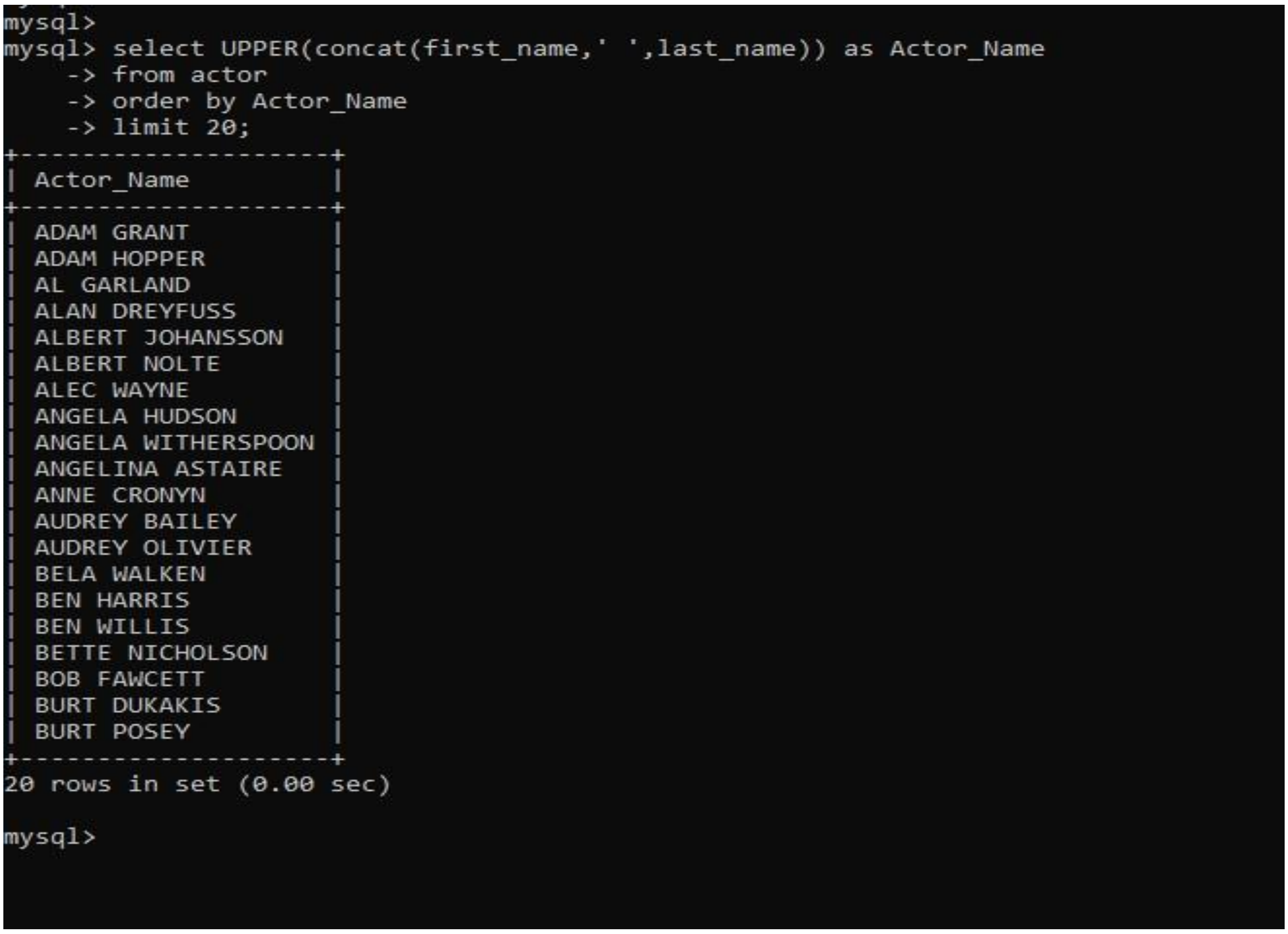
MySQL TASK – 3

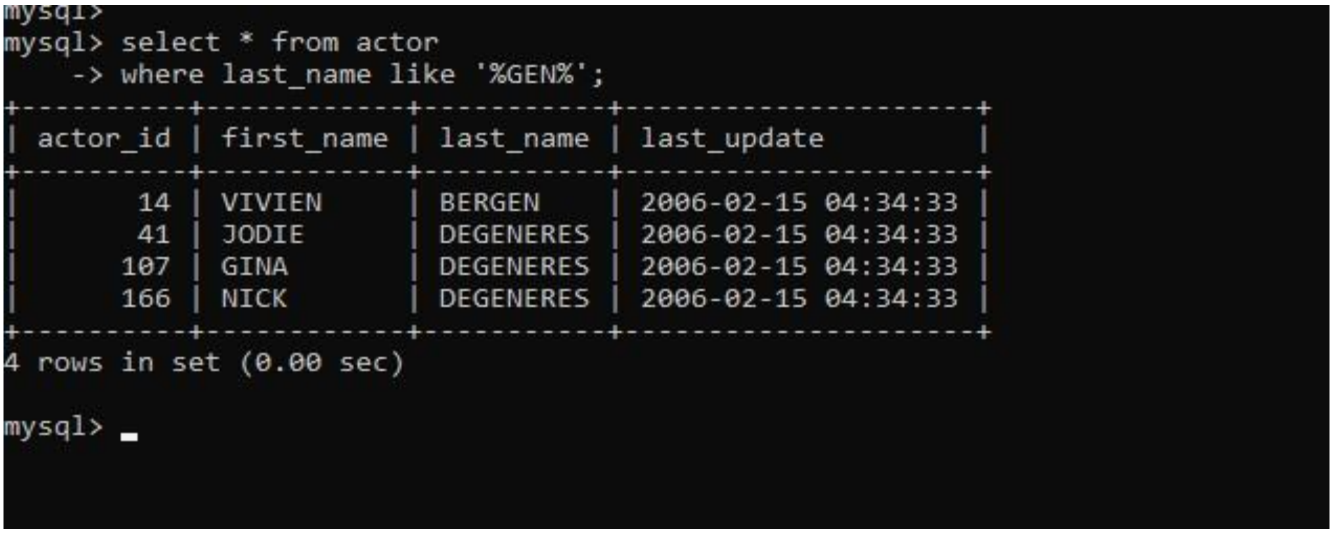
The Sakila database is a nicely normalised schema modelling a DVD rental store, featuring things like films, actors, film-actor relationships, and a central inventory table that connects films, stores, and rentals.

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



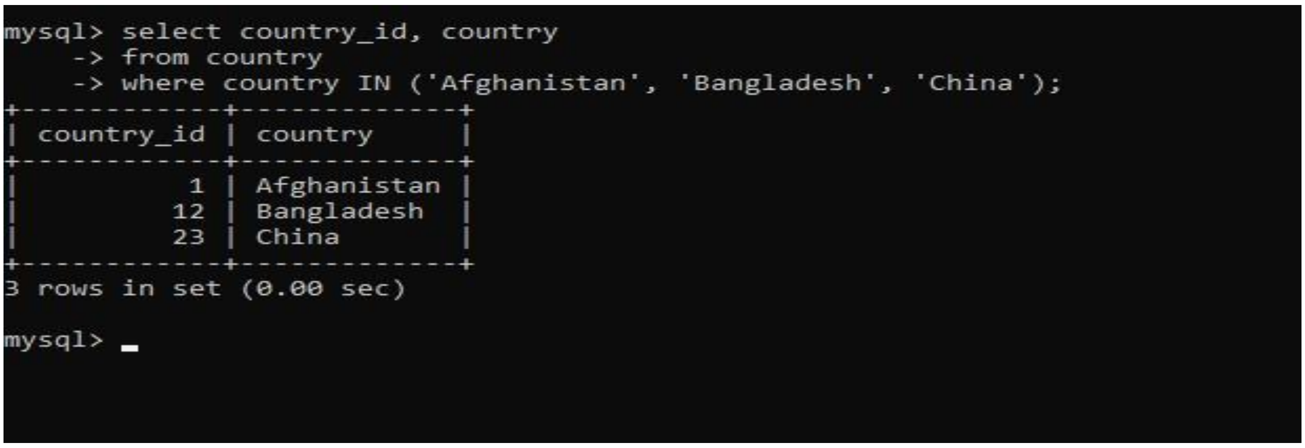
```
mysql>
mysql> select UPPER(concat(first_name,' ',last_name)) as Actor_Name
    -> from actor
    -> order by Actor_Name
    -> limit 20;
+--------------------+
| Actor_Name         |
+--------------------+
| ADAM GRANT         |
| ADAM HOPPER        |
| AL GARLAND         |
| ALAN DREYFUSS      |
| ALBERT JOHANSSON   |
| ALBERT NOLTE       |
| ALEC WAYNE         |
| ANGELA HUDSON      |
| ANGELA WITHERSPOON |
| ANGELINA ASTAIRE   |
| ANNE CRONYN        |
| AUDREY BAILEY      |
| AUDREY OLIVIER     |
| BELA WALKEN        |
| BEN HARRIS         |
| BEN WILLIS         |
| BETTE NICHOLSON    |
| BOB FAWCETT        |
| BURT DUKAKIS       |
| BURT POSEY         |
+--------------------+
20 rows in set (0.00 sec)

mysql>
```

