Domino’s

PIZZA DATA ANALYSIS

Arup Ranjan Dalai

CONTENT

Basic:

01 Retrieve the total number of orders placed.

02 Calculate the total revenue generated from pizza sales.

03 Identify the highest-priced pizza.

04 Identify the most common pizza size ordered.

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

Intermediate:

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

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

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

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

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

Advanced:

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

02 Analyze the cumulative revenue generated over time.

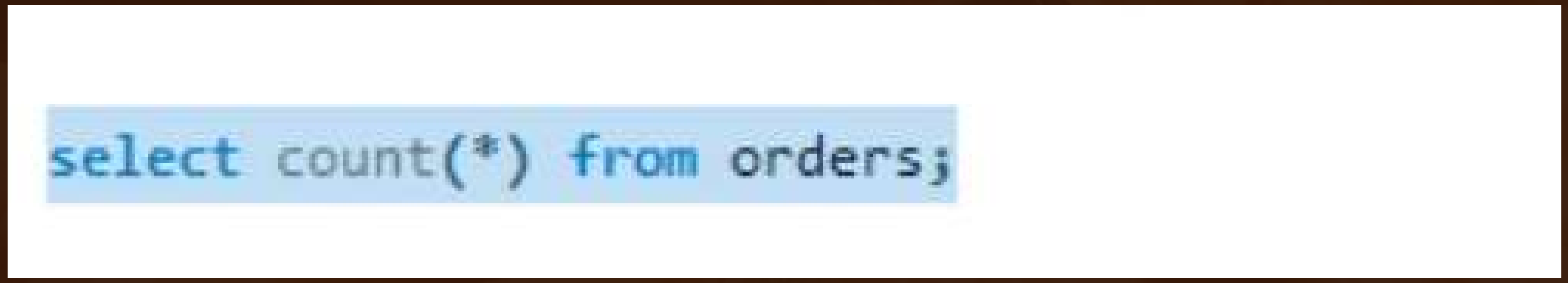
03 Determine the top 3 most ordered pizza types based on revenue for each pizza category.

01

BASIC

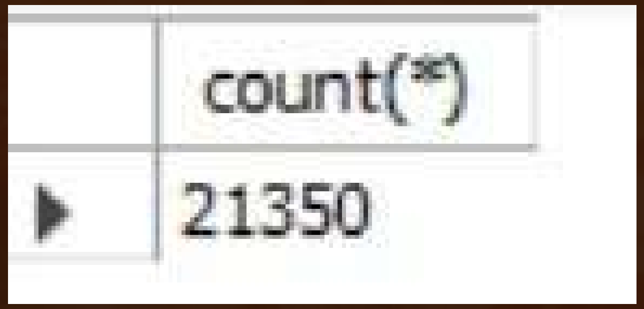
Retrieve the total number of orders placed.

code



```sql
select count(*) from orders;
```

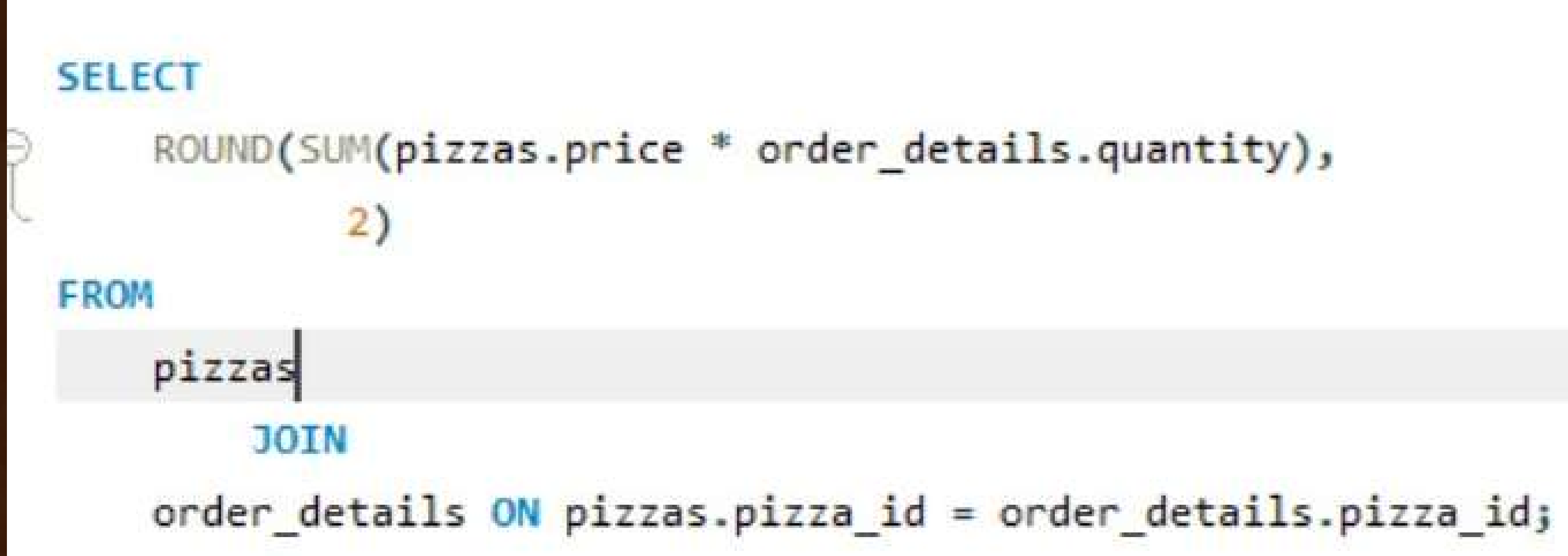
Output



```
count(*)
21350
```

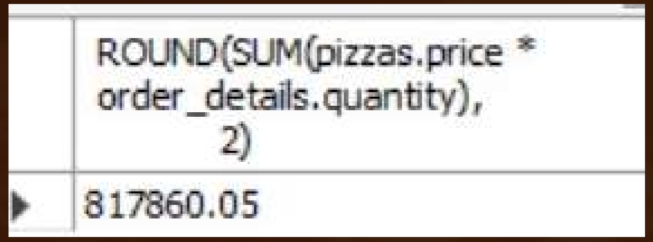
Calculate the total revenue generated from pizza sales.

code



```sql
SELECT
    ROUND(SUM(pizzas.price * order_details.quantity),
            2)
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id;
```

