Pizza Sales Data Analysis Using SQL

SQL-Based Business Insights Project

By - Devanshi Chouhan
Course - Integrated MBA (BBA Hons. + MBA)
Specialization - Artificial intelligence and Data science
tools used - MySQL workbench

Project Overview

This project analyzes a pizza sales dataset using SQL to extract meaningful business insights.
The objective is to demonstrate practical SQL skills including aggregation, joins, grouping, subqueries, and revenue analysis.
The analysis covers:

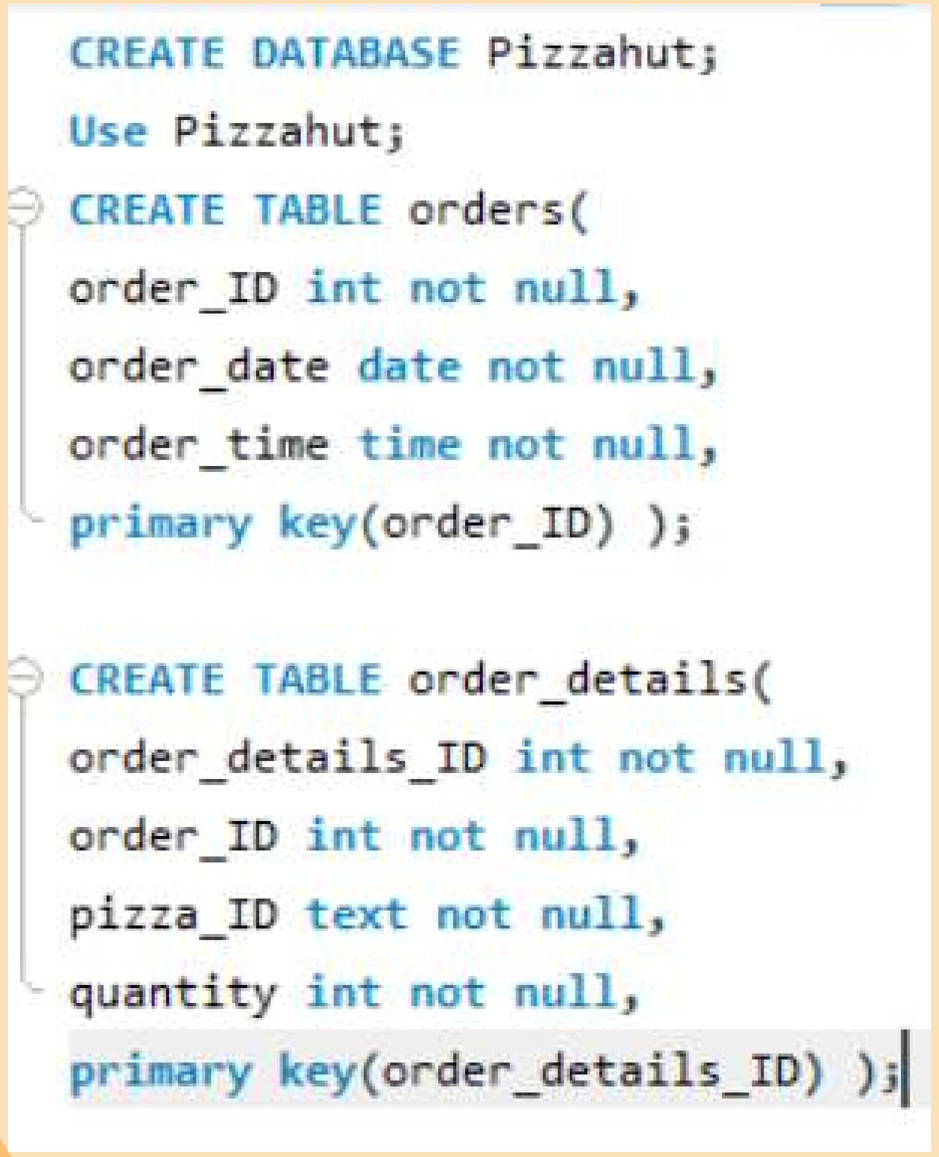
- Sales performance
- Customer ordering patterns
- Revenue contribution
- Category-wise and time-based insights

Business Objectives

The analysis aims to answer the following business questions:

- What is total revenue?
- Which pizza generates the highest revenue?
- What size is most preferred?
- What are peak ordering hours?
- Which categories drive the business?

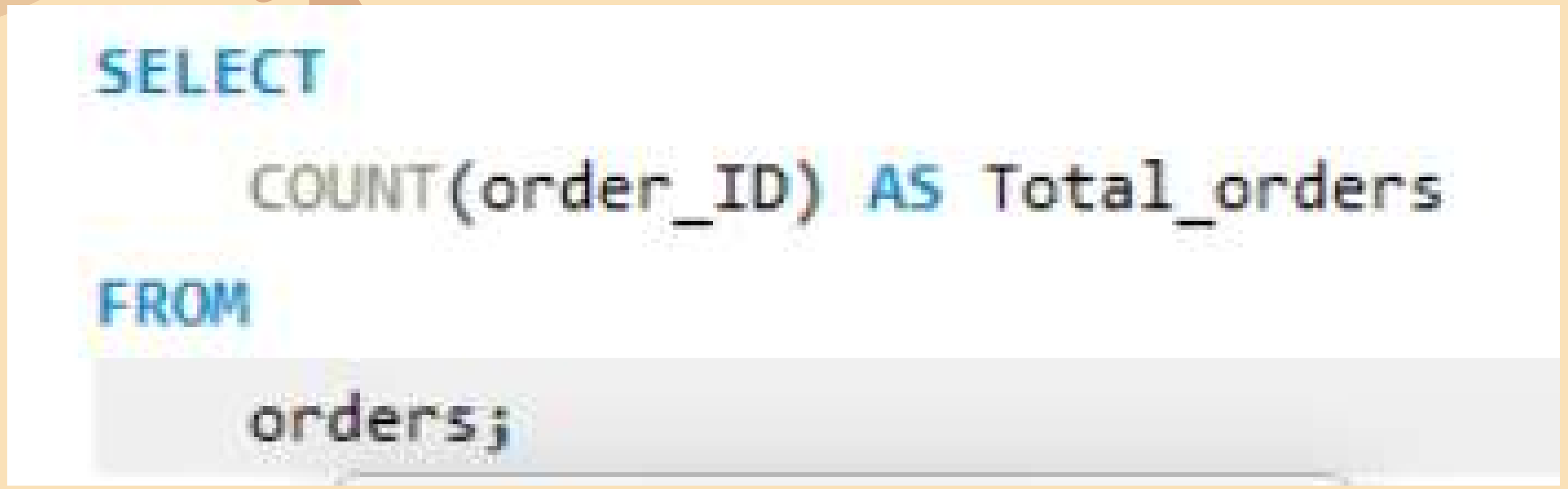
Dataset contains a Database named Pizzahut , including tables orders, order_details, pizzas and pizza_types



```
CREATE DATABASE Pizzahut;
Use Pizzahut;
CREATE TABLE orders(
order_ID int not null,
order_date date not null,
order_time time not null,
primary key(order_ID) );

CREATE TABLE order_details(
order_details_ID int not null,
order_ID int not null,
pizza_ID text not null,
quantity int not null,
primary key(order_details_ID) );
```

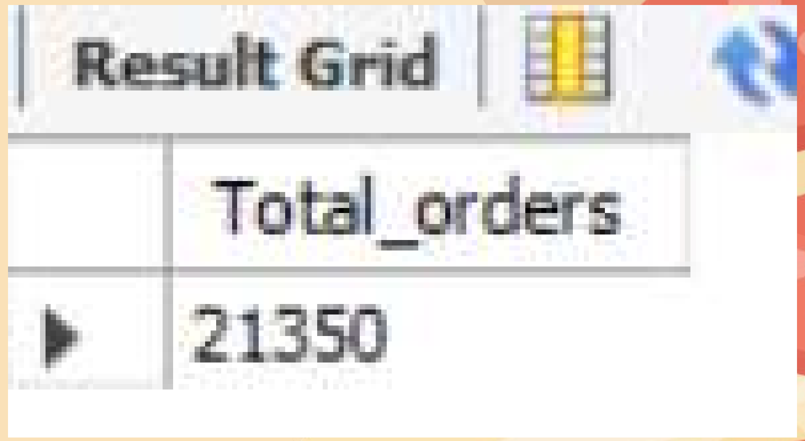
Retrieve the total number of orders placed



```
SELECT
    COUNT(order_ID) AS Total_orders
FROM
    orders;
```

