SQL PROJECT ON PIZZA SALES

INTRODUCTION

This project focuses on analyzing a dataset of pizza sales to gain insights into various aspects of pizza transactions. Using SQL, i have explored and examined key metrics, such as sales performance, customer preferences, and trends. By querying and aggregating data from the sales records, i have uncovered valuable information about the popularity of different pizza categories and types, peak sales time, and other relevant patterns. This analysis aims to provide actionable insights for optimizing sales strategies and enhancing customer satisfaction .

SCHEMAS

ORDER DETAILS

order_details_id

order_id

pizza_id

quantity

ORDERS

order_id

date

time

PIZZA TYPES

pizza_type_id

name

category

ingredients

PIZZAS

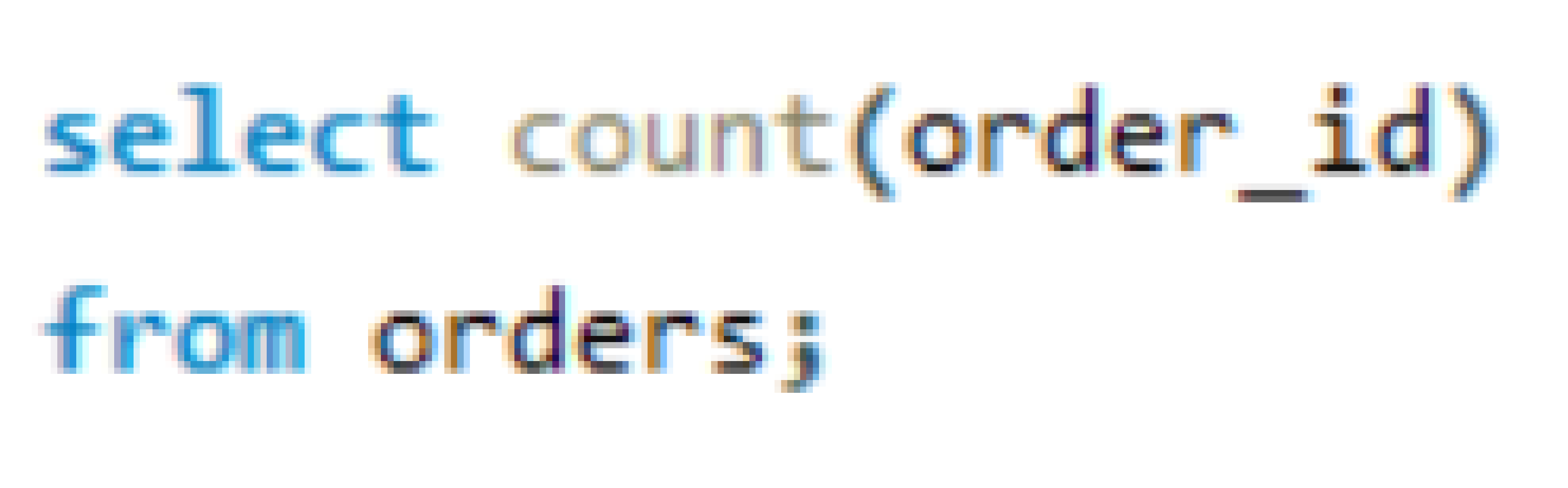
pizza_id

pizza_type_id

size

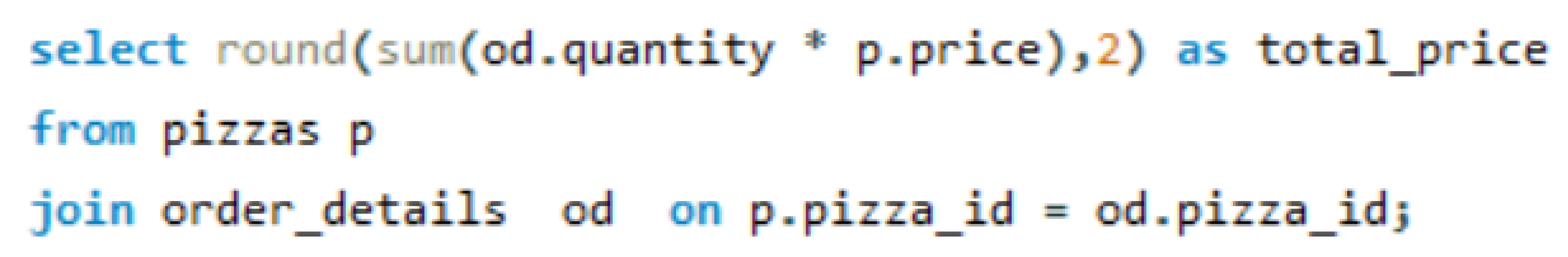
price

1.Retrieve the total number of orders placed.



