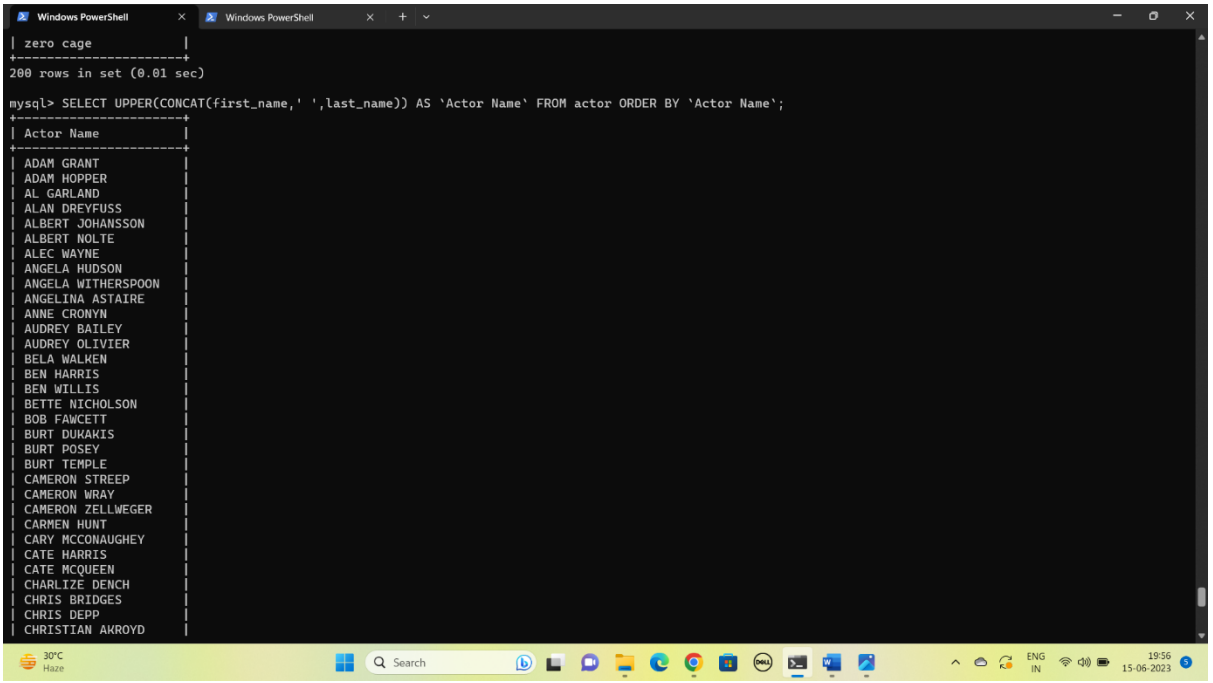
ASSIGNMENT 1

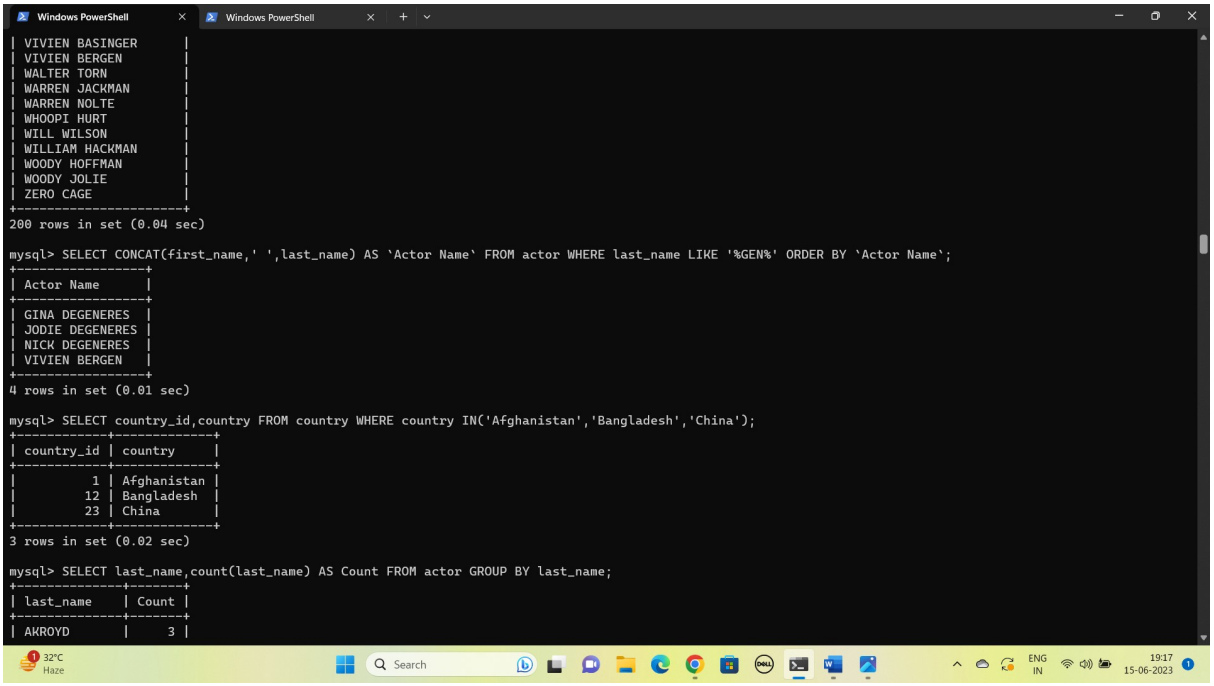
1. Display the first and last name of each actor in a single column in upper case letters in alphabetic order. Name the column Actor Name.

```
SELECT UPPER(CONCAT(first_name,' ',last_name)) AS `Actor Name` FROM actor ORDER BY `Actor Name`;
```



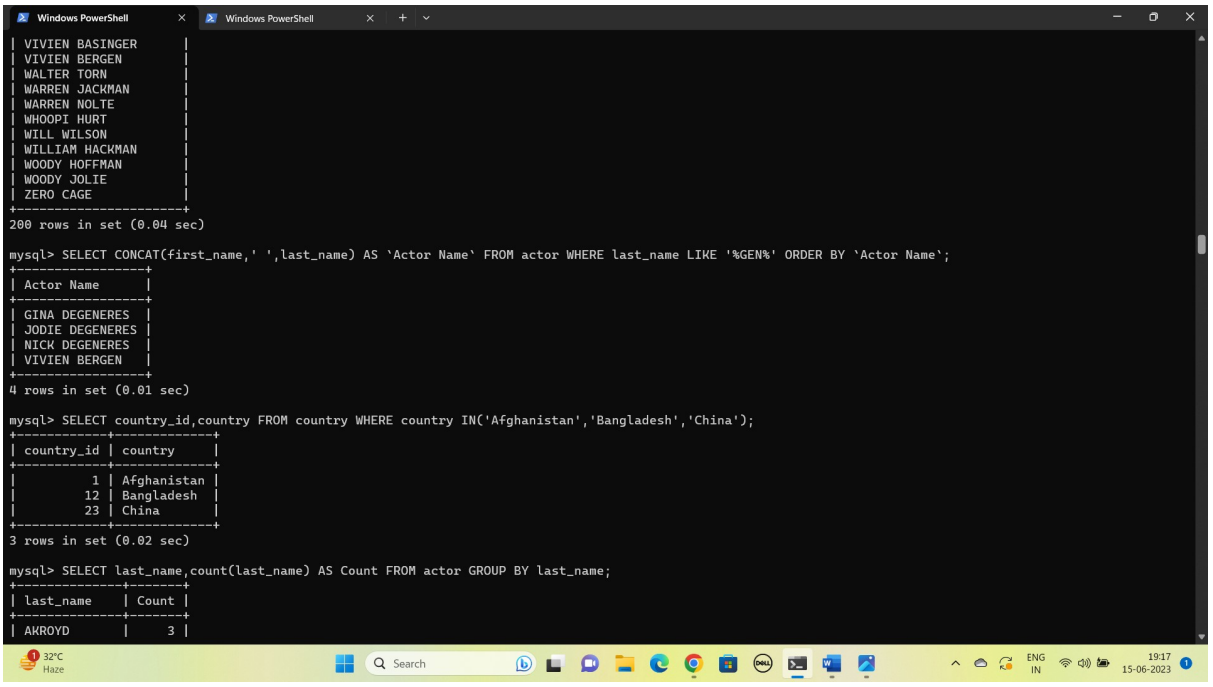
2. Find all actors whose last name contain the letters GEN:

```
SELECT CONCAT(first_name,' ',last_name) AS `Actor Name` FROM actor WHERE last_name LIKE '%GEN%' ORDER BY `Actor Name`;
```



