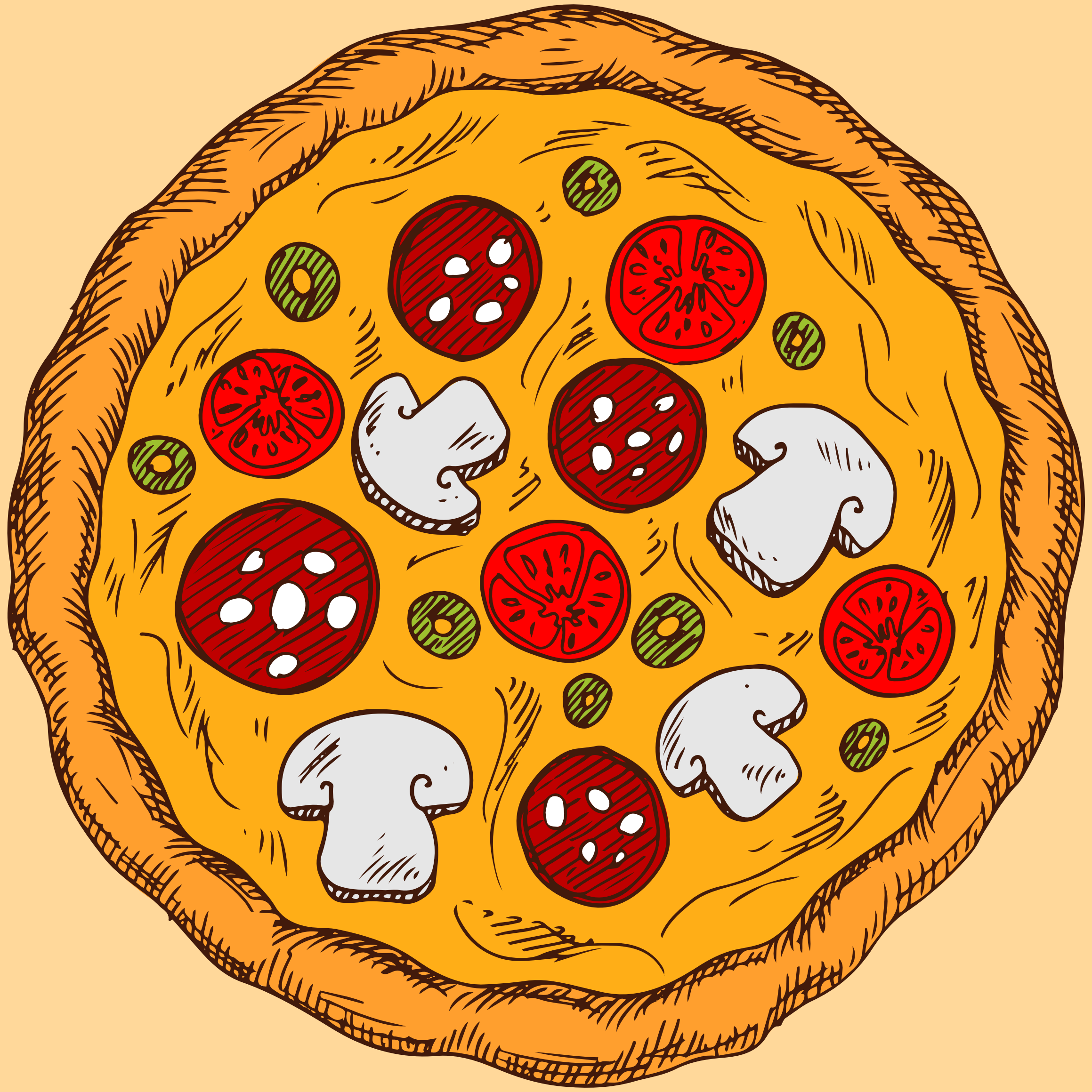
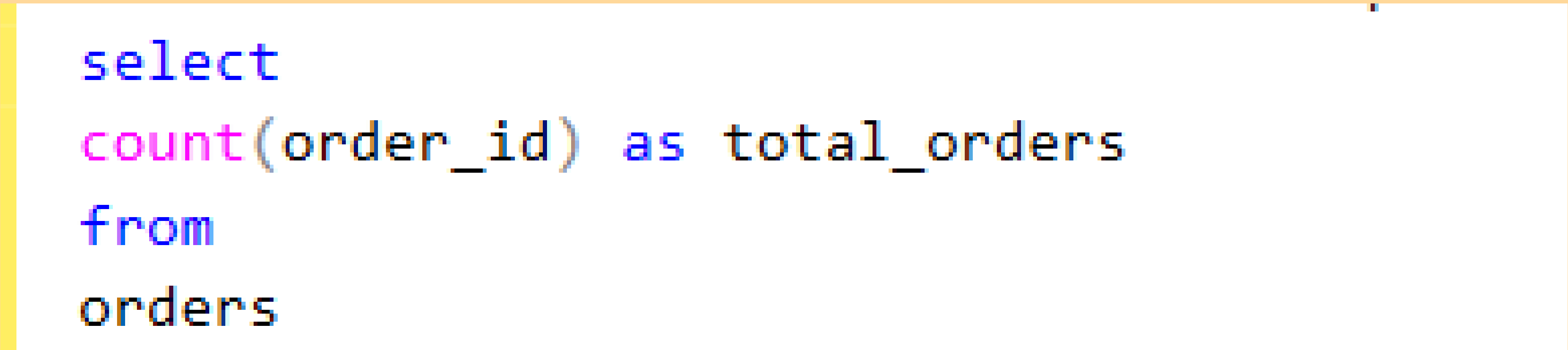


PIZZA SALES PROJECT:SQL

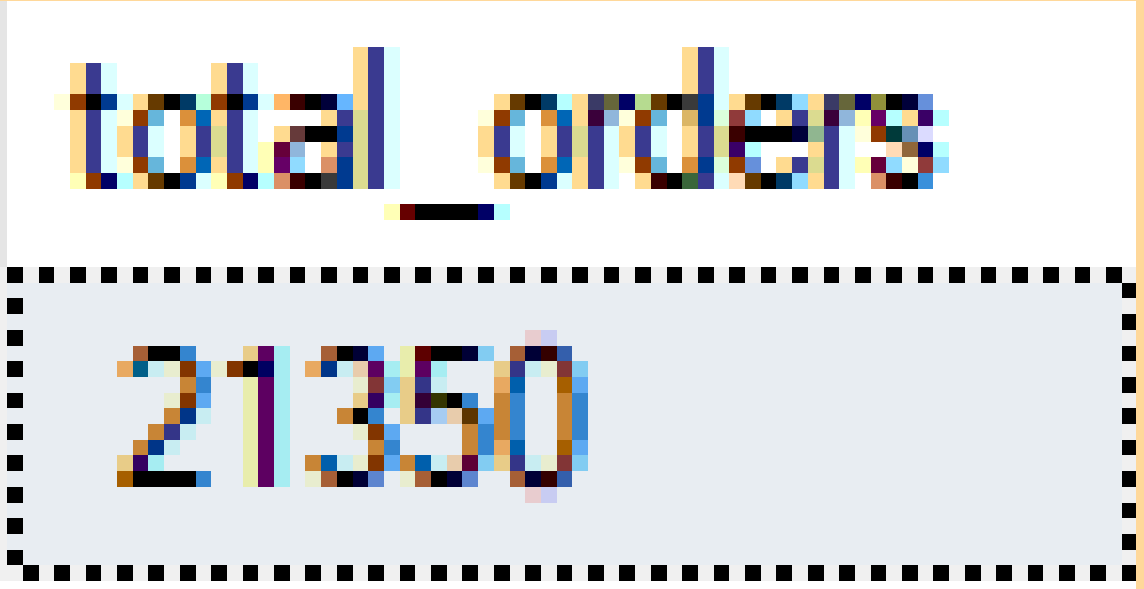
Mohd. Idrees

In this project, I have utilized SQL Queries to solve questions that were related to Pizza Sales.

