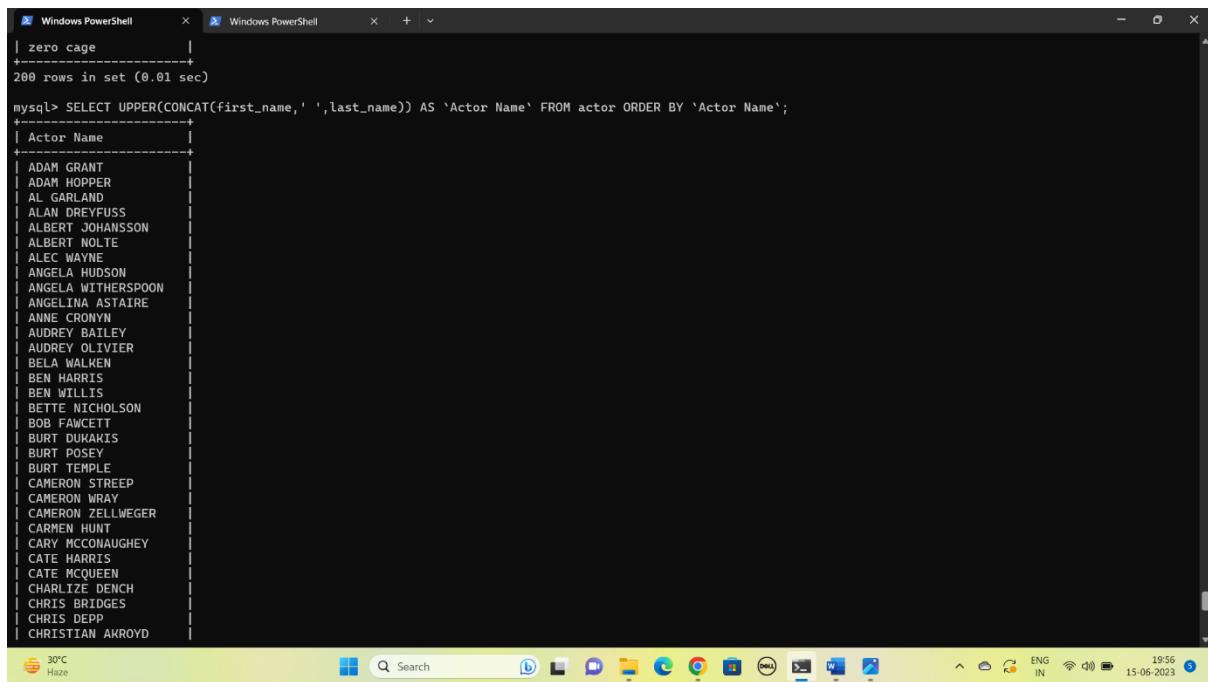


# ASSIGNMENT 1

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`;
```



```
| zero cage      |
+-----+
200 rows in set (0.01 sec)

mysql> SELECT UPPER(CONCAT(first_name,' ',last_name)) AS `Actor Name` FROM actor ORDER BY `Actor Name`;
+-----+
| 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    |
| BURT TEMPLE   |
| CAMERON STREEP |
| CAMERON WRAY   |
| CAMERON ZELLWEGER |
| CARMEN HUNT   |
| CARY MCCONAUGHEY |
| CATE HARRIS   |
| CATE MCQUEEN  |
| CHARLIZE DENCH |
| CHRIS BRIDGES |
| CHRIS DEPP    |
| CHRISTIAN AKROYD |
```

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`;
```

```

Windows PowerShell x Windows PowerShell x + -
+-----+
| VIVIEN BASINGER |
| VIVIEN BERGEN |
| WALTER TORN |
| WARREN JACKMAN |
| WARREN NOLTE |
| WHOOP! HURT |
| WILL WILSON |
| WILLIAM HACKMAN |
| WOODY HOFFMAN |
| WOODY JOLIE |
| ZERO CAGE |
+-----+
200 rows in set (0.04 sec)

mysql> SELECT CONCAT(first_name, ' ', last_name) AS 'Actor Name' FROM actor WHERE last_name LIKE '%GEN%' ORDER BY 'Actor Name';
+-----+
| Actor Name |
+-----+
| GINA DEGENERES |
| JODIE DEGENERES |
| NICK DEGENERES |
| VIVIEN BERGEN |
+-----+
4 rows in set (0.01 sec)

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 (0.02 sec)

mysql> SELECT last_name,count(last_name) AS Count FROM actor GROUP BY last_name;
+-----+-----+
| last_name | Count |
+-----+-----+
| AKROYD | 3 |
+-----+

