre_id = g.genre_id

    WHERE g.name = 'Drama'

) AND m.movie_id IN (

    SELECT mg.movie_id

    FROM movie_genres mg

    JOIN genres g ON mg.genre_id = g.genre_id

    WHERE g.name = 'Comedy'

);
```



```
title
3 Idiots
```

23. **List all 'Action' movies that have NOT been added to the 'Weekend Binge' playlist.**

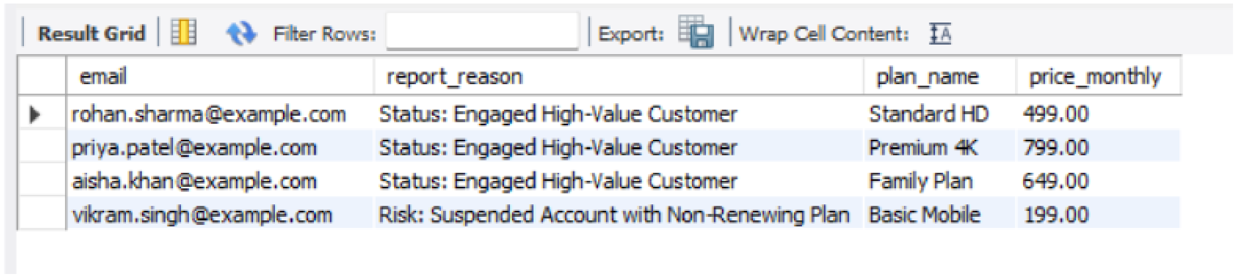
```
SELECT m.title

FROM movies m

JOIN movie_genres mg ON m.movie_id = mg.movie_id

JOIN genres g ON mg.genre_id = g.genre_id

WHERE g.name = 'Action'

AND m.movie_id NOT IN (

    SELECT pm.movie_id

    FROM playlist_movies pm

    JOIN playlists p ON pm.playlist_id = p.playlist_id

    WHERE p.name = '90s Classics'

);
```



```
title
Gangs of Wasseypur
```

24. **Generate a "Churn Risk Report" that identifies two types of at-risk users: 1) Users on expensive plans who haven't watched anything in the last 90 days, and 2) Users whose accounts are suspended but still have an active, paying subscription.**

