Swiggy SQL Project

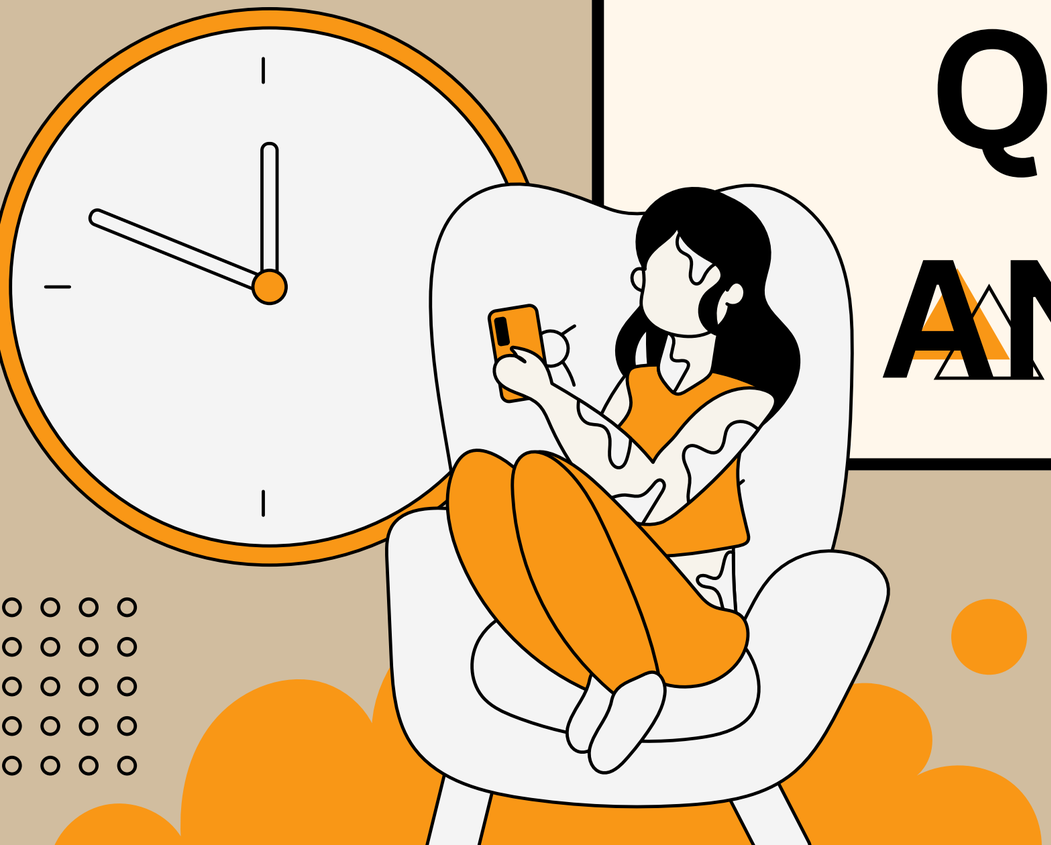
Presented by: Priyansh Bhangalia

About Swiggy

Swiggy is one of India’s largest online food delivery platforms, launched in 2014. Headquartered in Bengaluru, it connects users with a wide range of restaurants, offering an extensive menu selection across various cuisines. Swiggy stands out for its quick delivery times, user-friendly app, and robust customer support. It operates in major Indian cities, using a vast network of delivery personnel to ensure efficient service. Beyond food delivery, Swiggy has expanded its offerings to include grocery delivery through Swiggy Instamart and a pick-up and drop service called Swiggy Genie. The platform's growth is fueled by innovative technology, strategic partnerships, and a focus on customer experience. Swiggy has played a significant role in revolutionizing the food delivery industry in India, making dining more convenient and accessible.