```

32°C Haze 19:17 15-06-2023

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');`

```

Windows PowerShell x Windows PowerShell x + -
+-----+
| VIVIEN BASINGER |
| VIVIEN BERGEN |
| WALTER TORN |
| WARREN JACKMAN |
| WARREN NOLTE |
| WHOOP! HURT |
| WILL WILSON |
| WILLIAM HACKMAN |
| WOODY HOFFMAN |
| WOODY JOLIE |
| ZERO CAGE |
+-----+
200 rows in set (0.04 sec)

mysql> SELECT CONCAT(first_name, ' ',last_name) AS 'Actor Name' FROM actor WHERE last_name LIKE '%GEN%' ORDER BY 'Actor Name';
+-----+
| Actor Name |
+-----+
| GINA DEGENERES |
| JODIE DEGENERES |
| NICK DEGENERES |
| VIVIEN BERGEN |
+-----+
4 rows in set (0.01 sec)

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 (0.02 sec)

mysql> SELECT last_name,count(last_name) AS Count FROM actor GROUP BY last_name;
+-----+-----+
| last_name | Count |
+-----+-----+
| AKROYD | 3 |
+-----+

```

32°C Haze 19:17 15-06-2023

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;`

```
Windows PowerShell x Windows PowerShell x + - x
3 rows in set (0.02 sec)
mysql> SELECT last_name, count(last_name) AS Count FROM actor GROUP BY last_name;
+-----+-----+
| last_name | Count |
+-----+-----+
| AKROYD   | 3    |
| ALLEN    | 3    |
| ASTAIRE   | 1    |
| BACALL   | 1    |
| BATLEY   | 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    |
| CARREY   | 1    |
| CHAPLIN  | 1    |
| CHASE    | 2    |
| CLOSE    | 1    |
| COSTNER  | 1    |
| CRAWFORD | 2    |
| CRONYN   | 2    |
| CROWE    | 1    |
| CRUTSE   | 1    |
| CRUZ     | 1    |
| DAMON    | 1    |
| DAVIS    | 3    |
| DAY-LEWIS | 1    |
| DEAN     | 2    |
+-----+-----+
121 rows in set (0.01 sec)

mysql> SELECT last_name, count(last_name) AS Count FROM actor GROUP BY last_name HAVING Count>=2;
+-----+-----+
| last_name | Count |
+-----+-----+
| ZELLWEGER | 3    |
+-----+-----+
```

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;`

```
Windows PowerShell x Windows PowerShell x + - x
121 rows in set (0.01 sec)

mysql> SELECT last_name, count(last_name) AS Count FROM actor GROUP BY last_name HAVING Count>=2;
+-----+-----+
| last_name | Count |
+-----+-----+
| AKROYD   | 3    |
| ALLEN    | 3    |
| BATLEY   | 2    |
| BENING   | 2    |
| 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    |
| GOODING  | 2    |
| GUINNESS | 3    |
| HACKMAN  | 2    |
| HARRIS   | 3    |
| HOFFMAN  | 3    |
| HOPKINS  | 3    |
| HOPPER   | 2    |
| JACKMAN  | 2    |
| JOHANSSON | 3    |
| KEITEL   | 3    |
| KILMER   | 5    |
| MCCONAUGHEY | 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';
```

