ADAMA SCIENCE AND TECHNOLOGY UNIVERSITY

DEPARTMENET OF COMPUTER SCIENCE AND ENGINEERING

GROUP PROJECT:

MEMBER NAME	**ID**
Sirajudin Seid……	**ugr/31238/15**
Samuel Gemeda……	**ugr/03353/15**
Lidya Sisay ……	**ugr/30833/15**
Adonay Alemu……	**ugr/30112/15**

MOVIES COLLECTION MANAGEMENT SYSTEM

PROJECT

Database Management System

Overview:

➔the project is related with movies collection management system

The project include a lot of levels :-

- Director level
- Author level
- Editor level
- Movie studio level
- Actors level

✓ The main facilities available in this project are :

- ⇨ We sort the movies by ascending order or descending order
- ⇨ We can search the movies we want to watch by its title or identification number
- ⇨ Users can access the detail of the movie

Introduction

The Movies Collection Database is a structured repository for film enthusiasts and industry professionals, designed to organize and analyze extensive information about movies. It includes details such as titles, genres, release dates, directors, cast, and ratings, allowing users to explore trends within the cinematic landscape.

As the library of films continues to grow, a well-designed database is essential for efficient management. Whether for personal collections, academic research, or movie recommendation systems, this database serves as a vital tool.

Purpose :

1. Centralized Information: To consolidate all relevant data about movies—such as titles, genres, release dates, directors, cast members, and ratings—into a single, accessible platform.

2. Enhanced Searchability: To enable users to quickly search for specific films or filter results based on various criteria (e.g., genre, year, director) for ease of access.

3. Data Analysis: To facilitate analysis of trends in the film industry, such as popular genres over time, box office performance, and audience ratings, aiding in research and decision-making.

4. Collection Management: To assist collectors and libraries in managing their film inventories, tracking acquisitions, and maintaining records of viewed films.

5. Recommendation Systems: To support the development of personalized movie recommendation systems that suggest films based on user preferences and viewing history.

6. Historical Reference: To serve as a comprehensive reference for film historians, researchers, and enthusiasts interested in studying the evolution of cinema.

7. User Engagement: To enhance user interaction by allowing them to rate films, write reviews, and create watchlists, fostering a community around shared interests.

8. Data Integrity: To ensure accuracy and consistency in film information, reducing discrepancies that may arise from multiple sources.

By achieving these objectives, the Movies Collection Database not only enriches the user experience but also contributes to the broader understanding and appreciation of cinema as an art form.

Goals of proposed system

1. **Planned approach towards working:** - The working in the organization will be well planned and organized. The data will be stored properly in data stores, which will help in retrieval of information as well as its storage.
2. **Accuracy**: - The level of accuracy in the proposed system will be higher. All operation would be done correctly and it ensures that whatever information is coming from the center is accurate.
3. **Reliability**: - The reliability of the proposed system will be high due to the above stated reasons. The reason for the increased reliability of the system is that now there would be proper storage of information.
4. **No Redundancy**: - In the proposed system utmost care would be that no information is repeated anywhere, in storage or otherwise. This would assure economic use of storage space and consistency in the data stored.
5. **Immediate** retrieval of information: - The main objective of proposed system is to provide for a quick and efficient retrieval of information.

6. **Immediate storage of information**: - In manual system there are many problems to store the largest amount of information.
7. **Easy to Operate**: - The system should be easy to operate and should be such that it can be developed within a short period of time and fit in the limited budget of the user

TECHNICAL FEASIBILITY

In this project we have only implement the back end by using "MYSQL WORKBENCH"

AS an example we create 11 tables in this project:

