PIZZAAREA
DATABASE

BY USING MY SQL

PRESENTATION ON

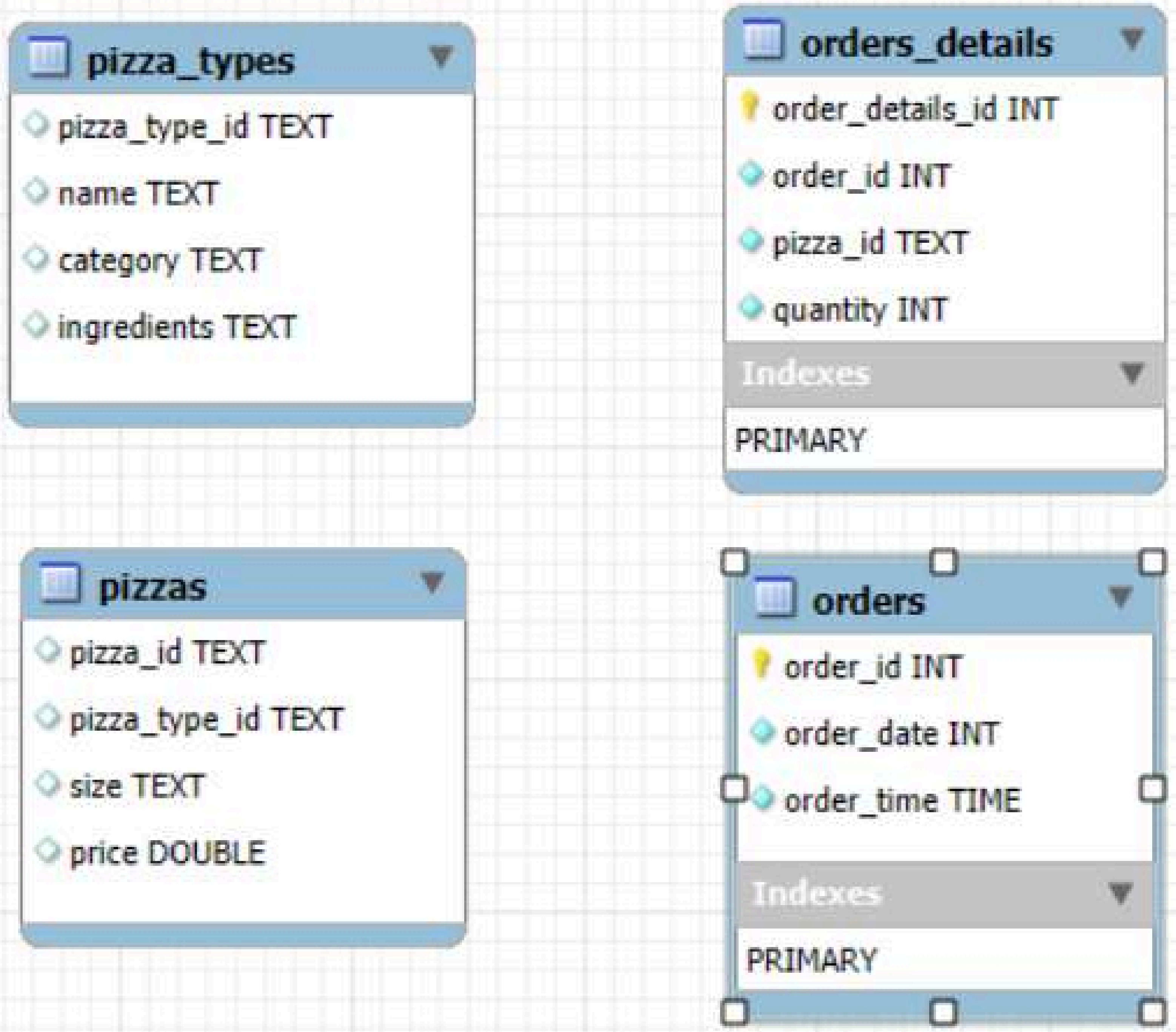
Pizza SALES ANALYSIS PROJECT

PRESENTED BY : NANDINI AGRAWAL

INTRODUCTION

In this discussion, we take a deep dive into analyzing sales data from a pizza store using MySQL's robust querying and analytical capabilities. The dataset consists of four primary tables: orders, order_details, pizza_types, and pizzas. These tables provide valuable insights into customer purchases, various pizza categories, and detailed sales records.

By exploring this data, we seek to identify patterns in sales performance, customer preferences, and the impact of marketing strategies. Our objective is to extract meaningful insights that can drive better business decisions and optimize operations in the pizza industry. Stay with us as we uncover key trends, highlight essential takeaways, and discuss how these findings can shape the future of pizza sales.



PROJECT OVERVIEW

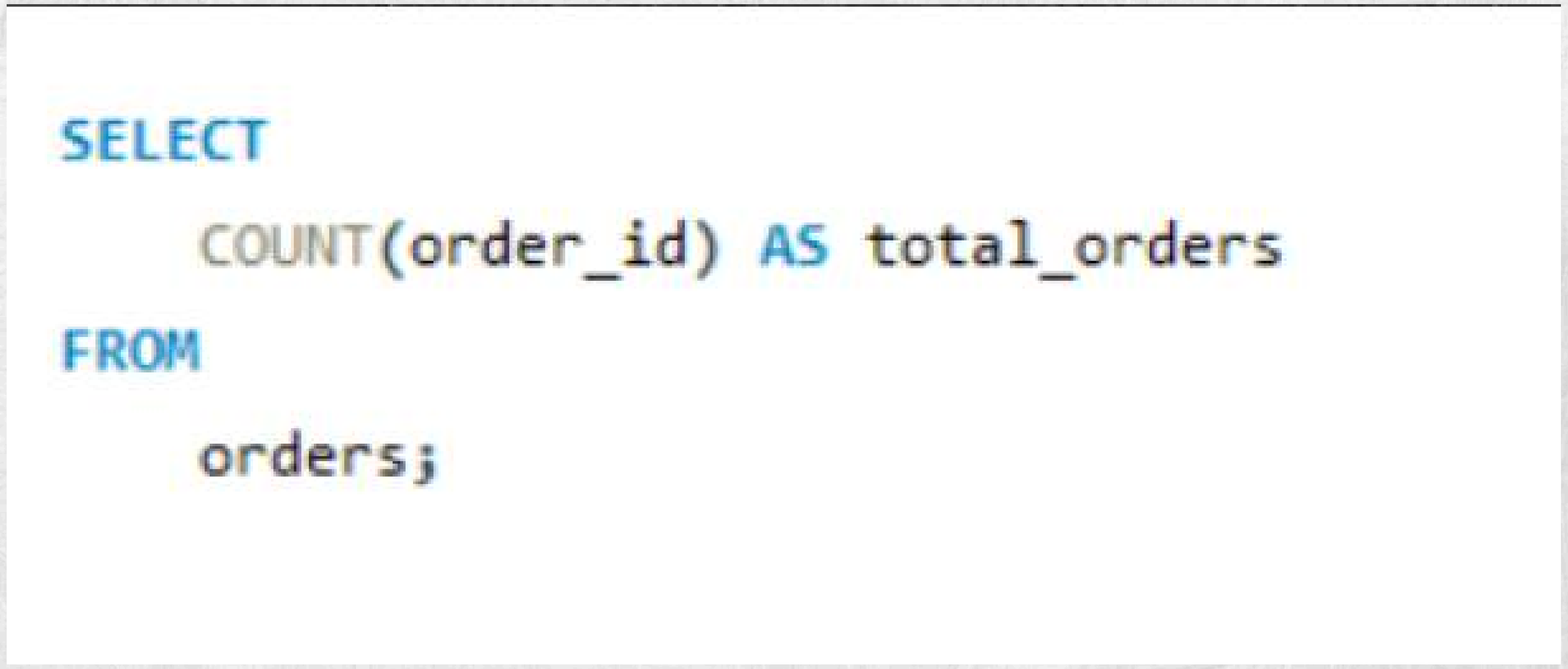
This project delves into a comprehensive examination of sales data from a pizza store across four inter connected tables: orders, order_details, pizza_types, and pizzas (as depicted in the accompanying Entity-Relationship (ER)diagram). Throughout our analysis, we address a spectrumof questions ranging from fundamental to advanced levels of complexity. By leveraging MySQL's robust querying capabilities, this project aims to find useful information that can improve how the pizza business operates and help it grow.

BASIC

- RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.
- LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.
- CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.
- IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.
- IDENTIFY THE HIGHEST-PRICED PIZZA.

