PIZZA STORE

DATA ANALYSIS

An SQL Project

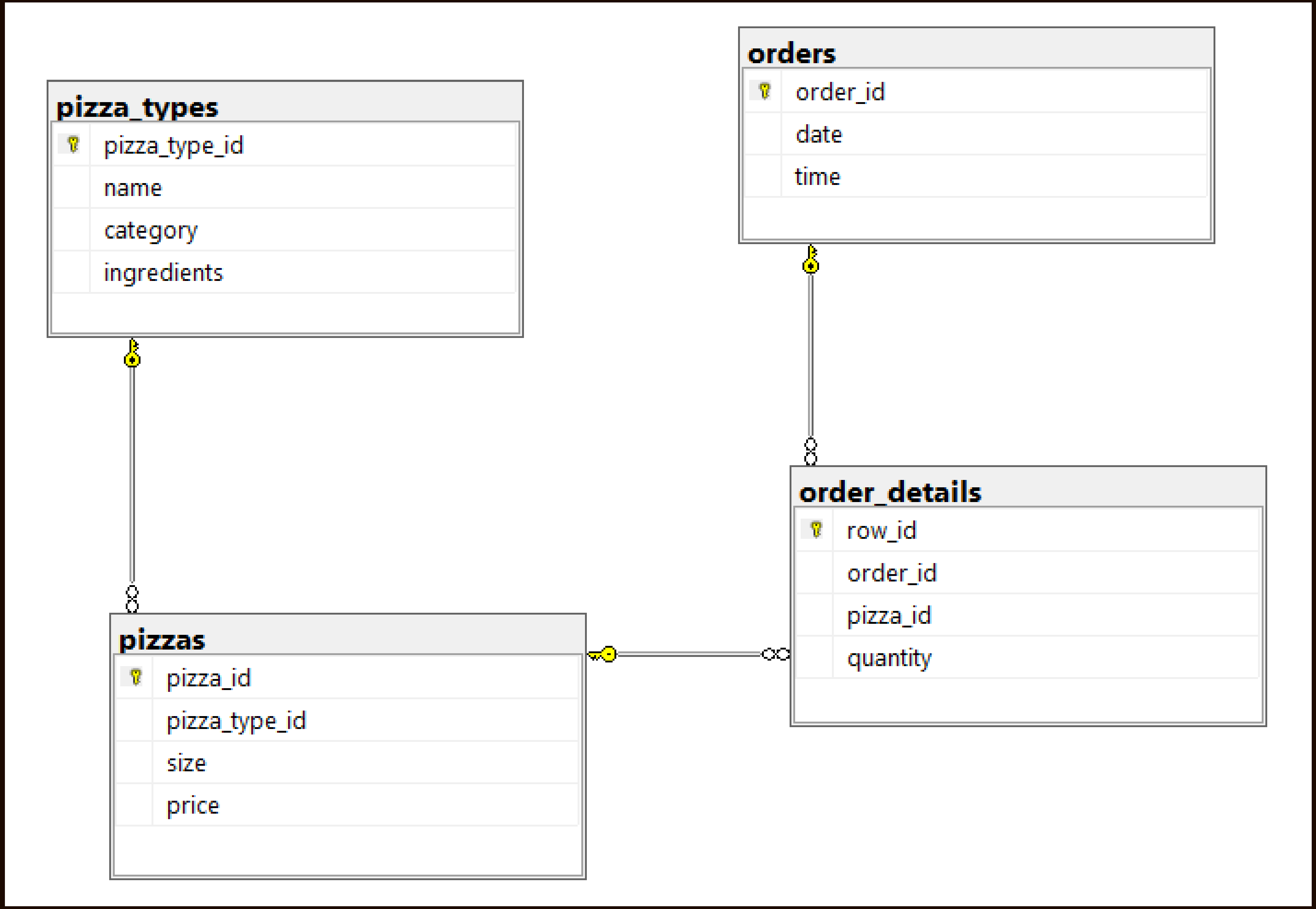
INTRODUCTION

This project involves the exploratory data analysis (EDA) of a pizza store's database to gain insights on the Pizza Business.

The aim is to answer various business questions to understand and optimize the store's operations.

This project analyzes sales data from a pizza store to uncover insights about customer ordering behavior, sales patterns and trends.

SCHEMA OVERVIEW



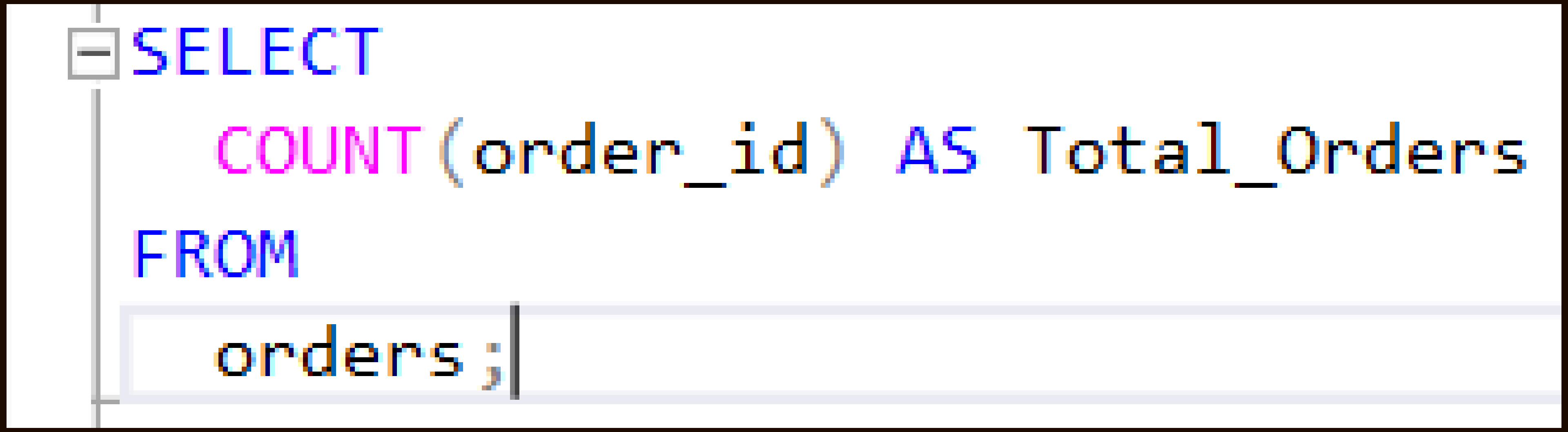
OBJECTIVE

To complete the objective of this project, we had to understand the table relationships and answer a few business questions by the store owner.