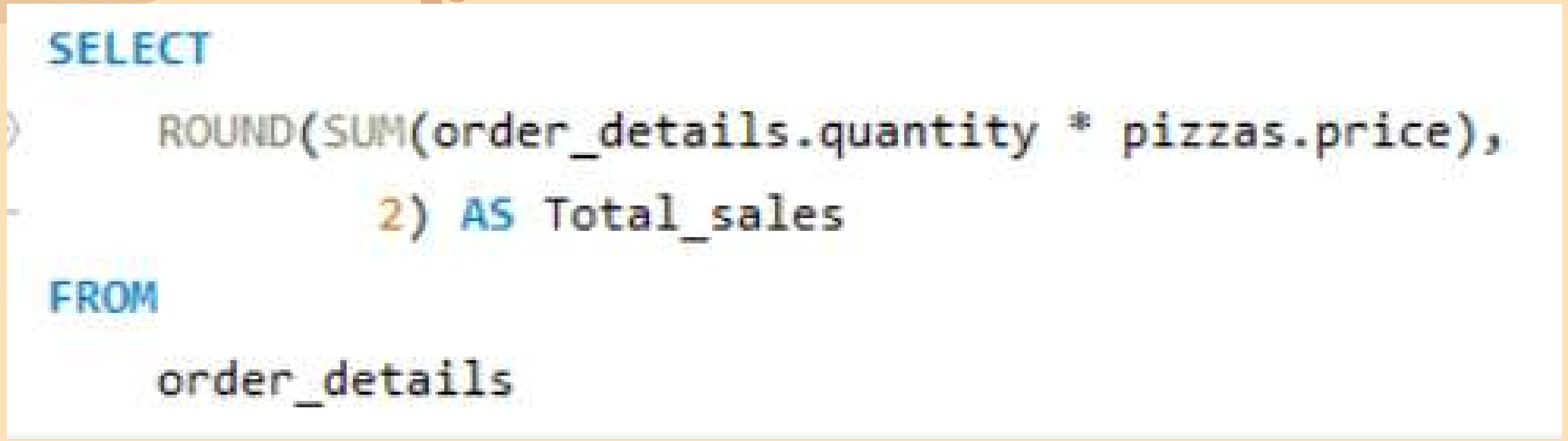
Syntax

Result



```
Total_orders
21350
```

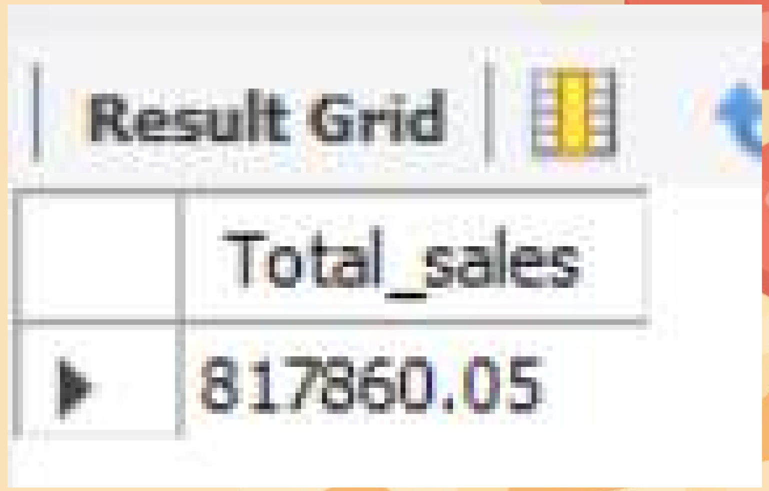
Calculate the total revenue generated from pizza sales.



```
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS Total_sales
FROM
    order_details
```

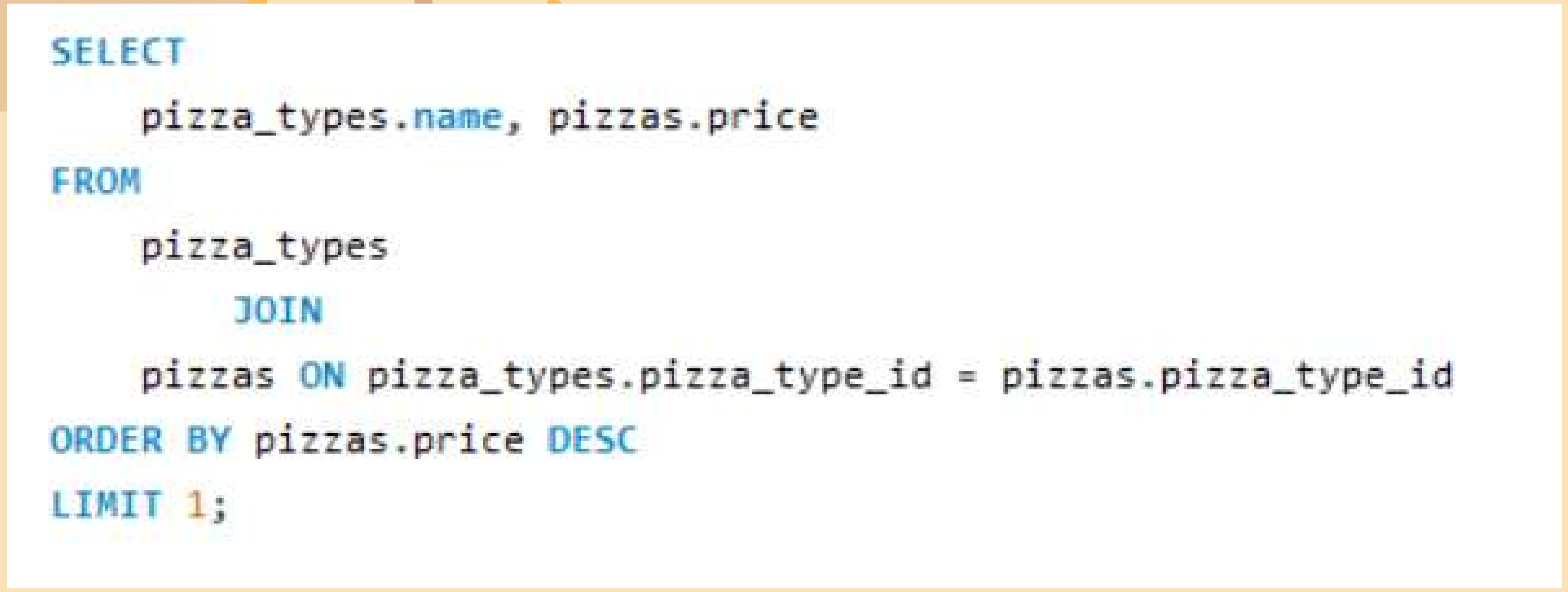
Syntax

Result



```
Total_sales
817860.05
```

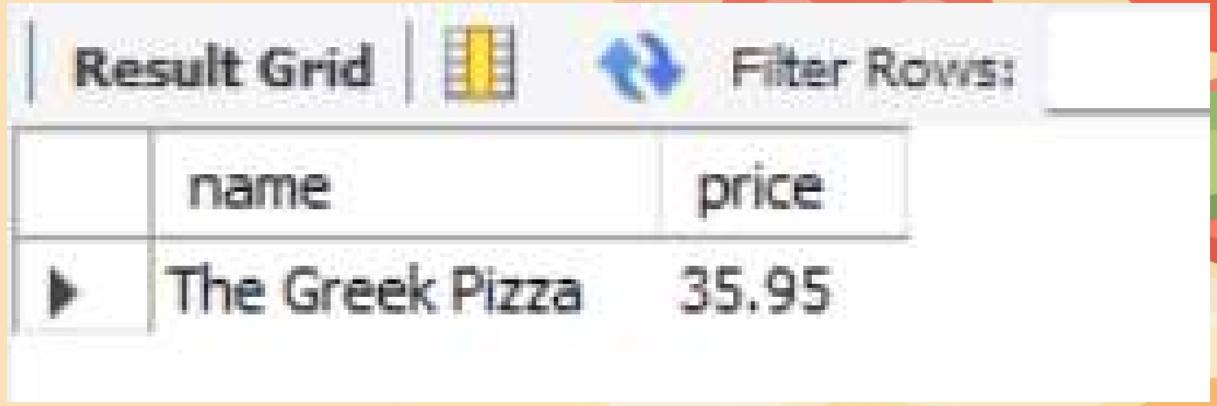
Identify the highest-priced pizza



```
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;
```

