CSE4/560 Project Milestone 1

"I have read and understood the course academic integrity policy in the syllabus of the class."

1. Project Details

Project Name: MovieGoods

Project Team Name: EliteWorks

Project Team Details:

NAME	UBID
Harsha Vardhan Sivadanam	hsivadan
Nikhil Kumar Vadigala	nvadigal
Deepana Nagireddy Gari	deepanan

2. Problem Statement

Movie enthusiasts often struggle to find reliable and up-to-date information about movies, such as cast and crew details, release dates, and other details. The aim of this project is to develop a comprehensive and accurate movie database that stores information about movies and their associated details. The database will include data on movie titles, cast and crew members, release dates, genres, ratings, and other relevant information. The project will involve collecting data from multiple sources and using data cleaning and data integration techniques to ensure that the database is accurate and consistent. The database will also need to be scalable to accommodate new movie releases and updates to existing movie information. The project will also involve developing a user-friendly interface for accessing the database and querying for movie information. The objective of this project is to provide movie enthusiasts with a reliable and comprehensive database that can be used to access information quickly and easily about their favorite movies.

Why we need a database instead of an excel file?

Databases are often preferred over Excel files for storing and managing data due to several advantages. First, databases are more scalable and performant than Excel files, which can become slow and unwieldy as data grows. Additionally, databases are designed to maintain data integrity and accuracy, minimizing the risk of errors and inconsistencies. Security is another advantage of databases, with robust authentication and permission features to protect sensitive data.

Whereas in our project, why we used database instead excel file because storing and maintaining a large amount of movie data, including directors, actors, and other details, can become complex and impractical using Excel sheets. To avoid duplication and errors, a database is needed to keep the data sorted, establish relationships between entities, and create a reliable source of information. This will also provide a long-term solution and a single source of truth for different movie database systems according to user needs.

3.Target user

User of the database:

The target users of the MovieGoods database could be movie enthusiasts, filmmakers, movie critics, researchers, and more importantly people who want to explore different movies. This database is useful for every age group. Mainly the target end users would be the ones between 18-35 age. It can be used by anyone who loves entertainment.

Administrator of the database:

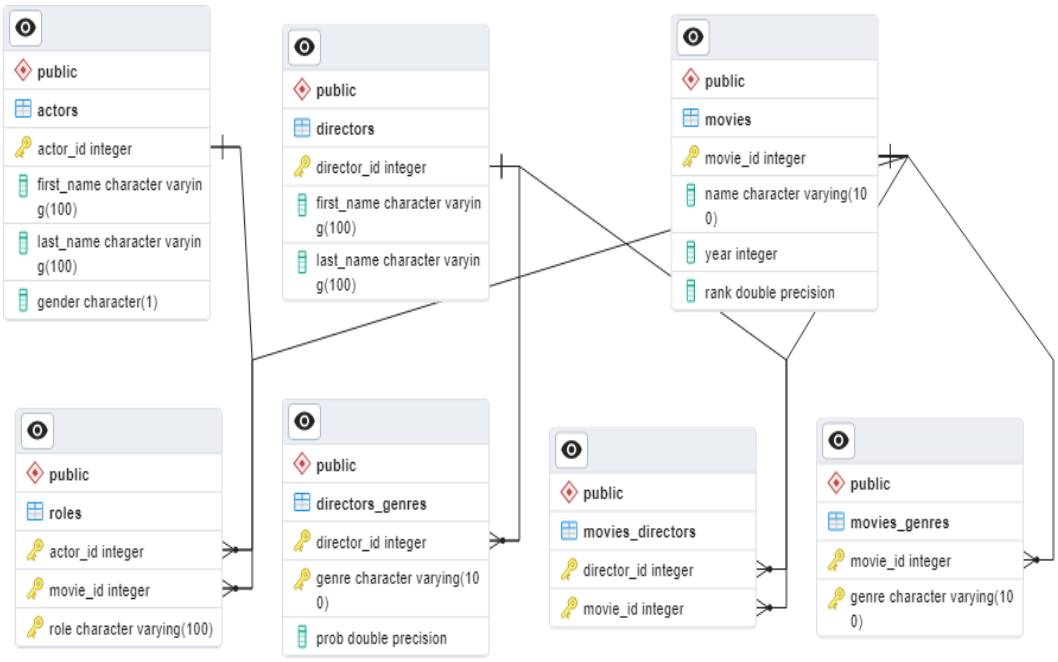
A team consisting of database administrators, developers, and system analysts are responsible for maintaining, updating, and securing the movie information system database.

