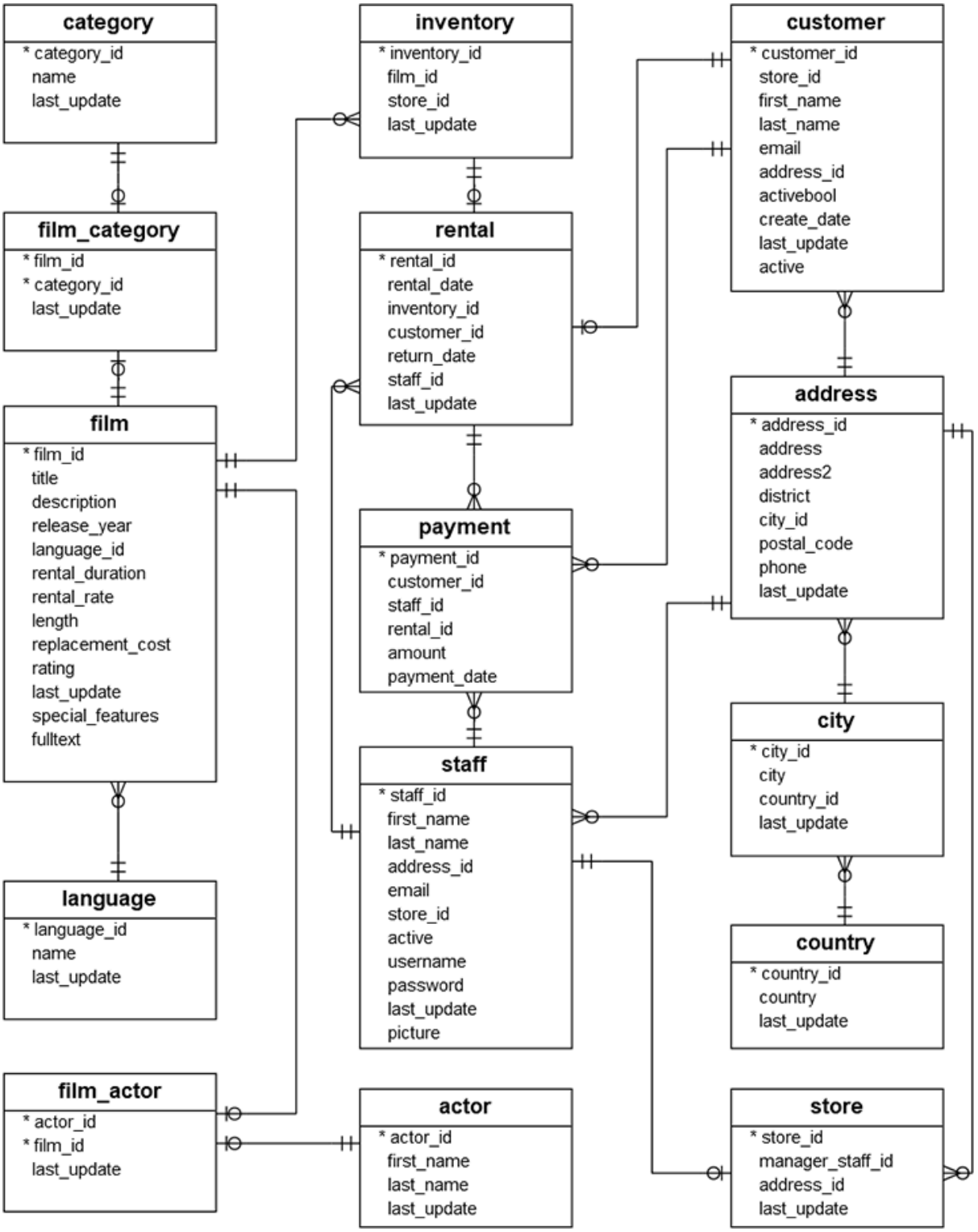
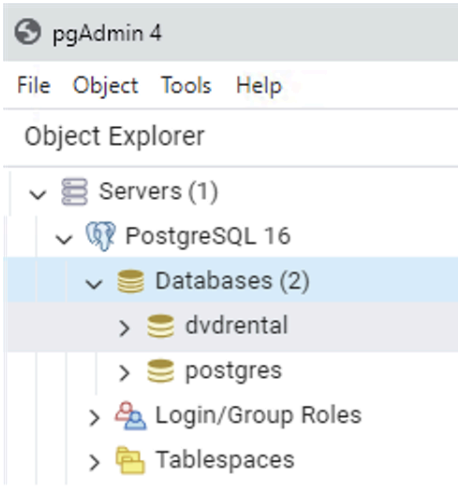
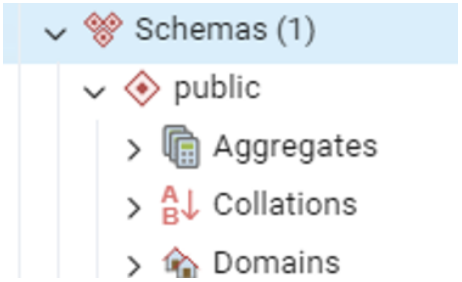
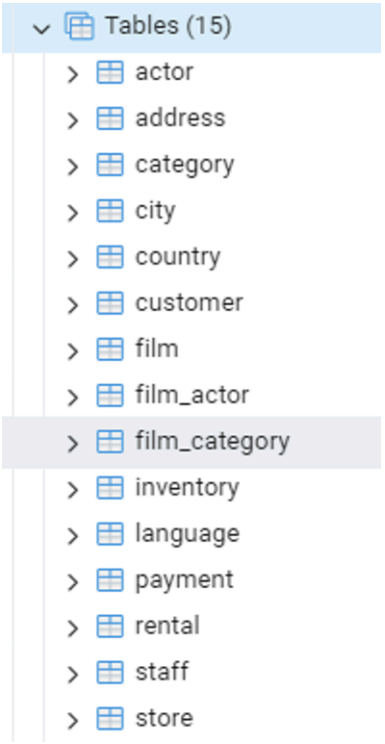
