DBMS – Mini Project
Netflix Database

Submitted By:

Name: Shashank Varma

SRN: PES1UG20CS395

V Semester: Section G

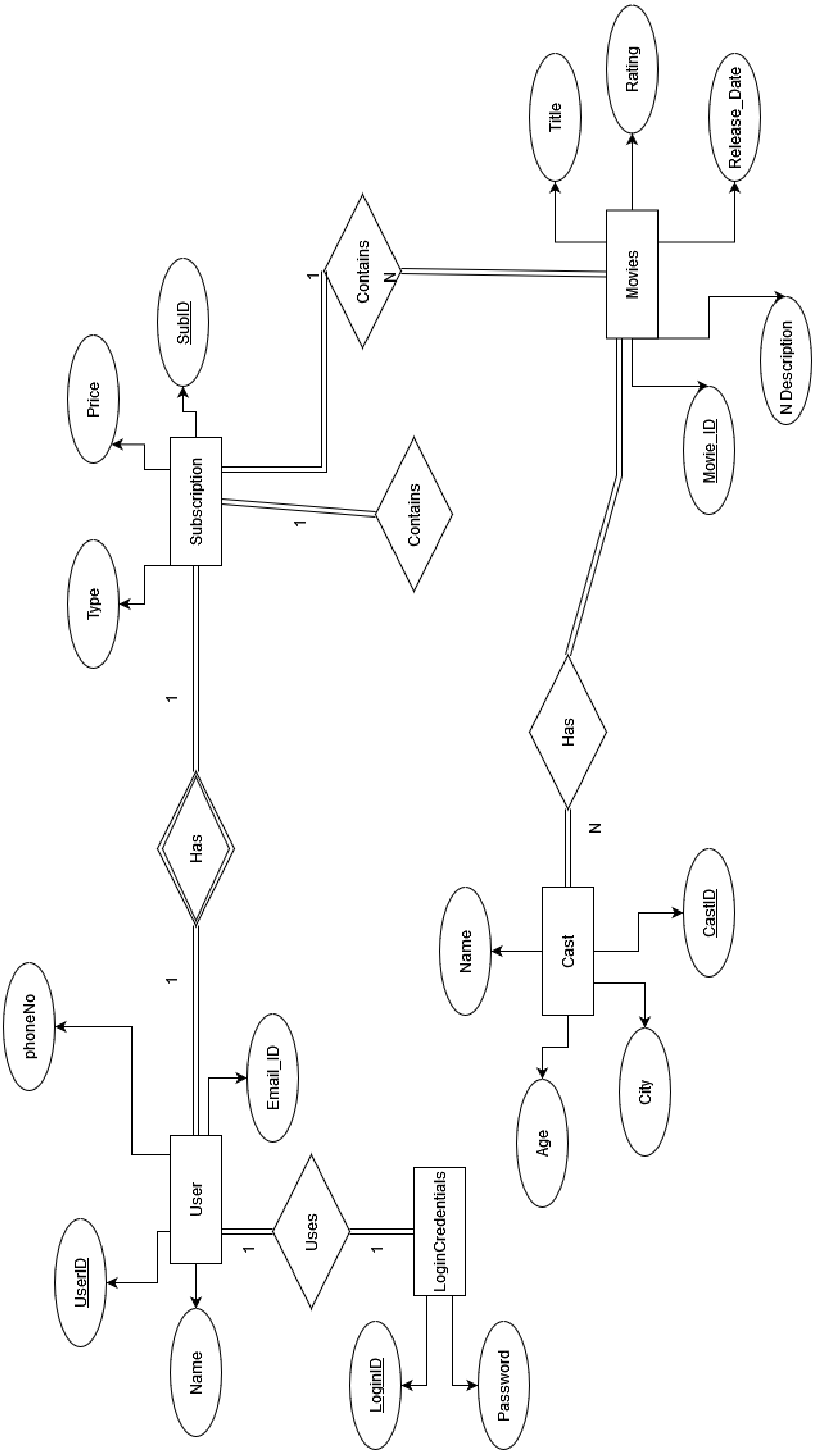
Short Description and Scope of Project

This is a clone of the Netflix Database done using mySQL and Streamlit.

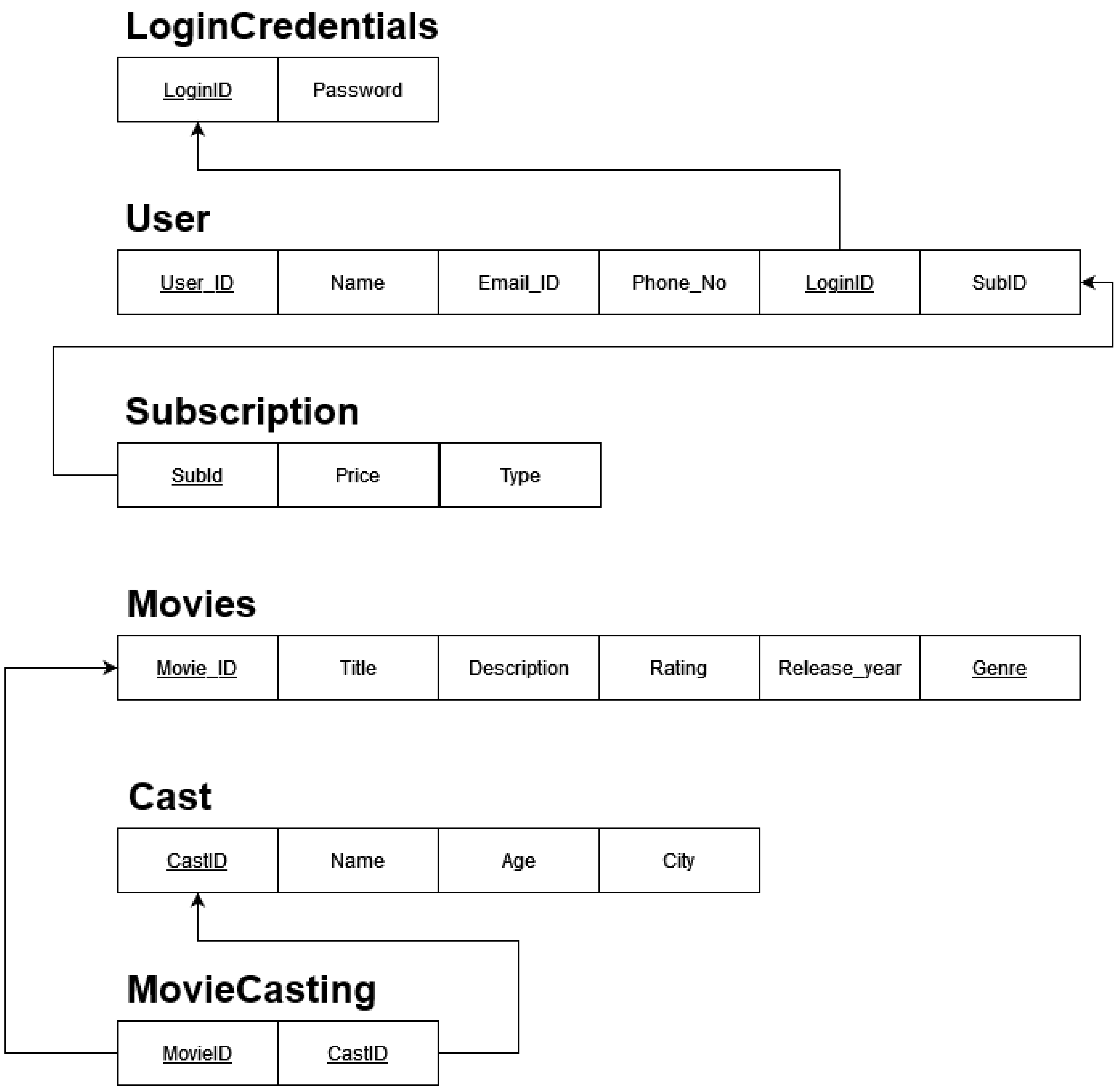
It consists of 5 tables – User, Login, Movies, Cast and MovieCasting relation. The user has to first login using his credentials. If he is a new user, he can register first and then login. He can then use the home page where all list of movies available are shown along with the average movie rating. After selecting one of them, he can view the full information of the movie in the movie page along with the cast, short summary and ratings. There is also an option to rate the movie. On rating, the rating gets updated to the average of the new and old rating.

The admin has some functions available at backend which can be used to get information regarding the users, their subscriptions, cast info and so on.

