ADAVANCED SQL PROJECT

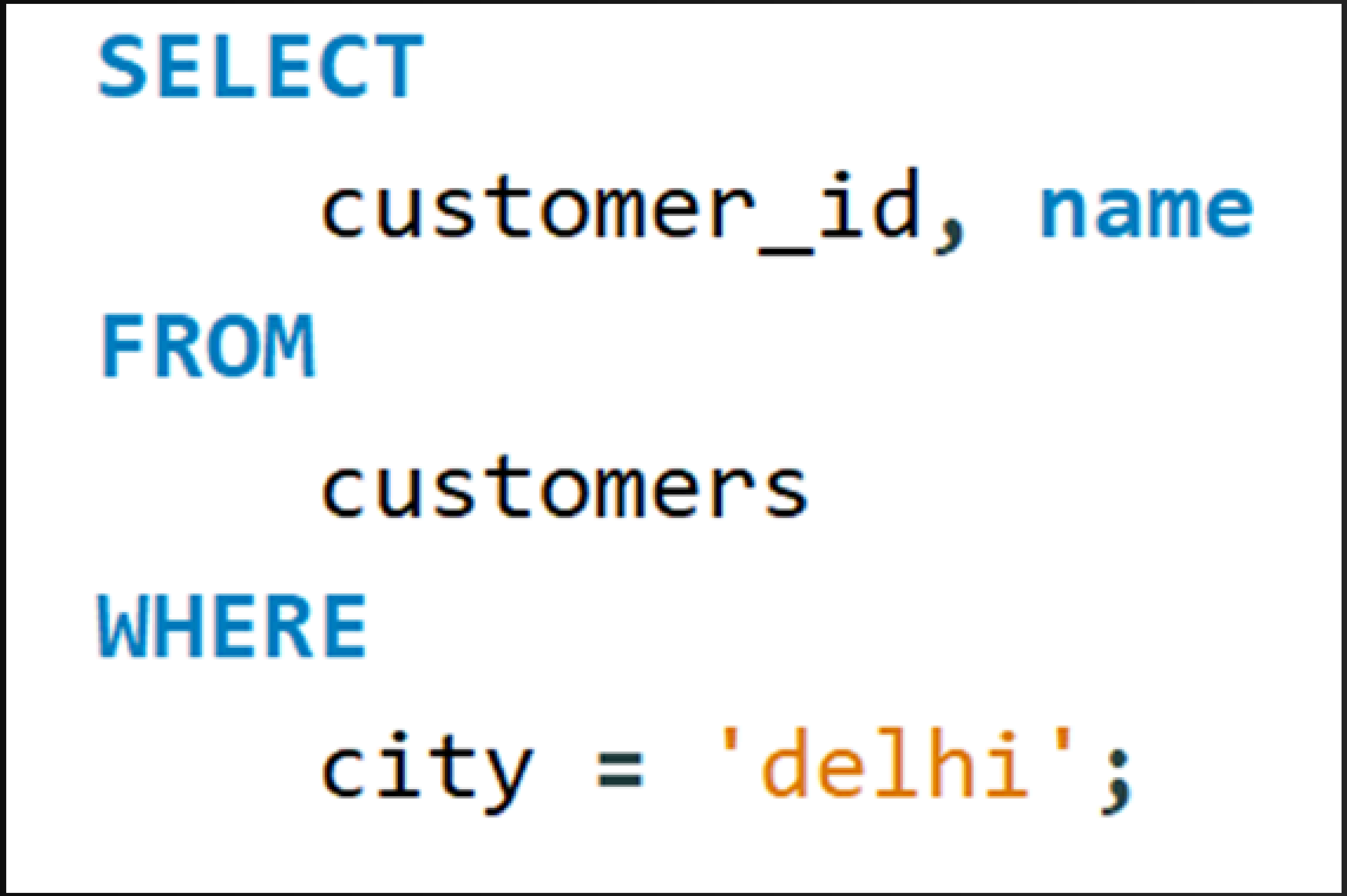
Sangeeta Rajput

Presented by:Sangeeta Rajput

Introduction

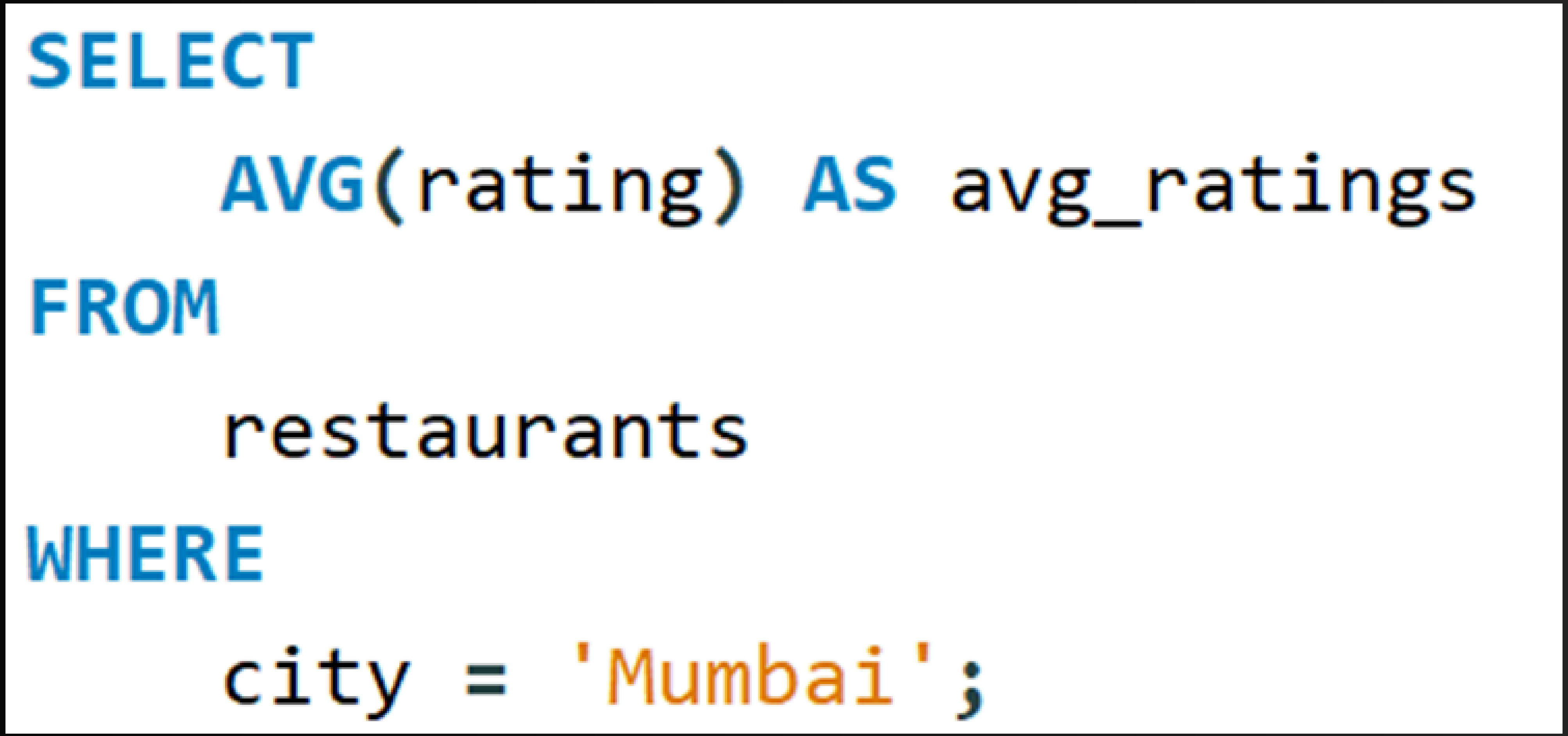
Swiggy is an Indian online food ordering and delivery company. Founded in 2014, Swiggy is headquartered in Bangalore and operates in more than 580 Indian cities, as of July 2023.[4] Besides food delivery, the platform also provides quick commerce services under the name Swiggy Instamart, and same-day package deliveries with Swiggy Genie.

