Where Every Slice is a Taste of Perfection

PIZZA SALES ANALYSIS USING SQL QUERIES

WELCOME TO PIZZA RESTO

HELLOOO!!!!

I'm Aishwarya Singh, and I have worked on SQL queries for analyzing pizza sales data. In this presentation, I will showcase how these queries help in understanding sales trends, customer preferences, and overall business performance. Through this, we can see how SQL is used to extract meaningful insights from data efficiently.

PIZZA TYPES

THE THAI CHICKEN PIZZA

THE BARBECUE CHICKEN PIZZA

THE ITALIAN SUPREME PIZZA

THE CALIFORNIA CHICKEN PIZZA

THE HAWAIIAN PIZZA

THE PEPPERONI PIZZA

THE SPICY ITALIAN PIZZA

THE SICILIAN PIZZA

THE MEXICANA PIZZA

THE FIVE CHEESE PIZZA

THE CLASSIC DELUXE PIZZA

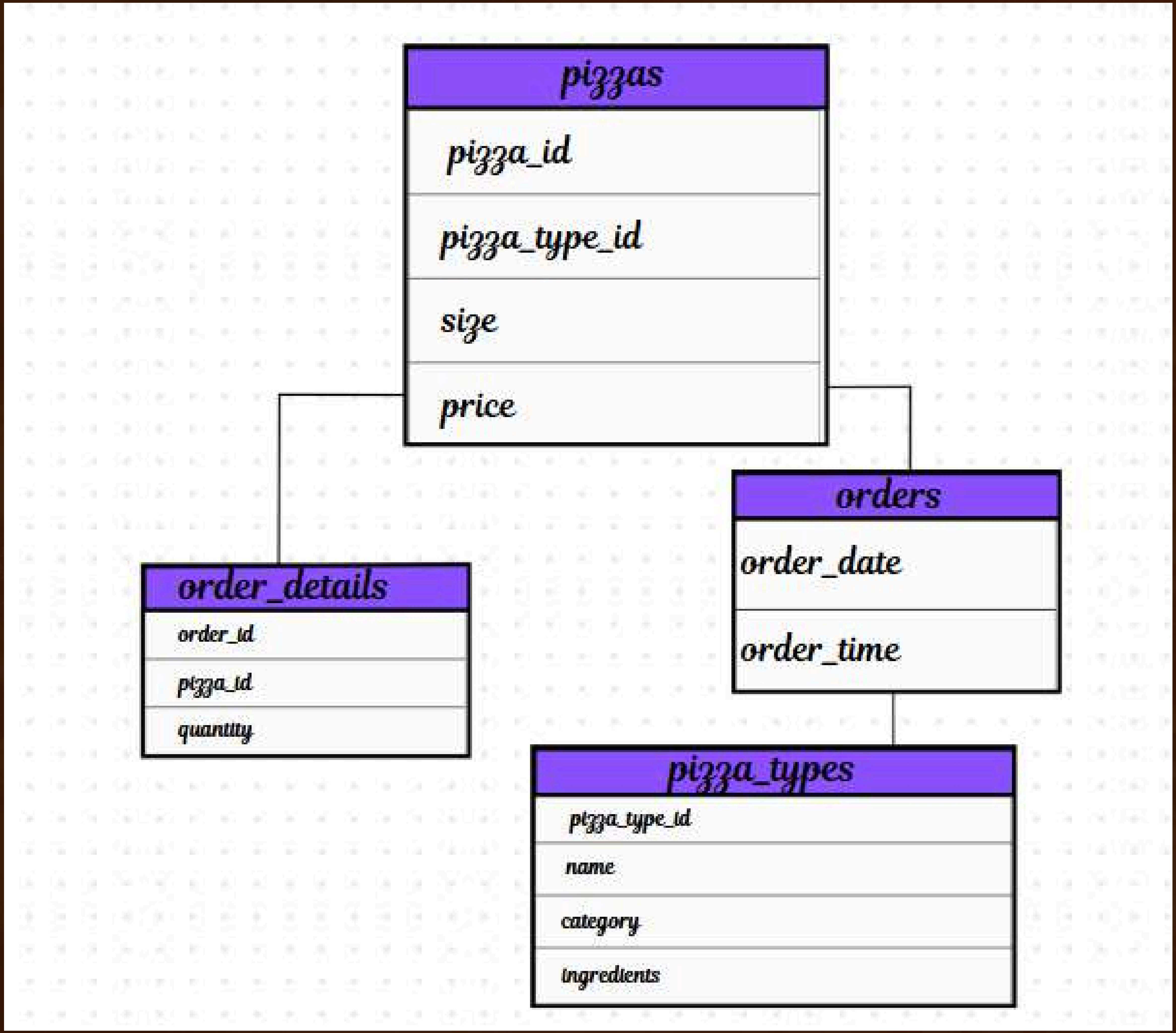
THE FOUR CHEESE PIZZA

In this presentation, I have solved a total of 14 SQL queries based on pizza sales data. It includes 6 basic queries, focusing on simple data retrieval and filtering, 5 intermediate queries, involving aggregations and joins, and 3 advanced queries, which handle complex analysis and optimization. This showcases how SQL can be used at different levels to extract meaningful insights from data efficiently.

The dataset contains the following files:

1. order_details – Contains details of each pizza in an order.
2. orders – Stores order-related information (like order date and time).
3. pizza_types– Describes different types of pizzas.
4. pizzas – Contains pizza details (size, price, and type).

SCHEMA



pizzas

- pizza_id
- pizza_type_id
- size
- price

order_details

- order_id
- pizza_id
- quantity

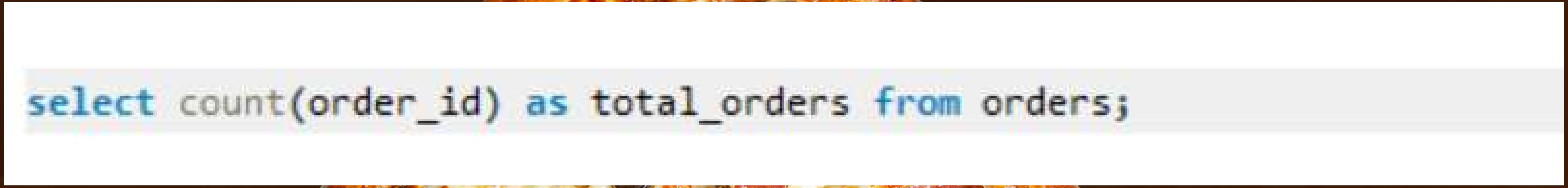
orders

- order_date
- order_time

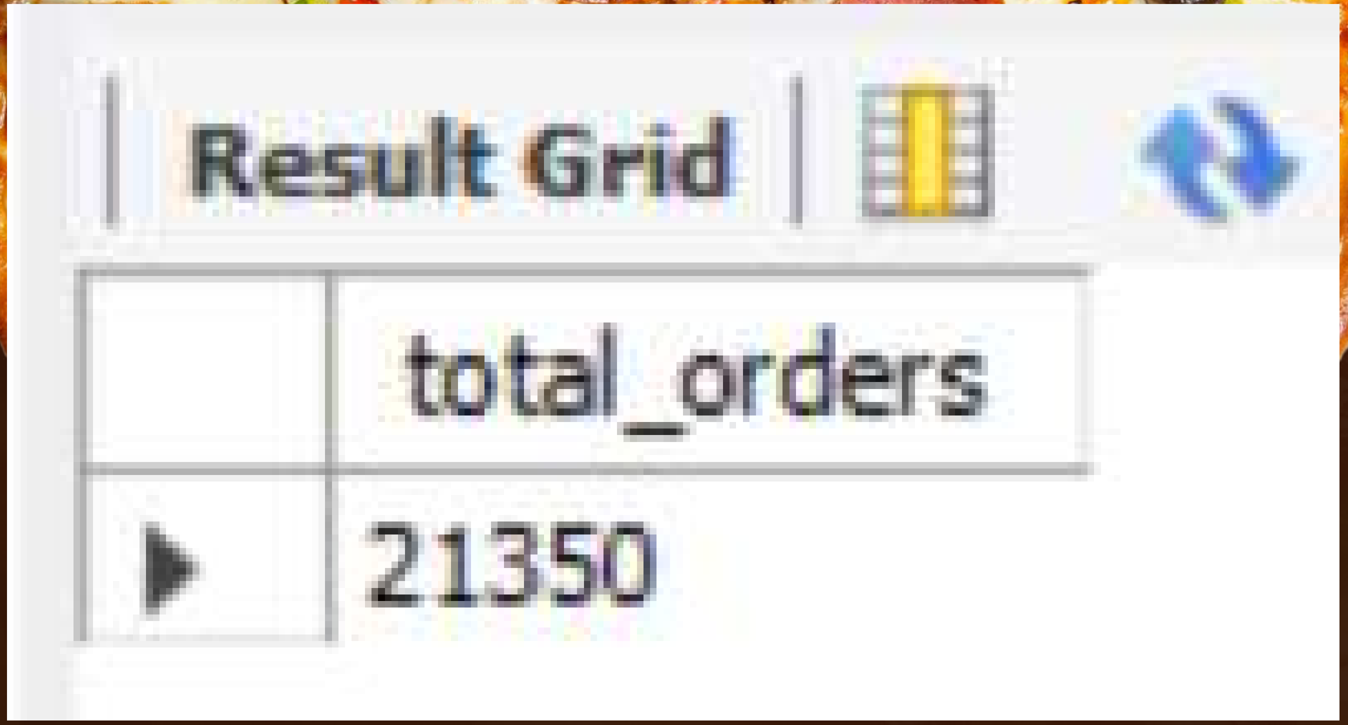
pizza_types

- pizza_type_id
- name
- category
- ingredients

Retrieve the total number of orders placed.



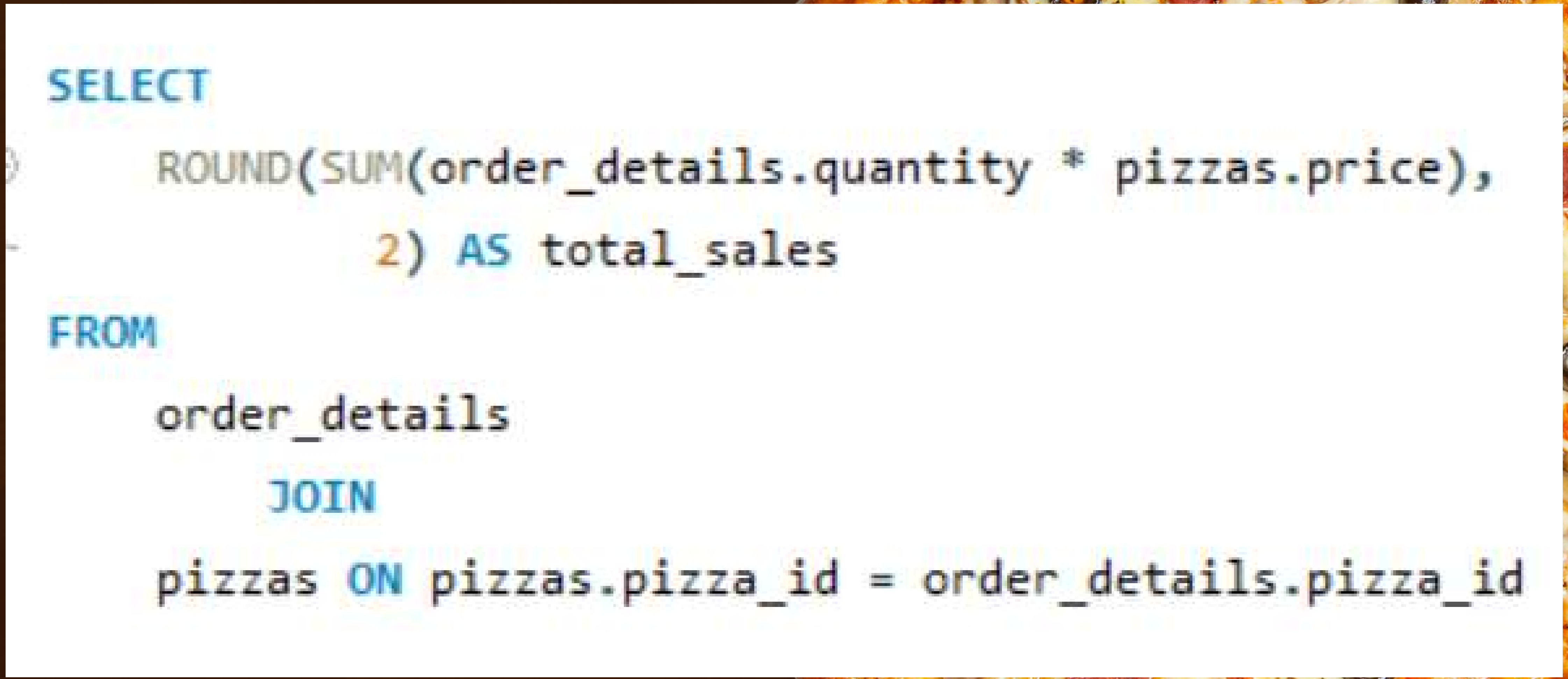
```sql
select count(order_id) as total_orders from orders;
```



