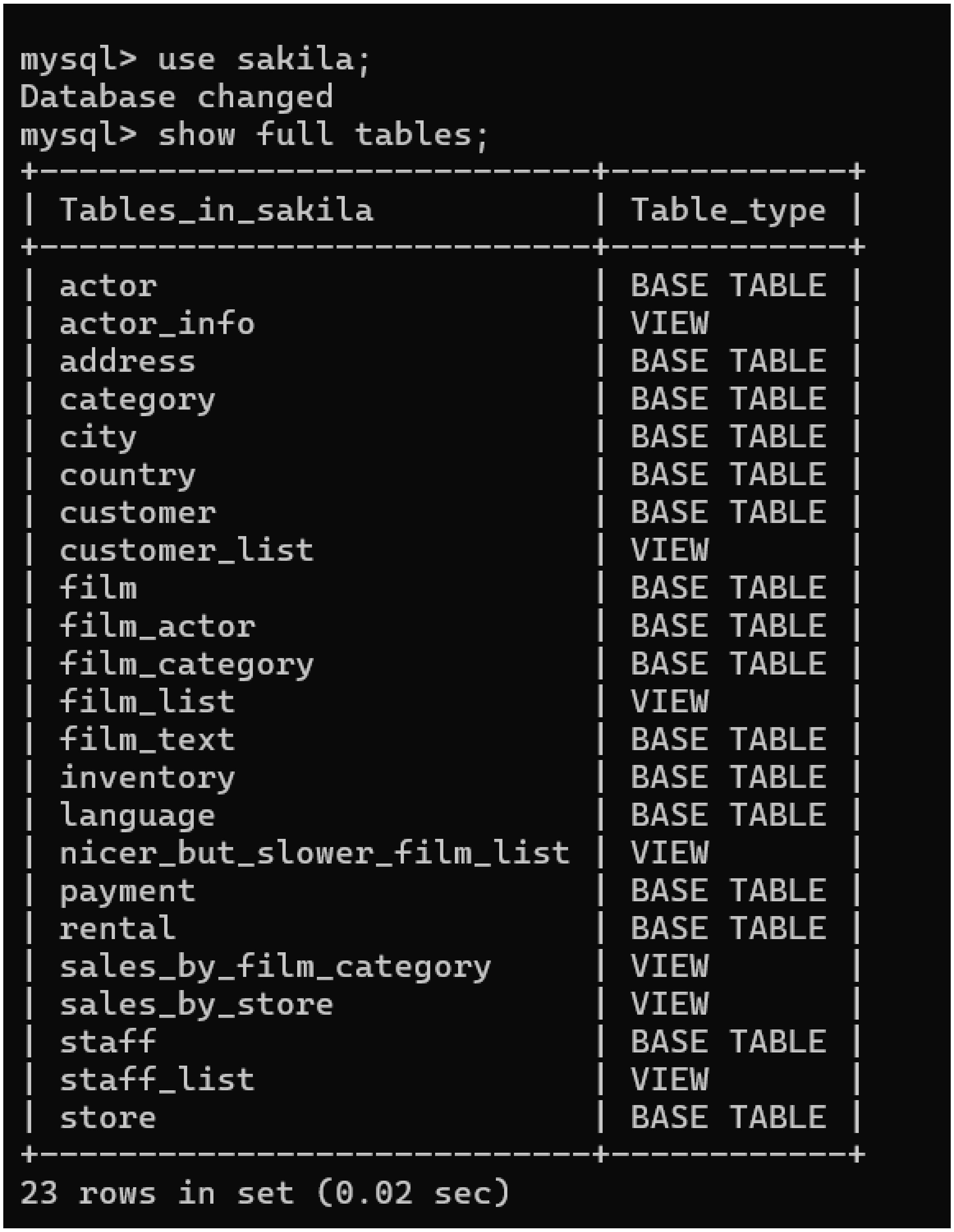
Mysql Task2

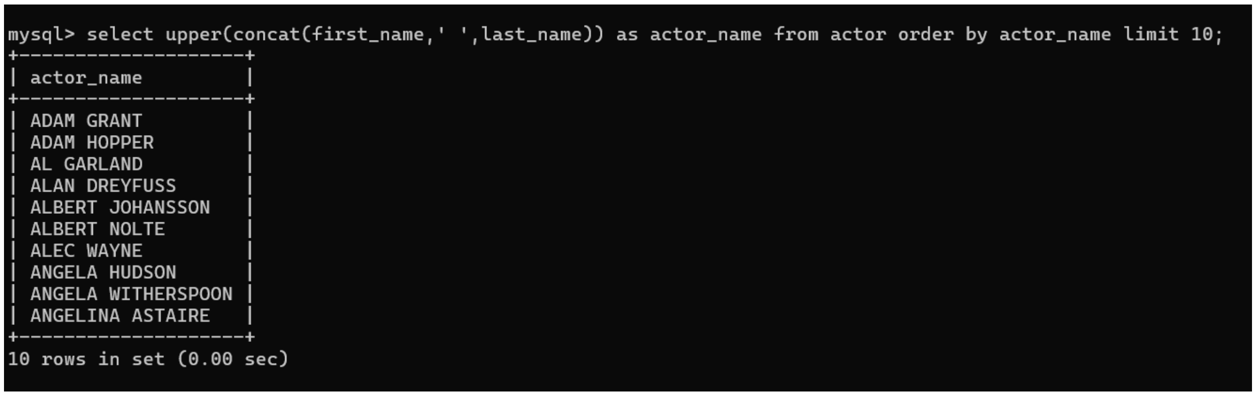
SAKILA



```
mysql> use sakila;
Database changed
mysql> show full tables;
+----------------------------+------------+
| Tables_in_sakila           | Table_type |
+----------------------------+------------+
| actor                      | BASE TABLE |
| actor_info                 | VIEW       |
| address                    | BASE TABLE |
| category                   | BASE TABLE |
| city                       | BASE TABLE |
| country                    | BASE TABLE |
| customer                   | BASE TABLE |
| customer_list              | VIEW       |
| film                       | BASE TABLE |
| film_actor                 | BASE TABLE |
| film_category              | BASE TABLE |
| film_list                  | VIEW       |
| film_text                  | BASE TABLE |
| inventory                  | BASE TABLE |
| language                   | BASE TABLE |
| nicer_but_slower_film_list | VIEW       |
| payment                    | BASE TABLE |
| rental                     | BASE TABLE |
| sales_by_film_category     | VIEW       |
| sales_by_store             | VIEW       |
| staff                      | BASE TABLE |
| staff_list                 | VIEW       |
| store                      | BASE TABLE |
+----------------------------+------------+
23 rows in set (0.02 sec)
```

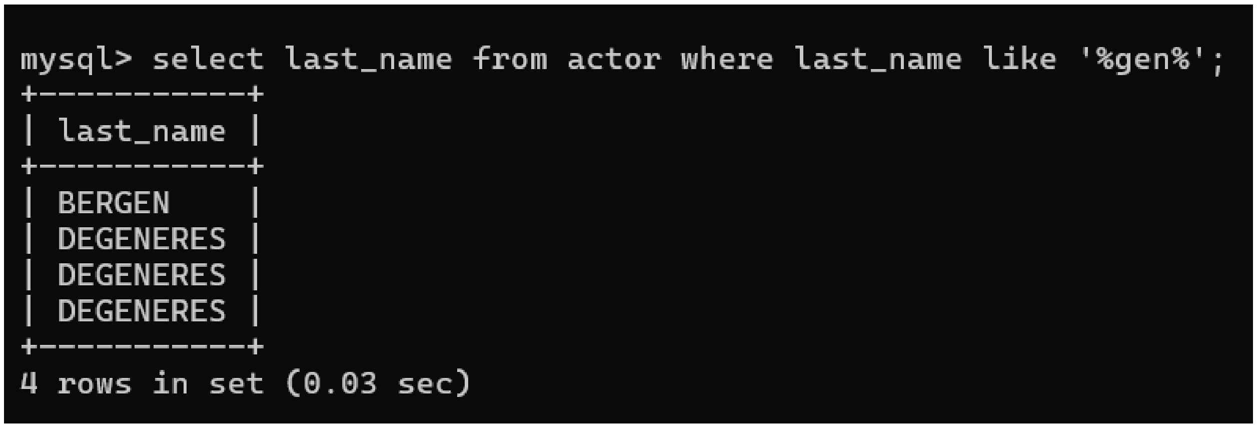
Exercises

1. 20Display the first and last name of each actor in a single column in upper case letters in alphabetical order. Name the column Actor Name.



