SYMBIOSIS INSTITUTE OF TECHNOLOGY

DBMS Project Report on

Movie and T.V. Series Recommendation System

SUBMITTED BY

DIVILI SATHVIK

18070124025

ARIPIRALA SAI SRIVATHSAV

18070124014

MAYANK ARORA

18070124040

PRANAV SHARMA

18070124051

Under the Guidance of

Prof. Shruti Patil

DEPARTMENT OF INFORMATION TECHNOLOGY

SYMBIOSIS INSTITUTE OF TECHNOLOGY, PUNE

PUNE-412115

2020-21

INDEX

1. Introduction

This is a report for the project "**Movie and T.V. series recommendation system**" which consists of problem we are trying to solve along with the solution, functional requirements, and all other initially essential details of the project. This project is an implementation model of movie recommendation system's database in MySQL.

2. Problem Statement, System Architecture, Functional Requirements

a. Problem Statement

With the rapid development and advancement of digital world especially in the field of media and entertainment, a lot of content to watch online is being accumulated on various streaming platforms like OTTs etc. This availability of huge amount of content, now turned out into a boon and bane situation because though it provides users with various choices and options to get entertained on the other hand it also results in few practical problems like:

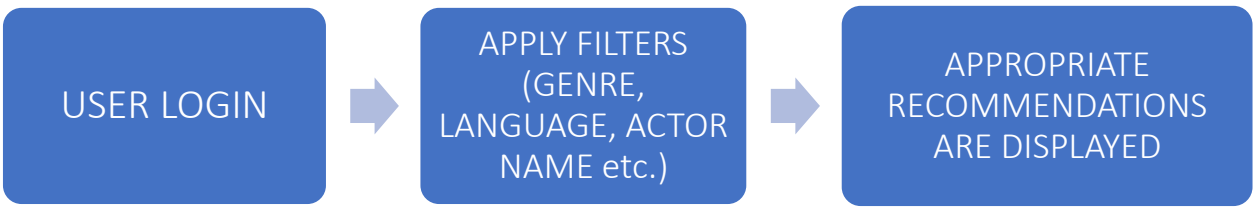
- User gets perplexed with more than enough choices in-hand.
- Finding the right OTT platform for a movie/T.V. series he wishes to watch becomes difficult.
- Lacks the right suggestion which suits his/her taste.

b. Objectives

So, the project "Movie and T.V. series recommendation system" is a solution that helps the user to get recommended with a proper movie or T.V. series based upon various parameters like genre, actor/actress, rating, languages etc. and help the user to know on which OTT platform the respective movie/ T.V. series is available. This helps the user to know the list of movies that perfectly matches his choices thus reducing the chance of getting confused and intimating where to watch it.

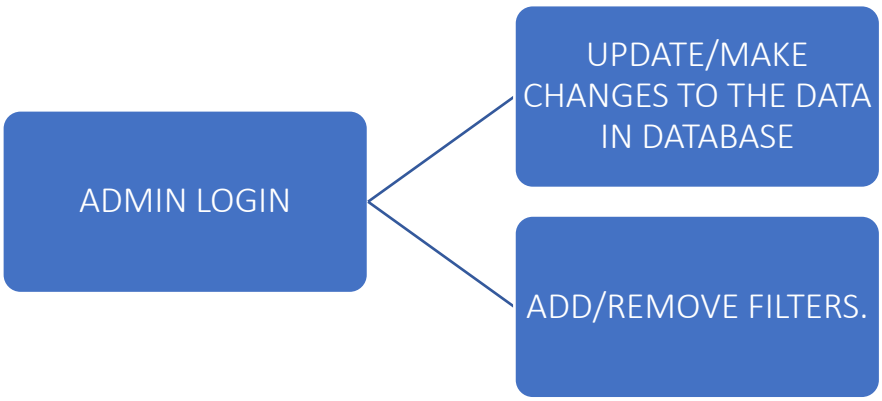
The System's Architecture/Modules would be as follows:

USER MODULE



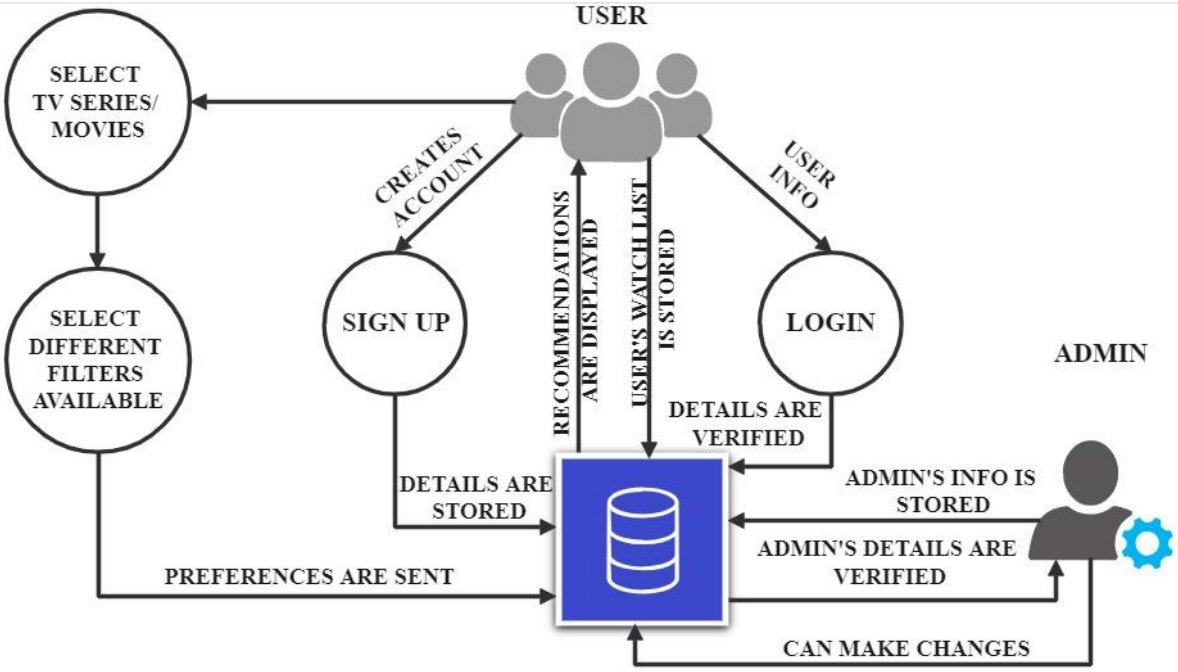
- Users need to login with his unique id and password.
- User can make use of various filters that are available, based upon his choices and can get appropriate recommendations.

ADMIN MODULE



- Admin needs to login with his unique id and password
- Admin can make changes to the database and can make changes to the filters (add new ones or delete few) that are available for users to use.

c. System Architecture



d. Functionalities

- Every user should sign up using his/her unique id and password.
- Initially user is provided with 2 options to select between movie and T.V. series which are under entertainment title.

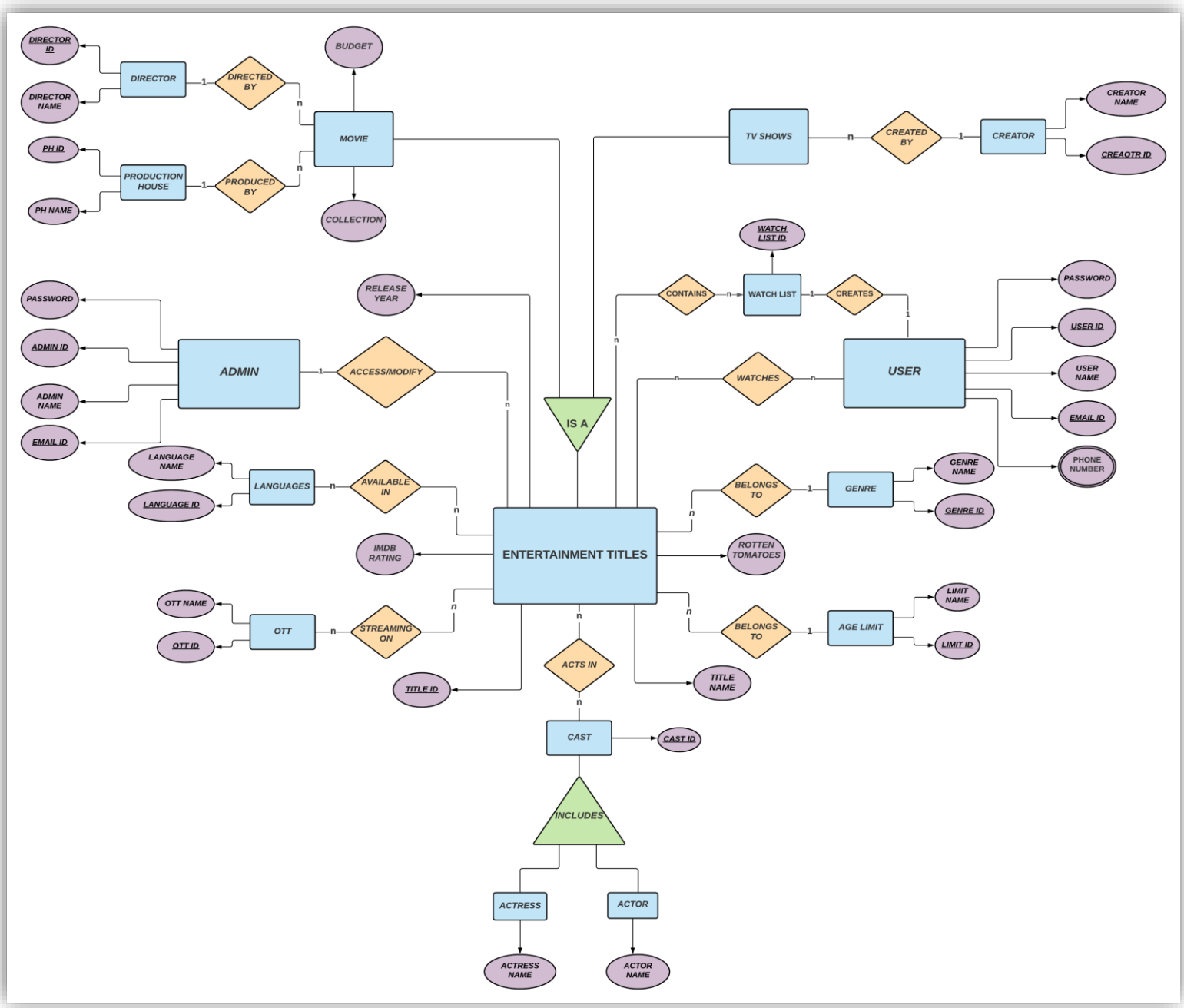
- After selecting one of the options, respective filters which are available under movies and T.V. series are displayed to choose between them.
- Available filters for movies are name of the actor, actress, director, genre, age limit, ratings, release year, language, budget, box-office collection, streaming platform and production house.
- Available filters for T.V. shows are name of the actor, actress, genre, age limit, ratings, release year, language, streaming platform, creator of the show.
- User can also maintain a common list of all the movies and T.V. shows which he had already watched.
- Admin should also sign up using his/her unique id and password.
- Admin has the access to both movies and T.V. series to modify them.

3. Entities, Attributes, and their Relationships.

ENTITES	ATTRIBUTES
Entertainment Titles (ET)	titleName, titleID, imdbRating, rottenTomatoesRating, Release year,
Movie	budget, collection
TVshows	-
Cast	castID
Actor	actorName
Actress	actressName
OTT	ottName, ottID
AgeLimit	limitName, limitID
Genre	genreName, genreID
Language	languageName, languageID
Admin	adminName, adminID, emailID, password
User	userName, userID, emailID, password, PhoneNumber
Watchlist	WatchID
Creator	creatorName, creatorID
Director	directorName, directorID
ProductionHouse	phName, phID

RELATIONSHIP	CARDINALITY
ET ***is a*** Movie.	-
ET ***is a*** TV show.	-
Cast ***acts in*** ET.	Many to many
Cast ***includes*** Actor.	-
Cast ***includes*** Actress.	-
ET ***has*** Age Limit.	Many to one
ET ***Streaming on*** OTT.	Many to many
ET ***belongs to a*** Genre.	Many to one
ET ***Available in*** Languages.	Many to many
Admin ***modifies*** ET.	One to many
User ***watches*** ET.	Many to many
User ***Creates*** Watchlist.	One to one
Watchlist ***Contains*** ET.	Many to many
TV Shows ***created by*** Creator.	Many to one
Movie ***directed by*** Director.	Many to one
Movie ***produced by*** Production House.	Many to one

4. ER (EER) Diagram



5. Relational Schema

a. Title

titleID	titleName	rottenTomatoesRating	imdbRating	releaseYear
genreID	*limitID*	*adminID*		

b. Movie

titleID	budget	collection	*directorID*	*phID*

c. TVshows

titleID	*creatorID*

d. Genre

genreID	genreName

e. AgeLimit

limitID	limitName

f. OTT

ottID	ottName

g. Languages

languageID	languageName

h. Cast

castID	castName

i. Director

directorID	directorName

j. ProductionHouse

phID	phName

k. Creator

creatorID	creatorName

l. User

userID	userName	password	emailID

m. PhoneNumber

phoneID	*userID*	phone

n. Admin

adminID	adminName	password	emailID

o. ActsIn

actingID	*titleID*	*castID*

p. StreamingOn

streamID	*titleID*	*ottID*

q. AvailableIn

availID	*titleID*	*languageID*

r. Watches

watchID	*titleID*	*userID*

s. Watchlist

listID	*userID*

t. Contain

containID	*titleID*	*listID*

6. Keys in each Relation

a. Title
- i. Candidate Key: titleID, titleName
- ii. Primary Key: titleID
- iii. Foreign Key: genreID, limitID, adminID
- iv. Alternate Key: titleName

b. Movie
- i. Candidate Key: titleID
- ii. Primary Key: titleID
- iii. Foreign Key: directorID, phID
- iv. Alternate Key: NULL

c. TV Show
- i. Candidate Key: titleID
- ii. Primary Key: titleID
- iii. Foreign Key: creatorID
- iv. Alternate Key: NULL

d. Genre
- i. Candidate Key: genreID, genreName
- ii. Primary Key: genreID
- iii. Foreign Key: NULL
- iv. Alternate Key: genreName

e. AgeLimit
   i. Candidate Key: limitID, limitName
   ii. Primary Key: limitID
   iii. Foreign Key: NULL
   iv. Alternate Key: limitName

f. OTT
   i. Candidate Key: ottID, ottName
   ii. Primary Key: ottID
   iii. Foreign Key: NULL
   iv. Alternate Key: ottName

g. Language
   i. Candidate Key: languageID, languageName
   ii. Primary Key: languageID
   iii. Foreign Key: NULL
   iv. Alternate Key: languageName

h. Actor
   i. Candidate Key: castID
   ii. Primary Key: castID
   iii. Foreign Key: NULL
   iv. Alternate Key: NULL

i. Director
   i. Candidate Key: directorID
   ii. Primary Key: directorID
   iii. Foreign Key: NULL
   iv. Alternate Key: NULL

j. ProductionHouse
   i. Candidate Key: phID, phName
   ii. Primary Key: phID
   iii. Foreign Key: NULL
   iv. Alternate Key: phName

k. Creator
   i. Candidate Key: creatorID
   ii. Primary Key: creatorID
   iii. Foreign Key: NULL
   iv. Alternate Key: NULL

l. User
   i. Candidate Key: userID, emailID
   ii. Primary Key: userID
   iii. Foreign Key: NULL
   iv. Alternate Key: emailID

m. PhoneNumber
   i. Candidate Key: phoneID, phone
   ii. Primary Key: phoneID
   iii. Foreign Key: userID
   iv. Alternate Key: phone

n. Admin
   i. Candidate Key: adminID, emailID

ii. Primary Key: adminID
iii. Foreign Key: NULL
iv. Alternate Key: emailID

o. ActsIn
i. Candidate Key: actingID
ii. Primary Key: actingID
iii. Foreign Key: titleID, castID
iv. Alternate Key: NULL

p. StreamingOn
i. Candidate Key: streamID
ii. Primary Key: streamID
iii. Foreign Key: titleID, ottID
iv. Alternate Key: NULL

q. AvailableIn
i. Candidate Key: availID
ii. Primary Key: availID
iii. Foreign Key: titleID, languageID
iv. Alternate Key: NULL

r. Watches
i. Candidate Key: watchID
ii. Primary Key: watchID
iii. Foreign Key: titleID, userID
iv. Alternate Key: NULL

s. Watchlist
i. Candidate Key: listID
ii. Primary Key: listID
iii. Foreign Key: userID
iv. Alternate Key: NULL

t. Contains
i. Candidate Key: containID
ii. Primary Key: containID
iii. Foreign Key: titleID, listID
iv. Alternate Key: NULL

7. Codd's Rules

Rule 0:

Definition: This rule states that for a system to qualify as an **RDBMS**, it must be able to manage database entirely through the relational capabilities.

Justification: Yes, this the primary rule for all databases to satisfy and this database is also able to manage entirely through its relational capabilities. Hence this rule is applied.

Rule 1: Information rule:

Definition: All information (including metadata) is to be represented as stored data in cells of tables. The rows and columns have to be strictly unordered.

Justification: This database has all the data in the form of rows and columns, also the data in each row is unique, every table has its own and unique value called primary key, each cell contains only one value and the order of rows and columns doesn't affect the meaning of the tables. Hence this rule is applied.

Rule 2: Guaranteed Access:

Definition: Each unique piece of data (atomic value) should be accessible by: **Table Name + Primary Key (Row) + Attribute(column)**.

NOTE: Ability to directly access via POINTER is a violation of this rule.

Justification: Since each table in this database has its own primary key it is possible to access every atomic value logically without using any pointer or physical address of the data. Hence this rule is applied.

Rule 3: Systematic treatment of NULL:

Definition: Null has several meanings, it can mean missing data, not applicable or no value. It should be handled consistently. Also, Primary key must not be null, ever. Expression on NULL must give null.

Justification: This database will not have any null value (missing value, unknown value or unacceptable value) and most importantly any primary key will not be null. Hence this rule is applied.

Rule 4: Active Online Catalog:

Definition: Database dictionary(catalogue) is the structure description of the complete **Database** and it must be stored online. The Catalogue must be governed by same rules as rest of the database. The same query language should be used on catalogue as used to query database.

Justification: As this database do not have any database dictionary this rule is not applied.

Rule 5: Powerful and Well-Structured Language:

Definition: One well-structured language must be there to provide all manners of access to the data stored in the database. Example: **SQL**, **QBE** etc. If the database allows access to the data without the use of this language, then that is a violation.

Justification: Since our database accepts SQL (which is one of SQL/QBE) to access, modify and manipulate this rule is also applied.

Rule 6: View Updation Rule:

Definition: All the view that are theoretically updatable should be updatable by the system as well.

Justification: The view that is given in the database to the user is also in the form of tables only and not any other alternative virtual way. Hence this rule is not applied.

Rule 7: Relational Level Operation:

Definition: There must be Insert, Delete, Update operations at each level of relations. Set operation like Union, Intersection and minus should also be supported.