1, actors

2,authors

3 .directors

4. editor

5.genres

6, movieactors

7,movies

8. moviestudios

9, producer

10,soundtracks

11, studios

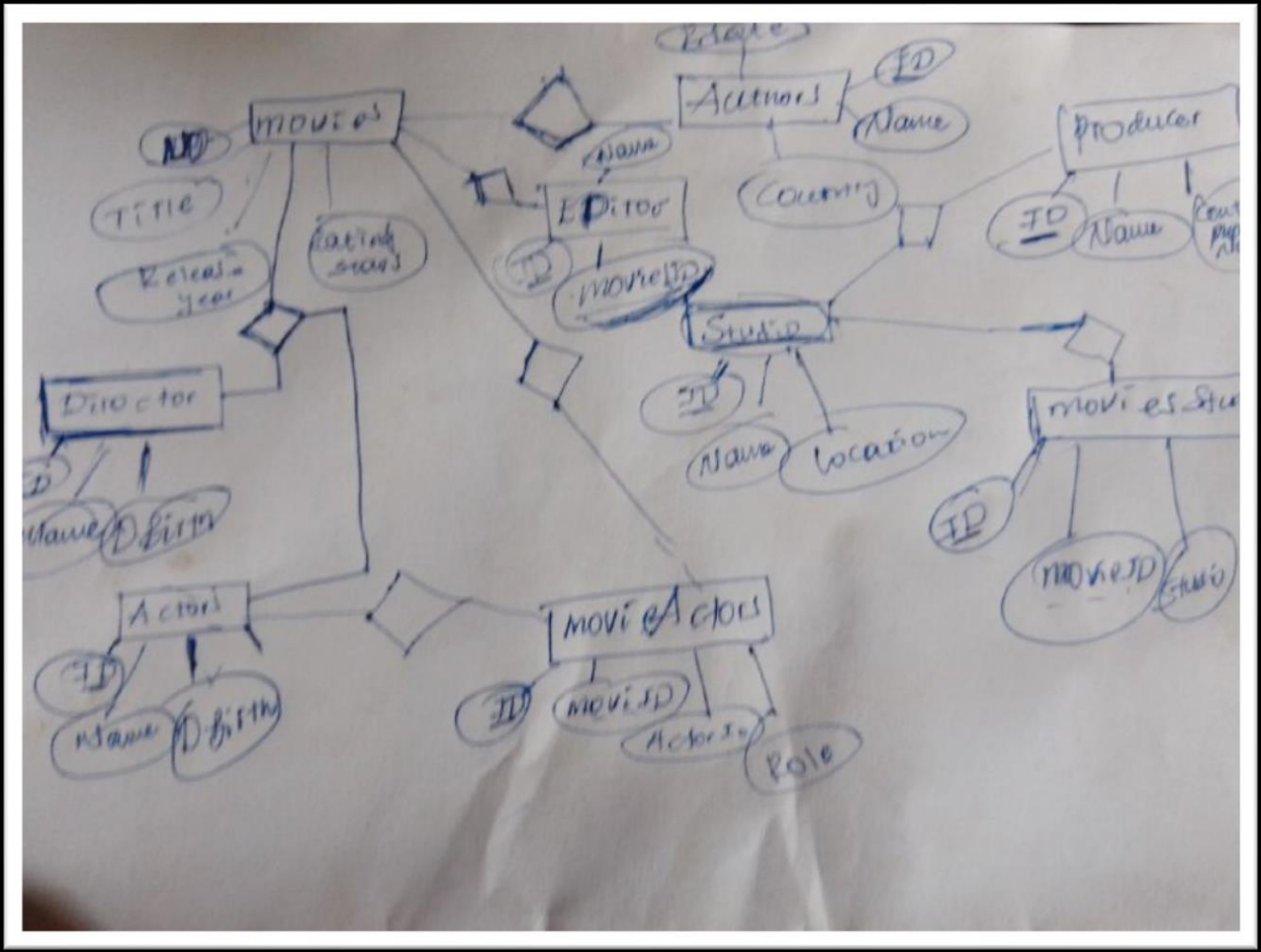
Summary

The Movies Collection Database is a comprehensive and organized platform designed to store, manage, and retrieve detailed information about films. It serves as a centralized repository for data such as titles, genres, release dates, directors, cast members, and audience ratings.