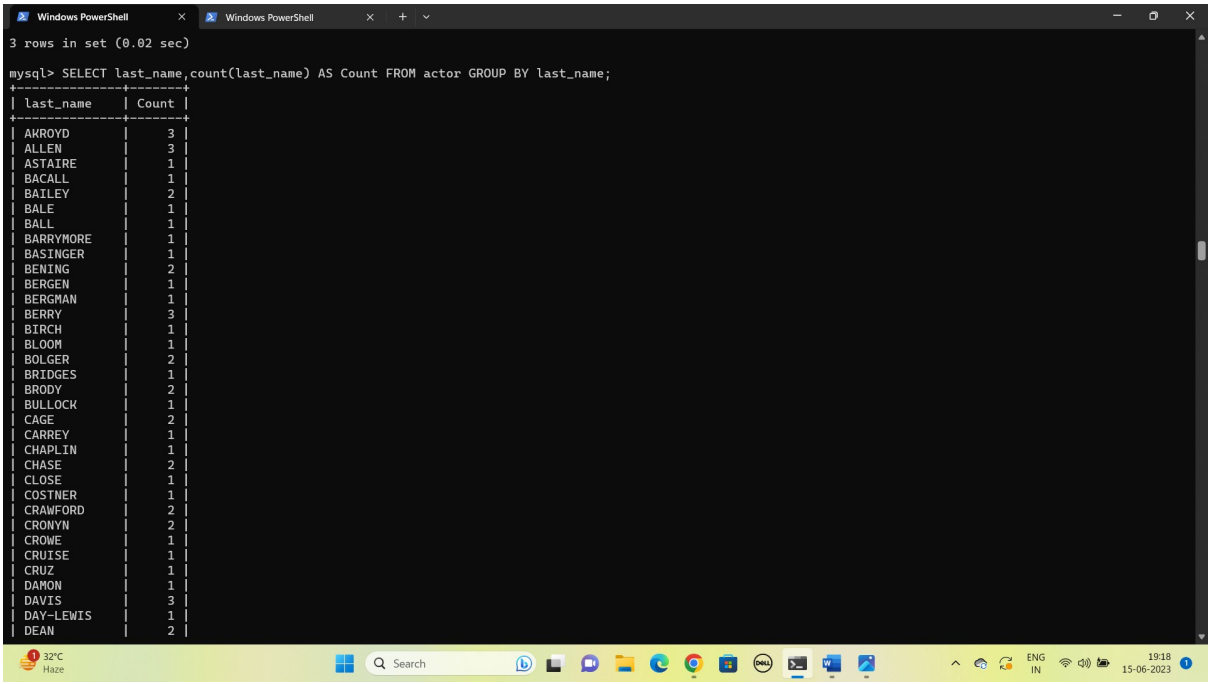
3. Using IN, display the country_id and country columns of the following countries: Afghanistan, Bangladesh, and China:

```
SELECT country_id,country FROM country WHERE country
IN('Afghanistan','Bangladesh','China');
```



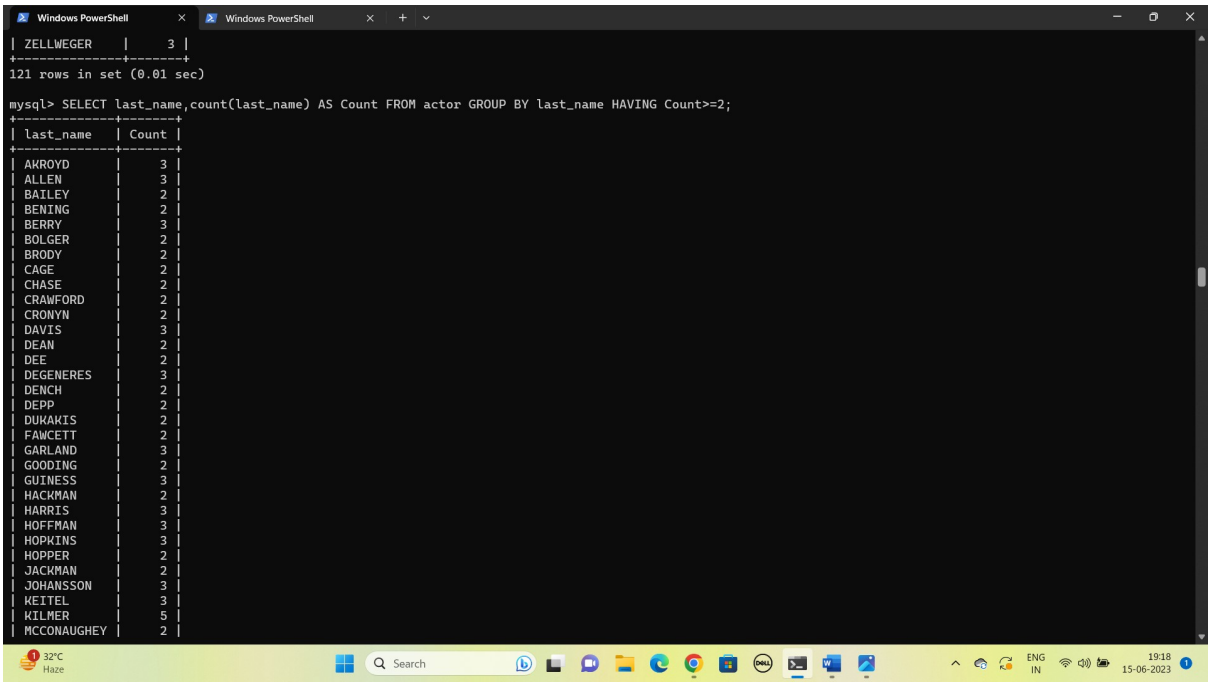
4. List the last names of actors, as well as how many actors have that last name.

```
SELECT last_name,count(last_name) AS Count FROM actor GROUP BY last_name;
```