Real life-life scenario:

A real-life scenario for the movie information system could be a popular movie streaming service such as Netflix or Amazon Prime. These services could use the database to recommend movies to their users based on their viewing history and preferences. The database could also be used by movie review websites to provide comprehensive information about a particular movie, including ratings and reviews from various sources, and relevant information about the movie's cast and crew. The database could also be used by cinema halls to promote and screen movies that are popular among their audience.

4. E/R diagram

Entity relationship diagram is basically a visual representation of entities. It depicts the logical structure of a database by showing the relationships between tables, columns and keys.

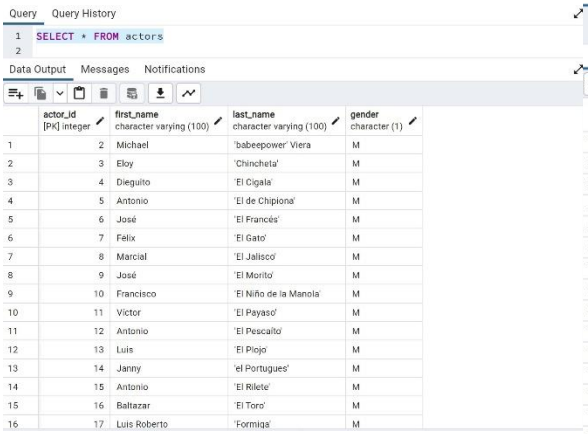


5. Relations and Data

```
create TABLE actors (
    actor_id INT NOT NULL DEFAULT '0',
    first_name VARCHAR(100) NULL DEFAULT NULL,
    last_name VARCHAR(100) NULL DEFAULT NULL ,
    gender CHAR(1) NULL DEFAULT NULL ,
    PRIMARY KEY (actor_id)
);

CREATE TABLE directors (
    director_id INT NOT NULL DEFAULT '0',
    first_name VARCHAR(100) NULL DEFAULT NULL,
    last_name VARCHAR(100) NULL DEFAULT NULL,
    PRIMARY KEY (director_id)
);

CREATE TABLE movies (
    movie_id INT NOT NULL DEFAULT '0',
    name VARCHAR(100) NULL DEFAULT NULL,
    year INT NULL DEFAULT NULL,
    rank FLOAT NULL DEFAULT NULL,
    PRIMARY KEY (movie_id)
);
```

```
CREATE TABLE directors_genres (
    director_id INT NOT NULL,
    genre VARCHAR(100) NOT NULL,
    prob FLOAT NULL DEFAULT NULL,
    PRIMARY KEY (director_id, genre),
    FOREIGN KEY (director_id) REFERENCES directors(director_id)
);

CREATE TABLE movies_directors (
    director_id INT NOT NULL,
    movie_id INT NOT NULL,
    PRIMARY KEY (director_id, movie_id) ,
    FOREIGN KEY (director_id) REFERENCES directors(director_id),
    FOREIGN KEY (movie_id) REFERENCES movies(movie_id)
);

CREATE TABLE movies_genres (
    movie_id INT NOT NULL,
    genre VARCHAR(100) NOT NULL ,
    PRIMARY KEY (movie_id, genre) ,
    FOREIGN KEY (movie_id) REFERENCES movies(movie_id)
);
```

```sql
CREATE TABLE roles (
    actor_id INT NOT NULL,
    movie_id INT NOT NULL,
    role VARCHAR(100) NOT NULL ,
    PRIMARY KEY (actor_id, movie_id, role) ,
    FOREIGN KEY (actor_id) REFERENCES actors(actor_id),
    FOREIGN KEY (movie_id) REFERENCES movies(movie_id)
);
```