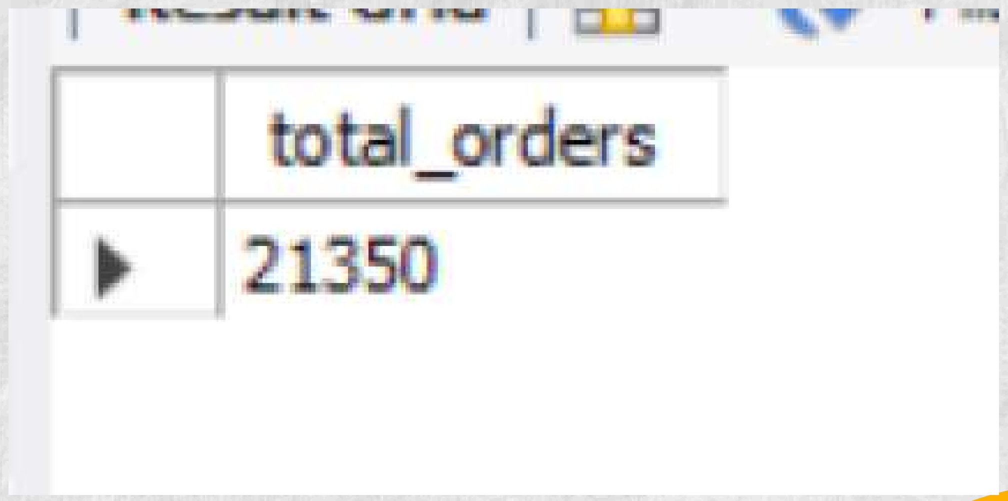
1. RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

QUERY:



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

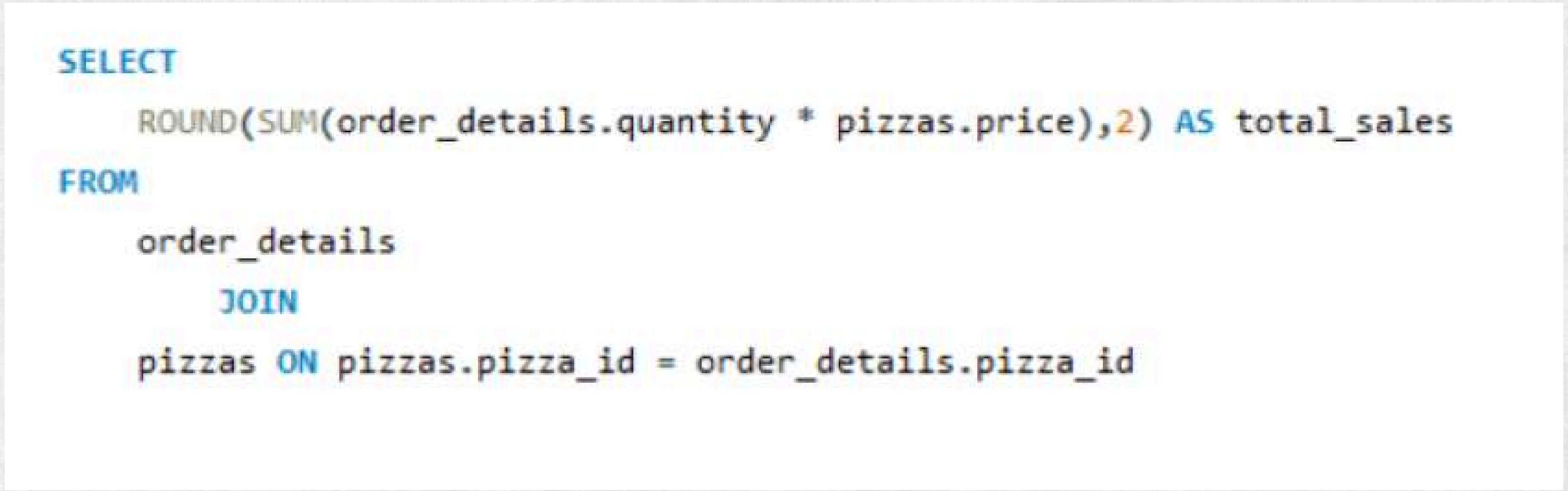
OUTPUT:



```
   total_orders
▶  21350
```

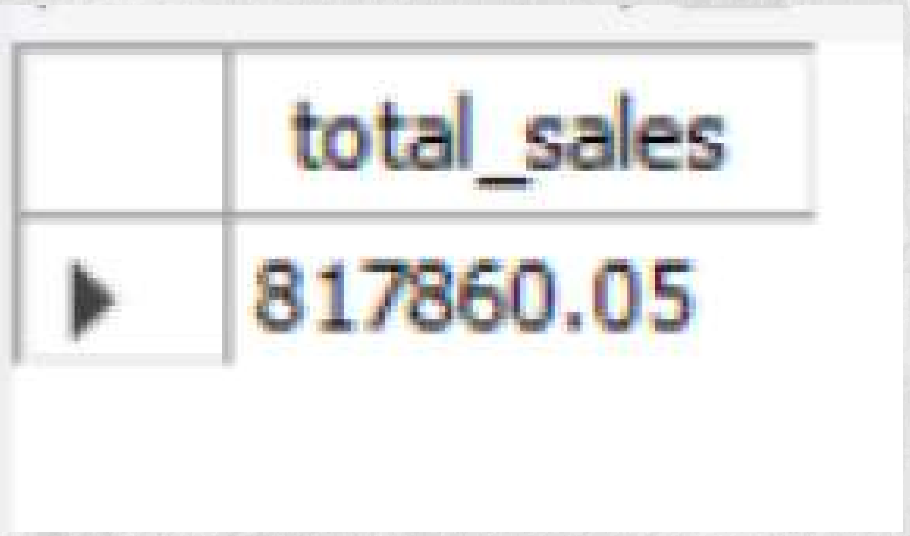
2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

QUERY:



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

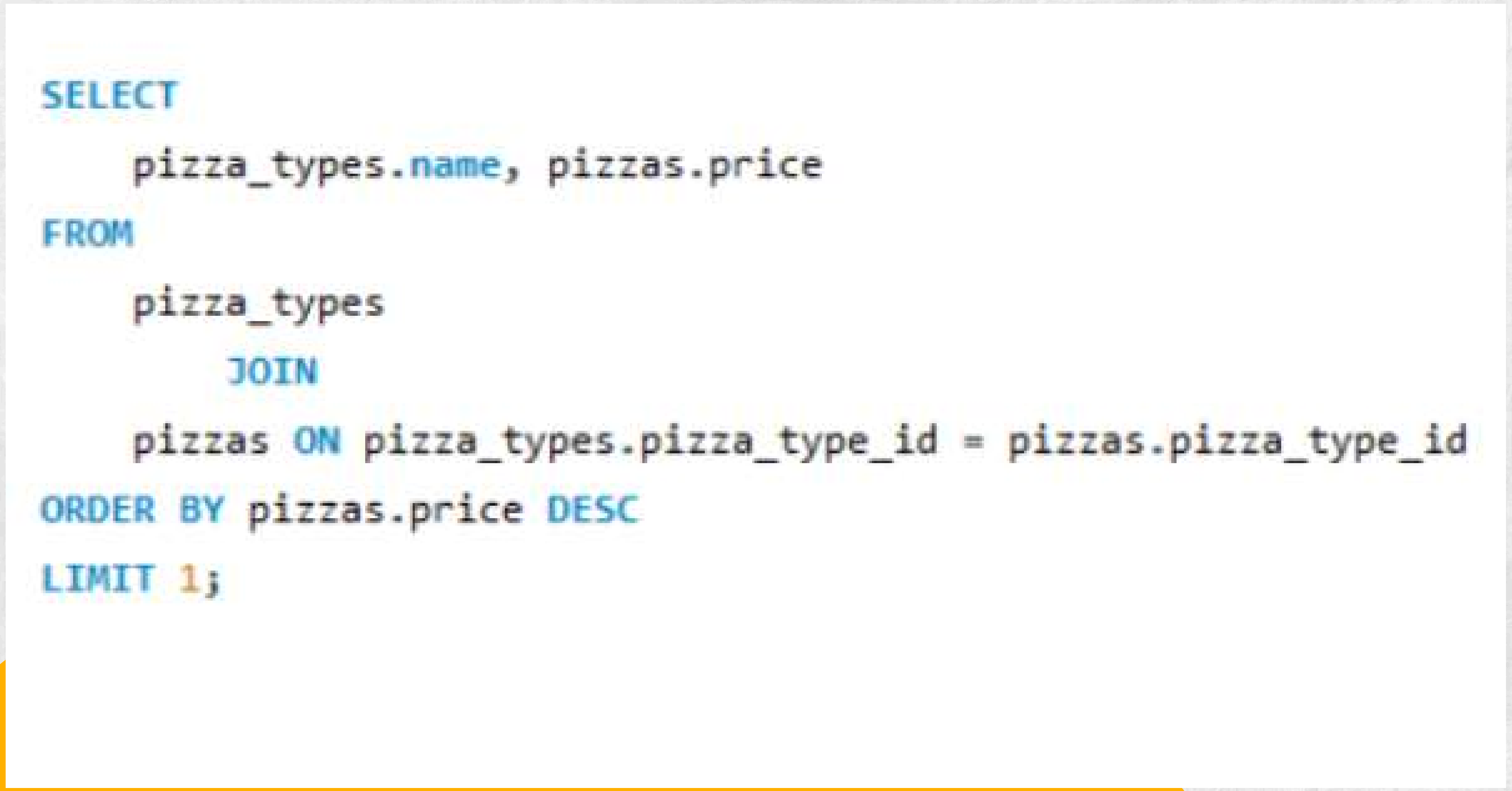
OUTPUT:



```
   total_sales
▶  817860.05
```

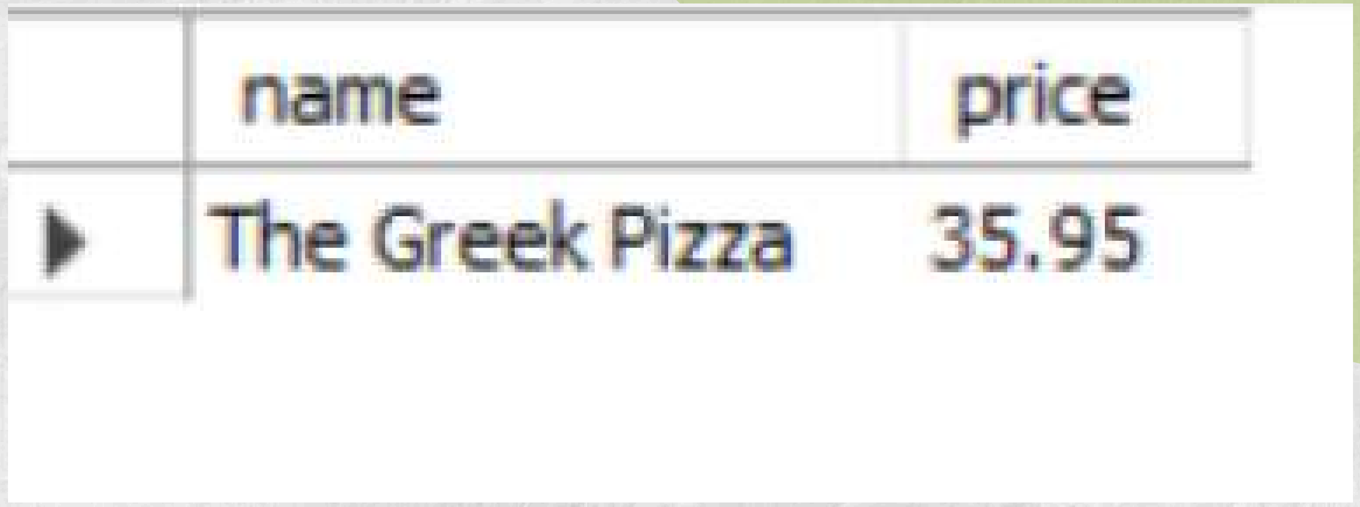
3. IDENTIFY THE HIGHEST-PRICED PIZZA.