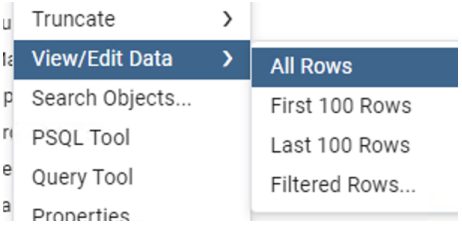
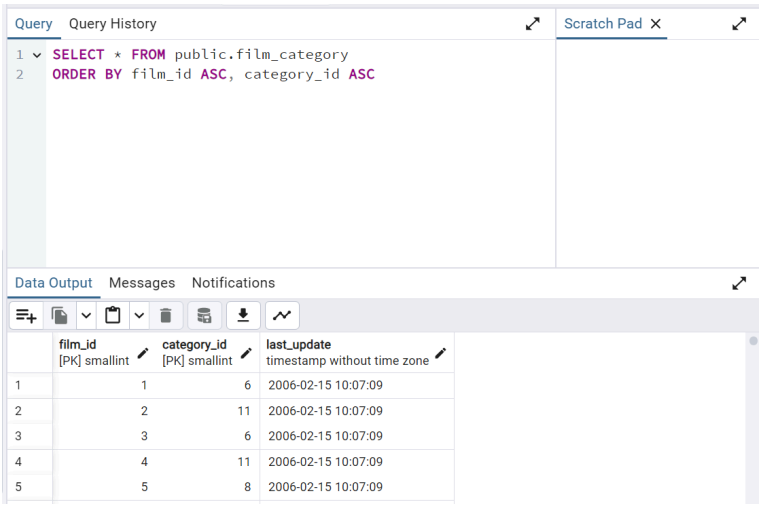
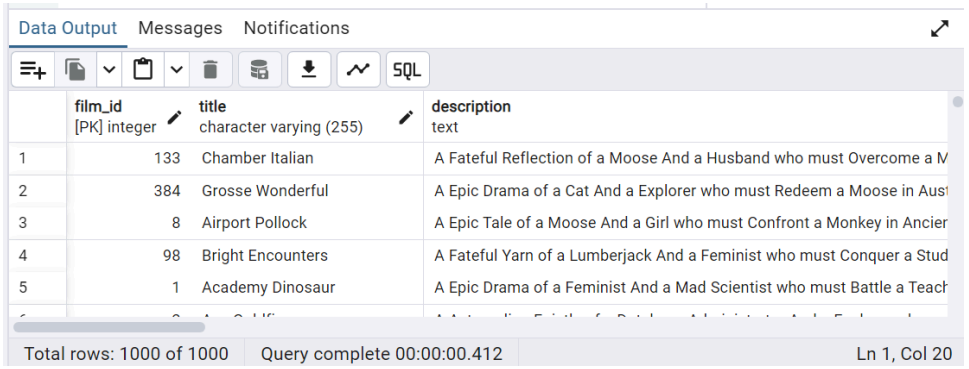
Java module 4