Justification: This database supports all the set operations and relational algebra which include join, union, intersection, minus, delete, insert etc. Hence this rule is also applied.

Rule 8: Physical Data Independence:

Definition: The physical storage of data should not matter to the system. If say, some file supporting table is renamed or moved from one disk to another, it should not affect the application.

Justification: This database and its data access methods will not be affected with the change in physical storage of data hence it satisfies the rule of physical data independence.

Rule 9: Logical Data Independence:

Definition: If there is change in the logical structure (table structures) of the database the user view of data should not change. Say, if a table is split into two tables, a new view should give result as the join of the two tables. This rule is most difficult to satisfy.

Justification: This rule is not applied by this database.

Rule 10: Integrity Independence:

Definition: The database should be able to enforce its own integrity rather than using other programs. Key and Check constraints, trigger etc, should be stored in Data Dictionary. This also make **RDBMS** independent of front-end.

Justification: Irrespective of the front-end, database has its own integrity and key values are stored in data dictionary. Hence this rule is applied.

Rule 11: Distribution Independence:

Definition: A database should work properly regardless of its distribution across a network. Even if a database is geographically distributed, with data stored in pieces, the end user should get an impression that it is stored at the same place. This lays the foundation of **distributed database**.

Justification: As we are not sure about the distribution of the database and how it could affect the data manipulation, we are not sure this rule could be applied or not. Hence considering this rule is not applied

Rule 12: Non-Subversion Rule:

Definition: If low level access is allowed to a system it should not be able to subvert or bypass integrity rules to change the data. This can be achieved by some sort of looking or encryption.

Justification: Since this database is made in MySQL and it forms an instant of the tables while we view data there will be no changes applied to the actual tables and data which

will not allow to circumvent data integrity and security at low level access. Hence this rule is applied.

8. Anomalies

a. Insertion: -

i. A new title cannot be added without a valid genreID, limitID, adminID, directorID, creatorID and phID.
ii. A new watchlist cannot be created without a valid userID

b. Updation: -

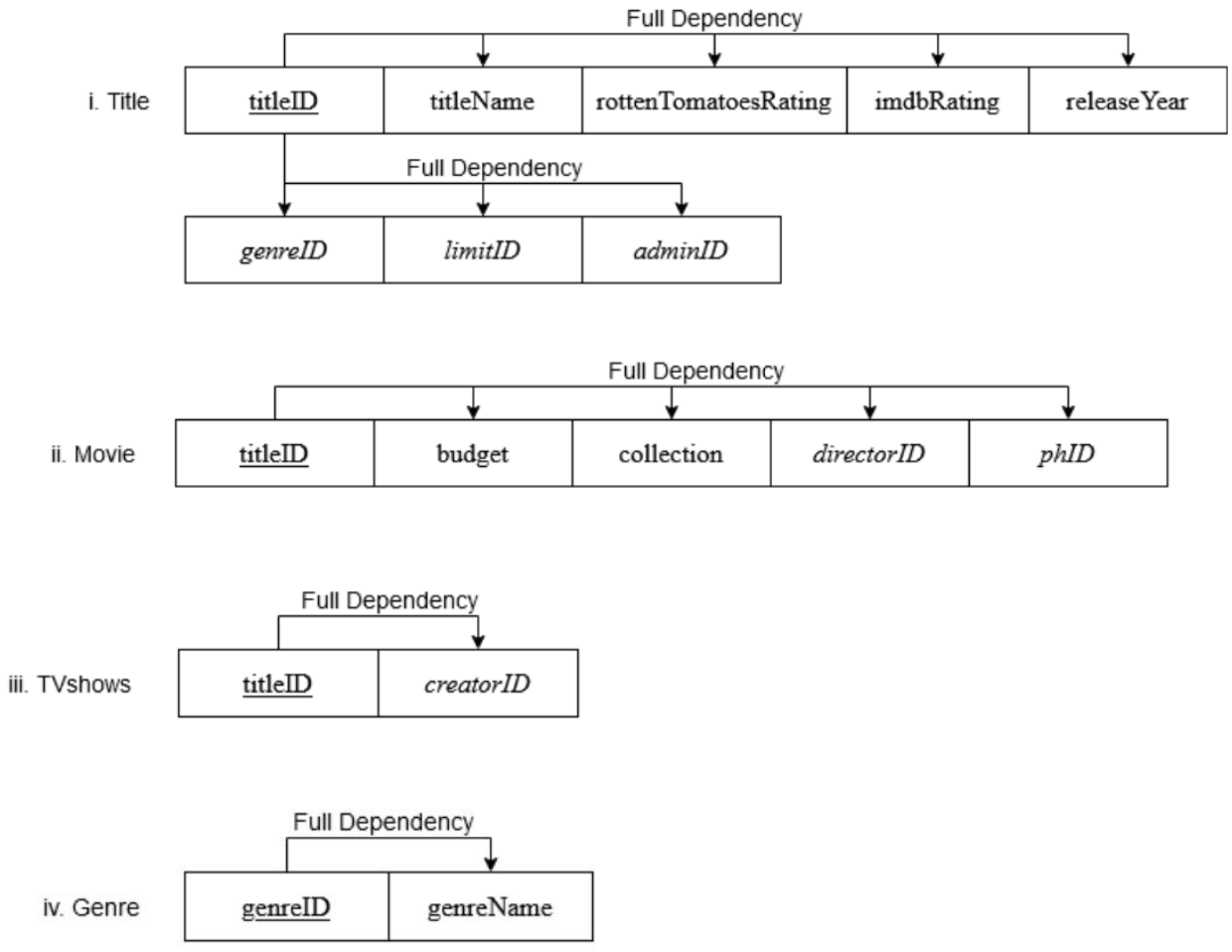
i. A title cannot be updated when the genreID, limitID, adminID, directorID, creatorID and phID is invalid.
ii. A watchlist cannot be updated so it contains an invalid userID.

c. Deletion: -

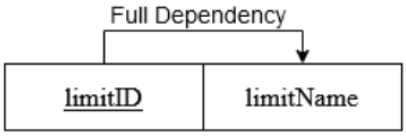
i. A user cannot be deleted if he/she has a non-empty watchlist.