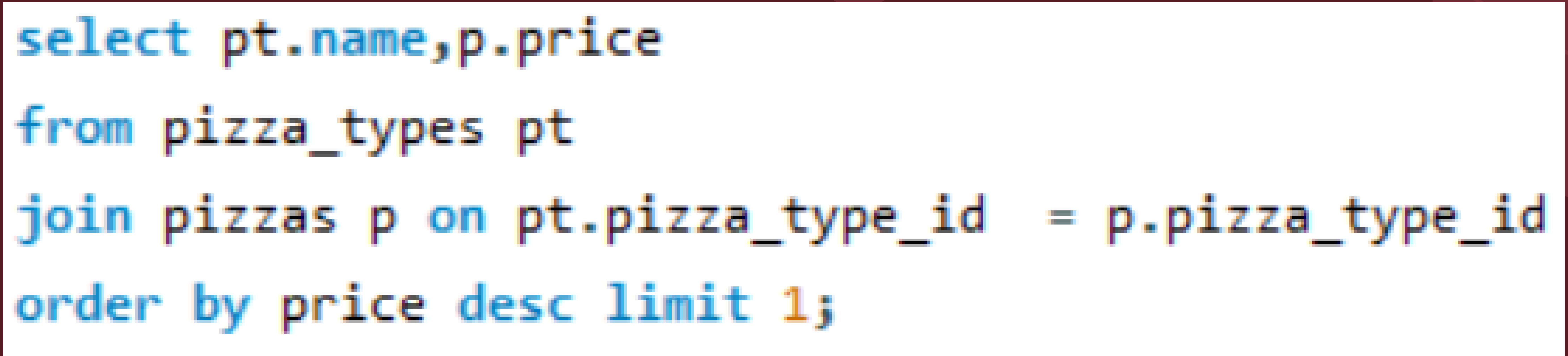
```
select count(order_id)
from orders;
```

2.Calculate the total revenue generated from pizza sales.



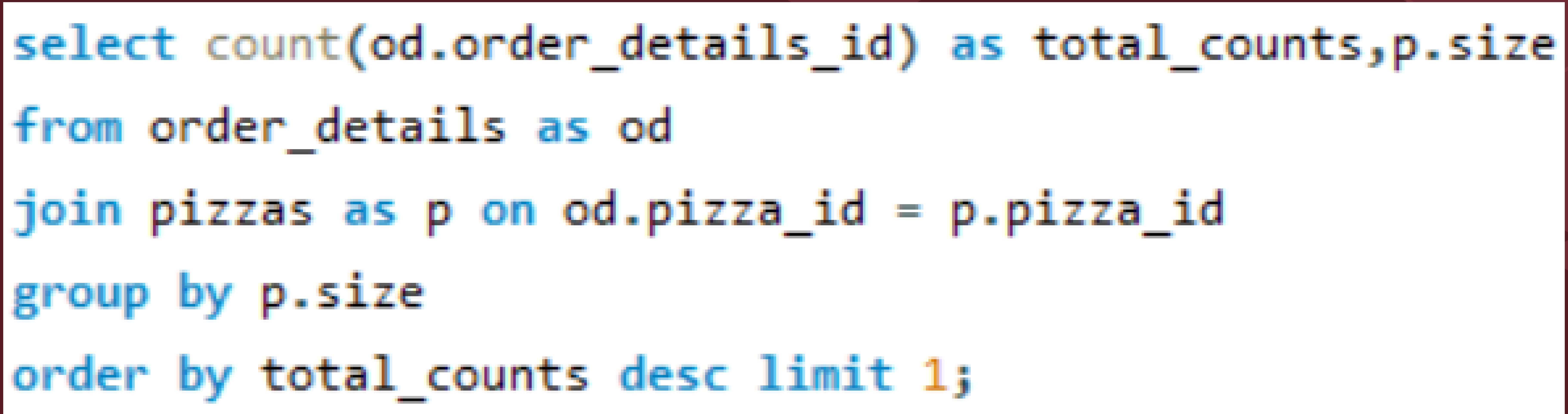
```sql
select round(sum(od.quantity * p.price),2) as total_price
from pizzas p
join order_details  od  on p.pizza_id = od.pizza_id;
```