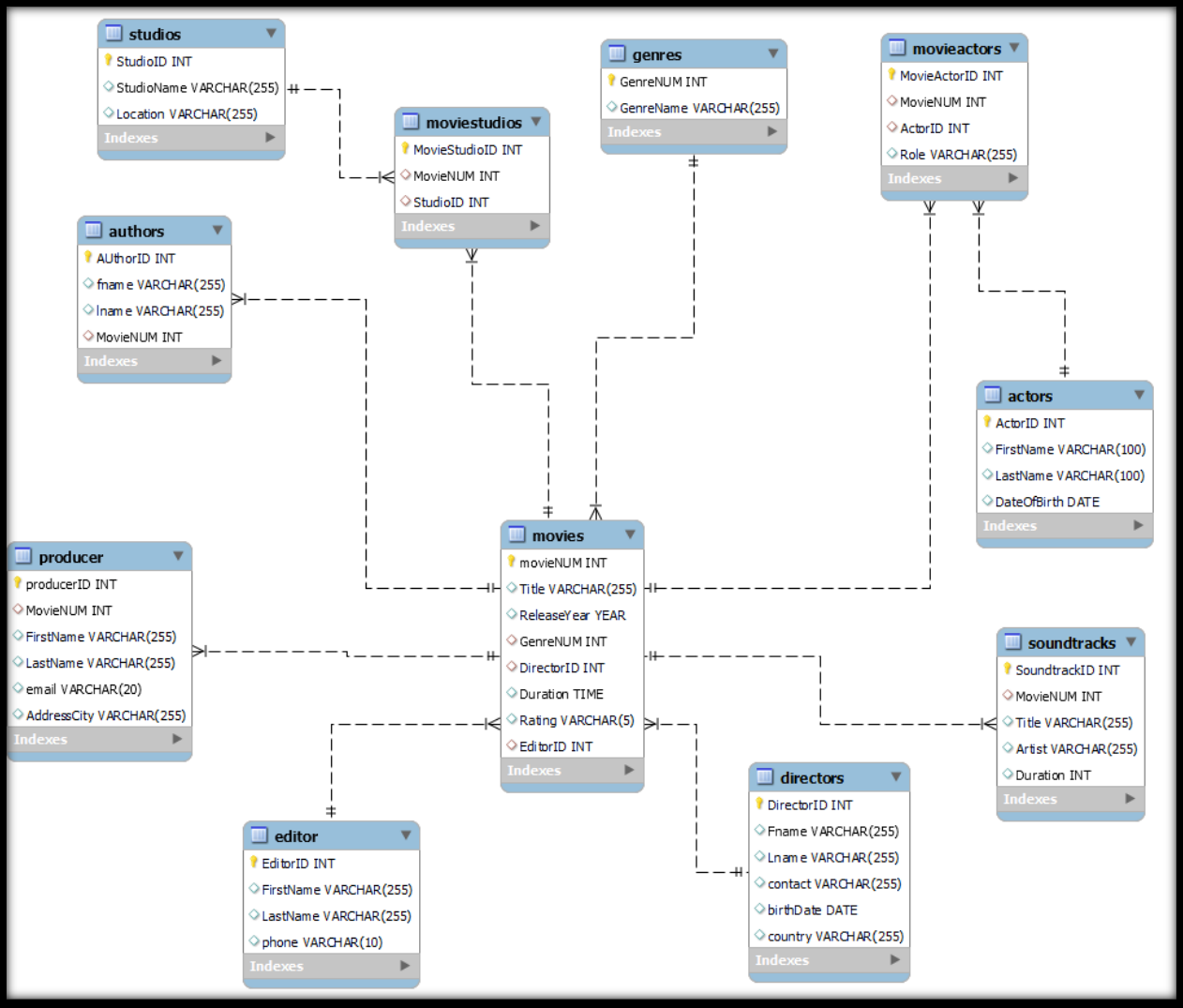
Key features include enhanced searchability, allowing users to filter and find specific films easily; robust collection management tools for tracking and organizing movie inventories; and analytical capabilities to identify trends in the film industry. The database also supports user engagement through features like ratings, reviews, and personalized recommendations.

By ensuring data integrity and providing a historical reference for researchers and enthusiasts, the Movies Collection Database enriches the user experience and promotes a deeper appreciation of cinema. It is an invaluable resource for collectors, libraries, and anyone passionate about films.