QUERY:



```
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;
```

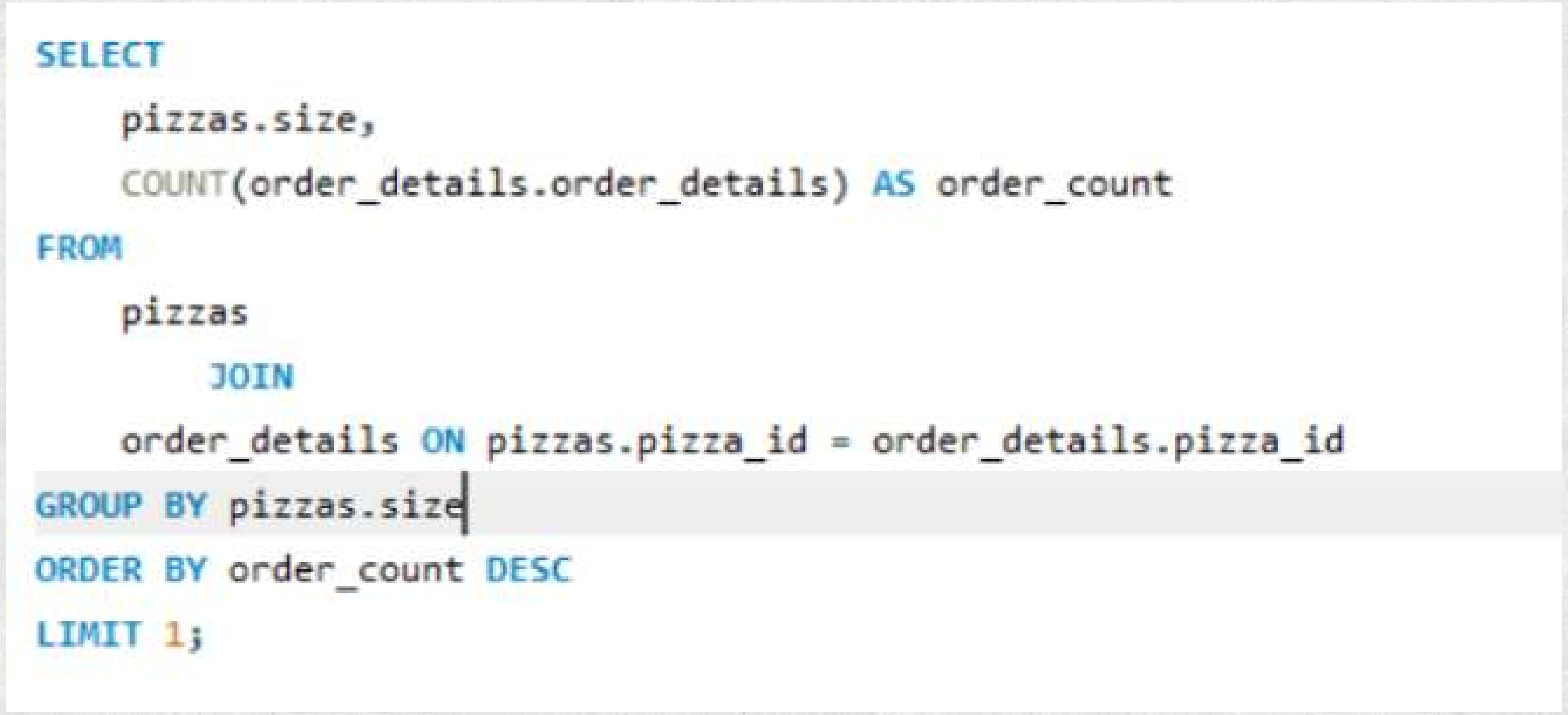
OUTPUT:



	name	price
▶	The Greek Pizza	35.95

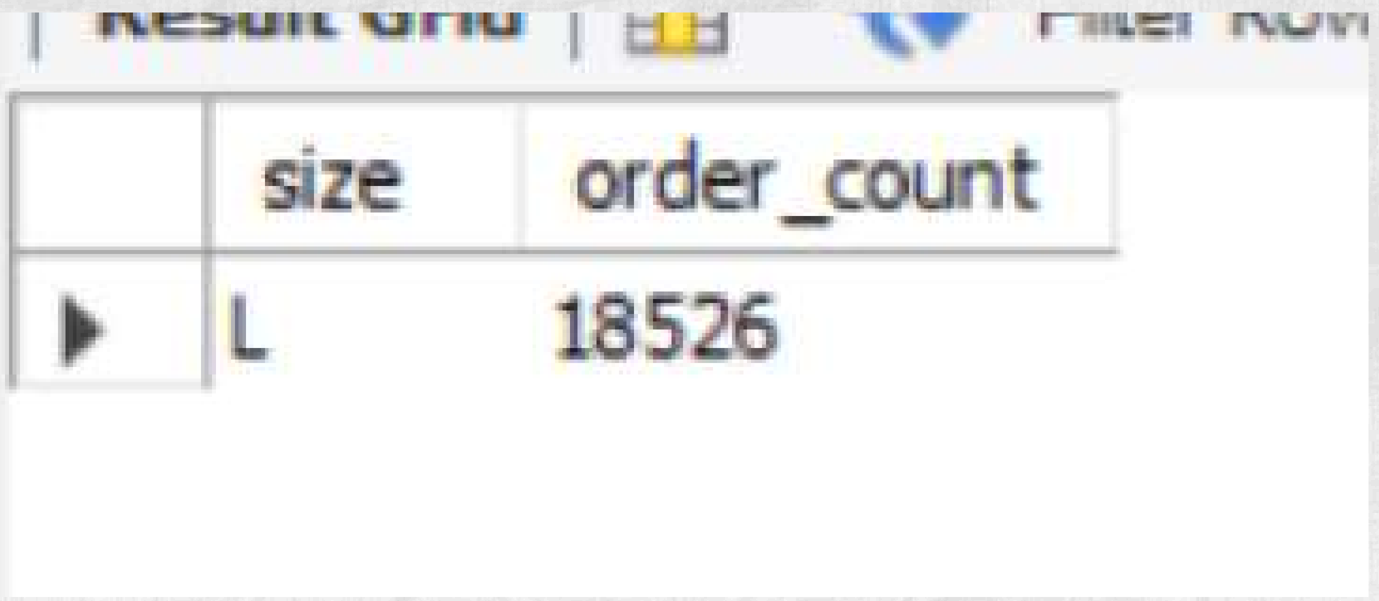
4. IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

QUERY:



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

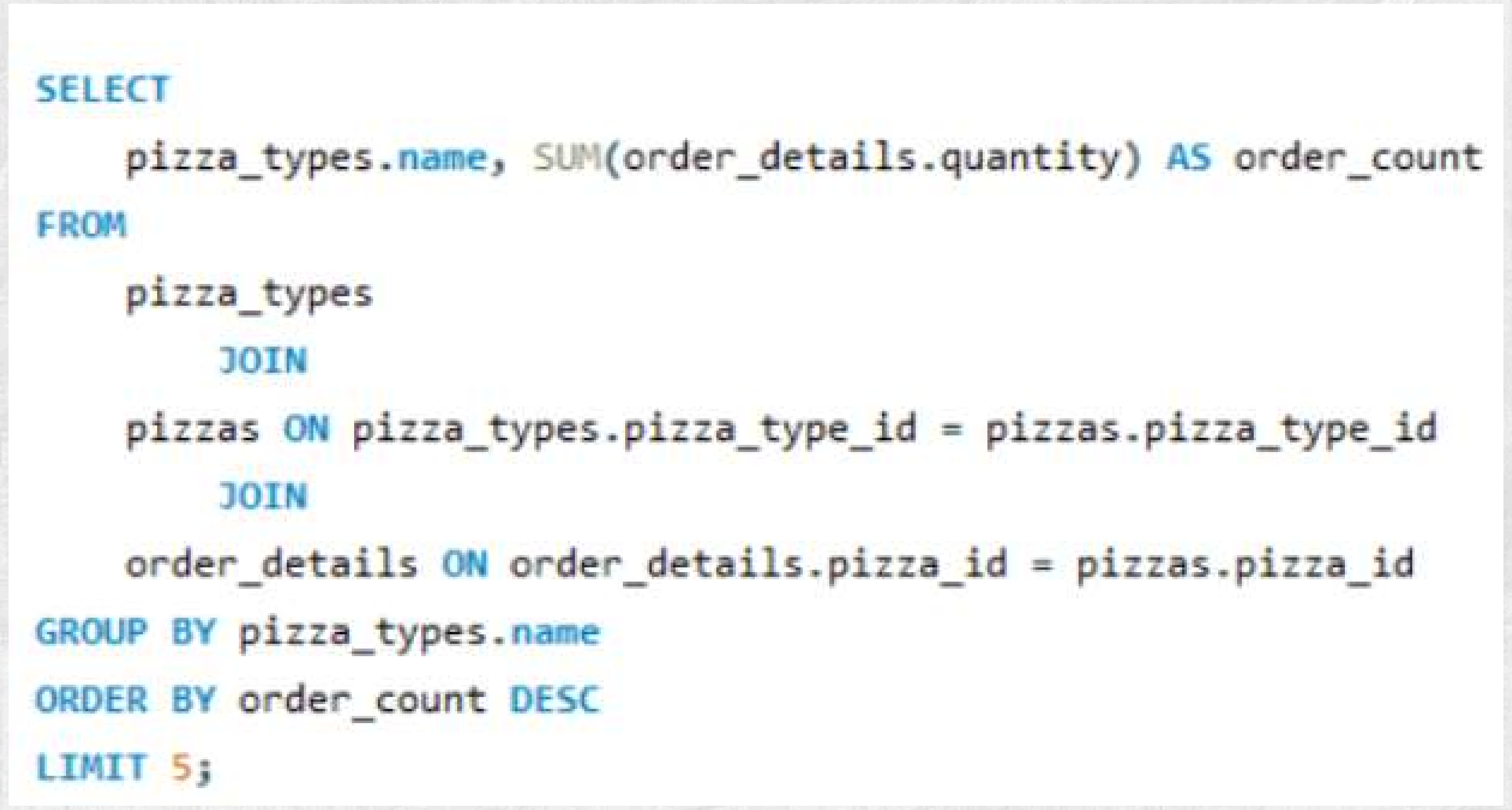
OUTPUT:



```
  size   order_count
▶ L      18526
```

