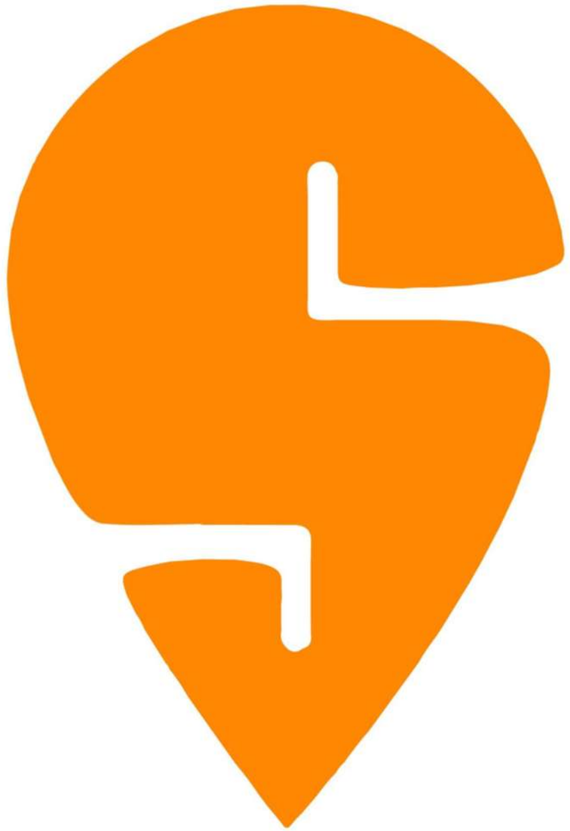


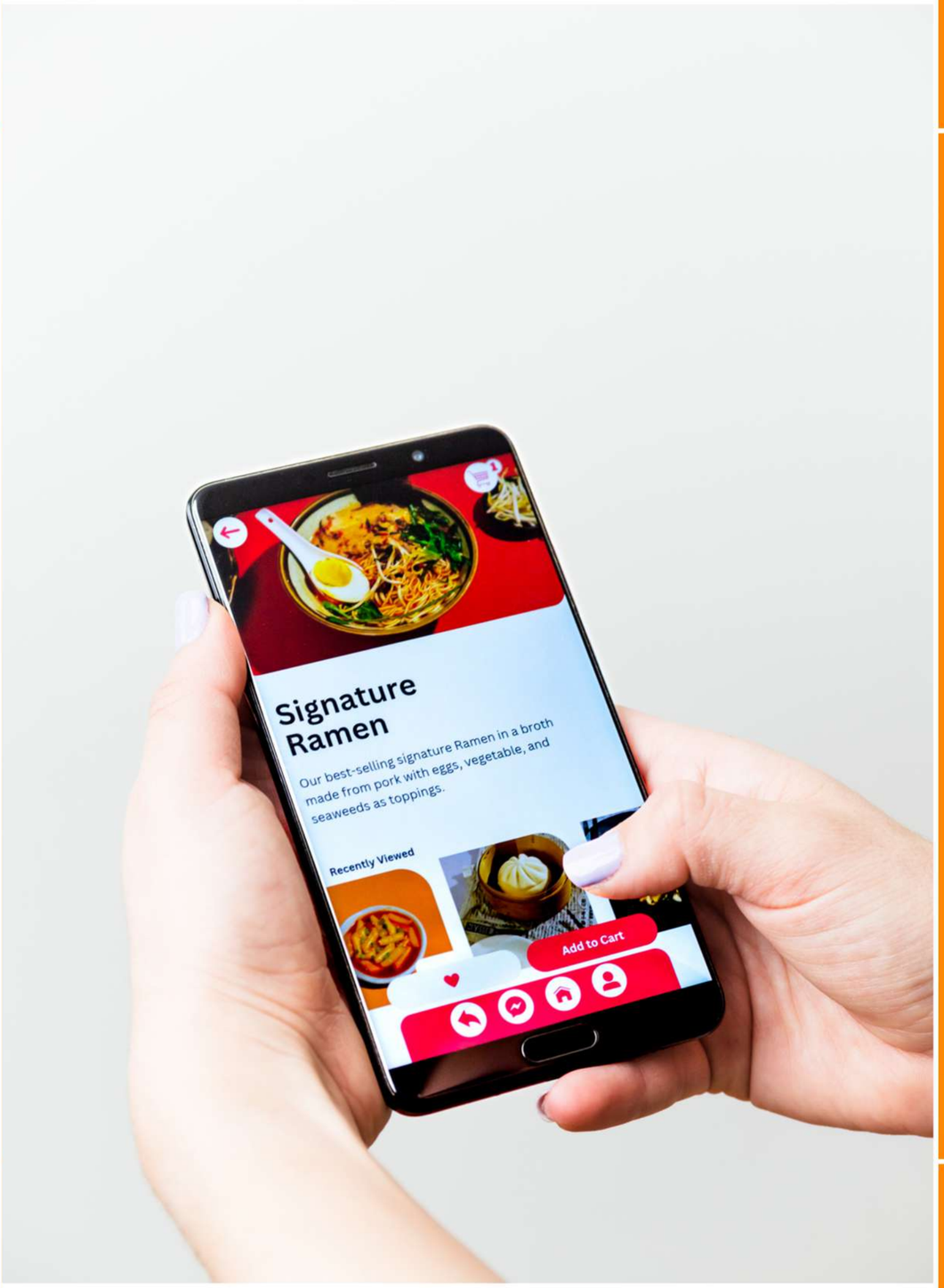
SWIGGY

Home About Us Contact

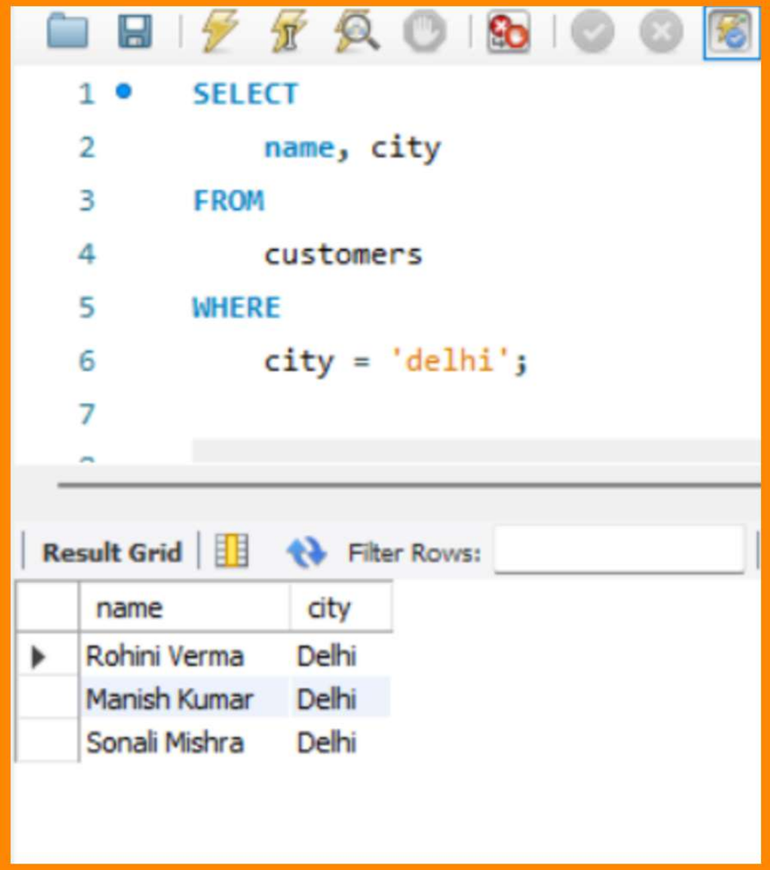
Swiggy is India’s leading online food delivery platform, founded in 2014 and headquartered in Bengaluru. It connects customers with a wide range of restaurants and food outlets through its app, offering fast and reliable doorstep delivery. Over the years, Swiggy has expanded its services to include Instamart for quick grocery delivery and Swiggy Genie for pickup and drop services, becoming a key player in the country's hyperlocal delivery ecosystem.

Project Overview

This project involves analyzing Swiggy's food delivery data using SQL to extract meaningful business insights. By solving real-world queries related to customer orders, restaurant performance, delivery times, and user behavior, the project aims to showcase the power of SQL in handling large datasets and driving data-driven decisions. The objective is to improve operational efficiency and enhance customer experience through structured data analysis.



