By VIJAY SINGH PARMAR

SQL PROJECT PIZZA SALES

VIEW NOW

Start Your VISIT

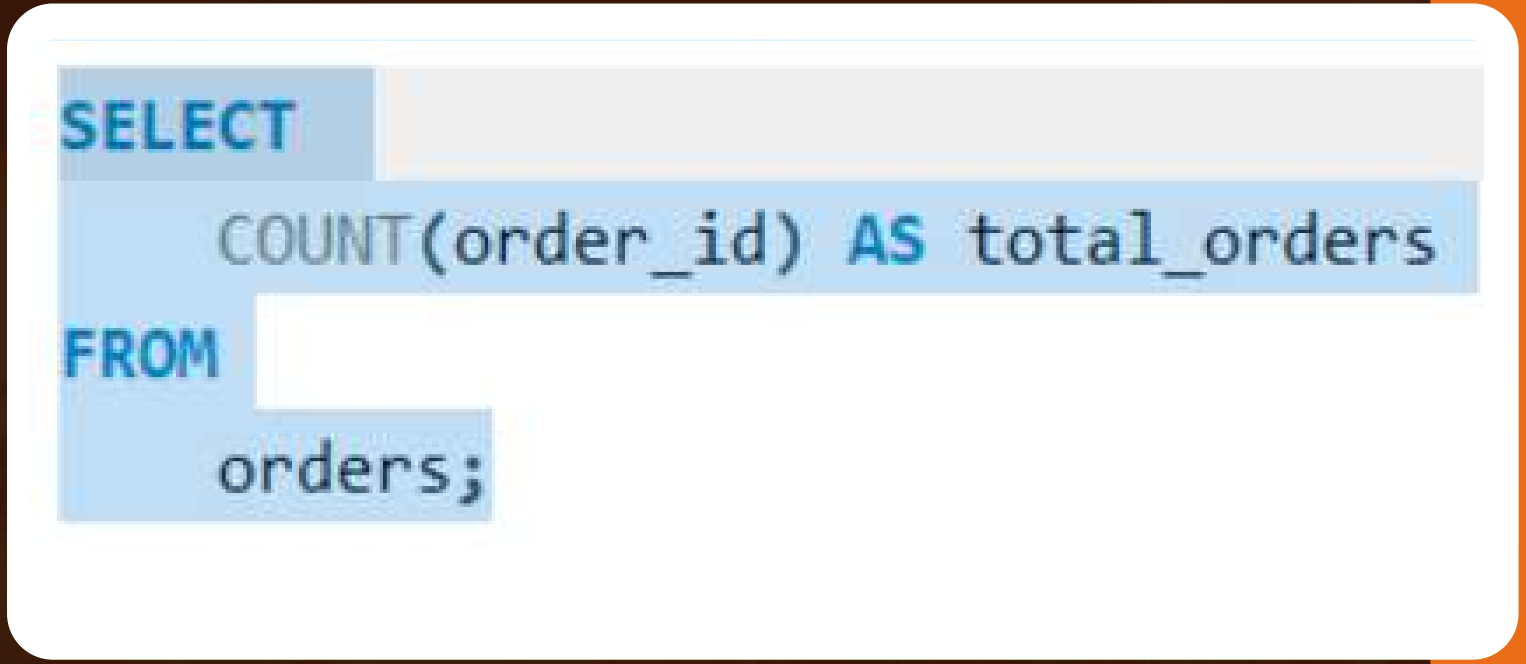
ABOUT ME

VIJAY SINGH PARMAR
DATA ANALYST

I am Vijay Singh Parmar, a passionate and detail-oriented Data Analyst with a strong foundation in data analysis, SQL, and problem-solving. I specialize in turning raw data into meaningful insights that drive informed business decisions.
My Pizza Sales Analysis project highlights my ability to extract insights, analyze trends, and deliver actionable solutions to support business decisions. I specialize in turning data into impactful strategies.

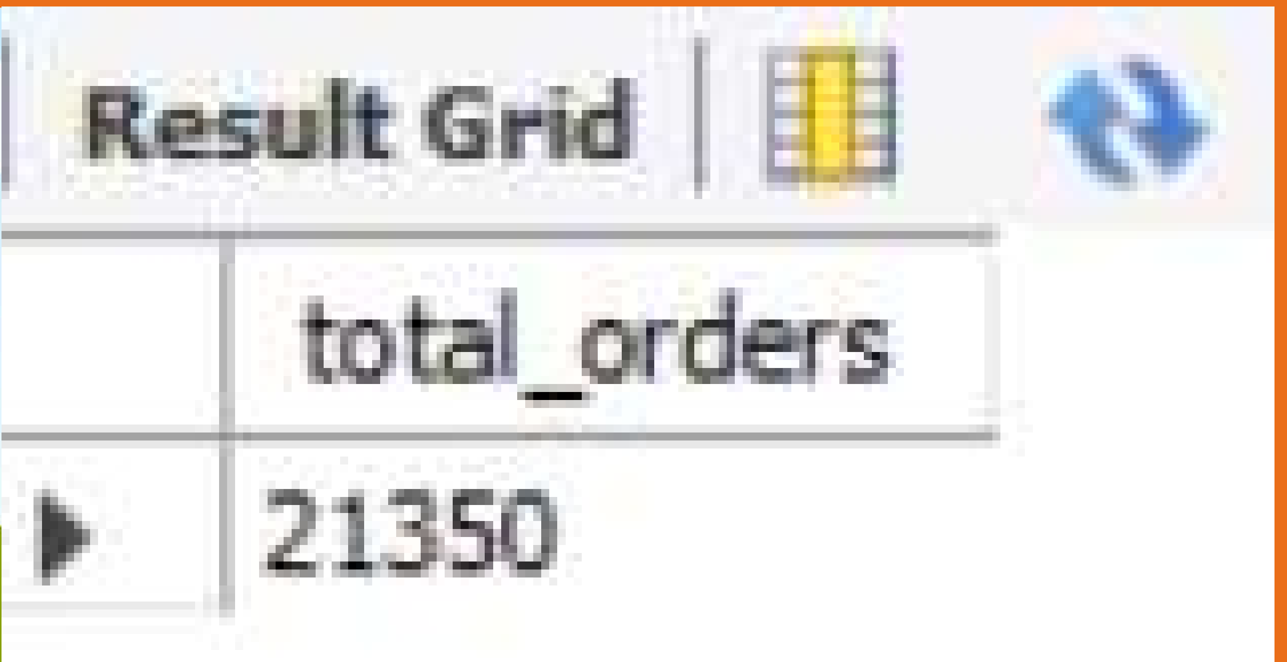
RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

Query

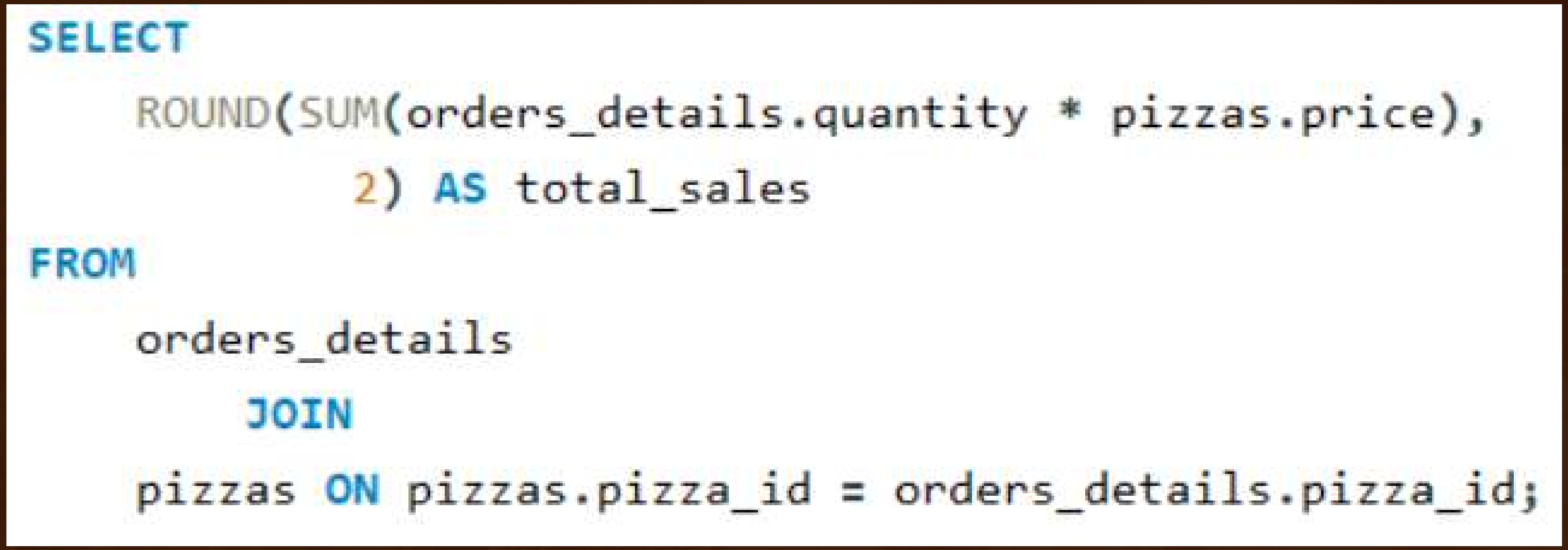


```
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

Output