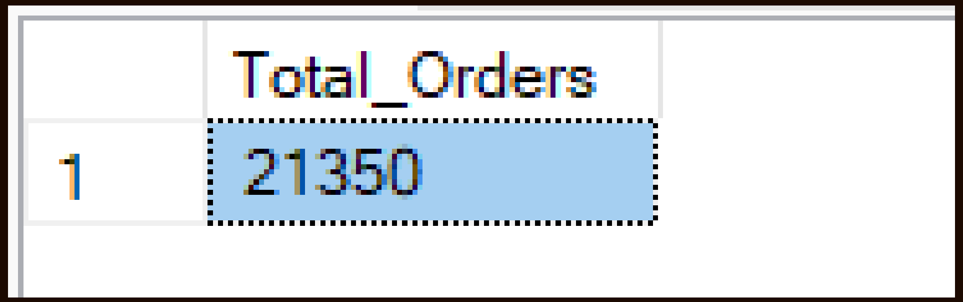
The results uncover necessary business insights required by the business owner to understand the business operations and the consumers better.

The next slides contain the requirements of the store owner and the queries that returned the insights which the owner required.

THE TOTAL NUMBER OF ORDERS PLACED.

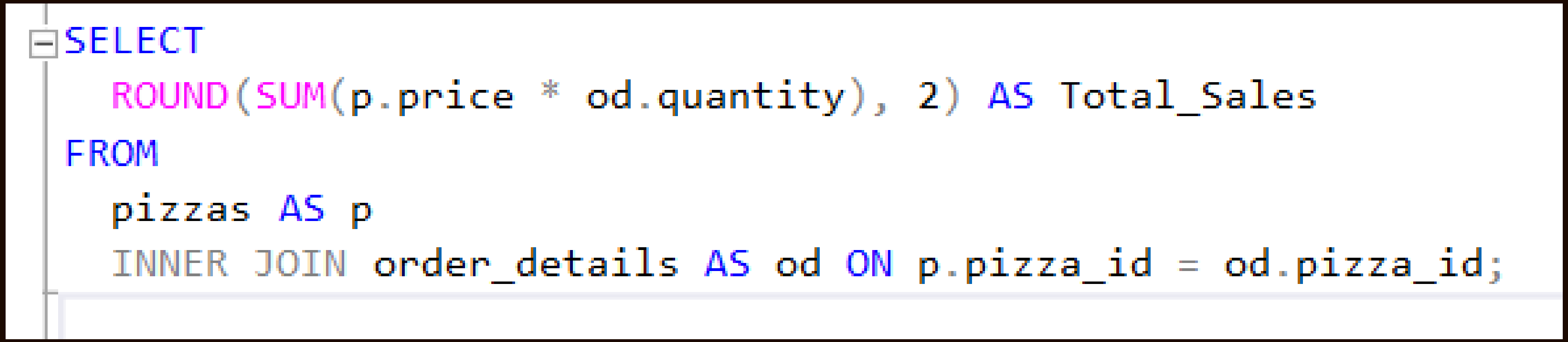


```
SELECT
   COUNT(order_id) AS Total_Orders
FROM
   orders;
```

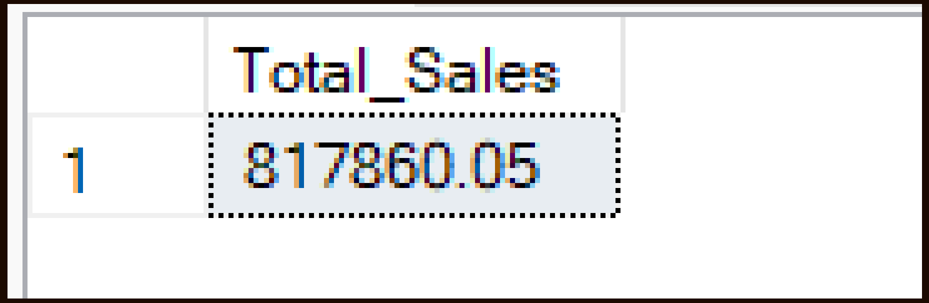


	Total_Orders
1	21350

TOTAL REVENUE GENERATED FROM PIZZA SALES.