Windows PowerShell

OLIVIER	2
PALTROW	2
PECK	3
PENN	2
SILVERSTONE	2
STREEP	2
TANDY	2
TEMPLE	4
TORN	3
TRACY	2
WAHLBERG	2
WEST	2
WILLIAMS	3
WILLIS	3
WINSLET	2
WOOD	2
ZELLWEGER	3

55 rows in set (0.00 sec)

mysql> UPDATE actor SET first\_name='GROUCHO' WHERE first\_name='HARPO' AND last\_name='WILLIAMS';  
Query OK, 1 row affected (0.04 sec)  
Rows matched: 1 Changed: 1 Warnings: 0

mysql> UPDATE actor SET first\_name='HARPO' WHERE first\_name='GROUCHO' AND last\_name='WILLIAMS';  
Query OK, 1 row affected (0.02 sec)  
Rows matched: 1 Changed: 1 Warnings: 0

mysql> SELECT first\_name,last\_name FROM actor WHERE last\_name='WILLIAMS';

first_name	last_name
SEAN	WILLIAMS
MORGAN	WILLIAMS
HARPO	WILLIAMS

3 rows in set (0.00 sec)

mysql> SELECT first\_name,last\_name,address FROM staff JOIN address USING(address\_id);

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);
```

Windows PowerShell

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

32°C Haze

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
| ALT 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 |   185.73
| GORDON    | ALLARD   |   169.68
| 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 | 185.73 |
| 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 | 115.68 |
| VICTOR | BARKLEY | 91.76 |
| RACHEL | BARNES | 84.78 |
| CAROLE | BARNETT | 108.70 |
| TRACEY | BARRETT | 118.73 |
| DATSY | 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.

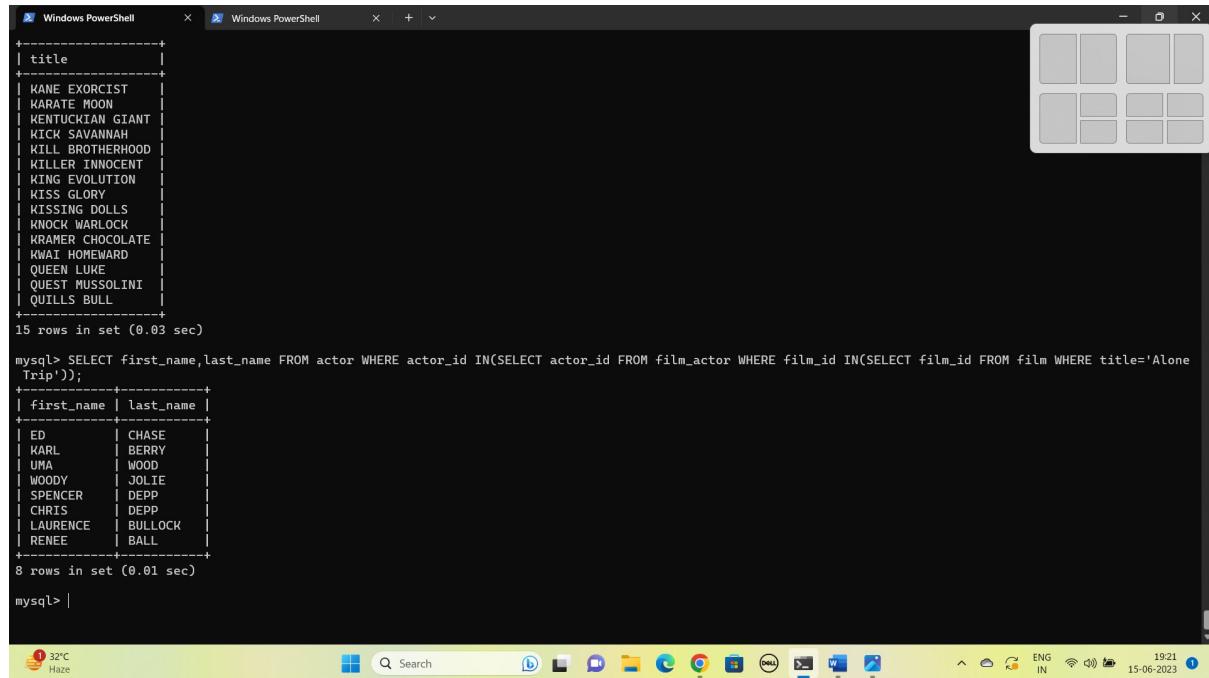
```
SELECT title FROM film WHERE title LIKE 'K%' OR title LIKE 'Q%' AND language_id
IN(SELECT language_id FROM language WHERE name='English');
```