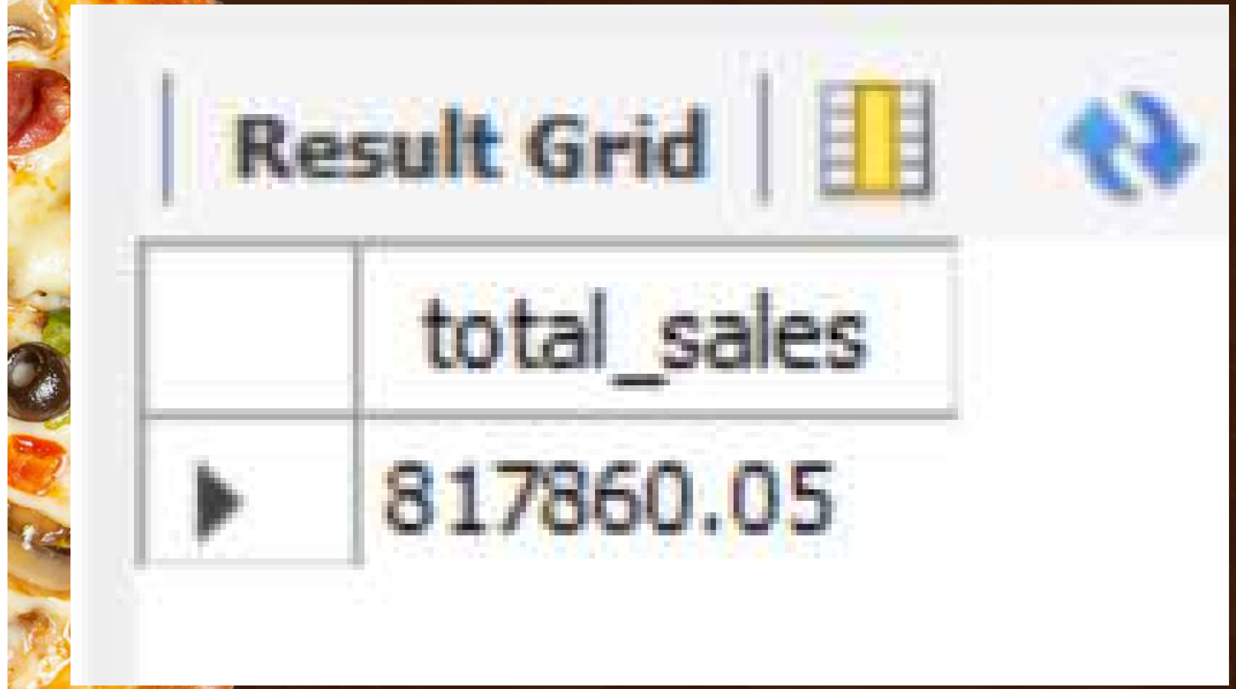
```
total_orders
21350
```

BASIC QUERY 1

Calculate the total revenue generated from pizza sales



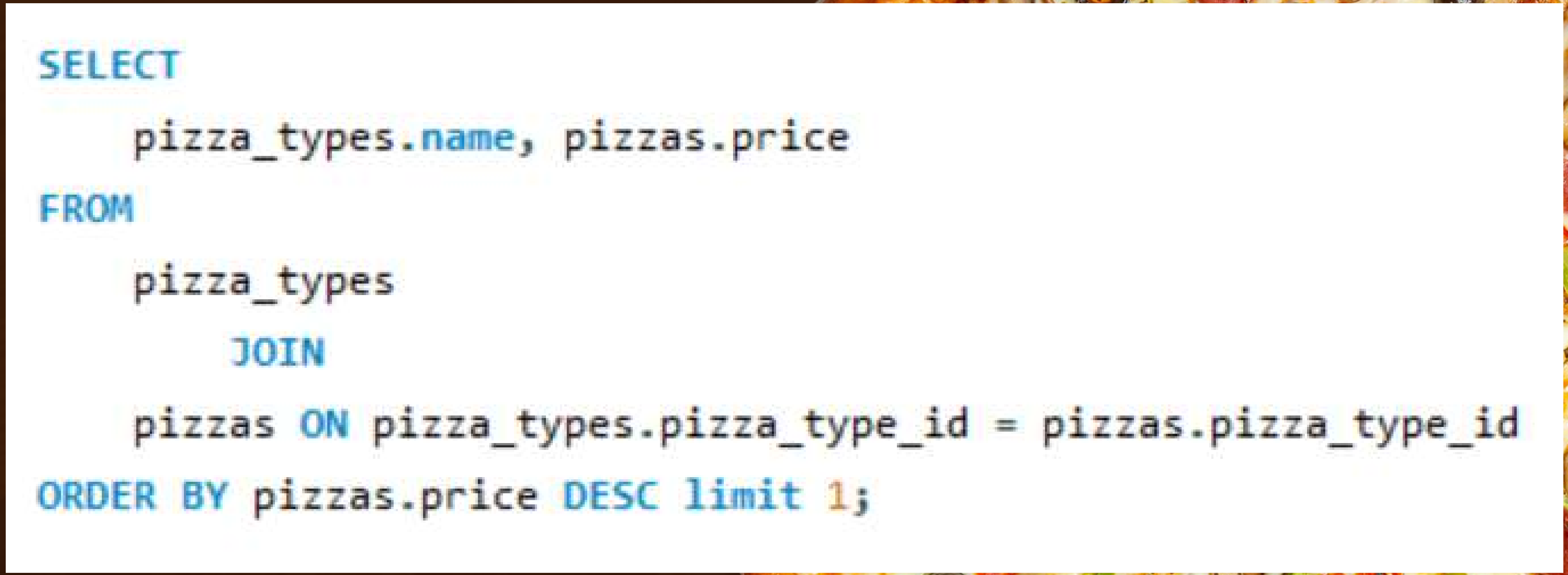
```
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS total_sales
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
```



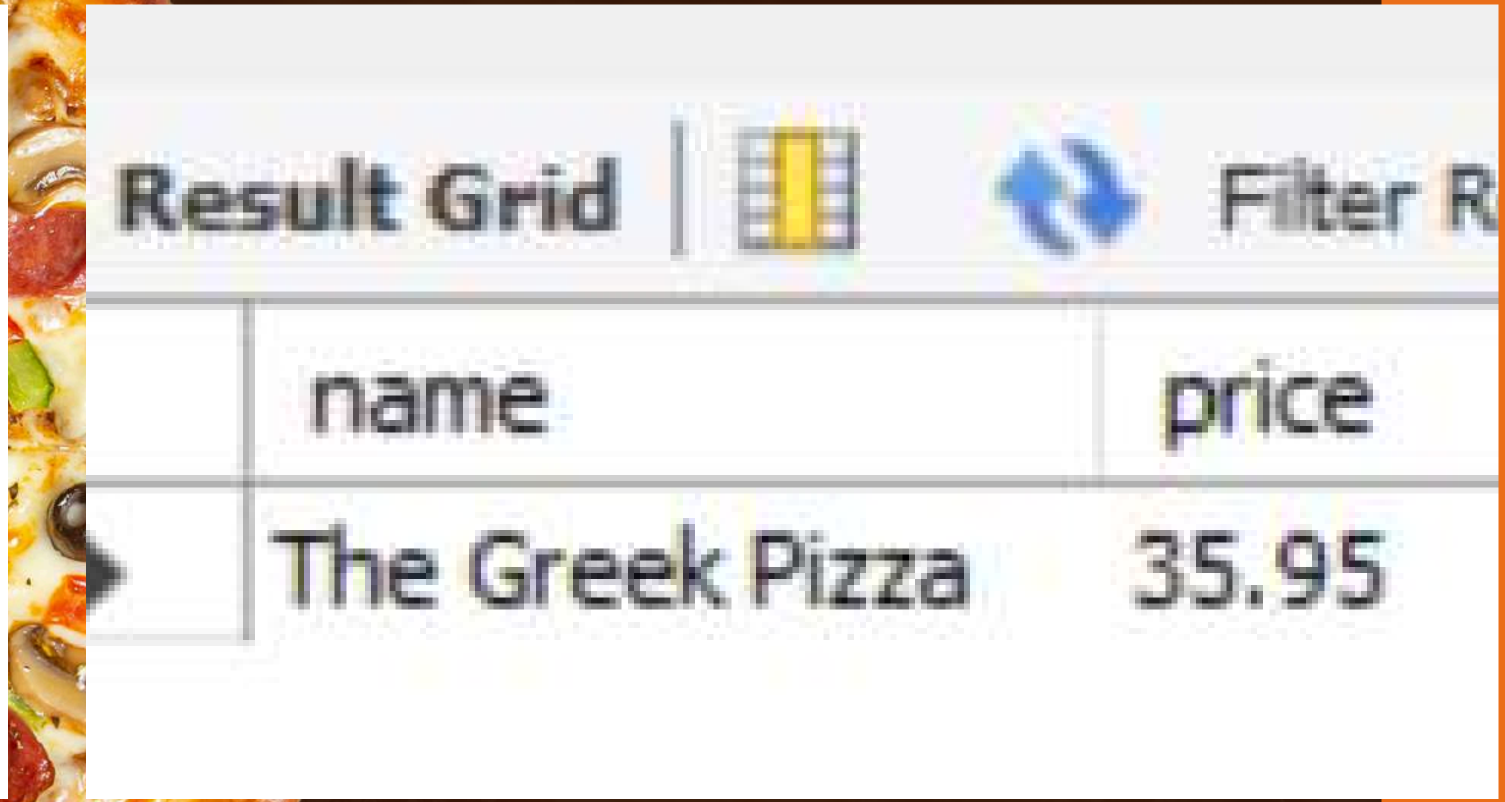
```
total_sales
▶ 817860.05
```

BASIC QUERY 2

Identify the highest-priced pizza.



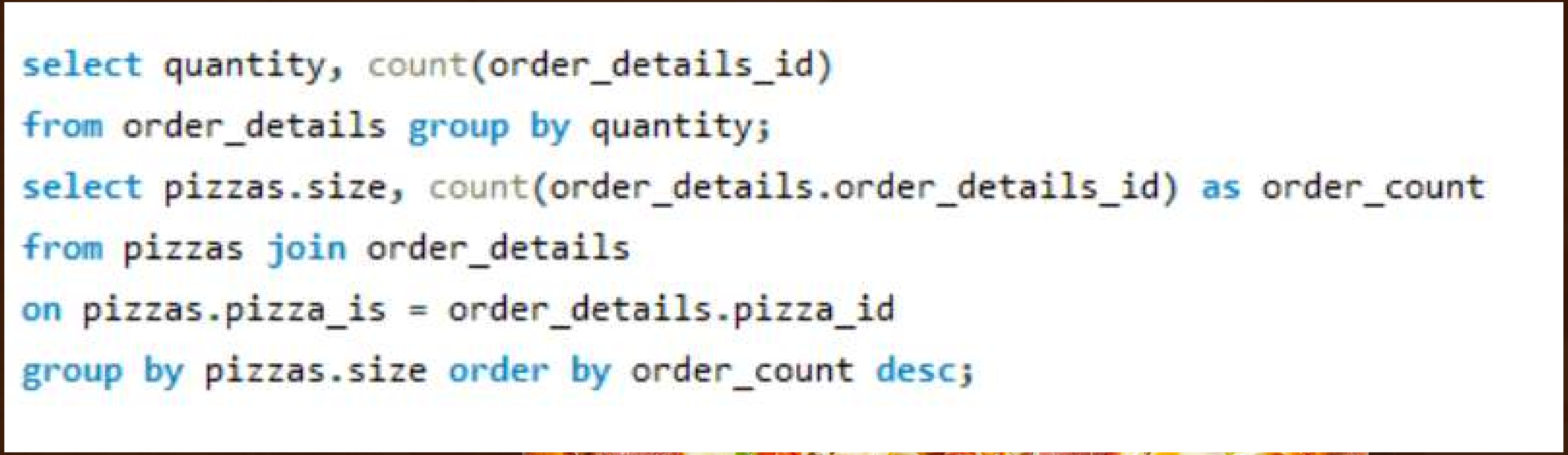
```
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC limit 1;
```