ER Diagram


User
UserID
Name
phoneNo
Email_ID
Uses
LoginCredentials
LoginID
Password
Has
Subscription
SubID
Price
Type
Contains
Contains
Movies
Title
Rating
Release_Date
Description
Movie_ID
Has
Cast
CastID
Name
Age
City
1
1
1
1
1
1
1
N
N
N

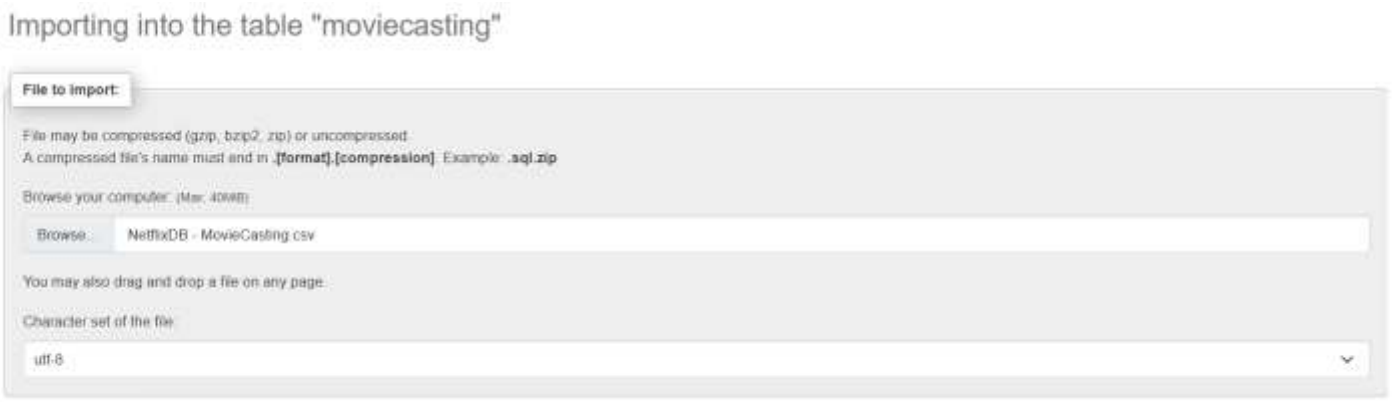
Relational Schema



DDL Statements – Building Database

```
CREATE TABLE loginCredentials (LoginID varchar(20) NOT NULL, password varchar(20),
PRIMARY KEY(LoginId));

CREATE TABLE user (userId varchar(20) NOT NULL, name varchar(20), emailId varchar(20),
phoneNo BIGINT(10),LoginId varchar(20), PRIMARY KEY(UserId), FOREIGN KEY(LoginId)
REFERENCES loginCredentials(LoginId));

CREATE TABLE movie (movieId int(10) NOT NULL, title varchar(20), description varchar(20),
rating float(10), releaseYear YEAR, genre varchar(20), PRIMARY KEY(movieId));

CREATE TABLE webSeries (seriesId int(10) NOT NULL, title varchar(20), description
varchar(20), rating float(10), releaseYear YEAR, genre varchar(20), noOfEpisodes INT(10),
PRIMARY KEY(seriesId));

CREATE TABLE subscription (subId INT(10) NOT NULL, type varchar(20), price float(10),
PRIMARY KEY(subId));

CREATE TABLE castInfo (castId INT(10) NOT NULL, name VARCHAR(20), age INT(10), city
VARCHAR(20), PRIMARY KEY(castId));

CREATE TABLE moviecasting (castId INT(10) NOT NULL, movieId int(10) NOT NULL, FOREIGN
KEY(castId) REFERENCES castInfo(castId), FOREIGN KEY(movieId) REFERENCES
movie(movieId));

ALTER TABLE user ADD COLUMN subId INT(10);

ALTER TABLE user ADD CONSTRAINT FOREIGN KEY(subId) REFERENCES subscription(subId);
```

Populating the Database



Importing into the table "moviecasting"

File to import:

File may be compressed (gzip, bzip2, zip) or uncompressed.
A compressed file's name must end in **.[format].[compression]**. Example: **.sql.zip**

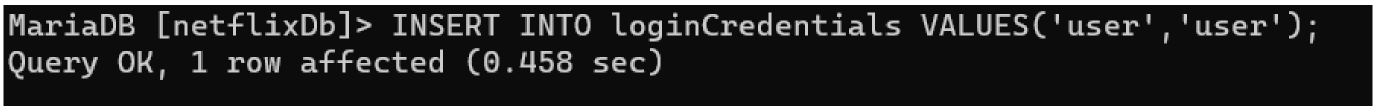
Browse your computer: (Max: 40MiB)

Browse... NetflixDB - MovieCasting.csv

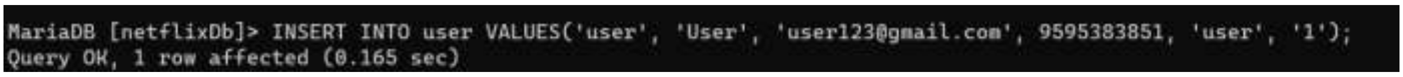
You may also drag and drop a file on any page.

Character set of the file:

utf-8



```
MariaDB [netflixDb]> INSERT INTO loginCredentials VALUES('user','user');
Query OK, 1 row affected (0.458 sec)
```

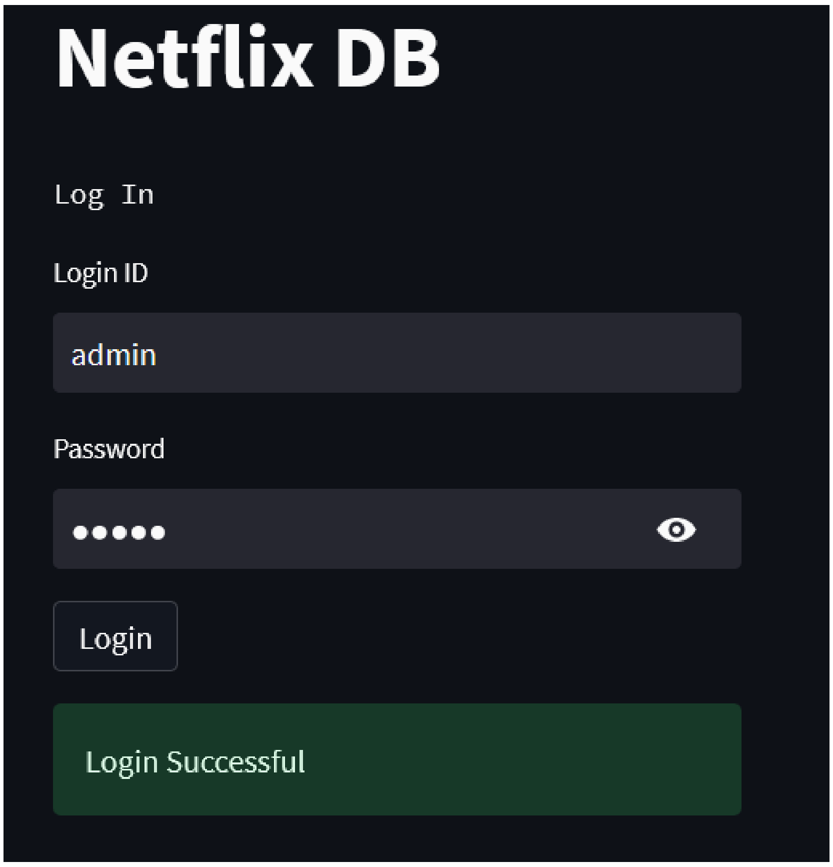


```
MariaDB [netflixDb]> INSERT INTO user VALUES('user', 'User', 'user123@gmail.com', 9595383851, 'user', '1');
Query OK, 1 row affected (0.165 sec)
```

Join Queries

1. This performs JOIN on user and login credentials to fetch the user information as soon as he logs in.

SELECT * FROM logincredentials, user WHERE logincredentials.LoginId = %s AND logincredentials.password = %s AND logincredentials.LoginId = user.LoginId


