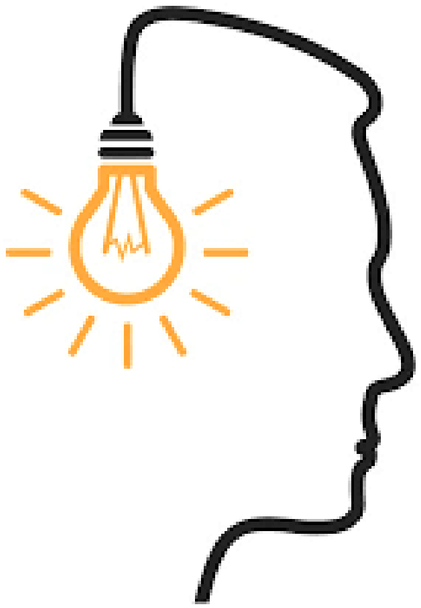
Strategic and In-depth Analysis

SWIGGY

Presented By Pankaj Kumar

Problem Statement



Swiggy, a leading food delivery platform, seeks to enhance its business operations and customer satisfaction by gaining deeper insights from its extensive SQL dataset. The objective is to leverage sophisticated SQL queries and intricate joins to perform in-depth analysis that will support strategic decision-making. The insights sought include understanding customer demographics, restaurant performance, order patterns, and delivery efficiency

SWIGGY

Methodology Used in the Analysis

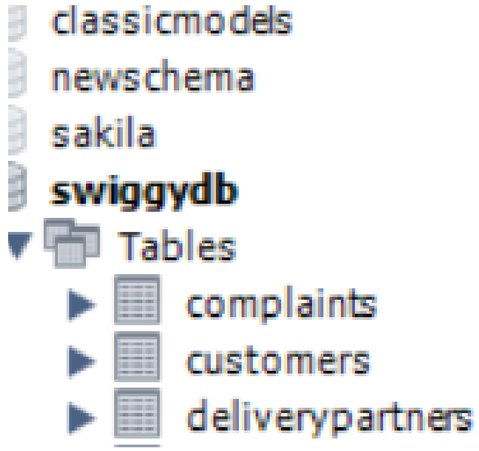
Data Source: Swiggy's Internal sales Database.

Tool: MySQL for Querying data from Database

SWIGGY

Problem: Display all customers who live in 'Delhi'.

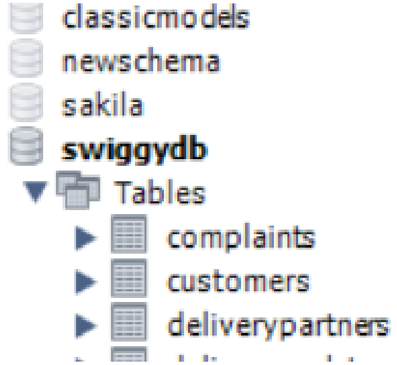
MySql Query for Above problem:




```
3 •    SELECT
4          name
5      FROM
6          customers
7      WHERE
8          city = 'Delhi';
```

Problem: Find the average rating of all restaurants in 'Mumbai'

MySql Query for Above problem:

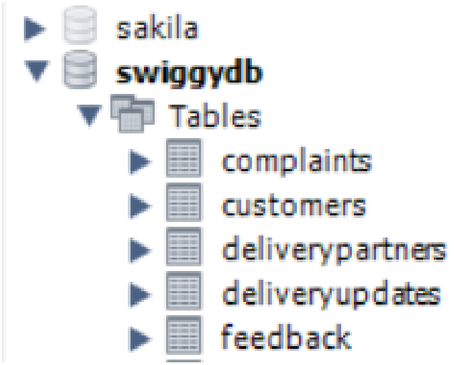



```
3 •  SELECT
4        ROUND(AVG(rating), 2) AS average_rating
5    FROM
6        restaurants
7    WHERE
8        city = 'Mumbai';
```

SWIGGY

Problem: List all customers who have placed at least one order.

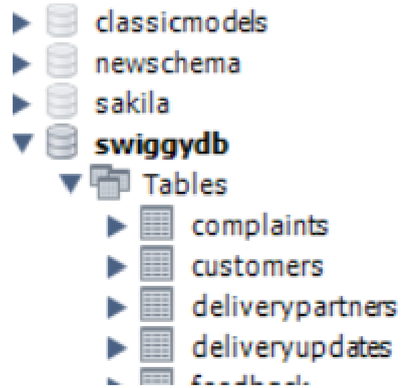
MySql Query for Above problem:




```
13    SELECT DISTINCT
14        customers.name
15    FROM
16        customers
17            INNER JOIN
18        orders ON customers.customer_id = orders.customer_id;
```

Problem: Display the total number of orders placed by each customer.