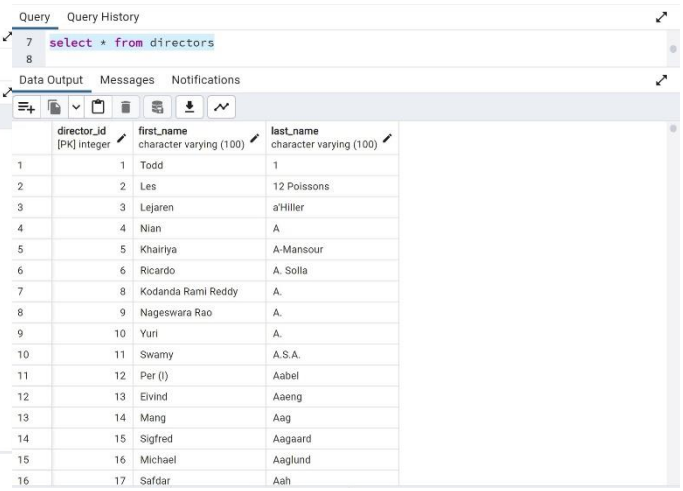


Query Query History

```sql
SELECT * FROM actors
```

Data Output Messages Notifications

	actor_id [PK] integer	first_name character varying (100)	last_name character varying (100)	gender character (1)
1	2	Michael	'babeepower' Viera	M
2	3	Eloy	'Chincheta'	M
3	4	Dieguito	'El Cigala'	M
4	5	Antonio	'El de Chipiona'	M
5	6	José	'El Francés'	M
6	7	Félix	'El Gato'	M
7	8	Marcial	'El Jalisco'	M
8	9	José	'El Morito'	M
9	10	Francisco	'El Niño de la Manola'	M
10	11	Víctor	'El Payaso'	M
11	12	Antonio	'El Pescaíto'	M
12	13	Luis	'El Piojo'	M
13	14	Janny	'el Portugues'	M
14	15	Antonio	'El Rilete'	M
15	16	Baltazar	'El Toro'	M
16	17	Luis Roberto	'Formiga'	M

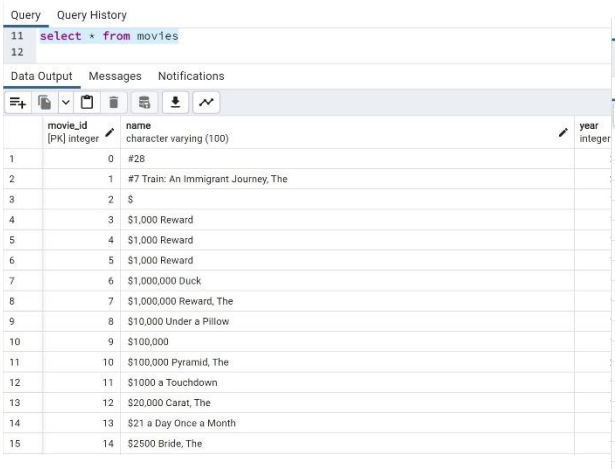


Query Query History

```sql
select * from directors
```

Data Output Messages Notifications

	director_id [PK] integer	first_name character varying (100)	last_name character varying (100)
1	1	Todd	1
2	2	Les	12 Poissons
3	3	Lejaren	a'Hiller
4	4	Nian	A
5	5	Khairiya	A-Mansour
6	6	Ricardo	A. Solla
7	8	Kodanda Rami Reddy	A.
8	9	Nageswara Rao	A.
9	10	Yuri	A.
10	11	Swamy	A.S.A.
11	12	Per (I)	Aabel
12	13	Eivind	Aaeng
13	14	Mang	Aag
14	15	Sigfred	Aagaard
15	16	Michael	Aaglund
16	17	Safdar	Aah

