SPECIAL PIZZA

Life is short, eat the pizza!

ORDER NOW

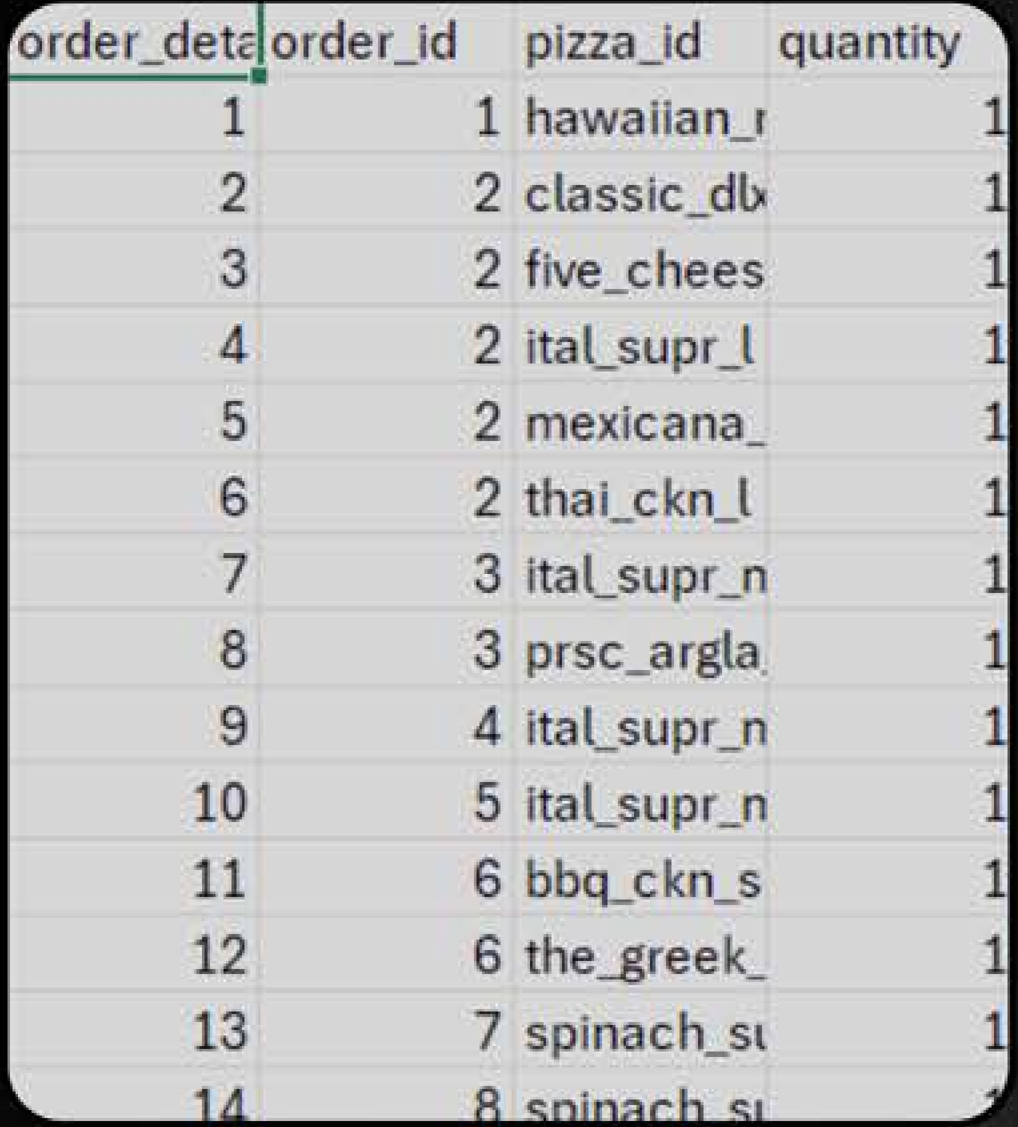
☎ +123-456-7890

📍 123 ANYWHERE ST., ANY CITY

HELLO!

My name is Keerthi and I have utilized advanced SQL queries to analyze and solve complex problems related to pizza sales.The project was not only intellectually stimulating but also enjoyable, highlighting the engaging and dynamic nature of data analysis in real-world scenarios.

ABOUT THE DATA

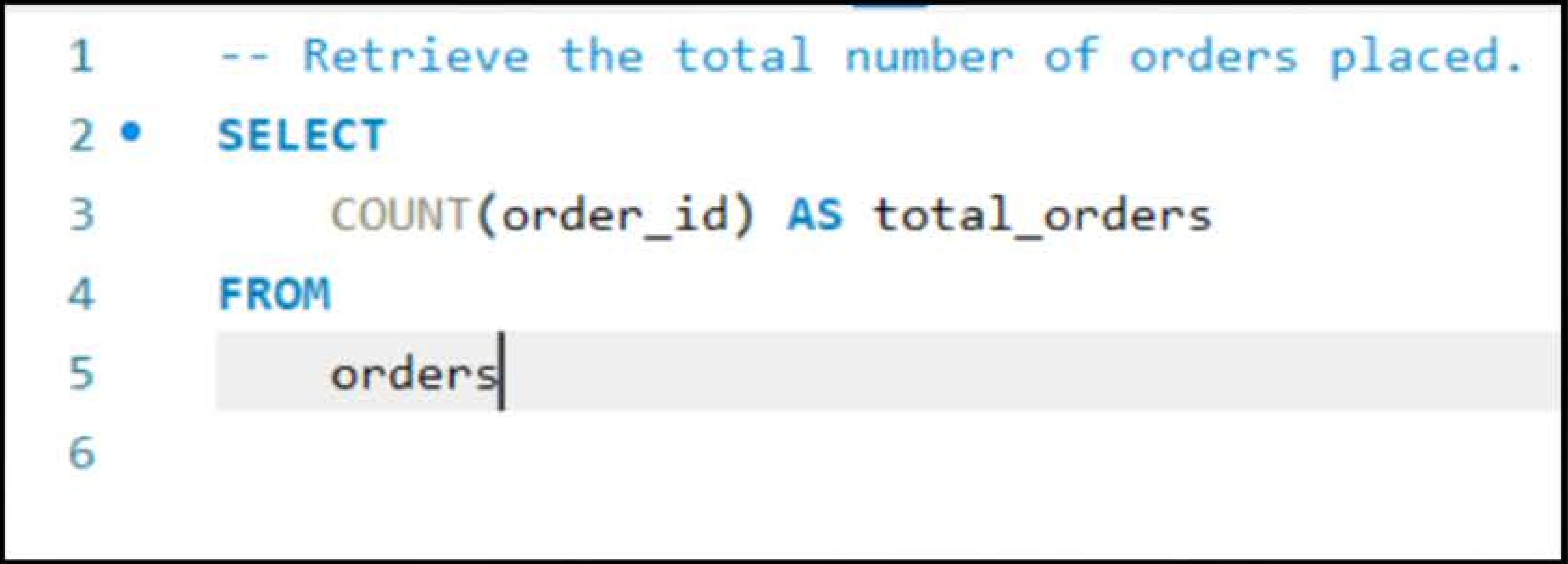


order_deta	order_id	pizza_id	quantity
1	1	hawaiian_	1
2	2	classic_dl	1
3	2	five_chees	1
4	2	ital_supr_l	1
5	2	mexicana_	1
6	2	thai_ckn_l	1
7	3	ital_supr_n	1
8	3	prsc_argla	1
9	4	ital_supr_n	1
10	5	ital_supr_n	1
11	6	bbq_ckn_s	1
12	6	the_greek_	1
13	7	spinach_su	1
14	8	spinach_su	1

The dataset contains comprehensive information on pizza sales, including orders, pizza types, sizes, and prices. It is structured to enable detailed analysis of sales performance, such as calculating total revenue, identifying popular pizza types and sizes, and examining sales trends over time. This data allows for insights into ordering patterns, revenue distribution,and the effectiveness of different pizza categories.

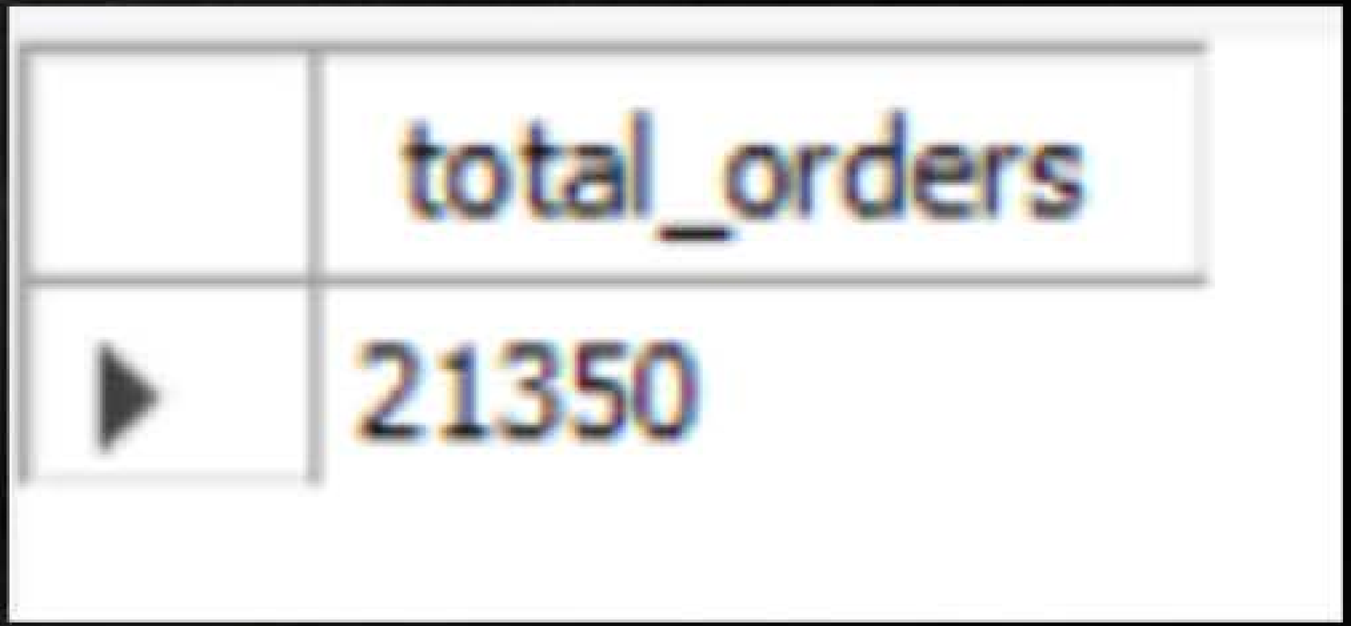
Retrieve the total number of orders placed.

