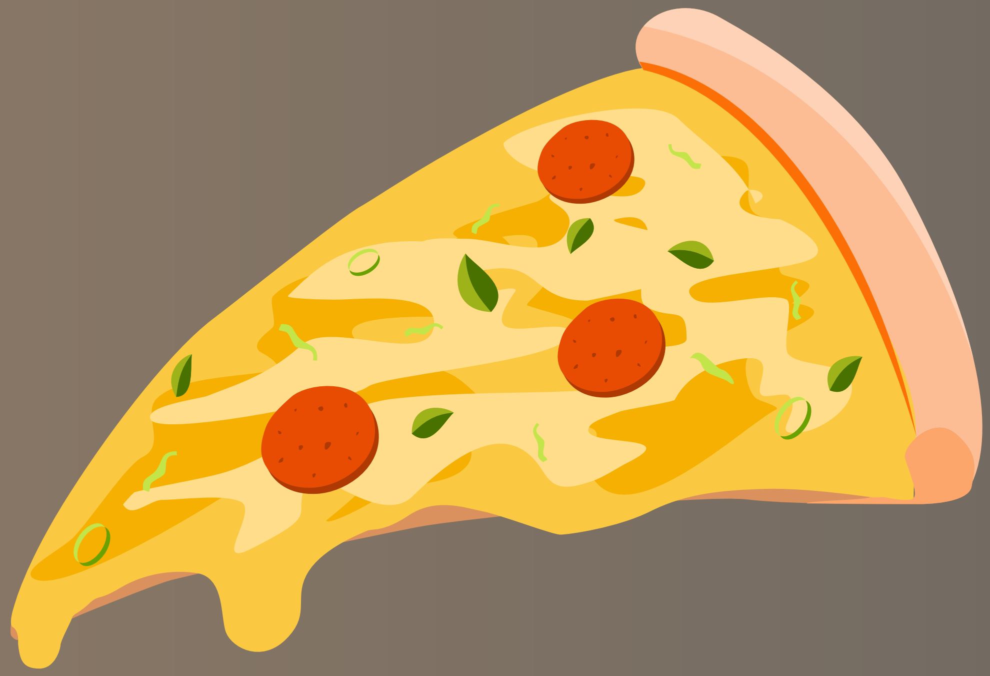
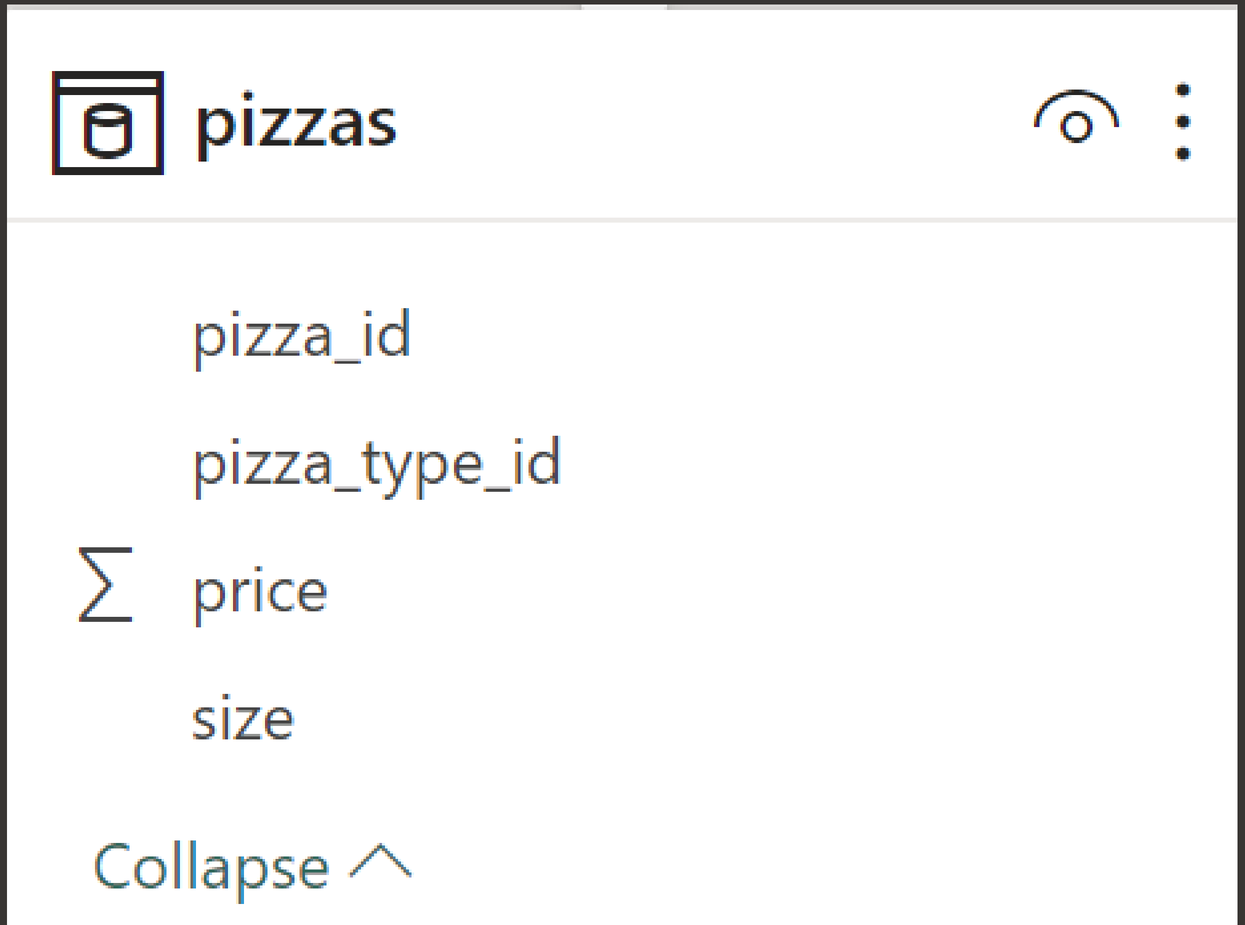
PIZZA MASTER SALES

# Delicious Pizza Sales



# HELLO!

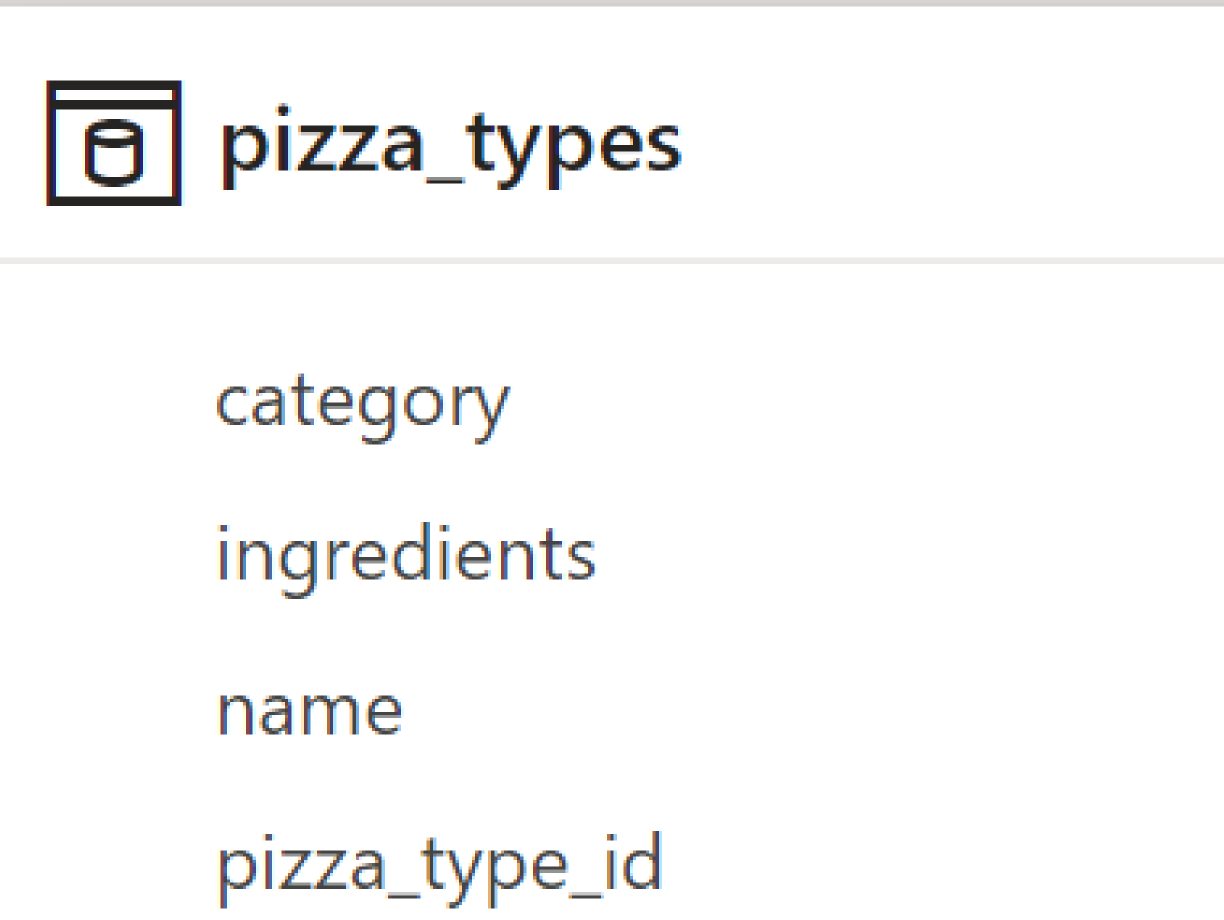
Hello my name is Yashashri
In this project I have utilized SQL Queries to solve questions that we related to pizza sales



