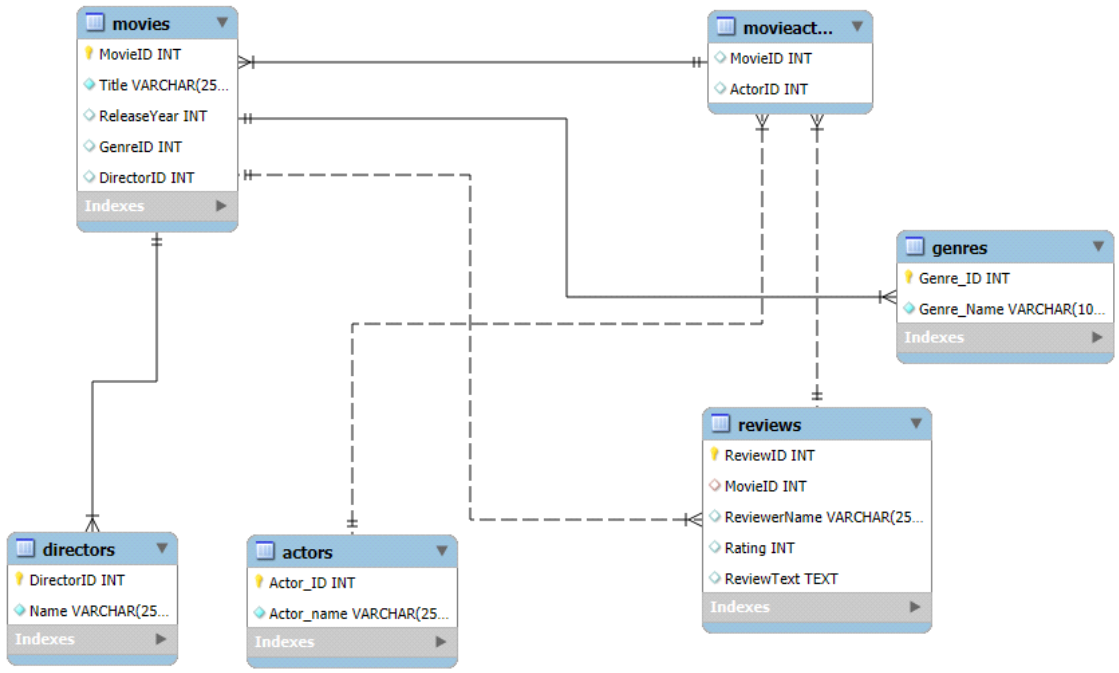
Movie Analysis

Introduction to Movie Analysis SQL Project:

The Movie Analysis SQL Project is designed to explore and analyze various aspects of movies, ranging from genres, directors, and actors to audience reviews and ratings. In an era where the film industry generates a vast amount of data, understanding and making sense of this information is crucial for movie producers, streaming platforms, and even viewers who want to discover new films to watch.

This project aims to harness the power of SQL (Structured Query Language) to build a comprehensive database that can store and analyze movie-related data. By creating a structured database, we can efficiently manage large datasets and run complex queries that can yield valuable insights.

EER Diagram:



Create Database:

```
CREATE database movies;
USE movies;
```

Create Table Genres:

```
CREATE TABLE Genres (
    Genre_ID INT PRIMARY KEY AUTO_INCREMENT,
    Genre_Name VARCHAR(100) NOT NULL
);
```

Create Table Directors:

```sql
CREATE TABLE Directors (
    DirectorID INT PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(255) NOT NULL
);
```

Create Table Movies:

```sql
CREATE TABLE Movies (
    MovieID INT PRIMARY KEY AUTO_INCREMENT,
    Title VARCHAR(255) NOT NULL,
    ReleaseYear INT,
    GenreID INT,
    DirectorID INT,
    FOREIGN KEY (GenreID) REFERENCES Genres(GenreID),
    FOREIGN KEY (DirectorID) REFERENCES Directors(DirectorID)
);
```

Create Table Actors:

```sql
CREATE TABLE Actors (
    Actor_ID INT PRIMARY KEY AUTO_INCREMENT,
    Actor_name VARCHAR(255) NOT NULL
);
```

Create Table Movie Actors:

```sql
CREATE TABLE MovieActors (
    MovieID INT,
    ActorID INT,
    PRIMARY KEY (MovieID, ActorID),
    FOREIGN KEY (MovieID) REFERENCES Movies(MovieID),
    FOREIGN KEY (ActorID) REFERENCES Actors(ActorID)
);
```