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



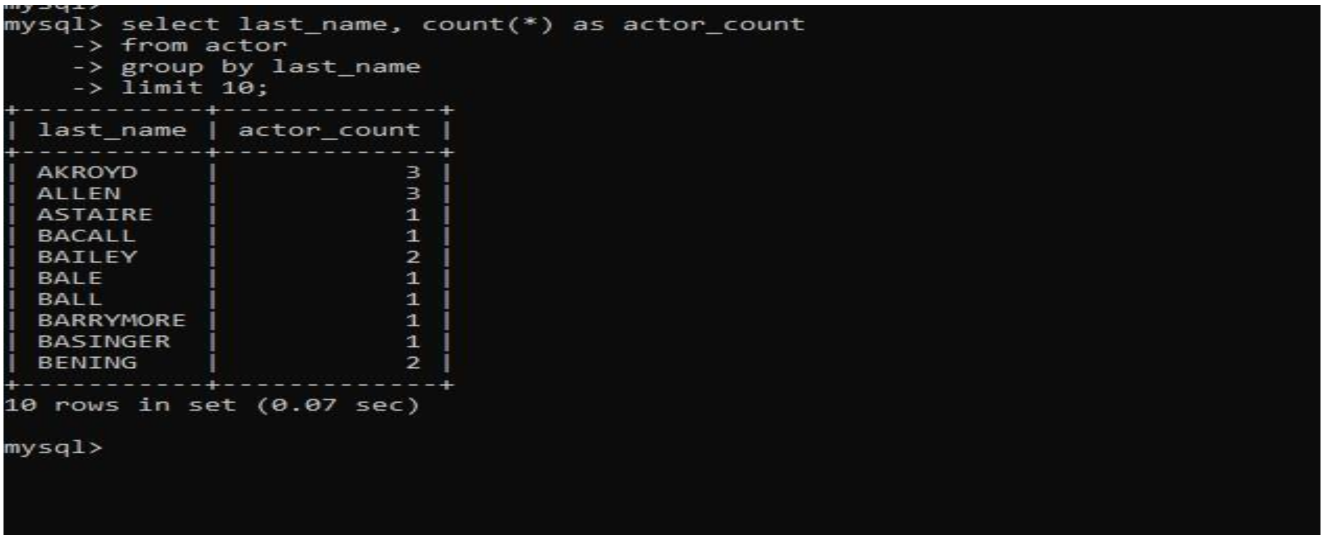
```
mysql>
mysql> select * from actor
    -> where last_name like '%GEN%';
+----------+------------+-----------+---------------------+
| actor_id | first_name | last_name | last_update         |
+----------+------------+-----------+---------------------+
|       14 | VIVIEN     | BERGEN    | 2006-02-15 04:34:33 |
|       41 | JODIE      | DEGENERES | 2006-02-15 04:34:33 |
|      107 | GINA       | DEGENERES | 2006-02-15 04:34:33 |
|      166 | NICK       | DEGENERES | 2006-02-15 04:34:33 |
+----------+------------+-----------+---------------------+
4 rows in set (0.00 sec)

mysql>
```

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



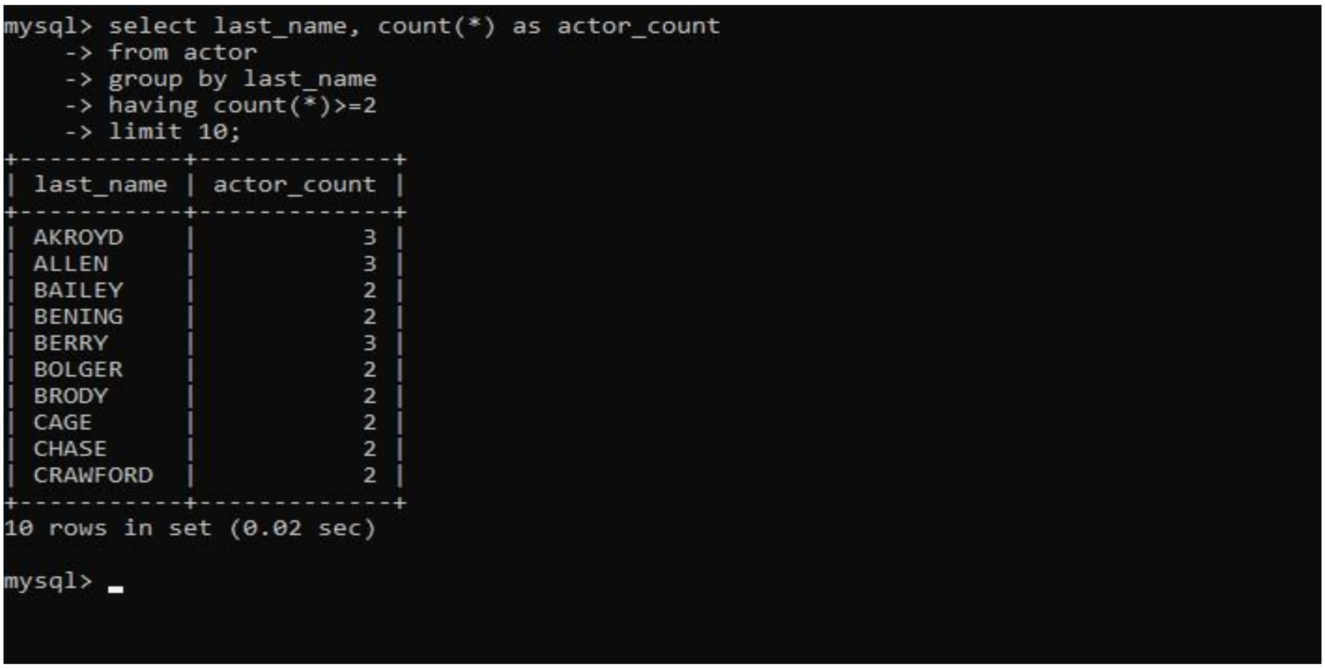
```
mysql> select country_id, country
    -> from country
    -> where country IN ('Afghanistan', 'Bangladesh', 'China');
+------------+-------------+
| country_id | country     |
+------------+-------------+
|          1 | Afghanistan |
|         12 | Bangladesh  |
|         23 | China       |
+------------+-------------+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

mysql> _
```

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



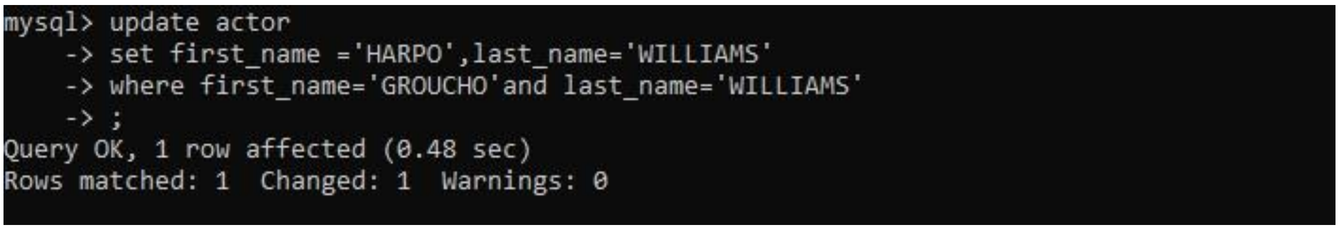
```
mysql> select last_name, count(*) as actor_count
    -> from actor
    -> group by last_name
    -> limit 10;
+-----------+-------------+
| last_name | actor_count |
+-----------+-------------+
| AKROYD    |           3 |
| ALLEN     |           3 |
| ASTAIRE   |           1 |
| BACALL    |           1 |
| BAILEY    |           2 |
| BALE      |           1 |
| BALL      |           1 |
| BARRYMORE |           1 |
| BASINGER  |           1 |
| BENING    |           2 |
+-----------+-------------+
10 rows in set (0.07 sec)

mysql>
```

