

Galaxy Studios Comprehensive SQL Database System Project

by

Gokul S – XXXXXXXX

Gokul S – XXXXXXXX

Gokul S - XXXXXXXX

Gokul S - XXXXXXXX

Gokul S - XXXXXXXX

Cleveland State University

Prof name

IST 634 | Enterprise Database

November 09, 2024

**Comprehensive SQL Database System**

**Company Selection:**

* **Company Name**: Galaxy Studios Inc
* **Industry**: Entertainment (Film Production)

**Company Overview**:

Galaxy Studios Inc is a dedicated film production company specializing in the development and production of high-quality cinematic content across various genres. With a commitment to innovation and creative storytelling, Galaxy Studios aims to produce visually compelling films that engage audiences. The studio is dedicated solely to pre-production and production activities, ensuring that each project benefits from comprehensive planning and professional execution.

**Company Operations**:

Galaxy Studios' operations encompass all aspects of film pre-production and production, focusing on creating a seamless experience from concept to completed film footage:

* **Pre-production**: During this phase, Galaxy Studios handles project development, including script refinement, casting, and recruitment of directors and essential crew members. The studio conducts a meticulous casting process to select actors for primary roles, such as male and female protagonists and antagonists, ensuring that each project has a talented and diverse cast.
* **Production**: In this stage, Galaxy Studios manages all on-set activities, including the coordination of filming, lighting, sound, and other technical elements. The company works closely with cinematographers, directors, and production crews to bring each story to life with high standards of quality and creativity. This phase also involves managing daily shoot schedules, equipment logistics, and team coordination, ensuring efficient operations on set.

**Services and Products**:

Galaxy Studios specializes in the pre-production and production of feature films across various genres, such as drama, action, and science fiction. The studio focuses on delivering polished, professional content for clients in the film industry, with projects that vary from independent films to high-budget productions.

**Data Management Needs**:

To support its operational objectives, Galaxy Studios requires a robust database system to manage data across its pre-production and production phases. Key data management needs include:

* **Talent Management**: Storing detailed information on actors, directors, cinematographers, and other crew members, including personal data, roles, and agent representation. This database supports efficient selection and booking of talent for each project.
* **Production Details and Financial Tracking**: Managing data for each film, such as production schedules, shooting locations, daily expenses, and overall budget allocation. This data helps Galaxy Studios ensure smooth project progression and effective resource management.
* **Awards and Recognition**: Tracking industry awards, nominations, and recognitions for each film. This data not only supports Galaxy Studios’ brand reputation but also highlights the contributions of key team members for future project engagements.
* **Audience Feedback and Critical Reviews**: Capturing reviews and ratings from professional critics, which inform Galaxy Studios about the success of its films and help shape the direction of future productions.
* **Operational Efficiency**: Enabling efficient management of production schedules, crew assignments, and inter-departmental communication through the database. This approach ensures that Galaxy Studios meets production deadlines, maintains high-quality standards, and optimizes resource utilization.

### Database Design

The database design for Galaxy Studios has been structured to support comprehensive data management needs in pre-production and production stages of film creation. This database system aims to ensure efficient data retrieval, reduce redundancy, and maintain data integrity through normalization up to the Third Normal Form (3NF). The following sections outline the key entities, their relationships, the Entity-Relationship (ER) diagram, and normalization techniques applied.

**Entities and Relationships**

The database includes a set of core entities that are critical to representing Galaxy Studios’ operations and personnel. Each entity reflects an essential aspect of the production process, from actors and crew members to movies and awards. Key entities include:

* **Movie**: Represents each film produced by Galaxy Studios, capturing essential information such as the title, genre, production start date, and production costs. This entity is pivotal in connecting other elements, including actors, directors, and awards.

**Attributes**: MovieID (Primary Key), Title, Genre, StartDate, ProductionCost, Duration, among others.

* **Actor: Contains detailed records of actors involved in various films, including personal information and their agency representation.**

**Attributes: ActorID (Primary Key), FirstName, LastName, BirthDate, Nationality, Gender, AgentCompany, AgentContactEmail.**

* **Director**: Holds information about directors, who are responsible for overseeing the artistic and dramatic aspects of each production.

**Attributes**: DirectorID (Primary Key), FirstName, LastName, BirthDate, Nationality, Gender, Email.

* **Producer**: Records data on producers, who manage film budgets and overall production coordination.

**Attributes**: ProducerID (Primary Key), FirstName, LastName, Company, ContactEmail.

* **Crew**: Represents technical crew members involved in production (excluding directors and actors), such as makeup artists and set designers.

**Attributes**: CrewID (Primary Key), FirstName, LastName, Role, BirthDate, Nationality, Gender, Email.

* **DOP (Director of Photography)**: Contains records of cinematographers, responsible for the visual components of each film.

**Attributes**: DOPID (Primary Key), FirstName, LastName, BirthDate, Nationality, Gender, Email.

* **Award**: Maintains a record of awards received by Galaxy Studios’ films, including award names, categories, and dates.

**Attributes**: AwardID (Primary Key), AwardName, Category, AwardDate, WinnerType.

* **Crew\_Linking**: Acts as a linking table to facilitate the many-to-many relationship between Movie and Crew entities, capturing the involvement of various crew members in multiple productions.

**Attributes**: MovieID (Foreign Key), CrewID (Foreign Key).

* **Movie\_Awards**: Establishes a relationship between Movie and Award, enabling the tracking of awards received by specific films.

**Attributes**: MovieID (Foreign Key), AwardID (Foreign Key), AwardDate.

* **Review**: Represents critical and audience reviews for each movie, including ratings and descriptive feedback.

**Attributes**: ReviewID (Primary Key), MovieID (Foreign Key), ReviewerID (Foreign Key), Rating, Description.

* **Reviewer**: Maintains records of reviewers, including their professional details and affiliations.

**Attributes**: ReviewerID (Primary Key), FirstName, LastName, Company.

**Entity-Relationship (ER) Diagram**

The ER diagram visually represents the relationships among these entities, illustrating the data flow and dependencies across the production process as shown in Fig A. Key relationships include:

* **Movie** and **Actor**: A film has specific roles assigned to actors, such as the MaleProtagonist, FemaleProtagonist, and Antagonist.
* **Movie** and **Director/Producer/DOP**: Each movie is associated with a director, producer, and cinematographer, captured through foreign keys in the Movie entity.
* **Movie** and **Crew\_Linking**: The linking table establishes the many-to-many relationship between Movie and Crew, enabling multiple crew members to be involved in a single movie and vice versa.
* **Movie\_Awards**: Connects Movie with Award, allowing a movie to receive multiple awards.
* **Review**: Connects Movie and Reviewer, providing a record of audience feedback and critical reviews for each film.

#### 

**Normalization to Third Normal Form (3NF)**

Normalization was applied to each entity to optimize data integrity and minimize redundancy:

* **First Normal Form (1NF)**: Ensures all attributes contain atomic values. For example, actor names are stored in separate FirstName and LastName fields, preventing any multi-valued attributes.
* **Second Normal Form (2NF)**: Ensures that all non-key attributes are fully functionally dependent on the primary key. For instance, attributes in the Movie table, such as Title and Genre, rely solely on MovieID.
* **Third Normal Form (3NF)**: Removes transitive dependencies so that all non-key attributes depend directly on the primary key. For example: In the Actor table, attributes related to agency representation (AgentCompany, AgentContactEmail) are stored only once, eliminating redundancy. Awards are isolated in a linking table to avoid duplicating award details in the Movie table.

By adhering to 3NF, this database design ensures efficient data retrieval and updates, avoiding redundant storage and supporting reliable data relationships essential to the production workflow at Galaxy Studios.

**SQL Implementation**

**Database Creation**

The SQL scripts below create a relational database for a movie production and review platform. The structure consists of tables representing key entities, such as movies, actors, directors, and reviews, designed based on the ER (Entity-Relationship) model. Constraints such as primary and foreign keys ensure data integrity, while the relationships between tables accurately reflect real-world associations within the industry. Below is a breakdown of each table and the specific data fields included:

**Tables**

* Movie: Stores details about each movie, including title, genre, release dates, duration, production cost, revenue, and associated crew members.
* Actor: Contains information about actors, including personal details and contact information.
* Director: Stores details about directors, with contact and background information.
* Producer: Holds details for producers associated with various films.
* Award: Represents awards won by movies or individuals, categorized by award type.
* DOP (Director of Photography): Contains details specific to the cinematographers involved in movie projects.
* Crew: Stores information about crew members and their specific roles.
* Crew\_Linking: Links movies to the respective crew members involved in their production.
* Movie\_Awards: Associates movies with the awards they have received.
* Review: Stores reviews for movies, including reviewer ratings and descriptions.
* Reviewer: Stores information about reviewers, including their affiliation and contact details.

**Example Script**

The SQL below represents a sample from the review, showcasing fields like MovieID, ReviewerID, ReviewerRating, ReviewerDescription, etc. After creating tables without foreign key constraints, additional SQL scripts using ALTER TABLE statements ensure relational integrity by linking primary keys to their corresponding foreign keys across tables, as listed below.

CREATE TABLE Review (

MovieID INT,

ReviewerID INT,

ReviewerRating varchar(50),

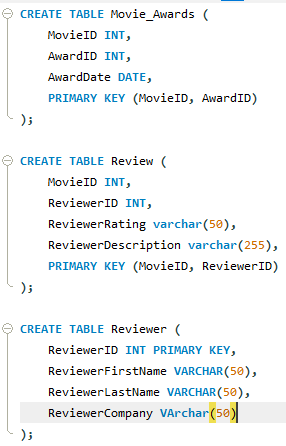
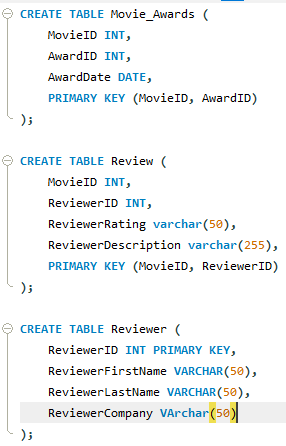
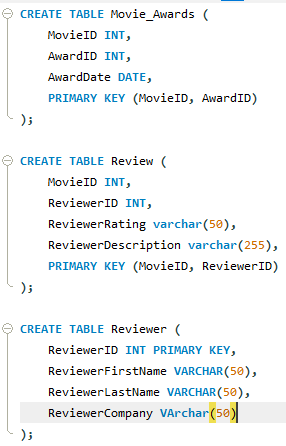
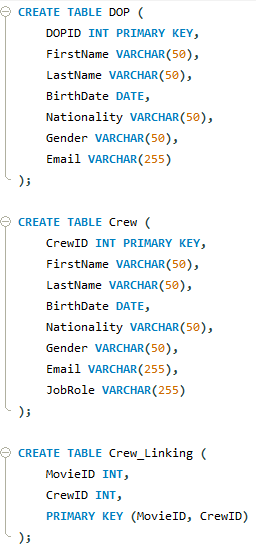
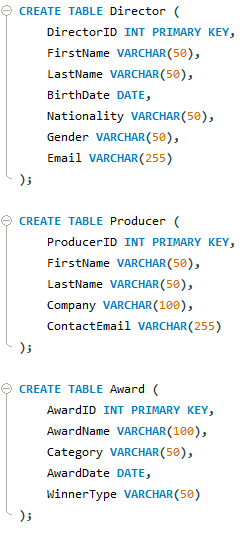
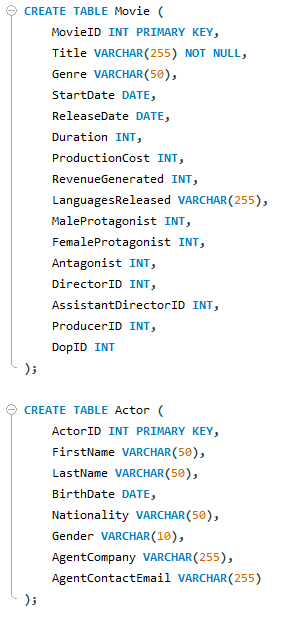
ReviewerDescription varchar(255),

PRIMARY KEY (MovieID, ReviewerID)

);

**Script:**

The SQL script for creating the database and establishing relationships is provided in Appendix A.



A screenshot of a computer program

Description automatically generated

**Data Population**

The database will be populated with realistic, sample data using AI-generated entries to represent each table accurately. This data should reflect typical movie production and review records and align with the company’s operational data. Each table will include at least 20 rows, maintaining consistency with a real-world database. All the Insert statements are listed in Appendix B.

**Example Script**

The SQL below represents a sample from the Movie and Actor tables, showcasing fields like MovieID, Title, Genre, StartDate, ReleaseDate, etc.

INSERT INTO Movie (MovieID, Title, Genre, StartDate, ReleaseDate, Duration, ProductionCost, RevenueGenerated, LanguagesReleased, MaleProtagonist, FemaleProtagonist, Antagonist, DirectorID, AssistantDirectorID, ProducerID, DopID)   
VALUES   
(1, 'Mystery of the Deep', 'Thriller', '2023-05-10', '2023-08-20', 120, 5000000, 25000000, 'English, Spanish', 1, 2, 3, 1, 2, 1, 1);

INSERT INTO Actor (ActorID, FirstName, LastName, BirthDate, Nationality, Gender, AgentCompany, AgentContactEmail)   
VALUES   
(1, 'John', 'Doe', '1980-04-22', 'American', 'Male', 'Top Talent Agency', 'johndoe@toptalent.com');

**Complex Queries:**

To support the company’s operational and analytical needs, a series of complex SQL queries were developed. These queries enable the organization to extract meaningful insights and efficiently manage various aspects of the business.

**Querry – 1** **:** **TOP 5 Revenue generated by Female antagonists**

SELECT

    m.Title AS "Movie Title",

    CONCAT(a.FirstName, ' ', a.LastName) AS "Antagonist Name",

    a.Gender AS "Antagonist Gender",

    aw.AwardName AS "Award Name",

    aw.Category AS "Award Category",

    ma.AwardDate AS "Award Date"

FROM

    Movie m

JOIN

    Actor a ON m.Antagonist = a.ActorID

JOIN

    Movie\_Awards ma ON m.MovieID = ma.MovieID

JOIN

    Award aw ON ma.AwardID = aw.AwardID

WHERE

    aw.Category LIKE '%Actor%'

    AND m.MovieID IN (SELECT MovieID FROM Movie WHERE Title IS NOT NULL)

ORDER BY

    DATE\_FORMAT(ma.AwardDate, '%Y-%m-%d');

A screenshot of a computer

Description automatically generated

Following the complex query presented here, additional queries and their outputs are provided in Appendix D. This appendix includes further examples that support the company’s analytical needs and offer deeper insights into various business aspects.

**Stored Procedures and Triggers:**

In the film production company’s database, stored procedures and triggers are used to automate routine tasks and enforce essential business rules, ensuring efficient and reliable data management.

