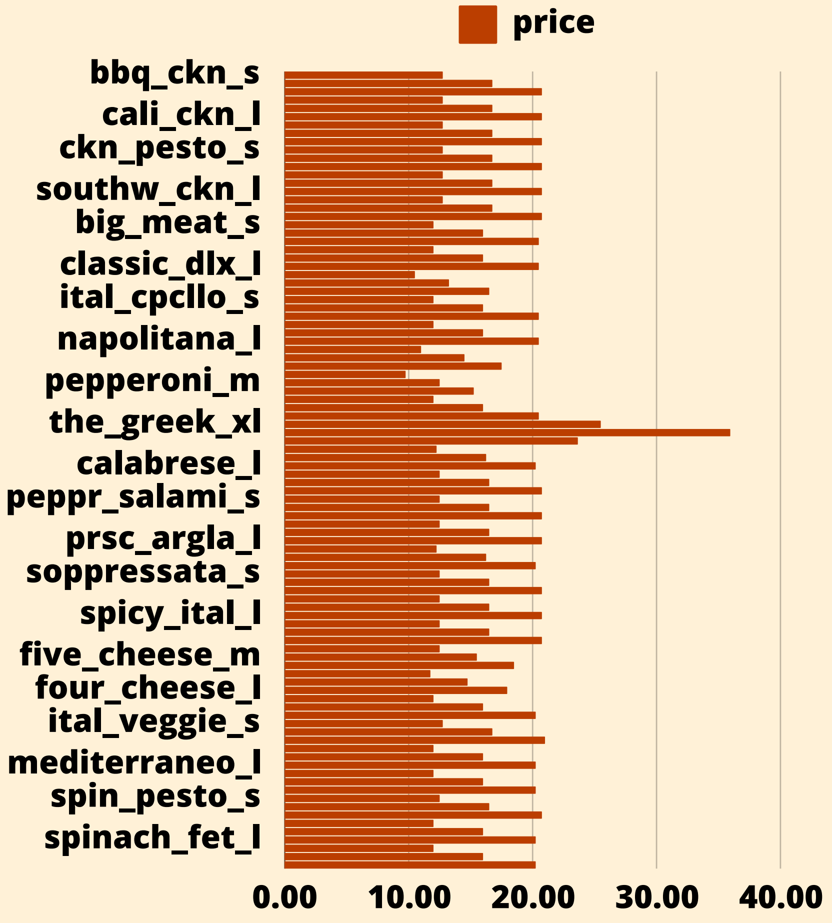
Pizza Sales Analysis using postgreSQL

LICERIA
& CO.

PIZZA SALES
IN 2015



PRESENTED BY
ISRAK JAHAN SAMIR

Dataset

For this project, I sourced pizza sales data from Kaggle (link) which contains four csv files: order_details.csv, orders.csv, pizza_types.csv, and pizzas.csv, which I subsequently imported into
my pgAdmin Workbench.

- orders.csv has columns : order_id, date, time
- order_details.csv has columns : order_details_id, order_id, pizza_id, quantity
- pizza_types.csv has columns : pizza_type_id, name, category, ingredients
- pizzas.csv has columns : pizza_id, pizza_type_id, size, price

Questions to answer

Basic:

1) Retrieve the total number of orders placed.
2) Calculate the total revenue generated from pizza sales.
3) Identify the highest-priced pizza.
4) Identify the most common pizza size ordered.
5) List the top 5 most ordered pizza types along with their quantities.

Intermediate:

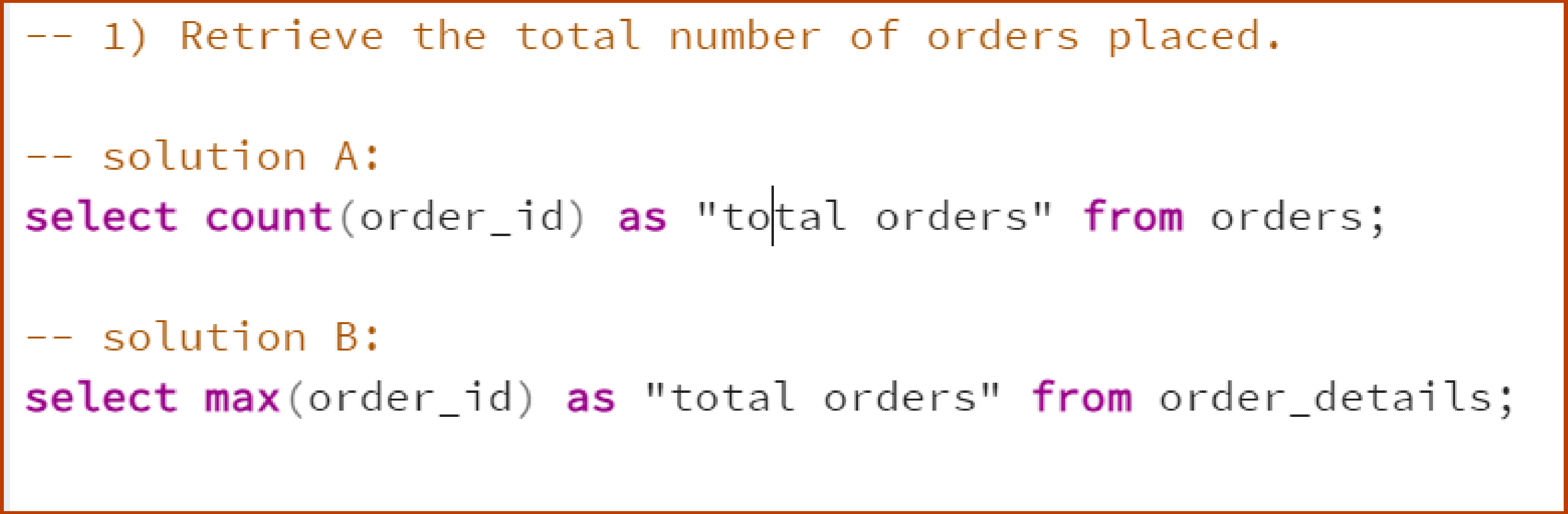
6) Join the necessary tables to find the total quantity of each pizza category ordered.
7) Determine the distribution of orders by hour of the day.
8) Join relevant tables to find the category-wise distribution of pizzas.
9) Group the orders by date and calculate the average number of pizzas ordered per day.
10) Determine the top 3 most ordered pizza types based on revenue.

Advanced:

11) Calculate the percentage contribution of each pizza type to total revenue.
12) Analyze the cumulative revenue generated over time.
13) Determine the top 3 most ordered pizza types based on revenue for each pizza category.

1) Retrieve the total number of orders placed.

Query:

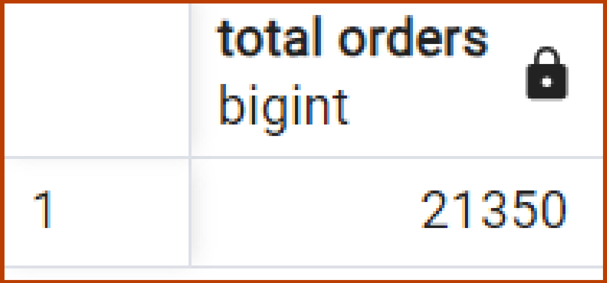


```
-- 1) Retrieve the total number of orders placed.

-- solution A:
select count(order_id) as "total orders" from orders;

-- solution B:
select max(order_id) as "total orders" from order_details;
```

