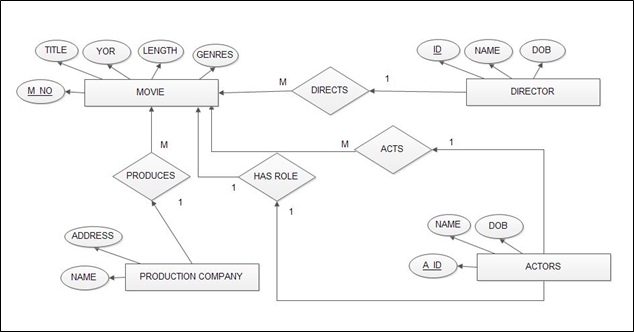
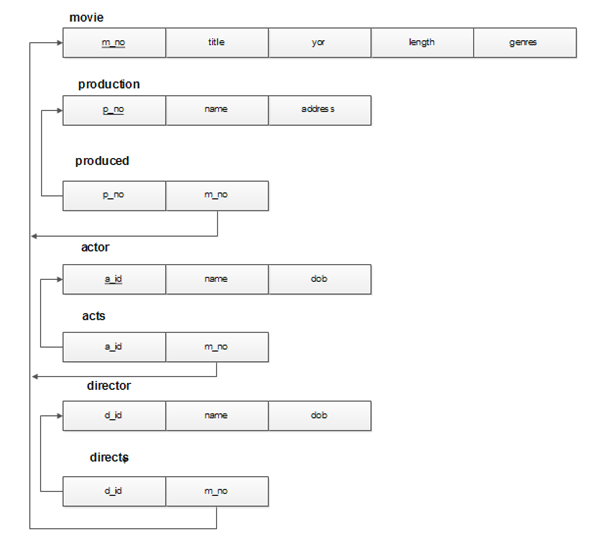
**Exercise 3**

Data requirements of the movie industry are captured. Each movie is identified by title and year of release. Each movie has length in minutes and is classified under one genre (like action, horror etc.). Each movie has a plot outline. Production companies are identified by name and each has an address. A production company produces one or more movies. Actors are identified by id. Other details like name and date of birth of actors are also stored. Each actor acts in one or more movies. Each actor has a role in a movie. Directors are identified by id. Other details like name and date of birth of directors are also stored. Each director directs one or more movies. Each movie has one or more actors and one or more directors and is produced by a production company.

ER-Diagram:



Relational schema:



Tables

-- Movie Table

CREATE TABLE movie (

m\_no INT PRIMARY KEY,

title VARCHAR(20),

yor INT,

length INT,

genres VARCHAR(10)

);

-- Production Table

CREATE TABLE production (

p\_no INT PRIMARY KEY,

name VARCHAR(20),

address VARCHAR(20)

);

-- Produced Table

CREATE TABLE produced (

p\_no INT,

m\_no INT,

FOREIGN KEY(p\_no) REFERENCES production(p\_no),

FOREIGN KEY(m\_no) REFERENCES movie(m\_no)

);

-- Actor Table

CREATE TABLE actor (

a\_id INT PRIMARY KEY,

name VARCHAR(30),

dob DATE

);

-- Acts Table

CREATE TABLE acts (

a\_id INT,

m\_no INT,

FOREIGN KEY(a\_id) REFERENCES actor(a\_id),

FOREIGN KEY(m\_no) REFERENCES movie(m\_no)

);

-- Director Table

CREATE TABLE director (

d\_id INT PRIMARY KEY,

name VARCHAR(20),

dob DATE

);

-- Directs Table

CREATE TABLE directs (

d\_id INT,

m\_no INT,

Primary key(d\_id,m\_no),

FOREIGN KEY(d\_id) REFERENCES director(d\_id),

FOREIGN KEY(m\_no) REFERENCES movie(m\_no)

);