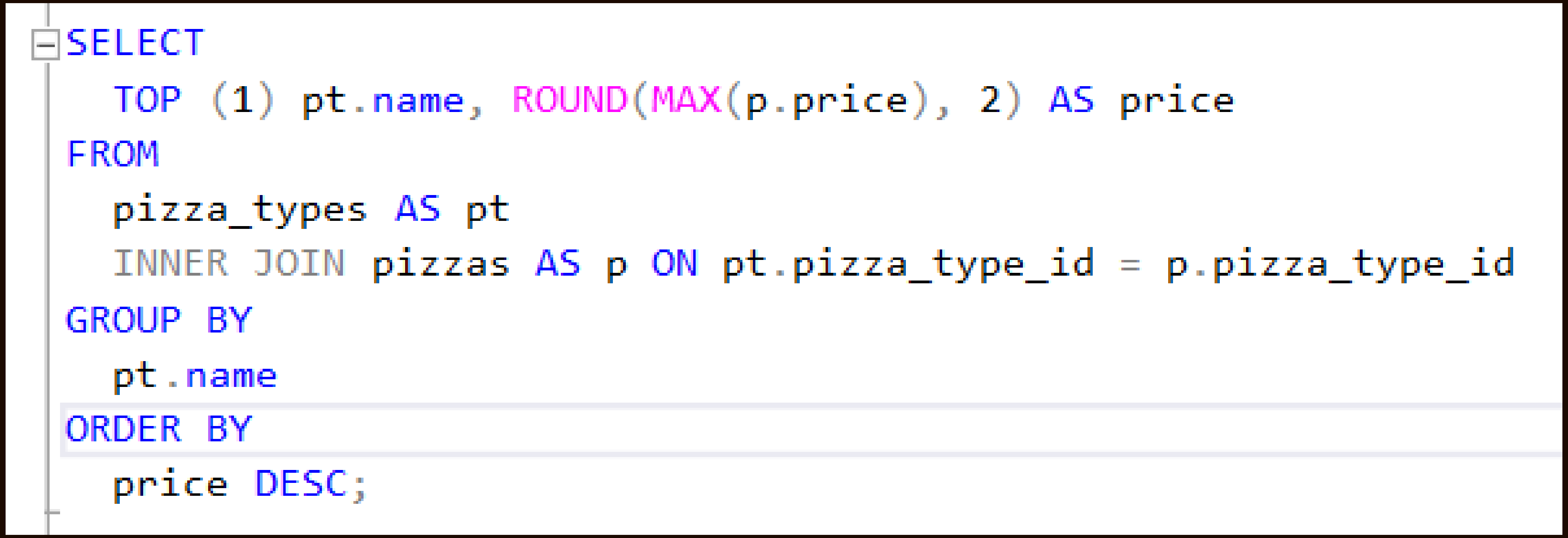


```
SELECT
  ROUND(SUM(p.price * od.quantity), 2) AS Total_Sales
FROM
  pizzas AS p
  INNER JOIN order_details AS od ON p.pizza_id = od.pizza_id;
```

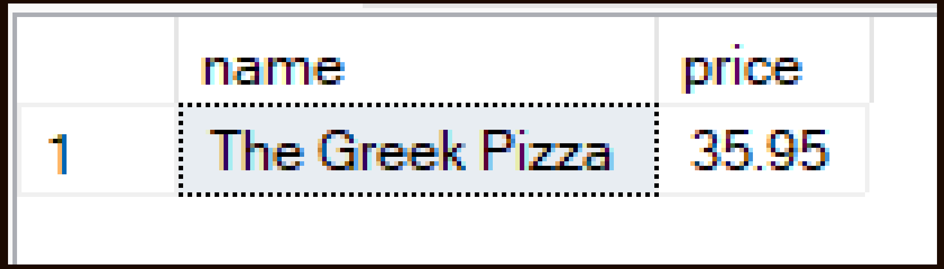


```
    Total_Sales
1   817860.05
```

IDENTIFY THE HIGHEST-PRICED PIZZA.

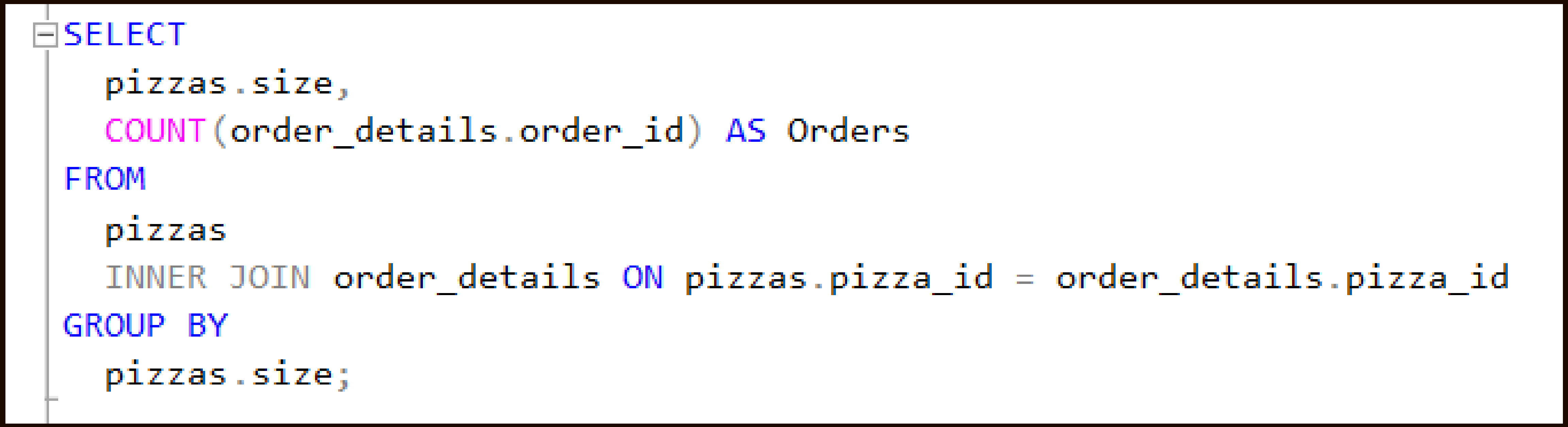


```
SELECT
  TOP (1) pt.name, ROUND(MAX(p.price), 2) AS price
FROM
  pizza_types AS pt
  INNER JOIN pizzas AS p ON pt.pizza_type_id = p.pizza_type_id
GROUP BY
  pt.name
ORDER BY
  price DESC;
```