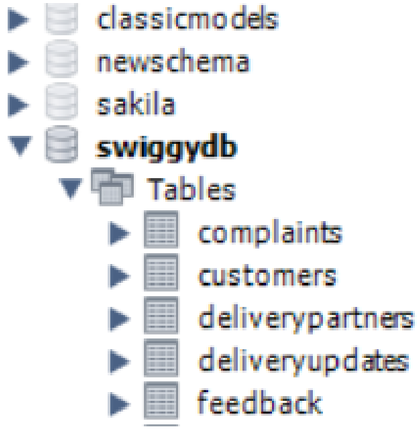
MySql Query for Above problem:




```
33 •  SELECT
34        customers.name, COUNT(orders.order_id)
35    FROM
36        customers
37            LEFT JOIN
38        orders ON customers.customer_id = orders.customer_id
39    GROUP BY customers.name;
```

Problem: Find the total revenue generated by each restaurant

MySql Query for Above problem:

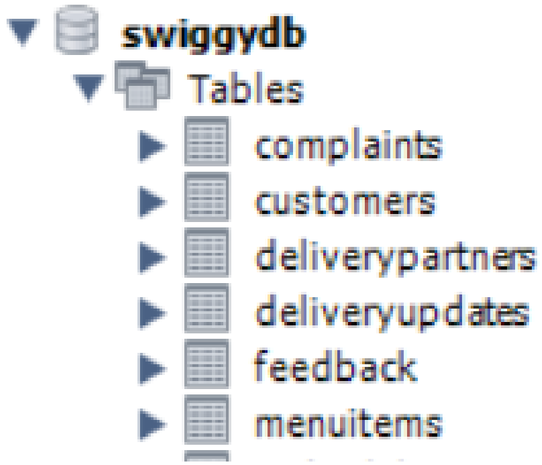



```
11 •    SELECT
12          restaurants.name, SUM(orders.total_amount) AS total_revenue
13      FROM
14          restaurants
15              LEFT JOIN
16          orders ON restaurants.restaurant_id = orders.restaurant_id
17      GROUP BY restaurants.name;
```

SWIGGY

Problem: Find the top 5 restaurants with the highest average rating.

MySql Query for Above problem:

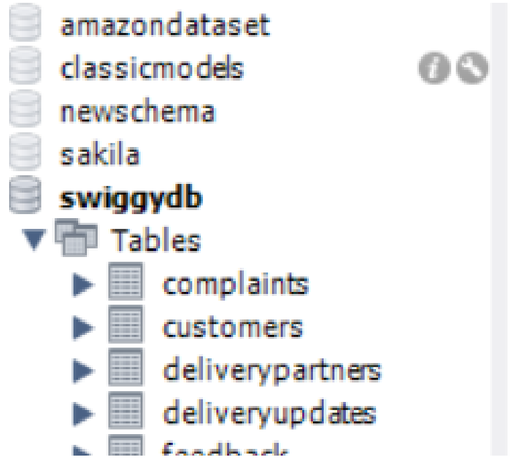



```sql
SELECT
    name, rating
FROM
    restaurants
ORDER BY rating DESC
LIMIT 5;
```

SWIGGY

Problem: Display all customers who have never placed an order

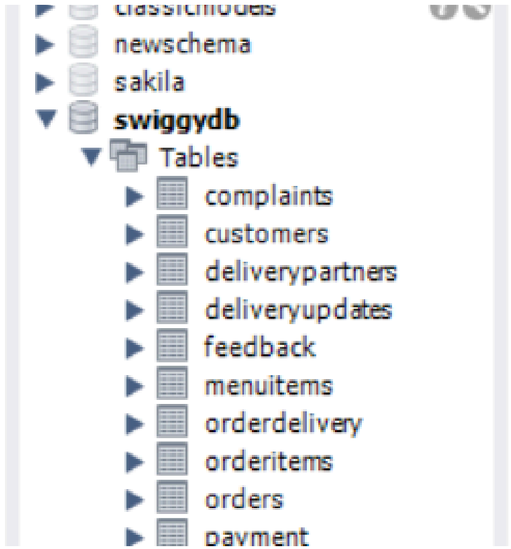
MySql Query for Above problem:




```
41 •  SELECT DISTINCT
42        customers.name
43    FROM
44        customers
45            LEFT JOIN
46        orders ON customers.customer_id = orders.order_id
47    WHERE
48        orders.order_id IS NULL;
```

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

MySql Query for Above problem:

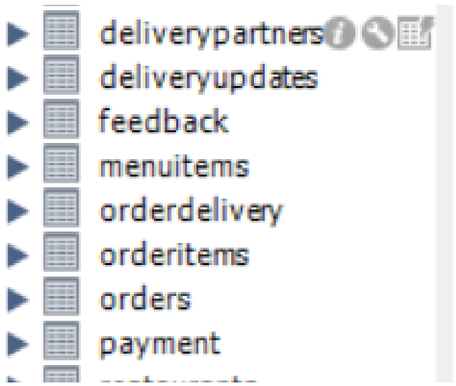



```mysql
SELECT
    customers.name,
    customers.city,
    COUNT(orders.order_id) AS numberoforders
FROM
    customers
        LEFT JOIN
    orders ON customers.customer_id = orders.customer_id
WHERE
    city = 'Mumbai'
GROUP BY customers.name , customers.city;
```

SWIGGY

Problem: Display all orders placed in the last 30 days.