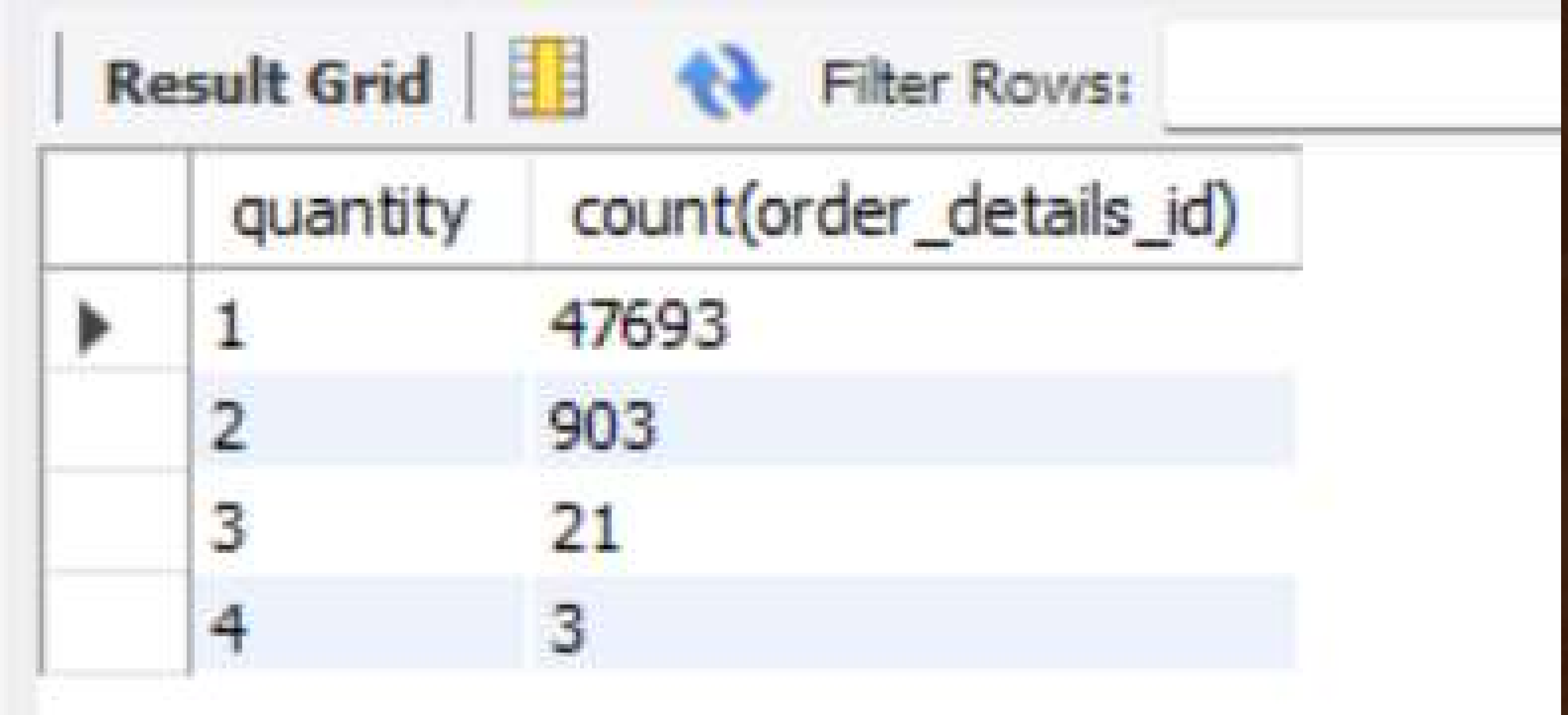
name	price
The Greek Pizza	35.95

BASIC QUERY 3

Identify the most common pizza size ordered.



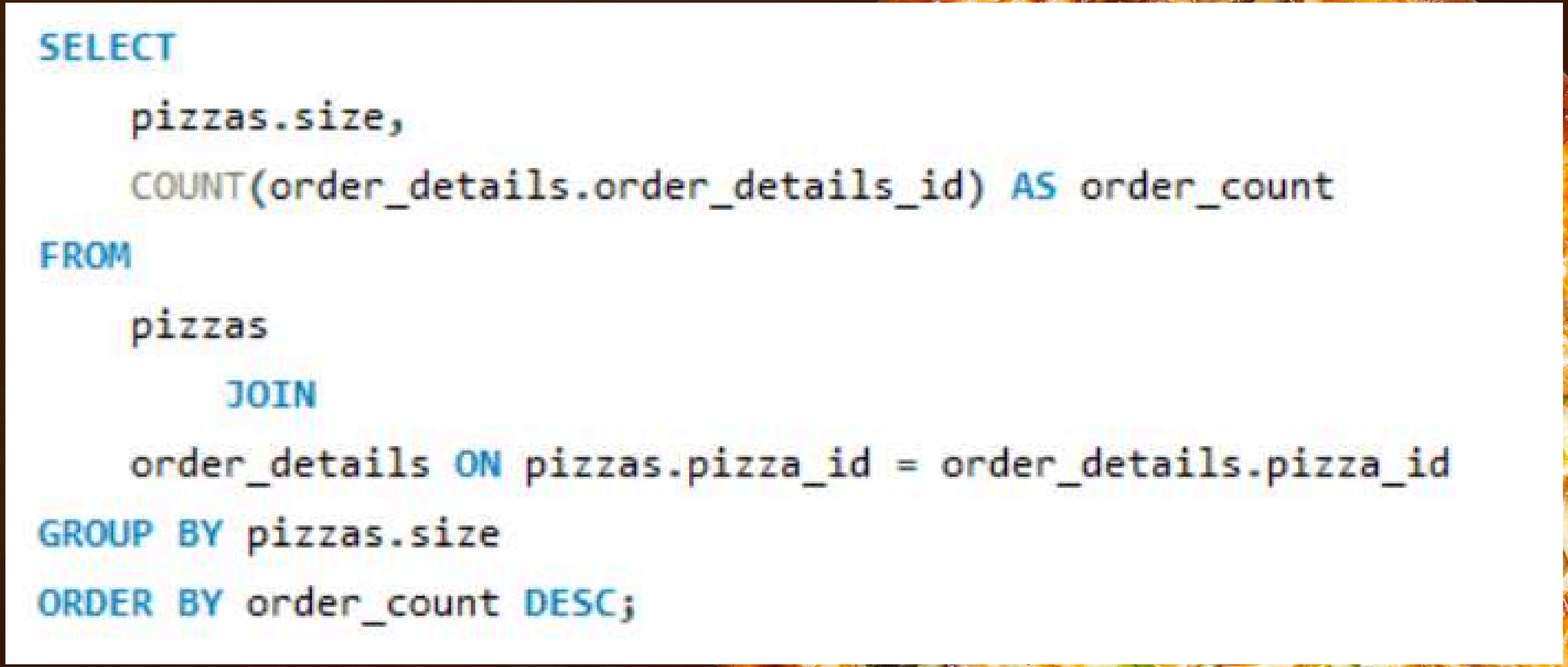
```
select quantity, count(order_details_id)
from order_details group by quantity;
select pizzas.size, count(order_details.order_details_id) as order_count
from pizzas join order_details
on pizzas.pizza_is = order_details.pizza_id
group by pizzas.size order by order_count desc;
```



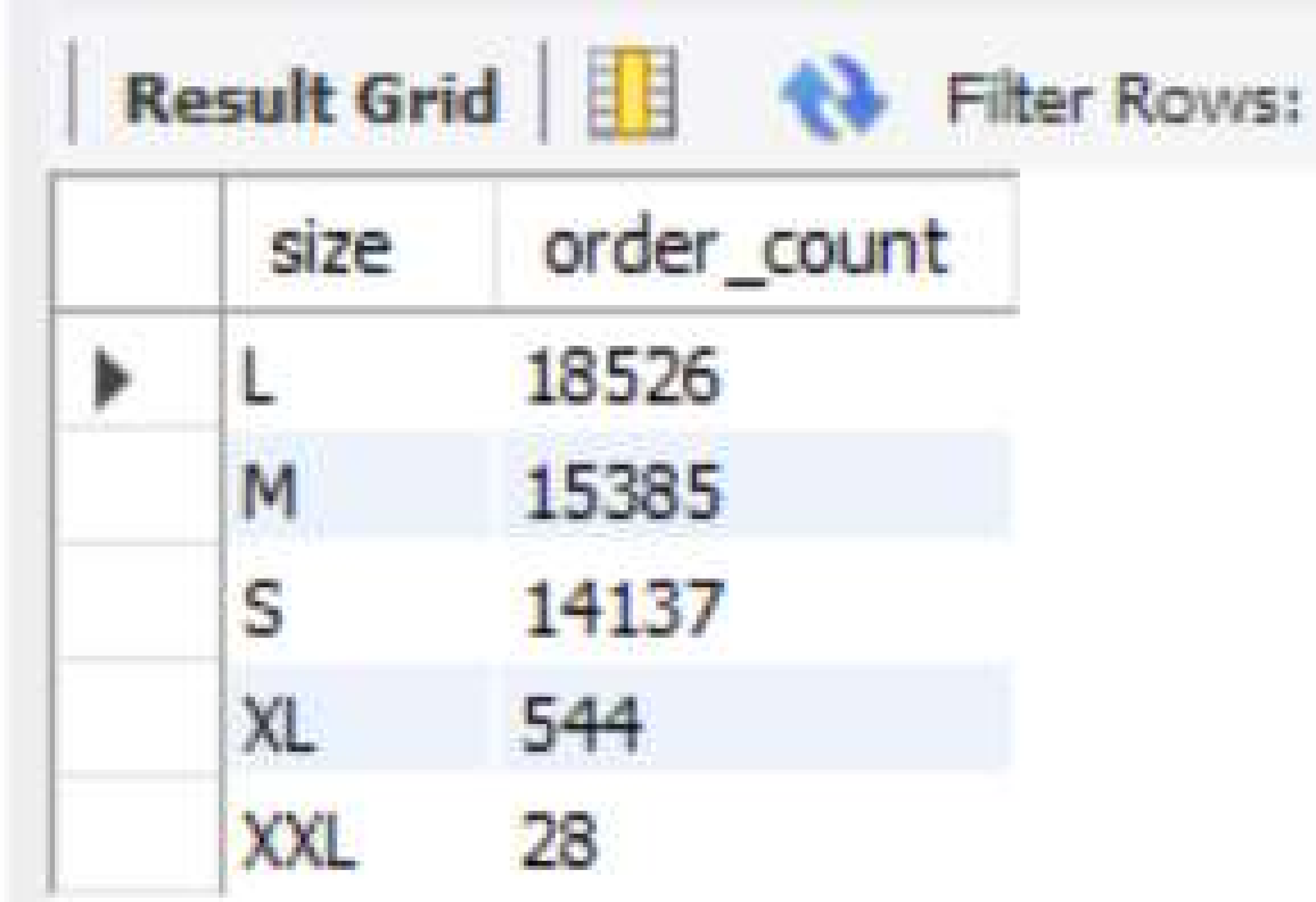
quantity	count(order_details_id)
1	47693
2	903
3	21
4	3

BASIC QUERY 4

Identify the most common pizza size ordered.



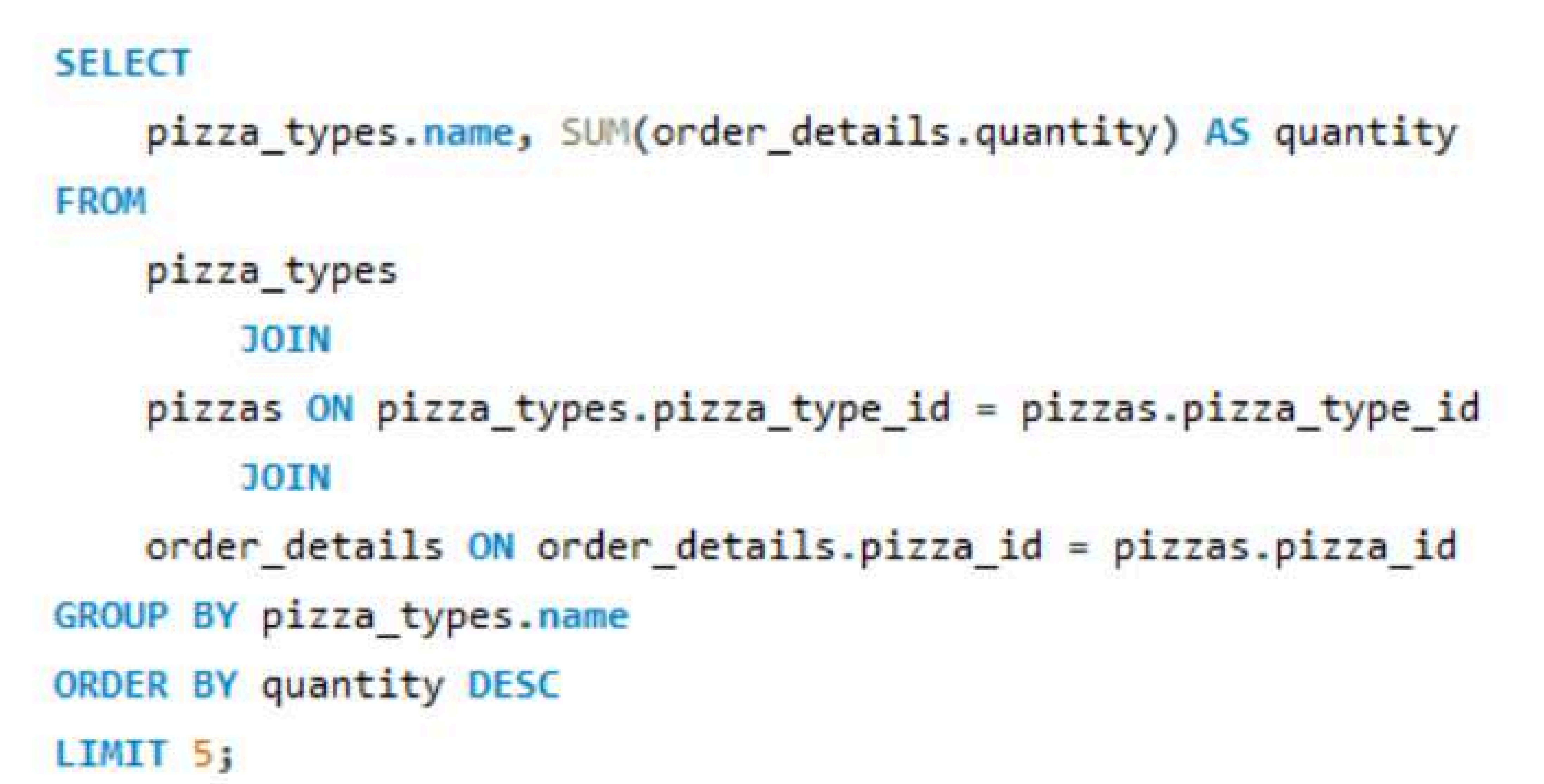
```
SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;
```