```
total_orders
21350
```

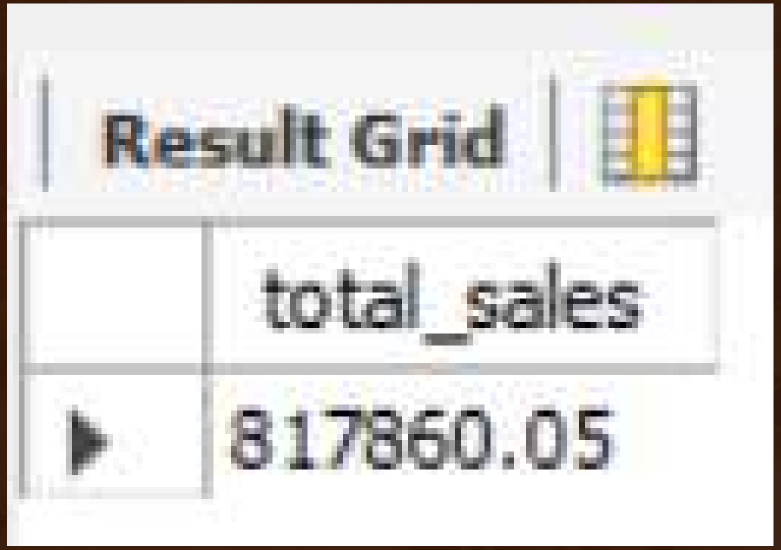


```
SELECT
    ROUND(SUM(orders_details.quantity * pizzas.price),
            2) AS total_sales
FROM
    orders_details
        JOIN
    pizzas ON pizzas.pizza_id = orders_details.pizza_id;
```

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

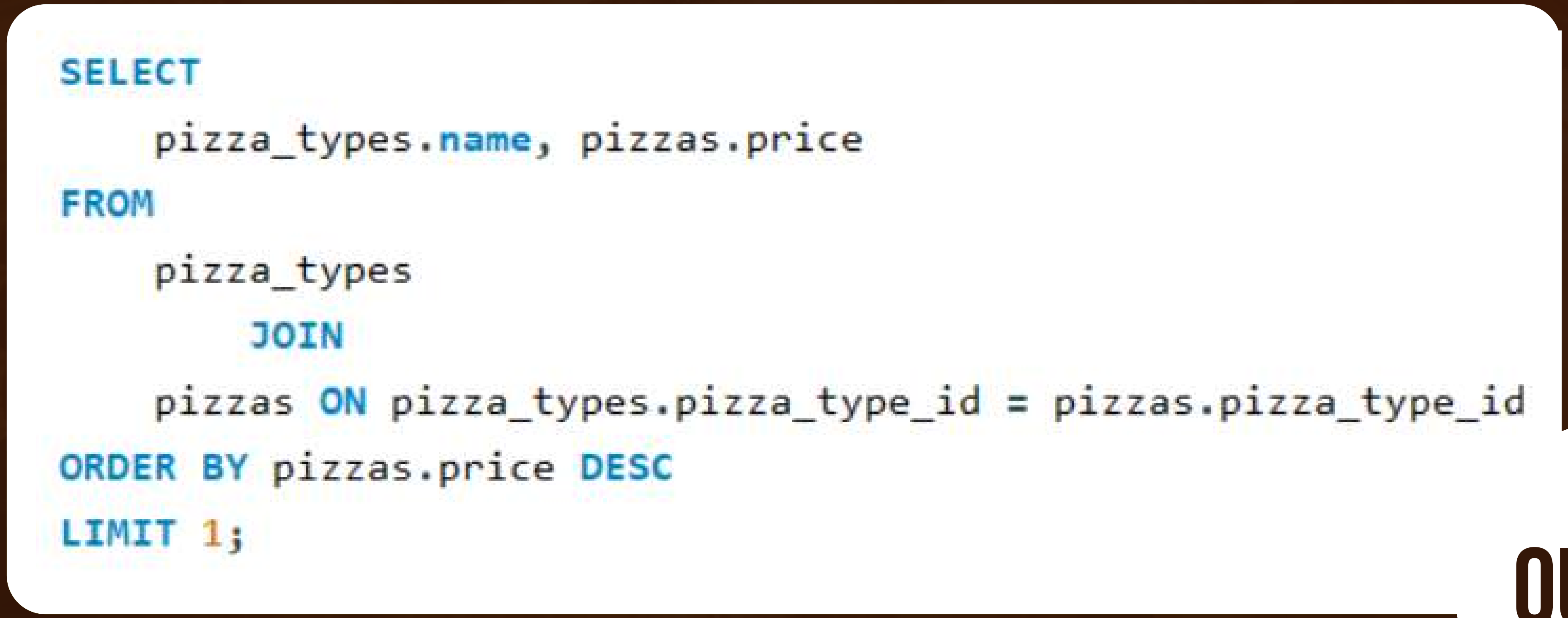
QUERY

- Output



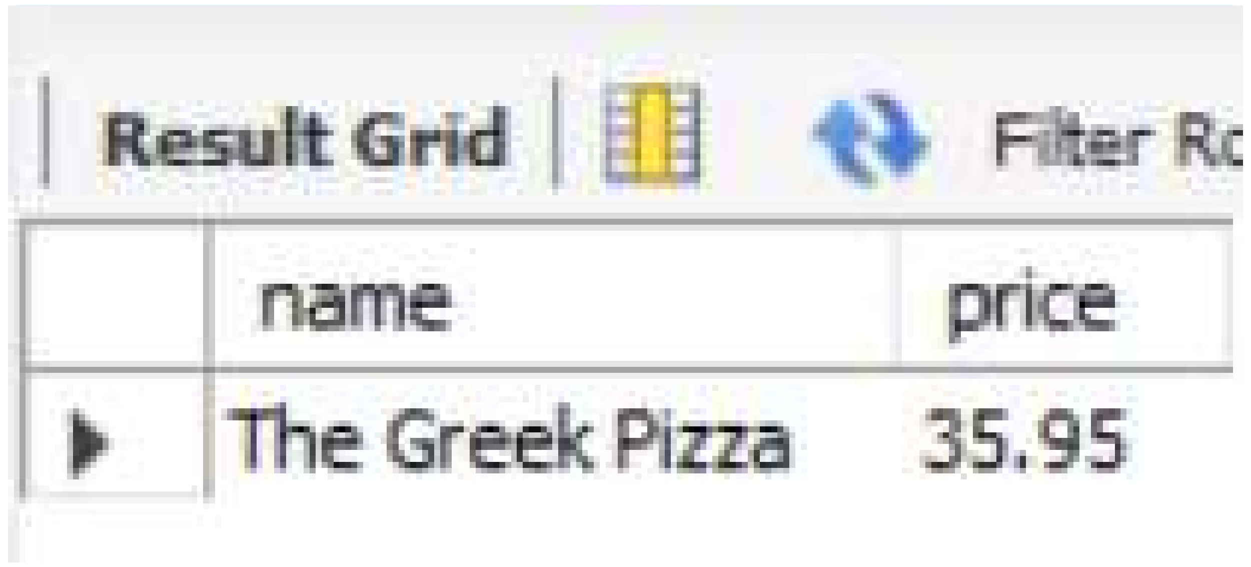
```
total_sales
817860.05
```

IDENTIFY THE HIGHEST-PRICED PIZZA



```
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;
```

QUERY

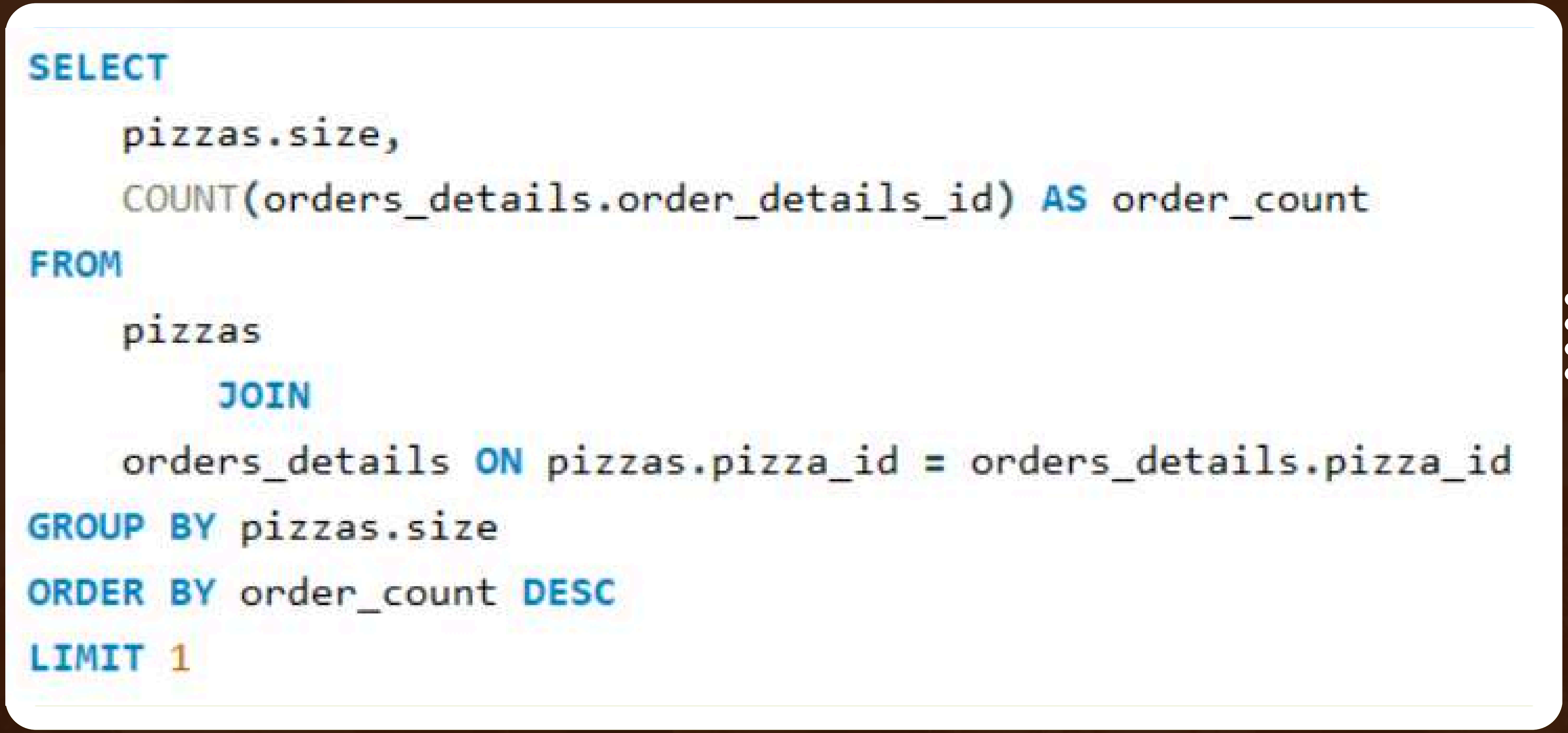


