# Food Delivery Data Analytics using SQL

**Q1.** Retrieve a list of user information with their name and date of registration who uses android phones.

**Query:**

```sql
SELECT name AS user_name, registration_date
FROM user_info
WHERE operating_system = 'Android';
```

**Result:**

| user_name | registration_date |
|---|---|
| John Doe | 2023-07-30 12:34:56 |
| Michael Johnson | 2023-07-28 15:23:10 |
| Robert Brown | 2023-07-26 11:22:33 |
| David Davis | 2023-07-24 17:19:56 |
| Richard Lee | 2023-07-22 16:18:56 |

**Q2.** Find out users who have registered on or after 14th of july and sort the list of users in ascending order.

**Query:**

```
SELECT name AS user_name
FROM user_info
WHERE registration_date >= '2023-07-14'
ORDER BY user_name;
```

**Result:**

| user_name |
| --- |
| Amanda Roberts |
| Daniel Evans |
| David Davis |
| Emily Williams |
| James Turner |

**Q3.** Retrieve a list of all restaurants and their menu items. Some restaurants may not have any menu items yet.

**Query:**

```
SELECT a.name AS restaurant_name, b.name AS item_name
FROM restaurant_info a
LEFT JOIN menuitems b
ON a.restaurant_id = b.restaurant_id;
```

**Result:**

| restaurant_name | item_name |
|---|---|
| Restaurant A | Item7 |
| Restaurant A | Item34 |
| Restaurant A | Item38 |
| Restaurant A | Item44 |
| Restaurant B | Item3 |
| Restaurant B | Item26 |

**Q4.** Extend the previous result to include the restaurant's contact number and the availability status of each menu item. If a restaurant does not have any menu items, display "No Menu Items" in the menu item column.

**Query:**

```
SELECT a.name AS restaurant_name, COALESCE(b.name, 'No Menu Items') AS item_name,
a.contact_number, b.availability
FROM restaurant_info a
LEFT JOIN menuitems b
ON a.restaurant_id = b.restaurant_id;
```

**Result:**

| restaurant_name | item_name | contact_number | availability |
|---|---|---|---|
| Restaurant A | Item7 | 9876543210 | 0 |
| Restaurant A | Item34 | 9876543210 | 1 |
| Restaurant A | Item38 | 9876543210 | 1 |
| Restaurant A | Item44 | 9876543210 | 0 |
| Restaurant B | Item3 | 1234567890 | 0 |

**Q5.** Retrieve the total number of orders placed by each user. Sort the results in descending order based on the number of orders.

**Query:**

```
SELECT name AS user_name, COUNT(order_id) AS total_order
FROM user_info a
LEFT JOIN orders b
ON a.id = b.user_id
GROUP BY user_name
ORDER BY total_order desc;
```

**Result:**

| user_name | total_order |
|---|---|
| Linda Miller | 5 |
| Laura Turner | 4 |
| Joseph Scott | 4 |
| William Turner | 4 |
| Mark Harris | 4 |

**Q6.** Find the average price of menu items for each restaurant. Sort the results in ascending order based on the restaurant name.

**Query:**

```sql
SELECT a.name AS restaurant_name, ROUND(AVG(b.price), 2) AS avg_menu_item_price
FROM restaurant_info a
LEFT JOIN menuitems b
ON a.restaurant_id = b.restaurant_id
GROUP BY restaurant_name
ORDER BY restaurant_name ASC;
```

**Result:**

| restaurant_name | avg_menu_item_price |
|---|---|
| Restaurant A | 52.00 |
| Restaurant B | 57.60 |
| Restaurant C | 63.50 |
| Restaurant D | 28.75 |
| Restaurant E | 59.00 |

**Q7.** Identify the restaurant with the highest total sales (sum of order amounts). Display the restaurant name and the total sales amount.