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.**

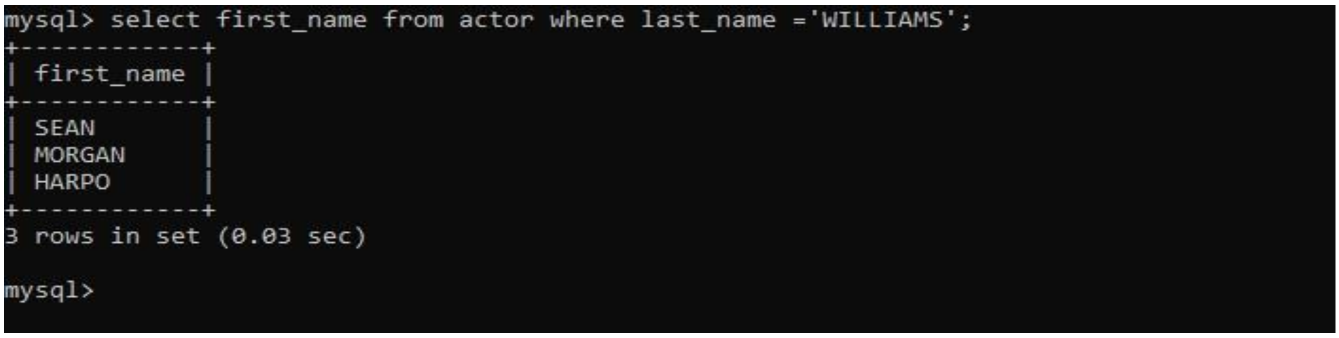


```
mysql> select last_name, count(*) as actor_count
    -> from actor
    -> group by last_name
    -> having count(*)>=2
    -> limit 10;
+-----------+-------------+
| last_name | actor_count |
+-----------+-------------+
| AKROYD    |           3 |
| ALLEN     |           3 |
| BAILEY    |           2 |
| BENING    |           2 |
| BERRY     |           3 |
| BOLGER    |           2 |
| BRODY     |           2 |
| CAGE      |           2 |
| CHASE     |           2 |
| CRAWFORD  |           2 |
+-----------+-------------+
10 rows in set (0.02 sec)

mysql> _
```

6. **The actor HARPO WILLIAMS was accidentally entered in the actor table as GROUCHO WILLIAMS. Write a query to fix the record.**

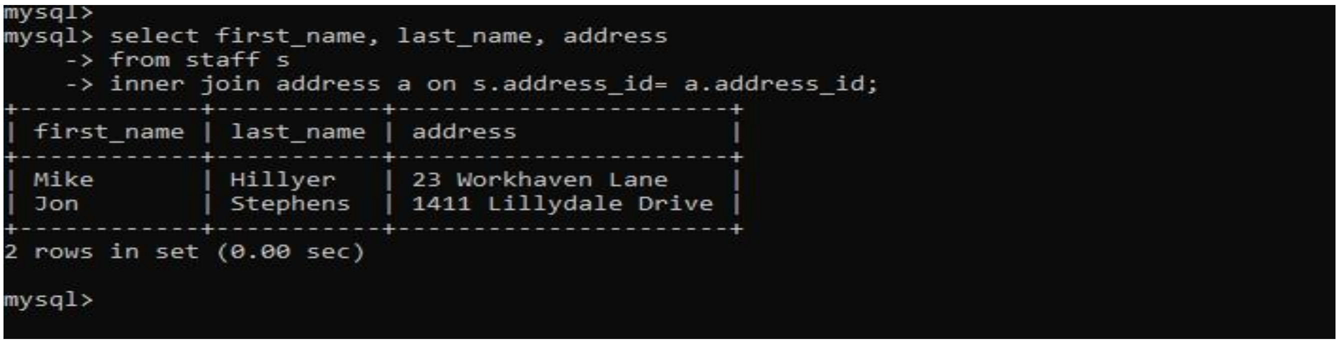


```
mysql> update actor
    -> set first_name ='HARPO',last_name='WILLIAMS'
    -> where first_name='GROUCHO'and last_name='WILLIAMS'
    -> ;
Query OK, 1 row affected (0.48 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```



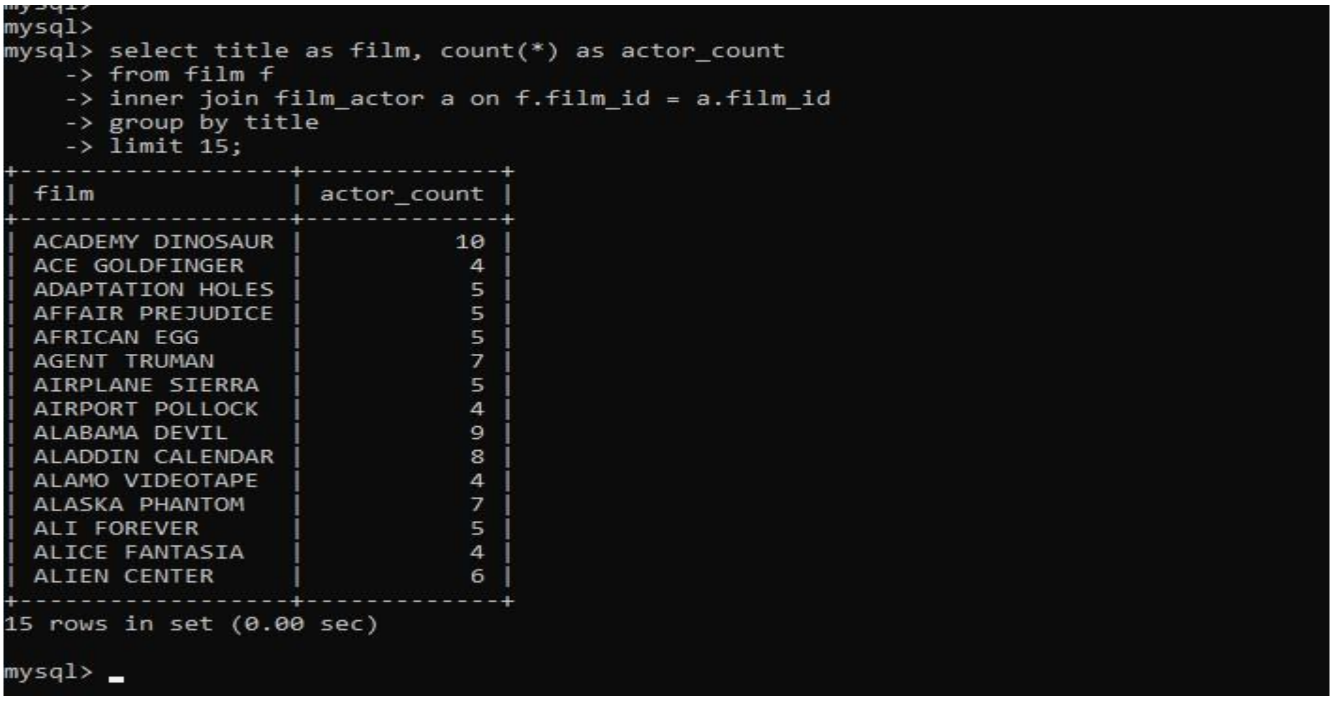
```
mysql> select first_name from actor where last_name ='WILLIAMS';
+------------+
| first_name |
+------------+
| SEAN       |
| MORGAN     |
| HARPO      |
+------------+
3 rows in set (0.03 sec)

mysql>
```

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:**



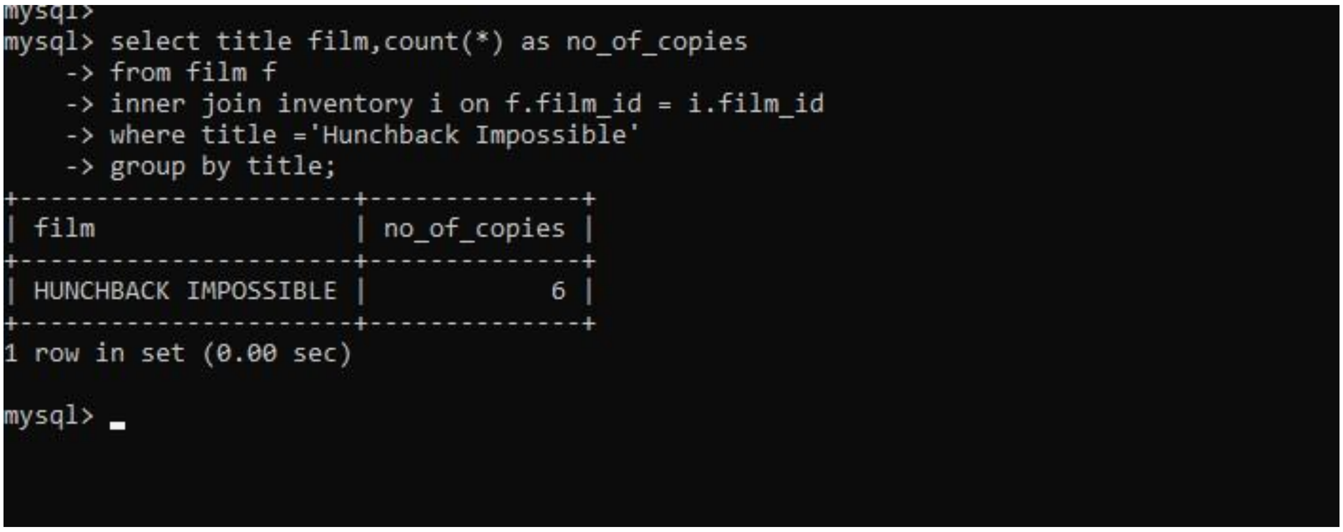
```
mysql>
mysql> select first_name, last_name, address
    -> from staff s
    -> inner join address a on s.address_id= a.address_id;
+------------+-----------+----------------------+
| first_name | last_name | address              |
+------------+-----------+----------------------+
| Mike       | Hillyer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