9. Functional Dependencies

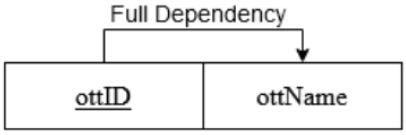
a. Functional Dependencies



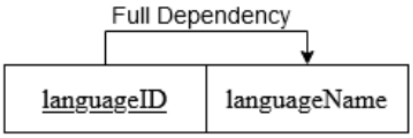
v. AgeLimit



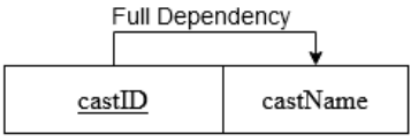
vi. OTT



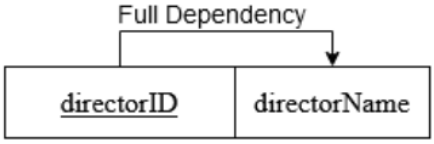
vii. Language



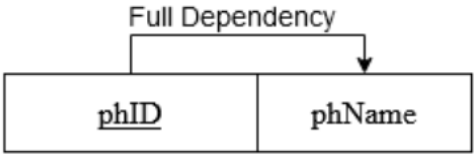
viii. Cast



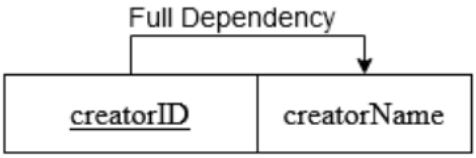
ix. Director



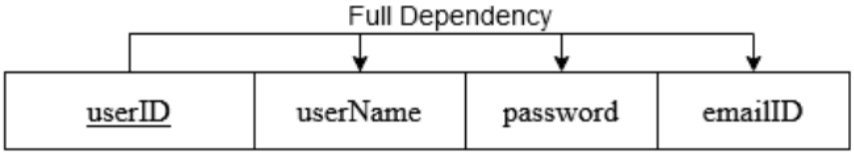
x. ProductionHouse



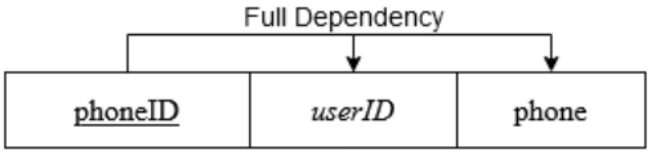
xi.Creator



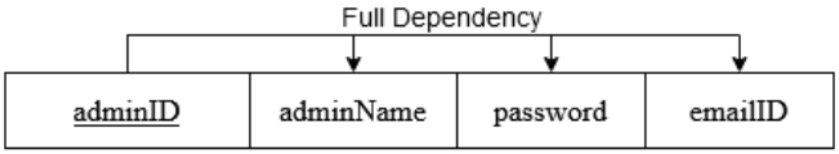
xii. User



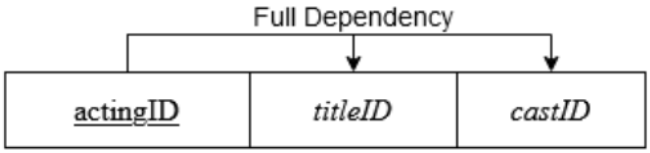
xiii. PhoneNumber



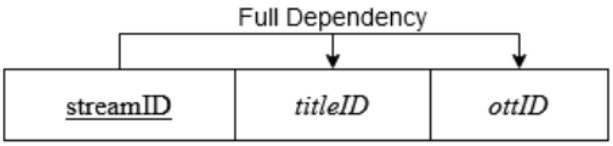
xiv. Admin



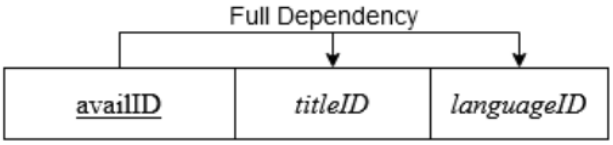
xv. ActsIn



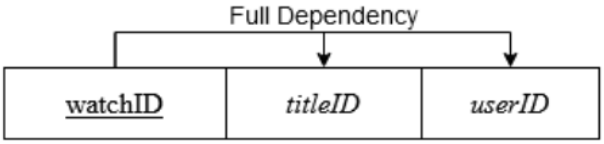
xvi. StreamingOn



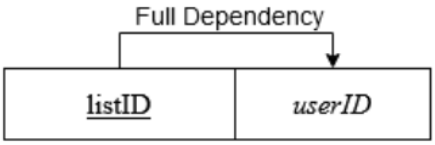
xvii. AvailableIn



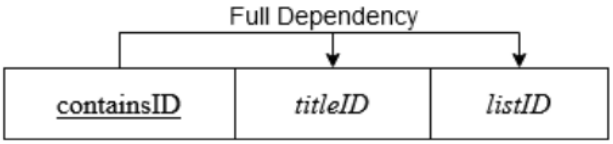
xviii. Watches



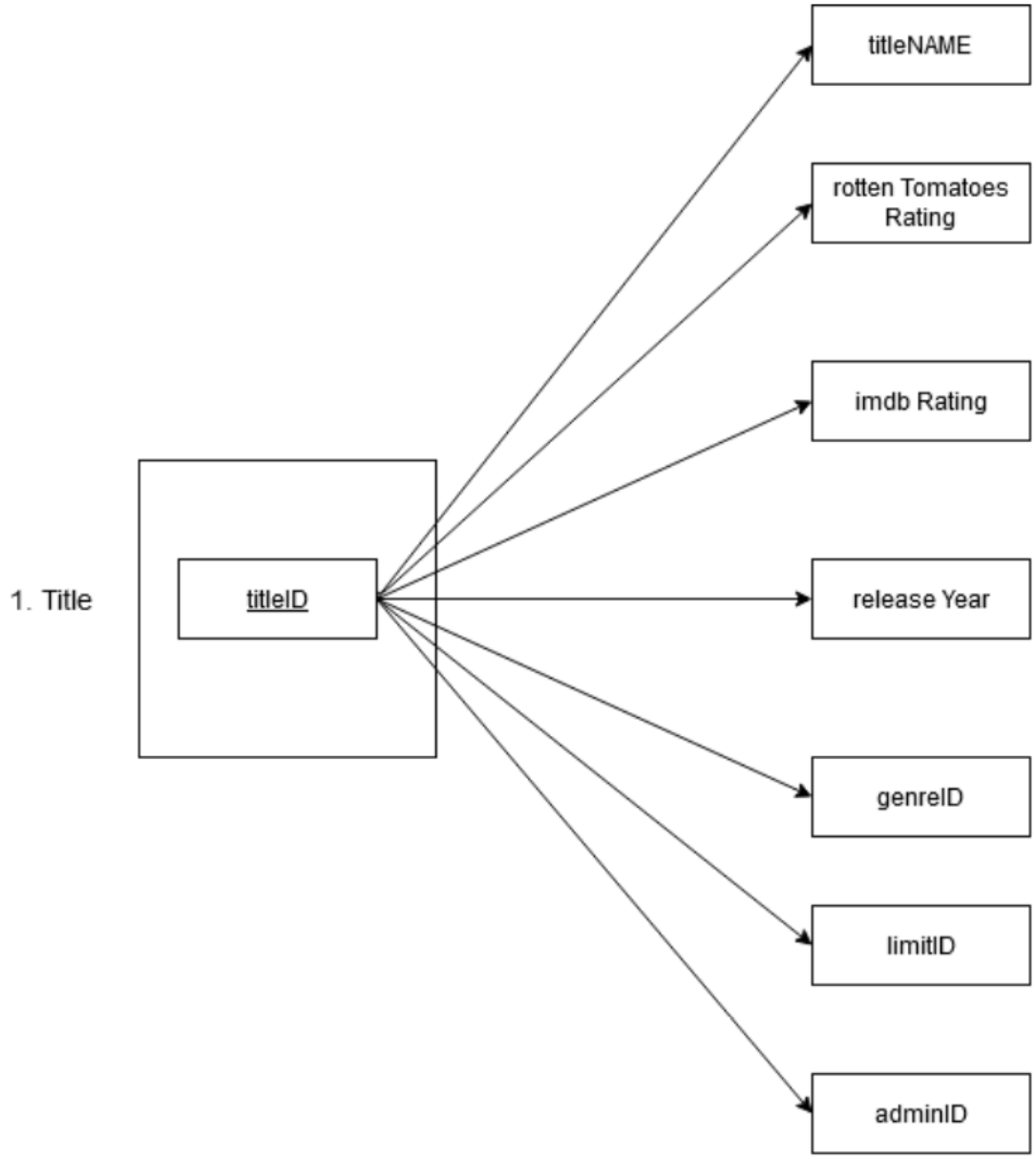
xix. WatchList

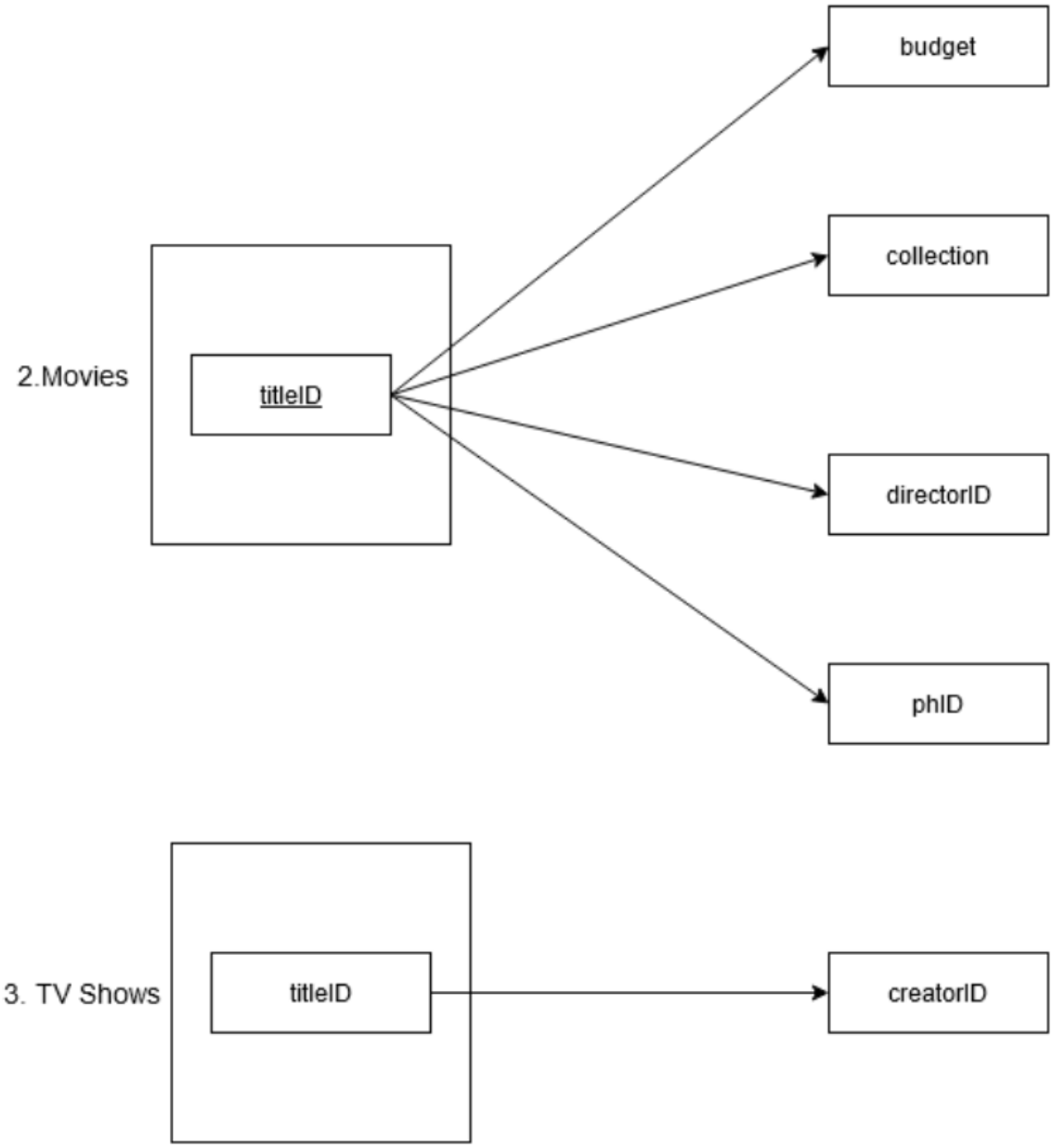


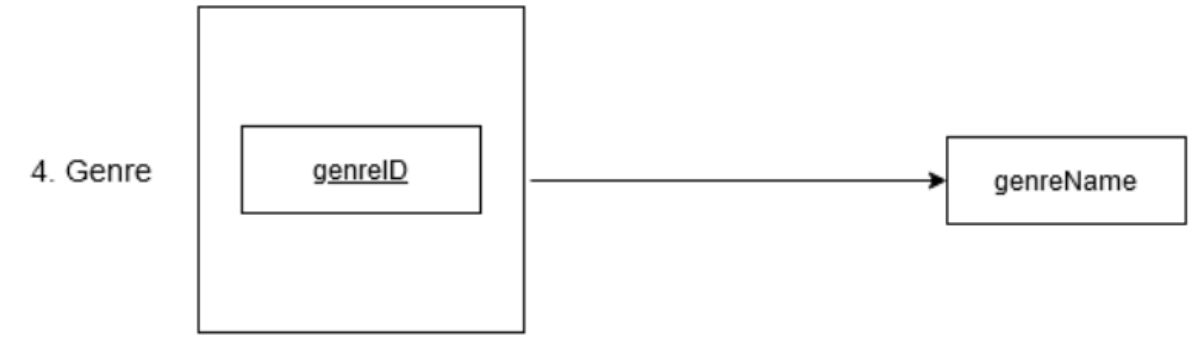
xx. Contains

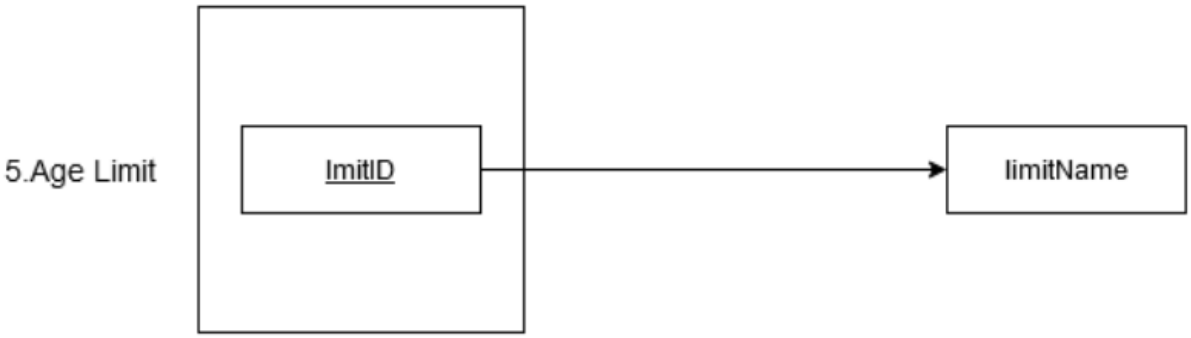


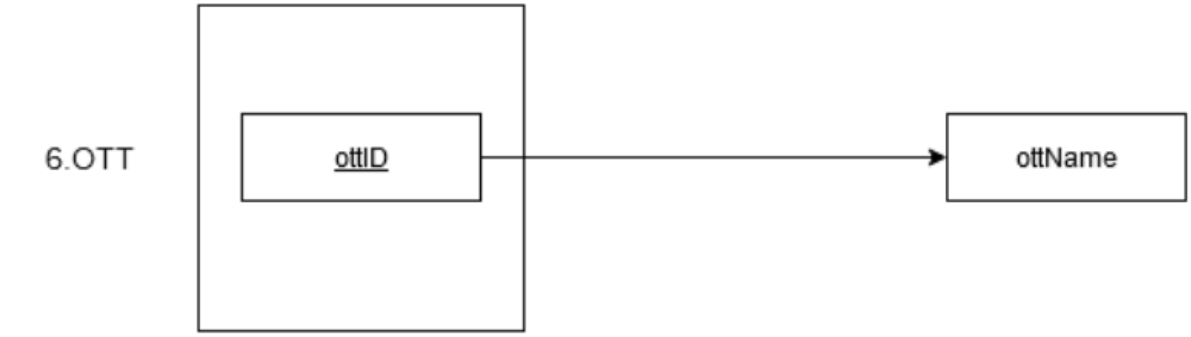
b. Functional Dependency Chart

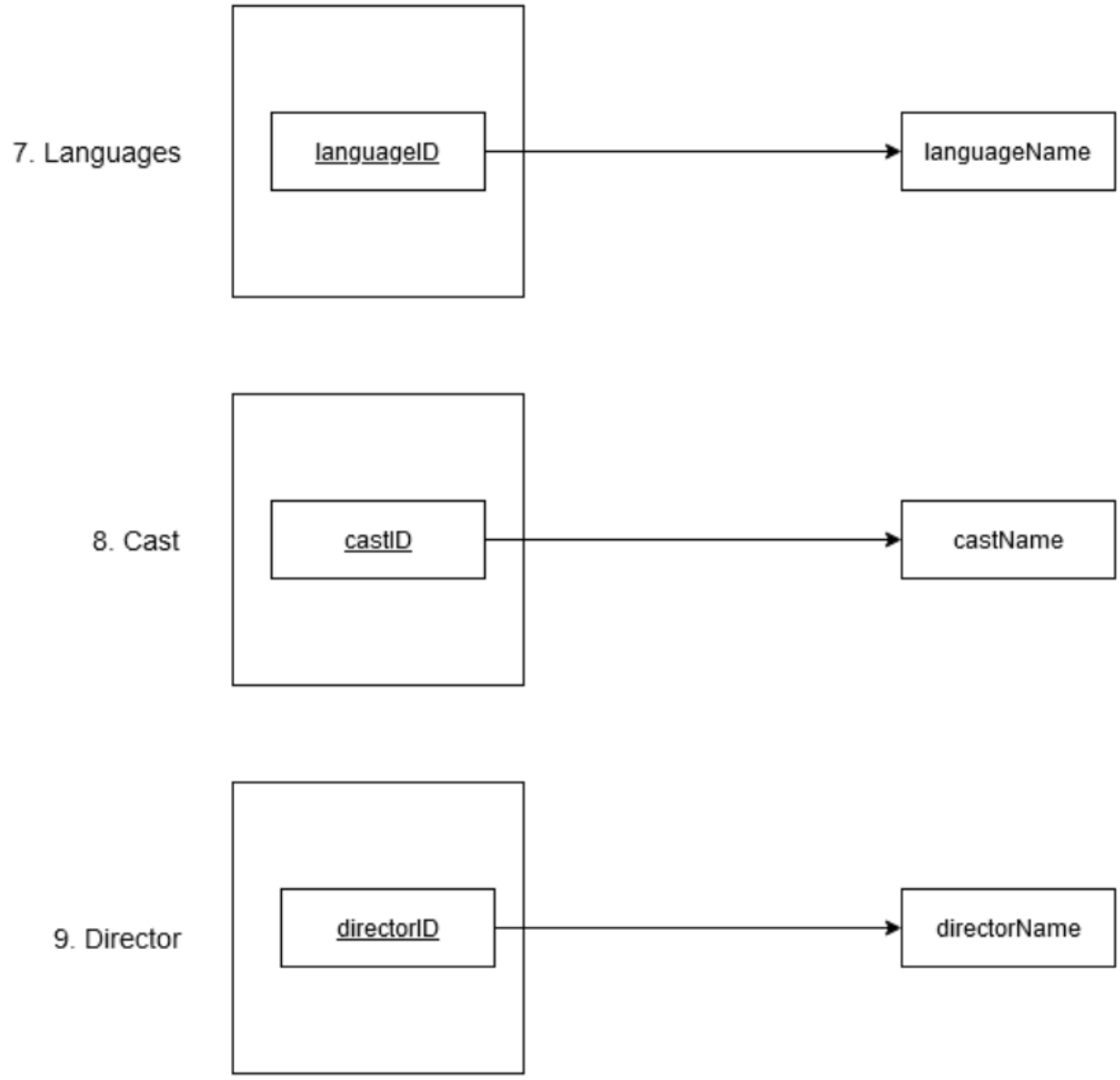


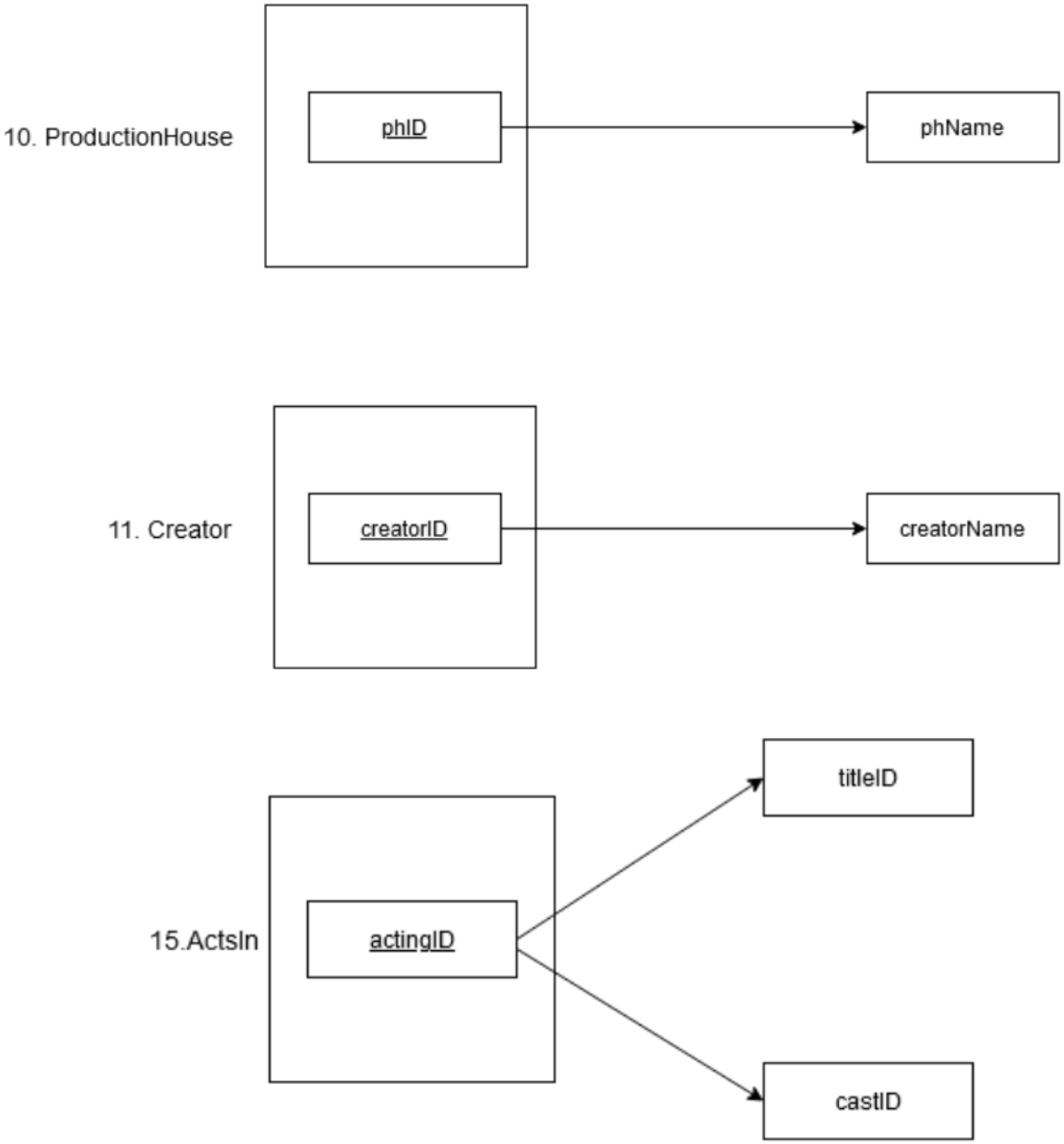

budget
collection
2.Movies
titleID
directorID
phID
3. TV Shows
titleID
creatorID


4. Genre
genreID
genreName

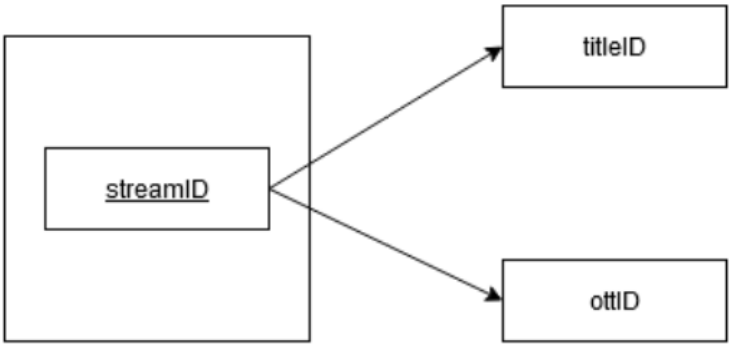

5.Age Limit
limitID
limitName


6.OTT
ottID
ottName

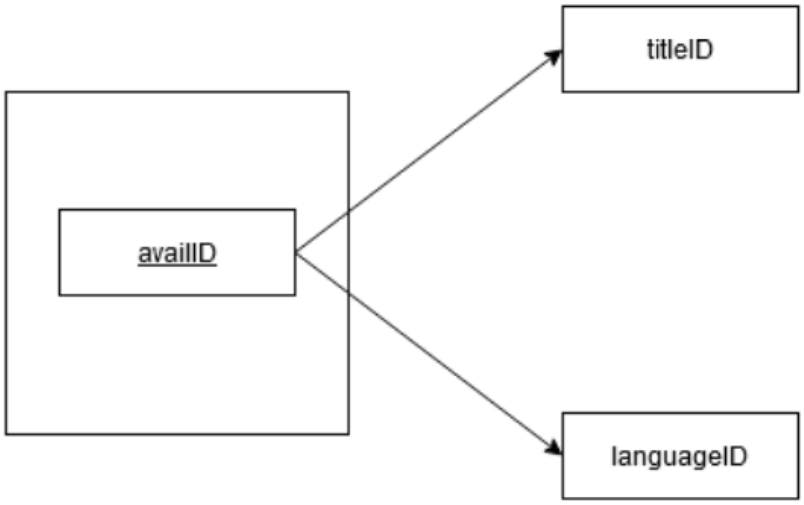

7. Languages
languageID
languageName
8. Cast
castID
castName
9. Director
directorID
directorName


10. ProductionHouse
phID
phName
11. Creator
creatorID
creatorName
15.ActsIn
actingID
titleID
castID

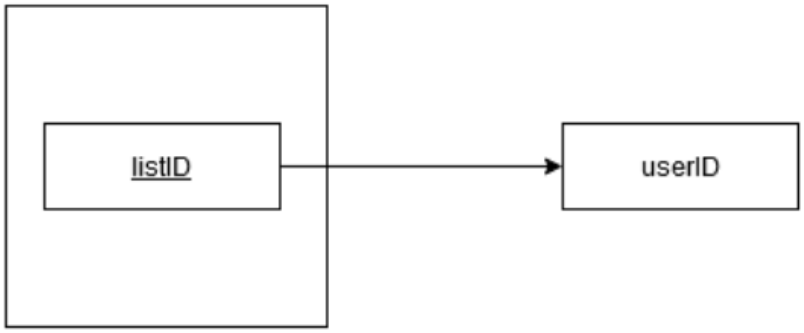
16.StreamingOn

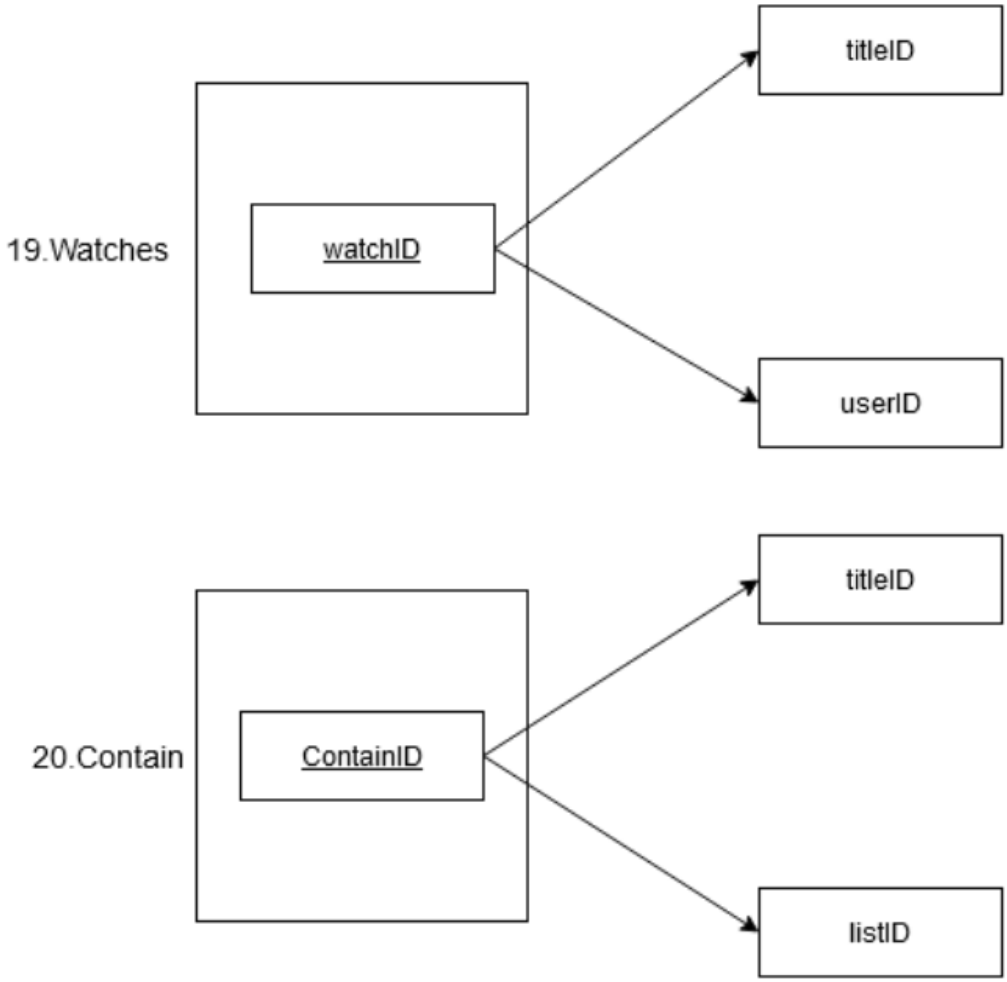


17.AvailableIn



18.Watchlist





10.Normalization

The database is in 3NF because there are no Transitive Dependencies.