**pizzas**

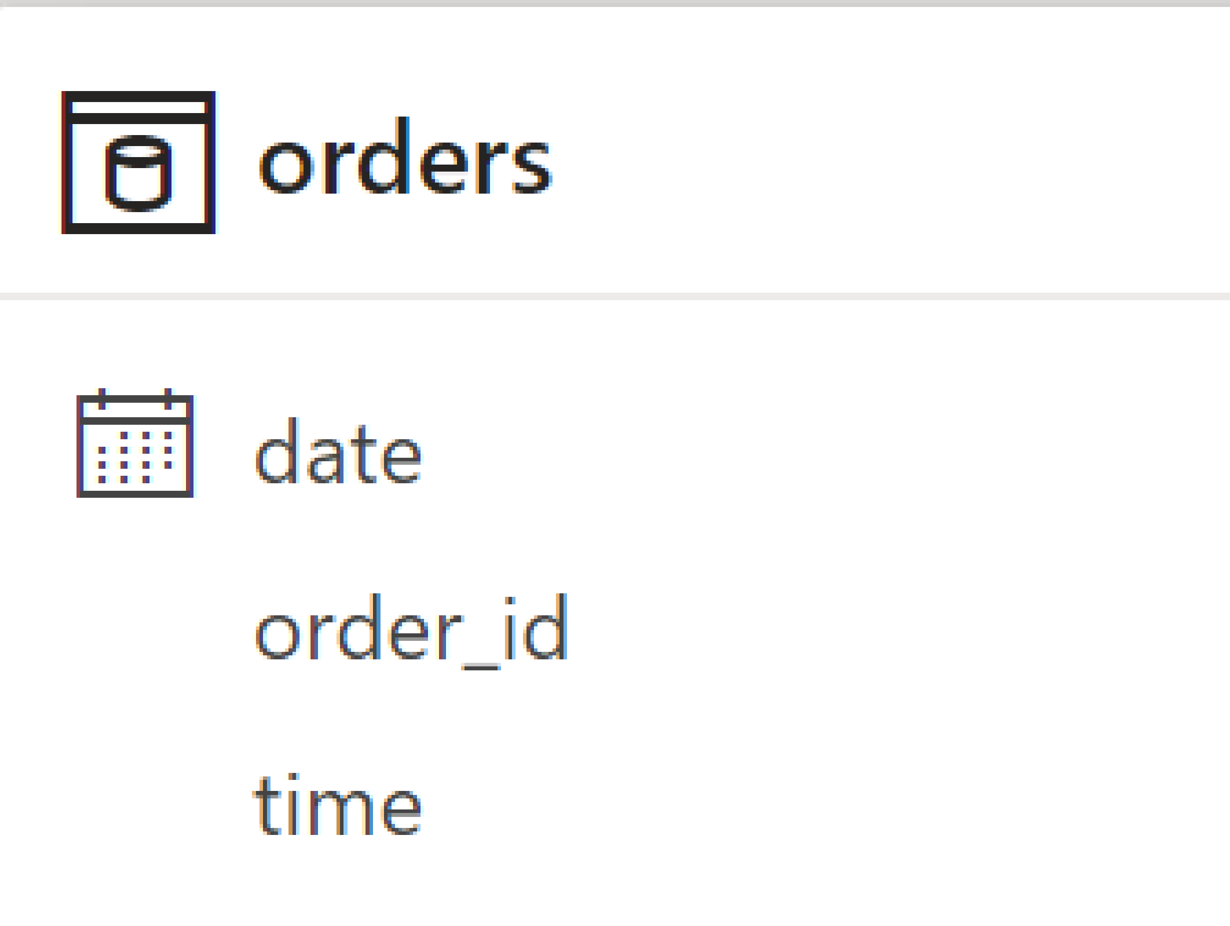
- pizza_id
- pizza_type_id
- Σ price
- size