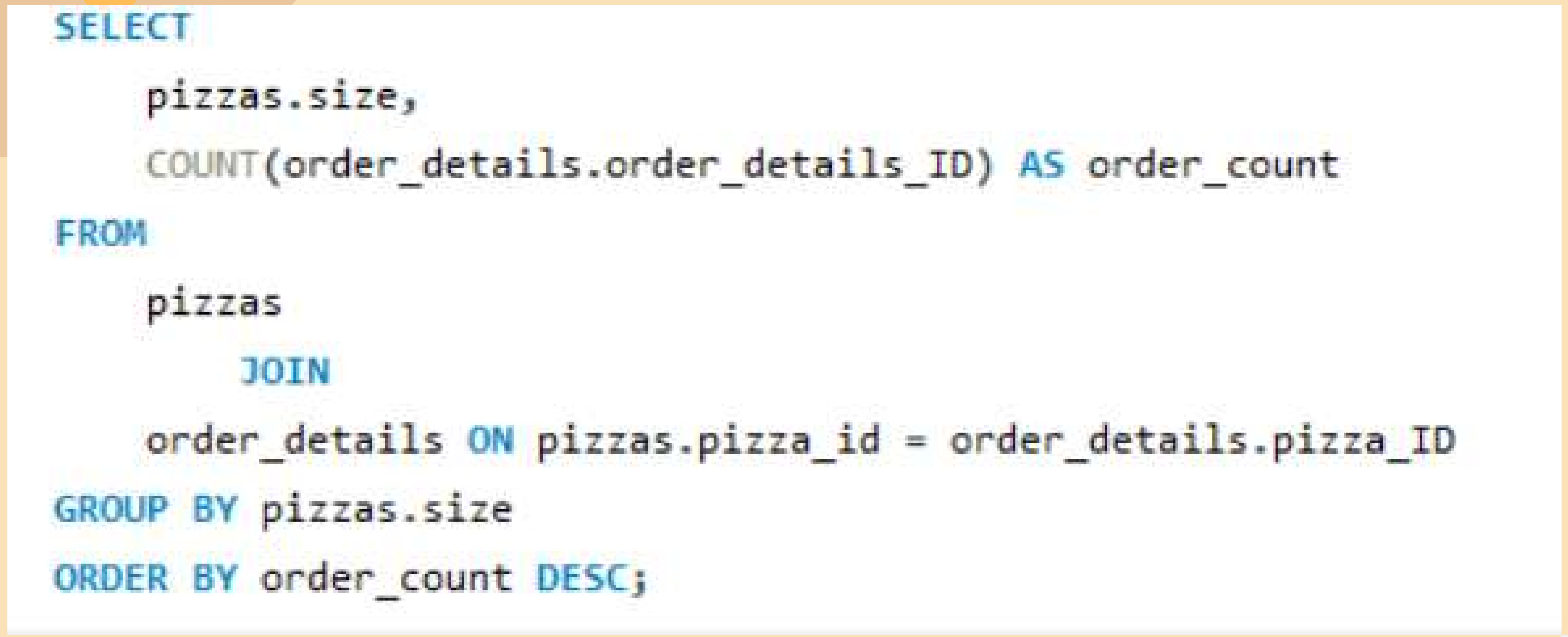
Syntax

Result



name	price
The Greek Pizza	35.95

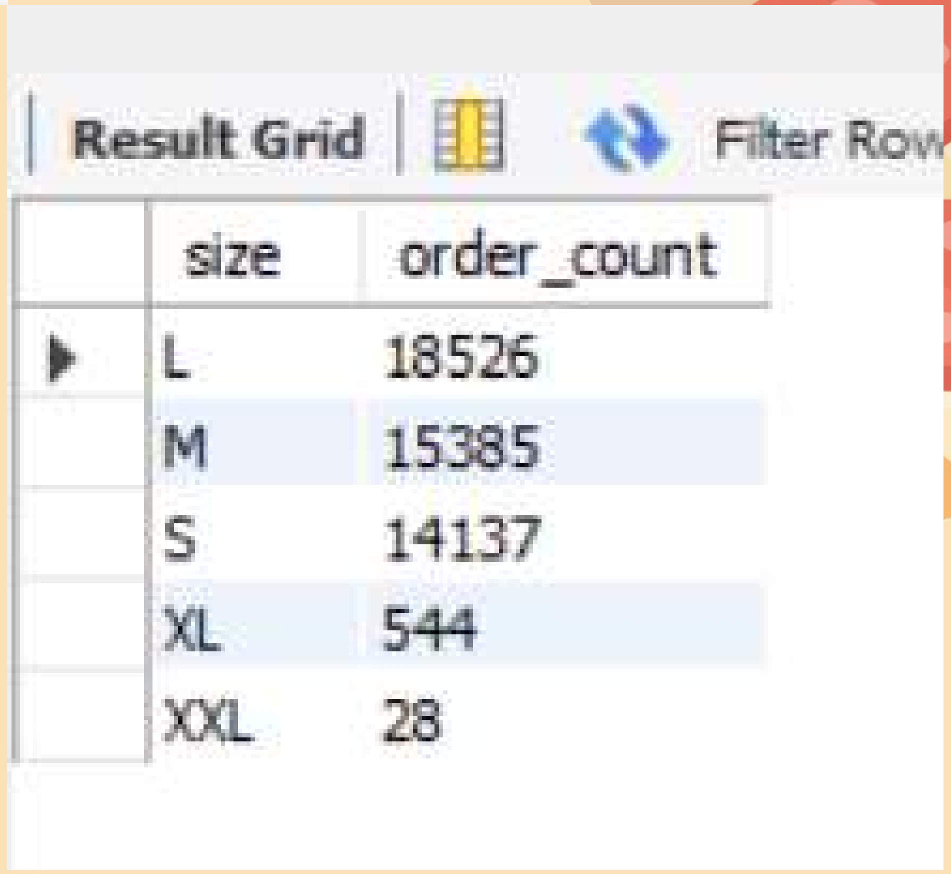
Identify the most common pizza size ordered



```
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;
```

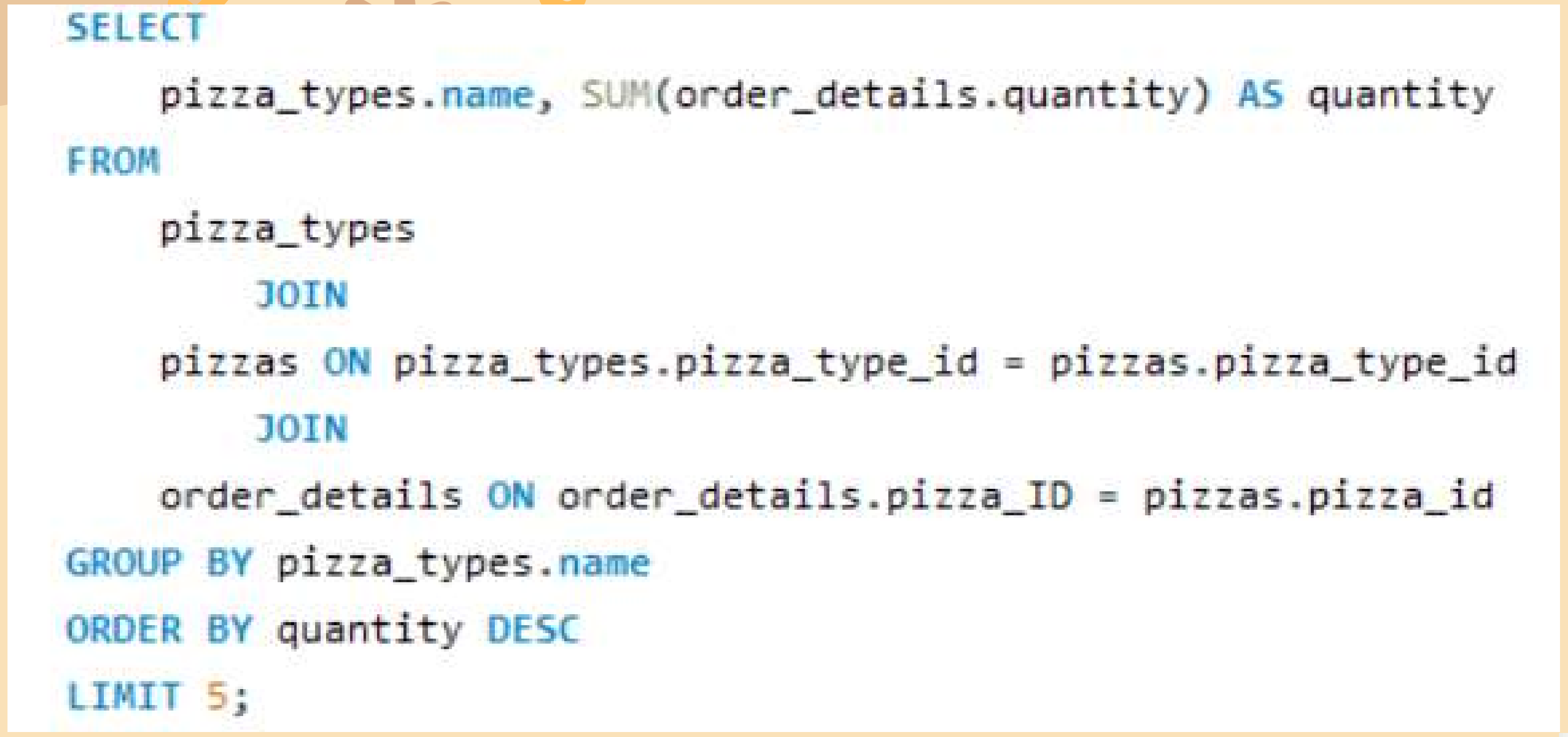
Syntax

Result



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

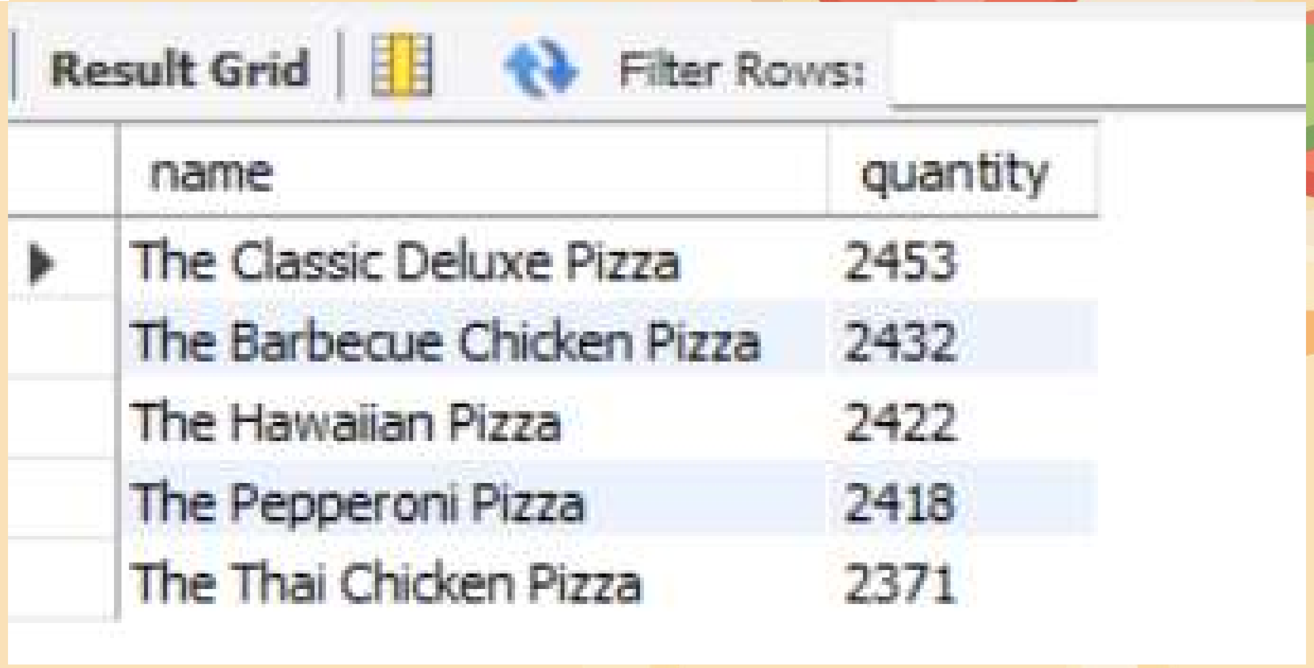
List the top 5 most ordered pizza types along with their quantities.



```
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;
```