11.Implementation

a. Tables

MySQL 8.0 Command Line Client

```
mysql> desc title;
+---------------------+-------------+------+-----+---------+-------+
| Field               | Type        | Null | Key | Default | Extra |
+---------------------+-------------+------+-----+---------+-------+
| titleID             | varchar(5)  | NO   | PRI | NULL    |       |
| titleName           | varchar(50) | YES  |     | NULL    |       |
| rottenTomatoesRating | tinyint(4)  | YES  |     | NULL    |       |
| imdbRating          | decimal(2,1) | YES  |     | NULL    |       |
| releaseYear         | year(4)     | YES  |     | NULL    |       |
| genreID             | int(11)     | NO   | MUL | NULL    |       |
| limitID             | int(11)     | NO   | MUL | NULL    |       |
| adminID             | int(11)     | NO   | MUL | NULL    |       |
+---------------------+-------------+------+-----+---------+-------+
8 rows in set (0.00 sec)

mysql> desc movie;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| titleID    | varchar(5)  | NO   | PRI | NULL    |       |
| budget     | varchar(15) | YES  |     | NULL    |       |
| collection | varchar(15) | YES  |     | NULL    |       |
| directorID | int(11)     | NO   | MUL | NULL    |       |
| phID       | int(11)     | NO   | MUL | NULL    |       |
+------------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)

mysql> desc tvshows;
+-----------+------------+------+-----+---------+-------+
```

MySQL 8.0 Command Line Client

```
mysql> desc tvshows;
+-----------+------------+------+-----+---------+-------+
| Field     | Type       | Null | Key | Default | Extra |
+-----------+------------+------+-----+---------+-------+
| titleID   | varchar(5) | NO   | PRI | NULL    |       |
| creatorID | int(11)    | NO   | MUL | NULL    |       |
+-----------+------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> desc genre;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| genreID   | int(11)     | NO   | PRI | NULL    |       |
| genreName | varchar(25) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> desc agelimit;
+-----------+------------+------+-----+---------+-------+
| Field     | Type       | Null | Key | Default | Extra |
+-----------+------------+------+-----+---------+-------+
| limitID   | int(11)    | NO   | PRI | NULL    |       |
| limitName | varchar(5) | YES  |     | NULL    |       |
+-----------+------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql>
```

MySQL 8.0 Command Line Client

```
mysql> desc ott;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| ottID   | int(11)     | NO   | PRI | NULL    |       |
| ottName | varchar(25) | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> desc languages;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| languageID   | int(11)     | NO   | PRI | NULL    |       |
| languageName | varchar(25) | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> desc casting;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| castID   | varchar(5)  | NO   | PRI | NULL    |       |
| castName | varchar(25) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql>
```

MySQL 8.0 Command Line Client

```
mysql> desc director;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| directorID   | int(11)     | NO   | PRI | NULL    |       |
| directorName | varchar(25) | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> desc productionhouse
    -> ;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| phID   | int(11)     | NO   | PRI | NULL    |       |
| phName | varchar(25) | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> desc creator;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| creatorID   | int(11)     | NO   | PRI | NULL    |       |
| creatorName | varchar(25) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql>
```

MySQL 8.0 Command Line Client

```
mysql> desc dbuser;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| userID   | int(11)     | NO   | PRI | NULL    |       |
| userName | varchar(25) | YES  |     | NULL    |       |
| passkey  | varchar(50) | YES  |     | NULL    |       |
| emailID  | varchar(50) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> desc phonenumber;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| phoneID | int(11)     | NO   | PRI | NULL    |       |
| userID  | int(11)     | NO   | MUL | NULL    |       |
| phone   | varchar(15) | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> desc dbadmin;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| adminID   | int(11)     | NO   | PRI | NULL    |       |
| adminName | varchar(25) | YES  |     | NULL    |       |
| passkey   | varchar(50) | YES  |     | NULL    |       |
| emailID   | varchar(50) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
```

MySQL 8.0 Command Line Client

```
mysql> desc actsin;
+----------+------------+------+-----+---------+-------+
| Field    | Type       | Null | Key | Default | Extra |
+----------+------------+------+-----+---------+-------+
| actingID | int(11)    | NO   | PRI | NULL    |       |
| titleID  | varchar(5) | NO   | MUL | NULL    |       |
| castID   | varchar(5) | NO   | MUL | NULL    |       |
+----------+------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> desc streamingon;
+----------+------------+------+-----+---------+-------+
| Field    | Type       | Null | Key | Default | Extra |
+----------+------------+------+-----+---------+-------+
| streamID | int(11)    | NO   | PRI | NULL    |       |
| titleID  | varchar(5) | NO   | MUL | NULL    |       |
| ottID    | int(11)    | NO   | MUL | NULL    |       |
+----------+------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> desc availablein;
+------------+------------+------+-----+---------+-------+
| Field      | Type       | Null | Key | Default | Extra |
+------------+------------+------+-----+---------+-------+
| availID    | int(11)    | NO   | PRI | NULL    |       |
| titleID    | varchar(5) | NO   | MUL | NULL    |       |
| languageID | int(11)    | NO   | MUL | NULL    |       |
+------------+------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

MySQL 8.0 Command Line Client

```
mysql> desc watches;
+---------+------------+------+-----+---------+-------+
| Field   | Type       | Null | Key | Default | Extra |
+---------+------------+------+-----+---------+-------+
| watchID | int(11)    | NO   | PRI | NULL    |       |
| titleID | varchar(5) | NO   | MUL | NULL    |       |
| userID  | int(11)    | NO   | MUL | NULL    |       |
+---------+------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> desc watchlist;
+--------+---------+------+-----+---------+-------+
| Field  | Type    | Null | Key | Default | Extra |
+--------+---------+------+-----+---------+-------+
| listID | int(11) | NO   | PRI | NULL    |       |
| userID | int(11) | NO   | MUL | NULL    |       |
+--------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> desc contain;
+-----------+------------+------+-----+---------+-------+
| Field     | Type       | Null | Key | Default | Extra |
+-----------+------------+------+-----+---------+-------+
| containID | int(11)    | NO   | PRI | NULL    |       |
| titleID   | varchar(5) | NO   | MUL | NULL    |       |
| listID    | int(11)    | NO   | MUL | NULL    |       |
+-----------+------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql>
```

b. Table Data

MySQL 8.0 Command Line Client

```
mysql> select * from title;
+---------+-----------------------------------+----------------------+------------+-------------+---------+---------+---------+
| titleID | titleName                         | rottenTomatoesRating | imdbRating | releaseYear | genreID | limitID | adminID |
+---------+-----------------------------------+----------------------+------------+-------------+---------+---------+---------+
| MO1     | Forrest Gump                      |                   71 |        8.8 |        1994 |       1 |       2 |       1 |
| MO2     | Avengers Endgame                  |                   94 |        8.4 |        2019 |       2 |       2 |       1 |
| MO3     | The Dark Knight                   |                   94 |        9.0 |        2008 |       3 |       1 |       1 |
| MO4     | Dumb and Dumber                   |                   68 |        7.3 |        1994 |       4 |       4 |       1 |
| MO5     | Titanic                           |                   89 |        7.8 |        1997 |       5 |       5 |       1 |
| TV1     | Breaking Bad                      |                   96 |        9.5 |        2008 |       6 |       6 |       1 |
| TV2     | Sacred Games                      |                   76 |        8.7 |        2018 |       6 |       7 |       1 |
| TV3     | Brooklyn Nine Nine                |                   95 |        8.4 |        2013 |       4 |       3 |       1 |
| TV4     | Dark                              |                   95 |        8.8 |        2017 |       8 |       5 |       1 |
| TV5     | Scam 1992: The Harshad Mehta Story |                 NULL |        9.5 |        2020 |       7 |       3 |       1 |
+---------+-----------------------------------+----------------------+------------+-------------+---------+---------+---------+
10 rows in set (0.00 sec)

mysql> select * from movie;
+---------+--------------+----------------+------------+------+
| titleID | budget       | collection     | directorID | phID |
+---------+--------------+----------------+------------+------+
| MO1     | $55 Million  | $683.1 Million |          1 |    1 |
| MO2     | $356 Million | $2.798 Billion |          2 |    2 |
| MO3     | $185 Million | $1.005 Billion |          3 |    3 |
| MO4     | $50 Million  | $169.8 Million |          4 |    4 |
| MO5     | $200 Million | $2.195 Billion |          5 |    5 |
+---------+--------------+----------------+------------+------+
5 rows in set (0.01 sec)

mysql>
```

MySQL 8.0 Command Line Client

```
mysql> select * from tvshows;
+---------+-----------+
| titleID | creatorID |
+---------+-----------+
| TV1     |         1 |
| TV2     |         2 |
| TV3     |         3 |
| TV4     |         4 |
| TV5     |         5 |
+---------+-----------+
5 rows in set (0.01 sec)

mysql> select * from genre;
+---------+-------------+
| genreID | genreName   |
+---------+-------------+
|       1 | Drama       |
|       2 | Superhero   |
|       3 | Thriller    |
|       4 | Comedy      |
|       5 | Romantic    |
|       6 | Crime Drama |
|       7 | Crime       |
|       8 | Sci-Fi      |
+---------+-------------+
8 rows in set (0.01 sec)

mysql> select * from agelimit;
+---------+-----------+
```

MySQL 8.0 Command Line Client

```
mysql> select * from agelimit;
+---------+-----------+
| limitID | limitName |
+---------+-----------+
|       1 | 12+       |
|       2 | 13+       |
|       3 | 14+       |
|       4 | 15+       |
|       5 | 16+       |
|       6 | 17+       |
|       7 | 18+       |
+---------+-----------+
7 rows in set (0.01 sec)

mysql> select * from ott;
+-------+-------------------+
| ottID | ottName           |
+-------+-------------------+
|     1 | Netflix           |
|     2 | Amazon Prime Video |
|     3 | Disney+ Hotstar   |
|     4 | Sony Liv          |
|     5 | Hulu              |
|     6 | Zee5              |
+-------+-------------------+
6 rows in set (0.01 sec)

mysql>
```

MySQL 8.0 Command Line Client

```
mysql> select * from languages;
+------------+--------------+
| languageID | languageName |
+------------+--------------+
|          1 | English      |
|          2 | Hindi        |
|          3 | French       |
|          4 | Spanish      |
|          5 | German       |
|          6 | Telugu       |
|          7 | Mandarin     |
|          8 | Swedish      |
|          9 | Persian      |
+------------+--------------+
9 rows in set (0.00 sec)

mysql> select * from director;
+------------+------------------+
| directorID | directorName     |
+------------+------------------+
|          1 | Robert Zemeckis  |
|          2 | Russo Brothers   |
|          3 | Christopher Nolan |
|          4 | Peter Farrelly   |
|          5 | James Cameron    |
+------------+------------------+
5 rows in set (0.01 sec)

mysql>
```

MySQL 8.0 Command Line Client

```
5 rows in set (0.01 sec)

mysql> select * from casting;
+--------+--------------------+
| castID | castName           |
+--------+--------------------+
| F1     | Robin Wright       |
| F10    | Shreya Dhanwanthary |
| F2     | Scarlett Johansson |
| F3     | Maggie Gyllenhaal  |
| F4     | Rachel Melvin      |
| F5     | Kate Winslet       |
| F6     | Anna Gunn          |
| F7     | Radhika Apte       |
| F8     | Melissa Fumero     |
| F9     | Lisa Vicari        |
| M1     | Tom Hanks          |
| M10    | Pratik Gandhi      |
| M2     | Robert Downey Jr.  |
| M3     | Christian Bale     |
| M4     | Jim Carrey         |
| M5     | Leonardo De Caprio |
| M6     | Bryan Cranston     |
| M7     | Nawazuddin Siddiqui |
| M8     | Andy Samberg       |
| M9     | Louis Hoffman      |
+--------+--------------------+
20 rows in set (0.00 sec)

mysql>
```

MySQL 8.0 Command Line Client

```
mysql> select * from productionhouse;
+------+--------------------+
| phID | phName             |
+------+--------------------+
|    1 | Paramount Pictures |
|    2 | Marvel Studios     |
|    3 | Waner Bros         |
|    4 | New Line Cinema    |
|    5 | 20th Century Fox   |
+------+--------------------+
5 rows in set (0.01 sec)

mysql> select * from creator;
+-----------+----------------+
| creatorID | creatorName    |
+-----------+----------------+
|         1 | Vince Gilligan |
|         2 | Vikram Chandra |
|         3 | Daniel J. Goor |
|         4 | Baran Bo Odar  |
|         5 | Hansal Mehta   |
+-----------+----------------+
5 rows in set (0.01 sec)

mysql> select * from dbuser;
+--------+----------+-----------+-----------------+
| userID | userName | passkey   | emailID         |
+--------+----------+-----------+-----------------+
|      1 | Henry    | henry123  | henry@mail.com  |
```

MySQL 8.0 Command Line Client

```
mysql> select * from dbuser;
+--------+----------+-----------+-----------------+
| userID | userName | passkey   | emailID         |
+--------+----------+-----------+-----------------+
|      1 | Henry    | henry123  | henry@mail.com  |
|      2 | Eddie    | eddie123  | eddie@mail.com  |
|      3 | Ashley   | ashley123 | ashley@mail.com |
|      4 | Daniel   | daniel123 | daniel@mail.com |
|      5 | Clara    | clara123  | clara@mail.com  |
+--------+----------+-----------+-----------------+
5 rows in set (0.00 sec)

mysql> select * from phonenumber;
+---------+--------+------------+
| phoneID | userID | phone      |
+---------+--------+------------+
|       1 |      1 | 7593528475 |
|       2 |      2 | 4238937410 |
|       3 |      2 | 7391034641 |
|       4 |      3 | 7824334231 |
|       5 |      4 | 7784873487 |
|       6 |      5 | 7083744940 |
+---------+--------+------------+
6 rows in set (0.00 sec)

mysql> select * from dbadmin;
+---------+-----------+---------+---------------+
| adminID | adminName | passkey | emailID       |
+---------+-----------+---------+---------------+
|       1 | Ravi      | ravi123 | ravi@mail.com |
```

MySQL 8.0 Command Line Client

```
mysql> select * from actsin;
+----------+---------+--------+
| actingID | titleID | castID |
+----------+---------+--------+
|        1 | MO1     | M1     |
|        2 | MO1     | F1     |
|        3 | MO2     | M2     |
|        4 | MO2     | F2     |
|        5 | MO3     | M3     |
|        6 | MO3     | F3     |
|        7 | MO4     | M4     |
|        8 | MO4     | F4     |
|        9 | MO5     | M5     |
|       10 | MO5     | F5     |
|       11 | TV1     | M6     |
|       12 | TV1     | F6     |
|       13 | TV2     | M7     |
|       14 | TV2     | F7     |
|       15 | TV3     | M8     |
|       16 | TV3     | F8     |
|       17 | TV4     | M9     |
|       18 | TV4     | F9     |
|       19 | TV5     | M10    |
|       20 | TV5     | F10    |
+----------+---------+--------+
20 rows in set (0.01 sec)

mysql>
```

MySQL 8.0 Command Line Client

```
mysql> select * from streamingon;
+----------+---------+-------+
| streamID | titleID | ottID |
+----------+---------+-------+
|        1 | MO1     |     1 |
|        2 | MO1     |     2 |
|        3 | MO2     |     1 |
|        4 | MO2     |     2 |
|        5 | MO2     |     3 |
|        6 | MO3     |     1 |
|        7 | MO3     |     2 |
|        8 | MO3     |     4 |
|        9 | MO4     |     1 |
|       10 | MO4     |     2 |
|       11 | MO4     |     5 |
|       12 | MO5     |     1 |
|       13 | MO5     |     2 |
|       14 | TV1     |     1 |
|       15 | TV1     |     2 |
|       16 | TV1     |     5 |
|       17 | TV2     |     1 |
|       18 | TV2     |     2 |
|       19 | TV2     |     6 |
|       20 | TV3     |     1 |
|       21 | TV3     |     2 |
|       22 | TV3     |     5 |
|       23 | TV4     |     1 |
|       24 | TV5     |     4 |
+----------+---------+-------+
24 rows in set (0.01 sec)
```

MySQL 8.0 Command Line Client

```
mysql> select * from availablein;
+---------+---------+------------+
| availID | titleID | languageID |
+---------+---------+------------+
|       1 | MO1     |          1 |
|       2 | MO1     |          2 |
|       3 | MO1     |          3 |
|       4 | MO1     |          4 |
|       5 | MO2     |          1 |
|       6 | MO2     |          2 |
|       7 | MO2     |          4 |
|       8 | MO2     |          5 |
|       9 | MO2     |          6 |
|      10 | MO3     |          1 |
|      11 | MO3     |          2 |
|      12 | MO3     |          7 |
|      13 | MO4     |          1 |
|      14 | MO4     |          2 |
|      15 | MO4     |          5 |
|      16 | MO4     |          8 |
|      17 | MO5     |          1 |
|      18 | MO5     |          2 |
|      19 | MO5     |          8 |
|      20 | TV1     |          1 |
|      21 | TV1     |          4 |
|      22 | TV1     |          5 |
|      23 | TV1     |          9 |
|      24 | TV2     |          1 |
|      25 | TV2     |          2 |
|      26 | TV3     |          1 |
|      27 | TV4     |          1 |
|      28 | TV4     |          5 |
|      29 | TV5     |          2 |
+---------+---------+------------+
29 rows in set (0.01 sec)
```

MySQL 8.0 Command Line Client

```
mysql> select * from watches;
+---------+---------+--------+
| watchID | titleID | userID |
+---------+---------+--------+
|       1 | MO1     |      1 |
|       2 | TV4     |      1 |
|       3 | MO2     |      2 |
|       4 | TV1     |      3 |
|       5 | TV5     |      3 |
|       6 | MO2     |      4 |
|       7 | MO5     |      5 |
|       8 | TV5     |      5 |
+---------+---------+--------+
8 rows in set (0.00 sec)

mysql> select * from watchlist;
+--------+--------+
| listID | userID |
+--------+--------+
|      1 |      1 |
|      2 |      2 |
|      3 |      3 |
|      4 |      4 |
|      5 |      5 |
+--------+--------+
5 rows in set (0.01 sec)

mysql>
```

MySQL 8.0 Command Line Client

```
mysql> select * from contain;
+-----------+---------+--------+
| containID | titleID | listID |
+-----------+---------+--------+
|         1 | TV4     |      1 |
|         2 | MO2     |      2 |
|         3 | TV1     |      3 |
|         4 | MO2     |      4 |
|         5 | MO5     |      5 |
+-----------+---------+--------+
5 rows in set (0.00 sec)

mysql>
```

12.25 Queries, 5 Functions, 5 Procedures, 5 Triggers

a. 25 Queries

i. Get the name of the cast for all the Movies.

MySQL 8.0 Command Line Client

```
mysql> select t.titleName, c.castName from title t inner join actsin a on a.titleID=t.titleID left join casting c on c.c
astID=a.castID where t.titleID like 'MO%';
+-----------------+--------------------+
| titleName       | castName           |
+-----------------+--------------------+
| Forrest Gump    | Tom Hanks          |
| Forrest Gump    | Robin Wright       |
| Avengers Endgame | Robert Downey Jr.  |
| Avengers Endgame | Scarlett Johansson |
| The Dark Knight | Christian Bale     |
| The Dark Knight | Maggie Gyllenhaal  |
| Dumb and Dumber | Jim Carrey         |
| Dumb and Dumber | Rachel Melvin      |
| Titanic         | Leonardo De Caprio |
| Titanic         | Kate Winslet       |
+-----------------+--------------------+
10 rows in set (0.00 sec)
```

ii. Get the name of OTTs on which the Movies are available.

MySQL 8.0 Command Line Client

