Exercises

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';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ACTOR NAME
PENELOPE GUINESS
NICK WAHLBERG
ED CHASE
JENNIFER DAVIS
JOHNNY LOLLOBRIGIDA
BETTE NICHOLSON
GRACE MOSTEL
MATTHEW JOHANSSON
JOE SWANK
CHRISTIAN GABLE
ZERO CAGE
KARL BERRY
UMA WOOD
VIVIEN BERGEN
CUBA OLIVIER
FRED COSTNER
HELEN VOIGHT

Result 18

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

```
SELECT * FROM actor WHERE last_name LIKE '%GEN%';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

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
NULL	NULL	NULL	NULL

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');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

country_id	country
1	Afghanistan
12	Bangladesh
23	China
NULL	NULL

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

```
SELECT last_name, COUNT(*) AS "Number of Actors" FROM actor
GROUP BY last_name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

last_name	Number of Actors
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

Result 21

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(*) AS "Number of Actors" FROM actor
GROUP BY last_name
HAVING COUNT(*) >= 2;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

last_name	Number of Actors
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

Result 22

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';
SELECT * FROM actor WHERE first_name = 'HARPO';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

actor_id	first_name	last_name	last_update
172	HARPO	WILLIAMS	2024-08-13 10:52:51
NULL	NULL	NULL	NULL

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 staff.first_name, staff.last_name, address.address
FROM staff
JOIN address ON staff.address_id = address.address_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name	address
Mike	Hillyer	23 Workhaven Lane
Jon	Stephens	1411 Lillydale Drive

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 film.title, COUNT(film_actor.actor_id) AS "Number of Actors"
FROM film
INNER JOIN film_actor ON film.film_id = film_actor.film_id
GROUP BY film.title;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

title	Number of Actors
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

Result 25

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

```
SELECT COUNT(*) AS "Number of Copies"
FROM inventory
INNER JOIN film ON inventory.film_id = film.film_id
WHERE film.title = 'Hunchback Impossible';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Number of Copies
6

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 c.first_name, c.last_name, SUM(p.amount) AS "Total Paid"
FROM payment p
JOIN customer c ON p.customer_id = c.customer_id
GROUP BY c.first_name, c.last_name
ORDER BY c.last_name, c.first_name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

Result 27

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 = (SELECT language_id FROM language WHERE name = 'English');
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

film 28

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 = (SELECT film_id FROM film WHERE title = 'Alone Trip'));
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name
ED	CHASE
KARL	BERRY
UMA	WOOD
WOODY	JOLIE
SPENCER	DEPP
CHRIS	DEPP
LAURENCE	BULLOCK
RENEE	BALL

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.

```sql
SELECT c.first_name, c.last_name, c.email
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';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_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

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

```
SELECT f.title 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';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

title
AFRICAN EGG
APACHE DIVINE
ATLANTIS CAUSE
BAKED CLEOPATRA
BANG KWAI
BEDAZZLED MARRIED
BILKO ANONYMOUS
BLANKET BEVERLY
BLOOD ARGONAUTS
BLUES INSTINCT
BRAVEHEART HUMAN
CHASING FIGHT
CHISUM BEHAVIOR
CHOCOLAT HARRY
CONFUSED CANDLES

Result 33

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

```
DELIMITER //

CREATE PROCEDURE GetFilmCountByCategory(
    IN category_name VARCHAR(255),
    OUT film_count INT
)
BEGIN
    SELECT COUNT(*) 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 GetFilmCountByCategory('Family', @film_count);
SELECT @film_count;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	@film_count
▶	69

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

```
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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

title	rental_count
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

Result 35

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

```
SELECT s.store_id, c.city, co.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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

store_id	city	country
1	Lethbridge	Canada
2	Woodridge	Australia

18. List the genres and its gross revenue.

```
SELECT c.name AS genre, SUM(p.amount) AS gross_revenue
FROM film f
JOIN film_category fc ON f.film_id = fc.film_id
JOIN category c ON fc.category_id = c.category_id
JOIN inventory i ON f.film_id = i.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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

Result 37

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

```
CREATE VIEW genre_gross_revenue AS
SELECT c.name AS genre, SUM(p.amount) AS gross_revenue
FROM film f
JOIN film_category fc ON f .film_id = fc.film_id
JOIN category c ON fc.category_id = c.category_id
JOIN inventory i ON f.film_id = i.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;

SELECT * FROM genre gross revenue;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

genre_gross_revenue 38

20. Select top 5 genres in gross revenue view.

```
SELECT genre, gross_revenue
FROM genre_gross_revenue
ORDER BY gross_revenue DESC
LIMIT 5;
```

Result Grid | Filter Rows:

genre	gross_revenue
Sports	5314.21
Sci-Fi	4756.98
Animation	4656.30
Drama	4587.39
Comedy	4383.58