**Query:**

```
SELECT a.name AS restaurant_name, SUM(b.total_amount) AS total_sales_amount
FROM restaurant_info a
LEFT JOIN orders b
ON a.restaurant_id = b.restaurant_id
GROUP BY restaurant_name
ORDER BY restaurant_name ASC;
```

**Result:**

| restaurant_name | total_sales_amount |
|---|---|
| Restaurant A | 1938 |
| Restaurant B | 1655 |
| Restaurant C | 2268 |
| Restaurant D | 1687 |
| Restaurant E | 1275 |

**Q8.** Find the number of orders placed in each city. Sort the results in descending order based on the number of orders.

**Query:**

```
SELECT a.city_name, COUNT(c.order_id) AS number_of_orders
FROM city a
INNER JOIN user_info b
ON a.city_id = b.city_id
INNER JOIN orders c
ON b.id = c.user_id
GROUP BY city_name
ORDER BY number_of_orders DESC;
```

**Result:**

| city_name | number_of_orders |
|---|---|
| Dhaka | 21 |
| Sylhet | 16 |
| Khulna | 14 |
| Chittagong | 9 |
| Rangpur | 7 |

**Q9.** Find the names of restaurants that have at least one menu item with a price greater than $10.

**Query:**

```
SELECT DISTINCT a.name AS restaurant_name
FROM restaurant_info a
INNER JOIN menuitems b
ON a.restaurant_id = b.restaurant_id
WHERE b.price > 10
ORDER BY restaurant_name;
```

**Result:**

| restaurant_name |
| --- |
| Restaurant A |
| Restaurant B |
| Restaurant C |
| Restaurant D |
| Restaurant E |

**Q10.** Retrieve the user names and their corresponding orders where the order total is greater than the average order total for all users.

**Query:**

```sql
SELECT a.name AS user_name, b.order_count AS total_order
FROM user_info a
INNER JOIN
(
    SELECT user_id, COUNT(order_id) AS order_count
    FROM orders
    GROUP BY user_id
) AS b
ON a.id = b.user_id
WHERE b.order_count >
(
    SELECT AVG(ORDER_COUNT)
    FROM
    (
        SELECT user_id, COUNT(order_id) AS order_count
        FROM orders
        GROUP BY user_id
    ) AS c
);
```

**Result:**

| user_name | total_order |
|---|---|
| John Doe | 3 |
| Robert Brown | 4 |
| Linda Miller | 5 |
| David Davis | 3 |
| Karen Allen | 3 |

**Q11.** List the names of users whose last names start with 'S' or ends with ‘e’.

**Query:**

```sql
SELECT name AS user_name
FROM user_info
WHERE SUBSTRING_INDEX(name, " ", -1) LIKE 'S%' OR SUBSTRING_INDEX(name, " ", -1) LIKE '%e';
```

**Result:**

| user_name |
| --- |
| John Doe |
| Jane Smith |
| Richard Lee |
| Joseph Scott |
| Stephanie White |

**Q12.** Find the total order amounts for each restaurant. If a restaurant has no orders, display the restaurant name and a total amount of 0.

**Query:**

```
SELECT a.name AS restaurant_name,
COALESCE(COUNT(b.order_id), 0) AS total_order,
COALESCE(SUM(b.total_amount), 0) AS total_order_amount
FROM restaurant_info a
LEFT JOIN orders b
ON a.restaurant_id = b.restaurant_id
GROUP BY restaurant_name;
```

**Result:**

| restaurant_name | total_order | total_order_amount |
|---|---|---|
| Restaurant A | 4 | 1938 |
| Restaurant B | 5 | 1655 |
| Restaurant C | 8 | 2268 |
| Restaurant D | 6 | 1687 |
| Restaurant E | 8 | 1275 |

**Q13.** Find out how many orders were placed using cash or credit.

**Query:**

```
SELECT a.name AS payment_type_name, COUNT(b.order_id) AS order_quantity
FROM payment_type a
LEFT JOIN payment_transactions b
ON a.pay_type_id = b.pay_type_id
GROUP BY payment_type_name;
```

**Result:**

| payment_type_name | order_quantity |
|---|---|
| Cash | 27 |
| Credit | 44 |

# Thanks for Reading