Collapse ^



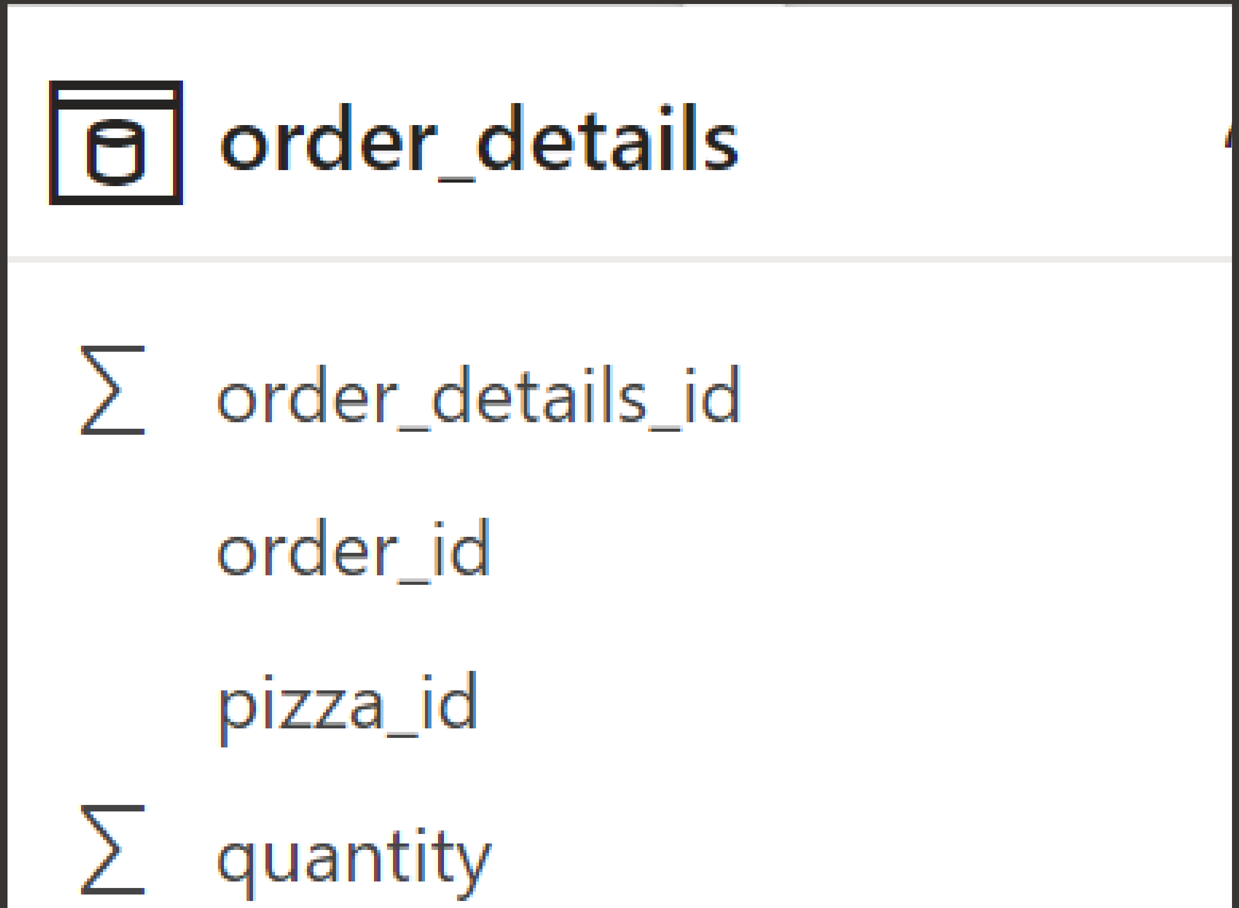
**pizza_types**

- category
- ingredients
- name
- pizza_type_id



**orders**

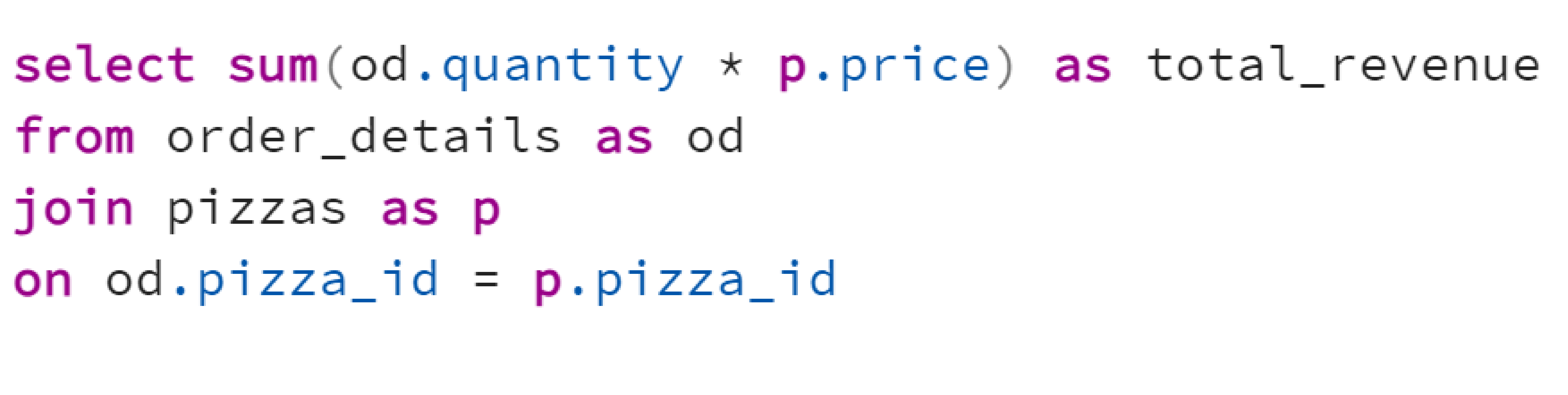
- date
- order_id
- time



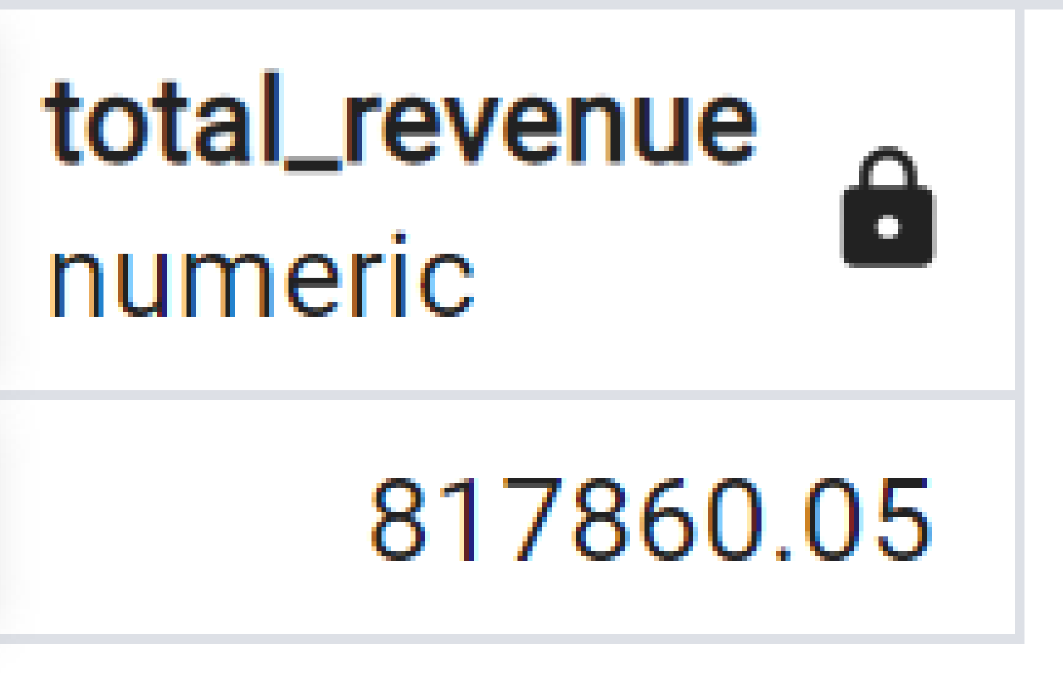
**order_details**

- Σ order_details_id
- order_id
- pizza_id
- Σ quantity

# Calculate the total revenue generated from pizza sales.

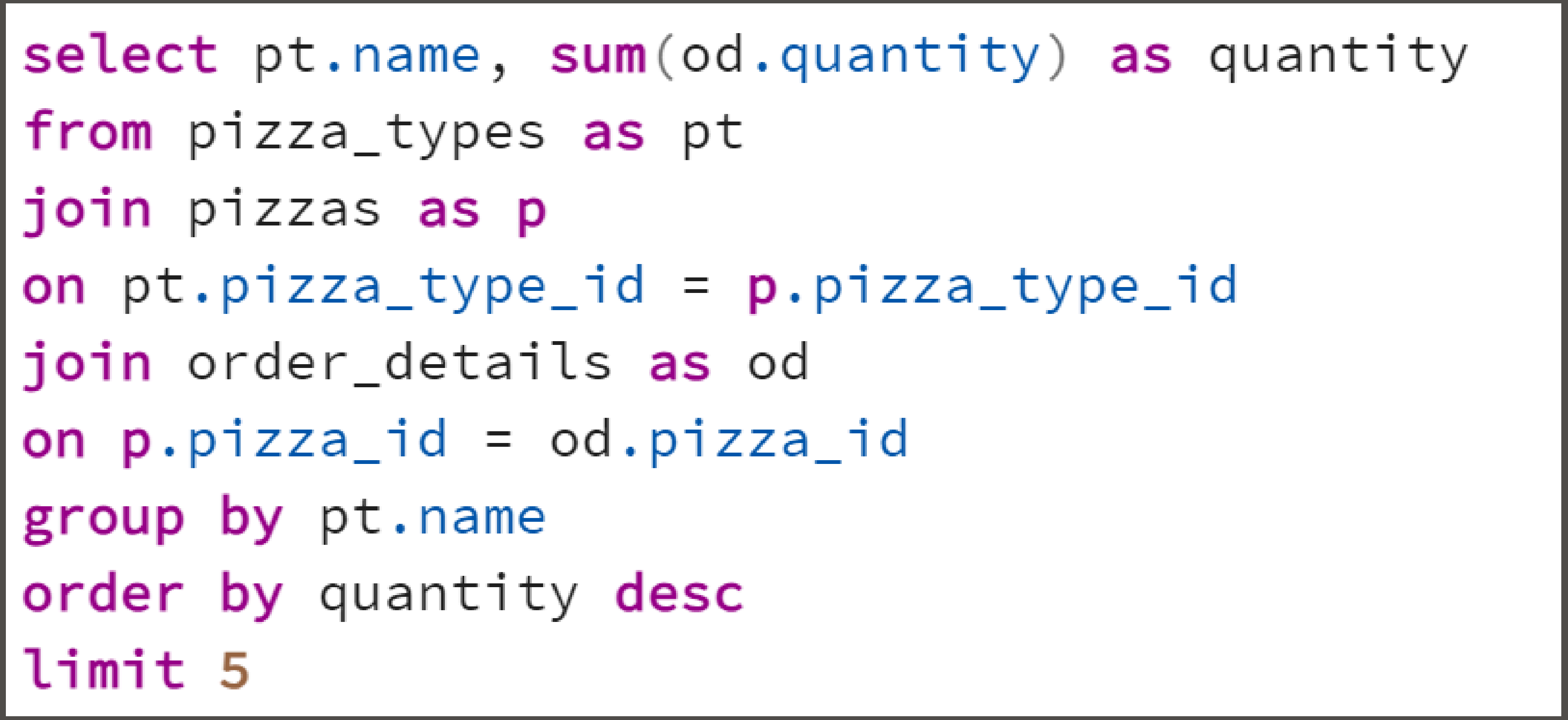


```
select sum(od.quantity * p.price) as total_revenue
from order_details as od
join pizzas as p
on od.pizza_id = p.pizza_id
```

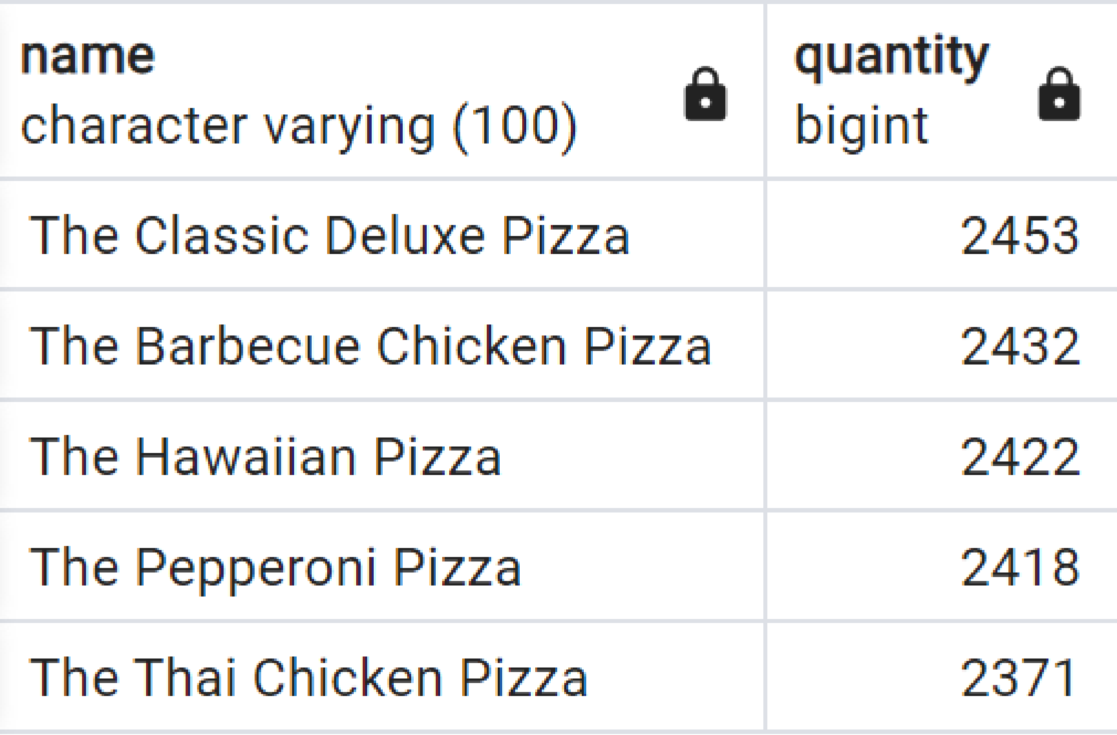


| total_revenue<br>numeric |
|---|
| 817860.05 |

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

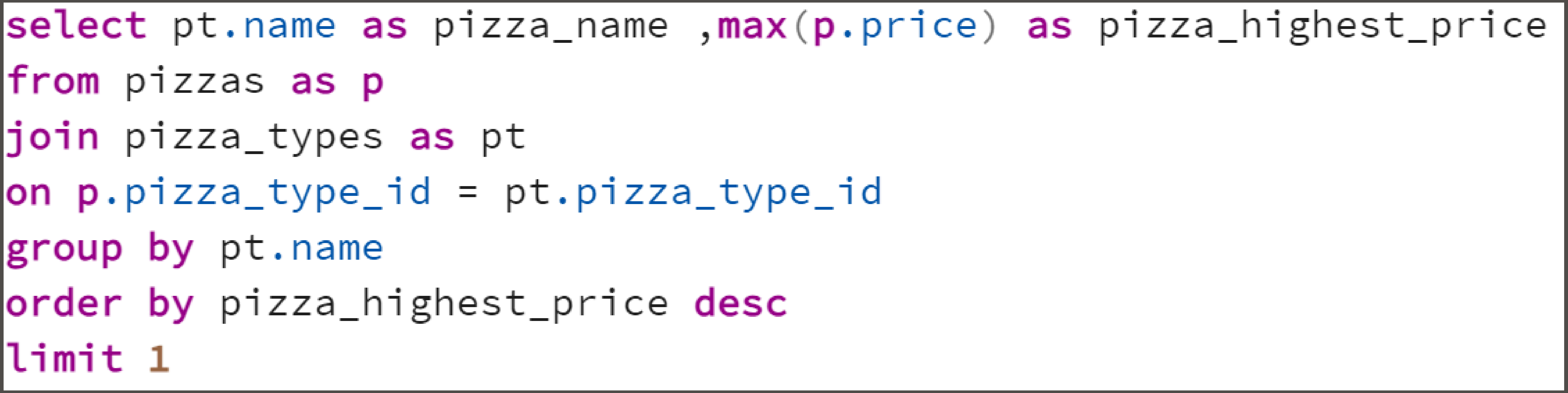


```
select pt.name, sum(od.quantity) as quantity
from pizza_types as pt
join pizzas as p
on pt.pizza_type_id = p.pizza_type_id
join order_details as od
on p.pizza_id = od.pizza_id
group by pt.name
order by quantity desc
limit 5
```