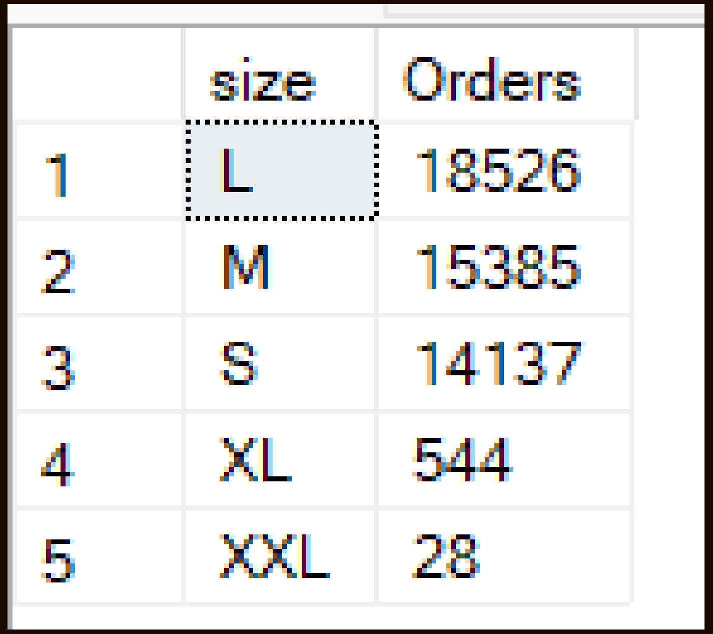


	name	price
1	The Greek Pizza	35.95

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

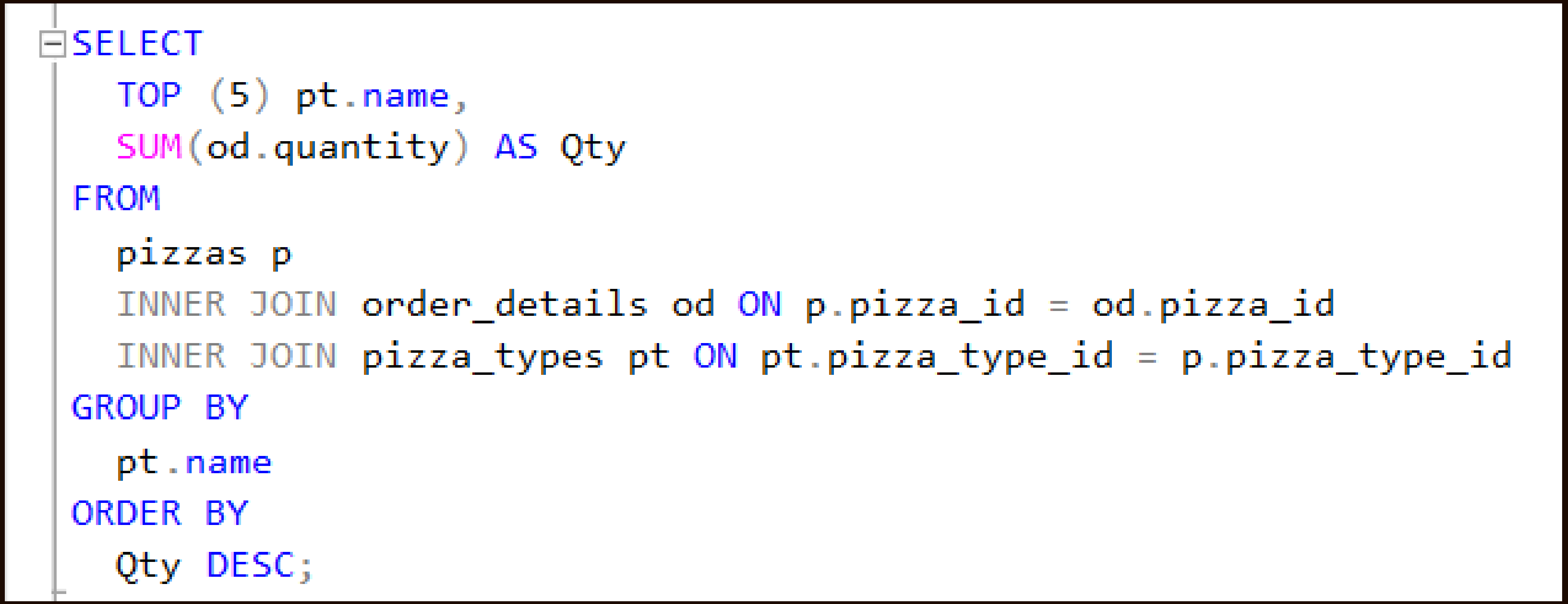


```
SELECT
  pizzas.size,
  COUNT(order_details.order_id) AS Orders
FROM
  pizzas
  INNER JOIN order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY
  pizzas.size;
```

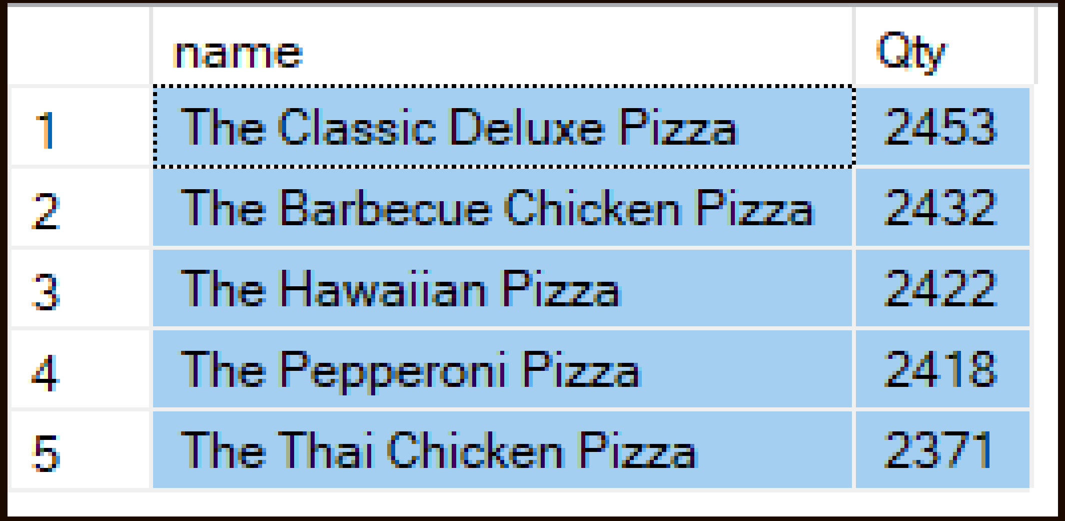


	size	Orders
1	L	18526
2	M	15385
3	S	14137
4	XL	544
5	XXL	28

TOP 5 MOST ORDERED PIZZA TYPES WITH THEIR QUANTITIES.