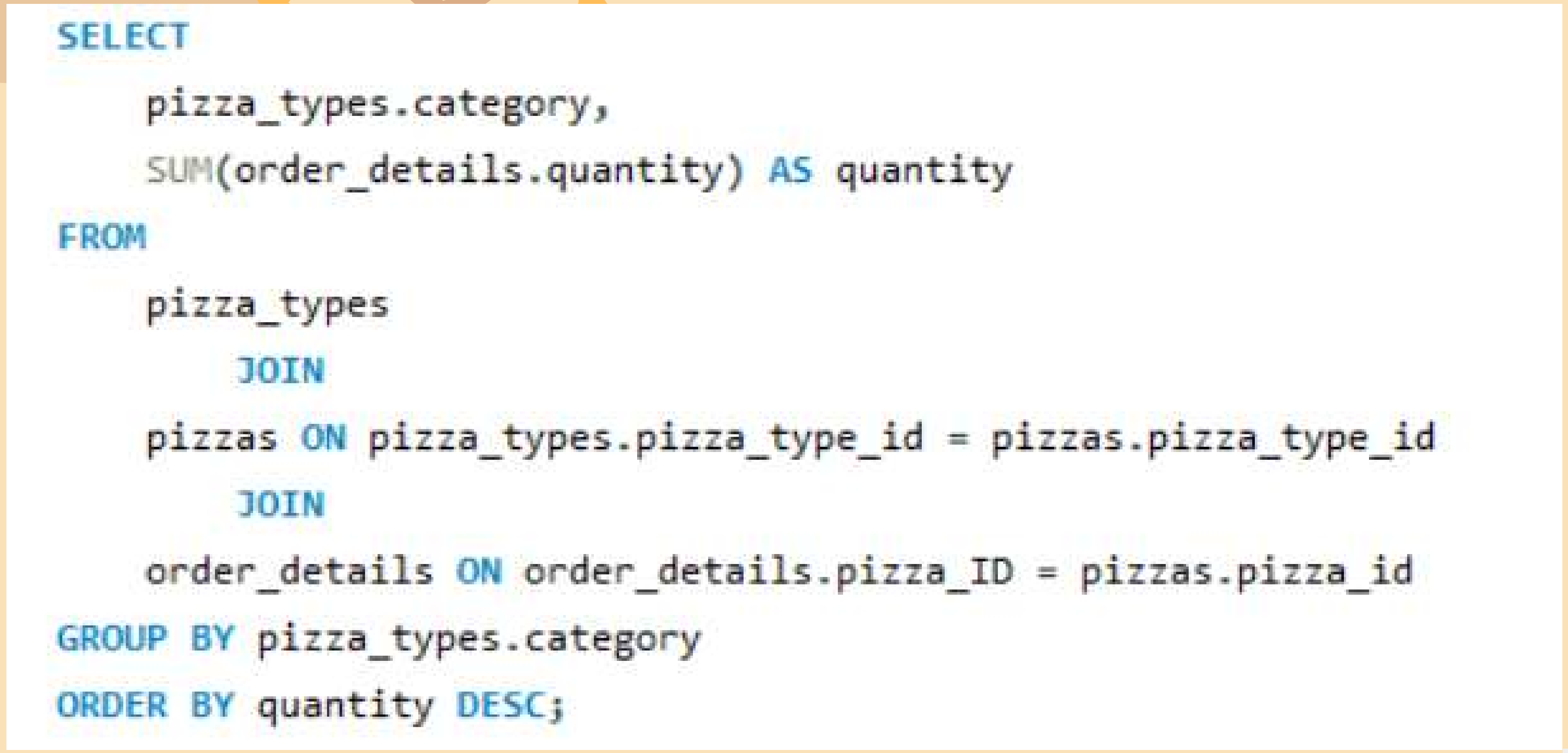
Syntax

Result



```
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
```

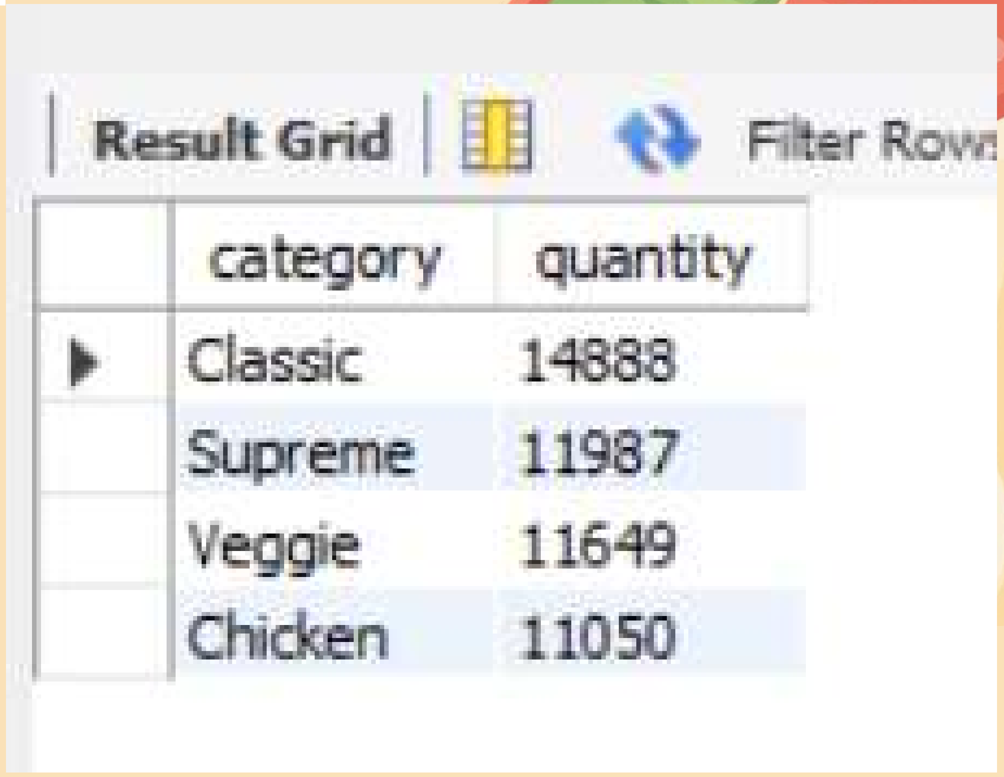
Join the necessary tables to find the total quantity of each pizza category ordered.



```
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;
```

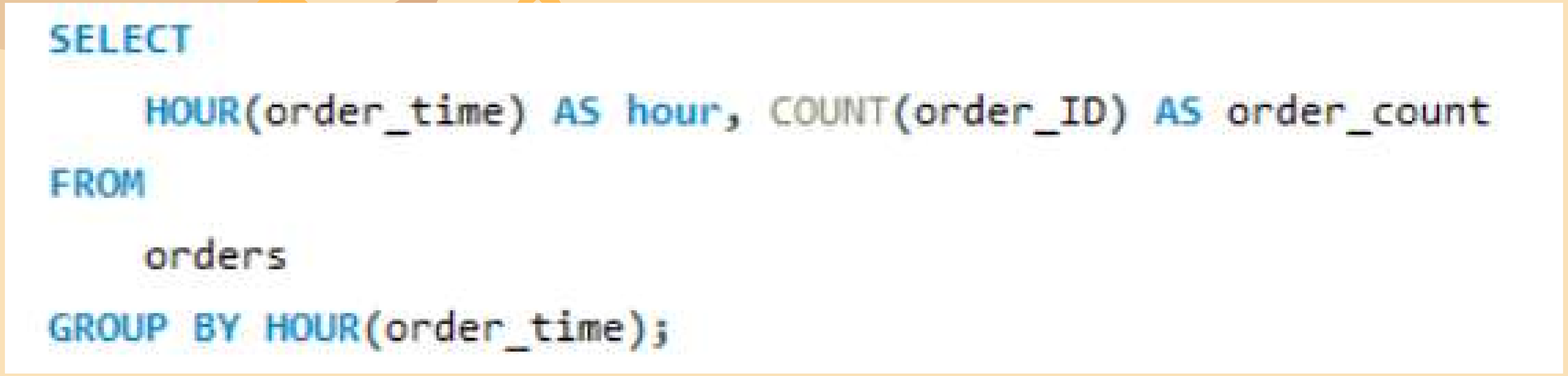
Syntax

Result



category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

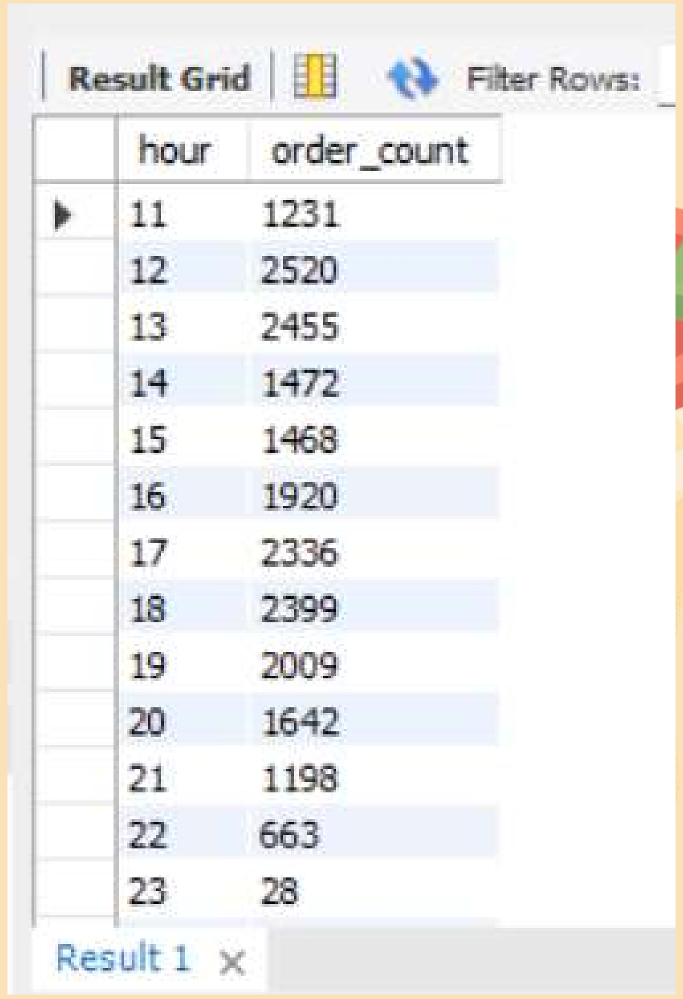
Determine the distribution of orders by hour of the day



