Name: Manaois, Ivan Bryan R. Date: 11/29/25
Year/Course: 2nd year BSCS C204 Subject: 7OOP

Problem:

Finals Lab Task 5. CLI using Mysql and Python

1. Make sure you have installed the following pre-requisites before proceeding:

a. Mysql-connector

b. Mysql-connector-python

c. Xampp is running along with Apache and Mysql in the background

2. Create the following database in Mysql;

a. Database name: moviesDB with the ff: fields:

movie_id int(10) Primary Key

title varchar(50) NOT NULL

main_actor varchar(50) NOT NULL

director varchar(50) NOT NULL

genre varchar(25) NOT NULL

gross_sales float

ratings (G, PG, R13, R16,X) varchar(5)

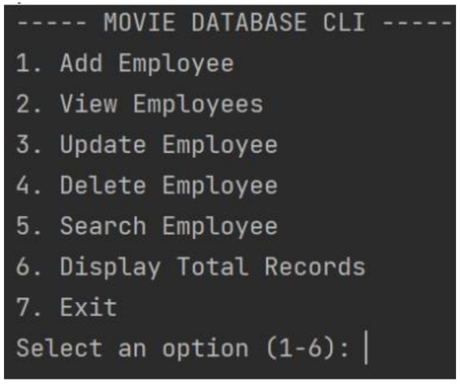
b. Insert at_least 5 records

c. Create a user named test_user and assign a password and give it an admin access by checking necessary SQL functions

3. Guided by the Demo code attached in this task. test_DemoDB.py

4. Kindly continue working on the code that will allow the user to navigate through the Database and perform simple CRUD operations. Follow the following CLI Menu

Options:



```
----- MOVIE DATABASE CLI -----
1. Add Employee
2. View Employees
3. Update Employee
4. Delete Employee
5. Search Employee
6. Display Total Records
7. Exit
Select an option (1-6): |
```

5. The user should be able perform the ff: in your program.

MOVIE DATABASE CRUD APP

1- Add New Record

2- View all records,

3- Update a Record and show the updates,

4- Delete a record

5- Search A Record

6- Display Total Numbers of Movies stored in the database

7- Exit

6. For additional challenge, Task – You are to add a SEARCH option in the MENU that will allow the user to search by Name or emp_id, then display the information about the record being search. You may use Like statement and fetchOne method in my SQL to do this,

7. You are also going to add a method that will display the the total number of records in your database – You may use SQL count statement for this.