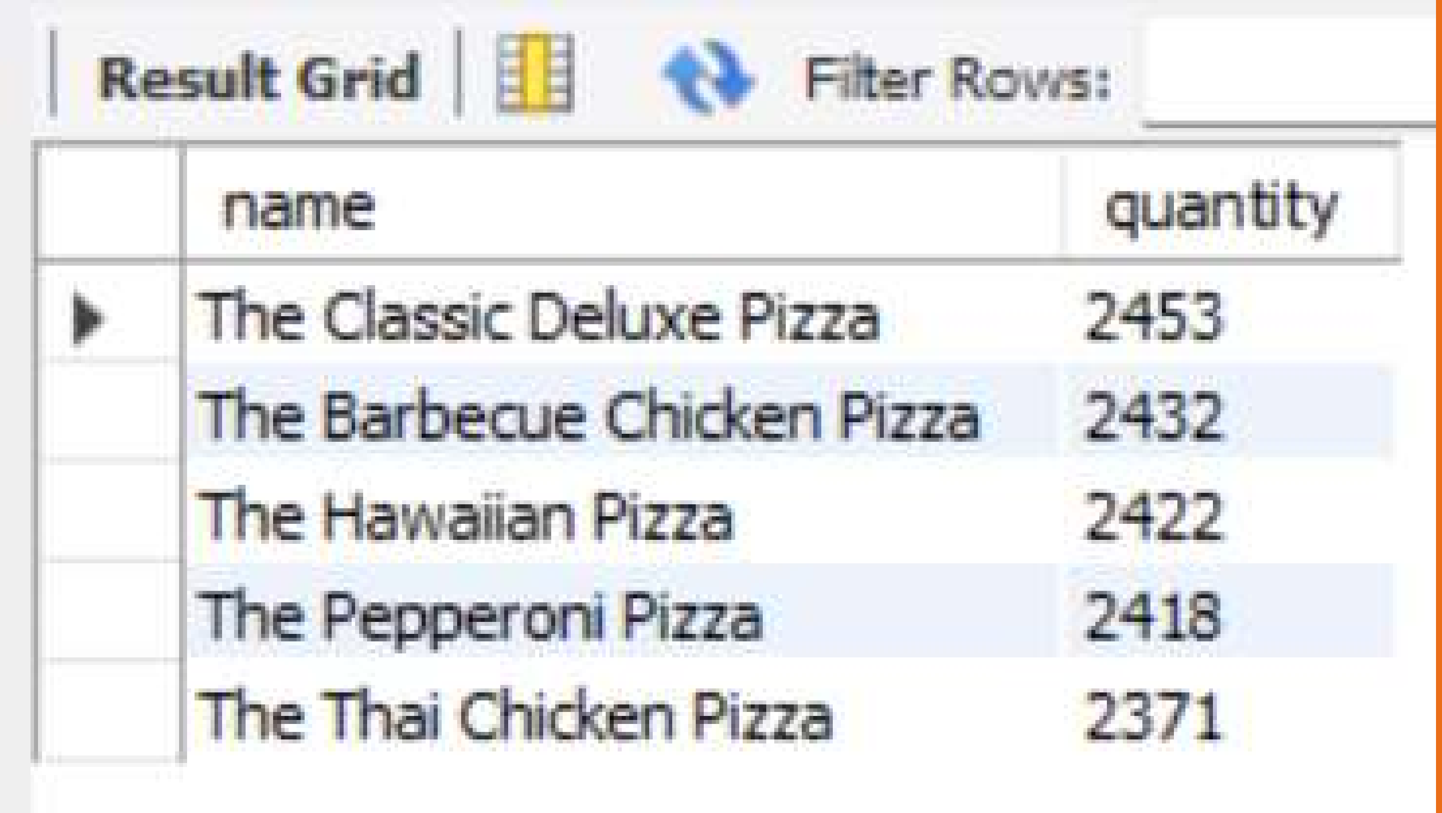
size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

BASIC QUERY 5

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



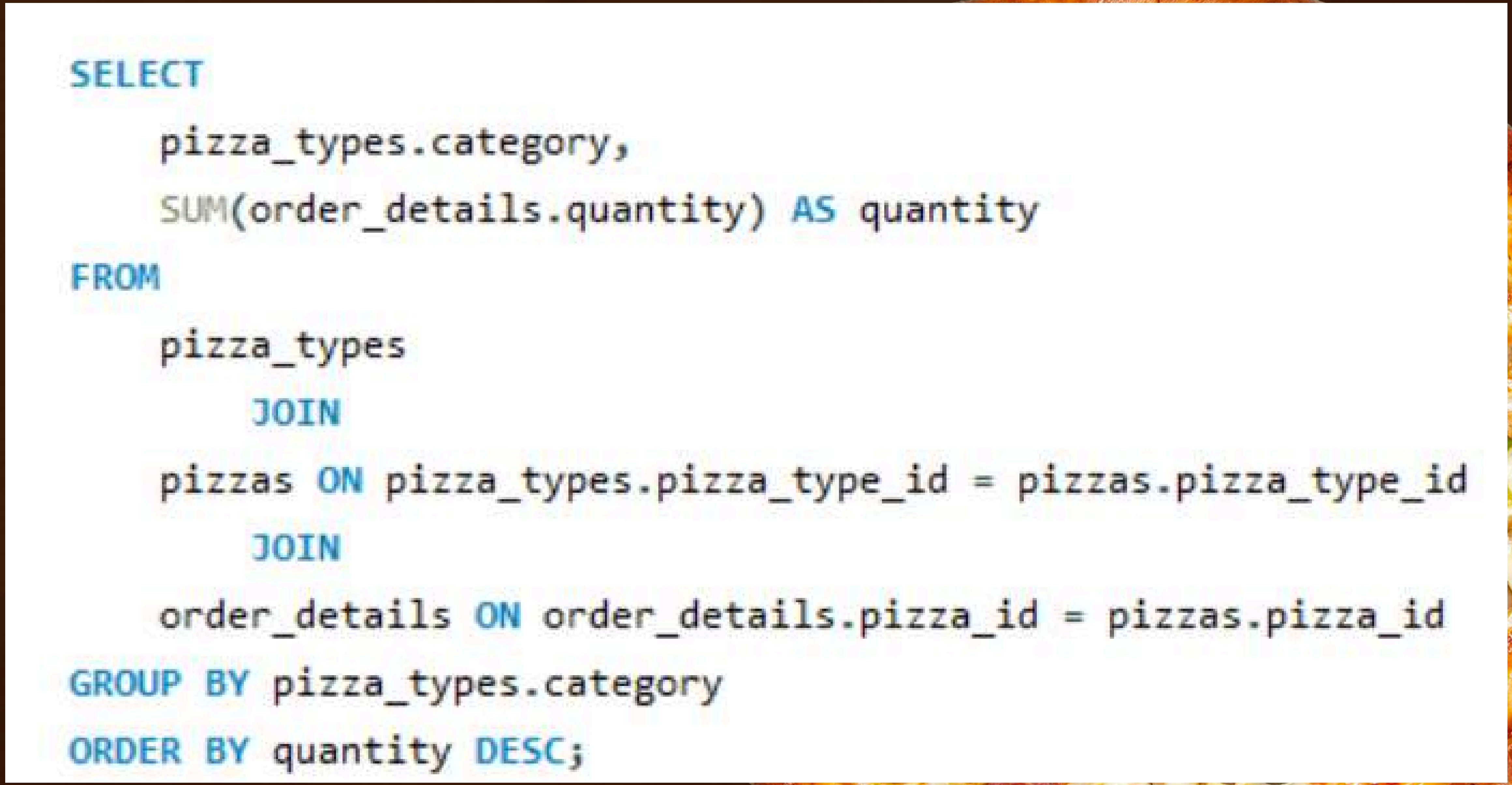
```
SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;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



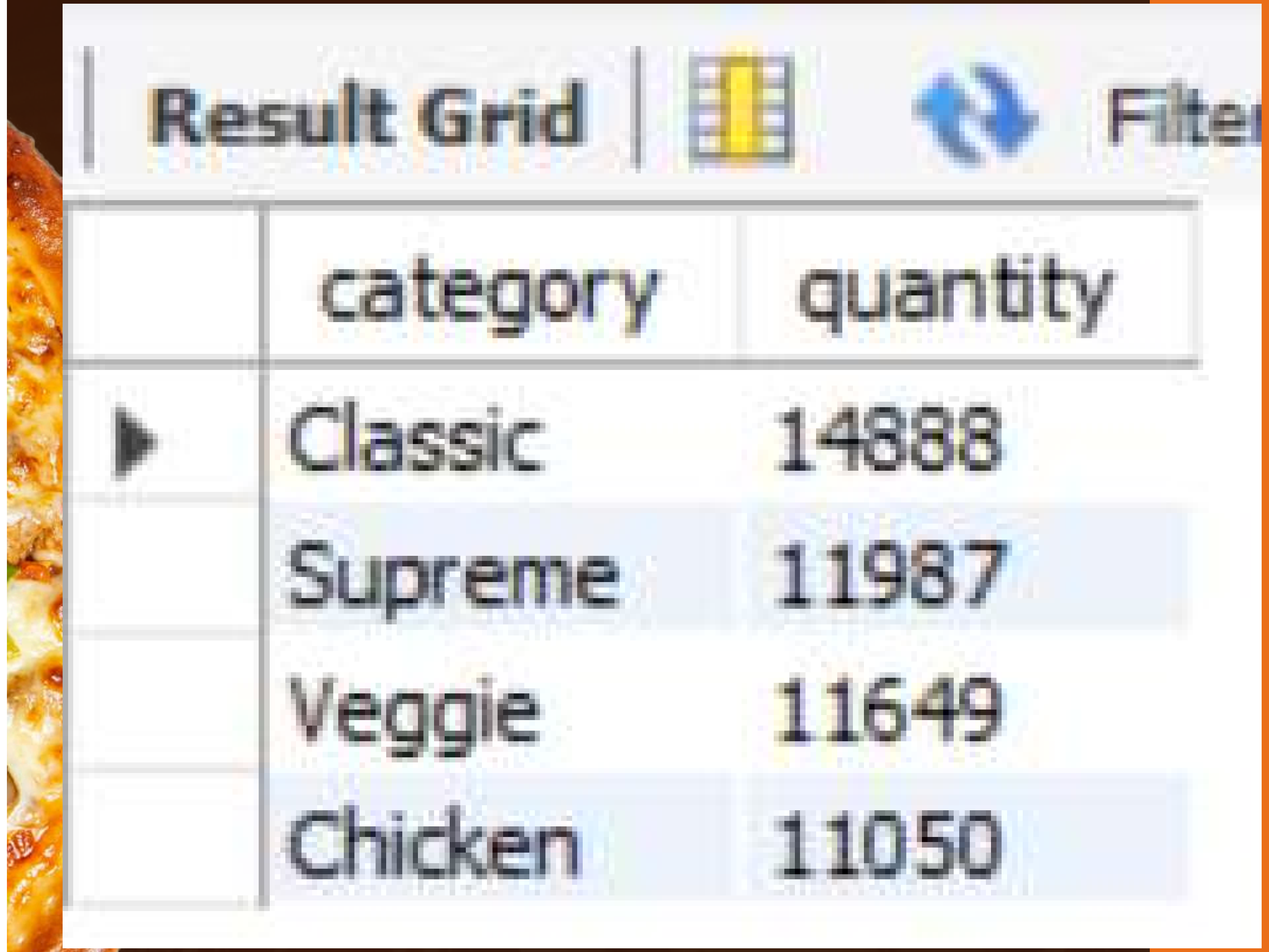
name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

BASIC QUERY 6

Quantity of each pizza category ordered



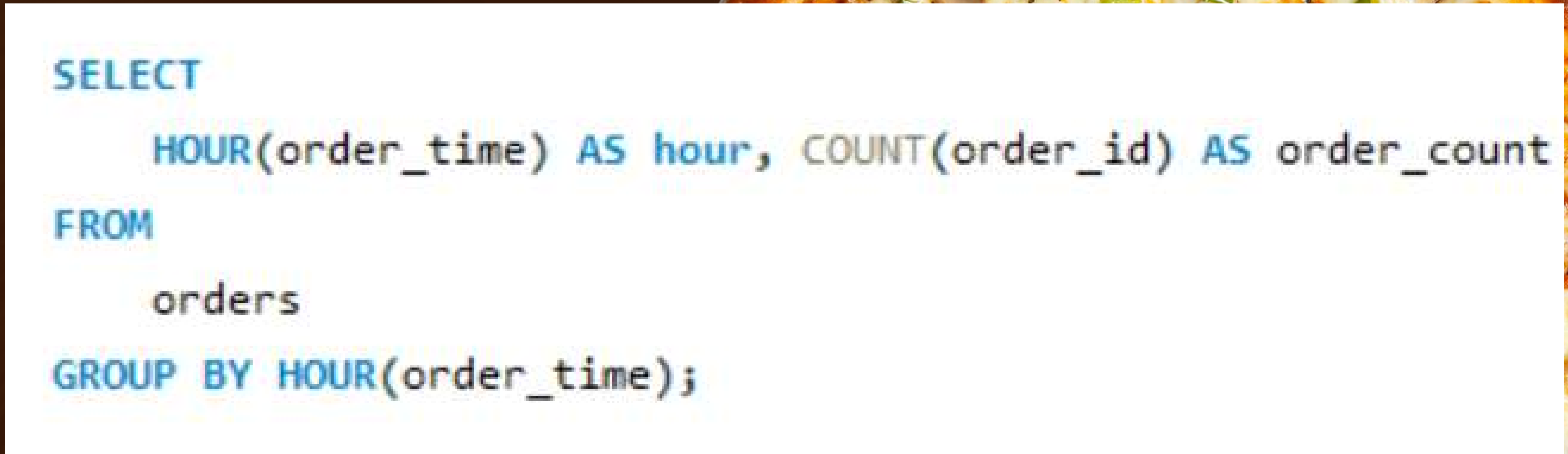
```
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity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