**Stored Procedures**

1. **Update Movie Revenue**

DELIMITER //

CREATE PROCEDURE UpdateMovieRevenue(

    IN p\_MovieID INT,

    IN p\_NewRevenue INT

)

BEGIN

    UPDATE Movie

    SET RevenueGenerated = p\_NewRevenue

    WHERE MovieID = p\_MovieID;

END //

DELIMITER ;

**Example Call**

CALL UpdateMovieRevenue(1, 900000000);

1. **Retrieve Movies with High Ratings**

DELIMITER //

CREATE PROCEDURE GetHighRatedMovies(

    IN p\_MinRating DECIMAL(3,2)

)

BEGIN

    SELECT M.Title, AVG(CAST(R.ReviewerRating AS DECIMAL(3,2))) AS AvgRating

    FROM Movie M

    JOIN Review R ON M.MovieID = R.MovieID

    GROUP BY M.MovieID

    HAVING AvgRating >= p\_MinRating;

END //

DELIMITER ;

**Example Call**

CALL GetHighRatedMovies(4.5);

1. **Insert New Movie Record**

DELIMITER //

CREATE PROCEDURE InsertNewMovie(

    IN p\_Title VARCHAR(255),

    IN p\_Genre VARCHAR(50),

    IN p\_StartDate DATE,

    IN p\_ReleaseDate DATE,

    IN p\_Duration INT,

    IN p\_ProductionCost INT,

    IN p\_RevenueGenerated INT,

    IN p\_LanguagesReleased VARCHAR(255),

    IN p\_MaleProtagonist INT,

    IN p\_FemaleProtagonist INT,

    IN p\_Antagonist INT,

    IN p\_DirectorID INT,

    IN p\_AssistantDirectorID INT,

    IN p\_ProducerID INT,

    IN p\_DopID INT

)

BEGIN

    INSERT INTO Movie (Title, Genre, StartDate, ReleaseDate, Duration, ProductionCost, RevenueGenerated, LanguagesReleased, MaleProtagonist, FemaleProtagonist, Antagonist, DirectorID, AssistantDirectorID, ProducerID, DopID)

    VALUES (p\_Title, p\_Genre, p\_StartDate, p\_ReleaseDate, p\_Duration, p\_ProductionCost, p\_RevenueGenerated, p\_LanguagesReleased, p\_MaleProtagonist, p\_FemaleProtagonist, p\_Antagonist, p\_DirectorID, p\_AssistantDirectorID, p\_ProducerID, p\_DopID);

END //

DELIMITER ;

**Example Call**

CALL InsertNewMovie('Inception', 'Sci-Fi', '2009-01-01', '2010-07-16', 148, 160000000, 829895144, 'English, Japanese, French', 1, 2, 3, 4, 5, 6, 7);

1. **Insert New Actor**

DELIMITER //

CREATE PROCEDURE InsertNewActor(

    IN p\_FirstName VARCHAR(50),

    IN p\_LastName VARCHAR(50),

    IN p\_BirthDate DATE,

    IN p\_Nationality VARCHAR(50),

    IN p\_Gender VARCHAR(10),

    IN p\_AgentCompany VARCHAR(255),

    IN p\_AgentContactEmail VARCHAR(255)

)

BEGIN

    INSERT INTO Actor (FirstName, LastName, BirthDate, Nationality, Gender, AgentCompany, AgentContactEmail)

    VALUES (p\_FirstName, p\_LastName, p\_BirthDate, p\_Nationality, p\_Gender, p\_AgentCompany, p\_AgentContactEmail);

END //

DELIMITER ;

**Example Call**

CALL InsertNewActor('Leonardo', 'DiCaprio', '1974-11-11', 'American', 'Male', 'Creative Artists Agency', 'leo@caa.com');

1. **Get Total Revenue by Genre**

DELIMITER //

CREATE PROCEDURE GetTotalRevenueByGenre(

    IN p\_Genre VARCHAR(50)

)

BEGIN

    SELECT Genre, SUM(RevenueGenerated) AS TotalRevenue

    FROM Movie

    WHERE Genre = p\_Genre

    GROUP BY Genre;

END //

DELIMITER ;

**Example Call**

CALL GetTotalRevenueByGenre('Sci-Fi');

1. **Assign Award to Movie**

DELIMITER //

CREATE PROCEDURE AssignAwardToMovie(

    IN p\_MovieID INT,

    IN p\_AwardID INT,

    IN p\_AwardDate DATE

)

BEGIN

    INSERT INTO Movie\_Awards (MovieID, AwardID, AwardDate)

    VALUES (p\_MovieID, p\_AwardID, p\_AwardDate);

END //

DELIMITER ;

**Example Call**

CALL AssignAwardToMovie(1, 5, '2023-03-15');

1. **Update Actor Agent Contact**

DELIMITER //

CREATE PROCEDURE UpdateActorAgentContact(

    IN p\_ActorID INT,

    IN p\_NewAgentEmail VARCHAR(255)

)

BEGIN

    UPDATE Actor

    SET AgentContactEmail = p\_NewAgentEmail

    WHERE ActorID = p\_ActorID;

END //

DELIMITER ;

**Example Call**

CALL UpdateActorAgentContact(3, 'newagent@example.com');

1. **Retrieve All Award-Winning Movies**

DELIMITER //

CREATE PROCEDURE GetAwardWinningMovies()

BEGIN

    SELECT M.Title, A.AwardName, MA.AwardDate

    FROM Movie M

    JOIN Movie\_Awards MA ON M.MovieID = MA.MovieID

    JOIN Award A ON MA.AwardID = A.AwardID

    ORDER BY MA.AwardDate DESC;

END //

DELIMITER ;

**Example Call**

CALL GetAwardWinningMovies();

1. **Insert Review for Movie**

DELIMITER //

CREATE PROCEDURE AddMovieReview(

    IN p\_MovieID INT,

    IN p\_ReviewerID INT,

    IN p\_ReviewerRating VARCHAR(50),

    IN p\_ReviewerDescription VARCHAR(255)

)

BEGIN

    INSERT INTO Review (MovieID, ReviewerID, ReviewerRating, ReviewerDescription)

    VALUES (p\_MovieID, p\_ReviewerID, p\_ReviewerRating, p\_ReviewerDescription);

END //

DELIMITER ;

**Example Call**

CALL AddMovieReview(1, 2, '4.5', 'Amazing movie with great visuals and story!');

1. **List Crew Members by Job Role**

DELIMITER //

CREATE PROCEDURE GetCrewByRole(

    IN p\_JobRole VARCHAR(255)

)

BEGIN

    SELECT CrewID, FirstName, LastName, JobRole

    FROM Crew

    WHERE JobRole = p\_JobRole;

END //

DELIMITER ;

**Example Call**

CALL GetCrewByRole('Director');

**Triggers**

1. **Trigger to Prevent Negative Revenue on Movie Insert or Update**

DELIMITER //

CREATE TRIGGER PreventNegativeRevenue

BEFORE INSERT ON Movie

FOR EACH ROW

BEGIN

    IF NEW.RevenueGenerated < 0 THEN

        SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'RevenueGenerated cannot be negative.';

    END IF;

END //

DELIMITER ;

1. **Trigger to Automatically Set Award Date to Today if Not Provided**

DELIMITER //

CREATE TRIGGER SetDefaultAwardDate

BEFORE INSERT ON Movie\_Awards

FOR EACH ROW

BEGIN

    IF NEW.AwardDate IS NULL THEN

        SET NEW.AwardDate = CURDATE();

    END IF;

END //

DELIMITER ;

1. **Trigger to Prevent Duplicate Awards for the Same Movie on the Same Date**

DELIMITER //

CREATE TRIGGER PreventDuplicateAwards

BEFORE INSERT ON Movie\_Awards

FOR EACH ROW

BEGIN

    DECLARE duplicate\_count INT;

    SELECT COUNT(\*) INTO duplicate\_count

    FROM Movie\_Awards

    WHERE MovieID = NEW.MovieID AND AwardID = NEW.AwardID AND AwardDate = NEW.AwardDate;

    IF duplicate\_count > 0 THEN

        SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'This award has already been assigned to the movie on this date.';

    END IF;

END //

DELIMITER ;

1. **Trigger to Automatically Insert New Review for Movies with No Review**

DELIMITER //

CREATE TRIGGER InsertDefaultReview

AFTER INSERT ON Movie

FOR EACH ROW

BEGIN

    DECLARE review\_count INT;

    SELECT COUNT(\*) INTO review\_count

    FROM Review

    WHERE MovieID = NEW.MovieID;

    IF review\_count = 0 THEN

        INSERT INTO Review (MovieID, ReviewerID, ReviewerRating, ReviewerDescription)

        VALUES (NEW.MovieID, 1, '0', 'Default review - no user reviews yet.');

    END IF;

END //

DELIMITER ;

1. **Trigger to Automatically Update LastModified Date on Movie Update**

ALTER TABLE Movie ADD COLUMN LastModified TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP;

DELIMITER //

CREATE TRIGGER UpdateMovieLastModified

BEFORE UPDATE ON Movie

FOR EACH ROW

BEGIN

    SET NEW.LastModified = NOW();

END //

DELIMITER ;

**Database Optimization**

1. **Indexing**

**A screenshot of a computer program

Description automatically generated**Indexing is a critical optimization technique in relational databases, particularly for enhancing query performance on frequently accessed columns. In this schema, indexes have been applied to select columns based on anticipated query frequency and expected data retrieval patterns. The SQL Script for Indexing is listing in Appendix C

* **Genre (Movie Table)**:

CREATE INDEX idx\_movie\_genre ON Movie (Genre);

Indexing the Genre column improves performance in queries that filter movies by genre, commonly required in genre-based reports or recommendations.

* **Release Date (Movie Table)**:

CREATE INDEX idx\_movie\_releasedate ON Movie (ReleaseDate);

This index facilitates efficient sorting and filtering on ReleaseDate, optimizing queries that organize movies chronologically or retrieve films from specific time periods.

* **Revenue Generated (Movie Table)**:

CREATE INDEX idx\_movie\_revenue ON Movie (RevenueGenerated);

This index enhances the speed of queries calculating top-grossing movies or generating revenue-based reports, a frequent requirement in revenue analytics.

* **Male Protagonist, Female Protagonist, and Antagonist (Movie Table)**:

CREATE INDEX idx\_movie\_maleprotagonist ON Movie (MaleProtagonist);  
CREATE INDEX idx\_movie\_femaleprotagonist ON Movie (FemaleProtagonist);  
CREATE INDEX idx\_movie\_antagonist ON Movie (Antagonist);

Indexing these foreign keys to the Actor table optimizes joins and filters in queries that retrieve movies based on specific actor roles, often needed for casting analytics or character studies.

* **Nationality (Actor Table)**:

CREATE INDEX idx\_actor\_nationality ON Actor (Nationality);

This index aids in filtering actors by nationality, which is beneficial for demographic-based analyses or regulatory compliance checks when working with actors from different countries.

* **Award Category (Award Table)**:

CREATE INDEX idx\_award\_category ON Award (Category);

The Category index expedites filtering and aggregation by award type, commonly required in award analytics to evaluate movie performance in specific categories (e.g., "Best Actor").

* **Job Role (Crew Table)**:

CREATE INDEX idx\_crew\_jobrole ON Crew (JobRole);

Indexing JobRole facilitates quick retrieval of crew members by their specific roles, such as cinematographers or set designers, enabling efficient crew management.

* **Movie and Crew IDs (Crew\_Linking Table)**:

CREATE INDEX idx\_crewlink\_movieid\_crewid ON Crew\_Linking (MovieID, CrewID);

This composite index optimizes joins and lookups between movies and crew members, especially for queries analyzing crew composition per movie.

* **Movie and Award IDs (Movie\_Awards Table)**:

CREATE INDEX idx\_movieawards\_movieid\_awardid ON Movie\_Awards (MovieID, AwardID);

This composite index improves the efficiency of award-based queries by allowing quick access to specific movies and their associated awards.

* **Award Date (Movie\_Awards Table)**:

CREATE INDEX idx\_movieawards\_awarddate ON Movie\_Awards (AwardDate);

Indexing AwardDate supports chronological queries and historical analyses of awards, commonly used to generate timelines or trend reports.

* **Movie and Reviewer IDs (Review Table)**:

CREATE INDEX idx\_review\_movieid\_reviewerid ON Review (MovieID, ReviewerID);

This composite index optimizes retrieval of reviews based on the combination of movies and reviewers, facilitating quick access to reviews by specific critics for particular movies.

* **Reviewer Rating (Review Table)**:

CREATE INDEX idx\_review\_rating ON Review (ReviewerRating);

Indexing ReviewerRating supports performance in queries filtering or sorting by rating, useful for quality assessments and sentiment analysis.

Each of these indexes was selected based on anticipated query patterns and data access needs, aiming to reduce response times and enhance overall system efficiency. Proper indexing reduces the amount of data scanned in common queries, lowering computational costs and making the database more responsive to user interactions.

**Query Optimization**

The unoptimized query includes several inefficiencies, which can be addressed to improve performance. Using the EXPLAIN command, we analyzed the query's execution plan, revealing potential bottlenecks and suboptimal operations, as seen in the screenshots provided.

**Unoptimized Query**

-- un-optimized

EXPLAIN

SELECT

m.Title AS "Movie Title",

CONCAT(a.FirstName, ' ', a.LastName) AS "Antagonist Name",

a.Gender AS "Antagonist Gender",

aw.AwardName AS "Award Name",

aw.Category AS "Award Category",

ma.AwardDate AS "Award Date"

FROM

Movie m

JOIN

Actor a ON m.Antagonist = a.ActorID

JOIN

Movie\_Awards ma ON m.MovieID = ma.MovieID

JOIN

Award aw ON ma.AwardID = aw.AwardID

WHERE

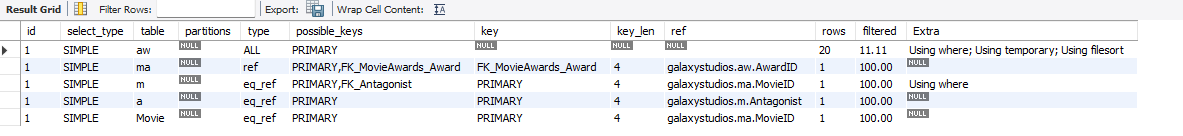
aw.Category LIKE '%Actor%' index use

AND m.MovieID IN (SELECT MovieID FROM Movie WHERE Title IS NOT NULL)

ORDER BY

DATE\_FORMAT(ma.AwardDate, '%Y-%m-%d');

**Explain Command Output for unoptimized query**



**Unoptimized Query Analysis**

1. **Broad Condition in WHERE Clause**:

The aw.Category LIKE '%Actor%' condition causes a full scan on the Award table because the wildcard at the beginning prevents the use of any index on Category. This can lead to increased execution time, especially if the Award table is large.

1. **Unnecessary Subquery**:

The m.MovieID IN (SELECT MovieID FROM Movie WHERE Title IS NOT NULL) condition is redundant because Title is a NOT NULL column. This subquery forces additional scanning without any meaningful filtering and impacts performance by adding complexity to the execution plan.

1. **Use of Function in ORDER BY**:

Applying DATE\_FORMAT(ma.AwardDate, '%Y-%m-%d') in the ORDER BY clause prevents MySQL from using any index on AwardDate for sorting, leading to an additional sorting step (Using filesort), which consumes extra resources.

**Optimized Query**

The optimized version of the query eliminates unnecessary operations and enhances index usage for better efficiency.

-- Optimized

EXPLAIN

SELECT

m.Title AS "Movie Title",

CONCAT(a.FirstName, ' ', a.LastName) AS "Antagonist Name",

a.Gender AS "Antagonist Gender",

aw.AwardName AS "Award Name",

aw.Category AS "Award Category",

ma.AwardDate AS "Award Date"

FROM

Movie m

JOIN

Actor a ON m.Antagonist = a.ActorID

JOIN

Movie\_Awards ma ON m.MovieID = ma.MovieID

JOIN

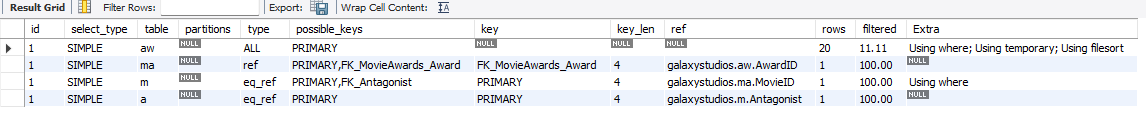
Award aw ON ma.AwardID = aw.AwardID

WHERE

aw.Category LIKE '%Actor%'

ORDER BY

ma.AwardDate DESC;

**Explain Command Output for optimized query**

**Optimization Techniques Applied**

1. **Removed the Unnecessary Subquery**:

The subquery (SELECT MovieID FROM Movie WHERE Title IS NOT NULL) was removed, as it did not filter the results in a meaningful way. This reduces redundant operations, lowering execution costs.

1. **Optimized ORDER BY Clause**:

Sorting is now based on ma.AwardDate DESC without applying a function, allowing MySQL to use an index on AwardDate (if one exists) for ordering. This avoids the need for a filesort operation, which is resource intensive.

1. **Evaluated Indexing for LIKE Clause**:

While LIKE '%Actor%' is still present, fully optimizing it would require re-evaluating how data is stored in Category and possibly adjusting the query structure. In the current context, this condition remains, but future improvements could involve using a different approach for partial matches.

**Performance Improvement Summary**

The optimized query minimizes redundant operations, potentially decreases the number of rows scanned, and eliminates resource-heavy sorting by filesort. Removing the function in the ORDER BY clause and the unnecessary subquery simplifies the execution plan, making the query more efficient. This streamlined approach results in faster execution and lower system resource usage, particularly beneficial for large datasets.

**Data Security and Backup**

**User Roles and Permissions**

The roles of the users will be provided to grant different types of permissions to the users as per the function they play in the company. This will ensure proper control of data security and access. Application of user role reduces unauthorized access and theft of sensitive information. Following are the recommended roles and their respective permissions:

1. **Admin**:

**Role Description**: Has full control over the database, including the ability to create, read, update, and delete (CRUD) data, manage user roles, and configure security settings.