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



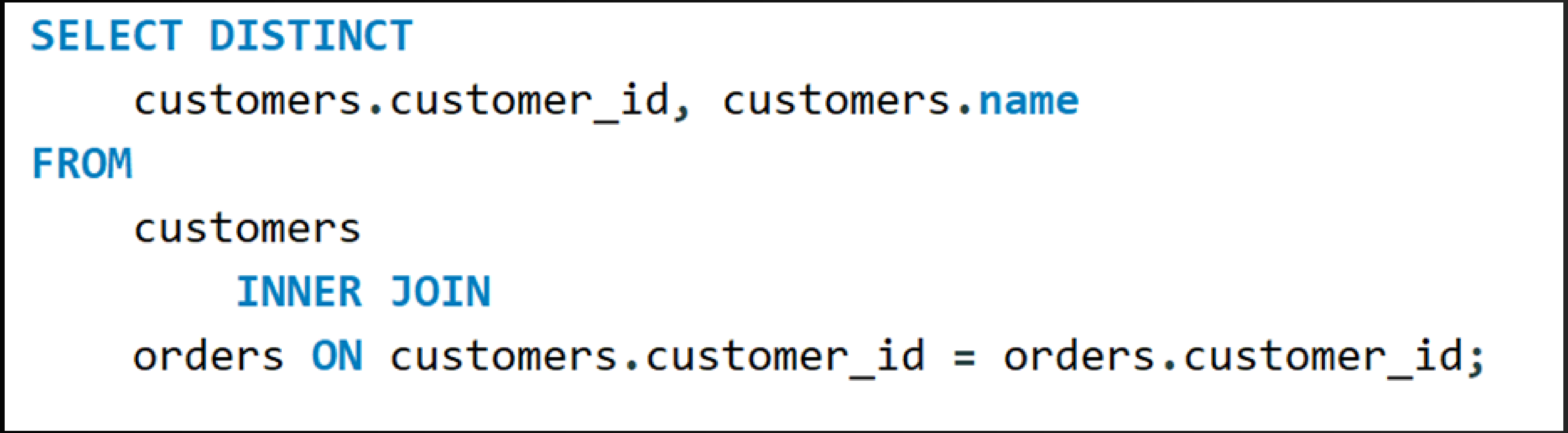
```sql
SELECT
    customer_id, name
FROM
    customers
WHERE
    city = 'delhi';
```