Netflix DB

Log In

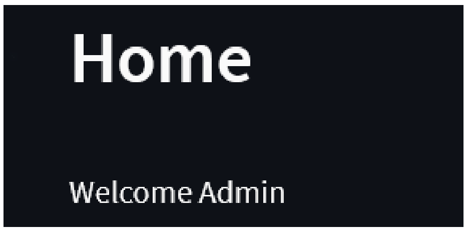
Login ID

admin

Password

●●●●●

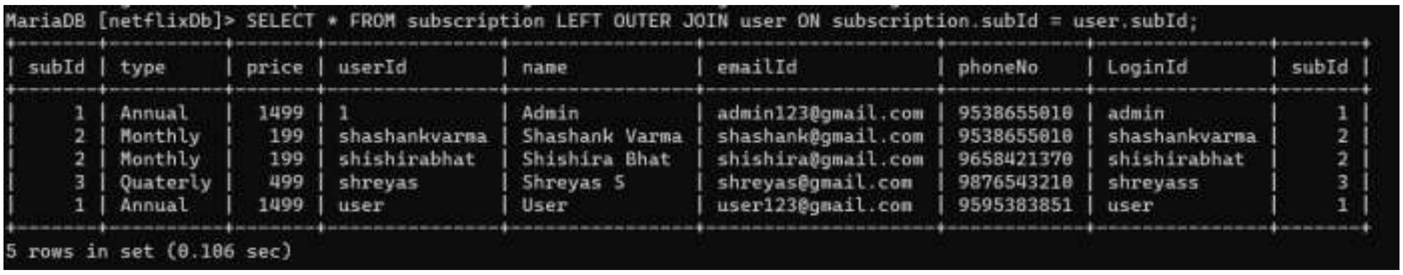
Login Successful


Home

Welcome Admin

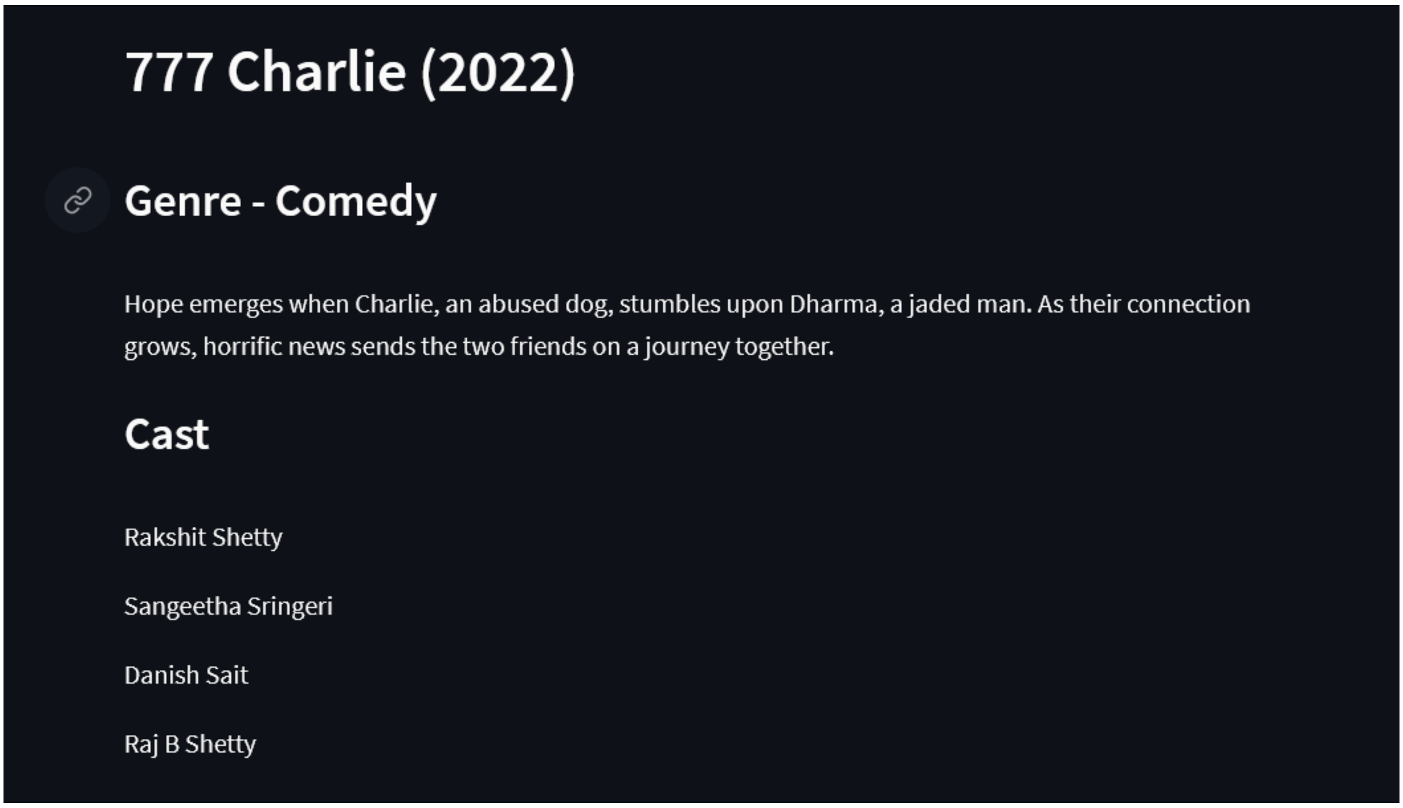
2. This performs a LEFT OUTER JOIN on the user and subscription tables for the admin to see the users and the list of subscription he/she has made.

SELECT * FROM subscription LEFT OUTER JOIN user ON subscription.subId = user.subId;


```
MariaDB [netflixDb]> SELECT * FROM subscription LEFT OUTER JOIN user ON subscription.subId = user.subId;
+-------+----------+-------+---------------+----------------+--------------------+------------+---------------+-------+
| subId | type     | price | userId        | name           | emailId            | phoneNo    | LoginId       | subId |
+-------+----------+-------+---------------+----------------+--------------------+------------+---------------+-------+
|     1 | Annual   |  1499 | 1             | Admin          | admin123@gmail.com | 9538655010 | admin         |     1 |
|     2 | Monthly  |   199 | shashankvarma | Shashank Varma | shashank@gmail.com | 9538655010 | shashankvarma |     2 |
|     2 | Monthly  |   199 | shishirabhat  | Shishira Bhat  | shishira@gmail.com | 9658421370 | shishirabhat  |     2 |
|     3 | Quaterly |   499 | shreyas       | Shreyas S      | shreyas@gmail.com  | 9876543210 | shreyass      |     3 |
|     1 | Annual   |  1499 | user          | User           | user123@gmail.com  | 9595383851 | user          |     1 |
+-------+----------+-------+---------------+----------------+--------------------+------------+---------------+-------+
5 rows in set (0.106 sec)
```

3. This performs a join on the movie and cast tables to retrieve the list of cast for a particular movie. This is displayed on the movie info page.

SELECT movieCast.name, movie.title FROM movieCast, movie WHERE movieCast.movieId=movie.movieId;



777 Charlie (2022)

Genre - Comedy

Hope emerges when Charlie, an abused dog, stumbles upon Dharma, a jaded man. As their connection grows, horrific news sends the two friends on a journey together.

Cast

Rakshit Shetty

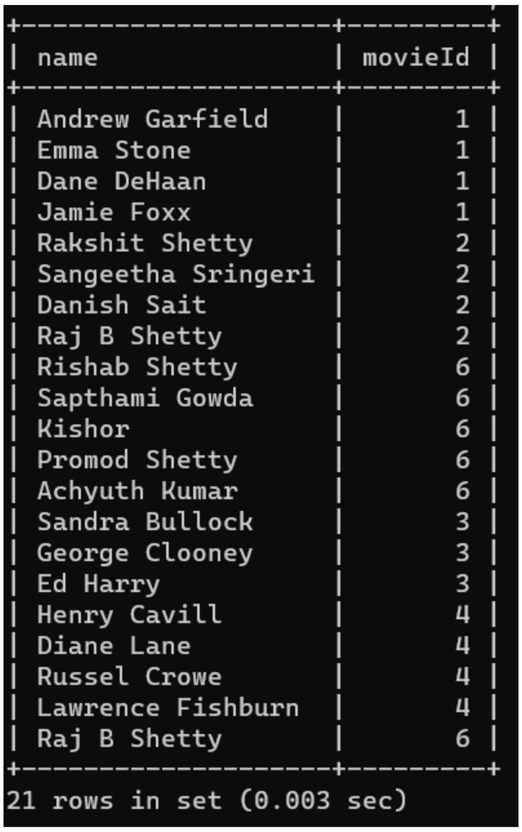
Sangeetha Sringeri

Danish Sait

Raj B Shetty

4. This join is FULL OUTER JOIN on the movieCasting relation and cast table to retrieve the list of all movies and all the cast.