SQL QUERIES ANALYSIS

Lets go



```sql
SELECT
    *
FROM
    customers
WHERE
    city = 'Delhi';
```

1

Display all customers who live in 'Delhi'.


```sql
SELECT
    city, AVG(rating)
FROM
    restaurants
WHERE
    city = 'Mumbai'
GROUP BY city;
```

2

Find the average rating of all restaurants in 'Mumbai'.


```sql
SELECT DISTINCT
    customers.name
FROM
    customers
        JOIN
    orders ON customers.customer_id = orders.customer_id;
```

3

List all customers who have placed at least one order.


```sql
SELECT
    customers.name, COUNT(orders.order_id) AS total_orders
FROM
    customers
        LEFT JOIN
    orders ON customers.customer_id = orders.customer_id
GROUP BY customers.name;
```

4

Display the total number of orders placed by each customer.


```sql
SELECT
    restaurants.name, COALESCE(SUM(orders.total_amount), 0)
FROM
    restaurants
        LEFT JOIN
    orders ON orders.restaurant_id = restaurants.restaurant_id
GROUP BY restaurants.name;
```

5

Find the total revenue generated by each restaurant.


```sql
SELECT
    name, rating
FROM
    restaurants
ORDER BY rating DESC
LIMIT 5;
```

6

Find the top 5 restaurants with the highest average rating.


```sql
SELECT DISTINCT
    customers.name
FROM
    customers
        LEFT JOIN
    orders ON customers.customer_id = orders.customer_id
WHERE
    orders.order_id IS NULL;
```

7

Display all customers who have never placed an order.


```sql
SELECT
    customers.name, COUNT(orders.order_id)
FROM
    customers
        LEFT JOIN
    orders ON customers.customer_id = orders.customer_id
WHERE
    customers.city = 'Mumbai'
GROUP BY customers.name;
```

8

Find the number of orders placed by each customer in 'Mumbai'.


```sql
SELECT
    *
FROM
    orders
WHERE
    order_date >= DATE_SUB(CURDATE(), INTERVAL 30 DAY);
```

9

Display all orders placed in the last 30 days.


```sql
SELECT
    deliverypartners.name, COUNT(orderdelivery.order_id)
FROM
    deliverypartners
        JOIN
    orderdelivery ON deliverypartners.partner_id = orderdelivery.partner_id
        JOIN
    deliveryupdates ON deliveryupdates.order_id = orderdelivery.order_id
WHERE
    deliveryupdates.status <> 'Failed'
        OR 'Returned'
GROUP BY deliverypartners.name
HAVING COUNT(orderdelivery.order_id) > 1;
```

10

List all delivery partners who have completed more than 1 delivery


```sql
SELECT
    customers.name
FROM
    customers
        JOIN
    orders ON customers.customer_id = orders.customer_id
GROUP BY customers.name
HAVING COUNT(DISTINCT orders.order_date) = 3;
```

11

Find the customers who have placed orders on exactly three different days.


```sql
SELECT deliverypartners.partner_id,
       deliverypartners.name,
       COUNT(DISTINCT orders.customer_id) AS diff_customers
FROM deliverypartners
JOIN orderdelivery ON deliverypartners.partner_id = orderdelivery.partner_id
JOIN orders ON orderdelivery.order_id = orders.order_id
GROUP BY deliverypartners.partner_id, deliverypartners.name
ORDER BY diff_customers DESC
LIMIT 1;
```

12

Find the delivery partner who has worked with the most different customers.


```sql
SELECT DISTINCT
    c1.name AS customer1, c2.name AS customer2, c1.city,
    o1.restaurant_id, r.name, DATE(o1.order_date) AS order_date1, DATE(o2.order_date) AS order_date2
FROM customers c1
        JOIN orders o1 ON c1.customer_id = o1.customer_id
        JOIN customers c2 ON c1.city = c2.city
        JOIN orders o2 ON c2.customer_id = o2.customer_id
        JOIN restaurants r ON r.restaurant_id = o1.restaurant_id
WHERE o1.restaurant_id = o2.restaurant_id
        AND DATE(o1.order_date) <> DATE(o2.order_date)
        AND c1.customer_id < c2.customer_id
ORDER BY c1.city , o1.restaurant_id , order_date1;
```

13

Identify customers who have the same city and have placed orders at the same restaurants, but on different dates.

Thankyou

Was it helpful?
Follow for more

Priyansh Bhangalia