PIZZA SALES ANALYSIS PROJECT USING SQL

A Comprehensive Analysis of Pizza Sales Using MySQL

Presented by : Samarth Yalsange

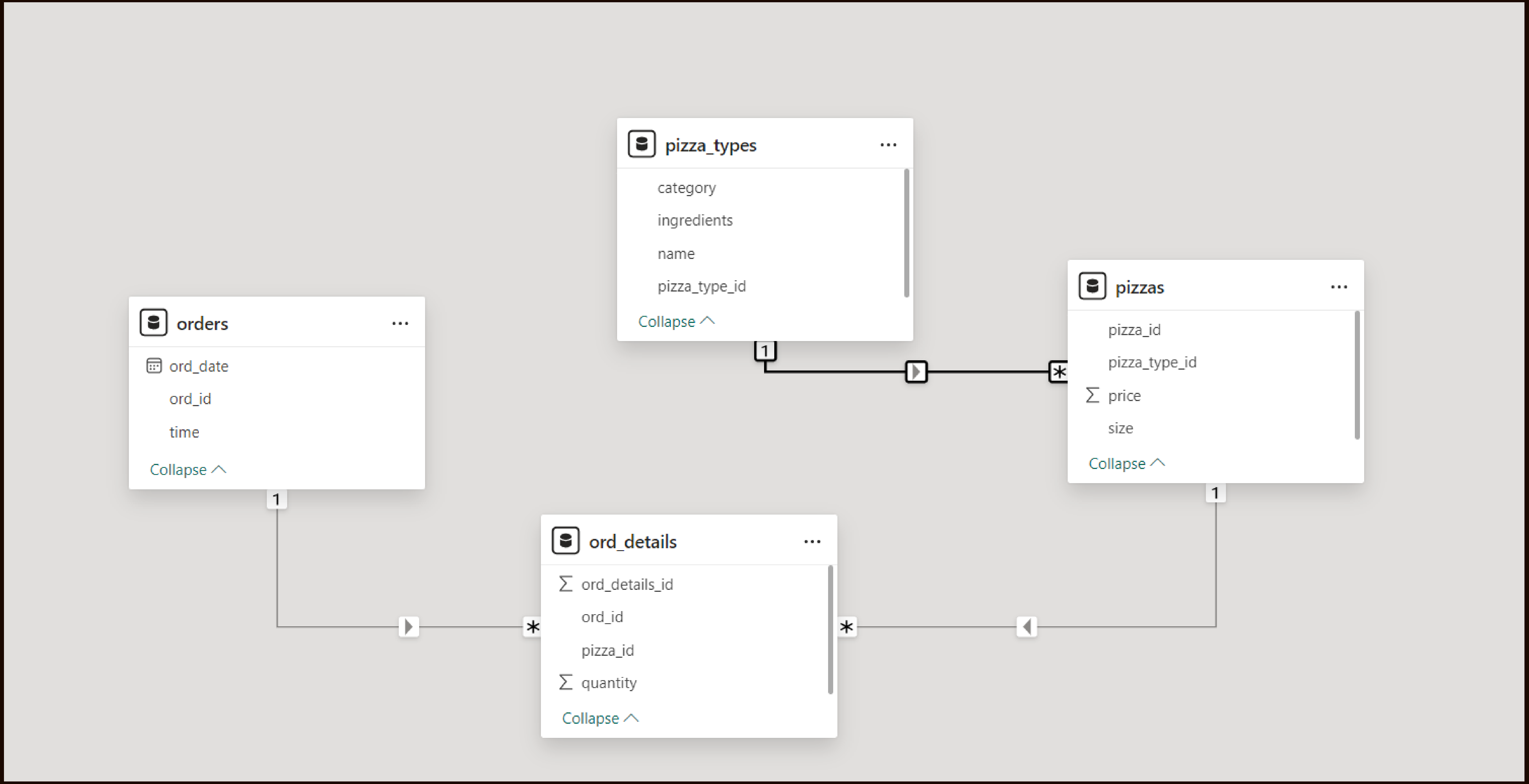
INTRODUCTION

- *Welcome to our data analysis project on pizza sales, designed to provide valuable insights into the performance and trends of a pizza business.*
- *Utilizing MySQL, we have systematically addressed 13 key questions to help better understand various aspects of sales, customer behavior, and operational efficiency*

PROJECT OVERVIEW

The Pizza Sales Analysis project leverages SQL to explore sales patterns, customer, preferences, and business performance of a pizza restaurant. Basic analysis includes calculating total order, revenue, identifying the highest-priced pizza. The most common pizza size., and the top 5 most ordered pizza types. Intermediate analysis involves joining tables to find total quantities of each pizza category. distribution of orders by hour, category-wise pizza distribution, daily average orders, and top 3 pizza type of revenue. Advanced Analysis calculate each pizza type's revenue contribution, cumulative revenue over time, and top3 pizza types by revenue within each category. This insights guide strategic decisions to optimize operations and boost profitability.

ENTITY RELATIONSHIP DIGRAM



Questions:

Basic:

- *Retrieve the total number of orders placed.*
- *Calculate the total revenue generated from pizza sales.*
- *Identify the highest-priced pizza.*
- *Identify the most common pizza size ordered.*
- *List the top 5 most ordered pizza types along with their quantities.*

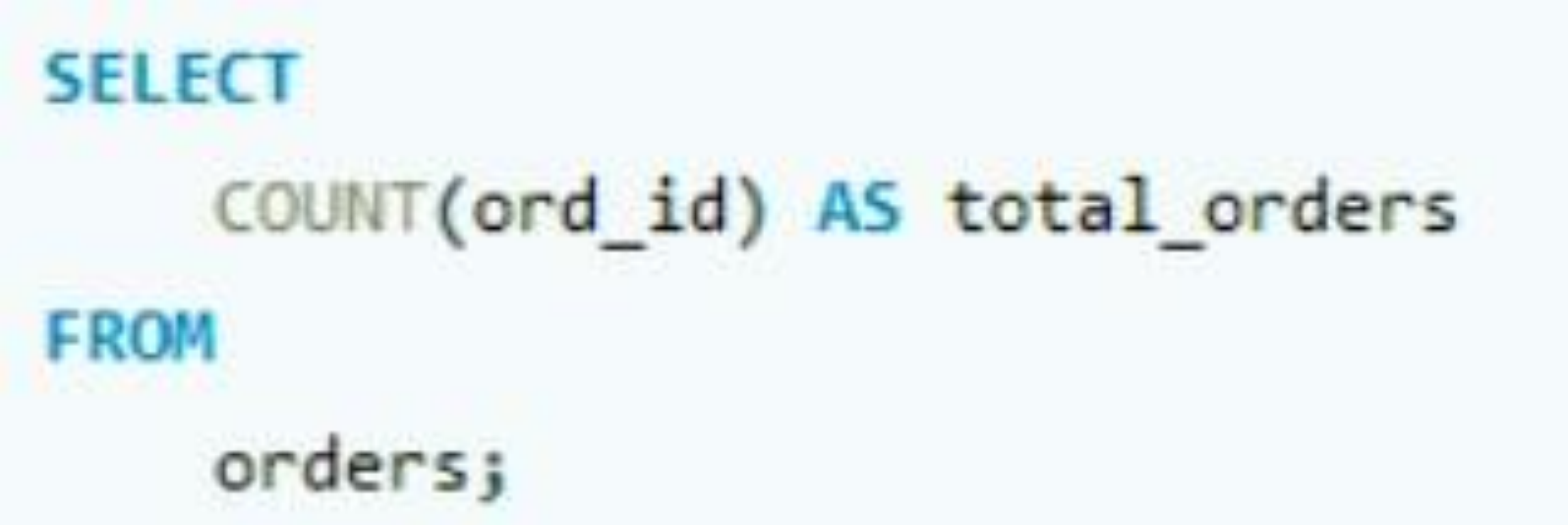
Intermediate:

- *Join the necessary tables to find the total quantity of each pizza category ordered.*
- *Determine the distribution of orders by hour of the day.*
- *Join relevant tables to find the category-wise distribution of pizzas.*
- *Group the orders by date and calculate the average number of pizzas ordered per day.*
- *Determine the top 3 most ordered pizza types based on revenue.*

Advanced:

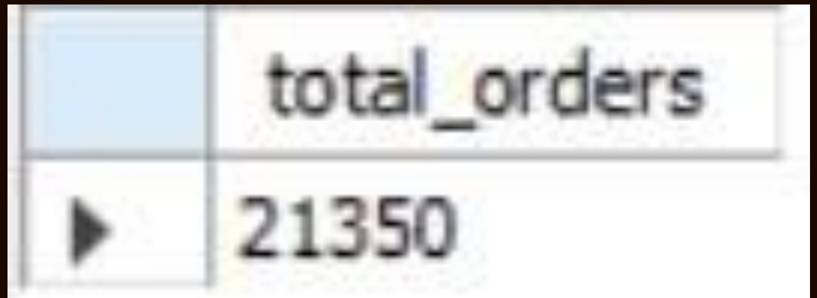
- *Calculate the percentage contribution of each pizza type to total revenue.*
- *Analyze the cumulative revenue generated over time.*
- *Determine the top 3 most ordered pizza types based on revenue for each pizza category.*

1.Retrieve the total count of orders placed.



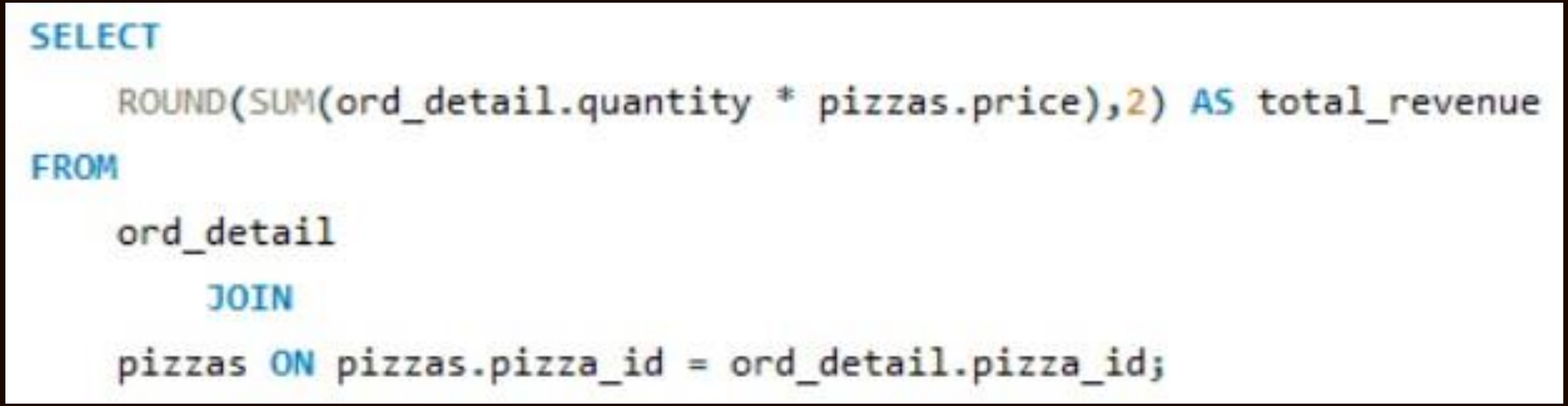
```
SELECT
    COUNT(ord_id) AS total_orders
FROM
    orders;
```

output:



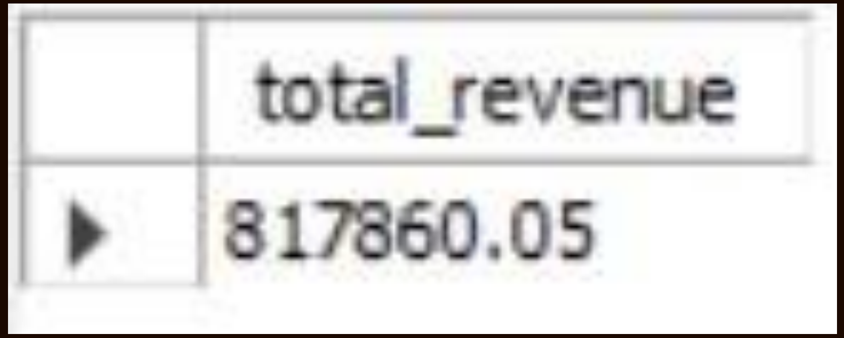
	total_orders
▶	21350

2.Calculate the total revenue generated from pizza sales.



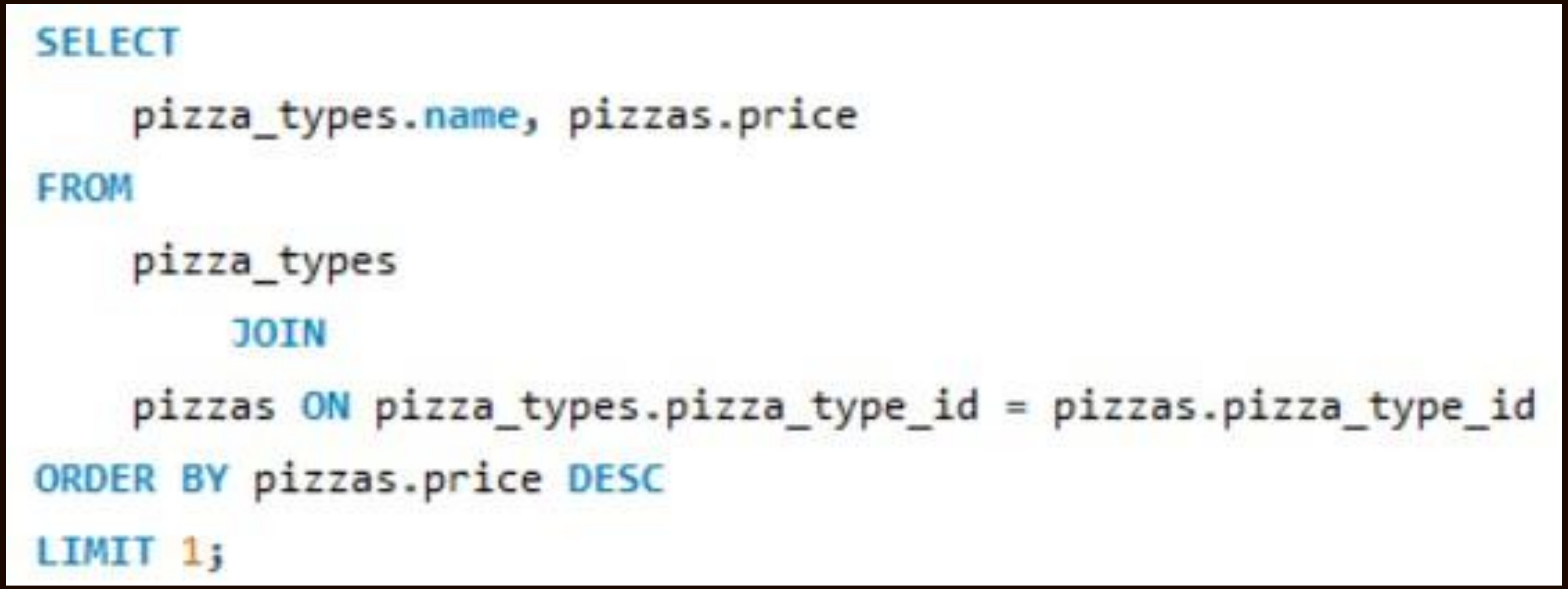
```sql
SELECT
    ROUND(SUM(ord_detail.quantity * pizzas.price),2) AS total_revenue
FROM
    ord_detail
        JOIN
    pizzas ON pizzas.pizza_id = ord_detail.pizza_id;
```

output:



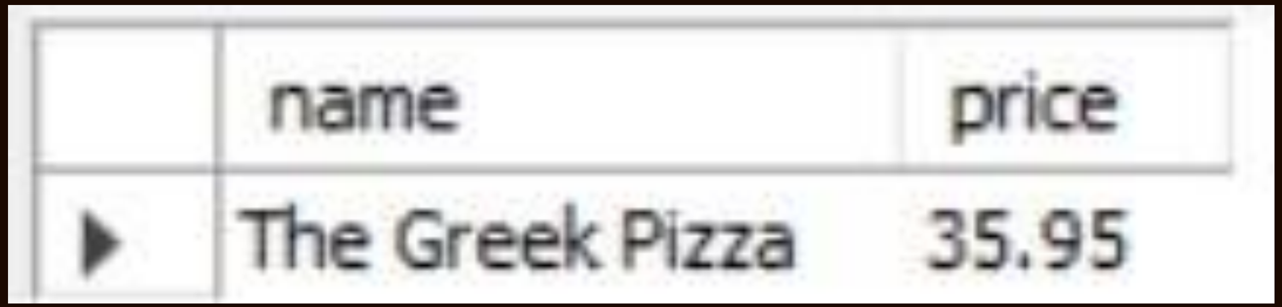
	total_revenue
▶	817860.05

3.Identify the highest-priced pizza.



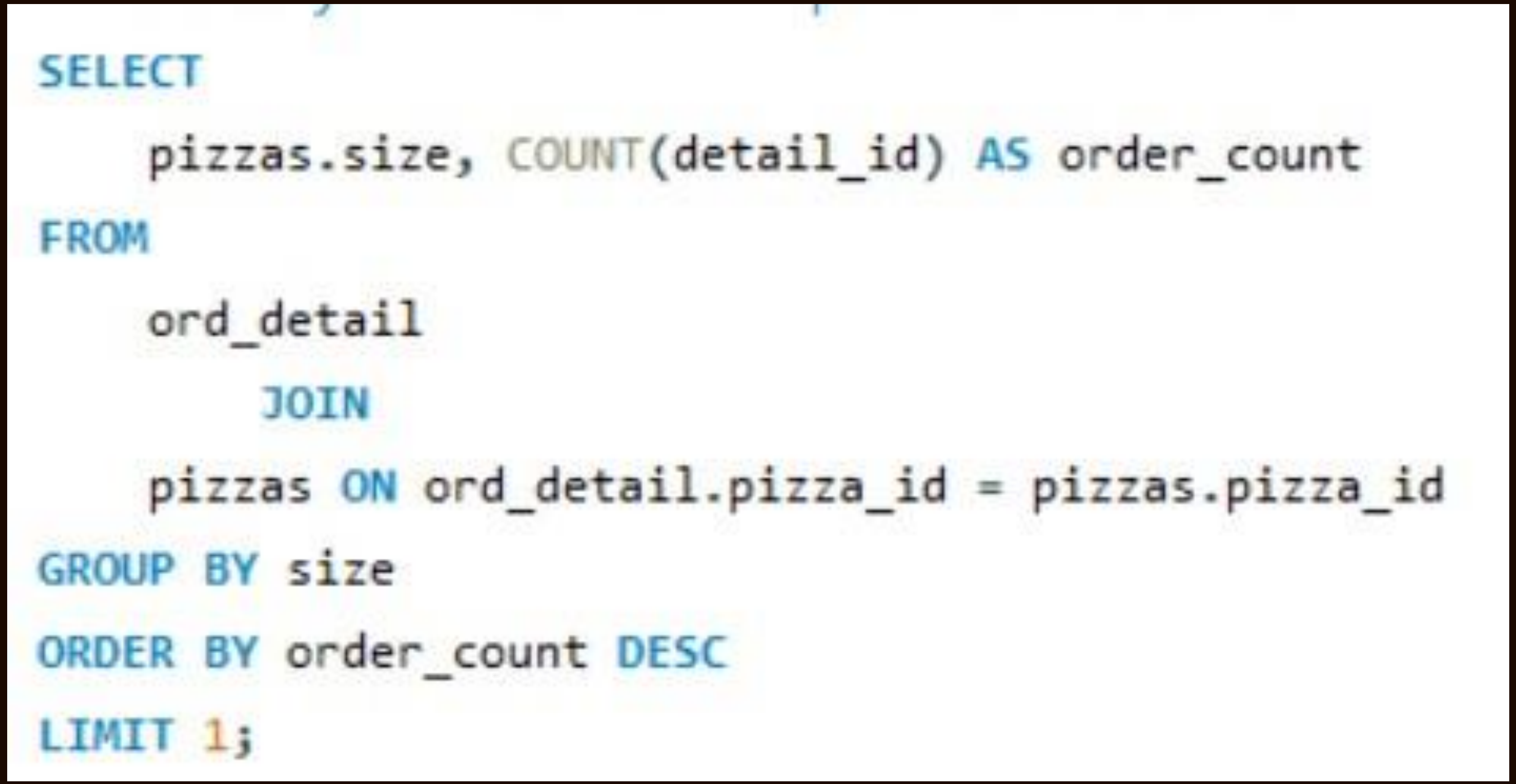
```sql
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

output:



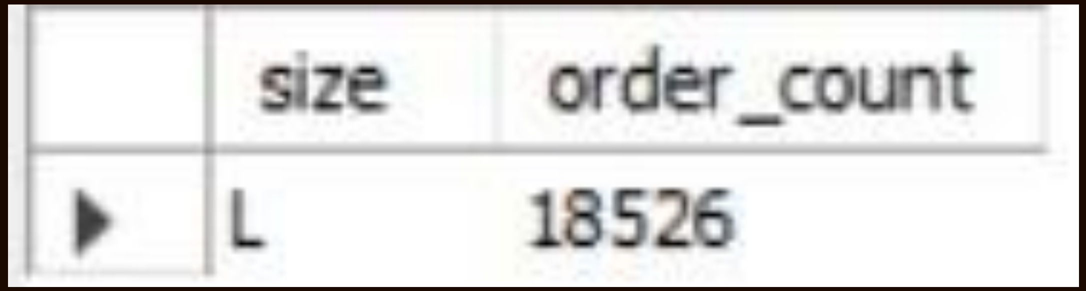
	name	price
▶	The Greek Pizza	35.95

4.Identify the most common pizza size ordered.



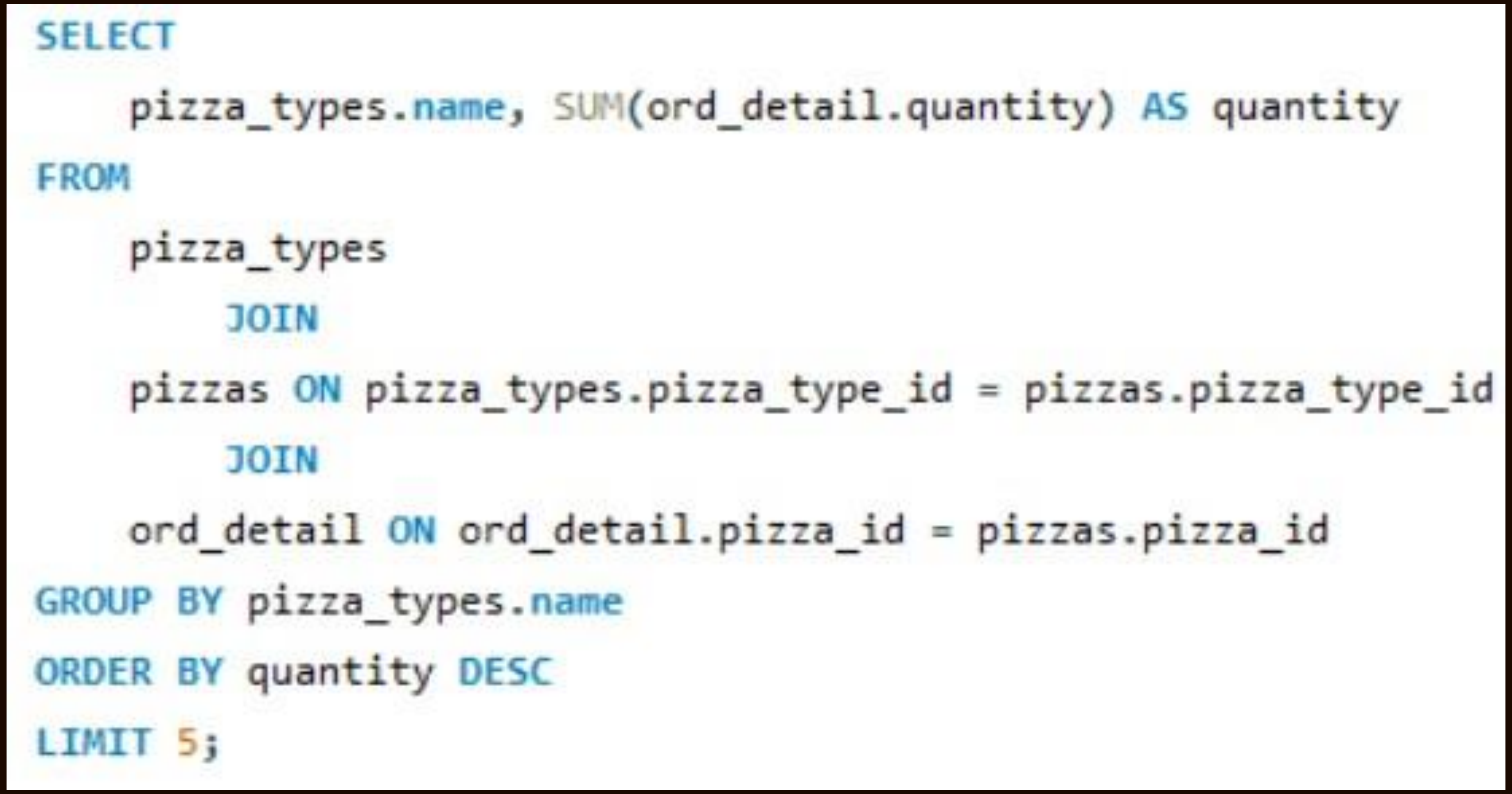
```
SELECT
    pizzas.size, COUNT(detail_id) AS order_count
FROM
    ord_detail
        JOIN
    pizzas ON ord_detail.pizza_id = pizzas.pizza_id
GROUP BY size
ORDER BY order_count DESC
LIMIT 1;
```

output:



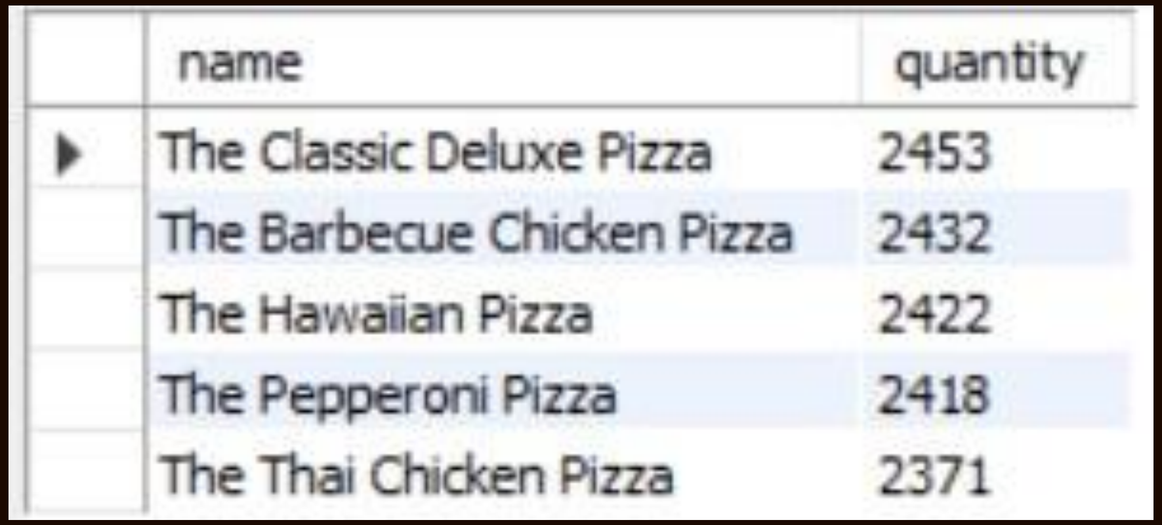
	size	order_count
▶	L	18526

5.List the top 5 most ordered pizza types along with their quantities.



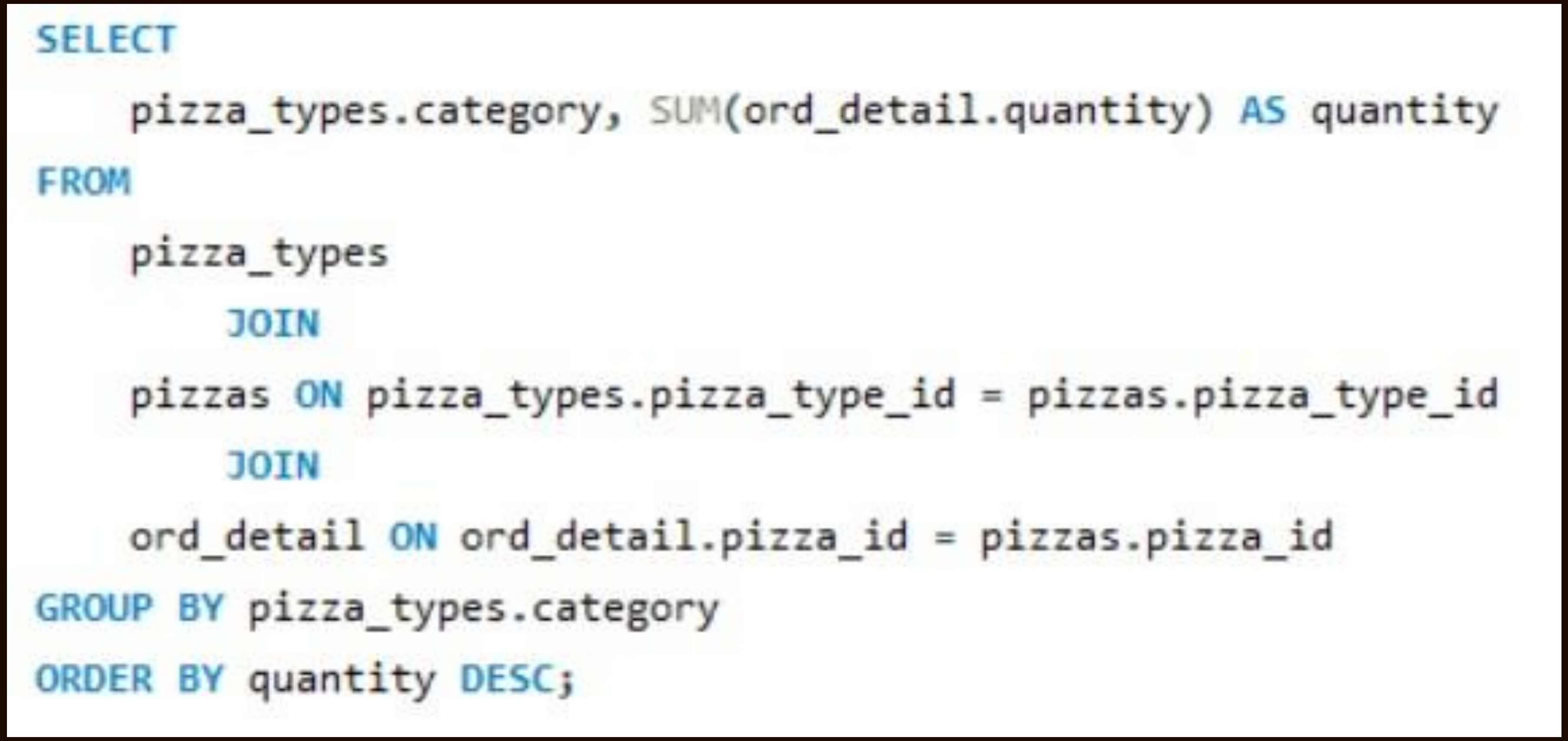
```
SELECT
    pizza_types.name, SUM(ord_detail.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    ord_detail ON ord_detail.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;
```

output:



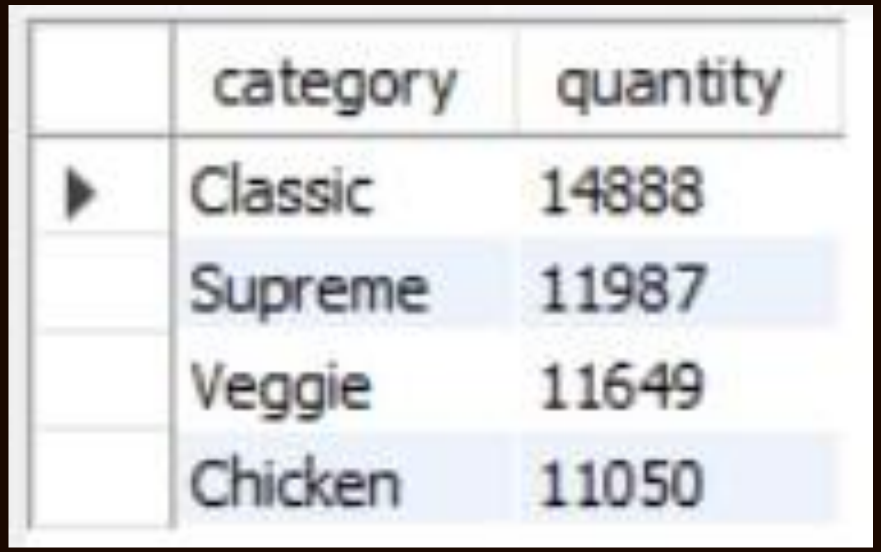
name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

6. Join the necessary tables to find the total quantity of each pizza category ordered.



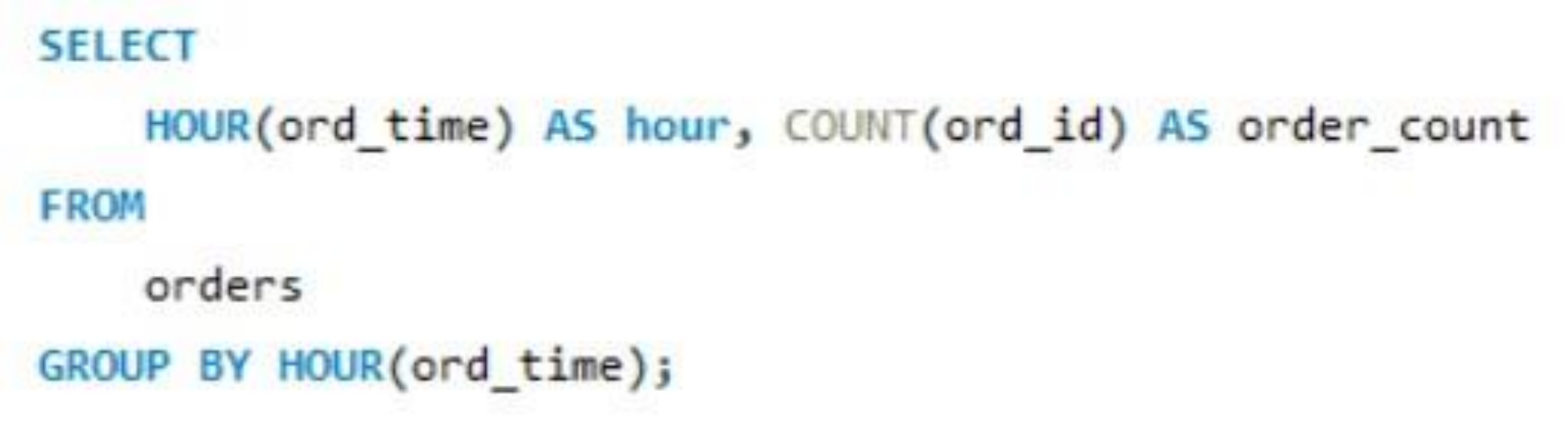
```
SELECT
    pizza_types.category, SUM(ord_detail.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    ord_detail ON ord_detail.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity DESC;
```

output:



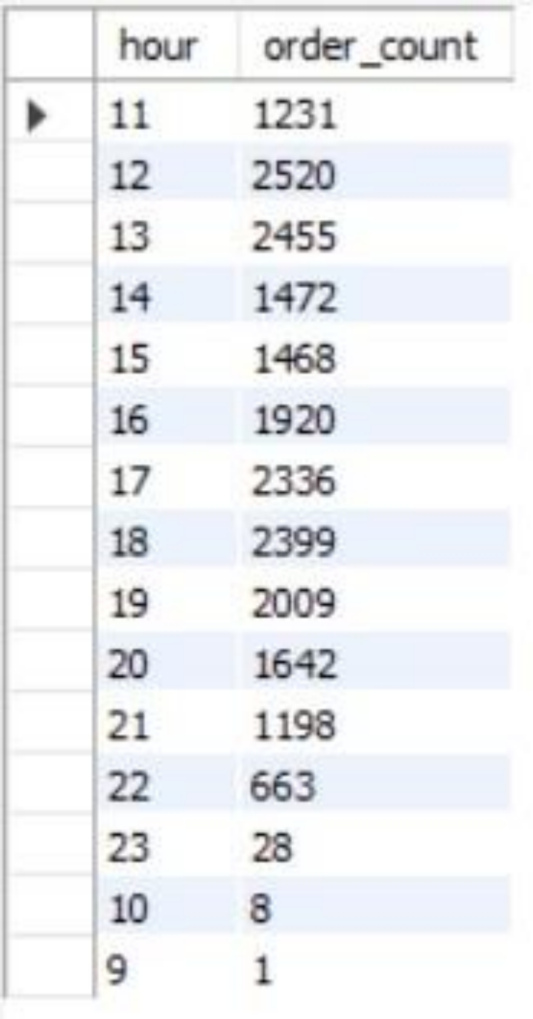
category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

7. Determine the distribution of orders by hour of the day.



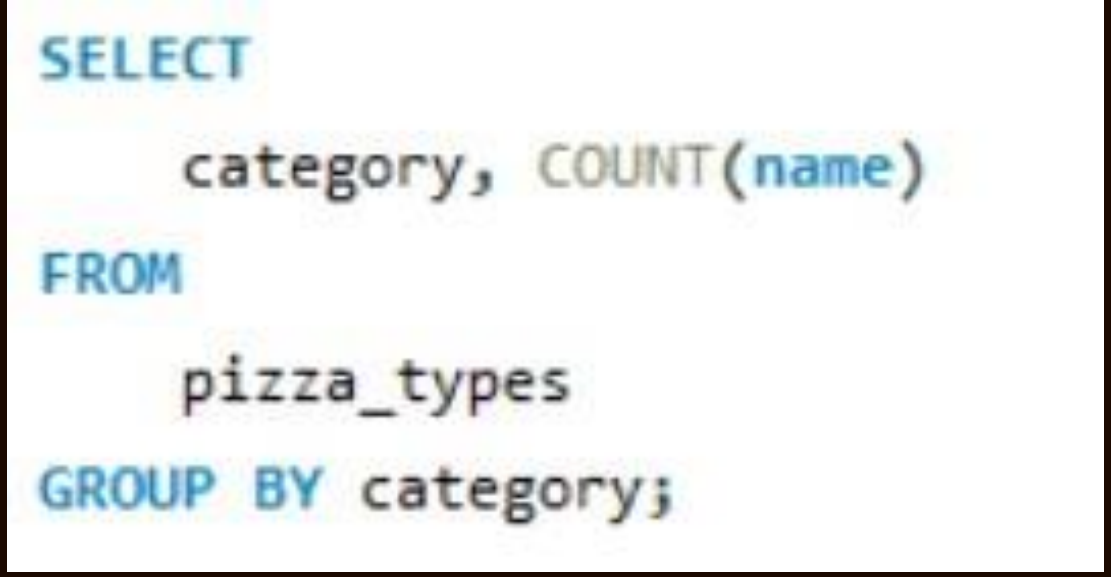
```
SELECT
    HOUR(ord_time) AS hour, COUNT(ord_id) AS order_count
FROM
    orders
GROUP BY HOUR(ord_time);
```

output:



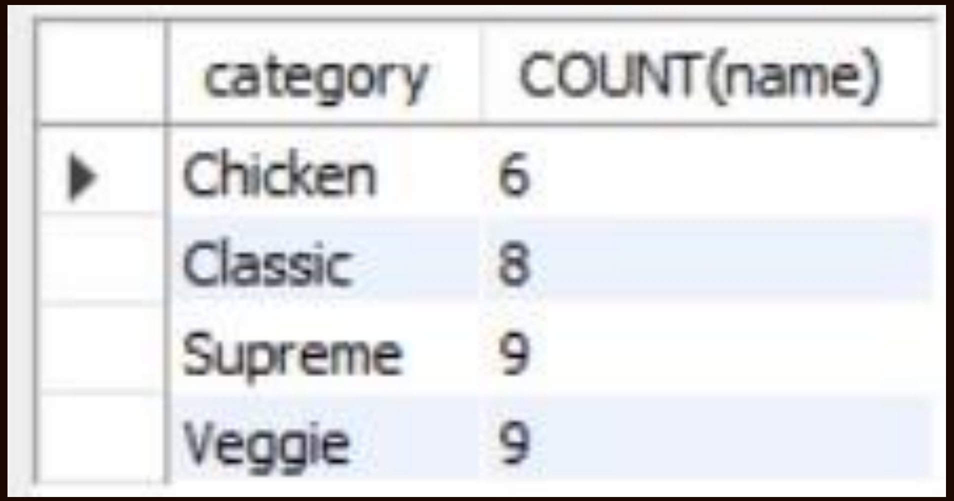
hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

8.Join relevant tables to find the category-wise distribution of pizzas.



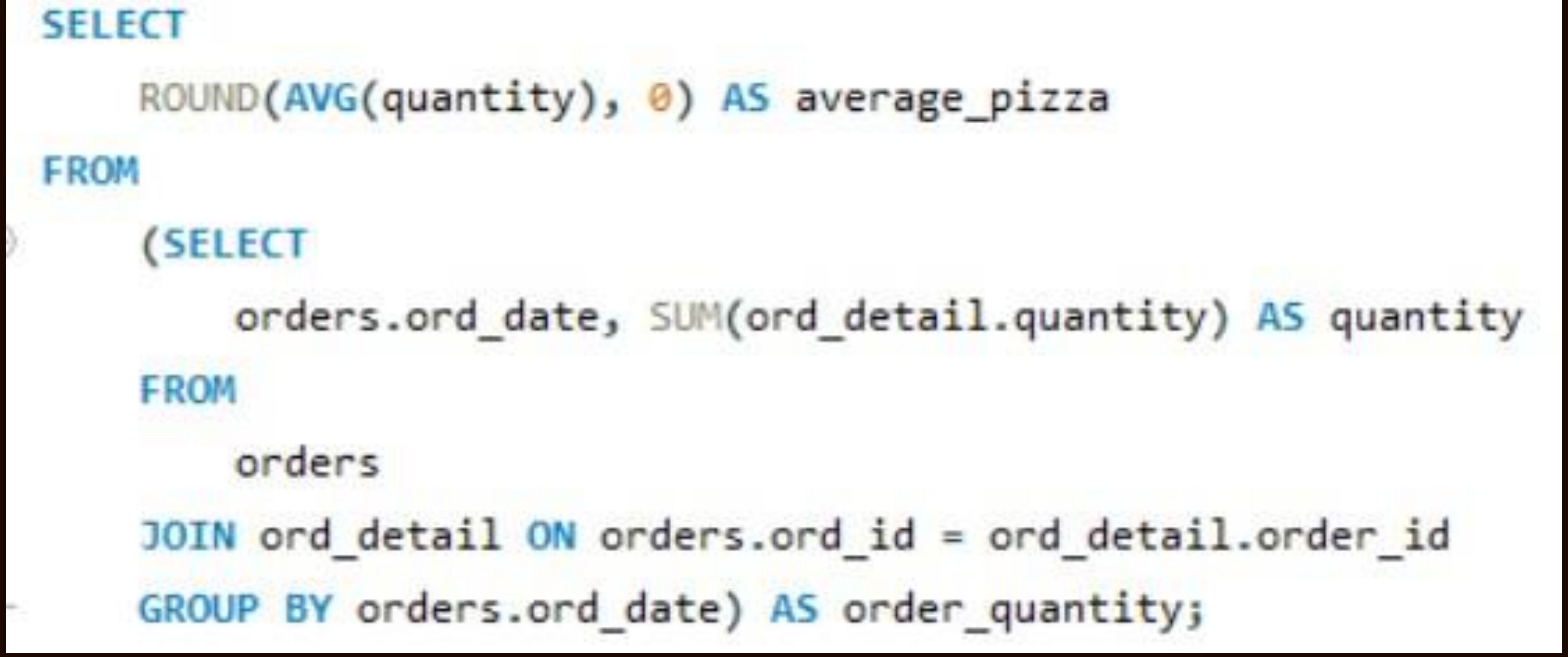
```
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

output:



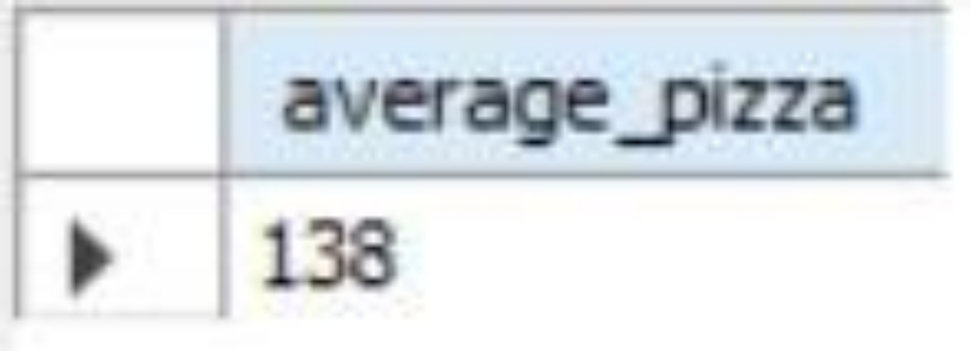
	category	COUNT(name)
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

9.Group the orders by date and calculate the average number of pizzas ordered per day



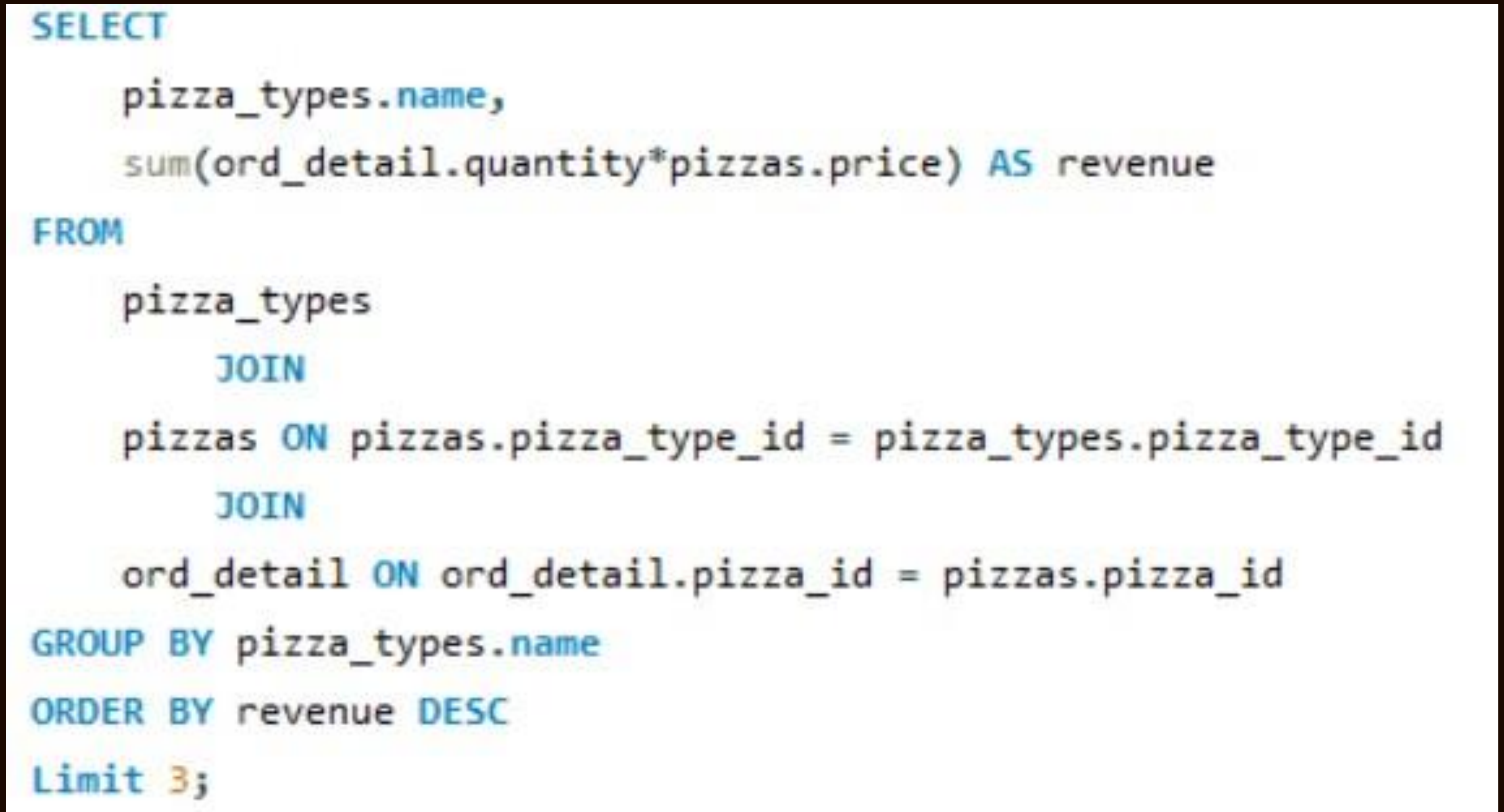
```
SELECT
    ROUND(AVG(quantity), 0) AS average_pizza
FROM
    (SELECT
        orders.ord_date, SUM(ord_detail.quantity) AS quantity
    FROM
        orders
    JOIN ord_detail ON orders.ord_id = ord_detail.order_id
    GROUP BY orders.ord_date) AS order_quantity;
```

output:



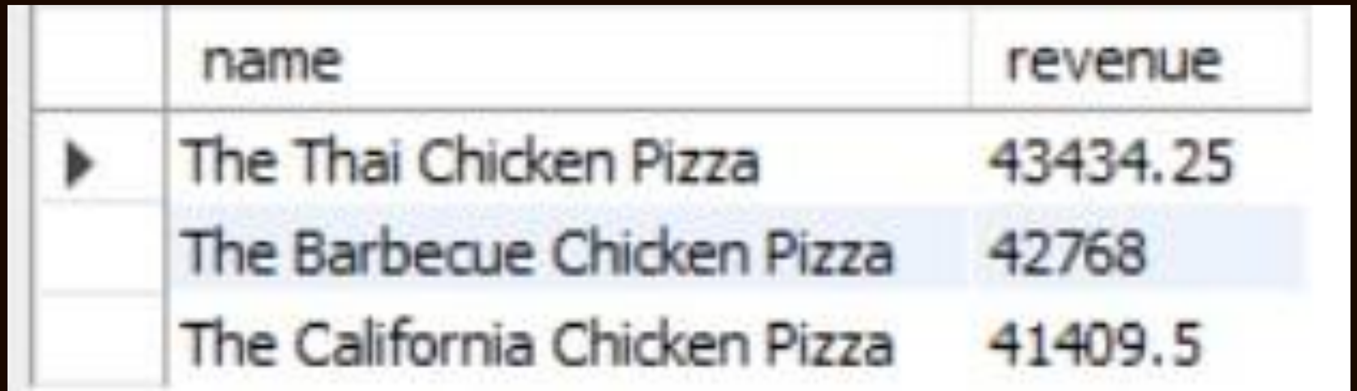
average_pizza
138

10.Determine the top 3 most ordered pizza types based on revenue.



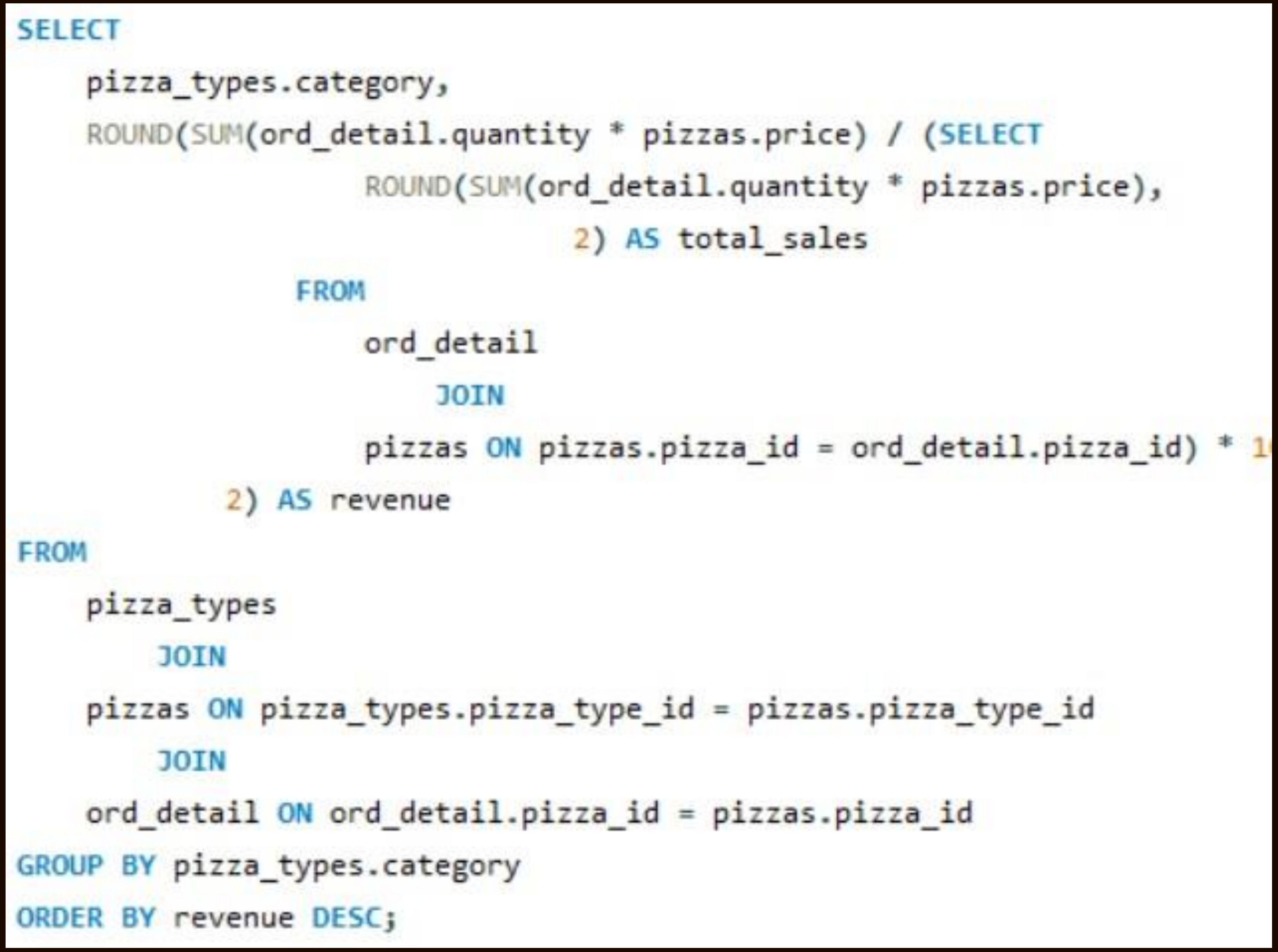
```
SELECT
    pizza_types.name,
    sum(ord_detail.quantity*pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    ord_detail ON ord_detail.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
Limit 3;
```

output:



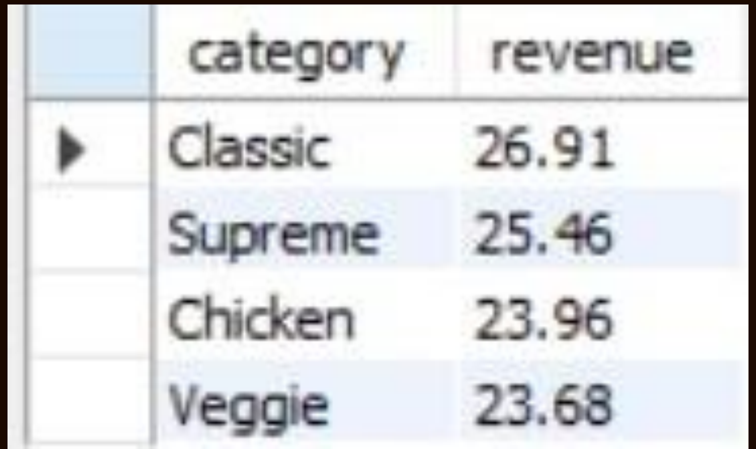
name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

11.Calculate the percentage contribution of each pizza type to total revenue.



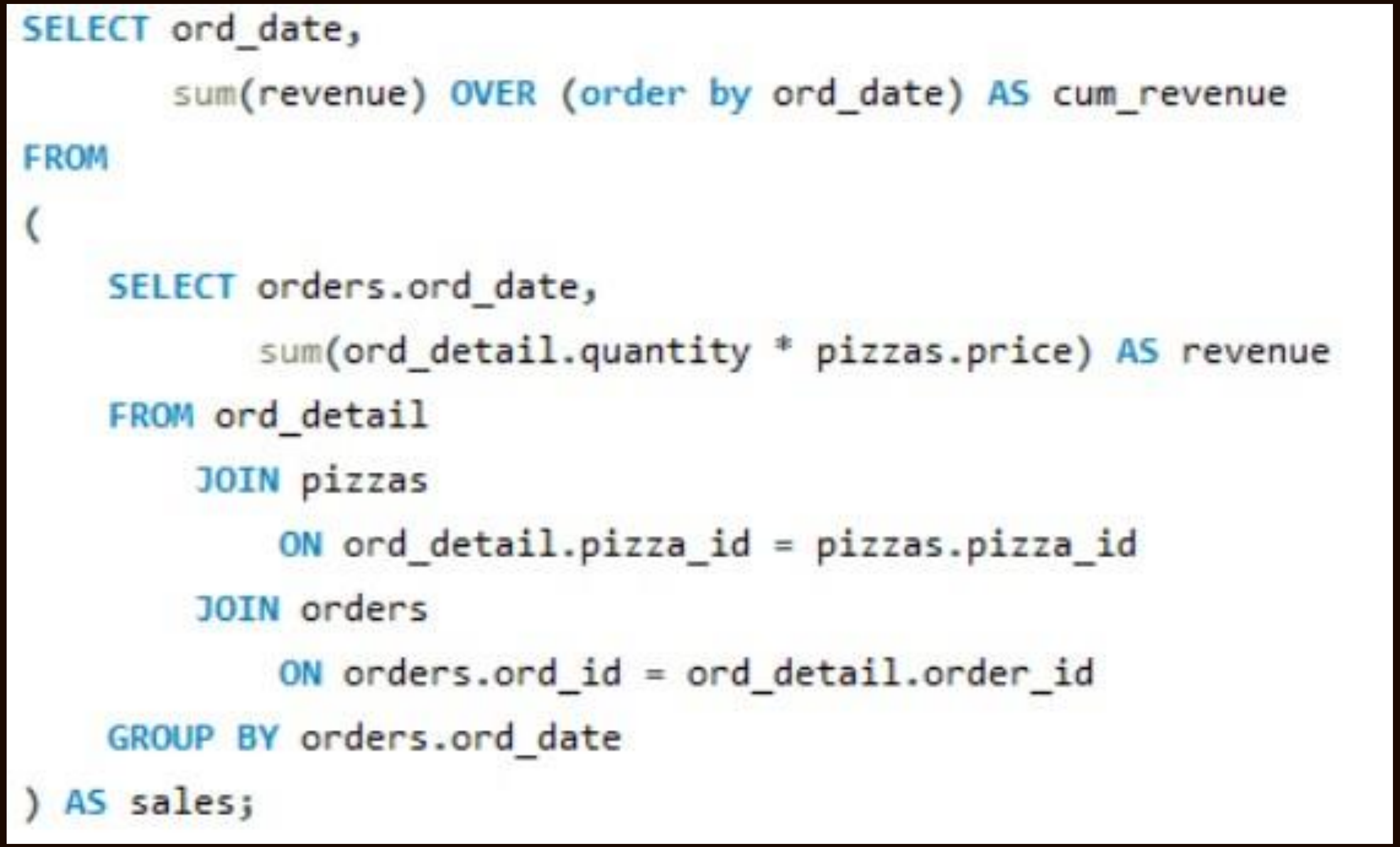
```
SELECT
  pizza_types.category,
  ROUND(SUM(ord_detail.quantity * pizzas.price) / (SELECT
          ROUND(SUM(ord_detail.quantity * pizzas.price),
                2) AS total_sales
        FROM
          ord_detail
            JOIN
          pizzas ON pizzas.pizza_id = ord_detail.pizza_id) * 1
      2) AS revenue
FROM
  pizza_types
    JOIN
  pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
    JOIN
  ord_detail ON ord_detail.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC;
```

output:



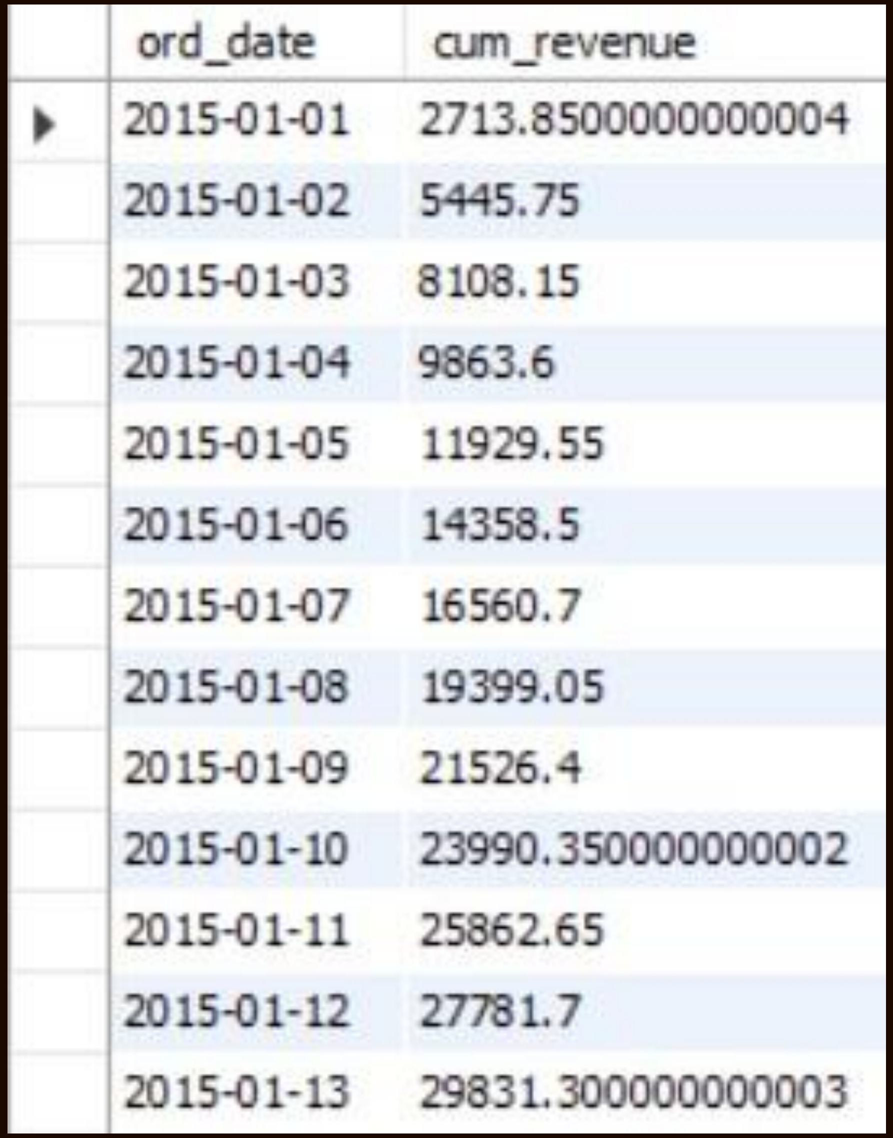
category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

12.Analyze the cumulative revenue generated over time.