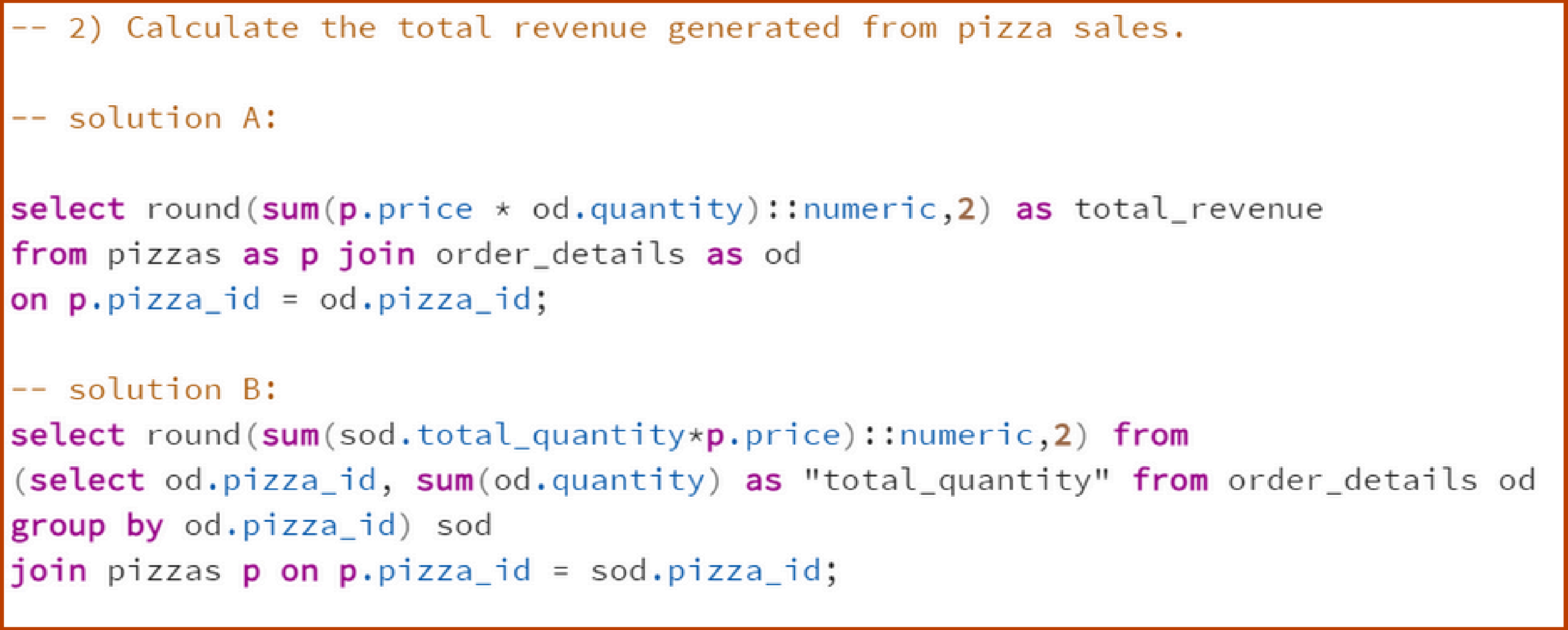
Output:



	total orders bigint
1	21350

2) Calculate the total revenue generated from pizza sales.

Query:



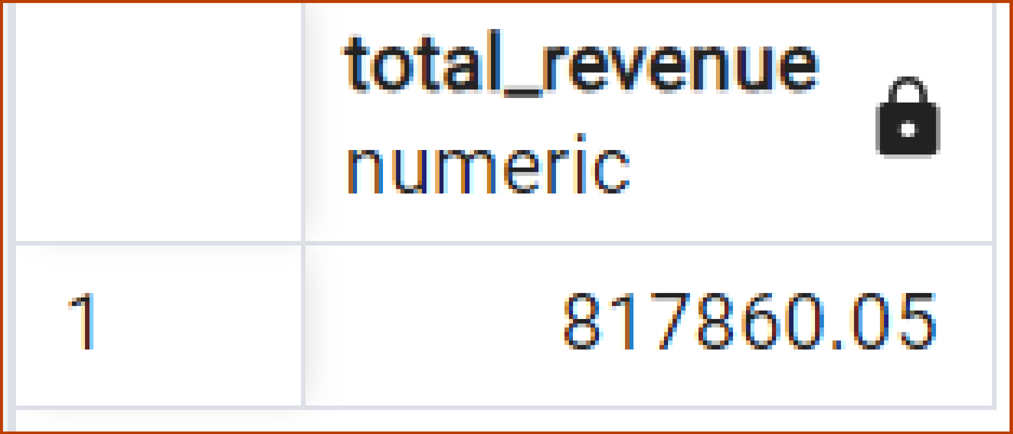
```sql
-- 2) Calculate the total revenue generated from pizza sales.

-- solution A:

select round(sum(p.price * od.quantity)::numeric,2) as total_revenue
from pizzas as p join order_details as od
on p.pizza_id = od.pizza_id;

-- solution B:
select round(sum(sod.total_quantity*p.price)::numeric,2) from
(select od.pizza_id, sum(od.quantity) as "total_quantity" from order_details od
group by od.pizza_id) sod
join pizzas p on p.pizza_id = sod.pizza_id;
```

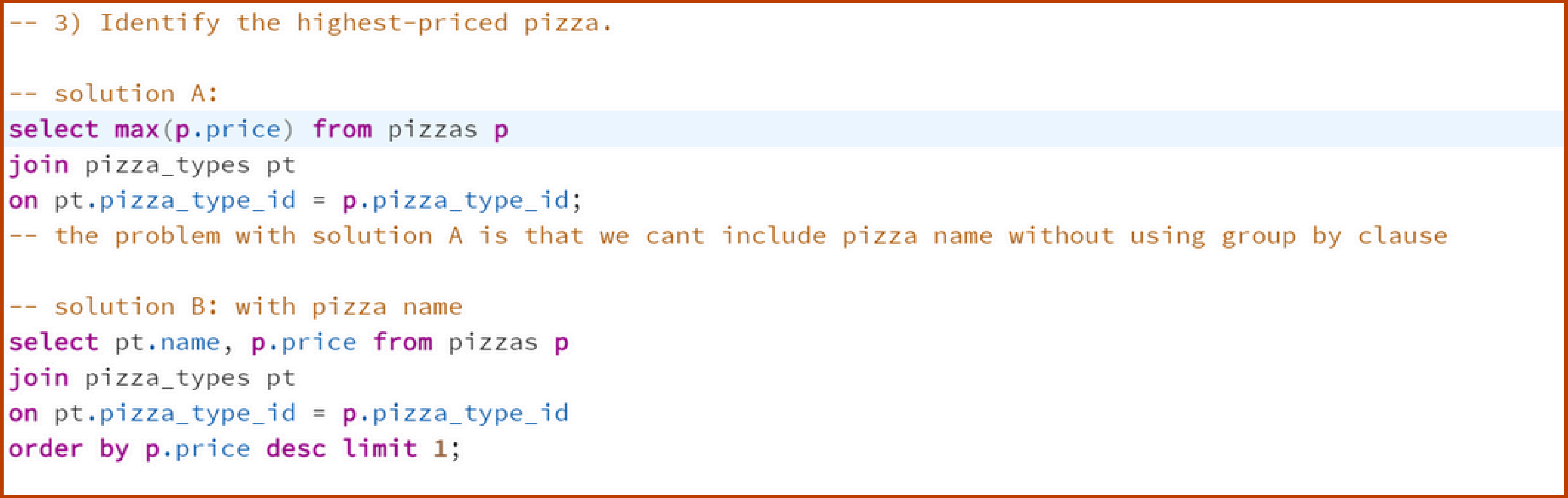
Output:



	total_revenue numeric
1	817860.05

3) Identify the highest-priced pizza.

Query:

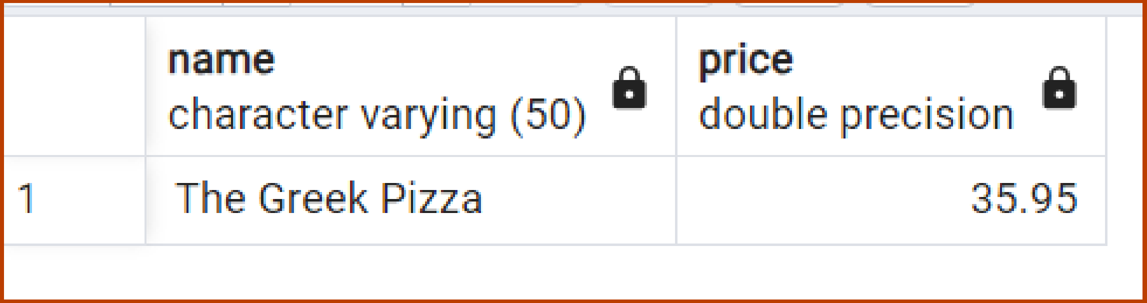


```
-- 3) Identify the highest-priced pizza.

-- solution A:
select max(p.price) from pizzas p
join pizza_types pt
on pt.pizza_type_id = p.pizza_type_id;
-- the problem with solution A is that we cant include pizza name without using group by clause

-- solution B: with pizza name
select pt.name, p.price from pizzas p
join pizza_types pt
on pt.pizza_type_id = p.pizza_type_id
order by p.price desc limit 1;
```