```
mysql> select t.titleName, o.ottName from title t inner join streamingon s on s.titleID=t.titleID left join ott o on o.o
ttID=s.ottID where t.titleID like 'MO%';
+-----------------+--------------------+
| titleName       | ottName            |
+-----------------+--------------------+
| Forrest Gump    | Netflix            |
| Forrest Gump    | Amazon Prime Video |
| Avengers Endgame | Netflix            |
| Avengers Endgame | Amazon Prime Video |
| Avengers Endgame | Disney+ Hotstar    |
| The Dark Knight | Netflix            |
| The Dark Knight | Amazon Prime Video |
| The Dark Knight | Sony Liv           |
| Dumb and Dumber | Netflix            |
| Dumb and Dumber | Amazon Prime Video |
| Dumb and Dumber | Hulu               |
| Titanic         | Netflix            |
| Titanic         | Amazon Prime Video |
+-----------------+--------------------+
13 rows in set (0.00 sec)
```

iii. Get the name of cast for all TV Shows.

MySQL 8.0 Command Line Client

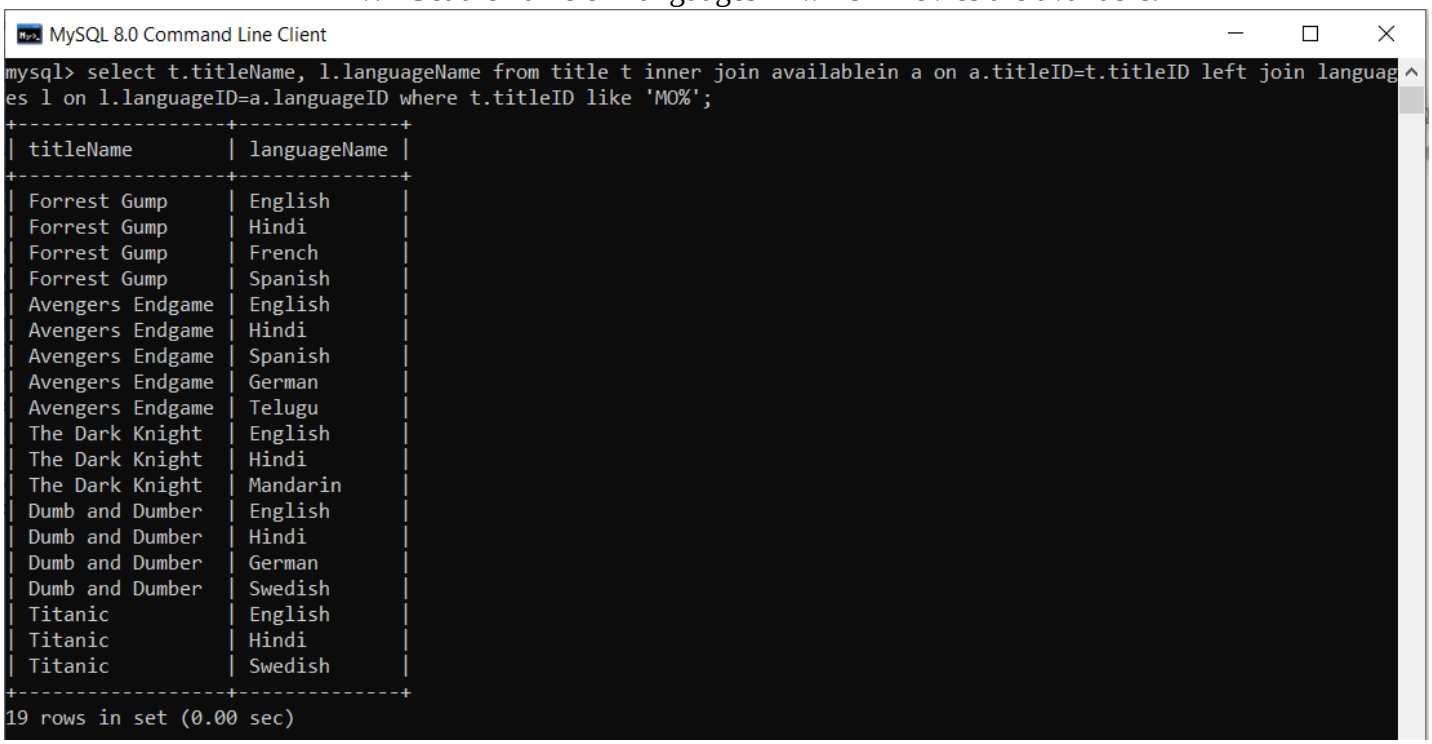
```
mysql> select t.titleName, c.castName from title t inner join actsin a on a.titleID=t.titleID left join casting c on c.c
astID=a.castID where t.titleID like 'TV%';
+----------------------------------+--------------------+
| titleName                        | castName           |
+----------------------------------+--------------------+
| Breaking Bad                     | Bryan Cranston     |
| Breaking Bad                     | Anna Gunn          |
| Sacred Games                     | Nawazuddin Siddiqui |
| Sacred Games                     | Radhika Apte       |
| Brooklyn Nine Nine               | Andy Samberg       |
| Brooklyn Nine Nine               | Melissa Fumero     |
| Dark                             | Louis Hoffman      |
| Dark                             | Lisa Vicari        |
| Scam 1992: The Harshad Mehta Story | Pratik Gandhi      |
| Scam 1992: The Harshad Mehta Story | Shreya Dhanwanthary |
+----------------------------------+--------------------+
10 rows in set (0.00 sec)
```

iv. Get the name of OTTs on which the TV Shows are available.

MySQL 8.0 Command Line Client

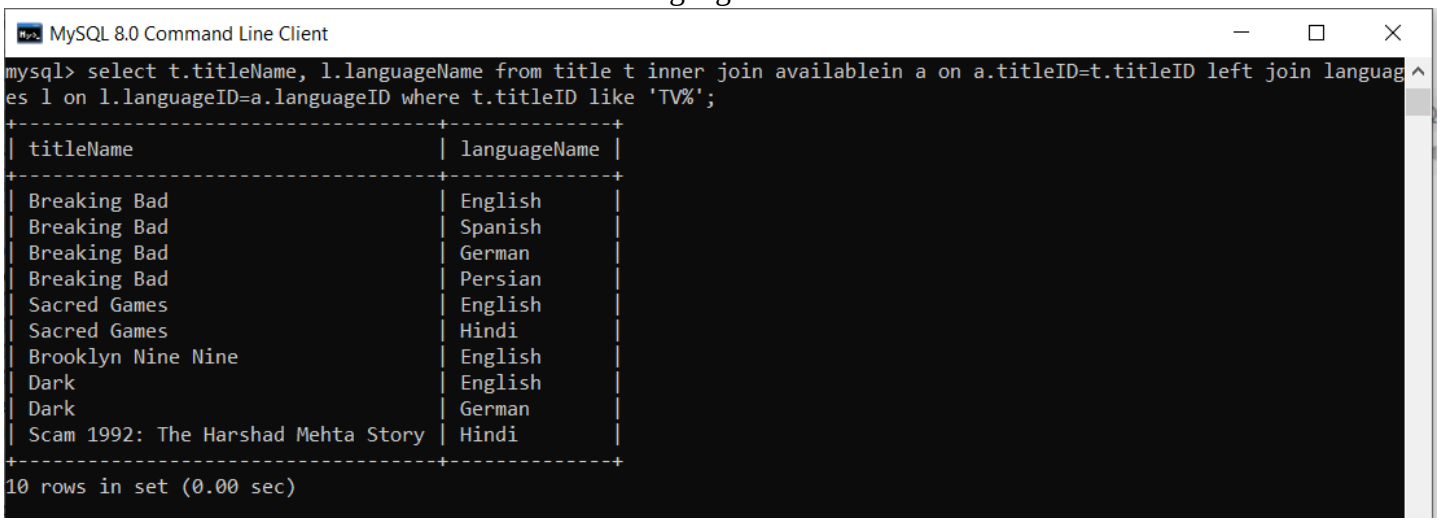
```
mysql> select t.titleName, o.ottName from title t inner join streamingon s on s.titleID=t.titleID left join ott o on o.o
ttID=s.ottID where t.titleID like 'TV%';
+----------------------------------+--------------------+
| titleName                        | ottName            |
+----------------------------------+--------------------+
| Breaking Bad                     | Netflix            |
| Breaking Bad                     | Amazon Prime Video |
| Breaking Bad                     | Hulu               |
| Sacred Games                     | Netflix            |
| Sacred Games                     | Amazon Prime Video |
| Sacred Games                     | Zee5               |
| Brooklyn Nine Nine               | Netflix            |
| Brooklyn Nine Nine               | Amazon Prime Video |
| Brooklyn Nine Nine               | Hulu               |
| Dark                             | Netflix            |
| Scam 1992: The Harshad Mehta Story | Sony Liv           |
+----------------------------------+--------------------+
11 rows in set (0.00 sec)
```

v. Get the name of Languages in which Movies are available.



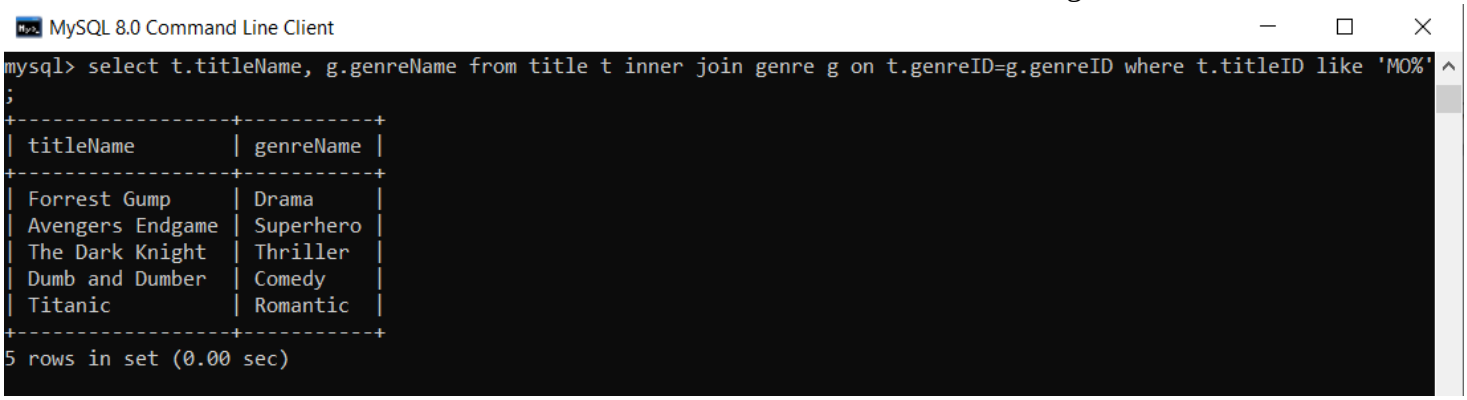
```
MySQL 8.0 Command Line Client
mysql> select t.titleName, l.languageName from title t inner join availablein a on a.titleID=t.titleID left join languag
es l on l.languageID=a.languageID where t.titleID like 'MO%';
+------------------+--------------+
| titleName        | languageName |
+------------------+--------------+
| Forrest Gump     | English      |
| Forrest Gump     | Hindi        |
| Forrest Gump     | French       |
| Forrest Gump     | Spanish      |
| Avengers Endgame | English      |
| Avengers Endgame | Hindi        |
| Avengers Endgame | Spanish      |
| Avengers Endgame | German       |
| Avengers Endgame | Telugu       |
| The Dark Knight  | English      |
| The Dark Knight  | Hindi        |
| The Dark Knight  | Mandarin     |
| Dumb and Dumber  | English      |
| Dumb and Dumber  | Hindi        |
| Dumb and Dumber  | German       |
| Dumb and Dumber  | Swedish      |
| Titanic          | English      |
| Titanic          | Hindi        |
| Titanic          | Swedish      |
+------------------+--------------+
19 rows in set (0.00 sec)
```

vi. Get the name of Languages in which TV Shows are available.



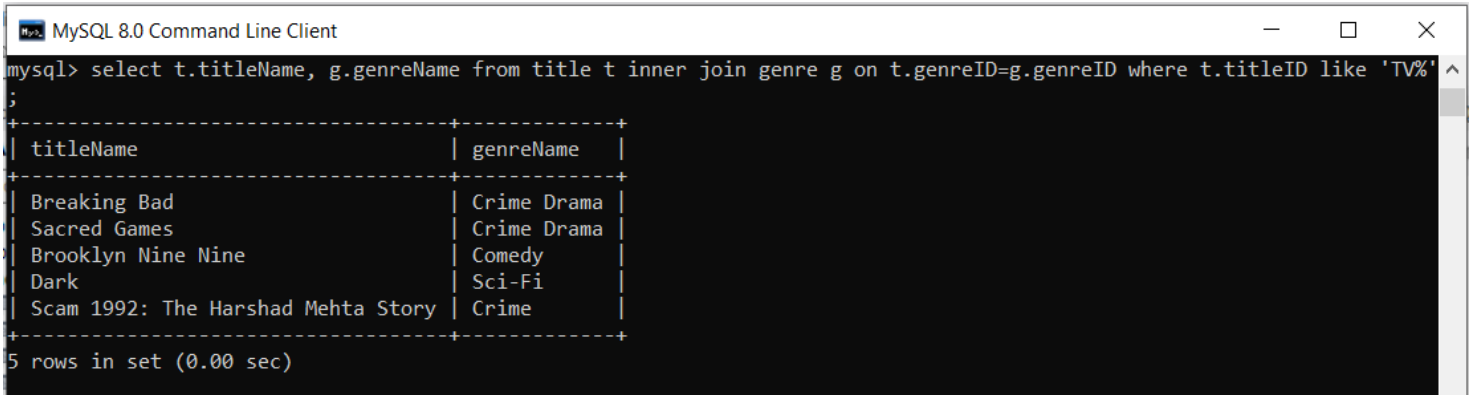
```
MySQL 8.0 Command Line Client
mysql> select t.titleName, l.languageName from title t inner join availablein a on a.titleID=t.titleID left join languag
es l on l.languageID=a.languageID where t.titleID like 'TV%';
+-----------------------------------+--------------+
| titleName                         | languageName |
+-----------------------------------+--------------+
| Breaking Bad                      | English      |
| Breaking Bad                      | Spanish      |
| Breaking Bad                      | German       |
| Breaking Bad                      | Persian      |
| Sacred Games                      | English      |
| Sacred Games                      | Hindi        |
| Brooklyn Nine Nine                | English      |
| Dark                              | English      |
| Dark                              | German       |
| Scam 1992: The Harshad Mehta Story | Hindi        |
+-----------------------------------+--------------+
10 rows in set (0.00 sec)
```

vii. Get the name of Genre to which the Movies belong.

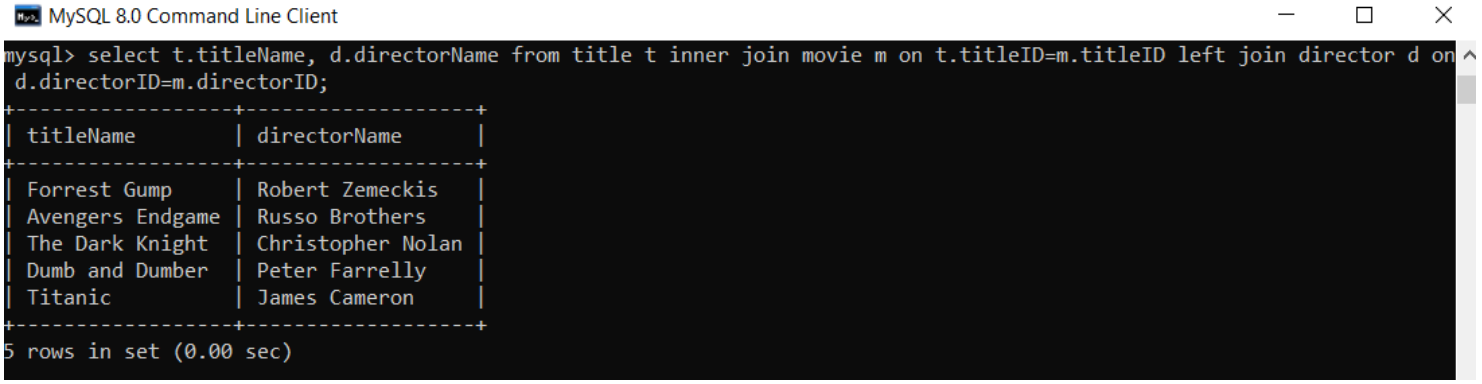