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



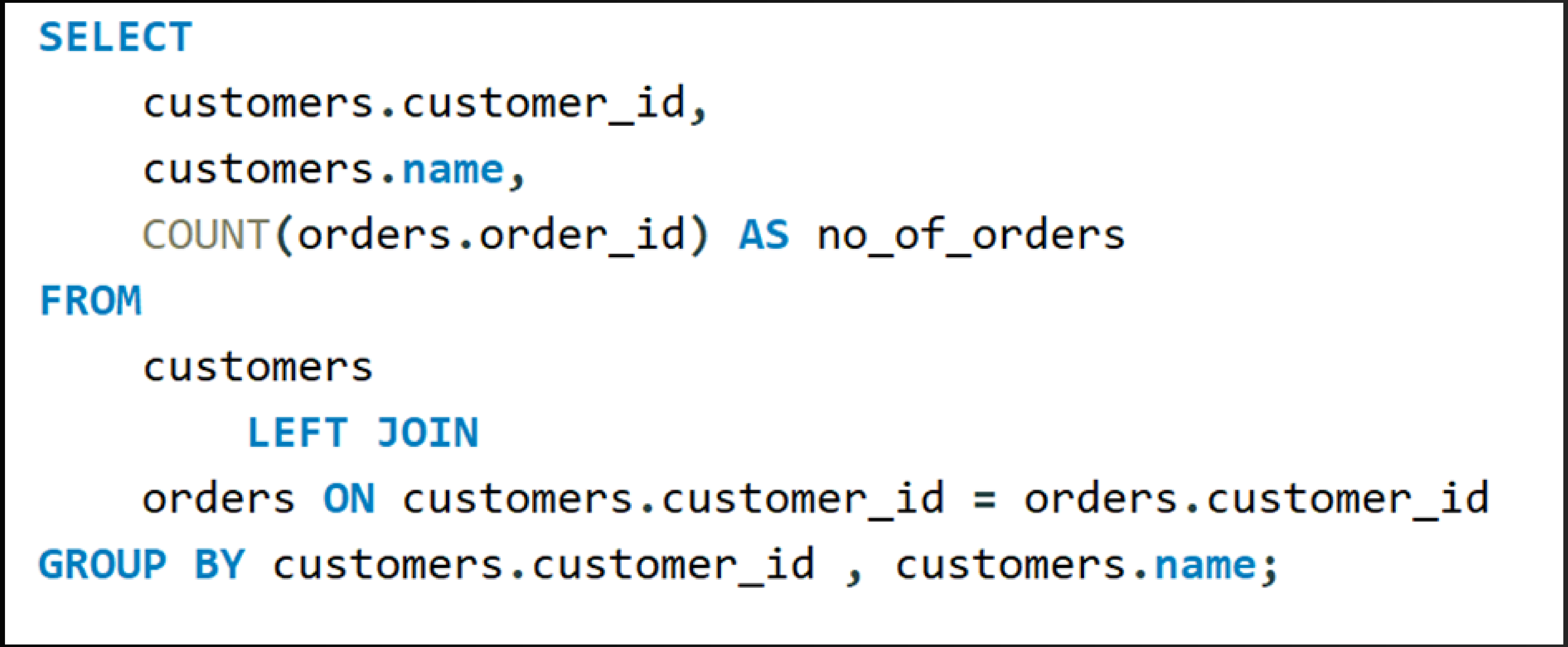
```sql
SELECT
    AVG(rating) AS avg_ratings
FROM
    restaurants
WHERE
    city = 'Mumbai';
```

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



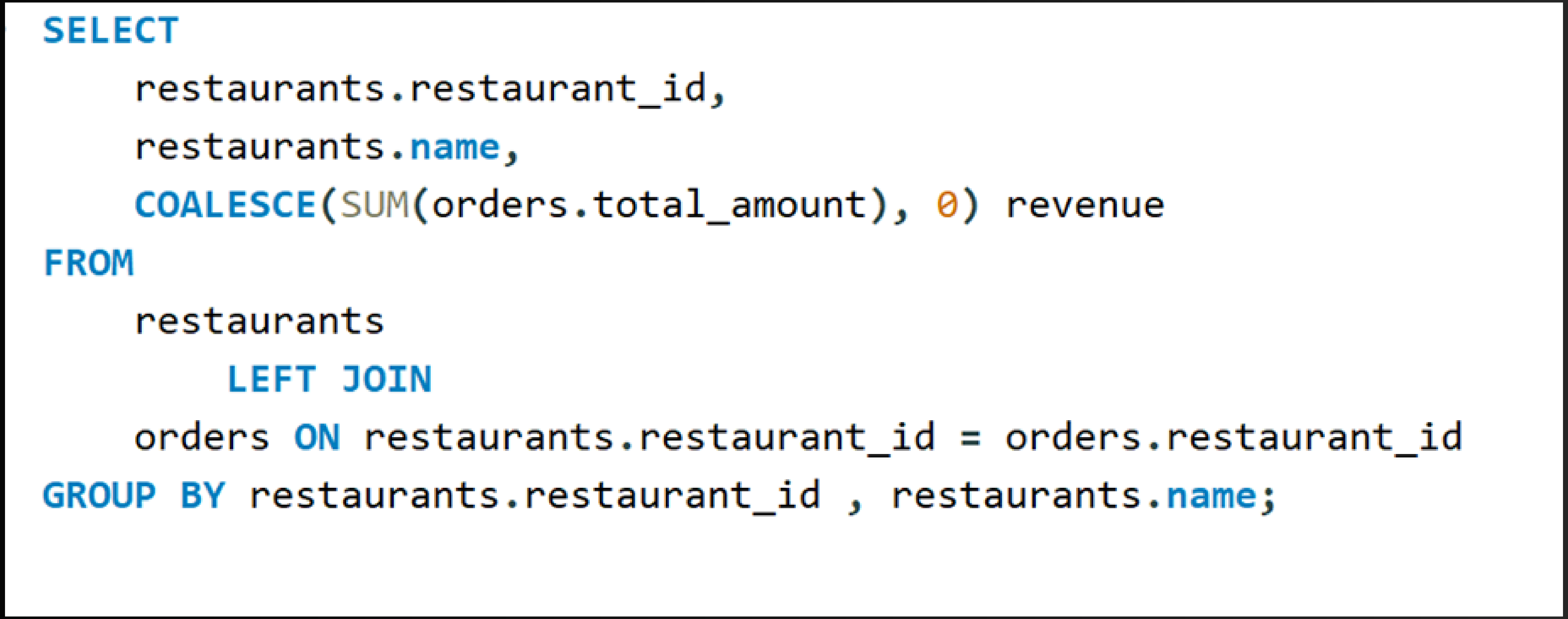
```sql
SELECT DISTINCT
    customers.customer_id, customers.name
FROM
    customers
        INNER JOIN
    orders ON customers.customer_id = orders.customer_id;
```

