Movie Database System

Krushna Subhash Paul
MIS: 612203101
Batch:T1 Div:2

1. Problem Statement

In today's digital era, users are constantly looking for platforms to watch movies across various genres and languages. With the increasing number of movies, streaming platforms, and diverse genres, it becomes challenging for users to find relevant information about movies and the platforms they are available on. This project aims to develop a movie database system that helps users search for movies based on multiple attributes such as genre, director, IMDb rating, language, etc. Additionally, it provides insights into streaming platforms where these movies are available.

2. Objectives

- **Create a Movie Database System**: Store information about movies, genres, directors, and streaming platforms in a structured format.
- **Provide Movie Information**: Enable users to retrieve movie details such as title, release year, certification, IMDb rating, revenue, runtime, and language.
- **Manage Genre and Director Information**: Maintain a record of genres and directors linked to the movies.
- **Streaming Platform Integration**: Allow users to find out which streaming platforms are offering specific movies.
- **Advanced Search Capabilities**: Allow users to filter movies based on various criteria, such as genre, rating, or available platforms.

3. Functional Requirements

- **Movie Details**: Users should be able to view detailed information about a movie, including its title, year of release, certification, IMDb rating, revenue, runtime, and language.
- **Director Information**: Users should be able to view the director(s) of a specific movie.
- **Genre Search**: The system should allow users to search for movies by genre.

- **Streaming Platform Search**: The system should enable users to see which streaming platforms a particular movie is available on.
- **Advanced Filtering**: Users should be able to filter movies based on their IMDb rating, release year, genre, and platforms.
- **Reports and Insights**: Provide insights such as the most popular genres, highest-grossing movies, and platforms with the largest movie catalog.

4. ER Diagram

The Entity-Relationship (ER) diagram will depict the relationship between various entities in the system such as Movies, Directors, Genres, Streaming Platforms, and their relationships.

- **Movies**: MovieID, Title, Year, Certification, IMDbRating, Revenue, Runtime, Language
- **Directors**: DirectorID, DirectorName
- **Genres**: GenreID, GenreName
- **MovieGenres**: MovieID, GenreID (Many-to-Many relation)
- **StreamingPlatforms**: PlatformID, PlatformName
- **MoviePlatforms**: MovieID, PlatformID (Many-to-Many relation)

The ER diagram will include relationships such as:

- Movies to Genres (Many-to-Many via MovieGenres)
- Movies to Directors (Many-to-One)
- Movies to Streaming Platforms (Many-to-Many via MoviePlatforms)

5. Relational Schemas Obtained from ER

```
CREATE DATABASE MovieDatabase;
USE MovieDatabase;
```

Movies Table

```
CREATE TABLE Movies (
  MovieID INT PRIMARY KEY AUTO_INCREMENT,
  Title VARCHAR(255),
  Year INT,
  Certification VARCHAR(10),
  IMDbRating DECIMAL(2, 1),
  Revenue DECIMAL(10, 2),
  Runtime INT,
```

```
  Language VARCHAR(50)
);
```

Directors Table

```
CREATE TABLE Directors (
  DirectorID INT PRIMARY KEY AUTO_INCREMENT,
  DirectorName VARCHAR(255)
);
```

Genres Table

```
CREATE TABLE Genres (
  GenreID INT PRIMARY KEY AUTO_INCREMENT,
  GenreName VARCHAR(50)
);
```

MovieGenres Table (Many-to-Many relation between Movies and Genres)

```
CREATE TABLE MovieGenres (
  MovieID INT,
  GenreID INT,
  FOREIGN KEY (MovieID) REFERENCES Movies(MovieID),
  FOREIGN KEY (GenreID) REFERENCES Genres(GenreID),
  PRIMARY KEY (MovieID, GenreID)
);
```

StreamingPlatforms Table

```
CREATE TABLE StreamingPlatforms (
  PlatformID INT PRIMARY KEY AUTO_INCREMENT,
  PlatformName VARCHAR(50)
);
```

MoviePlatforms Table (Many-to-Many relation between Movies and Streaming Platforms)

```
CREATE TABLE MoviePlatforms (
  MovieID INT,
  PlatformID INT,
  FOREIGN KEY (MovieID) REFERENCES Movies(MovieID),
  FOREIGN KEY (PlatformID) REFERENCES StreamingPlatforms(PlatformID),
  PRIMARY KEY (MovieID, PlatformID)
);
```

MovieDirectors Table (Many-to-One relation between Movies and Directors)

```
CREATE TABLE MovieDirectors (
  MovieID INT,
  DirectorID INT,
  PRIMARY KEY (MovieID, DirectorID),
  FOREIGN KEY (MovieID) REFERENCES Movies(MovieID),
  FOREIGN KEY (DirectorID) REFERENCES Directors(DirectorID)
);
```