8. What to submit:

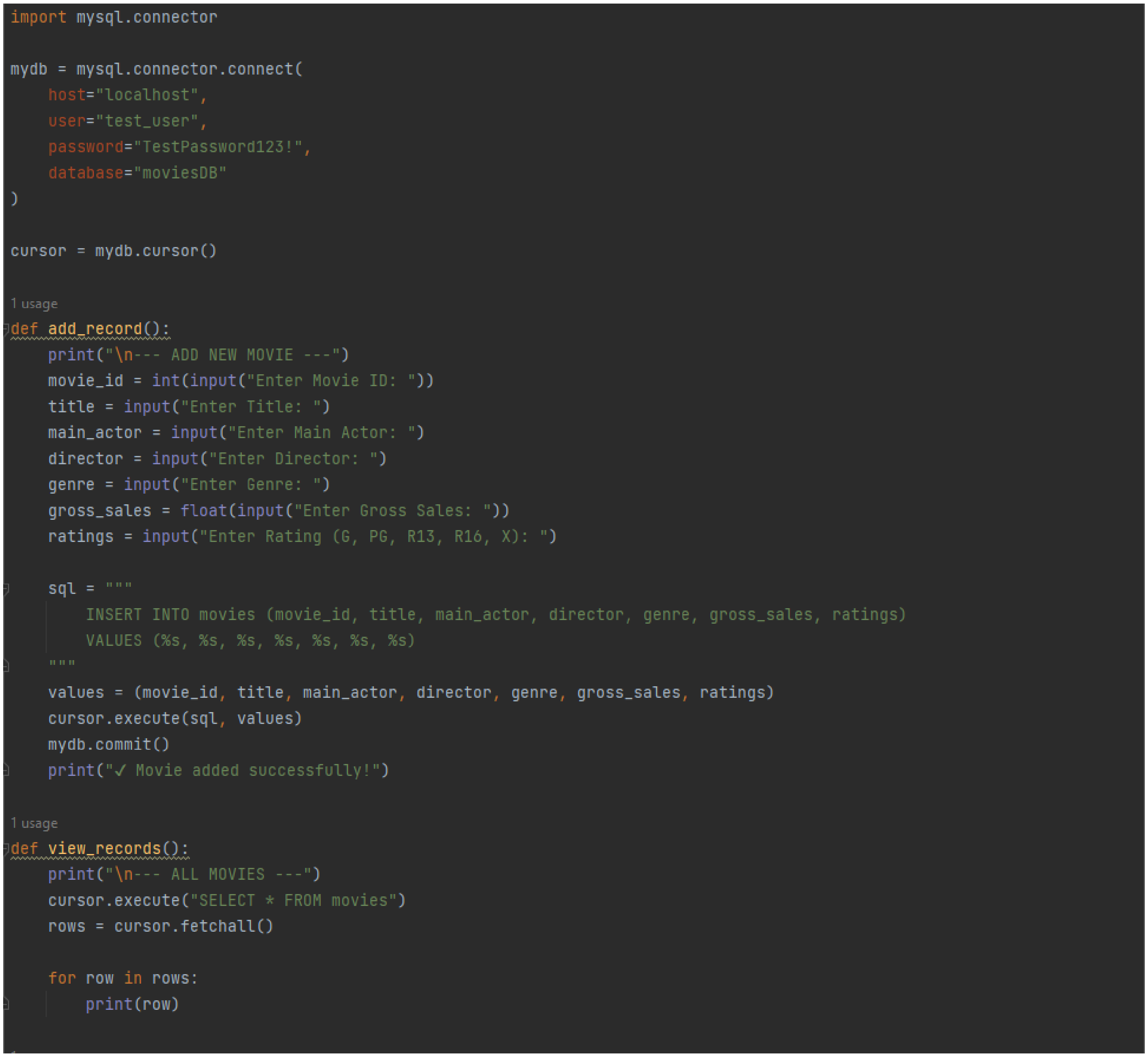
a. UI Menu

b. Sample Output

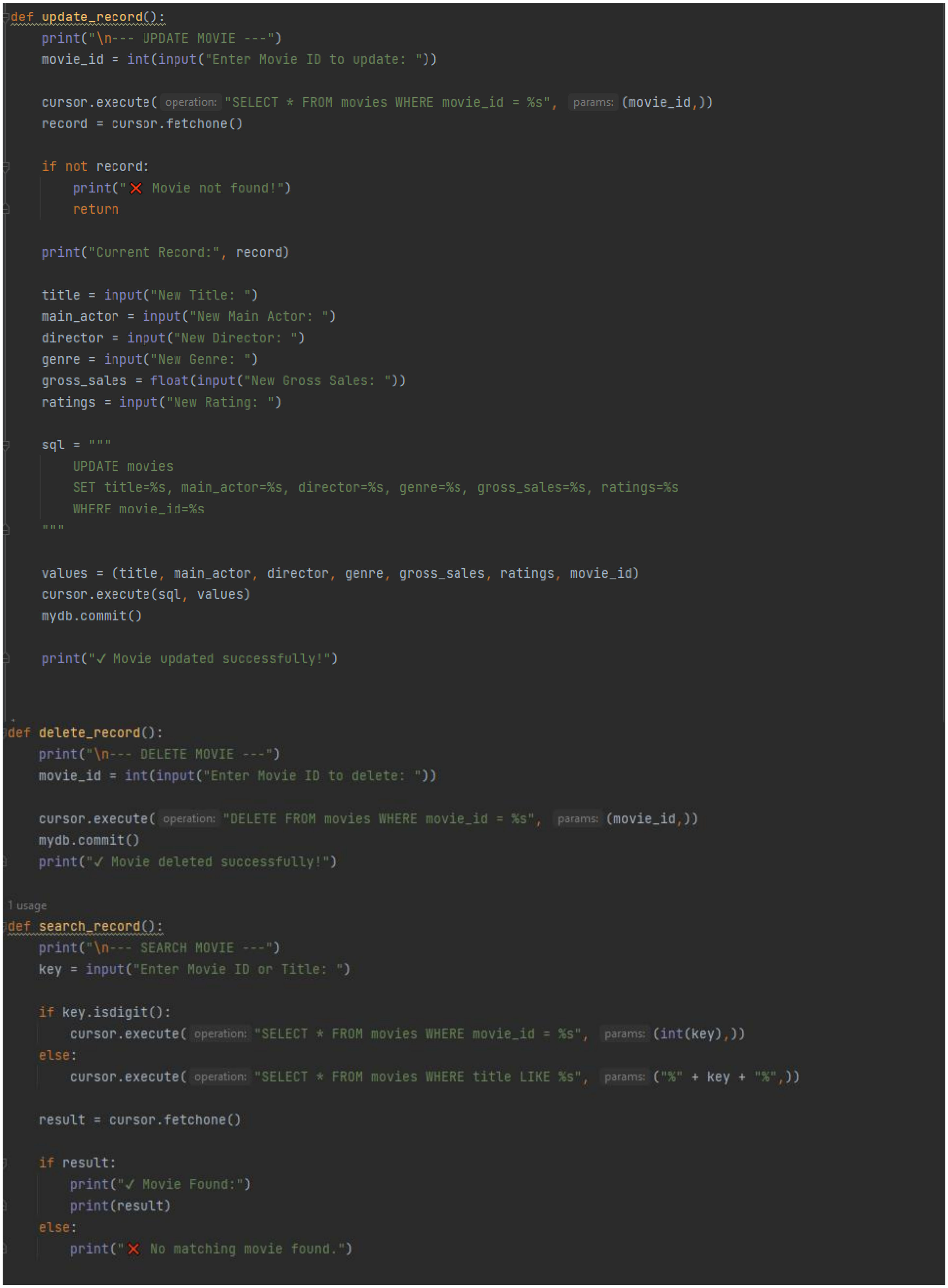
c. Source Code

d. Exported sql file

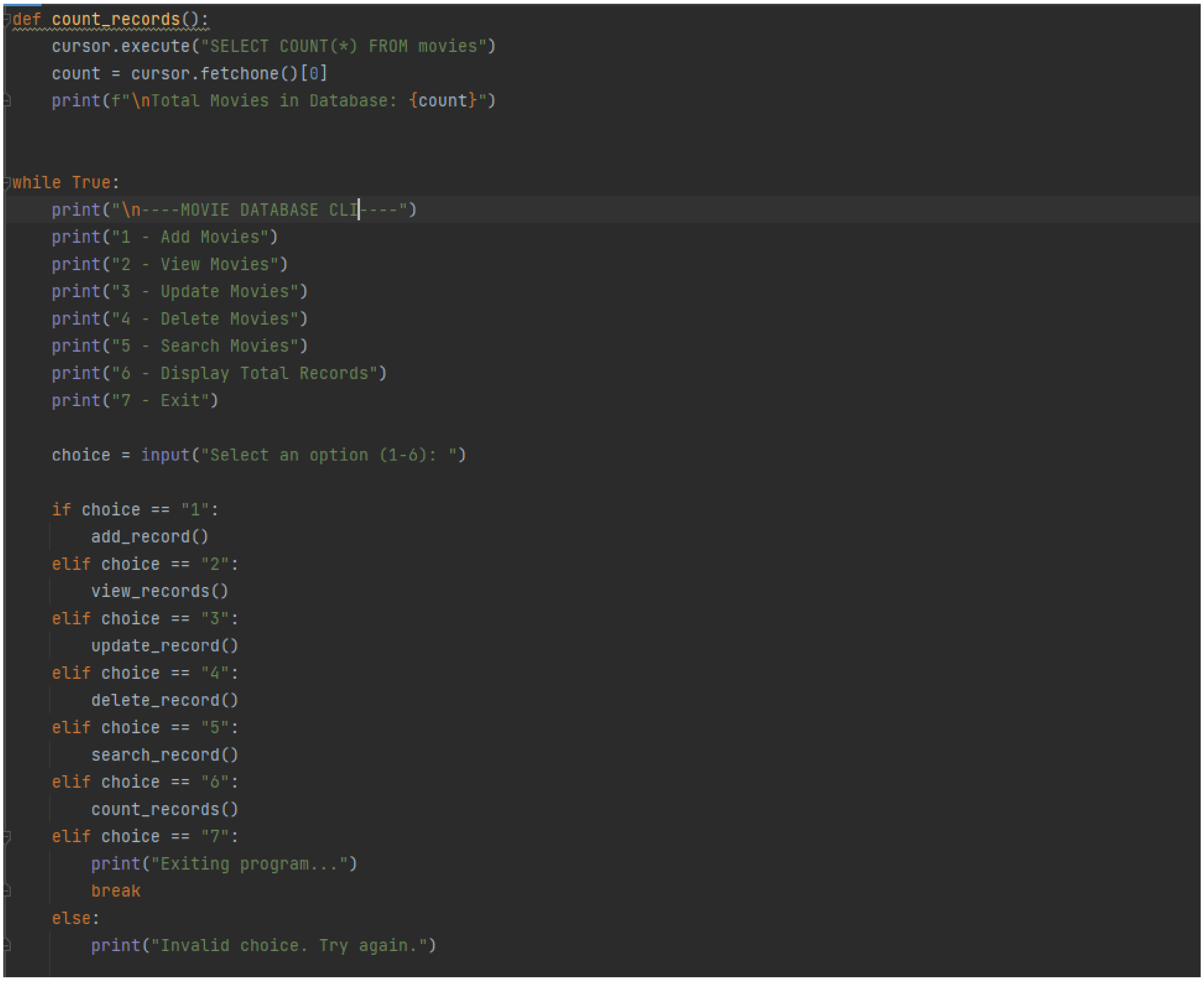
Code:



```python
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="test_user",
    password="TestPassword123!",
    database="moviesDB"
)

cursor = mydb.cursor()

1 usage
def add_record():
    print("\n--- ADD NEW MOVIE ---")
    movie_id = int(input("Enter Movie ID: "))
    title = input("Enter Title: ")
    main_actor = input("Enter Main Actor: ")
    director = input("Enter Director: ")
    genre = input("Enter Genre: ")
    gross_sales = float(input("Enter Gross Sales: "))
    ratings = input("Enter Rating (G, PG, R13, R16, X): ")

    sql = """
        INSERT INTO movies (movie_id, title, main_actor, director, genre, gross_sales, ratings)
        VALUES (%s, %s, %s, %s, %s, %s, %s)
    """
    values = (movie_id, title, main_actor, director, genre, gross_sales, ratings)
    cursor.execute(sql, values)
    mydb.commit()
    print("✓ Movie added successfully!")

1 usage
def view_records():
    print("\n--- ALL MOVIES ---")
    cursor.execute("SELECT * FROM movies")
    rows = cursor.fetchall()

    for row in rows:
        print(row)
```