```
name              price
The Greek Pizza   35.95
```

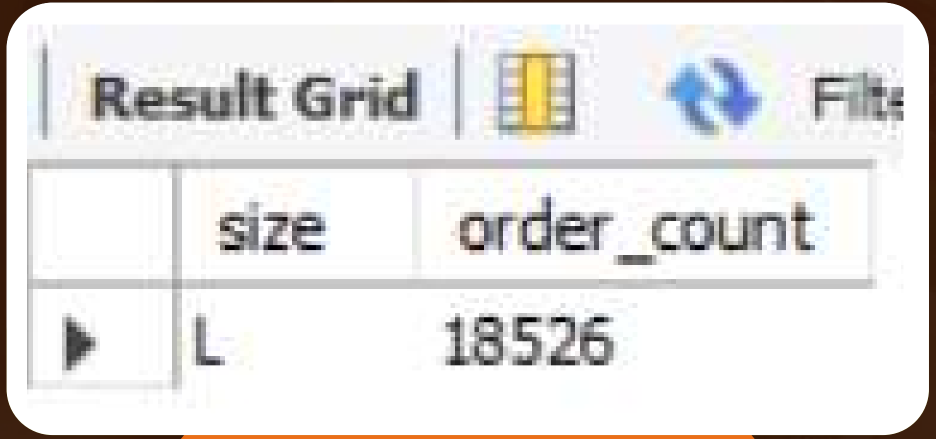
OUTPUT

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

Query



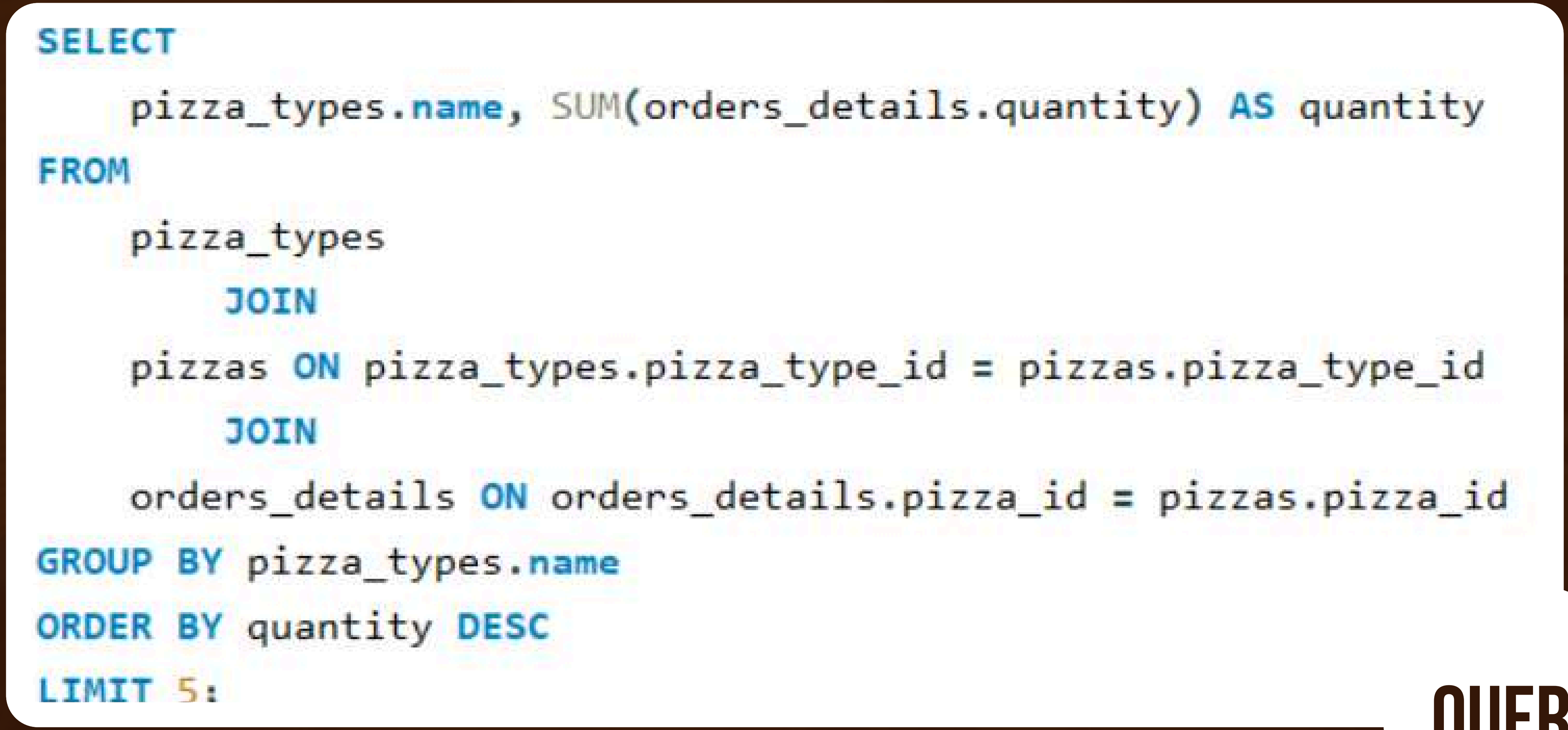
```
SELECT
    pizzas.size,
    COUNT(orders_details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC
LIMIT 1
```



