**Project Title: Movie Review and Recommendation Engine**

**Author:**

Dimple Mundhra  
**Tools Used:** PostgreSQL (SQL), pgAdmin / DBeaver (GUI)

**Objective:**

To build a relational database and recommendation system that analyzes user movie ratings and reviews to:

* Recommend highly-rated movies
* Understand genre-wise preferences
* Track user engagement
* Provide insightful data views

**Database Design**

The system follows a normalized structure with **4 relational tables**:

**1. Users**

Stores personal information of movie-watchers.

* user\_id: Unique ID for each user
* username: User's name
* gender: M/F
* age: Age of the user

**2. Movies**

Holds metadata about the movies.

* movie\_id: Unique movie ID
* title: Movie title
* genre: Genre (e.g., Sci-Fi, Drama)
* release\_year: Year of release

**3. Ratings**

Records the ratings given by users.

* rating\_id: Unique rating ID
* user\_id: References Users
* movie\_id: References Movies
* rating: Score from 0 to 10
* rated\_on: Date of rating

**4. Reviews**

Stores textual feedback and comments.

* review\_id: Unique review ID
* user\_id, movie\_id: References respective tables
* review: Free text comment
* review\_date: Timestamp of submission

**Key Features Implemented**

* **Data Integrity via Foreign Keys**  
  Ensures referential integrity between Users, Movies, and their related data.
* **Rating Normalization & Constraint**  
  Ratings allowed only within 0–10 range via CHECK.
* **Recommendation View**  
  Created Recommended\_Movies view for movies averaging ≥8.
* **Dynamic Movie Ranking**  
  Used DENSE\_RANK() window function for real-time movie ranking based on ratings.
* **Stored Procedure: add\_rating\_review()**  
  Automatically updates both Ratings and Reviews tables, based on optional review input.

**Analytical Reports & Views**

1. **Average Rating per Movie**  
   Displays mean rating and number of ratings each movie received.
2. **Top 3 Highest-Rated Movies**  
   Filters out movies with <3 reviews to avoid skewed results.
3. **Recommended\_Movies View**  
   A persistent SQL view suggesting movies with average rating ≥8.
4. **Ranking with DENSE\_RANK()**  
   Generates ranked list of movies by average rating (handles ties).
5. **Most Active Users**  
   Tracks user activity based on the number of ratings and reviews.
6. **Rating Distribution**  
   Shows frequency of specific ratings per movie.
7. **Collaborative Filtering Query**  
   Suggests movies liked by users with similar taste as a given user.
8. **Genre-wise Top Rated Movies**  
   Highlights highest-rated titles by genre, using aggregation.
9. **Recent Reviews**  
   Lists most recent reviews submitted by users, sorted by date.

**Export Capabilities**

* **GUI Export:** Easily export recommendation results from Recommended\_Movies view using DBeaver or pgAdmin.
* **Terminal Export (psql):** Use \copy command to export query results to CSV.
* **Python Export (Optional):** Integrate psycopg2 + pandas for automation.

**Observations**

* **Top Genres:** Sci-Fi emerged as the most frequently and highly-rated genre.
* **User Activity:** Users like Alice and Bob were the most active contributors.
* **High Scorers:** Movies like *Inception*, *Ironman*, and *The Dark Knight* consistently scored above 8.5.

**Deliverables**

* Full SQL schema creation script
* Sample data for movies, users, ratings, reviews
* Views for recommendations and top-rated content
* Stored procedure for adding ratings + reviews
* Analytical queries and reporting
* Export-ready structure for integration with external tools

**Future Enhancements**

* Integrate content-based filtering (based on genre/tags)
* Add user authentication + front-end interface (Flask/React)
* Build scheduled job to refresh views
* Implement hybrid recommender with collaborative + content-based filtering
* Use Python for automated report generation

**Conclusion**

This project demonstrates how relational databases and SQL can be leveraged to simulate a real-world movie recommendation engine. It combines well-structured schema design with analytics and procedural logic to enable meaningful recommendations.