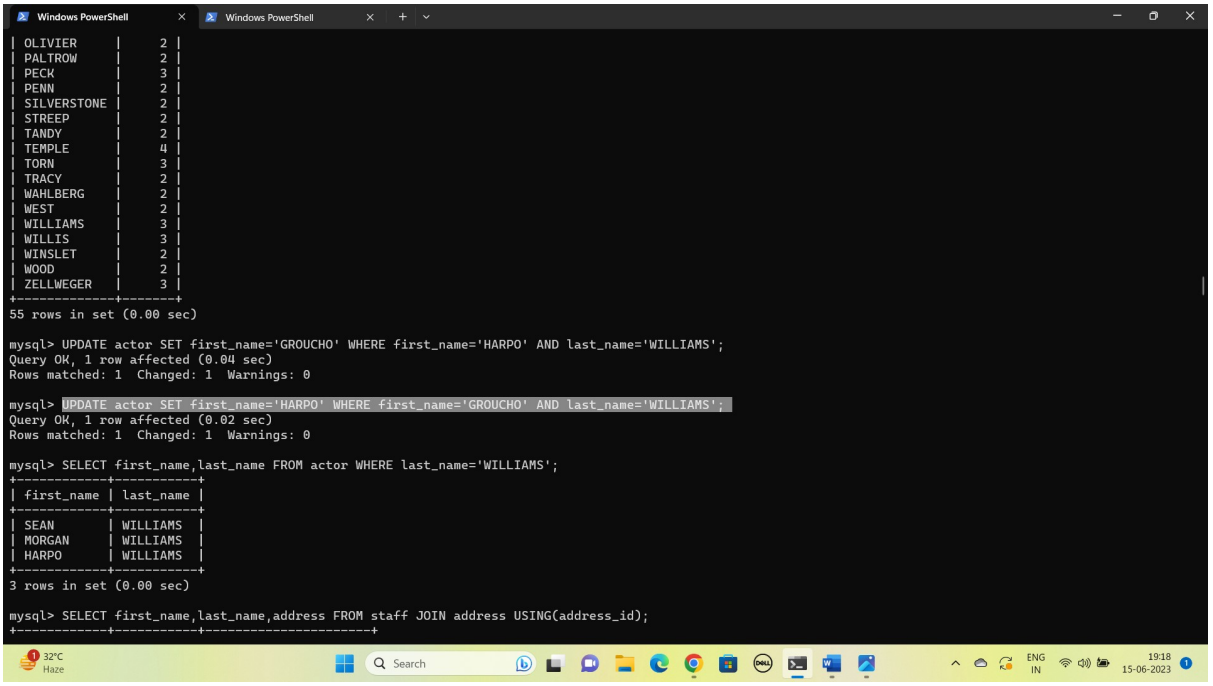
5. List last names of actors and the number of actors who have that last name, but only for names that are shared by at least two actors

SELECT last_name,count(last_name) AS Count FROM actor GROUP BY last_name HAVING Count>=2;



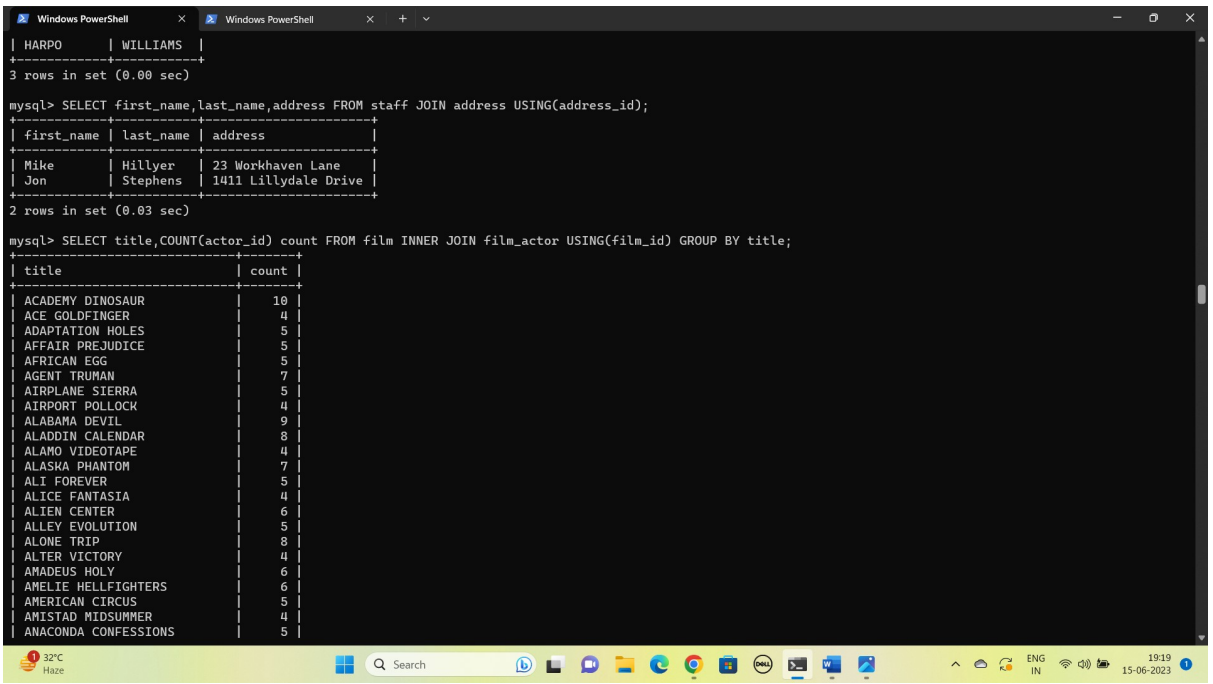
6. The actor HARPO WILLIAMS was accidentally entered in the actor table as GROUCHO WILLIAMS. Write a query to fix the record.

UPDATE actor SET first_name='HARPO' WHERE first_name='GROUCHO' AND last_name='WILLIAMS';



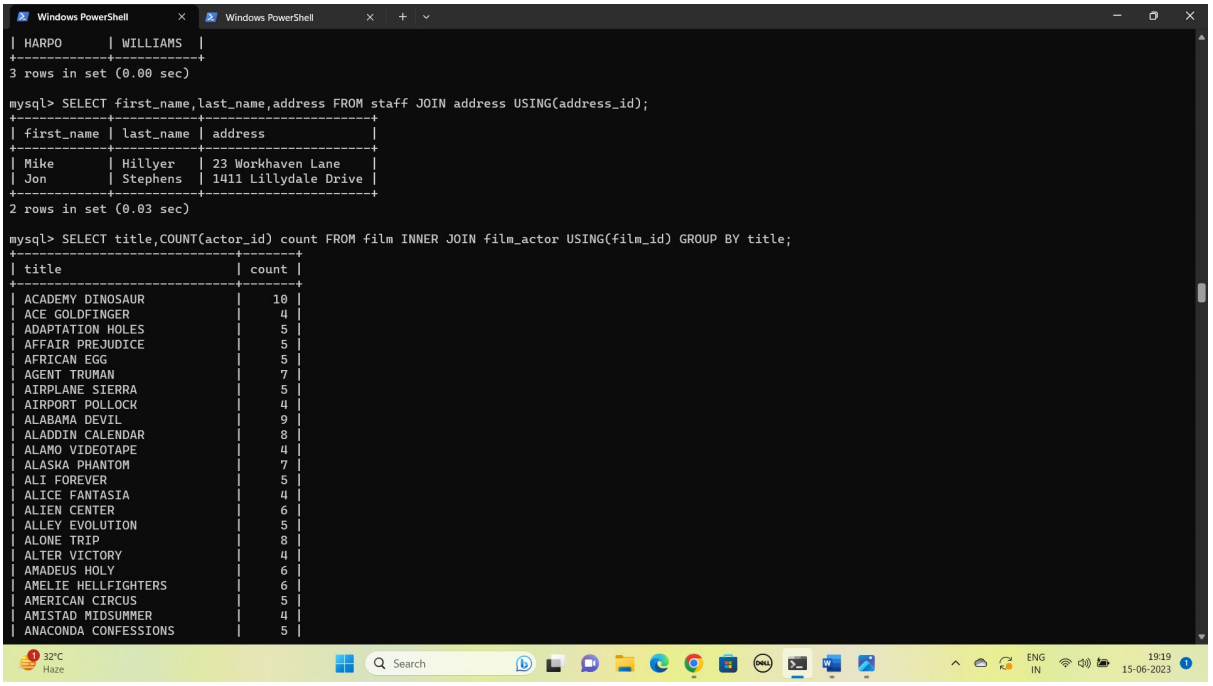
7. Use JOIN to display the first and last names, as well as the address, of each staff member. Use the tables staff and address:

SELECT first_name,last_name,address FROM staff JOIN address USING(address_id);



