Practice Question on SQL Joins and Subqueries

1. **Join Practice:**
   Write a query to display the customer's first name, last name, email, and city they live in.

2. **Subquery Practice (Single Row):**
   Retrieve the film title, description, and release year for the film that has the longest duration.

3. **Join Practice (Multiple Joins):**
   List the customer name, rental date, and film title for each rental made. Include customers who have never rented a film.

4. **Subquery Practice (Multiple Rows):**
   Find the number of actors for each film. Display the film title and the number of actors for each film.

5. **Join Practice (Using Aliases):**
   Display the first name, last name, and email of customers along with the rental date, film title, and rental return date.

6. **Subquery Practice (Conditional):**
   Retrieve the film titles that are rented by customers whose email domain ends with '.net'.

7. **Join Practice (Aggregation):**
   Show the total number of rentals made by each customer, along with their first and last names.

8. **Subquery Practice (Aggregation):**
   List the customers who have made more rentals than the average number of rentals made by all customers.

9. **Join Practice (Self Join):**
   Display the customer first name, last name, and email along with the names of other customers living in the same city.

10. **Subquery Practice (Correlated Subquery):**
    Retrieve the film titles with a rental rate higher than the average rental rate of films in the same category.

11. **Subquery Practice (Nested Subquery):**

Retrieve the film titles along with their descriptions and lengths that have a rental rate greater than the average rental rate of films released in the same year.

12. **Subquery Practice (IN Operator):**
List the first name, last name, and email of customers who have rented at least one film in the 'Documentary' category.

13. **Subquery Practice (Scalar Subquery):**
Show the title, rental rate, and difference from the average rental rate for each film.

14. **Subquery Practice (Existence Check):**
Retrieve the titles of films that have never been rented.

15. **Subquery Practice (Correlated Subquery - Multiple Conditions):**
List the titles of films whose rental rate is higher than the average rental rate of films released in the same year and belong to the 'Sci-Fi' category.

16. **Subquery Practice (Conditional Aggregation):**
Find the number of films rented by each customer, excluding customers who have rented fewer than five films.

Answers :

Absolutely! Here are the queries along with their solutions using the Sakila database:

1. **Join Practice:**
```sql
SELECT c.first_name, c.last_name, c.email, a.city
FROM customer c
JOIN address a ON c.address_id = a.address_id;
```

2. **Subquery Practice (Single Row):**
```sql
SELECT title, description, release_year
FROM film
WHERE length = (SELECT MAX(length) FROM film);
```

3. **Join Practice (Multiple Joins):**
   ```sql
   SELECT c.first_name, c.last_name, r.rental_date, f.title
   FROM customer c
   LEFT JOIN rental r ON c.customer_id = r.customer_id
   LEFT JOIN inventory i ON r.inventory_id = i.inventory_id
   LEFT JOIN film f ON i.film_id = f.film_id;
   ```

4. **Subquery Practice (Multiple Rows):**
   ```sql
   SELECT f.title, COUNT(actor_id) AS actor_count
   FROM film f
   LEFT JOIN film_actor fa ON f.film_id = fa.film_id
   GROUP BY f.title;
   ```

5. **Join Practice (Using Aliases):**
   ```sql
   SELECT c.first_name, c.last_name, c.email, r.rental_date, f.title, r.return_date
   FROM customer c
   JOIN rental r ON c.customer_id = r.customer_id
   JOIN inventory i ON r.inventory_id = i.inventory_id
   JOIN film f ON i.film_id = f.film_id;
   ```

6. **Subquery Practice (Conditional):**
   ```sql
   SELECT title
   FROM film
   WHERE film_id IN (
       SELECT film_id
       FROM inventory
       WHERE store_id IN (
           SELECT store_id
           FROM customer
           WHERE email LIKE '%.net'
       )
   );
   ```

7. **Join Practice (Aggregation):**
   ```sql
   ```

```sql
SELECT c.first_name, c.last_name, COUNT(r.rental_id) AS total_rentals
FROM customer c
LEFT JOIN rental r ON c.customer_id = r.customer_id
GROUP BY c.customer_id
ORDER BY total_rentals DESC;
```

8. **Subquery Practice (Aggregation):**
```sql
SELECT first_name, last_name
FROM customer
WHERE customer_id IN (
    SELECT customer_id
    FROM rental
    GROUP BY customer_id
    HAVING COUNT(rental_id) > (
        SELECT AVG(rental_count)
        FROM (
            SELECT COUNT(rental_id) AS rental_count
            FROM rental
            GROUP BY customer_id
        ) AS avg_rentals
    )
);
```

9. **Join Practice (Self Join):**
```sql
SELECT c1.first_name, c1.last_name, c1.email, c2.first_name AS other_first_name,
c2.last_name AS other_last_name
FROM customer c1
JOIN address a1 ON c1.address_id = a1.address_id
JOIN address a2 ON a1.city = a2.city AND a1.address_id != a2.address_id
JOIN customer c2 ON a2.address_id = c2.address_id;
```

10. **Subquery Practice (Correlated Subquery):**
```sql
SELECT title, rental_rate
FROM film f
WHERE rental_rate > (
    SELECT AVG(rental_rate)
    FROM film
    WHERE category_id = f.category_id
```

```
);
```

11. **Subquery Practice (Nested Subquery):**
```sql
SELECT title, description, length
FROM film
WHERE rental_rate > (
    SELECT AVG(rental_rate)
    FROM film
    WHERE release_year = film.release_year
);
```

12. **Subquery Practice (IN Operator):**

```sql
SELECT first_name, last_name, email
FROM customer
WHERE customer_id IN (
    SELECT DISTINCT c.customer_id
    FROM customer c
    JOIN rental r ON c.customer_id = r.customer_id
    JOIN inventory i ON r.inventory_id = i.inventory_id
    JOIN film_category fc ON i.film_id = fc.film_id
    JOIN category cat ON fc.category_id = cat.category_id
    WHERE cat.name = 'Documentary'
);
```

13. **Subquery Practice (Scalar Subquery):**

```sql
SELECT title, rental_rate,
    rental_rate - (SELECT AVG(rental_rate) FROM film) AS rate_difference
FROM film;
```

14. **Subquery Practice (Existence Check):**

```sql
SELECT title
FROM film
WHERE film_id NOT IN (
```

```sql
    SELECT DISTINCT film_id
    FROM inventory
    WHERE film_id IS NOT NULL
);
```

15. **Subquery Practice (Correlated Subquery - Multiple Conditions):**

```sql
SELECT title
FROM film f
WHERE rental_rate > (
    SELECT AVG(rental_rate)
    FROM film
    WHERE release_year = f.release_year
)
AND film_id IN (
    SELECT fc.film_id
    FROM film_category fc
    JOIN category c ON fc.category_id = c.category_id
    WHERE c.name = 'Sci-Fi'
);
```

16. **Subquery Practice (Conditional Aggregation):**

```sql
SELECT customer_id, COUNT(rental_id) AS film_count
FROM rental
GROUP BY customer_id
HAVING COUNT(rental_id) >= 5;
```