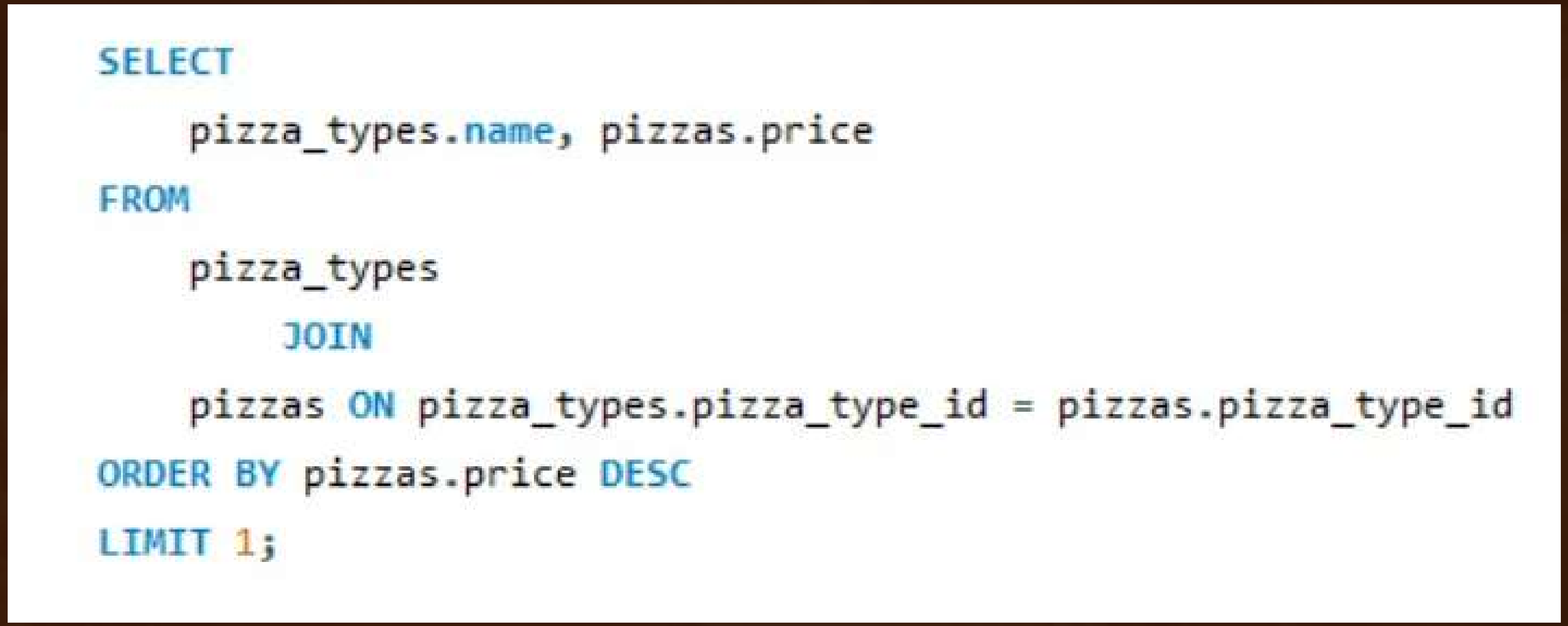
Output



```
ROUND(SUM(pizzas.price *
order_details.quantity),
        2)
817860.05
```

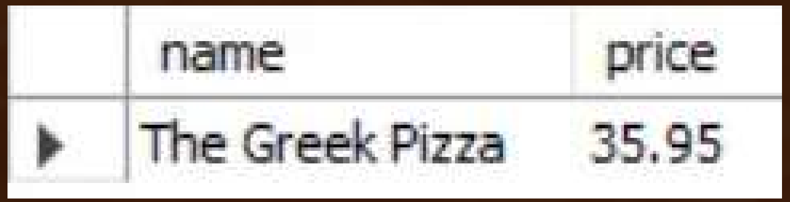
Identify the highest-priced pizza.

code



```
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;
```

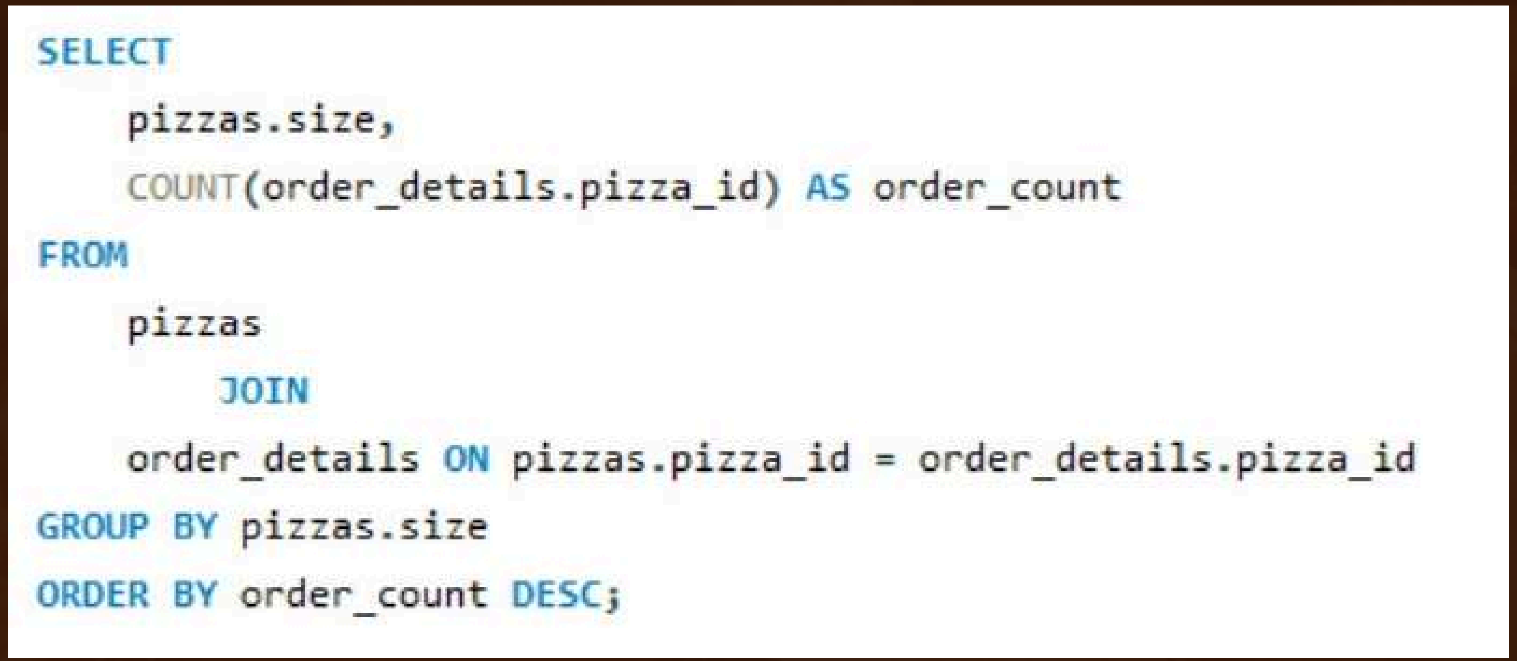
Output



name	price
The Greek Pizza	35.95

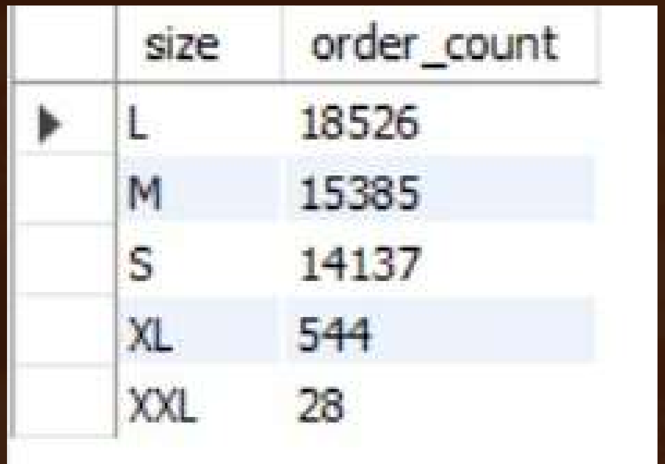
Identify the most common pizza size ordered.

code



```
SELECT
    pizzas.size,
    COUNT(order_details.pizza_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;
```