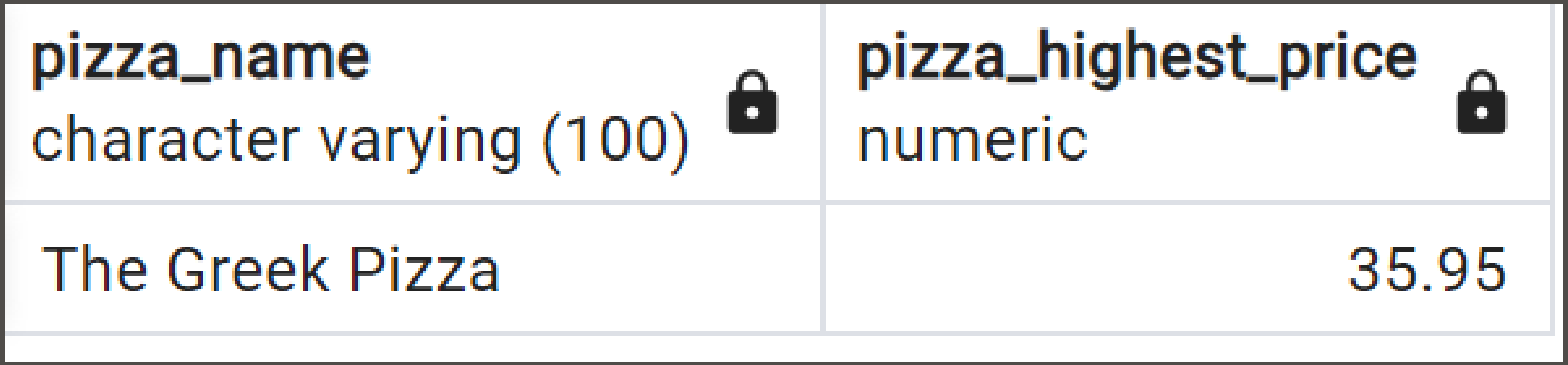


| name<br>character varying (100) | quantity<br>bigint |
|---|---|
| The Classic Deluxe Pizza | 2453 |
| The Barbecue Chicken Pizza | 2432 |
| The Hawaiian Pizza | 2422 |
| The Pepperoni Pizza | 2418 |
| The Thai Chicken Pizza | 2371 |

# Identify the highest-priced pizza.

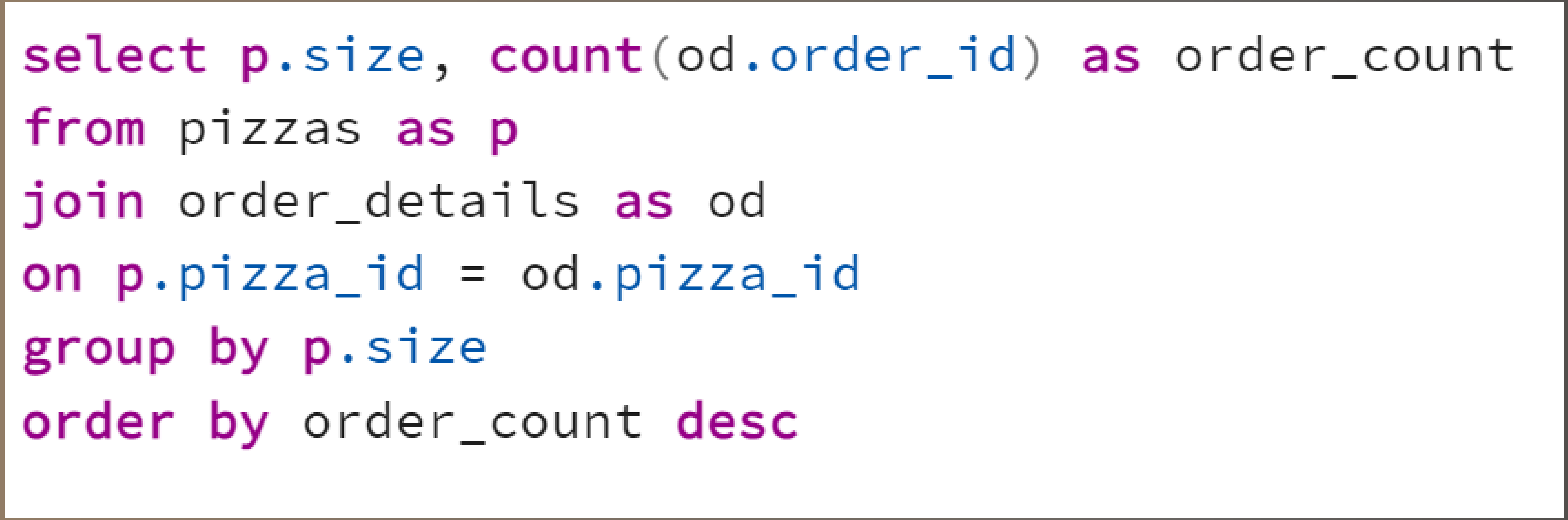


```sql
select pt.name as pizza_name ,max(p.price) as pizza_highest_price
from pizzas as p
join pizza_types as pt
on p.pizza_type_id = pt.pizza_type_id
group by pt.name
order by pizza_highest_price desc
limit 1
```

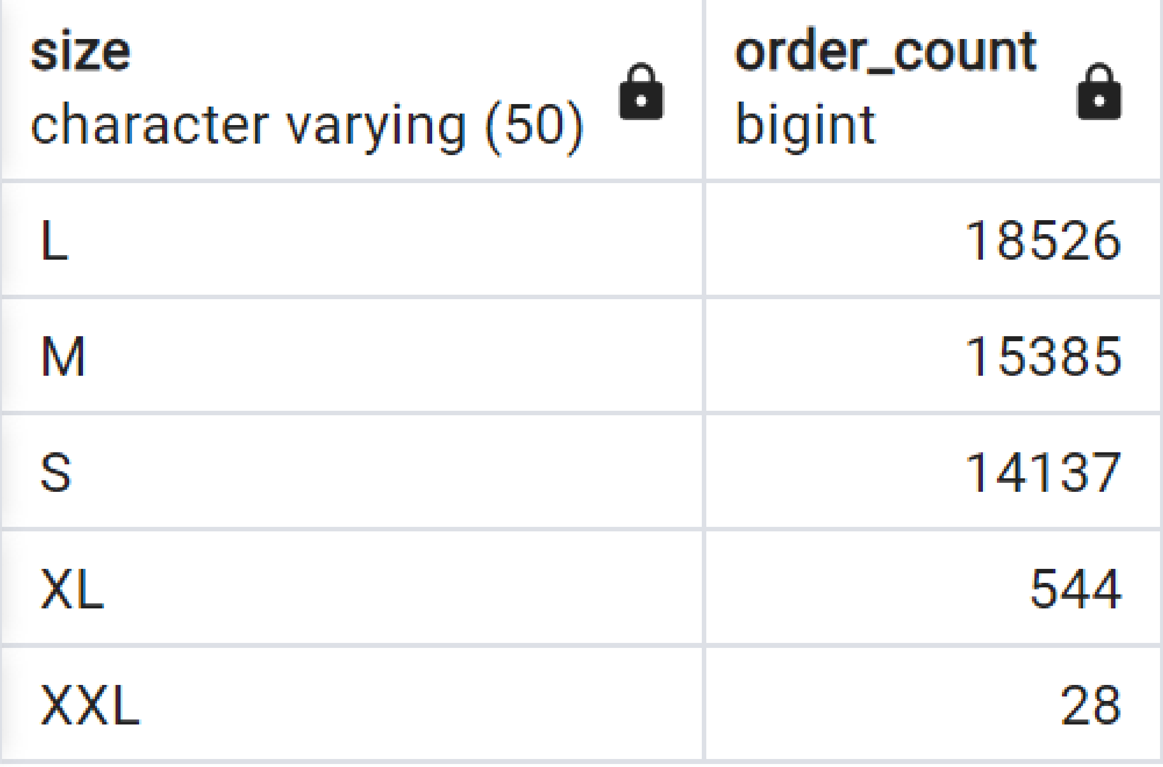


| pizza_name | pizza_highest_price |
|---|---|
| character varying (100) | numeric |
| The Greek Pizza | 35.95 |

# Identify the most common pizza size ordered.

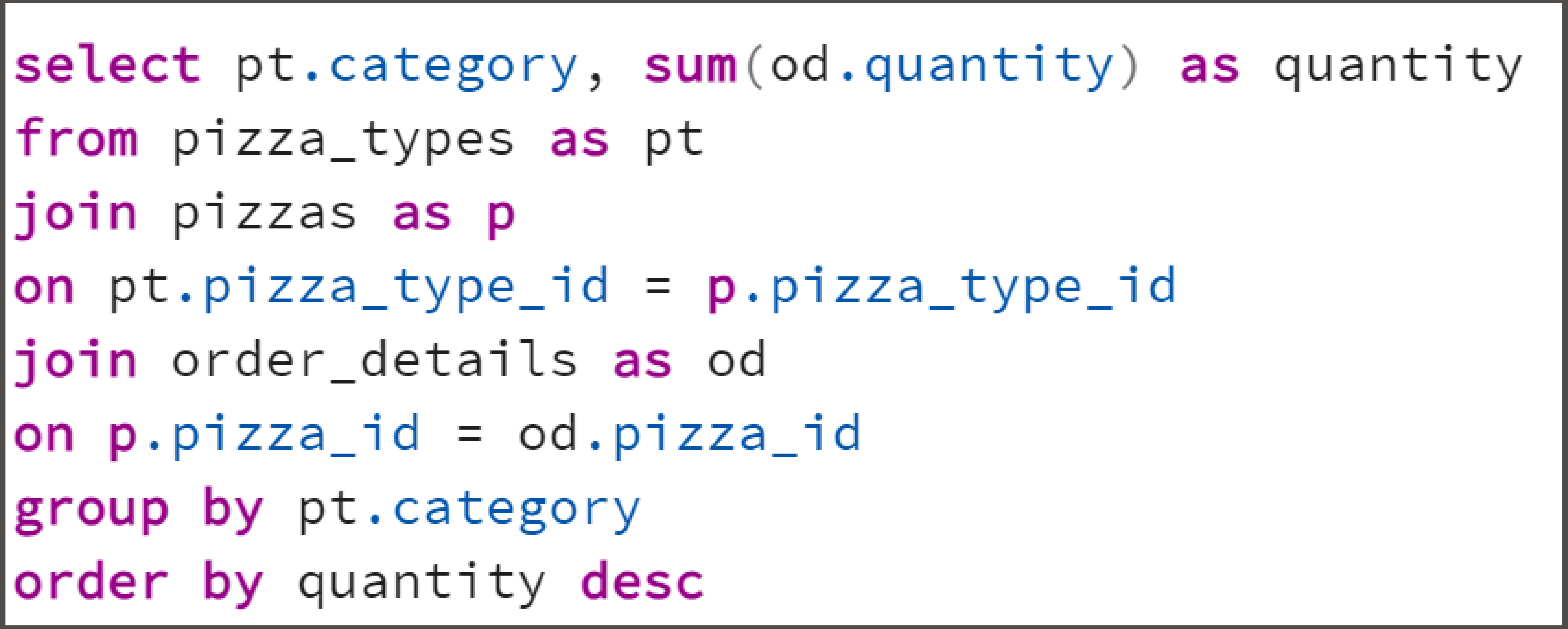


```
select p.size, count(od.order_id) as order_count
from pizzas as p
join order_details as od
on p.pizza_id = od.pizza_id
group by p.size
order by order_count desc
```