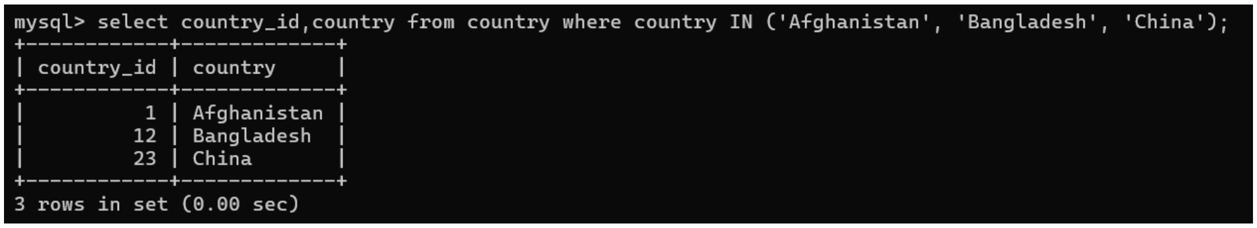
```
mysql> select upper(concat(first_name,' ',last_name)) as actor_name from actor order by actor_name limit 10;
+--------------------+
| actor_name         |
+--------------------+
| ADAM GRANT         |
| ADAM HOPPER        |
| AL GARLAND         |
| ALAN DREYFUSS      |
| ALBERT JOHANSSON   |
| ALBERT NOLTE       |
| ALEC WAYNE         |
| ANGELA HUDSON      |
| ANGELA WITHERSPOON |
| ANGELINA ASTAIRE   |
+--------------------+
10 rows in set (0.00 sec)
```

2. Find all actors whose last name contain the letters GEN:



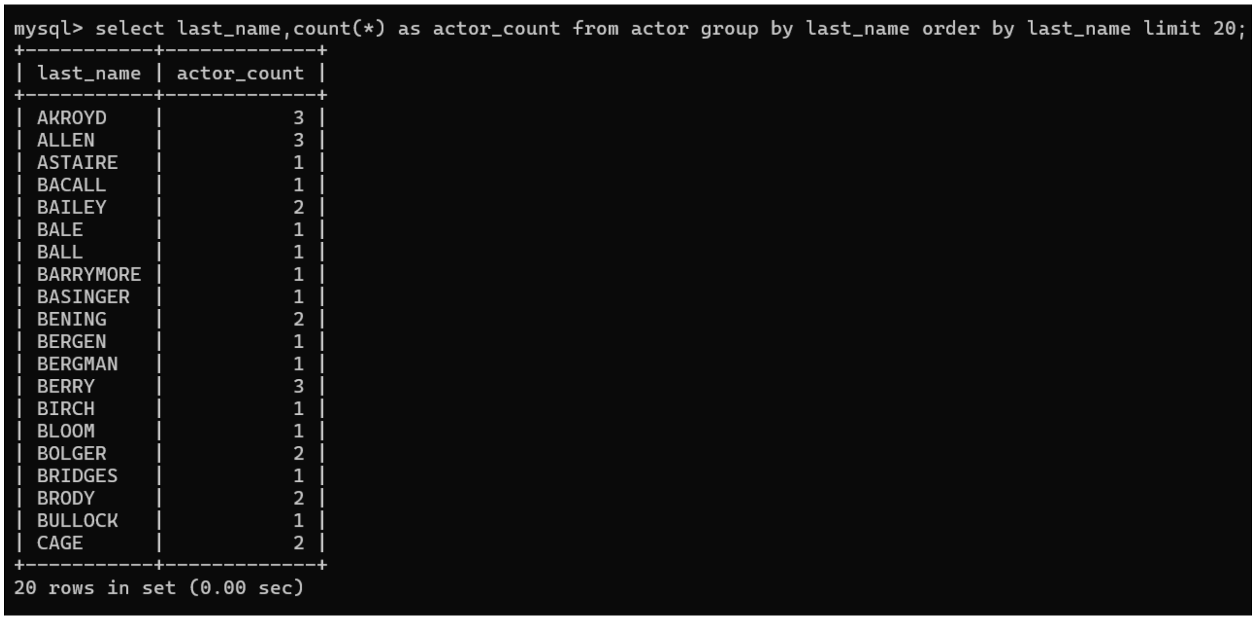
```
mysql> select last_name from actor where last_name like '%gen%';
+-----------+
| last_name |
+-----------+
| BERGEN    |
| DEGENERES |
| DEGENERES |
| DEGENERES |
+-----------+
4 rows in set (0.03 sec)
```

3. Using IN, display the country_id and country columns of the following countries: Afghanistan, Bangladesh, and China:



```
mysql> select country_id,country from country where country IN ('Afghanistan', 'Bangladesh', 'China');
+------------+-------------+
| country_id | country     |
+------------+-------------+
|          1 | Afghanistan |
|         12 | Bangladesh  |
|         23 | China       |
+------------+-------------+
3 rows in set (0.00 sec)
```

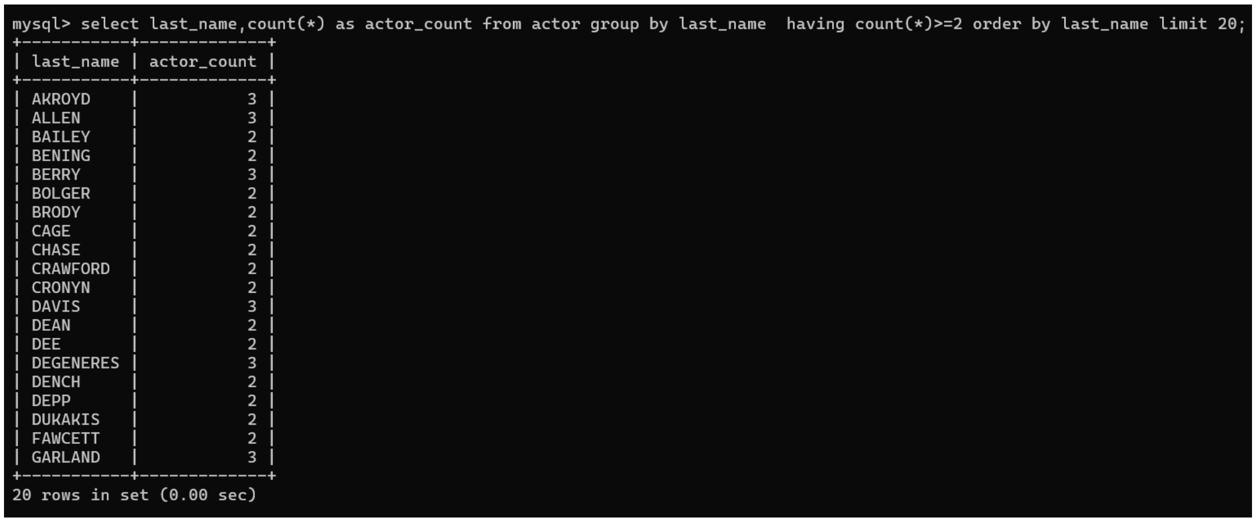
4. List the last names of actors, as well as how many actors have that last name



```
mysql> select last_name,count(*) as actor_count from actor group by last_name order by last_name limit 20;
+-----------+-------------+
| last_name | actor_count |
+-----------+-------------+
| AKROYD    |           3 |
| ALLEN     |           3 |
| ASTAIRE   |           1 |
| BACALL    |           1 |
| BAILEY    |           2 |
| BALE      |           1 |
| BALL      |           1 |
| BARRYMORE |           1 |
| BASINGER  |           1 |
| BENING    |           2 |
| BERGEN    |           1 |
| BERGMAN   |           1 |
| BERRY     |           3 |
| BIRCH     |           1 |
| BLOOM     |           1 |
| BOLGER    |           2 |
| BRIDGES   |           1 |
| BRODY     |           2 |
| BULLOCK   |           1 |
| CAGE      |           2 |
+-----------+-------------+
20 rows in set (0.00 sec)
```