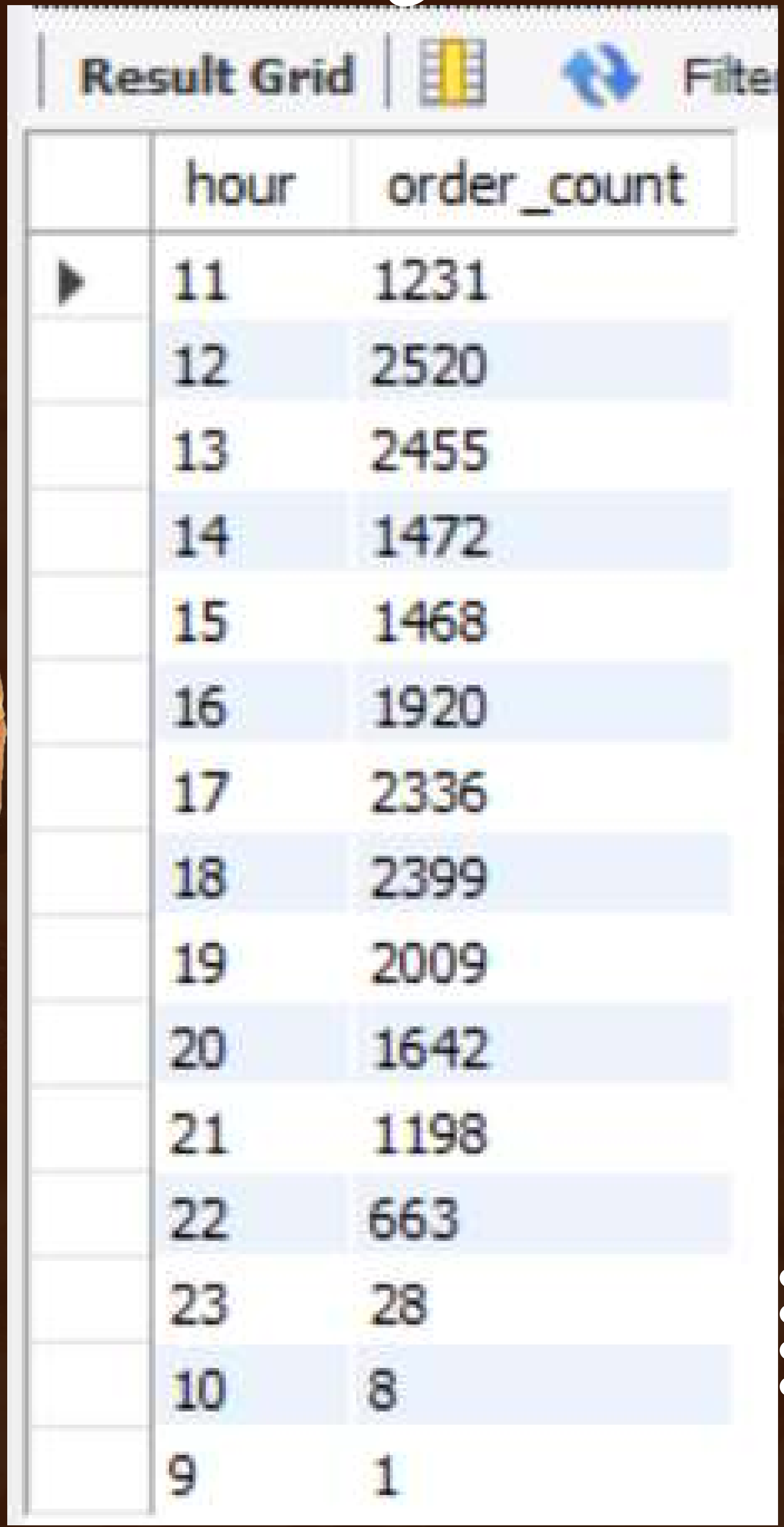
category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

INTERMEDIATE QUERY 1

Determine the distribution of orders by hour of the day.



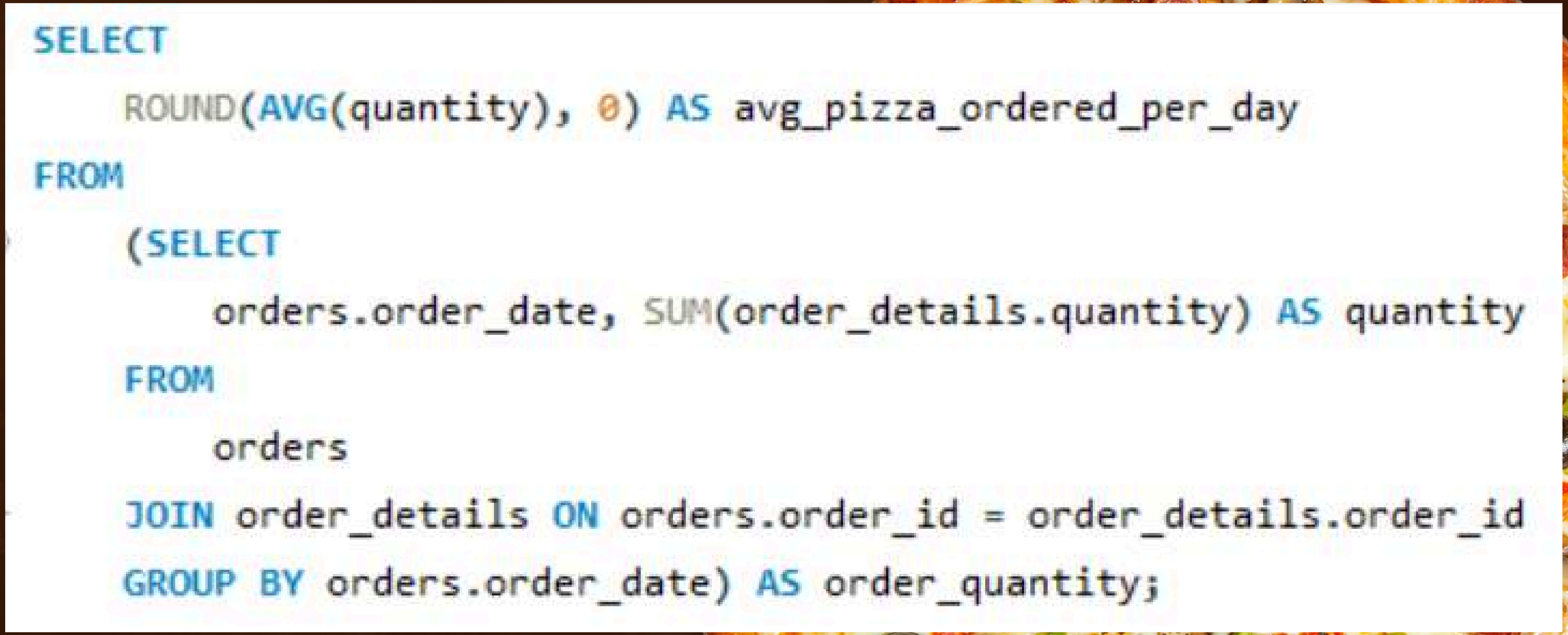
```
SELECT
    HOUR(order_time) AS hour, COUNT(order_id) AS order_count
FROM
    orders
GROUP BY HOUR(order_time);
```



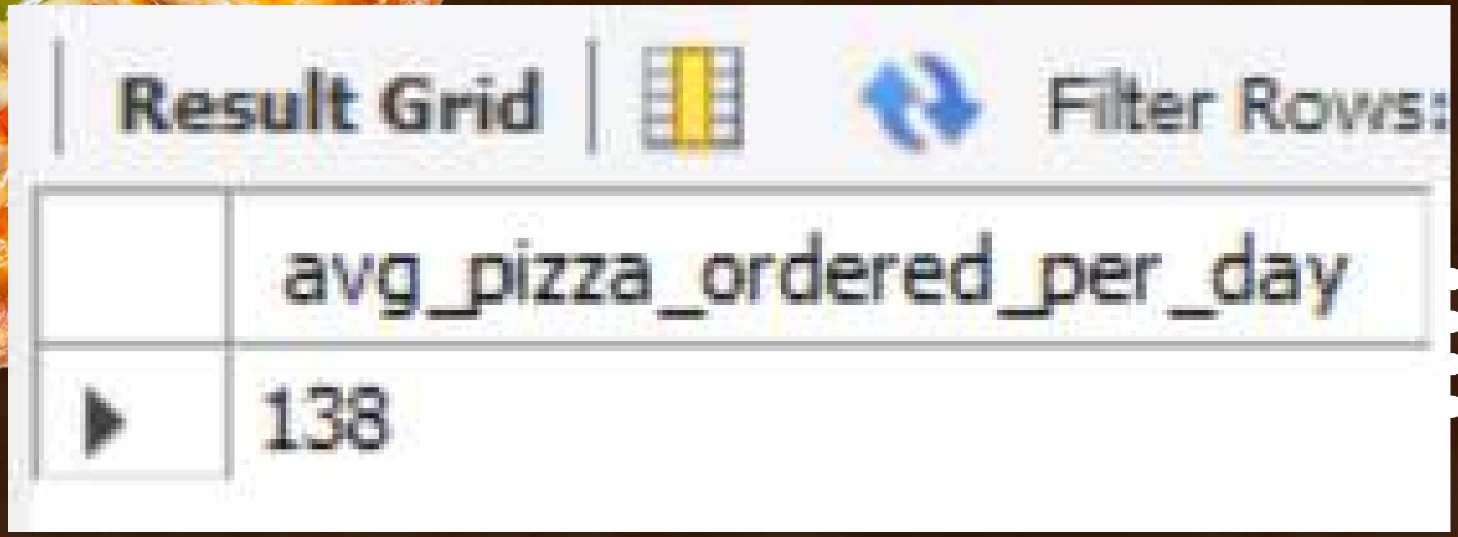
hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