```
SELECT

    u.email,

    'Status: Engaged High-Value Customer' AS report_reason,

    p.name AS plan_name,
    p.price_monthly
FROM users u
JOIN subscriptions s ON u.user_id = s.user_id
JOIN plans p ON s.plan_id = p.plan_id
WHERE
    p.price_monthly > 400.00
    AND s.status = 'active'
    AND u.user_id IN (
        SELECT DISTINCT p.user_id
        FROM profiles p
        JOIN watch_history wh ON p.profile_id = wh.profile_id
        WHERE wh.watched_at >= CURDATE() - INTERVAL 90 DAY
    )
UNION ALL
SELECT
    u.email,
    'Risk: Suspended Account with Non-Renewing Plan' AS report_reason,
    p.name AS plan_name,
    p.price_monthly
FROM users u
JOIN subscriptions s ON u.user_id = s.user_id
JOIN plans p ON s.plan_id = p.plan_id
WHERE
    u.status = 'suspended'
    AND s.status = 'cancelled';
```

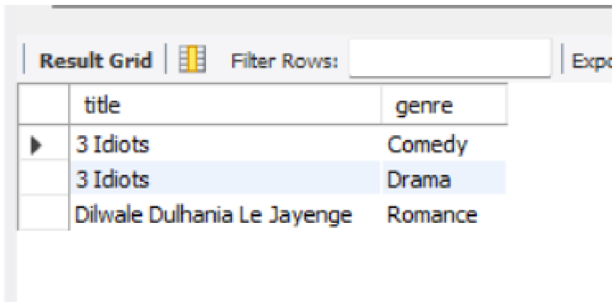

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

email	report_reason	plan_name	price_monthly
rohan.sharma@example.com	Status: Engaged High-Value Customer	Standard HD	499.00
priya.patel@example.com	Status: Engaged High-Value Customer	Premium 4K	799.00
aisha.khan@example.com	Status: Engaged High-Value Customer	Family Plan	649.00
vikram.singh@example.com	Risk: Suspended Account with Non-Renewing Plan	Basic Mobile	199.00

f. Advanced SQL

25. **Using a CTE, first find all movies with an average rating above 4.0, and then display their titles and genres.**

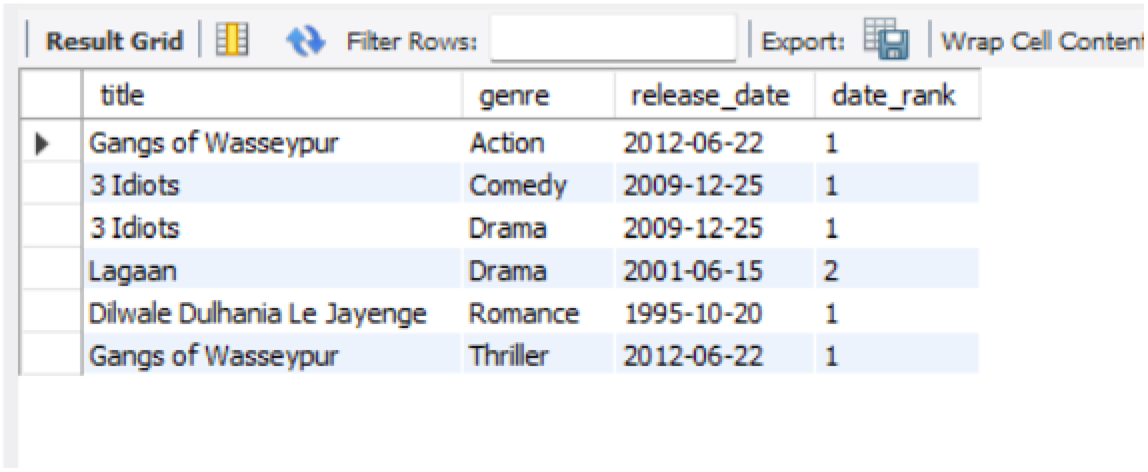
```
WITH HighlyRatedMovies AS (

    SELECT movie_id, AVG(rating) AS avg_rating

    FROM ratings

    GROUP BY movie_id

    HAVING AVG(rating) > 4.0

)

SELECT

    m.title,

    g.name AS genre

FROM movies m

JOIN HighlyRatedMovies hrm ON m.movie_id = hrm.movie_id

JOIN movie_genres mg ON m.movie_id = mg.movie_id

JOIN genres g ON mg.genre_id = g.genre_id

ORDER BY m.title;
```


Result Grid | Filter Rows: | Expo

title	genre
3 Idiots	Comedy
3 Idiots	Drama
Dilwale Dulhania Le Jayenge	Romance

26. **For each genre, rank movies by their release date (newest first).**