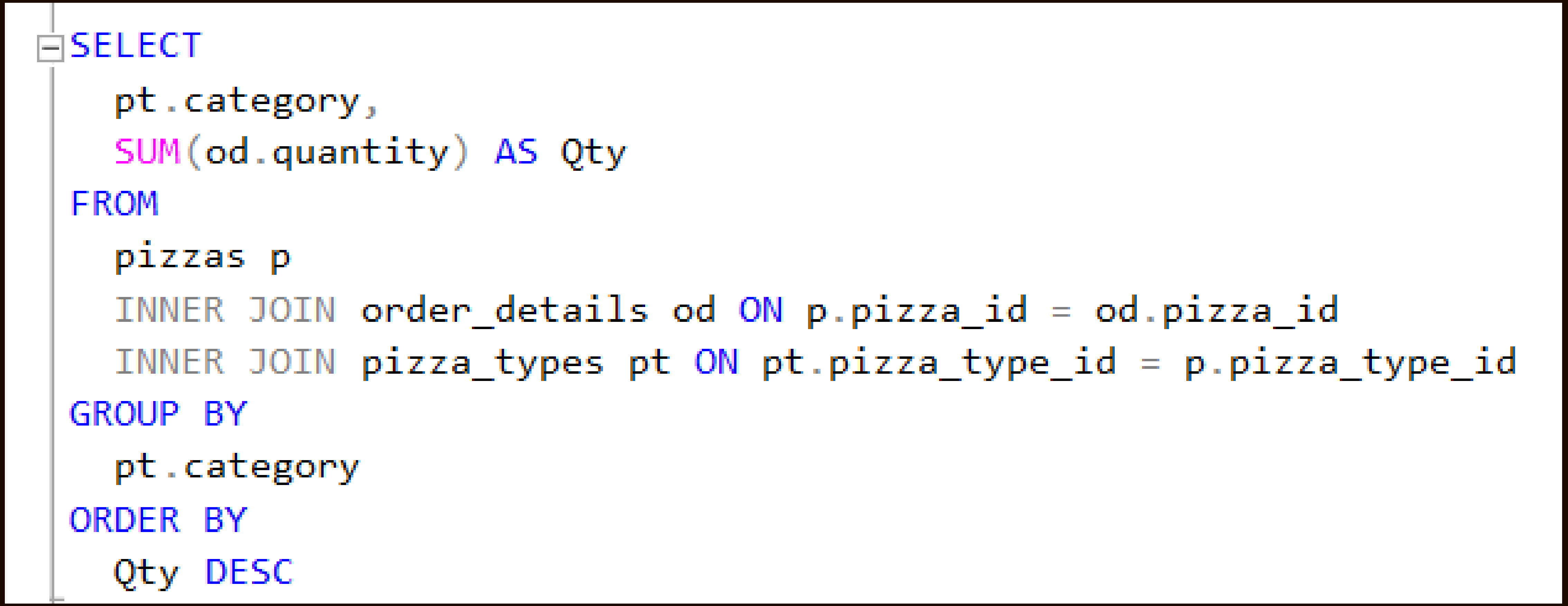


```
SELECT
  TOP (5) pt.name,
  SUM(od.quantity) AS Qty
FROM
  pizzas p
  INNER JOIN order_details od ON p.pizza_id = od.pizza_id
  INNER JOIN pizza_types pt ON pt.pizza_type_id = p.pizza_type_id
GROUP BY
  pt.name
ORDER BY
  Qty DESC;
```

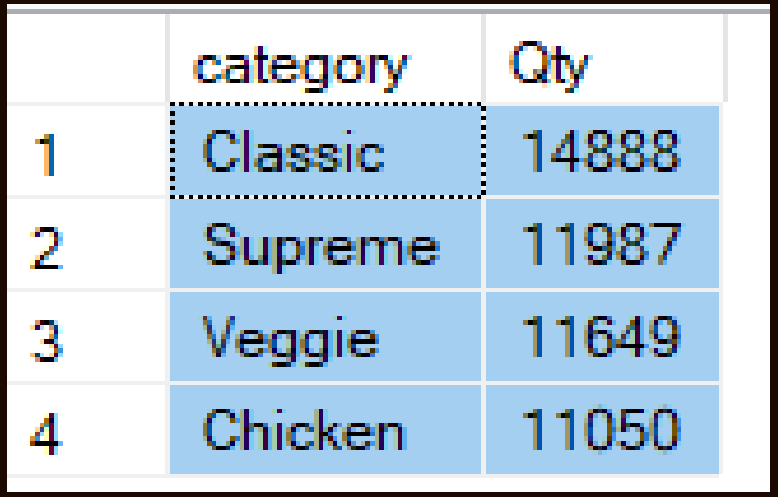


	name	Qty
1	The Classic Deluxe Pizza	2453
2	The Barbecue Chicken Pizza	2432
3	The Hawaiian Pizza	2422
4	The Pepperoni Pizza	2418
5	The Thai Chicken Pizza	2371

TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

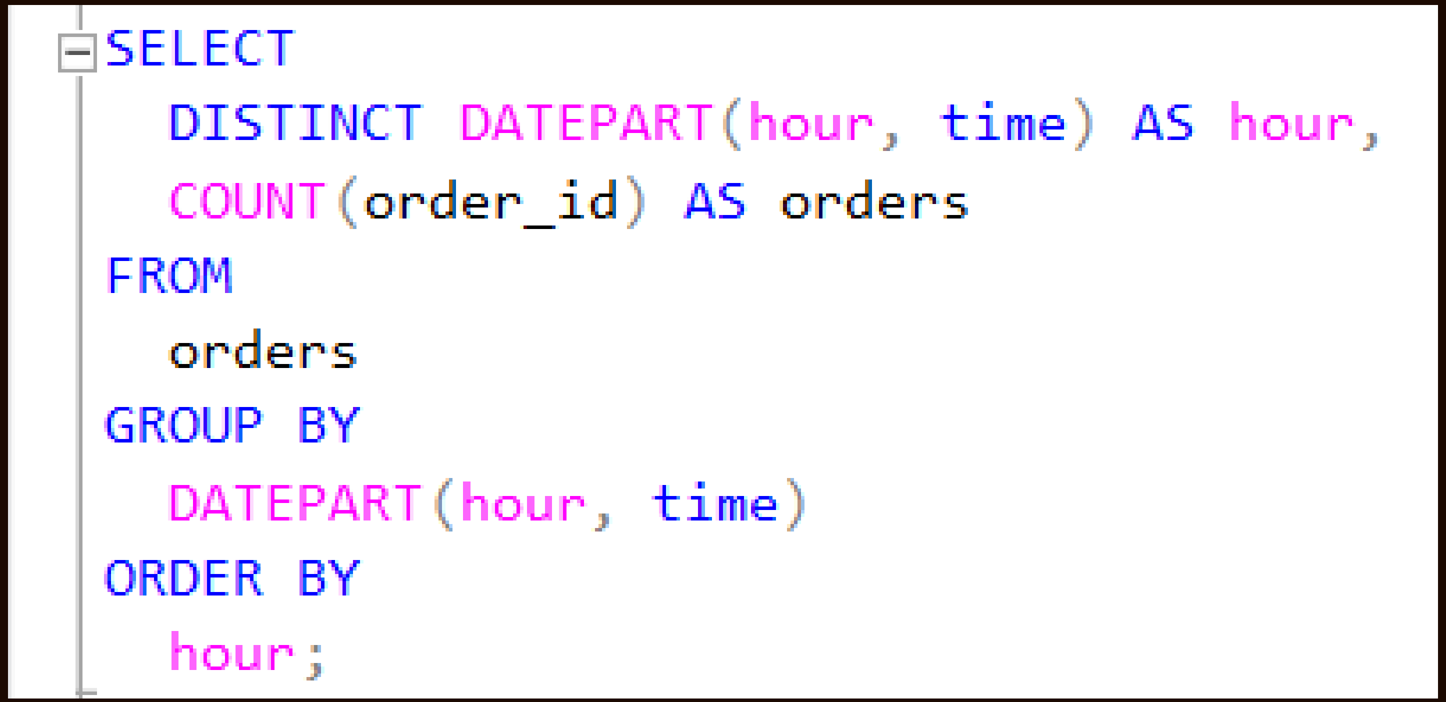


```
SELECT
  pt.category,
  SUM(od.quantity) AS Qty
FROM
  pizzas p
  INNER JOIN order_details od ON p.pizza_id = od.pizza_id
  INNER JOIN pizza_types pt ON pt.pizza_type_id = p.pizza_type_id
GROUP BY
  pt.category
ORDER BY
  Qty DESC
```