```
MySQL 8.0 Command Line Client
mysql> select t.titleName, g.genreName from title t inner join genre g on t.genreID=g.genreID where t.titleID like 'MO%'
;
+------------------+-----------+
| titleName        | genreName |
+------------------+-----------+
| Forrest Gump     | Drama     |
| Avengers Endgame | Superhero |
| The Dark Knight  | Thriller  |
| Dumb and Dumber  | Comedy    |
| Titanic          | Romantic  |
+------------------+-----------+
5 rows in set (0.00 sec)
```

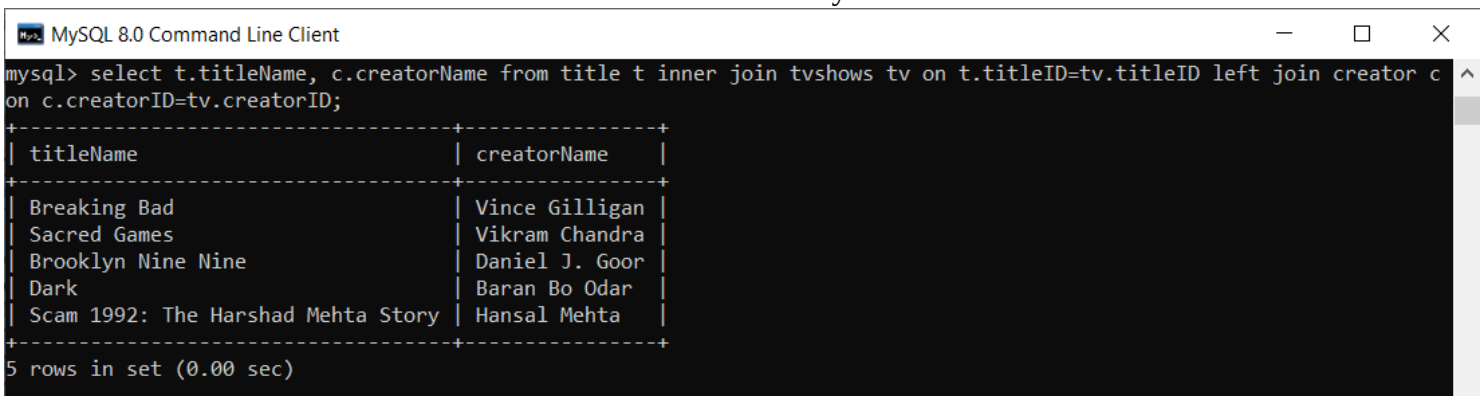
viii. Get the name of Genre to which the TV Shows belong.



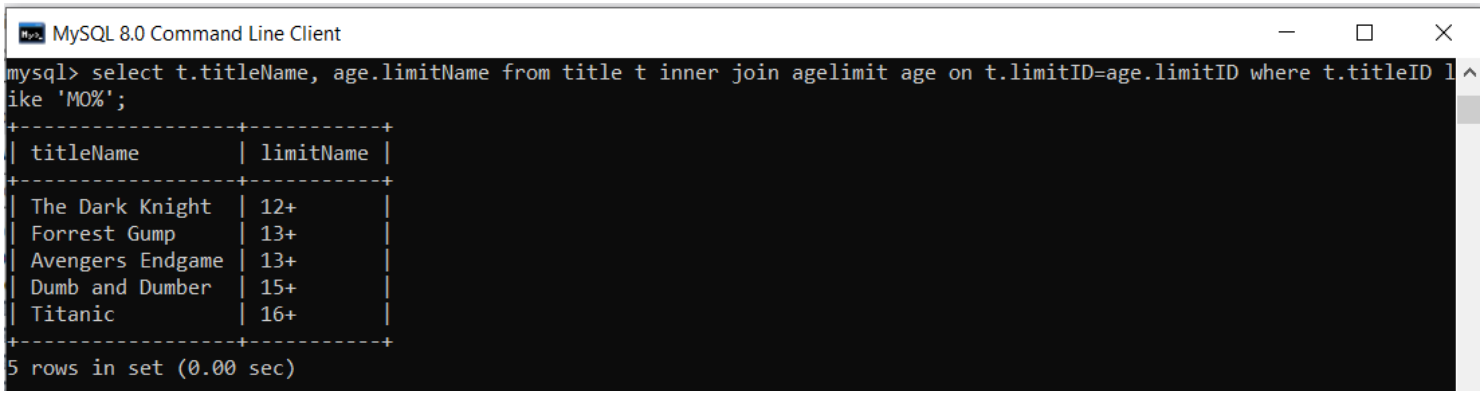
ix. Get the name of director of every Movie.



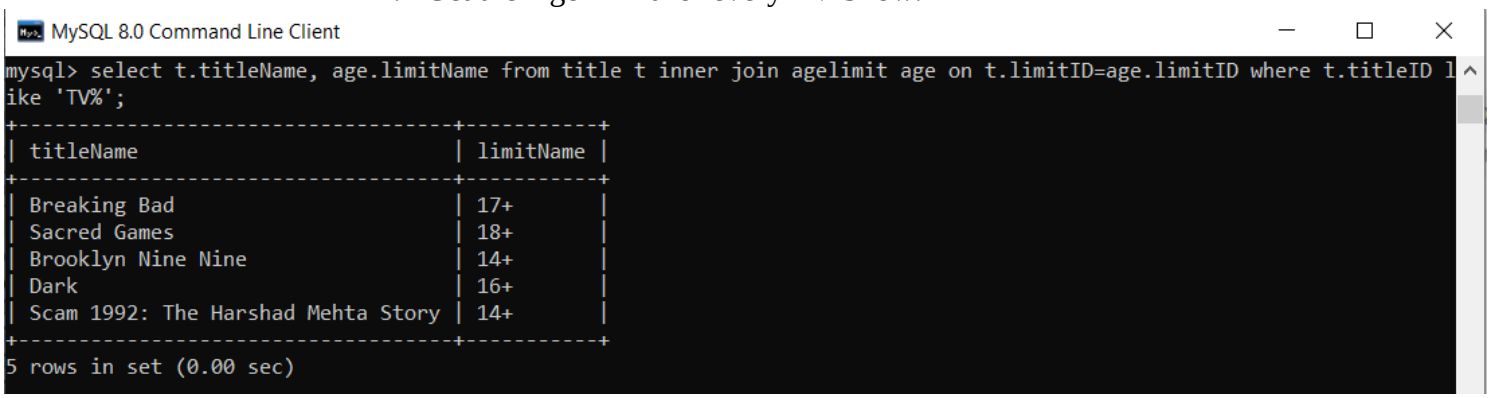
x. Get the name of Creator of every TV Show.



xi. Get the Age Limit for every Movie.

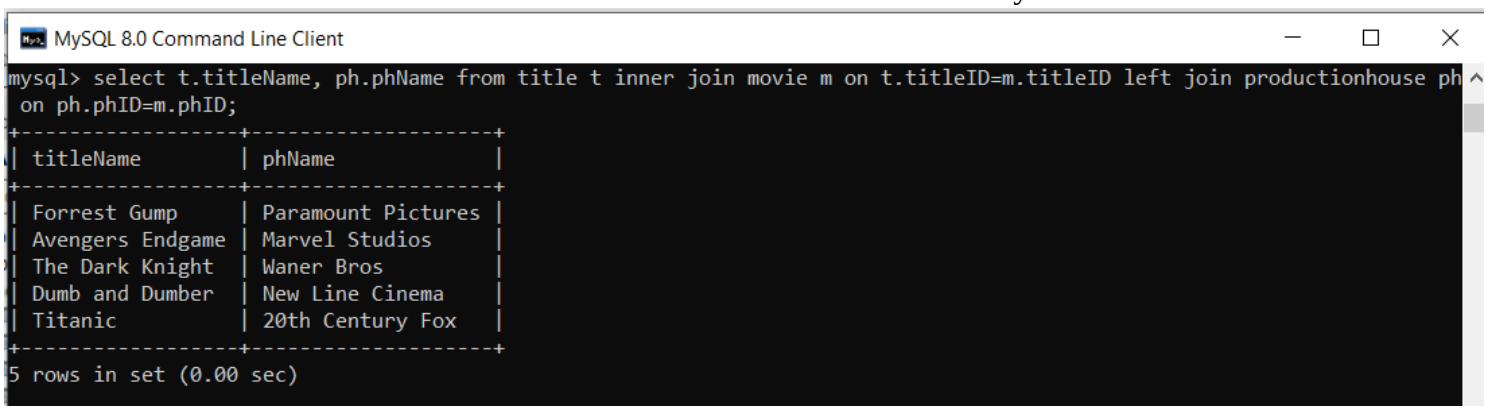


xii. Get the Age Limit for every TV Show.



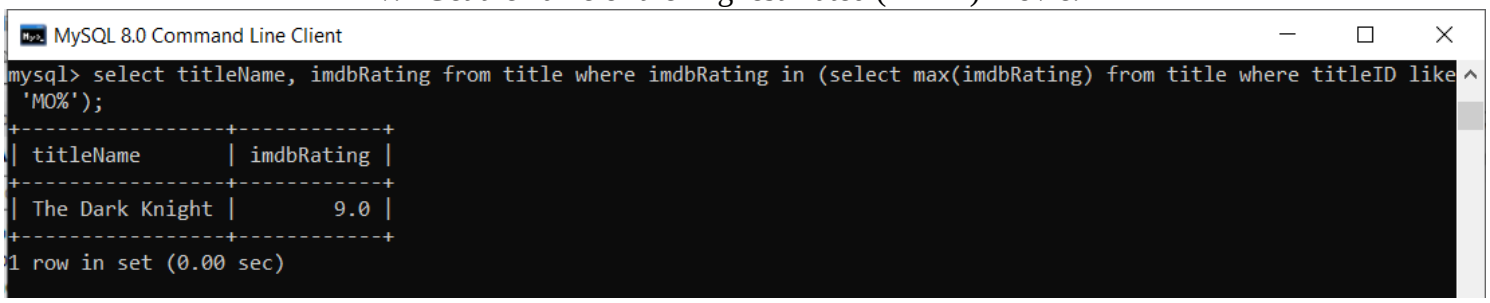
```
MySQL 8.0 Command Line Client
mysql> select t.titleName, age.limitName from title t inner join agelimit age on t.limitID=age.limitID where t.titleID l
ike 'TV%';
+-----------------------------------+-----------+
| titleName                         | limitName |
+-----------------------------------+-----------+
| Breaking Bad                      | 17+       |
| Sacred Games                      | 18+       |
| Brooklyn Nine Nine                | 14+       |
| Dark                              | 16+       |
| Scam 1992: The Harshad Mehta Story | 14+       |
+-----------------------------------+-----------+
5 rows in set (0.00 sec)
```

xiii. Get the name of the Production House of every Movie.



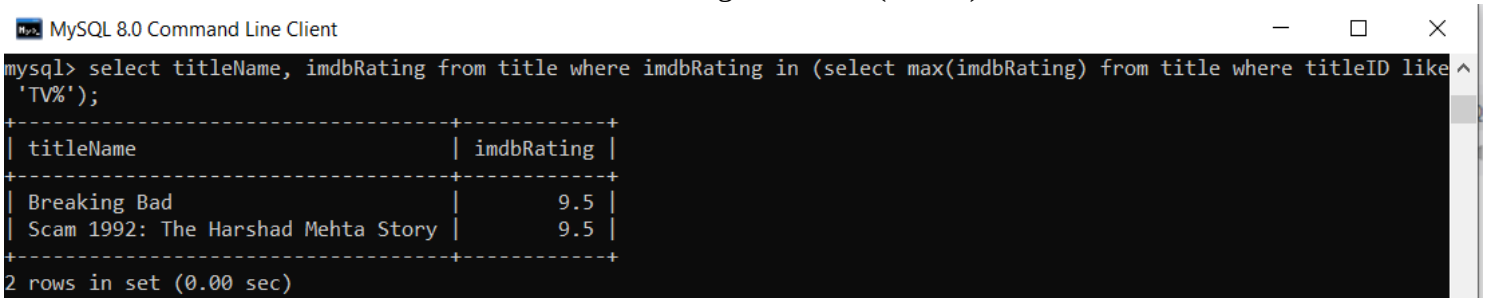
```
MySQL 8.0 Command Line Client
mysql> select t.titleName, ph.phName from title t inner join movie m on t.titleID=m.titleID left join productionhouse ph
 on ph.phID=m.phID;
+-----------------+--------------------+
| titleName       | phName             |
+-----------------+--------------------+
| Forrest Gump    | Paramount Pictures |
| Avengers Endgame | Marvel Studios     |
| The Dark Knight | Waner Bros         |
| Dumb and Dumber | New Line Cinema    |
| Titanic         | 20th Century Fox   |
+-----------------+--------------------+
5 rows in set (0.00 sec)
```

xiv. Get the name of the Highest Rated (IMDB) Movie.



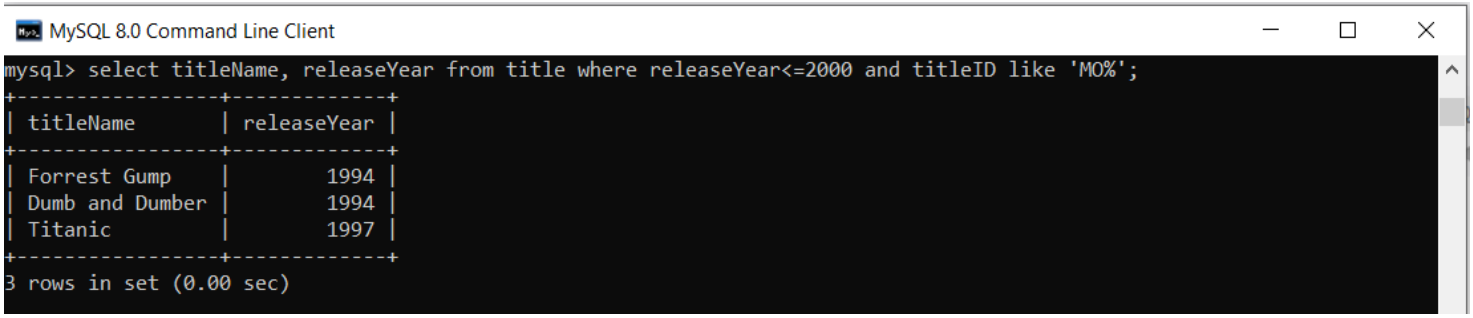
```
MySQL 8.0 Command Line Client
mysql> select titleName, imdbRating from title where imdbRating in (select max(imdbRating) from title where titleID like
 'MO%');
+-----------------+------------+
| titleName       | imdbRating |
+-----------------+------------+
| The Dark Knight |        9.0 |
+-----------------+------------+
1 row in set (0.00 sec)
```

xv. Get the name of the Highest Rated (IMDB) TV Show.

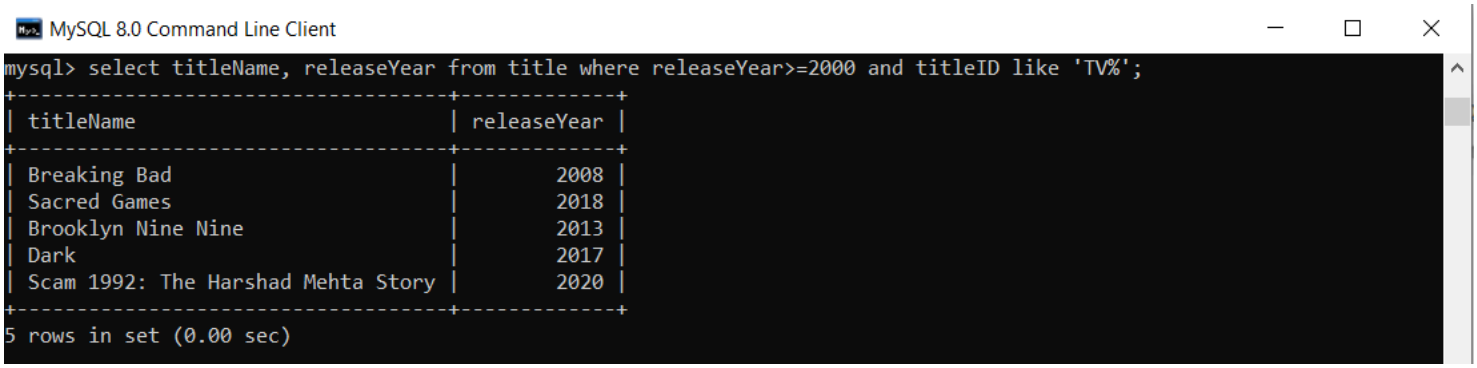


```
MySQL 8.0 Command Line Client
mysql> select titleName, imdbRating from title where imdbRating in (select max(imdbRating) from title where titleID like
 'TV%');
+-----------------------------------+------------+
| titleName                         | imdbRating |
+-----------------------------------+------------+
| Breaking Bad                      |        9.5 |
| Scam 1992: The Harshad Mehta Story |        9.5 |
+-----------------------------------+------------+
2 rows in set (0.00 sec)
```

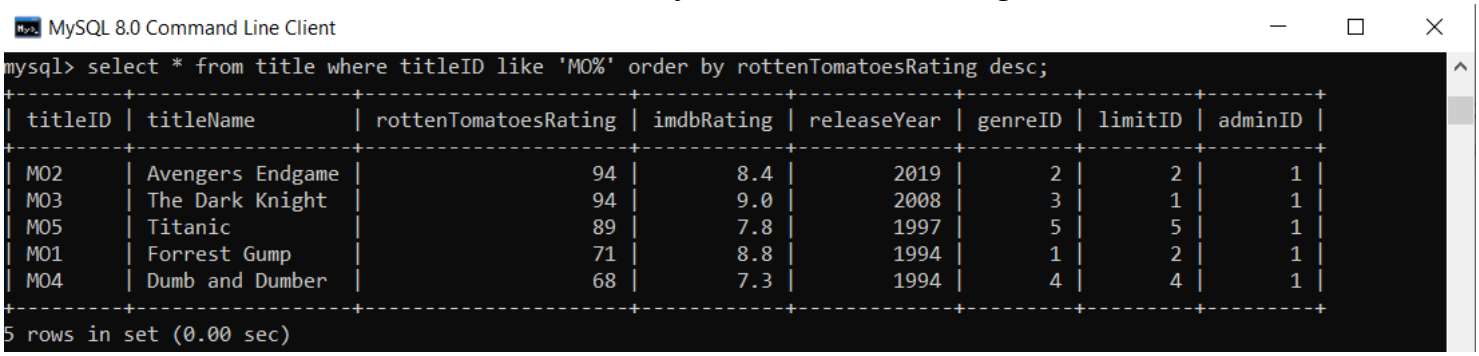
xvi. Get the names of Movies released before 2000.



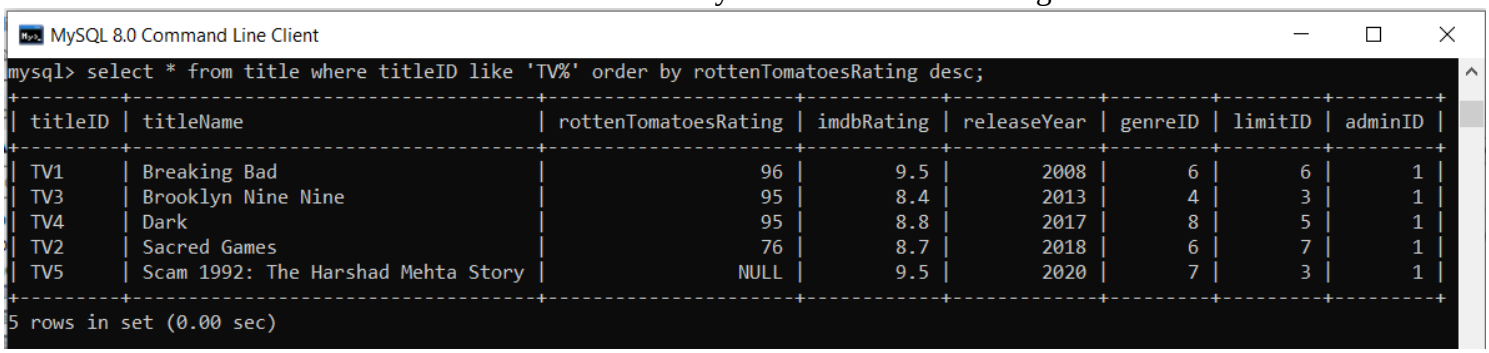
xvii. Get the names of TV Shows released after 2000.



xviii. Order the Movies by Rotten Tomatoes Rating.



xix. Order the TV Shows by Rotten Tomatoes Rating.



xx. Group the Titles according to their Genre.

MySQL 8.0 Command Line Client

```
mysql> select t.titleName, g.genreName from title t inner join genre g on t.genreID=g.genreID group by genreName;
+----------------------------------+-------------+
| titleName                        | genreName   |
+----------------------------------+-------------+
| Forrest Gump                     | Drama       |
| Avengers Endgame                 | Superhero   |
| The Dark Knight                  | Thriller    |
| Dumb and Dumber                  | Comedy      |
| Titanic                          | Romantic    |
| Breaking Bad                     | Crime Drama |
| Scam 1992: The Harshad Mehta Story | Crime       |
| Dark                             | Sci-Fi      |
+----------------------------------+-------------+
8 rows in set (0.00 sec)
```

xxi. Get all the Titles that have an Age Limit of '18+'.

MySQL 8.0 Command Line Client

```
mysql> select * from title where limitID in (select limitID from agelimit where limitName='18+');
+---------+-------------+----------------------+------------+-------------+---------+---------+---------+
| titleID | titleName   | rottenTomatoesRating | imdbRating | releaseYear | genreID | limitID | adminID |
+---------+-------------+----------------------+------------+-------------+---------+---------+---------+
| TV2     | Sacred Games |                   76 |        8.7 |        2018 |       6 |       7 |       1 |
+---------+-------------+----------------------+------------+-------------+---------+---------+---------+
1 row in set (0.00 sec)
```

xxii. Get all the Titles whose Genre is 'Crime Drama'.

MySQL 8.0 Command Line Client

```
mysql> select * from title where genreID in (select genreID from genre where genreName='Crime Drama');
+---------+-------------+----------------------+------------+-------------+---------+---------+---------+
| titleID | titleName   | rottenTomatoesRating | imdbRating | releaseYear | genreID | limitID | adminID |
+---------+-------------+----------------------+------------+-------------+---------+---------+---------+
| TV1     | Breaking Bad |                   96 |        9.5 |        2008 |       6 |       6 |       1 |
| TV2     | Sacred Games |                   76 |        8.7 |        2018 |       6 |       7 |       1 |
+---------+-------------+----------------------+------------+-------------+---------+---------+---------+
2 rows in set (0.00 sec)
```

xxiii. Get all the Movies produced by 'Marvel Studios'.

MySQL 8.0 Command Line Client