3.Identify the highest priced pizza.



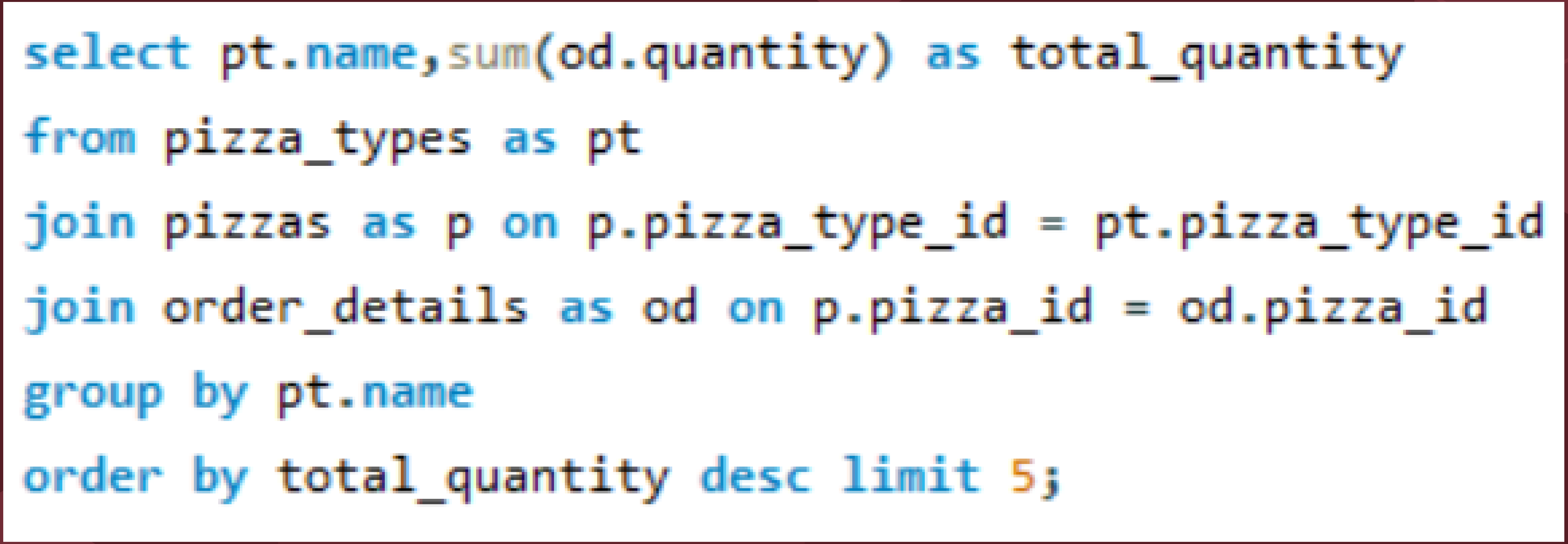
```sql
select pt.name,p.price
from pizza_types pt
join pizzas p on pt.pizza_type_id  = p.pizza_type_id
order by price desc limit 1;
```

4.Identify the most common pizza size ordered.