**Permissions**:

* + Full access to all tables, views, and procedures.
  + Ability to create, alter, and drop database structures (e.g., tables, indexes).
  + Access to audit logs and backup/restore functionality.
  + Management of user roles and permissions.

1. **Sales Manager**:

**Role Description**: Focused on managing sales-related data, with read and limited write access to sales information, while being restricted from accessing other sensitive data.

**Permissions**:

* + Read-only access to Movie, Revenue, Awards, and Review tables.
  + Write access to update or insert records related to Review for sales tracking and customer feedback.
  + No access to modify or delete core data (e.g., actors, production costs).

1. **Inventory Manager**:

**Role Description**: Manages assets and inventory associated with movie production, with permissions to update relevant data but restricted from accessing financial and sales data.

**Permissions**:

* + Read and write access to tables associated with Movie, Crew, and Actor for tracking asset availability.
  + No access to financial data, awards, or sensitive information like salaries or revenue.

1. **Reviewer**:

**Role Description**: External or third-party reviewers with limited access to submit or update review data but restricted from viewing or modifying any other data.

**Permissions**:

* + Write-only access to the Review table for submitting ratings and descriptions.
  + No access to any other tables or sensitive company data.

1. **Read-Only Analyst**:

**Role Description**: Data analyst with read-only access to data across the database for reporting and analytics, without any ability to modify data.

**Permissions**:

* + Read-only access to all tables, enabling data extraction and analysis.
  + No write, update, or delete permissions to preserve data integrity.

These roles put in place ensure that access to data is controlled, based on the function of each user, hence minimizing unauthorized modification or exposure of data.

**Backup Strategy**

A complete backup and restore strategy is highly important in order not to lose crucial data of the company, corrupt it, or face system failure. Such a strategy is composed of the following parts:

1. **Backup Types**:

* **Full Backup**: A complete copy of the entire database taken weekly. This ensures that a recent version of the entire dataset is available for a full restoration.
* **Differential Backup**: Taken daily, this includes only the data modified since the last full backup. Differential backups are faster to execute than full backups and reduce the storage requirement while maintaining up-to-date backups.
* **Transaction Log Backup**: Taken every hour to capture ongoing changes within the database, enabling point-in-time recovery in case of data loss or corruption.

1. **Storage of Backups**:

* **On-Site Storage**: Backups are stored locally on encrypted storage devices with restricted access to administrators only. This facilitates quick recovery in case of minor failures or data loss.
* **Off-Site Storage**: Copies of weekly full backups and daily differential backups are stored in a secure, off-site location or a cloud storage solution with strong encryption protocols. This mitigates the risk of data loss due to on-site disasters (e.g., fire or flood).
* **Retention Policy**: Maintain daily backups for a week, weekly full backups for a month, and monthly backups for a year to balance data availability and storage costs.

1. **Testing and Validation**:

* **Regular Backup Testing**: Perform monthly restore tests to ensure backups are functional and can be used to restore data accurately. This involves periodically restoring backups in a staging environment and validating data integrity.
* **Automated Alerts**: Set up automated notifications to report backup completion, failures, or storage issues, allowing the team to respond promptly to any issues that might compromise data security.

1. **Disaster Recovery Plan**:

* **Restore Procedure**: Document a step-by-step restore procedure for each backup type (full, differential, transaction log) to ensure a quick recovery process during a failure. The procedure should be accessible to the admin team but secured to prevent unauthorized access.
* **Recovery Time Objectives (RTO) and Recovery Point Objectives (RPO)**:
  + **RTO**: Set a target of 4 hours for critical systems, ensuring that essential operations can resume swiftly after data loss.
  + **RPO**: Limit data loss to a maximum of 1 hour through frequent transaction log backups, minimizing disruption.

By defining user roles and permissions and implementing a robust backup strategy, the company can secure its data against unauthorized access and ensure data recoverability in case of unexpected data loss. This plan will help maintain business continuity and uphold data integrity, essential for operational stability and regulatory compliance.

**APPENDIX A**

CREATE TABLE Movie (

MovieID INT PRIMARY KEY,

Title VARCHAR(255) NOT NULL,

Genre VARCHAR(50),

StartDate DATE,

ReleaseDate DATE,

Duration INT,

ProductionCost INT,

RevenueGenerated INT,

LanguagesReleased VARCHAR(255),

MaleProtagonist INT,

FemaleProtagonist INT,

Antagonist INT,

DirectorID INT,

AssistantDirectorID INT,

ProducerID INT,

DopID INT

);

CREATE TABLE Actor (

ActorID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

BirthDate DATE,

Nationality VARCHAR(50),

Gender VARCHAR(10),

AgentCompany VARCHAR(255),

AgentContactEmail VARCHAR(255)

);

CREATE TABLE Director (

DirectorID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

BirthDate DATE,

Nationality VARCHAR(50),

Gender VARCHAR(50),

Email VARCHAR(255)

);

CREATE TABLE Producer (

ProducerID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Company VARCHAR(100),

ContactEmail VARCHAR(255)

);

CREATE TABLE Award (

AwardID INT PRIMARY KEY,

AwardName VARCHAR(100),

Category VARCHAR(50),

AwardDate DATE,

WinnerType VARCHAR(50)

);

CREATE TABLE DOP (

DOPID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

BirthDate DATE,

Nationality VARCHAR(50),

Gender VARCHAR(50),

Email VARCHAR(255)

);

CREATE TABLE Crew (

CrewID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

BirthDate DATE,

Nationality VARCHAR(50),

Gender VARCHAR(50),

Email VARCHAR(255),

JobRole VARCHAR(255)

);

CREATE TABLE Crew\_Linking (

MovieID INT,

CrewID INT,

PRIMARY KEY (MovieID, CrewID)

);

CREATE TABLE Movie\_Awards (

MovieID INT,

AwardID INT,

AwardDate DATE,

PRIMARY KEY (MovieID, AwardID)

);

CREATE TABLE Review (

MovieID INT,

ReviewerID INT,

ReviewerRating varchar(50),

ReviewerDescription varchar(255),

PRIMARY KEY (MovieID, ReviewerID)

);

CREATE TABLE Reviewer (

ReviewerID INT PRIMARY KEY,

ReviewerFirstName VARCHAR(50),

ReviewerLastName VARCHAR(50),

ReviewerCompany VArchar(50)

);

ALTER TABLE Movie

ADD CONSTRAINT FK\_MaleProtagonist FOREIGN KEY (MaleProtagonist) REFERENCES Actor(ActorID),

ADD CONSTRAINT FK\_FemaleProtagonist FOREIGN KEY (FemaleProtagonist) REFERENCES Actor(ActorID),

ADD CONSTRAINT FK\_Antagonist FOREIGN KEY (Antagonist) REFERENCES Actor(ActorID),

ADD CONSTRAINT FK\_Director FOREIGN KEY (DirectorID) REFERENCES Director(DirectorID),

ADD CONSTRAINT FK\_AssistantDirector FOREIGN KEY (AssistantDirectorID) REFERENCES Director(DirectorID),

ADD CONSTRAINT FK\_Producer FOREIGN KEY (ProducerID) REFERENCES Producer(ProducerID),

ADD CONSTRAINT FK\_Dop FOREIGN KEY (DopID) REFERENCES DOP(DOPID);

ALTER TABLE Crew\_Linking

ADD CONSTRAINT FK\_CrewLinking\_Movie FOREIGN KEY (MovieID) REFERENCES Movie(MovieID),

ADD CONSTRAINT FK\_CrewLinking\_Crew FOREIGN KEY (CrewID) REFERENCES Crew(CrewID);

ALTER TABLE Movie\_Awards

ADD CONSTRAINT FK\_MovieAwards\_Movie FOREIGN KEY (MovieID) REFERENCES Movie(MovieID),

ADD CONSTRAINT FK\_MovieAwards\_Award FOREIGN KEY (AwardID) REFERENCES Award(AwardID);

ALTER TABLE Review

ADD CONSTRAINT FK\_review\_movie FOREIGN KEY (MovieID) REFERENCES Movie(MovieID),

ADD CONSTRAINT FK\_reviewer\_review FOREIGN KEY (ReviewerID) REFERENCES Reviewer(ReviewerID);

**APPENDIX B**

INSERT INTO Crew (CrewID, FirstName, LastName, BirthDate, Nationality, Gender, Email, JobRole) VALUES

(1, 'John', 'Doe', '1980-01-15', 'American', 'Male', 'john.doe@example.com', 'Production Designer'),

(2, 'Emily', 'Brown', '1990-11-05', 'British', 'Female', 'emily.brown@example.com', 'Production Assistant'),

(3, 'Michael', 'Smith', '1983-05-24', 'Canadian', 'Male', 'michael.smith@example.com', 'Sound'),

(4, 'Jessica', 'Taylor', '1987-09-12', 'American', 'Female', 'jessica.taylor@example.com', 'Stunt Coordinator'),

(5, 'David', 'Johnson', '1975-12-30', 'Irish', 'Male', 'david.johnson@example.com', 'Art Department'),

(6, 'Sarah', 'Williams', '1985-03-22', 'Australian', 'Female', 'sarah.williams@example.com', 'Art Director'),

(7, 'James', 'Garcia', '1973-08-08', 'Mexican', 'Male', 'james.garcia@example.com', 'Grip'),

(8, 'Laura', 'Martinez', '1988-10-17', 'Spanish', 'Female', 'laura.martinez@example.com', 'Production Coordinator'),

(9, 'Thomas', 'Rodriguez', '1991-06-26', 'American', 'Male', 'thomas.rodriguez@example.com', 'Sound Mixer'),

(10, 'Sophia', 'Lopez', '1995-04-05', 'Brazilian', 'Female', 'sophia.lopez@example.com', '1st Assistant Camera'),

(11, 'Daniel', 'Harris', '1981-07-18', 'Colombian', 'Male', 'daniel.harris@example.com', 'Assistant Cameraman'),

(12, 'Olivia', 'Young', '1989-11-11', 'American', 'Female', 'olivia.young@example.com', 'Gaffer'),

(13, 'Matthew', 'Green', '1978-03-15', 'British', 'Male', 'matthew.green@example.com', 'Makeup Artist'),

(14, 'Chloe', 'Clark', '1993-12-22', 'American', 'Female', 'chloe.clark@example.com', 'Production Manager'),

(15, 'Anthony', 'Lewis', '1982-10-02', 'American', 'Male', 'anthony.lewis@example.com', 'Script Supervisor'),

(16, 'Emma', 'Walker', '1984-09-20', 'British', 'Female', 'emma.walker@example.com', 'Executive Producer'),

(17, 'David', 'Lee', '1979-04-19', 'American', 'Male', 'david.lee@example.com', 'Location Manager'),

(18, 'Lily', 'Hall', '1992-05-10', 'American', 'Female', 'lily.hall@example.com', 'Prop Master'),

(19, 'Benjamin', 'Allen', '1979-07-07', 'British', 'Male', 'benjamin.allen@example.com', 'Casting Director'),

(20, 'Grace', 'Scott', '1987-11-02', 'Canadian', 'Female', 'grace.scott@example.com', 'Production Designer'),

(21, 'Ryan', 'King', '1984-05-08', 'American', 'Male', 'ryan.king@example.com', 'Production Assistant'),

(22, 'Ella', 'Mitchell', '1989-03-19', 'Australian', 'Female', 'ella.mitchell@example.com', 'Sound'),

(23, 'Ethan', 'Campbell', '1992-04-16', 'Irish', 'Male', 'ethan.campbell@example.com', 'Stunt Coordinator'),

(24, 'Sophia', 'Parker', '1986-08-04', 'American', 'Female', 'sophia.parker@example.com', 'Art Department'),

(25, 'Joshua', 'Evans', '1985-09-30', 'American', 'Male', 'joshua.evans@example.com', 'Art Director'),

(26, 'Mia', 'Carter', '1991-12-29', 'Australian', 'Female', 'mia.carter@example.com', 'Grip'),

(27, 'Andrew', 'Nelson', '1980-01-18', 'Irish', 'Male', 'andrew.nelson@example.com', 'Production Coordinator'),

(28, 'Isabella', 'Perez', '1985-06-27', 'American', 'Female', 'isabella.perez@example.com', 'Sound Mixer'),

(29, 'Jonathan', 'Cook', '1975-08-18', 'Mexican', 'Male', 'jonathan.cook@example.com', '1st Assistant Camera'),

(30, 'Madison', 'Morris', '1991-11-13', 'American', 'Female', 'madison.morris@example.com', 'Assistant Cameraman'),

(31, 'Owen', 'Morgan', '1989-02-27', 'Canadian', 'Male', 'owen.morgan@example.com', 'Gaffer'),

(32, 'Ava', 'Bell', '1995-12-11', 'Mexican', 'Female', 'ava.bell@example.com', 'Makeup Artist'),

(33, 'Hunter', 'Adams', '1984-01-09', 'American', 'Male', 'hunter.adams@example.com', 'Production Manager'),

(34, 'Charlotte', 'Parker', '1990-09-19', 'American', 'Female', 'charlotte.parker@example.com', 'Script Supervisor'),

(35, 'Oliver', 'Brooks', '1988-07-30', 'British', 'Male', 'oliver.brooks@example.com', 'Executive Producer'),

(36, 'Liam', 'Reed', '1981-06-04', 'Irish', 'Male', 'liam.reed@example.com', 'Location Manager'),

(37, 'Amelia', 'Wood', '1993-03-01', 'American', 'Female', 'amelia.wood@example.com', 'Prop Master'),

(38, 'Zoe', 'Bailey', '1992-12-17', 'American', 'Female', 'zoe.bailey@example.com', 'Casting Director'),

(39, 'Henry', 'Gray', '1990-02-25', 'Irish', 'Male', 'henry.gray@example.com', 'Production Designer'),

(40, 'Mila', 'Cox', '1983-07-14', 'Canadian', 'Female', 'mila.cox@example.com', 'Production Assistant'),

(41, 'Lucas', 'Howard', '1986-10-23', 'American', 'Male', 'lucas.howard@example.com', 'Sound'),

(42, 'Violet', 'Murphy', '1984-05-17', 'Irish', 'Female', 'violet.murphy@example.com', 'Stunt Coordinator'),

(43, 'Nathan', 'Rogers', '1985-08-29', 'American', 'Male', 'nathan.rogers@example.com', 'Art Department'),

(44, 'Sofia', 'Barnes', '1987-09-30', 'American', 'Female', 'sofia.barnes@example.com', 'Art Director'),

(45, 'Alexander', 'Reyes', '1994-12-05', 'Canadian', 'Male', 'alexander.reyes@example.com', 'Grip'),

(46, 'Hazel', 'Sanders', '1991-11-25', 'British', 'Female', 'hazel.sanders@example.com', 'Production Coordinator'),

(47, 'Leo', 'Russell', '1989-04-22', 'American', 'Male', 'leo.russell@example.com', 'Sound Mixer'),

(48, 'Ruby', 'Foster', '1982-02-02', 'American', 'Female', 'ruby.foster@example.com', '1st Assistant Camera'),

(49, 'Dominic', 'Morales', '1979-10-28', 'Irish', 'Male', 'dominic.morales@example.com', 'Assistant Cameraman'),

(50, 'Aurora', 'Bennett', '1988-09-18', 'American', 'Female', 'aurora.bennett@example.com', 'Gaffer'),

(51, 'Sam', 'Coleman', '1981-03-15', 'American', 'Male', 'sam.coleman@example.com', 'Makeup Artist'),

(52, 'Natalie', 'Jenkins', '1985-12-29', 'Canadian', 'Female', 'natalie.jenkins@example.com', 'Production Manager'),

(53, 'Julian', 'Torres', '1982-04-08', 'American', 'Male', 'julian.torres@example.com', 'Script Supervisor'),

(54, 'Ella', 'Kim', '1993-10-12', 'South Korean', 'Female', 'ella.kim@example.com', 'Executive Producer'),

(55, 'Evelyn', 'Ward', '1994-06-09', 'Australian', 'Female', 'evelyn.ward@example.com', 'Location Manager');

INSERT INTO Actor (ActorID, FirstName, LastName, BirthDate, Nationality, Gender, AgentCompany, AgentContactEmail) VALUES

