Project Report: Movie Database Management System (MySQL)

Prepared By: Likitha Shatdarsanam

Date: 10/12/2023

---

1. Introduction

The Movie Database Management System (MDBMS) is designed to efficiently catalog and manage movie content, including movie details, actors, directors, genres, and user reviews. The system allows for easy access to information, such as movie titles, actors, directors, and reviews. Additionally, it supports queries to retrieve various types of data, such as top-rated movies, movies by genre, and average ratings.

---

2. Objectives

The objectives of the project were to:

- Design a comprehensive and relational movie database.
- Efficiently store and categorize movie-related data such as movies, actors, directors, genres, and reviews.
- Allow users to query and retrieve movie information.
- Enable users to review movies and provide ratings.
- Implement robust data management with a focus on scalability and future enhancements.

---

3. Scope of the Project

The Movie Database Management System has the following scope:

- Store and manage information for 100 movies, 50 actors, 30 directors, 10 genres, and 200 user reviews.
- Provide easy retrieval of movie information, including directors, actors, and genres.

- Store user reviews and ratings for each movie.
- Facilitate queries such as top-rated movies, movies by genre, reviews by actors, etc.

---

4. System Design

4.1. Database Schema

The system is built using a relational database model with the following tables:

1. **Movies Table**
    - Stores the basic details of movies such as movie_id, title, release_date, runtime, language, actor_id, director_id, genre_id, and rating.
2. **Actors Table**
    - Stores information about actors, including actor_id, name, and birthdate.
3. **Directors Table**
    - Stores information about directors, including director_id, name, and birthdate.
4. **Genres Table**
    - Stores movie genres with fields genre_id and genre_name.
5. **Reviews Table**
    - Stores movie reviews, including review_id, movie_id, review_text, rating, and review_date.

4.2. Entity-Relationship (ER) Model

The ER model consists of five entities: Movies, Actors, Directors, Genres, and Reviews. Each movie is associated with a director, one or more actors, a genre, and multiple reviews.

Key Relationships:

- Each movie is linked to one director and one genre.
- Each movie can have multiple actors and multiple reviews.
- The reviews entity is connected to the movies entity via a foreign key movie_id.