INTERMEDIATE QUERY 2

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



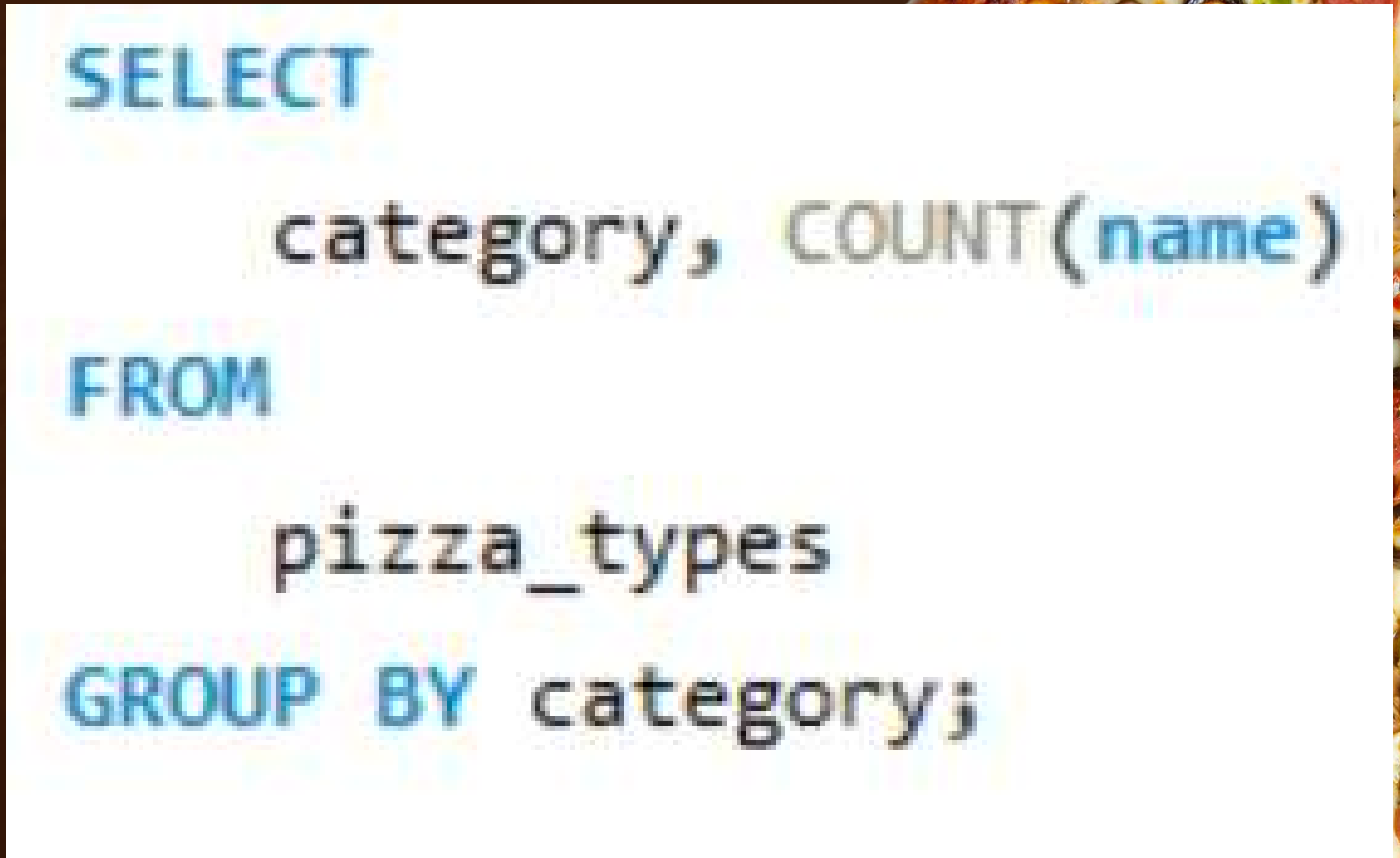
```
SELECT
    ROUND(AVG(quantity), 0) AS avg_pizza_ordered_per_day
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```



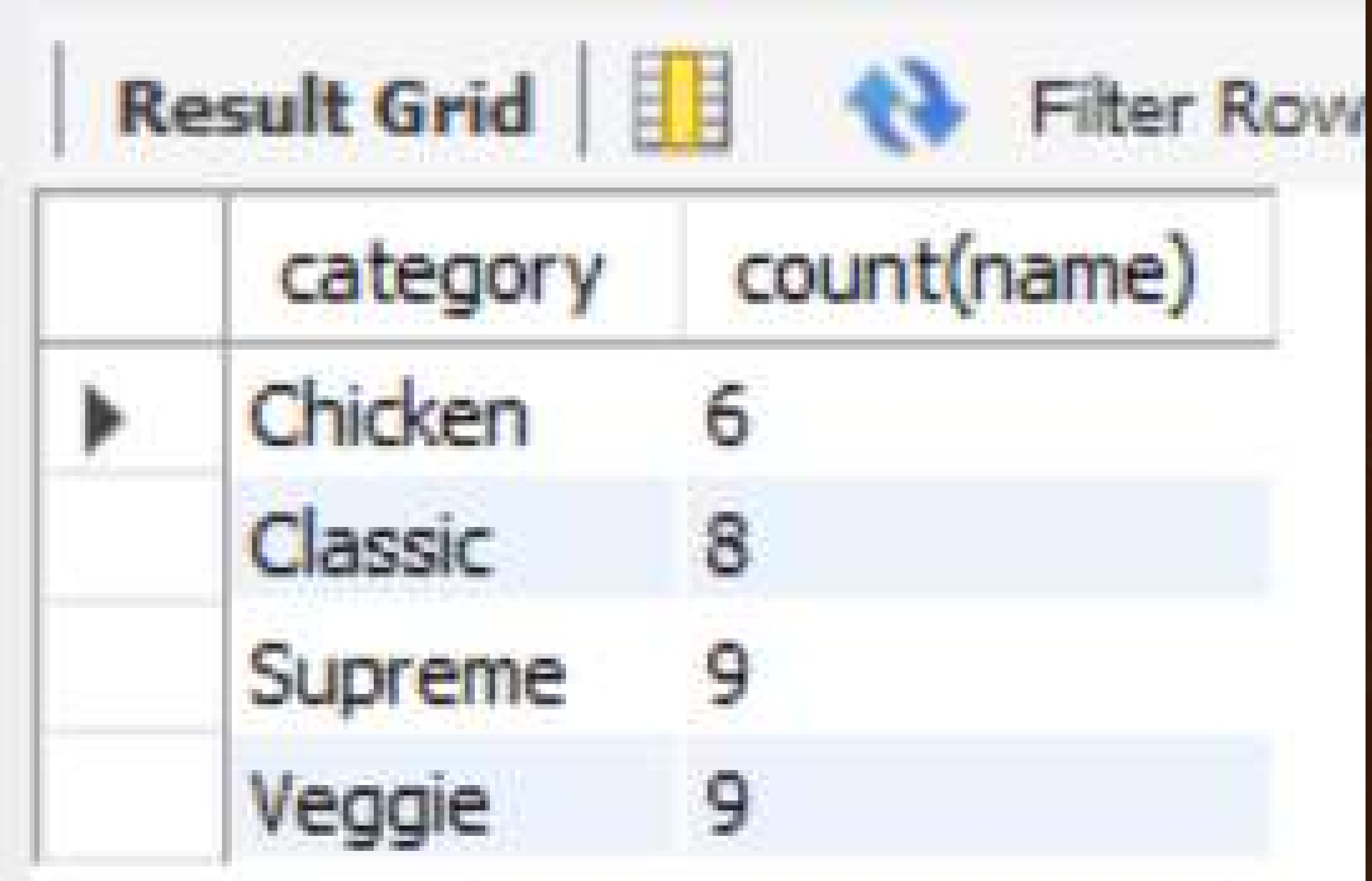
	avg_pizza_ordered_per_day
▶	138

INTERMEDIATE QUERY 3

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



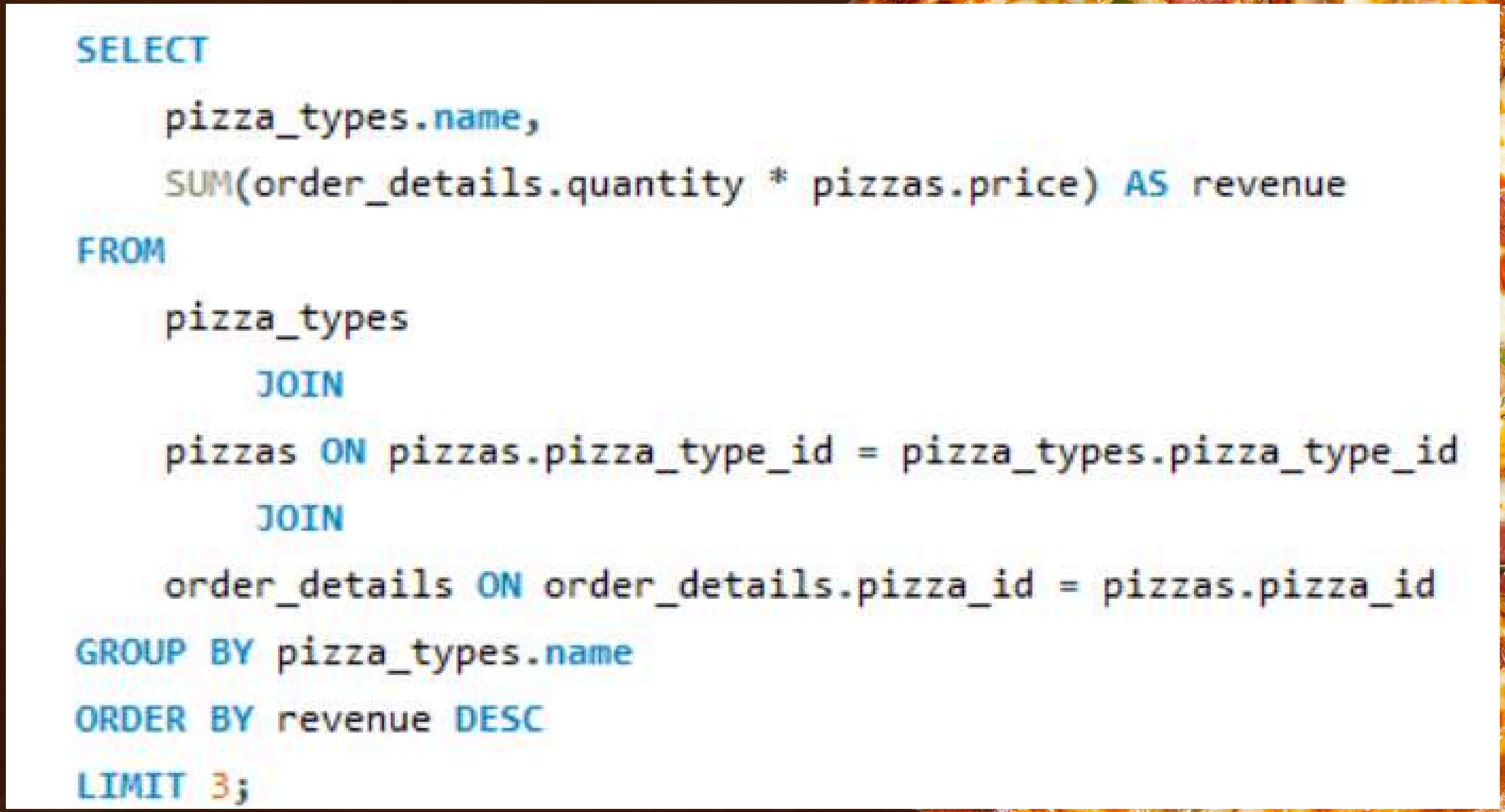
```
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```



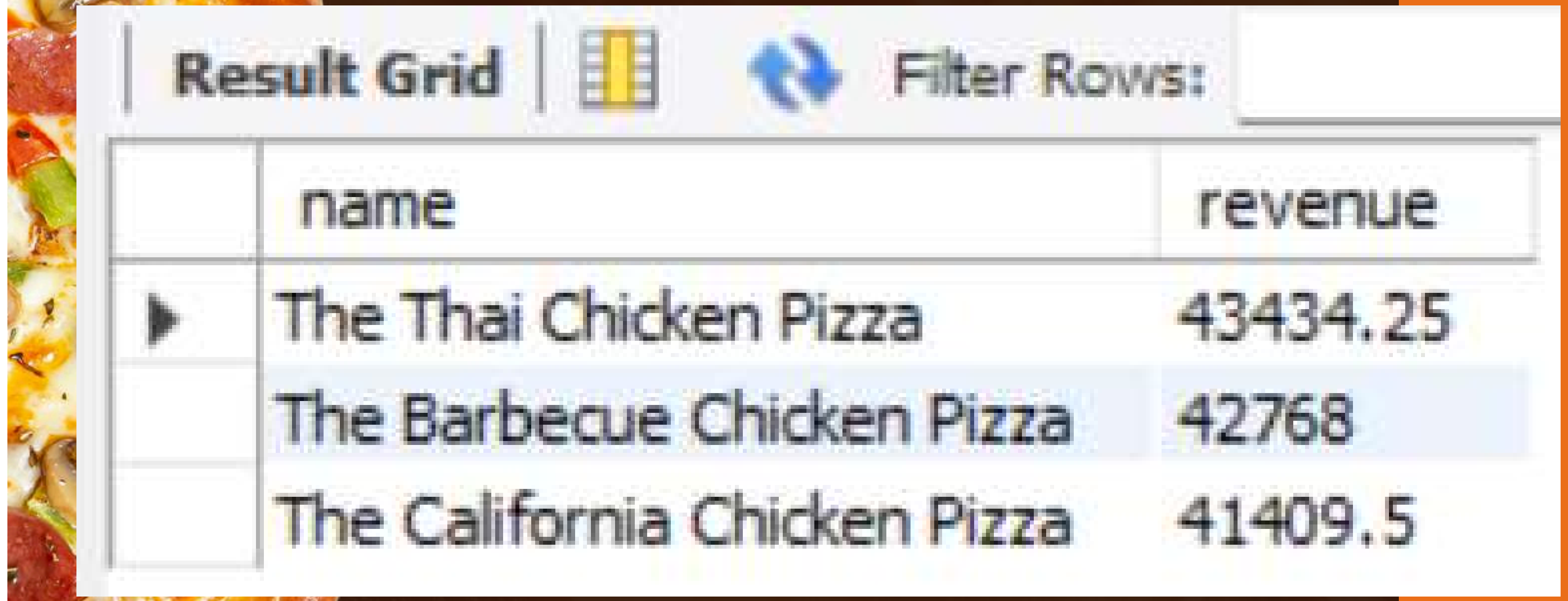
category	count(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