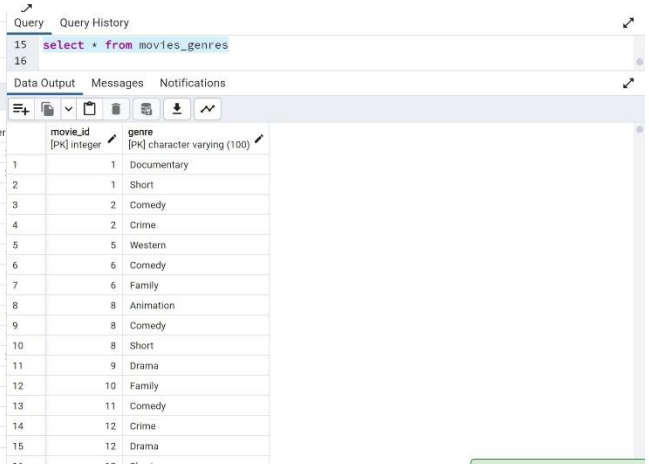


Query Query History

```sql
select * from movies
```

Data Output Messages Notifications

	movie_id [PK] integer	name character varying (100)	year integer
1	0	#28	
2	1	#7 Train: An Immigrant Journey, The	
3	2	$	
4	3	$1,000 Reward	
5	4	$1,000 Reward	
6	5	$1,000 Reward	
7	6	$1,000,000 Duck	
8	7	$1,000,000 Reward, The	
9	8	$10,000 Under a Pillow	
10	9	$100,000	
11	10	$100,000 Pyramid, The	
12	11	$1000 a Touchdown	
13	12	$20,000 Carat, The	
14	13	$21 a Day Once a Month	
15	14	$2500 Bride, The	

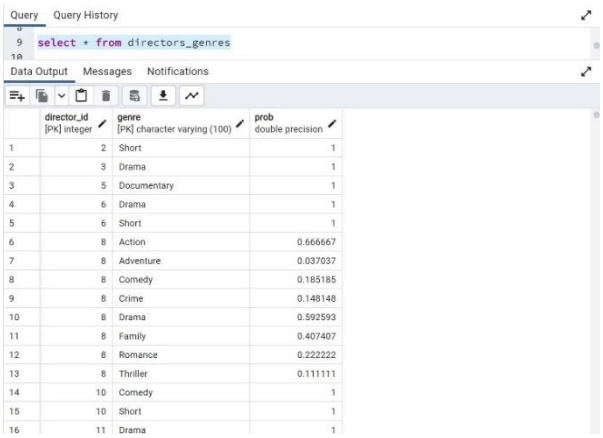


Query Query History

```sql
select * from movies_genres
```

Data Output Messages Notifications

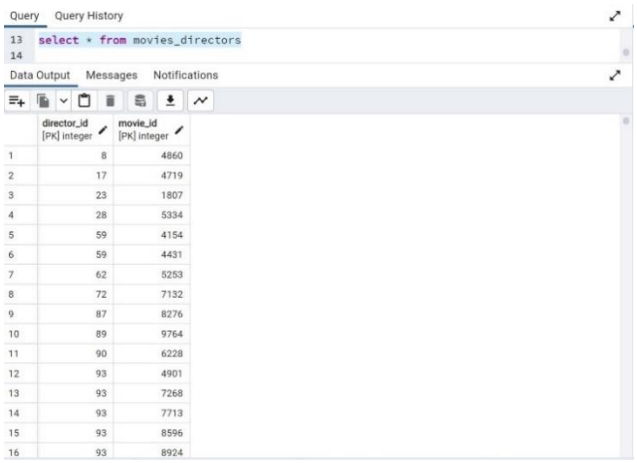
	movie_id [PK] integer	genre [PK] character varying (100)
1	1	Documentary
2	1	Short
3	2	Comedy
4	2	Crime
5	5	Western
6	6	Comedy
7	6	Family
8	8	Animation
9	8	Comedy
10	8	Short
11	9	Drama
12	10	Family
13	11	Comedy
14	12	Crime
15	12	Drama



```
select * from directors_genres
```