ER DIAGRAM



RELATIONAL SCHEMA



--MYSQL CODE IMPLEMENTATION

```
create database MOVIES_COLLECTION;
use MOVIES_COLLECTION;

DROP TABLE IF EXISTS Editors;
DROP TABLE IF EXISTS Actors ;
DROP TABLE IF EXISTS Studios;
DROP TABLE IF EXISTS MovieActors;
DROP TABLE IF EXISTS Studios;
DROP TABLE IF EXISTS MovieStudios;
DROP TABLE IF EXISTS Soundtracks;
DROP TABLE IF EXISTS Producers;
DROP TABLE IF EXISTS Movies;
DROP TABLE IF EXISTS Directors ;
DROP TABLE IF EXISTS Authors;

create table Genres(
```

```
GenreNUM int  auto_increment primary key,
GenreName varchar(255)
);

create table Directors (
DirectorID int auto_increment primary key,
Fname varchar(255) ,
Lname varchar (255),
contact varchar (255),
birthDate date,
country varchar(255)
);

 create table Movies(
 movieNUM int auto_increment primary key,
 Title varchar(255) not null,
 ReleaseYear year,
 GenreNUM int ,
 DirectorID int ,
 Duration time,
 Rating varchar(5),
 foreign key (GenreNUM) references Genres (GenreNUM),
```

```
 foreign key (DirectorID) references Directors (DirectorID)
 );
alter table Movies add column EditorID int;



create table Authors (
AUthorID int auto_increment primary key ,
fname varchar (255) ,
lname varchar (255) ,
MovieNUM int  ,
foreign key (movieNUM)  references Movies (movienum)
);

create table Editor (
EditorID int auto_increment primary key,
FirstName varchar(255),
LastName varchar (255),
phone varchar(10)
);
alter table Editor
drop column movieNUM;
```

```
ALTER TABLE Movies
ADD CONSTRAINT fk_editor
FOREIGN KEY (EditorID)
REFERENCES Editor (EditorID);



create table Producer(
producerID int auto_increment primary key,
MovieNUM int,
FirstName varchar (255),
LastName varchar(255),
email varchar(20),
AddressCity varchar(255),
foreign key (movienum) references Movies (movienum)

);

CREATE TABLE Studios (
   StudioID INT AUTO_INCREMENT PRIMARY KEY,
   StudioName VARCHAR(255),
   Location VARCHAR(255)
);
```

```
CREATE TABLE Actors (
    ActorID INT AUTO_INCREMENT PRIMARY KEY,
    FirstName VARCHAR(100),
    LastName VARCHAR(100),
    DateOfBirth DATE
);

CREATE TABLE MovieActors (
    MovieActorID INT AUTO_INCREMENT PRIMARY KEY,
    MovieNUM INT,
    ActorID INT,
    Role VARCHAR(255),
    FOREIGN KEY (MovieNUM) REFERENCES Movies(MovieNUM),
    FOREIGN KEY (ActorID) REFERENCES Actors(ActorID)
);

CREATE TABLE MovieStudios (
    MovieStudioID INT AUTO_INCREMENT PRIMARY KEY,
    MovieNUM INT,
    StudioID INT,
    FOREIGN KEY (MovieNUM) REFERENCES Movies(MovieNUM),
```

```
    FOREIGN KEY (StudioID) REFERENCES Studios(StudioID)
);

CREATE TABLE Soundtracks (
    SoundtrackID INT AUTO_INCREMENT PRIMARY KEY,
    MovieNUM INT,
    Title VARCHAR(255),
    Artist VARCHAR(255),
    Duration INT,
    FOREIGN KEY (MovieNUM) REFERENCES Movies(MovieNUM)
);
```