Create Table Reviews:

```sql
CREATE TABLE Reviews (
    ReviewID INT PRIMARY KEY AUTO_INCREMENT,
    MovieID INT,
    ReviewerName VARCHAR(255),
    Rating INT CHECK (Rating BETWEEN 1 AND 10),
    ReviewText TEXT,
    FOREIGN KEY (MovieID) REFERENCES Movies(MovieID)
);
```

Insert Value

```sql
INSERT INTO Genres  VALUES
(1,'Action'),
(2,'Comedy'),
(3,'Drama'),
(4,'Sci-Fi');

INSERT INTO Directors(DirectorID,Name) VALUES
(1,'Steven Spielberg'),
(2,'Christopher Nolan'),
(3,'Quentin Tarantino'),
(4,'James Cameron');

INSERT INTO Movies (MovieID,Title, ReleaseYear, GenreID, DirectorID) VALUES
(101,'Inception', 2010,1,1),
(102,'Pulp_Fiction', 1994,2,2),
(103,'Titanic', 1997, 3,3),
(104,'Jurassic_Park', 1993,4,4);
```

Case Study Questions

Question 1:

Write an SQL query to retrieve the titles and release years of all movies directed by 'Christopher Nolan'.

```sql
SELECT Title, ReleaseYear
FROM Movies
WHERE DirectorID = (SELECT DirectorID FROM Directors WHERE Name = 'Christopher Nolan');
```

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

Title	ReleaseYear
Pulp Fiction	1994

Question 2:

Write an SQL query to find the names of all actors who acted in the movie 'Inception'.

```sql
SELECT Actor_name
FROM Actors
WHERE Actor_ID IN (
    SELECT ActorID
    FROM MovieActors
    WHERE MovieID = (SELECT MovieID FROM Movies WHERE Title = 'Inception')
);
```

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

Actor_name
Leonardo DiCaprio

Question 3:

Write an SQL query to list all movies and their corresponding genres.

```sql
SELECT M.Title, G.Genre_Name
FROM Movies M
JOIN Genres G ON M.GenreID = G.Genre_ID;
```

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

Title	Genre_Name
Inception	Action
Pulp Fiction	Comedy
Titanic	Drama
Jurassic Park	Sci-Fi

Question 4:

Write an SQL query to count the number of movies released in each genre.

```sql
SELECT ReviewerName, Rating, ReviewText
FROM Reviews
WHERE MovieID = (SELECT MovieID FROM Movies WHERE Title = 'Titanic');
```

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

ReviewerName	Rating	ReviewText
Alice Johnson	7	timeless love story.

Question 5:

Write an SQL query to count the number of movies released in each genre.

```sql
SELECT G.Genre_Name, COUNT(M.MovieID) AS MovieCount
FROM Movies M
JOIN Genres G ON M.GenreID = G.Genre_ID
GROUP BY G.Genre_Name;
```

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

Genre_Name	MovieCount
Action	1
Comedy	1
Drama	1
Sci-Fi	1

Question 6:

Write an SQL query to find the name of the director who directed the movie 'Jurassic Park'.

```sql
SELECT D.Name
FROM Directors D
JOIN Movies M ON D.DirectorID = M.DirectorID
WHERE M.Title = 'Jurassic_Park';
```

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

Name

Question 7:

Write an SQL query to list all movies with their corresponding directors' names.

```sql
SELECT M.Title, D.Name AS DirectorName
FROM Movies M
JOIN Directors D ON M.DirectorID = D.DirectorID;
```

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

Title	DirectorName
Inception	Steven Spielberg
Pulp Fiction	Christopher Nolan
Titanic	Quentin Tarantino
Jurassic Park	James Cameron

Question 8:

Write an SQL query to find the total number of movies directed by 'Steven Spielberg'.

```sql
SELECT COUNT(*) AS TotalMovies
FROM Movies
WHERE DirectorID = (SELECT DirectorID FROM Directors WHERE Name = 'Steven Spielberg');
```

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

TotalMovies
1

Question 9:

Write an SQL query to list all the actors who have worked in more than one movie.

```sql
SELECT A.Actor_name, COUNT(MA.MovieID) AS MovieCount
FROM Actors A
JOIN MovieActors MA ON A.Actor_ID = MA.ActorID
GROUP BY A.Actor_name
HAVING COUNT(MA.MovieID)>1;
```

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

Actor_name	MovieCount

Question 10:

Write an SQL query to find the highest-rated movie (by rating) and its director's name.

```sql
SELECT M.Title, D.Name AS DirectorName, R.Rating
FROM Movies M
JOIN Directors D ON M.DirectorID = D.DirectorID
JOIN Reviews R ON M.MovieID = R.MovieID
ORDER BY R.Rating DESC
LIMIT 1;
```

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

Title	DirectorName	Rating
Jurassic Park	James Cameron	10