Retrieve the total number of orders placed

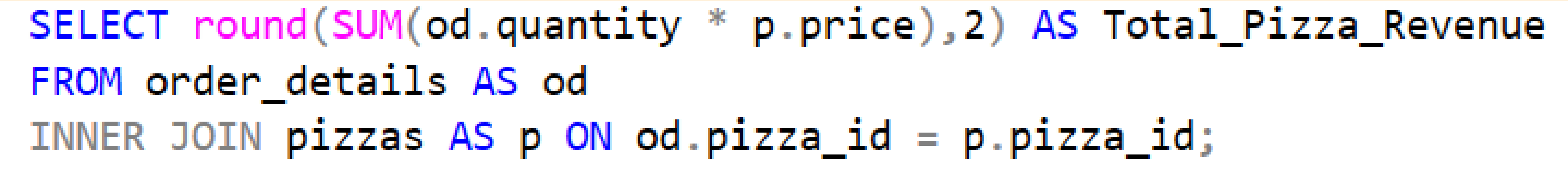


```
select
count(order_id) as total_orders
from
orders
```

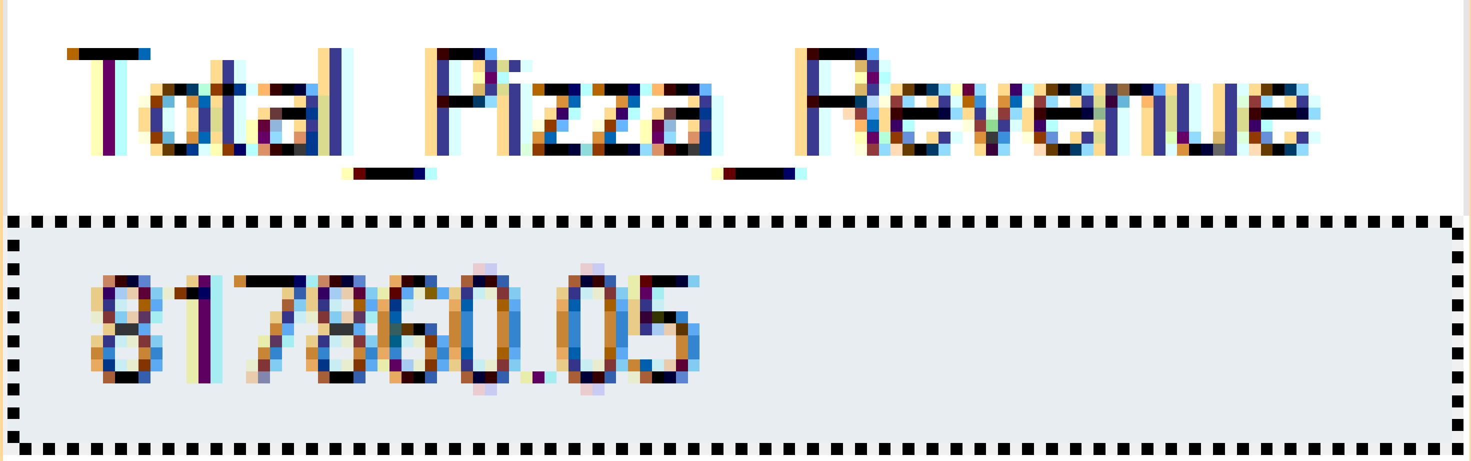


```
total_orders
21350
```

Calculate the total revenue generated from pizza sales

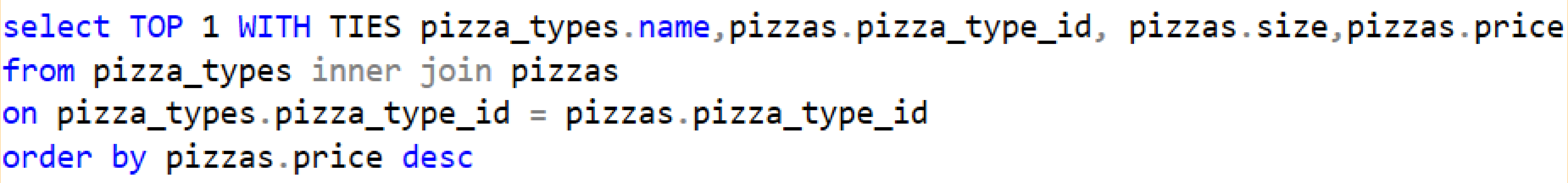


```sql
SELECT round(SUM(od.quantity * p.price),2) AS Total_Pizza_Revenue
FROM order_details AS od
INNER JOIN pizzas AS p ON od.pizza_id = p.pizza_id;
```

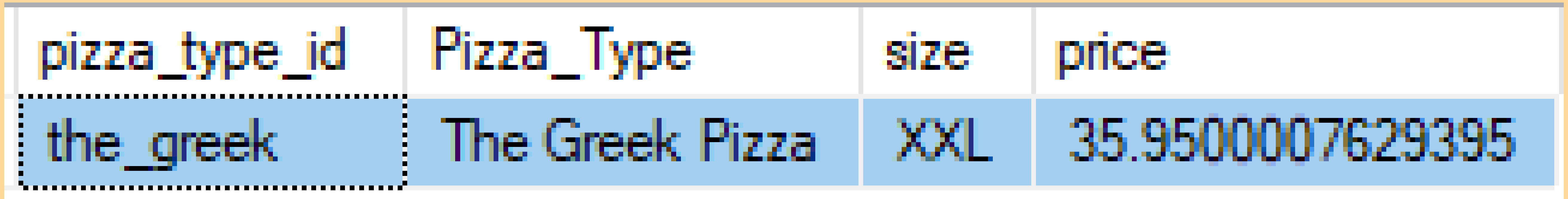


Total_Pizza_Revenue
817860.05

Identify the highest-priced pizza

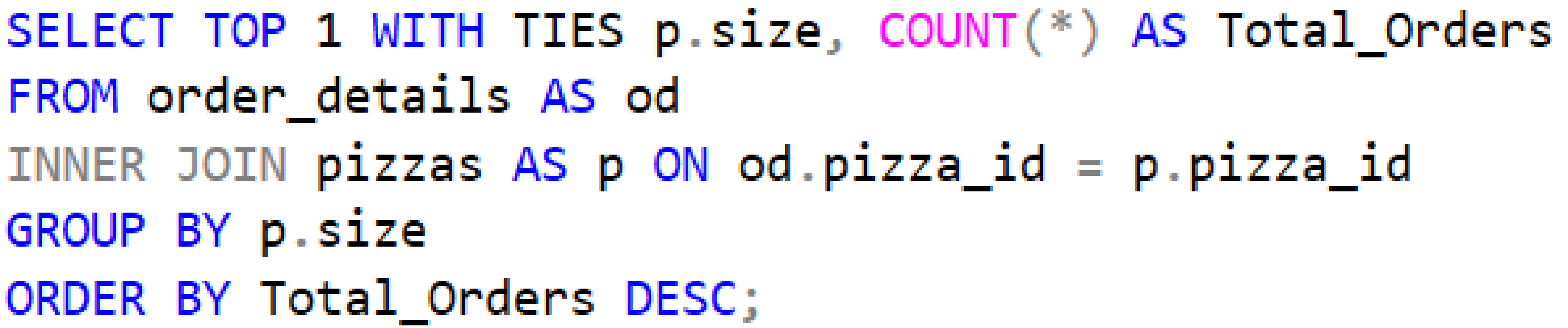


```
select TOP 1 WITH TIES pizza_types.name,pizzas.pizza_type_id, pizzas.size,pizzas.price
from pizza_types inner join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
order by pizzas.price desc
```