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

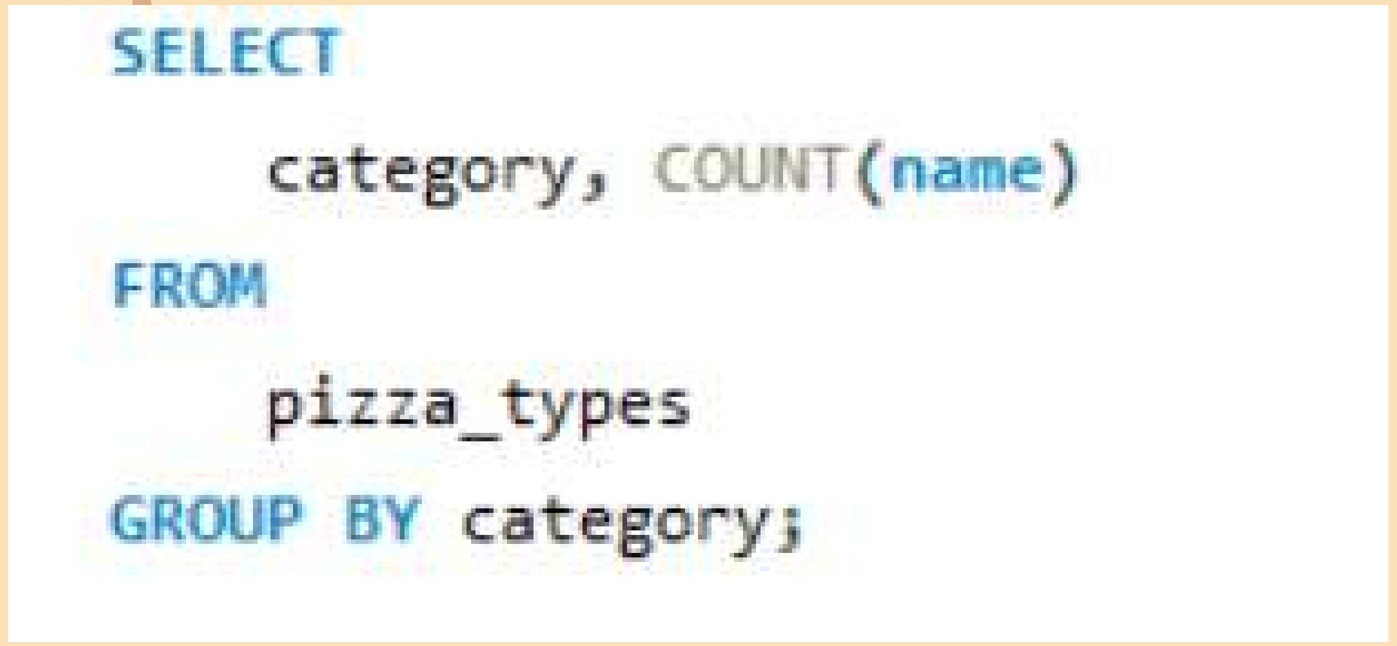
Syntax

Result



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

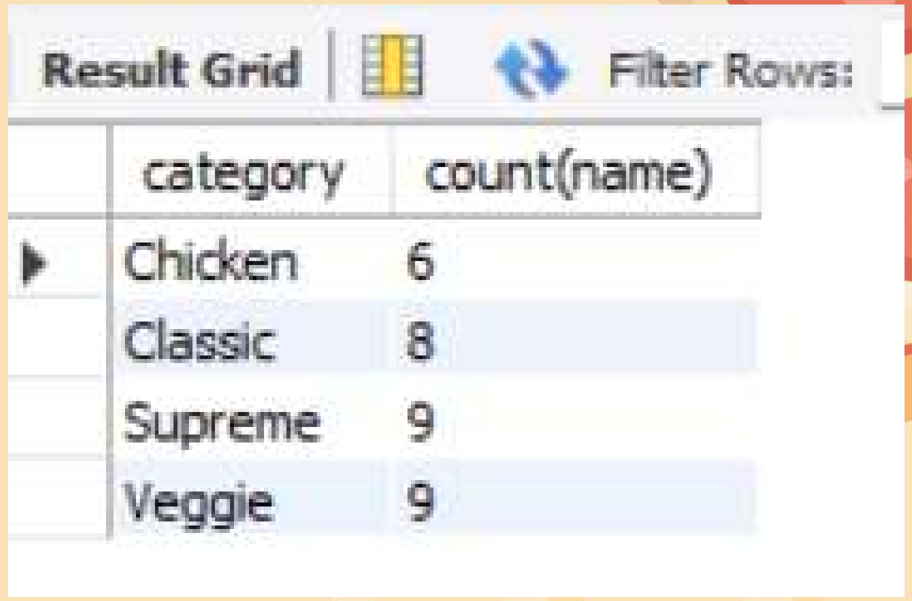
Join relevant tables to find the category-wise distribution of pizzas.



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

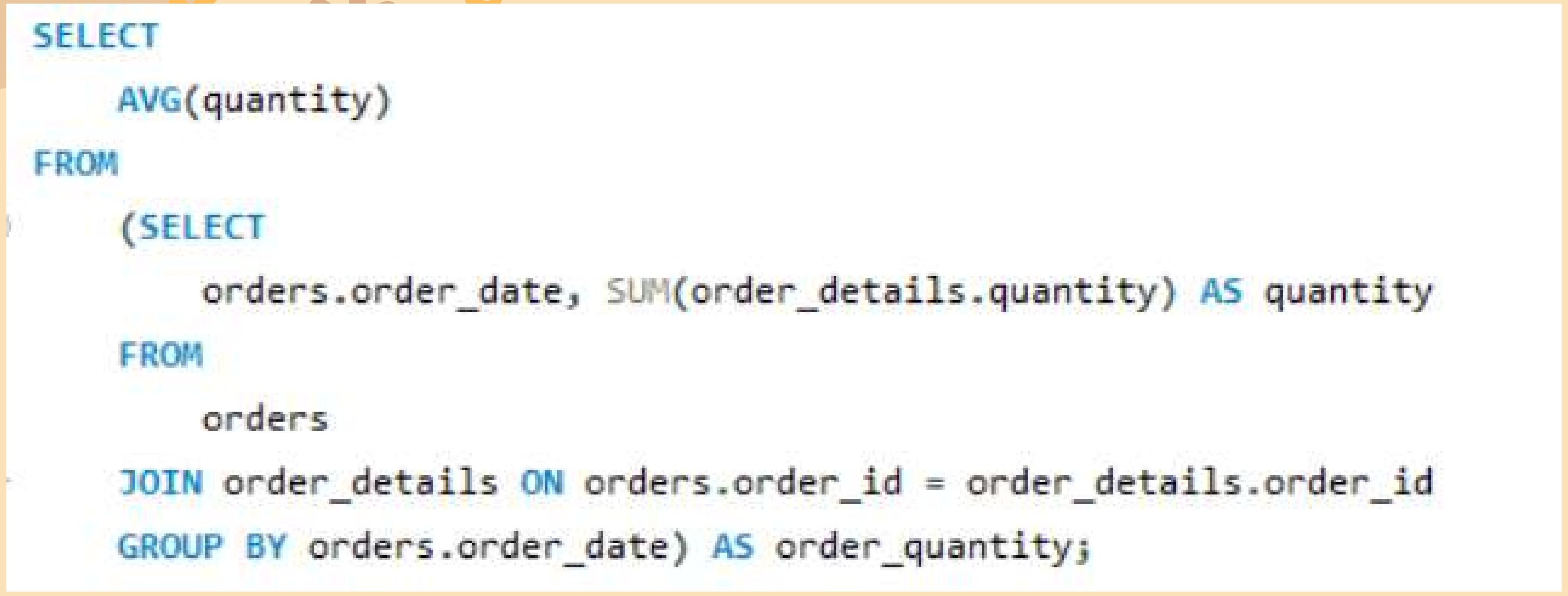
Syntax

Result



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

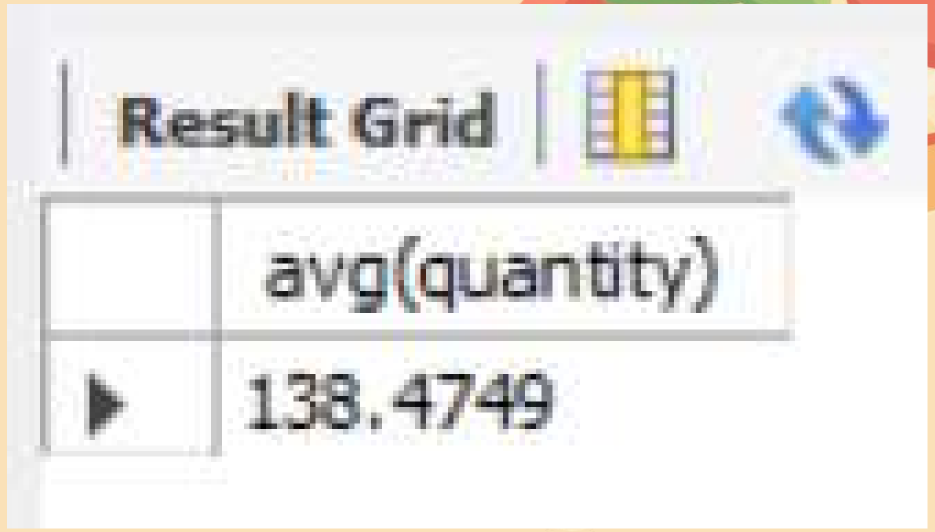
Group the orders by date and calculate the average number of pizzas ordered per day.



```
SELECT
    AVG(quantity)
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;
```