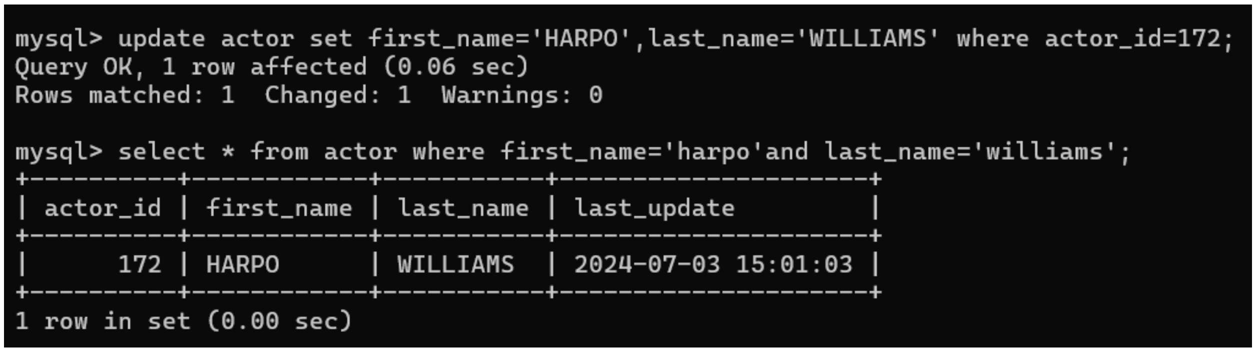
.

5. List last names of actors and the number of actors who have that last name, but only for names that are shared by at least two actors



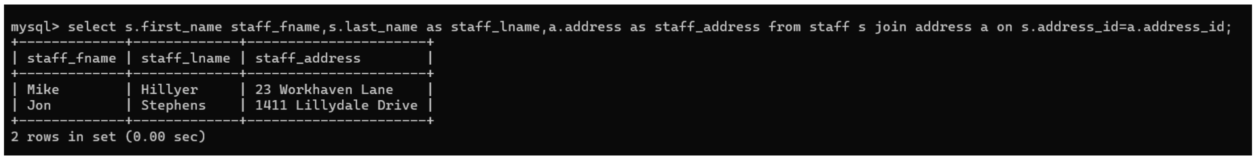
```
mysql> select last_name,count(*) as actor_count from actor group by last_name  having count(*)>=2 order by last_name limit 20;
+-----------+-------------+
| last_name | actor_count |
+-----------+-------------+
| AKROYD    |           3 |
| ALLEN     |           3 |
| BAILEY    |           2 |
| BENING    |           2 |
| BERRY     |           3 |
| BOLGER    |           2 |
| BRODY     |           2 |
| CAGE      |           2 |
| CHASE     |           2 |
| CRAWFORD  |           2 |
| CRONYN    |           2 |
| DAVIS     |           3 |
| DEAN      |           2 |
| DEE       |           2 |
| DEGENERES |           3 |
| DENCH     |           2 |
| DEPP      |           2 |
| DUKAKIS   |           2 |
| FAWCETT   |           2 |
| GARLAND   |           3 |
+-----------+-------------+
20 rows in set (0.00 sec)
```

6. T-he actor HARPO WILLIAMS was accidentally entered in the actor table as GROUCHO WILLIAMS. Write a query to fix the record.



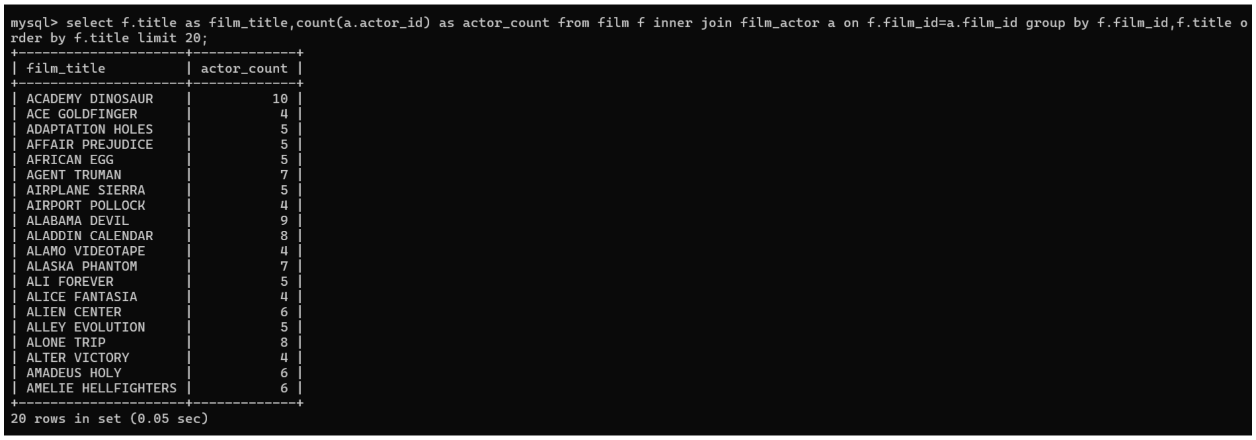
```
mysql> update actor set first_name='HARPO',last_name='WILLIAMS' where actor_id=172;
Query OK, 1 row affected (0.06 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from actor where first_name='harpo'and last_name='williams';
+----------+------------+-----------+---------------------+
| actor_id | first_name | last_name | last_update         |
+----------+------------+-----------+---------------------+
|      172 | HARPO      | WILLIAMS  | 2024-07-03 15:01:03 |
+----------+------------+-----------+---------------------+
1 row in set (0.00 sec)
```

7. Use JOIN to display the first and last names, as well as the address, of each staff member. Use the tables staff and address:



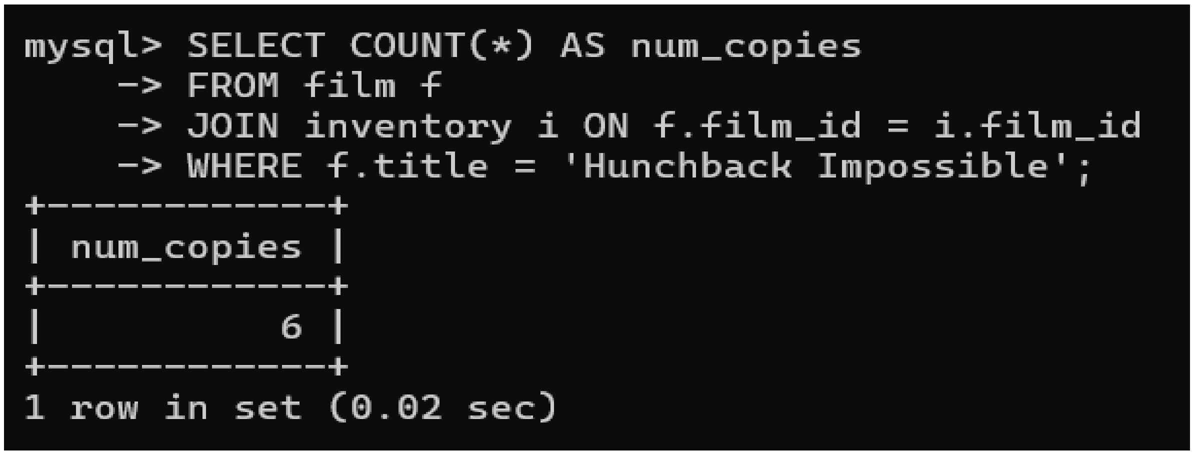
```
mysql> select s.first_name staff_fname,s.last_name as staff_lname,a.address as staff_address from staff s join address a on s.address_id=a.address_id;
+-------------+-------------+----------------------+
| staff_fname | staff_lname | staff_address        |
+-------------+-------------+----------------------+
| Mike        | Hillyer     | 23 Workhaven Lane    |
| Jon         | Stephens    | 1411 Lillydale Drive |
+-------------+-------------+----------------------+
2 rows in set (0.00 sec)
```