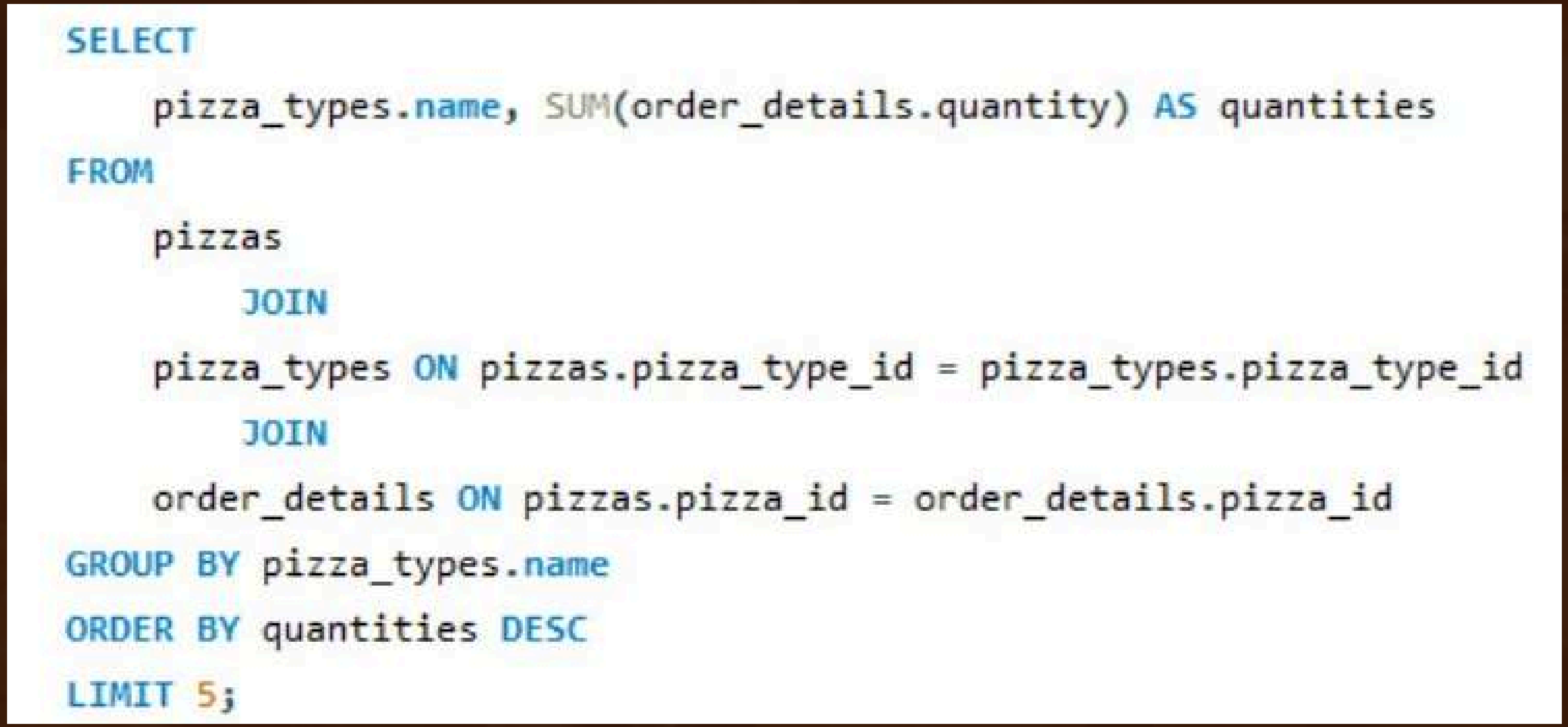
Output



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

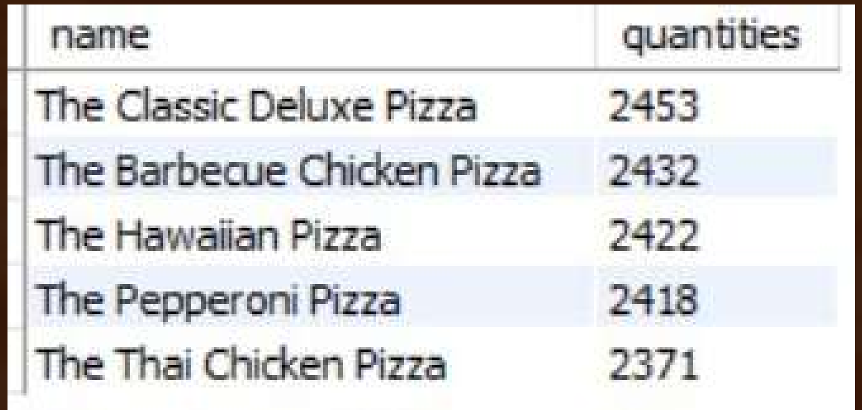
List the top 5 most ordered pizza types along with their quantities.

code



```
SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantities
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY quantities DESC
LIMIT 5;
```

Output



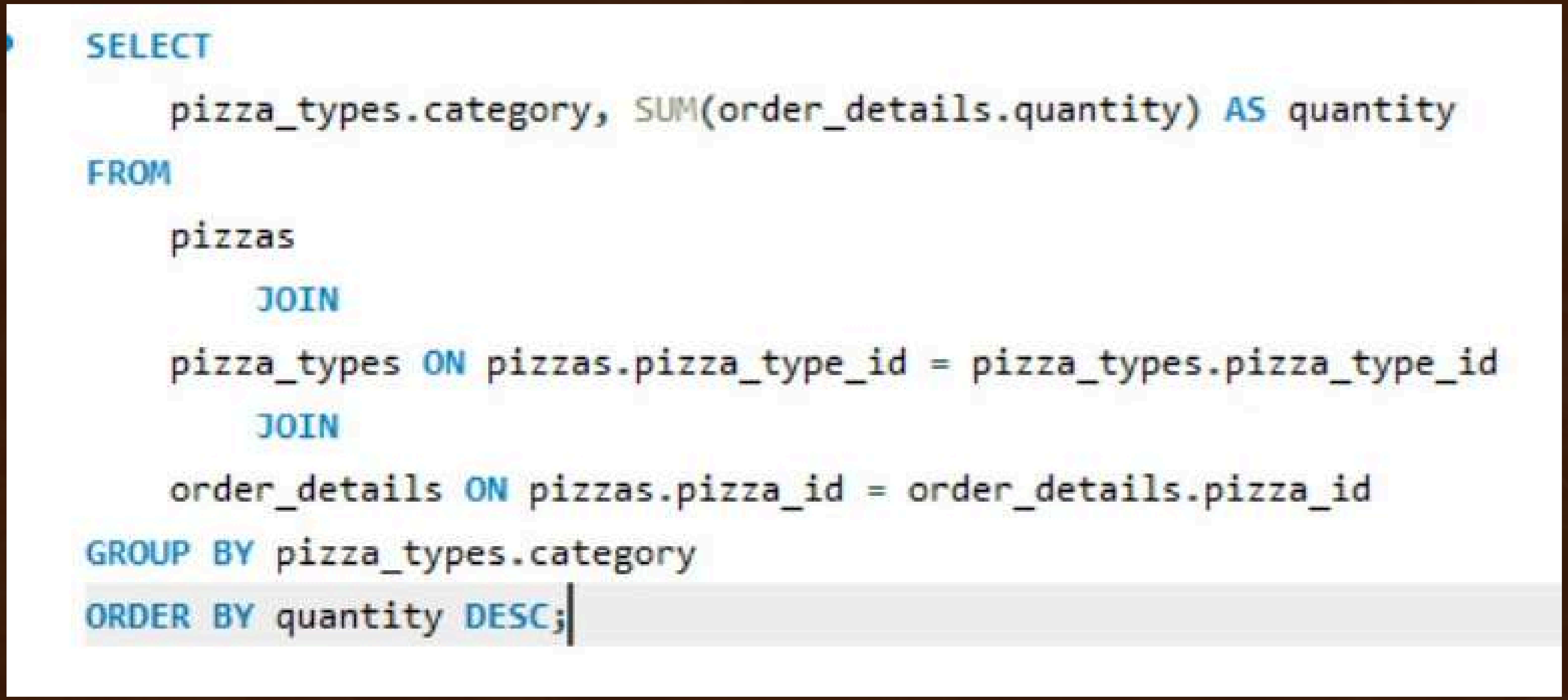
name	quantities
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