MySql Query for Above problem:

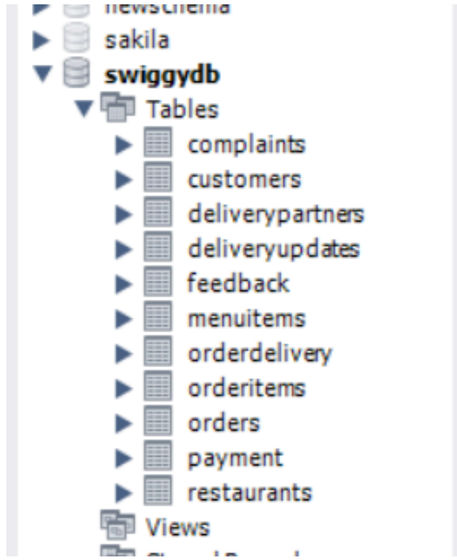



```sql
SELECT
    *
FROM
    orders
WHERE
    order_date >= DATE_SUB(NOW(), INTERVAL 30 DAY);
```

SWIGGY

Problem: List all delivery partners who have completed more than 1 delivery

MySql Query for Above problem:

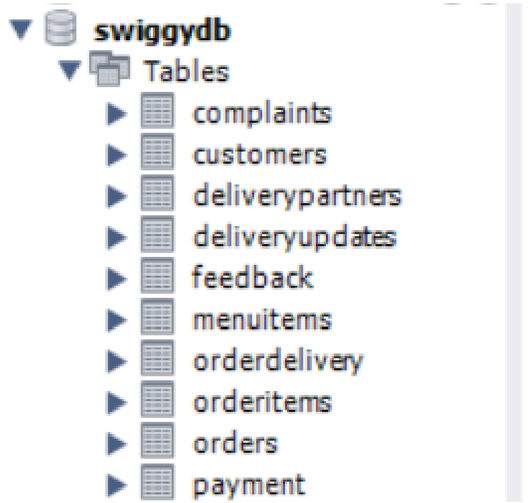



```sql
SELECT
    deliverypartners.name, COUNT(orderdelivery.order_id)
FROM
    deliverypartners
        INNER JOIN
    orderdelivery ON deliverypartners.partner_id = orderdelivery.partner_id
        INNER JOIN
    deliveryupdates ON deliveryupdates.order_id = orderdelivery.order_id
WHERE
    deliveryupdates.status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

SWIGGY

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

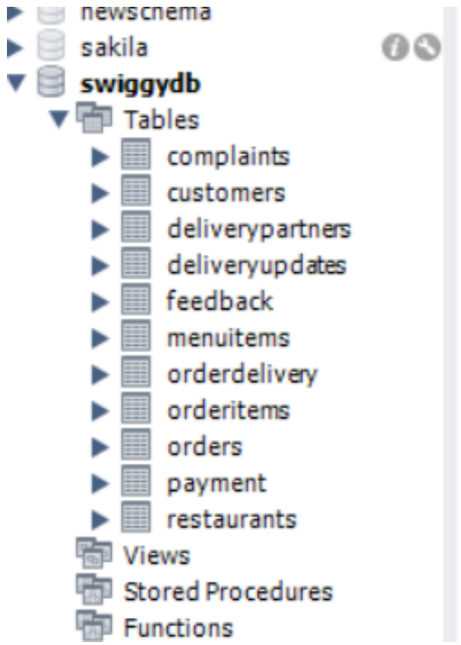
MySql Query for Above problem:




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

Problem: Find the delivery partner who has worked with the most different customers

MySql Query for Above problem:




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

SWIGGY

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

MySql Query for Above problem:

```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;
```

SWIGGY

Conclusion

By using sophisticated SQL queries and joins, Swiggy can gain valuable insights into customer demographics, restaurant performance, order patterns, and delivery efficiency. These insights will enable data-driven decision-making, helping to improve customer satisfaction, optimize operations, and drive business growth.

SWIGGY

Follow for more Data Analytics Insights

www.linkedin.com/in/pankaj-kumar-da