8. List each film and the number of actors who are listed for that film. Use tables film_actor and film. Use inner join.



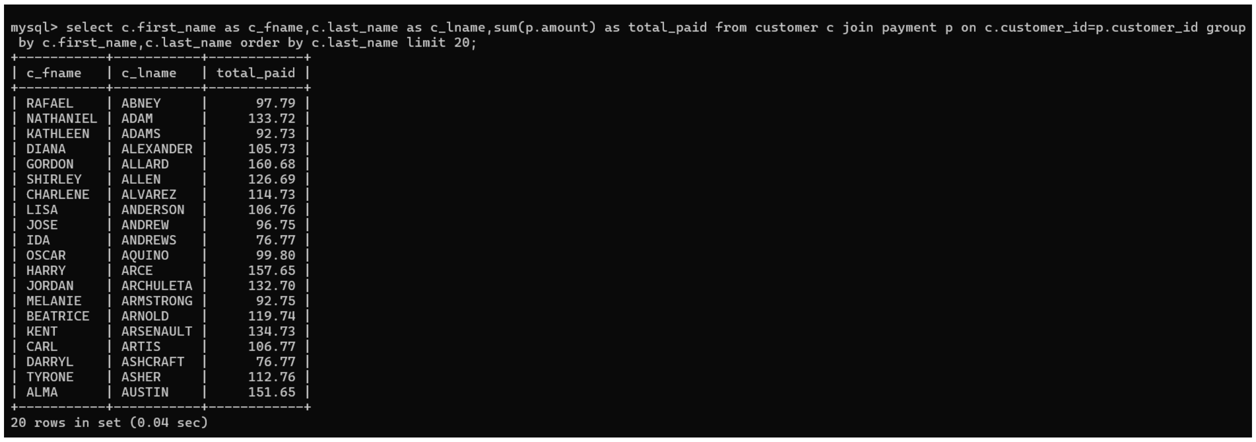
```
mysql> select f.title as film_title,count(a.actor_id) as actor_count from film f inner join film_actor a on f.film_id=a.film_id group by f.film_id,f.title o
rder by f.title limit 20;
+---------------------+-------------+
| film_title          | actor_count |
+---------------------+-------------+
| ACADEMY DINOSAUR    |          10 |
| ACE GOLDFINGER      |           4 |
| ADAPTATION HOLES    |           5 |
| AFFAIR PREJUDICE    |           5 |
| AFRICAN EGG         |           5 |
| AGENT TRUMAN        |           7 |
| AIRPLANE SIERRA     |           5 |
| AIRPORT POLLOCK     |           4 |
| ALABAMA DEVIL       |           9 |
| ALADDIN CALENDAR    |           8 |
| ALAMO VIDEOTAPE     |           4 |
| ALASKA PHANTOM      |           7 |
| ALI FOREVER         |           5 |
| ALICE FANTASIA      |           4 |
| ALIEN CENTER        |           6 |
| ALLEY EVOLUTION     |           5 |
| ALONE TRIP          |           8 |
| ALTER VICTORY       |           4 |
| AMADEUS HOLY        |           6 |
| AMELIE HELLFIGHTERS |           6 |
+---------------------+-------------+
20 rows in set (0.05 sec)
```

9. How many copies of the film Hunchback Impossible exist in the inventory system?



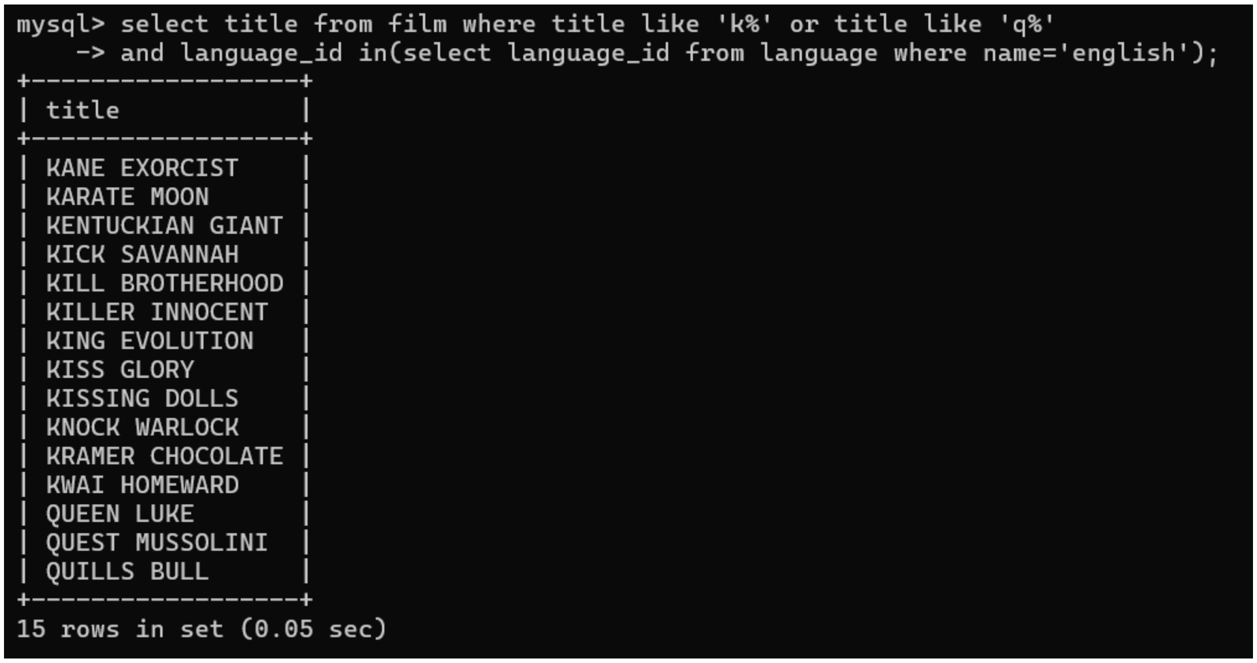
```
mysql> SELECT COUNT(*) AS num_copies
    -> FROM film f
    -> JOIN inventory i ON f.film_id = i.film_id
    -> WHERE f.title = 'Hunchback Impossible';
+------------+
| num_copies |
+------------+
|          6 |
+------------+
1 row in set (0.02 sec)
```

10. Using the tables payment and customer and the JOIN command, list the total paid by each customer. List the customers alphabetically by last name



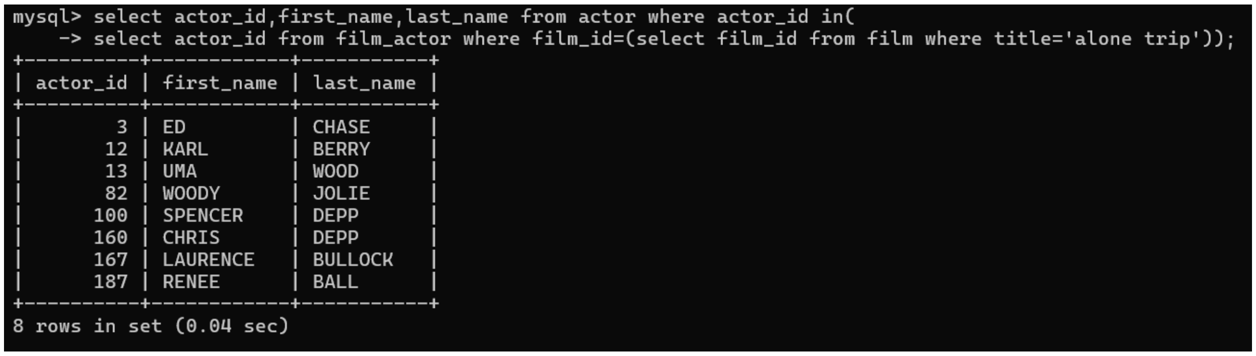
```
mysql> select c.first_name as c_fname,c.last_name as c_lname,sum(p.amount) as total_paid from customer c join payment p on c.customer_id=p.customer_id group
 by c.first_name,c.last_name order by c.last_name limit 20;
+-----------+-----------+------------+
| c_fname   | c_lname   | total_paid |
+-----------+-----------+------------+
| RAFAEL    | ABNEY     |      97.79 |
| NATHANIEL | ADAM      |     133.72 |
| KATHLEEN  | ADAMS     |      92.73 |
| DIANA     | ALEXANDER |     105.73 |
| GORDON    | ALLARD    |     160.68 |
| SHIRLEY   | ALLEN     |     126.69 |
| CHARLENE  | ALVAREZ   |     114.73 |
| LISA      | ANDERSON  |     106.76 |
| JOSE      | ANDREW    |      96.75 |
| IDA       | ANDREWS   |      76.77 |
| OSCAR     | AQUINO    |      99.80 |
| HARRY     | ARCE      |     157.65 |
| JORDAN    | ARCHULETA |     132.70 |
| MELANIE   | ARMSTRONG |      92.75 |
| BEATRICE  | ARNOLD    |     119.74 |
| KENT      | ARSENAULT |     134.73 |
| CARL      | ARTIS     |     106.77 |
| DARRYL    | ASHCRAFT  |      76.77 |
| TYRONE    | ASHER     |     112.76 |
| ALMA      | AUSTIN    |     151.65 |
+-----------+-----------+------------+
20 rows in set (0.04 sec)
```

11. The music of Queen and Kris Kristofferson have seen an unlikely resurgence. As an unintended consequence, films starting with the letters `K` and `Q` have also soared in popularity. Use subqueries to display the titles of movies starting with the letters `K` and `Q` whose language is English.