(1, 'Robert', 'Downey Jr.', '1965-04-04', 'American', 'Male', 'Creative Artists Agency', 'rdj@example.com'),

(2, 'Chris', 'Hemsworth', '1983-08-11', 'Australian', 'Male', 'Management 360', 'chris.hemsworth@example.com'),

(3, 'Scarlett', 'Johansson', '1984-11-22', 'American', 'Female', 'CAA', 'scarlett.j@example.com'),

(4, 'Tom', 'Holland', '1996-06-01', 'British', 'Male', 'William Morris Endeavor', 'tom.holland@example.com'),

(5, 'Chris', 'Evans', '1981-06-13', 'American', 'Male', 'ICM Partners', 'chris.evans@example.com'),

(6, 'Emma', 'Stone', '1988-11-06', 'American', 'Female', 'Anonymous Content', 'emma.stone@example.com'),

(7, 'Samuel', 'Jackson', '1948-12-21', 'American', 'Male', 'United Talent Agency', 'sam.jackson@example.com'),

(8, 'Anne', 'Hathaway', '1982-11-12', 'American', 'Female', 'CAA', 'anne.hathaway@example.com'),

(9, 'Leonardo', 'DiCaprio', '1974-11-11', 'American', 'Male', 'Appian Way Productions', 'leo.d@example.com'),

(10, 'Jennifer', 'Lawrence', '1990-08-15', 'American', 'Female', 'CAA', 'jlaw@example.com'),

(11, 'Mark', 'Ruffalo', '1967-11-22', 'American', 'Male', 'CAA', 'mark.ruffalo@example.com'),

(12, 'Ryan', 'Reynolds', '1976-10-23', 'Canadian', 'Male', 'William Morris Endeavor', 'ryan.reynolds@example.com'),

(13, 'Natalie', 'Portman', '1981-06-09', 'Israeli-American', 'Female', 'United Talent Agency', 'natalie.portman@example.com'),

(14, 'Brad', 'Pitt', '1963-12-18', 'American', 'Male', 'Plan B Entertainment', 'brad.pitt@example.com'),

(15, 'Margot', 'Robbie', '1990-07-02', 'Australian', 'Female', 'Management 360', 'margot.robbie@example.com'),

(16, 'Tom', 'Cruise', '1962-07-03', 'American', 'Male', 'Creative Artists Agency', 'tom.cruise@example.com'),

(17, 'Zoe', 'Saldana', '1978-06-19', 'American', 'Female', 'CAA', 'zoe.saldana@example.com'),

(18, 'Dwayne', 'Johnson', '1972-05-02', 'American', 'Male', 'Seven Bucks Productions', 'dwayne.j@example.com'),

(19, 'Gal', 'Gadot', '1985-04-30', 'Israeli', 'Female', 'William Morris Endeavor', 'gal.gadot@example.com'),

(20, 'Will', 'Smith', '1968-09-25', 'American', 'Male', 'Overbrook Entertainment', 'will.smith@example.com'),

(21, 'Angelina', 'Jolie', '1975-06-04', 'American', 'Female', 'United Talent Agency', 'angelina.j@example.com'),

(22, 'Hugh', 'Jackman', '1968-10-12', 'Australian', 'Male', 'Creative Artists Agency', 'hugh.jackman@example.com'),

(23, 'Cate', 'Blanchett', '1969-05-14', 'Australian', 'Female', 'United Talent Agency', 'cate.b@example.com'),

(24, 'Keanu', 'Reeves', '1964-09-02', 'Canadian', 'Male', 'William Morris Endeavor', 'keanu.reeves@example.com'),

(25, 'Emma', 'Watson', '1990-04-15', 'British', 'Female', 'CAA', 'emma.watson@example.com');

INSERT INTO Producer (ProducerID, FirstName, LastName, Company, ContactEmail) VALUES

(1, 'Kevin', 'Feige', 'Marvel Studios', 'kevin.feige@marvel.com'),

(2, 'Kathleen', 'Kennedy', 'Lucasfilm', 'kathleen.kennedy@lucasfilm.com'),

(3, 'Emma', 'Thomas', 'Syncopy', 'emma.thomas@syncopy.com'),

(4, 'Jerry', 'Bruckheimer', 'Jerry Bruckheimer Films', 'jerry.bruckheimer@jbf.com'),

(5, 'Scott', 'Rudin', 'Scott Rudin Productions', 'scott.rudin@srp.com'),

(6, 'Ava', 'DuVernay', 'Forward Movement', 'ava.duvernay@forwardmovement.com'),

(7, 'David', 'Heyman', 'Heyday Films', 'david.heyman@heyday.com'),

(8, 'Amy', 'Pascal', 'Pascal Pictures', 'amy.pascal@pascalpictures.com'),

(9, 'James', 'Cameron', 'Lightstorm Entertainment', 'james.cameron@lightstorm.com'),

(10, 'Deborah', 'Snyder', 'The Stone Quarry', 'deborah.snyder@stonequarry.com'),

(11, 'Jon', 'Landau', 'Lightstorm Entertainment', 'jon.landau@lightstorm.com'),

(12, 'Jason', 'Blum', 'Blumhouse Productions', 'jason.blum@blumhouse.com'),

(13, 'Lorenzo', 'di Bonaventura', 'di Bonaventura Pictures', 'lorenzo.dibonaventura@dbp.com'),

(14, 'Dana', 'Brunetti', 'Trigger Street Productions', 'dana.brunetti@triggerstreet.com'),

(15, 'Frank', 'Marshall', 'Kennedy/Marshall Company', 'frank.marshall@kmc.com'),

(16, 'Denise', 'Di Novi', 'Di Novi Pictures', 'denise.dinovi@dinovipictures.com'),

(17, 'Brian', 'Grazer', 'Imagine Entertainment', 'brian.grazer@imagine-ent.com'),

(18, 'Barbara', 'Broccoli', 'EON Productions', 'barbara.broccoli@eon.com'),

(19, 'Neal', 'Moritz', 'Original Film', 'neal.moritz@originalfilm.com'),

(20, 'Gale', 'Anne Hurd', 'Valhalla Motion Pictures', 'gale.hurd@valhalla.com');

INSERT INTO Director (DirectorID, FirstName, LastName, BirthDate, Nationality, Gender, Email) VALUES

(1, 'Steven', 'Spielberg', '1946-12-18', 'American', 'Male', 'steven.spielberg@example.com'),

(2, 'Christopher', 'Nolan', '1970-07-30', 'British-American', 'Male', 'christopher.nolan@example.com'),

(3, 'Martin', 'Scorsese', '1942-11-17', 'American', 'Male', 'martin.scorsese@example.com'),

(4, 'Quentin', 'Tarantino', '1963-03-27', 'American', 'Male', 'quentin.tarantino@example.com'),

(5, 'James', 'Cameron', '1954-08-16', 'Canadian', 'Male', 'james.cameron@example.com'),

(6, 'Kathryn', 'Bigelow', '1951-11-27', 'American', 'Female', 'kathryn.bigelow@example.com'),

(7, 'Sofia', 'Coppola', '1971-05-14', 'American', 'Female', 'sofia.coppola@example.com'),

(8, 'Greta', 'Gerwig', '1983-08-04', 'American', 'Female', 'greta.gerwig@example.com'),

(9, 'Guillermo', 'del Toro', '1964-10-09', 'Mexican', 'Male', 'guillermo.del.toro@example.com'),

(10, 'Jordan', 'Peele', '1979-02-21', 'American', 'Male', 'jordan.peele@example.com'),

(11, 'Ridley', 'Scott', '1937-11-30', 'British', 'Male', 'ridley.scott@example.com'),

(12, 'Patty', 'Jenkins', '1971-07-24', 'American', 'Female', 'patty.jenkins@example.com'),

(13, 'Taika', 'Waititi', '1975-08-16', 'New Zealander', 'Male', 'taika.waititi@example.com'),

(14, 'David', 'Fincher', '1962-08-28', 'American', 'Male', 'david.fincher@example.com'),

(15, 'Alfonso', 'Cuarón', '1961-11-28', 'Mexican', 'Male', 'alfonso.cuaron@example.com'),

(16, 'Ang', 'Lee', '1954-10-23', 'Taiwanese', 'Male', 'ang.lee@example.com'),

(17, 'Lilly', 'Wachowski', '1967-12-29', 'American', 'Female', 'lilly.wachowski@example.com'),

(18, 'Spike', 'Lee', '1957-03-20', 'American', 'Male', 'spike.lee@example.com'),

(19, 'Denis', 'Villeneuve', '1967-10-03', 'Canadian', 'Male', 'denis.villeneuve@example.com'),

(20, 'Chloé', 'Zhao', '1982-03-31', 'Chinese', 'Female', 'chloe.zhao@example.com');

INSERT INTO DOP (DOPID, FirstName, LastName, BirthDate, Nationality, Gender, Email) VALUES

(1, 'Roger', 'Deakins', '1949-05-24', 'British', 'Male', 'roger.deakins@example.com'),

(2, 'Emmanuel', 'Lubezki', '1964-06-21', 'Mexican', 'Male', 'emmanuel.lubezki@example.com'),

(3, 'Rachel', 'Morrison', '1978-04-27', 'American', 'Female', 'rachel.morrison@example.com'),

(4, 'Wally', 'Pfister', '1961-07-08', 'American', 'Male', 'wally.pfister@example.com'),

(5, 'Greig', 'Fraser', '1975-10-03', 'Australian', 'Male', 'greig.fraser@example.com'),

(6, 'Vittorio', 'Storaro', '1940-06-24', 'Italian', 'Male', 'vittorio.storaro@example.com'),

(7, 'Ellen', 'Kuras', '1959-07-10', 'American', 'Female', 'ellen.kuras@example.com'),

(8, 'Robert', 'Richardson', '1955-08-27', 'American', 'Male', 'robert.richardson@example.com'),

(9, 'Mandy', 'Walker', '1963-10-07', 'Australian', 'Female', 'mandy.walker@example.com'),

(10, 'Janusz', 'Kaminski', '1959-06-27', 'Polish', 'Male', 'janusz.kaminski@example.com'),

(11, 'Hoyte', 'van Hoytema', '1971-10-04', 'Swedish', 'Male', 'hoyte.vanhoytema@example.com'),

(12, 'Claudio', 'Miranda', '1965-03-07', 'Chilean', 'Male', 'claudio.miranda@example.com'),

(13, 'Linus', 'Sandgren', '1972-12-05', 'Swedish', 'Male', 'linus.sandgren@example.com'),

(14, 'Charlotte', 'Bruus Christensen', '1978-06-20', 'Danish', 'Female', 'charlotte.christensen@example.com'),

(15, 'Adam', 'Arkapaw', '1984-03-19', 'Australian', 'Male', 'adam.arkapaw@example.com'),

(16, 'Seamus', 'McGarvey', '1967-06-29', 'Irish', 'Male', 'seamus.mcgarvey@example.com'),

(17, 'Bradford', 'Young', '1977-07-06', 'American', 'Male', 'bradford.young@example.com'),

(18, 'Maryse', 'Alberti', '1954-03-10', 'French', 'Female', 'maryse.alberti@example.com'),

(19, 'Caleb', 'Deschanel', '1944-09-21', 'American', 'Male', 'caleb.deschanel@example.com'),

(20, 'Haris', 'Zambarloukos', '1970-03-11', 'Greek', 'Male', 'haris.zambarloukos@example.com');

INSERT INTO Reviewer (ReviewerID, ReviewerFirstName, ReviewerLastName, ReviewerCompany) VALUES

(1, 'Peter', 'Travers', 'Rolling Stone'),

(2, 'Richard', 'Roeper', 'Chicago Sun-Times'),

(3, 'A.O.', 'Scott', 'The New York Times'),

(4, 'Claudia', 'Puig', 'USA Today'),

(5, 'Michael', 'Phillips', 'Chicago Tribune'),

(6, 'Ann', 'Hornaday', 'The Washington Post'),

(7, 'Kenneth', 'Turan', 'Los Angeles Times'),

(8, 'Owen', 'Gleiberman', 'Variety'),

(9, 'Manohla', 'Dargis', 'The New York Times'),

(10, 'Todd', 'McCarthy', 'The Hollywood Reporter'),

(11, 'David', 'Edelstein', 'New York Magazine'),

(12, 'Leonard', 'Maltin', 'Film Critic'),

(13, 'Mick', 'LaSalle', 'San Francisco Chronicle'),

(14, 'Rex', 'Reed', 'Observer'),

(15, 'Lisa', 'Schwarzbaum', 'Entertainment Weekly'),

(16, 'Stephen', 'Whitty', 'The Star-Ledger'),

(17, 'Ty', 'Burr', 'The Boston Globe'),

(18, 'Dana', 'Stevens', 'Slate'),

(19, 'Chris', 'Stuckmann', 'Independent Critic'),

(20, 'Alison', 'Wilmore', 'Vulture');

INSERT INTO Award (AwardID, AwardName, Category, AwardDate, WinnerType) VALUES

(1, 'Academy Awards', 'Best Picture', '2024-03-10', 'Movie'),

(2, 'Academy Awards', 'Best Actor', '2024-03-10', 'Actor'),

(3, 'Academy Awards', 'Best Actress', '2024-03-10', 'Actor'),

(4, 'Academy Awards', 'Best Director', '2024-03-10', 'Director'),

(5, 'Golden Globe Awards', 'Best Motion Picture - Drama', '2024-01-07', 'Movie'),

(6, 'Golden Globe Awards', 'Best Actor - Drama', '2024-01-07', 'Actor'),

(7, 'Golden Globe Awards', 'Best Actress - Drama', '2024-01-07', 'Actor'),

(8, 'BAFTA Awards', 'Best Film', '2024-02-18', 'Movie'),

(9, 'BAFTA Awards', 'Best Actor', '2024-02-18', 'Actor'),

(10, 'BAFTA Awards', 'Best Actress', '2024-02-18', 'Actor'),

(11, 'Screen Actors Guild Awards', 'Outstanding Performance by a Male Actor', '2024-01-21', 'Actor'),

(12, 'Screen Actors Guild Awards', 'Outstanding Performance by a Female Actor', '2024-01-21', 'Actor'),

(13, 'Critics Choice Awards', 'Best Picture', '2024-01-14', 'Movie'),

(14, 'Critics Choice Awards', 'Best Director', '2024-01-14', 'Director'),

(15, 'Directors Guild of America', 'Outstanding Directorial Achievement', '2024-01-13', 'Director'),

(16, 'Producers Guild of America', 'Best Theatrical Motion Picture', '2024-01-20', 'Movie'),

(17, 'Writers Guild of America', 'Best Original Screenplay', '2024-01-28', 'Movie'),

(18, 'Writers Guild of America', 'Best Adapted Screenplay', '2024-01-28', 'Movie'),

(19, 'Independent Spirit Awards', 'Best Feature', '2024-03-03', 'Movie'),

(20, 'Independent Spirit Awards', 'Best Cinematography', '2024-03-03', 'DOP');

INSERT INTO Movie (MovieID, Title, Genre, StartDate, ReleaseDate, Duration, ProductionCost, RevenueGenerated, LanguagesReleased, MaleProtagonist, FemaleProtagonist, Antagonist, DirectorID, AssistantDirectorID, ProducerID, DopID) VALUES

(1, 'Tales of the Deep', 'Western', '2022-08-21', '2023-12-14', 99, 76368296, 479929495, 'English, Spanish', 24, 23, 13, 12, 17, 7, 8),

(2, 'The Forbidden Tower', 'Animation', '2022-07-26', '2023-03-30', 100, 26226317, 493479215, 'English', 5, 23, 17, 3, 13, 11, 12),

(3, 'The Vanishing Light', 'Action', '2022-08-25', '2023-12-23', 134, 29470006, 128131201, 'English, Spanish', 18, 13, 22, 4, 18, 2, 13),

(4, 'The Sacred Oath', 'Documentary', '2021-10-06', '2023-04-09', 110, 23720801, 206079360, 'Mandarin', 20, 10, 10, 18, 14, 19, 19),

(5, 'Fragments of Fate', 'War', '2022-08-26', '2023-05-06', 175, 56630765, 328177398, 'Portuguese', 11, 19, 10, 20, 20, 5, 4),

(6, 'Oceans of Silence', 'Drama', '2022-01-03', '2023-08-12', 95, 51203670, 215409876, 'Japanese', 4, 25, 16, 2, 15, 6, 7),

(7, 'Guardians of the Realm', 'Action', '2022-03-11', '2023-06-10', 118, 39280000, 478909000, 'Korean', 15, 21, 6, 10, 19, 3, 2),