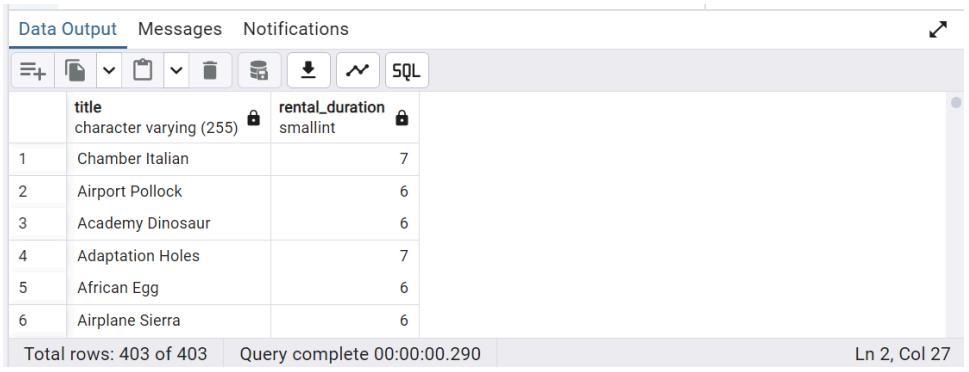
Exercises Day 1

https://www.postgresqltutorial.com/postgresql-getting-started/postgresql-sample-database/

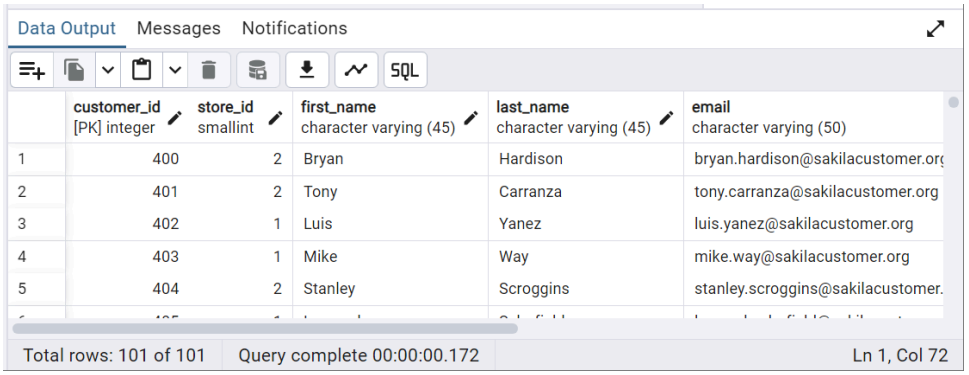


1.1 - pgAdmin	pgAdmin interface
Instructions	Step 1  Step 2  Step 3  Step 4 
Expected output	