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



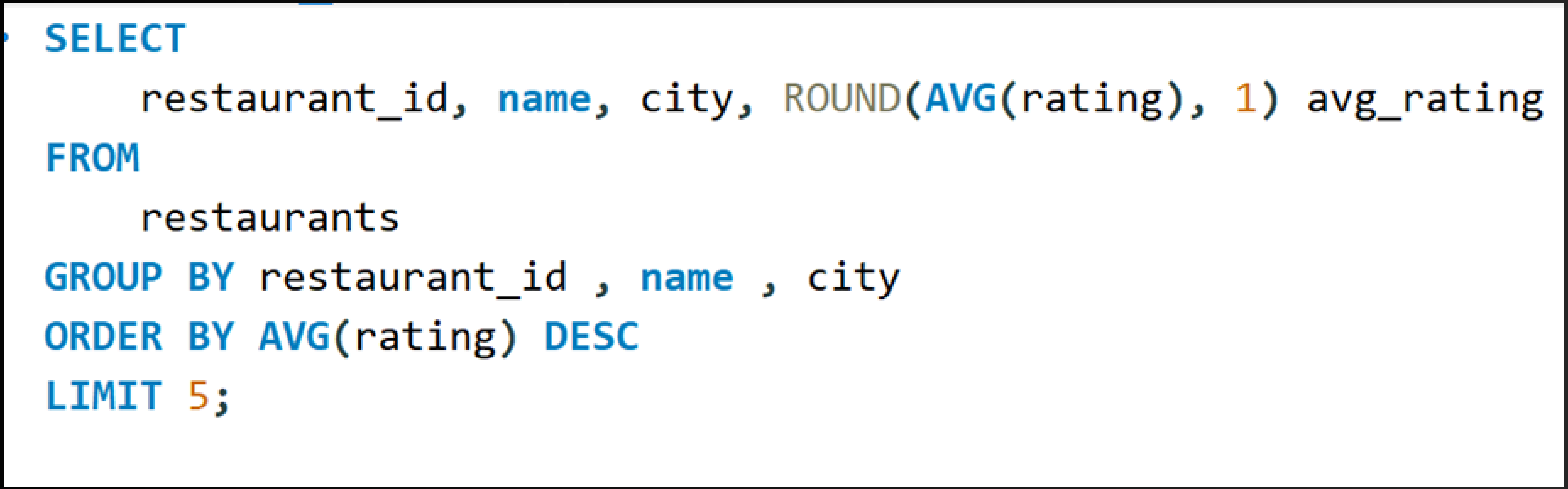
```sql
SELECT
    customers.customer_id,
    customers.name,
    COUNT(orders.order_id) AS no_of_orders
FROM
    customers
        LEFT JOIN
    orders ON customers.customer_id = orders.customer_id
GROUP BY customers.customer_id , customers.name;
```

5.Find the total revenue generated by each restaurant.



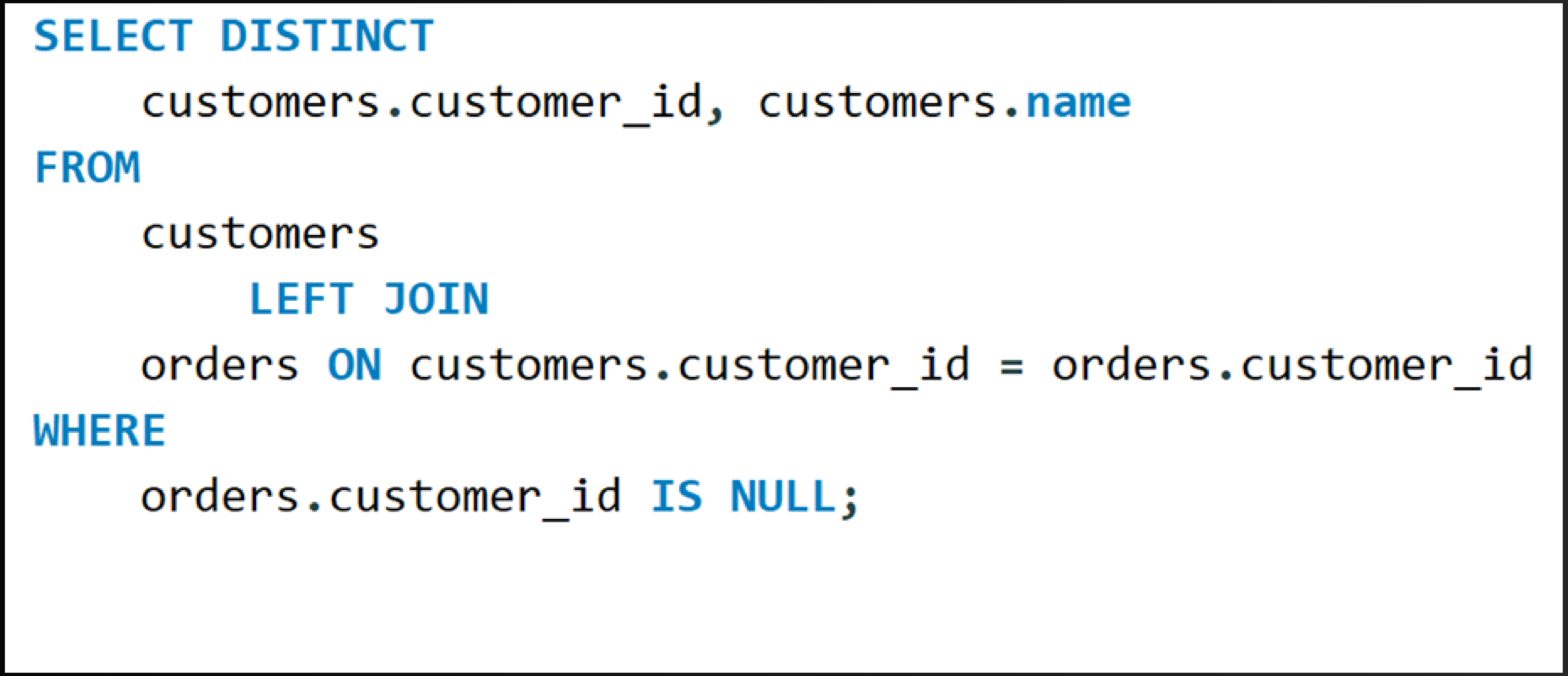
```sql
SELECT
    restaurants.restaurant_id,
    restaurants.name,
    COALESCE(SUM(orders.total_amount), 0) revenue
FROM
    restaurants
        LEFT JOIN
    orders ON restaurants.restaurant_id = orders.restaurant_id
GROUP BY restaurants.restaurant_id , restaurants.name;
```