```
size   order_count
L      18526
```

Output

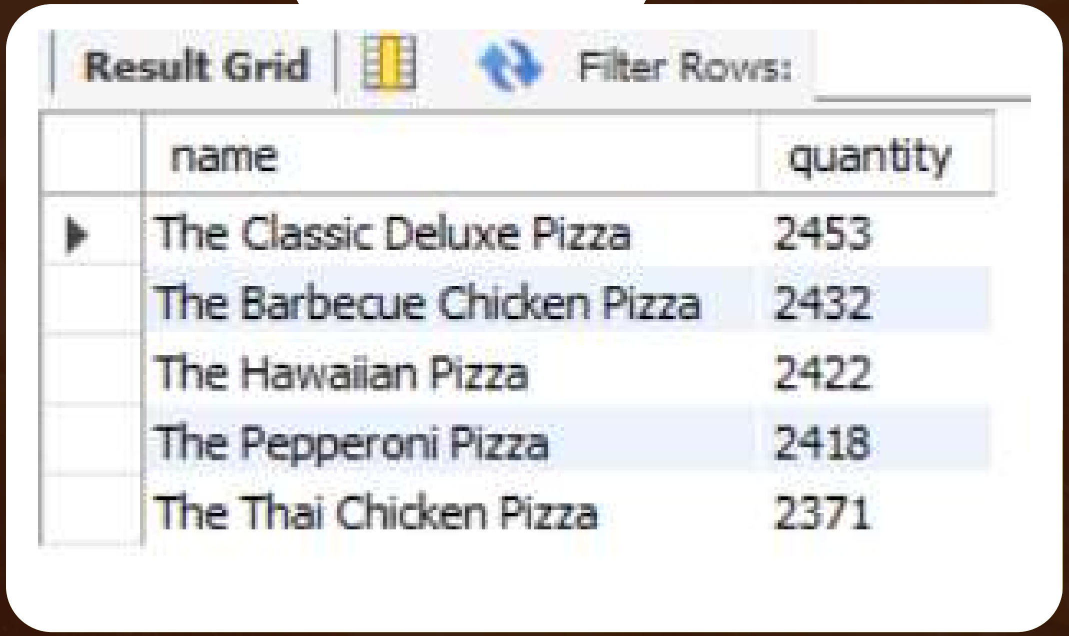
LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.



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

QUERY

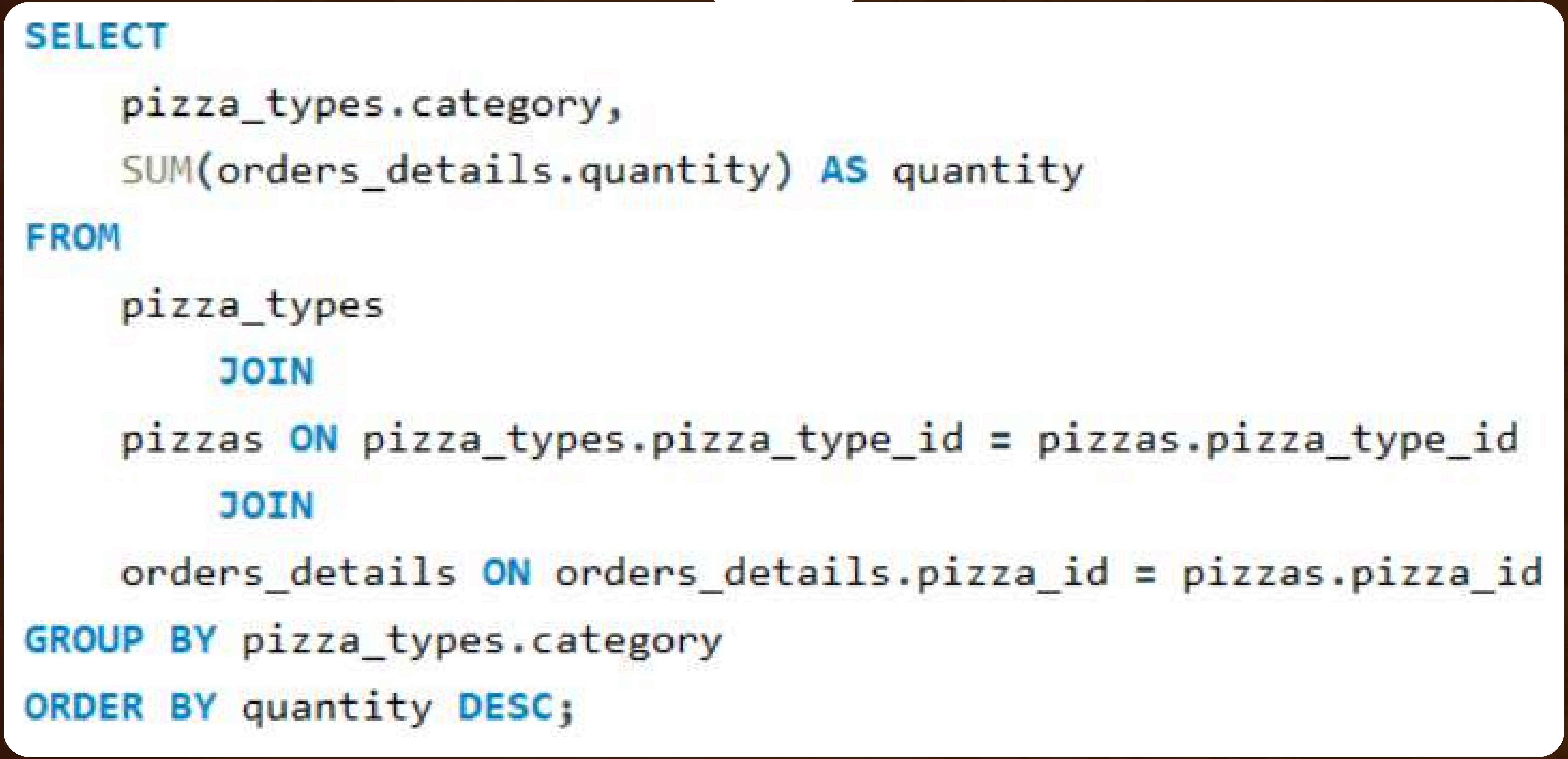
OUTPUT



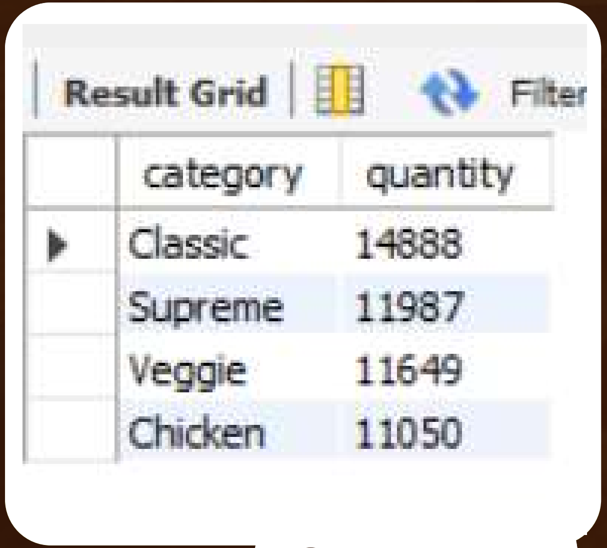
name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED

QUERY



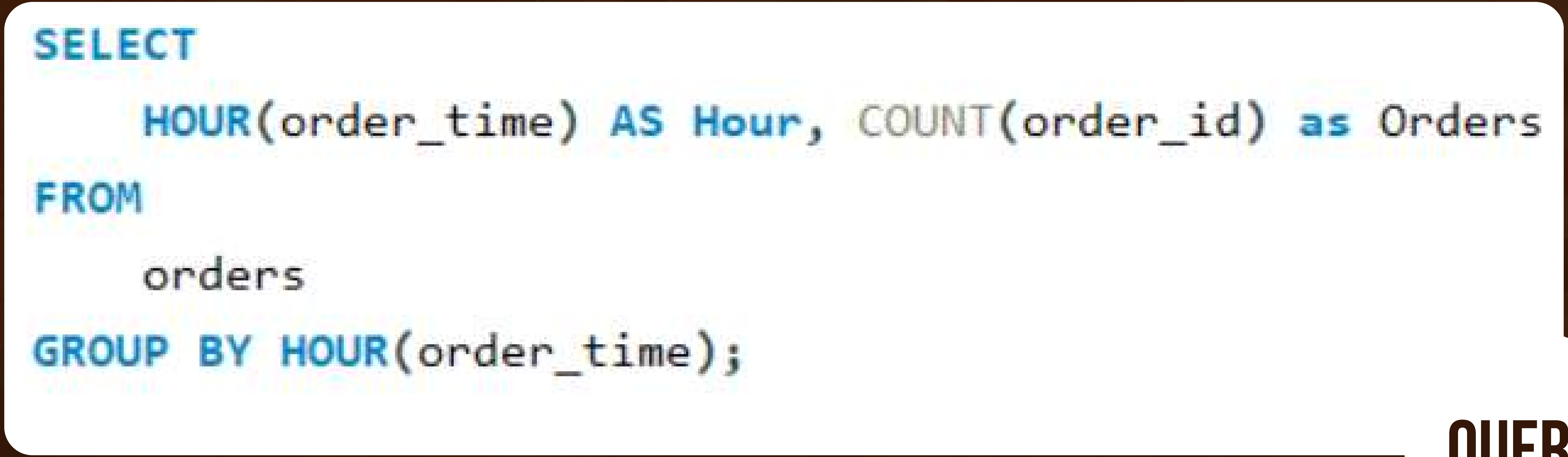
```
SELECT
    pizza_types.category,
    SUM(orders_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity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



category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

OUTPUT

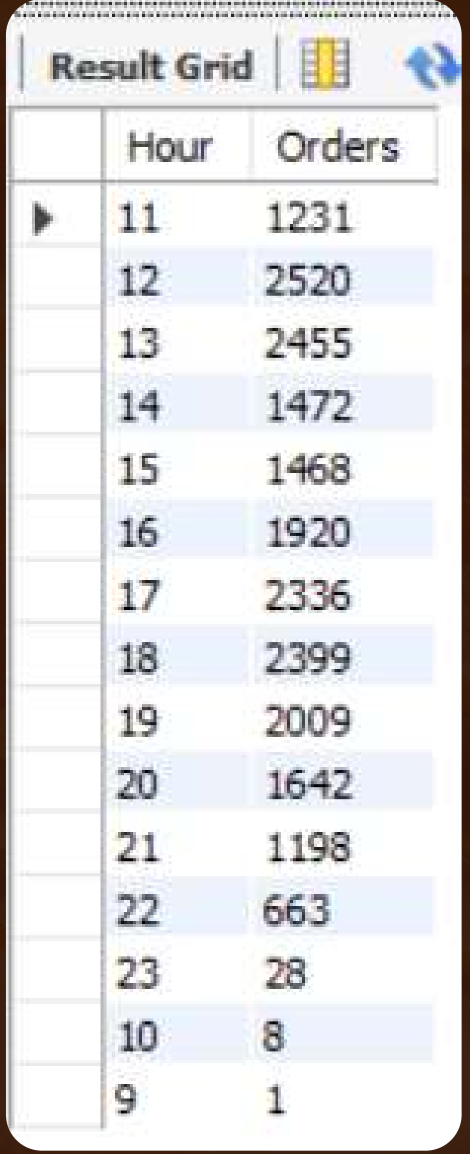
DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY



```
SELECT
    HOUR(order_time) AS Hour, COUNT(order_id) as Orders
FROM
    orders
GROUP BY HOUR(order_time);
```