01

INTERMEDIATE

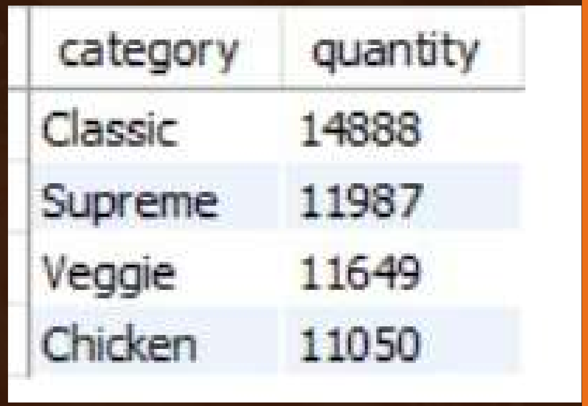
Join the necessary tables to find the total quantity of each pizza category ordered.

code



```
SELECT
    pizza_types.category, SUM(order_details.quantity) AS quantity
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity DESC;
```

Output

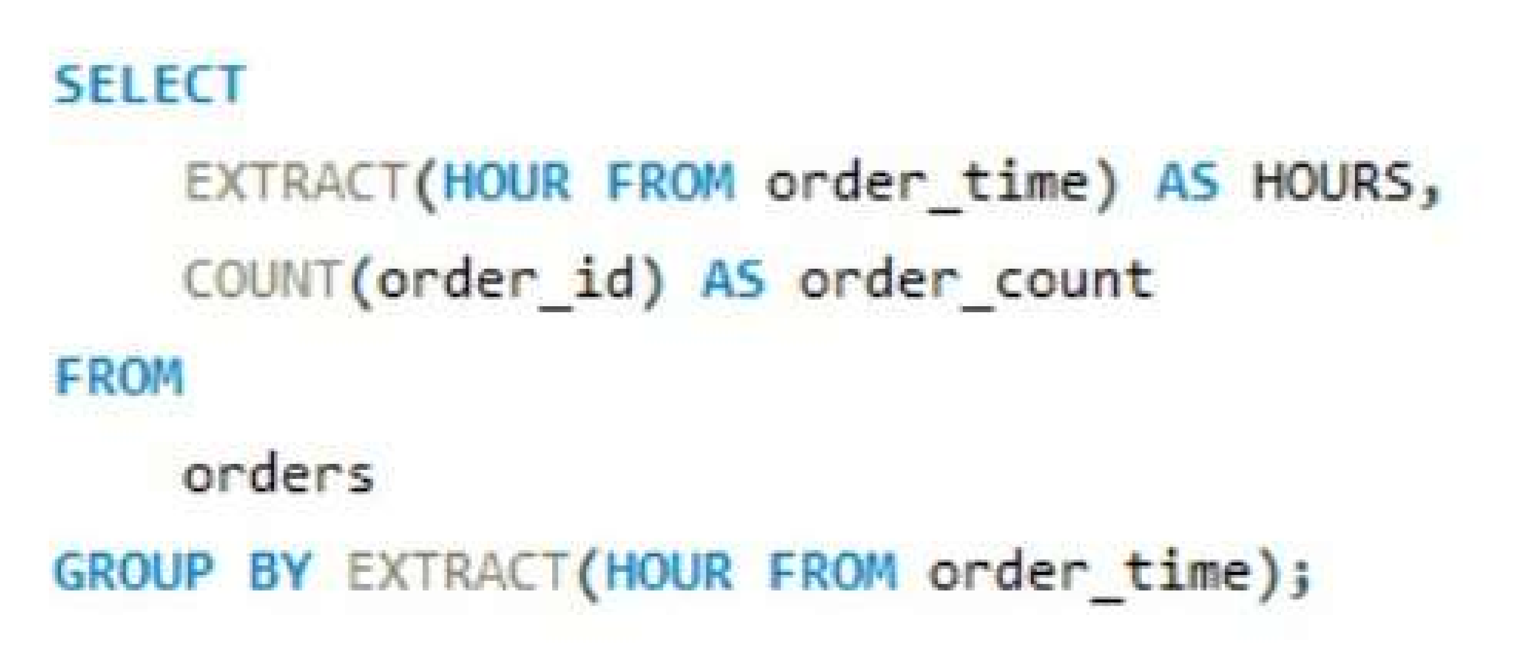


category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

02

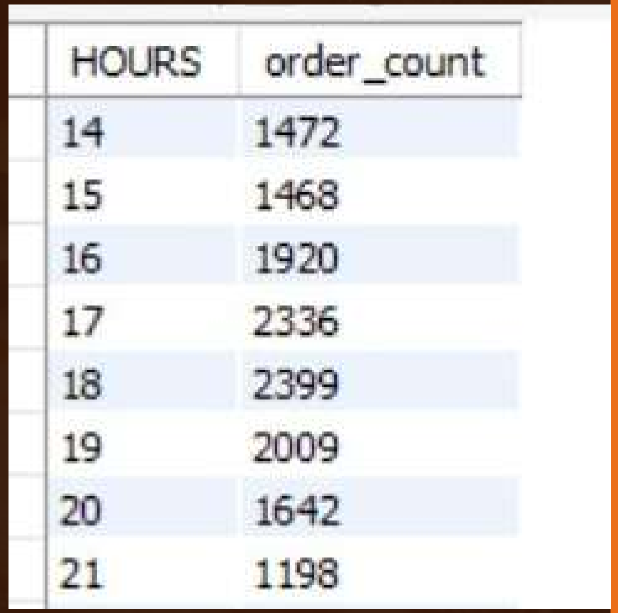
Determine the distribution of orders by hour of the day.

code



```
SELECT
    EXTRACT(HOUR FROM order_time) AS HOURS,
    COUNT(order_id) AS order_count
FROM
    orders
GROUP BY EXTRACT(HOUR FROM order_time);
```

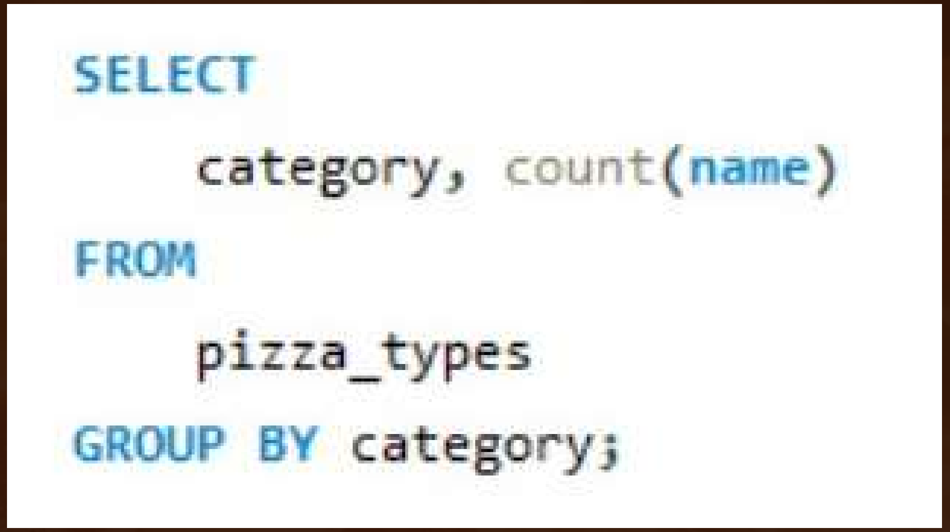
Output



HOURS	order_count
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198

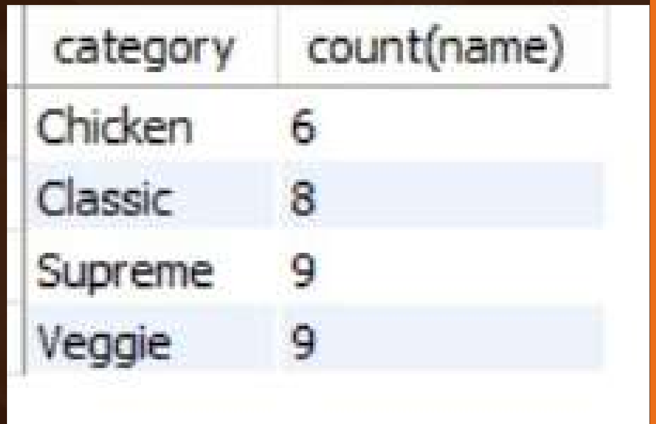
Join relevant tables to find the category-wise distribution of pizzas.