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



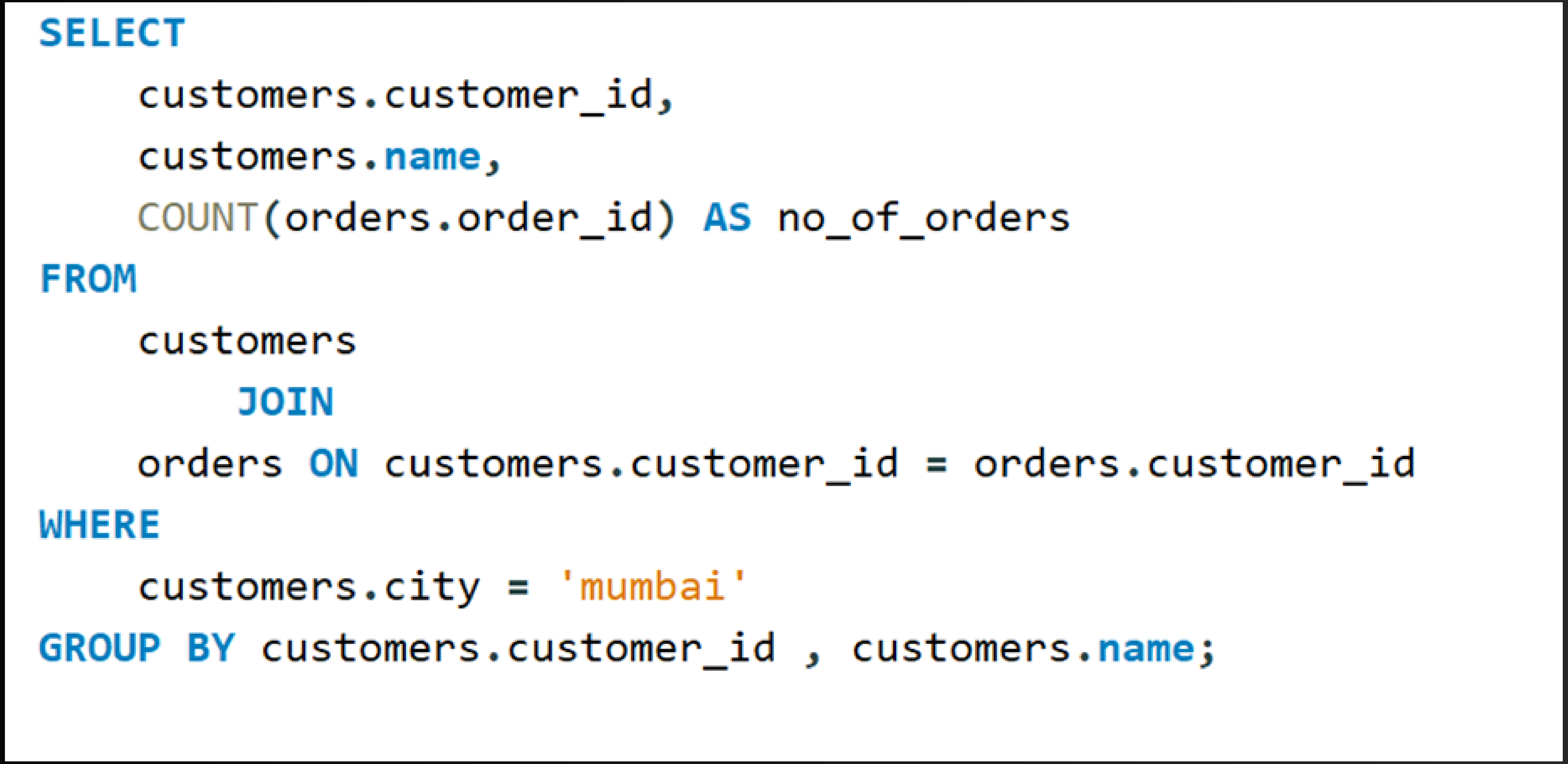
```sql
SELECT
    restaurant_id, name, city, ROUND(AVG(rating), 1) avg_rating
FROM
    restaurants
GROUP BY restaurant_id , name , city
ORDER BY AVG(rating) DESC
LIMIT 5;
```

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



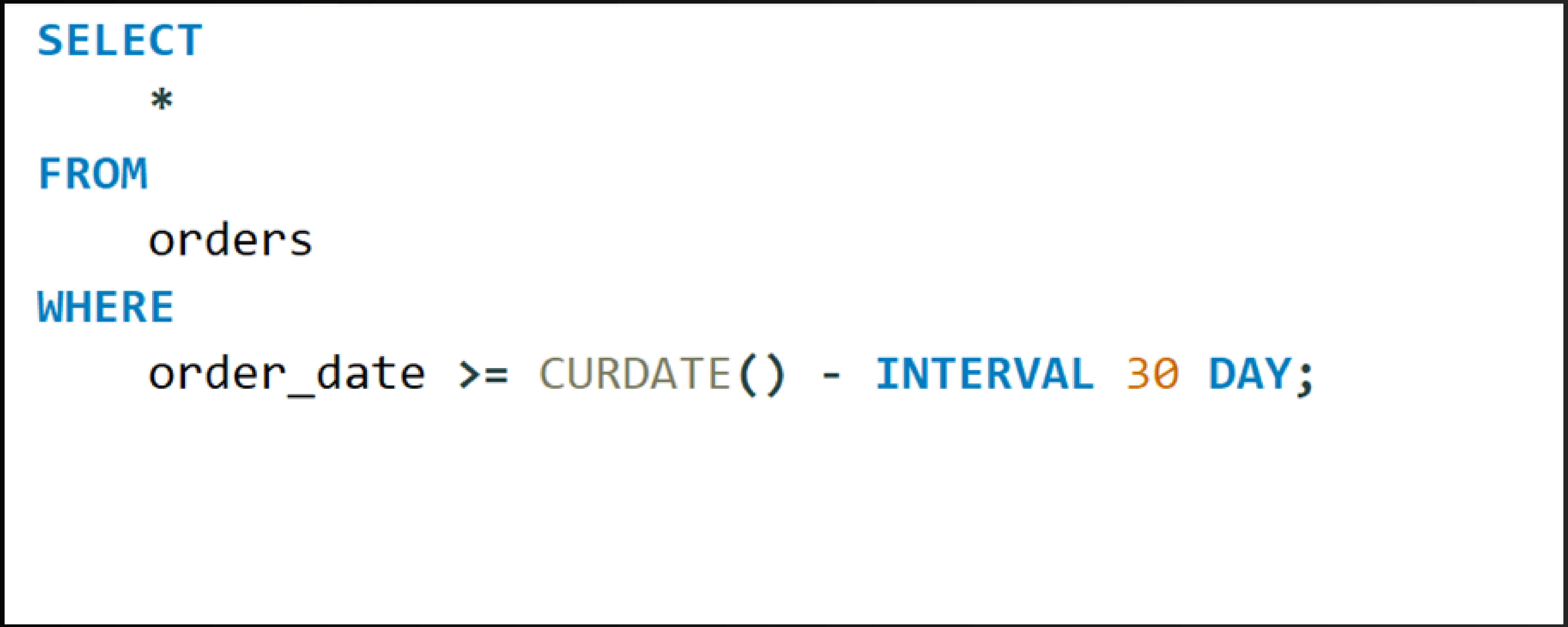
```sql
SELECT DISTINCT
    customers.customer_id, customers.name
FROM
    customers
        LEFT JOIN
    orders ON customers.customer_id = orders.customer_id
WHERE
    orders.customer_id IS NULL;
```