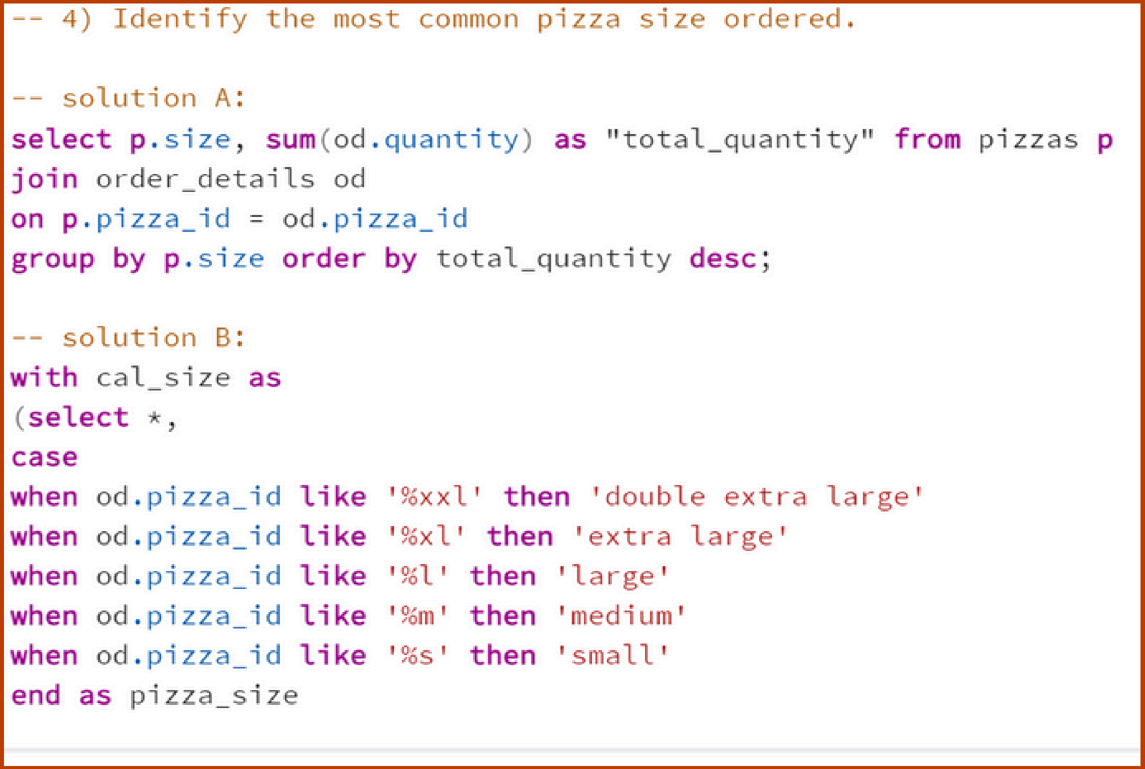
Output:



	name character varying (50)	price double precision
1	The Greek Pizza	35.95

4) Identify the most common pizza size ordered.

Query:

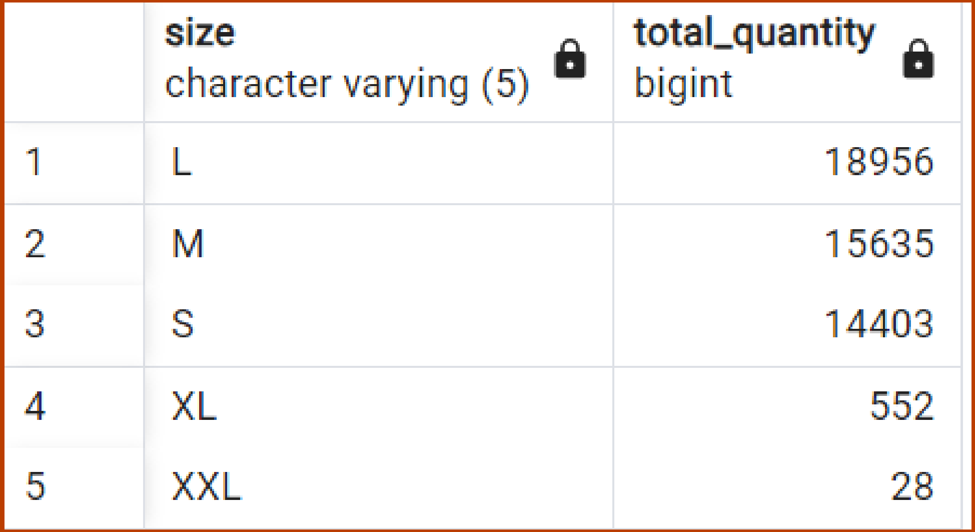


```
-- 4) Identify the most common pizza size ordered.

-- solution A:
select p.size, sum(od.quantity) as "total_quantity" from pizzas p
join order_details od
on p.pizza_id = od.pizza_id
group by p.size order by total_quantity desc;

-- solution B:
with cal_size as
(select *,
case
when od.pizza_id like '%xxl' then 'double extra large'
when od.pizza_id like '%xl' then 'extra large'
when od.pizza_id like '%l' then 'large'
when od.pizza_id like '%m' then 'medium'
when od.pizza_id like '%s' then 'small'
end as pizza_size
```

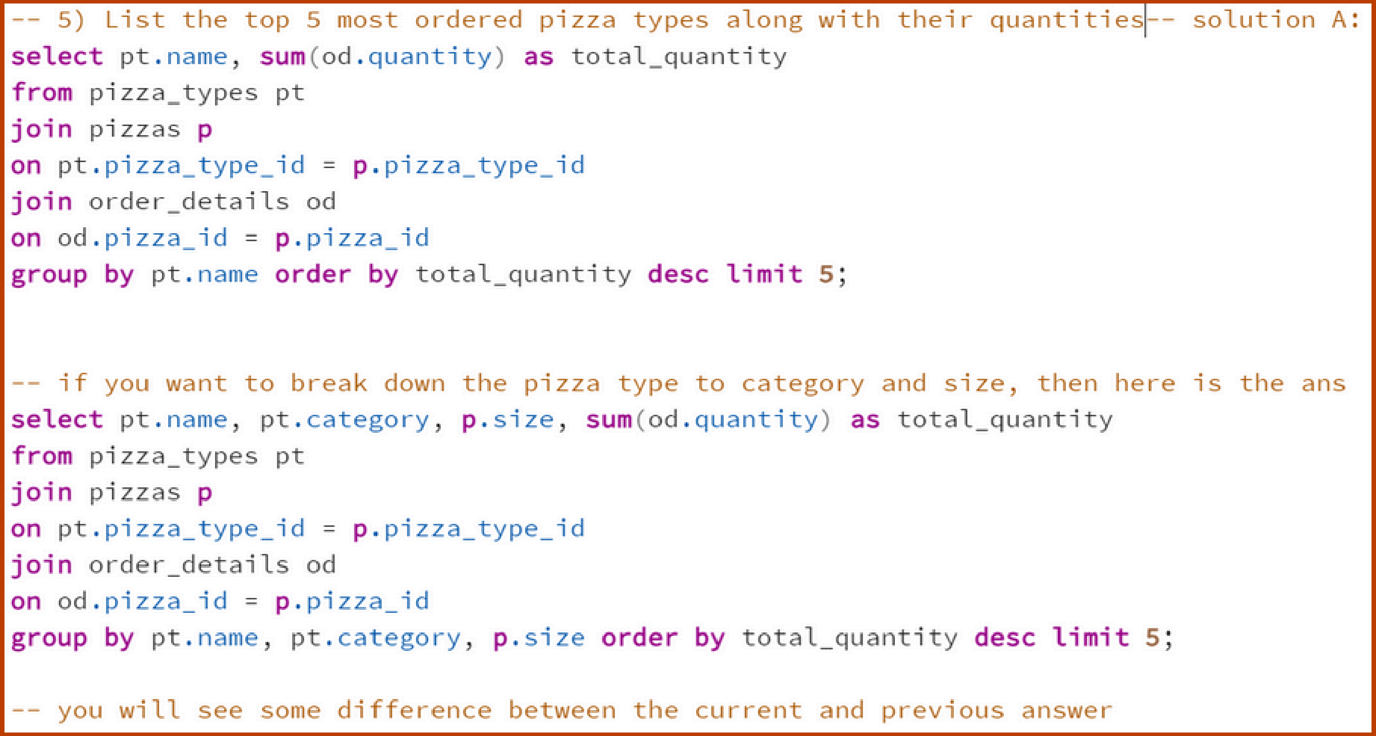
Output:



	size character varying (5)	total_quantity bigint
1	L	18956
2	M	15635
3	S	14403
4	XL	552
5	XXL	28