---

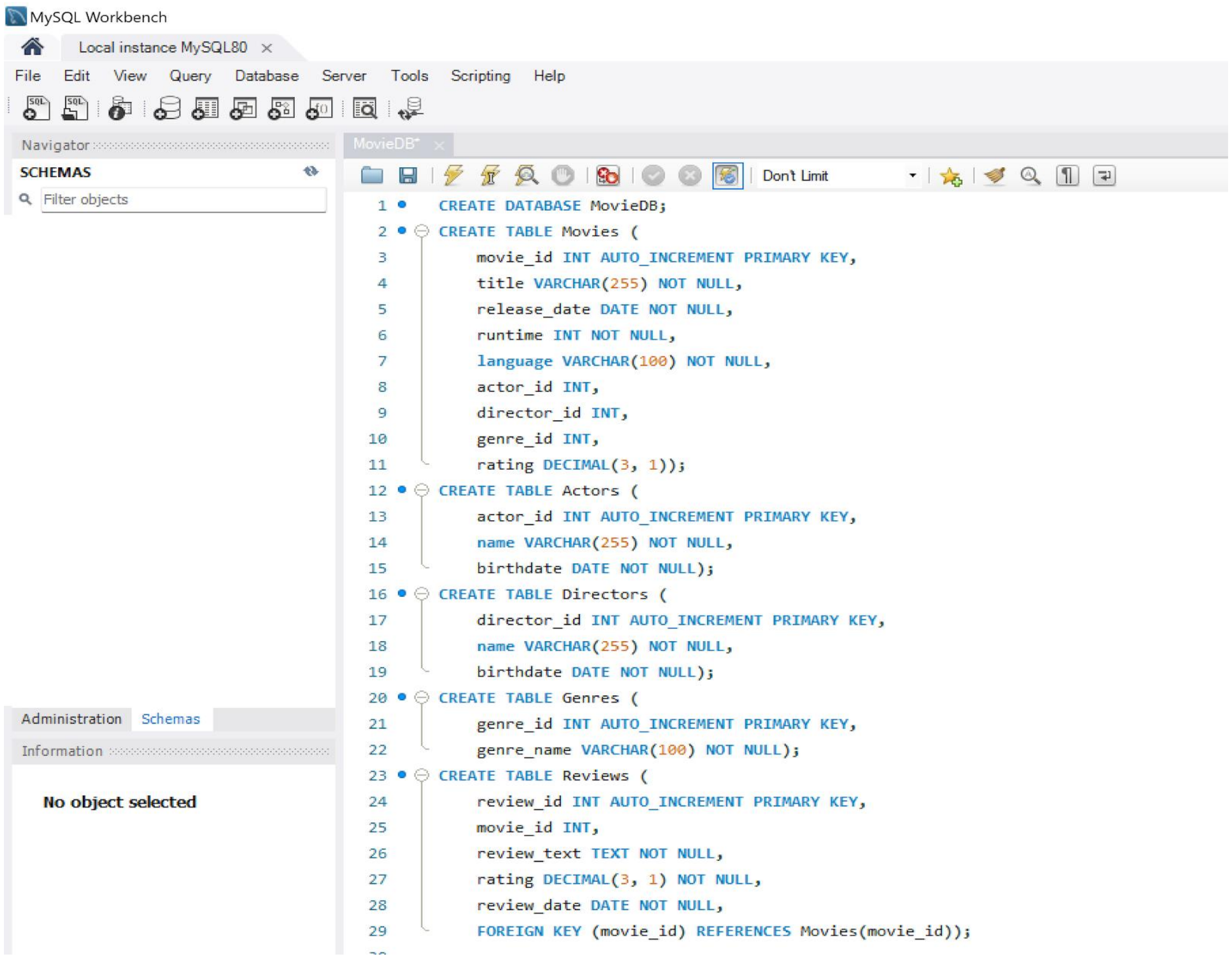
5. Implementation

5.1. Tools and Technologies

- **Database System:** MySQL
- **Development Environment:** MySQL Workbench
- **Programming Languages:** SQL for database management and queries
- **Operating System:** [Specify your OS, e.g., Windows, Linux, macOS]

5.2. Database Creation

The database schema was created using SQL commands in MySQL. Below are the SQL queries for creating the primary tables:

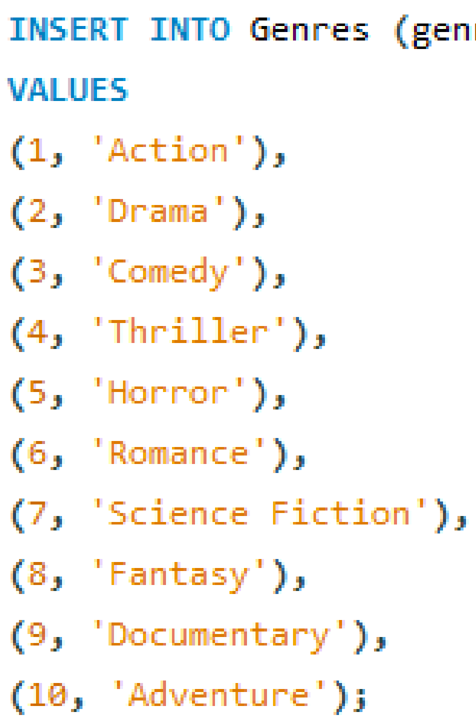


```sql
CREATE DATABASE MovieDB;
CREATE TABLE Movies (
    movie_id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    release_date DATE NOT NULL,
    runtime INT NOT NULL,
    language VARCHAR(100) NOT NULL,
    actor_id INT,
    director_id INT,
    genre_id INT,
    rating DECIMAL(3, 1));
CREATE TABLE Actors (
    actor_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    birthdate DATE NOT NULL);
CREATE TABLE Directors (
    director_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    birthdate DATE NOT NULL);
CREATE TABLE Genres (
    genre_id INT AUTO_INCREMENT PRIMARY KEY,
    genre_name VARCHAR(100) NOT NULL);
CREATE TABLE Reviews (
    review_id INT AUTO_INCREMENT PRIMARY KEY,
    movie_id INT,
    review_text TEXT NOT NULL,
    rating DECIMAL(3, 1) NOT NULL,
    review_date DATE NOT NULL,
    FOREIGN KEY (movie_id) REFERENCES Movies(movie_id));
```

5.3. Data Insertion

After creating the database schema, data was inserted into the tables. A total of 100 movies, 50 actors, 30 directors, 10 genres, and 200 reviews were added to the respective tables.