6. Set of Functional Dependencies

- Movies(MovieID) → Title, Year, Certification, IMDbRating, Revenue, Runtime, Language
- Directors(DirectorID) → DirectorName
- Genres(GenreID) → GenreName
- MovieGenres(MovieID, GenreID) → Movies, Genres (Composite Key)
- StreamingPlatforms(PlatformID) → PlatformName
- MoviePlatforms(MovieID, PlatformID) → Movies, StreamingPlatforms (Composite Key)
- MovieDirectors(MovieID) → DirectorID (Each movie has one director, establishing a many-to-one relationship)

7. Normalization Up to 3NF

- **1NF**: All columns contain atomic values (no multi-valued attributes).
- **2NF**: Every non-key attribute is fully dependent on the primary key. For example, in the MovieGenres table, the composite key (MovieID, GenreID) uniquely identifies each record.
- **3NF**: There is no transitive dependency, ensuring that all non-key attributes depend only on the primary key.

Conclusion

In conclusion, the Movie Database System provides a comprehensive solution to manage and access movie information, including details about genres, directors, and streaming platforms. With the use of advanced filtering options and a structured database design, this system enhances users' ability to find relevant movies quickly. The normalized schema ensures efficient storage and retrieval of data, maintaining data integrity and avoiding redundancy. Overall, this project demonstrates an effective use of relational database principles to create a functional and scalable solution for movie enthusiasts and streaming platform users.