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

Query:

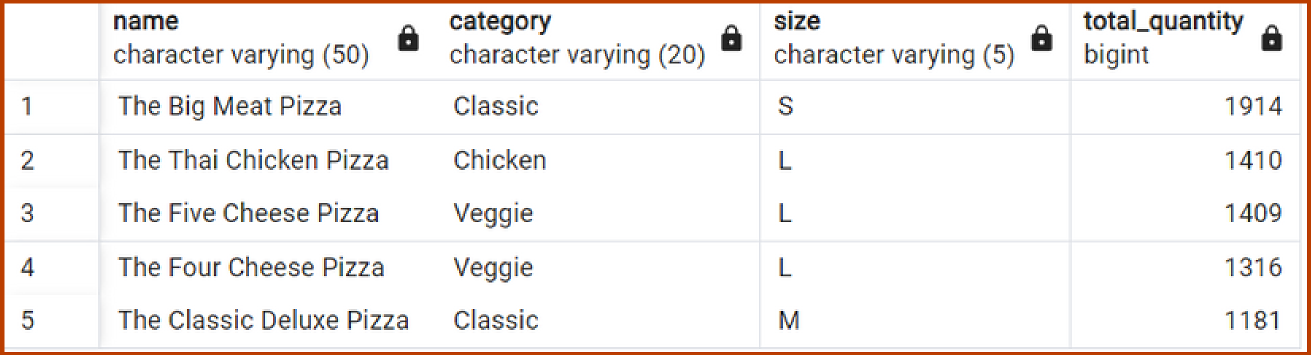


```
-- 5) List the top 5 most ordered pizza types along with their quantities-- solution A:
select pt.name, sum(od.quantity) as total_quantity
from pizza_types pt
join pizzas p
on pt.pizza_type_id = p.pizza_type_id
join order_details od
on od.pizza_id = p.pizza_id
group by pt.name order by total_quantity desc limit 5;


-- if you want to break down the pizza type to category and size, then here is the ans
select pt.name, pt.category, p.size, sum(od.quantity) as total_quantity
from pizza_types pt
join pizzas p
on pt.pizza_type_id = p.pizza_type_id
join order_details od
on od.pizza_id = p.pizza_id
group by pt.name, pt.category, p.size order by total_quantity desc limit 5;

-- you will see some difference between the current and previous answer
```

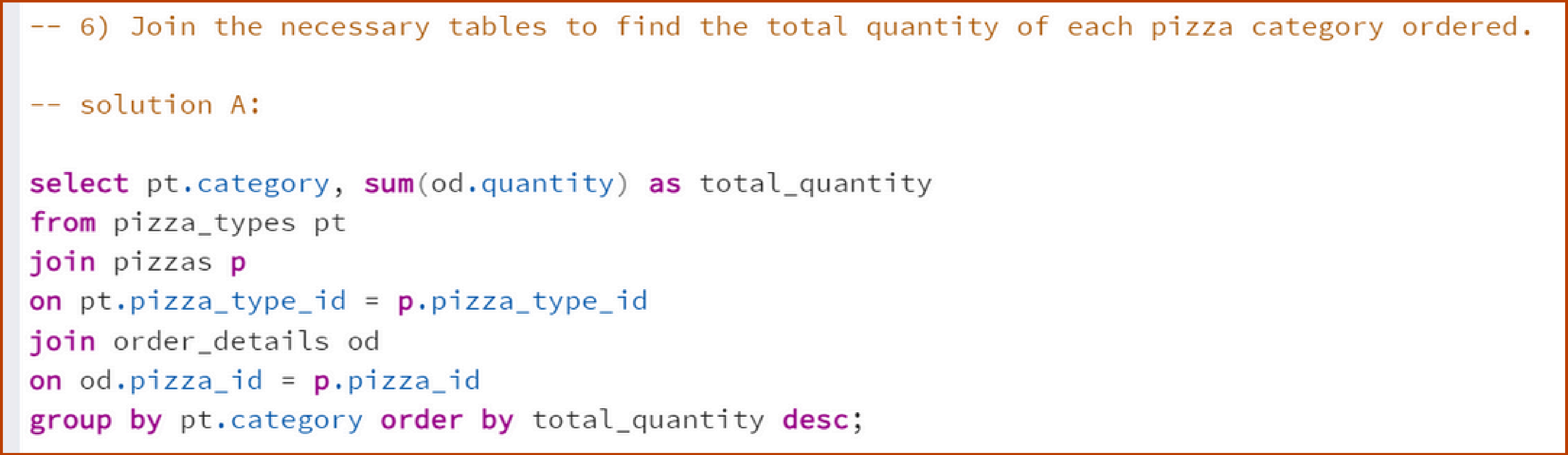
Output:



	name character varying (50)	category character varying (20)	size character varying (5)	total_quantity bigint
1	The Big Meat Pizza	Classic	S	1914
2	The Thai Chicken Pizza	Chicken	L	1410
3	The Five Cheese Pizza	Veggie	L	1409
4	The Four Cheese Pizza	Veggie	L	1316
5	The Classic Deluxe Pizza	Classic	M	1181

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

Query:

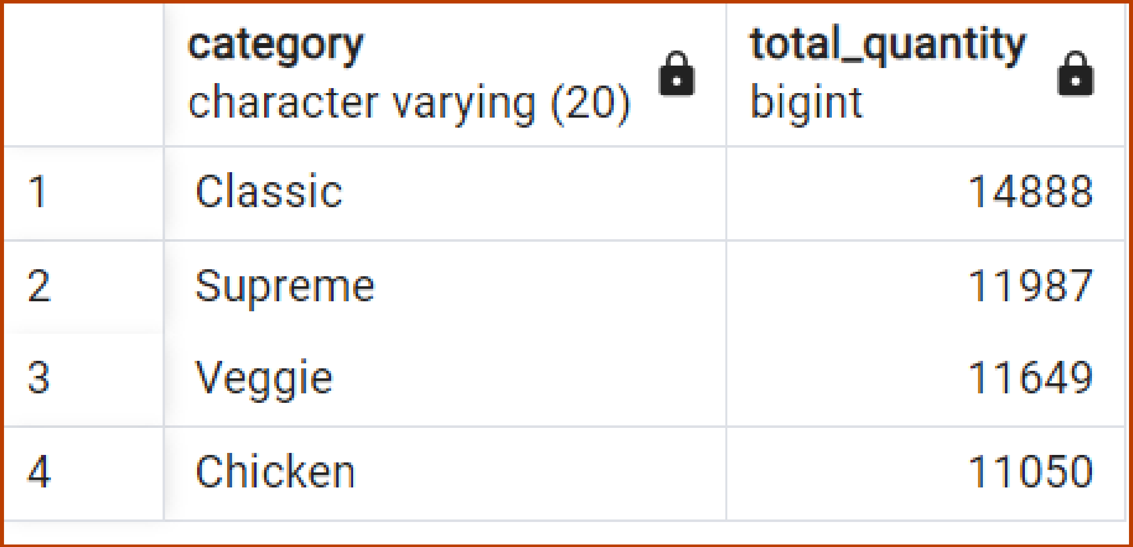


```
-- 6) Join the necessary tables to find the total quantity of each pizza category ordered.

-- solution A:

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

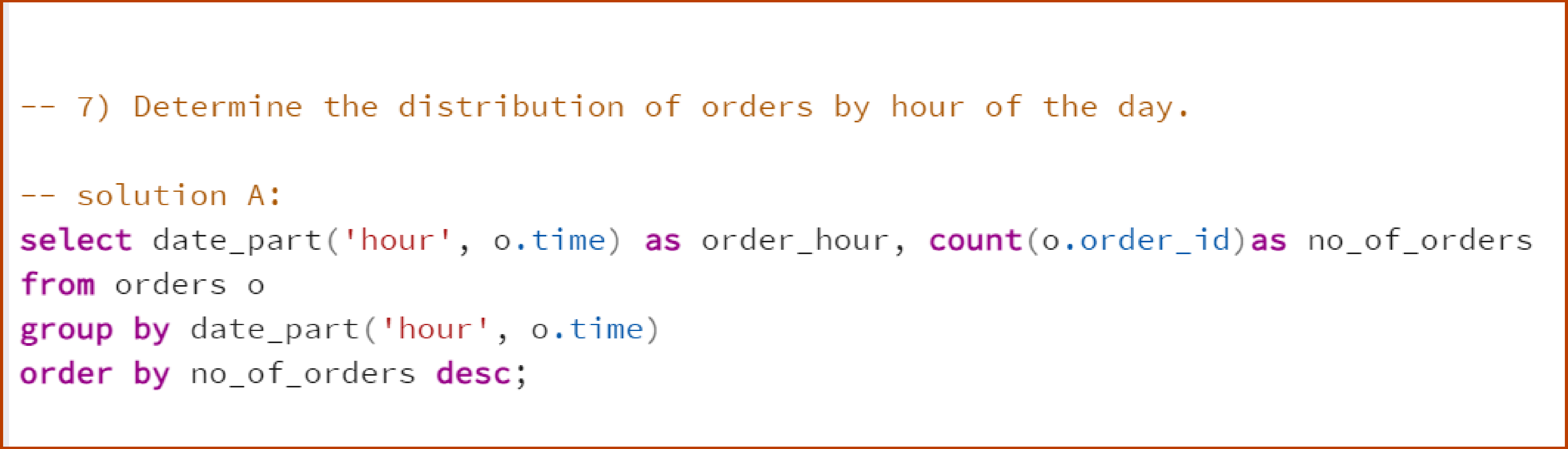
Output:



	category character varying (20)	total_quantity bigint
1	Classic	14888
2	Supreme	11987
3	Veggie	11649
4	Chicken	11050

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

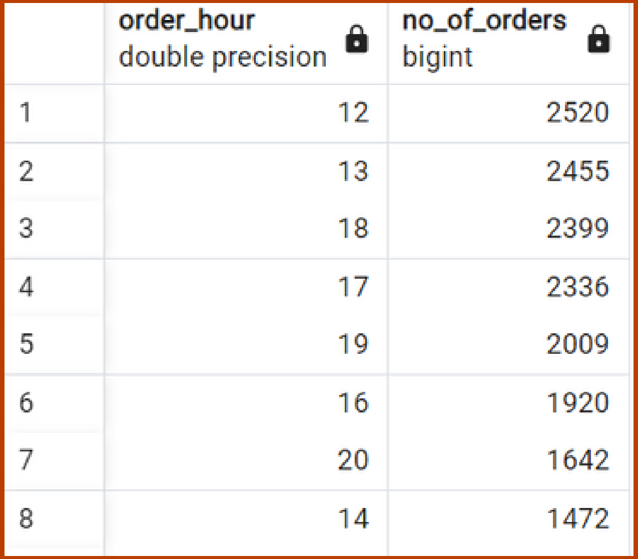
Query:



```
-- 7) Determine the distribution of orders by hour of the day.

-- solution A:
select date_part('hour', o.time) as order_hour, count(o.order_id)as no_of_orders
from orders o
group by date_part('hour', o.time)
order by no_of_orders desc;
```

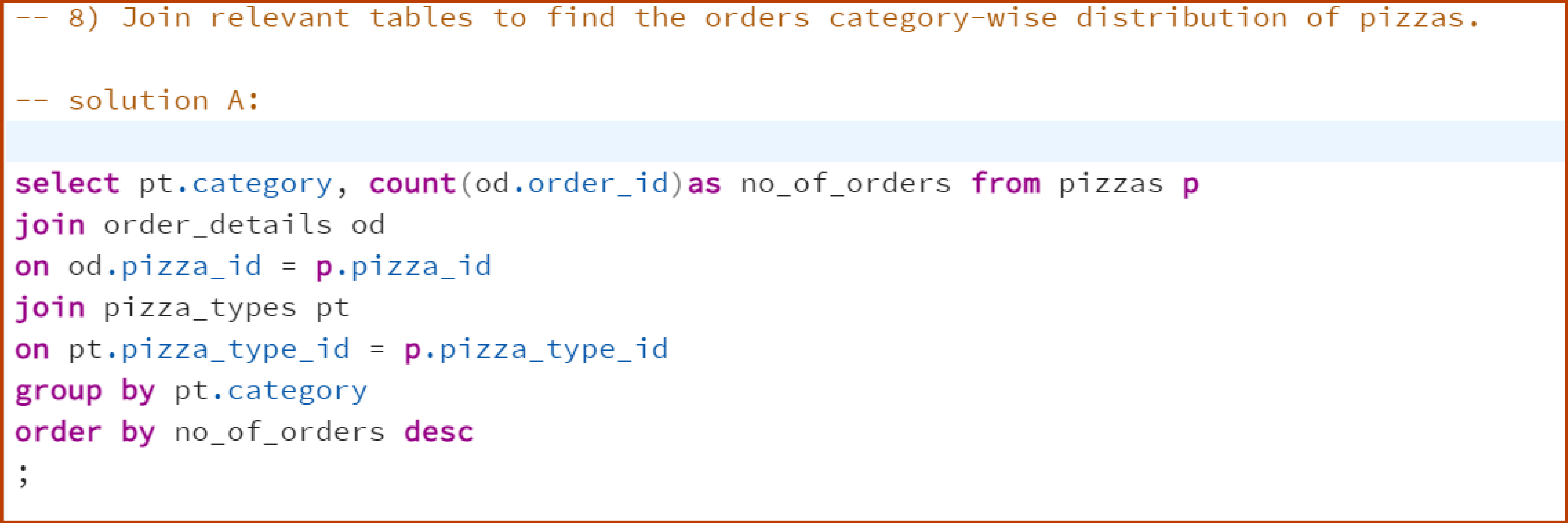
Output:



	order_hour double precision	no_of_orders bigint
1	12	2520
2	13	2455
3	18	2399
4	17	2336
5	19	2009
6	16	1920
7	20	1642
8	14	1472

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

Query:

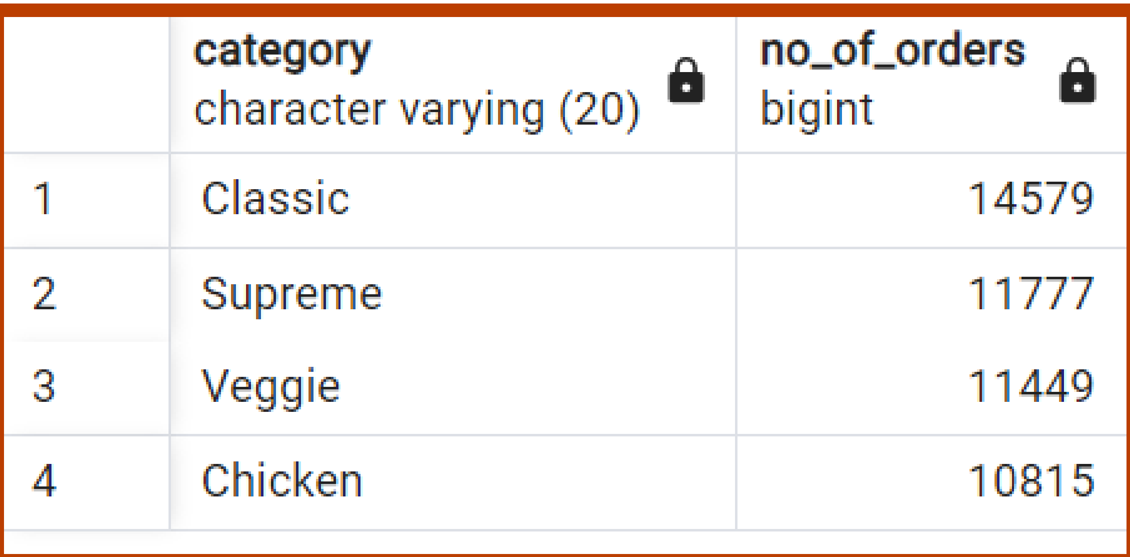


```
-- 8) Join relevant tables to find the orders category-wise distribution of pizzas.