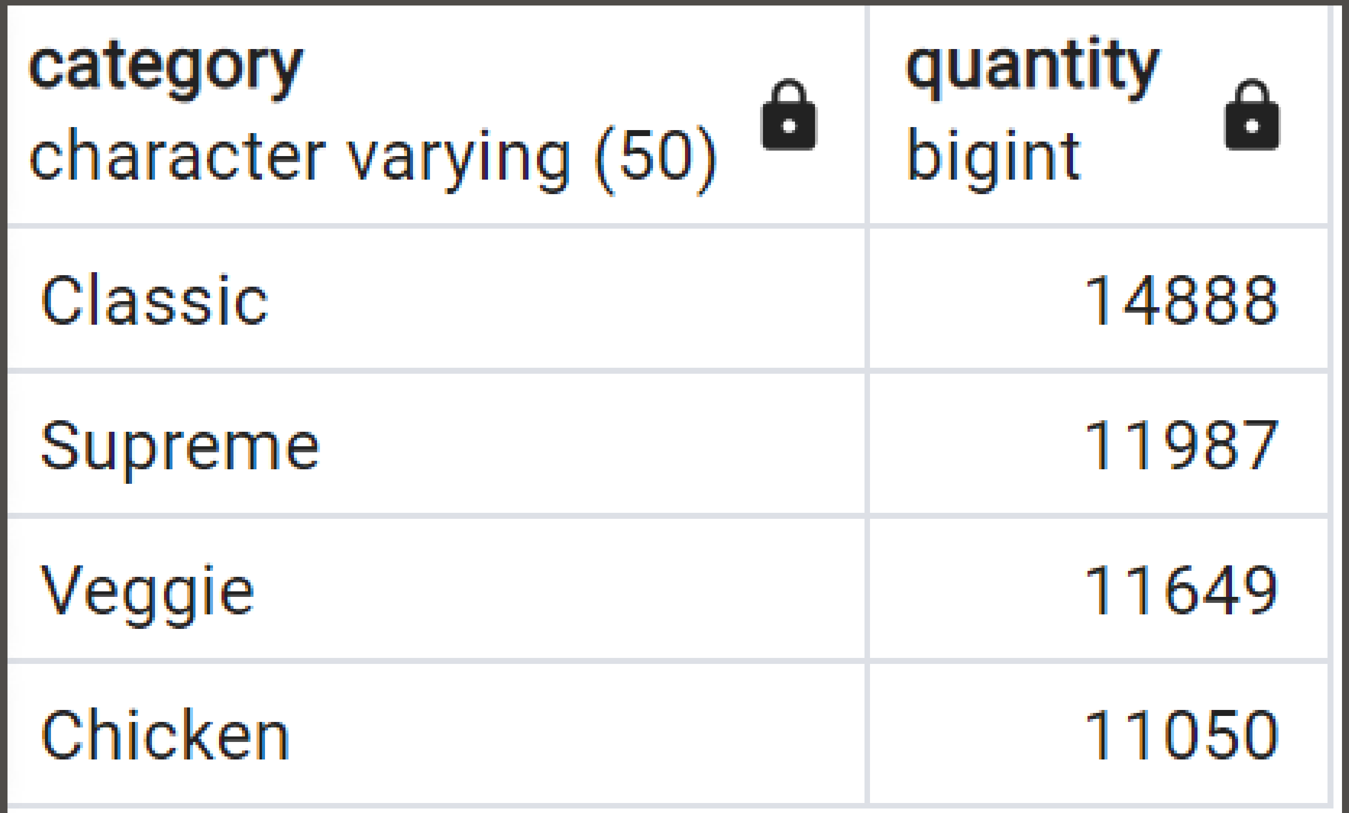


| size<br>character varying (50) | order_count<br>bigint |
|---|---|
| L | 18526 |
| M | 15385 |
| S | 14137 |
| XL | 544 |
| XXL | 28 |

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

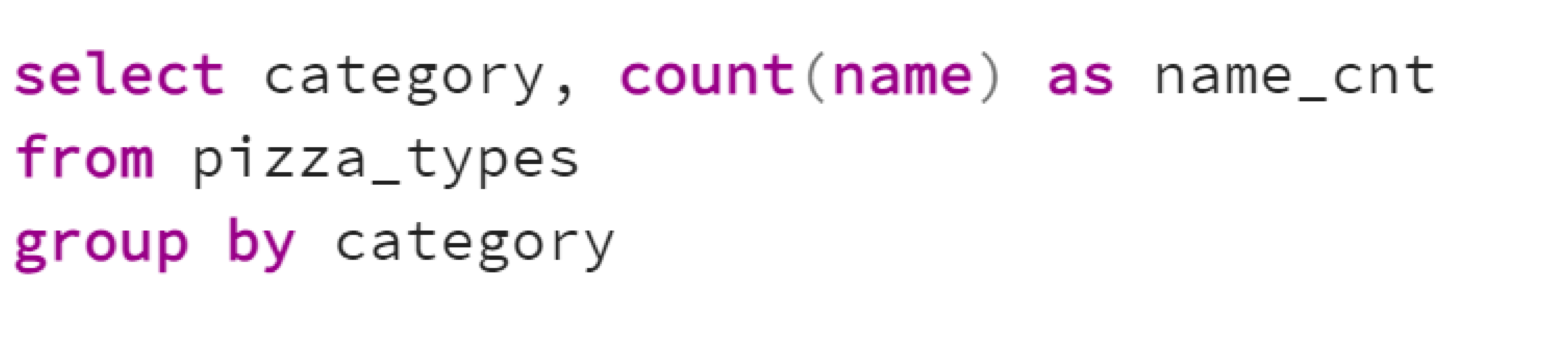


```
select pt.category, sum(od.quantity) as quantity
from pizza_types as pt
join pizzas as p
on pt.pizza_type_id = p.pizza_type_id
join order_details as od
on p.pizza_id = od.pizza_id
group by pt.category
order by quantity desc
```

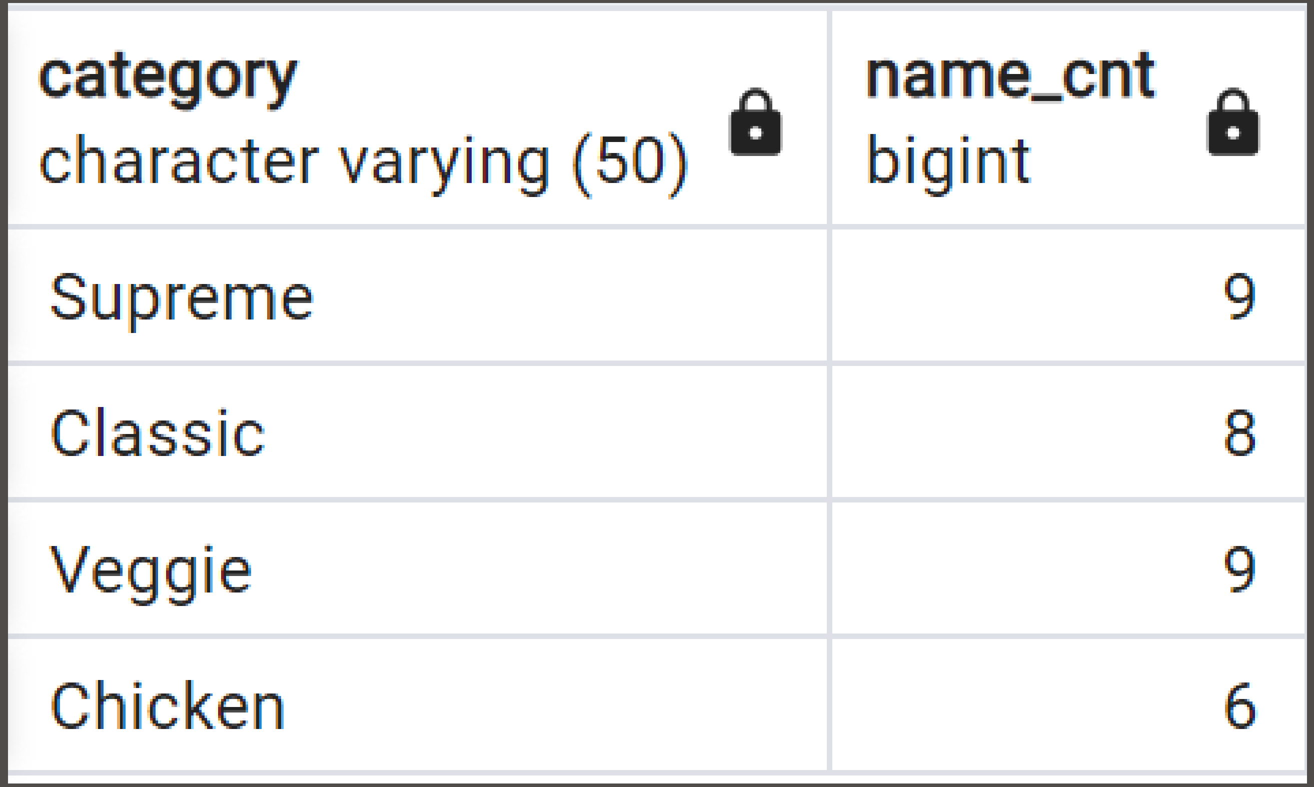


| category<br>character varying (50) | quantity<br>bigint |
|---|---|
| Classic | 14888 |
| Supreme | 11987 |
| Veggie | 11649 |
| Chicken | 11050 |