code



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

Output

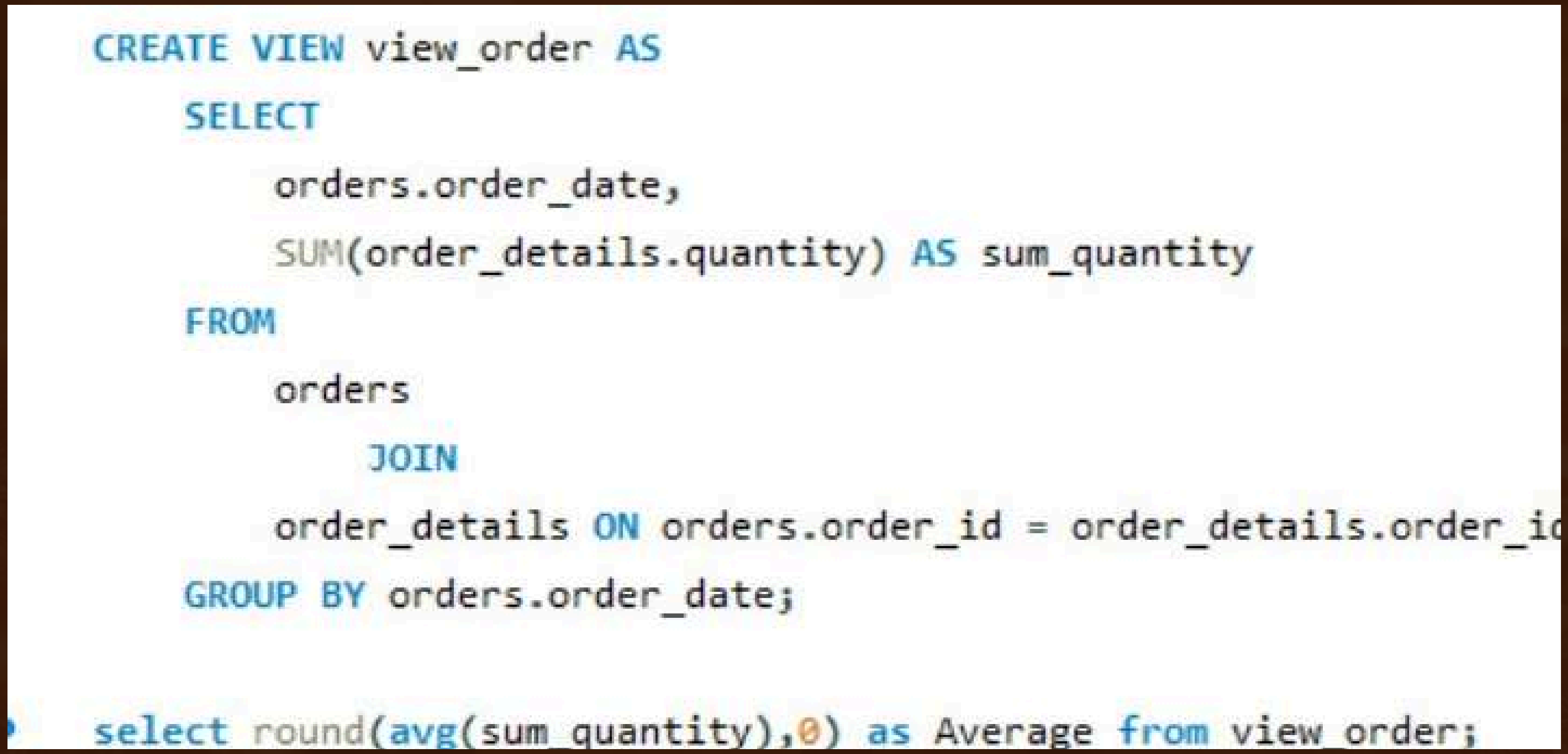


category	count(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

04

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

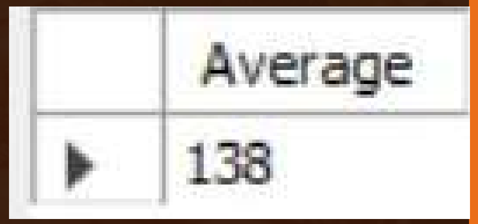
code



```
CREATE VIEW view_order AS
    SELECT
        orders.order_date,
        SUM(order_details.quantity) AS sum_quantity
    FROM
        orders
            JOIN
        order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date;

select round(avg(sum_quantity),0) as Average from view_order;
```