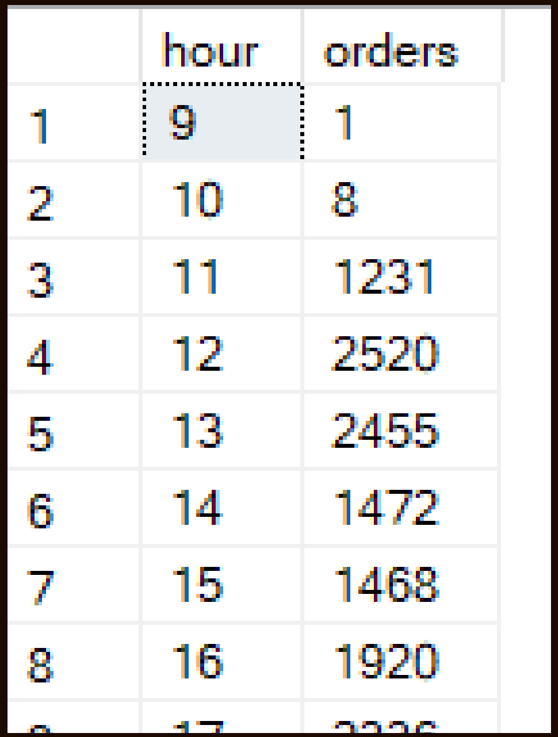


	category	Qty
1	Classic	14888
2	Supreme	11987
3	Veggie	11649
4	Chicken	11050

DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

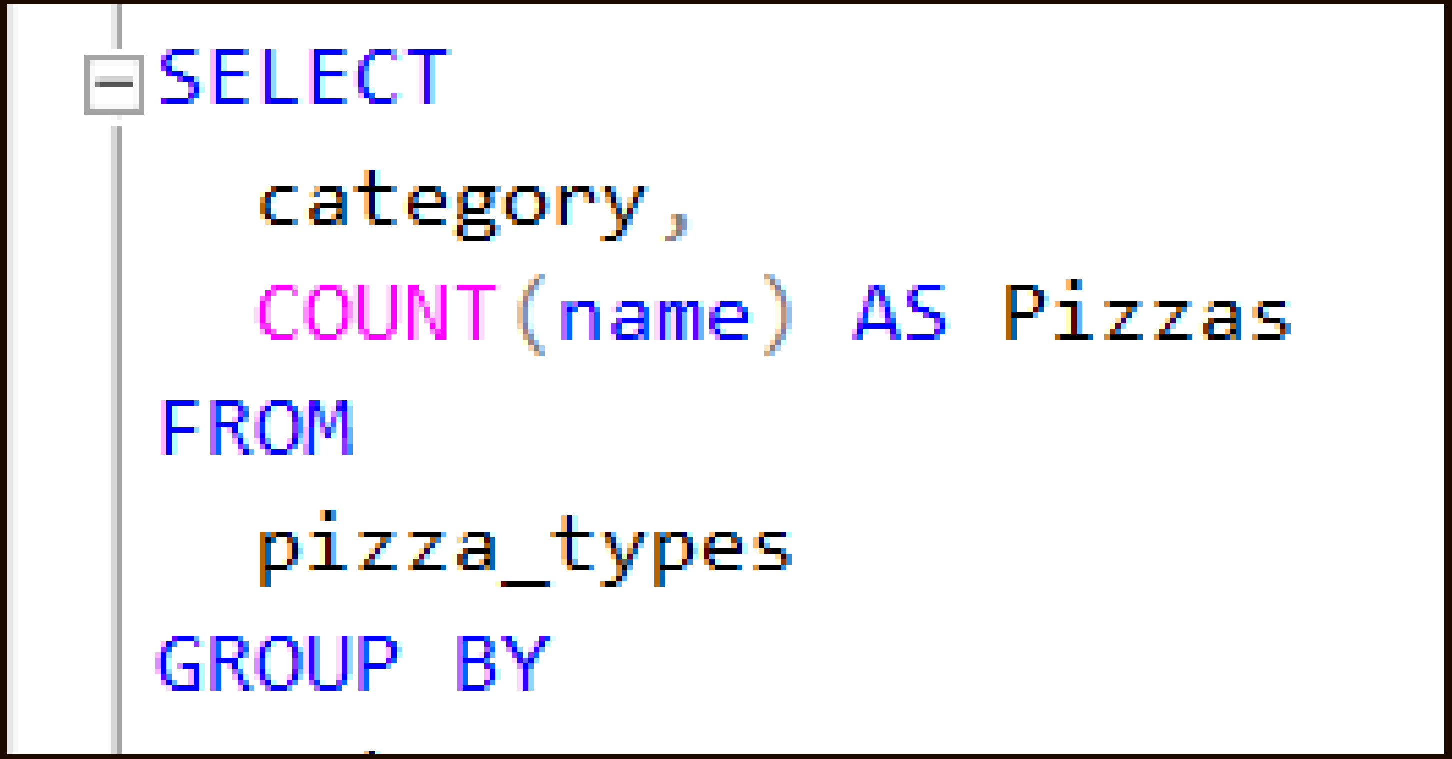


```
SELECT
   DISTINCT DATEPART(hour, time) AS hour,
   COUNT(order_id) AS orders
FROM
   orders
GROUP BY
   DATEPART(hour, time)
ORDER BY
   hour;
```

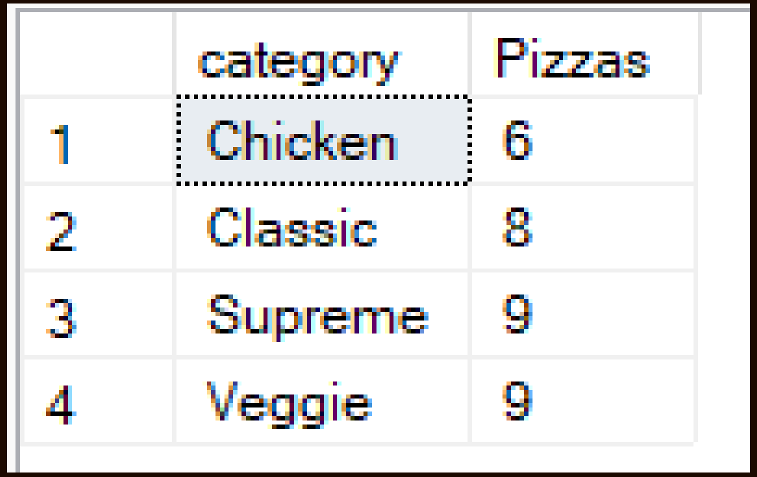


	hour	orders
1	9	1
2	10	8
3	11	1231
4	12	2520
5	13	2455
6	14	1472
7	15	1468
8	16	1920

CATEGORY-WISE DISTRIBUTION OF PIZZAS.

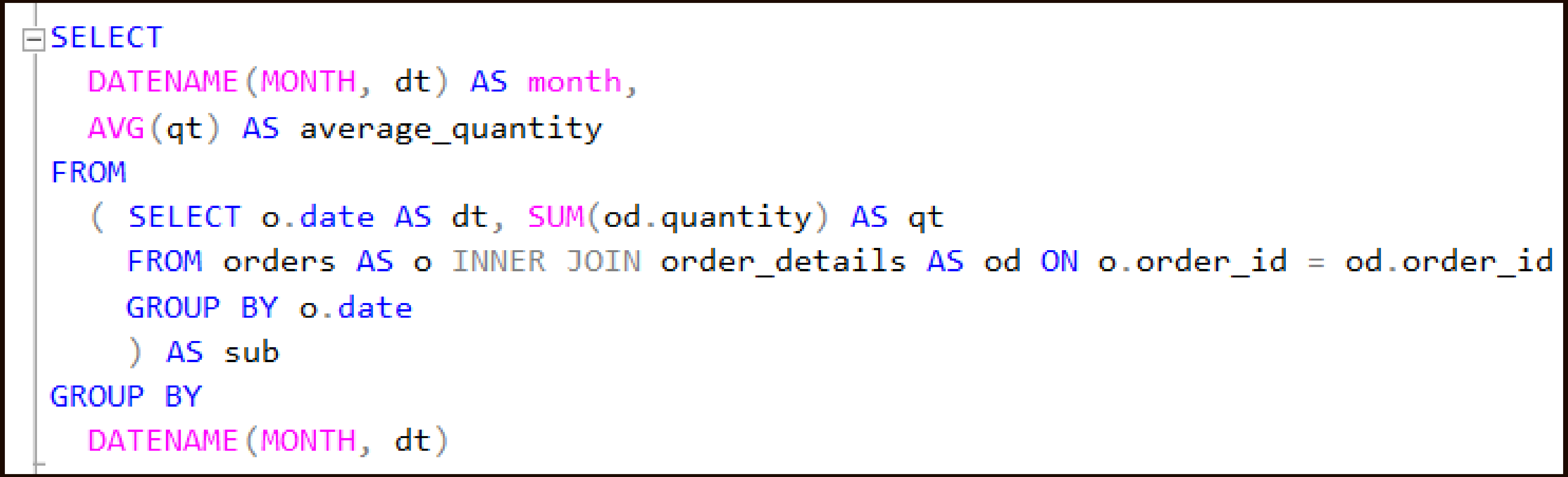


```
SELECT
  category,
  COUNT(name) AS Pizzas
FROM
  pizza_types
GROUP BY
```