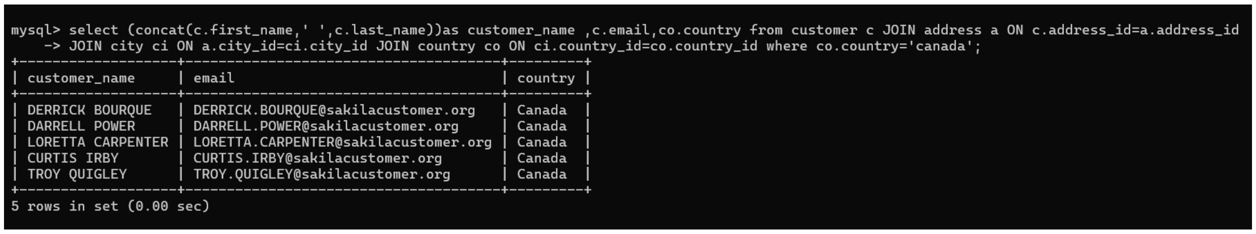
```
mysql> select title from film where title like 'k%' or title like 'q%'
    -> and language_id in(select language_id from language where name='english');
+-------------------+
| title             |
+-------------------+
| KANE EXORCIST     |
| KARATE MOON       |
| KENTUCKIAN GIANT  |
| KICK SAVANNAH     |
| KILL BROTHERHOOD  |
| KILLER INNOCENT   |
| KING EVOLUTION    |
| KISS GLORY        |
| KISSING DOLLS     |
| KNOCK WARLOCK     |
| KRAMER CHOCOLATE  |
| KWAI HOMEWARD     |
| QUEEN LUKE        |
| QUEST MUSSOLINI   |
| QUILLS BULL       |
+-------------------+
15 rows in set (0.05 sec)
```

12. Use subqueries to display all actors who appear in the film `Alone Trip`.



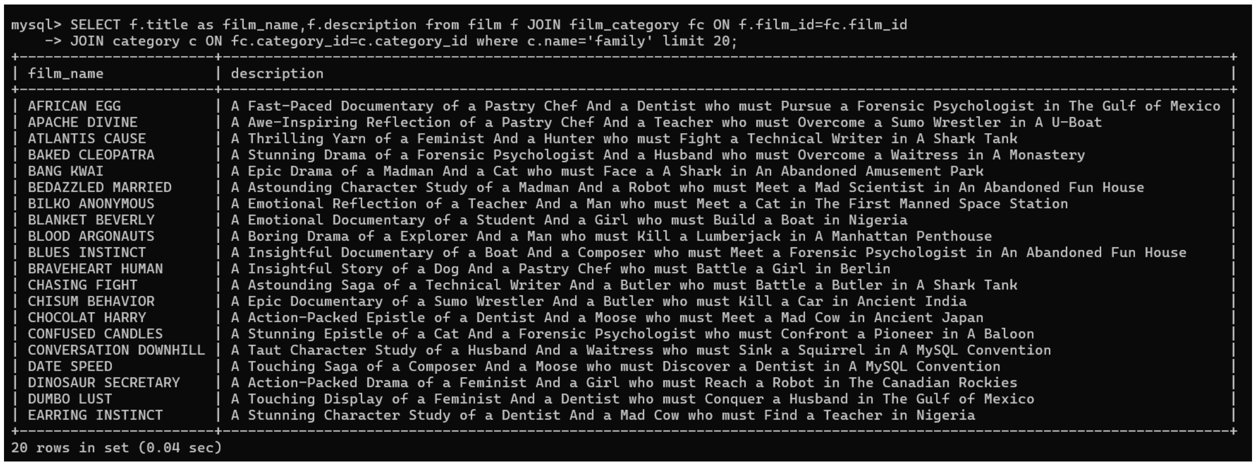
```
mysql> select actor_id,first_name,last_name from actor where actor_id in(
    -> select actor_id from film_actor where film_id=(select film_id from film where title='alone trip'));
+----------+------------+-----------+
| actor_id | first_name | last_name |
+----------+------------+-----------+
|        3 | ED         | CHASE     |
|       12 | KARL       | BERRY     |
|       13 | UMA        | WOOD      |
|       82 | WOODY      | JOLIE     |
|      100 | SPENCER    | DEPP      |
|      160 | CHRIS      | DEPP      |
|      167 | LAURENCE   | BULLOCK   |
|      187 | RENEE      | BALL      |
+----------+------------+-----------+
8 rows in set (0.04 sec)
```

13. You want to run an email marketing campaign in Canada, for which you will need the names and email addresses of all Canadian customers. Use joins to retrieve this information.



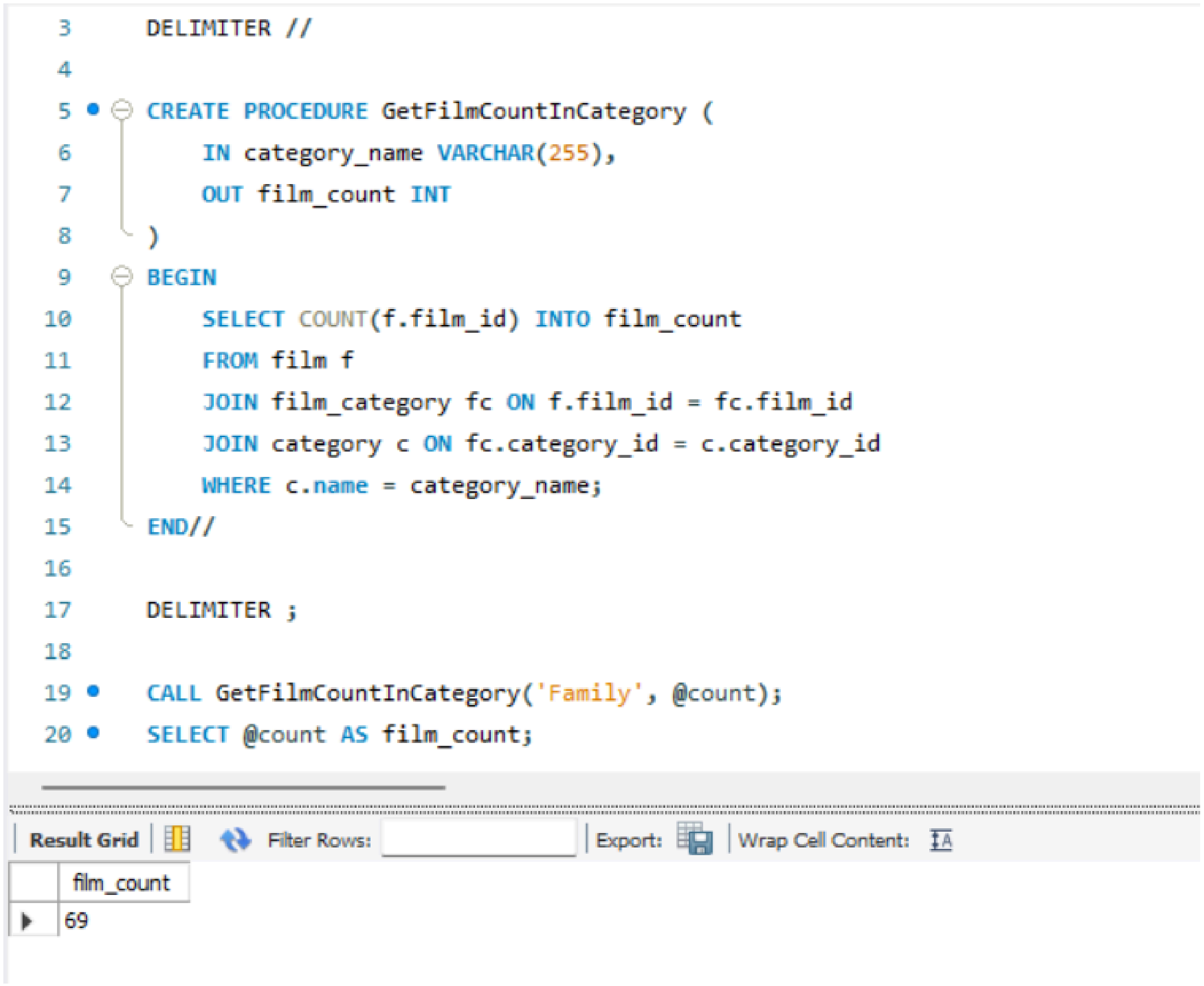
```
mysql> select (concat(c.first_name,' ',c.last_name))as customer_name ,c.email,co.country from customer c JOIN address a ON c.address_id=a.address_id
    -> JOIN city ci ON a.city_id=ci.city_id JOIN country co ON ci.country_id=co.country_id where co.country='canada';
+-------------------+---------------------------------------+---------+
| customer_name     | email                                 | country |
+-------------------+---------------------------------------+---------+
| DERRICK BOURQUE   | DERRICK.BOURQUE@sakilacustomer.org    | Canada  |
| DARRELL POWER     | DARRELL.POWER@sakilacustomer.org      | Canada  |
| LORETTA CARPENTER | LORETTA.CARPENTER@sakilacustomer.org  | Canada  |
| CURTIS IRBY       | CURTIS.IRBY@sakilacustomer.org        | Canada  |
| TROY QUIGLEY      | TROY.QUIGLEY@sakilacustomer.org       | Canada  |
+-------------------+---------------------------------------+---------+
5 rows in set (0.00 sec)
```