```
SELECT

    m.title,

    g.name AS genre,

    m.release_date,

    RANK() OVER (PARTITION BY g.name ORDER BY m.release_date DESC) as
date_rank

FROM movies m

JOIN movie_genres mg ON m.movie_id = mg.movie_id

JOIN genres g ON mg.genre_id = g.genre_id;
```

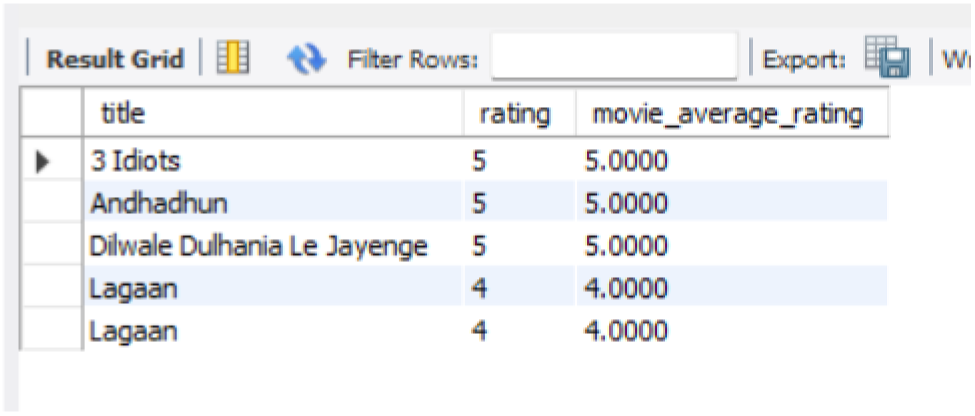


Result Grid | Filter Rows: | Export: | Wrap Cell Content

title	genre	release_date	date_rank
Gangs of Wasseypur	Action	2012-06-22	1
3 Idiots	Comedy	2009-12-25	1
3 Idiots	Drama	2009-12-25	1
Lagaan	Drama	2001-06-15	2
Dilwale Dulhania Le Jayenge	Romance	1995-10-20	1
Gangs of Wasseypur	Thriller	2012-06-22	1

27. **For each movie rating, show the rating itself and the average rating for that specific movie.**

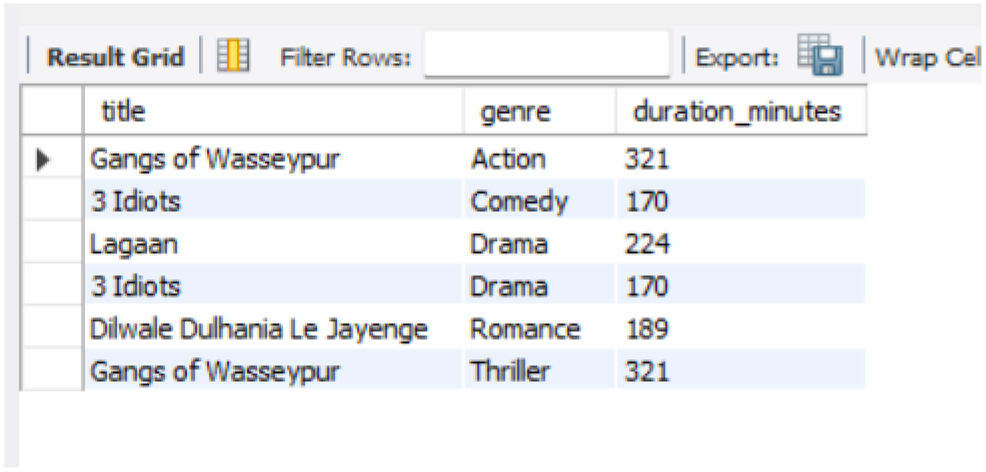
```
SELECT

    m.title,

    r.rating,

    AVG(r.rating) OVER (PARTITION BY m.title) AS movie_average_rating

FROM ratings r

JOIN movies m ON r.movie_id = m.movie_id

ORDER BY m.title;
```



title	rating	movie_average_rating
3 Idiots	5	5.0000
Andhadhun	5	5.0000
Dilwale Dulhania Le Jayenge	5	5.0000
Lagaan	4	4.0000
Lagaan	4	4.0000

28. **Find the top 3 longest movies in each genre.**

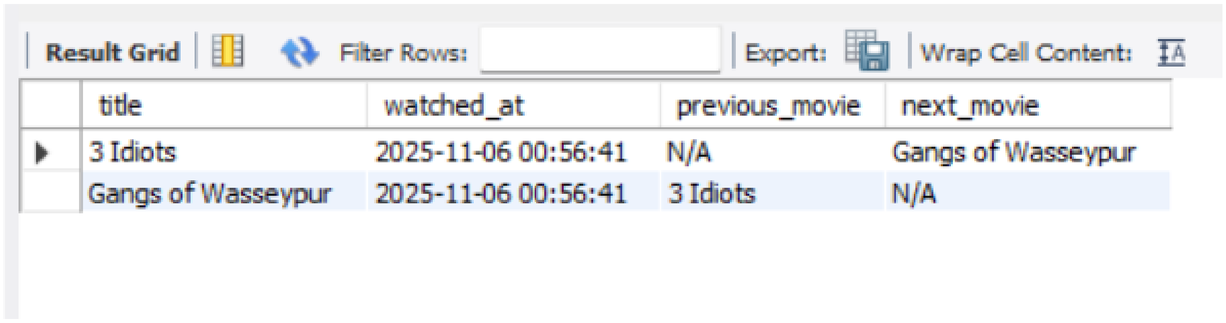
```
WITH RankedMovies AS (