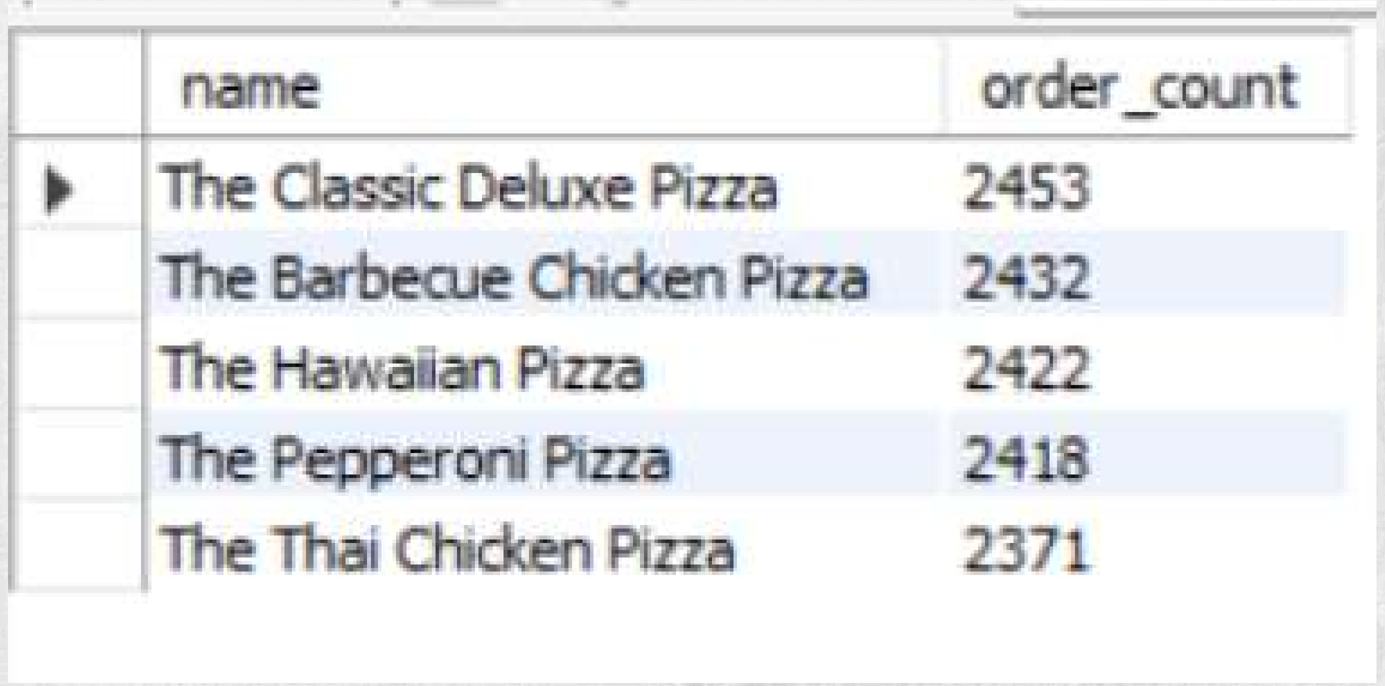
5. LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES

QUERY:



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

OUTPUT:



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

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

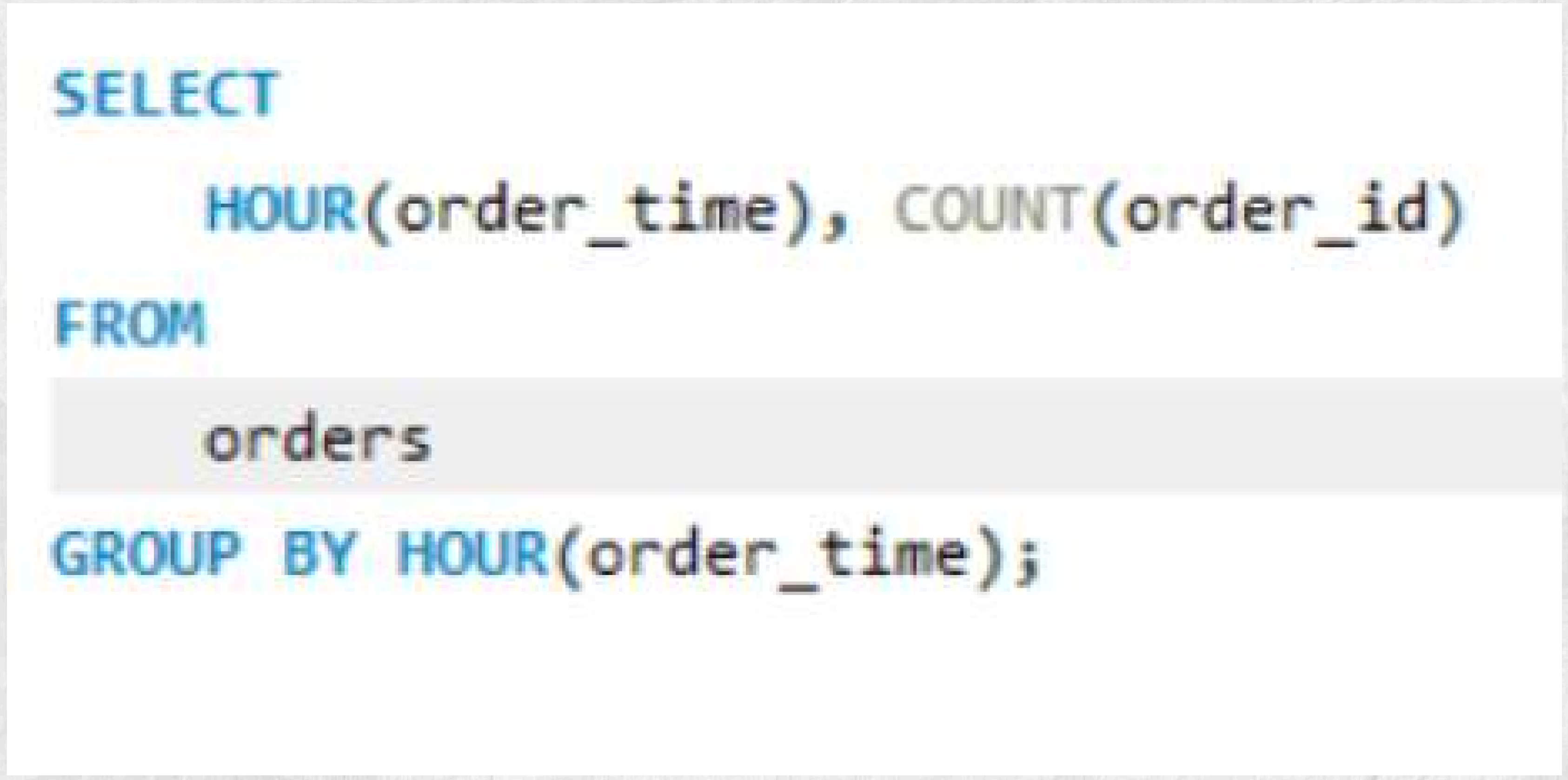
- GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