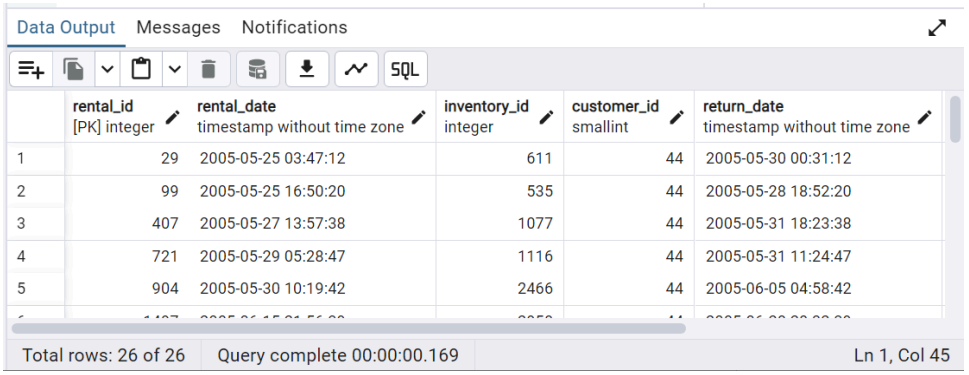
1.2 - pgAdmin	Retrieve information
Instructions	Using pgAdmin's Query Tool, retrieve all information from the **film** table
Expected output	
Solution	SELECT * FROM film;

1.3 - pgAdmin	Retrieve filtered results
Instructions	Using pgAdmin's Query Tool, find all **movies** with a **rental duration** longer than 5 days
Expected output	
Solution	SELECT title, rental_duration FROM film WHERE rental_duration > 5;

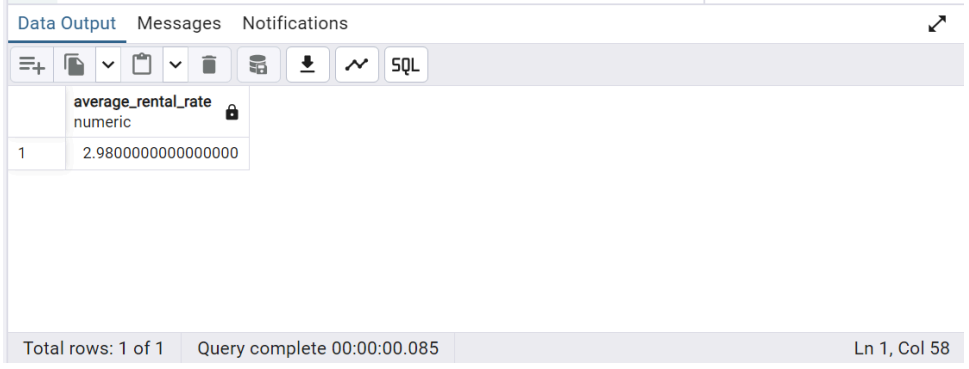
1.4 - pgAdmin	Retrieve filtered results
Instructions	Using pgAdmin's Query Tool, retrieve all **customers** with **customer_id** between 400 & 500.

Expected output	
Solution	SELECT * FROM customer WHERE customer_id >= 400 AND customer_id <= 500;