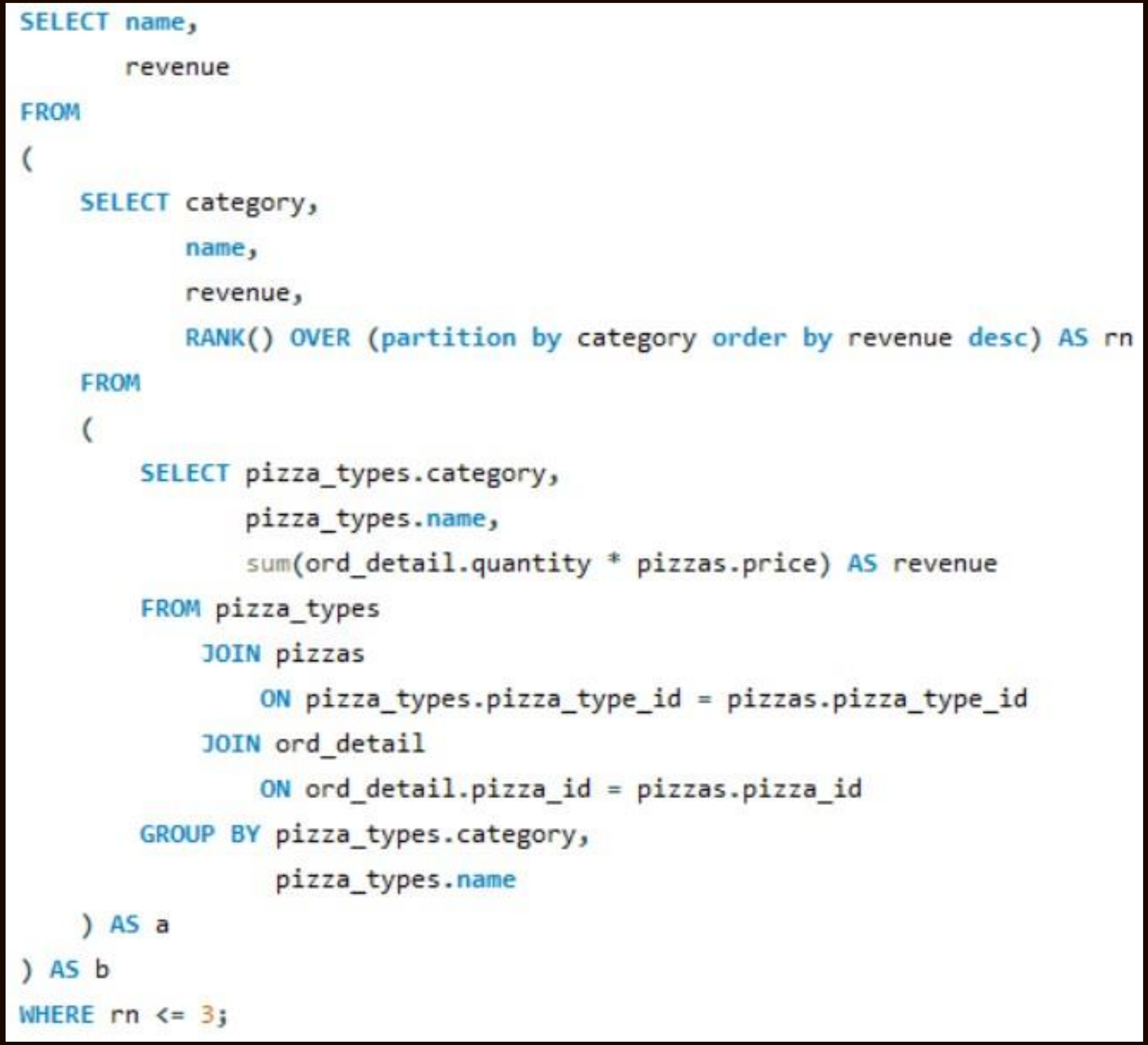
```
SELECT ord_date,
       sum(revenue) OVER (order by ord_date) AS cum_revenue
FROM
(
    SELECT orders.ord_date,
           sum(ord_detail.quantity * pizzas.price) AS revenue
    FROM ord_detail
        JOIN pizzas
            ON ord_detail.pizza_id = pizzas.pizza_id
        JOIN orders
            ON orders.ord_id = ord_detail.order_id
    GROUP BY orders.ord_date
) AS sales;
```

output:



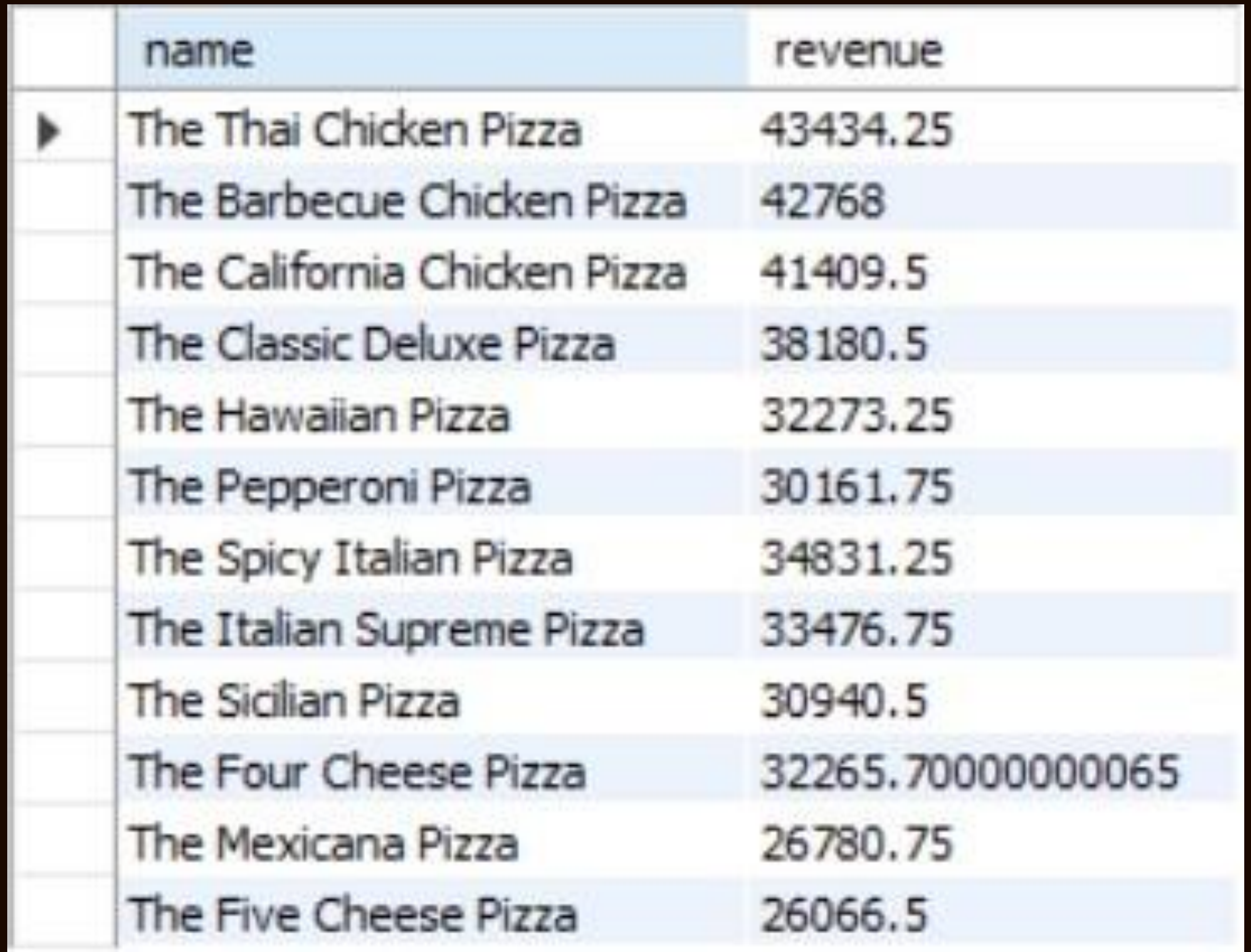
ord_date	cum_revenue
2015-01-01	2713.8500000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.4
2015-01-10	23990.350000000002
2015-01-11	25862.65
2015-01-12	27781.7
2015-01-13	29831.300000000003

13.Determine the top 3 most ordered pizza types based on revenue for each pizza category.



```
SELECT name,
       revenue
FROM
(
    SELECT category,
           name,
           revenue,
           RANK() OVER (partition by category order by revenue desc) AS rn
    FROM
    (
        SELECT pizza_types.category,
               pizza_types.name,
               sum(ord_detail.quantity * pizzas.price) AS revenue
        FROM pizza_types
            JOIN pizzas
               ON pizza_types.pizza_type_id = pizzas.pizza_type_id
            JOIN ord_detail
               ON ord_detail.pizza_id = pizzas.pizza_id
        GROUP BY pizza_types.category,
                 pizza_types.name
    ) AS a
) AS b
WHERE rn <= 3;
```

output:



name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5
The Classic Deluxe Pizza	38180.5
The Hawaiian Pizza	32273.25
The Pepperoni Pizza	30161.75
The Spicy Italian Pizza	34831.25
The Italian Supreme Pizza	33476.75
The Sicilian Pizza	30940.5
The Four Cheese Pizza	32265.70000000065
The Mexicana Pizza	26780.75
The Five Cheese Pizza	26066.5

CONCLUSION:

- In conclusion, our project analyzing pizza sales data using MySQL has effectively turned raw data into valuable insights.
- Through tackling thirteen targeted questions, we have pinpointed growth opportunities and areas that require enhancement.

THANK YOU