```
mysql> select * from title t where t.titleID in (select m.titleID from movie m where phID in (select phID from productionhouse wh
ere phName="Marvel Studios"));
+---------+-----------------+----------------------+------------+-------------+---------+---------+---------+
| titleID | titleName       | rottenTomatoesRating | imdbRating | releaseYear | genreID | limitID | adminID |
+---------+-----------------+----------------------+------------+-------------+---------+---------+---------+
| MO2     | Avengers Endgame |                   94 |        8.4 |        2019 |       2 |       2 |       1 |
+---------+-----------------+----------------------+------------+-------------+---------+---------+---------+
1 row in set (0.00 sec)
```

xxiv. Get all the Titles which are available in ‘German’ or ‘Spanish’.

MySQL 8.0 Command Line Client

```
mysql> select * from title where titleID in (select titleID from availablein where languageID in (select languageID from language
s where languageName in ('German', 'Spanish')));
+---------+-----------------+----------------------+------------+-------------+---------+---------+---------+
| titleID | titleName       | rottenTomatoesRating | imdbRating | releaseYear | genreID | limitID | adminID |
+---------+-----------------+----------------------+------------+-------------+---------+---------+---------+
| MO1     | Forrest Gump    |                   71 |        8.8 |        1994 |       1 |       2 |       1 |
| MO2     | Avengers Endgame|                   94 |        8.4 |        2019 |       2 |       2 |       1 |
| MO4     | Dumb and Dumber |                   68 |        7.3 |        1994 |       4 |       4 |       1 |
| TV1     | Breaking Bad    |                   96 |        9.5 |        2008 |       6 |       6 |       1 |
| TV4     | Dark            |                   95 |        8.8 |        2017 |       8 |       5 |       1 |
+---------+-----------------+----------------------+------------+-------------+---------+---------+---------+
5 rows in set (0.00 sec)
```

xxv. Get all the Titles streaming on ‘Sony Liv’.

MySQL 8.0 Command Line Client

```
mysql> select * from title where titleID in (select titleID from streamingon where ottID in (select ottID from ott where ottName=
'Sony Liv'));
+---------+-----------------------------------+----------------------+------------+-------------+---------+---------+---------+
| titleID | titleName                         | rottenTomatoesRating | imdbRating | releaseYear | genreID | limitID | adminID |
+---------+-----------------------------------+----------------------+------------+-------------+---------+---------+---------+
| MO3     | The Dark Knight                   |                   94 |        9.0 |        2008 |       3 |       1 |       1 |
| TV5     | Scam 1992: The Harshad Mehta Story|                 NULL |        9.5 |        2020 |       7 |       3 |       1 |
+---------+-----------------------------------+----------------------+------------+-------------+---------+---------+---------+
2 rows in set (0.00 sec)
```

b. 5 Functions

i. Function to get the number of Movies/TV Shows that are present in the database.

MySQL 8.0 Command Line Client

```
mysql> delimiter @@
mysql> create function noOfRecords(titleType text)
    -> returns int
    -> deterministic
    -> begin
    -> declare noRecords int;
    -> declare recordType text;
    -> select concat(titleType, '%') into recordType;
    -> select count(titleID) from title where titleID like recordType into noRecords;
    -> return noRecords;
    -> end @@
Query OK, 0 rows affected (0.01 sec)

mysql> select noOfRecords('TV') as NoOfRecords;
    -> \g
+-------------+
| NoOfRecords |
+-------------+
|           5 |
+-------------+
1 row in set (0.00 sec)

mysql> select noOfRecords('MO') as NoOfRecords\g
+-------------+
| NoOfRecords |
+-------------+
|           5 |
+-------------+
1 row in set (0.00 sec)
```

ii. Function to get the number of years that have passed since a particular title was released.

MySQL 8.0 Command Line Client

```
mysql> delimiter @@
mysql> create function noOfYears(tName text)
    -> returns int
    -> deterministic
    -> begin
    -> declare currentYear year(4);
    -> declare titleYear year(4);
    -> select extract(year from current_date()) into currentYear;
    -> select releaseYear from title where titleName=tName into titleYear;
    -> return currentYear-titleYear;
    -> end @@
Query OK, 0 rows affected (0.01 sec)

mysql> select noOfYears('Dumb and Dumber') as NoOfYears
    -> \g
+-----------+
| NoOfYears |
+-----------+
|        26 |
+-----------+
1 row in set (0.00 sec)
```

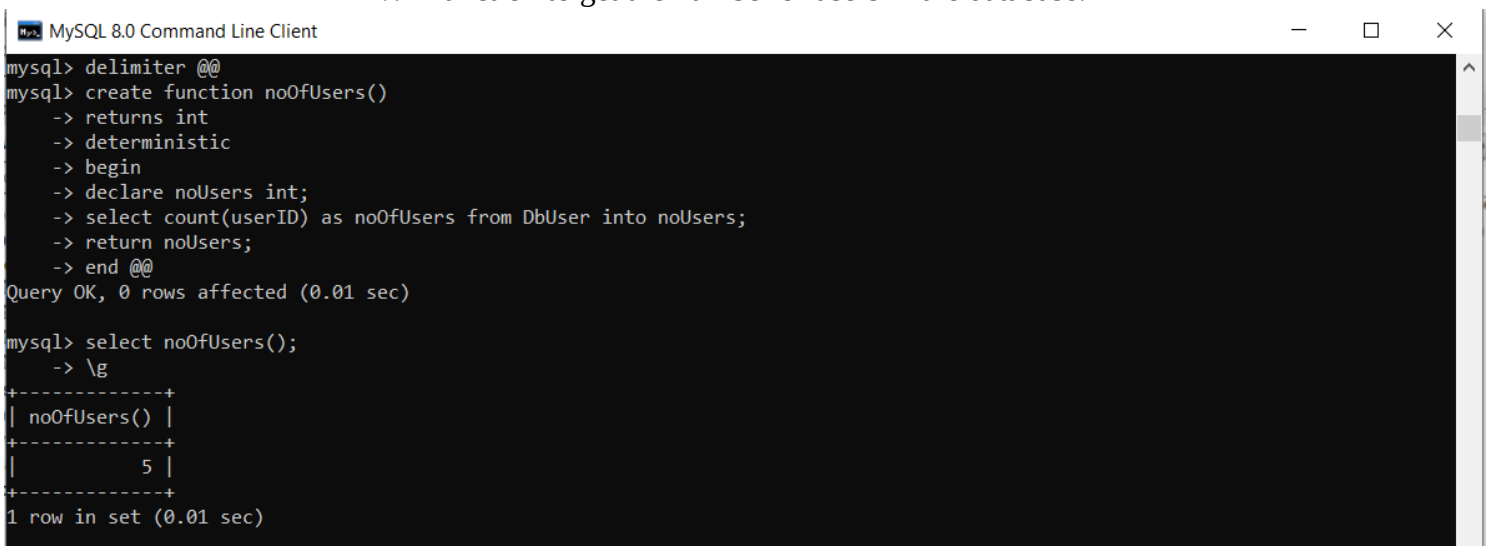
iii. Function to get the highest rated title (IMDB) of a particular genre.

MySQL 8.0 Command Line Client

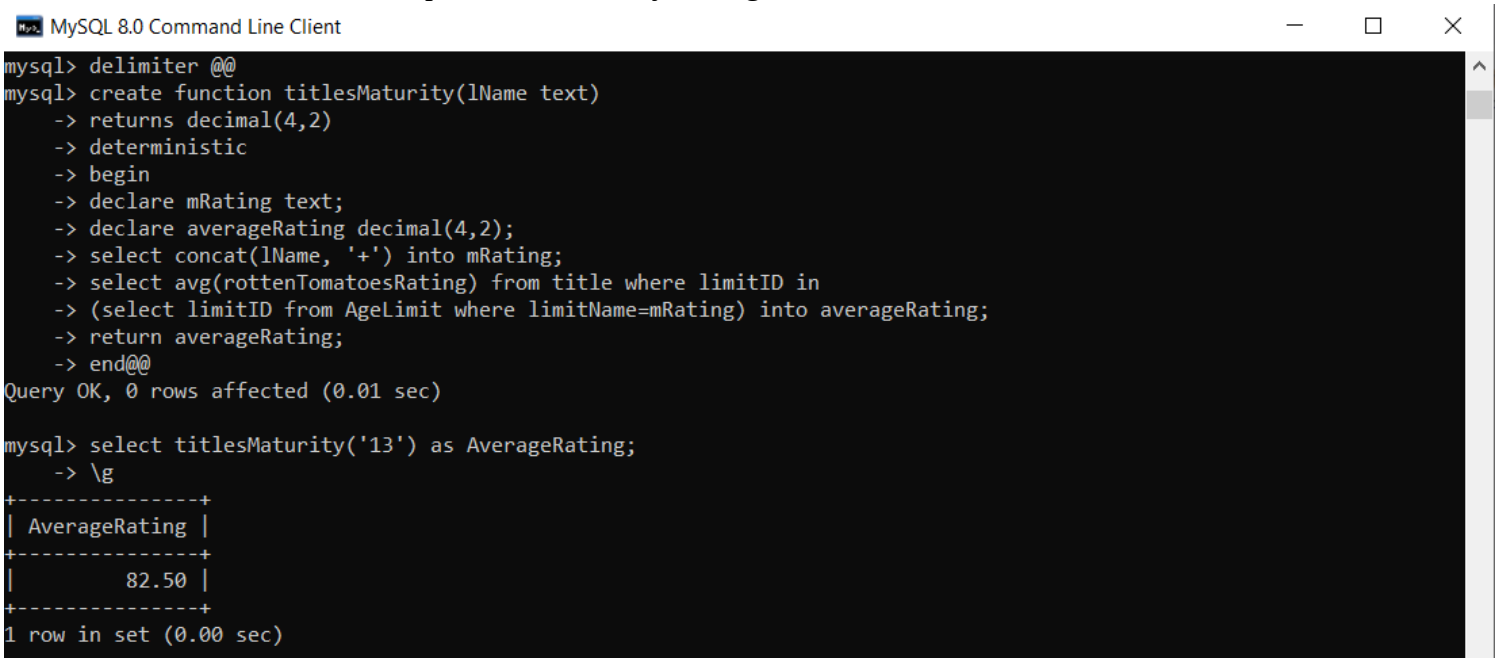
```
mysql> delimiter @@
mysql> create function maxRating(gName varchar(25))
    -> returns varchar(50)
    -> deterministic
    -> begin
    -> declare MaxRatingShow varchar(50);
    -> declare MaxImdbRating decimal(2,1);
    -> declare MaxGenre int;
    -> select g.genreID from genre g where g.genreName=gName into MaxGenre;
    -> select max(imdbRating) from title group by genreID having genreID=MaxGenre into MaxImdbRating;
    -> select t.titleName from title t where (t.imdbRating=MaxImdbRating and t.genreID=MaxGenre) group by imdbRating into MaxRati
ngShow;
    -> return MaxRatingShow;
    -> end @@
Query OK, 0 rows affected (0.01 sec)

mysql> select maxRating('Comedy') as MaxRatingShow from title
    -> \g
+--------------------+
| MaxRatingShow      |
+--------------------+
| Brooklyn Nine Nine |
+--------------------+
1 row in set (0.00 sec)
```

iv. Function to get the number of users in the database.

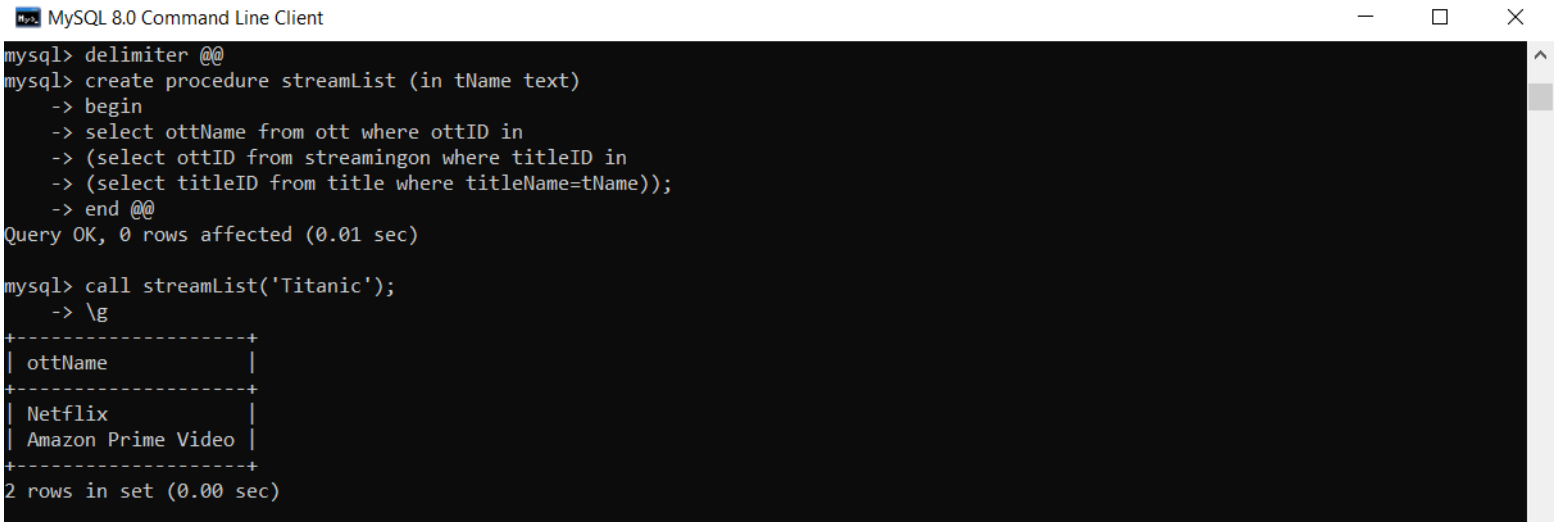


v. Function to get the average Rotten Tomatoes Rating of titles of a particular Maturity Rating.



c. 5 Procedures

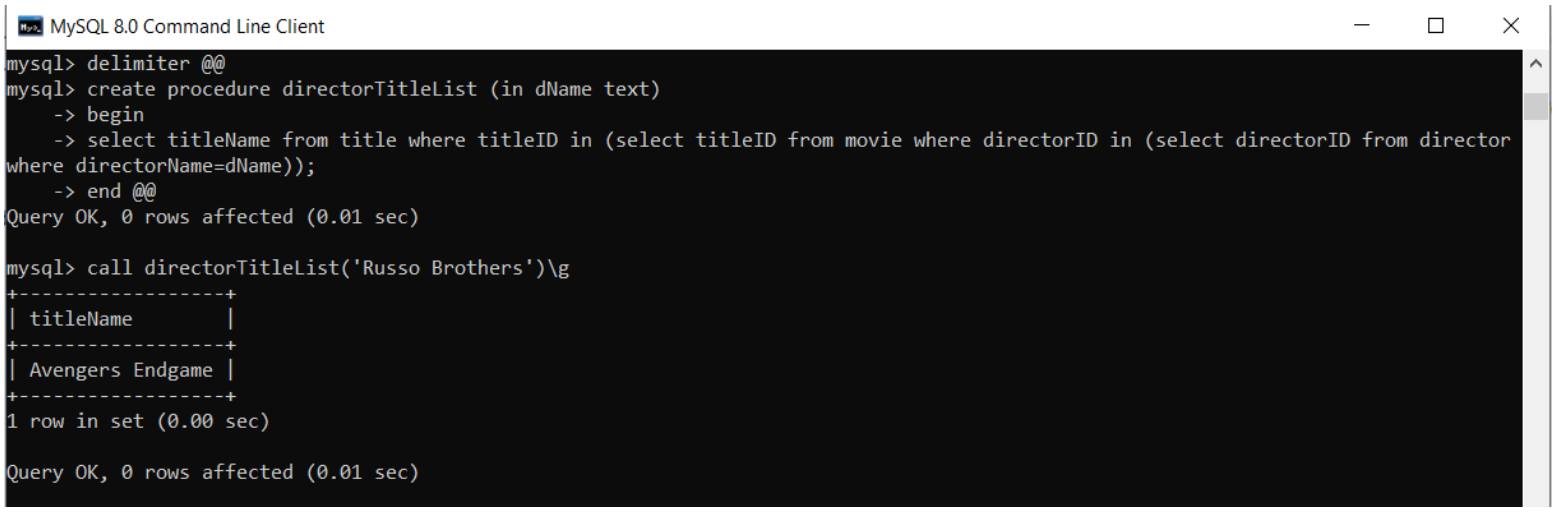
i. Procedure to get the names of OTTs on which a particular title is available.