mysql>
```

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



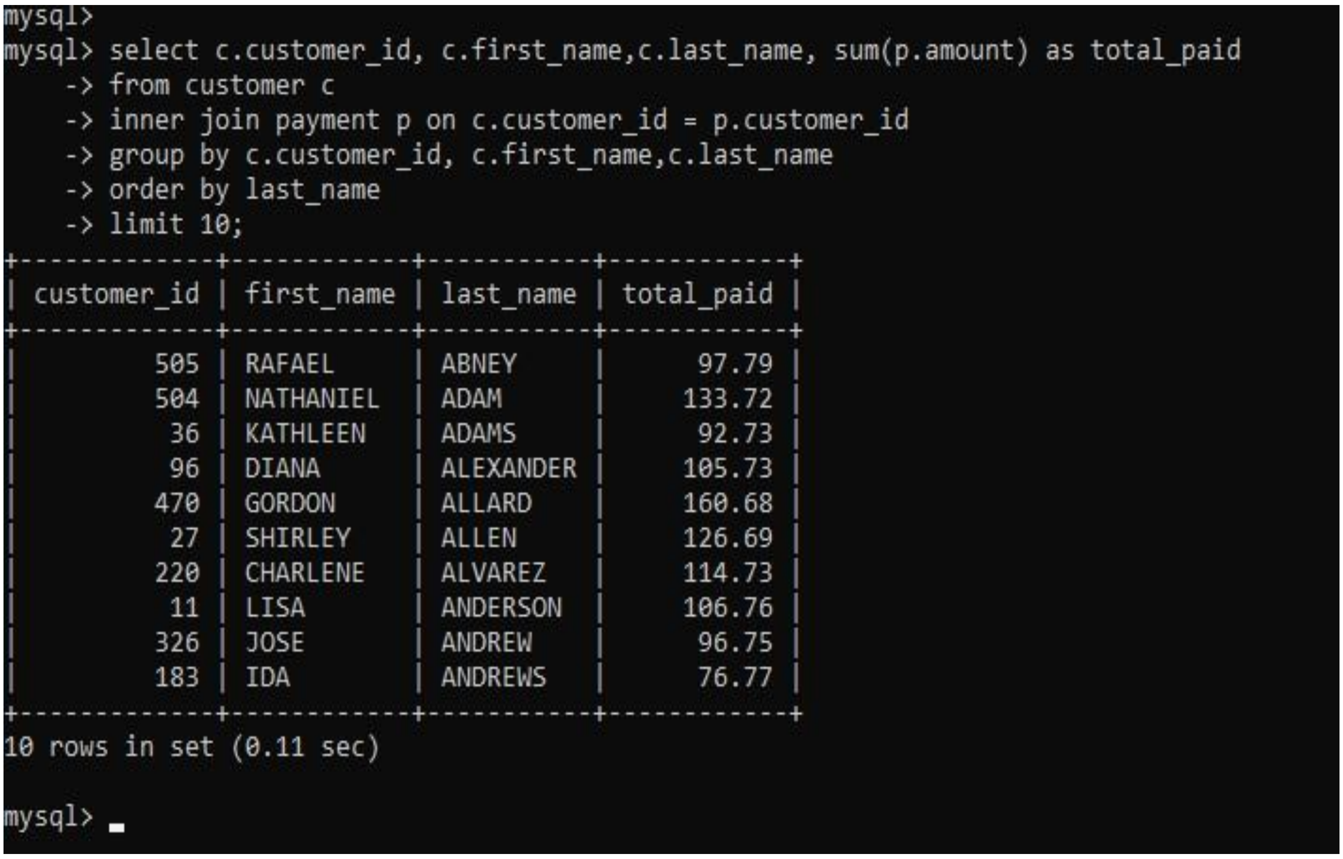
```
mysql>
mysql> select title as film, count(*) as actor_count
    -> from film f
    -> inner join film_actor a on f.film_id = a.film_id
    -> group by title
    -> limit 15;
+------------------+-------------+
| film             | actor_count |
+------------------+-------------+
| ACADEMY DINOSAUR |          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 |
+------------------+-------------+
15 rows in set (0.00 sec)

mysql> _
```

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



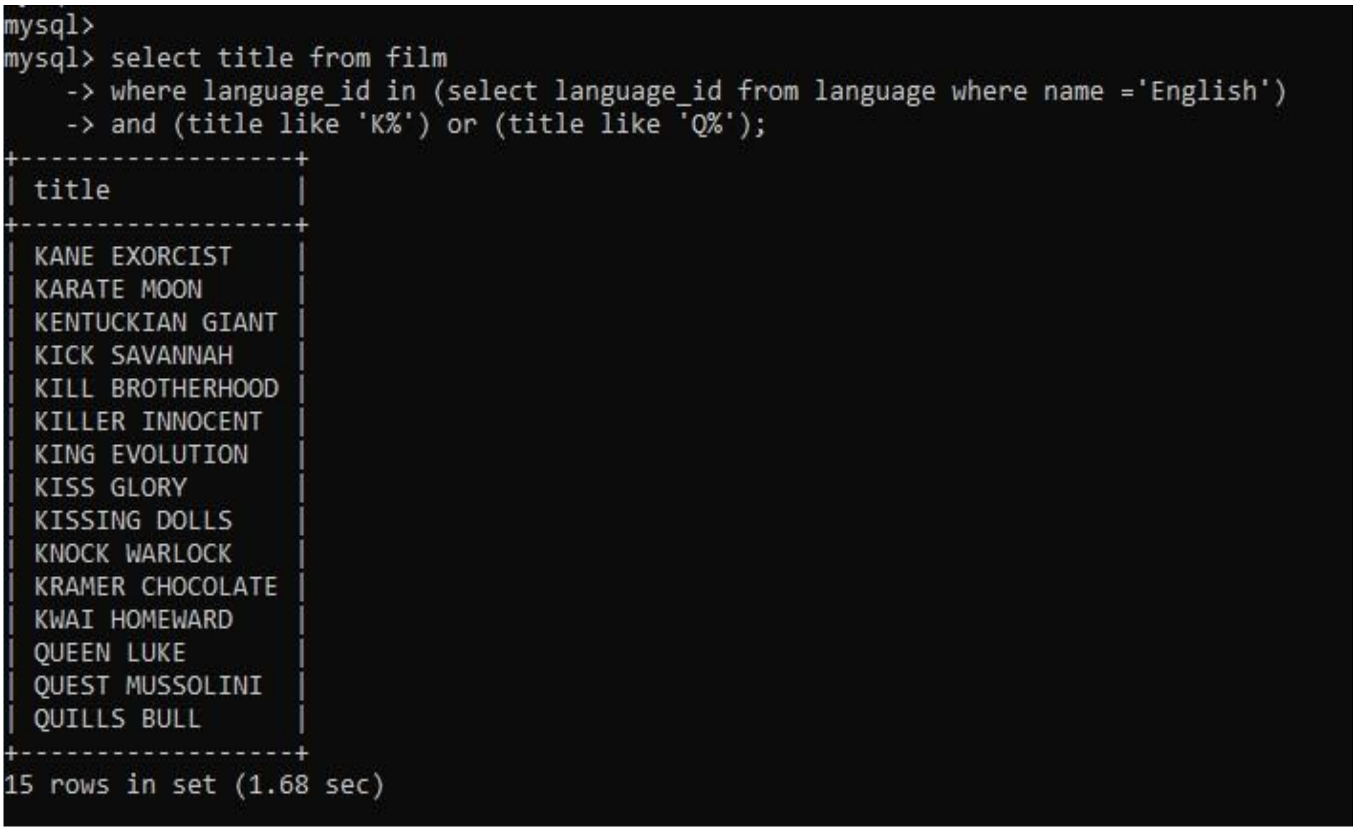
```
mysql>
mysql> select title film,count(*) as no_of_copies
    -> from film f
    -> inner join inventory i on f.film_id = i.film_id
    -> where title ='Hunchback Impossible'
    -> group by title;
+----------------------+--------------+
| film                 | no_of_copies |
+----------------------+--------------+
| HUNCHBACK IMPOSSIBLE |            6 |
+----------------------+--------------+
1 row in set (0.00 sec)