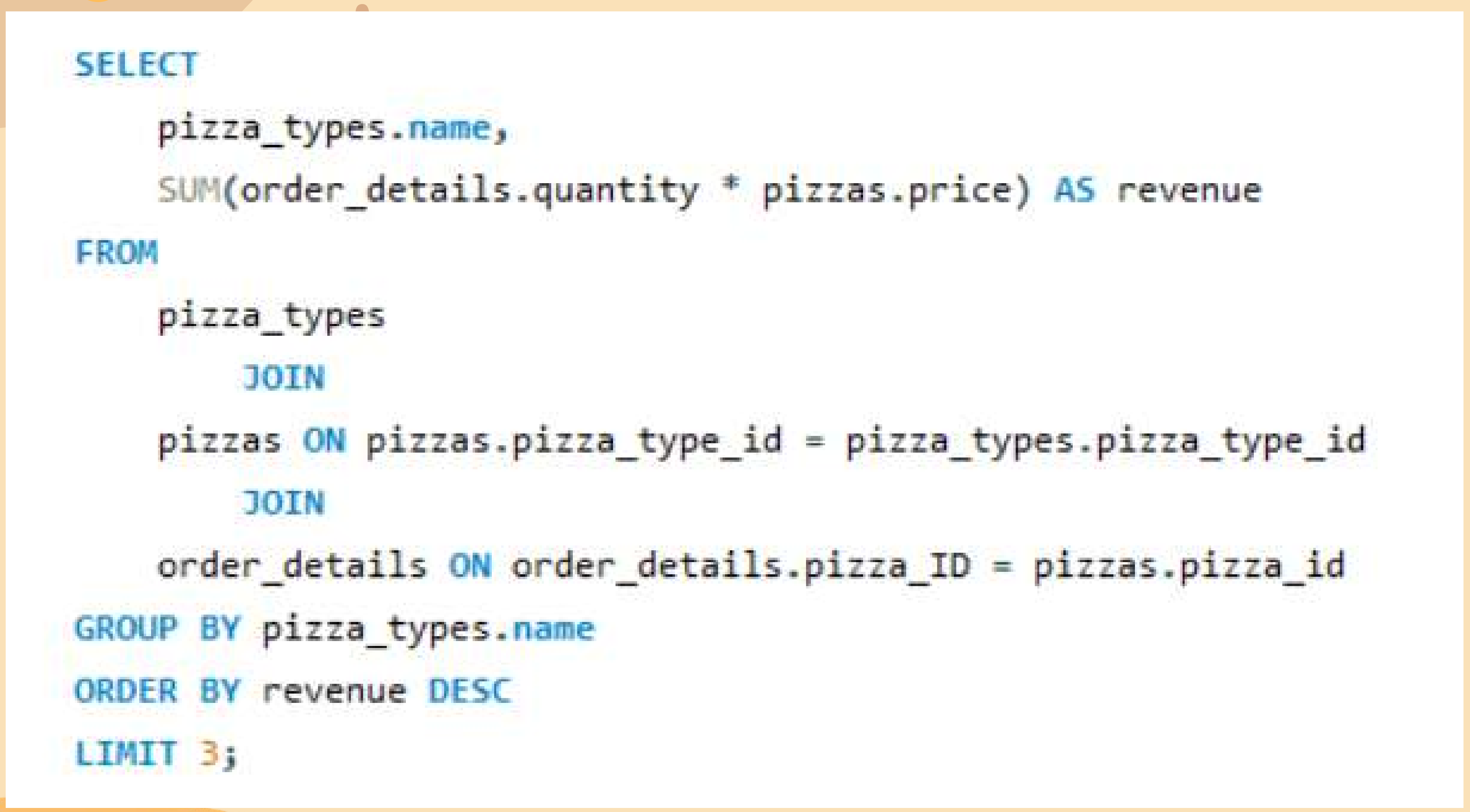
Syntax

Result



```
Result Grid
   avg(quantity)
▶  138.4749
```

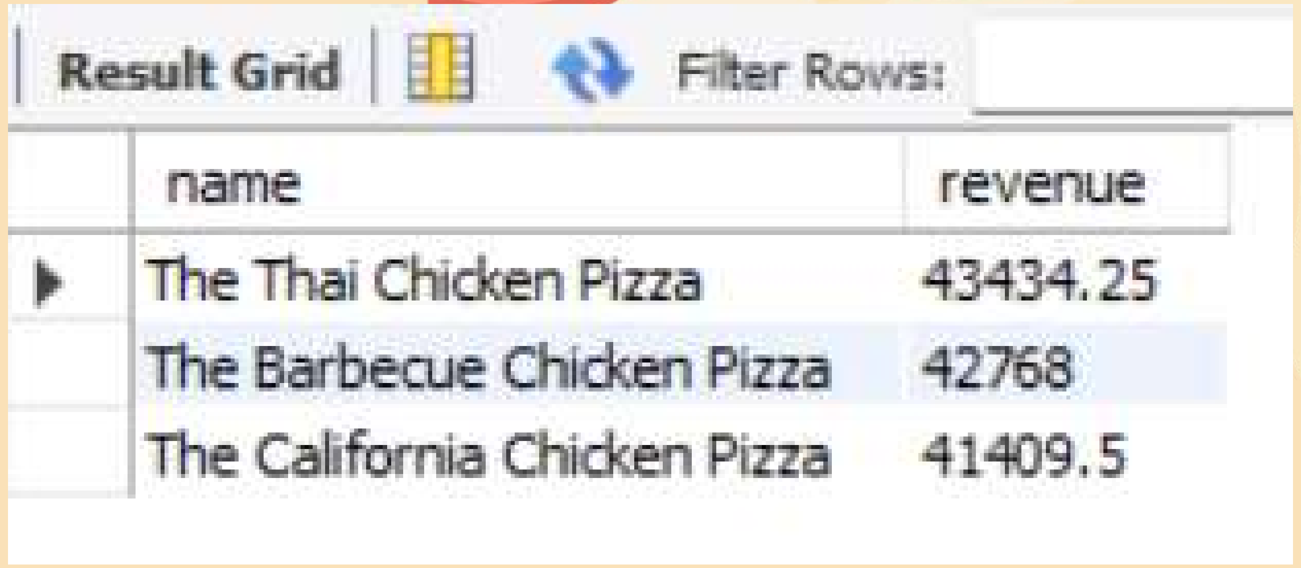
Determine the top 3 most ordered pizza types based on revenue.



```
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;
```