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

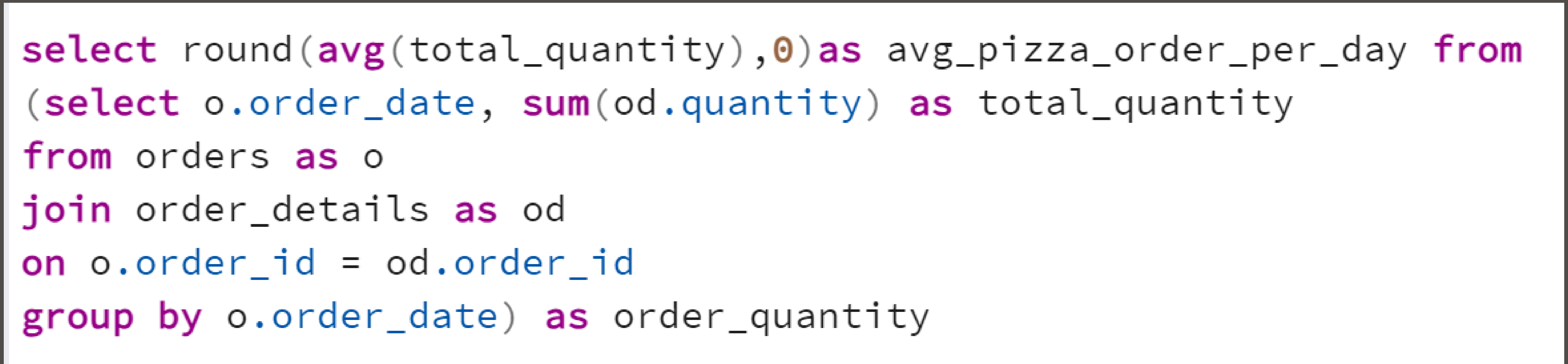


```
select category, count(name) as name_cnt
from pizza_types
group by category
```

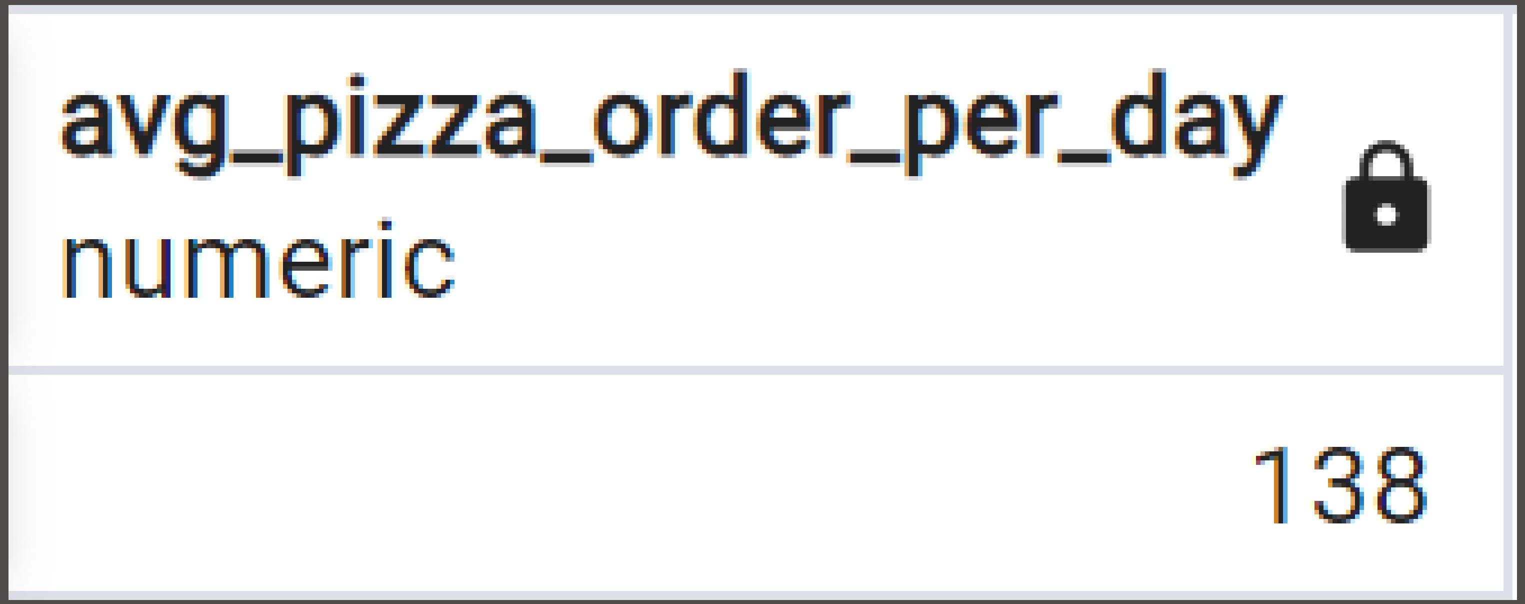


| category<br>character varying (50) | name_cnt<br>bigint |
|---|---:|
| Supreme | 9 |
| Classic | 8 |
| Veggie | 9 |
| Chicken | 6 |

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

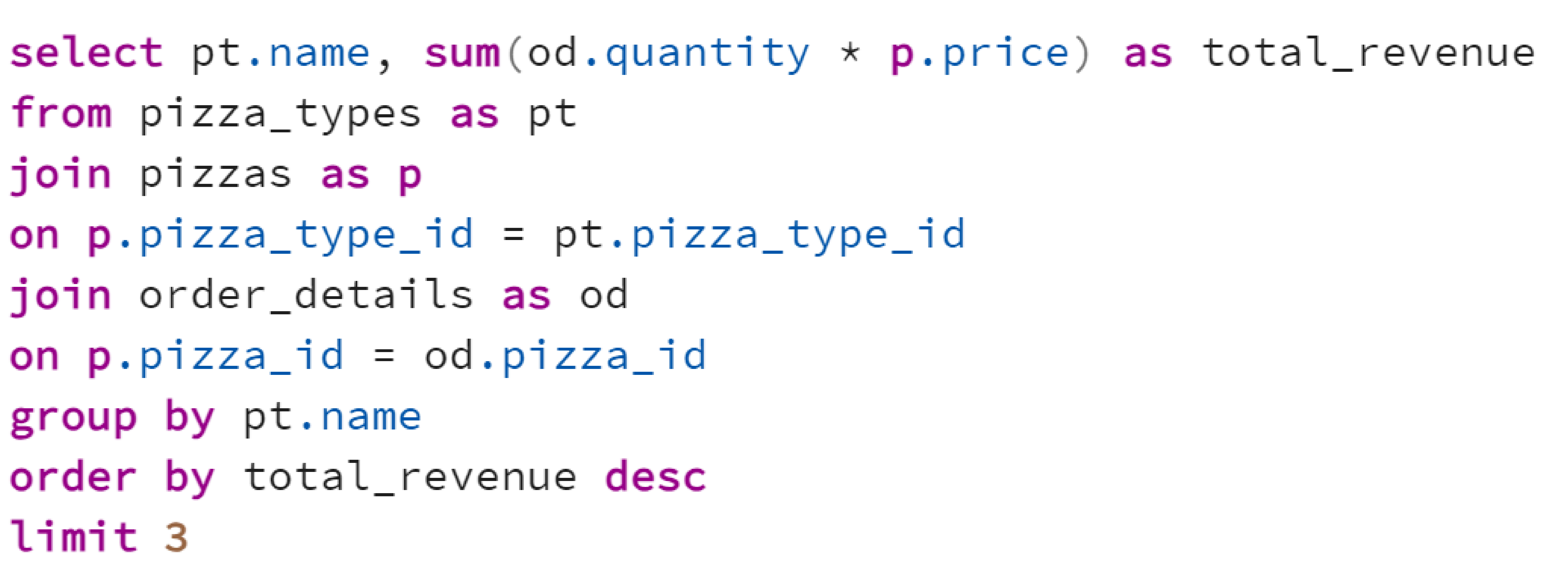


```
select round(avg(total_quantity),0)as avg_pizza_order_per_day from
(select o.order_date, sum(od.quantity) as total_quantity
from orders as o
join order_details as od
on o.order_id = od.order_id
group by o.order_date) as order_quantity
```

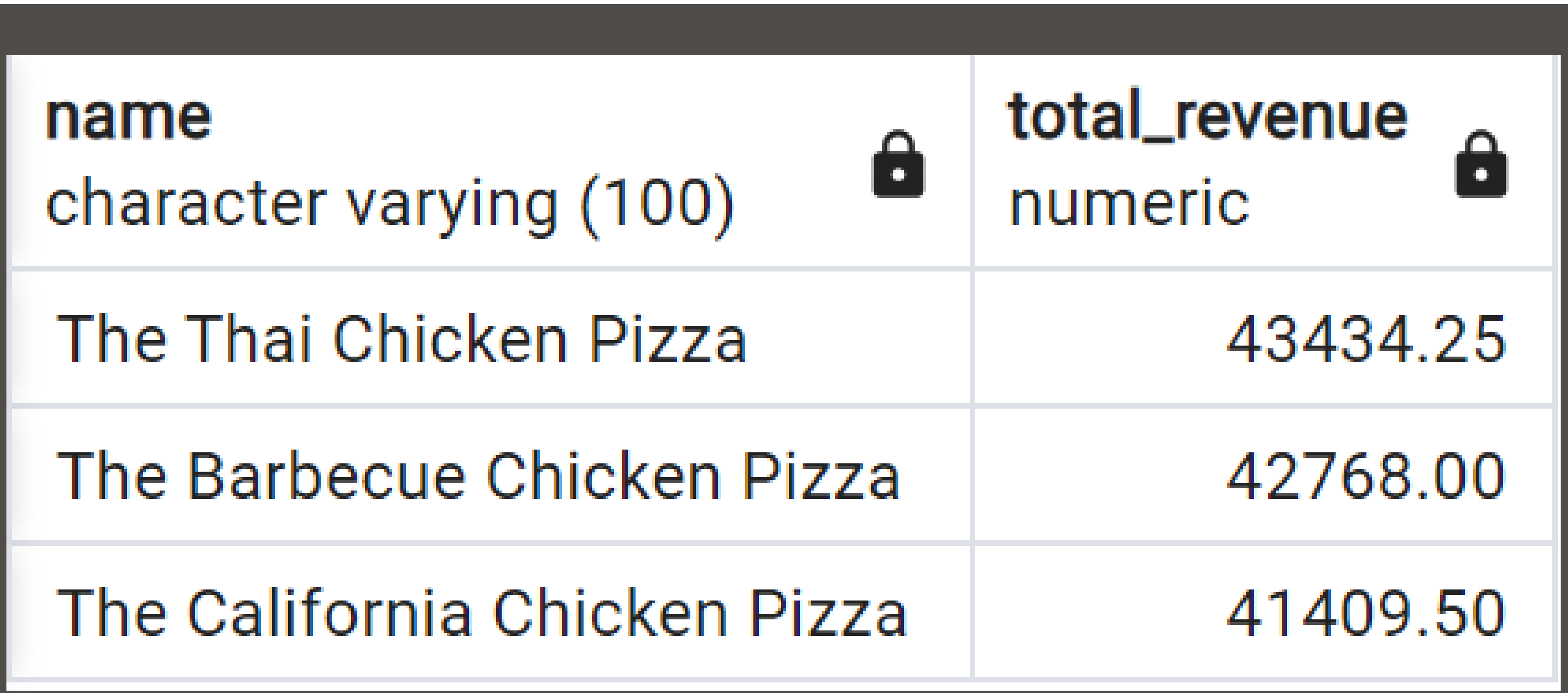


```
avg_pizza_order_per_day
numeric
                    138
```