input



```sql
-- Retrieve the total number of orders placed.
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders
```

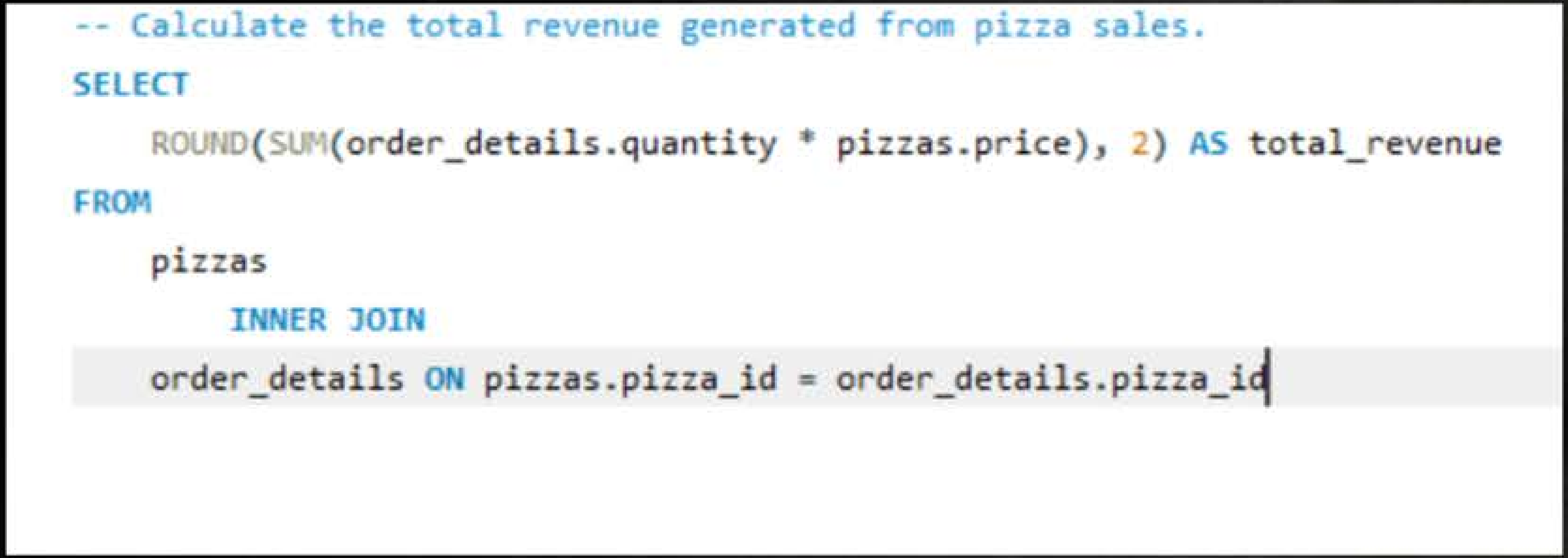
output



total_orders
21350

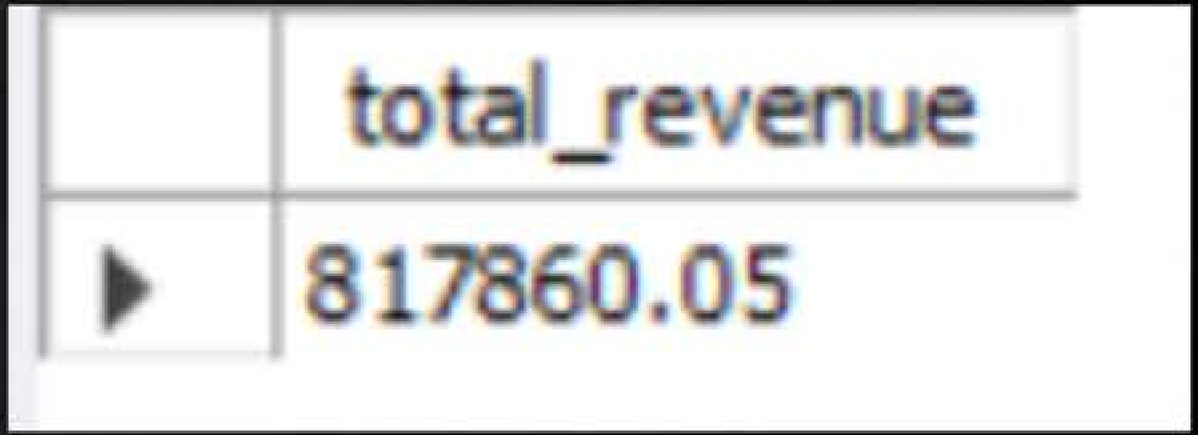
Calculate the total revenue generated from pizza sales.

input



```
-- Calculate the total revenue generated from pizza sales.
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price), 2) AS total_revenue
FROM
    pizzas
        INNER JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
```

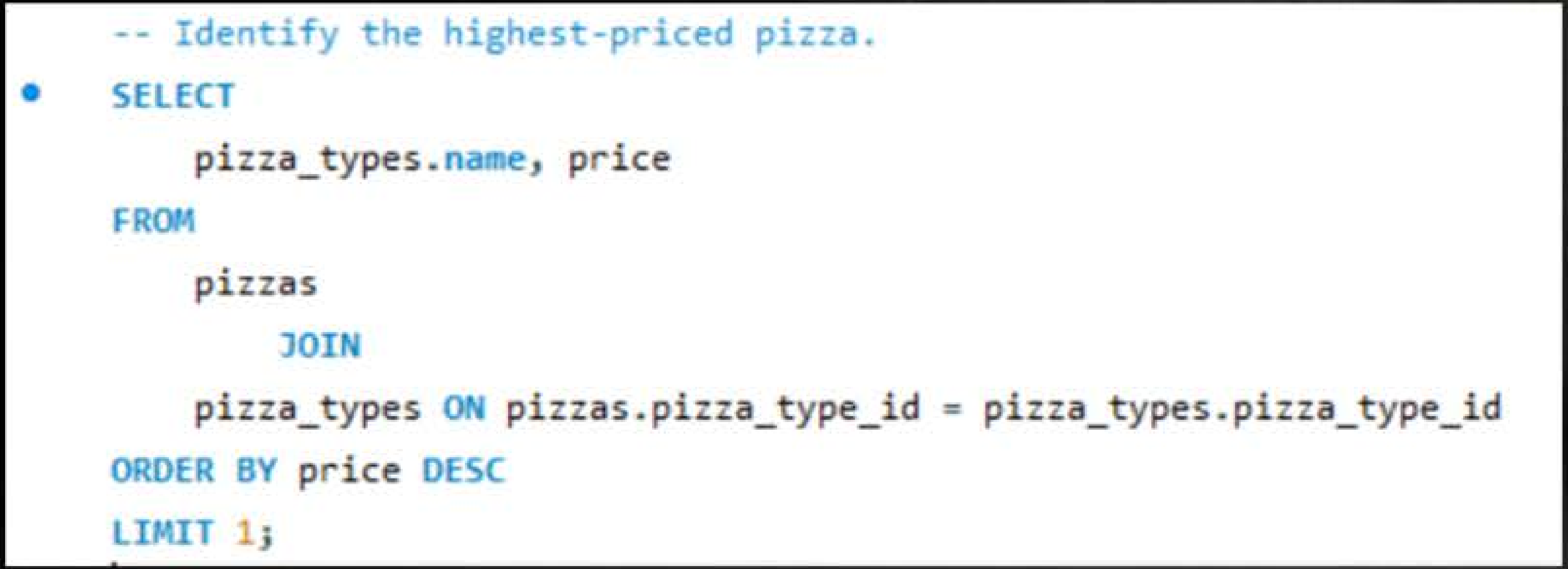
output



	total_revenue
▶	817860.05

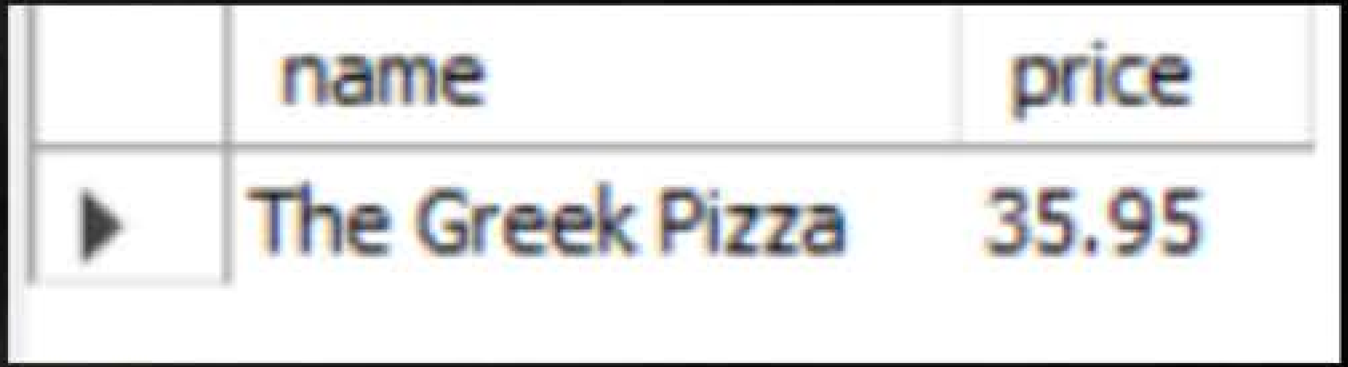
Identify the highest-priced pizza.