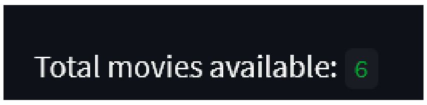
SELECT * FROM (SELECT castInfo.name,movieCasting.movieId FROM castInfo RIGHT OUTER JOIN movieCasting ON castInfo.castId=movieCasting.castId UNION SELECT castInfo.name,movieCasting.movieId FROM castInfo LEFT OUTER JOIN movieCasting ON castInfo.castId=movieCasting.castId) as c;



```
+--------------------+---------+
| name               | movieId |
+--------------------+---------+
| Andrew Garfield    |       1 |
| Emma Stone         |       1 |
| Dane DeHaan        |       1 |
| Jamie Foxx         |       1 |
| Rakshit Shetty     |       2 |
| Sangeetha Sringeri |       2 |
| Danish Sait        |       2 |
| Raj B Shetty       |       2 |
| Rishab Shetty      |       6 |
| Sapthami Gowda     |       6 |
| Kishor             |       6 |
| Promod Shetty      |       6 |
| Achyuth Kumar      |       6 |
| Sandra Bullock     |       3 |
| George Clooney     |       3 |
| Ed Harry           |       3 |
| Henry Cavill       |       4 |
| Diane Lane         |       4 |
| Russel Crowe       |       4 |
| Lawrence Fishburn  |       4 |
| Raj B Shetty       |       6 |
+--------------------+---------+
21 rows in set (0.003 sec)
```

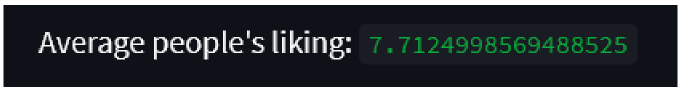
Aggregate Function

1. This is used to get a total count of movies available to watch.
   SELECT COUNT(*) FROM movie;



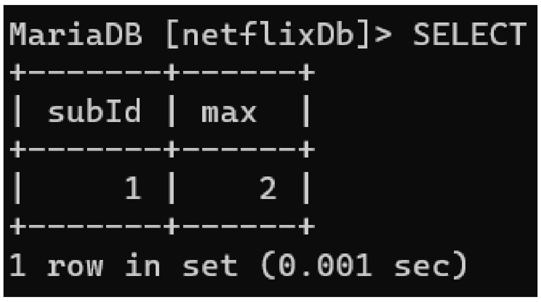
```
Total movies available: 6
```

2. This command is used to get the average rating of the movies on Netflix. This shows the people's liking towards current movies.
   SELECT AVG(movie.rating) FROM movie;



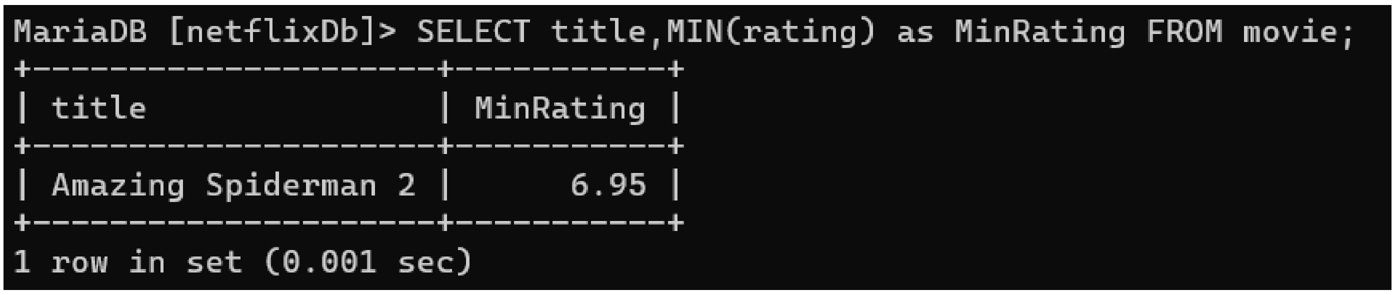
```
Average people's liking: 7.7124998569488525
```

3. This command is used to get the subscription with the most users.
   SELECT subId,MAX(subCount) AS max FROM (SELECT subId, COUNT(*) AS subCount FROM user GROUP BY subId) AS a;



```
MariaDB [netflixDb]> SELECT
+-------+------+
| subId | max  |
+-------+------+
|     1 |    2 |
+-------+------+
1 row in set (0.001 sec)
```

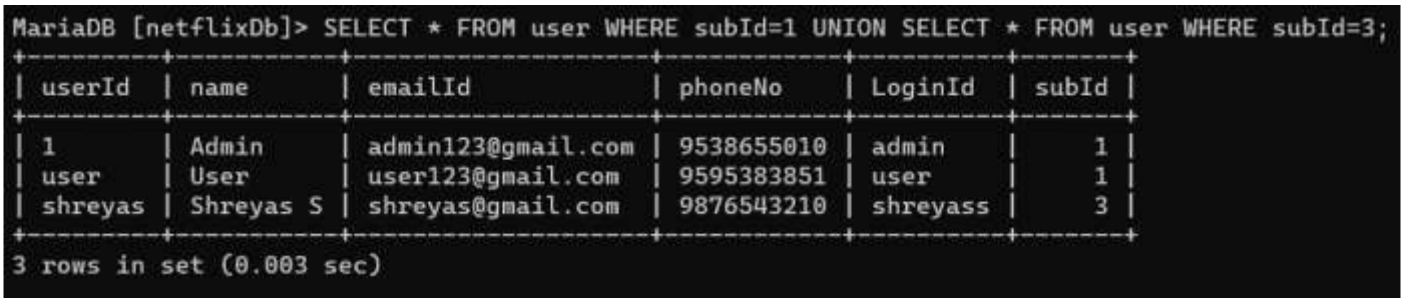
4. This command is used to find the movie with the least rating.
   SELECT title,MIN(rating) as MinRating FROM movie;



```
MariaDB [netflixDb]> SELECT title,MIN(rating) as MinRating FROM movie;
+---------------------+-----------+
| title               | MinRating |
+---------------------+-----------+
| Amazing Spiderman 2 |      6.95 |
+---------------------+-----------+
1 row in set (0.001 sec)
```

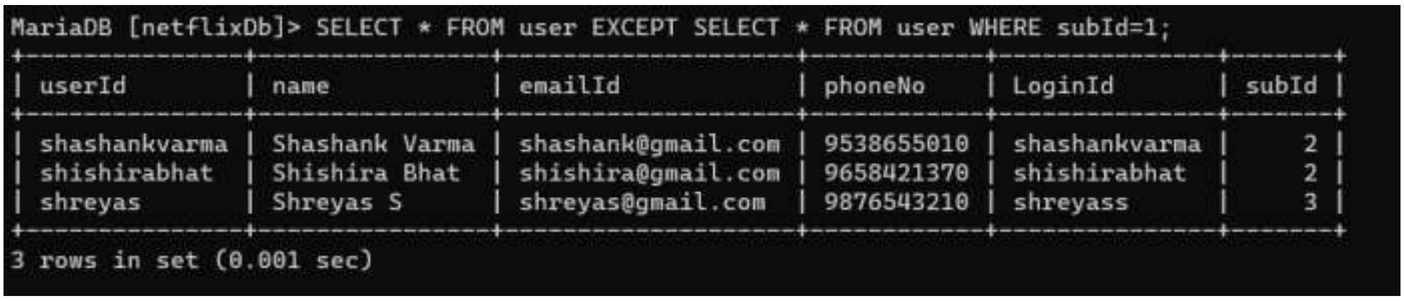
Set Operations

1. This command might be used to get list of all users with the annual and quarterly subscription in order to provide them some offer.
   SELECT * FROM user WHERE subId=1 UNION SELECT * FROM user WHERE subId=3;



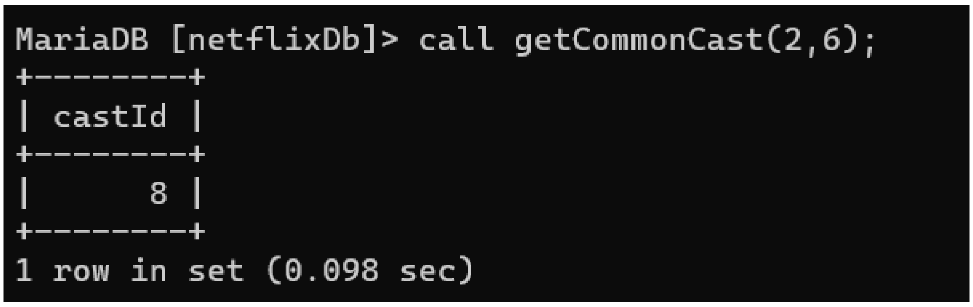
```
MariaDB [netflixDb]> SELECT * FROM user WHERE subId=1 UNION SELECT * FROM user WHERE subId=3;
+---------+-----------+-------------------+------------+----------+-------+
| userId  | name      | emailId           | phoneNo    | LoginId  | subId |
+---------+-----------+-------------------+------------+----------+-------+
| 1       | Admin     | admin123@gmail.com | 9538655010 | admin    |     1 |
| user    | User      | user123@gmail.com | 9595383851 | user     |     1 |
| shreyas | Shreyas S | shreyas@gmail.com | 9876543210 | shreyass |     3 |
+---------+-----------+-------------------+------------+----------+-------+
3 rows in set (0.003 sec)
```