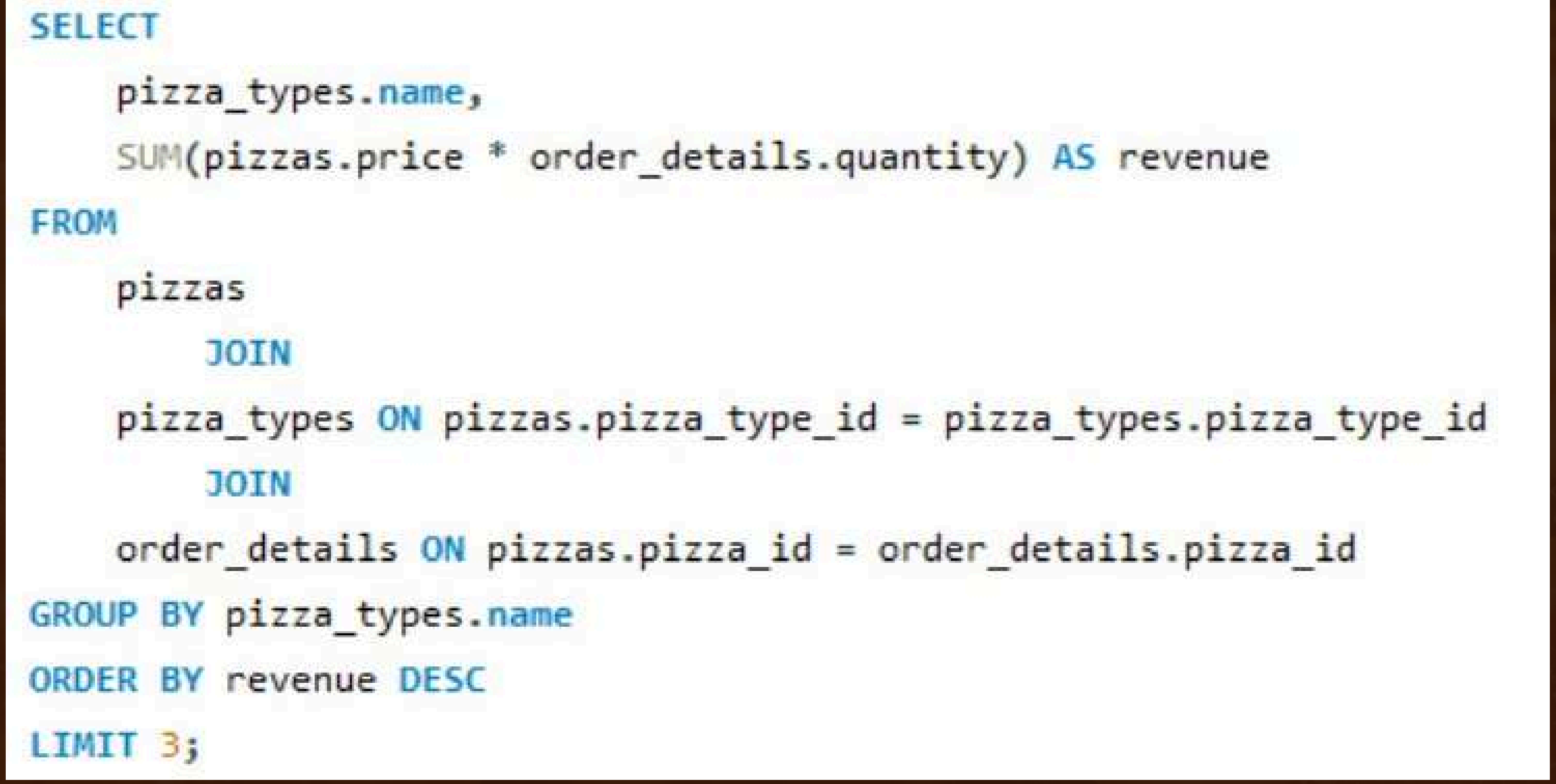
Output



```
  Average
▶ 138
```

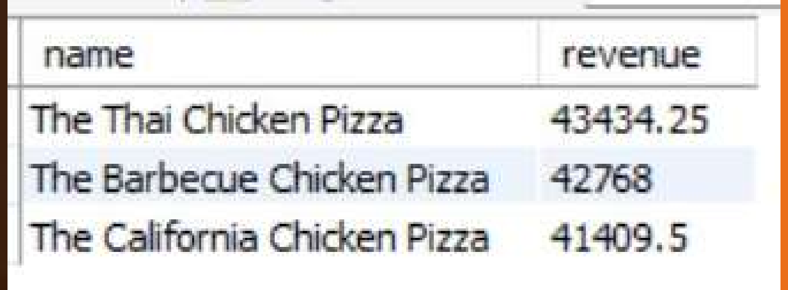
Determine the top 3 most ordered pizza types based on revenue.

code



```
SELECT
    pizza_types.name,
    SUM(pizzas.price * order_details.quantity) AS revenue
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```

Output



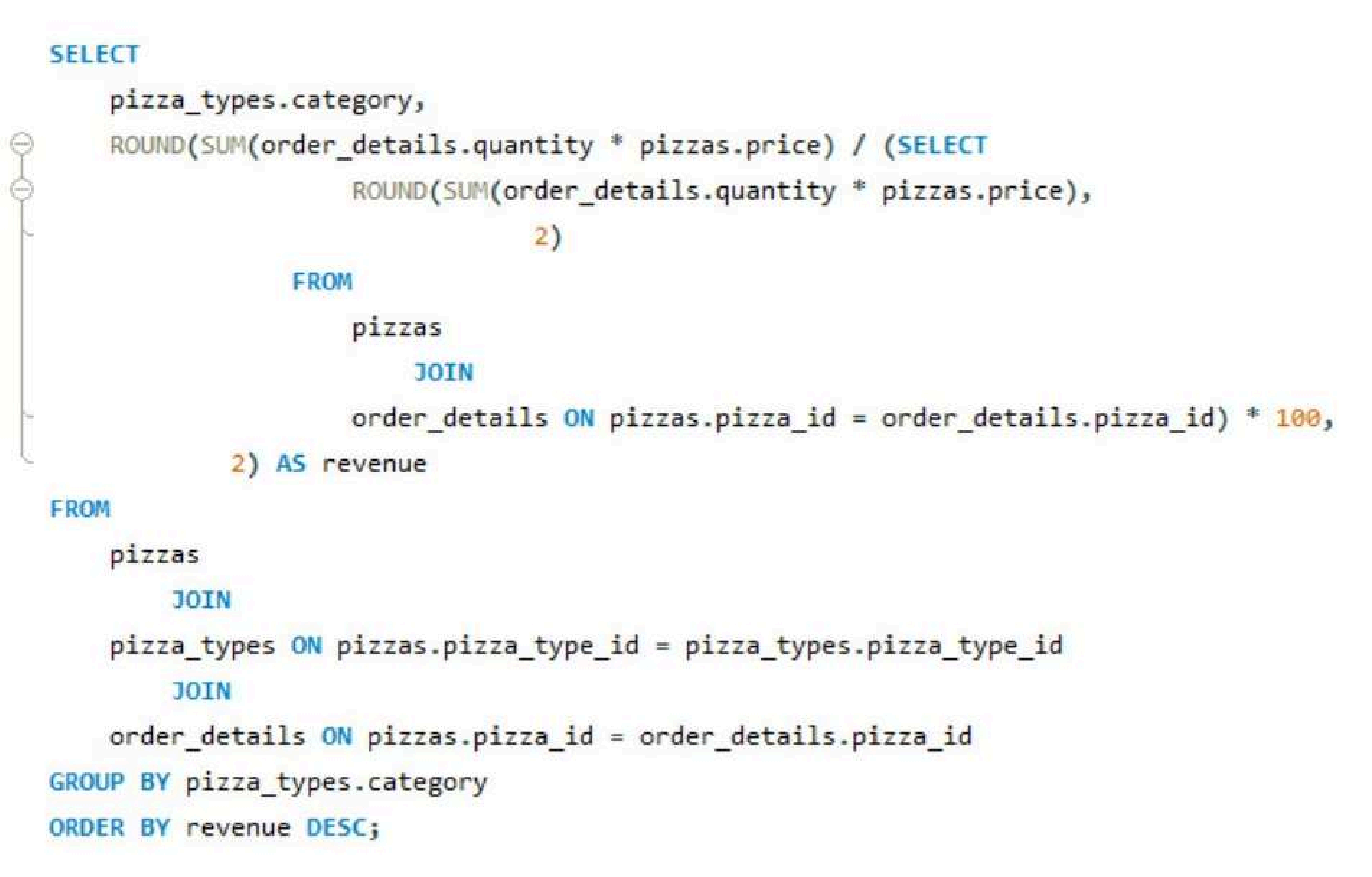
name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

01

ADVANCED

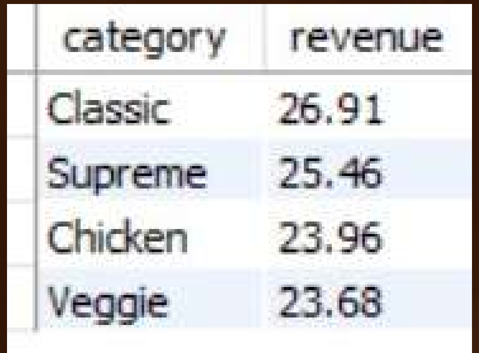
Calculate the percentage contribution of each pizza type to total revenue.

code