(8, 'Echoes of Eternity', 'Sci-Fi', '2021-10-23', '2023-04-22', 130, 98000000, 420000000, 'English', 2, 3, 14, 5, 6, 1, 1),

(9, 'Shadows of Valor', 'Fantasy', '2022-12-18', '2023-12-05', 140, 78000000, 500000000, 'French', 1, 17, 11, 13, 12, 8, 9),

(10, 'Beyond the Stars', 'Sci-Fi', '2022-01-16', '2023-10-18', 125, 87000000, 420000000, 'English, Japanese', 14, 7, 4, 4, 16, 2, 3),

(11, 'Kingdom of Silence', 'Thriller', '2021-09-30', '2023-07-20', 110, 46000000, 240000000, 'Mandarin', 6, 18, 8, 9, 3, 4, 11),

(12, 'The Phoenix Rises', 'Action', '2021-11-11', '2023-06-25', 125, 58000000, 350000000, 'German', 5, 25, 21, 2, 5, 9, 10),

(13, 'Legacy of Fire', 'Horror', '2022-05-15', '2023-11-21', 140, 62000000, 300000000, 'English', 3, 10, 9, 3, 10, 7, 6),

(14, 'The Lost Hero', 'Fantasy', '2021-08-29', '2023-10-14', 130, 75000000, 380000000, 'English, Korean', 13, 20, 12, 6, 18, 11, 5),

(15, 'Dawn of Legends', 'Biography', '2021-07-10', '2023-09-15', 100, 31000000, 150000000, 'English, Spanish', 7, 19, 23, 7, 8, 14, 4),

(16, 'Infinite Horizon', 'Adventure', '2022-04-01', '2023-05-30', 98, 34000000, 210000000, 'Italian', 16, 11, 3, 15, 12, 13, 8),

(17, 'The Final Stand', 'Action', '2022-09-08', '2023-11-28', 160, 66000000, 270000000, 'Russian', 22, 8, 5, 14, 3, 7, 6),

(18, 'Chaos Theory', 'Mystery', '2021-05-12', '2023-06-19', 113, 45000000, 200000000, 'English, Russian', 17, 2, 13, 6, 10, 5, 17),

(19, 'Broken Chains', 'Western', '2021-09-20', '2023-09-12', 144, 52000000, 310000000, 'French, German', 19, 14, 7, 18, 16, 12, 3),

(20, 'The Hidden Oasis', 'Fantasy', '2021-11-25', '2023-07-01', 150, 68000000, 290000000, 'Korean', 9, 6, 15, 11, 19, 10, 15),

(21, 'Endless Journey', 'Documentary', '2022-06-16', '2023-08-18', 102, 39000000, 160000000, 'Mandarin', 25, 12, 18, 13, 7, 18, 13),

(22, 'Crimson Destiny', 'Romance', '2022-10-19', '2023-12-08', 140, 32000000, 120000000, 'Spanish', 20, 16, 2, 12, 14, 19, 7),

(23, 'Rise of the Titans', 'Action', '2021-08-22', '2023-03-25', 133, 71000000, 420000000, 'Italian', 10, 9, 1, 5, 1, 16, 12),

(24, 'Dreams of the Future', 'Sci-Fi', '2021-04-10', '2023-08-21', 125, 84000000, 300000000, 'English, French', 15, 13, 5, 1, 2, 20, 2),

(25, 'The Chosen Ones', 'Drama', '2022-02-14', '2023-09-30', 108, 55000000, 190000000, 'English, Italian', 6, 24, 22, 8, 17, 15, 18),

(26, 'Celestial Empire', 'Fantasy', '2022-03-17', '2023-07-05', 144, 64000000, 370000000, 'German', 19, 25, 17, 2, 14, 8, 14),

(27, 'Lands of Glory', 'Western', '2021-07-29', '2023-11-10', 155, 72000000, 260000000, 'Russian', 18, 22, 11, 16, 5, 6, 9),

(28, 'Odyssey of Souls', 'Horror', '2021-08-13', '2023-10-05', 110, 68000000, 320000000, 'Japanese', 23, 14, 19, 17, 9, 3, 7),

(29, 'Veil of Shadows', 'Fantasy', '2021-12-07', '2023-12-28', 138, 45000000, 250000000, 'English', 12, 15, 16, 20, 6, 18, 10),

(30, 'Winds of Change', 'Action', '2022-04-23', '2023-05-14', 148, 47000000, 280000000, 'Hindi', 7, 18, 4, 19, 3, 6, 18),

(31, 'The Silver Shield', 'Adventure', '2021-09-12', '2023-07-15', 130, 30000000, 250000000, 'English, Spanish', 9, 25, 20, 11, 8, 4, 13),

(32, 'Path of the Phoenix', 'Fantasy', '2022-05-18', '2023-11-22', 120, 54000000, 380000000, 'Mandarin', 13, 17, 21, 12, 19, 7, 5),

(33, 'Shattered Dreams', 'Drama', '2022-08-10', '2023-06-13', 140, 47000000, 410000000, 'Portuguese', 11, 14, 22, 8, 4, 10, 2),

(34, 'Nightfall Chronicles', 'Sci-Fi', '2021-03-22', '2023-08-17', 108, 61000000, 280000000, 'French', 15, 16, 9, 3, 10, 18, 6),

(35, 'The Cursed City', 'Horror', '2021-07-01', '2023-04-21', 100, 33000000, 210000000, 'Russian', 18, 23, 6, 16, 15, 5, 12),

(36, 'Mystic Isles', 'Fantasy', '2021-05-10', '2023-05-18', 125, 42000000, 240000000, 'English, Korean', 4, 10, 13, 19, 1, 9, 7),

(37, 'The Lost Expedition', 'Documentary', '2021-02-25', '2023-10-30', 115, 28000000, 190000000, 'Japanese', 1, 6, 15, 2, 8, 11, 14),

(38, 'The Savage Hunt', 'Adventure', '2022-11-19', '2023-12-07', 135, 52000000, 290000000, 'German', 24, 12, 5, 7, 6, 17, 15),

(39, 'Ashes to Ashes', 'Western', '2021-10-04', '2023-09-19', 140, 40000000, 180000000, 'English', 20, 19, 10, 12, 14, 3, 8),

(40, 'Echoes in the Mist', 'Mystery', '2022-12-23', '2023-06-06', 112, 47000000, 280000000, 'Italian', 5, 15, 4, 10, 11, 12, 9),

(41, 'Heart of Stone', 'Romance', '2021-01-27', '2023-11-26', 100, 26000000, 220000000, 'French, Spanish', 3, 9, 1, 18, 5, 14, 2),

(42, 'Legacy of the Fallen', 'Fantasy', '2021-06-22', '2023-08-01', 115, 55000000, 350000000, 'English', 14, 13, 7, 15, 4, 16, 5),

(43, 'Crimson Dawn', 'Adventure', '2022-02-15', '2023-03-12', 140, 62000000, 400000000, 'Korean', 17, 25, 20, 4, 6, 15, 3),

(44, 'Beneath the Waves', 'Drama', '2021-11-17', '2023-07-25', 110, 32000000, 150000000, 'Spanish', 8, 21, 14, 9, 18, 8, 10),

(45, 'Forsaken Lands', 'War', '2022-03-02', '2023-09-16', 148, 47000000, 320000000, 'English', 23, 2, 3, 5, 13, 20, 14),

(46, 'The Iron Crown', 'Fantasy', '2022-08-04', '2023-05-10', 130, 34000000, 270000000, 'Portuguese', 19, 11, 18, 7, 10, 17, 6),

(47, 'Whispers of the Night', 'Mystery', '2022-06-07', '2023-08-15', 104, 46000000, 180000000, 'Russian', 6, 8, 9, 1, 14, 3, 4),

(48, 'Realm of the Ancients', 'Sci-Fi', '2021-09-06', '2023-11-30', 115, 30000000, 200000000, 'English', 7, 24, 5, 3, 11, 13, 7),

(49, 'Lost Souls', 'Horror', '2022-04-10', '2023-10-22', 138, 56000000, 280000000, 'English, Spanish', 16, 18, 10, 19, 9, 5, 15),

(50, 'The Darkened Forest', 'Fantasy', '2022-01-19', '2023-07-21', 120, 52000000, 250000000, 'French', 2, 17, 12, 16, 3, 6, 12),

(51, 'The Star Chaser', 'Adventure', '2022-09-08', '2023-12-11', 140, 44000000, 230000000, 'Italian', 25, 6, 21, 8, 7, 12, 11),

(52, 'Wings of Valor', 'Action', '2021-10-28', '2023-04-18', 112, 36000000, 260000000, 'English, French', 12, 20, 14, 18, 4, 11, 8),

(53, 'A New Dawn', 'Romance', '2021-06-16', '2023-08-19', 100, 43000000, 290000000, 'Japanese', 11, 3, 13, 6, 19, 16, 13),

(54, 'Beyond the Horizon', 'Biography', '2022-05-04', '2023-06-08', 120, 51000000, 370000000, 'Korean', 9, 22, 8, 10, 17, 7, 9),

(55, 'Fragments of Eternity', 'Sci-Fi', '2021-02-23', '2023-10-19', 140, 58000000, 420000000, 'Mandarin', 4, 15, 23, 11, 18, 5, 10),

(56, 'The Eternal Flame', 'Fantasy', '2022-03-21', '2023-11-01', 130, 50000000, 250000000, 'English', 13, 7, 19, 15, 14, 2, 14),

(57, 'The Sacred Prophecy', 'Adventure', '2021-08-11', '2023-09-29', 148, 37000000, 210000000, 'French', 21, 16, 6, 17, 10, 9, 3),

(58, 'Echoes of War', 'War', '2021-07-15', '2023-04-24', 130, 34000000, 280000000, 'Russian', 5, 25, 11, 19, 12, 15, 6),

(59, 'The Final Hour', 'Drama', '2022-06-01', '2023-07-09', 90, 47000000, 150000000, 'Portuguese', 3, 14, 17, 12, 1, 20, 7),

(60, 'Into the Void', 'Sci-Fi', '2022-08-14', '2023-08-20', 135, 61000000, 270000000, 'Hindi', 10, 13, 24, 16, 4, 8, 9),

(61, 'Valley of the Lost', 'Adventure', '2022-04-12', '2023-05-29', 125, 52000000, 210000000, 'English, Italian', 22, 2, 19, 7, 13, 18, 9),

(62, 'The Fire Within', 'Action', '2021-07-19', '2023-08-22', 143, 57000000, 360000000, 'German', 19, 9, 5, 14, 6, 16, 15),

(63, 'The Silent Path', 'Thriller', '2022-02-09', '2023-07-15', 130, 63000000, 220000000, 'English, French', 6, 11, 8, 8, 18, 4, 10),

(64, 'The Eternal Quest', 'Fantasy', '2021-05-14', '2023-11-04', 115, 39000000, 210000000, 'Italian', 2, 23, 18, 11, 12, 19, 12),

(65, 'The Forbidden Realm', 'Mystery', '2021-10-06', '2023-06-11', 138, 42000000, 280000000, 'Korean', 20, 4, 10, 5, 15, 14, 8),

(66, 'Legacy of Light', 'Documentary', '2022-03-24', '2023-10-08', 92, 25000000, 170000000, 'Russian', 23, 13, 21, 3, 7, 8, 6),

(67, 'Empire of Ash', 'Fantasy', '2021-09-17', '2023-08-29', 148, 68000000, 330000000, 'Portuguese', 4, 15, 7, 2, 10, 18, 3),

(68, 'Secrets of the Past', 'Adventure', '2021-12-22', '2023-05-03', 127, 49000000, 270000000, 'Japanese', 11, 21, 14, 1, 9, 13, 4),

(69, 'Whispering Shadows', 'Horror', '2021-02-27', '2023-07-30', 125, 58000000, 250000000, 'Spanish', 14, 10, 13, 6, 16, 3, 9),

(70, 'The Last Frontier', 'Sci-Fi', '2022-06-15', '2023-09-21', 133, 72000000, 400000000, 'English, Korean', 15, 19, 12, 7, 11, 6, 15),

(71, 'Beyond Redemption', 'Drama', '2022-04-03', '2023-12-10', 100, 33000000, 290000000, 'Mandarin', 5, 24, 9, 17, 13, 15, 8),

(72, 'The Forgotten Legacy', 'Biography', '2021-11-09', '2023-03-18', 150, 56000000, 310000000, 'French', 7, 20, 6, 18, 2, 12, 17),

(73, 'Celestial Quest', 'Sci-Fi', '2021-08-23', '2023-10-17', 140, 61000000, 280000000, 'Russian', 3, 18, 10, 4, 7, 1, 5),

(74, 'The Shining Blade', 'Action', '2021-09-14', '2023-04-29', 130, 47000000, 320000000, 'English', 16, 12, 3, 9, 5, 17, 11),

(75, 'Odyssey of Shadows', 'Fantasy', '2022-05-19', '2023-11-05', 110, 50000000, 280000000, 'English, Italian', 8, 25, 15, 10, 6, 13, 14),

(76, 'Echoes of Eternity', 'Mystery', '2022-07-21', '2023-06-30', 120, 65000000, 410000000, 'English, Spanish', 1, 7, 20, 19, 8, 14, 3),

(77, 'Kingdom Reborn', 'War', '2021-10-31', '2023-07-27', 110, 55000000, 240000000, 'Italian', 25, 13, 4, 15, 14, 20, 2),

(78, 'Chronicles of Valor', 'Adventure', '2022-03-28', '2023-10-13', 135, 62000000, 320000000, 'French', 12, 23, 11, 8, 19, 9, 18),

(79, 'Rise of Shadows', 'Fantasy', '2021-07-29', '2023-12-02', 145, 70000000, 410000000, 'English, French', 18, 17, 2, 3, 20, 15, 16),

(80, 'Echoes from the Past', 'Drama', '2022-09-15', '2023-11-02', 102, 41820640, 245318009, 'German', 6, 13, 14, 7, 15, 12, 9);

INSERT INTO Review (MovieID, ReviewerID, ReviewerRating, ReviewerDescription) VALUES

(1, 14, '4/5', 'A refreshing take on the Western genre with well-executed themes.'),

(2, 3, '4.5/5', 'An animation masterpiece that enchants viewers of all ages.'),

(3, 1, '3.5/5', 'A thrilling action flick with impressive stunts but a familiar plot.'),

(4, 5, '4/5', 'Insightful documentary with a unique perspective on its subject.'),

(5, 13, '3.5/5', 'A gritty war film with powerful visuals and emotional depth.'),

(6, 6, '4/5', 'An emotional drama that resonates deeply with the audience.'),

(7, 1, '4/5', 'Non-stop action and captivating performances make this a hit.'),

(8, 11, '4.5/5', 'A thought-provoking Sci-Fi journey with stunning visuals.'),

(9, 7, '4/5', 'A beautifully crafted fantasy that transports viewers to another world.'),

(10, 11, '5/5', 'A Sci-Fi marvel with a compelling plot and stunning special effects.'),

(11, 12, '3.5/5', 'A thriller that keeps viewers on the edge of their seats.'),

(12, 1, '4/5', 'Action-packed with a solid storyline that entertains.'),

(13, 8, '3/5', 'A horror movie with some scares but an unremarkable plot.'),

(14, 7, '4.5/5', 'Enchanting fantasy that brings characters to life in a magical setting.'),

(15, 4, '4/5', 'A captivating biography that delves deep into its subject’s life.'),

(16, 2, '4/5', 'A thrilling adventure with memorable moments and scenery.'),

(17, 1, '3.5/5', 'High-paced action but lacks depth in the storyline.'),

(18, 9, '4/5', 'An engaging mystery that keeps you guessing until the end.'),

(19, 14, '3.5/5', 'A Western with solid performances but a predictable plot.'),

(20, 10, '4/5', 'A heartfelt romance with genuine chemistry between the leads.'),

(21, 5, '4.5/5', 'A documentary that provides a powerful look at its subject.'),

(22, 10, '3/5', 'A romance that struggles to make an emotional impact.'),

(23, 1, '4/5', 'A well-crafted action movie with gripping scenes.'),

(24, 11, '5/5', 'An epic Sci-Fi adventure that explores deep themes.'),

(25, 6, '4/5', 'A well-acted drama that deals with complex issues.'),

(26, 7, '4/5', 'A fantasy epic with beautiful visuals and a grand storyline.'),

(27, 14, '4/5', 'An old-school Western that delivers on atmosphere and style.'),

(28, 8, '3.5/5', 'Chilling horror with good scares, though the plot feels thin.'),