2. This can be used to select all users who haven't subscribed for annual to remind them to subscribe or to provide them with some offer.
   SELECT * FROM user EXCEPT SELECT * FROM user WHERE subId=1;



```
MariaDB [netflixDb]> SELECT * FROM user EXCEPT SELECT * FROM user WHERE subId=1;
+---------------+----------------+--------------------+------------+---------------+-------+
| userId        | name           | emailId            | phoneNo    | LoginId       | subId |
+---------------+----------------+--------------------+------------+---------------+-------+
| shashankvarma | Shashank Varma | shashank@gmail.com | 9538655010 | shashankvarma |     2 |
| shishirabhat  | Shishira Bhat  | shishira@gmail.com | 9658421370 | shishirabhat  |     2 |
| shreyas       | Shreyas S      | shreyas@gmail.com  | 9876543210 | shreyass      |     3 |
+---------------+----------------+--------------------+------------+---------------+-------+
3 rows in set (0.001 sec)
```

3. This command is used to get common cast between 2 movies to display in the "Also acted in" section.
   SELECT castId FROM movieCasting where movieId=movie1 INTERSECT SELECT castId FROM movieCasting where movieId=movie2;



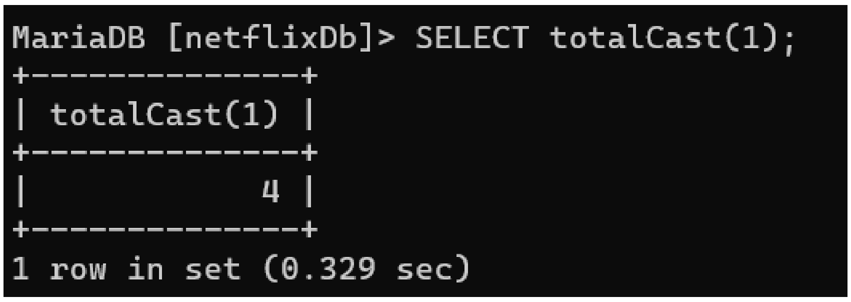
```
MariaDB [netflixDb]> call getCommonCast(2,6);
+--------+
| castId |
+--------+
|      8 |
+--------+
1 row in set (0.098 sec)
```

   We can see that cast with id=8 has acted in both movie 2 and movie 6.

Functions And Procedures

1. This function can be used to get the total number of actors in the movie.

```
DELIMITER $$
CREATE FUNCTION totalCast(movId INT(10))
   RETURNS INT(10)
   DETERMINISTIC
   BEGIN
   DECLARE res INT(10);
   SELECT COUNT(*) INTO res FROM movieCasting GROUP BY movieId HAVING
movieCasting.movieId=movId;
   RETURN res;
   END;$$
DELIMITER ;
SELECT totalCast(1);
```

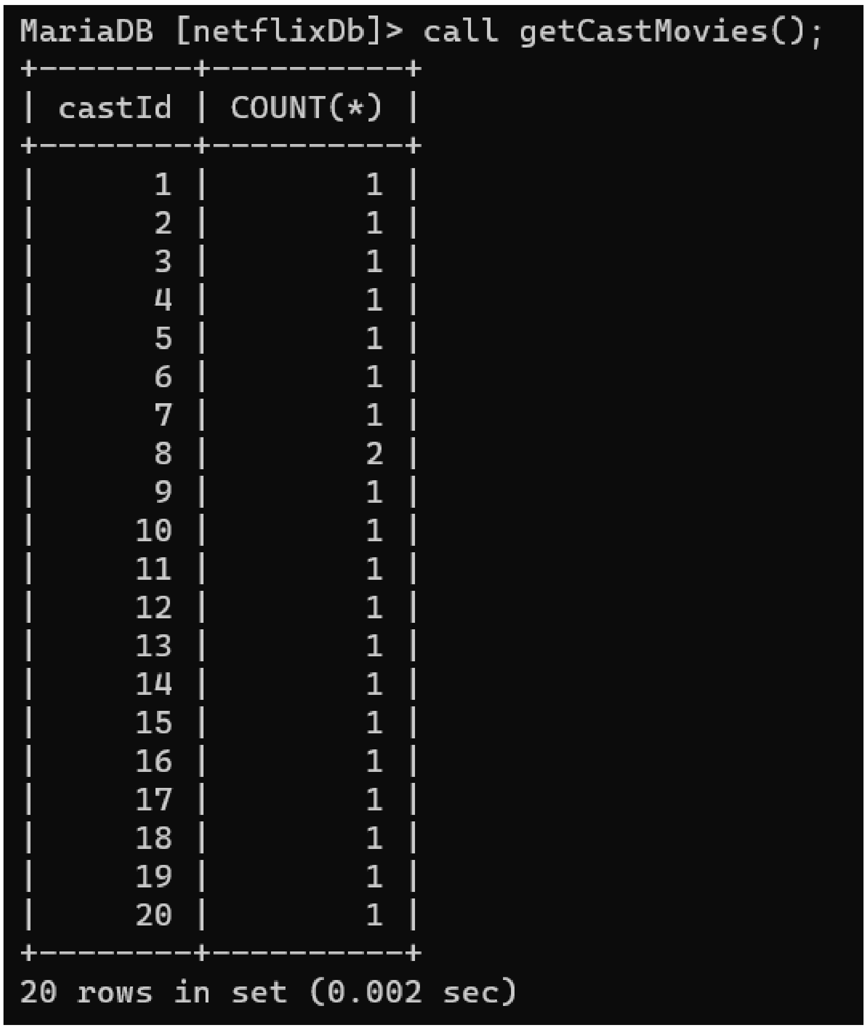


```
MariaDB [netflixDb]> SELECT totalCast(1);
+--------------+
| totalCast(1) |
+--------------+
|            4 |
+--------------+
1 row in set (0.329 sec)
```

We can see that a total of 4 actors are present in the movie with id 1.

2. This procedure can be used to check how many movies each actor has acted in.

```
DROP PROCEDURE getCastMovies;
DELIMITER $$
CREATE PROCEDURE getCastMovies()
BEGIN
SELECT castId,COUNT(*) from movieCasting GROUP BY castId;
END;$$
DELIMITER ;
call getCastMovies();
```



```
MariaDB [netflixDb]> call getCastMovies();
+--------+----------+
| castId | COUNT(*) |
+--------+----------+
|      1 |        1 |
|      2 |        1 |
|      3 |        1 |
|      4 |        1 |
|      5 |        1 |
|      6 |        1 |
|      7 |        1 |
|      8 |        2 |
|      9 |        1 |
|     10 |        1 |
|     11 |        1 |
|     12 |        1 |
|     13 |        1 |
|     14 |        1 |
|     15 |        1 |
|     16 |        1 |
|     17 |        1 |
|     18 |        1 |
|     19 |        1 |
|     20 |        1 |
+--------+----------+
20 rows in set (0.002 sec)
```

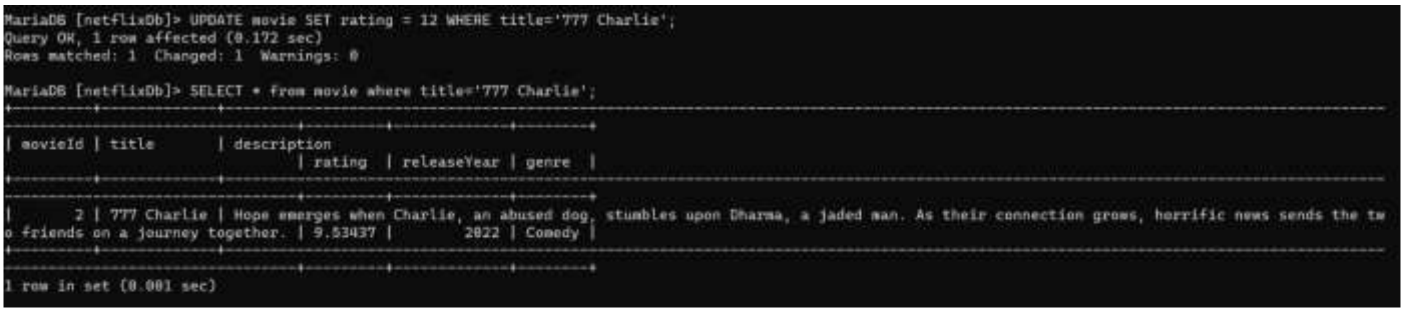
Triggers and Cursors

1. This trigger is used to make sure the rating stays between 0 and 10 while updating. If it is above 10, it made 10 and if it is below zero, it is made zero before updating.

```
DELIMITER $$
CREATE TRIGGER rating_check BEFORE UPDATE ON movie
      FOR EACH ROW
      BEGIN
         IF NEW.rating < 0 THEN
            SET NEW.rating = 0;
         ELSEIF NEW.rating > 10 THEN
            SET NEW.rating = 10;
         END IF;
      END;$$
DELIMITER ;
```