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

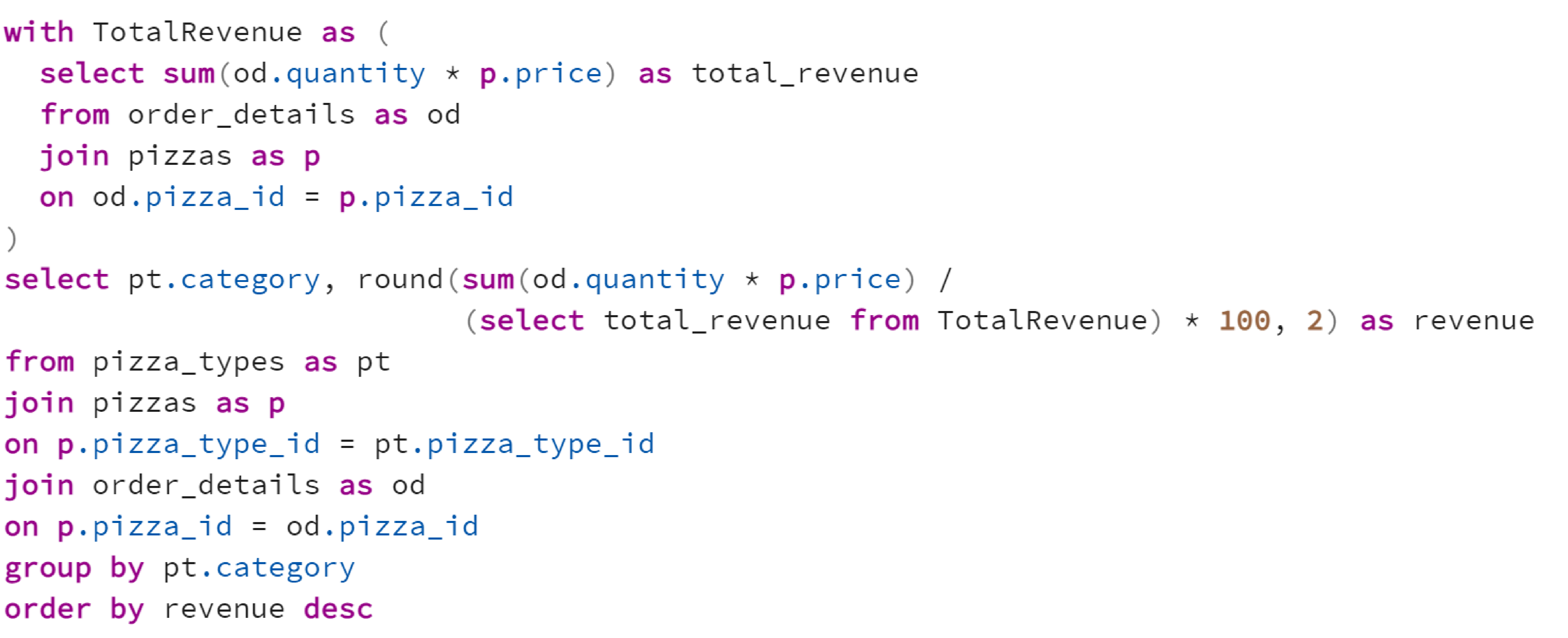


```
select pt.name, sum(od.quantity * p.price) as total_revenue
from pizza_types as pt
join pizzas as p
on p.pizza_type_id = pt.pizza_type_id
join order_details as od
on p.pizza_id = od.pizza_id
group by pt.name
order by total_revenue desc
limit 3
```

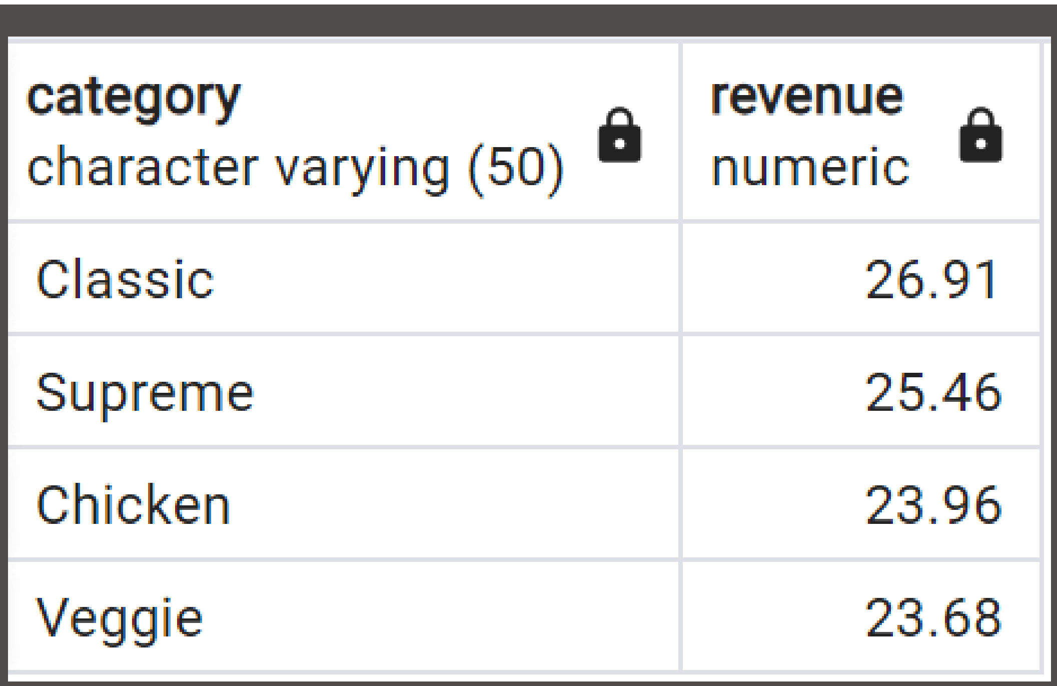


| name<br>character varying (100) | total_revenue<br>numeric |
|---|---:|
| The Thai Chicken Pizza | 43434.25 |
| The Barbecue Chicken Pizza | 42768.00 |
| The California Chicken Pizza | 41409.50 |

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

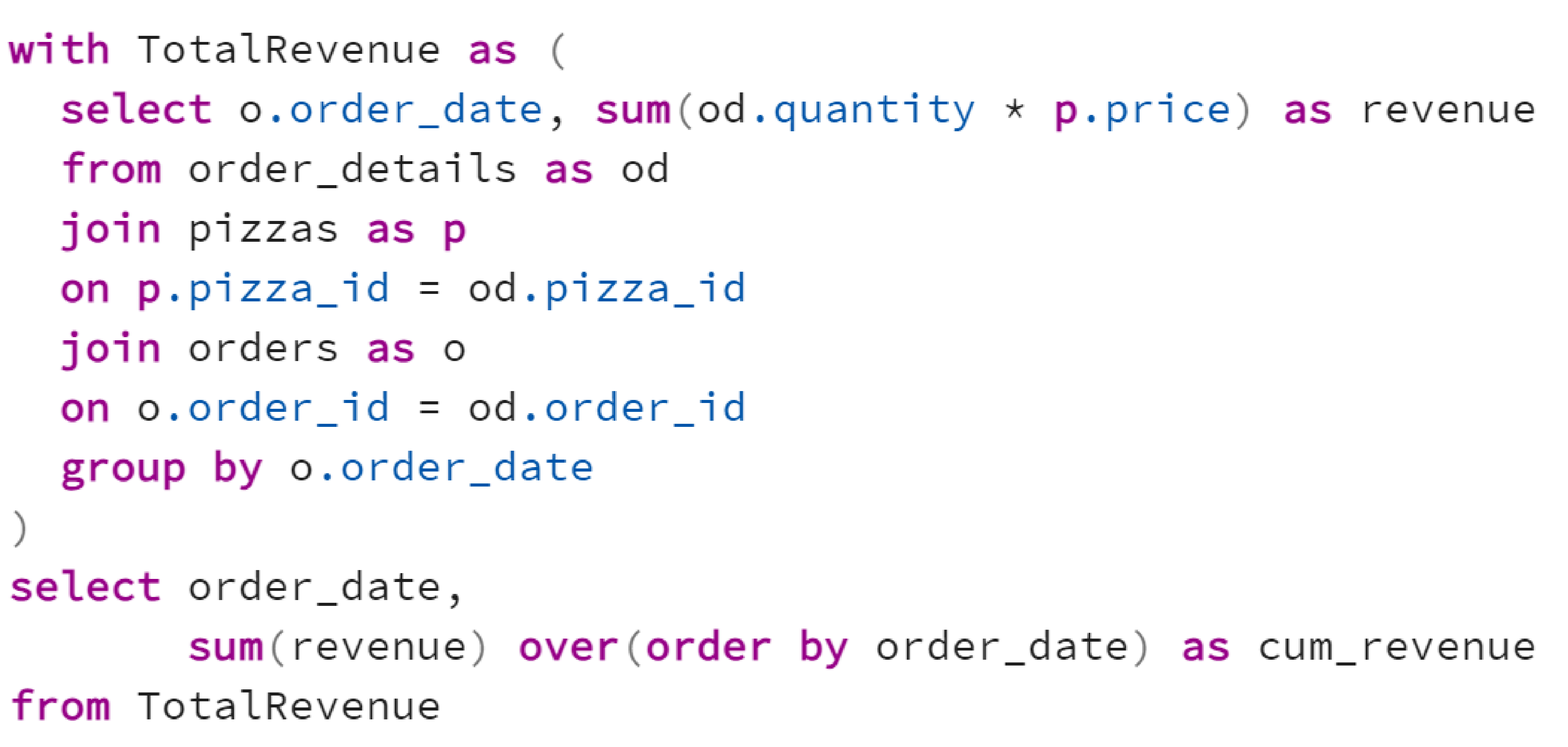


```
with TotalRevenue as (
  select sum(od.quantity * p.price) as total_revenue
  from order_details as od
  join pizzas as p
  on od.pizza_id = p.pizza_id
)
select pt.category, round(sum(od.quantity * p.price) /
                          (select total_revenue from TotalRevenue) * 100, 2) as revenue
from pizza_types as pt
join pizzas as p
on p.pizza_type_id = pt.pizza_type_id
join order_details as od
on p.pizza_id = od.pizza_id
group by pt.category
order by revenue desc
```

