🎬 **Database Name: streamflix**

Tables:

1. **users**

```
CREATE TABLE users (
   user_id INT PRIMARY KEY,
   name VARCHAR(100),
   city VARCHAR(50),
   join_date DATE
);
```

2. **plans**

```
CREATE TABLE plans (
   plan_id INT PRIMARY KEY,
   plan_name VARCHAR(50),
   monthly_cost DECIMAL(6,2)
);
```

3. **subscriptions**

```
CREATE TABLE subscriptions (
   subscription_id INT PRIMARY KEY,
   user_id INT,
   plan_id INT,
   start_date DATE,
   end_date DATE,
   FOREIGN KEY (user_id) REFERENCES users(user_id),
   FOREIGN KEY (plan_id) REFERENCES plans(plan_id)
);
```

4. **movies**

```
CREATE TABLE movies (
   movie_id INT PRIMARY KEY,
   title VARCHAR(100),
   genre VARCHAR(50),
   release_year INT,
   rating DECIMAL(3,1)
);
```

5. **watch_history**

```
CREATE TABLE watch_history (
   watch_id INT PRIMARY KEY,
   user_id INT,
   movie_id INT,
   watch_date DATE,
   watch_duration INT,  -- in minutes
```

```
  FOREIGN KEY (user_id) REFERENCES users(user_id),
  FOREIGN KEY (movie_id) REFERENCES movies(movie_id)
);
```

🎯 Questions

1. Show each user's name, plan name, and monthly cost.
2. Find the total number of movies watched per genre.
3. Get the top 5 most-watched movies (by watch count).
4. Find users who have **never watched any movie**.
5. Show the average movie rating by genre.
6. Find the user who has spent the **most total time watching movies**.
7. List movies watched by **more than 3 users**.
8. Find users whose subscriptions expired before today's date.
9. Update all users on the **"Basic" plan** to the **"Standard" plan**.
10. Find which city has the highest total watch duration.

🎬 STREAMFLIX ANSWER KEY

1️⃣ Show each user's name, plan name, and monthly cost.

```
SELECT u.name AS user_name, p.plan_name, p.monthly_cost
FROM users u
JOIN subscriptions s ON u.user_id = s.user_id
JOIN plans p ON s.plan_id = p.plan_id;
```

2️⃣ Find the total number of movies watched per genre.

```
SELECT m.genre, COUNT(w.watch_id) AS total_watched
FROM watch_history w
JOIN movies m ON w.movie_id = m.movie_id
GROUP BY m.genre;
```

3️⃣ Get the top 5 most-watched movies (by watch count).

```
SELECT m.title, COUNT(w.watch_id) AS watch_count
FROM watch_history w
JOIN movies m ON w.movie_id = m.movie_id
GROUP BY m.title
ORDER BY watch_count DESC
LIMIT 5;
```

4️⃣ Find users who have never watched any movie.

```
SELECT u.name
FROM users u
LEFT JOIN watch_history w ON u.user_id = w.user_id
WHERE w.watch_id IS NULL;
```

5️⃣ Show the average movie rating by genre.

```
SELECT genre, ROUND(AVG(rating), 2) AS avg_rating
FROM movies
GROUP BY genre;
```

6️⃣ Find the user who has spent the most total time watching movies.

```
SELECT u.name, SUM(w.watch_duration) AS total_minutes
FROM watch_history w
JOIN users u ON w.user_id = u.user_id
GROUP BY u.user_id
ORDER BY total_minutes DESC
LIMIT 1;
```

7️⃣ List movies watched by more than 3 users.

```
SELECT m.title, COUNT(DISTINCT w.user_id) AS unique_viewers
FROM watch_history w
JOIN movies m ON w.movie_id = m.movie_id
GROUP BY m.movie_id
HAVING COUNT(DISTINCT w.user_id) > 3;
```

8️⃣ Find users whose subscriptions expired before today's date.

```
SELECT u.name, s.end_date
FROM subscriptions s
JOIN users u ON s.user_id = u.user_id
WHERE s.end_date < CURDATE();
```

9️⃣ Update all users on the "Basic" plan to the "Standard" plan.

```
UPDATE subscriptions
SET plan_id = (
  SELECT plan_id FROM plans WHERE plan_name = 'Standard'
)
WHERE plan_id = (
  SELECT plan_id FROM plans WHERE plan_name = 'Basic'
);
```

🔟 Find which city has the highest total watch duration.

```
SELECT u.city, SUM(w.watch_duration) AS total_watch_time
FROM users u
JOIN watch_history w ON u.user_id = w.user_id
GROUP BY u.city
ORDER BY total_watch_time DESC
LIMIT 1;
```