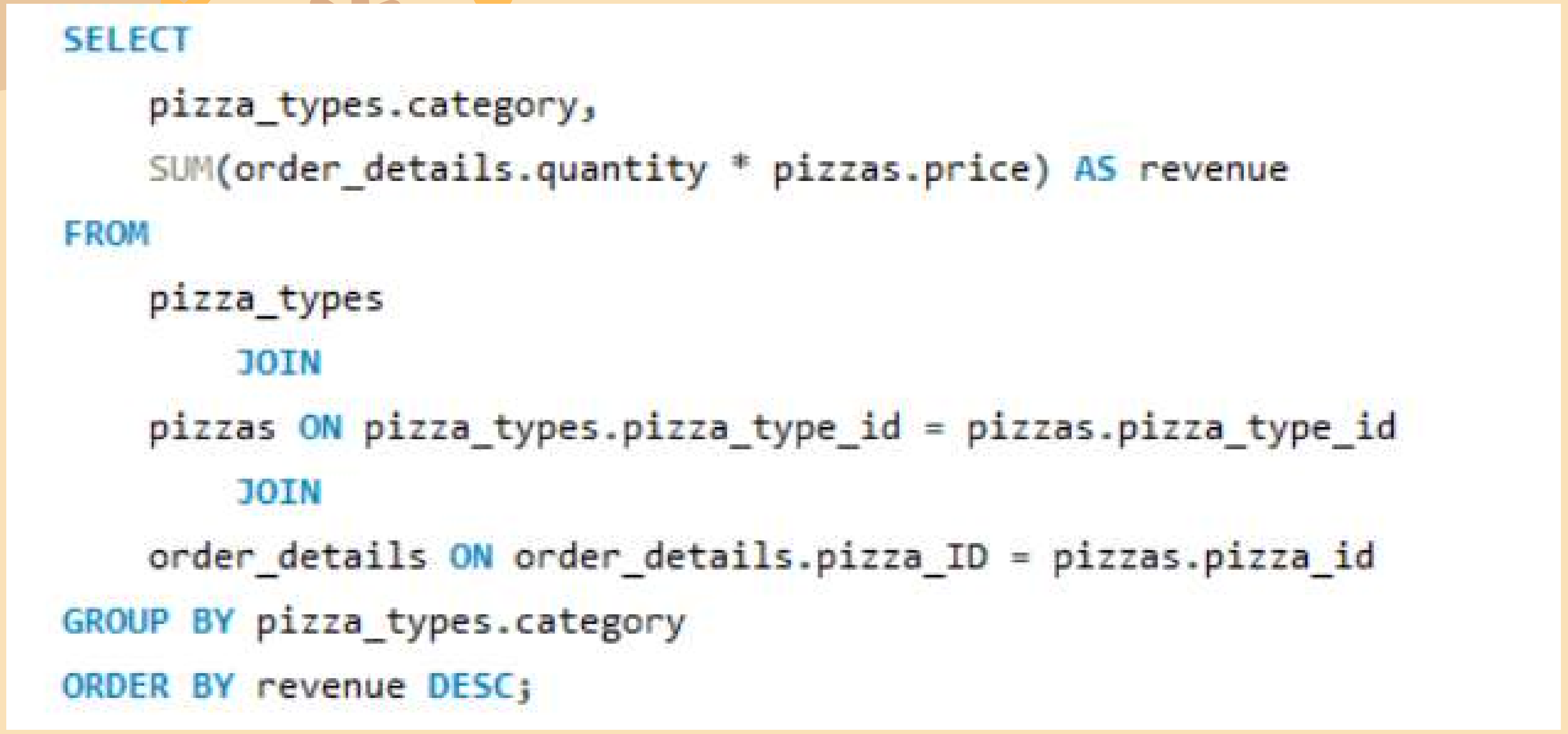
Syntax

Result



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

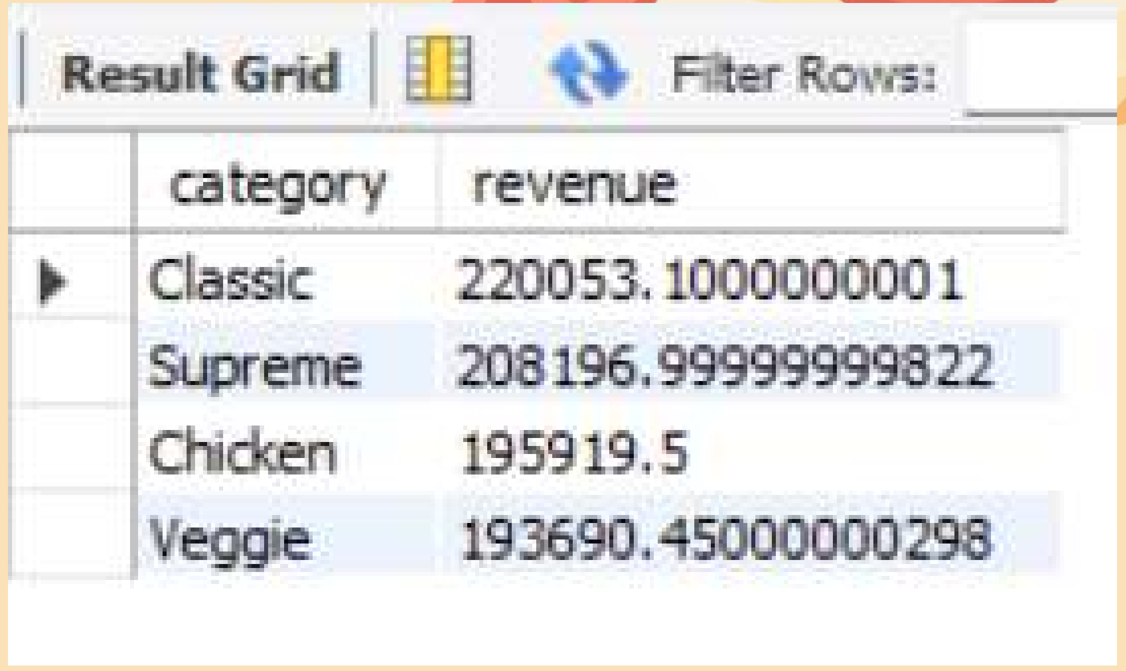
Calculate the percentage contribution of each pizza type to total revenue.



```
SELECT
    pizza_types.category,
    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
ORDER BY revenue DESC;
```

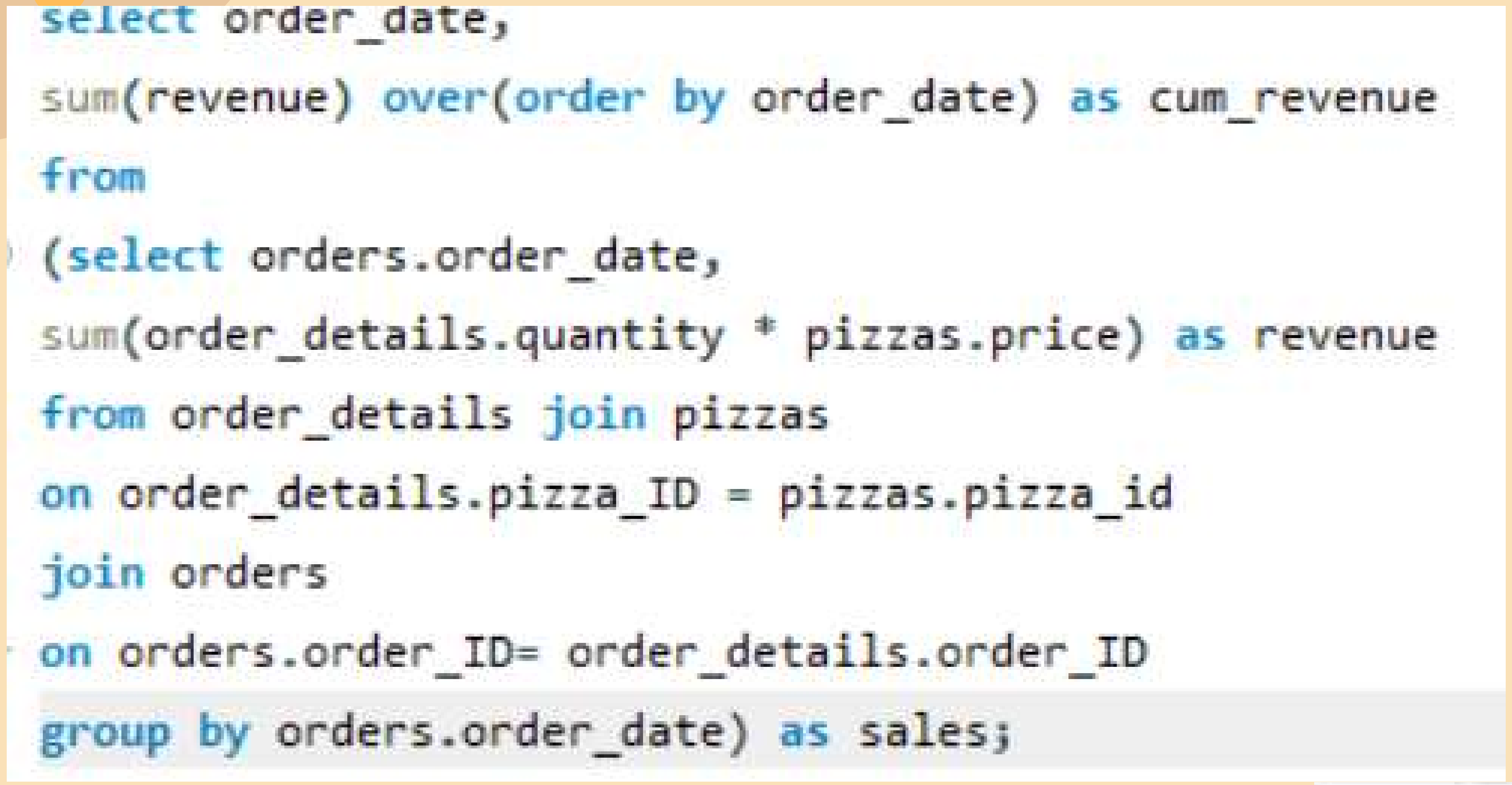
Syntax

Result



category	revenue
Classic	220053.1000000001
Supreme	208196.99999999822
Chicken	195919.5
Veggie	193690.45000000298

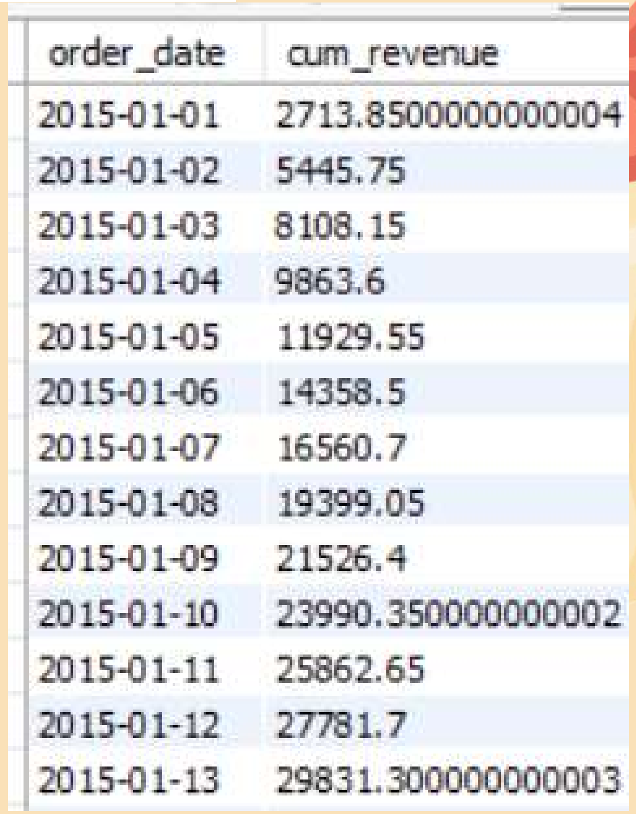
Analyze the cumulative revenue generated over time.