    SELECT

        m.title,

        g.name AS genre,

        m.duration_minutes,

        ROW_NUMBER() OVER(PARTITION BY g.name ORDER BY m.duration_minutes
DESC) as rn

    FROM movies m

    JOIN movie_genres mg ON m.movie_id = mg.movie_id

    JOIN genres g ON mg.genre_id = g.genre_id

)

SELECT title, genre, duration_minutes

FROM RankedMovies

WHERE rn <= 3;
```


Result Grid | Filter Rows: | Export: | Wrap Cel

title	genre	duration_minutes
Gangs of Wasseypur	Action	321
3 Idiots	Comedy	170
Lagaan	Drama	224
3 Idiots	Drama	170
Dilwale Dulhania Le Jayenge	Romance	189
Gangs of Wasseypur	Thriller	321

29. **For a specific profile's watch history, show what movie was watched immediately before and after each movie.**

```
SELECT

    m.title,

    wh.watched_at,

    LAG(m.title, 1, 'N/A') OVER (ORDER BY wh.watched_at) AS previous_movie,

    LEAD(m.title, 1, 'N/A') OVER (ORDER BY wh.watched_at) AS next_movie

FROM watch_history wh

JOIN movies m ON wh.movie_id = m.movie_id

WHERE wh.profile_id = 1 -- Assuming we are checking for the profile with ID 1

ORDER BY wh.watched_at;
```


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

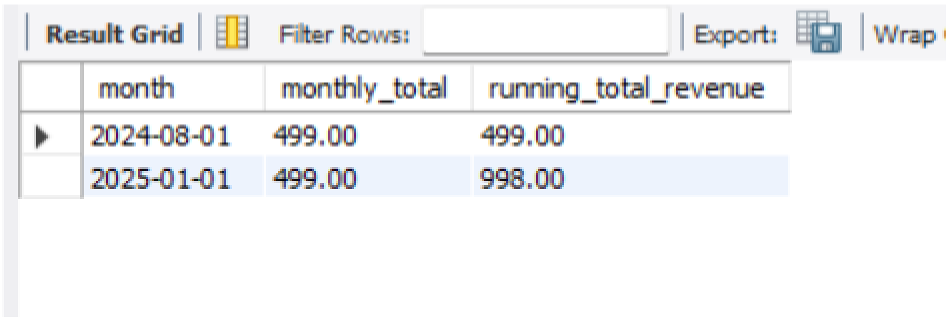
title	watched_at	previous_movie	next_movie
3 Idiots	2025-11-06 00:56:41	N/A	Gangs of Wasseypur
Gangs of Wasseypur	2025-11-06 00:56:41	3 Idiots	N/A

30. **Calculate the running total of monthly revenue from all 'Standard HD' subscriptions.**

```
WITH MonthlyRevenue AS (
```

BT23CSE218 - Yashita Bahrani

```
    SELECT
        DATE_FORMAT(start_date, '%Y-%m-01') AS month,
        SUM(p.price_monthly) as monthly_total
    FROM subscriptions s
    JOIN plans p ON s.plan_id = p.plan_id
    WHERE p.name = 'Standard HD'
    GROUP BY month
)
SELECT
    month,
    monthly_total,
    SUM(monthly_total) OVER (ORDER BY month) AS running_total_revenue
FROM MonthlyRevenue;
```



month	monthly_total	running_total_revenue
2024-08-01	499.00	499.00
2025-01-01	499.00	998.00