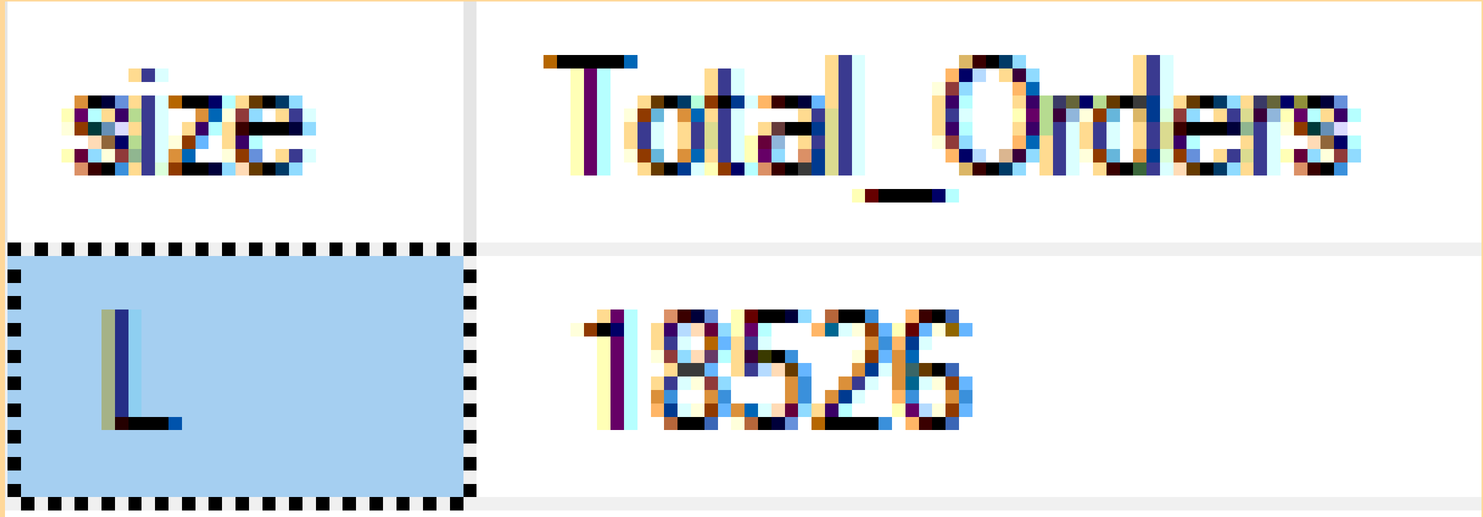


pizza_type_id	Pizza_Type	size	price
the_greek	The Greek Pizza	XXL	35.9500007629395

Identify the most common pizza size ordered

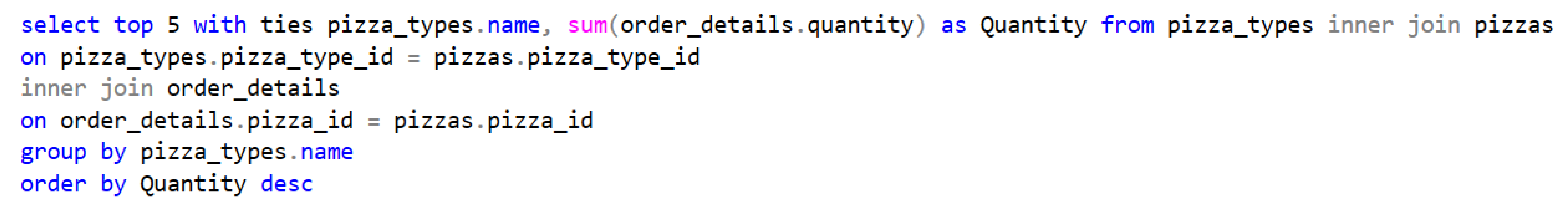


```sql
SELECT TOP 1 WITH TIES p.size, COUNT(*) AS Total_Orders
FROM order_details AS od
INNER JOIN pizzas AS p ON od.pizza_id = p.pizza_id
GROUP BY p.size
ORDER BY Total_Orders DESC;
```

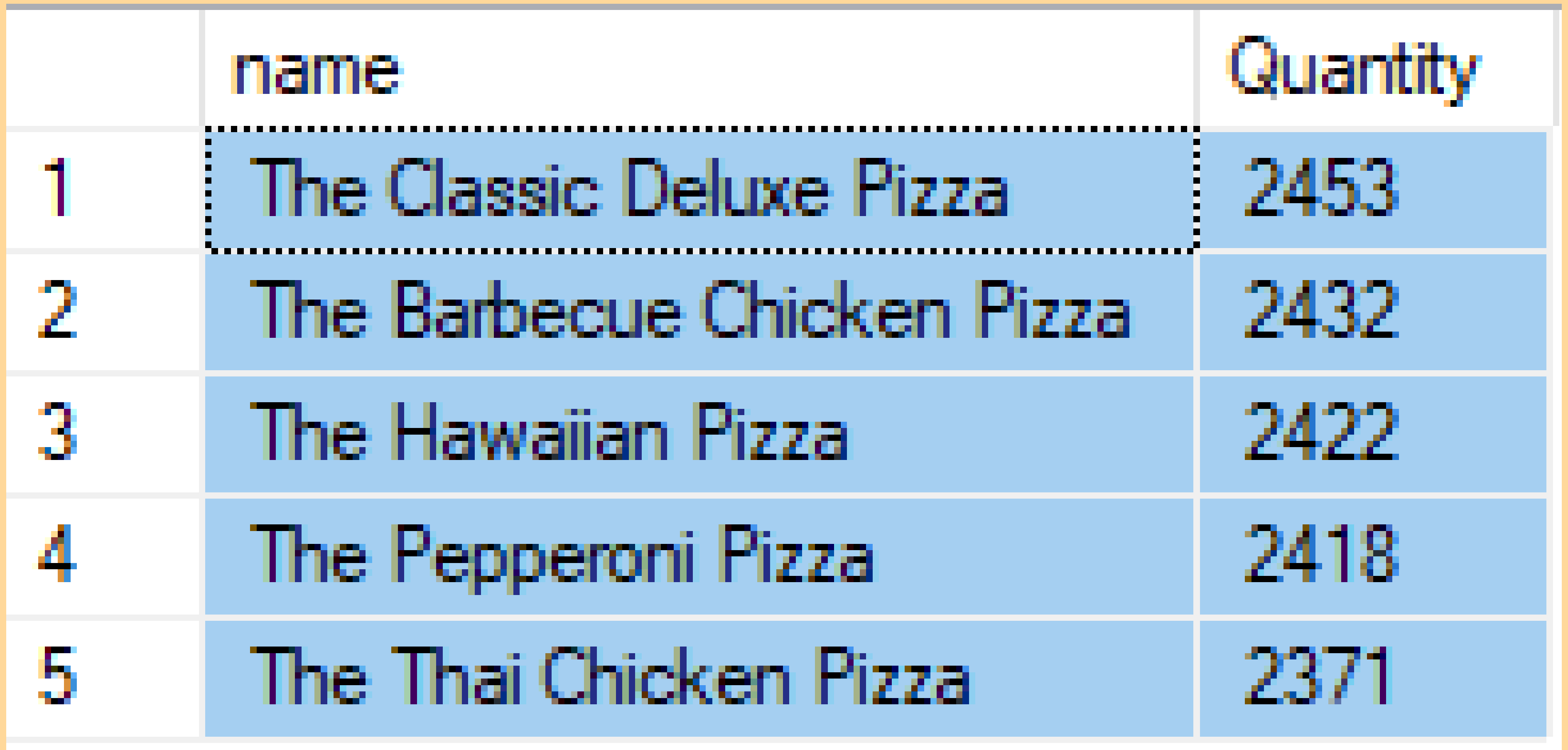


size	Total_Orders
L	18526

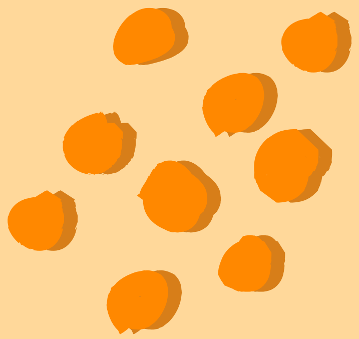
List the top 5 most ordered pizza types along with their quantities