14. Sales have been lagging among young families, and you wish to target all family movies for a promotion. Identify all movies categorized as family films.



```
mysql> SELECT f.title as film_name,f.description from film f JOIN film_category fc ON f.film_id=fc.film_id
    -> JOIN category c ON fc.category_id=c.category_id where c.name='family' limit 20;
+-----------------------+-----------------------------------------------------------------------------------------------------------------------+
| film_name             | description                                                                                                           |
+-----------------------+-----------------------------------------------------------------------------------------------------------------------+
| AFRICAN EGG           | A Fast-Paced Documentary of a Pastry Chef And a Dentist who must Pursue a Forensic Psychologist in The Gulf of Mexico |
| APACHE DIVINE         | A Awe-Inspiring Reflection of a Pastry Chef And a Teacher who must Overcome a Sumo Wrestler in A U-Boat               |
| ATLANTIS CAUSE        | A Thrilling Yarn of a Feminist And a Hunter who must Fight a Technical Writer in A Shark Tank                         |
| BAKED CLEOPATRA       | A Stunning Drama of a Forensic Psychologist And a Husband who must Overcome a Waitress in A Monastery                 |
| BANG KWAI             | A Epic Drama of a Madman And a Cat who must Face a A Shark in An Abandoned Amusement Park                             |
| BEDAZZLED MARRIED     | A Astounding Character Study of a Madman And a Robot who must Meet a Mad Scientist in An Abandoned Fun House          |
| BILKO ANONYMOUS       | A Emotional Reflection of a Teacher And a Man who must Meet a Cat in The First Manned Space Station                   |
| BLANKET BEVERLY       | A Emotional Documentary of a Student And a Girl who must Build a Boat in Nigeria                                      |
| BLOOD ARGONAUTS       | A Boring Drama of a Explorer And a Man who must Kill a Lumberjack in A Manhattan Penthouse                            |
| BLUES INSTINCT        | A Insightful Documentary of a Boat And a Composer who must Meet a Forensic Psychologist in An Abandoned Fun House     |
| BRAVEHEART HUMAN      | A Insightful Story of a Dog And a Pastry Chef who must Battle a Girl in Berlin                                        |
| CHASING FIGHT         | A Astounding Saga of a Technical Writer And a Butler who must Battle a Butler in A Shark Tank                         |
| CHISUM BEHAVIOR       | A Epic Documentary of a Sumo Wrestler And a Butler who must Kill a Car in Ancient India                               |
| CHOCOLAT HARRY        | A Action-Packed Epistle of a Dentist And a Moose who must Meet a Mad Cow in Ancient Japan                             |
| CONFUSED CANDLES      | A Stunning Epistle of a Cat And a Forensic Psychologist who must Confront a Pioneer in A Baloon                       |
| CONVERSATION DOWNHILL | A Taut Character Study of a Husband And a Waitress who must Sink a Squirrel in A MySQL Convention                     |
| DATE SPEED            | A Touching Saga of a Composer And a Moose who must Discover a Dentist in A MySQL Convention                           |
| DINOSAUR SECRETARY    | A Action-Packed Drama of a Feminist And a Girl who must Reach a Robot in The Canadian Rockies                         |
| DUMBO LUST            | A Touching Display of a Feminist And a Dentist who must Conquer a Husband in The Gulf of Mexico                       |
| EARRING INSTINCT      | A Stunning Character Study of a Dentist And a Mad Cow who must Find a Teacher in Nigeria                              |
+-----------------------+-----------------------------------------------------------------------------------------------------------------------+
20 rows in set (0.04 sec)
```

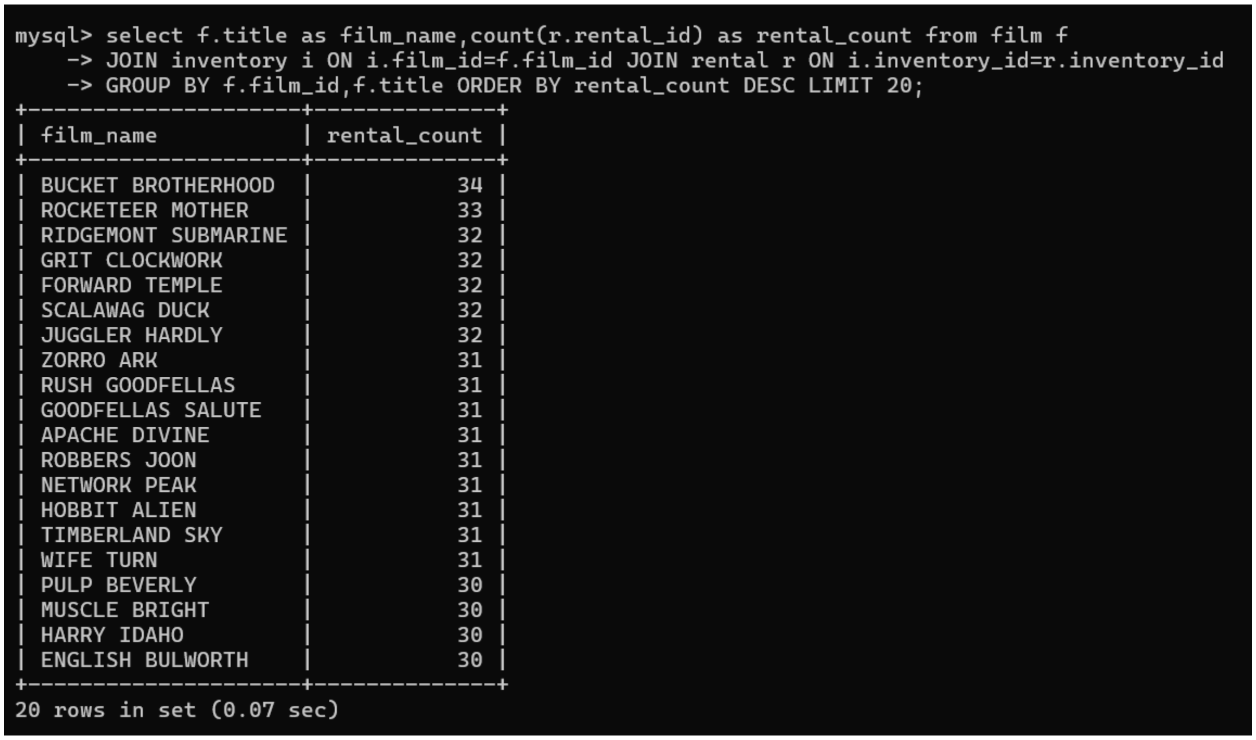
15. Create a Stored procedure to get the count of films in the input category (IN category_name, OUT count)



```sql
DELIMITER //

CREATE PROCEDURE GetFilmCountInCategory (
    IN category_name VARCHAR(255),
    OUT film_count INT
)
BEGIN
    SELECT COUNT(f.film_id) INTO film_count
    FROM film f
    JOIN film_category fc ON f.film_id = fc.film_id
    JOIN category c ON fc.category_id = c.category_id
    WHERE c.name = category_name;
END//

DELIMITER ;

CALL GetFilmCountInCategory('Family', @count);
SELECT @count AS film_count;
```