1. Display all customers who live in 'Delhi'.

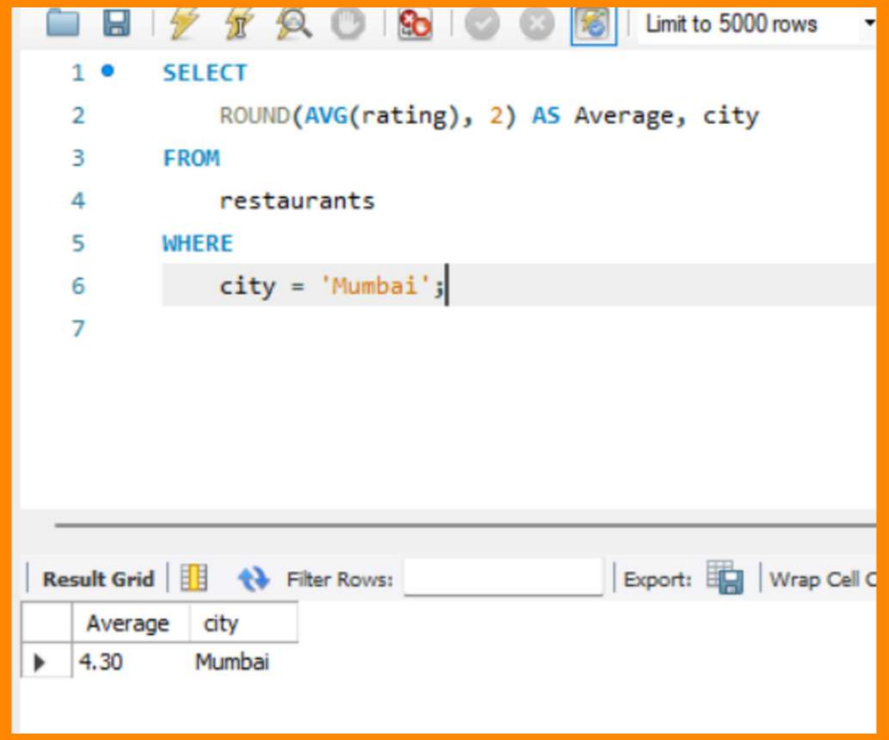


```
SELECT
    name, city
FROM
    customers
WHERE
    city = 'delhi';
```

Result Grid | Filter Rows:

name	city
Rohini Verma	Delhi
Manish Kumar	Delhi
Sonali Mishra	Delhi

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



```
SELECT
    ROUND(AVG(rating), 2) AS Average, city
FROM
    restaurants
WHERE
    city = 'Mumbai';
```

Limit to 5000 rows

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

Average	city
4.30	Mumbai

3.List all customers who have placed at least one order.

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

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

name
Amit Sharma
Rohini Verma
Rajesh Gupta
Sneha Mehta
Manish Kumar
Priya Singh
Vikas Reddy
Anjali Patel
Suresh Nair
Kavita Deshmukh
Vivek Bhatt

4. Display the total number of orders placed by each customer.

```sql
SELECT
    customers.name, count(orders.order_id)AS no_of_orders
FROM
    customers
        left JOIN
    orders ON customers.customer_id = orders.customer_id
    group by  customers.name ;
```

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

name	no_of_orders
Amit Sharma	2
Rohini Verma	3
Rajesh Gupta	3
Sneha Mehta	2
Manish Kumar	4
Priya Singh	3
Vikas Reddy	3
Anjali Patel	3
Suresh Nair	1
Kavita Deshmukh	2
Vivek Bhatt	2

5. Find the total revenue generated by each restaurant.

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

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

name	revenue_generated
Spice of India	1100.00
Tandoori Flames	1200.00
Biryani House	5300.00
Curry Pot	3200.00
Taste of Punjab	600.00
Royal Biryani	650.00
Coastal Delight	2100.00
Veggie Delight	1600.00
Gujarat Express	2550.00
Andhra Spice	4050.00
Punjabi Tadka	900.00

6. Find the top 5 restaurants with the highest average rating.

```sql
SELECT
    name, ROUND(AVG(rating), 2)
FROM
    restaurants
WHERE
    rating IS NOT NULL
GROUP BY name
ORDER BY ROUND(AVG(rating), 2) DESC
LIMIT 5;
```

Result Grid | Filter Rows: | Export: | Wrap

name	round(avg(rating),2)
Biryani House	4.80
Paradise Biryani	4.80
Lucknowi Nawabi	4.70
Royal Biryani	4.70
Flavours of Bengal	4.60