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

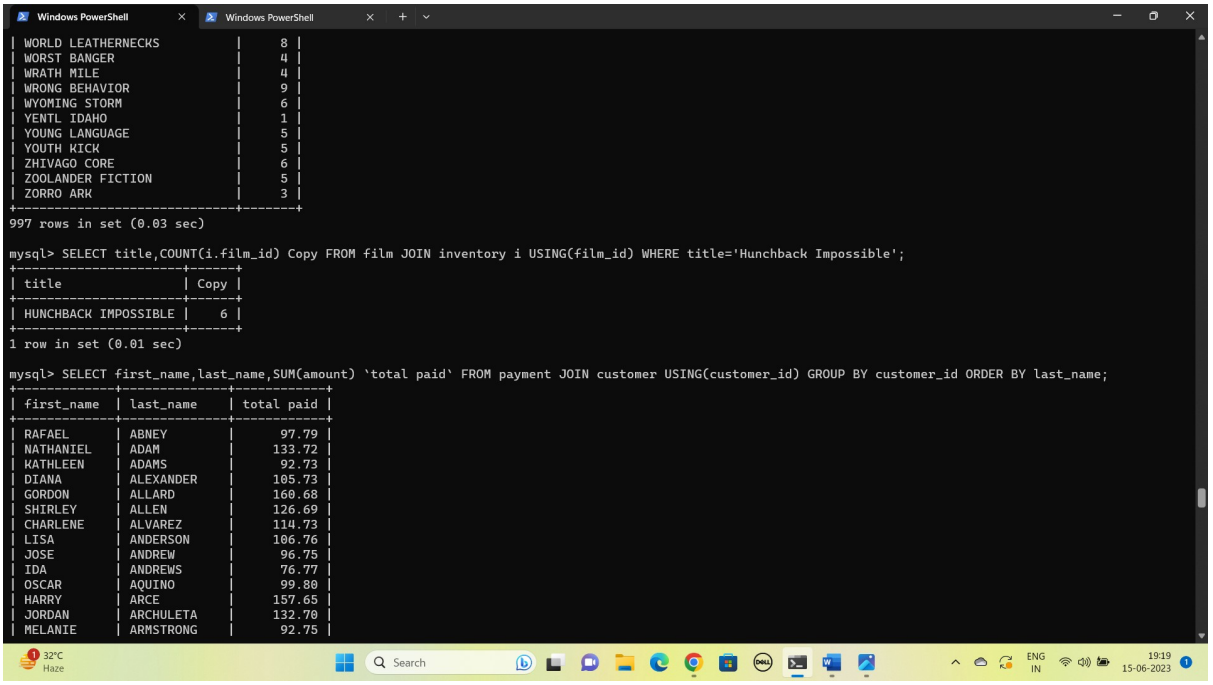
SELECT title,COUNT(actor_id) count FROM film INNER JOIN film_actor USING(film_id) GROUP BY title;



```
| HARPO      | WILLIAMS  |
+------------+-----------+
3 rows in set (0.00 sec)

mysql> SELECT first_name,last_name,address FROM staff JOIN address USING(address_id);
+------------+-----------+----------------------+
| first_name | last_name | address              |
+------------+-----------+----------------------+
| Mike       | Hillyer   | 23 Workhaven Lane    |
| Jon        | Stephens  | 1411 Lillydale Drive |
+------------+-----------+----------------------+
2 rows in set (0.03 sec)

mysql> SELECT title,COUNT(actor_id) count FROM film INNER JOIN film_actor USING(film_id) GROUP BY title;
+----------------------------+-------+
| title                      | 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 |
| AMERICAN CIRCUS            |     5 |
| AMISTAD MIDSUMMER          |     4 |
| ANACONDA CONFESSIONS       |     5 |
```

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

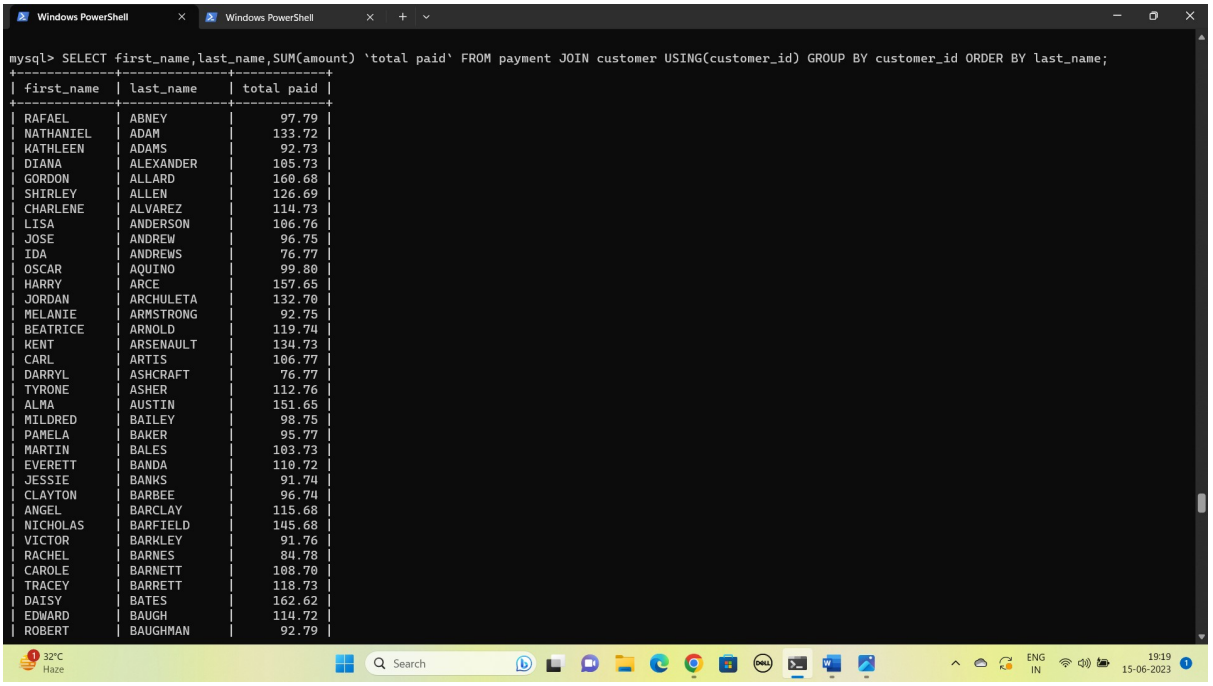
SELECT title,COUNT(i.film_id) Copy FROM film JOIN inventory i USING(film_id) WHERE title='Hunchback Impossible';



```
| WORLD LEATHERNECKS         |     8 |
| WORST BANGER               |     4 |
| WRATH MILE                 |     4 |
| WRONG BEHAVIOR             |     9 |
| WYOMING STORM              |     6 |
| YENTL IDAHO                |     1 |
| YOUNG LANGUAGE             |     5 |
| YOUTH KICK                 |     5 |
| ZHIVAGO CORE               |     6 |
| ZOOLANDER FICTION          |     5 |
| ZORRO ARK                  |     3 |
+----------------------------+-------+
997 rows in set (0.03 sec)

mysql> SELECT title,COUNT(i.film_id) Copy FROM film JOIN inventory i USING(film_id) WHERE title='Hunchback Impossible';
+----------------------+------+
| title                | Copy |
+----------------------+------+
| HUNCHBACK IMPOSSIBLE |    6 |
+----------------------+------+
1 row in set (0.01 sec)

mysql> SELECT first_name,last_name,SUM(amount) `total paid` FROM payment JOIN customer USING(customer_id) GROUP BY customer_id ORDER BY last_name;
+------------+--------------+------------+
| first_name | last_name    | 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 |
```