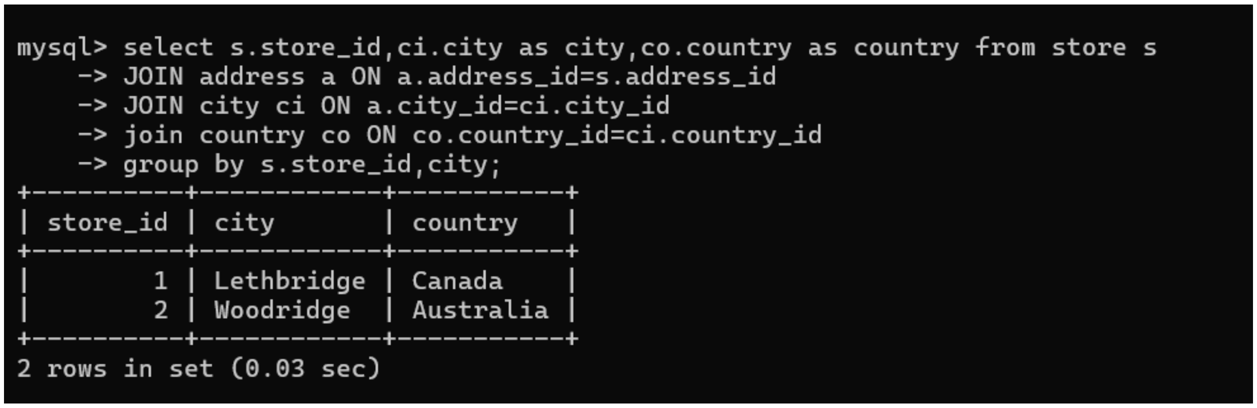
film_count
69

16. Display the most frequently rented movies in descending order.



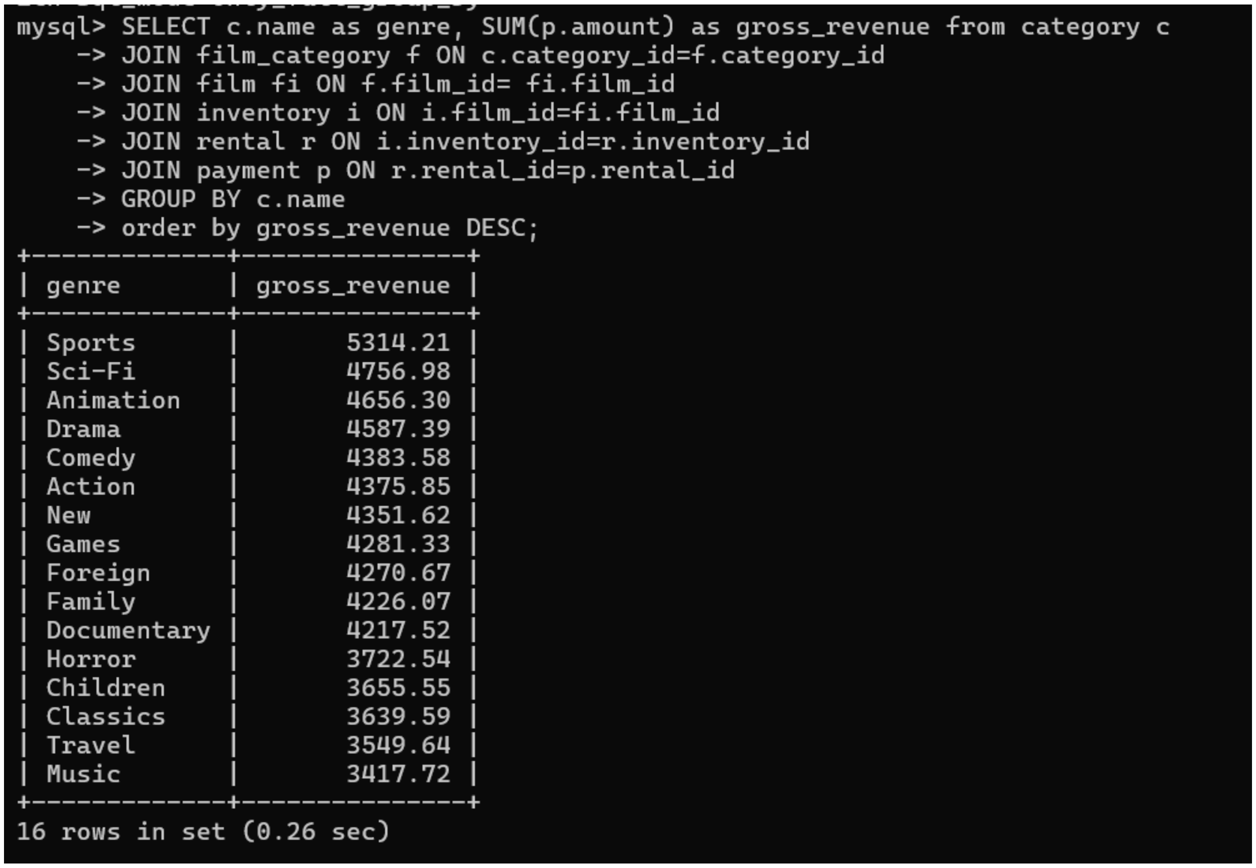
```
mysql> select f.title as film_name,count(r.rental_id) as rental_count from film f
    -> JOIN inventory i ON i.film_id=f.film_id JOIN rental r ON i.inventory_id=r.inventory_id
    -> GROUP BY f.film_id,f.title ORDER BY rental_count DESC LIMIT 20;
+---------------------+--------------+
| film_name           | rental_count |
+---------------------+--------------+
| BUCKET BROTHERHOOD  |           34 |
| ROCKETEER MOTHER    |           33 |
| RIDGEMONT SUBMARINE |           32 |
| GRIT CLOCKWORK      |           32 |
| FORWARD TEMPLE      |           32 |
| SCALAWAG DUCK       |           32 |
| JUGGLER HARDLY      |           32 |
| ZORRO ARK           |           31 |
| RUSH GOODFELLAS     |           31 |
| GOODFELLAS SALUTE   |           31 |
| APACHE DIVINE       |           31 |
| ROBBERS JOON        |           31 |
| NETWORK PEAK        |           31 |
| HOBBIT ALIEN        |           31 |
| TIMBERLAND SKY      |           31 |
| WIFE TURN           |           31 |
| PULP BEVERLY        |           30 |
| MUSCLE BRIGHT       |           30 |
| HARRY IDAHO         |           30 |
| ENGLISH BULWORTH    |           30 |
+---------------------+--------------+
20 rows in set (0.07 sec)
```