(29, 7, '4.5/5', 'A fantastical journey that captivates and enchants.'),

(30, 13, '3.5/5', 'A war drama that falls short of being truly memorable.'),

(31, 2, '4.5/5', 'A breathtaking adventure with stunning locations and thrills.'),

(32, 7, '4/5', 'A richly imagined fantasy with a compelling plot.'),

(33, 6, '4/5', 'A moving drama that leaves a lasting impression.'),

(34, 9, '3.5/5', 'A decent mystery with intriguing twists.'),

(35, 13, '4/5', 'An intense war movie that portrays the horrors of battle.'),

(36, 7, '4/5', 'An enchanting fantasy that takes the audience on a magical journey.'),

(37, 5, '4/5', 'A documentary with a powerful message and strong storytelling.'),

(38, 2, '4/5', 'An exciting adventure that combines action and beauty.'),

(39, 14, '3.5/5', 'A Western that offers style, but the plot is lacking.'),

(40, 9, '3.5/5', 'A mystery that pulls you in with interesting twists.'),

(41, 10, '4/5', 'A romance that stands out with its genuine emotion.'),

(42, 7, '4.5/5', 'An epic fantasy with a world that feels fully alive.'),

(43, 2, '4.5/5', 'An adventure filled with thrilling moments and memorable characters.'),

(44, 6, '4/5', 'A drama that explores complex relationships.'),

(45, 13, '4/5', 'A war film that delves into the human side of battle.'),

(46, 7, '4/5', 'A fantastical adventure with rich visuals and a compelling story.'),

(47, 9, '3.5/5', 'A mystery that intrigues, though it lacks a strong ending.'),

(48, 11, '4/5', 'A Sci-Fi experience that captivates with its vision of the future.'),

(49, 8, '3.5/5', 'A horror film with some eerie moments but an average plot.'),

(50, 7, '4/5', 'A richly imagined fantasy with a captivating plot.'),

(51, 2, '4.5/5', 'An adventure with breathtaking scenery and heart-pounding moments.'),

(52, 1, '4/5', 'Action-packed and entertaining from start to finish.'),

(53, 10, '3.5/5', 'A romance with strong leads but a predictable plot.'),

(54, 4, '4.5/5', 'An inspiring biography that sheds light on a remarkable life.'),

(55, 11, '5/5', 'A sci-fi epic that dives deep into thought-provoking ideas.'),

(56, 7, '4/5', 'A fantasy tale that enchants and inspires.'),

(57, 2, '4/5', 'An adventure with stunning visuals and a well-paced plot.'),

(58, 13, '4.5/5', 'A war movie that captures the intensity of conflict.'),

(59, 6, '4/5', 'A drama with outstanding performances and emotional depth.'),

(60, 11, '4/5', 'A sci-fi journey that engages both mind and heart.'),

(61, 2, '4.5/5', 'An exhilarating adventure with a gripping storyline.'),

(62, 1, '4/5', 'High-octane action with a well-crafted plot.'),

(63, 12, '3.5/5', 'A thriller that keeps you guessing until the end.'),

(64, 7, '4/5', 'A captivating fantasy with memorable characters.'),

(65, 9, '4/5', 'A mystery that draws you in with unexpected twists.'),

(66, 5, '4.5/5', 'An impactful documentary with a strong message.'),

(67, 7, '4.5/5', 'A fantasy epic that immerses you in a vivid world.'),

(68, 2, '4/5', 'An adventurous tale that entertains and excites.'),

(69, 8, '3.5/5', 'A horror film with effective scares but lacks depth.'),

(70, 11, '4.5/5', 'A sci-fi story with intriguing ideas and powerful visuals.'),

(71, 6, '4/5', 'A deeply moving drama with exceptional performances.'),

(72, 4, '4/5', 'An inspiring biography that does justice to its subject.'),

(73, 11, '4/5', 'A sci-fi thriller with suspenseful moments and stunning effects.'),

(74, 1, '3.5/5', 'Action-packed but falls short on plot depth.'),

(75, 7, '4/5', 'A fantasy with rich storytelling and vivid characters.'),

(76, 9, '4/5', 'A mystery that keeps you engaged throughout.'),

(77, 13, '4/5', 'A war story that portrays the grit and sacrifice of soldiers.'),

(78, 2, '4.5/5', 'An exhilarating adventure that keeps you hooked.'),

(79, 7, '4.5/5', 'A captivating fantasy with a breathtaking world.'),

(80, 6, '4/5', 'A powerful drama that resonates with its emotional depth.');

INSERT INTO Movie\_Awards (MovieID, AwardID, AwardDate) VALUES

(1, 1, '2024-03-10'), (1, 13, '2024-01-14'), (1, 8, '2024-02-18'), (2, 5, '2024-01-07'), (3, 7, '2024-01-07'), (3, 9, '2024-02-18'), (4, 13, '2024-01-14'), (4, 16, '2024-01-20'), (5, 1, '2024-03-10'), (5, 6, '2024-01-07'), (6, 4, '2024-03-10'), (7, 10, '2024-02-18'), (7, 13, '2024-01-14'), (8, 5, '2024-01-07'), (8, 2, '2024-03-10'), (9, 1, '2024-03-10'), (10, 8, '2024-02-18'), (10, 14, '2024-01-14'), (11, 7, '2024-01-07'), (12, 15, '2024-01-13'), (13, 9, '2024-02-18'), (14, 1, '2024-03-10'), (15, 19, '2024-03-03'), (16, 3, '2024-03-10'), (17, 6, '2024-01-07'), (18, 12, '2024-01-21'), (19, 10, '2024-02-18'), (20, 20, '2024-03-03'), (21, 11, '2024-01-21'), (22, 18, '2024-01-28'), (23, 8, '2024-02-18'), (24, 5, '2024-01-07'), (25, 16, '2024-01-20'), (26, 14, '2024-01-14'), (27, 15, '2024-01-13'), (28, 3, '2024-03-10');