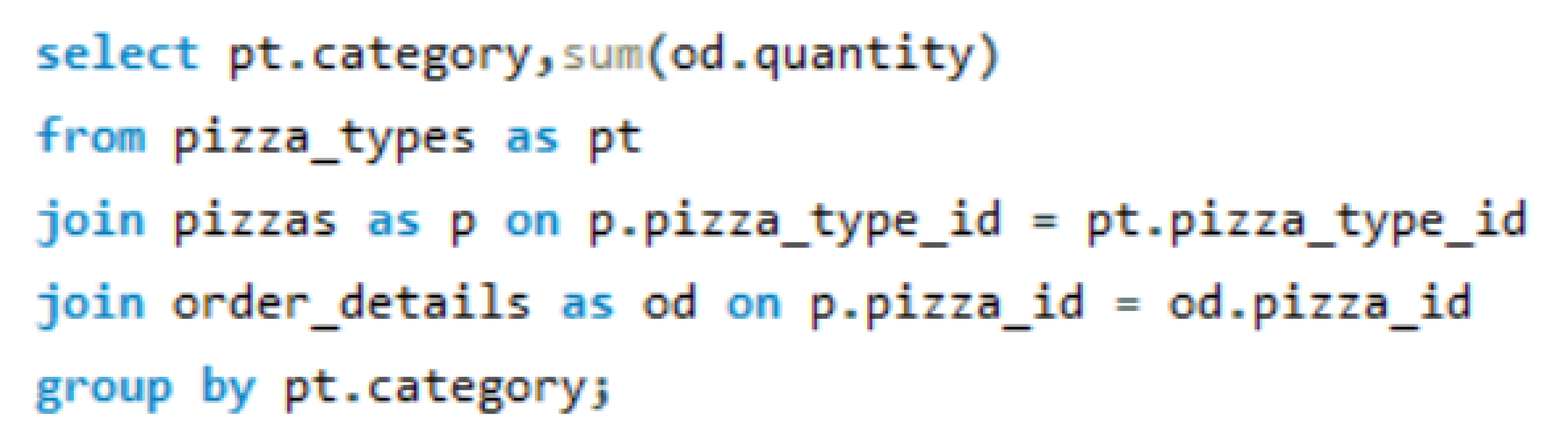
```
select count(od.order_details_id) as total_counts,p.size
from order_details as od
join pizzas as p on od.pizza_id = p.pizza_id
group by p.size
order by total_counts desc limit 1;
```

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



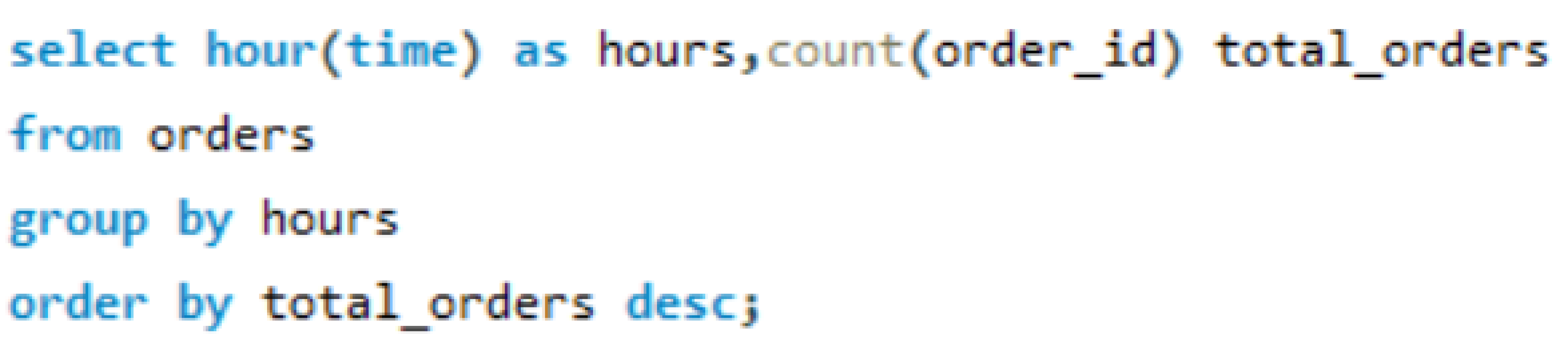
```sql
select pt.name,sum(od.quantity) as total_quantity
from pizza_types as pt
join pizzas as p on p.pizza_type_id = pt.pizza_type_id
join order_details as od on p.pizza_id = od.pizza_id
group by pt.name
order by total_quantity desc limit 5;
```