--INSERTION THE VALUES FOR EACH TABLE

```
show databases ;
use movies_collection;
show tables;

-- insertiEditorIDng the values in existing table
INSERT INTO ACTORS
VALUES (21,"GARNACHO","TELAVIE","1993-04-23"),
                (01,"LULAZ","BONEY","2000-01-01"),
                (56,"KENT","JOFFERY","2006-8-10"),
```

```
        (90,"NEWTON ","ISAK","1993-01-13"),
        (11,"TOM","ROCK","2001-11-21");


INSERT INTO EDITOR
VALUES (001,"SAMUEL","SHUMI","091006...."),
    (002,"HABTAMU","TADESE","094576...."),
    (003,"DAGIM","BEKELE","091656...."),
    (004,"NATINAEL","BRUK","091896...."),
    (005,"MUSA","JEMAL","091165....");

INSERT INTO DIRECTORS
VALUES (101,"SAMUEL","NEGESE","091006....","ETHIOPIA"),
    (102,"BRATTIAN","STANNIS","024576....","ENGLAND"),
    (103,"ZECHIPING ","YOUNG","031656....","CHINA"),
    (104,"MICHEAL","ERICKSON","011896....","USA"),
    (105,"MUFAS","ODAGU","091165....","ETIOPIA");

 INSERT INTO STUDIOS
 VALUES (200,"A2M PRODUCTION","ADDIS ABABA"),
      (201,"NEFLIM","KERGISTAN"),
      (202,"FIREFOX","FLORIDA"),
```

```
        (203,"AUTOSTUDIO","FLORIDA"),
        (204,"WASEGARECORDS","ADDIS ABABA");

INSERT INTO GENRES
 VALUES (300,"COMEDY"),
        (301,"FICTION"),
        (302,"ADVENTURE"),
        (303,"ACTION"),
        (304,"ADVENTURE");
INSERT INTO movies
VALUES (400,"TIMBER THE WOLF","1990",303,105,"2:35:31","*****",003),
      (401,"THE CHRONICLES","1930",300 ,101,"1:55:47","***",001),
      (402,"CYBER CRACKER","2001",301,102,"2:35:01","**",002),
      (403,"GAME ZONE","2005",302,104,"3:01:31","*****",005),
      (404,"INSTALLER","1960",304,103,"2:56:21","****",004);

INSERT INTO MOVIEACTORS (movieactorID,ROLE)
VALUES (500,"TO ACT LIKE CYBER SECURITY"),
      (501,"ACT LIKE PROTECTOR"),
      (502,"UNKNOWN"),
      (503,"FASTEST RUNNER"),
      (504,"TO ACT LIKE CYBER SECURITY");
```

```
 INSERT INTO PRODUCER
 VALUES (600,402,"NIGATU","DAGNE","nigatu@gmail.com","AYERTENA"),
        (601,401,"DARVISH","ERDOHAN","darvish@gmail.com","stanbul"),
        (602,400,"NIKOLA","KARDATE","NIKO@gmail.com","AMSTERDAM"),
        (603,404,"HABTAMU","DEGAFE","HABTAMu@gmail.com","MEXICO"),
        (604,403,"MEKONIN","DAGNE","moke@gmail.com","AYErtena");
INSERT INTO MOVIESTUDIOS
VALUES (700,404,204),
       (701,402,202),
       (702,403,201),
       (703,400,200),
       (705,401,203);
INSERT INTO SOUNDTRACKS
VALUES (800,404,"HEART_SOFTING","UNKNOWN","1:02:00"),
      (801,402,"GAMING_MUSIC","UNKNOWN","00:30:00"),
      (802,401,"NATURAL_HEAT","UNKNOWN","2:00:00"),
      (803,403,"THE RAPPED BOY","UNKNOWN","3:02:00"),
      (804,400,"LYRICSCOLLECTION","UNKNOWN","2:32:00");

INSERT INTO AUTHORS
VALUES (900,"LEONARDO","COVIN",403),
```

```
(901,"JOHN","KAEVEN",401),
(902,"LOLAZ","FRERICK",402),
(903,"ERICK","JOSEF",404),
(904,"MICHEAL","BRANDON",400);
```

--- DATA FETCHING FROM EXSISTING TABLES

```
use movies_collection;
show tables;
-- data fetching for individual tables
select * from actors;
select * from authors;
select * from directors;
select * from editor;
select * from genres;
select * from movieactors;
select * from movies;
select * from moviestudios;
select * from producer;
select * from soundtracks;
select *from studios;

-- RELATED DATA FETCHING BY USING INNER JOIN
select *
from authors as au inner join movies as m
on au.MovieNUM=m.movieNUM;

select AU.AUthorID,AU.FNAME,AU.LNAME,M.TITLE
from authors as au INNER join movies as M
ON AU.MOVIENUM=M.MOVIENUM;


-- RIGHT JOIN
select AU.AUthorID,AU.FNAME,AU.LNAME,M.TITLE
from authors as au RIGHT join movies as M
```

```
ON AU.MOVIENUM=M.MOVIENUM;


-- LEFT JOIN

select AU.AUthorID,AU.FNAME,AU.LNAME,M.TITLE
from authors as au LEFT join movies as M
ON AU.MOVIENUM=M.MOVIENUM;

select *
from movies as m inner join directors as d
on m.DirectorID=d.DirectorID;

select m.title,m.releaseyear,m.duration,m.rating,
d.directorid,d.fname,d.lname,d.contact
from movies as m inner join directors as d
on m.DirectorID=d.DirectorID
where Duration > "01:35:01";
```

To implement the code please use the statements individually !!!!