Below is an example SQL query for inserting data into the Genres table:

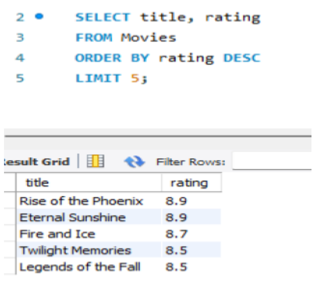


```sql
INSERT INTO Genres (genre_id, genre_name)
VALUES
(1, 'Action'),
(2, 'Drama'),
(3, 'Comedy'),
(4, 'Thriller'),
(5, 'Horror'),
(6, 'Romance'),
(7, 'Science Fiction'),
(8, 'Fantasy'),
(9, 'Documentary'),
(10, 'Adventure');
```

---

6. Example Queries

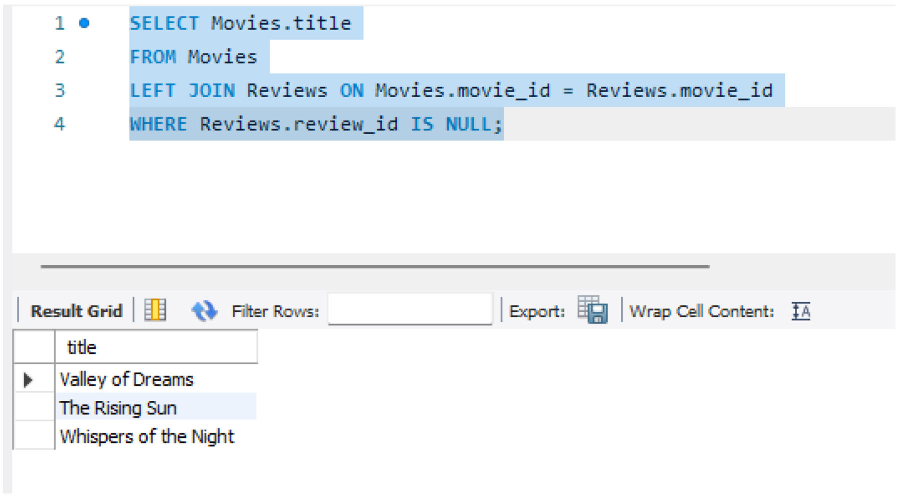
6.1. Top 5 Highest-Rated Movies



```sql
SELECT title, rating
FROM Movies
ORDER BY rating DESC
LIMIT 5;
```

title	rating
Rise of the Phoenix	8.9
Eternal Sunshine	8.9
Fire and Ice	8.7
Twilight Memories	8.5
Legends of the Fall	8.5

6.2. Find Movies with No Reviews



```sql
SELECT Movies.title
FROM Movies
LEFT JOIN Reviews ON Movies.movie_id = Reviews.movie_id
WHERE Reviews.review_id IS NULL;
```

title
Valley of Dreams
The Rising Sun
Whispers of the Night

---

7. Results

The Movie Database Management System successfully achieves the objectives:

- Efficient storage and retrieval of movie data.
- User-friendly queries to retrieve data about movies, actors, directors, and reviews.
- The ability to calculate and display top-rated movies, movies by genre, and average ratings for movies.
- The system scales to handle more data as required, and the relational design ensures easy extension for new features.

---

8. Conclusion

The Movie Database Management System was successfully implemented using MySQL and provides a comprehensive platform to manage and access movie-related data. The system allows users to query information, view reviews, and retrieve insights about top-rated movies, actors, and directors.

---

9. Future Enhancements

Future enhancements to the Movie Database Management System could include:

- **User Authentication:** Adding user accounts to track individual reviews and ratings.

- **Recommendation System:** Implementing a recommendation engine to suggest movies based on user preferences.
- **Advanced Search:** Providing advanced search features based on multiple criteria (e.g., by actor, director, genre, etc.).
- **API Integration:** Creating a RESTful API for external applications to access the movie data.

Project Completed By: Likitha Shatdarsanam
Date: 10/12/2023
Tools Used: MySQL, MySQL Workbench
Technologies: SQL, Relational Database Management System (RDBMS)