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



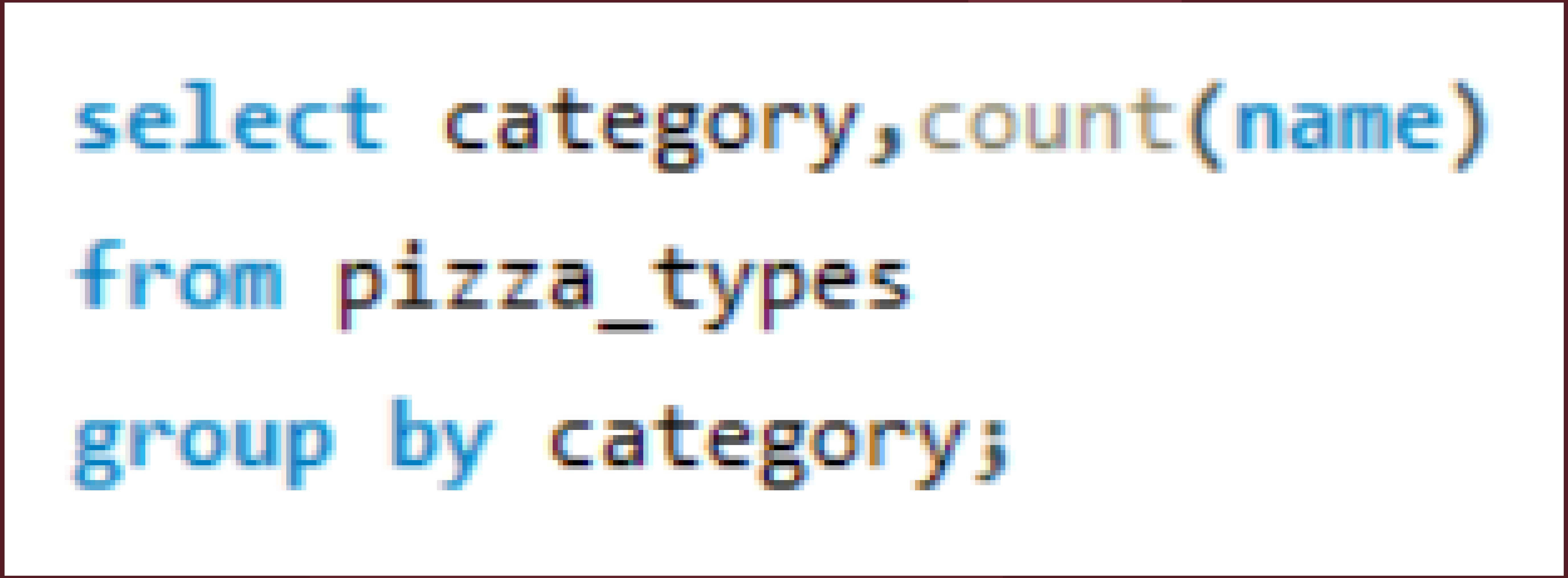
```sql
select pt.category,sum(od.quantity)
from pizza_types as pt
join pizzas as p on p.pizza_type_id = pt.pizza_type_id
join order_details as od on p.pizza_id = od.pizza_id
group by pt.category;
```

7.Determine the distribution of orders by hour of the day.