7. Display all customers who have never placed an order.

```sql
SELECT
    customers.customer_id,
    customers.name,
    COALESCE(orders.order_id, 0) AS orders
FROM
    customers
        LEFT JOIN
    orders ON customers.customer_id = orders.customer_id
WHERE
    orders.order_id IS NULL;
```

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

customer_id	name	orders
24	Sonal Kaur	0
25	Vivek Malhotra	0
26	Divya Iyer	0
27	Rakesh Yadav	0
28	Mona Sharma	0
29	Sudha Pillai	0
30	Gaurav Khanna	0

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

```sql
SELECT DISTINCT
    customers.name,
    count(orders.order_id) AS Total_orders,
    customers.city
FROM
    customers
        JOIN
    orders ON customers.customer_id = orders.customer_id
WHERE
    city = 'Mumbai'
GROUP BY customers.name;
```

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

name	Total_orders	city
Amit Sharma	2	Mumbai
Rajesh Gupta	3	Mumbai
Arjun Desai	2	Mumbai
Ravi Singh	2	Mumbai

9. Display all orders placed in the last 30 days.

```sql
SELECT
    *
FROM
    orders
WHERE
    DATEDIFF(NOW(), 30);
```

Result Grid

order_id	customer_id	restaurant_id	order_date	total_amount	status
NULL	NULL	NULL	NULL	NULL	NULL

10. List all delivery partners who have completed more than 1 delivery

```sql
SELECT
    deliverypartners.name,
    COUNT(orderdelivery.partner_id) AS no_of_deliveries
FROM
    deliverypartners
        JOIN
    orderdelivery ON deliverypartners.partner_id = orderdelivery.partner_id
        JOIN
    orders ON orderdelivery.order_id = orders.order_id
WHERE
    orders.status = 'completed'
GROUP BY deliverypartners.name
HAVING COUNT(orderdelivery.partner_id) > 1;
```

Result Grid

name	no_of_deliveries
Suresh Reddy	4
Anita Desai	4
Rajesh Gupta	2
Priya Patel	3
Amit Sharma	2
Sonia Agarwal	3
Ravi Kumar	3
Vikram Singh	2
Mohit Saini	2

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

SQL File 4* SQL File 5* SQL File 6* SQL File 7* SQL File 8* SQL File 9*

Limit to 5000 rows

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

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

name
Anjali Patel
Ashok Kumar
Nidhi Saxena
Priya Singh
Rohini Verma

Result 5

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

Limit to 5000 rows

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

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

name	unique_customer
Suresh Reddy	6

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

SQL File 1* | swiggy | SQL File 2* | SQL File 3* | SQL File 4* | SQL File 5* | SQL File 6* | SQL F

Limit to 5000 rows

```sql
with C1 AS (SELECT
    customers.city,
    customers.customer_id,
    orders.order_date,
    orders.restaurant_id
FROM
    customers
        JOIN
    orders ON customers.customer_id = orders.customer_id),

    C2 AS (SELECT
    customers.city,
    customers.customer_id,
    orders.order_date,
    orders.restaurant_id
FROM
    customers
        JOIN
    orders ON customers.customer_id = orders.customer_id)

 select C1.city,C2.city,
 C1.customer_id,C2.customer_id,
 C1.order_date, C2.order_date,
 C1.restaurant_id, C2.restaurant_id
 from C1 Join C2 on
 C1.customer_id<>C2.customer_id and C1.city= C2.city and
 C1.restaurant_id=C2.restaurant_id and C1.order_date<>C2.order_date;
```

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

city	city	customer_id	customer_id	order_date	order_date	restaurant_id	restaurant_id
Delhi	Delhi	5	18	2024-08-04 00:00:00	2024-08-05 00:00:00	3	3
Delhi	Delhi	18	5	2024-08-05 00:00:00	2024-08-04 00:00:00	3	3
Delhi	Delhi	18	5	2024-08-05 00:00:00	2024-08-07 00:00:00	3	3
Mumbai	Mumbai	19	23	2024-08-03 00:00:00	2024-08-09 00:00:00	8	8
Delhi	Delhi	5	18	2024-08-07 00:00:00	2024-08-05 00:00:00	3	3
Mumbai	Mumbai	23	19	2024-08-09 00:00:00	2024-08-03 00:00:00	8	8

Thank You

PRESENTED BY-
Anshika Tejwani