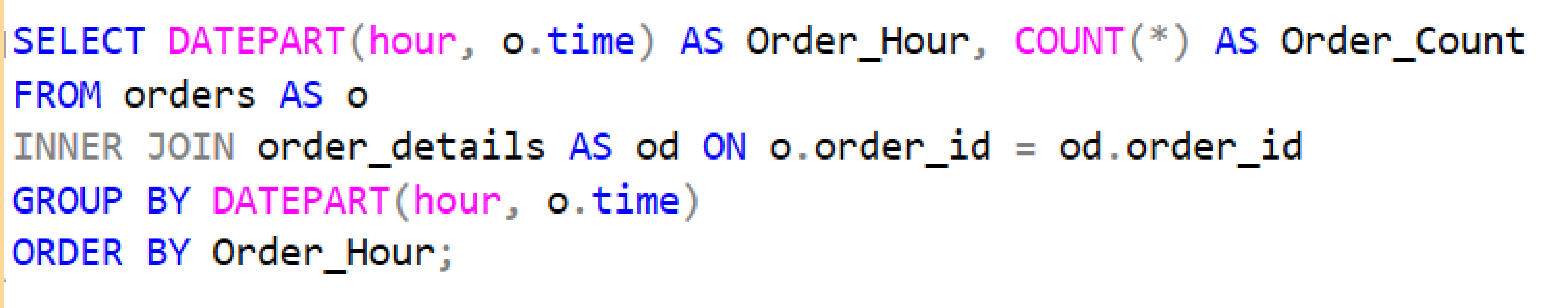
```
select top 5 with ties pizza_types.name, sum(order_details.quantity) as Quantity from pizza_types inner join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
inner join order_details
on order_details.pizza_id = pizzas.pizza_id
group by pizza_types.name
order by Quantity desc
```



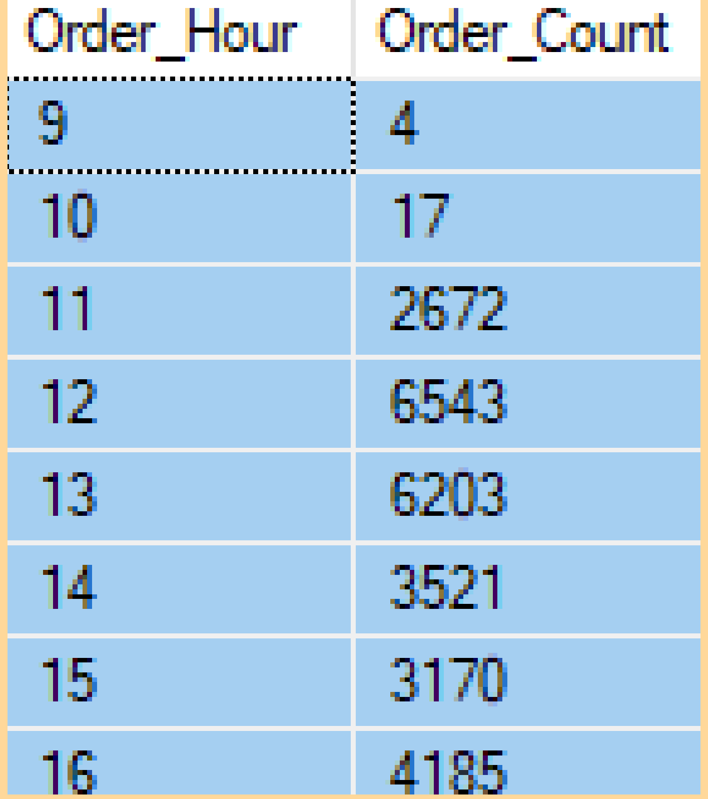
	name	Quantity
1	The Classic Deluxe Pizza	2453
2	The Barbecue Chicken Pizza	2432
3	The Hawaiian Pizza	2422
4	The Pepperoni Pizza	2418
5	The Thai Chicken Pizza	2371



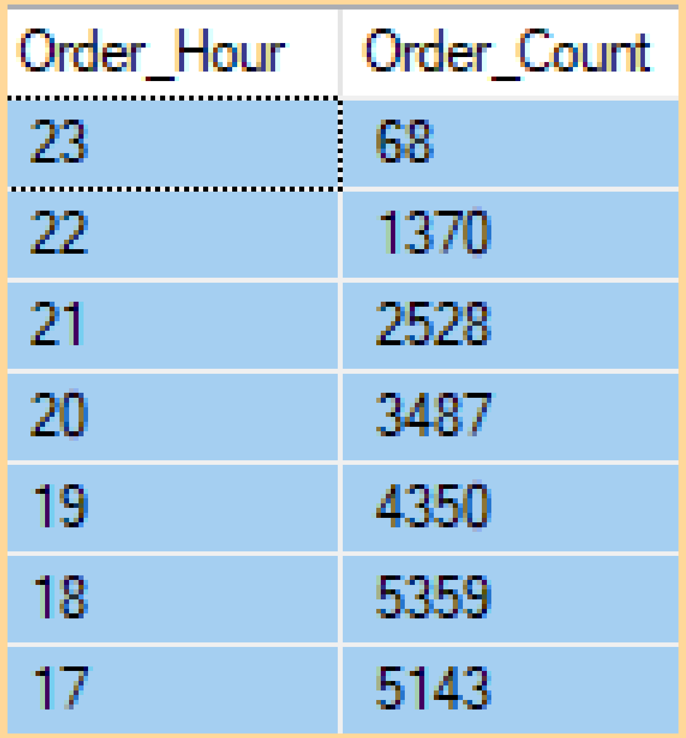
Determine the distribution of orders by hour of the day



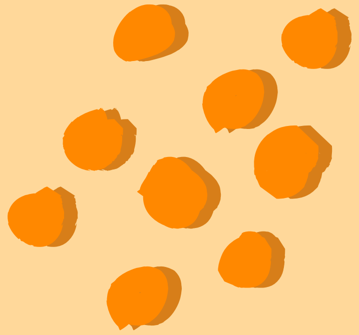
```
SELECT DATEPART(hour, o.time) AS Order_Hour, COUNT(*) AS Order_Count
FROM orders AS o
INNER JOIN order_details AS od ON o.order_id = od.order_id
GROUP BY DATEPART(hour, o.time)
ORDER BY Order_Hour;
```



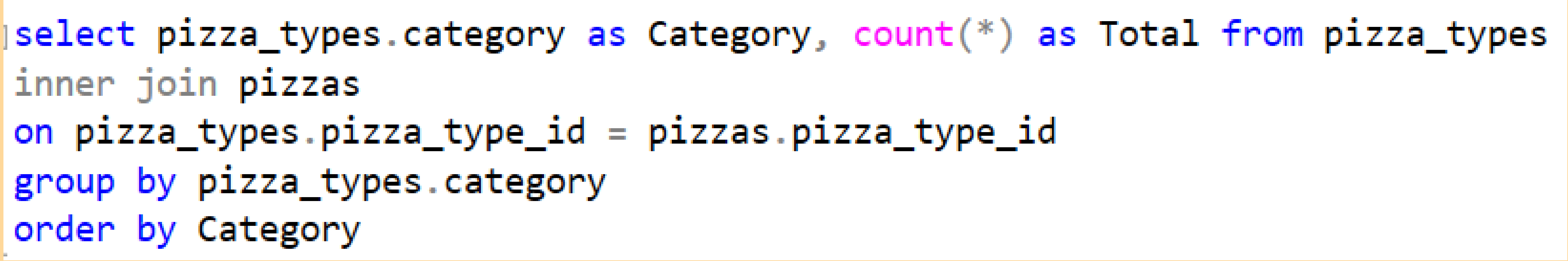
Order_Hour	Order_Count
9	4
10	17
11	2672
12	6543
13	6203
14	3521
15	3170
16	4185



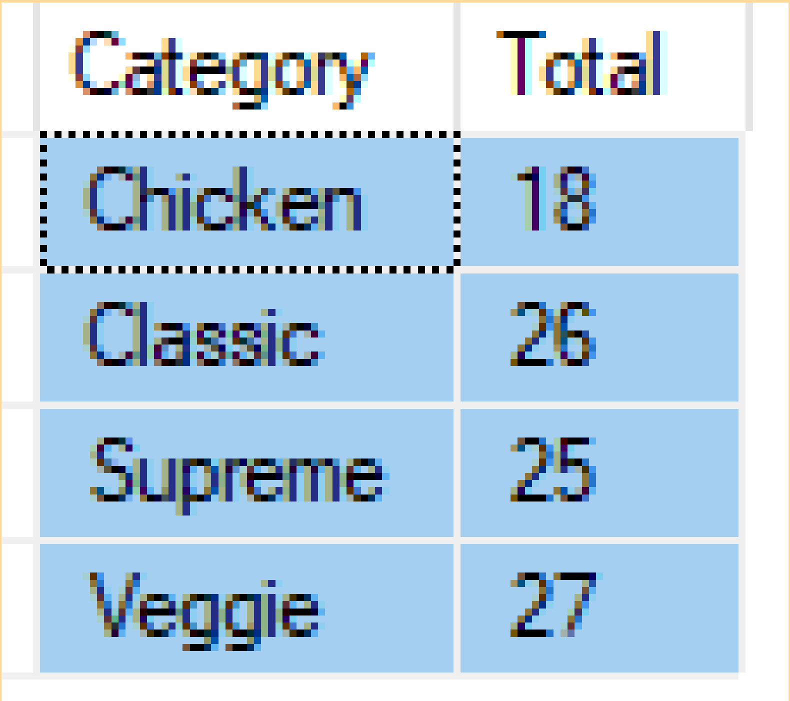
Order_Hour	Order_Count
23	68
22	1370
21	2528
20	3487
19	4350
18	5359
17	5143