mysql> _
```

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.**



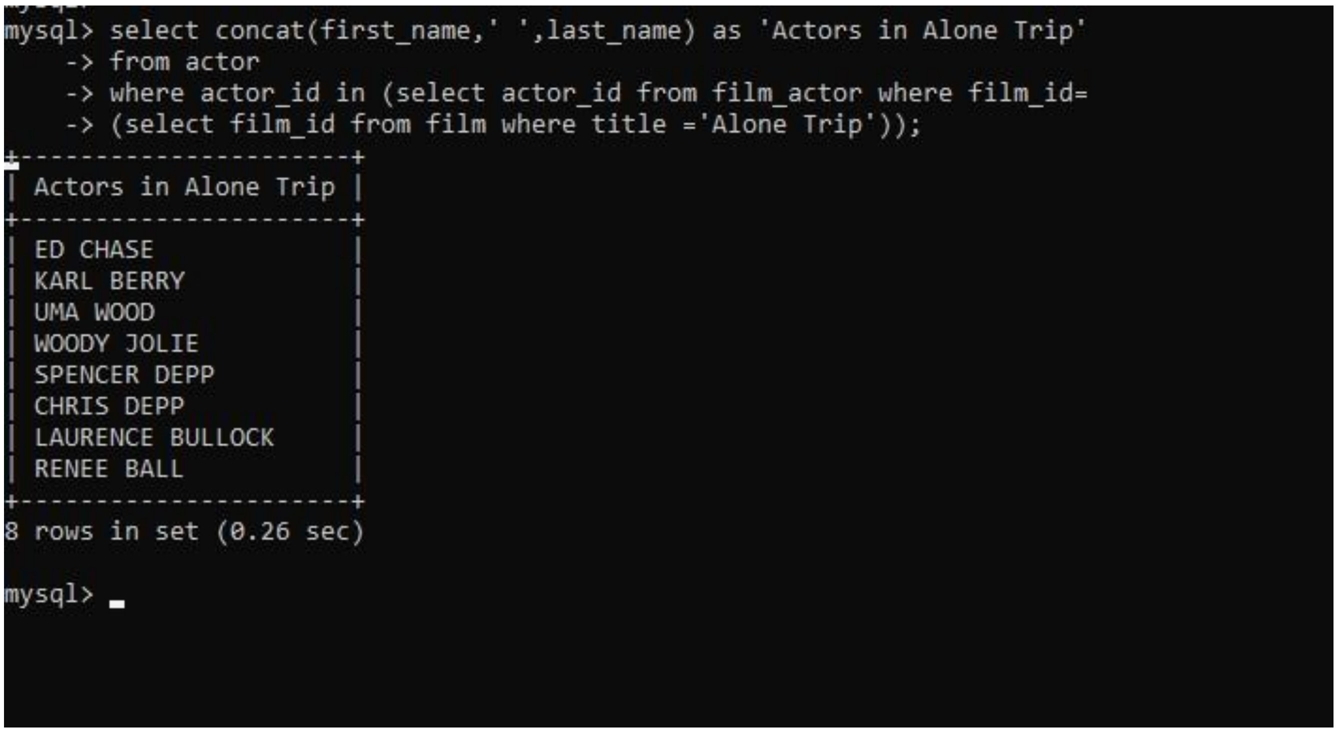
```
mysql>
mysql> select c.customer_id, c.first_name,c.last_name, sum(p.amount) as total_paid
    -> from customer c
    -> inner join payment p on c.customer_id = p.customer_id
    -> group by c.customer_id, c.first_name,c.last_name
    -> order by last_name
    -> limit 10;
+-------------+------------+-----------+------------+
| customer_id | first_name | last_name | total_paid |
+-------------+------------+-----------+------------+
|         505 | RAFAEL     | ABNEY     |      97.79 |
|         504 | NATHANIEL  | ADAM      |     133.72 |
|          36 | KATHLEEN   | ADAMS     |      92.73 |
|          96 | DIANA      | ALEXANDER |     105.73 |
|         470 | GORDON     | ALLARD    |     160.68 |
|          27 | SHIRLEY    | ALLEN     |     126.69 |
|         220 | CHARLENE   | ALVAREZ   |     114.73 |
|          11 | LISA       | ANDERSON  |     106.76 |
|         326 | JOSE       | ANDREW    |      96.75 |
|         183 | IDA        | ANDREWS   |      76.77 |
+-------------+------------+-----------+------------+
10 rows in set (0.11 sec)

mysql> _
```

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**



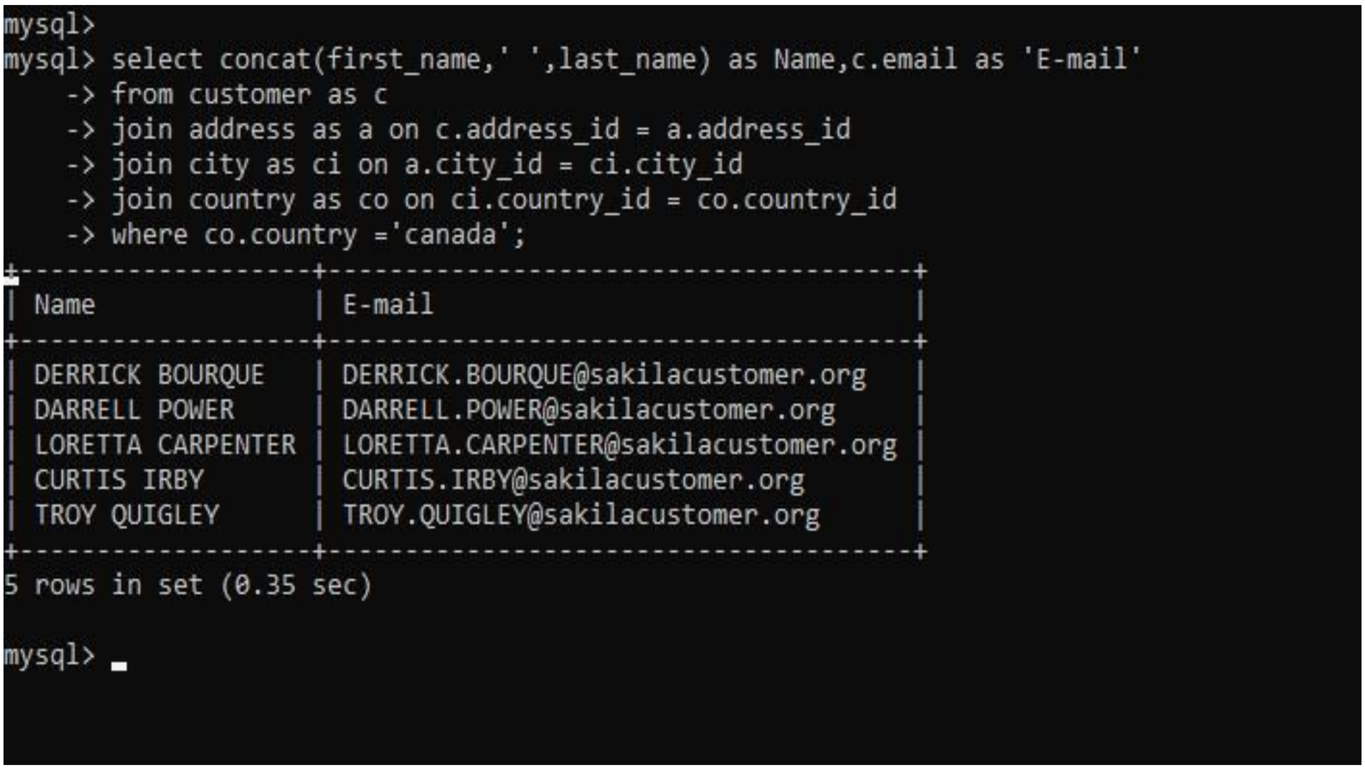
```
mysql>
mysql> select title from film
    -> where language_id in (select language_id from language where name ='English')
    -> and (title like 'K%') or (title like 'Q%');
+------------------+
| title            |
+------------------+
| KANE EXORCIST    |
| KARATE MOON      |
| KENTUCKIAN GIANT |
| 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 (1.68 sec)
```

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

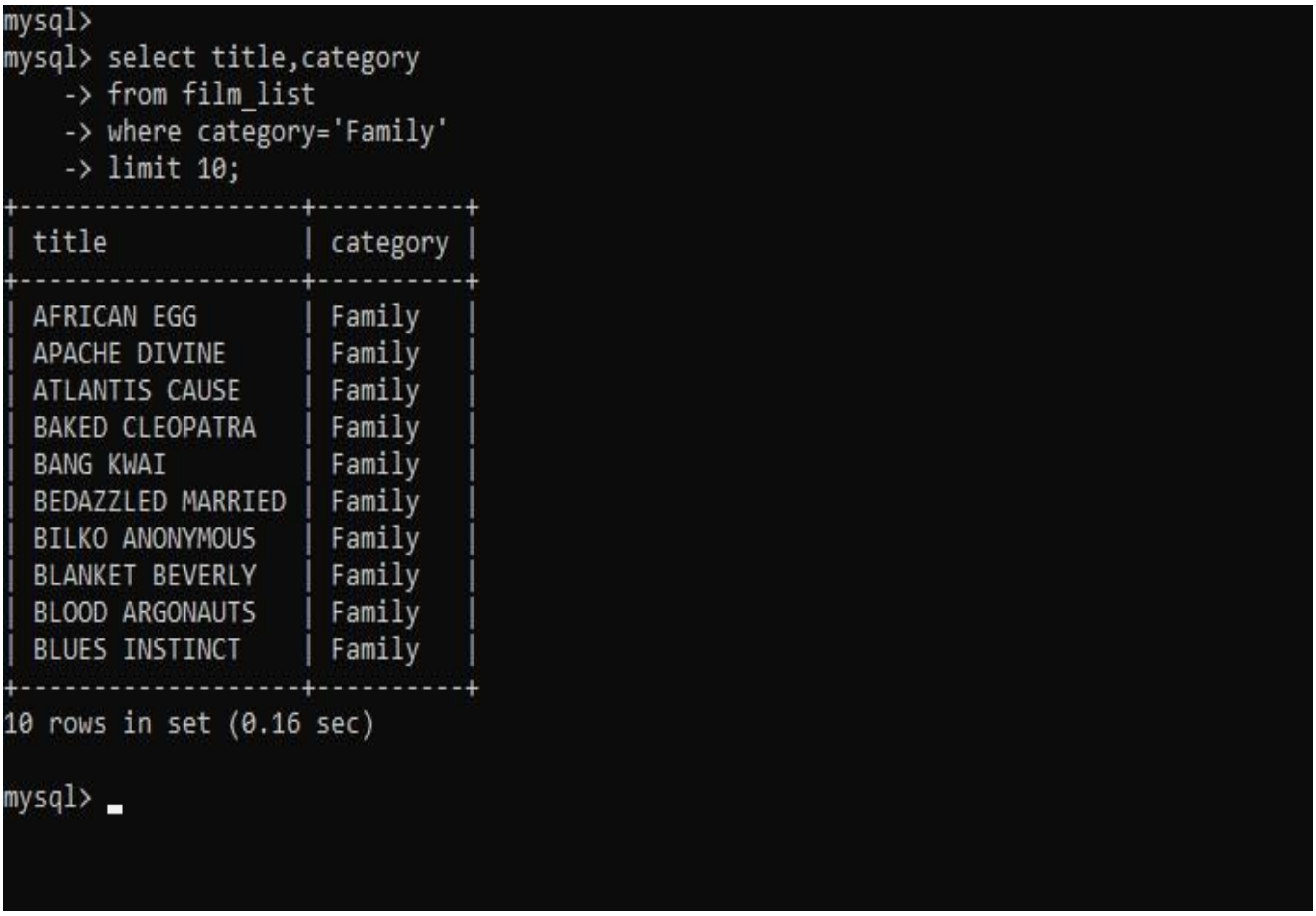