input



```
-- Identify the highest-priced pizza.
SELECT
    pizza_types.name, price
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
ORDER BY price DESC
LIMIT 1;
```

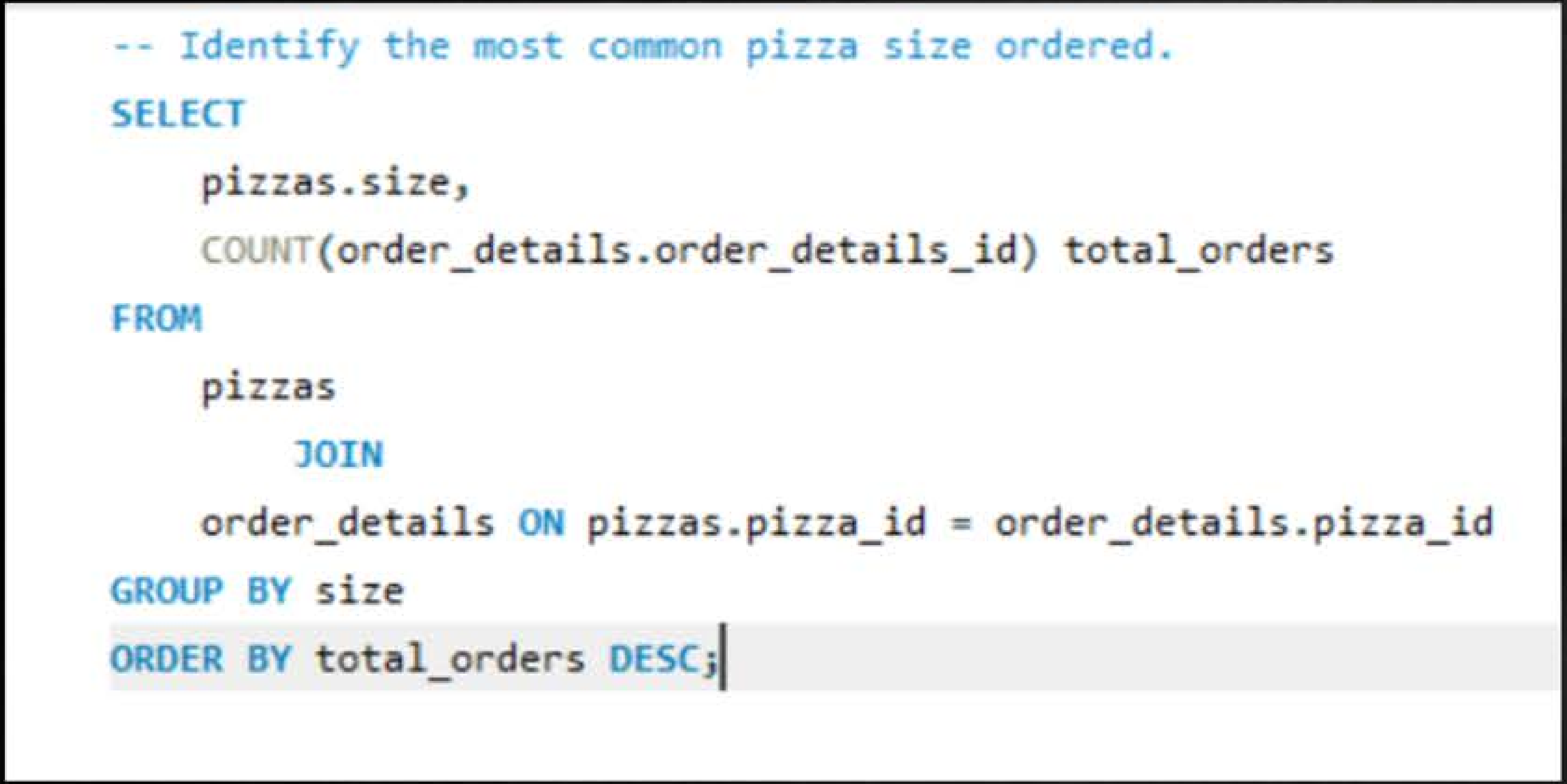
output



	name	price
▶	The Greek Pizza	35.95

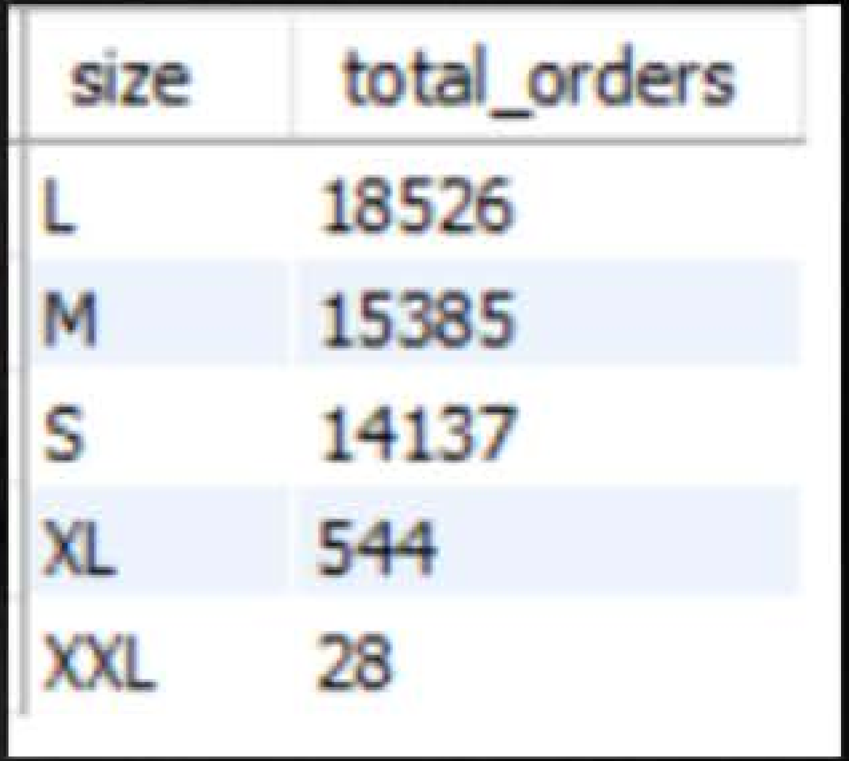
Identify the most common pizza size ordered.

input



```sql
-- Identify the most common pizza size ordered.
SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) total_orders
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY size
ORDER BY total_orders DESC;
```