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



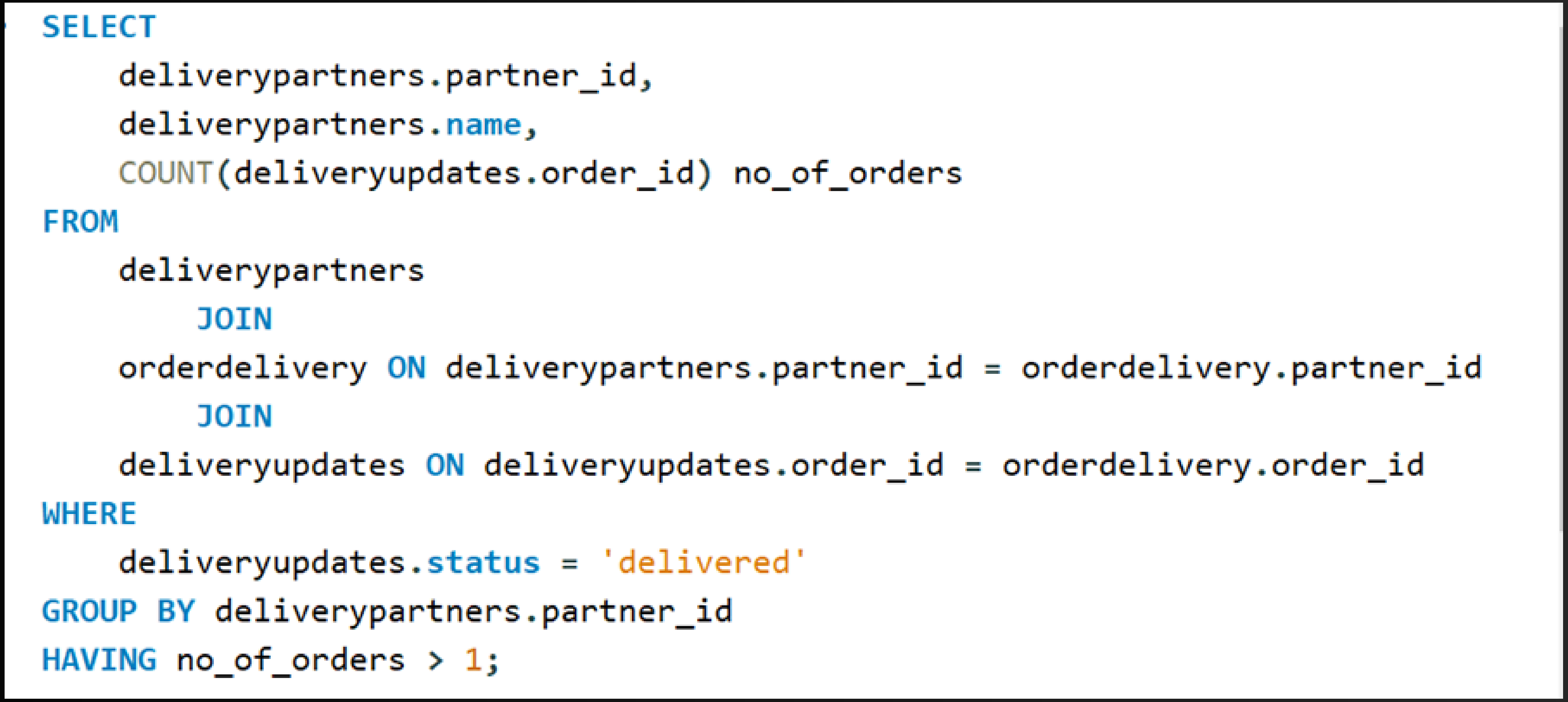
```
SELECT
    customers.customer_id,
    customers.name,
    COUNT(orders.order_id) AS no_of_orders
FROM
    customers
        JOIN
    orders ON customers.customer_id = orders.customer_id
WHERE
    customers.city = 'mumbai'
GROUP BY customers.customer_id , customers.name;
```