```
SELECT
    pizza_types.category,
    ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
                ROUND(SUM(order_details.quantity * pizzas.price),
                            2)
            FROM
                pizzas
                    JOIN
                order_details ON pizzas.pizza_id = order_details.pizza_id) * 100,
            2) AS revenue
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC;
```

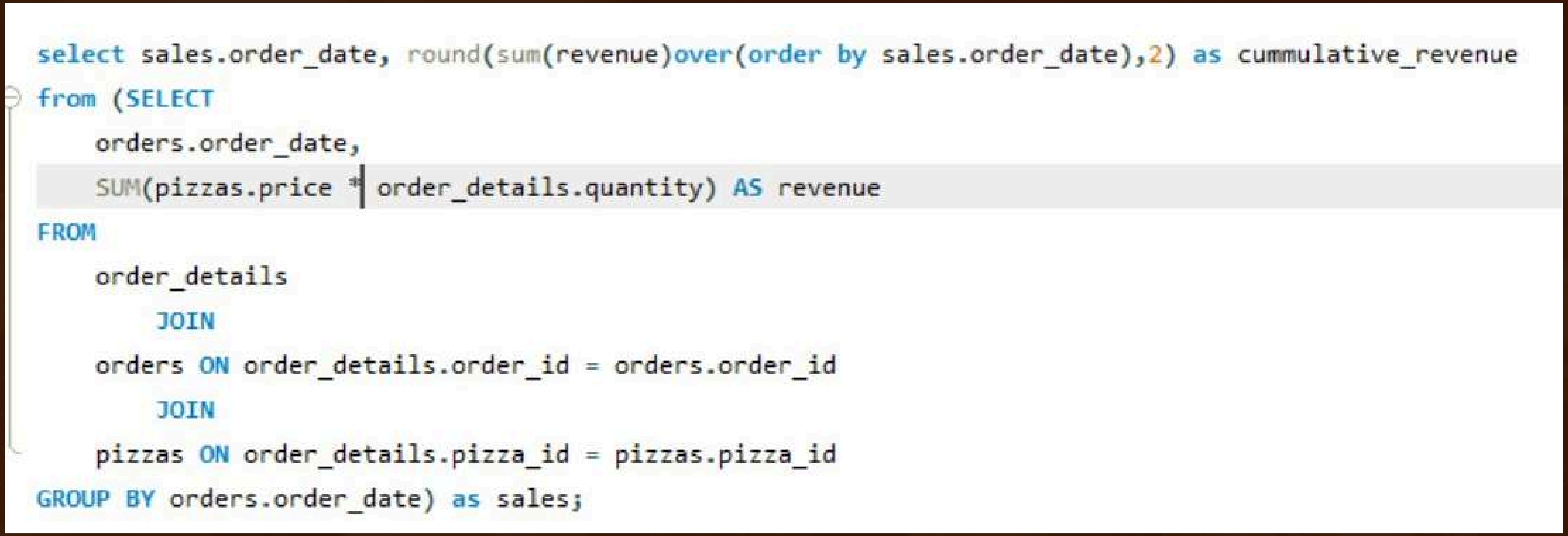
Output



category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

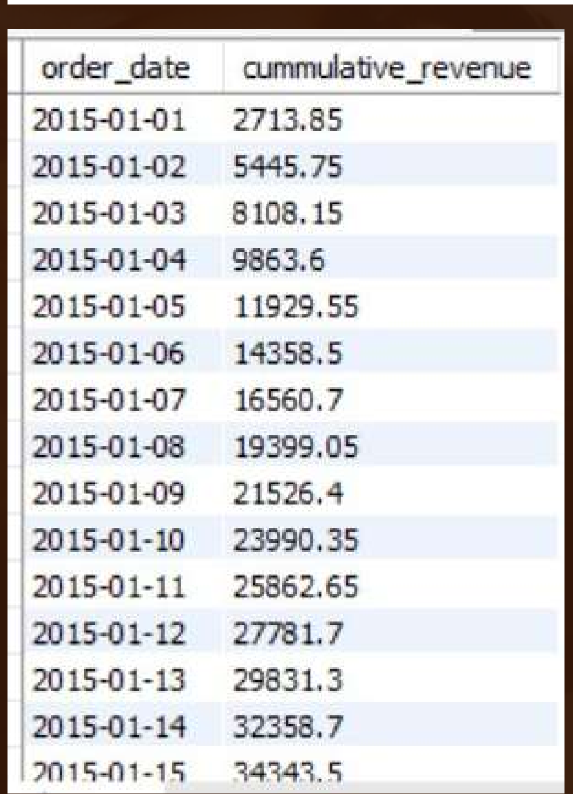
Analyze the cumulative revenue generated over time.

code



```
select sales.order_date, round(sum(revenue)over(order by sales.order_date),2) as cummulative_revenue
from (SELECT
    orders.order_date,
    SUM(pizzas.price * order_details.quantity) AS revenue
FROM
    order_details
        JOIN
    orders ON order_details.order_id = orders.order_id
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id
GROUP BY orders.order_date) as sales;
```

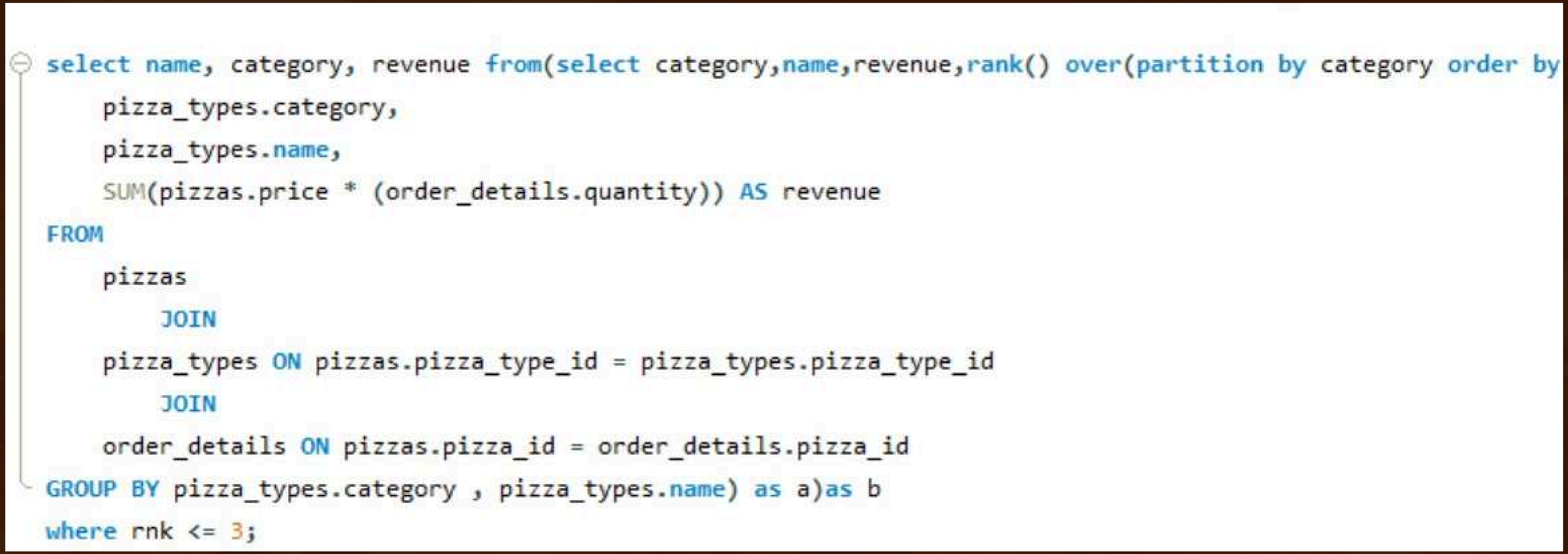
Output