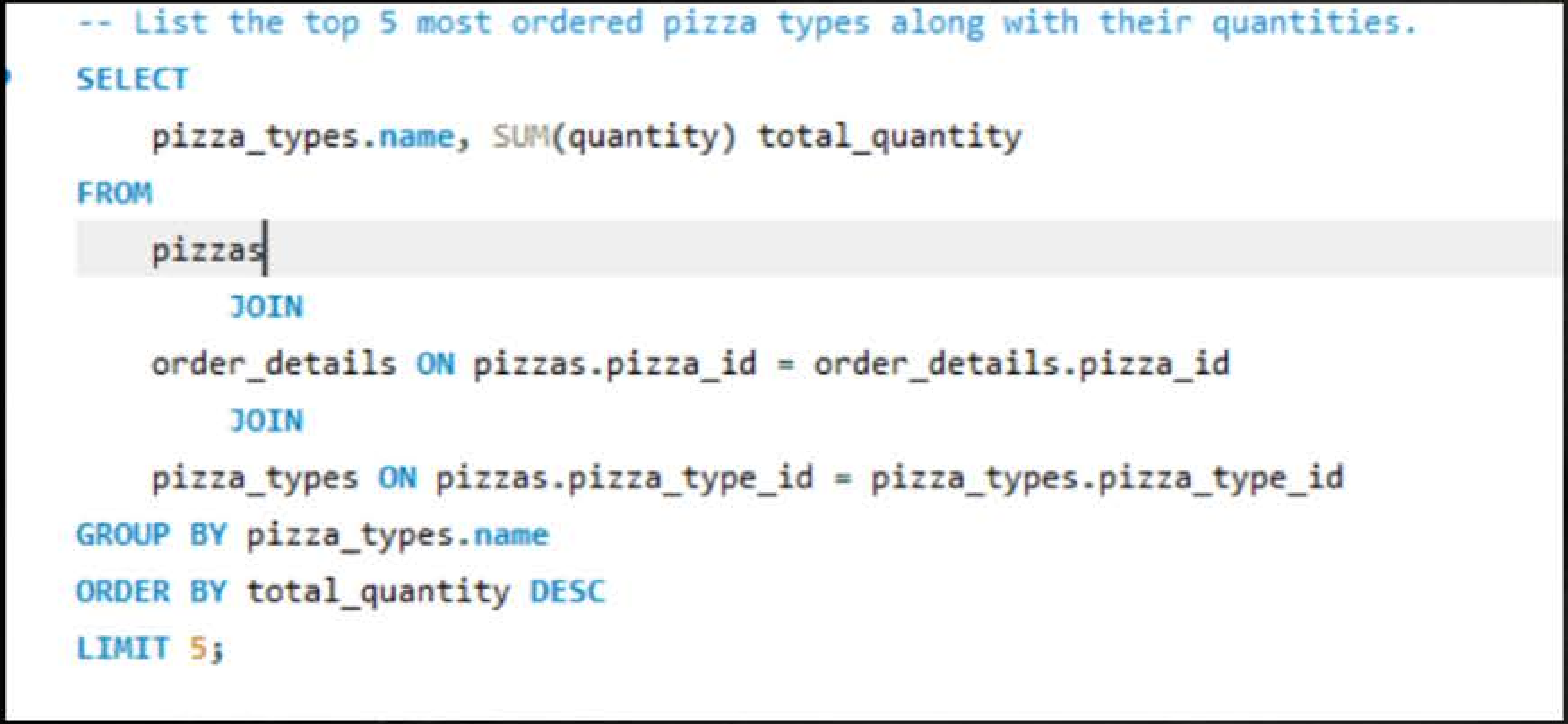
output



size	total_orders
L	18526
M	15385
S	14137
XL	544
XXL	28

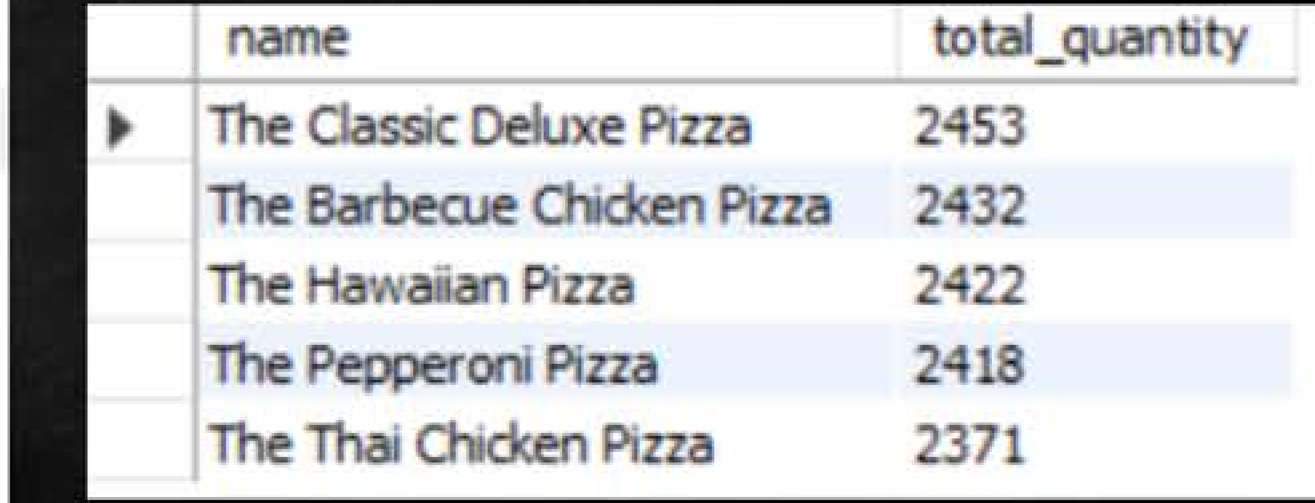
List the top 5 most ordered pizza types along with their quantities.

input



```
-- List the top 5 most ordered pizza types along with their quantities.
SELECT
    pizza_types.name, SUM(quantity) total_quantity
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
GROUP BY pizza_types.name
ORDER BY total_quantity DESC
LIMIT 5;
```

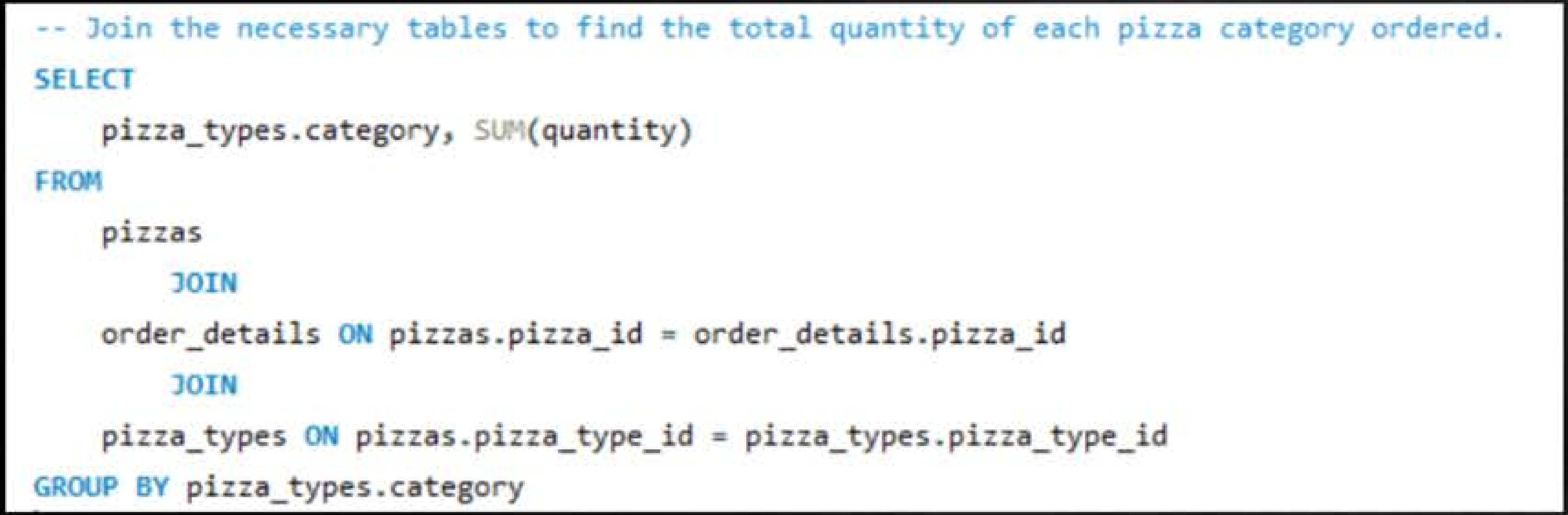
output



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

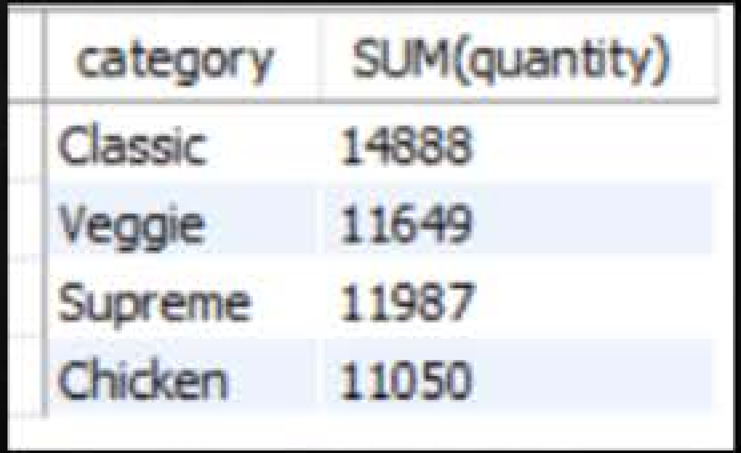
Join the necessary tables to find the total quantity of each pizza category ordered.

input



```
-- Join the necessary tables to find the total quantity of each pizza category ordered.
SELECT
    pizza_types.category, SUM(quantity)
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
GROUP BY pizza_types.category
```