INTERMEDIATE QUERY 4

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



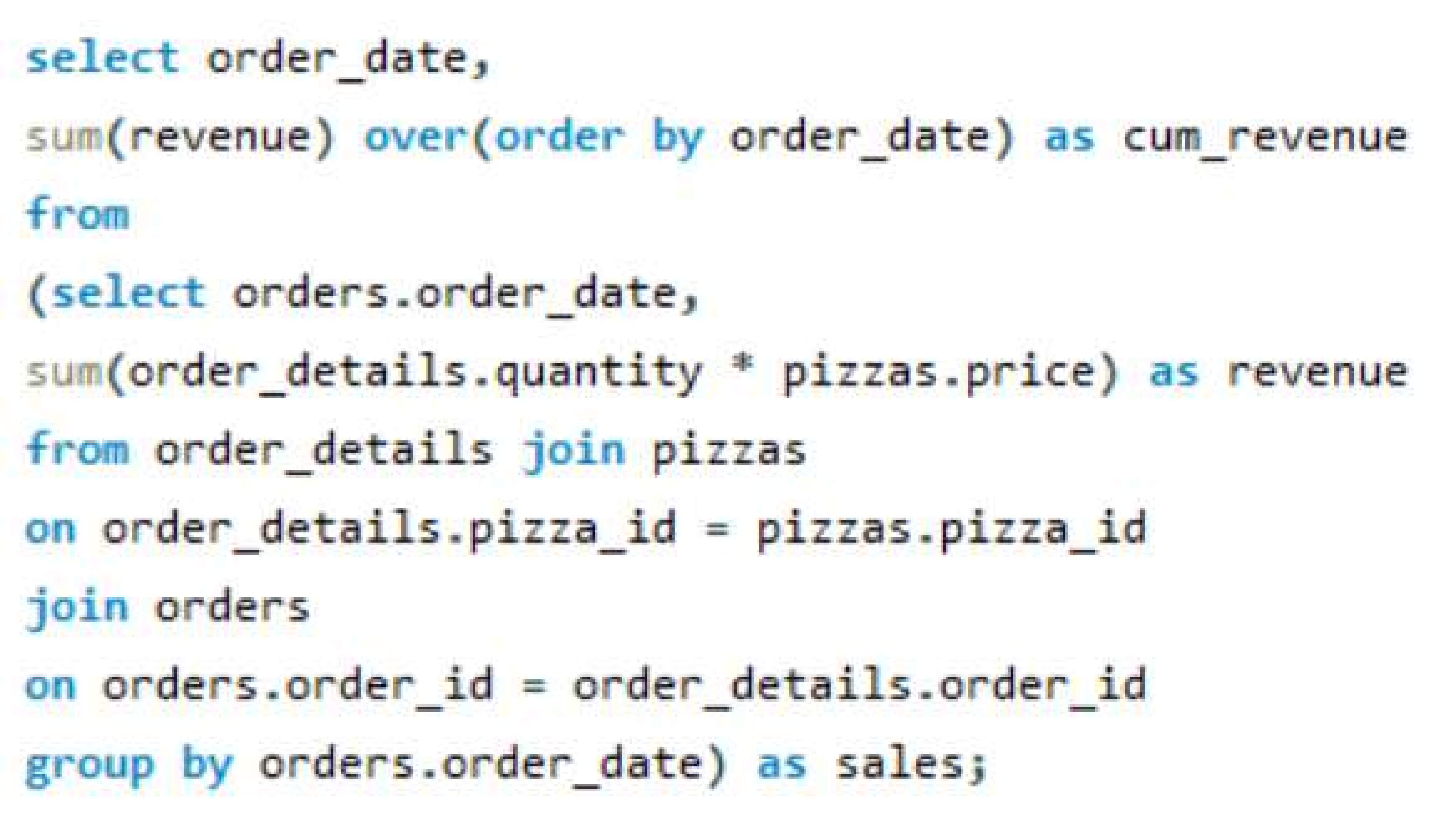
```
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue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



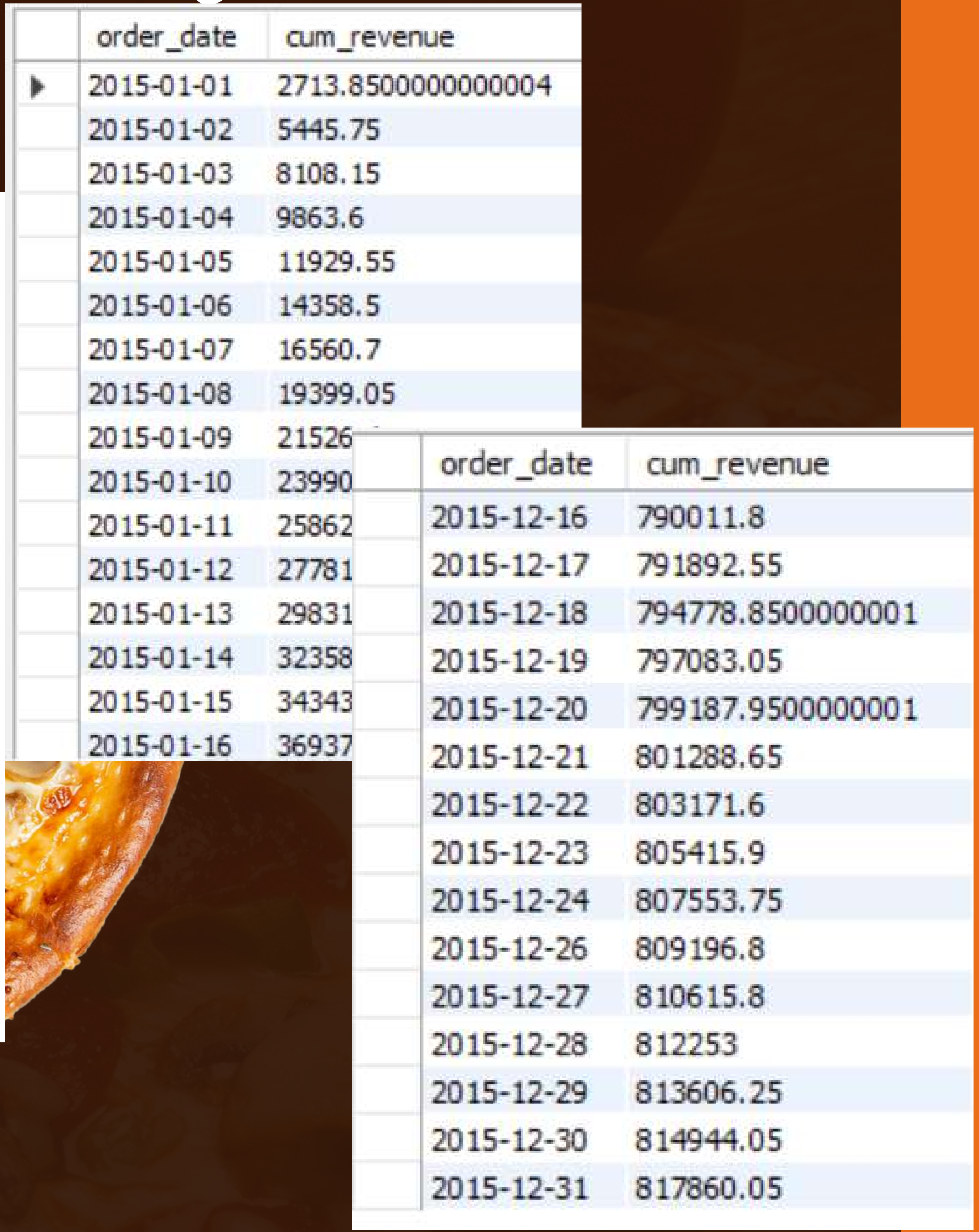
	name	revenue
▶	The Thai Chicken Pizza	43434.25
	The Barbecue Chicken Pizza	42768
	The California Chicken Pizza	41409.5

INTERMEDIATE QUERY 5

Analyze the cumulative revenue generated over time.