```
mysql> select concat(first_name,' ',last_name) as 'Actors in Alone Trip'
    -> from actor
    -> where actor_id in (select actor_id from film_actor where film_id=
    -> (select film_id from film where title ='Alone Trip'));
+---------------------+
| Actors in Alone Trip |
+---------------------+
| ED CHASE             |
| KARL BERRY           |
| UMA WOOD             |
| WOODY JOLIE          |
| SPENCER DEPP         |
| CHRIS DEPP           |
| LAURENCE BULLOCK     |
| RENEE BALL           |
+---------------------+
8 rows in set (0.26 sec)

mysql>
```

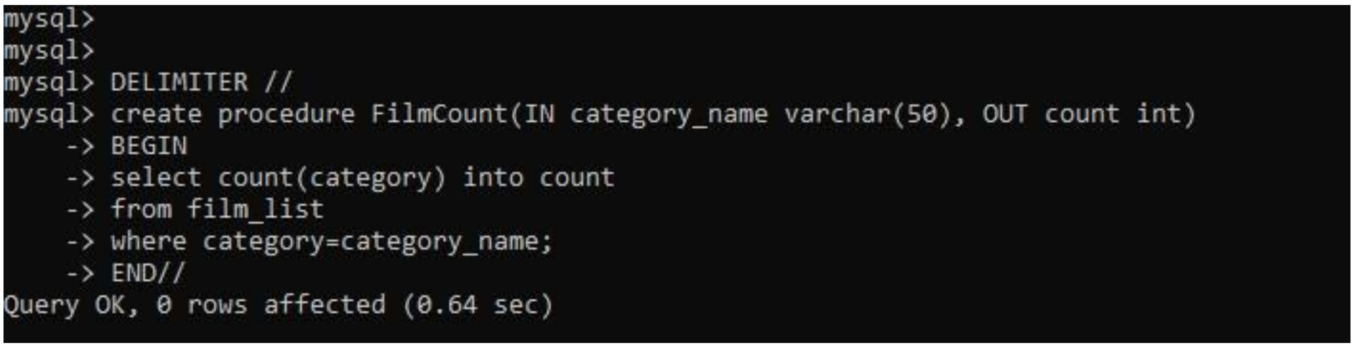
13. You want to run an email marketing campaign in Canada, for which you will need the names and email addresses of all Canadian customers. Use joins to retrieve this information.


```
mysql>
mysql> select concat(first_name,' ',last_name) as Name,c.email as 'E-mail'
    -> from customer as c
    -> join address as a on c.address_id = a.address_id
    -> join city as ci on a.city_id = ci.city_id
    -> join country as co on ci.country_id = co.country_id
    -> where co.country ='canada';
+-------------------+-------------------------------------+
| Name              | E-mail                              |
+-------------------+-------------------------------------+
| DERRICK BOURQUE   | DERRICK.BOURQUE@sakilacustomer.org  |
| DARRELL POWER     | DARRELL.POWER@sakilacustomer.org    |
| LORETTA CARPENTER | LORETTA.CARPENTER@sakilacustomer.org |
| CURTIS IRBY       | CURTIS.IRBY@sakilacustomer.org      |
| TROY QUIGLEY      | TROY.QUIGLEY@sakilacustomer.org     |
+-------------------+-------------------------------------+
5 rows in set (0.35 sec)

mysql>
```

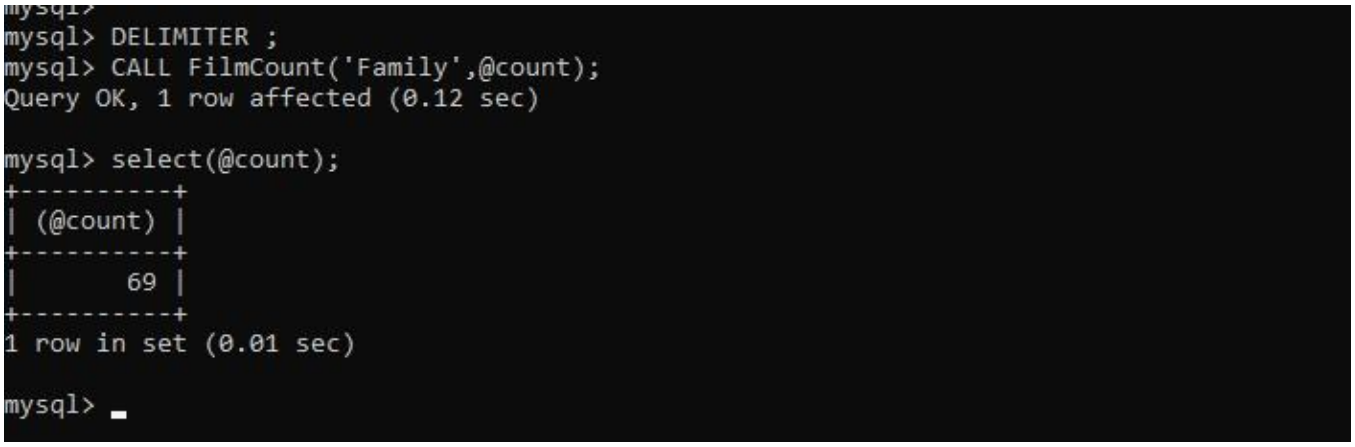
14. **Sales have been lagging among young families, and you wish to target all family movies for a promotion. Identify all movies categorized as family films.**


```
mysql>
mysql> select title,category
    -> from film_list
    -> where category='Family'
    -> limit 10;
+------------------+----------+
| title            | category |
+------------------+----------+
| AFRICAN EGG      | Family   |
| APACHE DIVINE    | Family   |
| ATLANTIS CAUSE   | Family   |
| BAKED CLEOPATRA  | Family   |
| BANG KWAI        | Family   |
| BEDAZZLED MARRIED | Family  |
| BILKO ANONYMOUS  | Family   |
| BLANKET BEVERLY  | Family   |
| BLOOD ARGONAUTS  | Family   |
| BLUES INSTINCT   | Family   |
+------------------+----------+
10 rows in set (0.16 sec)