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

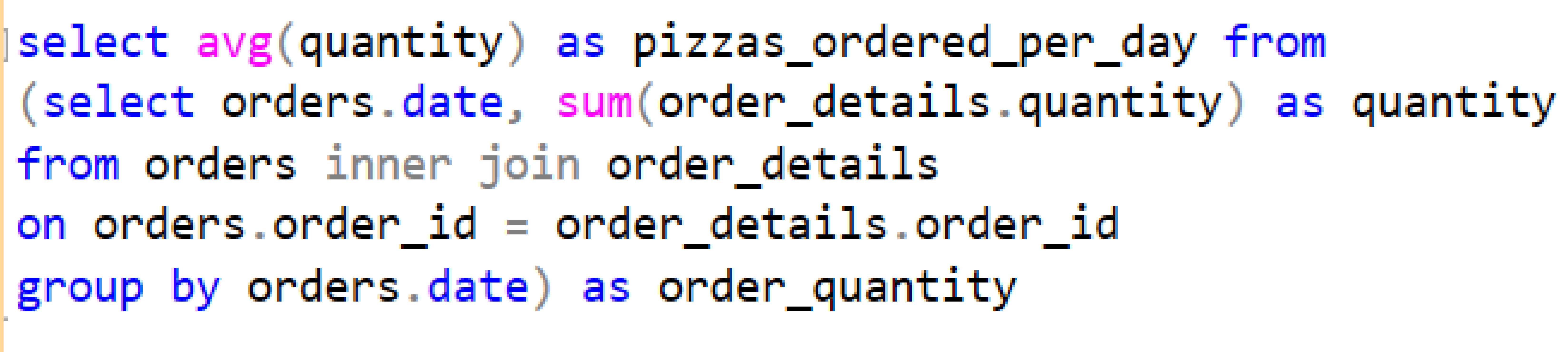


```
select pizza_types.category as Category, count(*) as Total from pizza_types
inner join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
group by pizza_types.category
order by Category
```

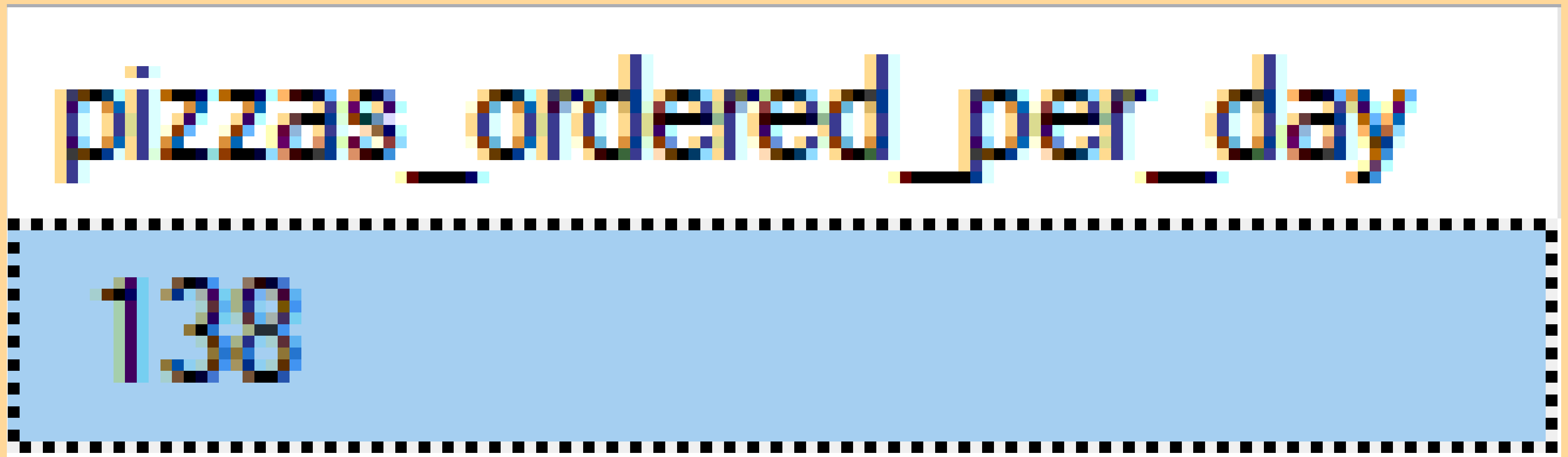


Category	Total
Chicken	18
Classic	26
Supreme	25
Veggie	27

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

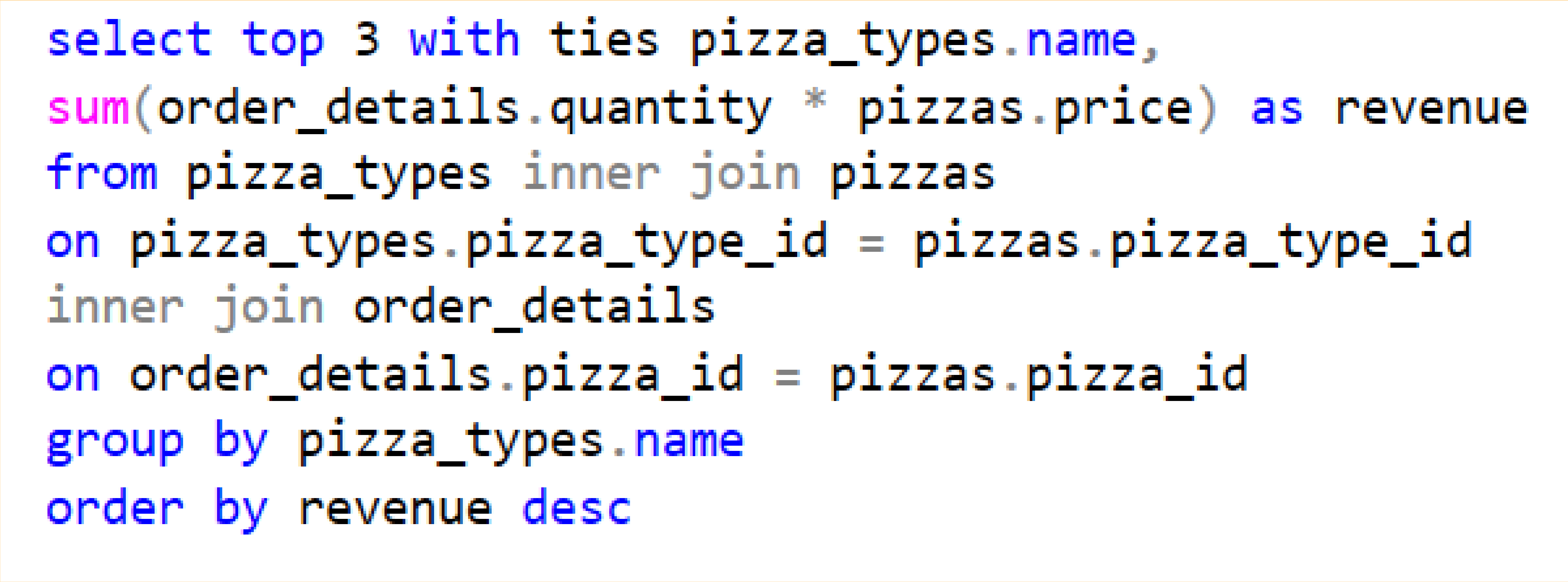


```
select avg(quantity) as pizzas_ordered_per_day from
(select orders.date, sum(order_details.quantity) as quantity
from orders inner join order_details
on orders.order_id = order_details.order_id
group by orders.date) as order_quantity
```