```
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 orders.order_id = order_details.order_id
group by orders.order_date) as sales;
```

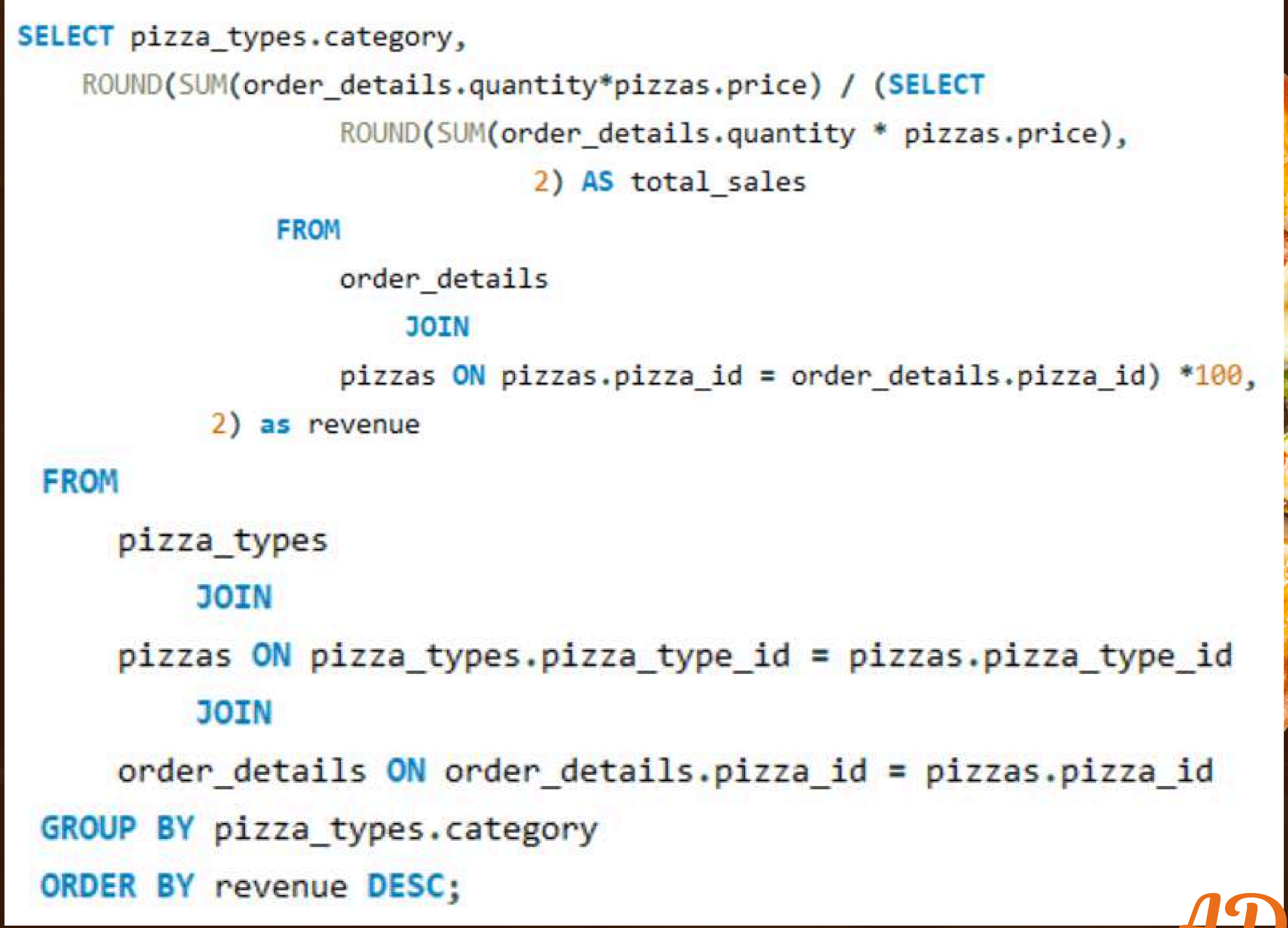


order_date	cum_revenue
2015-01-01	2713.8500000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526
2015-01-10	23990
2015-01-11	25862
2015-01-12	27781
2015-01-13	29831
2015-01-14	32358
2015-01-15	34343
2015-01-16	36937

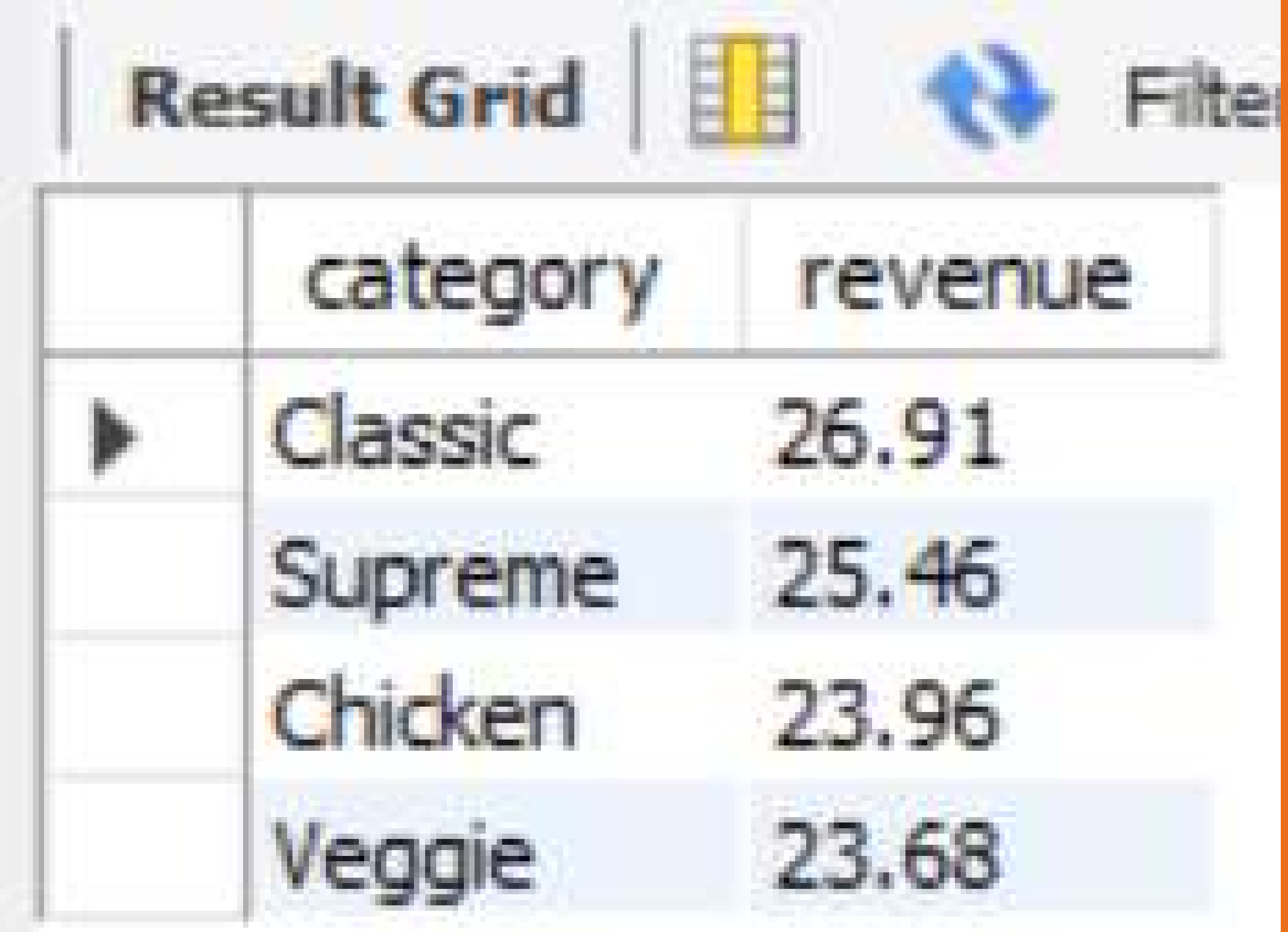
order_date	cum_revenue
2015-12-16	790011.8
2015-12-17	791892.55
2015-12-18	794778.8500000001
2015-12-19	797083.05
2015-12-20	799187.9500000001
2015-12-21	801288.65
2015-12-22	803171.6
2015-12-23	805415.9
2015-12-24	807553.75
2015-12-26	809196.8
2015-12-27	810615.8
2015-12-28	812253
2015-12-29	813606.25
2015-12-30	814944.05
2015-12-31	817860.05

ADVANCED QUERY 1

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



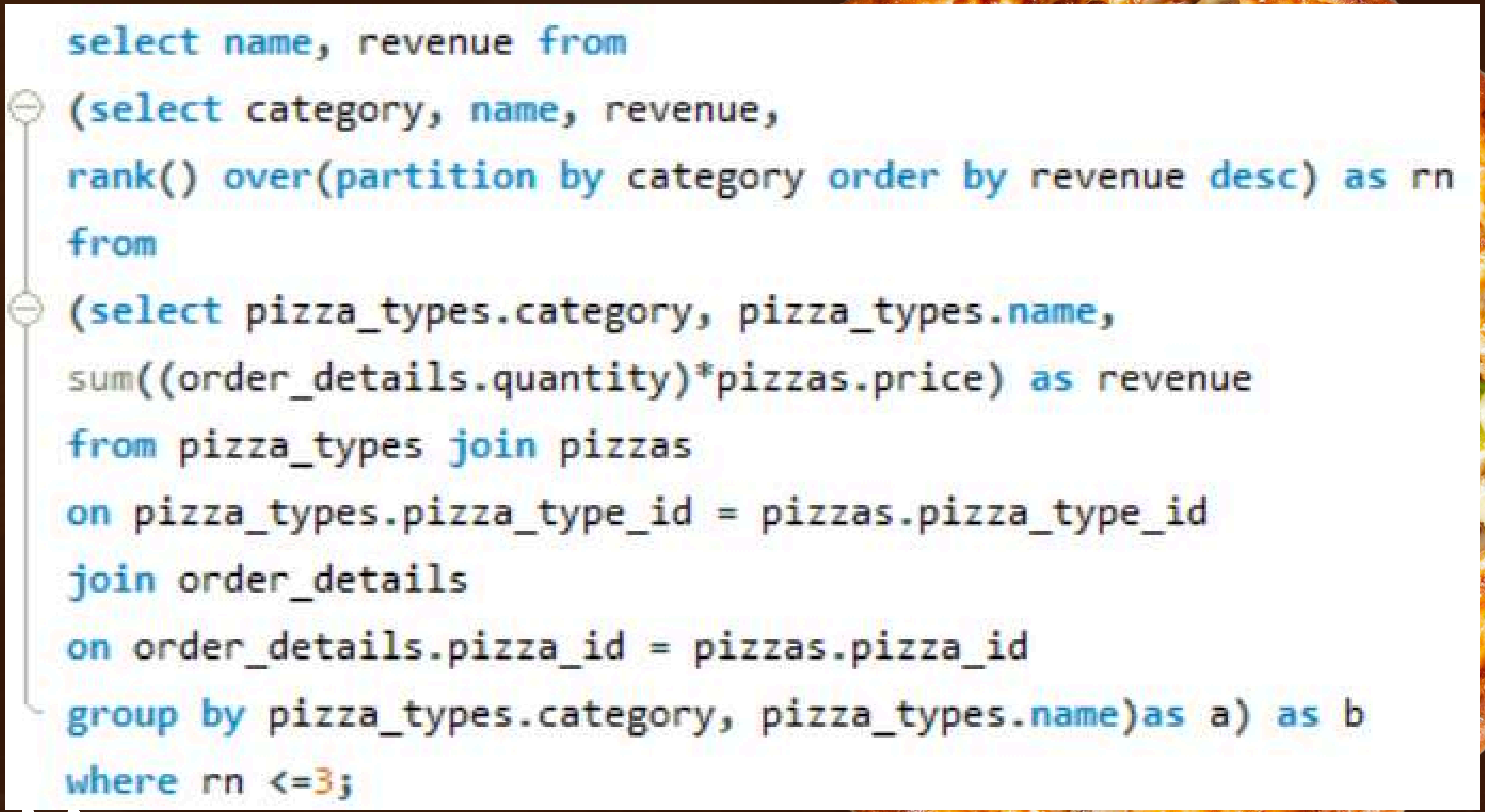
```
SELECT pizza_types.category,
    ROUND(SUM(order_details.quantity*pizzas.price) / (SELECT
                    ROUND(SUM(order_details.quantity * pizzas.price),
                                2) AS total_sales
                FROM
                    order_details
                        JOIN
                    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;
```



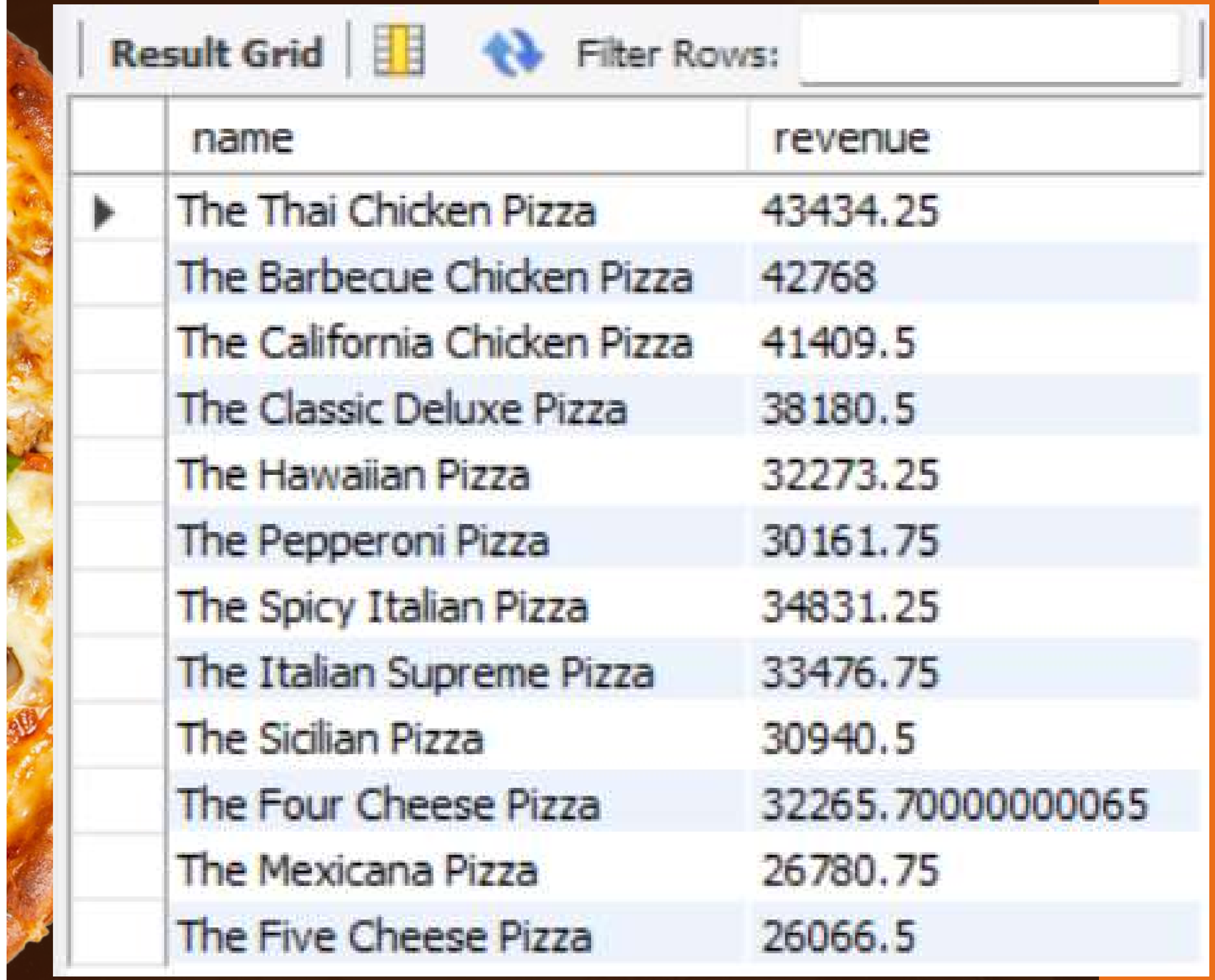
category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

ADUANCED QUERY 2

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((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;
```



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

ADVANCED QUERY 3

PIZZA SALES ANALYSIS USING SQL QUERIES

THANK YOU FOR ATTENTION