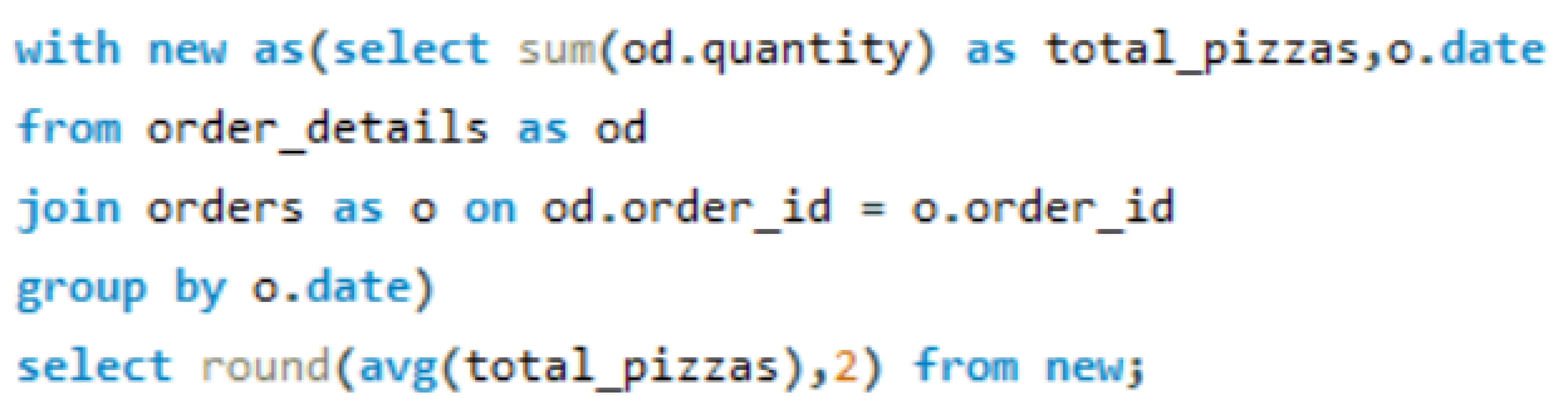
```sql
select hour(time) as hours,count(order_id) total_orders
from orders
group by hours
order by total_orders desc;
```

8. Find the category-wise distribution of pizzas.



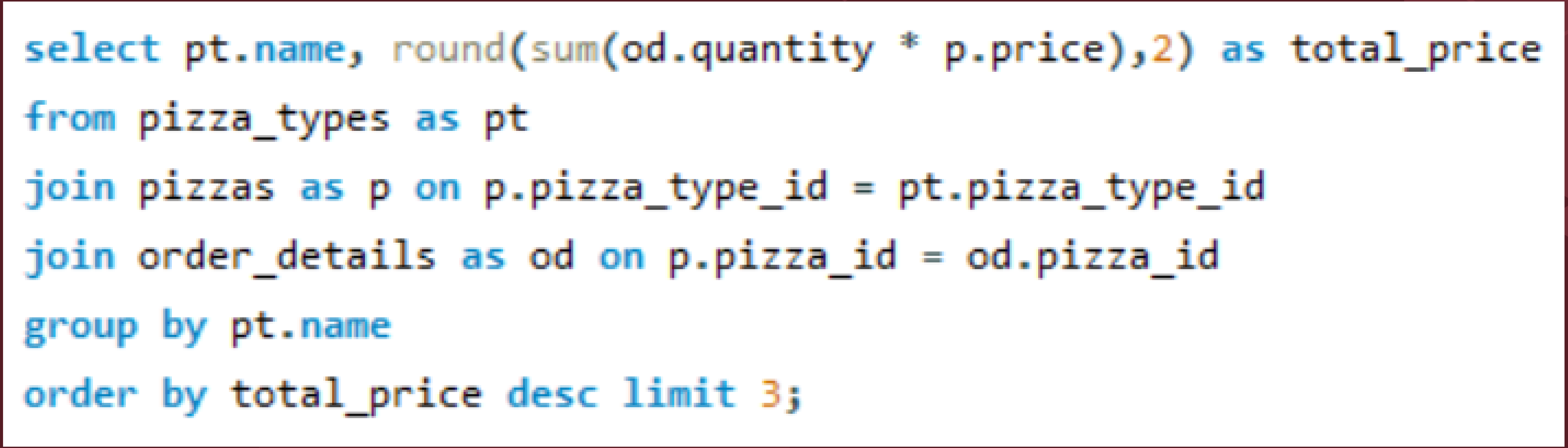
```
select category,count(name)
from pizza_types
group by category;
```