```
order_date   cummulative_revenue
2015-01-01   2713.85
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   23990.35
2015-01-11   25862.65
2015-01-12   27781.7
2015-01-13   29831.3
2015-01-14   32358.7
2015-01-15   34343.5
```

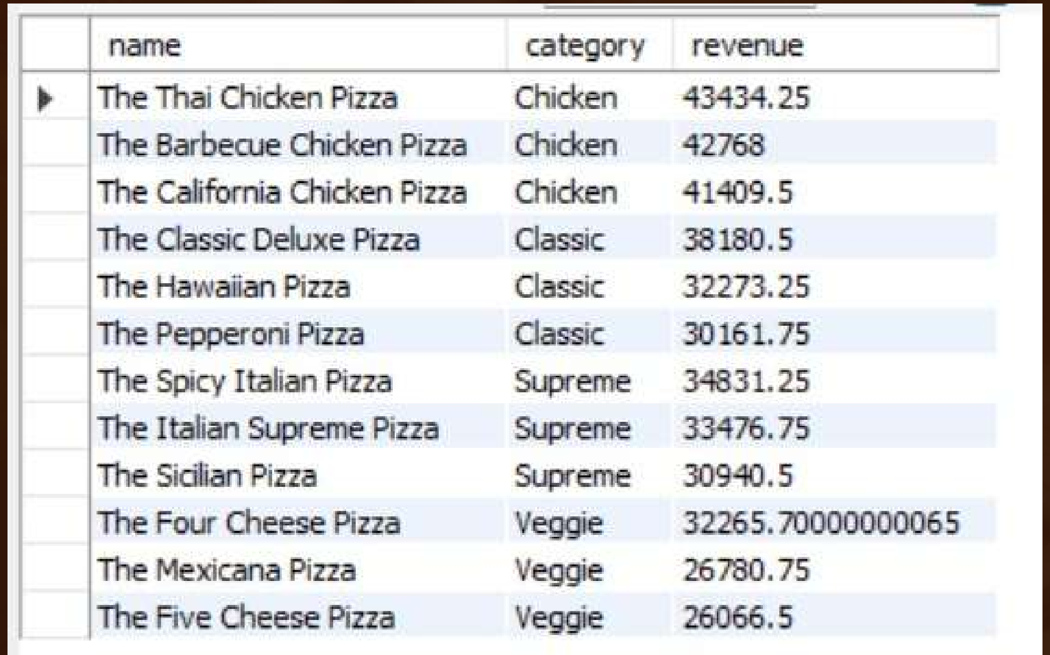
Determine the top 3 most ordered pizza types based on revenue for each pizza category.

code



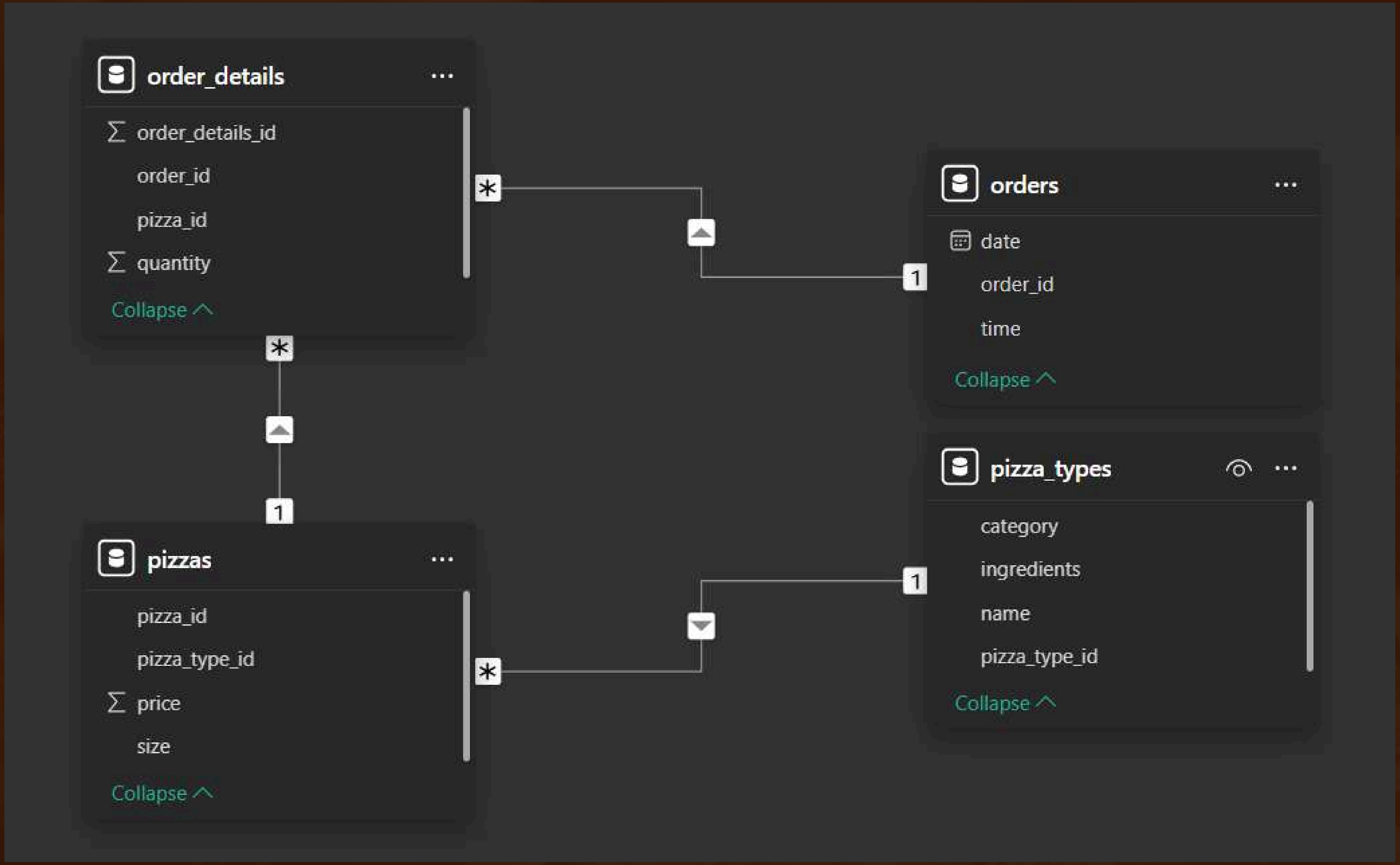
```
select name, category, revenue from(select category,name,revenue,rank() over(partition by category order by
    pizza_types.category,
    pizza_types.name,
    SUM(pizzas.price * (order_details.quantity)) AS revenue
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category , pizza_types.name) as a)as b
where rnk <= 3;
```

Output



name	category	revenue
The Thai Chicken Pizza	Chicken	43434.25
The Barbecue Chicken Pizza	Chicken	42768
The California Chicken Pizza	Chicken	41409.5
The Classic Deluxe Pizza	Classic	38180.5
The Hawaiian Pizza	Classic	32273.25
The Pepperoni Pizza	Classic	30161.75
The Spicy Italian Pizza	Supreme	34831.25
The Italian Supreme Pizza	Supreme	33476.75
The Sicilian Pizza	Supreme	30940.5
The Four Cheese Pizza	Veggie	32265.70000000065
The Mexicana Pizza	Veggie	26780.75
The Five Cheese Pizza	Veggie	26066.5

ER DIGRAM



order_details
∑ order_details_id
order_id
pizza_id
∑ quantity
Collapse
orders
date
order_id
time
Collapse
pizzas
pizza_id
pizza_type_id
∑ price
size
Collapse
pizza_types
category
ingredients
name
pizza_type_id
Collapse
*
1
*
1
*
1

Domino's Pizza Data Analysis

THANK YOU FOR ATTENTION