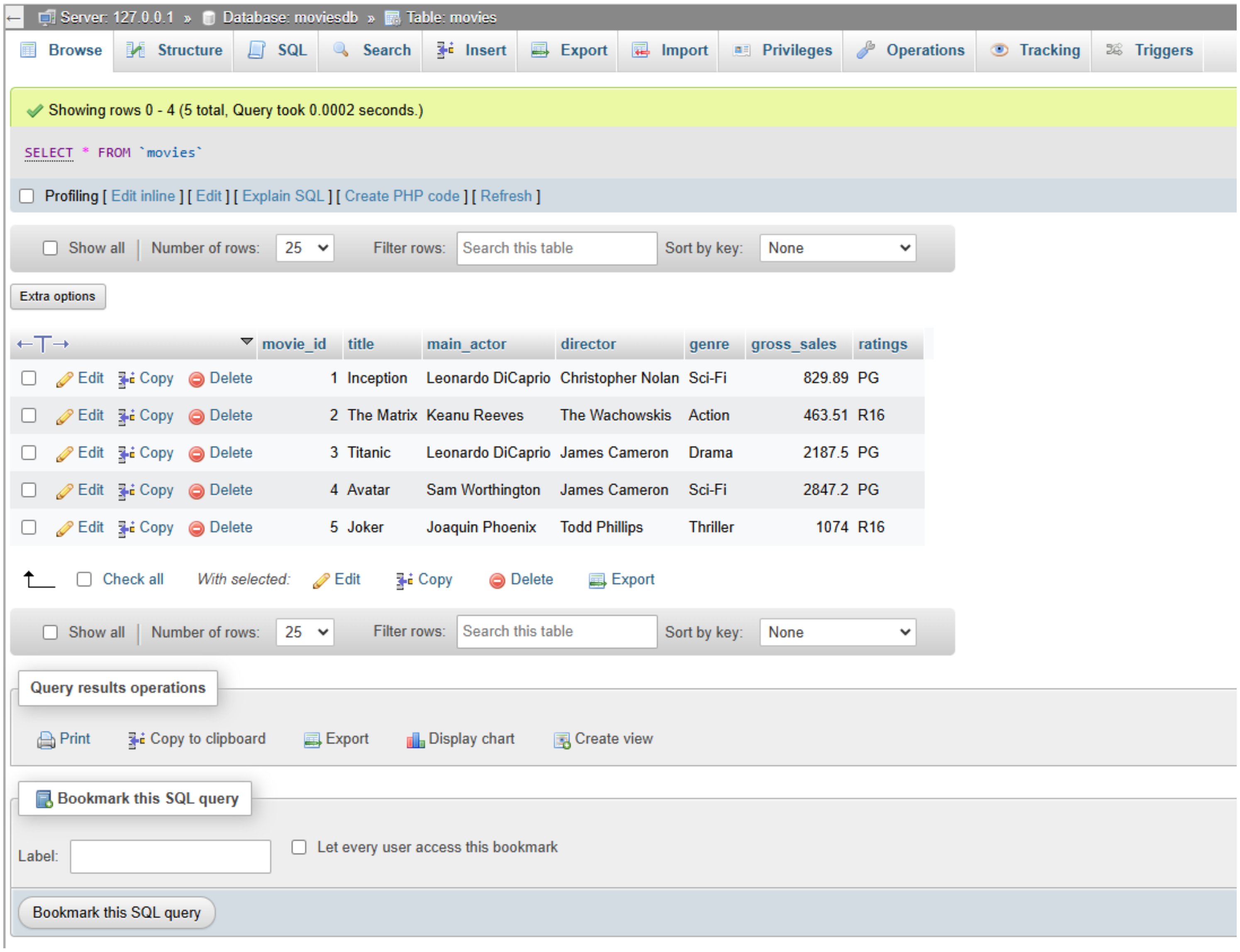


```python
def update_record():
    print("\n--- UPDATE MOVIE ---")
    movie_id = int(input("Enter Movie ID to update: "))

    cursor.execute( operation: "SELECT * FROM movies WHERE movie_id = %s",  params: (movie_id,))
    record = cursor.fetchone()

    if not record:
        print("❌ Movie not found!")
        return

    print("Current Record:", record)

    title = input("New Title: ")
    main_actor = input("New Main Actor: ")
    director = input("New Director: ")
    genre = input("New Genre: ")
    gross_sales = float(input("New Gross Sales: "))
    ratings = input("New Rating: ")

    sql = """
        UPDATE movies
        SET title=%s, main_actor=%s, director=%s, genre=%s, gross_sales=%s, ratings=%s
        WHERE movie_id=%s
    """

    values = (title, main_actor, director, genre, gross_sales, ratings, movie_id)
    cursor.execute(sql, values)
    mydb.commit()

    print("✓ Movie updated successfully!")


def delete_record():
    print("\n--- DELETE MOVIE ---")
    movie_id = int(input("Enter Movie ID to delete: "))

    cursor.execute( operation: "DELETE FROM movies WHERE movie_id = %s",  params: (movie_id,))
    mydb.commit()
    print("✓ Movie deleted successfully!")

1 usage
def search_record():
    print("\n--- SEARCH MOVIE ---")
    key = input("Enter Movie ID or Title: ")

    if key.isdigit():
        cursor.execute( operation: "SELECT * FROM movies WHERE movie_id = %s",  params: (int(key),))
    else:
        cursor.execute( operation: "SELECT * FROM movies WHERE title LIKE %s",  params: ("%" + key + "%",))

    result = cursor.fetchone()

    if result:
        print("✓ Movie Found:")
        print(result)
    else:
        print("❌ No matching movie found.")
```



```
def count_records():
    cursor.execute("SELECT COUNT(*) FROM movies")
    count = cursor.fetchone()[0]
    print(f"\nTotal Movies in Database: {count}")


while True:
    print("\n----MOVIE DATABASE CLI----")
    print("1 - Add Movies")
    print("2 - View Movies")
    print("3 - Update Movies")
    print("4 - Delete Movies")
    print("5 - Search Movies")
    print("6 - Display Total Records")
    print("7 - Exit")

    choice = input("Select an option (1-6): ")

    if choice == "1":
        add_record()
    elif choice == "2":
        view_records()
    elif choice == "3":
        update_record()
    elif choice == "4":
        delete_record()
    elif choice == "5":
        search_record()
    elif choice == "6":
        count_records()
    elif choice == "7":
        print("Exiting program...")
        break
    else:
        print("Invalid choice. Try again.")
```