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

INTERMEDIATE

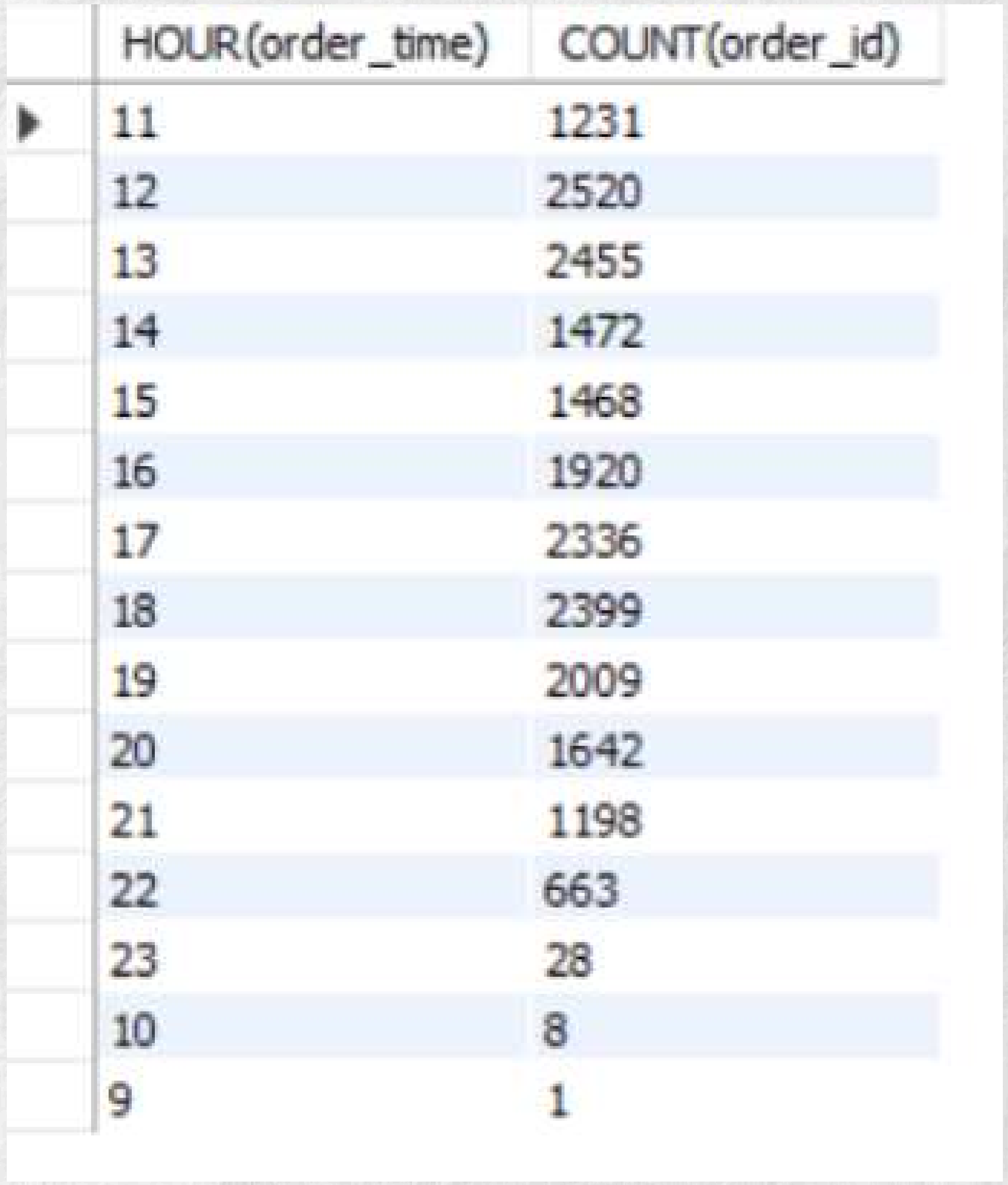
1. DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

QUERY:



```
SELECT
    HOUR(order_time), COUNT(order_id)
FROM
    orders
GROUP BY HOUR(order_time);
```

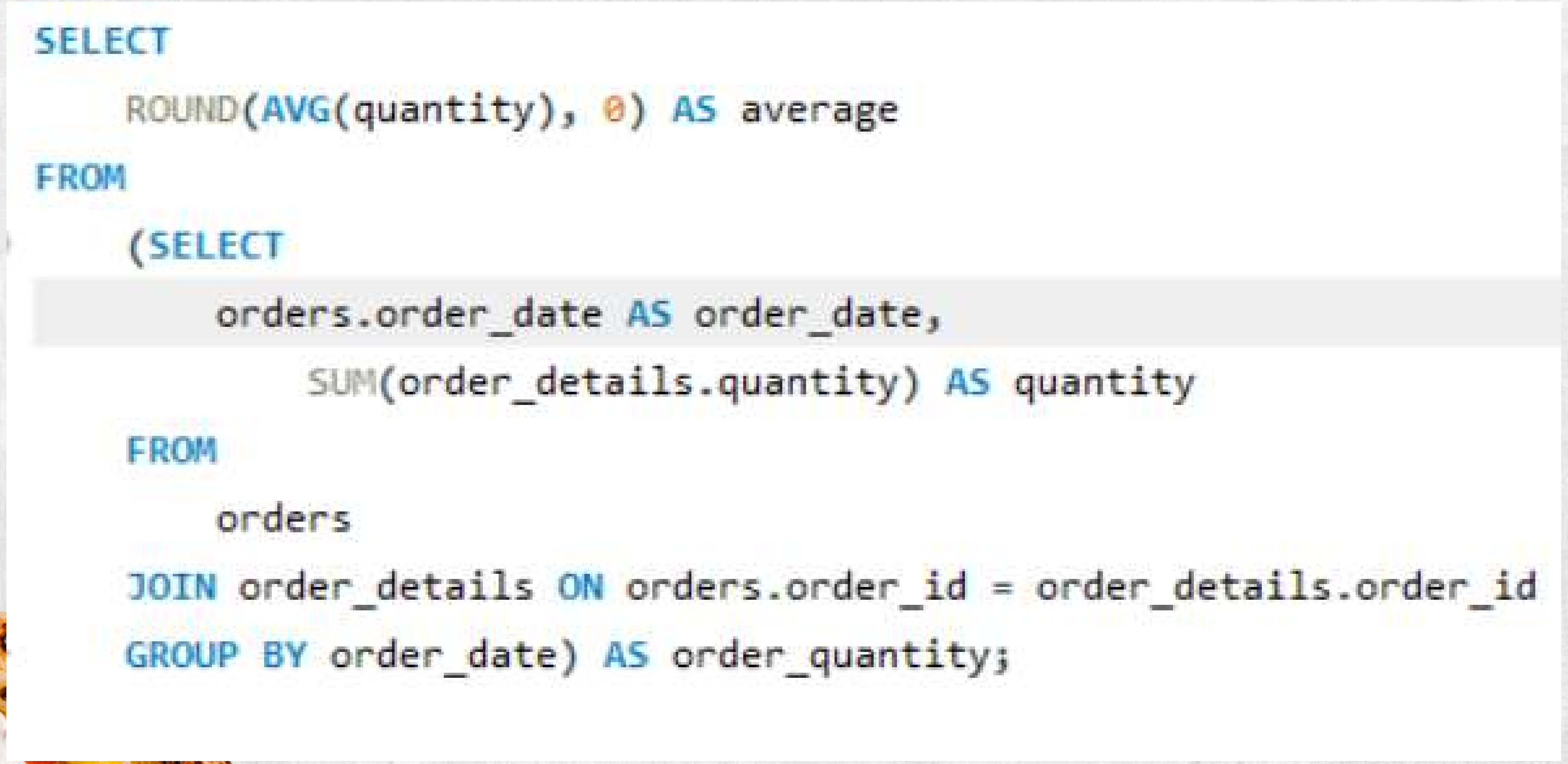
OUTPUT:



HOUR(order_time)	COUNT(order_id)
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

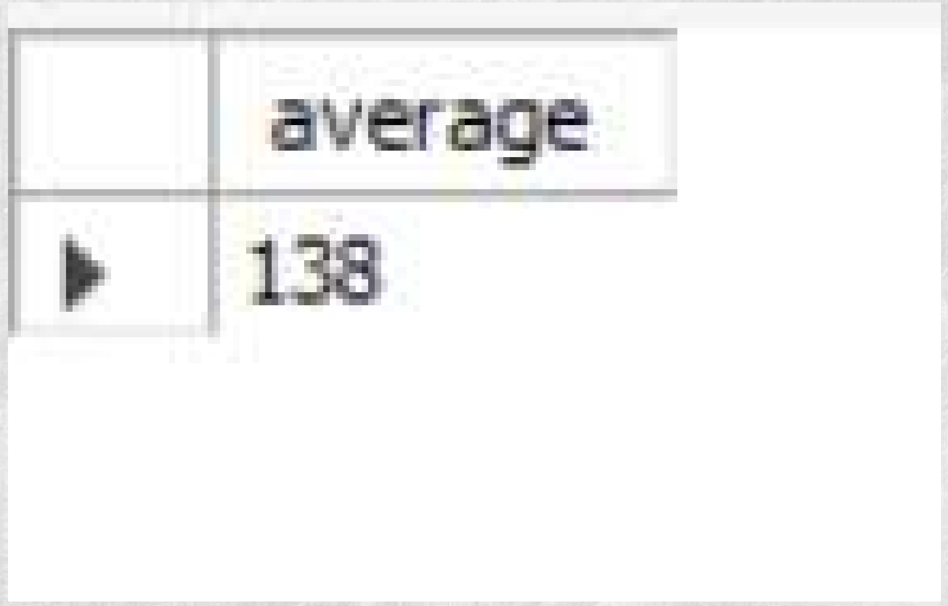
2. GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZASORDERED PER DAY.

QUERY:



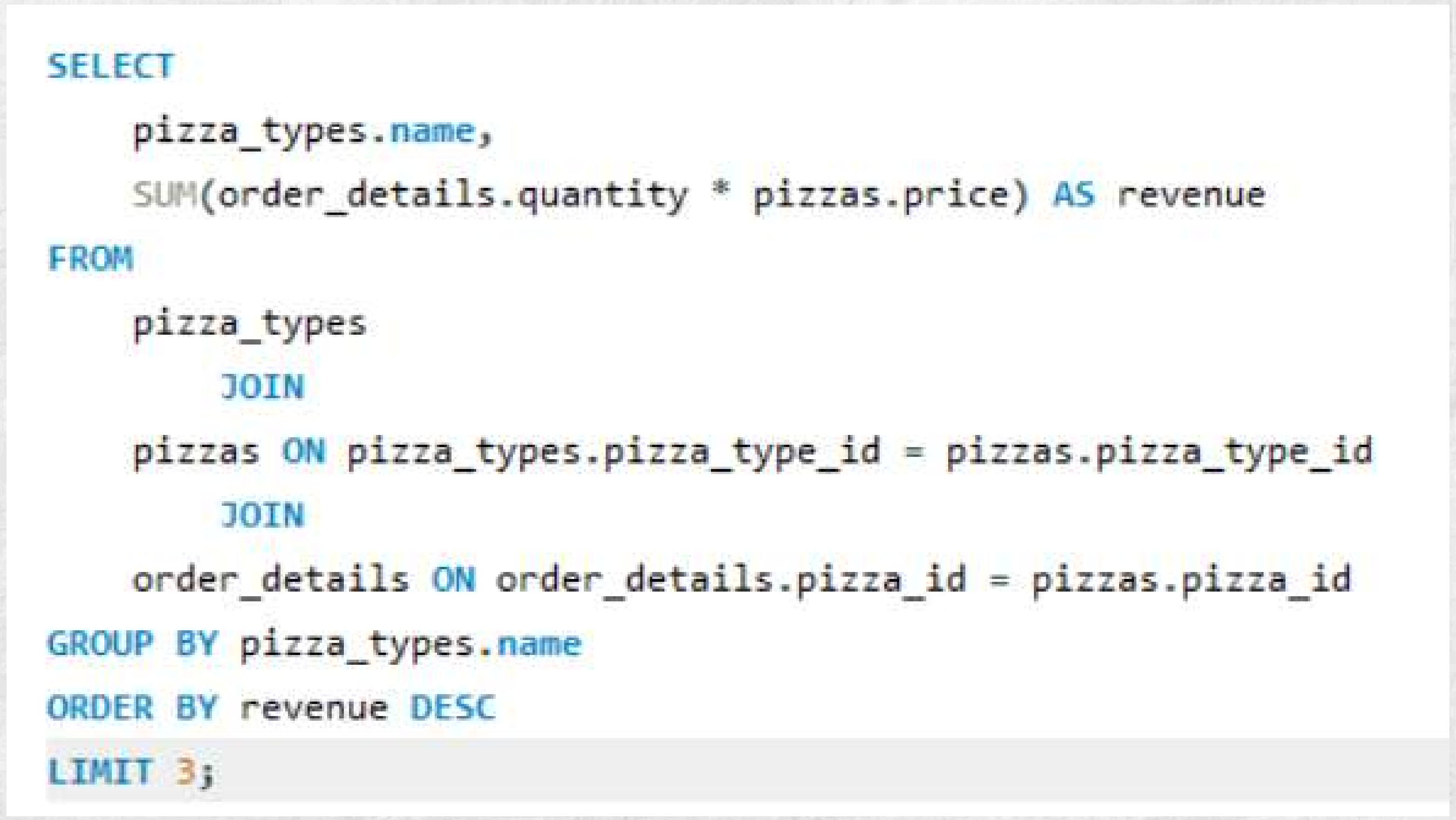
```
SELECT
    ROUND(AVG(quantity), 0) AS average
FROM
    (SELECT
        orders.order_date AS order_date,
            SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY order_date) AS order_quantity;
```

OUTPUT:



```
  average
▶ 138
```

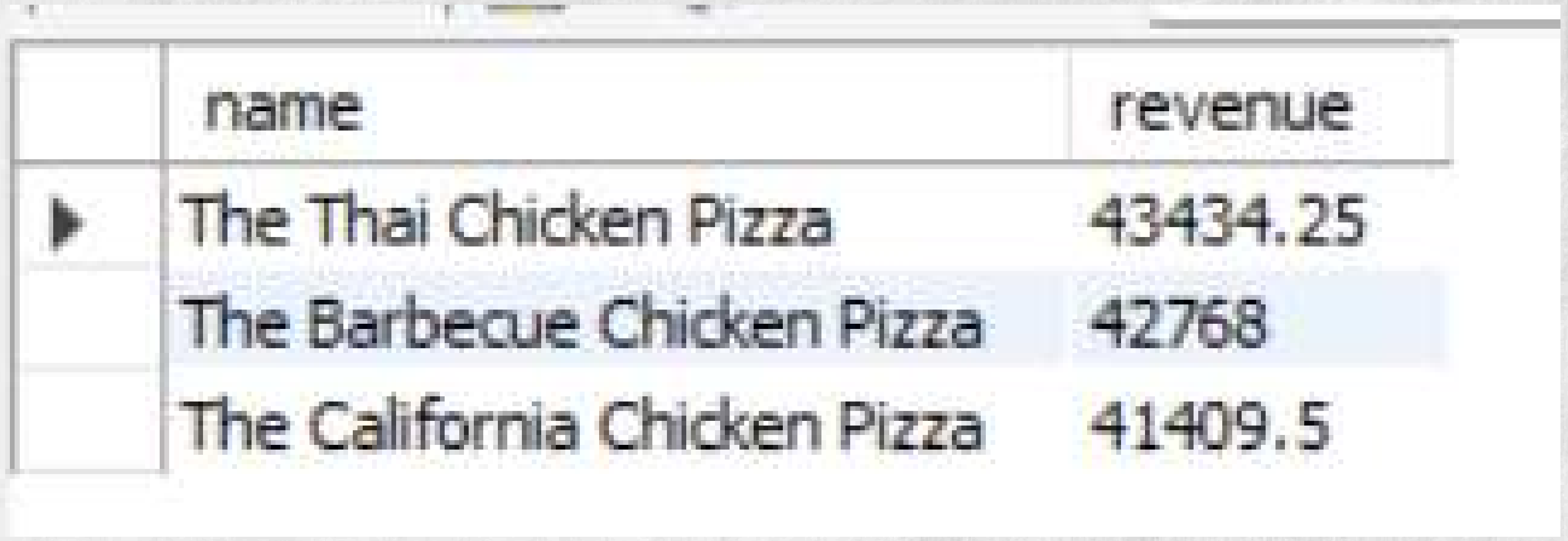
3.DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.



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

:QUERY

OUTPUT:



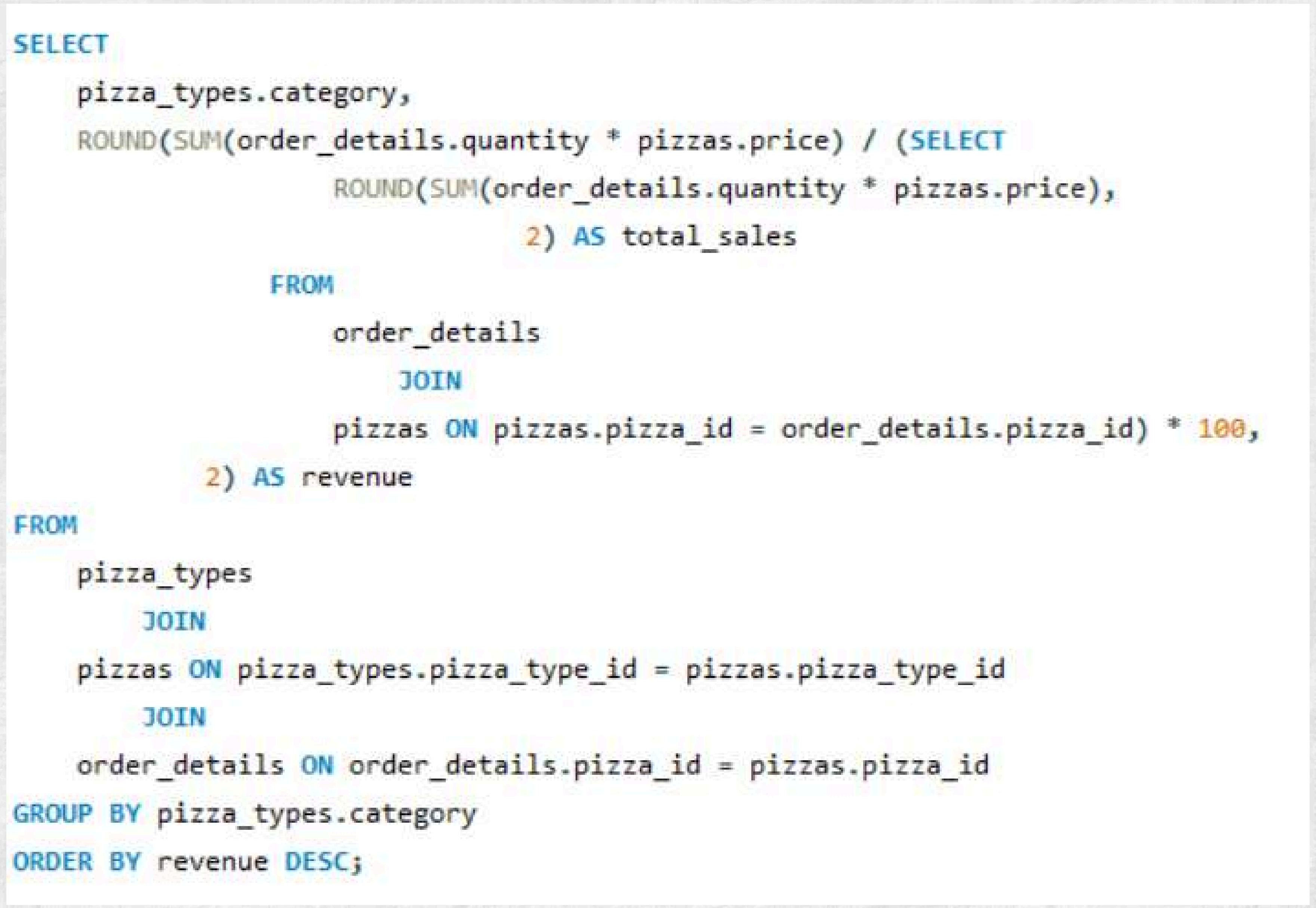
```
name                              revenue
The Thai Chicken Pizza            43434.25
The Barbecue Chicken Pizza        42768
The California Chicken Pizza      41409.5
```

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.