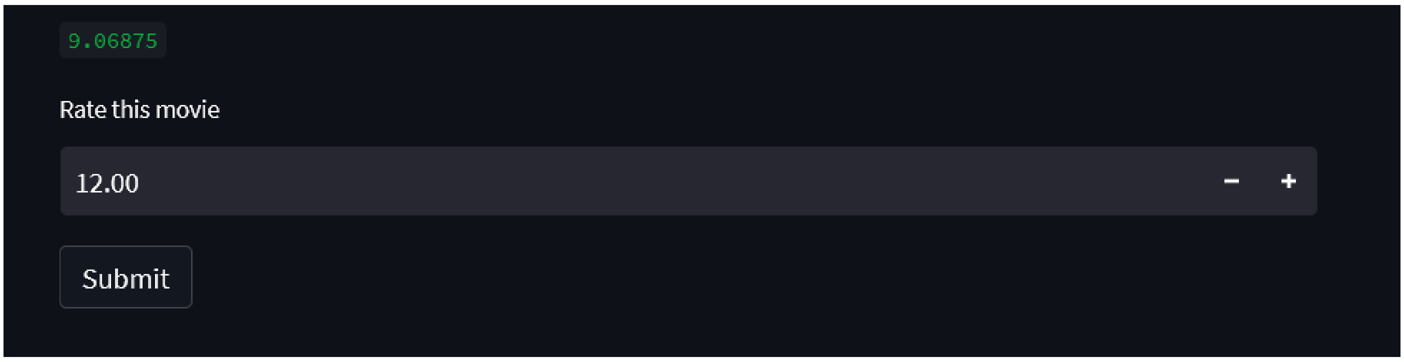
2. This trigger is used to set the rating of a movie to be the average of the old rating and the new rating entered by the user.

```
DELIMITER $$
CREATE TRIGGER update_rating BEFORE UPDATE ON movie
      FOR EACH ROW
      BEGIN
         SET NEW.rating=(OLD.rating+NEW.rating)/2;
      END;$$
DELIMITER ;
```



```
MariaDB [netflixDb]> UPDATE movie SET rating = 12 WHERE title='777 Charlie';
Query OK, 1 row affected (0.172 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [netflixDb]> SELECT * from movie where title='777 Charlie';
+---------+-------------+---------------------------------------------------------------------------------------------------------------------------------
---------------------------------+----------+-------------+--------+
| movieId | title       | description
                                  | rating   | releaseYear | genre  |
+---------+-------------+---------------------------------------------------------------------------------------------------------------------------------
---------------------------------+----------+-------------+--------+
|       2 | 777 Charlie | Hope emerges when Charlie, an abused dog, stumbles upon Dharma, a jaded man. As their connection grows, horrific news sends the tw
o friends on a journey together. | 9.53437 |        2022 | Comedy |
+---------+-------------+---------------------------------------------------------------------------------------------------------------------------------
---------------------------------+----------+-------------+--------+
1 row in set (0.001 sec)
```

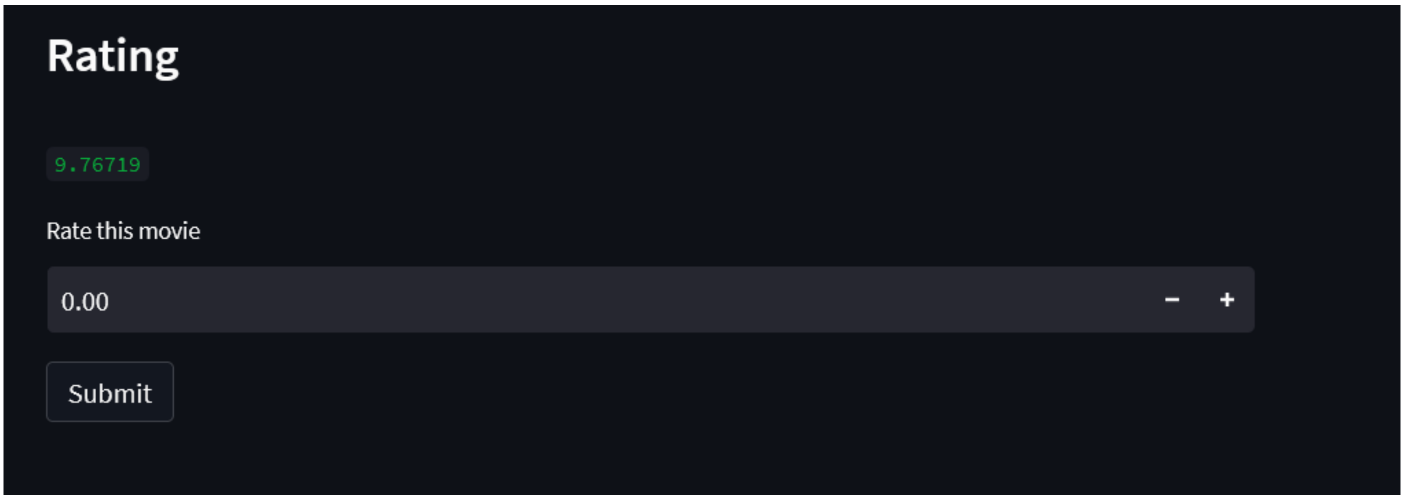
Before update:


9.06875

Rate this movie

12.00

After update:


Rating

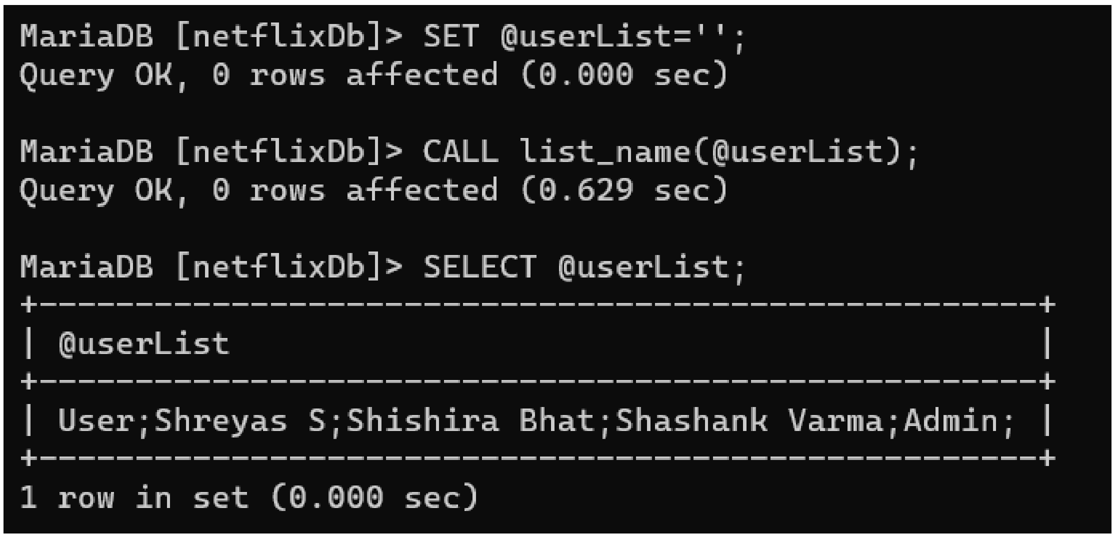
9.76719

Rate this movie

0.00

3. This cursor is used to get all the names of users as list. This can be used by the admin to get the list of users.

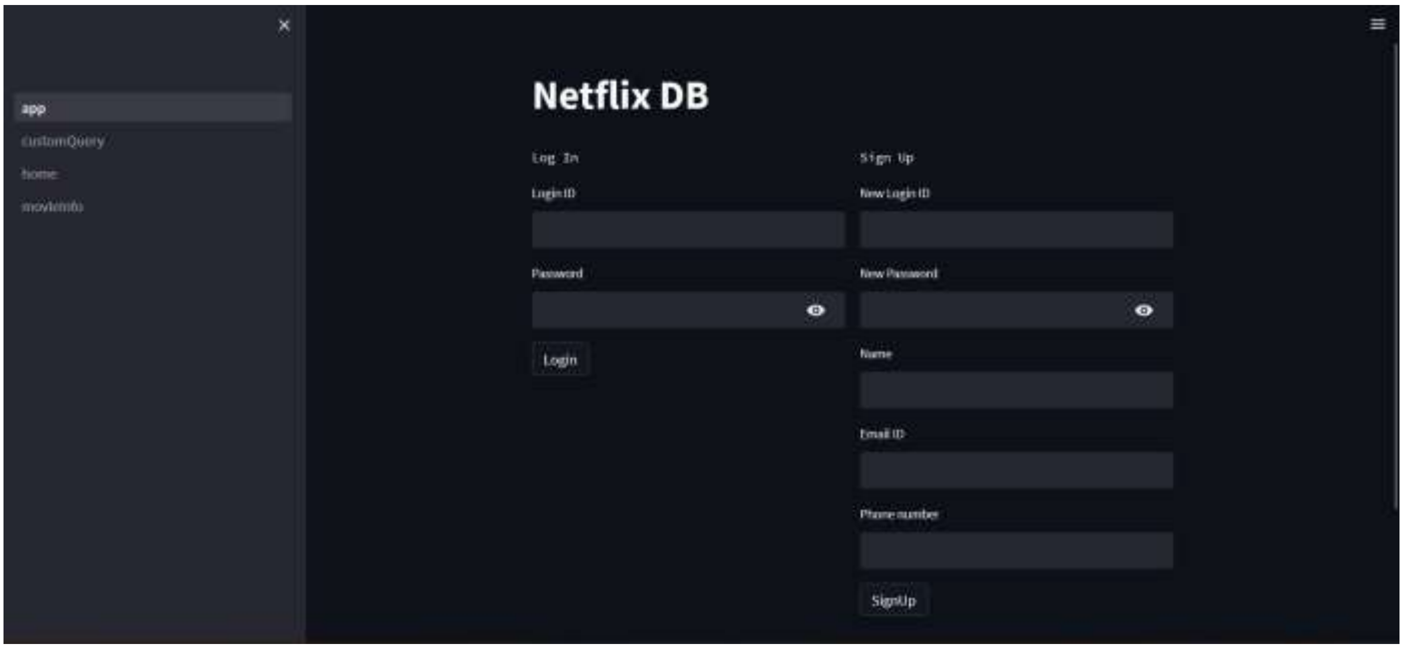
```
DELIMITER $$
CREATE PROCEDURE list_name(INOUT user_list varchar(1000))
BEGIN
DECLARE is_done INTEGER DEFAULT 0;
DECLARE uName varchar(20) DEFAULT "";
DECLARE getNames CURSOR FOR
SELECT name FROM user;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET is_done=1;
OPEN getNames;
get_names: LOOP
FETCH getNames INTO uName;
IF is_done=1 THEN
LEAVE get_names;
END IF;
SET user_list=CONCAT(uName, ";",user_list);
END LOOP get_names;
END;$$
DELIMITER ;
```



```
MariaDB [netflixDb]> SET @userList='';
Query OK, 0 rows affected (0.000 sec)

MariaDB [netflixDb]> CALL list_name(@userList);
Query OK, 0 rows affected (0.629 sec)

MariaDB [netflixDb]> SELECT @userList;
+------------------------------------------------------+
| @userList                                            |
+------------------------------------------------------+
| User;Shreyas S;Shishira Bhat;Shashank Varma;Admin; |
+------------------------------------------------------+
1 row in set (0.000 sec)
```

Developing Frontend

Login page



Netflix DB

Log In

Login ID

Password

Login

Sign Up

New Login ID

New Password

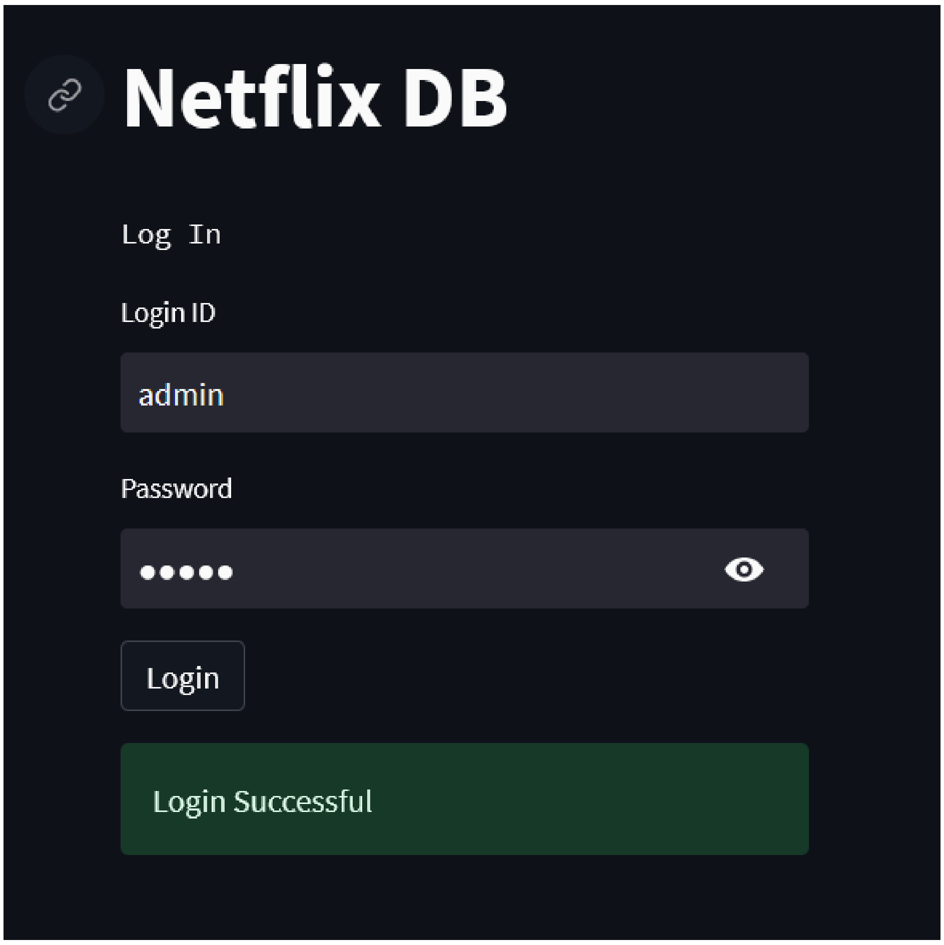
Name

Email ID

Phone number

SignUp

After successful login:



Netflix DB

Log In

Login ID

admin

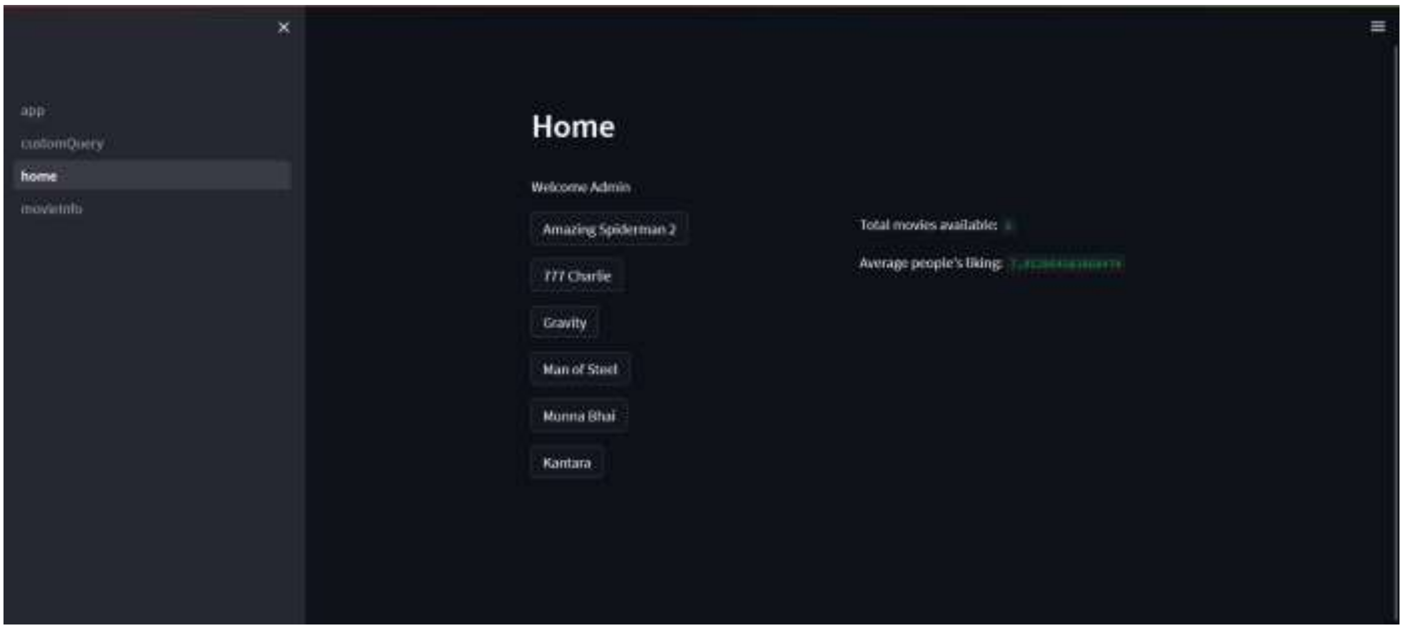
Password

●●●●●

Login

Login Successful

Home page:



Home

Welcome Admin

Amazing Spiderman 2

777 Charlie

Gravity

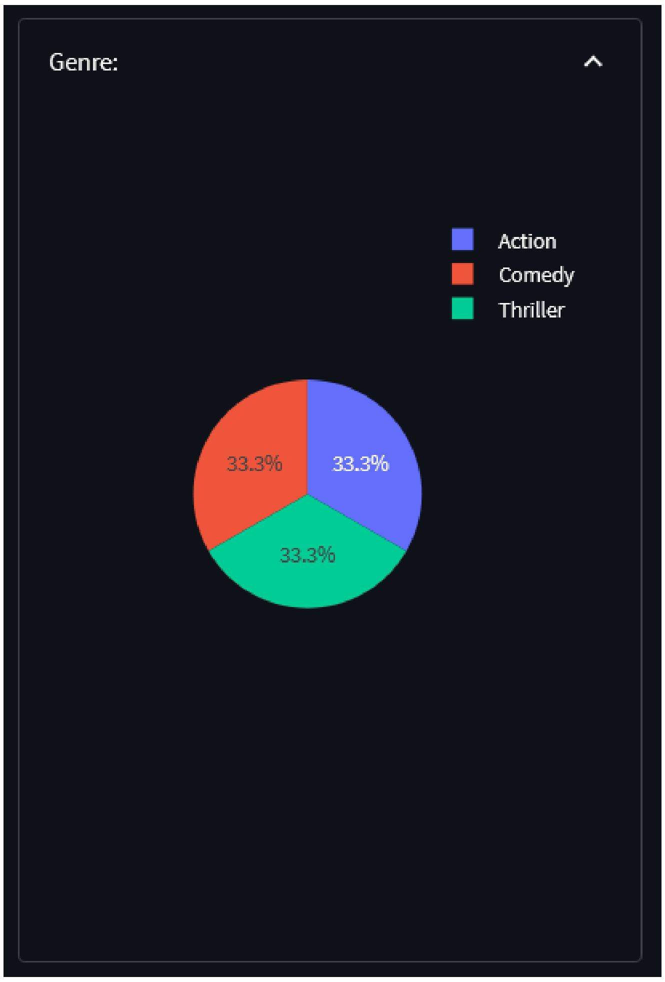
Man of Steel

Munna Bhai

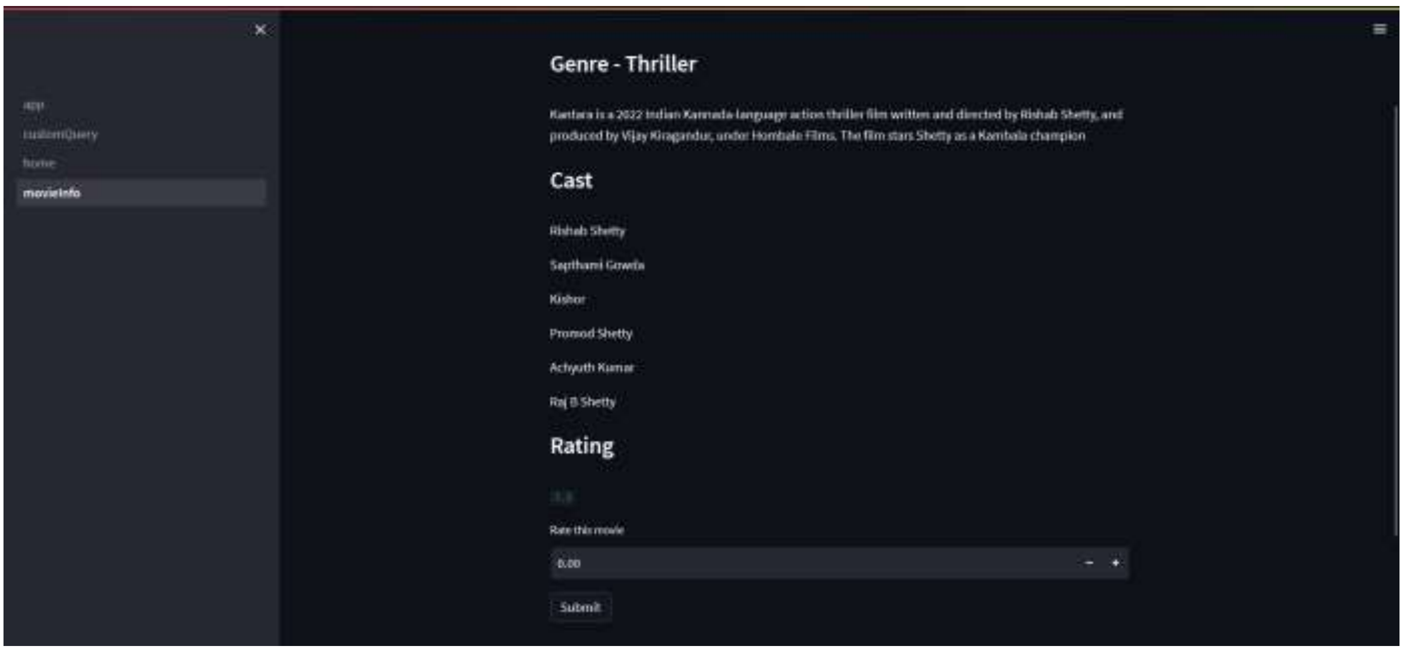
Kantara

Total movies available: [illegible]

Average people's liking: [illegible]



Movie information page:



Genre - Thriller

Kantara is a 2022 Indian Kannada-language action thriller film written and directed by Rishab Shetty, and produced by Vijay Kiragandur, under Hombale Films. The film stars Shetty as a Kambala champion

Cast

Rishab Shetty

Sapthami Gowda

Kishor

Promod Shetty

Achyuth Kumar

Raj B Shetty

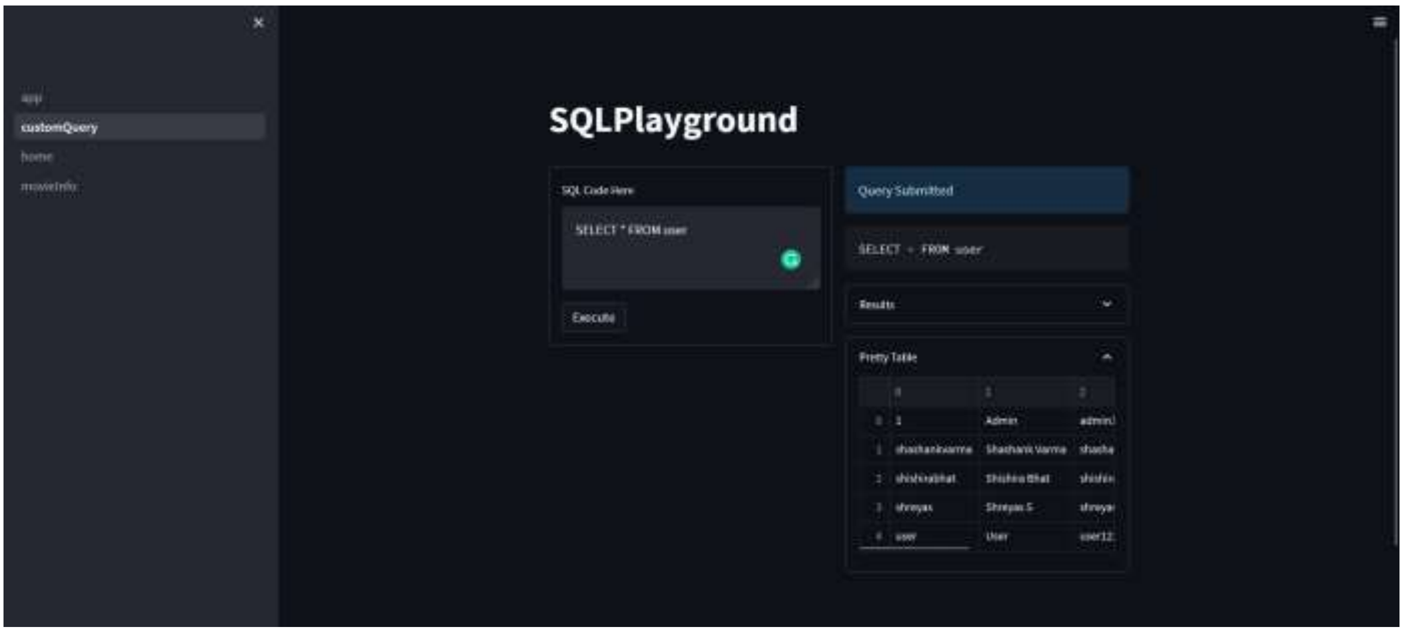
Rating

[illegible]

Rate this movie

0.00

Custom query page:



SQLPlayground

SQL Code Here

```
SELECT * FROM user
```

Query Submitted

```
SELECT * FROM user
```

	0	1	2
0	1	Admin	admin
1	shashankvarma	Shashank Varma	shasha
2	shishirabhat	Shishira Bhat	shishi
3	shreyas	Shreyas S	shreya
4	user	User	user12