```
MySQL 8.0 Command Line Client
mysql> delimiter @@
mysql> create procedure streamList (in tName text)
    -> begin
    -> select ottName from ott where ottID in
    -> (select ottID from streamingon where titleID in
    -> (select titleID from title where titleName=tName));
    -> end @@
Query OK, 0 rows affected (0.01 sec)

mysql> call streamList('Titanic');
    -> \g
+--------------------+
| ottName            |
+--------------------+
| Netflix            |
| Amazon Prime Video |
+--------------------+
2 rows in set (0.00 sec)
```

ii. Procedure to get the names of the Titles directed by a particular Director.

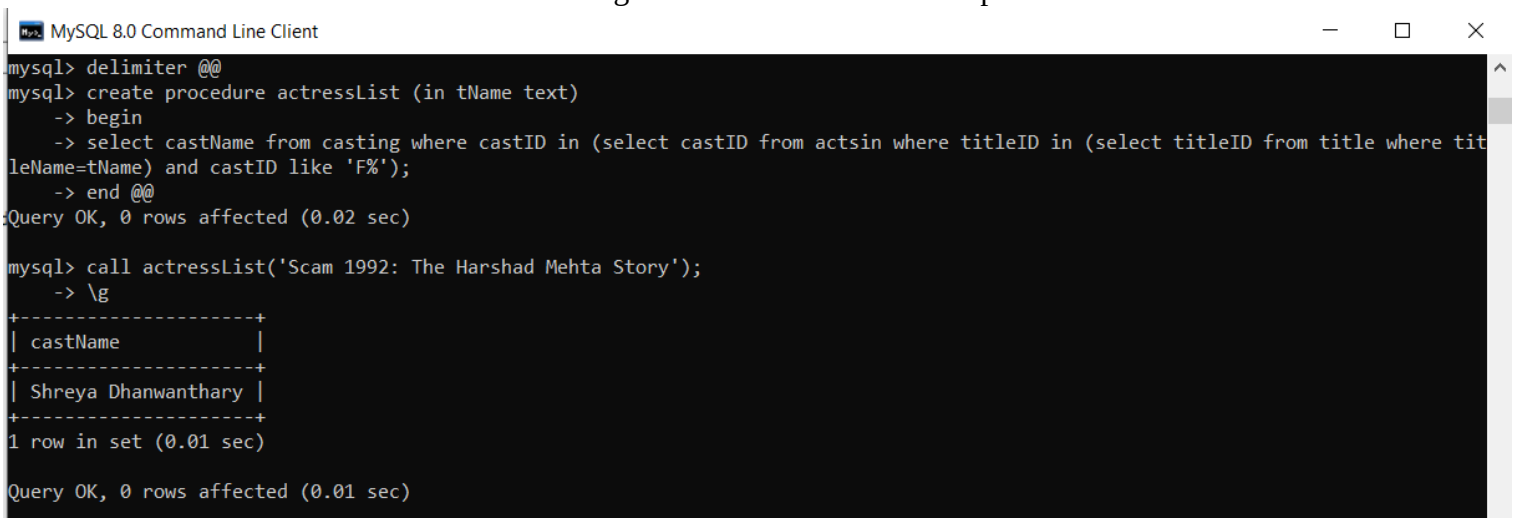


```
MySQL 8.0 Command Line Client
mysql> delimiter @@
mysql> create procedure directorTitleList (in dName text)
    -> begin
    -> select titleName from title where titleID in (select titleID from movie where directorID in (select directorID from director
where directorName=dName));
    -> end @@
Query OK, 0 rows affected (0.01 sec)

mysql> call directorTitleList('Russo Brothers')\g
+-----------------+
| titleName       |
+-----------------+
| Avengers Endgame |
+-----------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)
```

iii. Procedure to get the name of Actress of a particular Title.



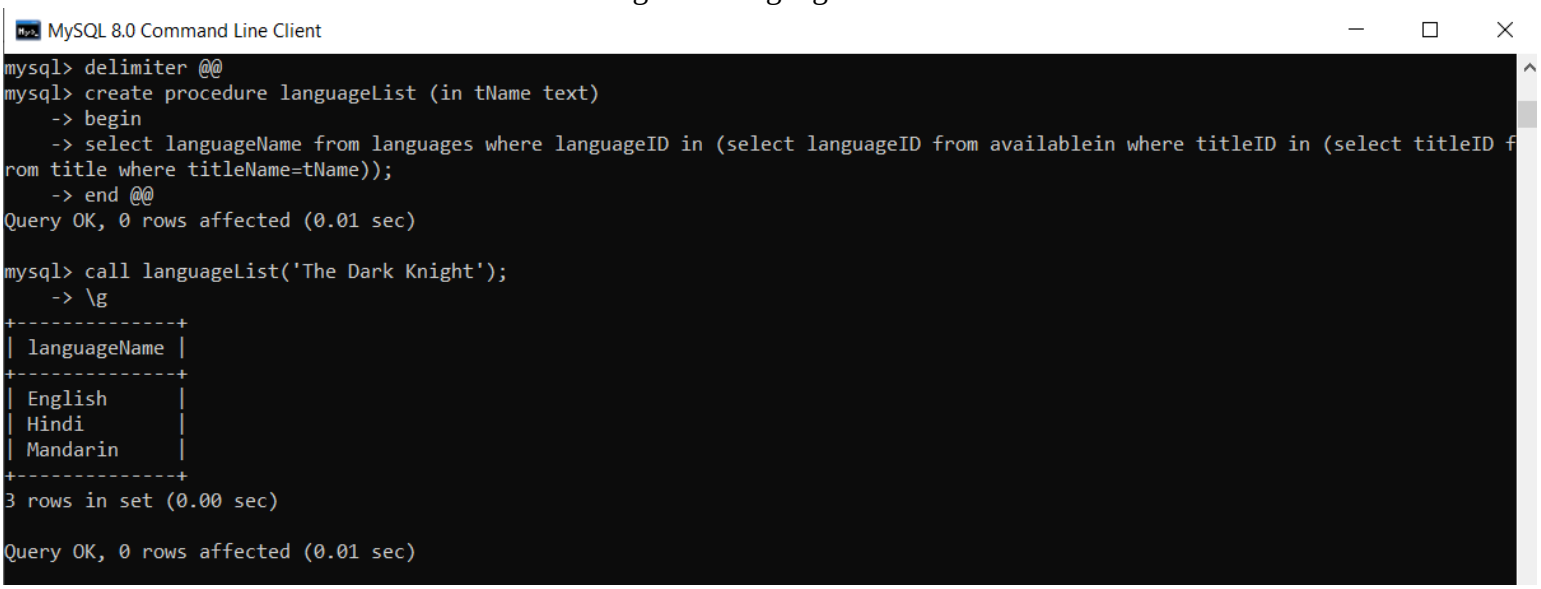
```
MySQL 8.0 Command Line Client
mysql> delimiter @@
mysql> create procedure actressList (in tName text)
    -> begin
    -> select castName from casting where castID in (select castID from actsin where titleID in (select titleID from title where tit
leName=tName) and castID like 'F%');
    -> end @@
Query OK, 0 rows affected (0.02 sec)

mysql> call actressList('Scam 1992: The Harshad Mehta Story');
    -> \g
+---------------------+
| castName            |
+---------------------+
| Shreya Dhanwanthary |
+---------------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)
```

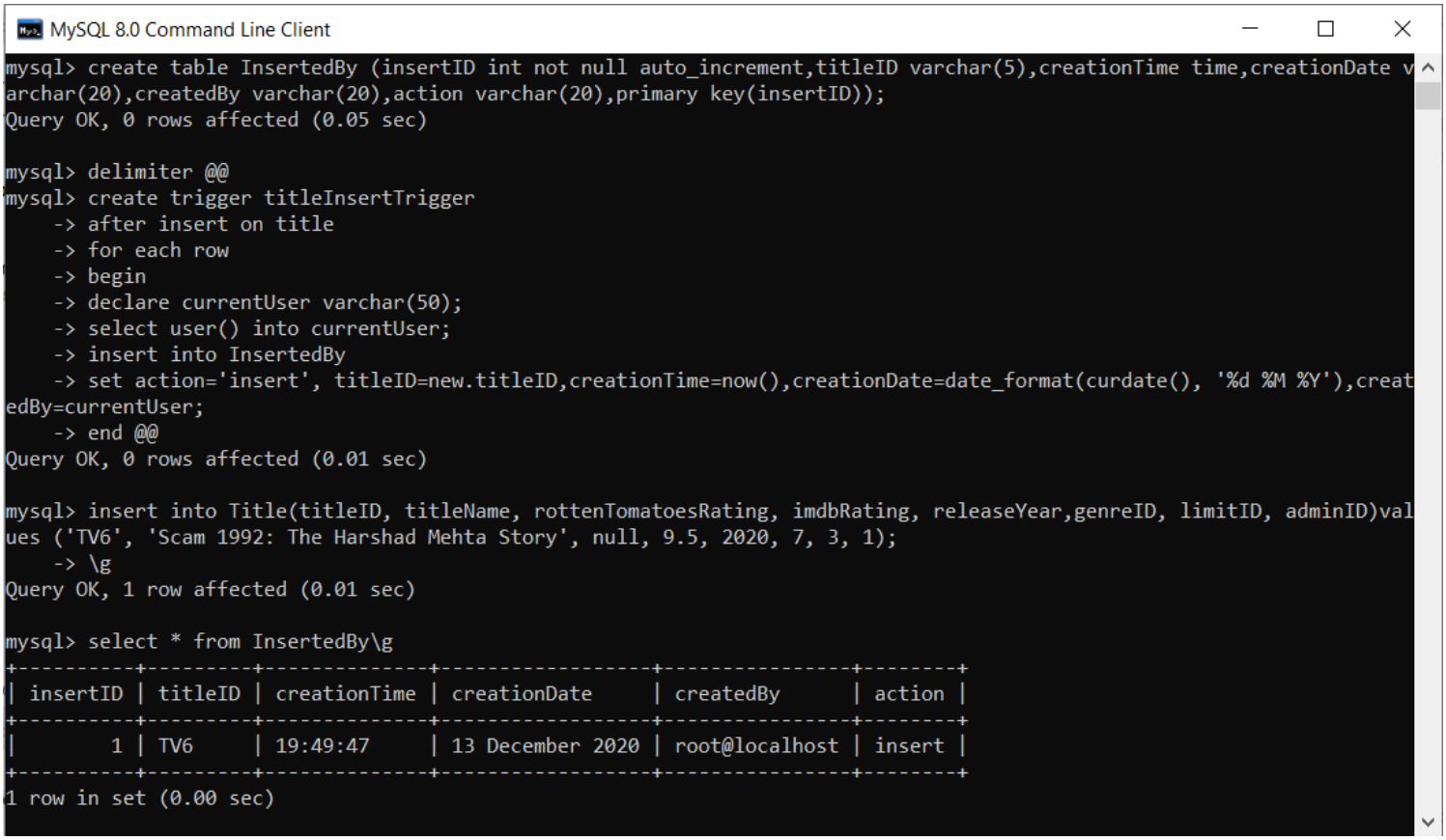
iv. Procedure to get the name of Titles belonging to a particular Genre.

v. Procedure to get the languages in which the title is available.



d. 5 Triggers

i. Trigger to store the time at which a Title was inserted into the 'title' table.



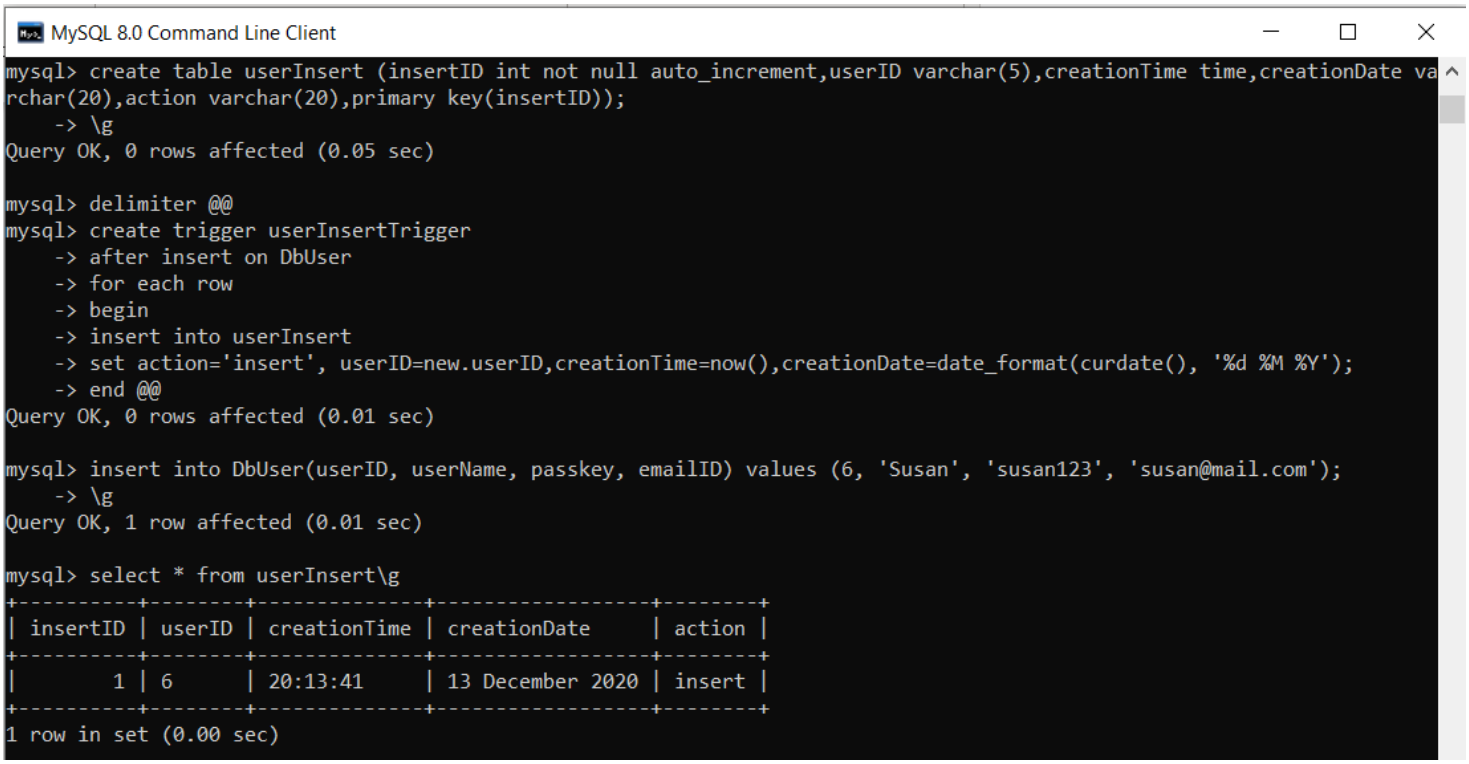
```
MySQL 8.0 Command Line Client
mysql> create table InsertedBy (insertID int not null auto_increment,titleID varchar(5),creationTime time,creationDate v
archar(20),createdBy varchar(20),action varchar(20),primary key(insertID));
Query OK, 0 rows affected (0.05 sec)

mysql> delimiter @@
mysql> create trigger titleInsertTrigger
    -> after insert on title
    -> for each row
    -> begin
    -> declare currentUser varchar(50);
    -> select user() into currentUser;
    -> insert into InsertedBy
    -> set action='insert', titleID=new.titleID,creationTime=now(),creationDate=date_format(curdate(), '%d %M %Y'),creat
edBy=currentUser;
    -> end @@
Query OK, 0 rows affected (0.01 sec)

mysql> insert into Title(titleID, titleName, rottenTomatoesRating, imdbRating, releaseYear,genreID, limitID, adminID)val
ues ('TV6', 'Scam 1992: The Harshad Mehta Story', null, 9.5, 2020, 7, 3, 1);
    -> \g
Query OK, 1 row affected (0.01 sec)

mysql> select * from InsertedBy\g
+----------+---------+--------------+------------------+----------------+--------+
| insertID | titleID | creationTime | creationDate     | createdBy      | action |
+----------+---------+--------------+------------------+----------------+--------+
|        1 | TV6     | 19:49:47     | 13 December 2020 | root@localhost | insert |
+----------+---------+--------------+------------------+----------------+--------+
1 row in set (0.00 sec)
```

ii. Trigger to store the time and date at which user information was inserted into the database.



```
MySQL 8.0 Command Line Client
mysql> create table userInsert (insertID int not null auto_increment,userID varchar(5),creationTime time,creationDate va
rchar(20),action varchar(20),primary key(insertID));
    -> \g
Query OK, 0 rows affected (0.05 sec)

mysql> delimiter @@
mysql> create trigger userInsertTrigger
    -> after insert on DbUser
    -> for each row
    -> begin
    -> insert into userInsert
    -> set action='insert', userID=new.userID,creationTime=now(),creationDate=date_format(curdate(), '%d %M %Y');
    -> end @@
Query OK, 0 rows affected (0.01 sec)

mysql> insert into DbUser(userID, userName, passkey, emailID) values (6, 'Susan', 'susan123', 'susan@mail.com');
    -> \g
Query OK, 1 row affected (0.01 sec)

mysql> select * from userInsert\g
+----------+--------+--------------+------------------+--------+
| insertID | userID | creationTime | creationDate     | action |
+----------+--------+--------------+------------------+--------+
|        1 | 6      | 20:13:41     | 13 December 2020 | insert |
+----------+--------+--------------+------------------+--------+
1 row in set (0.00 sec)
```

iii. Trigger to store old user data when user updates his/her data.

MySQL 8.0 Command Line Client

```
mysql> create table userUpdate (updateID int not null auto_increment,userID int,userName varchar(25),passkey varchar(50)
,emailID varchar(50),action varchar(20),primary key(updateID))\g
Query OK, 0 rows affected (0.05 sec)

mysql> delimiter @@
mysql> create trigger userUpdateTrigger
    -> after update on DbUser
    -> for each row
    -> begin
    -> insert into userUpdate
    -> set action='update', userID=old.userID,userName=old.userName,passkey=old.passkey,emailID=old.emailID;
    -> end @@
Query OK, 0 rows affected (0.01 sec)

mysql> update DbUser set emailID=
    ->
    -> 'susan2000@mail.com' where emailID='susan@mail.com'\g
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from userUpdate\g
+----------+--------+----------+----------+----------------+--------+
| updateID | userID | userName | passkey  | emailID        | action |
+----------+--------+----------+----------+----------------+--------+
|        1 |      6 | Susan    | susan123 | susan@mail.com | update |
+----------+--------+----------+----------+----------------+--------+
1 row in set (0.00 sec)

mysql> select * from DbUser;
    -> \g
```

MySQL 8.0 Command Line Client

```
mysql> select * from DbUser;
    -> \g
+--------+----------+-----------+--------------------+
| userID | userName | passkey   | emailID            |
+--------+----------+-----------+--------------------+
|      1 | Henry    | henry123  | henry@mail.com     |
|      2 | Eddie    | eddie123  | eddie@mail.com     |
|      3 | Ashley   | ashley123 | ashley@mail.com    |
|      4 | Daniel   | daniel123 | daniel@mail.com    |
|      5 | Clara    | clara123  | clara@mail.com     |
|      6 | Susan    | susan123  | susan2000@mail.com |
+--------+----------+-----------+--------------------+
6 rows in set (0.00 sec)
```

iv. Trigger to delete the old user data (which gets stored when user data is updated) if the user gets deleted

```
MySQL 8.0 Command Line Client
mysql> select * from userUpdate\g
+----------+--------+----------+----------+----------------+--------+
| updateID | userID | userName | passkey  | emailID        | action |
+----------+--------+----------+----------+----------------+--------+
|        1 |      6 | Susan    | susan123 | susan@mail.com | update |
+----------+--------+----------+----------+----------------+--------+
1 row in set (0.00 sec)

mysql> delimiter @@
mysql> create trigger userUpdateDeleteTrigger
    -> after delete on DbUser
    -> for each row
    -> begin
    -> delete from userUpdate where userID=old.userID;
    -> end @@
Query OK, 0 rows affected (0.01 sec)

mysql> delete from DbUser where userID=6\g
Query OK, 1 row affected (0.02 sec)

mysql> select * from userUpdate\g
Empty set (0.00 sec)
```

v. Trigger to store the time and date at which the user information was deleted from the database.

```
MySQL 8.0 Command Line Client
mysql> create table userDelete (deleteID int not null auto_increment,userID varchar(5),deletionTime time,deletionDate va
rchar(20),action varchar(20),primary key(deleteID));
    -> \g
Query OK, 0 rows affected (0.07 sec)

mysql> delimiter @@
mysql> create trigger userDeleteTrigger
    -> after delete on DbUser
    -> for each row
    -> begin
    -> insert into userDelete
    -> set action='delete', userID=old.userID,deletionTime=now(),deletionDate=date_format(curdate(), '%d %M %Y');
    -> end @@
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> delete from DbUser where userID=6\g
Query OK, 1 row affected (0.02 sec)

mysql> select * from userUpdate\g
Empty set (0.00 sec)

mysql> select * from userDelete\g
+----------+--------+--------------+------------------+--------+
| deleteID | userID | deletionTime | deletionDate     | action |
+----------+--------+--------------+------------------+--------+
|        1 | 6      | 20:24:09     | 13 December 2020 | delete |
+----------+--------+--------------+------------------+--------+
1 row in set (0.00 sec)
```