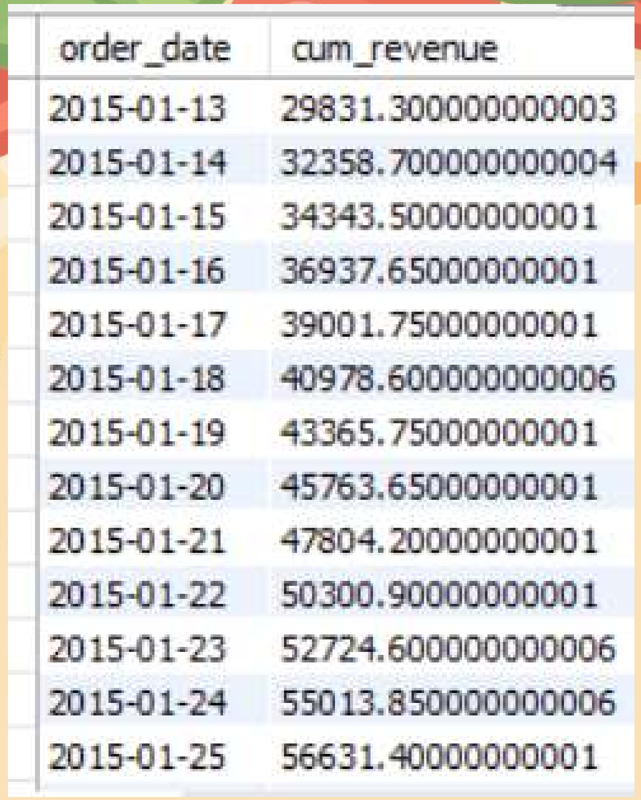
```
select order_date,
sum(revenue) over(order by order_date) as cum_revenue
from
(select orders.order_date,
sum(order_details.quantity * pizzas.price) as revenue
from order_details join pizzas
on order_details.pizza_ID = pizzas.pizza_id
join orders
on orders.order_ID= order_details.order_ID
group by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

Syntax

Result

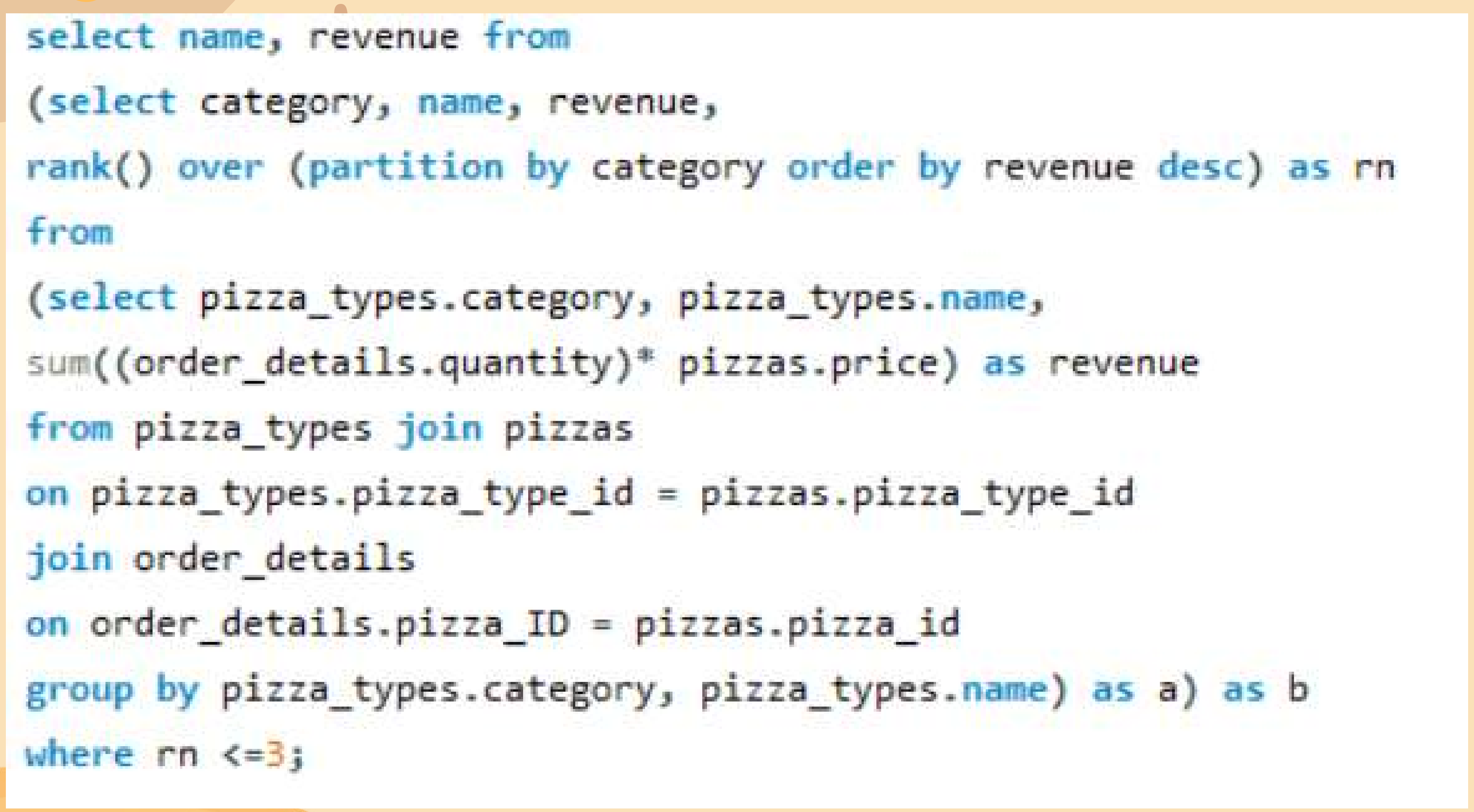


order_date	cum_revenue
2015-01-01	2713.8500000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.4
2015-01-10	23990.350000000002
2015-01-11	25862.65
2015-01-12	27781.7
2015-01-13	29831.300000000003



order_date	cum_revenue
2015-01-13	29831.300000000003
2015-01-14	32358.700000000004
2015-01-15	34343.50000000001
2015-01-16	36937.65000000001
2015-01-17	39001.75000000001
2015-01-18	40978.600000000006
2015-01-19	43365.75000000001
2015-01-20	45763.65000000001
2015-01-21	47804.20000000001
2015-01-22	50300.90000000001
2015-01-23	52724.600000000006
2015-01-24	55013.850000000006
2015-01-25	56631.40000000001

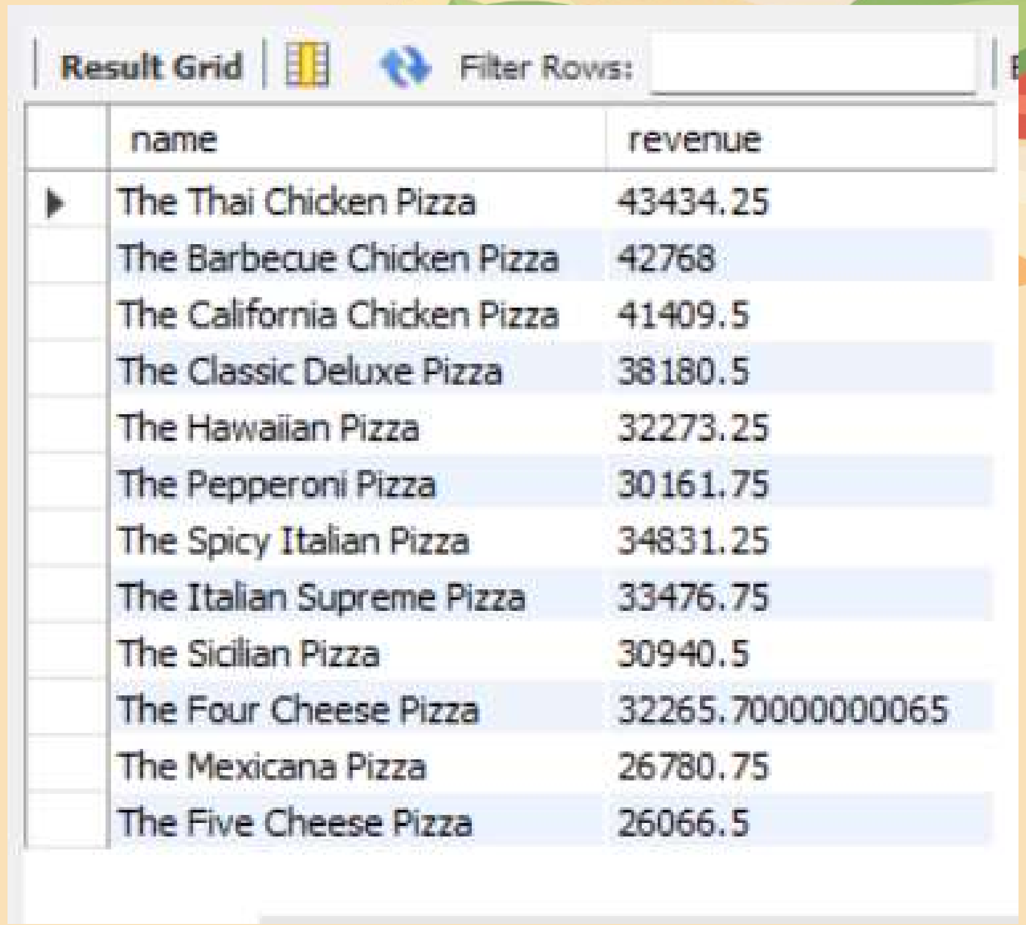
Determine the top 3 most ordered pizza types based on revenue for each pizza category.



```
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;
```

Syntax

Result



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

Thank You