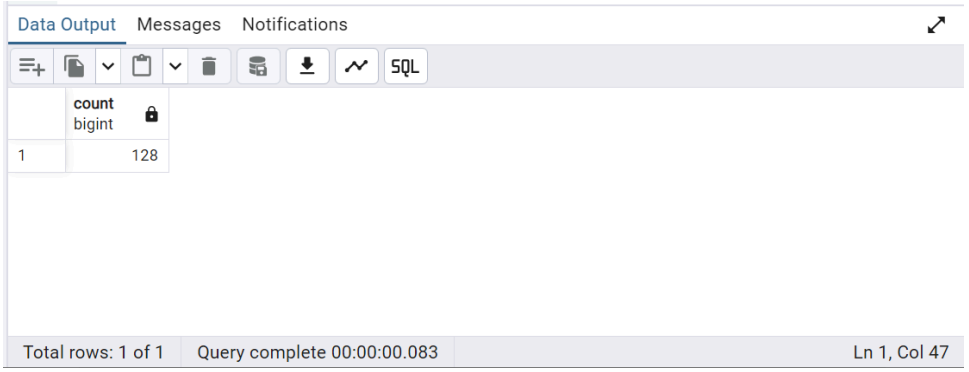
	customer_id [PK] integer	store_id smallint	first_name character varying (45)	last_name character varying (45)	email character varying (50)
1	400	2	Bryan	Hardison	bryan.hardison@sakilacustomer.org
2	401	2	Tony	Carranza	tony.carranza@sakilacustomer.org
3	402	1	Luis	Yanez	luis.yanez@sakilacustomer.org
4	403	1	Mike	Way	mike.way@sakilacustomer.org
5	404	2	Stanley	Scroggins	stanley.scroggins@sakilacustomer.

1.5 - pgAdmin	Retrieve filtered results
Instructions	Using pgAdmin's Query Tool, retrieve all rentals for the **customer** "Marie Turner".
Expected output	
Solution	SELECT * FROM rental WHERE customer_id = 44;

	rental_id [PK] integer	rental_date timestamp without time zone	inventory_id integer	customer_id smallint	return_date timestamp without time zone
1	29	2005-05-25 03:47:12	611	44	2005-05-30 00:31:12
2	99	2005-05-25 16:50:20	535	44	2005-05-28 18:52:20
3	407	2005-05-27 13:57:38	1077	44	2005-05-31 18:23:38
4	721	2005-05-29 05:28:47	1116	44	2005-05-31 11:24:47
5	904	2005-05-30 10:19:42	2466	44	2005-06-05 04:58:42

1.6 - pgAdmin	Calculate the average
Instructions	Using pgAdmin's Query Tool, calculate the **average rental rate** for all **films**.
Expected output	
Solution	SELECT AVG(rental_rate) AS average_rental_rate FROM film;

	average_rental_rate numeric
1	2.9800000000000000

1.7 - pgAdmin	Count entries
Instructions	Using pgAdmin's Query Tool, calculate the distinct **first names** of **actors**
Expected output	
Solution	SELECT COUNT(DISTINCT (first_name)) FROM actor;

1.8 - pgAdmin	Group by and count
Instructions	Using pgAdmin's Query Tool, count the number of films in each rating category. Hint: Use GROUP BY
Expected output	
Solution	SELECT rating, COUNT(*) AS film_count FROM film GROUP BY rating;

2.1 - JDBC	Connection
Instructions	Create a Java program to connect to the dvdrental database.
Expected output	Connected to the PostgreSQL servers successfully.
Solution	`import java.sql.DriverManager;` `import java.sql.Connection;` `import java.sql.SQLException;`

	`` public class DvdRentalConnection { public static void main(String[] args) { String url = "jdbc:postgresql://localhost:5432/dvdrental"; String user = "postgres"; String password = "admin"; try { Connection myConnection = DriverManager.getConnection(url, user, password); System.out.println("Connection successful"); } catch (SQLException e) { System.out.println("Connection error!"); } } } ```

2.2 - JDBC	Class to table mapping
Instructions	Create a Java Class for the actor table
Solution	``` public class Actor { private int actorId; private String firstName; private String lastName; public Actor(int actorId, String firstName, String lastName) { this.actorId = actorId; this.firstName = firstName; this.lastName = lastName; } public int getActorId() { return actorId; } public void setActorId(int actorId) { ```

	`` this.actorId = actorId; } public String getFirstName() { return firstName; } public void setFirstName(String firstName) { this.firstName = firstName; } public String getLastName() { return lastName; } public void setLastName(String lastName) { this.lastName = lastName; } } ```