QUERY

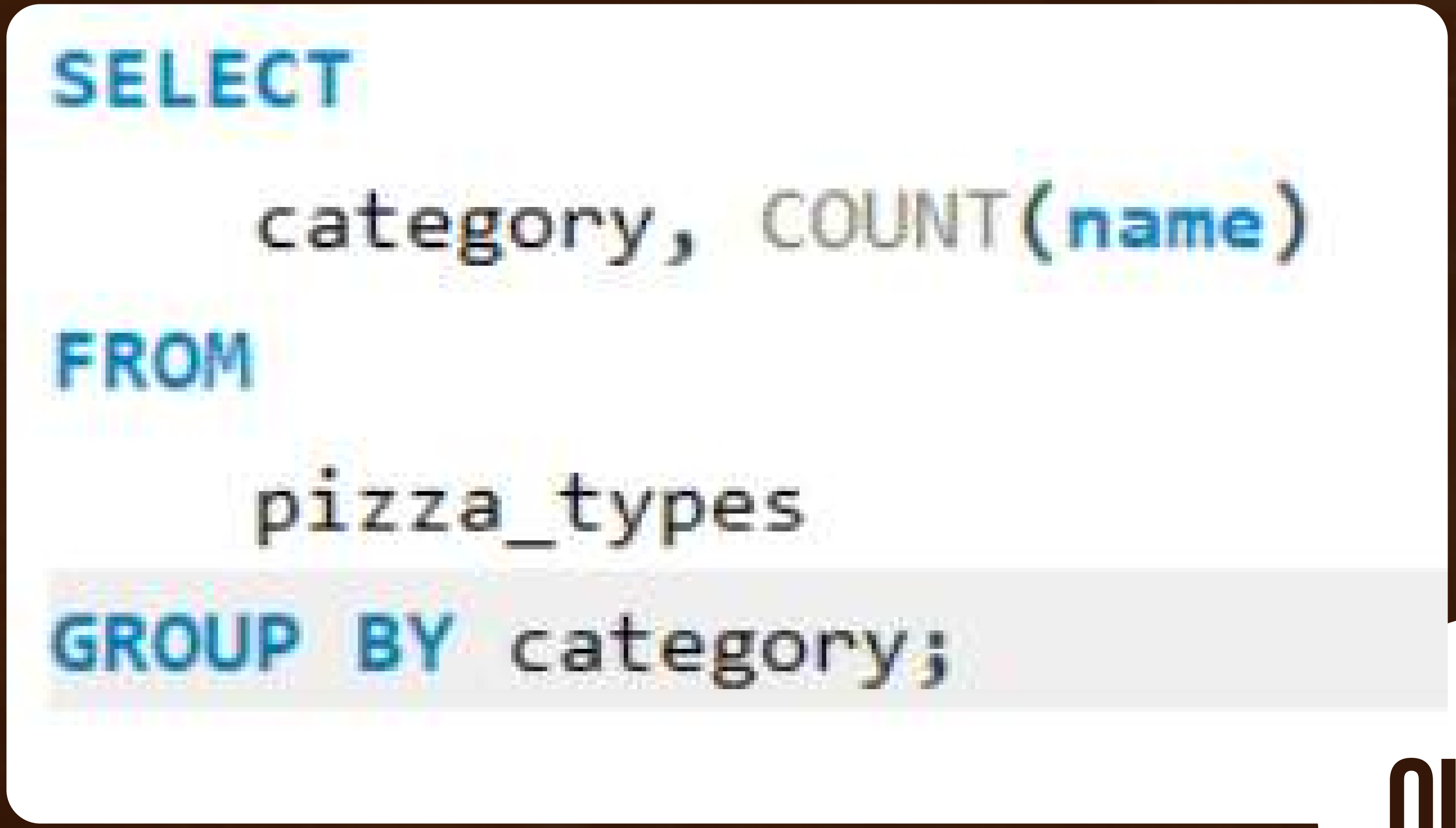
OUTPUT



Result Grid

Hour	Orders
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

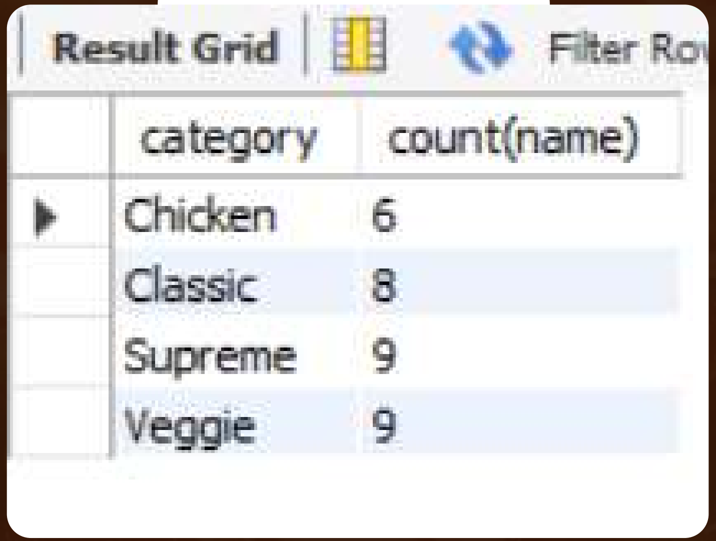
JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS



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

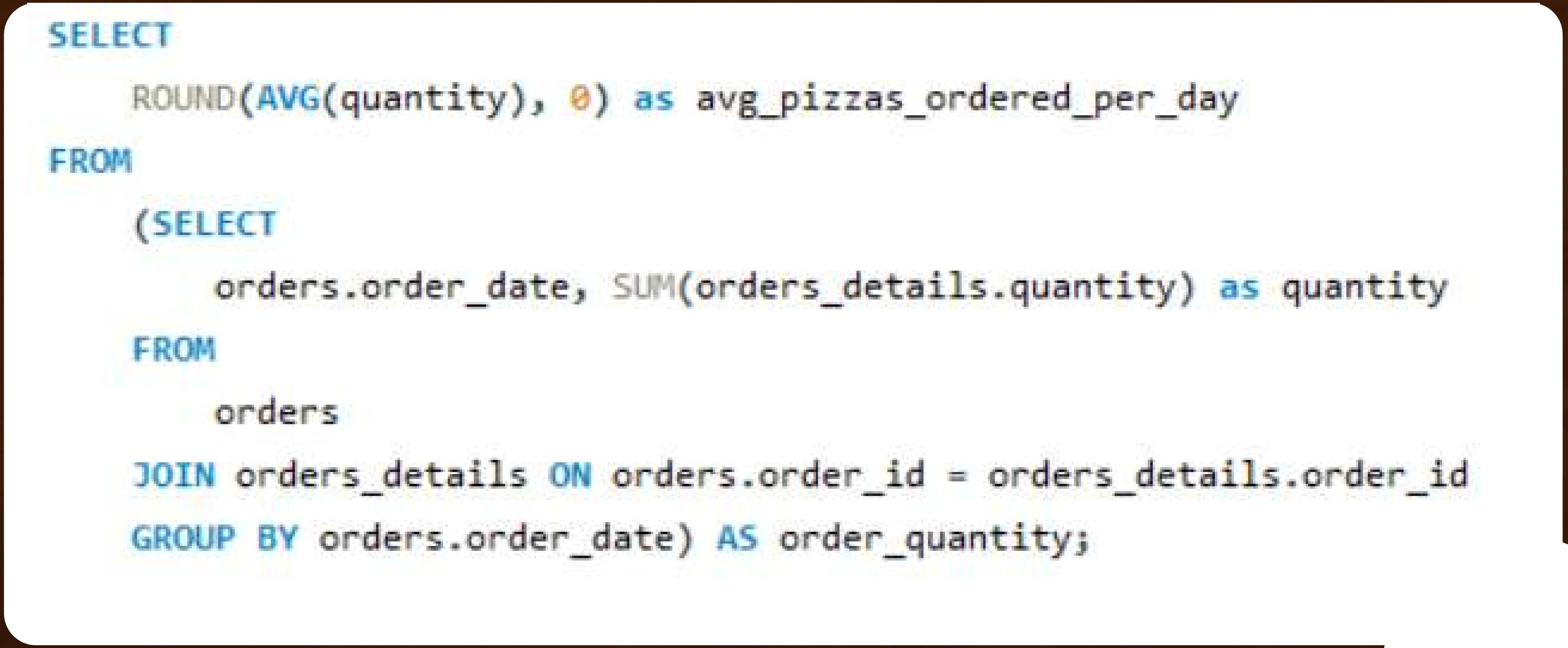
QUERY

OUTPUT



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

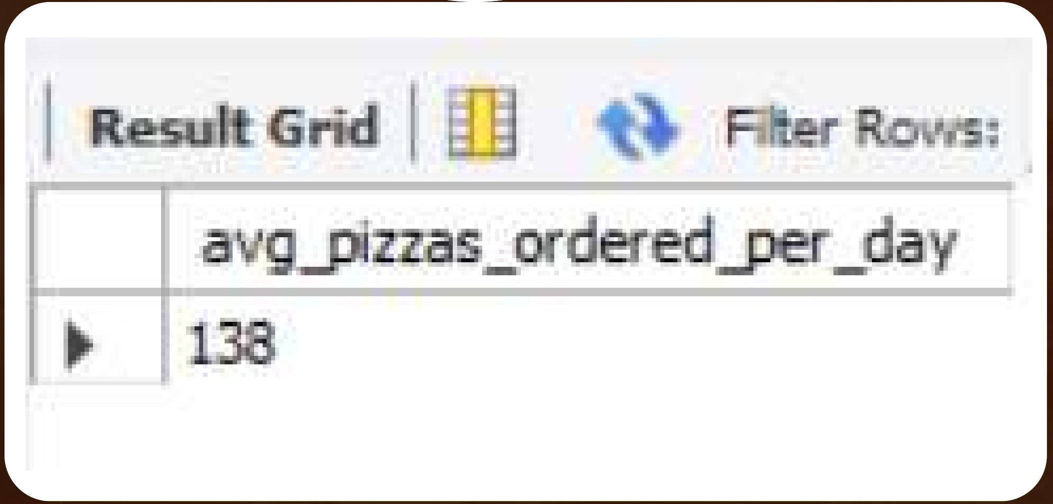
GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.



```
SELECT
    ROUND(AVG(quantity), 0) as avg_pizzas_ordered_per_day
FROM
    (SELECT
        orders.order_date, SUM(orders_details.quantity) as quantity
    FROM
        orders
    JOIN orders_details ON orders.order_id = orders_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

QUERY

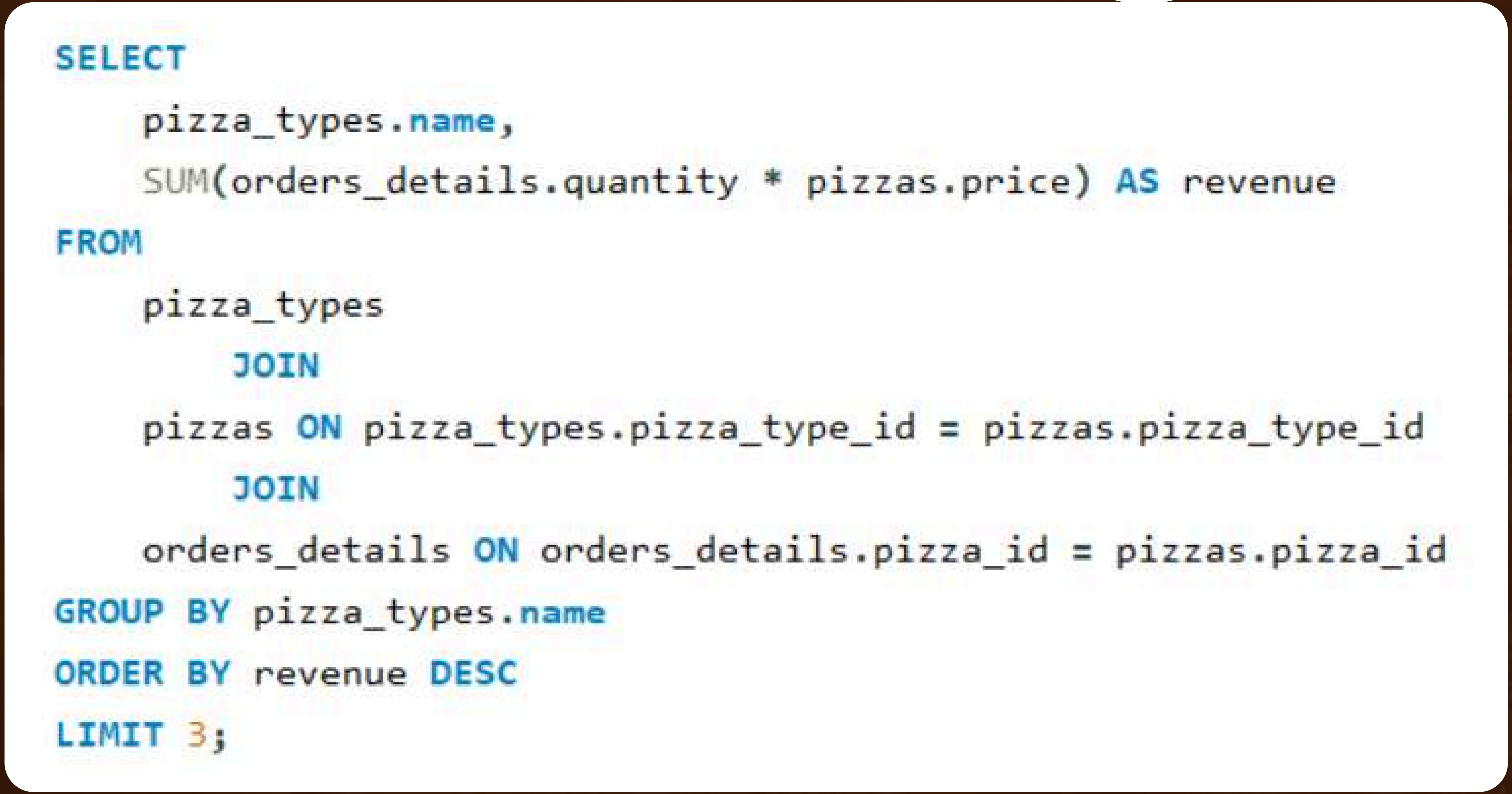
OUTPUT



```
avg_pizzas_ordered_per_day
138
```

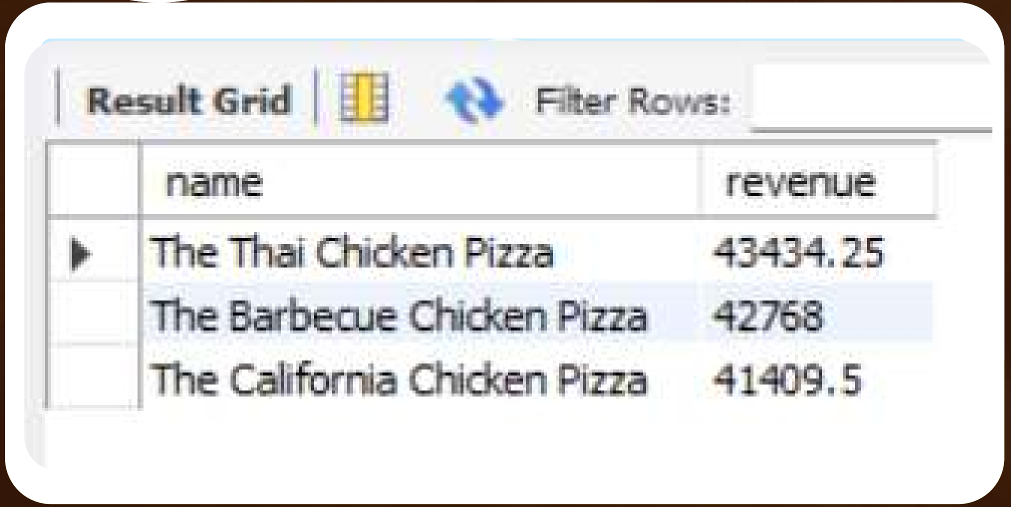
DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

QUERY



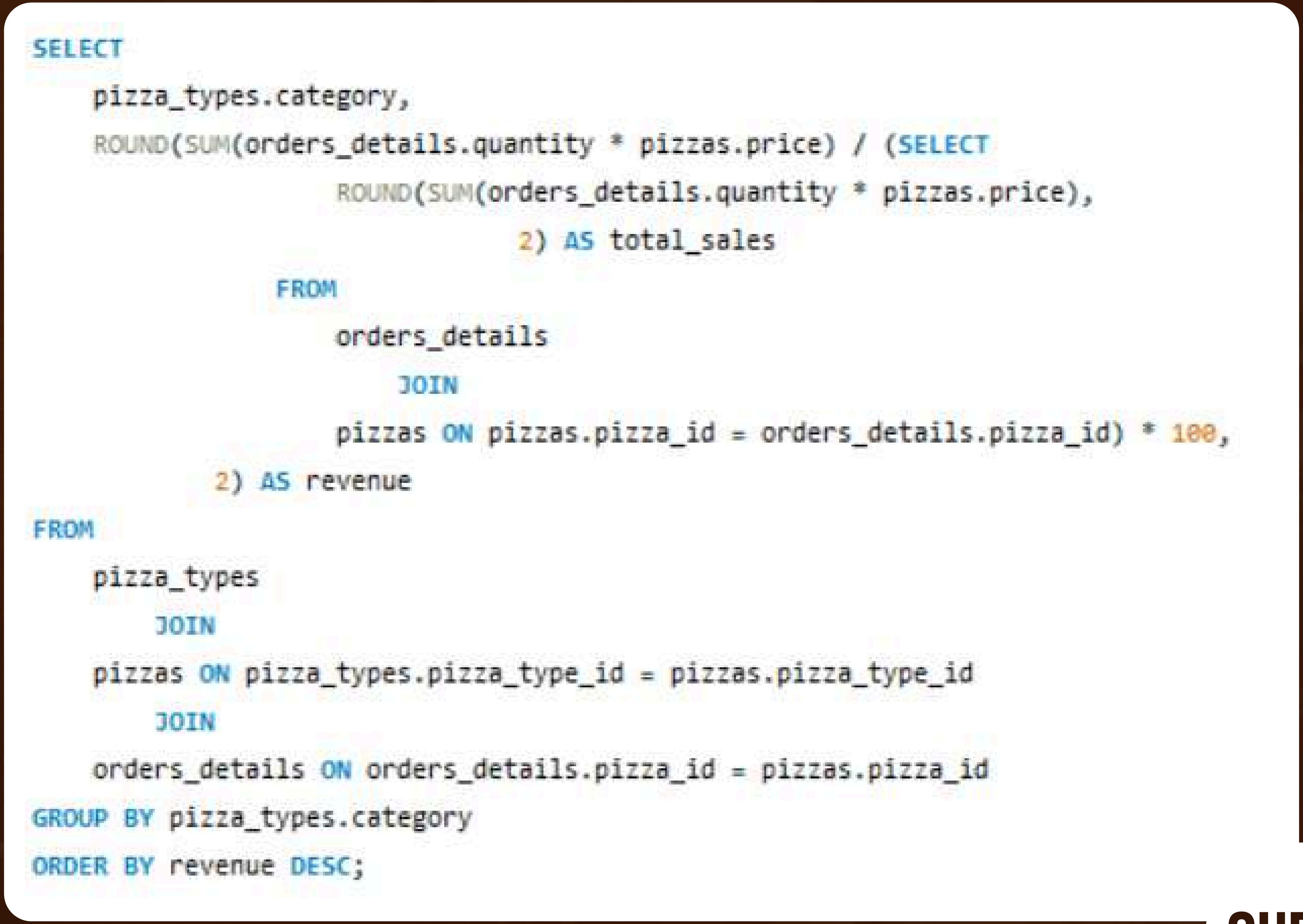
```
SELECT
    pizza_types.name,
    SUM(orders_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY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