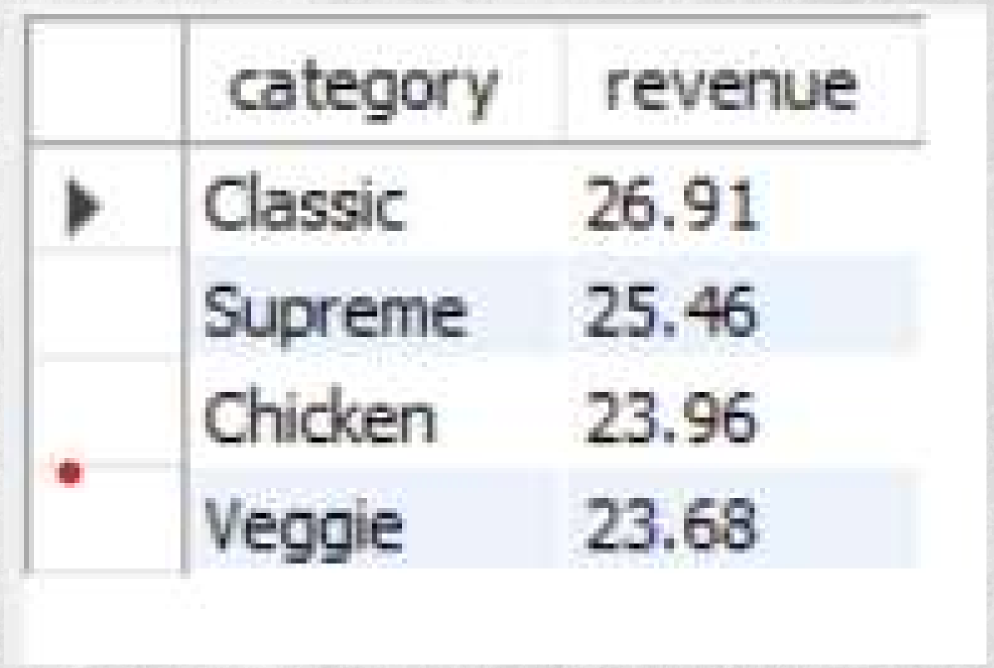
1. CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.

QUERY:



```
SELECT
    pizza_types.category,
    ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
                ROUND(SUM(order_details.quantity * pizzas.price),
                            2) AS total_sales
            FROM
                order_details
                    JOIN
                pizzas ON pizzas.pizza_id = order_details.pizza_id) * 100,
            2) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC;
```

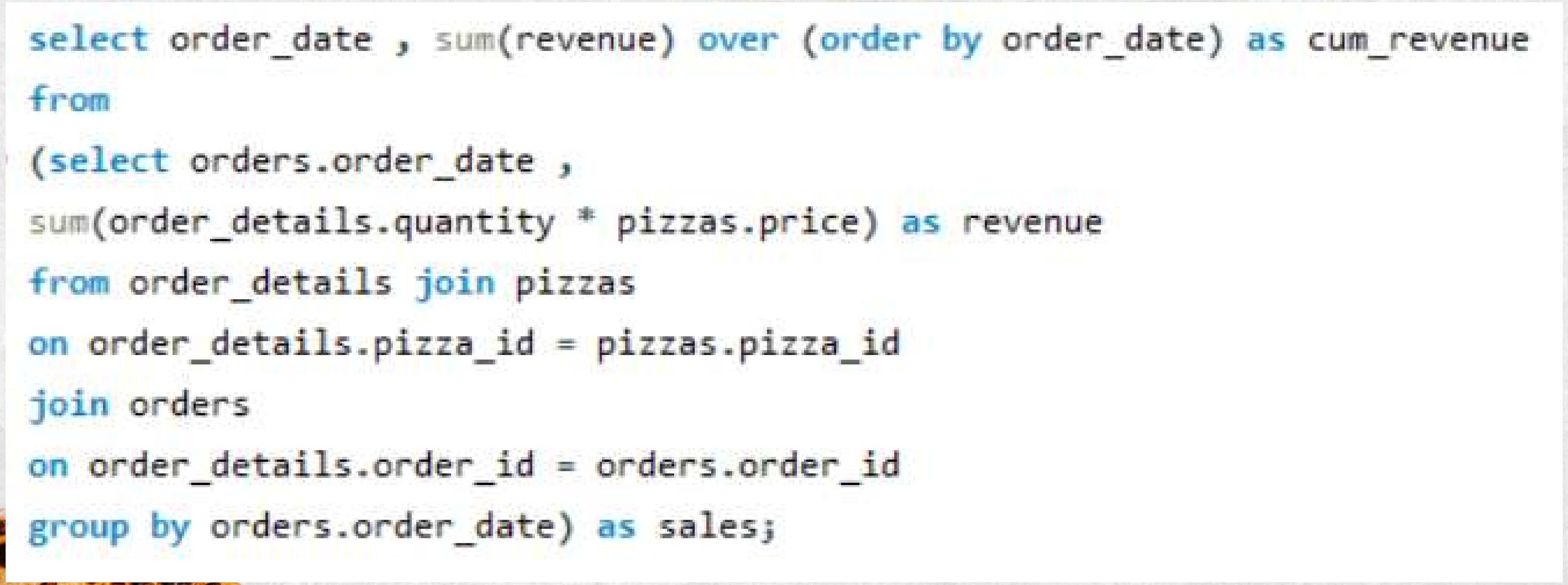
OUTPUT:



category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

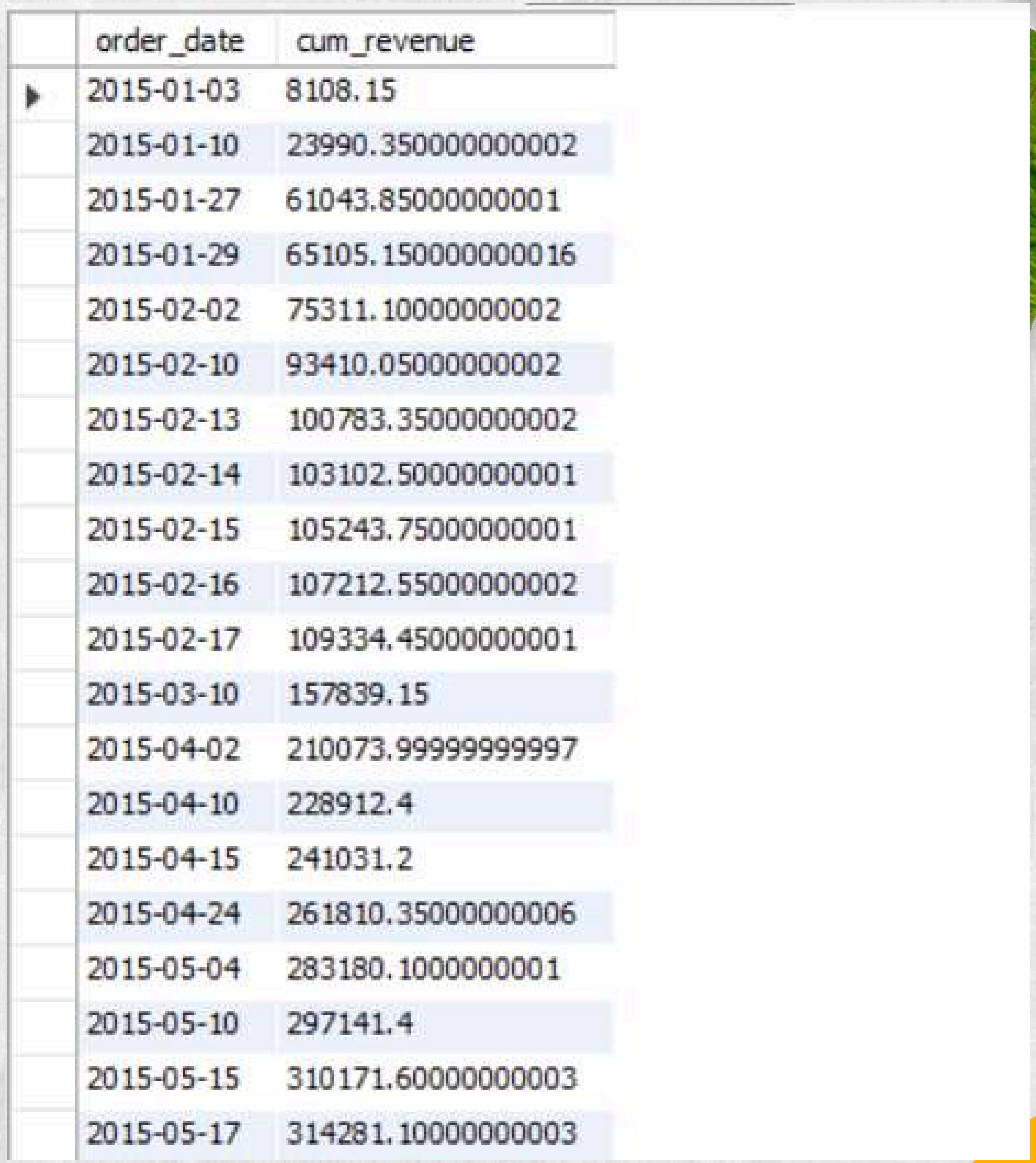
2. GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZASORDERED PER DAY.