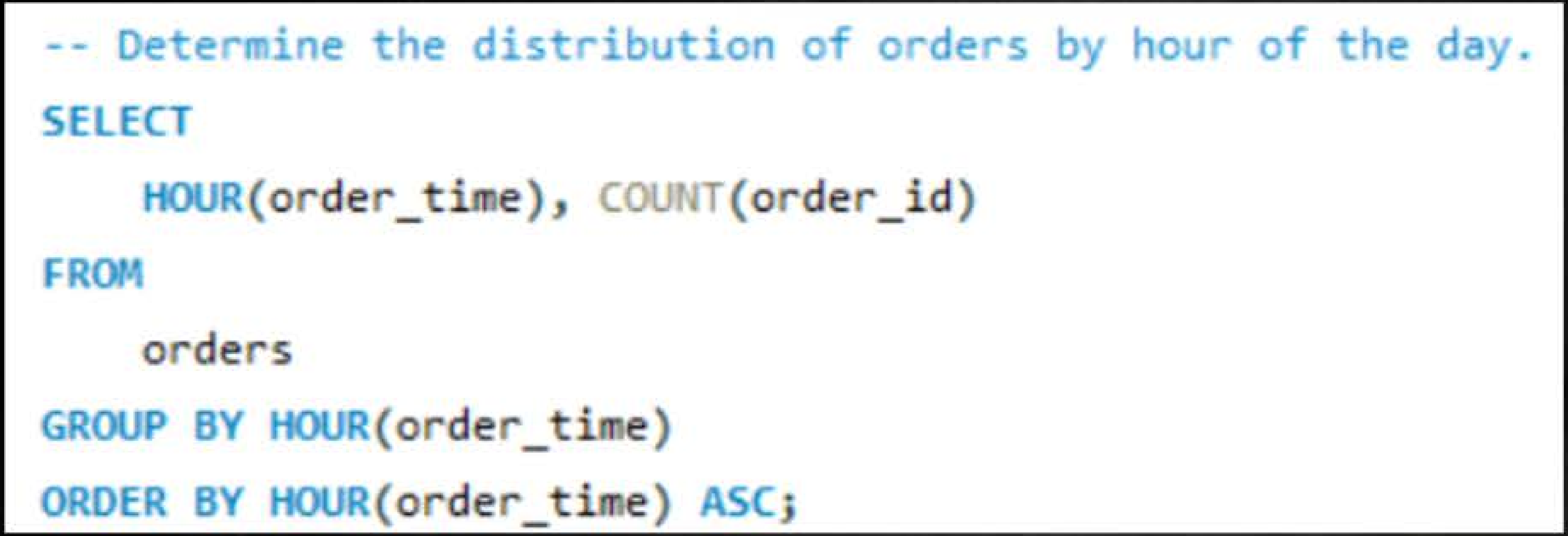
output



category	SUM(quantity)
Classic	14888
Veggie	11649
Supreme	11987
Chicken	11050

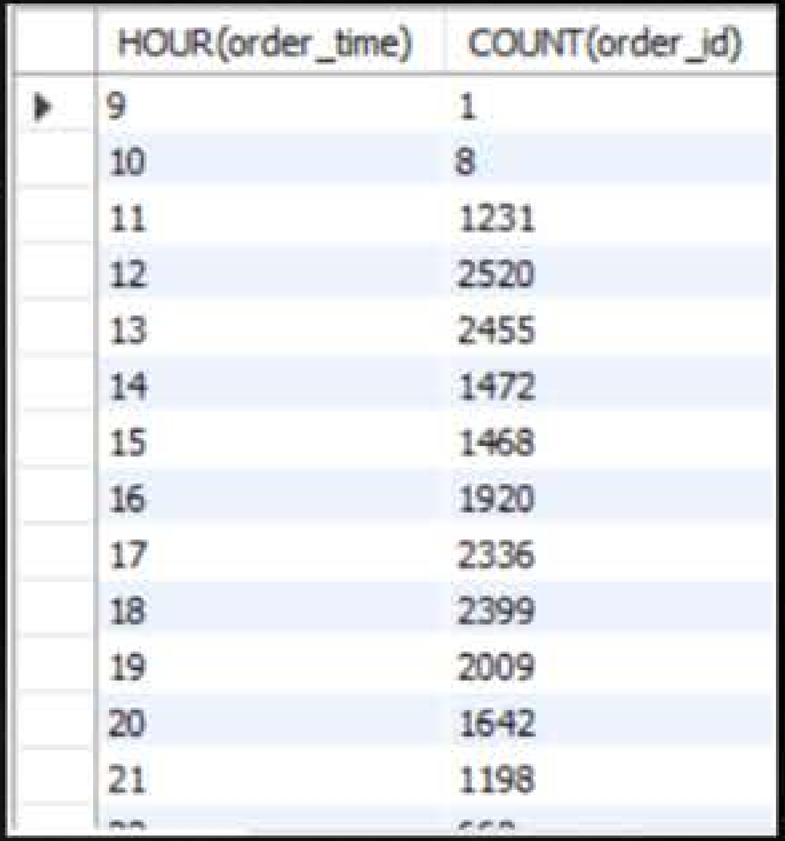
Determine the distribution of orders by hour of the day.

input



```
-- Determine the distribution of orders by hour of the day.
SELECT
    HOUR(order_time), COUNT(order_id)
FROM
    orders
GROUP BY HOUR(order_time)
ORDER BY HOUR(order_time) ASC;
```

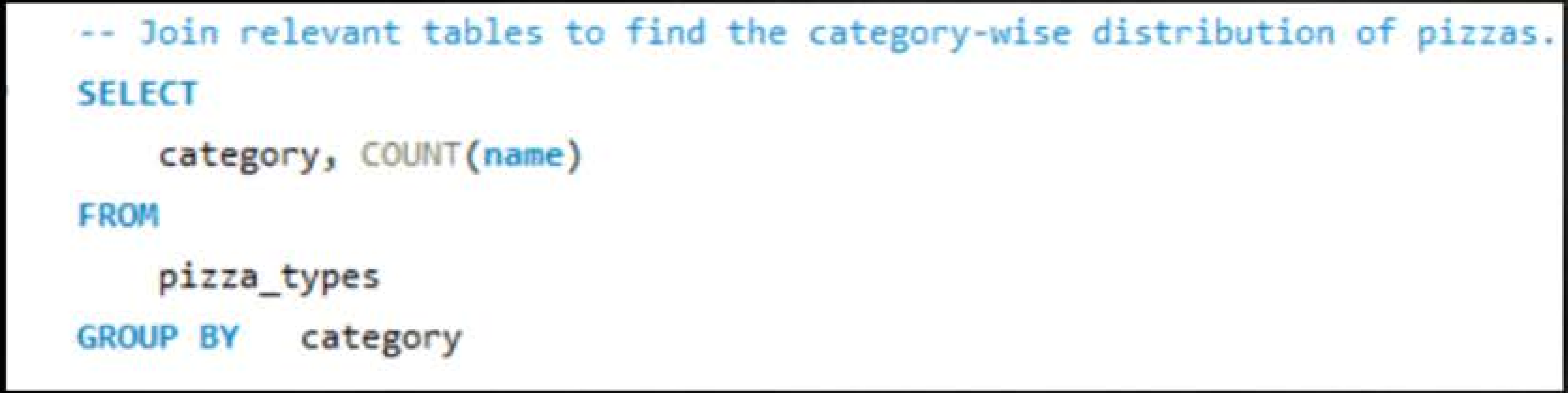
output



HOUR(order_time)	COUNT(order_id)
9	1
10	8
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198

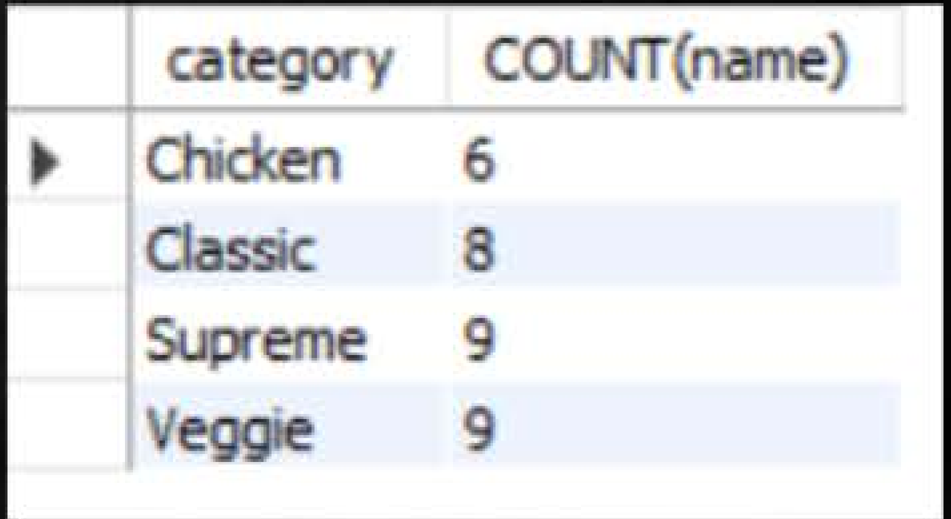
Join relevant tables to find the category-wise distribution of pizzas.

input



```
-- Join relevant tables to find the category-wise distribution of pizzas.
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY   category
```