OUTPUT



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

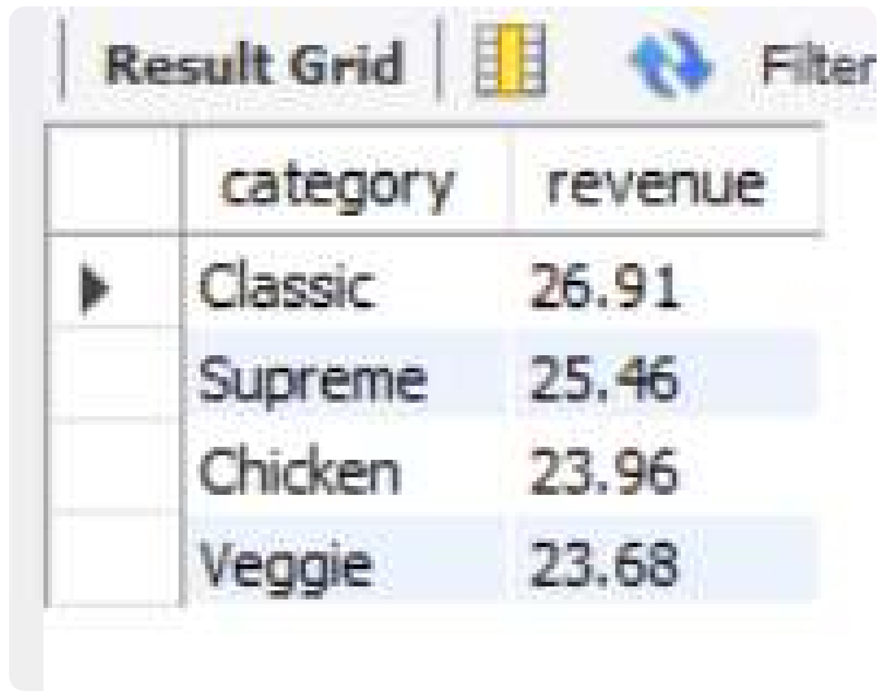
CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE BASED ON CATEGORY



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

QUERY

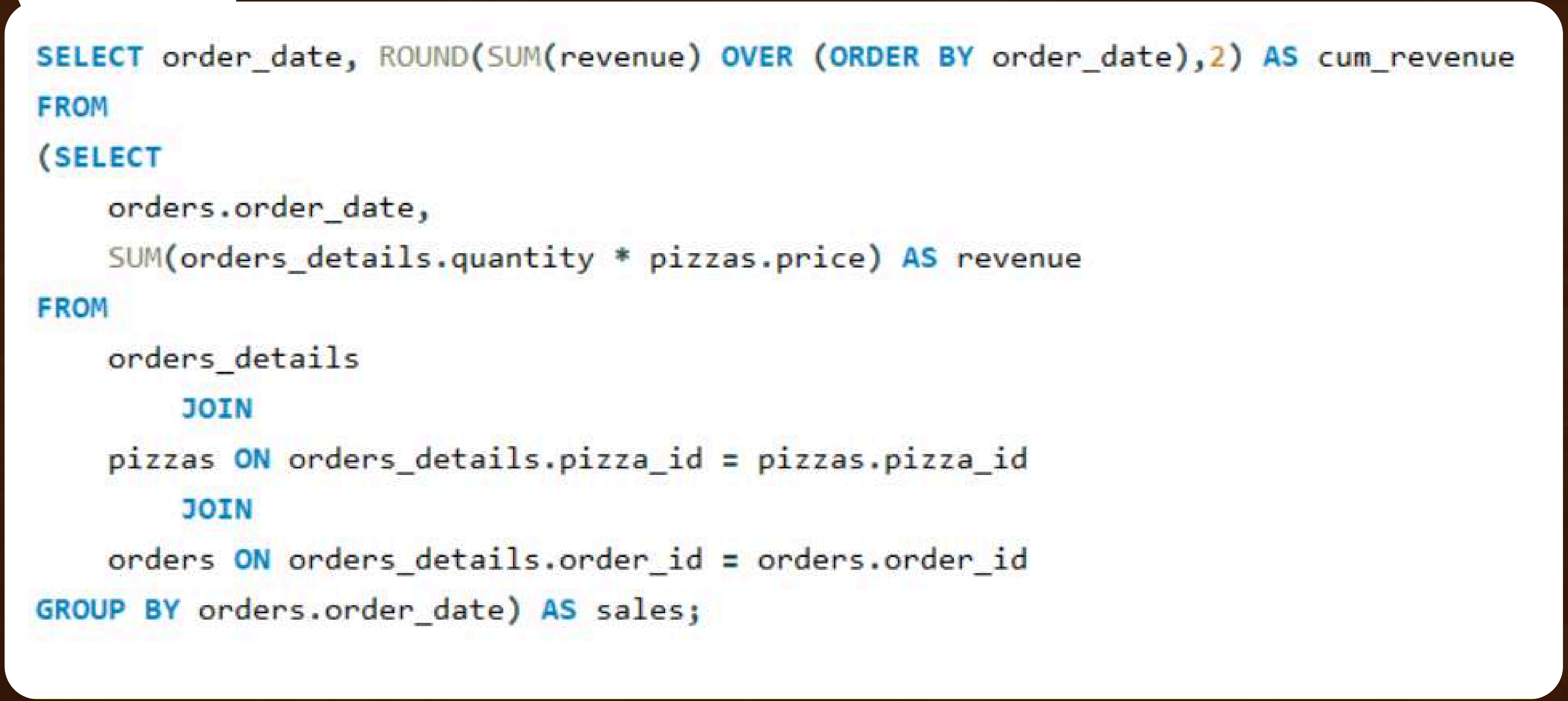
OUTPUT



category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

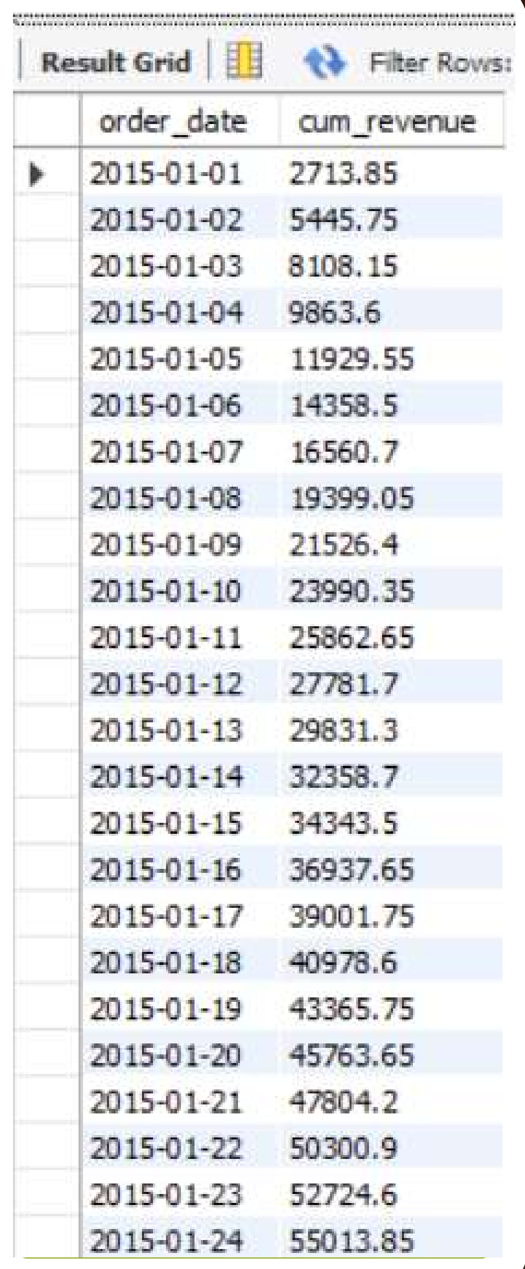
ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME

QUERY



```
SELECT order_date, ROUND(SUM(revenue) OVER (ORDER BY order_date),2) AS cum_revenue
FROM
(SELECT
    orders.order_date,
    SUM(orders_details.quantity * pizzas.price) AS revenue
FROM
    orders_details
        JOIN
    pizzas ON orders_details.pizza_id = pizzas.pizza_id
        JOIN
    orders ON orders_details.order_id = orders.order_id
GROUP BY orders.order_date) AS sales;
```

OUTPUT

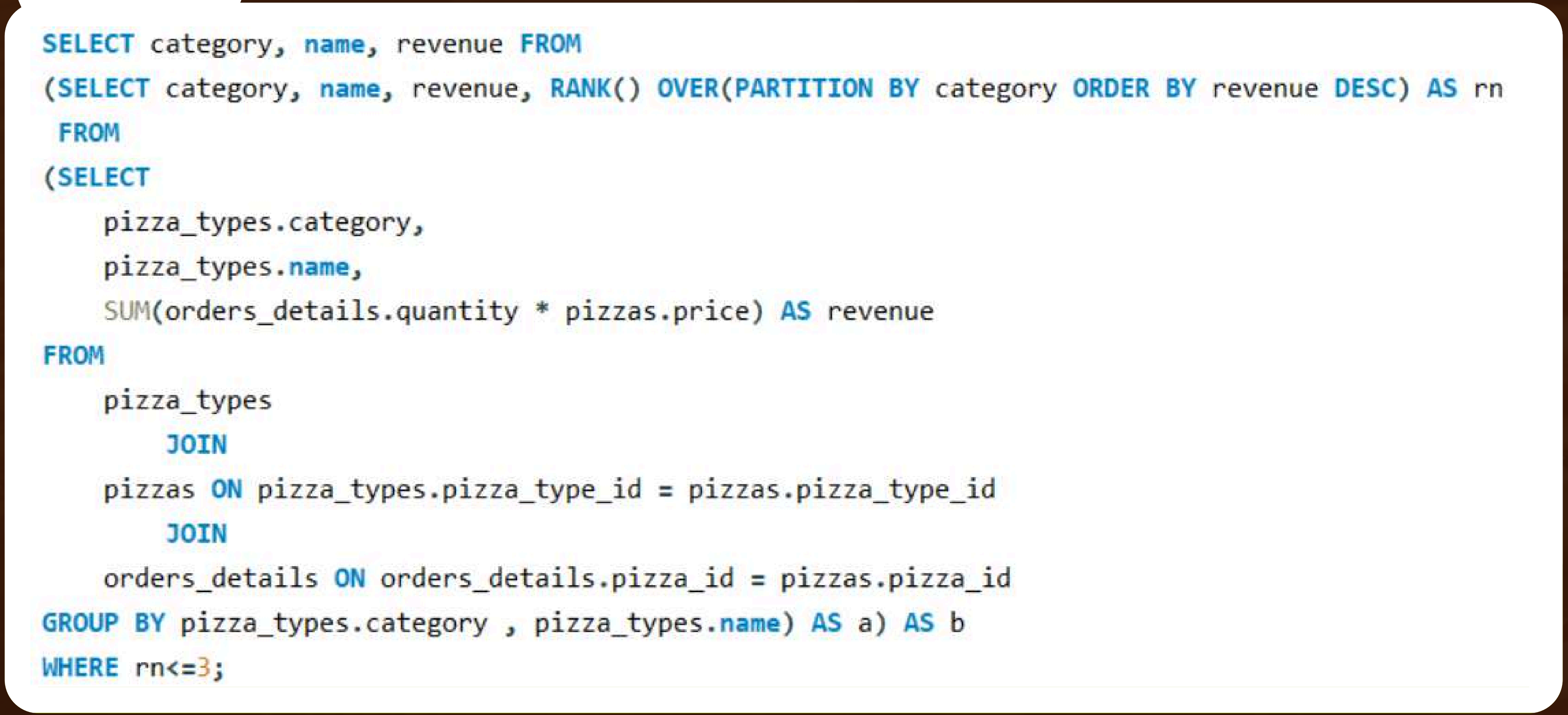


Result Grid | Filter Rows:

order_date	cum_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
2015-01-16	36937.65
2015-01-17	39001.75
2015-01-18	40978.6
2015-01-19	43365.75
2015-01-20	45763.65
2015-01-21	47804.2
2015-01-22	50300.9
2015-01-23	52724.6
2015-01-24	55013.85

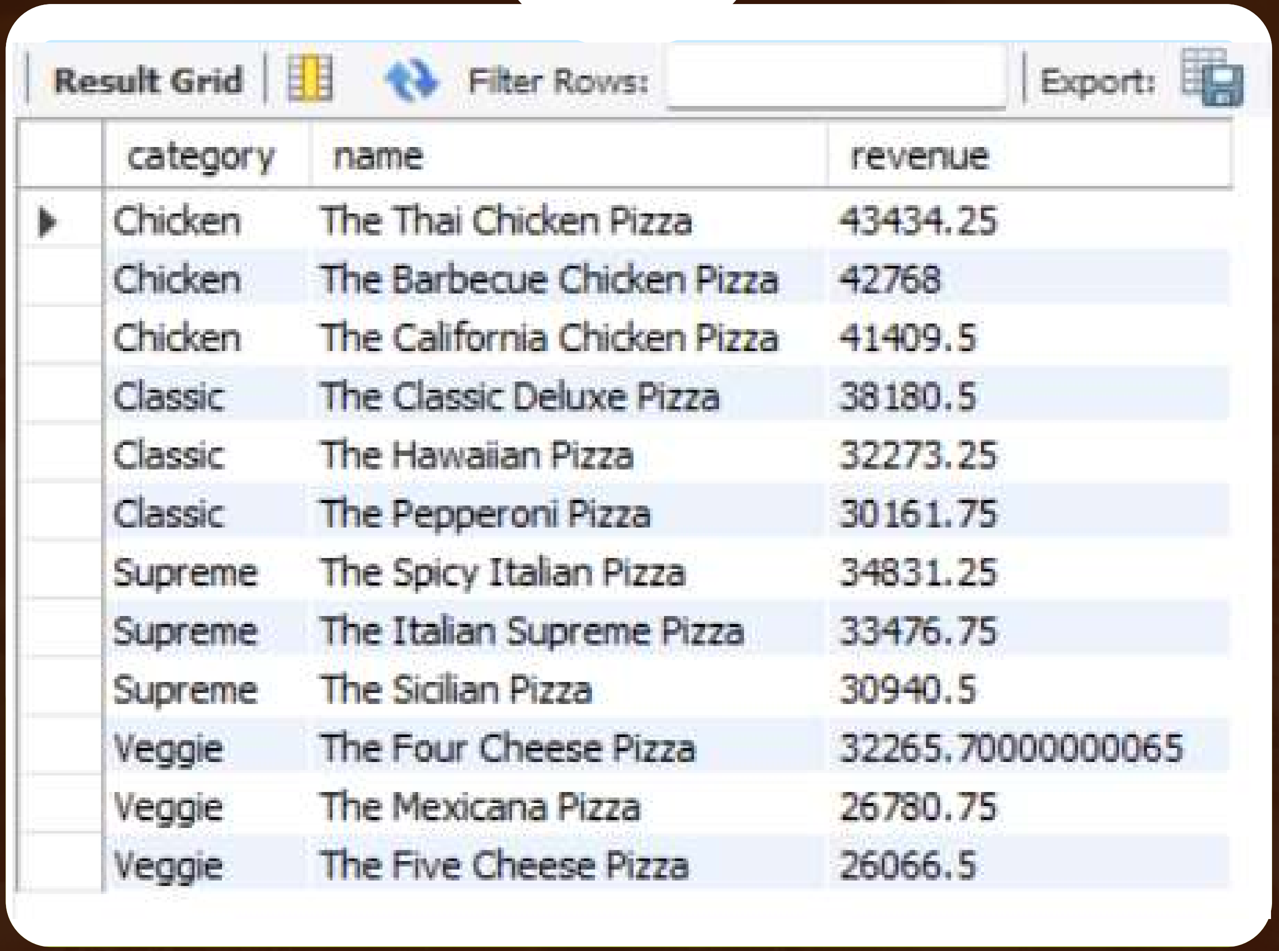
ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME

QUERY



```sql
SELECT category, name, revenue FROM
(SELECT category, name, revenue, RANK() OVER(PARTITION BY category ORDER BY revenue DESC) AS rn
 FROM
(SELECT
    pizza_types.category,
    pizza_types.name,
    SUM(orders_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category , pizza_types.name) AS a) AS b
WHERE rn<=3;
```

OUTPUT



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

Pizza Sales SQL Presentation

THANK YOU FOR ATTENTION

See You Next

Vijay Singh Parmar