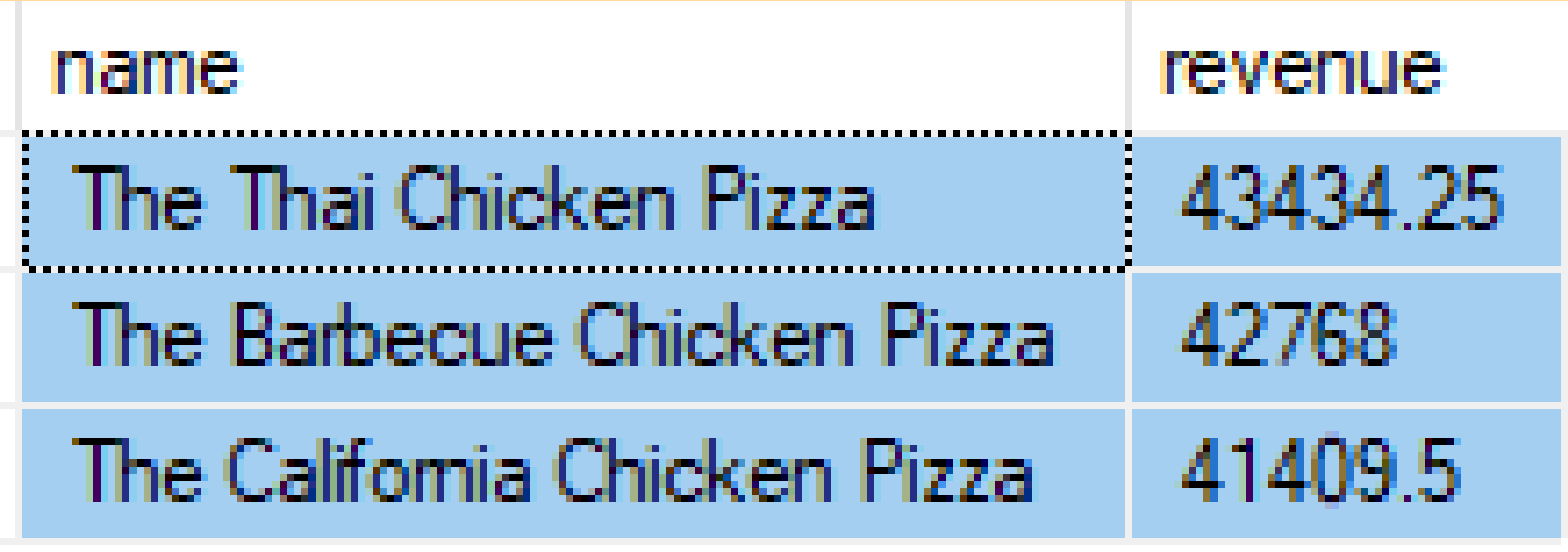


```
pizzas_ordered_per_day
138
```

Determine the top 3 most ordered pizza types based on revenue

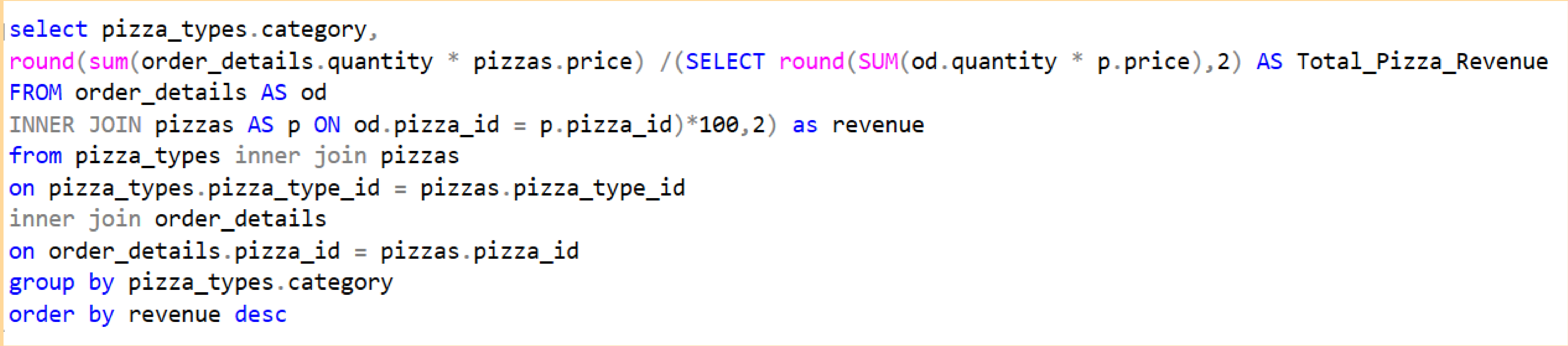


```sql
select top 3 with ties pizza_types.name,
sum(order_details.quantity * pizzas.price) as revenue
from pizza_types inner join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
inner join order_details
on order_details.pizza_id = pizzas.pizza_id
group by pizza_types.name
order by revenue desc
```