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



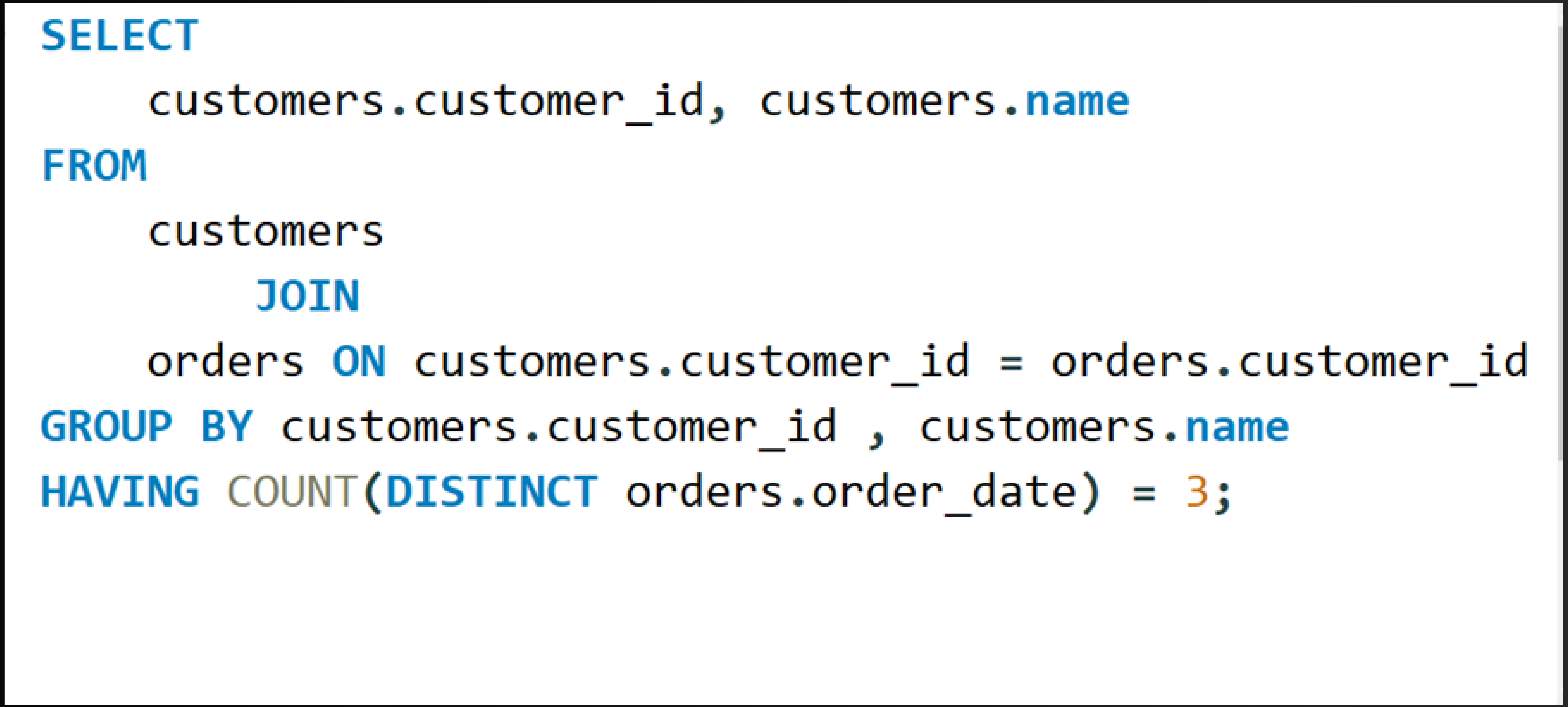
```sql
SELECT
    *
FROM
    orders
WHERE
    order_date >= CURDATE() - INTERVAL 30 DAY;
```