mysql>
```

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

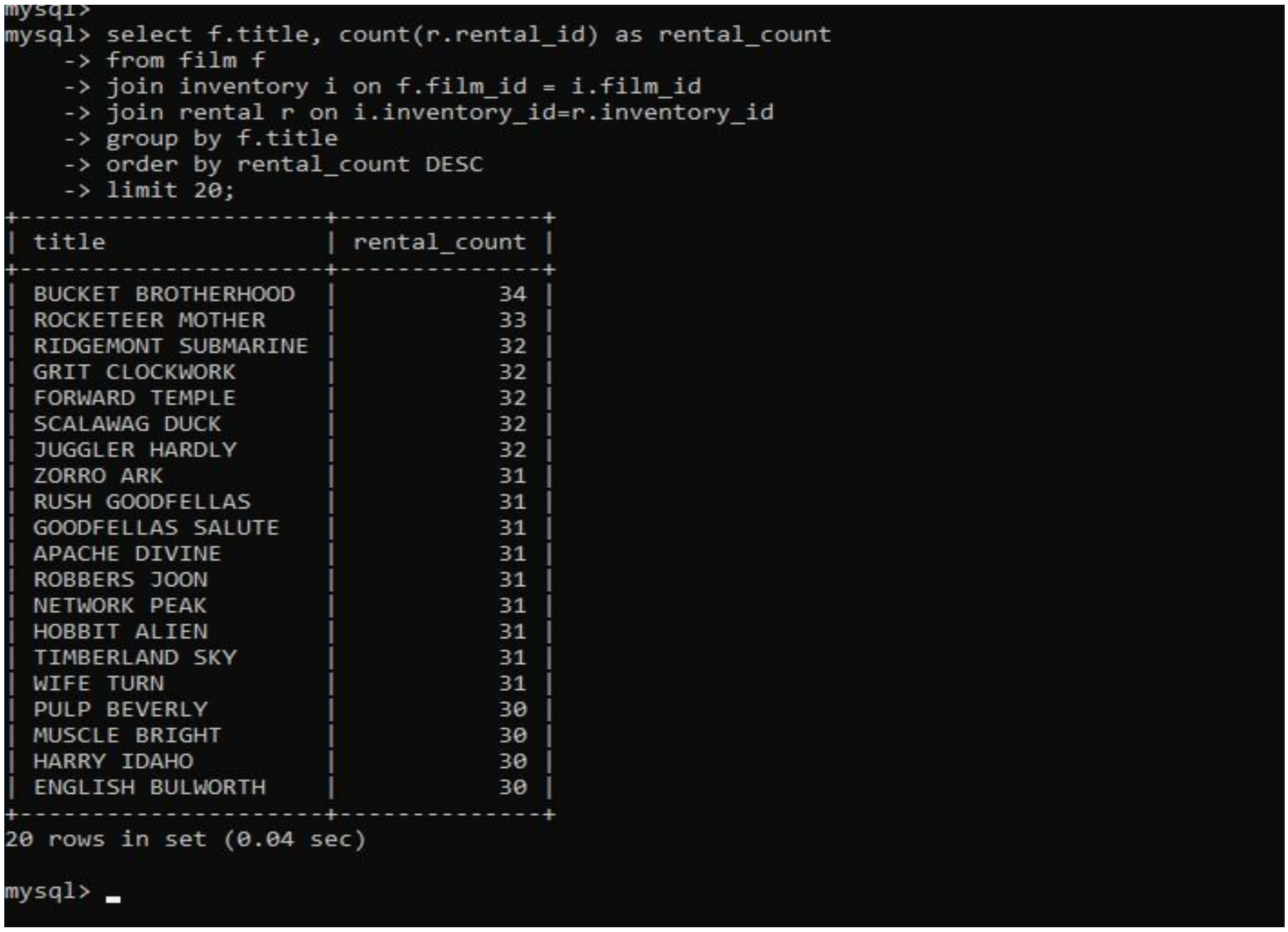


```
mysql>
mysql>
mysql> DELIMITER //
mysql> create procedure FilmCount(IN category_name varchar(50), OUT count int)
    -> BEGIN
    -> select count(category) into count
    -> from film_list
    -> where category=category_name;
    -> END//
Query OK, 0 rows affected (0.64 sec)
```



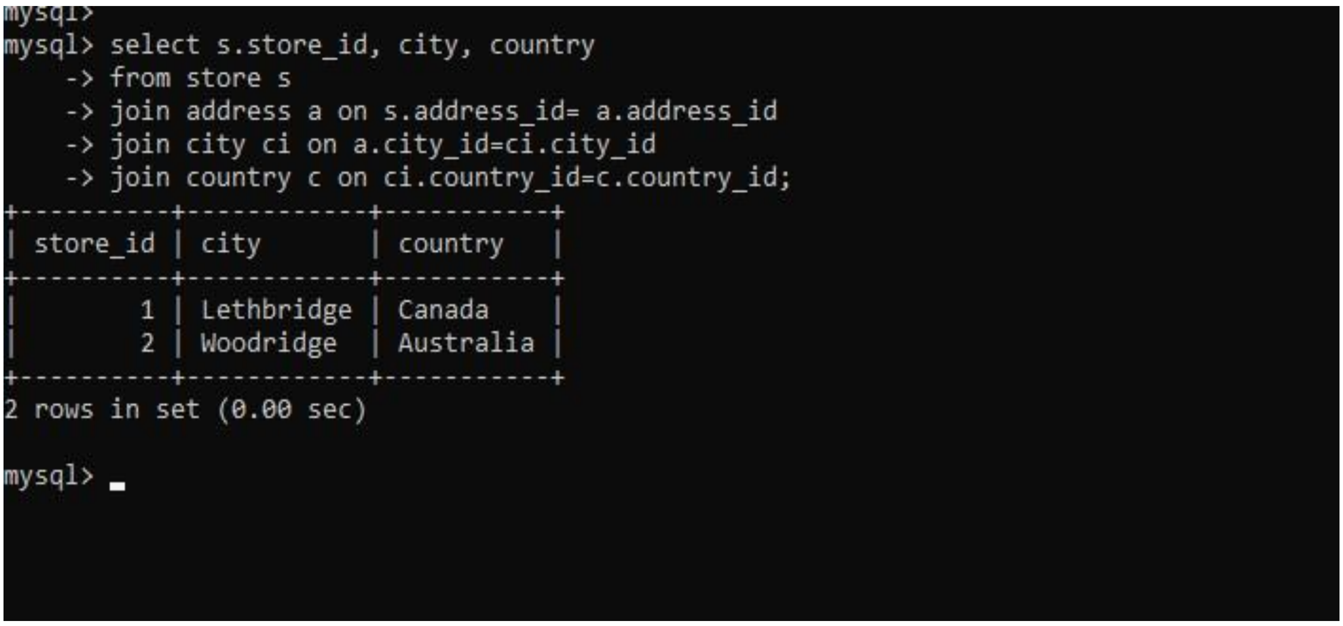
```
mysql>
mysql> DELIMITER ;
mysql> CALL FilmCount('Family',@count);
Query OK, 1 row affected (0.12 sec)

mysql> select(@count);
+----------+
| (@count) |
+----------+
|       69 |
+----------+
1 row in set (0.01 sec)

mysql> _
```

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



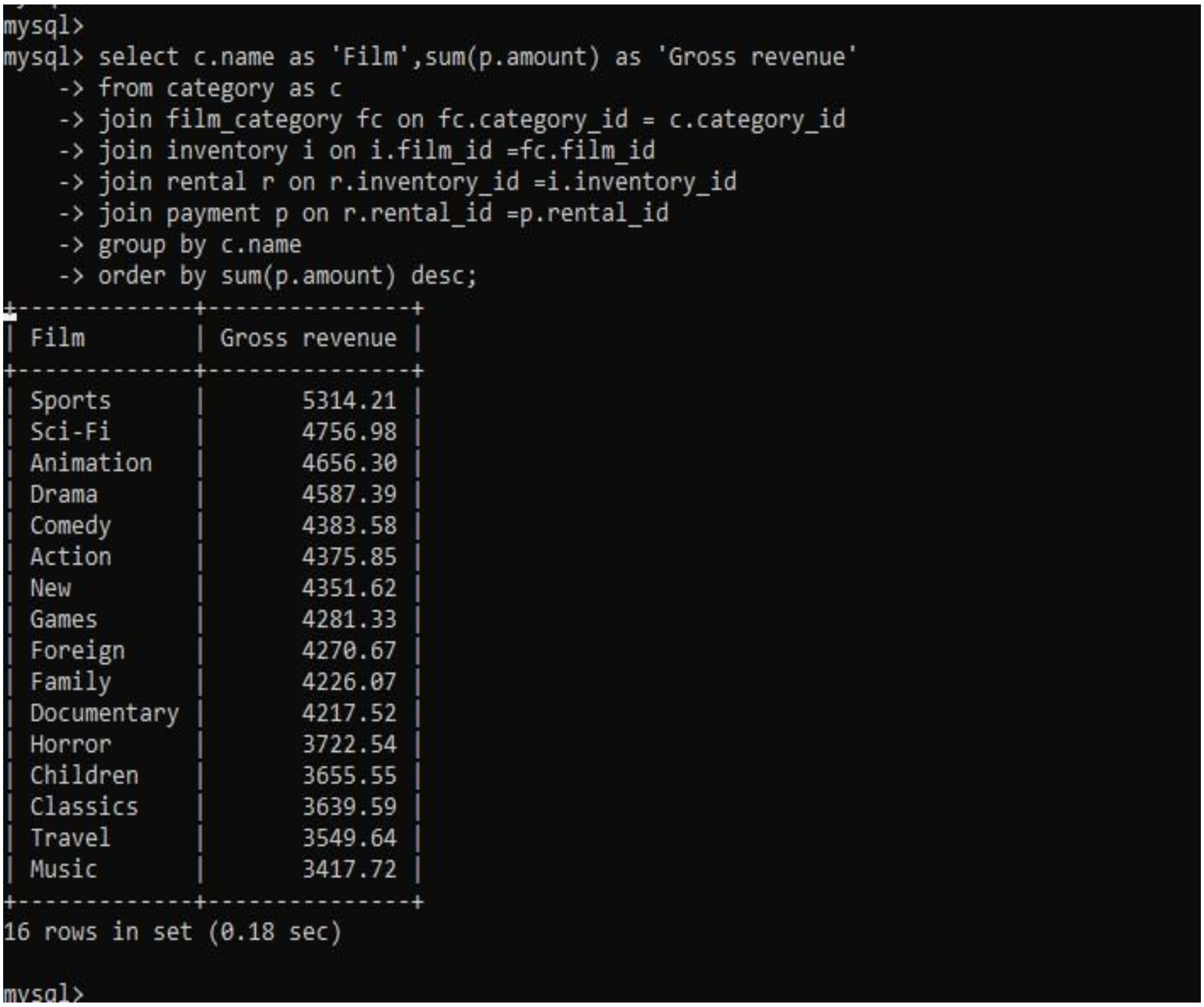
```
mysql>
mysql> select f.title, count(r.rental_id) as rental_count
    -> from film f
    -> join inventory i on f.film_id = i.film_id
    -> join rental r on i.inventory_id=r.inventory_id
    -> group by f.title
    -> order by rental_count DESC
    -> limit 20;
+--------------------+--------------+
| title              | 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 (0.04 sec)

