Assignment 1

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.

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

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

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

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

Ans:
```
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.

Ans :
```
SELECT last_name, COUNT(*) AS actor_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

Ans :
```
SELECT last_name, COUNT(*) AS actor_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.

Ans:
```
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:

Ans.:
```
SELECT staff.first_name, staff.last_name, address.address
FROM staff
JOIN address ON staff.address_id = address.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.

Ans:
```
SELECT
film.film_id,
film.title,
COUNT(film_actor.actor_id) AS actor_count
FROM
film
INNER JOIN
film_actor ON film.film_id = film_actor.film_id
GROUP BY film.film_id , film.title;
```

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

Ans:
```
SELECT COUNT(*) AS copy_count
FROM inventory
JOIN film ON inventory.film_id = film.film_id
WHERE film.title = 'Hunchback Impossible';
```

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

Ans:
```
SELECT customer.last_name, customer.first_name, SUM(payment.amount) AS    .
.   .        total_paid
         FROM customer
        JOIN payment ON customer.customer_id = payment.customer_id
        GROUP BY customer.last_name, customer.first_name
        ORDER BY customer.last_name ASC;
```

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.

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

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

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

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.

Ans:
```
SELECT customer.first_name, customer.last_name, customer.email
FROM customer
JOIN address ON customer.address_id = address.address_id
JOIN city ON address.city_id = city.city_id
JOIN country ON city.country_id = country.country_id
WHERE country.country = 'Canada';
```

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.

Ans:
```
SELECT film.title
FROM film
JOIN film_category ON film.film_id = film_category.film_id
JOIN category ON film_category.category_id = category.category_id
WHERE category.name = 'Family';
```

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

Ans:
```
DELIMITER //
CREATE PROCEDURE GetFilmCountInCategory(
IN category_name VARCHAR(255),
OUT count INT )
BEGIN
SELECT COUNT(*) INTO count
FROM film
JOIN film_category ON film.film_id = film_category.film_id
JOIN category ON film_category.category_id = category.category_id
WHERE category.name = category_name;
END //
DELIMITER ;
```

To call this stored procedure and retrieve the film count

```
CALL GetFilmCountInCategory('Your Category Name', @film_count);
SELECT @film_count;
```

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

Ans:
```
SELECT film.title, COUNT(rental.rental_id) AS rental_count
FROM film
JOIN inventory ON film.film_id = inventory.film_id
JOIN rental ON inventory.inventory_id = rental.inventory_id
GROUP BY film.film_id, film.title
ORDER BY rental_count DESC;
```

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

Ans:
```
SELECT store.store_id, city.city, country.country
 FROM store
JOIN address ON store.address_id = address.address_id
JOIN city ON address.city_id = city.city_id
JOIN country ON city.country_id = country.country_id;
```

18. List the genres and its gross revenue

Ans:

```
SELECT category.name AS category, SUM(payment.amount) AS gross_income
FROM category
JOIN film_category ON category.category_id = film_category.category_id
JOIN film ON film_category.film_id = film.film_id
JOIN inventory ON film.film_id = inventory.film_id
JOIN rental ON inventory.inventory_id = rental.inventory_id
JOIN payment ON rental.rental_id = payment.rental_id
GROUP BY category.name;
```

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

Ans:

```
CREATE VIEW category_gross_income AS
SELECT category.name AS category, SUM(payment.amount) AS gross_income
FROM category
JOIN film_category ON category.category_id = film_category.category_id
JOIN film ON film_category.film_id = film.film_id
JOIN inventory ON film.film_id = inventory.film_id
JOIN rental ON inventory.inventory_id = rental.inventory_id
JOIN payment ON rental.rental_id = payment.rental_id

GROUP BY category.name;
```

20. Select top 5 category in gross revenue view.

Ans:

```
CREATE VIEW category_gross_income AS

SELECT category.name AS category, SUM(payment.amount) AS gross_income

FROM category

JOIN film_category ON category.category_id = film_category.category_id

JOIN film ON film_category.film_id = film.film_id

JOIN inventory ON film.film_id = inventory.film_id

JOIN rental ON inventory.inventory_id = rental.inventory_id

JOIN payment ON rental.rental_id = payment.rental_id

GROUP BY category.name;


SELECT category, gross_income

FROM category_gross_income

ORDER BY gross_income DESC

LIMIT 5;
```