17. Write a query to display for each store its store ID, city, and country.



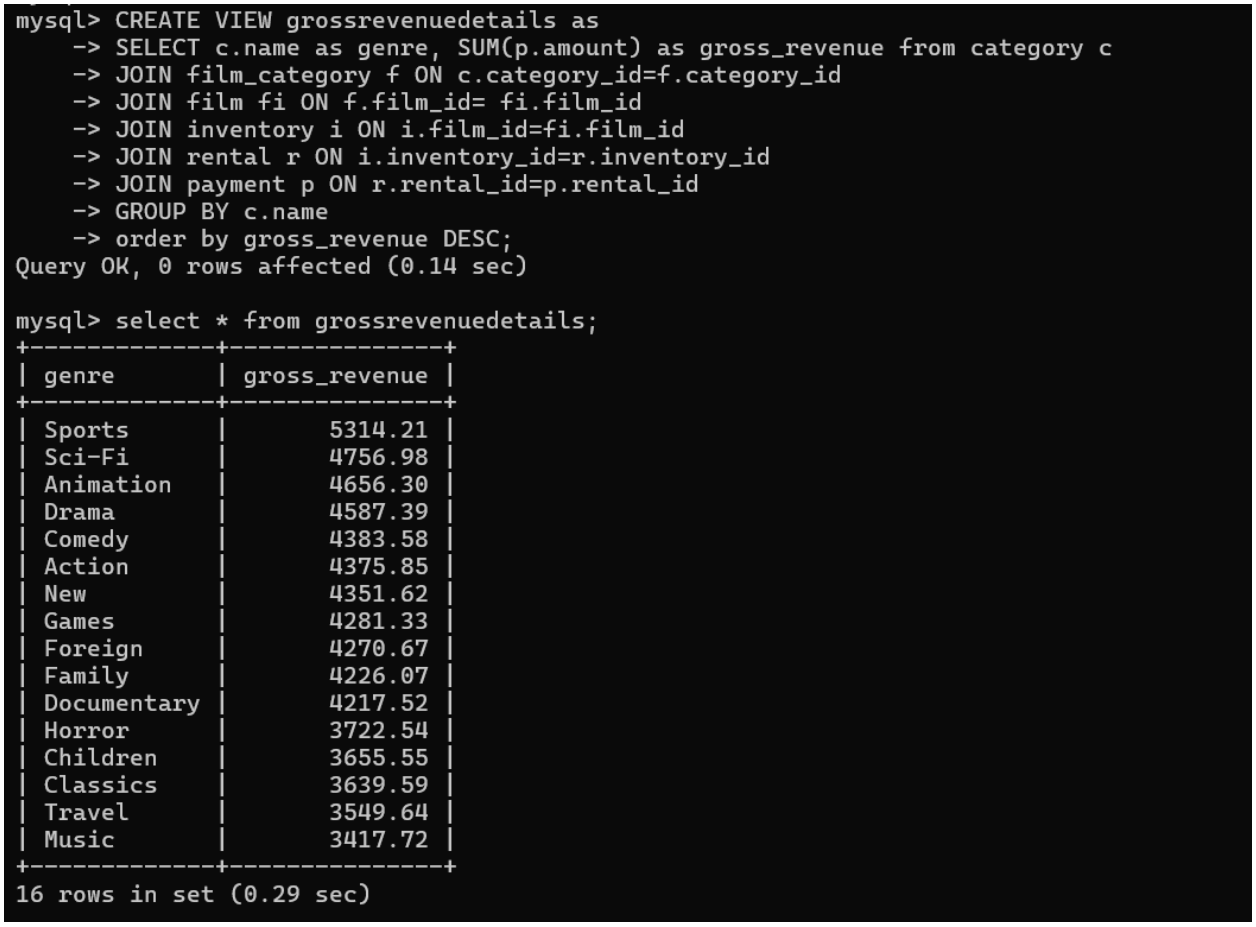
```
mysql> select s.store_id,ci.city as city,co.country as country from store s
    -> JOIN address a ON a.address_id=s.address_id
    -> JOIN city ci ON a.city_id=ci.city_id
    -> join country co ON co.country_id=ci.country_id
    -> group by s.store_id,city;
+----------+------------+-----------+
| store_id | city       | country   |
+----------+------------+-----------+
|        1 | Lethbridge | Canada    |
|        2 | Woodridge  | Australia |
+----------+------------+-----------+
2 rows in set (0.03 sec)
```

18. List the genres and its gross revenue.



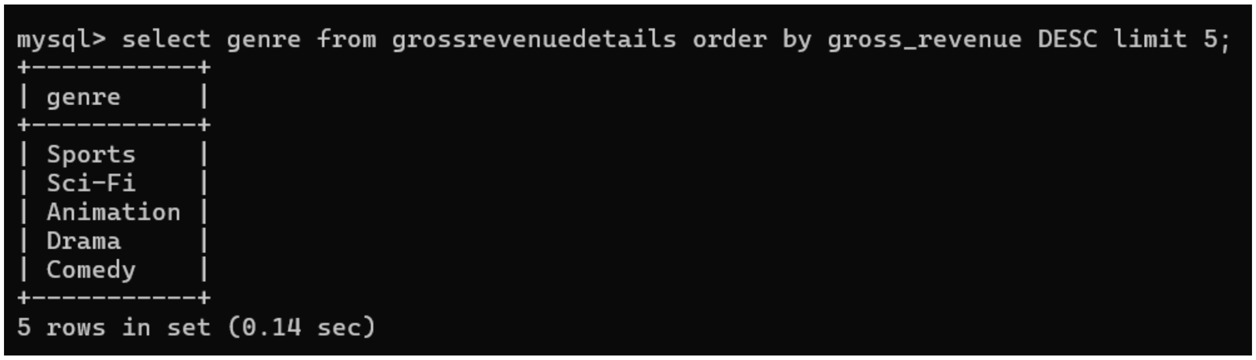
```
mysql> SELECT c.name as genre, SUM(p.amount) as gross_revenue from category c
    -> JOIN film_category f ON c.category_id=f.category_id
    -> JOIN film fi ON f.film_id= fi.film_id
    -> JOIN inventory i ON i.film_id=fi.film_id
    -> JOIN rental r ON i.inventory_id=r.inventory_id
    -> JOIN payment p ON r.rental_id=p.rental_id
    -> GROUP BY c.name
    -> order by gross_revenue DESC;
+-------------+---------------+
| genre       | gross_revenue |
+-------------+---------------+
| Sports      |       5314.21 |
| Sci-Fi      |       4756.98 |
| Animation   |       4656.30 |
| Drama       |       4587.39 |
| Comedy      |       4383.58 |
| Action      |       4375.85 |
| New         |       4351.62 |
| Games       |       4281.33 |
| Foreign     |       4270.67 |
| Family      |       4226.07 |
| Documentary |       4217.52 |
| Horror      |       3722.54 |
| Children    |       3655.55 |
| Classics    |       3639.59 |
| Travel      |       3549.64 |
| Music       |       3417.72 |
+-------------+---------------+
16 rows in set (0.26 sec)
```

19.Create a View for the above query(18)



```
mysql> CREATE VIEW grossrevenuedetails as
    -> SELECT c.name as genre, SUM(p.amount) as gross_revenue from category c
    -> JOIN film_category f ON c.category_id=f.category_id
    -> JOIN film fi ON f.film_id= fi.film_id
    -> JOIN inventory i ON i.film_id=fi.film_id
    -> JOIN rental r ON i.inventory_id=r.inventory_id
    -> JOIN payment p ON r.rental_id=p.rental_id
    -> GROUP BY c.name
    -> order by gross_revenue DESC;
Query OK, 0 rows affected (0.14 sec)

mysql> select * from grossrevenuedetails;
+-------------+---------------+
| genre       | gross_revenue |
+-------------+---------------+
| Sports      |       5314.21 |
| Sci-Fi      |       4756.98 |
| Animation   |       4656.30 |
| Drama       |       4587.39 |
| Comedy      |       4383.58 |
| Action      |       4375.85 |
| New         |       4351.62 |
| Games       |       4281.33 |
| Foreign     |       4270.67 |
| Family      |       4226.07 |
| Documentary |       4217.52 |
| Horror      |       3722.54 |
| Children    |       3655.55 |
| Classics    |       3639.59 |
| Travel      |       3549.64 |
| Music       |       3417.72 |
+-------------+---------------+
16 rows in set (0.29 sec)
```

20.Select top 5 genres in gross revenue view.



```
mysql> select genre from grossrevenuedetails order by gross_revenue DESC limit 5;
+-----------+
| genre     |
+-----------+
| Sports    |
| Sci-Fi    |
| Animation |
| Drama     |
| Comedy    |
+-----------+
5 rows in set (0.14 sec)
```