mysql> _
```

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



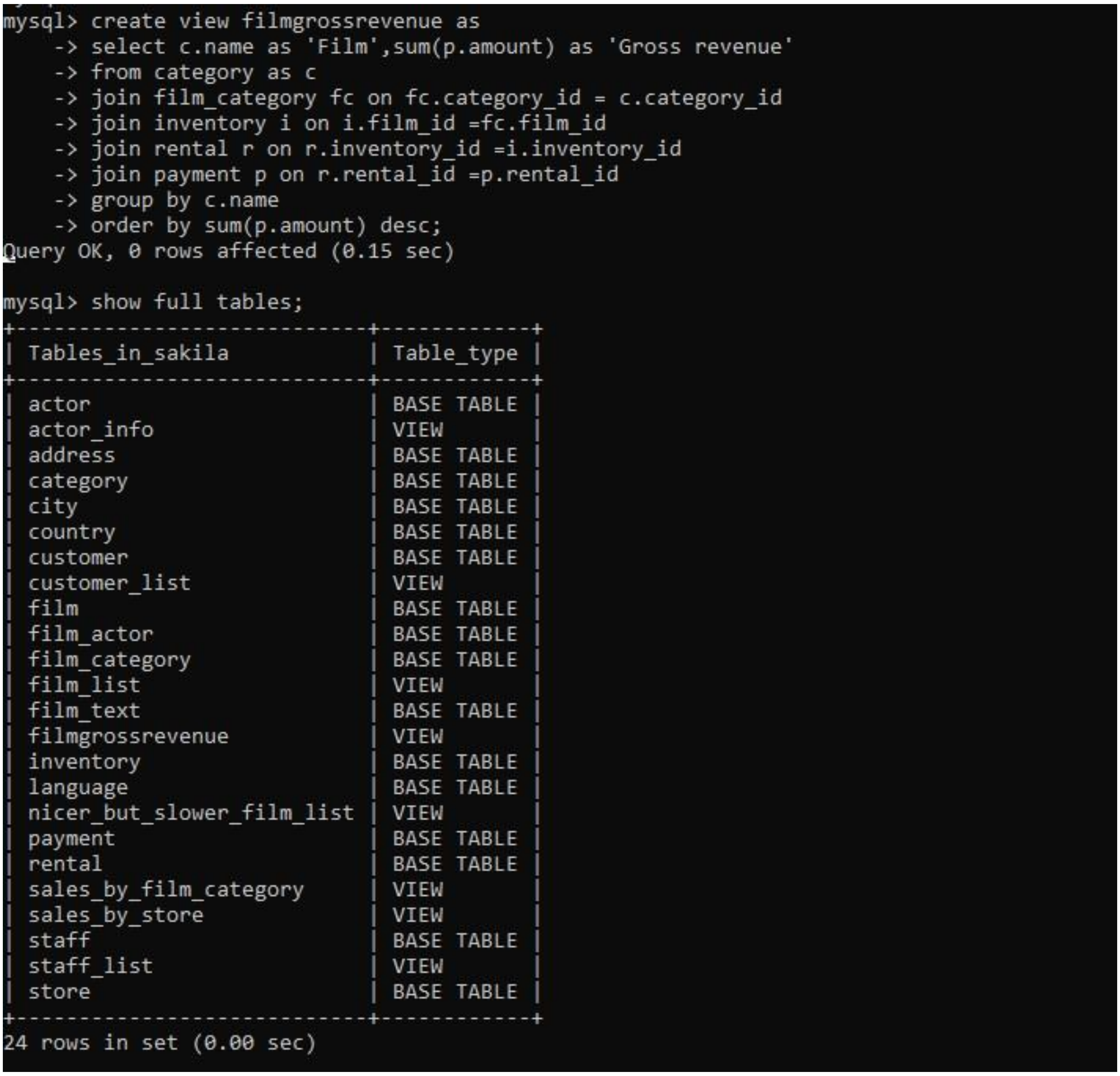
```
mysql>
mysql> select s.store_id, city, country
    -> from store s
    -> join address a on s.address_id= a.address_id
    -> join city ci on a.city_id=ci.city_id
    -> join country c on ci.country_id=c.country_id;
+----------+------------+-----------+
| store_id | city       | country   |
+----------+------------+-----------+
|        1 | Lethbridge | Canada    |
|        2 | Woodridge  | Australia |
+----------+------------+-----------+
2 rows in set (0.00 sec)

mysql>
```

18. List the genres and its gross revenue.



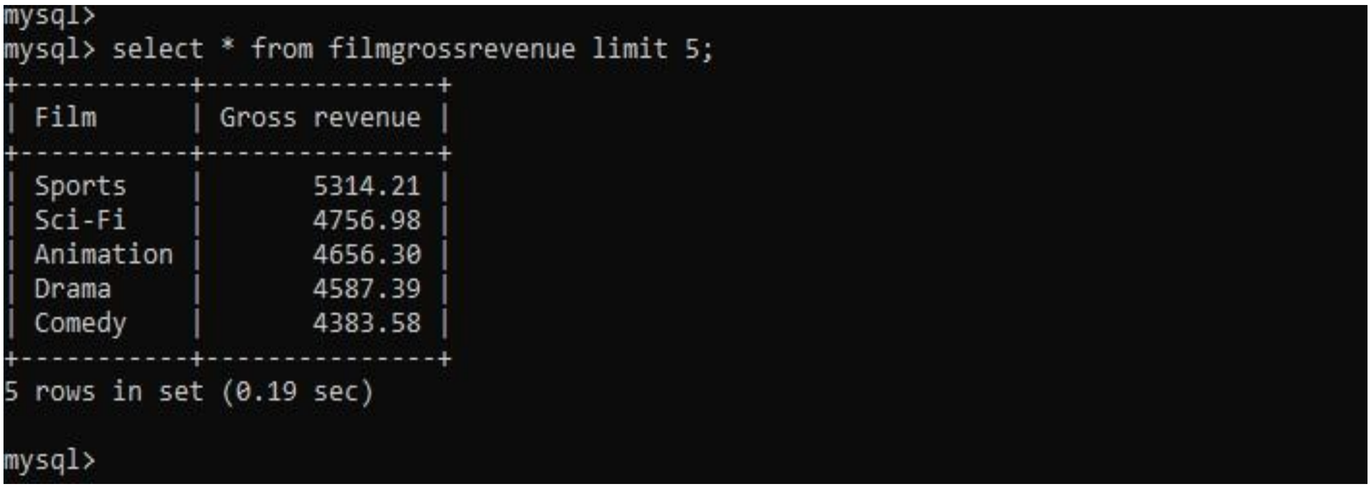
```
mysql>
mysql> select c.name as 'Film',sum(p.amount) as 'Gross revenue'
    -> from category as c
    -> join film_category fc on fc.category_id = c.category_id
    -> join inventory i on i.film_id =fc.film_id
    -> join rental r on r.inventory_id =i.inventory_id
    -> join payment p on r.rental_id =p.rental_id
    -> group by c.name
    -> order by sum(p.amount) desc;
+-------------+---------------+
| Film        | 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 (0.18 sec)

mysql>
```

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



```
mysql> create view filmgrossrevenue as
    -> select c.name as 'Film',sum(p.amount) as 'Gross revenue'
    -> from category as c
    -> join film_category fc on fc.category_id = c.category_id
    -> join inventory i on i.film_id =fc.film_id
    -> join rental r on r.inventory_id =i.inventory_id
    -> join payment p on r.rental_id =p.rental_id
    -> group by c.name
    -> order by sum(p.amount) desc;
Query OK, 0 rows affected (0.15 sec)

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 |
| filmgrossrevenue           | VIEW       |
| 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 |
+----------------------------+------------+
24 rows in set (0.00 sec)
```

20. Select top 5 genres in gross revenue view.



```
mysql>
mysql> select * from filmgrossrevenue limit 5;
+-----------+---------------+
| Film      | Gross revenue |
+-----------+---------------+
| Sports    |       5314.21 |
| Sci-Fi    |       4756.98 |
| Animation |       4656.30 |
| Drama     |       4587.39 |
| Comedy    |       4383.58 |
+-----------+---------------+
5 rows in set (0.19 sec)

mysql>
```