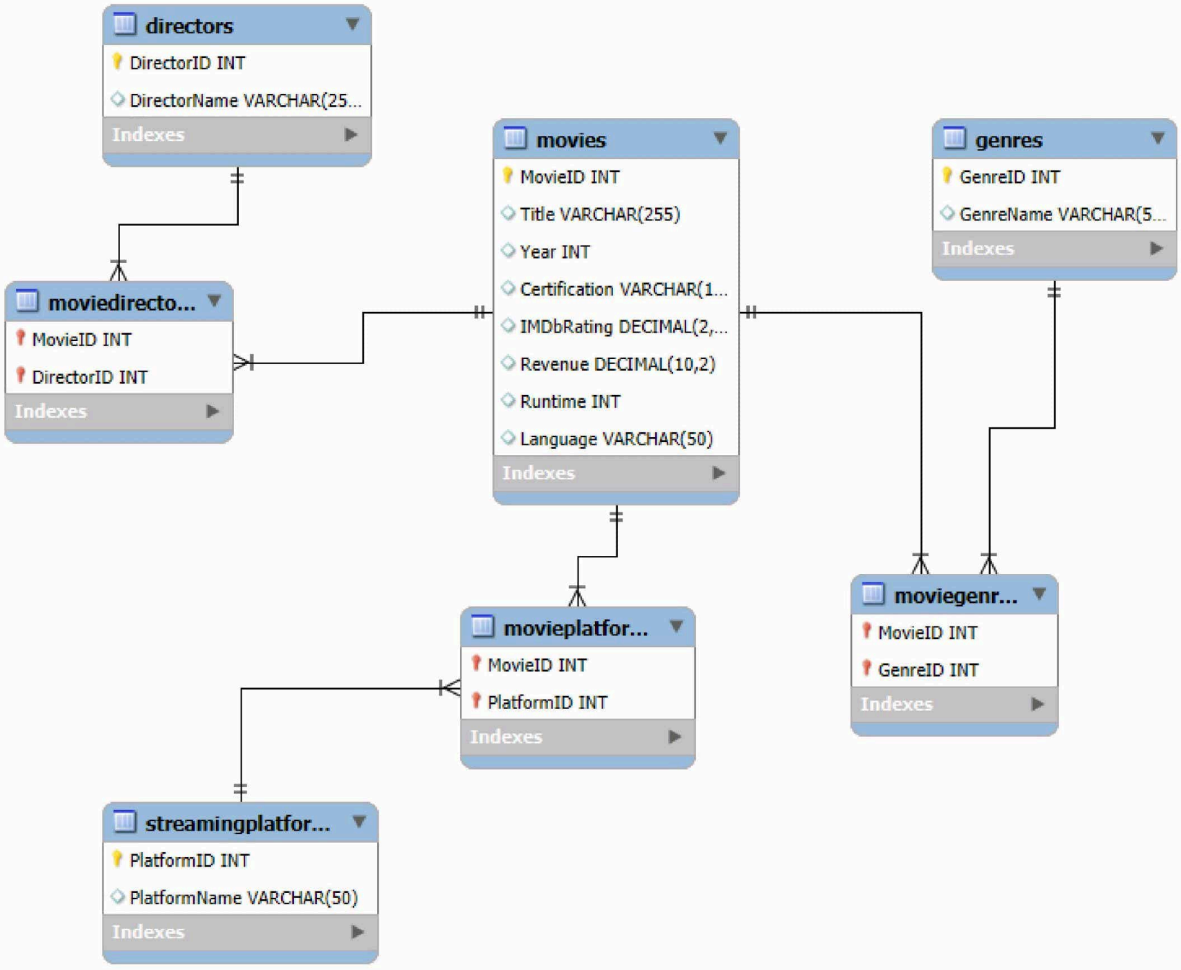
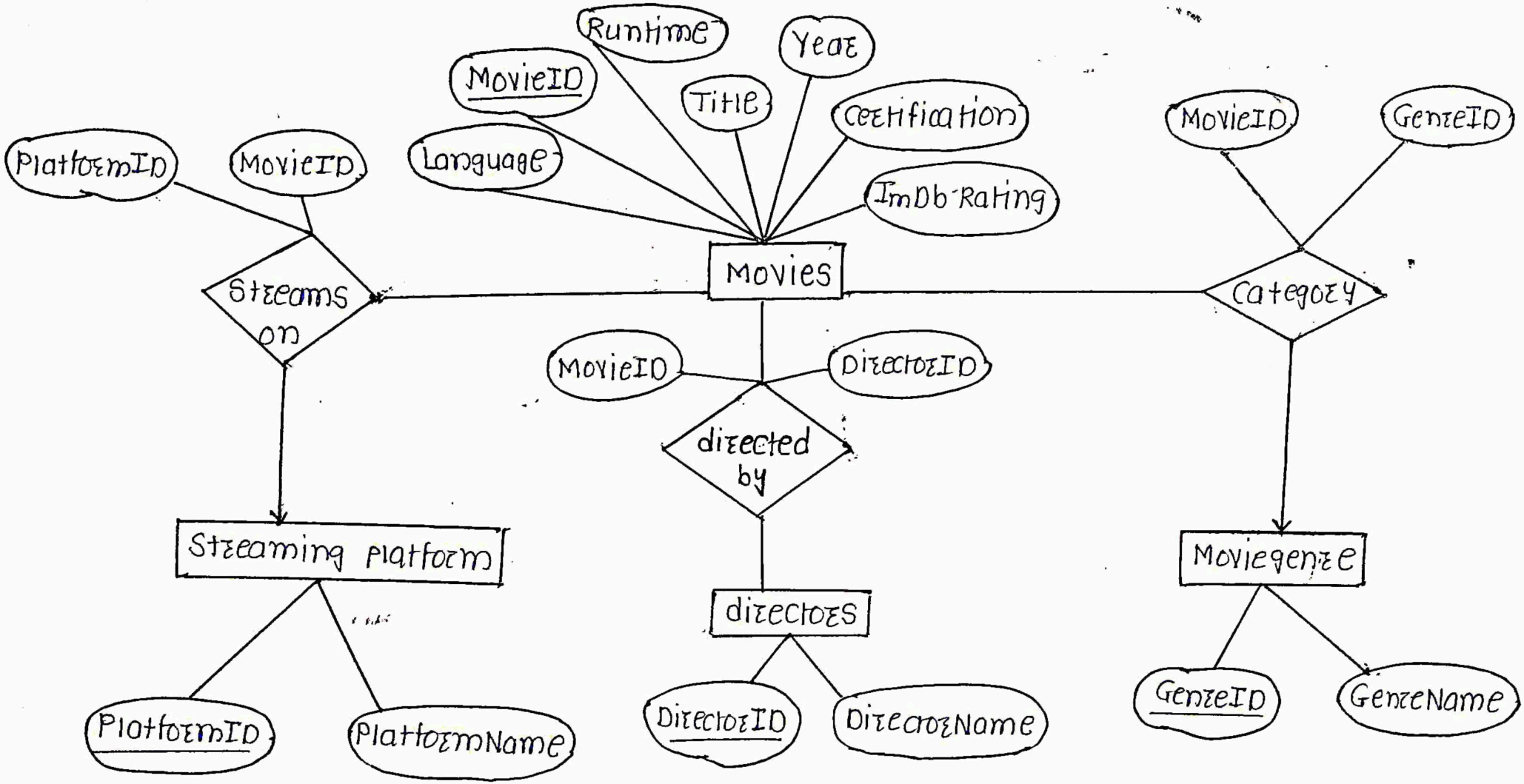


Figure 2: Relational Schema



ER Model of Movie Database