QUERY:



```
select order_date , sum(revenue) over (order by order_date) as cum_revenue
from
(select orders.order_date ,
sum(order_details.quantity * pizzas.price) as revenue
from order_details join pizzas
on order_details.pizza_id = pizzas.pizza_id
join orders
on order_details.order_id = orders.order_id
group by orders.order_date) as sales;
```

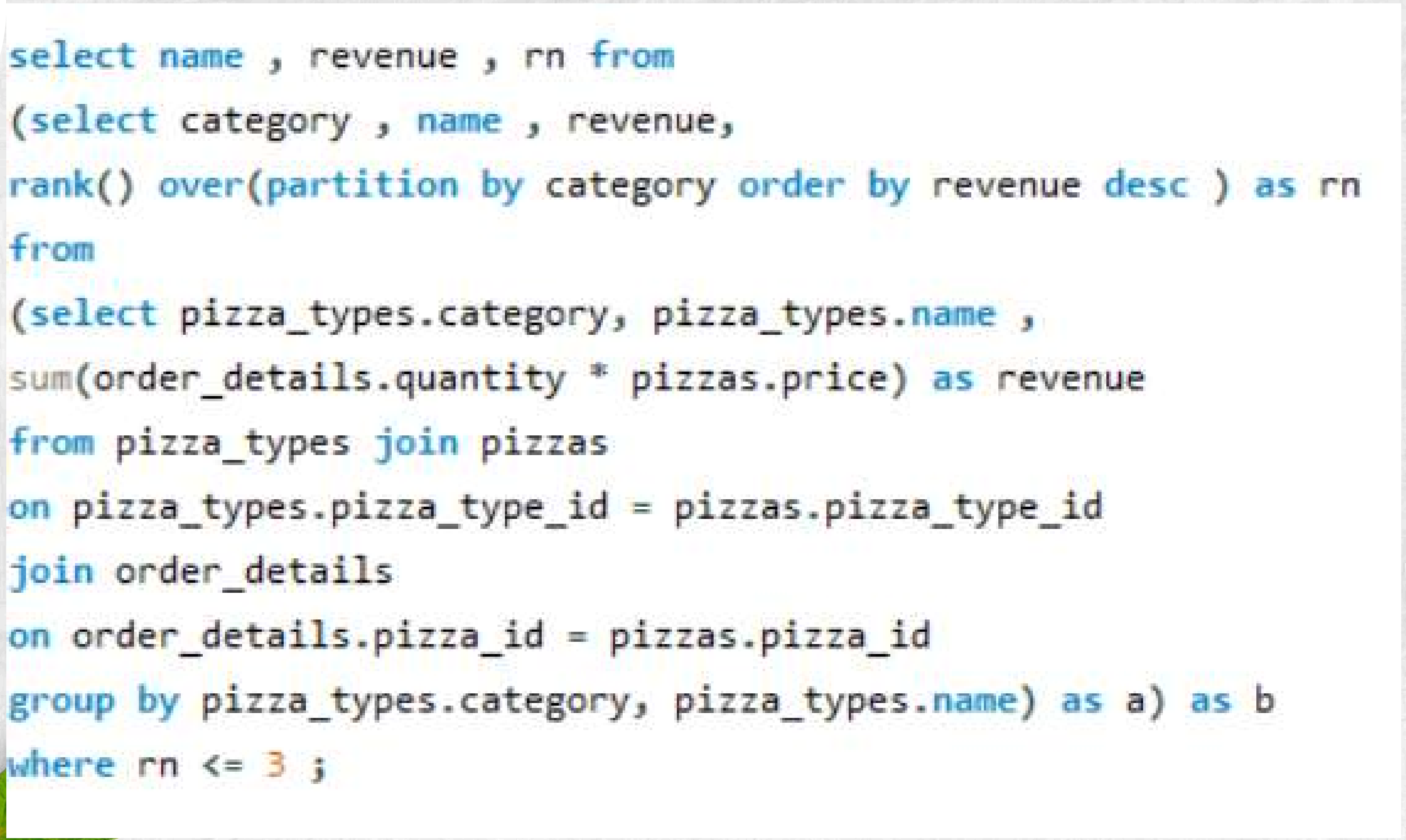
OUTPUT:



```
order_date   cum_revenue
2015-01-03   8108.15
2015-01-10   23990.350000000002
2015-01-27   61043.85000000001
2015-01-29   65105.150000000016
2015-02-02   75311.10000000002
2015-02-10   93410.05000000002
2015-02-13   100783.35000000002
2015-02-14   103102.50000000001
2015-02-15   105243.75000000001
2015-02-16   107212.55000000002
2015-02-17   109334.45000000001
2015-03-10   157839.15
2015-04-02   210073.99999999997
2015-04-10   228912.4
2015-04-15   241031.2
2015-04-24   261810.35000000006
2015-05-04   283180.1000000001
2015-05-10   297141.4
2015-05-15   310171.60000000003
2015-05-17   314281.10000000003
```

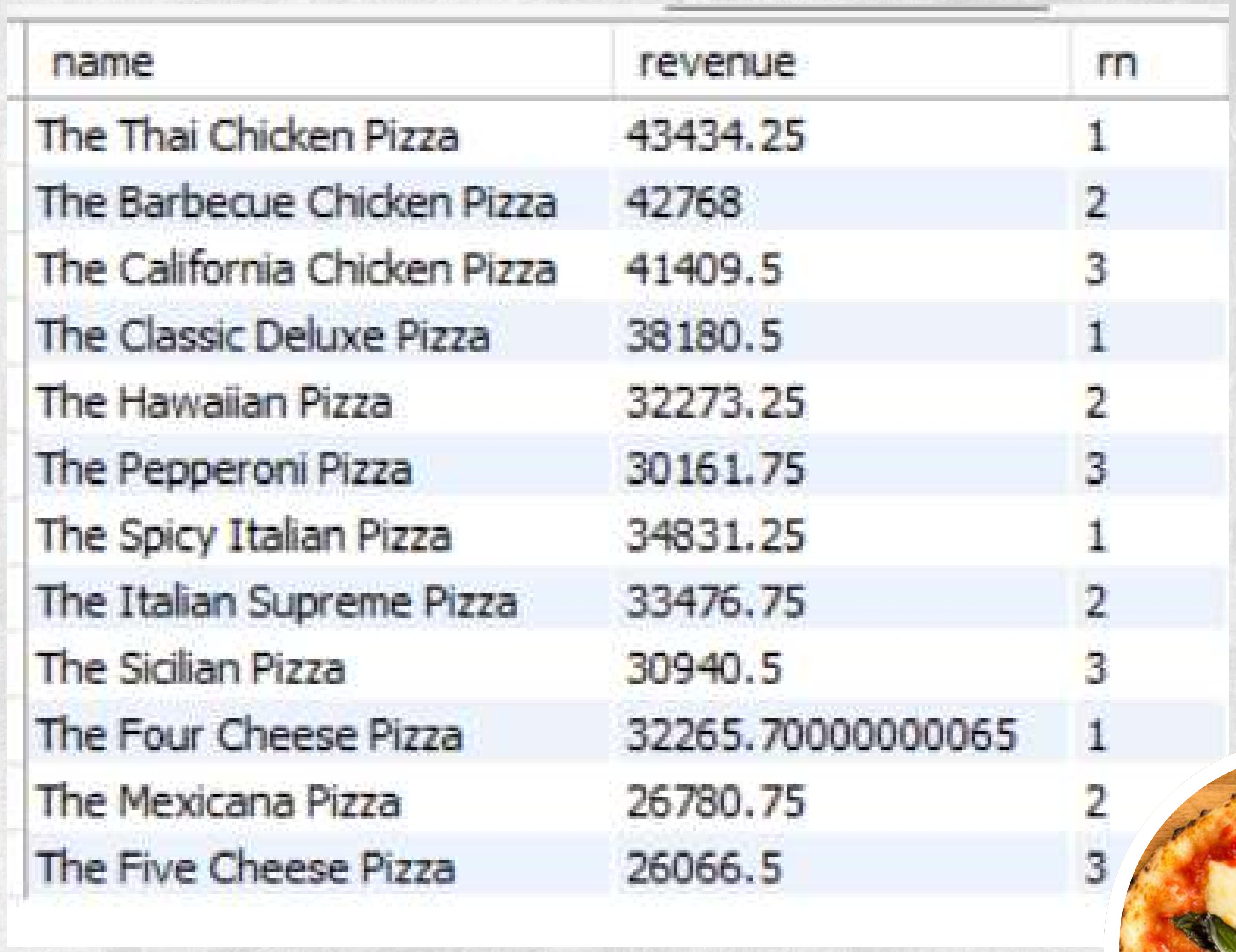
3. DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACHPIZZA CATEGORY.

QUERY:

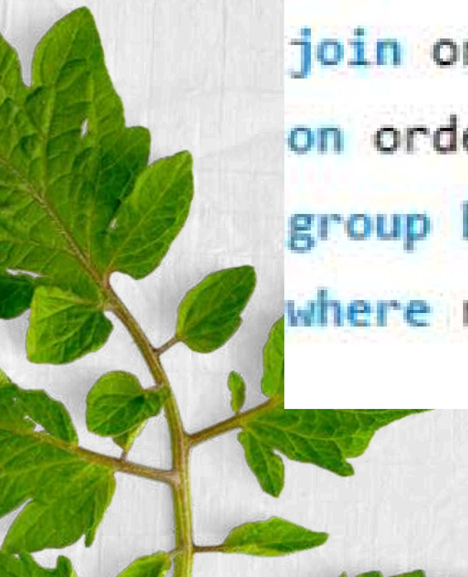


```
select name , revenue , rn from
(select category , name , revenue,
rank() over(partition by category order by revenue desc ) as rn
from
(select pizza_types.category, pizza_types.name ,
sum(order_details.quantity * pizzas.price) as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join order_details
on order_details.pizza_id = pizzas.pizza_id
group by pizza_types.category, pizza_types.name) as a) as b
where rn <= 3 ;
```

OUTPUT:



name	revenue	rn
The Thai Chicken Pizza	43434.25	1
The Barbecue Chicken Pizza	42768	2
The California Chicken Pizza	41409.5	3
The Classic Deluxe Pizza	38180.5	1
The Hawaiian Pizza	32273.25	2
The Pepperoni Pizza	30161.75	3
The Spicy Italian Pizza	34831.25	1
The Italian Supreme Pizza	33476.75	2
The Sicilian Pizza	30940.5	3
The Four Cheese Pizza	32265.7000000065	1
The Mexicana Pizza	26780.75	2
The Five Cheese Pizza	26066.5	3



CONCLUSION

In this discussion, we take a deep dive into analyzing sales data from a pizza store using MySQL’s robust querying and analytical capabilities. The dataset consists of four primary tables: orders, order_details, pizza_types, and pizzas. These tables provide valuable insights into customer purchases, various pizza categories, and detailed sales records. By exploring this data, we seek to identify patterns in sales performance, customer preferences, and the impact of marketing strategies. Our objective is to extract meaningful insights that can drive better business decisions and optimize operations in the pizza industry. Stay with us as we uncover key trends, highlight essential takeaways, and discuss how these findings can shape the future of pizza sales.

THANK YOU