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



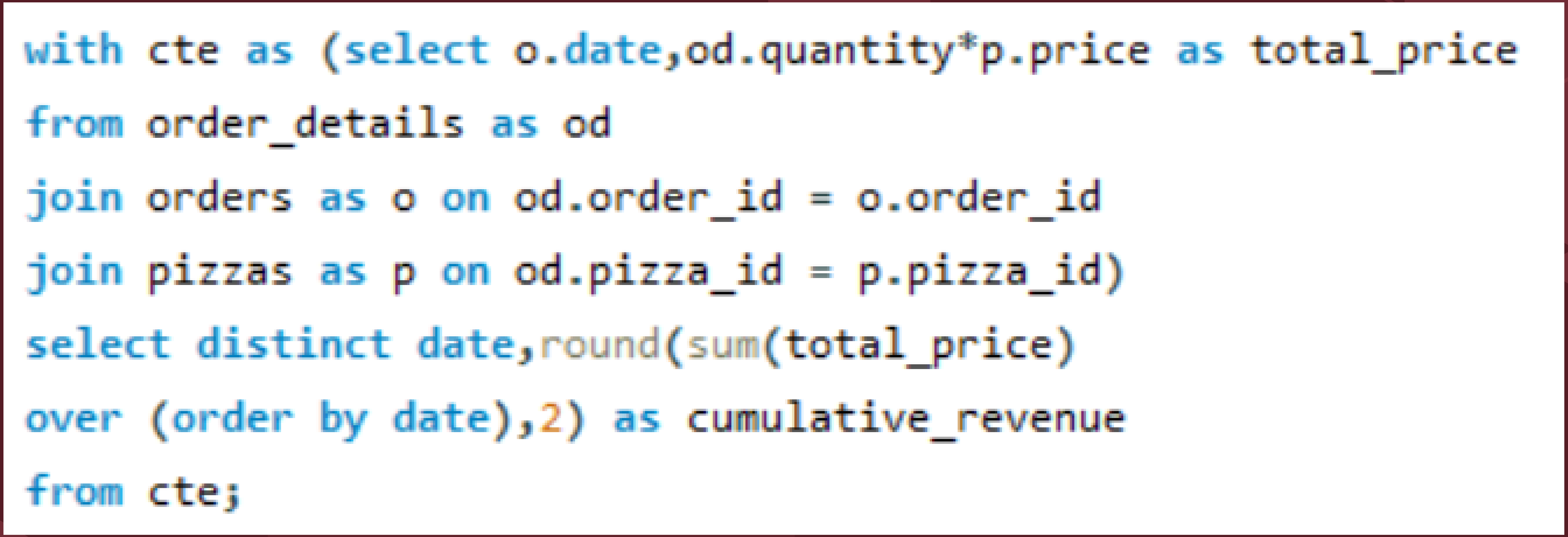
```
with new as(select sum(od.quantity) as total_pizzas,o.date
from order_details as od
join orders as o on od.order_id = o.order_id
group by o.date)
select round(avg(total_pizzas),2) from new;
```

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



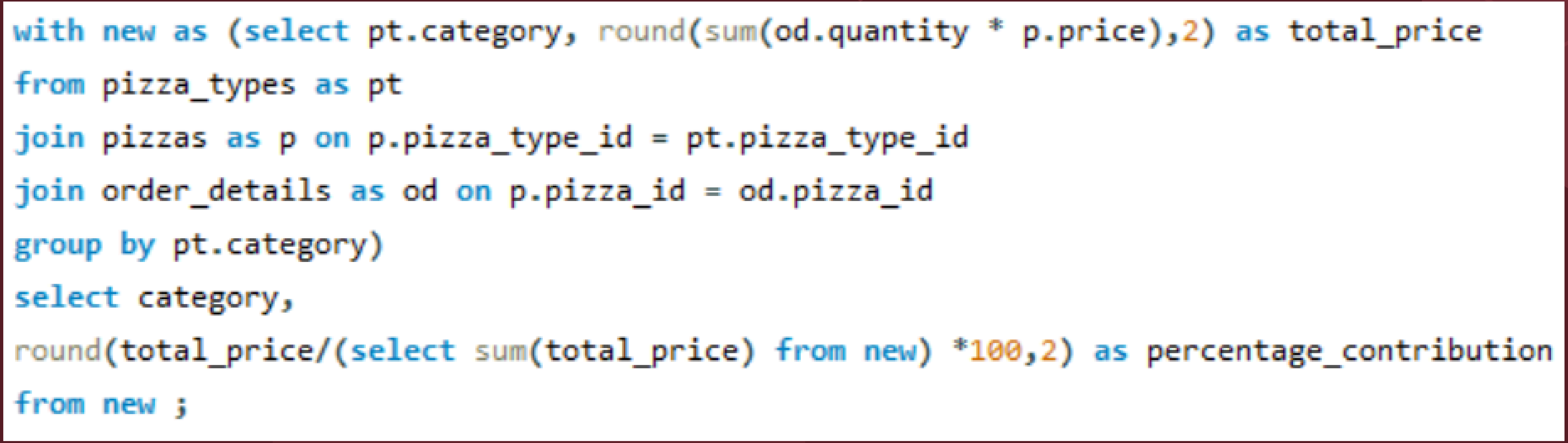
```
select pt.name, round(sum(od.quantity * p.price),2) as total_price
from pizza_types as pt
join pizzas as p on p.pizza_type_id = pt.pizza_type_id
join order_details as od on p.pizza_id = od.pizza_id
group by pt.name
order by total_price desc limit 3;
```