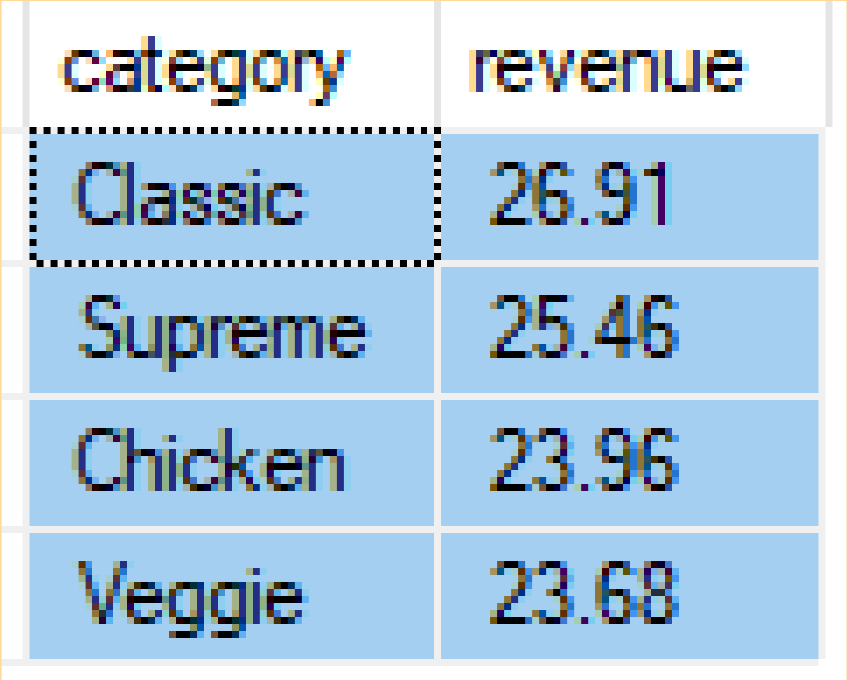


name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

Calculate the percentage contribution of each pizza type to total revenue

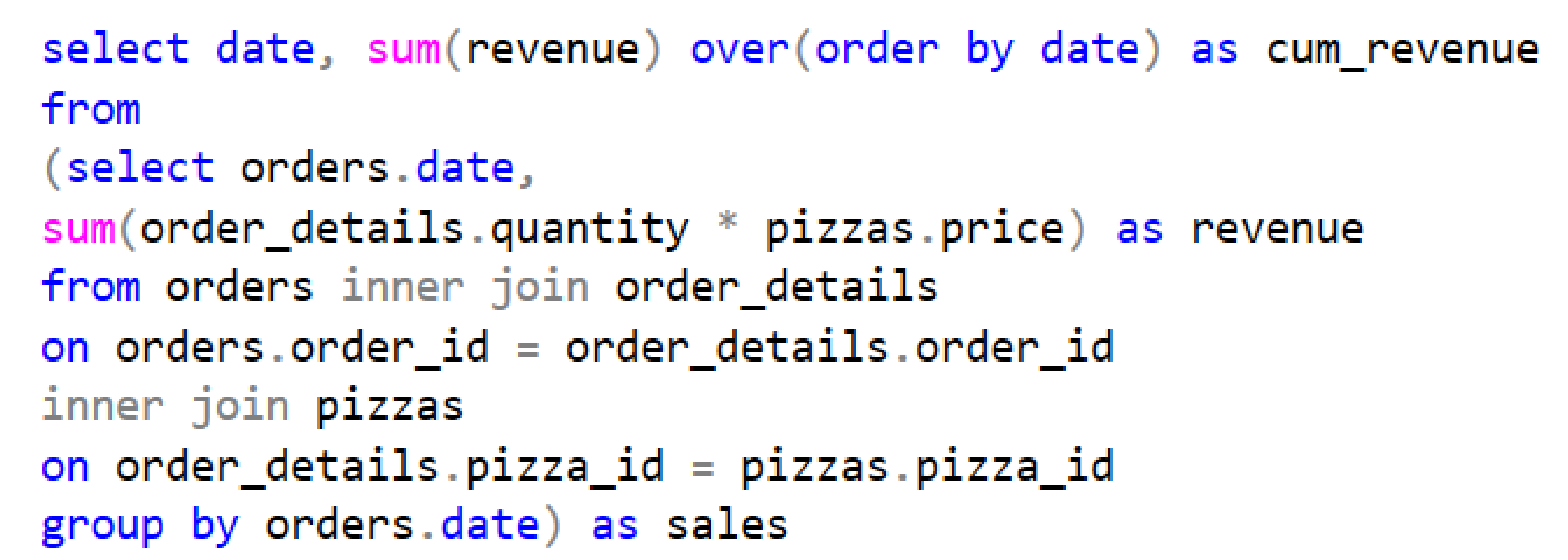


```
select pizza_types.category,
round(sum(order_details.quantity * pizzas.price) /(SELECT round(SUM(od.quantity * p.price),2) AS Total_Pizza_Revenue
FROM order_details AS od
INNER JOIN pizzas AS p ON od.pizza_id = p.pizza_id)*100,2) as revenue
from pizza_types inner join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
inner join order_details
on order_details.pizza_id = pizzas.pizza_id
group by pizza_types.category
order by revenue desc
```

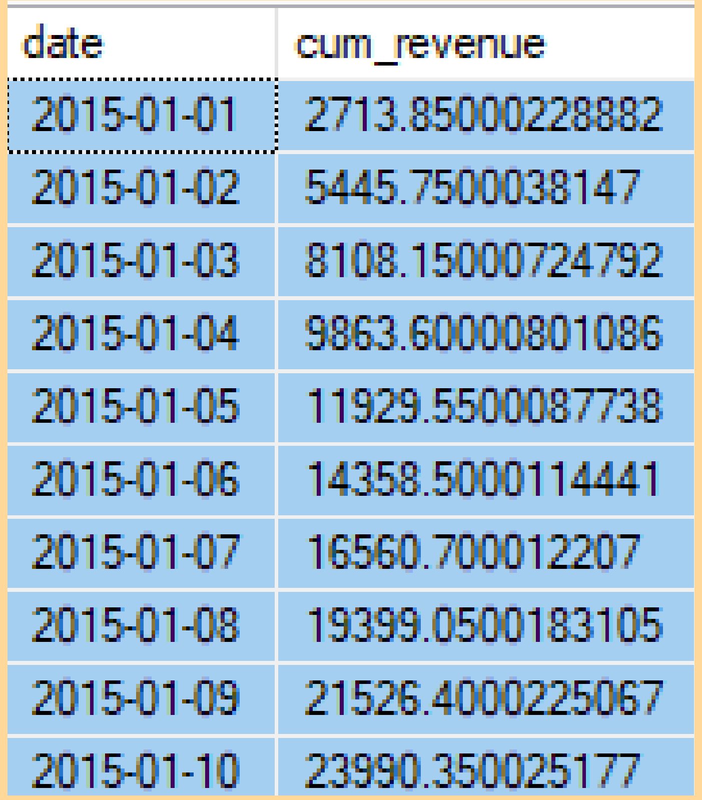


category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Analyze the cumulative revenue generated over time