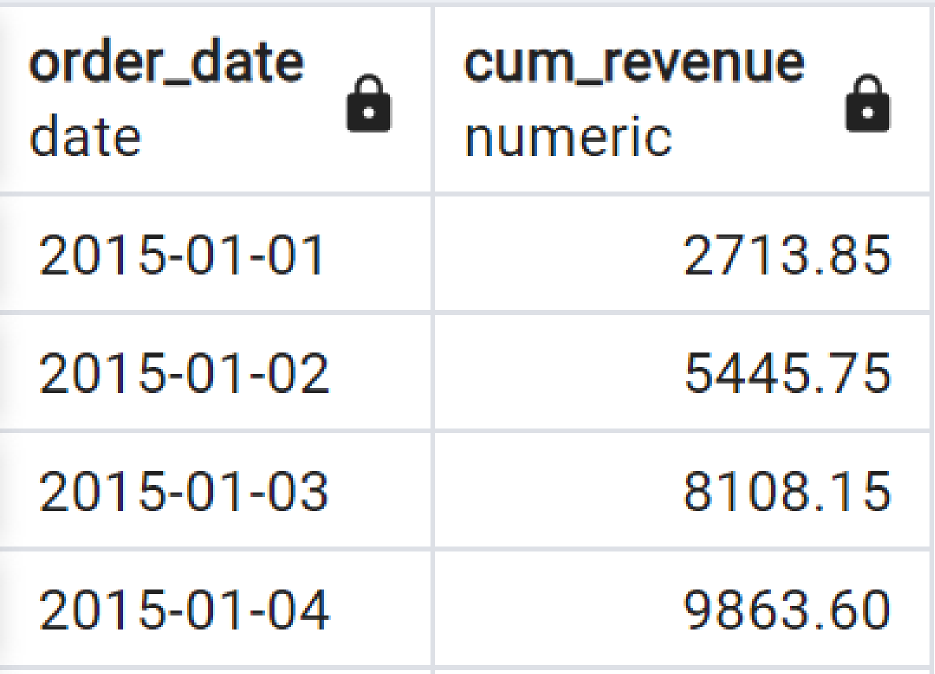


| category<br>character varying (50) | revenue<br>numeric |
|---|---:|
| Classic | 26.91 |
| Supreme | 25.46 |
| Chicken | 23.96 |
| Veggie | 23.68 |

# Analyze the cumulative revenue generated over time.

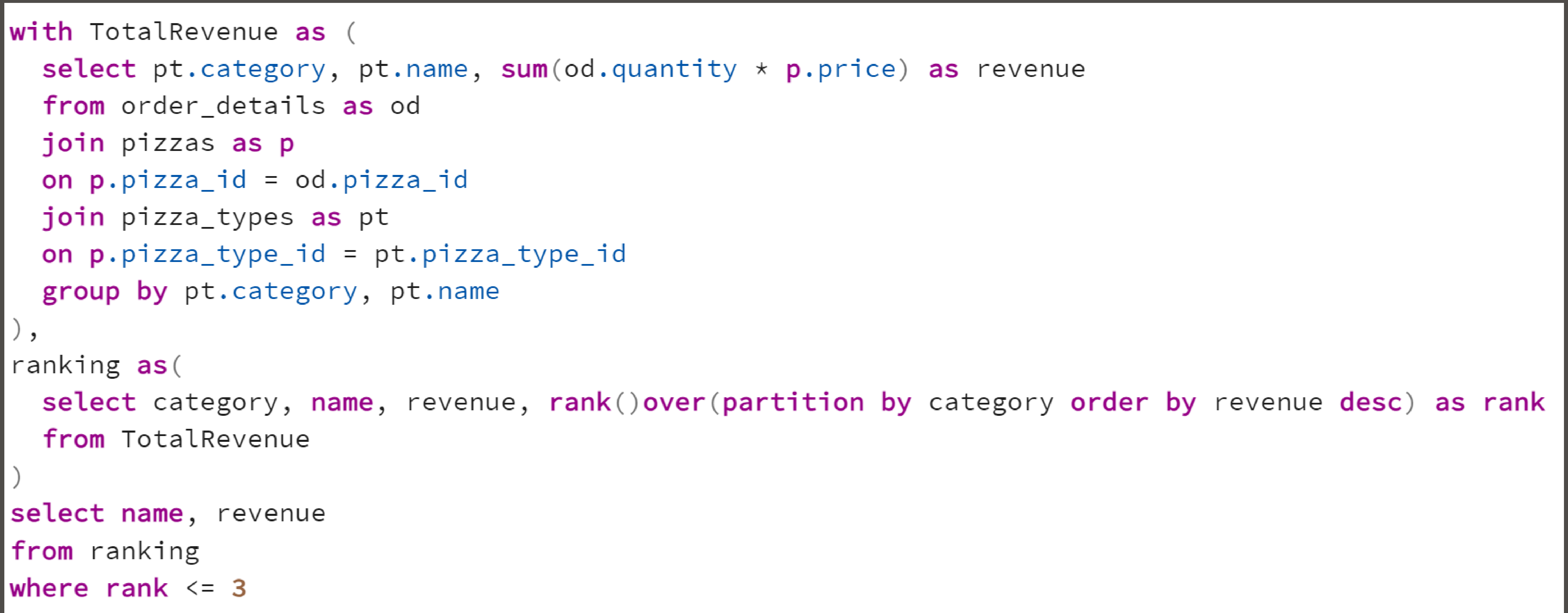


```sql
with TotalRevenue as (
  select o.order_date, sum(od.quantity * p.price) as revenue
  from order_details as od
  join pizzas as p
  on p.pizza_id = od.pizza_id
  join orders as o
  on o.order_id = od.order_id
  group by o.order_date
)
select order_date,
       sum(revenue) over(order by order_date) as cum_revenue
from TotalRevenue
```

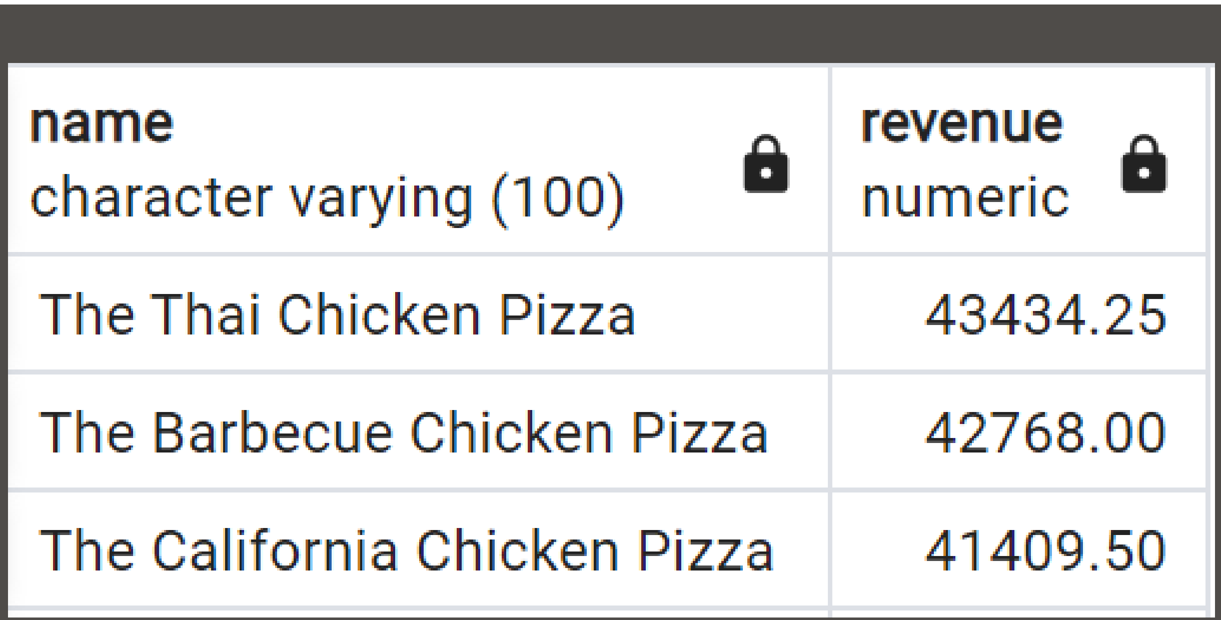


| order_date<br>date | cum_revenue<br>numeric |
|---|---:|
| 2015-01-01 | 2713.85 |
| 2015-01-02 | 5445.75 |
| 2015-01-03 | 8108.15 |
| 2015-01-04 | 9863.60 |

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



```
with TotalRevenue as (
  select pt.category, pt.name, sum(od.quantity * p.price) as revenue
  from order_details as od
  join pizzas as p
  on p.pizza_id = od.pizza_id
  join pizza_types as pt
  on p.pizza_type_id = pt.pizza_type_id
  group by pt.category, pt.name
),
ranking as(
  select category, name, revenue, rank()over(partition by category order by revenue desc) as rank
  from TotalRevenue
)
select name, revenue
from ranking
where rank <= 3
```



| **name**<br>character varying (100) | **revenue**<br>numeric |
|---|---:|
| The Thai Chicken Pizza | 43434.25 |
| The Barbecue Chicken Pizza | 42768.00 |
| The California Chicken Pizza | 41409.50 |