-- Insert queries for the given tables' data

-- Inserting data into `director` table

INSERT INTO director (d\_id, name, dob) VALUES

(302, 'S Krishna', '1983-04-06'),

(303, 'Santhosh Ananddram', '1986-01-08'),

(304, 'Tharun Sudhir', '1986-12-28'),

(305, 'Chethan Kumar', '1988-12-07');

INSERT INTO movie (m\_no, title, yor, length, genres) VALUES

(1, 'KGF: Chapter 2', 2022, 180, 'action'),

(2, 'Pailwaan', 2019, 155, 'sports'),

(3, 'Yuvarathnaa', 2021, 160, 'drama'),

(4, 'Roberrt', 2021, 145, 'action'),

(5, 'James', 2022, 150, 'thriller'),

(6, 'Horror Movie 1', 2012, 120, 'horror'),

(7, 'Horror Movie 2', 2012, 110, 'horror'),

(8,'Yash',2008,156,'comedy');

-- Inserting data into `directs` table

INSERT INTO directs (d\_id, m\_no) VALUES

(302, 2),

(303, 3),

(304, 4),

(305, 5),

(303,6),(304,6),

(302,7),(305,7),(303,8);

-- Inserting data into `actor` table

INSERT INTO actor (a\_id, name, dob) VALUES

(201, 'Yash', '1986-01-08'),

(202, 'Sudeep', '1973-09-02'),

(203, 'Darshan', '1976-10-16'),

(204, 'Puneeth Rajkumar', '1975-03-17'),

(205, 'Rakshit Shetty', '1983-06-06');

-- Inserting data into `acts` table

INSERT INTO acts (a\_id, m\_no) VALUES

(201, 1),

(202, 1),

(203, 2),

(201, 1),

(202, 1),

(203, 2),

(201,8);

-- Inserting data into `movie` table

-- Inserting data into `production` table

INSERT INTO production (p\_no, name, address) VALUES

(101, 'Hombale Films', 'Bangalore'),

(102, 'PRK Productions', 'Bangalore'),

(103, 'Umapathy Films', 'Mysore');

-- Inserting data into `produced` table

INSERT INTO produced (p\_no, m\_no) VALUES

(101, 1),

(102, 2),

(103, 3),

(103, 4),

(101, 5);

Queries:

a)List the details of horror movies released in 2012 and directed by more than 2 directors.

mysql> SELECT m.\* FROM movie m JOIN directs d ON m.m\_no = d.m\_no WHERE m.genres = 'horror' AND m.yor = 2012 GROUP BY m.m\_no HAVING COUNT(d.d\_id) >= 2;

b) List the details of actors who acted in movies having the same titles but released before 2000 and after 2010.

mysql> select a.name from actor a, acts ac, movie m where a.a\_id=ac.a\_id and ac.m\_no=m.m\_no and m.yor between 2000 and 2010 and a.name=m.title;

c) List the details of production companies producing maximum movies.

mysql> SELECT p.name, COUNT(\*) AS movie\_count

FROM production p

JOIN produced pr ON p.p\_no = pr.p\_no

GROUP BY p.name

HAVING movie\_count = (

SELECT MAX(movie\_count)

FROM (

SELECT COUNT(\*) AS movie\_count

FROM produced

GROUP BY p\_no

) AS movie\_counts

);

d) List the details of movies where director and actor have the same date of birth.

mysql> select m.title,m.m\_no from movie m, acts ac, actor a, directs ds, director d where m.m\_no=ac.m\_no and m.m\_no=ds.m\_no and ac.a\_id=a.a\_id and ds.d\_id=d.d\_id and d.dob=a.dob;

e) Retrieve the names of directors directed to all the movies produced by any one production company.

mysql> select p.name from movie m, production p, produced pd, directs ds, director d where d.d\_id=ds.d\_id and ds.m\_no=m.m\_no and m.m\_no=pd.m\_no and pd.p\_no=p.p\_no group by p.name having count(d.d\_id)=1;