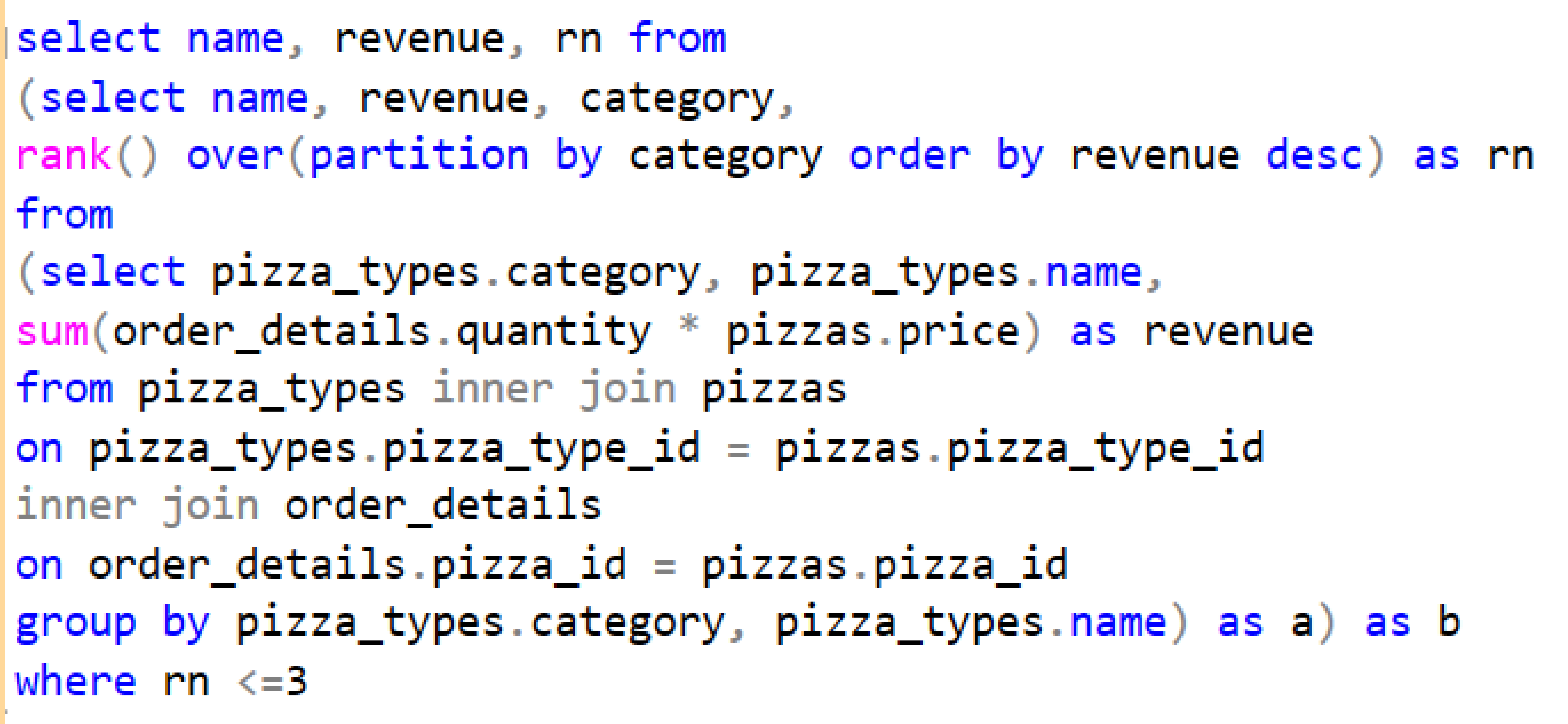


```sql
select date, sum(revenue) over(order by date) as cum_revenue
from
(select orders.date,
sum(order_details.quantity * pizzas.price) as revenue
from orders inner join order_details
on orders.order_id = order_details.order_id
inner join pizzas
on order_details.pizza_id = pizzas.pizza_id
group by orders.date)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

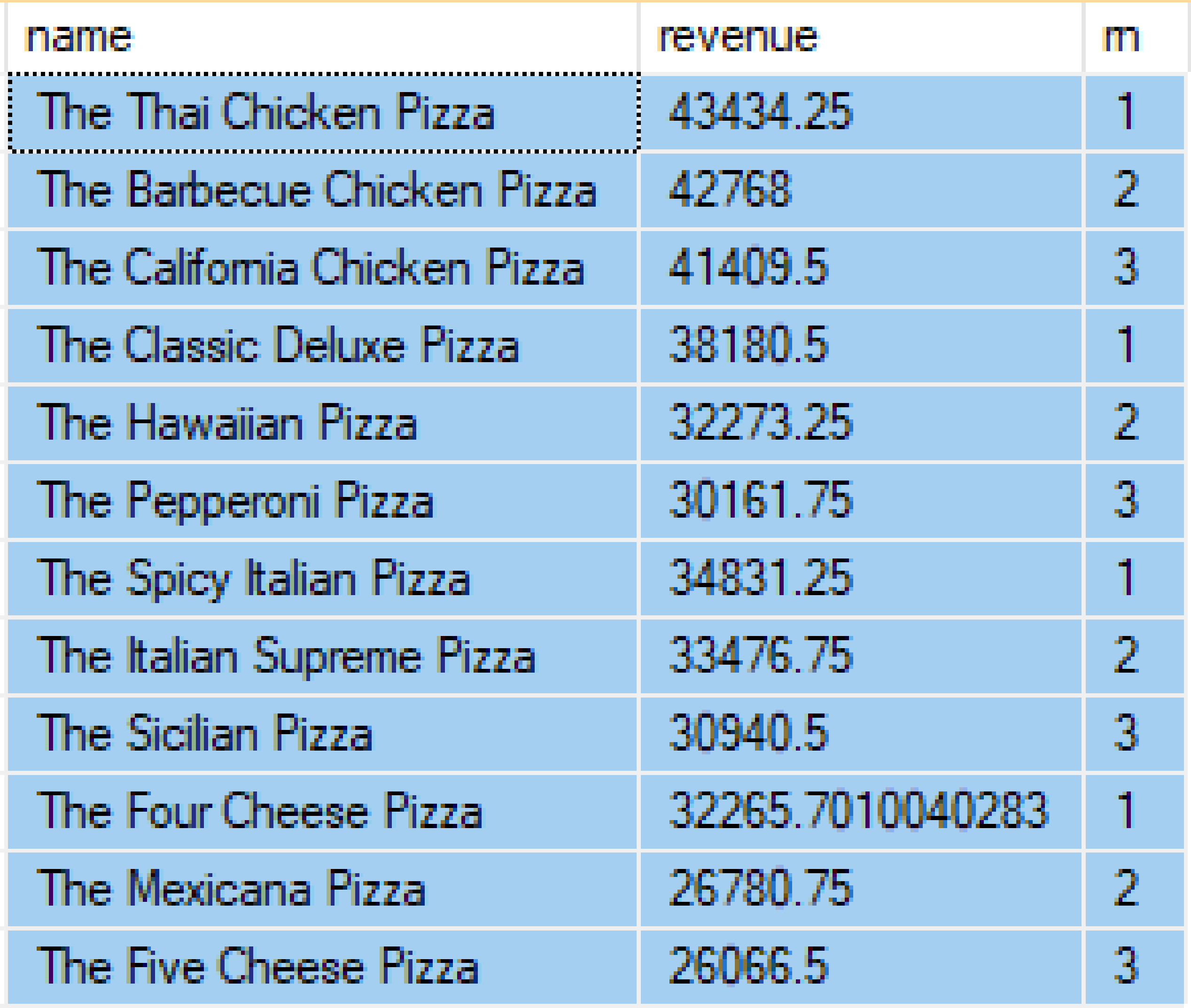


date	cum_revenue
2015-01-01	2713.85000228882
2015-01-02	5445.7500038147
2015-01-03	8108.15000724792
2015-01-04	9863.60000801086
2015-01-05	11929.5500087738
2015-01-06	14358.5000114441
2015-01-07	16560.700012207
2015-01-08	19399.0500183105
2015-01-09	21526.4000225067
2015-01-10	23990.350025177

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



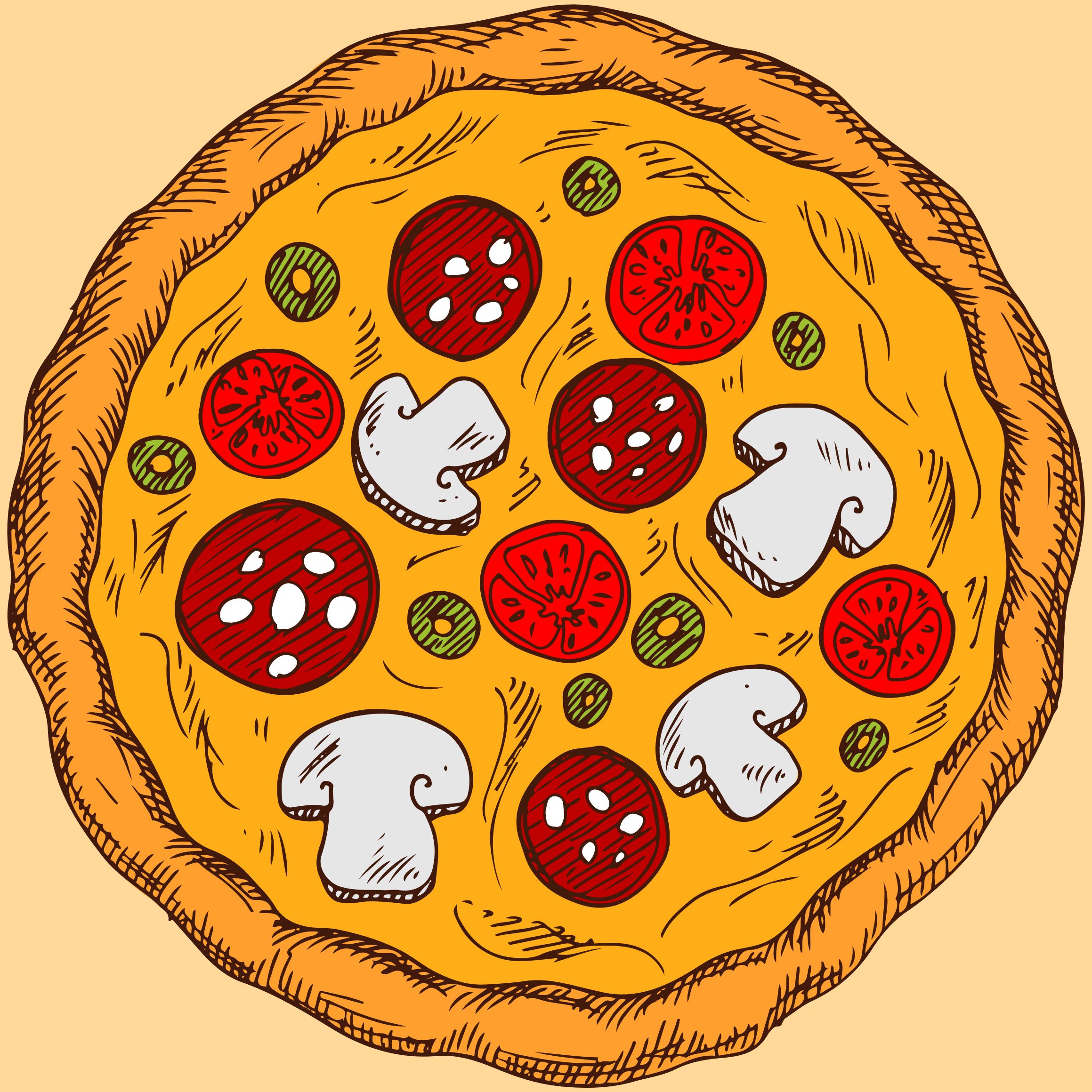
```
select name, revenue, rn from
(select name, revenue, category,
rank() over(partition by category order by revenue desc) as rn
from
(select pizza_types.category, pizza_types.name,
sum(order_details.quantity * pizzas.price) as revenue
from pizza_types inner join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
inner 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	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	32265.7010040283	1
The Mexicana Pizza	26780.75	2
The Five Cheese Pizza	26066.5	3

Where we are?

we have answered 13 queries that will help us make decisions and drive the business ahead.



THANK YOU