```
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 |
| WILL 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.

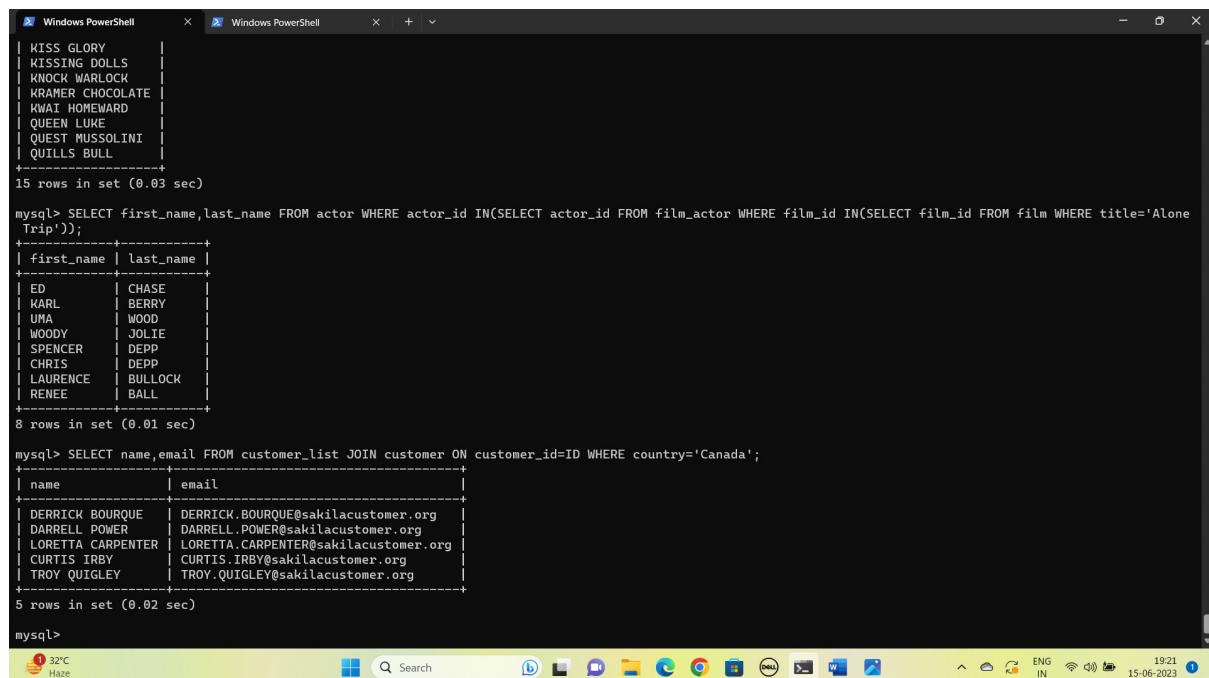
```
SELECT first_name,last_name FROM actor WHERE actor_id IN(SELECT actor_id FROM film_actor WHERE film_id IN(SELECT film_id FROM film WHERE title='Alone Trip'));
```



```
+-----+  
| 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';
```



```
+-----+  
| title |  
+-----+  
| 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 ;
```

```

>> ^C
mysql> use sakila;
Database changed
mysql> Delimiter !
mysql> 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 !
Query OK, 0 rows affected (0.06 sec)

mysql> DELIMITER ;
mysql> CALL film_count('Family',@count);
Query OK, 1 row affected (0.01 sec)

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

mysql> select * from category;
+-----+-----+-----+
| category_id | name      | last_update |
+-----+-----+-----+
|      1 | Action    | 2006-02-15 04:46:27 |
|      2 | Animation | 2006-02-15 04:46:27 |
|      3 | Children   | 2006-02-15 04:46:27 |
|      4 | Classics   | 2006-02-15 04:46:27 |
|      5 | Comedy    | 2006-02-15 04:46:27 |
|      6 | Documentary| 2006-02-15 04:46:27 |
|      7 | Drama     | 2006-02-15 04:46:27 |
|      8 | Family    | 2006-02-15 04:46:27 |
|      9 | Foreign   | 2006-02-15 04:46:27 |
|     10 | Games    | 2006-02-15 04:46:27 |
|     11 | Horror    | 2006-02-15 04:46:27 |
|     12 | Music    | 2006-02-15 04:46:27 |
+-----+-----+-----+