11.Analyze the cumulative revenue generated over time.



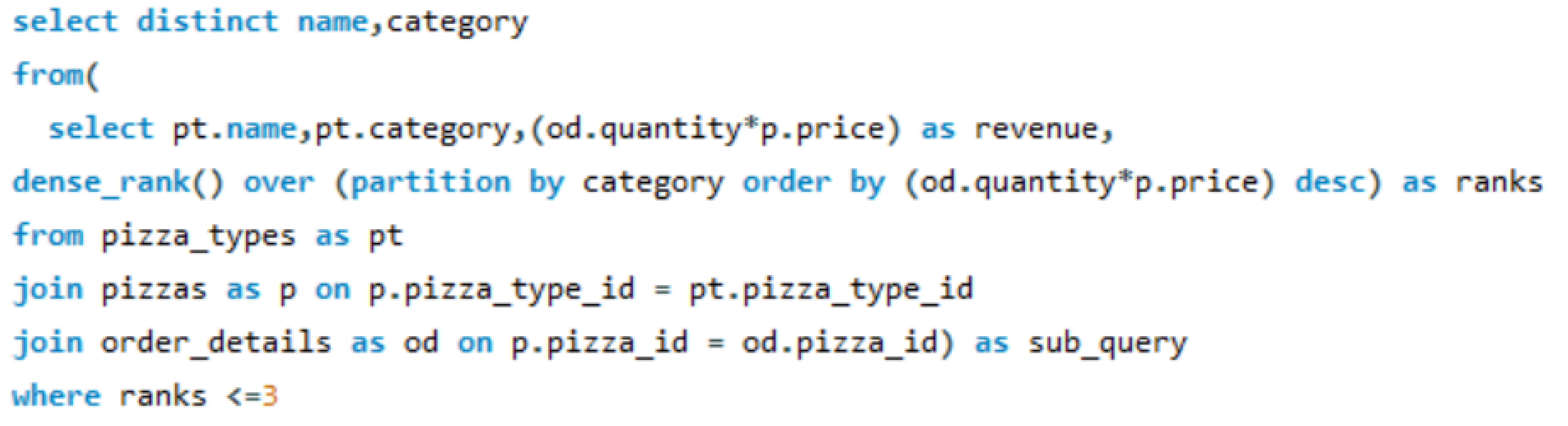
```
with cte as (select o.date,od.quantity*p.price as total_price
from order_details as od
join orders as o on od.order_id = o.order_id
join pizzas as p on od.pizza_id = p.pizza_id)
select distinct date,round(sum(total_price)
over (order by date),2) as cumulative_revenue
from cte;
```

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



```
with new as (select pt.category, round(sum(od.quantity * p.price),2) as total_price
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)
select category,
round(total_price/(select sum(total_price) from new) *100,2) as percentage_contribution
from new ;
```

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



```sql
select distinct name,category
from(
  select pt.name,pt.category,(od.quantity*p.price) as revenue,
dense_rank() over (partition by category order by (od.quantity*p.price) desc) as ranks
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) as sub_query
where ranks <=3
```

THANK YOU