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

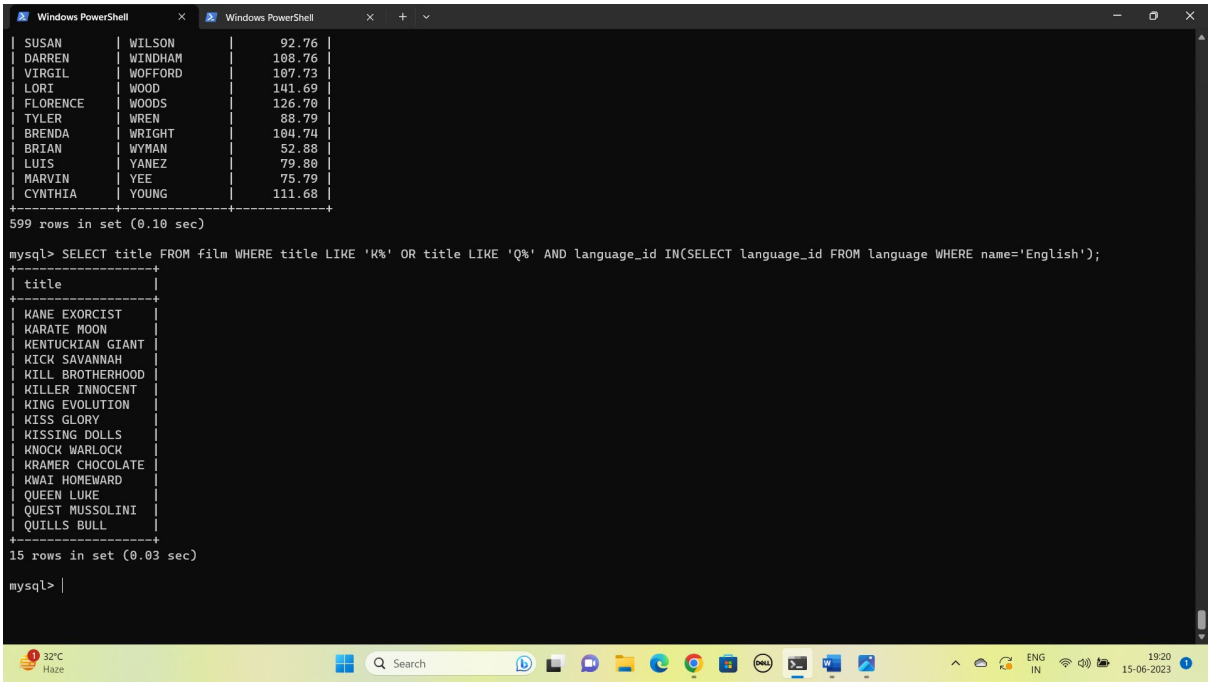
```
SELECT first_name,last_name,SUM(amount) `total paid` FROM payment JOIN customer USING(customer_id) GROUP BY customer_id ORDER BY last_name;
```



```
mysql> SELECT first_name,last_name,SUM(amount) `total paid` FROM payment JOIN customer USING(customer_id) GROUP BY customer_id ORDER BY last_name;
+-------------+--------------+------------+
| first_name  | last_name    | 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 |
| MILDRED     | BAILEY       |      98.75 |
| PAMELA      | BAKER        |      95.77 |
| MARTIN      | BALES        |     103.73 |
| EVERETT     | BANDA        |     110.72 |
| JESSIE      | BANKS        |      91.74 |
| CLAYTON     | BARBEE       |      96.74 |
| ANGEL       | BARCLAY      |     115.68 |
| NICHOLAS    | BARFIELD     |     145.68 |
| VICTOR      | BARKLEY      |      91.76 |
| RACHEL      | BARNES       |      84.78 |
| CAROLE      | BARNETT      |     108.70 |
| TRACEY      | BARRETT      |     118.73 |
| DAISY       | BATES        |     162.62 |
| EDWARD      | BAUGH        |     114.72 |
| ROBERT      | BAUGHMAN     |      92.79 |
```

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.

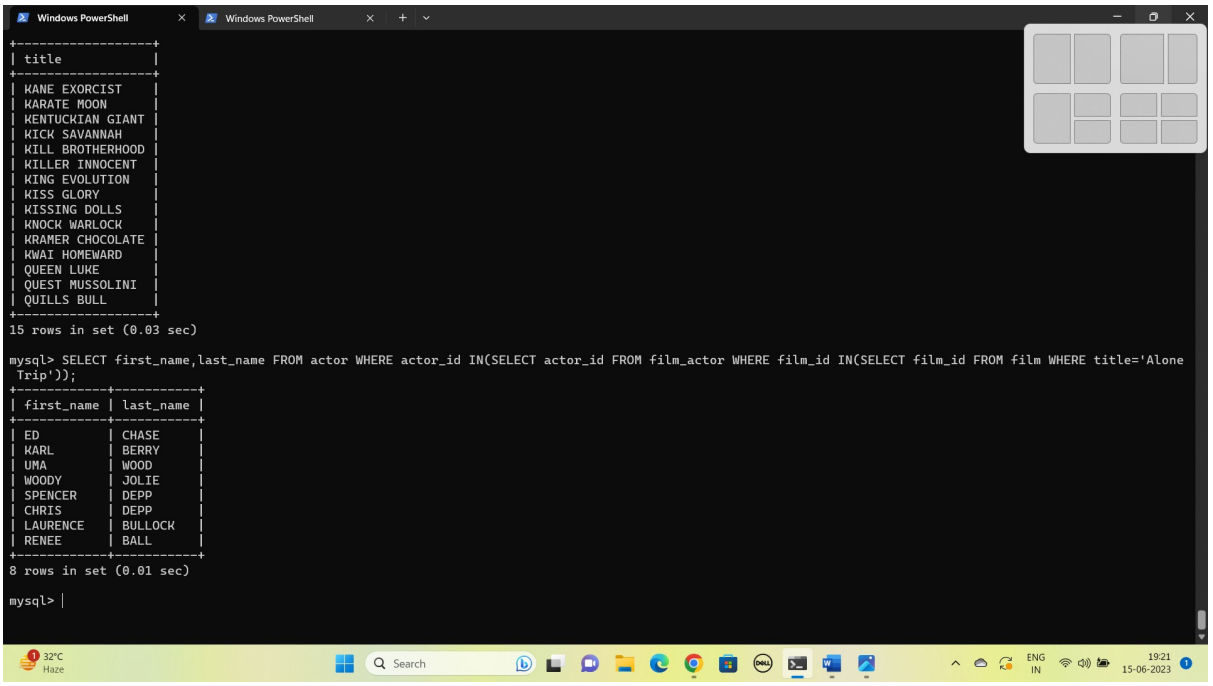
```
SELECT title FROM film WHERE title LIKE 'K%' OR title LIKE 'Q%' AND language_id IN(SELECT language_id FROM language WHERE name='English');
```



```
| SUSAN       | WILSON       |      92.76 |
| DARREN      | WINDHAM      |     108.76 |
| VIRGIL      | WOFFORD      |     107.73 |
| LORI        | WOOD         |     141.69 |
| FLORENCE    | WOODS        |     126.70 |
| TYLER       | WREN         |      88.79 |
| BRENDA      | WRIGHT       |     104.74 |
| BRIAN       | WYMAN        |      52.88 |
| LUIS        | YANEZ        |      79.80 |
| MARVIN      | YEE          |      75.79 |
| CYNTHIA     | YOUNG        |     111.68 |
+-------------+--------------+------------+
599 rows in set (0.10 sec)

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 (0.03 sec)

mysql>
```

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

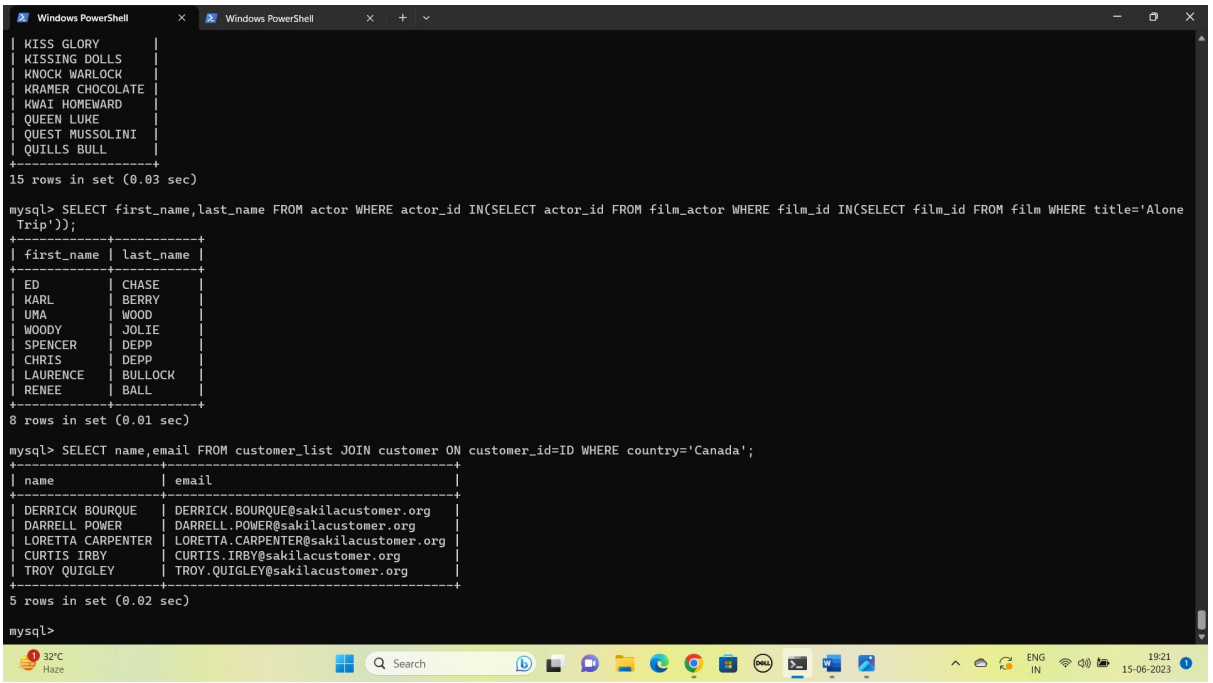
```
SELECT first_name,last_name FROM actor WHERE actor_id IN(SELECT actor_id FROM film_actor WHERE film_id IN(SELECT film_id FROM film WHERE title='Alone Trip'));
```



```
+-------------------+
| 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 (0.03 sec)

mysql> SELECT first_name,last_name FROM actor WHERE actor_id IN(SELECT actor_id FROM film_actor WHERE film_id IN(SELECT film_id FROM film WHERE title='Alone
 Trip'));
+------------+-----------+
| first_name | last_name |
+------------+-----------+
| ED         | CHASE     |
| KARL       | BERRY     |
| UMA        | WOOD      |
| WOODY      | JOLIE     |
| SPENCER    | DEPP      |
| CHRIS      | DEPP      |
| LAURENCE   | BULLOCK   |
| RENEE      | BALL      |
+------------+-----------+
8 rows in set (0.01 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.

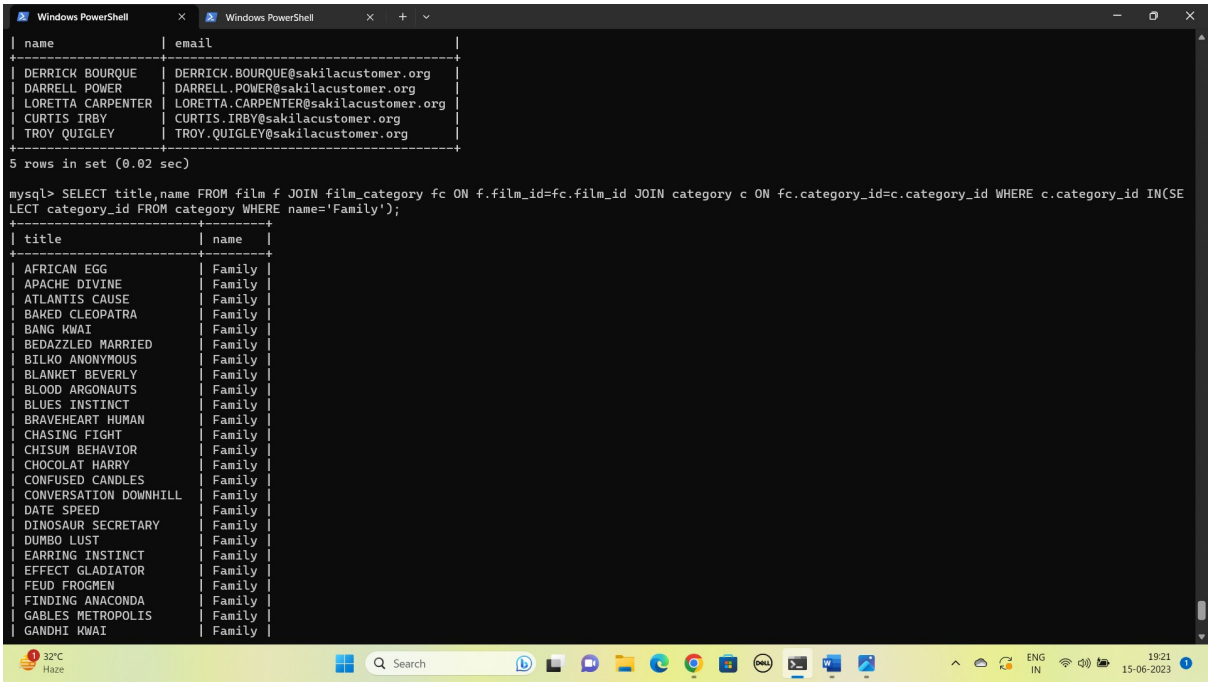
```
SELECT name,email FROM customer_list JOIN customer ON customer_id=ID WHERE country='Canada';
```



```
| KISS GLORY        |
| KISSING DOLLS     |
| KNOCK WARLOCK     |
| KRAMER CHOCOLATE  |
| KWAI HOMEWARD     |
| QUEEN LUKE        |
| QUEST MUSSOLINI   |
| QUILLS BULL       |
+-------------------+
15 rows in set (0.03 sec)

mysql> SELECT first_name,last_name FROM actor WHERE actor_id IN(SELECT actor_id FROM film_actor WHERE film_id IN(SELECT film_id FROM film WHERE title='Alone
 Trip'));
+------------+-----------+
| first_name | last_name |
+------------+-----------+
| ED         | CHASE     |
| KARL       | BERRY     |
| UMA        | WOOD      |
| WOODY      | JOLIE     |
| SPENCER    | DEPP      |
| CHRIS      | DEPP      |
| LAURENCE   | BULLOCK   |
| RENEE      | BALL      |
+------------+-----------+
8 rows in set (0.01 sec)

mysql> SELECT name,email FROM customer_list JOIN customer ON customer_id=ID WHERE country='Canada';
+-------------------+--------------------------------------+
| name              | email                                |
+-------------------+--------------------------------------+
| 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 (0.02 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.

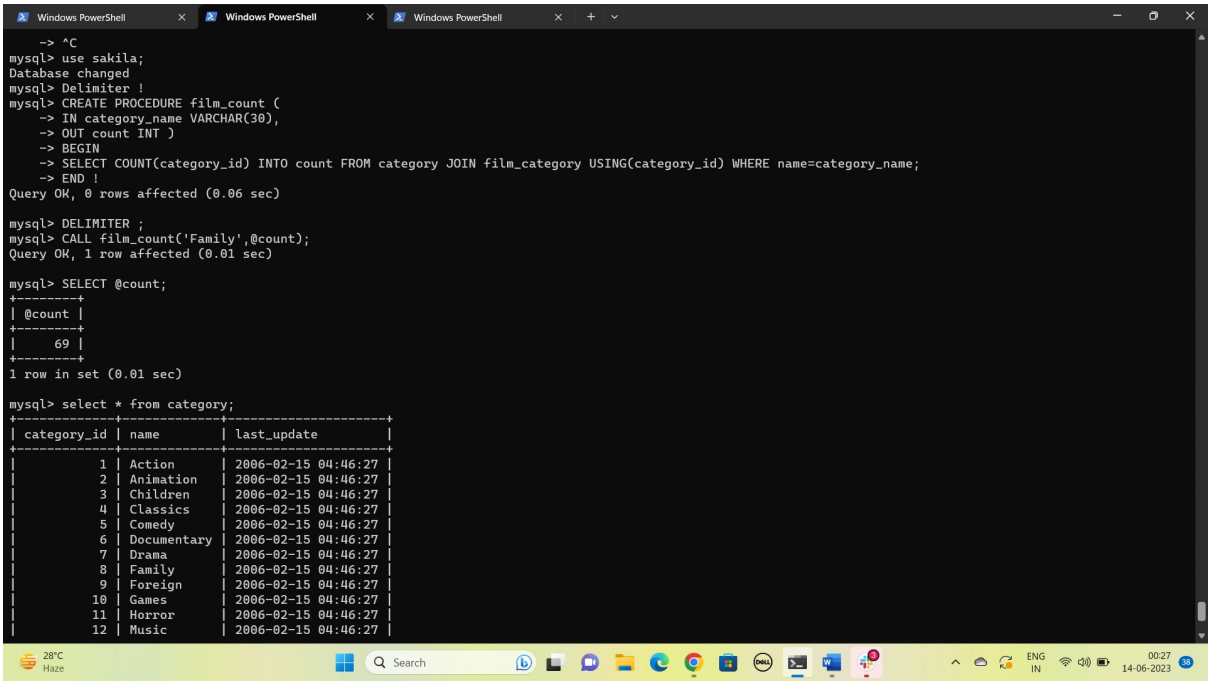
```
SELECT title,name FROM film f JOIN film_category fc ON f.film_id=fc.film_id JOIN category c ON fc.category_id=c.category_id WHERE c.category_id IN(SELECT category_id FROM category WHERE name='Family');
```



```
| name              | email                                  |
+-------------------+----------------------------------------+
| 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 (0.02 sec)

mysql> SELECT title,name 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.category_id IN(SELECT category_id FROM category WHERE name='Family');
+----------------------+--------+
| title                | name   |
+----------------------+--------+
| 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 |
| BRAVEHEART HUMAN     | Family |
| CHASING FIGHT        | Family |
| CHISUM BEHAVIOR      | Family |
| CHOCOLAT HARRY       | Family |
| CONFUSED CANDLES     | Family |
| CONVERSATION DOWNHILL| Family |
| DATE SPEED           | Family |
| DINOSAUR SECRETARY   | Family |
| DUMBO LUST           | Family |
| EARRING INSTINCT     | Family |
| EFFECT GLADIATOR     | Family |
| FEUD FROGMEN         | Family |
| FINDING ANACONDA     | Family |
| GABLES METROPOLIS    | Family |
| GANDHI KWAI          | Family |
```

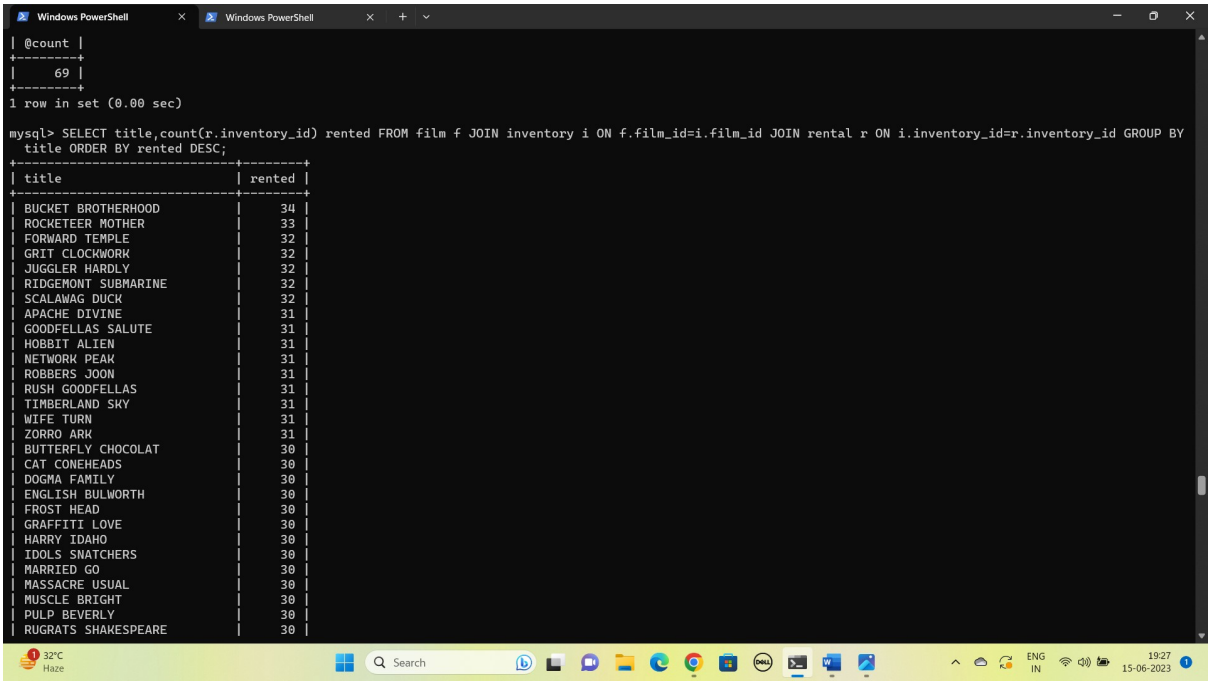
15. Create a Stored procedure to get the count of films in the input category (IN category_name, OUT count)

```
DELIMITER !

CREATE PROCEDURE film_count (
    IN category_name VARCHAR(30),

    OUT count INT )

     BEGIN

    SELECT COUNT(category_id) INTO count FROM category JOIN film_category
    USING(category_id) WHERE name=category_name;

    END !

DELIMITER ;
```



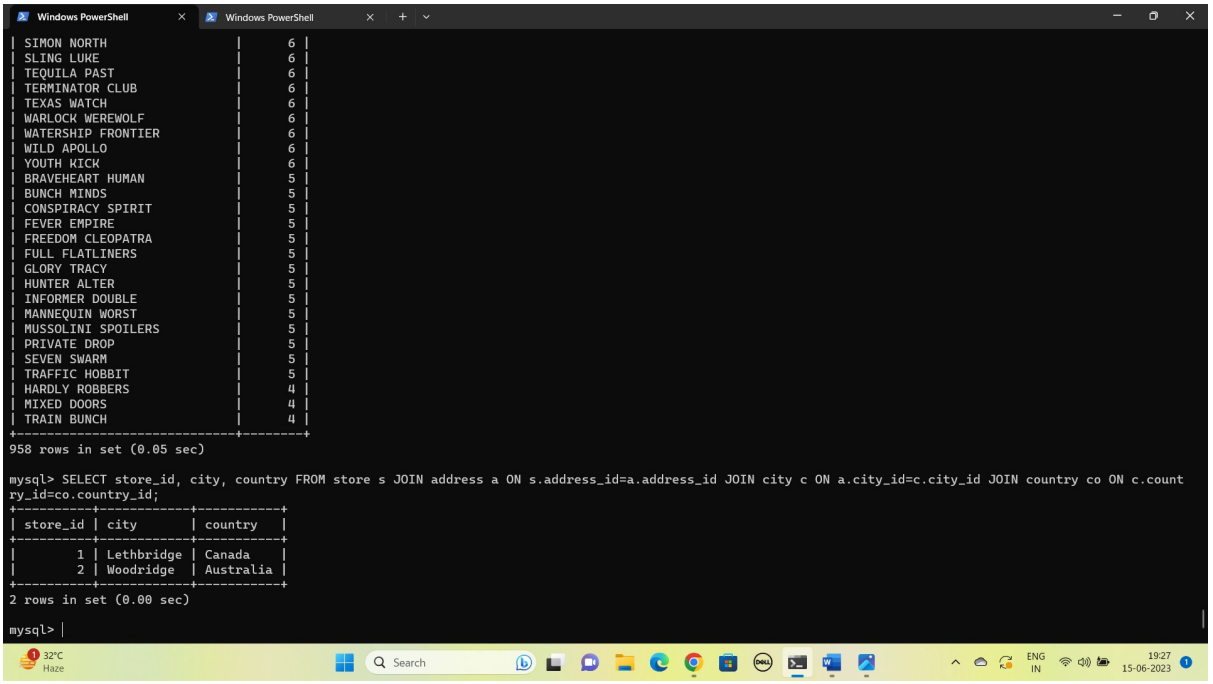
16. Display the most frequently rented movies in descending order.

```
SELECT title,count(r.inventory_id) rented 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  title ORDER BY rented DESC;
```



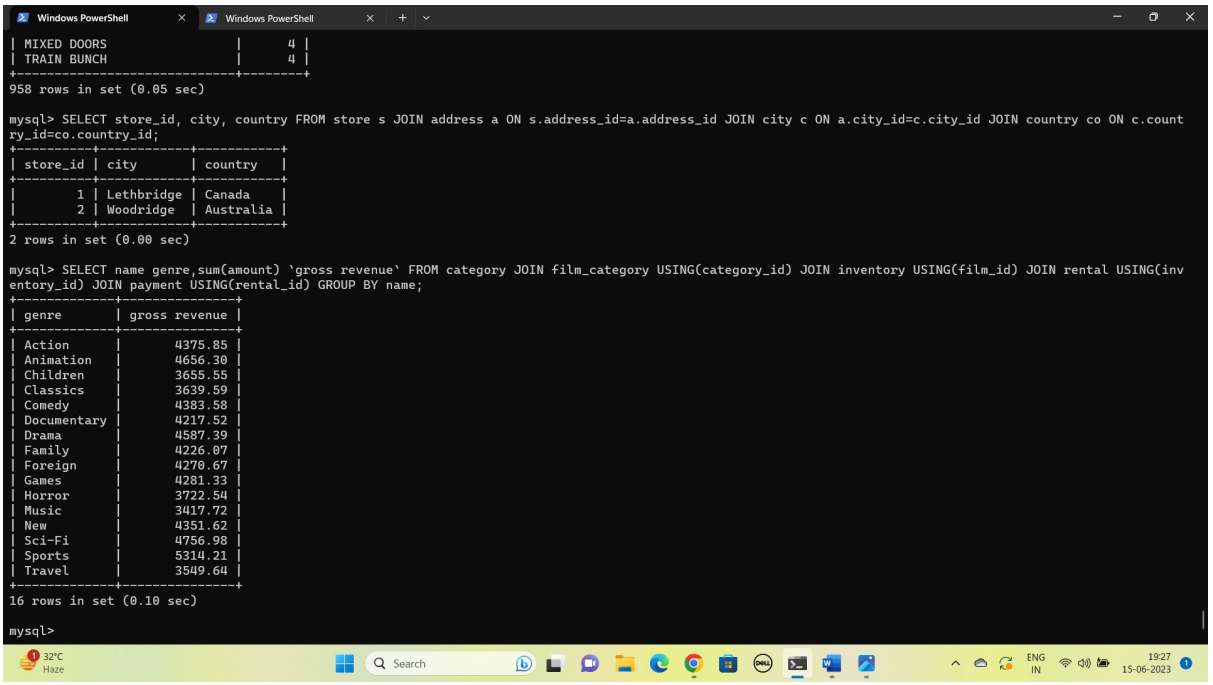
17. Write a query to display for each store its store ID, city, and country.

```
SELECT store_id, city, country FROM store s JOIN address a ON
s.address_id=a.address_id JOIN city c ON a.city_id=c.city_id JOIN country co ON
c.country_id=co.country_id;
```



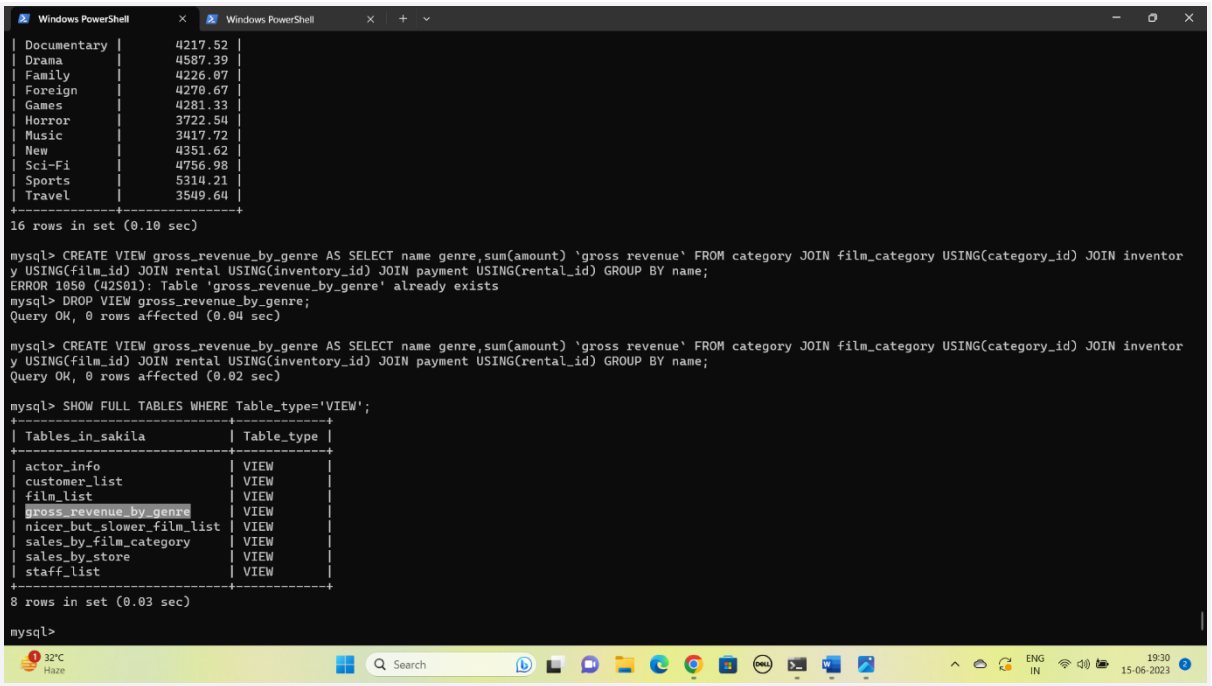
18. List the genres and its gross revenue.

```
SELECT name genre,sum(amount) `gross revenue` FROM category JOIN film_category
USING(category_id) JOIN inventory USING(film_id) JOIN rental USING(inventory_id)
JOIN payment USING(rental_id) GROUP BY name;
```



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

```
CREATE VIEW gross_revenue_by_genre AS SELECT name genre,sum(amount) `gross revenue` FROM category JOIN film_category USING(category_id) JOIN inventory USING(film_id) JOIN rental USING(inventory_id) JOIN payment USING(rental_id) GROUP BY name;
```



20. Select top 5 genres in gross revenue view.

```
SELECT * FROM gross_revenue_by_genre ORDER BY `gross revenue` DESC LIMIT 5;
```