INSERT INTO Crew\_Linking (MovieID, CrewID) VALUES

(1, 1), (2, 20), (3, 39), (4, 13), (5, 1), (6, 20), (7, 39), (8, 13), (9, 1), (10, 20), (11, 39), (12, 13), (13, 1), (14, 20), (15, 39), (16, 13), (17, 1), (18, 20), (19, 39), (20, 13), (21, 1), (22, 20), (23, 39), (24, 13), (25, 1), (26, 20), (27, 39), (28, 13), (29, 1), (30, 20), (31, 39), (32, 13), (33, 1), (34, 20), (35, 39), (36, 13), (37, 1), (38, 20), (39, 39), (40, 13), (41, 1), (42, 20), (43, 39), (44, 13), (45, 1), (46, 20), (47, 39), (48, 13), (49, 1), (50, 20), (51, 39), (52, 13), (53, 1), (54, 20), (55, 39), (56, 13), (57, 1), (58, 20), (59, 39), (60, 13), (61, 1), (62, 20), (63, 39), (64, 13), (65, 1), (66, 20), (67, 39), (68, 13), (69, 1), (70, 20), (71, 39), (72, 13), (73, 1), (74, 20), (75, 39), (76, 13), (77, 1), (78, 20), (79, 39), (80, 13), (1, 2), (2, 40), (3, 21), (4, 8), (5, 2), (6, 40), (7, 21), (8, 8), (9, 2), (10, 40), (11, 21), (12, 8), (13, 2), (14, 40), (15, 21), (16, 8), (17, 2), (18, 40), (19, 21), (20, 8), (21, 2), (22, 40), (23, 21), (24, 8), (25, 2), (26, 40), (27, 21), (28, 8), (29, 2), (30, 40), (31, 21), (32, 8), (33, 2), (34, 40), (35, 21), (36, 8), (37, 2), (38, 40), (39, 21), (40, 8), (41, 2), (42, 40), (43, 21), (44, 8), (45, 2), (46, 40), (47, 21), (48, 8), (49, 2), (50, 40), (51, 21), (52, 8), (53, 2), (54, 40), (55, 21), (56, 8), (57, 2), (58, 40), (59, 21), (60, 8), (61, 2), (62, 40), (63, 21), (64, 8), (65, 2), (66, 40), (67, 21), (68, 8), (69, 2), (70, 40), (71, 21), (72, 8), (73, 2), (74, 40), (75, 21), (76, 8), (77, 2), (78, 40), (79, 21), (80, 8), (1, 3), (2, 9), (3, 22), (4, 41), (5, 3), (6, 9), (7, 22), (8, 41), (9, 3), (10, 9), (11, 22), (12, 41), (13, 3), (14, 9), (15, 22), (16, 41), (17, 3), (18, 9), (19, 22), (20, 41), (21, 3), (22, 9), (23, 22), (24, 41), (25, 3), (26, 9), (27, 22), (28, 41), (29, 3), (30, 9), (31, 22), (32, 41), (33, 3), (34, 9), (35, 22), (36, 41), (37, 3), (38, 9), (39, 22), (40, 41), (41, 3), (42, 9), (43, 22), (44, 41), (45, 3), (46, 9), (47, 22), (48, 41), (49, 3), (50, 9), (51, 22), (52, 41), (53, 3), (54, 9), (55, 22), (56, 41), (57, 3), (58, 9), (59, 22), (60, 41), (61, 3), (62, 9), (63, 22), (64, 41), (65, 3), (66, 9), (67, 22), (68, 41), (69, 3), (70, 9), (71, 22), (72, 41), (73, 3), (74, 9), (75, 22), (76, 41), (77, 3), (78, 9), (79, 22), (80, 41), (1, 4), (2, 23), (3, 42), (4, 14), (5, 4), (6, 23), (7, 42), (8, 14), (9, 4), (10, 23), (11, 42), (12, 14), (13, 4), (14, 23), (15, 42), (16, 14), (17, 4), (18, 23), (19, 42), (20, 14), (21, 4), (22, 23), (23, 42), (24, 14), (25, 4), (26, 23), (27, 42), (28, 14), (29, 4), (30, 23), (31, 42), (32, 14), (33, 4), (34, 23), (35, 42), (36, 14), (37, 4), (38, 23), (39, 42), (40, 14), (41, 4), (42, 23), (43, 42), (44, 14), (45, 4), (46, 23), (47, 42), (48, 14), (49, 4), (50, 23), (51, 42), (52, 14), (53, 4), (54, 23), (55, 42), (56, 14), (57, 4), (58, 23), (59, 42), (60, 14), (61, 4), (62, 23), (63, 42), (64, 14), (65, 4), (66, 23), (67, 42), (68, 14), (69, 4), (70, 23), (71, 42), (72, 14), (73, 4), (74, 23), (75, 42), (76, 14), (77, 4), (78, 23), (79, 42), (80, 14), (1, 5), (2, 24), (3, 43), (4, 15), (5, 5), (6, 24), (7, 43), (8, 15), (9, 5), (10, 24), (11, 43), (12, 15), (13, 5), (14, 24), (15, 43), (16, 15), (17, 5), (18, 24), (19, 43), (20, 15), (21, 5), (22, 24), (23, 43), (24, 15), (25, 5), (26, 24), (27, 43), (28, 15), (29, 5), (30, 24), (31, 43), (32, 15), (33, 5), (34, 24), (35, 43), (36, 15), (37, 5), (38, 24), (39, 43), (40, 15), (41, 5), (42, 24), (43, 43), (44, 15), (45, 5), (46, 24), (47, 43), (48, 15), (49, 5), (50, 24), (51, 43), (52, 15), (53, 5), (54, 24), (55, 43), (56, 15), (57, 5), (58, 24), (59, 43), (60, 15), (61, 5), (62, 24), (63, 43), (64, 15), (65, 5), (66, 24), (67, 43), (68, 15), (69, 5), (70, 24), (71, 43), (72, 15), (73, 5), (74, 24), (75, 43), (76, 15), (77, 5), (78, 24), (79, 43), (80, 15), (1, 6), (2, 25), (3, 44), (4, 16), (5, 6), (6, 25), (7, 44), (8, 16), (9, 6), (10, 25), (11, 44), (12, 16), (13, 6), (14, 25), (15, 44), (16, 16), (17, 6), (18, 25), (19, 44), (20, 16), (21, 6), (22, 25), (23, 44), (24, 16), (25, 6), (26, 25), (27, 44), (28, 16), (29, 6), (30, 25), (31, 44), (32, 16), (33, 6), (34, 25), (35, 44), (36, 16), (37, 6), (38, 25), (39, 44), (40, 16), (41, 6), (42, 25), (43, 44), (44, 16), (45, 6), (46, 25), (47, 44), (48, 16), (49, 6), (50, 25), (51, 44), (52, 16), (53, 6), (54, 25), (55, 44), (56, 16), (57, 6), (58, 25), (59, 44), (60, 16), (61, 6), (62, 25), (63, 44), (64, 16), (65, 6), (66, 25), (67, 44), (68, 16), (69, 6), (70, 25), (71, 44), (72, 16), (73, 6), (74, 25), (75, 44), (76, 16), (77, 6), (78, 25), (79, 44), (80, 16), (1, 7), (2, 26), (3, 45), (4, 17), (5, 7), (6, 26), (7, 45), (8, 17), (9, 7), (10, 26), (11, 45), (12, 17), (13, 7), (14, 26), (15, 45), (16, 17), (17, 7), (18, 26), (19, 45), (20, 17), (21, 7), (22, 26), (23, 45), (24, 17), (25, 7), (26, 26), (27, 45), (28, 17), (29, 7), (30, 26), (31, 45), (32, 17), (33, 7), (34, 26), (35, 45), (36, 17), (37, 7), (38, 26), (39, 45), (40, 17), (41, 7), (42, 26), (43, 45), (44, 17), (45, 7), (46, 26), (47, 45), (48, 17), (49, 7), (50, 26), (51, 45), (52, 17), (53, 7), (54, 26), (55, 45), (56, 17), (57, 7), (58, 26), (59, 45), (60, 17), (61, 7), (62, 26), (63, 45), (64, 17), (65, 7), (66, 26), (67, 45), (68, 17), (69, 7), (70, 26), (71, 45), (72, 17), (73, 7), (74, 26), (75, 45), (76, 17), (77, 7), (78, 26), (79, 45), (80, 17), (1, 8), (2, 27), (3, 46), (4, 18), (5, 8), (6, 27), (7, 46), (8, 18), (9, 8), (10, 27), (11, 46), (12, 18), (13, 8), (14, 27), (15, 46), (16, 18), (17, 8), (18, 27), (19, 46), (20, 18), (21, 8), (22, 27), (23, 46), (24, 18), (25, 8), (26, 27), (27, 46), (28, 18), (29, 8), (30, 27), (31, 46), (32, 18), (33, 8), (34, 27), (35, 46), (36, 18), (37, 8), (38, 27), (39, 46), (40, 18), (41, 8), (42, 27), (43, 46), (44, 18), (45, 8), (46, 27), (47, 46), (48, 18), (49, 8), (50, 27), (51, 46), (52, 18), (53, 8), (54, 27), (55, 46), (56, 18), (57, 8), (58, 27), (59, 46), (60, 18), (61, 8), (62, 27), (63, 46), (64, 18), (65, 8), (66, 27), (67, 46), (68, 18), (69, 8), (70, 27), (71, 46), (72, 18), (73, 8), (74, 27), (75, 46), (76, 18), (77, 8), (78, 27), (79, 46), (80, 18), (1, 9), (2, 28), (3, 47), (4, 19), (5, 9), (6, 28), (7, 47), (8, 19), (9, 9), (10, 28), (11, 47), (12, 19), (13, 9), (14, 28), (15, 47), (16, 19), (17, 9), (18, 28), (19, 47), (20, 19), (21, 9), (22, 28), (23, 47), (24, 19), (25, 9), (26, 28), (27, 47), (28, 19), (29, 9), (30, 28), (31, 47), (32, 19), (33, 9), (34, 28), (35, 47), (36, 19), (37, 9), (38, 28), (39, 47), (40, 19), (41, 9), (42, 28), (43, 47), (44, 19), (45, 9), (46, 28), (47, 47), (48, 19), (49, 9), (50, 28), (51, 47), (52, 19), (53, 9), (54, 28), (55, 47), (56, 19), (57, 9), (58, 28), (59, 47), (60, 19), (61, 9), (62, 28), (63, 47), (64, 19), (65, 9), (66, 28), (67, 47), (68, 19), (69, 9), (70, 28), (71, 47), (72, 19), (73, 9), (74, 28), (75, 47), (76, 19), (77, 9), (78, 28), (79, 47), (80, 19), (1, 10), (2, 29), (3, 48), (4, 20), (5, 10), (6, 29), (7, 48), (8, 20), (9, 10), (10, 29), (11, 48), (12, 20), (13, 10), (14, 29), (15, 48), (16, 20), (17, 10), (18, 29), (19, 48), (20, 20), (21, 10), (22, 29), (23, 48), (24, 20), (25, 10), (26, 29), (27, 48), (28, 20), (29, 10), (30, 29), (31, 48), (32, 20), (33, 10), (34, 29), (35, 48), (36, 20), (37, 10), (38, 29), (39, 48), (40, 20), (41, 10), (42, 29), (43, 48), (44, 20), (45, 10), (46, 29), (47, 48), (48, 20), (49, 10), (50, 29), (51, 48), (52, 20), (53, 10), (54, 29), (55, 48), (56, 20), (57, 10), (58, 29), (59, 48), (60, 20), (61, 10), (62, 29), (63, 48), (64, 20), (65, 10), (66, 29), (67, 48), (68, 20), (69, 10), (70, 29), (71, 48), (72, 20), (73, 10), (74, 29), (75, 48), (76, 20), (77, 10), (78, 29), (79, 48), (80, 20), (1, 11), (2, 30), (3, 49), (4, 21), (5, 11), (6, 30), (7, 49), (8, 21), (9, 11), (10, 30), (11, 49), (12, 21), (13, 11), (14, 30), (15, 49), (16, 21), (17, 11), (18, 30), (19, 49), (20, 21), (21, 11), (22, 30), (23, 49), (24, 21), (25, 11), (26, 30), (27, 49), (28, 21), (29, 11), (30, 30), (31, 49), (32, 21), (33, 11), (34, 30), (35, 49), (36, 21), (37, 11), (38, 30), (39, 49), (40, 21), (41, 11), (42, 30), (43, 49), (44, 21), (45, 11), (46, 30), (47, 49), (48, 21), (49, 11), (50, 30), (51, 49), (52, 21), (53, 11), (54, 30), (55, 49), (56, 21), (57, 11), (58, 30), (59, 49), (60, 21), (61, 11), (62, 30), (63, 49), (64, 21), (65, 11), (66, 30), (67, 49), (68, 21), (69, 11), (70, 30), (71, 49), (72, 21), (73, 11), (74, 30), (75, 49), (76, 21), (77, 11), (78, 30), (79, 49), (80, 21), (1, 12), (2, 31), (3, 50), (4, 22), (5, 12), (6, 31), (7, 50), (8, 22), (9, 12), (10, 31), (11, 50), (12, 22), (13, 12), (14, 31), (15, 50), (16, 22), (17, 12), (18, 31), (19, 50), (20, 22), (21, 12), (22, 31), (23, 50), (24, 22), (25, 12), (26, 31), (27, 50), (28, 22), (29, 12), (30, 31), (31, 50), (32, 22), (33, 12), (34, 31), (35, 50), (36, 22), (37, 12), (38, 31), (39, 50), (40, 22), (41, 12), (42, 31), (43, 50), (44, 22), (45, 12), (46, 31), (47, 50), (48, 22), (49, 12), (50, 31), (51, 50), (52, 22), (53, 12), (54, 31), (55, 50), (56, 22), (57, 12), (58, 31), (59, 50), (60, 22), (61, 12), (62, 31), (63, 50), (64, 22), (65, 12), (66, 31), (67, 50), (68, 22), (69, 12), (70, 31), (71, 50), (72, 22), (73, 12), (74, 31), (75, 50), (76, 22), (77, 12), (78, 31), (79, 50), (80, 22), (1, 13), (2, 32), (3, 51), (4, 23), (5, 13), (6, 32), (7, 51), (8, 23), (9, 13), (10, 32), (11, 51), (12, 23), (13, 13), (14, 32), (15, 51), (16, 23), (17, 13), (18, 32), (19, 51), (20, 23), (21, 13), (22, 32), (23, 51), (24, 23), (25, 13), (26, 32), (27, 51), (28, 23), (29, 13), (30, 32), (31, 51), (32, 23), (33, 13), (34, 32), (35, 51), (36, 23), (37, 13), (38, 32), (39, 51), (40, 23), (41, 13), (42, 32), (43, 51), (44, 23), (45, 13), (46, 32), (47, 51), (48, 23), (49, 13), (50, 32), (51, 51), (52, 23), (53, 13), (54, 32), (55, 51), (56, 23), (57, 13), (58, 32), (59, 51), (60, 23), (61, 13), (62, 32), (63, 51), (64, 23), (65, 13), (66, 32), (67, 51), (68, 23), (69, 13), (70, 32), (71, 51), (72, 23), (73, 13), (74, 32), (75, 51), (76, 23), (77, 13), (78, 32), (79, 51), (80, 23), (1, 14), (2, 33), (3, 52), (4, 24), (5, 14), (6, 33), (7, 52), (8, 24), (9, 14), (10, 33), (11, 52), (12, 24), (13, 14), (14, 33), (15, 52), (16, 24), (17, 14), (18, 33), (19, 52), (20, 24), (21, 14), (22, 33), (23, 52), (24, 24), (25, 14), (26, 33), (27, 52), (28, 24), (29, 14), (30, 33), (31, 52), (32, 24), (33, 14), (34, 33), (35, 52), (36, 24), (37, 14), (38, 33), (39, 52), (40, 24), (41, 14), (42, 33), (43, 52), (44, 24), (45, 14), (46, 33), (47, 52), (48, 24), (49, 14), (50, 33), (51, 52), (52, 24), (53, 14), (54, 33), (55, 52), (56, 24), (57, 14), (58, 33), (59, 52), (60, 24), (61, 14), (62, 33), (63, 52), (64, 24), (65, 14), (66, 33), (67, 52), (68, 24), (69, 14), (70, 33), (71, 52), (72, 24), (73, 14), (74, 33), (75, 52), (76, 24), (77, 14), (78, 33), (79, 52), (80, 24), (1, 15), (2, 34), (3, 53), (4, 25), (5, 15), (6, 34), (7, 53), (8, 25), (9, 15), (10, 34), (11, 53), (12, 25), (13, 15), (14, 34), (15, 53), (16, 25), (17, 15), (18, 34), (19, 53), (20, 25), (21, 15), (22, 34), (23, 53), (24, 25), (25, 15), (26, 34), (27, 53), (28, 25), (29, 15), (30, 34), (31, 53), (32, 25), (33, 15), (34, 34), (35, 53), (36, 25), (37, 15), (38, 34), (39, 53), (40, 25), (41, 15), (42, 34), (43, 53), (44, 25), (45, 15), (46, 34), (47, 53), (48, 25), (49, 15), (50, 34), (51, 53), (52, 25), (53, 15), (54, 34), (55, 53), (56, 25), (57, 15), (58, 34), (59, 53), (60, 25), (61, 15), (62, 34), (63, 53), (64, 25), (65, 15), (66, 34), (67, 53), (68, 25), (69, 15), (70, 34), (71, 53), (72, 25), (73, 15), (74, 34), (75, 53), (76, 25), (77, 15), (78, 34), (79, 53), (80, 25), (1, 16), (2, 35), (3, 54), (4, 26), (5, 16), (6, 35), (7, 54), (8, 26), (9, 16), (10, 35), (11, 54), (12, 26), (13, 16), (14, 35), (15, 54), (16, 26), (17, 16), (18, 35), (19, 54), (20, 26), (21, 16), (22, 35), (23, 54), (24, 26), (25, 16), (26, 35), (27, 54), (28, 26), (29, 16), (30, 35), (31, 54), (32, 26), (33, 16), (34, 35), (35, 54), (36, 26), (37, 16), (38, 35), (39, 54), (40, 26), (41, 16), (42, 35), (43, 54), (44, 26), (45, 16), (46, 35), (47, 54), (48, 26), (49, 16), (50, 35), (51, 54), (52, 26), (53, 16), (54, 35), (55, 54), (56, 26), (57, 16), (58, 35), (59, 54), (60, 26), (61, 16), (62, 35), (63, 54), (64, 26), (65, 16), (66, 35), (67, 54), (68, 26), (69, 16), (70, 35), (71, 54), (72, 26), (73, 16), (74, 35), (75, 54), (76, 26), (77, 16), (78, 35), (79, 54), (80, 26),(1, 17), (2, 36), (3, 55), (4, 27), (5, 17), (6, 36), (7, 55), (8, 27), (9, 17), (10, 36), (11, 55), (12, 27), (13, 17), (14, 36), (15, 55), (16, 27), (17, 17), (18, 36), (19, 55), (20, 27), (21, 17), (22, 36), (23, 55), (24, 27), (25, 17), (26, 36), (27, 55), (28, 27), (29, 17), (30, 36), (31, 55), (32, 27), (33, 17), (34, 36), (35, 55), (36, 27), (37, 17), (38, 36), (39, 55), (40, 27), (41, 17), (42, 36), (43, 55), (44, 27), (45, 17), (46, 36), (47, 55), (48, 27), (49, 17), (50, 36), (51, 55), (52, 27), (53, 17), (54, 36), (55, 55), (56, 27), (57, 17), (58, 36), (59, 55), (60, 27), (61, 17), (62, 36), (63, 55), (64, 27), (65, 17), (66, 36), (67, 55), (68, 27), (69, 17), (70, 36), (71, 55), (72, 27), (73, 17), (74, 36), (75, 55), (76, 27), (77, 17), (78, 36), (79, 55), (80, 27),(1, 18), (2, 37), (3, 18), (4, 28), (5, 18), (6, 37), (7, 37), (8, 28), (9, 18), (10, 37), (11, 18), (12, 28), (13, 18), (14, 37), (15, 37), (16, 28), (17, 18), (18, 37), (19, 18), (20, 28), (21, 18), (22, 37), (23, 37), (24, 28), (25, 18), (26, 37), (27, 18), (28, 28), (29, 18), (30, 37), (31, 37), (32, 28), (33, 18), (34, 37), (35, 18), (36, 28), (37, 18), (38, 37), (39, 37), (40, 28), (41, 18), (42, 37), (43, 18), (44, 28), (45, 18), (46, 37), (47, 37), (48, 28), (49, 18), (50, 37), (51, 18), (52, 28), (53, 18), (54, 37), (55, 37), (56, 28), (57, 18), (58, 37), (59, 18), (60, 28), (61, 18), (62, 37), (63, 37), (64, 28), (65, 18), (66, 37), (67, 18), (68, 28), (69, 18), (70, 37), (71, 37), (72, 28), (73, 18), (74, 37), (75, 18), (76, 28), (77, 18), (78, 37), (79, 37), (80, 28),(1, 19), (2, 38), (3, 19), (4, 29), (5, 19), (6, 38), (7, 19), (8, 29), (9, 19), (10, 38), (11, 19), (12, 29), (13, 19), (14, 38), (15, 19), (16, 29), (17, 19), (18, 38), (19, 19), (20, 29), (21, 19), (22, 38), (23, 19), (24, 29), (25, 19), (26, 38), (27, 19), (28, 29), (29, 19), (30, 38), (31, 19), (32, 29), (33, 19), (34, 38), (35, 19), (36, 29), (37, 19), (38, 38), (39, 19), (40, 29), (41, 19), (42, 38), (43, 19), (44, 29), (45, 19), (46, 38), (47, 19), (48, 29), (49, 19), (50, 38), (51, 19), (52, 29), (53, 19), (54, 38), (55, 19), (56, 29), (57, 19), (58, 38), (59, 19), (60, 29), (61, 19), (62, 38), (63, 19), (64, 29), (65, 19), (66, 38), (67, 19), (68, 29), (69, 19), (70, 38), (71, 19), (72, 29), (73, 19), (74, 38), (75, 19), (76, 29), (77, 19), (78, 38), (79, 19), (80, 29);

**APPENDIX C**

CREATE INDEX idx\_movie\_genre ON Movie (Genre);

CREATE INDEX idx\_movie\_releasedate ON Movie (ReleaseDate);

CREATE INDEX idx\_movie\_revenue ON Movie (RevenueGenerated);

CREATE INDEX idx\_movie\_maleprotagonist ON Movie (MaleProtagonist);

CREATE INDEX idx\_movie\_femaleprotagonist ON Movie (FemaleProtagonist);

CREATE INDEX idx\_movie\_antagonist ON Movie (Antagonist);

CREATE INDEX idx\_actor\_nationality ON Actor (Nationality);

CREATE INDEX idx\_award\_category ON Award (Category);

CREATE INDEX idx\_crew\_jobrole ON Crew (JobRole);

CREATE INDEX idx\_crewlink\_movieid\_crewid ON Crew\_Linking (MovieID, CrewID);

CREATE INDEX idx\_movieawards\_movieid\_awardid ON Movie\_Awards (MovieID, AwardID);

CREATE INDEX idx\_movieawards\_awarddate ON Movie\_Awards (AwardDate);

CREATE INDEX idx\_review\_movieid\_reviewerid ON Review (MovieID, ReviewerID);

CREATE INDEX idx\_review\_rating ON Review (ReviewerRating);

**APPENDIX D**

1. Retrieve the top 10 highest-grossing movies with their directors and main cast.

SELECT

    m.Title AS "Movie Title",

    m.RevenueGenerated AS "Revenue Generated",

    CONCAT(d.FirstName, ' ', d.LastName) AS "Director Name",

    CONCAT(maleProtag.FirstName, ' ', maleProtag.LastName) AS

"Male Protagonist Name",

    CONCAT(femaleProtag.FirstName, ' ', femaleProtag.LastName) AS "Female Protagonist Name",

    CONCAT(antagonist.FirstName, ' ', antagonist.LastName) AS "Antagonist Name",

    a.AwardName AS "Award Name",

    a.Category AS "Award Category",

    ma.AwardDate AS "Award Date"

FROM

    Movie m

JOIN

    Director d ON m.DirectorID = d.DirectorID

JOIN

    Actor maleProtag ON m.MaleProtagonist = maleProtag.ActorID

JOIN

    Actor femaleProtag ON m.FemaleProtagonist = femaleProtag.ActorID

JOIN

    Actor antagonist ON m.Antagonist = antagonist.ActorID

LEFT JOIN

    Movie\_Awards ma ON m.MovieID = ma.MovieID

LEFT JOIN

    Award a ON ma.AwardID = a.AwardID

ORDER BY

    m.RevenueGenerated DESCA screenshot of a computer

Description automatically generated, LIMIT 10;

1. List all movies with awards won, including the award category and date.

SELECT

    m.Title AS "Movie Title",

    CONCAT(d.FirstName, ' ', d.LastName) AS "Director Name",

    CONCAT(maleProtag.FirstName, ' ', maleProtag.LastName) AS "Male Protagonist Name",

    CONCAT(femaleProtag.FirstName, ' ', femaleProtag.LastName) AS "Female Protagonist Name",

    CONCAT(antagonist.FirstName, ' ', antagonist.LastName) AS "Antagonist Name",

    a.AwardName AS "Award Name",

    a.Category AS "Award Category",

    ma.AwardDate AS "Award Date"

FROM

    Movie m

JOIN

    Director d ON m.DirectorID = d.DirectorID

JOIN

    Actor maleProtag ON m.MaleProtagonist = maleProtag.ActorID

JOIN

    Actor femaleProtag ON m.FemaleProtagonist = femaleProtag.ActorID

JOIN

    Actor antagonist ON m.Antagonist = antagonist.ActorID

JOIN

    Movie\_Awards ma ON m.MovieID = ma.MovieID

JOIN

    Award a ON ma.AwardID = a.AwardID

ORDER BY

A screenshot of a computer

Description automatically generated    ma.AwardDate DESC;

1. Find actors who starred in movies across at least three different genres.

SELECT

    CONCAT(a.FirstName, ' ', a.LastName) AS "Actor Name",

    COUNT(DISTINCT m.Genre) AS "Genre Count"

FROM

    Actor a

JOIN

    Movie m ON a.ActorID = m.MaleProtagonist

            OR a.ActorID = m.FemaleProtagonist

            OR a.ActorID = m.Antagonist

GROUP BY

    a.ActorID, a.FirstName, a.LastName

HAVING

A screenshot of a computer

Description automatically generated    COUNT(DISTINCT m.Genre) >= 3;