-- solution A:

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

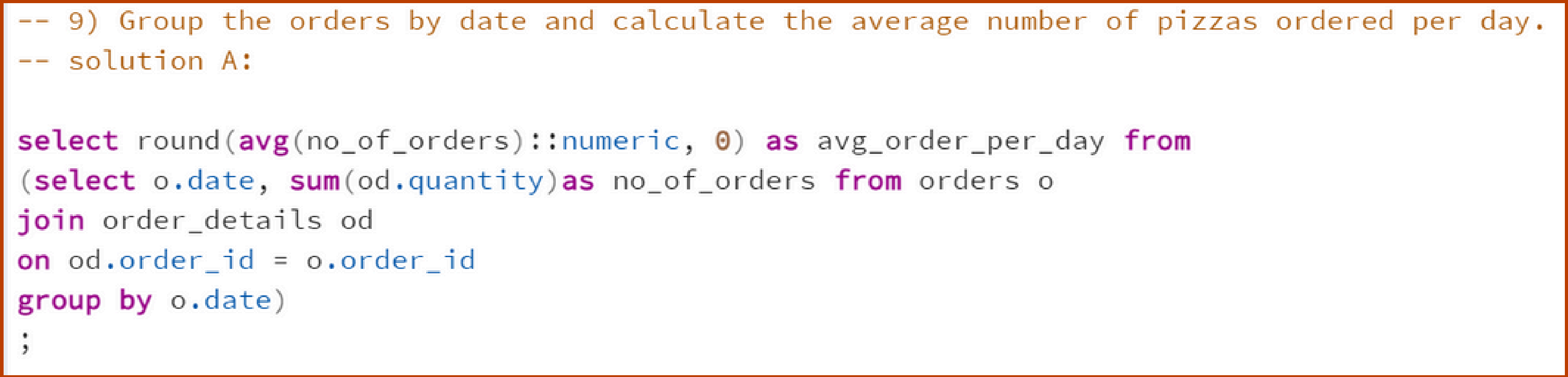
Output:



	category character varying (20)	no_of_orders bigint
1	Classic	14579
2	Supreme	11777
3	Veggie	11449
4	Chicken	10815

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

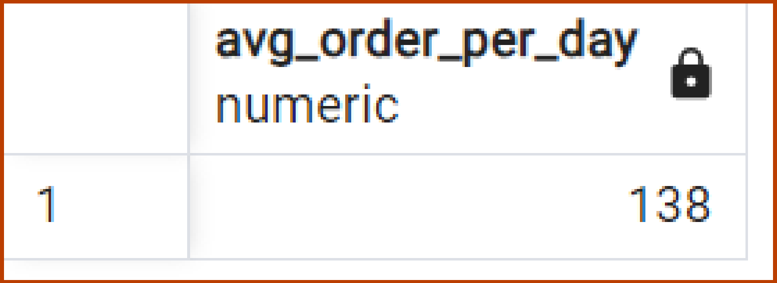
Query:



```
-- 9) Group the orders by date and calculate the average number of pizzas ordered per day.
-- solution A:

select round(avg(no_of_orders)::numeric, 0) as avg_order_per_day from
(select o.date, sum(od.quantity)as no_of_orders from orders o
join order_details od
on od.order_id = o.order_id
group by o.date)
;
```

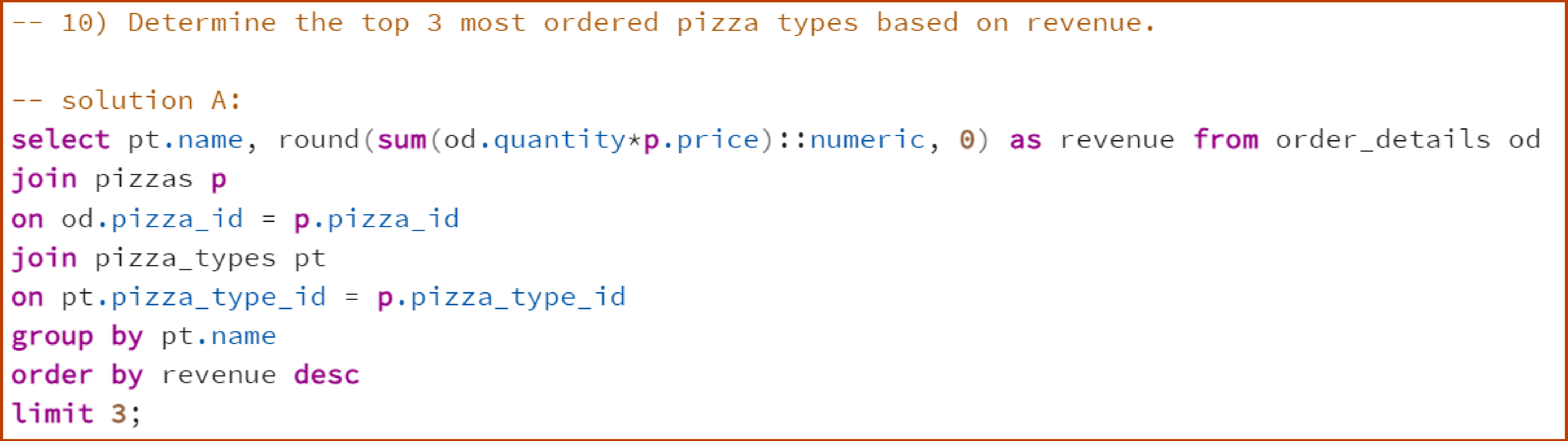
Output:



	avg_order_per_day numeric
1	138

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

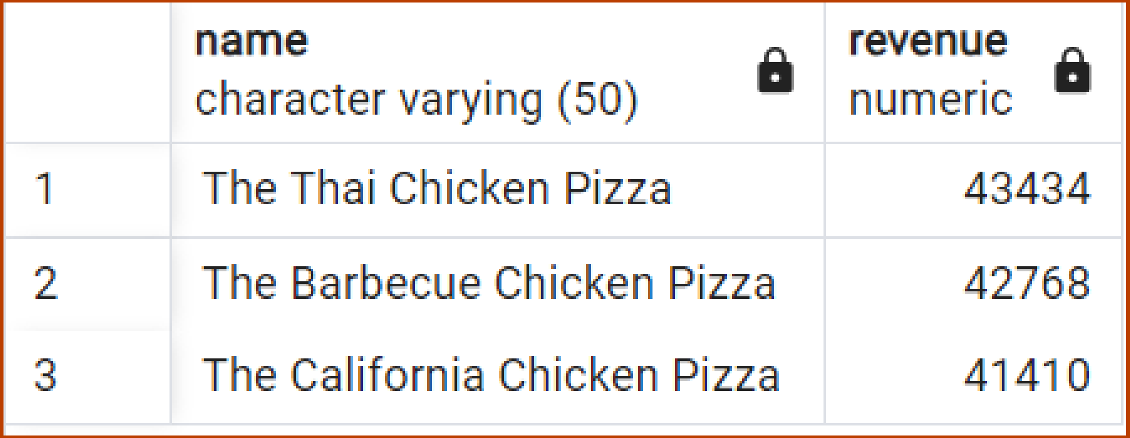
Query:



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

-- solution A:
select pt.name, round(sum(od.quantity*p.price)::numeric, 0) as revenue from order_details od
join pizzas p
on od.pizza_id = p.pizza_id
join pizza_types pt
on pt.pizza_type_id = p.pizza_type_id
group by pt.name
order by revenue desc
limit 3;
```

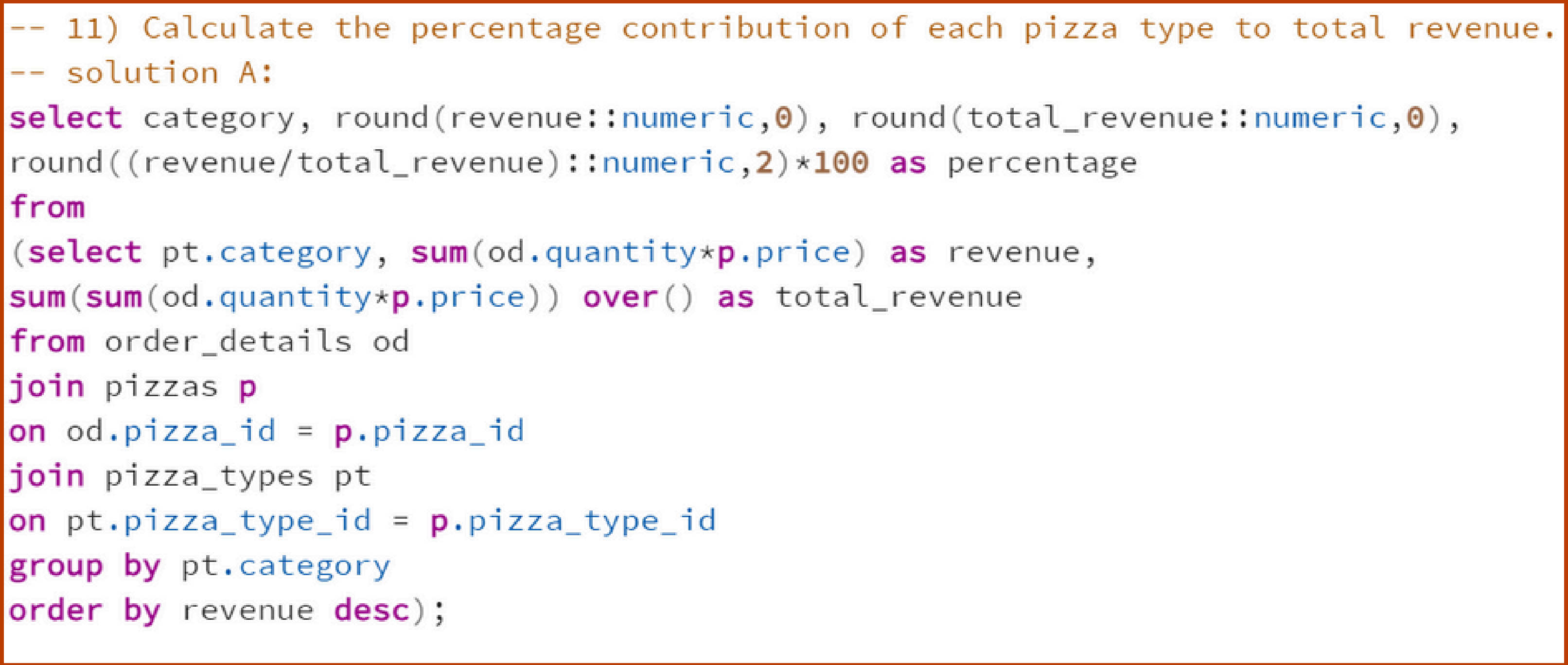
Output:



	name character varying (50)	revenue numeric
1	The Thai Chicken Pizza	43434
2	The Barbecue Chicken Pizza	42768
3	The California Chicken Pizza	41410

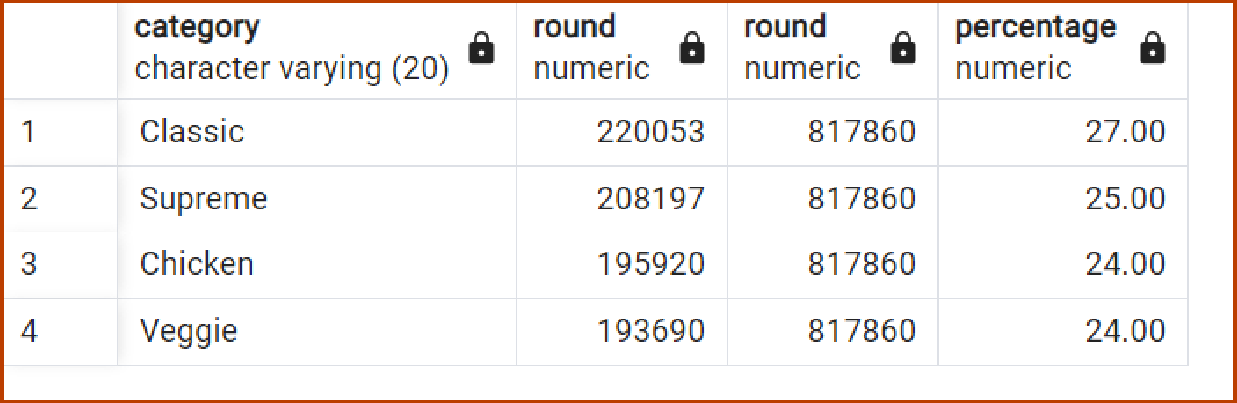
11) Calculate the percentage contribution of each pizza type to total revenue.

Query:



```
-- 11) Calculate the percentage contribution of each pizza type to total revenue.
-- solution A:
select category, round(revenue::numeric,0), round(total_revenue::numeric,0),
round((revenue/total_revenue)::numeric,2)*100 as percentage
from
(select pt.category, sum(od.quantity*p.price) as revenue,
sum(sum(od.quantity*p.price)) over() as total_revenue
from order_details od
join pizzas p
on od.pizza_id = p.pizza_id
join pizza_types pt
on pt.pizza_type_id = p.pizza_type_id
group by pt.category
order by revenue desc);
```

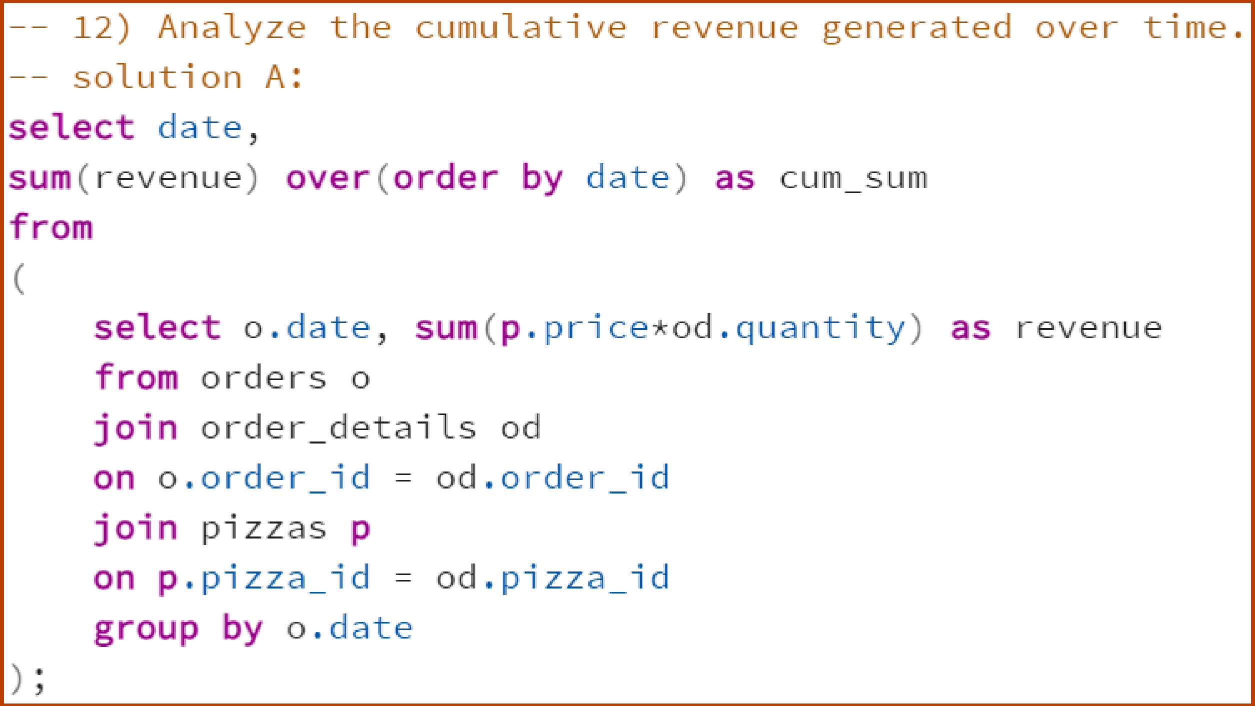
Output:



	category character varying (20)	round numeric	round numeric	percentage numeric
1	Classic	220053	817860	27.00
2	Supreme	208197	817860	25.00
3	Chicken	195920	817860	24.00
4	Veggie	193690	817860	24.00

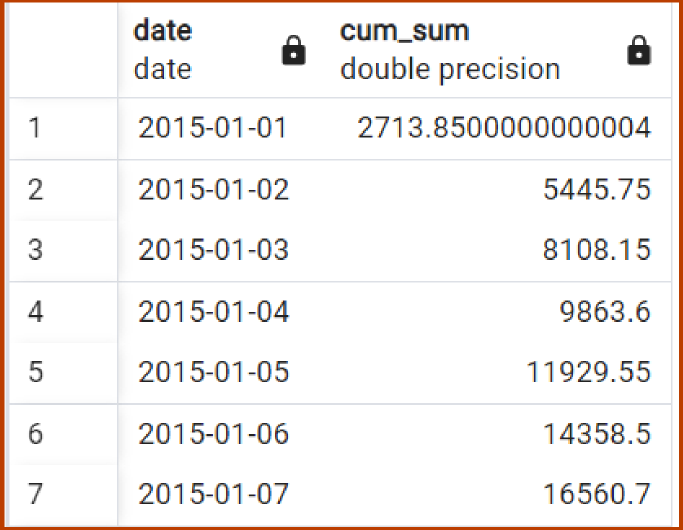
12) Analyze the cumulative revenue generated over time.

Query:



```
-- 12) Analyze the cumulative revenue generated over time.
-- solution A:
select date,
sum(revenue) over(order by date) as cum_sum
from
(
    select o.date, sum(p.price*od.quantity) as revenue
    from orders o
    join order_details od
    on o.order_id = od.order_id
    join pizzas p
    on p.pizza_id = od.pizza_id
    group by o.date
);
```

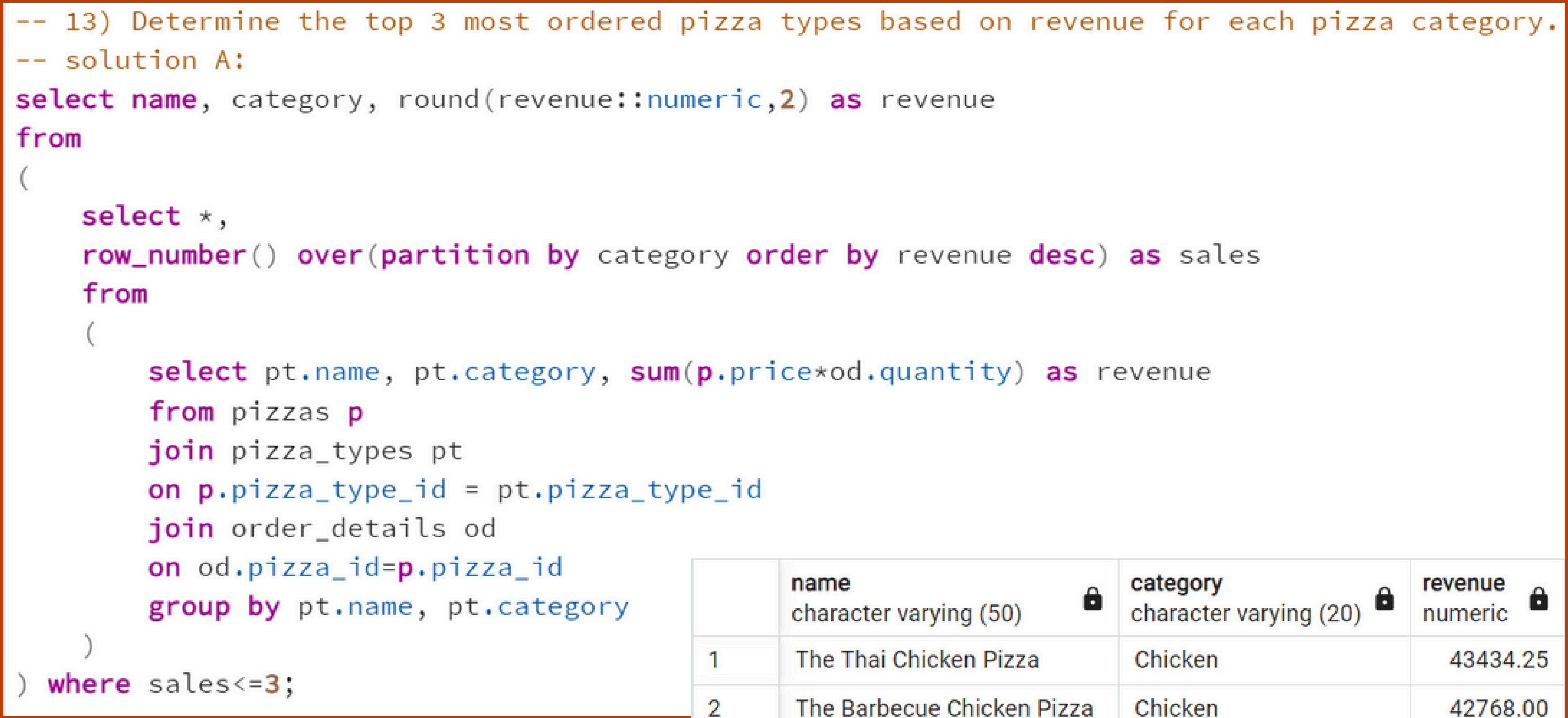
Output:



	date date	cum_sum double precision
1	2015-01-01	2713.8500000000004
2	2015-01-02	5445.75
3	2015-01-03	8108.15
4	2015-01-04	9863.6
5	2015-01-05	11929.55
6	2015-01-06	14358.5
7	2015-01-07	16560.7

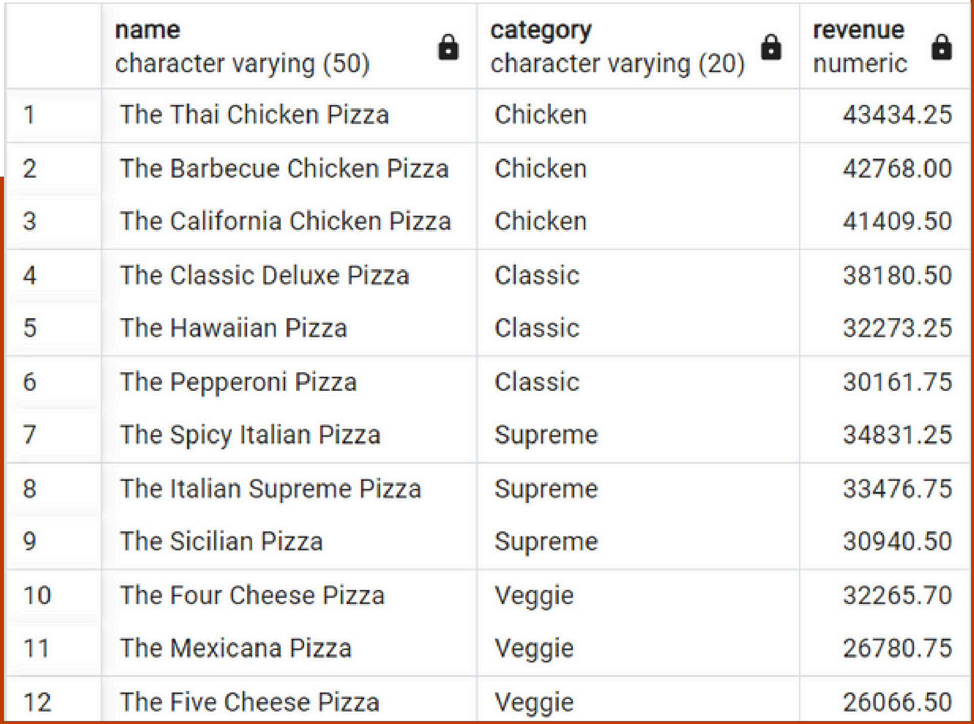
13) Determine the top 3 most ordered pizza types based on revenue for each pizza category.

Query:



```
-- 13) Determine the top 3 most ordered pizza types based on revenue for each pizza category.
-- solution A:
select name, category, round(revenue::numeric,2) as revenue
from
(
    select *,
    row_number() over(partition by category order by revenue desc) as sales
    from
    (
        select pt.name, pt.category, sum(p.price*od.quantity) as revenue
        from pizzas p
        join pizza_types pt
        on p.pizza_type_id = pt.pizza_type_id
        join order_details od
        on od.pizza_id=p.pizza_id
        group by pt.name, pt.category
    )
) where sales<=3;
```

Output:



	name character varying (50)	category character varying (20)	revenue numeric
1	The Thai Chicken Pizza	Chicken	43434.25
2	The Barbecue Chicken Pizza	Chicken	42768.00
3	The California Chicken Pizza	Chicken	41409.50
4	The Classic Deluxe Pizza	Classic	38180.50
5	The Hawaiian Pizza	Classic	32273.25
6	The Pepperoni Pizza	Classic	30161.75
7	The Spicy Italian Pizza	Supreme	34831.25
8	The Italian Supreme Pizza	Supreme	33476.75
9	The Sicilian Pizza	Supreme	30940.50
10	The Four Cheese Pizza	Veggie	32265.70
11	The Mexicana Pizza	Veggie	26780.75
12	The Five Cheese Pizza	Veggie	26066.50

Thank you