Data Output | Messages | Notifications

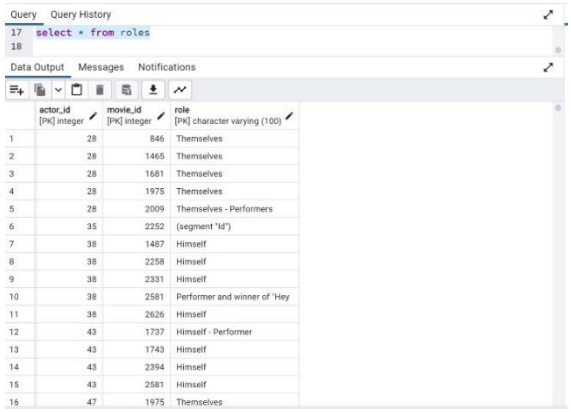
	director_id [PK] integer	genre [PK] character varying (100)	prob double precision
1	2	Short	1
2	3	Drama	1
3	5	Documentary	1
4	6	Drama	1
5	6	Short	1
6	8	Action	0.666667
7	8	Adventure	0.037037
8	8	Comedy	0.185185
9	8	Crime	0.148148
10	8	Drama	0.592593
11	8	Family	0.407407
12	8	Romance	0.222222
13	8	Thriller	0.111111
14	10	Comedy	1
15	10	Short	1
16	11	Drama	1



```
select * from movies_directors
```

Data Output | Messages | Notifications

	director_id [PK] integer	movie_id [PK] integer
1	8	4860
2	17	4719
3	23	1807
4	28	5334
5	59	4154
6	59	4431
7	62	5253
8	72	7132
9	87	8276
10	89	9764
11	90	6228
12	93	4901
13	93	7268
14	93	7713
15	93	8596
16	93	8924



```
select * from roles
```

Data Output | Messages | Notifications

	actor_id [PK] integer	movie_id [PK] integer	role [PK] character varying (100)
1	28	846	Themselves
2	28	1465	Themselves
3	28	1681	Themselves
4	28	1975	Themselves
5	28	2009	Themselves - Performers
6	35	2252	(segment "Id")
7	38	1487	Himself
8	38	2258	Himself
9	38	2331	Himself
10	38	2581	Performer and winner of "Hey
11	38	2626	Himself
12	43	1737	Himself - Performer
13	43	1743	Himself
14	43	2394	Himself
15	43	2581	Himself
16	47	1975	Themselves

A relationship between two database tables presupposes that one of them has a foreign key that references the primary key of another table.

- To form a relationship between movies_genre and movies, we have movie_id, which forms a link between.
- In director_genres table, we have director_id, which create the relationship between it and directors table.
- Coming to movies_directors, it can be linket to two tables, which are director_id from directors table & movie_id from movies table.
- Roles table can have two relations, one can be with movies table which has movie_id, and, secondly with actors which has actor_id.

6. Constraints

Primary Keys:

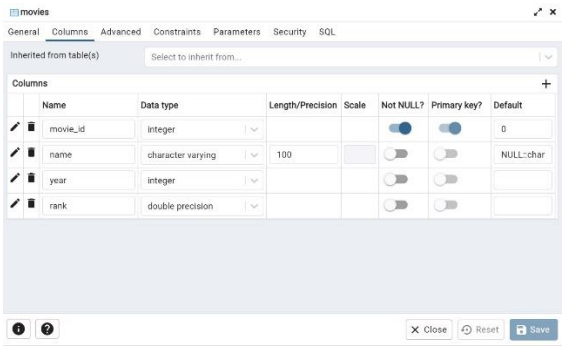
actors table: actor_id
directors table: director_id
movies table: movie_id
directors_genres: genre
movies_genres: genre
roles: role

Foreign Keys:

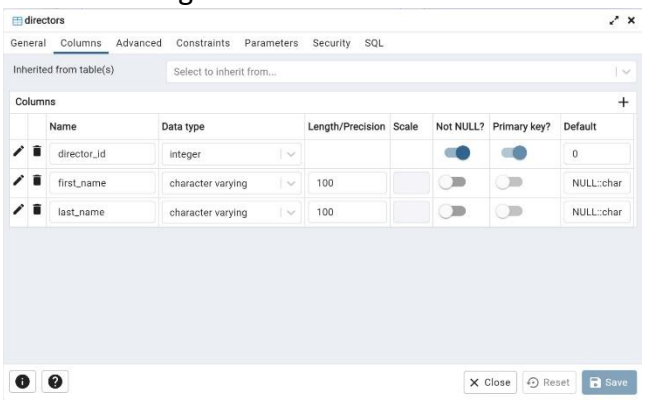
directors_genres: director_id
movies_directors: director_id, movie_id
movies_genres: movie_id
roles: actor_id, movie_id

7. Attributes

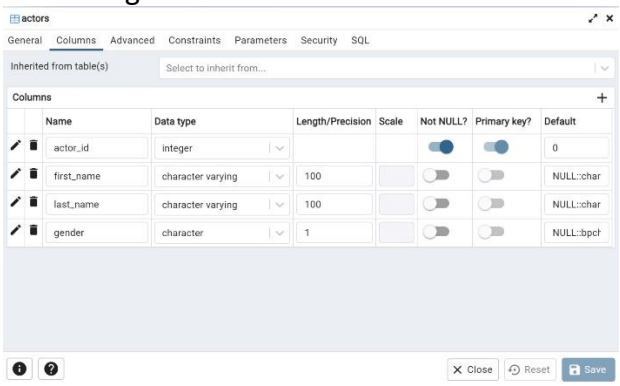
- It gives the information about the name of the movie, year in which it was released, and the rank it holds.



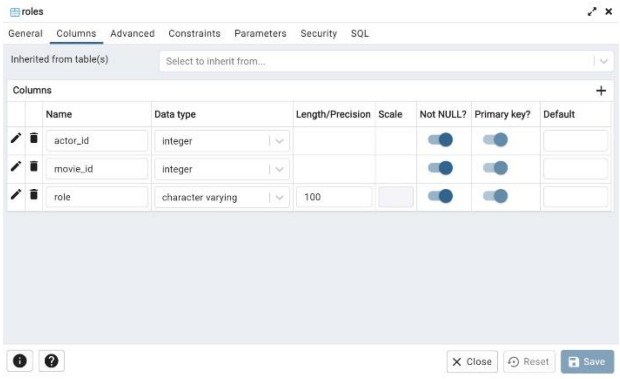
- It gives the information about the full name of the director who made the film you are looking for.



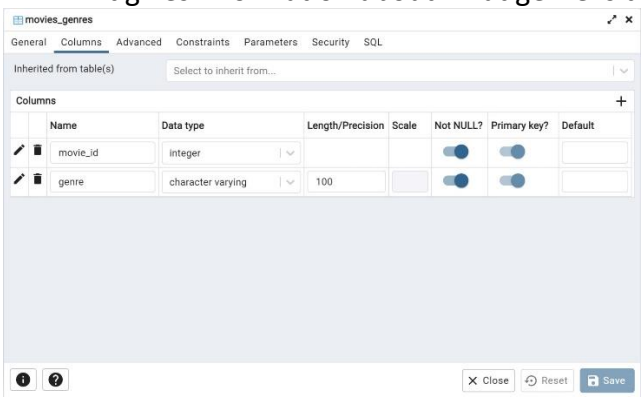
- It gives the information about actors who acted in the movie.



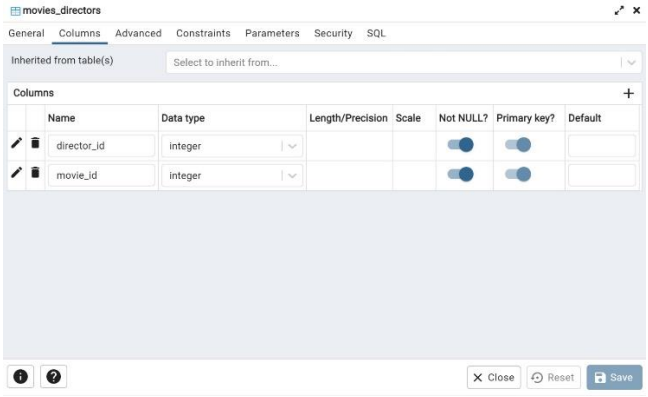
- It talks about which person is playing what role as in cast of movie.