1. Identify directors who have directed movies in multiple languages.

SELECT

    CONCAT(d.FirstName, ' ', d.LastName) AS "Director Name",

    COUNT(DISTINCT m.LanguagesReleased) AS "Languages Count"

FROM

    Director d

JOIN

    Movie m ON d.DirectorID = m.DirectorID

GROUP BY

    d.DirectorID, d.FirstName, d.LastName

HAVING

A screenshot of a computer

Description automatically generated    COUNT(DISTINCT m.LanguagesReleased) > 1;

1. List movies that feature both a male and female protagonist, along with their genres.

SELECT

    m.Title AS "Movie Title",

    m.Genre AS "Genre",

    CONCAT(maleProtag.FirstName, ' ', maleProtag.LastName) AS "Male Protagonist Name",

    CONCAT(femaleProtag.FirstName, ' ', femaleProtag.LastName) AS "Female Protagonist Name"

FROM

    Movie m

JOIN

    Actor maleProtag ON m.MaleProtagonist = maleProtag.ActorID

JOIN

    Actor femaleProtag ON m.FemaleProtagonist = femaleProtag.ActorID

WHERE

    m.MaleProtagonist IS NOT NULL

A screenshot of a computer

Description automatically generated    AND m.FemaleProtagonist IS NOT NULL;

1. Retrieve movies with production costs exceeding $60 million and their revenue generated.

SELECT

    m.Title AS "Movie Title",

    m.ProductionCost AS "Production Cost",

    m.RevenueGenerated AS "Revenue Generated"

FROM

    Movie m

WHERE

    m.ProductionCost > 60000000

ORDER BY

A screenshot of a computer

Description automatically generated    m.RevenueGenerated DESC;

1. Find all movies reviewed by a specific reviewer, including review ratings and comments. Using first name (here “Peter”)

SELECT

    m.Title AS "Movie Title",

    r.ReviewerRating AS "Reviewer Rating",

    r.ReviewerDescription AS "Review Comment"

FROM

    Movie m

JOIN

    Review r ON m.MovieID = r.MovieID

JOIN

    Reviewer rev ON r.ReviewerID = rev.ReviewerID

WHERE

A screenshot of a computer

Description automatically generated    rev.ReviewerFirstName = 'Peter'; 

1. Show actors who have worked with the same director more than 2 movies.

SELECT

    CONCAT(a.FirstName, ' ', a.LastName) AS "Actor Name",

    CONCAT(d.FirstName, ' ', d.LastName) AS "Director Name",

    COUNT(m.MovieID) AS "Movies Together"

FROM

    Actor a

JOIN

    Movie m ON a.ActorID = m.MaleProtagonist

            OR a.ActorID = m.FemaleProtagonist

            OR a.ActorID = m.Antagonist

JOIN

    Director d ON m.DirectorID = d.DirectorID

GROUP BY

    a.ActorID, d.DirectorID, a.FirstName, a.LastName, d.FirstName, d.LastName

HAVING

    COUNT(m.MovieID) > 2;

A screenshot of a computer

Description automatically generated

1. List movies where the antagonist's nationality differs from the protagonists.

SELECT

    m.Title AS "Movie Title",

    CONCAT(maleProtag.FirstName, ' ', maleProtag.LastName) AS "Male Protagonist",

    maleProtag.Nationality AS "Male Protagonist Nationality",

    CONCAT(femaleProtag.FirstName, ' ', femaleProtag.LastName) AS "Female Protagonist",

    femaleProtag.Nationality AS "Female Protagonist Nationality",

    CONCAT(antagonist.FirstName, ' ', antagonist.LastName) AS "Antagonist",

    antagonist.Nationality AS "Antagonist Nationality"

FROM

    Movie m

JOIN

    Actor maleProtag ON m.MaleProtagonist = maleProtag.ActorID

JOIN

    Actor femaleProtag ON m.FemaleProtagonist = femaleProtag.ActorID

JOIN

    Actor antagonist ON m.Antagonist = antagonist.ActorID

WHERE

    (maleProtag.Nationality <> antagonist.Nationality)

    OR (femaleProtag.Nationality <> antagonist.Nationality);

A screenshot of a computer

Description automatically generated

1. Retrieve the total revenue generated by each genre, sorted by highest revenue.

SELECT

    m.Genre AS "Genre",

    SUM(m.RevenueGenerated) AS "Total Revenue"

FROM

    Movie m

GROUP BY

    m.Genre

ORDER BY

A screenshot of a computer

Description automatically generated    "Total Revenue" DESC;

1. Find producers who produced more than one movie in the same genre.

SELECT

    CONCAT(p.FirstName, ' ', p.LastName) AS "Producer Name",

    m.Genre AS "Genre",

    COUNT(m.MovieID) AS "Movies Produced"

FROM

    Producer p

JOIN

    Movie m ON p.ProducerID = m.ProducerID

GROUP BY

    p.ProducerID, p.FirstName, p.LastName, m.Genre

HAVING

A screenshot of a computer

Description automatically generated    COUNT(m.MovieID) > 1;

1. Show movies that received an award in the "Best Director" category.

SELECT

    m.Title AS "Movie Title",

    a.AwardName AS "Award Name",

    a.Category AS "Award Category",

    ma.AwardDate AS "Award Date",

    CONCAT(d.FirstName, ' ', d.LastName) AS "Director Name"

FROM

    Movie m

JOIN

    Movie\_Awards ma ON m.MovieID = ma.MovieID

JOIN

    Award a ON ma.AwardID = a.AwardID

JOIN

    Director d ON m.DirectorID = d.DirectorID

WHERE

    a.Category = 'Best Director'

ORDER BY

    ma.AwardDate DESC;

A screenshot of a computer

Description automatically generated

1. List directors who have worked with the same assistant director on multiple movies.

SELECT

    CONCAT(d.FirstName, ' ', d.LastName) AS "Director Name",

    CONCAT(ad.FirstName, ' ', ad.LastName) AS "Assistant Director Name",

    COUNT(m.MovieID) AS "Movies Together"

FROM

    Movie m

JOIN

    Director d ON m.DirectorID = d.DirectorID

JOIN

    Director ad ON m.AssistantDirectorID = ad.DirectorID

GROUP BY

    d.DirectorID, ad.DirectorID, d.FirstName, d.LastName, ad.FirstName, ad.LastName

HAVING

A screenshot of a computer

Description automatically generated    COUNT(m.MovieID) > 1;

1. Retrieve movies where the main protagonist has won at least one acting award.

SELECT

    m.Title AS "Movie Title",

    CONCAT(maleProtag.FirstName, ' ', maleProtag.LastName) AS "Male Protagonist",

    CONCAT(femaleProtag.FirstName, ' ', femaleProtag.LastName) AS "Female Protagonist",

    a.AwardName AS "Award Name",

    a.Category AS "Award Category",

    ma.AwardDate AS "Award Date"

FROM

    Movie m

LEFT JOIN

    Actor maleProtag ON m.MaleProtagonist = maleProtag.ActorID

LEFT JOIN

    Actor femaleProtag ON m.FemaleProtagonist = femaleProtag.ActorID

JOIN

    Movie\_Awards ma ON m.MovieID = ma.MovieID

JOIN

    Award a ON ma.AwardID = a.AwardID

WHERE

    (a.Category = 'Best Actor' AND maleProtag.ActorID IS NOT NULL)

    OR (a.Category = 'Best Actress' AND femaleProtag.ActorID IS NOT NULL)

ORDER BY

    ma.AwardDate DESC;

A screenshot of a computer

Description automatically generated

1. Show actors whose age at the time of the movie release was under 30.

SELECT

    CONCAT(a.FirstName, ' ', a.LastName) AS "Actor Name",

    m.Title AS "Movie Title",

    m.ReleaseDate AS "Release Date",

    a.BirthDate AS "Actor Birth Date",

    TIMESTAMPDIFF(YEAR, a.BirthDate, m.ReleaseDate) AS "Age at Release"

FROM

    Actor a

JOIN

    Movie m ON a.ActorID = m.MaleProtagonist

            OR a.ActorID = m.FemaleProtagonist

            OR a.ActorID = m.Antagonist

WHERE

    TIMESTAMPDIFF(YEAR, a.BirthDate, m.ReleaseDate) < 30;

A screenshot of a computer

Description automatically generated

1. Find all female actors who have worked with at least three different directors.

SELECT

    CONCAT(a.FirstName, ' ', a.LastName) AS "Actor Name",

    COUNT(DISTINCT m.DirectorID) AS "Different Directors"

FROM

    Actor a

JOIN

    Movie m ON a.ActorID = m.FemaleProtagonist

WHERE

    a.Gender = 'Female'

GROUP BY

    a.ActorID, a.FirstName, a.LastName

HAVING

    COUNT(DISTINCT m.DirectorID) >= 3;

A screenshot of a computer

Description automatically generated

1. Retrieve movies with more than five crew members listed in the Crew\_Linking table.

SELECT

    m.Title AS "Movie Title",

    COUNT(cl.CrewID) AS "Crew Members Count"

FROM

    Movie m

JOIN

    Crew\_Linking cl ON m.MovieID = cl.MovieID

GROUP BY

    m.MovieID, m.Title

HAVING

    COUNT(cl.CrewID) > 5;

A screenshot of a computer

Description automatically generated

1. List all movies with awards, grouped by movie and sorted by award date.

SELECT

    m.Title AS "Movie Title",

    a.AwardName AS "Award Name",

    a.Category AS "Award Category",

    ma.AwardDate AS "Award Date"

FROM

    Movie m

JOIN

    Movie\_Awards ma ON m.MovieID = ma.MovieID

JOIN

    Award a ON ma.AwardID = a.AwardID

ORDER BY

    m.Title, ma.AwardDate;

A screenshot of a computer screen

Description automatically generated

1. List the top 5 longest movies with their duration and director names.

SELECT

    m.Title AS "Movie Title",

    m.Duration AS "Duration (minutes)",

    CONCAT(d.FirstName, ' ', d.LastName) AS "Director Name"

FROM

    Movie m

JOIN

    Director d ON m.DirectorID = d.DirectorID

ORDER BY

    m.Duration DESC

LIMIT 5;

A screenshot of a computer

Description automatically generated

1. Find movies released in the same year as their director's birthday month.

SELECT

    m.Title AS "Movie Title",

    m.ReleaseDate AS "Release Date",

    CONCAT(d.FirstName, ' ', d.LastName) AS "Director Name",

    d.BirthDate AS "Director Birth Date"

FROM

    Movie m

JOIN

    Director d ON m.DirectorID = d.DirectorID

WHERE

    MONTH(m.ReleaseDate) = MONTH(d.BirthDate);

A screenshot of a computer

Description automatically generated

1. List all reviews with ratings, filtered by movies with revenue over $400 million.

SELECT

    r.MovieID AS "Movie ID",

    m.Title AS "Movie Title",

    r.ReviewerRating AS "Reviewer Rating",

    r.ReviewerDescription AS "Review Comment"

FROM

    Review r

JOIN

    Movie m ON r.MovieID = m.MovieID

WHERE

    m.RevenueGenerated > 400000000;

A screenshot of a computer

Description automatically generated

1. Show movies where the male protagonist and antagonist are of the same nationality.

SELECT

    m.Title AS "Movie Title",

    CONCAT(maleProtag.FirstName, ' ', maleProtag.LastName) AS "Male Protagonist",

    CONCAT(antagonist.FirstName, ' ', antagonist.LastName) AS "Antagonist",

    maleProtag.Nationality AS "Nationality"

FROM

    Movie m

JOIN

    Actor maleProtag ON m.MaleProtagonist = maleProtag.ActorID

JOIN

    Actor antagonist ON m.Antagonist = antagonist.ActorID

WHERE

A screenshot of a computer

Description automatically generated    maleProtag.Nationality = antagonist.Nationality;

1. List directors who have worked with at least two producers on different movies.

SELECT

    CONCAT(d.FirstName, ' ', d.LastName) AS "Director Name",

    COUNT(DISTINCT m.ProducerID) AS "Different Producers"

FROM

    Director d

JOIN

    Movie m ON d.DirectorID = m.DirectorID

GROUP BY

    d.DirectorID, d.FirstName, d.LastName

HAVING

    COUNT(DISTINCT m.ProducerID) >= 3

A screenshot of a computer

Description automatically generated

1. Retrieve actors who have starred as both protagonist and antagonist in different movies.

SELECT

    CONCAT(a.FirstName, ' ', a.LastName) AS "Actor Name"

FROM

    Actor a

JOIN

    Movie m1 ON a.ActorID = m1.MaleProtagonist

              OR a.ActorID = m1.FemaleProtagonist

JOIN

    Movie m2 ON a.ActorID = m2.Antagonist

WHERE

    m1.MovieID <> m2.MovieID

GROUP BY

    a.ActorID, a.FirstName, a.LastName;

A screenshot of a computer

Description automatically generated

1. Show all movies released between two specific years with production costs and revenue.

SELECT

    m.Title AS "Movie Title",

    m.ReleaseDate AS "Release Date",

    m.ProductionCost AS "Production Cost",

    m.RevenueGenerated AS "Revenue Generated"

FROM

    Movie m

WHERE

    YEAR(m.ReleaseDate) BETWEEN 2020 AND 2023;

A screenshot of a data

Description automatically generated

1. Find movies where the duration exceeds the average movie duration.

SELECT

    m.Title AS "Movie Title",

    m.Duration AS "Duration"

FROM

    Movie m

WHERE

    m.Duration > (SELECT AVG(Duration) FROM Movie);

A screenshot of a computer

Description automatically generated

1. Retrieve all movies with both male and female protagonists, ordered by release date.

SELECT

    m.Title AS "Movie Title",

    m.ReleaseDate AS "Release Date",

    CONCAT(maleProtag.FirstName, ' ', maleProtag.LastName) AS "Male Protagonist",

    CONCAT(femaleProtag.FirstName, ' ', femaleProtag.LastName) AS "Female Protagonist"

FROM

    Movie m

JOIN

    Actor maleProtag ON m.MaleProtagonist = maleProtag.ActorID

JOIN

    Actor femaleProtag ON m.FemaleProtagonist = femaleProtag.ActorID

WHERE

    m.MaleProtagonist IS NOT NULL

    AND m.FemaleProtagonist IS NOT NULL

ORDER BY

A screenshot of a computer

Description automatically generated    m.ReleaseDate;

1. TOP 5 Revenue generated by Female antagonists

SELECT

    m.Title AS "Movie Title",

    CONCAT(a.FirstName, ' ', a.LastName) AS "Female Antagonist",

    m.RevenueGenerated AS "Revenue Generated",

    m.ProductionCost AS "Production Cost"

FROM

    Movie m

JOIN

    Actor a ON m.Antagonist = a.ActorID

LEFT JOIN

    Review r ON m.MovieID = r.MovieID

WHERE

    a.Gender = 'Female'

GROUP BY

    m.MovieID, m.Title, a.FirstName, a.LastName, m.RevenueGenerated, m.ProductionCost

ORDER BY

m.RevenueGenerated DESC

A screenshot of a computer

Description automatically generatedlimit 5;

1. Show movies where antagoninsts got an award

SELECT

    m.Title AS "Movie Title",

    CONCAT(a.FirstName, ' ', a.LastName) AS "Antagonist Name",

    a.Gender AS "Antagonist Gender",

    aw.AwardName AS "Award Name",

    aw.Category AS "Award Category",

    ma.AwardDate AS "Award Date"

FROM

    Movie m

JOIN

    Actor a ON m.Antagonist = a.ActorID

JOIN

    Movie\_Awards ma ON m.MovieID = ma.MovieID

JOIN

    Award aw ON ma.AwardID = aw.AwardID

WHERE

    aw.Category LIKE '%Actor%'  -- Adjust if award categories include specific antagonist awards

ORDER BY

    ma.AwardDate DESC;

A screenshot of a computer

Description automatically generated

1. Top 10 Award winning movies with highest revenue

SELECT

    m.Title AS "Movie Title",

    m.RevenueGenerated AS "Revenue Generated",

    m.ProductionCost AS "Production Cost",

    aw.AwardName AS "Award Name",

    aw.Category AS "Award Category",

    ma.AwardDate AS "Award Date"

FROM

    Movie m

JOIN

    Movie\_Awards ma ON m.MovieID = ma.MovieID

JOIN

    Award aw ON ma.AwardID = aw.AwardID

ORDER BY

    m.RevenueGenerated DESC;

A screenshot of a computer

Description automatically generated