```

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;
```

```

+-----+
| @count |
+-----+
|    69 |
+-----+
1 row in set (0.00 sec)

mysql> 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;
+-----+-----+
| title          | rented |
+-----+-----+
| BUCKET BROTHERHOOD | 34 |
| ROCKETEER MOTHER | 33 |
| FORWARD TEMPLE | 32 |
| GRIT CLOCKWORK | 32 |
| JUGGLER HARDLY | 32 |
| RIDGEMONT SUBMARINE | 32 |
| SCALAWAG DUCK | 32 |
| APACHE DIVINE | 31 |
| GOODFELLAS SALUTE | 31 |
| HOBBIT ALIEN | 31 |
| NETWORK PEAK | 31 |
| ROBBERS JOON | 31 |
| RUSH GOODFELLAS | 31 |
| TIMBERLAND SKY | 31 |
| WIFE TURN | 31 |
| ZORRO ARK | 31 |
| BUTTERFLY CHOCOLAT | 30 |
| CAT CONEHEADS | 30 |
| DOGMA FAMILY | 30 |
| ENGLISH BULWORTH | 30 |
| FROST HEAD | 30 |
| GRAFFITI LOVE | 30 |
| HARRY IDAHO | 30 |
| IDOLS SNATCHERS | 30 |
| MARRIED GO | 30 |
| MASSACRE USUAL | 30 |
| MUSCLE BRIGHT | 30 |
| PULP BEVERLY | 30 |
| RUGRATS SHAKESPEARE | 30 |
+-----+-----+

```

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;
```

```
| SIMON NORTH | 6 |
| SLING LUKE | 6 |
| TEQUILA PAST | 6 |
| TERMINATOR CLUB | 6 |
| TEXAS WATCH | 6 |
| WARLOCK WEREWOLF | 6 |
| WATERSHIP FRONTIER | 6 |
| WILD APOLLO | 6 |
| YOUTH KICK | 6 |
| BRAVEHEART HUMAN | 5 |
| BUNCH MINDS | 5 |
| CONSPIRACY SPIRIT | 5 |
| FEVER EMPIRE | 5 |
| FREEDOM CLEOPATRA | 5 |
| FULL FLATLINERS | 5 |
| GLORY TRACY | 5 |
| HUNTER ALTER | 5 |
| INFORMER DOUBLE | 5 |
| MANNEQUIN WORST | 5 |
| MUSSOLINI SPOILERS | 5 |
| PRIVATE DROP | 5 |
| SEVEN SWARM | 5 |
| TRAFFIC HOBBIT | 5 |
| HARDLY ROBBERS | 4 |
| MIXED DOORS | 4 |
| TRAIN BUNCH | 4 |
+-----+
958 rows in set (0.05 sec)

mysql> 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;
+-----+-----+-----+
| 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.