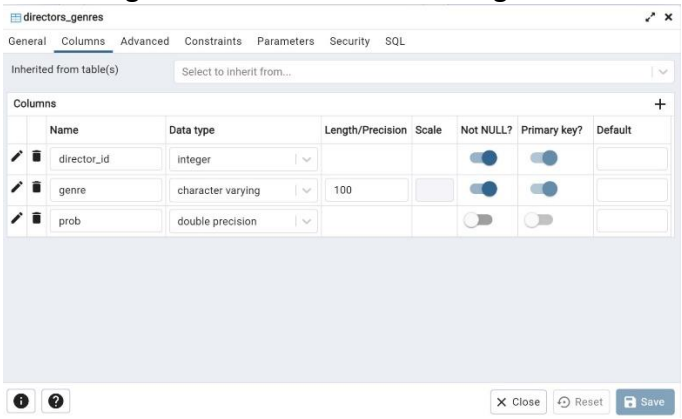
- It gives information about what genre is the movie you are looking for.



- It gives details about the movie and director.

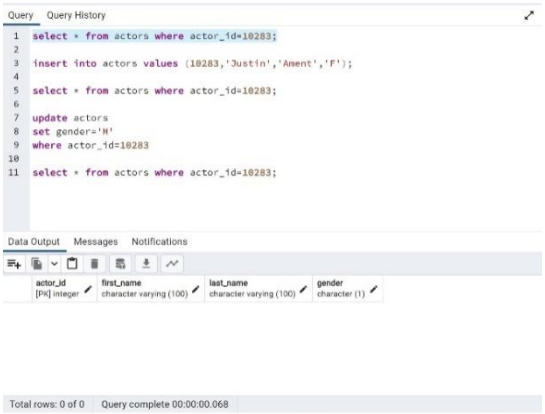


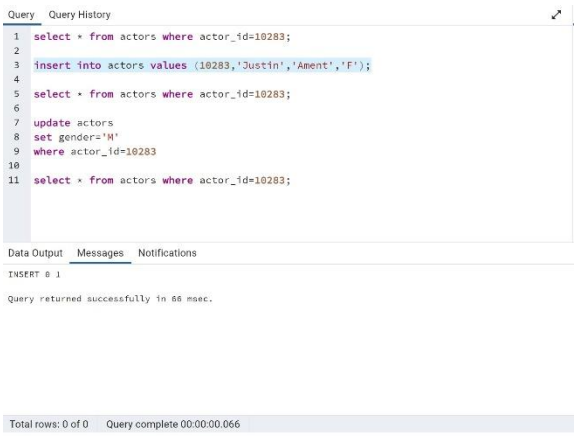
- It gives the information about genre and rating of the director's movies.

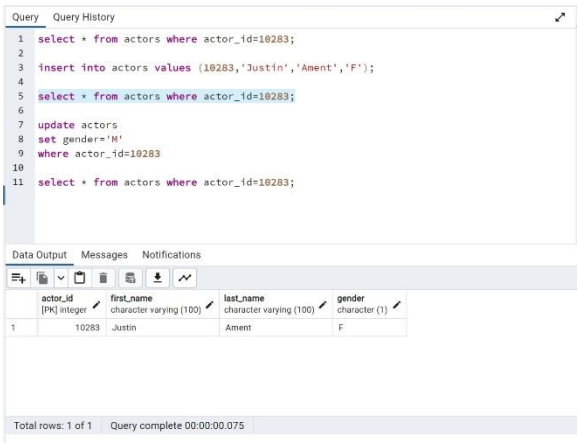


8. Queries

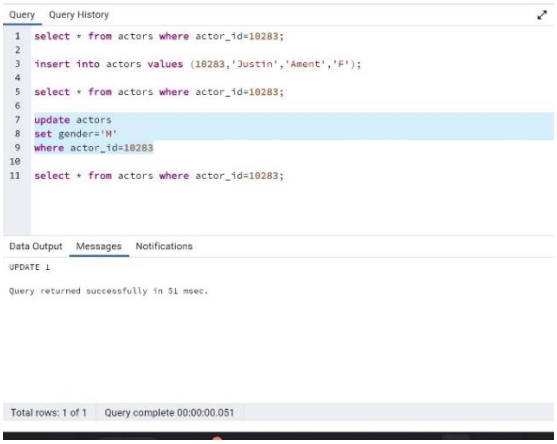
Select & Insert:

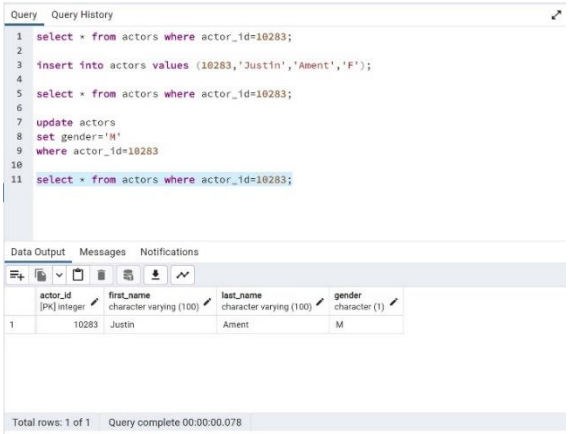




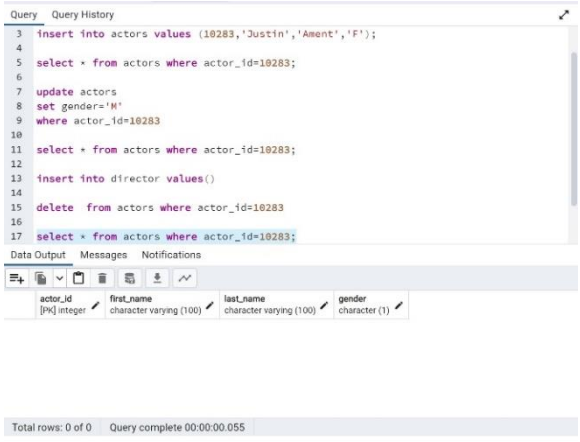


Update:

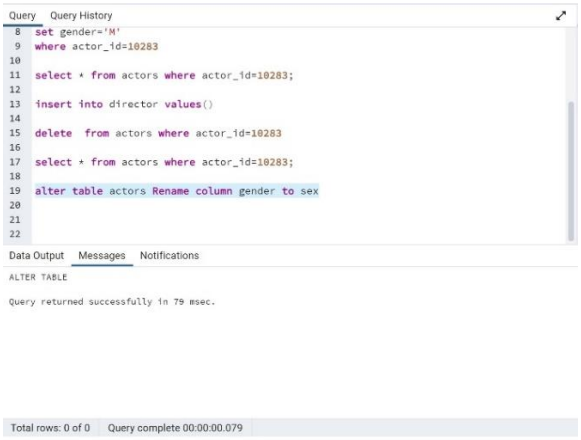


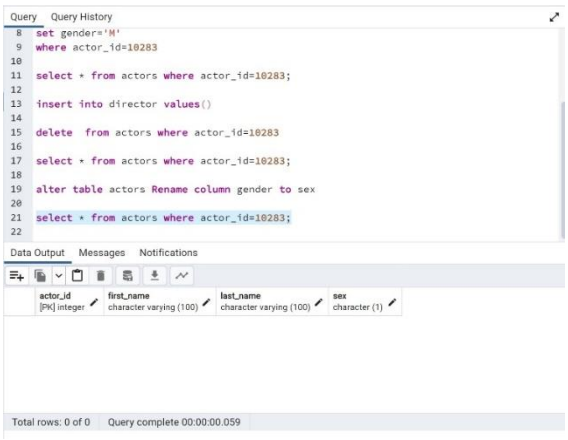


Delete:



Alter:





Data Load: