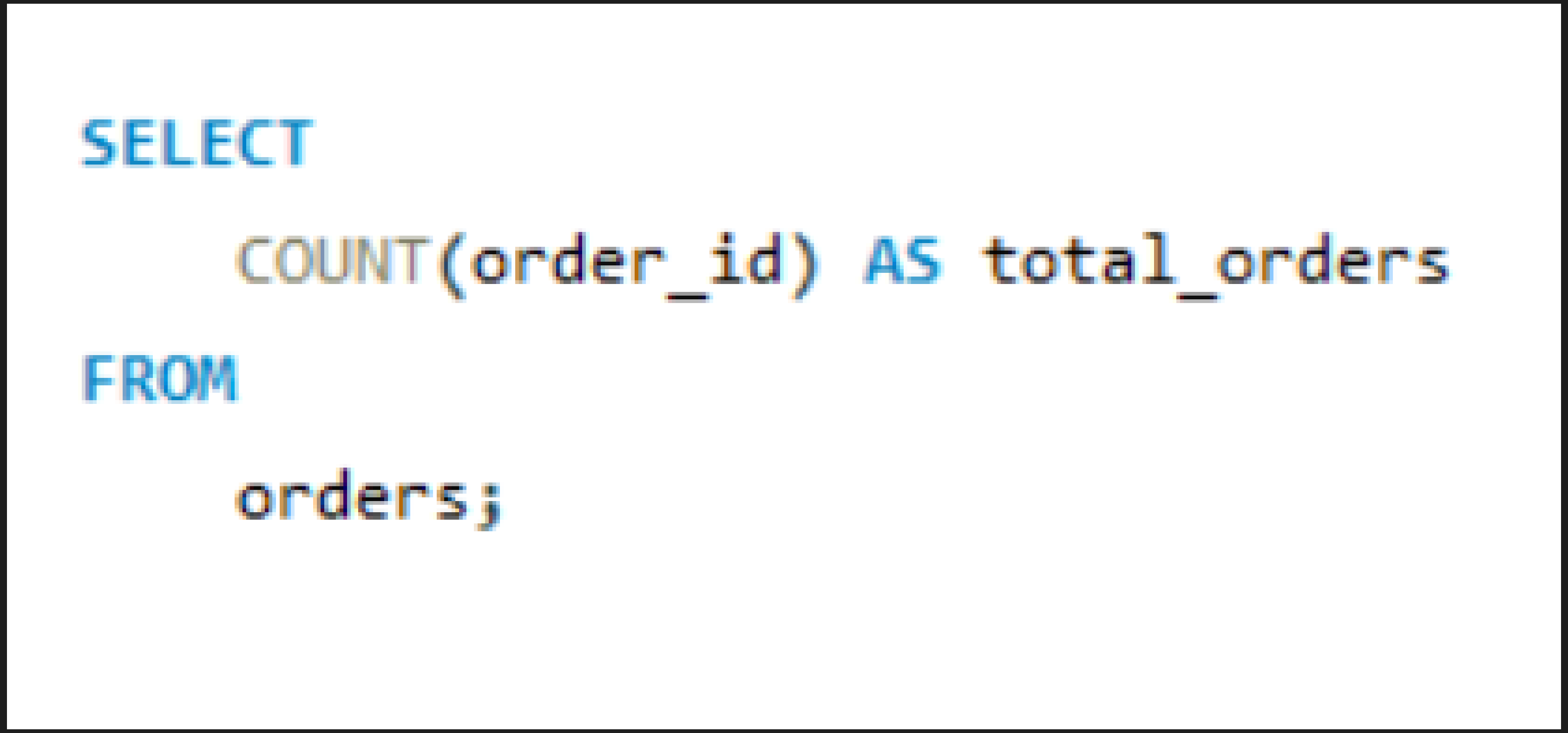
SQL PROJECT: PIZZA SALES DATA ANALYSIS

-Saurabh Kumar Yadav

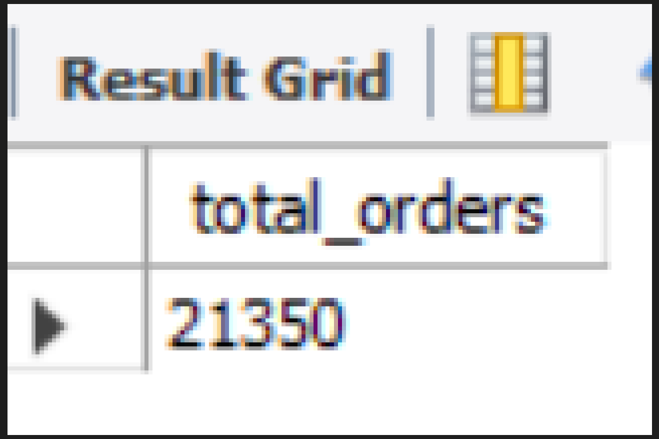
INTRODUCTION

- THIS PROJECT DEMONSTRATES THE USE OF SQL QUERIES TO ANALYZE PIZZA SALES DATA.
- A TOTAL OF 10 QUERIES WERE WRITTEN, COVERING EASY, INTERMEDIATE, AND ADVANCED DIFFICULTY LEVELS.
- KEY SQL CONCEPTS USED:
  - JOINS (TO COMBINE DATA ACROSS MULTIPLE TABLES)
  - SUBQUERIES (TO HANDLE COMPLEX ANALYTICAL REQUIREMENTS)
- THE QUERIES AIM TO EXTRACT MEANINGFUL INSIGHTS SUCH AS:
  - SALES TRENDS
  - CUSTOMER PREFERENCES
  - REVENUE PATTERNS
- THIS PROJECT HIGHLIGHTS THE PRACTICAL APPLICATION OF SQL IN SOLVING REAL-WORLD BUSINESS PROBLEMS THROUGH DATA ANALYSIS.

1- RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED

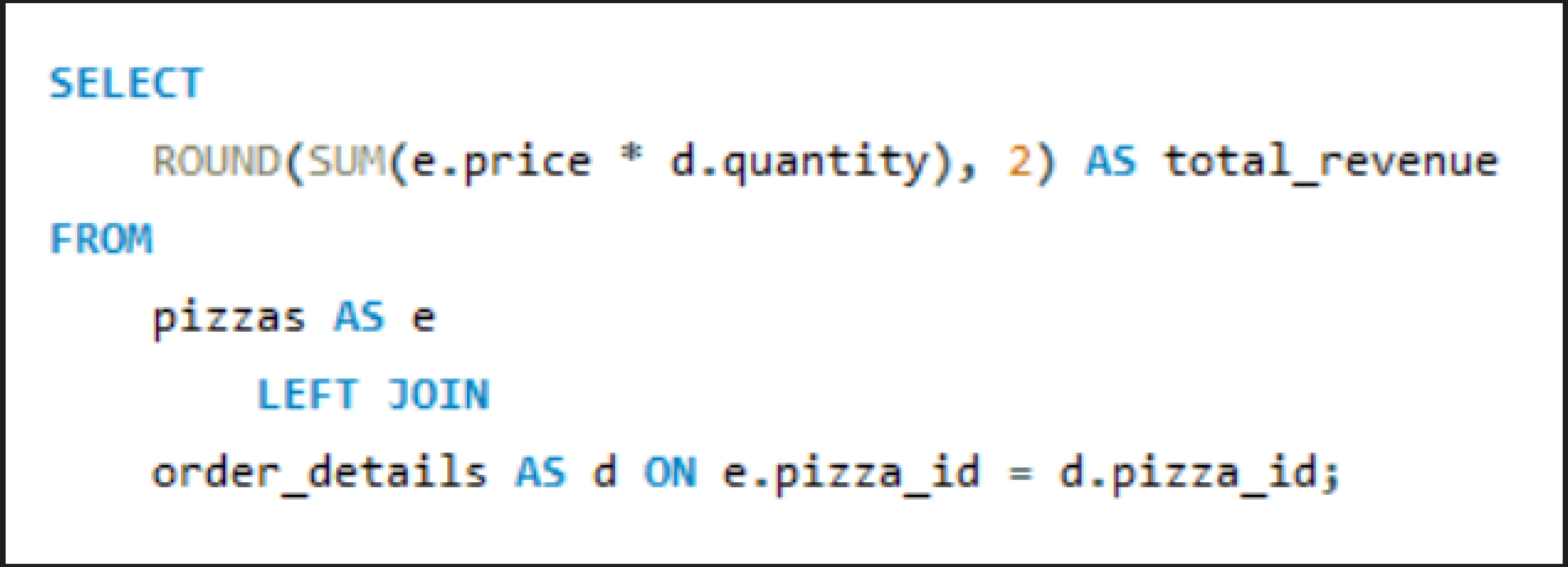


```sql
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

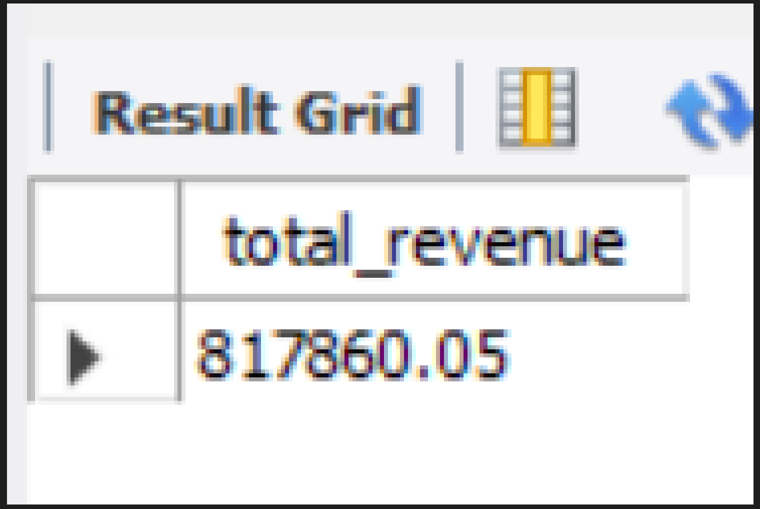


```
total_orders
21350
```

2- CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES

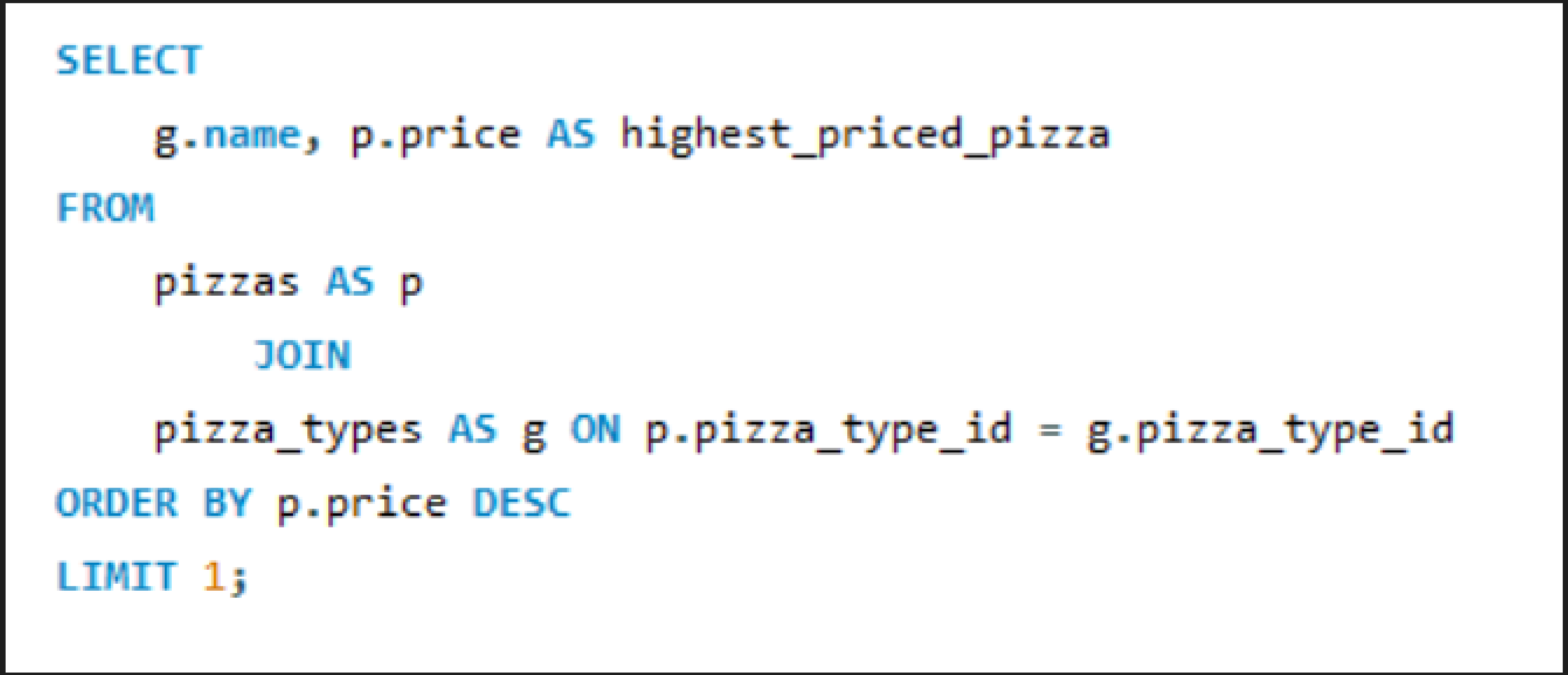


```
SELECT
    ROUND(SUM(e.price * d.quantity), 2) AS total_revenue
FROM
    pizzas AS e
        LEFT JOIN
    order_details AS d ON e.pizza_id = d.pizza_id;
```

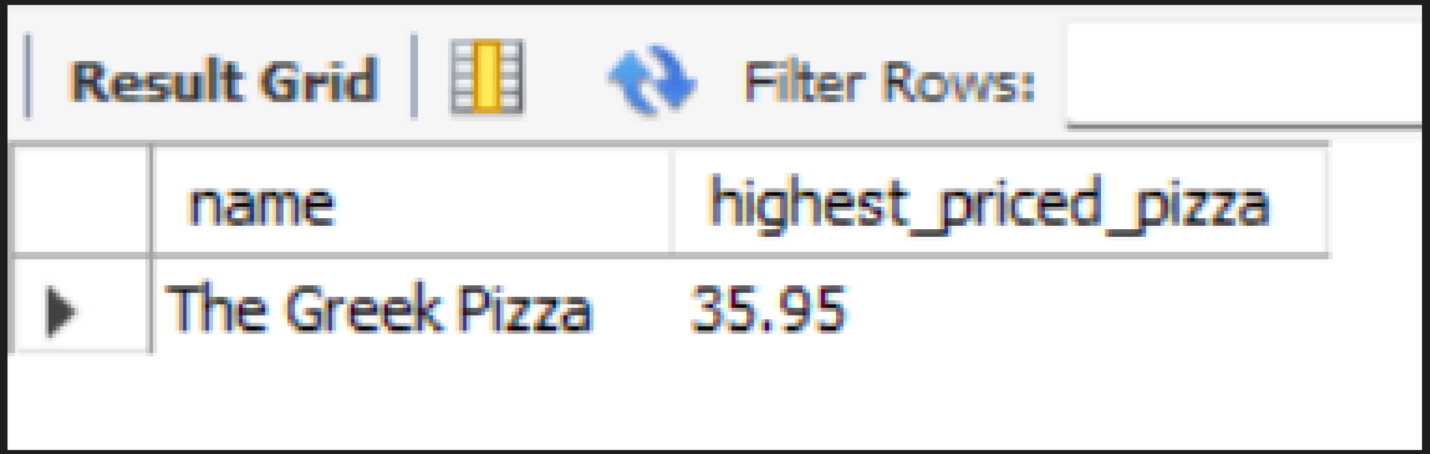


```
total_revenue
817860.05
```

3- IDENTIFY THE HIGHEST-PRICED PIZZA

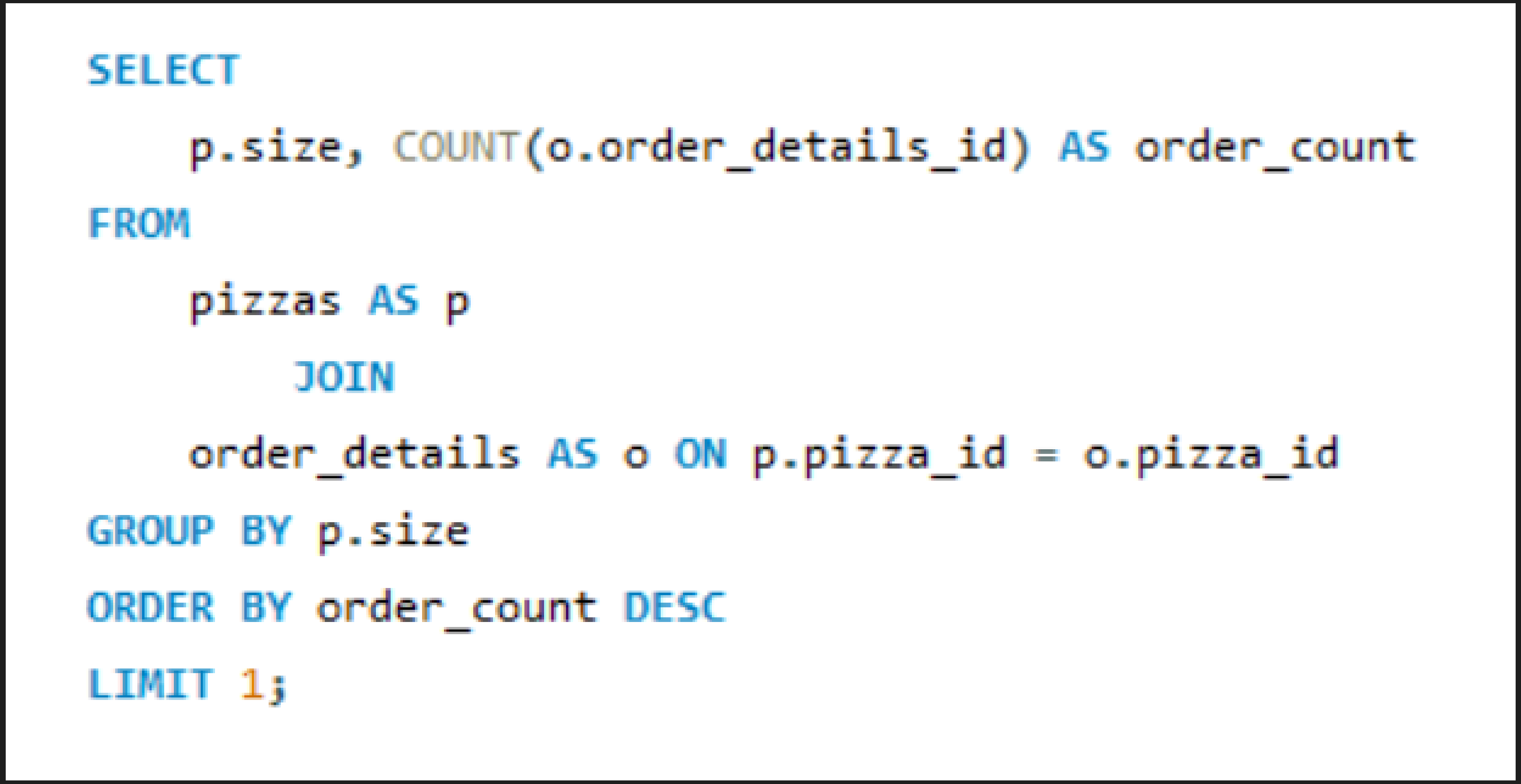


```
SELECT
    g.name, p.price AS highest_priced_pizza
FROM
    pizzas AS p
        JOIN
    pizza_types AS g ON p.pizza_type_id = g.pizza_type_id
ORDER BY p.price DESC
LIMIT 1;
```

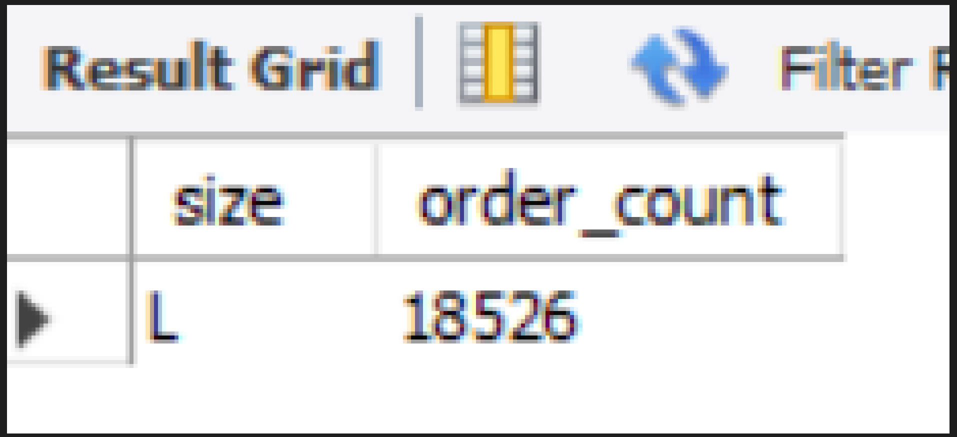


name	highest_priced_pizza
The Greek Pizza	35.95

4- IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED



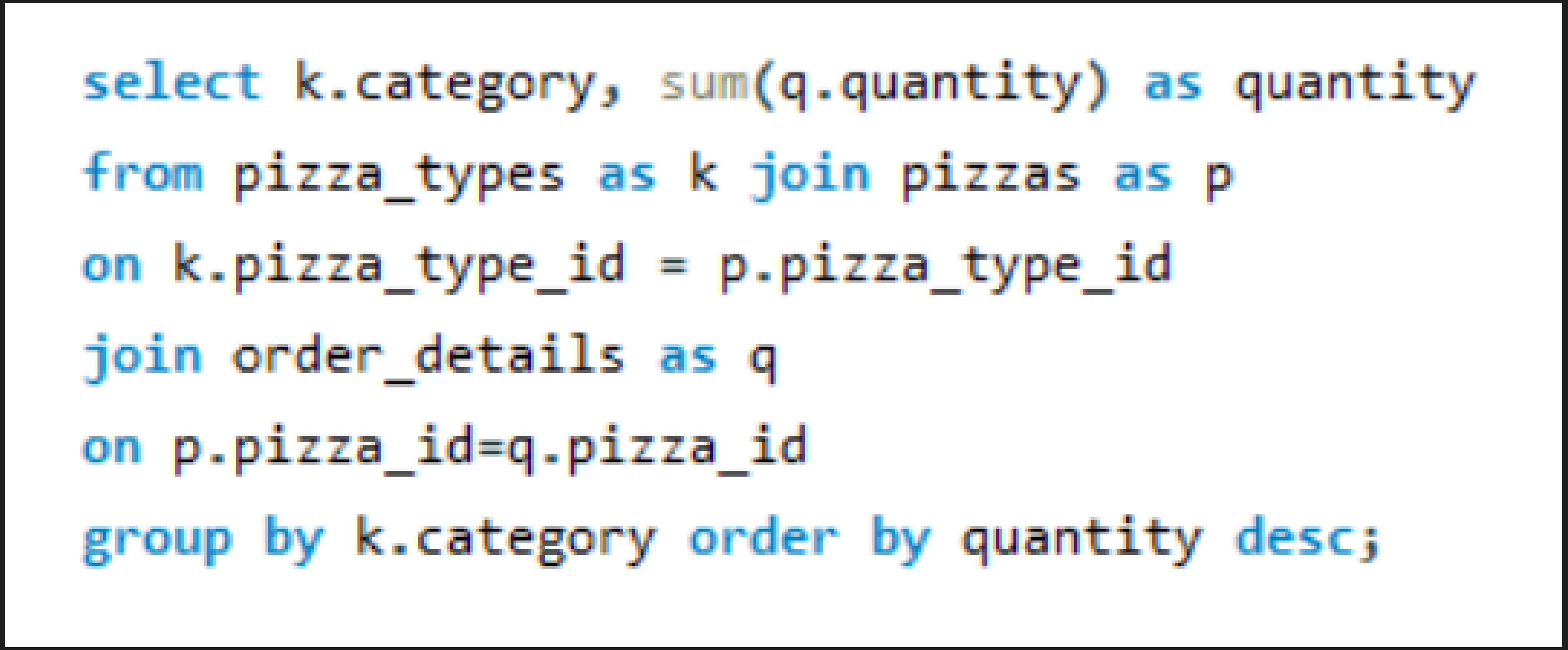
```
SELECT
    p.size, COUNT(o.order_details_id) AS order_count
FROM
    pizzas AS p
        JOIN
    order_details AS o ON p.pizza_id = o.pizza_id
GROUP BY p.size
ORDER BY order_count DESC
LIMIT 1;
```



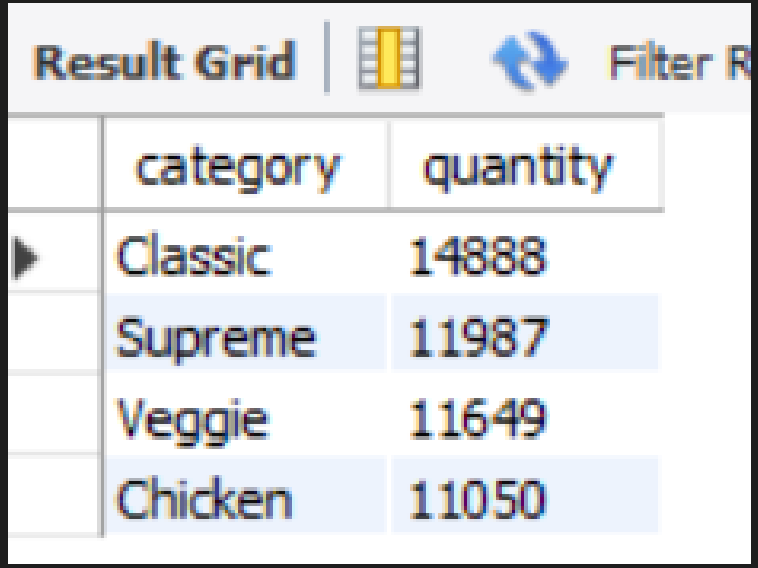
Result Grid | Filter R

```
size  order_count
L     18526
```

5- JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED

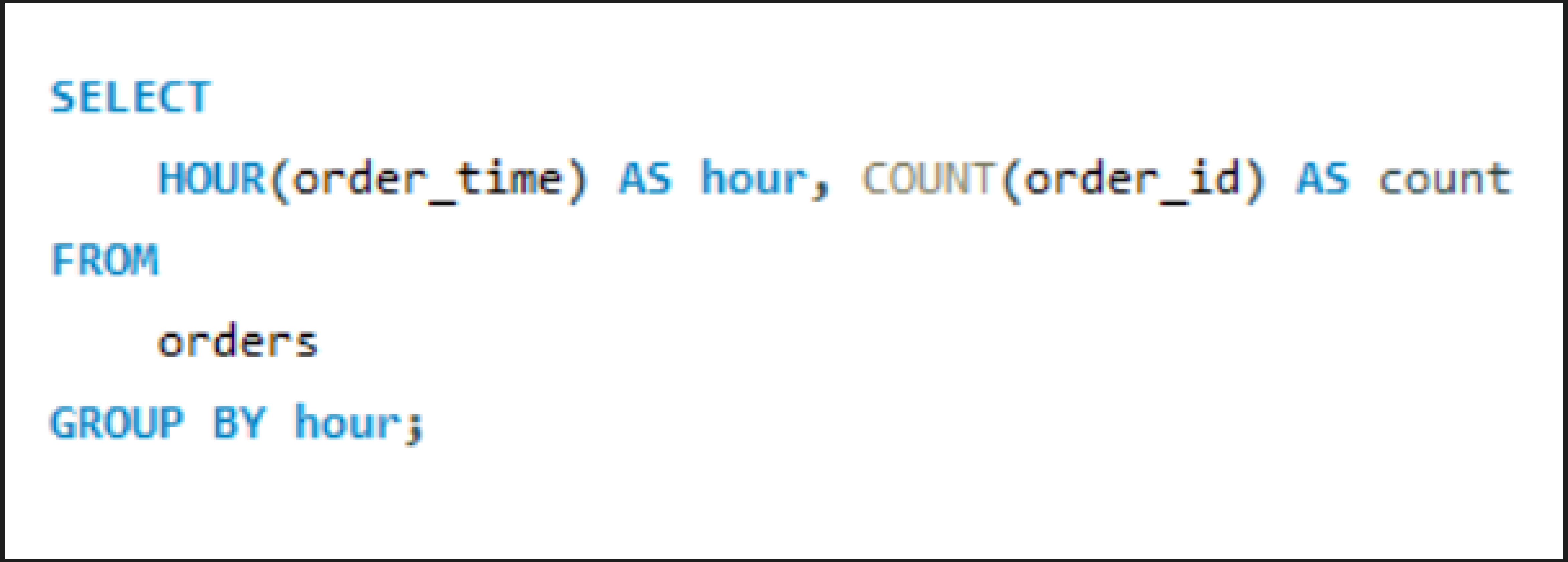


```sql
select k.category, sum(q.quantity) as quantity
from pizza_types as k join pizzas as p
on k.pizza_type_id = p.pizza_type_id
join order_details as q
on p.pizza_id=q.pizza_id
group by k.category order by quantity desc;
```

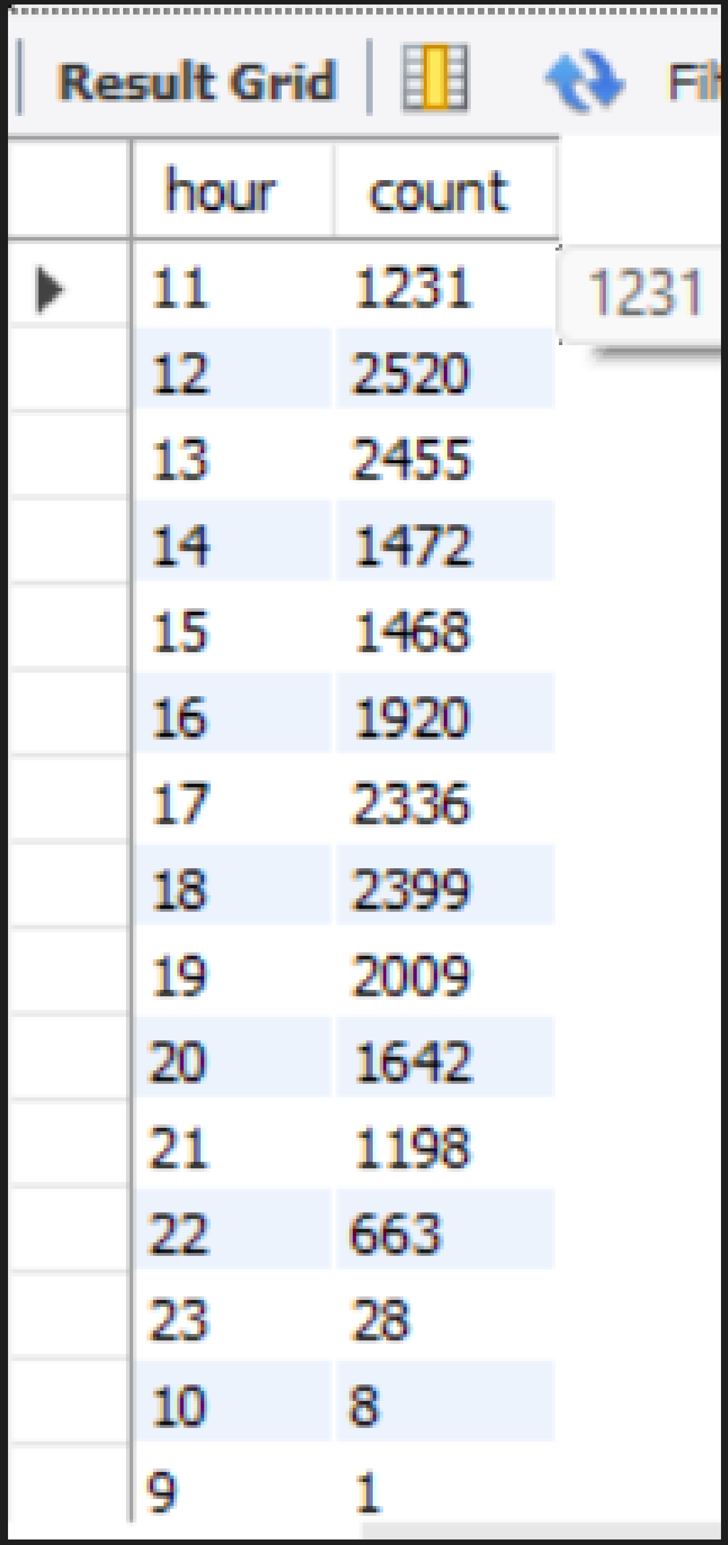


category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

6- DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY

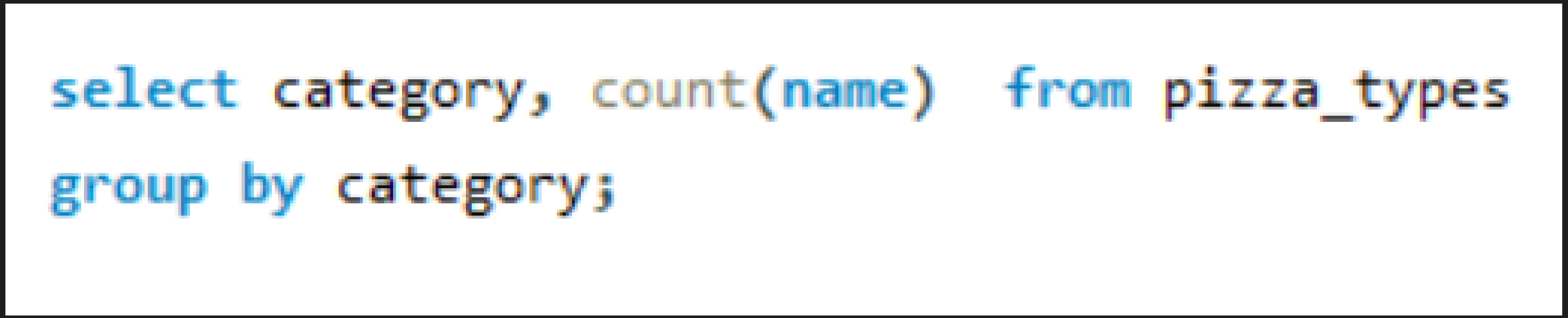