Database: MoviesDB



Server: 127.0.0.1 » Database: moviesdb » Table: movies

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Tracking | Triggers

Showing rows 0 - 4 (5 total, Query took 0.0002 seconds.)

```
SELECT * FROM `movies`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

movie_id	title	main_actor	director	genre	gross_sales	ratings
1	Inception	Leonardo DiCaprio	Christopher Nolan	Sci-Fi	829.89	PG
2	The Matrix	Keanu Reeves	The Wachowskis	Action	463.51	R16
3	Titanic	Leonardo DiCaprio	James Cameron	Drama	2187.5	PG
4	Avatar	Sam Worthington	James Cameron	Sci-Fi	2847.2	PG
5	Joker	Joaquin Phoenix	Todd Phillips	Thriller	1074	R16

Check all | With selected: Edit | Copy | Delete | Export

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Query results operations

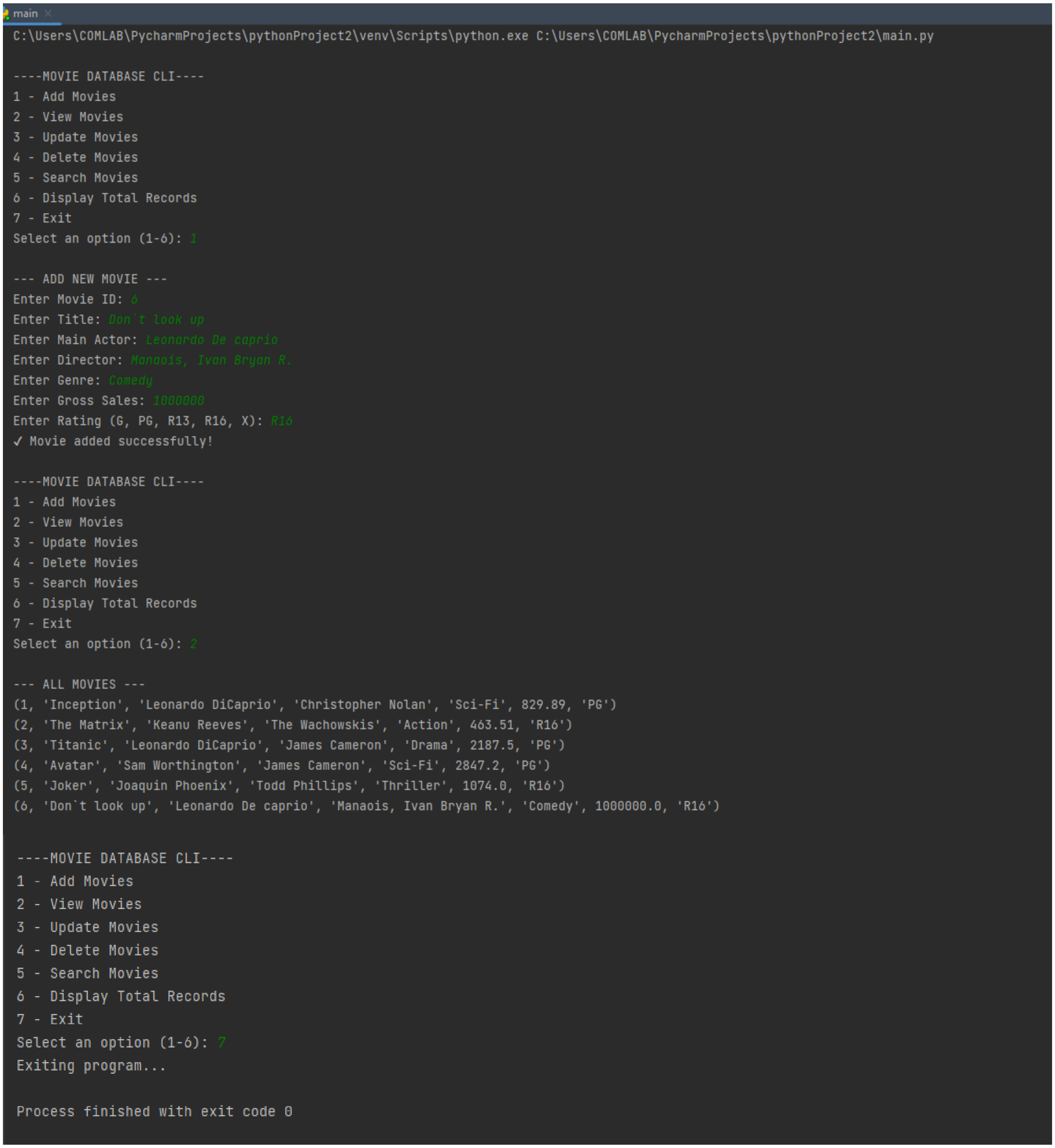
Print | Copy to clipboard | Export | Display chart | Create view

Bookmark this SQL query

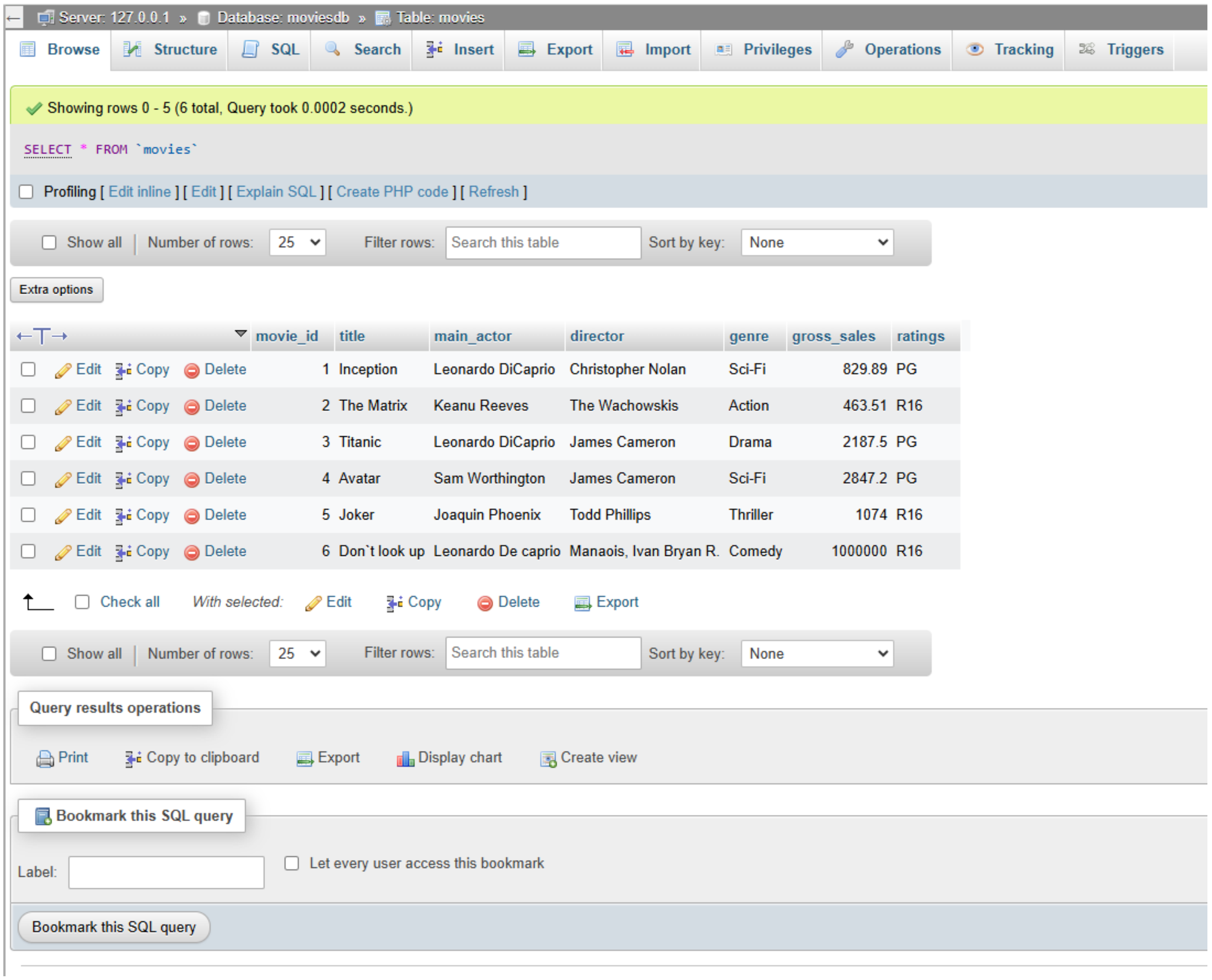
Label: | Let every user access this bookmark

Bookmark this SQL query

Sample output:



```
main
C:\Users\COMLAB\PycharmProjects\pythonProject2\venv\Scripts\python.exe C:\Users\COMLAB\PycharmProjects\pythonProject2\main.py

----MOVIE DATABASE CLI----
1 - Add Movies
2 - View Movies
3 - Update Movies
4 - Delete Movies
5 - Search Movies
6 - Display Total Records
7 - Exit
Select an option (1-6): 1

--- ADD NEW MOVIE ---
Enter Movie ID: 6
Enter Title: Don`t look up
Enter Main Actor: Leonardo De caprio
Enter Director: Manaois, Ivan Bryan R.
Enter Genre: Comedy
Enter Gross Sales: 1000000
Enter Rating (G, PG, R13, R16, X): R16
✓ Movie added successfully!

----MOVIE DATABASE CLI----
1 - Add Movies
2 - View Movies
3 - Update Movies
4 - Delete Movies
5 - Search Movies
6 - Display Total Records
7 - Exit
Select an option (1-6): 2

--- ALL MOVIES ---
(1, 'Inception', 'Leonardo DiCaprio', 'Christopher Nolan', 'Sci-Fi', 829.89, 'PG')
(2, 'The Matrix', 'Keanu Reeves', 'The Wachowskis', 'Action', 463.51, 'R16')
(3, 'Titanic', 'Leonardo DiCaprio', 'James Cameron', 'Drama', 2187.5, 'PG')
(4, 'Avatar', 'Sam Worthington', 'James Cameron', 'Sci-Fi', 2847.2, 'PG')
(5, 'Joker', 'Joaquin Phoenix', 'Todd Phillips', 'Thriller', 1074.0, 'R16')
(6, 'Don`t look up', 'Leonardo De caprio', 'Manaois, Ivan Bryan R.', 'Comedy', 1000000.0, 'R16')

----MOVIE DATABASE CLI----
1 - Add Movies
2 - View Movies
3 - Update Movies
4 - Delete Movies
5 - Search Movies
6 - Display Total Records
7 - Exit
Select an option (1-6): 7
Exiting program...

Process finished with exit code 0
```



Server: 127.0.0.1 » Database: moviesdb » Table: movies

Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Tracking
Triggers

Showing rows 0 - 5 (6 total, Query took 0.0002 seconds.)

```
SELECT * FROM `movies`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
Extra options

movie_id	title	main_actor	director	genre	gross_sales	ratings
1	Inception	Leonardo DiCaprio	Christopher Nolan	Sci-Fi	829.89	PG
2	The Matrix	Keanu Reeves	The Wachowskis	Action	463.51	R16
3	Titanic	Leonardo DiCaprio	James Cameron	Drama	2187.5	PG
4	Avatar	Sam Worthington	James Cameron	Sci-Fi	2847.2	PG
5	Joker	Joaquin Phoenix	Todd Phillips	Thriller	1074	R16
6	Don`t look up	Leonardo De caprio	Manaois, Ivan Bryan R.	Comedy	1000000	R16

Check all With selected: Edit Copy Delete Export
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
Query results operations
Print
Copy to clipboard
Export
Display chart
Create view
Bookmark this SQL query
Label:
Let every user access this bookmark
Bookmark this SQL query