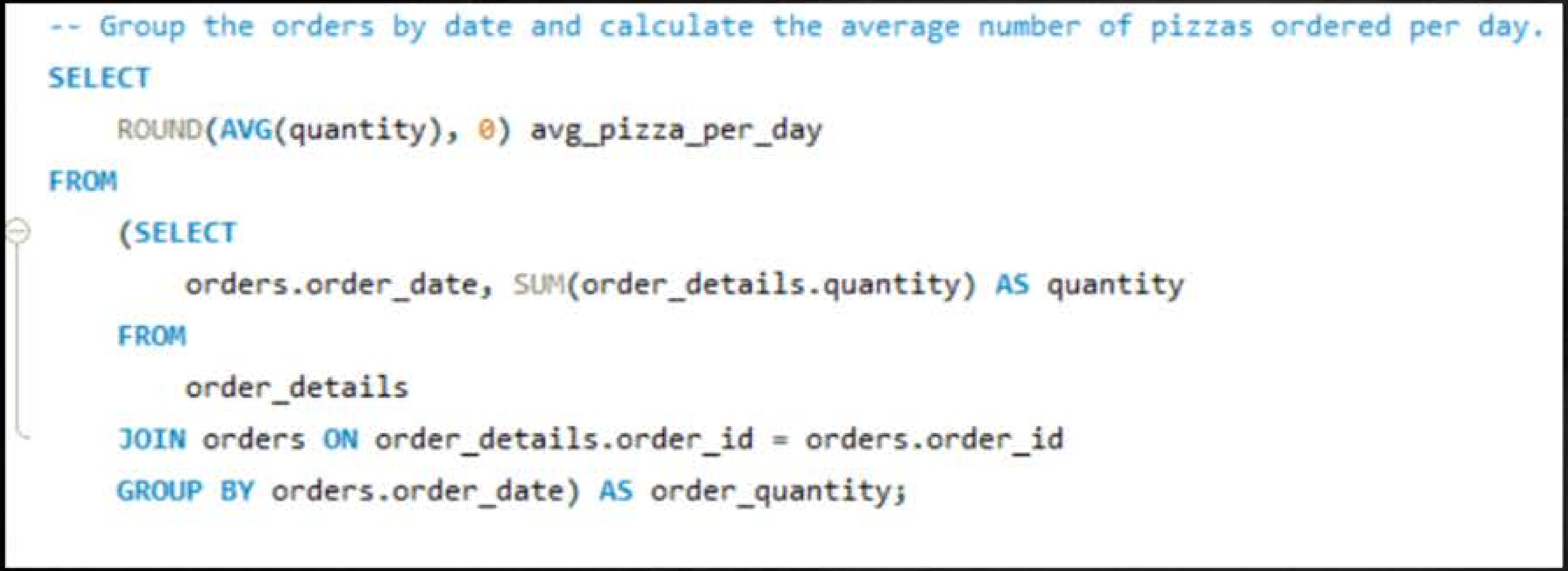
output



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

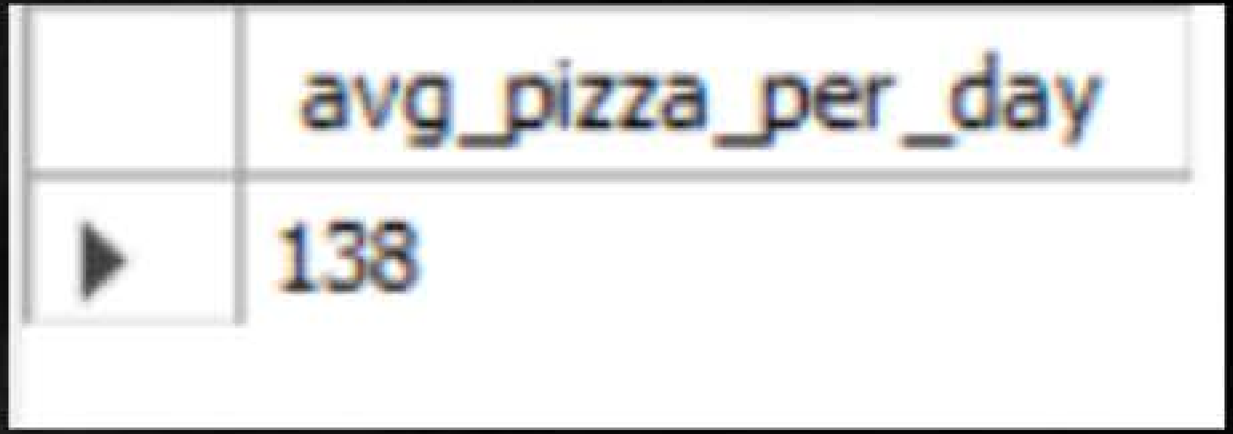
Group the orders by date and calculate the average number of pizzas ordered per day.

input



```
-- Group the orders by date and calculate the average number of pizzas ordered per day.
SELECT
    ROUND(AVG(quantity), 0) avg_pizza_per_day
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        order_details
    JOIN orders ON order_details.order_id = orders.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

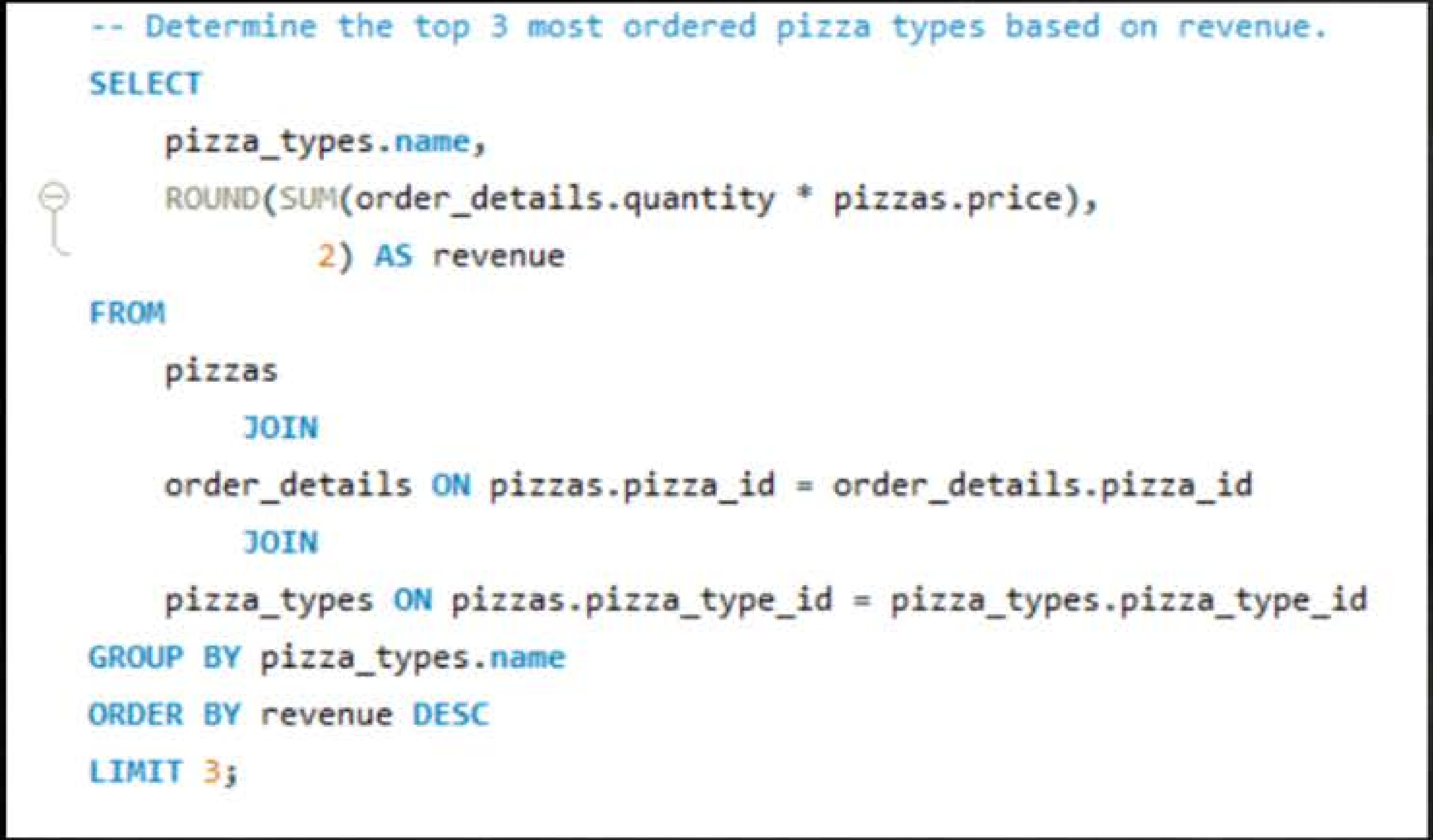
output



	avg_pizza_per_day
▶	138

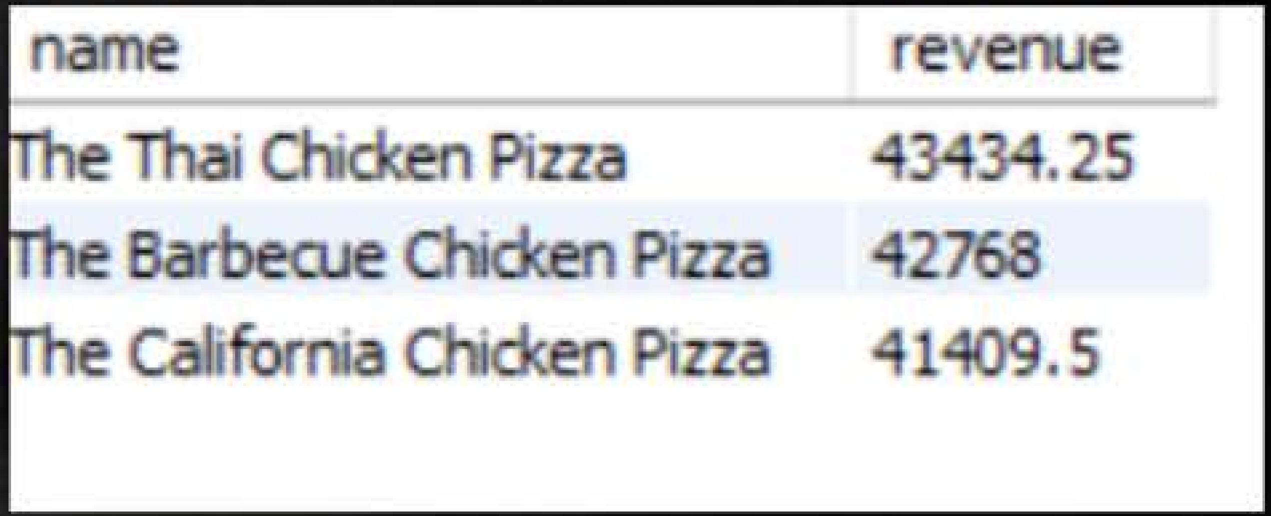
Determine the top 3 most ordered pizza types based on revenue.

input



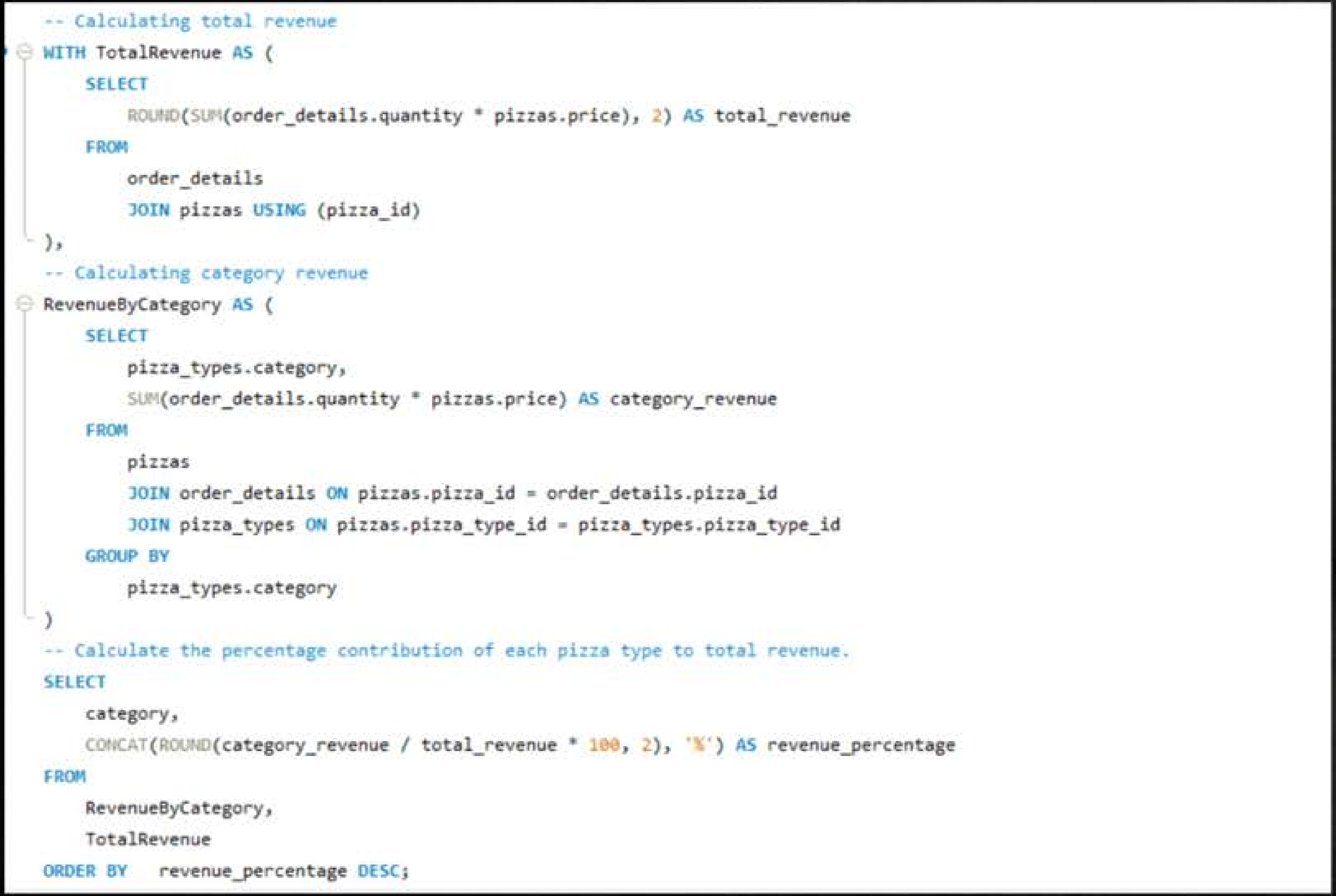
```sql
-- Determine the top 3 most ordered pizza types based on revenue.
SELECT
    pizza_types.name,
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS revenue
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```