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



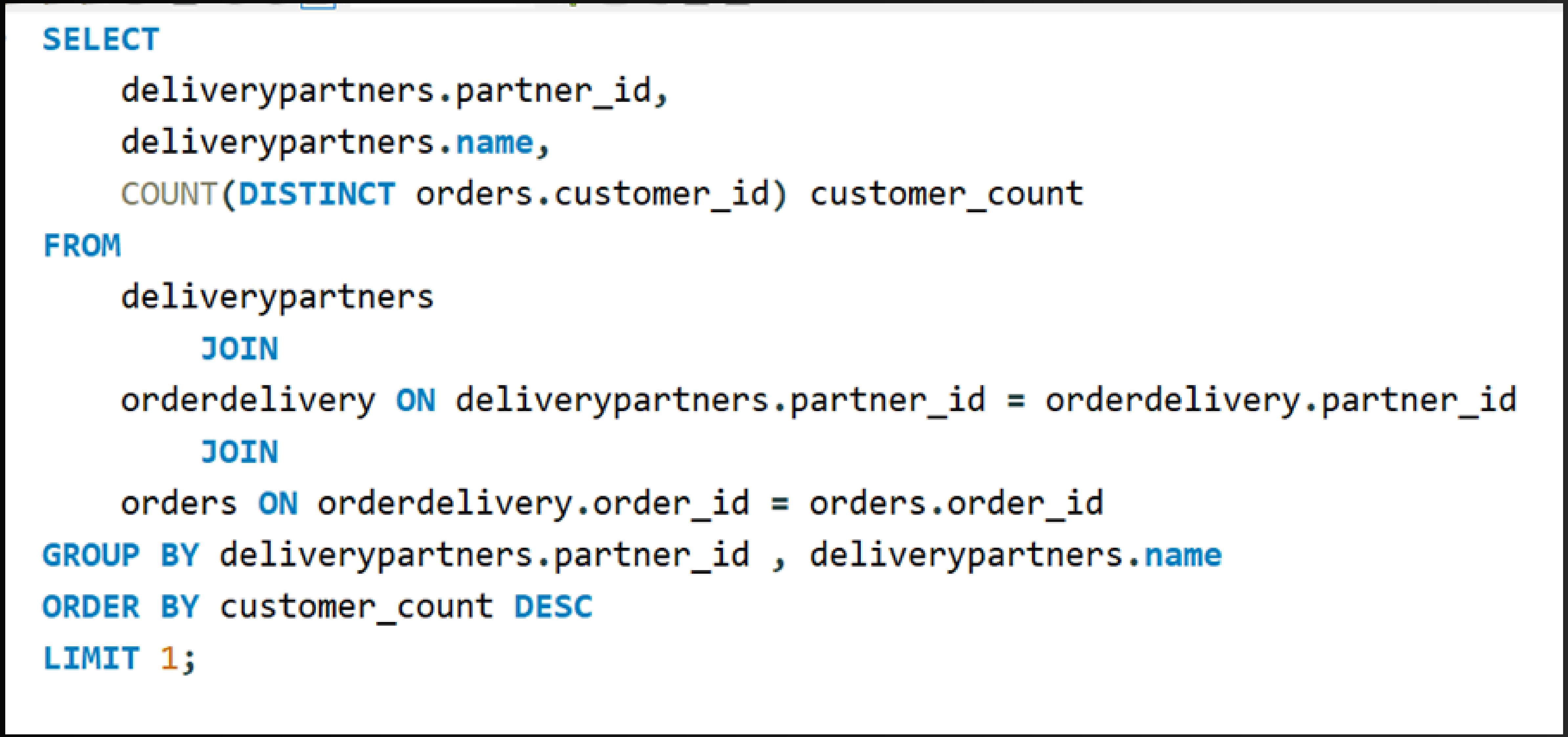
```sql
SELECT
    deliverypartners.partner_id,
    deliverypartners.name,
    COUNT(deliveryupdates.order_id) no_of_orders
FROM
    deliverypartners
        JOIN
    orderdelivery ON deliverypartners.partner_id = orderdelivery.partner_id
        JOIN
    deliveryupdates ON deliveryupdates.order_id =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

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



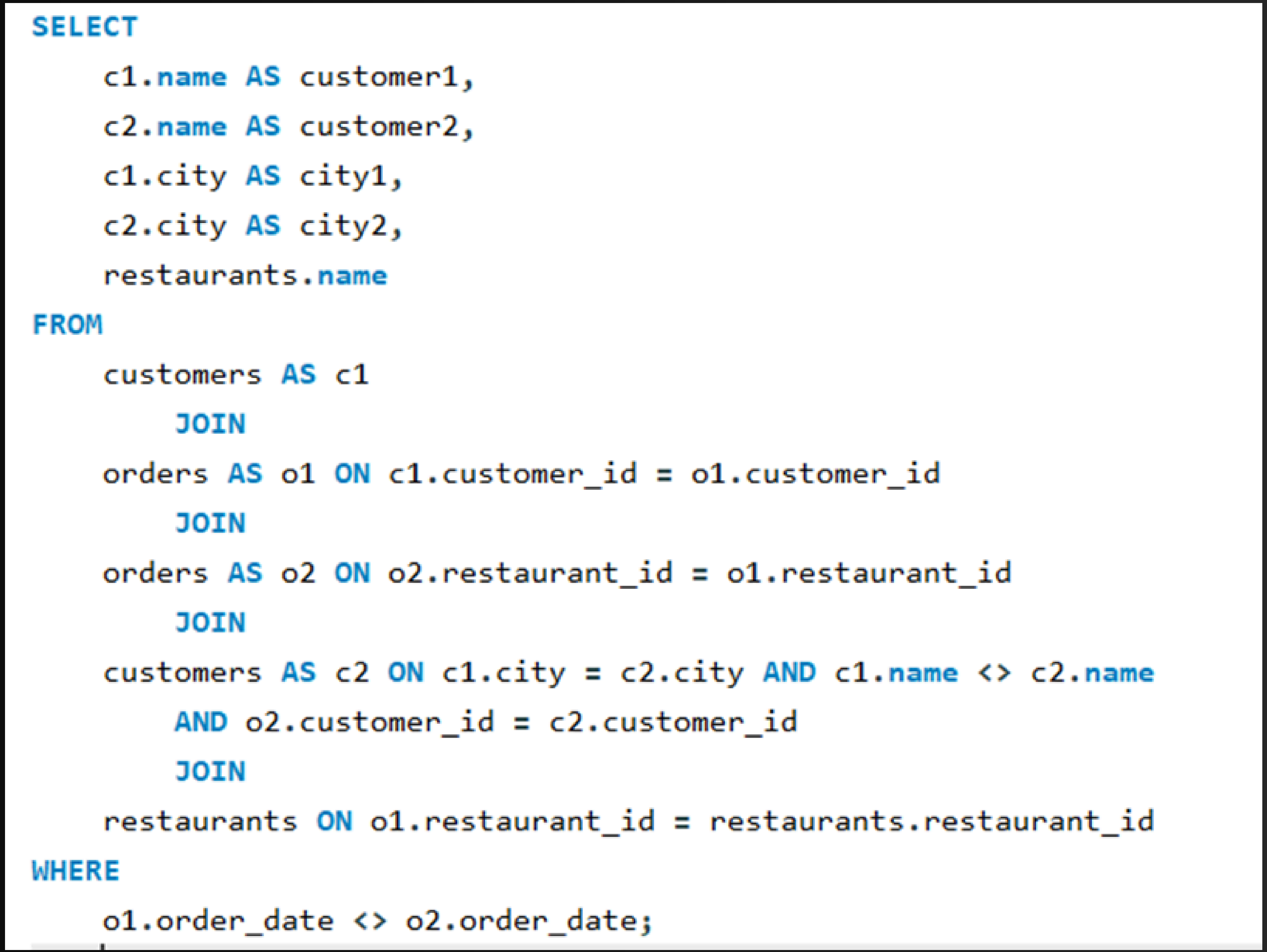
```sql
SELECT
    customers.customer_id, customers.name
FROM
    customers
        JOIN
    orders ON customers.customer_id = orders.customer_id
GROUP BY customers.customer_id , customers.name
HAVING COUNT(DISTINCT orders.order_date) = 3;
```

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



```
SELECT
    deliverypartners.partner_id,
    deliverypartners.name,
    COUNT(DISTINCT orders.customer_id) customer_count
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 customer_count DESC
LIMIT 1;
```

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



```sql
SELECT
    c1.name AS customer1,
    c2.name AS customer2,
    c1.city AS city1,
    c2.city AS city2,
    restaurants.name
FROM
    customers AS c1
        JOIN
    orders AS o1 ON c1.customer_id = o1.customer_id
        JOIN
    orders AS o2 ON o2.restaurant_id = o1.restaurant_id
        JOIN
    customers AS c2 ON c1.city = c2.city AND c1.name <> c2.name
        AND o2.customer_id = c2.customer_id
        JOIN
    restaurants ON o1.restaurant_id = restaurants.restaurant_id
WHERE
    o1.order_date <> o2.order_date;
```

THANK YOU

Like, Comment, Share, and Follow

Sangeeta Rajput

SWIGGY