2.3 - JDBC	Retrieve information
Instructions	Update the main program to retrieve all actors from the actor table and print their names.
Expected output	Actor ID: 1, Name: Penelope Guiness Actor ID: 2, Name: Nick Wahlberg Actor ID: 3, Name: Ed Chase Actor ID: 4, Name: Jennifer Davis Actor ID: 5, Name: Johnny Lollobrigida Actor ID: 6, Name: Bette Nicholson Actor ID: 7, Name: Grace Mostel Actor ID: 8, Name: Matthew Johansson Actor ID: 9, Name: Joe Swank Actor ID: 10, Name: Christian Gable Actor ID: 11, Name: Zero Cage Actor ID: 12, Name: Karl Berry Actor ID: 13, Name: Uma Wood Actor ID: 14, Name: Vivien Bergen Actor ID: 15, Name: Cuba Olivier …
Solution	``` import java.sql.DriverManager; import java.sql.ResultSet; import java.sql.Connection; import java.sql.SQLException; import java.sql.Statement; public class DvdRentalConnection { ```

```
    public static void main(String[] args) {

        String url =
"jdbc:postgresql://localhost:5432/dvdrental";
        String user = "postgres";
        String password = "admin";

        String query = "SELECT * FROM actor";

        try {
            Connection myConnection =
DriverManager.getConnection(url, user, password);
            System.out.println("Connection successful");
            Statement statement =
myConnection.createStatement();
            ResultSet results = statement.executeQuery(query);

            while (results.next()) {
                int id = results.getInt("actor_id");
                String firstName =
results.getString("first_name");
                String lastName =
results.getString("last_name");
                System.out.println("Actor ID: " + id + ",
Name: " + firstName + " " + lastName);
            }
        } catch (SQLException e) {
            System.out.println("There was an error: " +
e.getMessage());
        }
    }
}
```

2.4 - JDBC	Filter information
Instructions	Update the main program to allow the retrieval of actors from the actor table by their last name.
Expected output	Enter a last name: Penn Actor ID: 73, Name: Gary Penn Actor ID: 133, Name: Richard Penn

Solution	

```
import java.sql.*;
import java.util.Scanner;

public class FetchActorsByLastName {
    public static void main(String[] args) {
        String url =
"jdbc:postgresql://localhost:5432/dvdrental";
        String user = "postgres";
        String password = "admin";

        String query = "SELECT actor_id, first_name, last_name
FROM actor WHERE last_name LIKE ?";

        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a lastname: ");
        String searchLastName = scanner.nextLine();

        try (Connection conn =
DriverManager.getConnection(url, user, password);
             PreparedStatement pstmt =
conn.prepareStatement(query)) {

            pstmt.setString(1, searchLastName);
            try (ResultSet rs = pstmt.executeQuery()) {
                while (rs.next()) {
                    int id = rs.getInt("actor_id");
                    String firstName =
rs.getString("first_name");
                    String lastName =
rs.getString("last_name");
                    System.out.println("Actor ID: " + id + ",
Name: " + firstName + " " + lastName);
                }
            }
        } catch (SQLException e) {
            System.out.println(e.getMessage());
        } finally {
            scanner.close();
        }
    }
}
```

2.5 - JDBC	Counting
Instructions	Update the main program to allow printing how many actors there are in the table.
Expected output	Total number of actors: 200
Solution	

```java
import java.sql.*;

public class CountActors {
    public static void main(String[] args) {
        String url =
"jdbc:postgresql://localhost:5432/dvdrental";
        String user = "postgres";
        String password = "admin";

        String query = "SELECT COUNT(*) AS total_actors FROM
actor";

        try (Connection conn =
DriverManager.getConnection(url, user, password);
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(query)) {

            if (rs.next()) {
                int count = rs.getInt("total_actors");
                System.out.println("Total number of actors: "
+ count);
            }
        } catch (SQLException e) {
            System.out.println(e.getMessage());
        }
      }
}
```