output



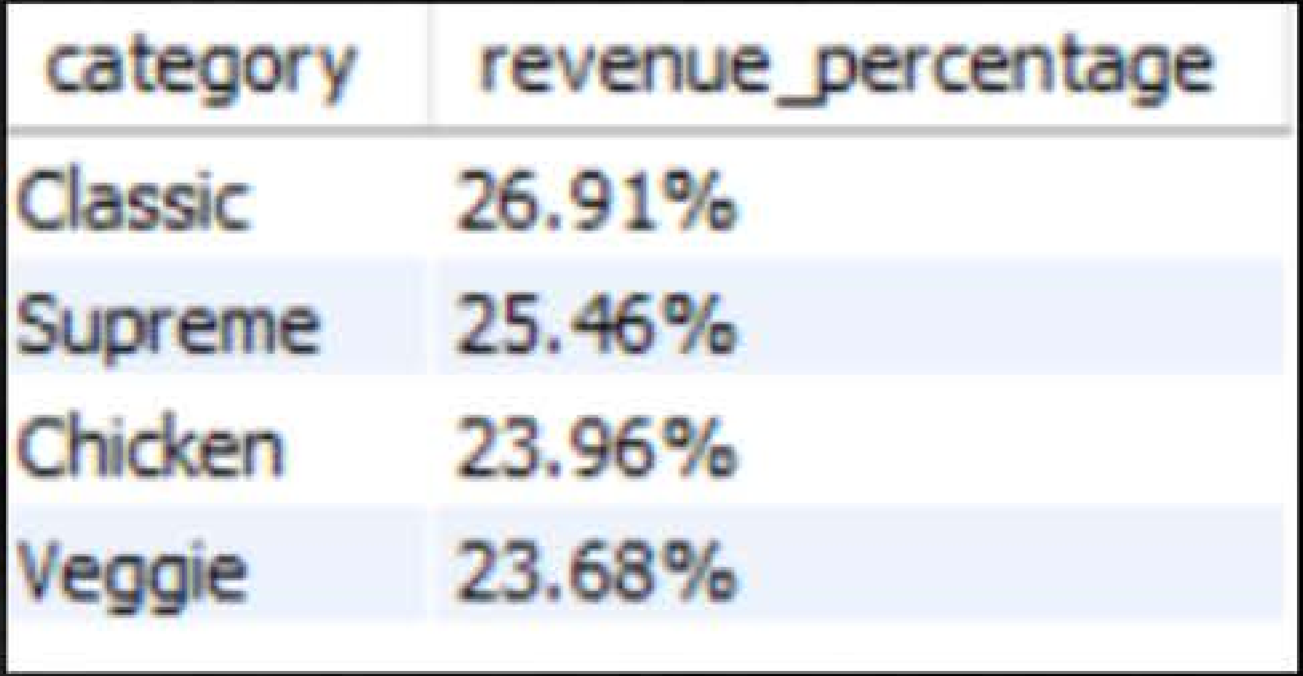
name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

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



```
-- Calculating total revenue
WITH TotalRevenue AS (
    SELECT
        ROUND(SUM(order_details.quantity * pizzas.price), 2) AS total_revenue
    FROM
        order_details
        JOIN pizzas USING (pizza_id)
),
-- Calculating category revenue
RevenueByCategory AS (
    SELECT
        pizza_types.category,
        SUM(order_details.quantity * pizzas.price) AS category_revenue
    FROM
        pizzas
        JOIN order_details ON pizzas.pizza_id = order_details.pizza_id
        JOIN pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
    GROUP BY
        pizza_types.category
)
-- Calculate the percentage contribution of each pizza type to total revenue.
SELECT
    category,
    CONCAT(ROUND(category_revenue / total_revenue * 100, 2), '%') AS revenue_percentage
FROM
    RevenueByCategory,
    TotalRevenue
ORDER BY   revenue_percentage DESC;
```

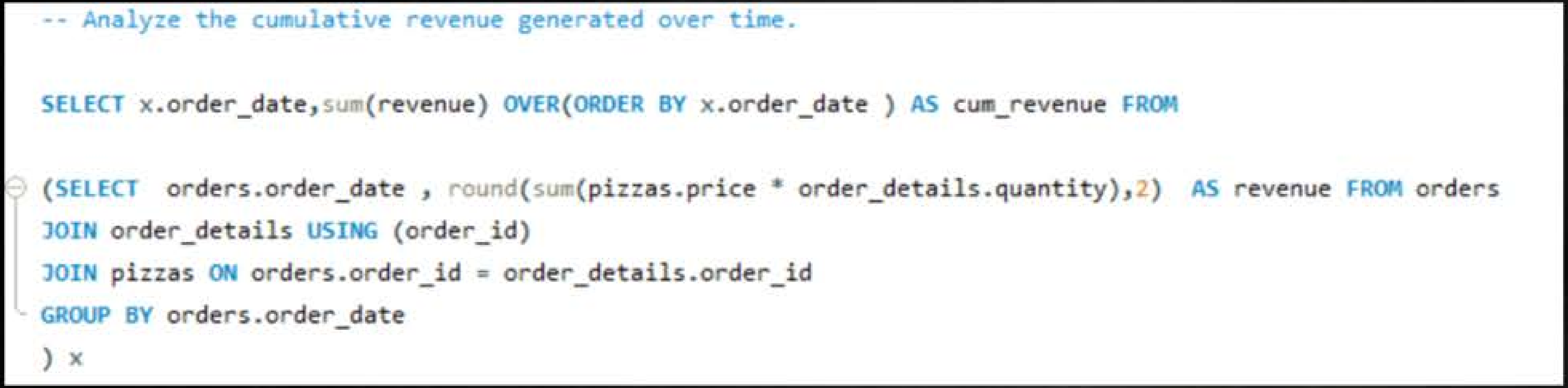
output



category	revenue_percentage
Classic	26.91%
Supreme	25.46%
Chicken	23.96%
Veggie	23.68%

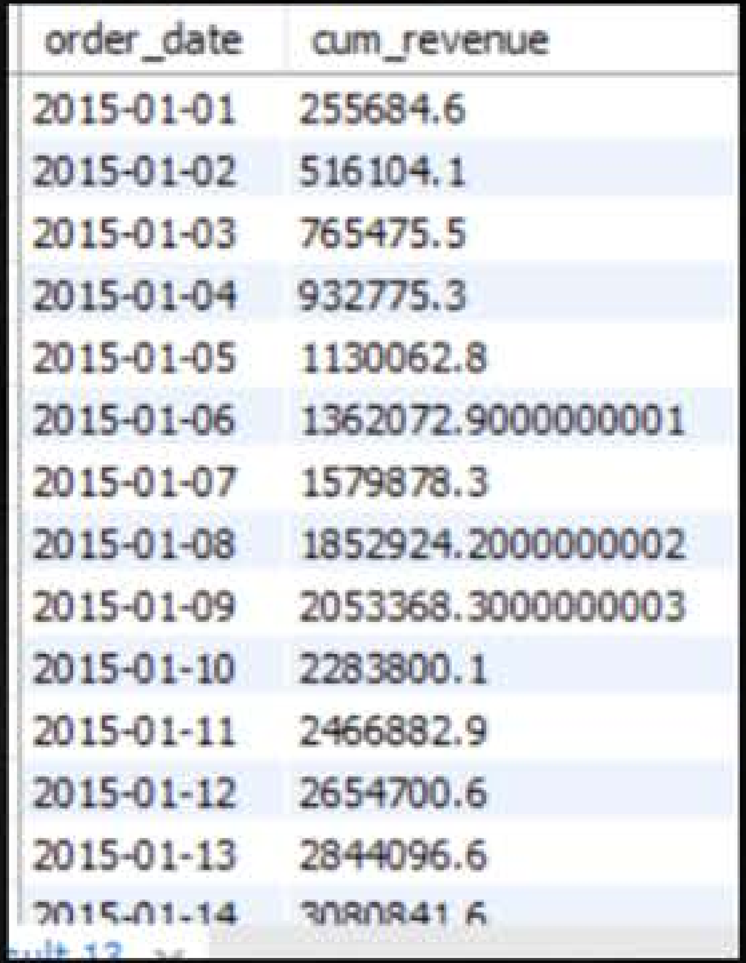
Analyze the cumulative revenue generated over time.