```
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;
```

```
| MIXED DOORS | 4 |
| TRAIN BUNCH | 4 |
+-----+
958 rows in set (0.05 sec)

mysql> 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;
+-----+-----+-----+
| store_id | city      | country   |
+-----+-----+-----+
| 1       | Lethbridge | Canada    |
| 2       | Woodridge  | Australia |
+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> 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;
+-----+-----+
| genre      | gross revenue |
+-----+-----+
| Action     | 4375.85      |
| Animation  | 4656.30      |
| Children   | 3655.55      |
| Classics   | 3639.59      |
| Comedy     | 4383.58      |
| Documentary| 4217.52      |
| Drama      | 4587.39      |
| Family     | 4226.07      |
| Foreign    | 4270.67      |
| Games      | 4281.33      |
| Horror     | 3722.54      |
| Music      | 3417.72      |
| New        | 4351.62      |
| Sci-Fi     | 4756.98      |
| Sports     | 5314.21      |
| Travel     | 3549.64      |
+-----+-----+
16 rows in set (0.10 sec)

mysql> |
```

### 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;
```

```

Windows PowerShell x Windows PowerShell x + -
+-----+
| Documentary | 4217.52 |
| Drama       | 4587.39 |
| Family      | 4226.07 |
| Foreign     | 4278.67 |
| Games       | 4281.33 |
| Horror      | 3722.54 |
| Music       | 3417.72 |
| New         | 4351.62 |
| Sci-Fi      | 4756.98 |
| Sports      | 5314.21 |
| Travel      | 3549.64 |
+-----+
16 rows in set (0.10 sec)

mysql> 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;
ERROR 1050 (42S01): Table 'gross_revenue_by_genre' already exists
mysql> DROP VIEW gross_revenue_by_genre;
Query OK, 0 rows affected (0.04 sec)

mysql> 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;
Query OK, 0 rows affected (0.02 sec)

mysql> SHOW FULL TABLES WHERE Table_type='VIEW';
+-----+
| Tables_in_sakila | Table_type |
+-----+
| actor_info      | VIEW      |
| customer_list   | VIEW      |
| film_list       | VIEW      |
| gross_revenue_by_genre | VIEW      |
| nicer_but_slower_film_list | VIEW      |
| sales_by_film_category | VIEW      |
| sales_by_store  | VIEW      |
| staff_list      | VIEW      |
+-----+
8 rows in set (0.03 sec)

mysql>

```

## 20. Select top 5 genres in gross revenue view.

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

```

Windows PowerShell x Windows PowerShell x + -
16 rows in set (0.10 sec)

mysql> 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;
ERROR 1050 (42S01): Table 'gross_revenue_by_genre' already exists
mysql> DROP VIEW gross_revenue_by_genre;
Query OK, 0 rows affected (0.04 sec)

mysql> 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;
Query OK, 0 rows affected (0.02 sec)

mysql> SHOW FULL TABLES WHERE Table_type='VIEW';
+-----+
| Tables_in_sakila | Table_type |
+-----+
| actor_info      | VIEW      |
| customer_list   | VIEW      |
| film_list       | VIEW      |
| gross_revenue_by_genre | VIEW      |
| nicer_but_slower_film_list | VIEW      |
| sales_by_film_category | VIEW      |
| sales_by_store  | VIEW      |
| staff_list      | VIEW      |
+-----+
8 rows in set (0.03 sec)

mysql> SELECT * FROM gross_revenue_by_genre ORDER BY `gross revenue` DESC LIMIT 5;
+-----+
| genre | gross revenue |
+-----+
| Sports | 5314.21 |
| Sci-Fi | 4756.98 |
| Animation | 4656.38 |
| Drama | 4587.39 |
| Comedy | 4383.58 |
+-----+
5 rows in set (0.12 sec)

mysql>

```