```sql
SELECT
    HOUR(order_time) AS hour, COUNT(order_id) AS count
FROM
    orders
GROUP BY hour;
```

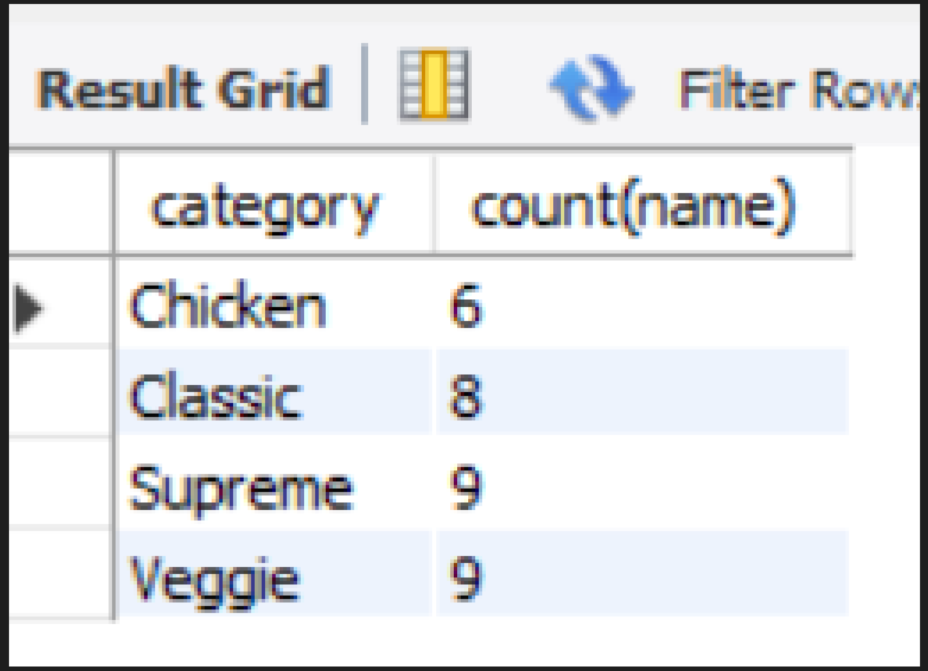


hour	count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

7- JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS

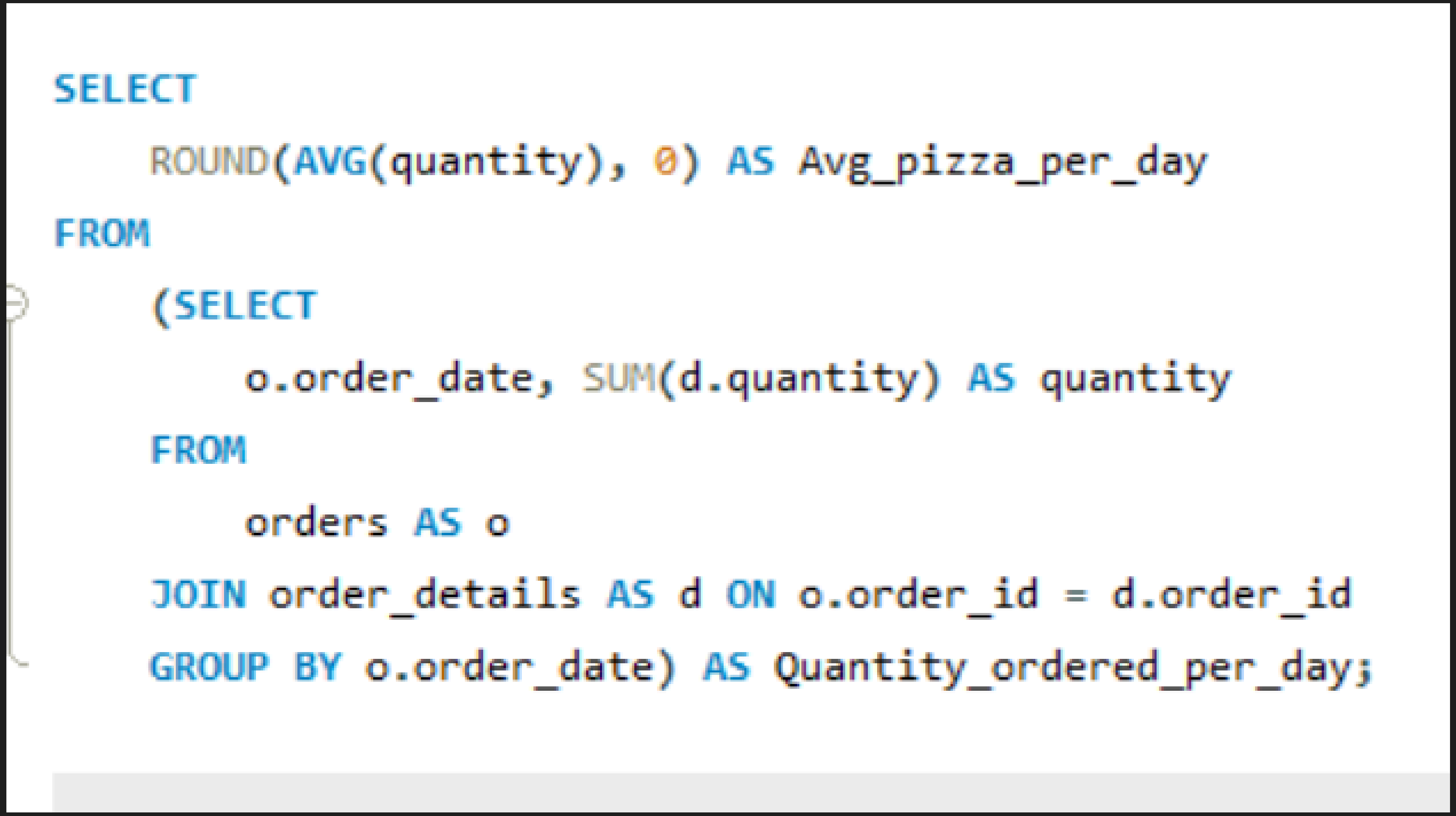


```
select category, count(name)  from pizza_types
group by category;
```

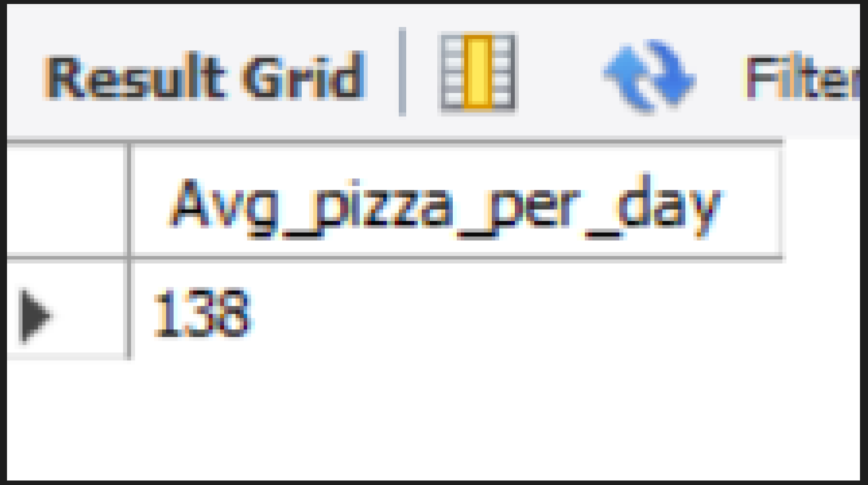


category	count(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

8- GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY



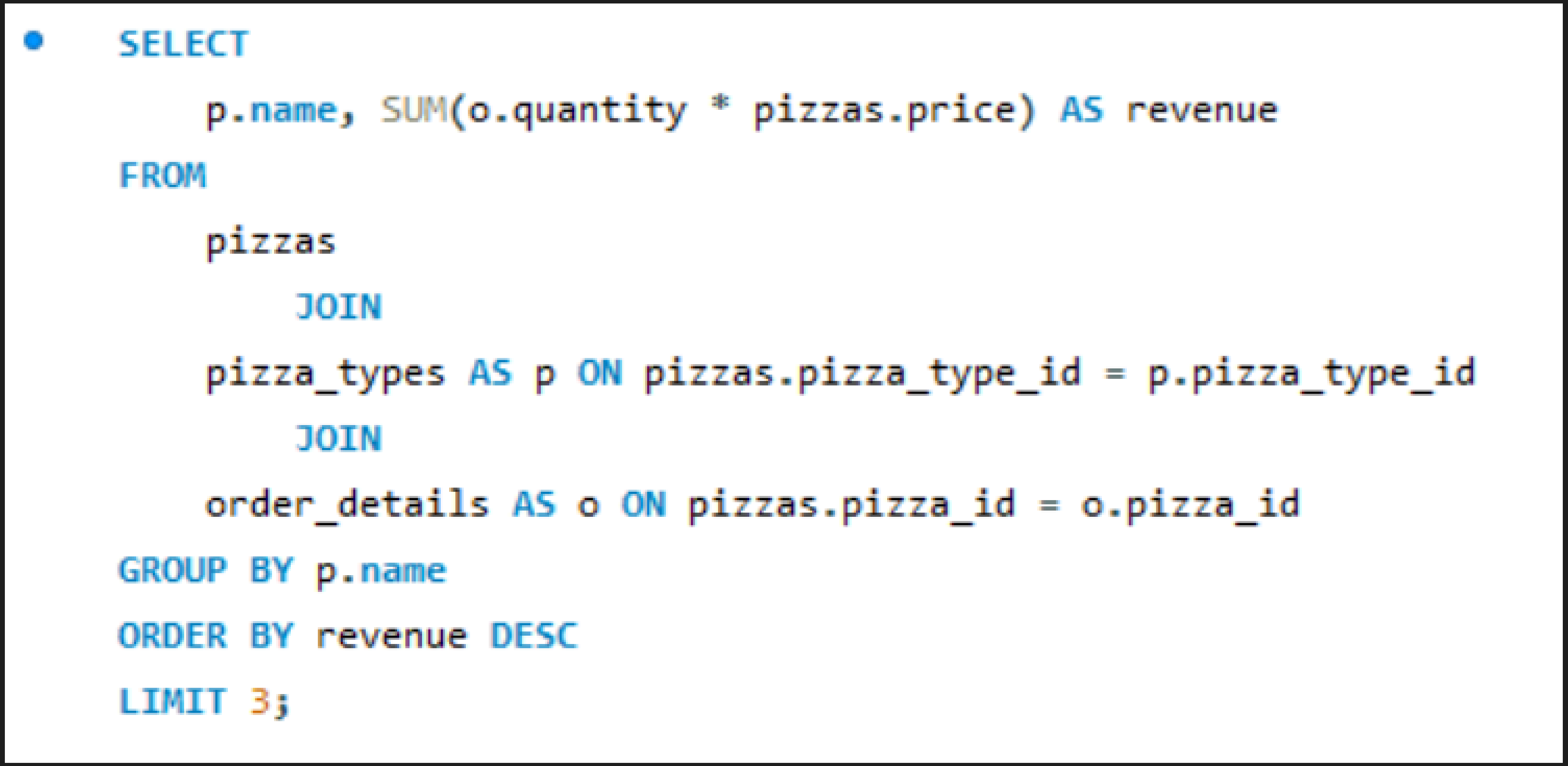
```sql
SELECT
    ROUND(AVG(quantity), 0) AS Avg_pizza_per_day
FROM
    (SELECT
        o.order_date, SUM(d.quantity) AS quantity
    FROM
        orders AS o
    JOIN order_details AS d ON o.order_id = d.order_id
    GROUP BY o.order_date) AS Quantity_ordered_per_day;
```



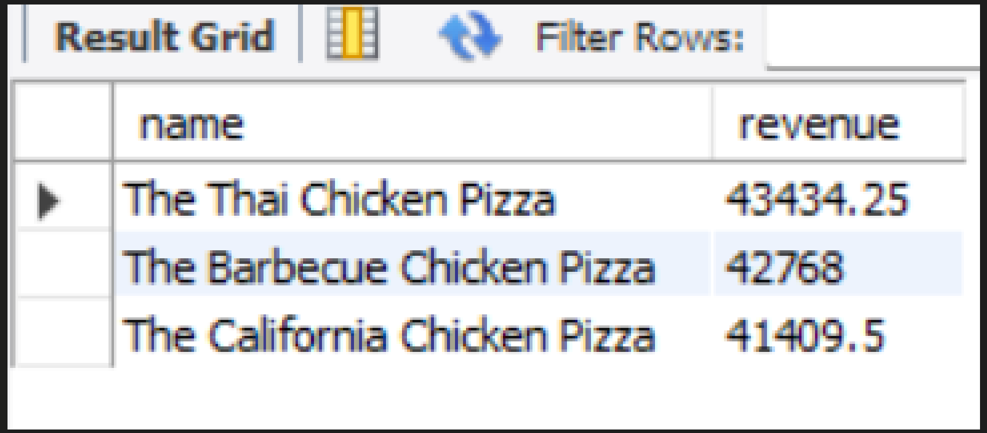
Result Grid | Filter

Avg_pizza_per_day
138

9- DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE

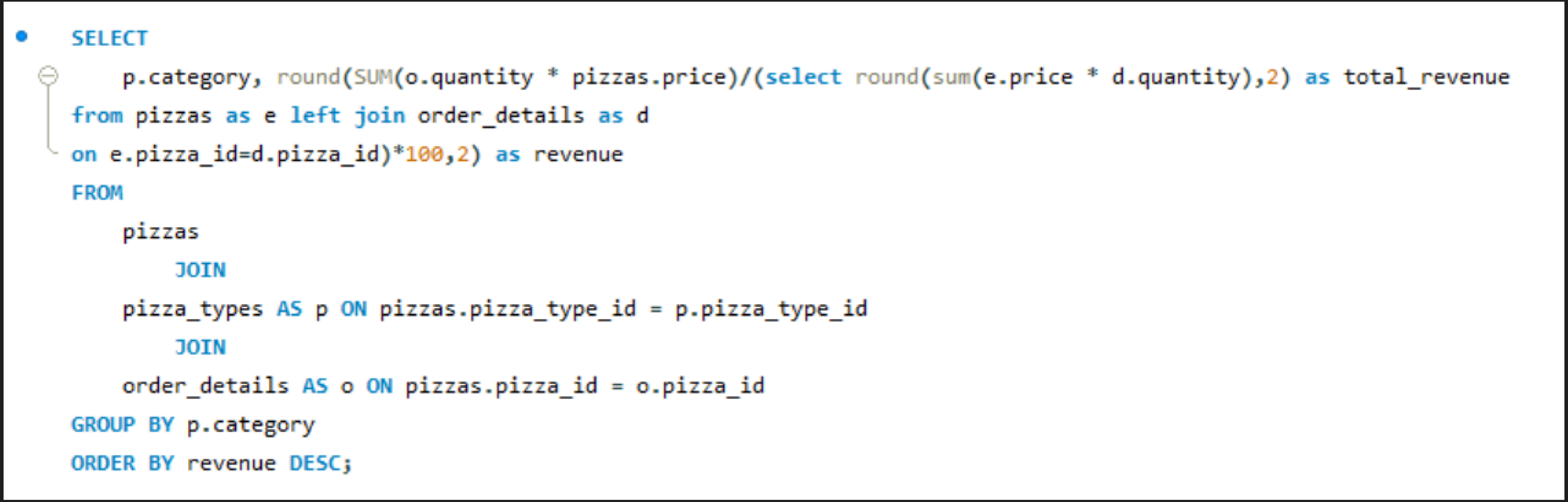


```
SELECT
    p.name, SUM(o.quantity * pizzas.price) AS revenue
FROM
    pizzas
        JOIN
    pizza_types AS p ON pizzas.pizza_type_id = p.pizza_type_id
        JOIN
    order_details AS o ON pizzas.pizza_id = o.pizza_id
GROUP BY p.name
ORDER BY revenue DESC
LIMIT 3;
```

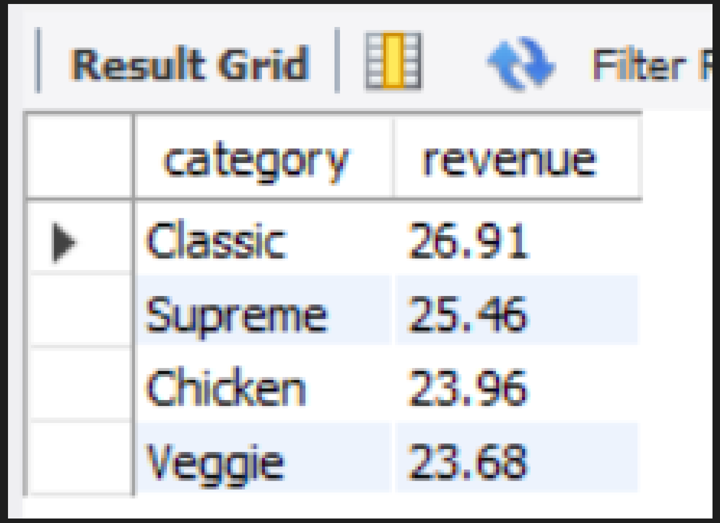


name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

10- CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE



```
SELECT
    p.category, round(SUM(o.quantity * pizzas.price)/(select round(sum(e.price * d.quantity),2) as total_revenue
from pizzas as e left join order_details as d
on e.pizza_id=d.pizza_id)*100,2) as revenue
FROM
    pizzas
        JOIN
    pizza_types AS p ON pizzas.pizza_type_id = p.pizza_type_id
        JOIN
    order_details AS o ON pizzas.pizza_id = o.pizza_id
GROUP BY p.category
ORDER BY revenue DESC;
```



Result Grid | Filter

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

THANK YOU FOR YOUR ATTENTION