input



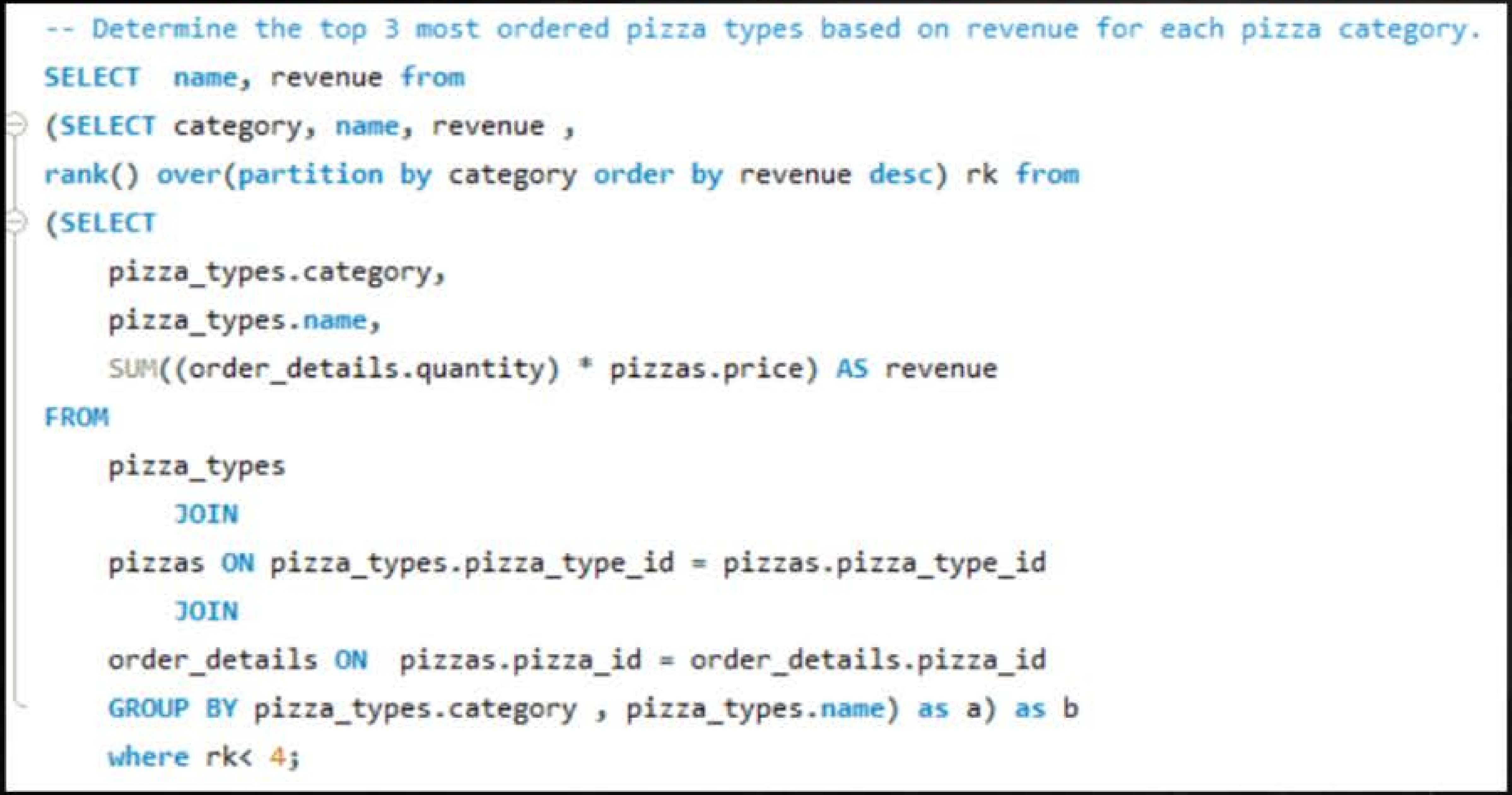
```
-- Analyze the cumulative revenue generated over time.

SELECT x.order_date,sum(revenue) OVER(ORDER BY x.order_date ) AS cum_revenue FROM

(SELECT  orders.order_date , round(sum(pizzas.price * order_details.quantity),2)  AS revenue FROM orders
JOIN order_details USING (order_id)
JOIN pizzas ON orders.order_id = order_details.order_id
GROUP BY orders.order_date
) x
```

output



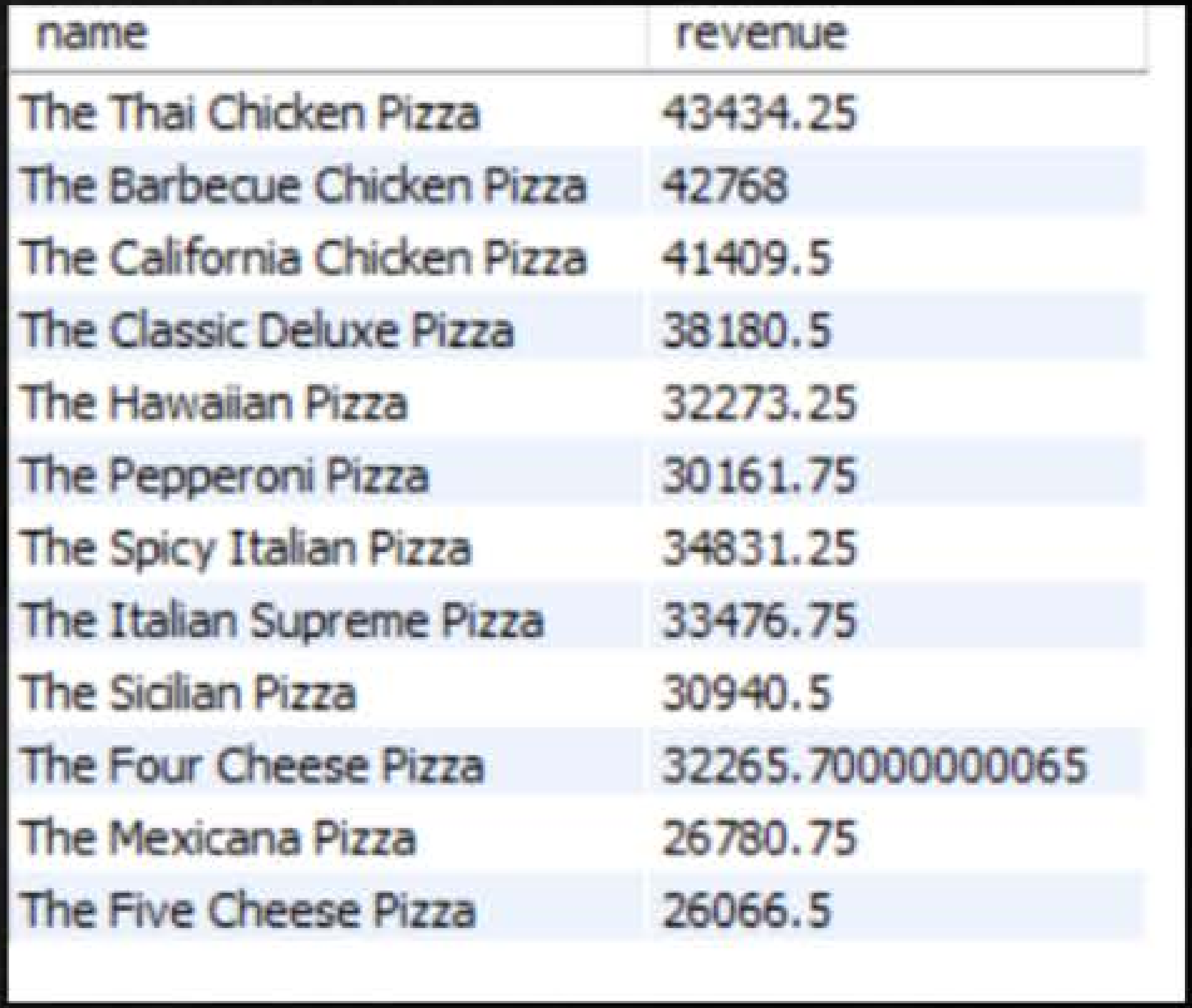
order_date	cum_revenue
2015-01-01	255684.6
2015-01-02	516104.1
2015-01-03	765475.5
2015-01-04	932775.3
2015-01-05	1130062.8
2015-01-06	1362072.9000000001
2015-01-07	1579878.3
2015-01-08	1852924.2000000002
2015-01-09	2053368.3000000003
2015-01-10	2283800.1
2015-01-11	2466882.9
2015-01-12	2654700.6
2015-01-13	2844096.6
2015-01-14	3080841.6

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



```
-- 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) rk 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  pizzas.pizza_id = order_details.pizza_id
    GROUP BY pizza_types.category , pizza_types.name) as a) as b
    where rk< 4;
```

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