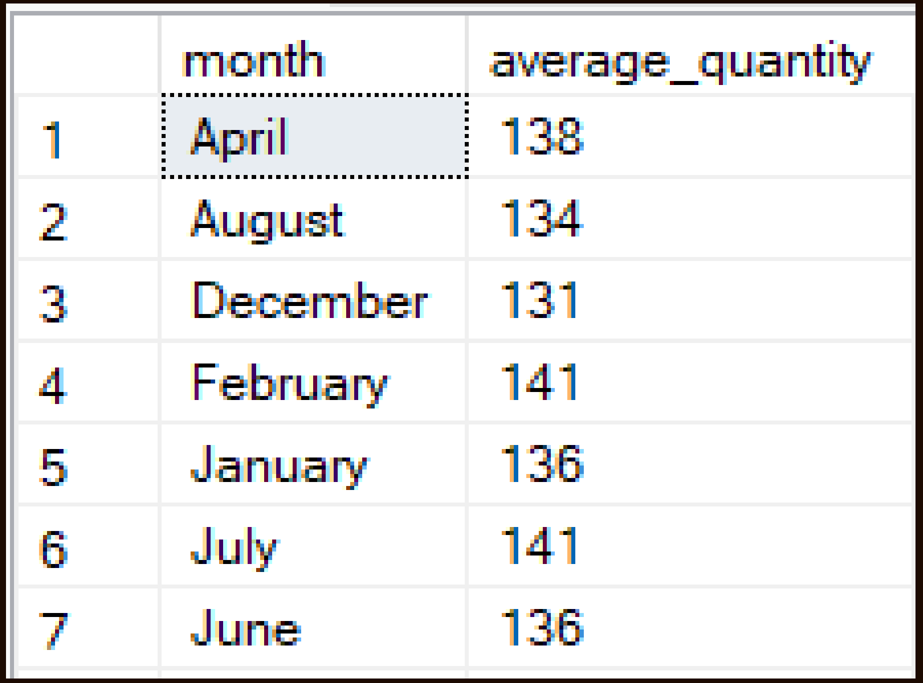


	category	Pizzas
1	Chicken	6
2	Classic	8
3	Supreme	9
4	Veggie	9

CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER MONTH.

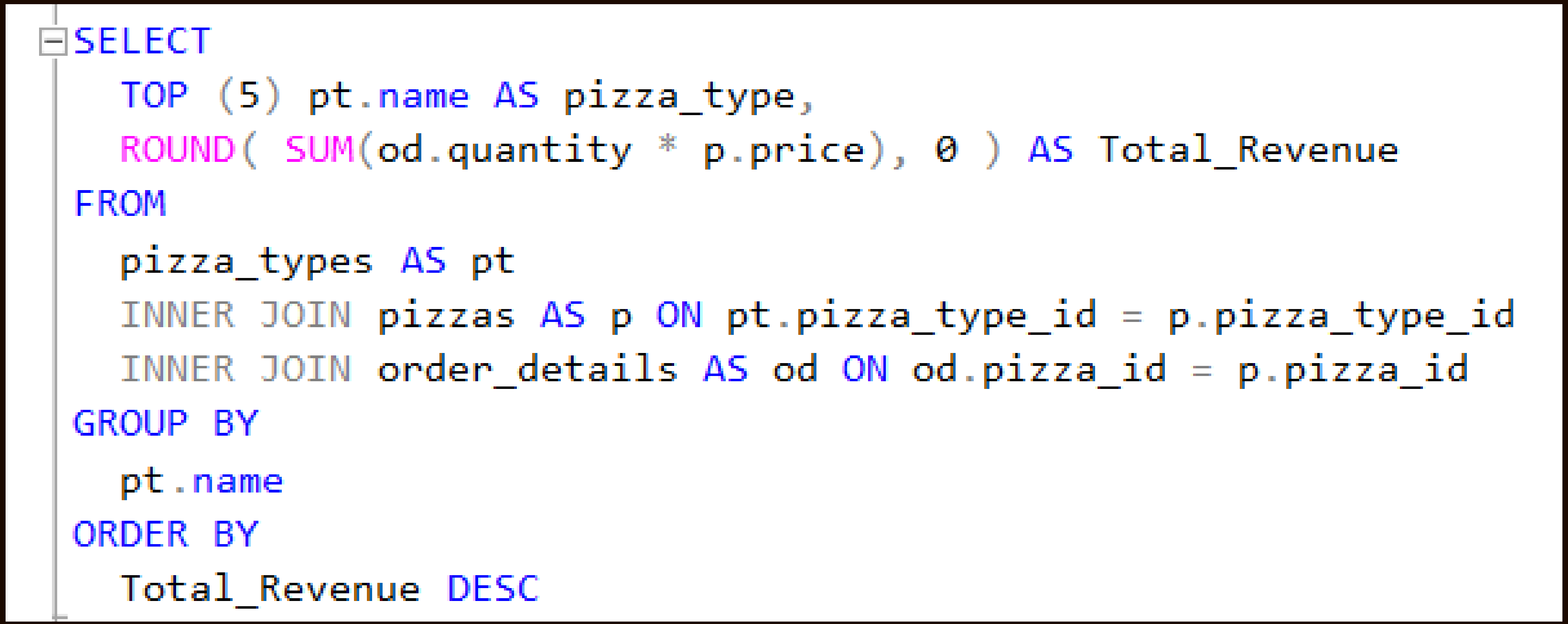


```
SELECT
  DATENAME(MONTH, dt) AS month,
  AVG(qt) AS average_quantity
FROM
  ( SELECT o.date AS dt, SUM(od.quantity) AS qt
    FROM orders AS o INNER JOIN order_details AS od ON o.order_id = od.order_id
    GROUP BY o.date
    ) AS sub
GROUP BY
  DATENAME(MONTH, dt)
```

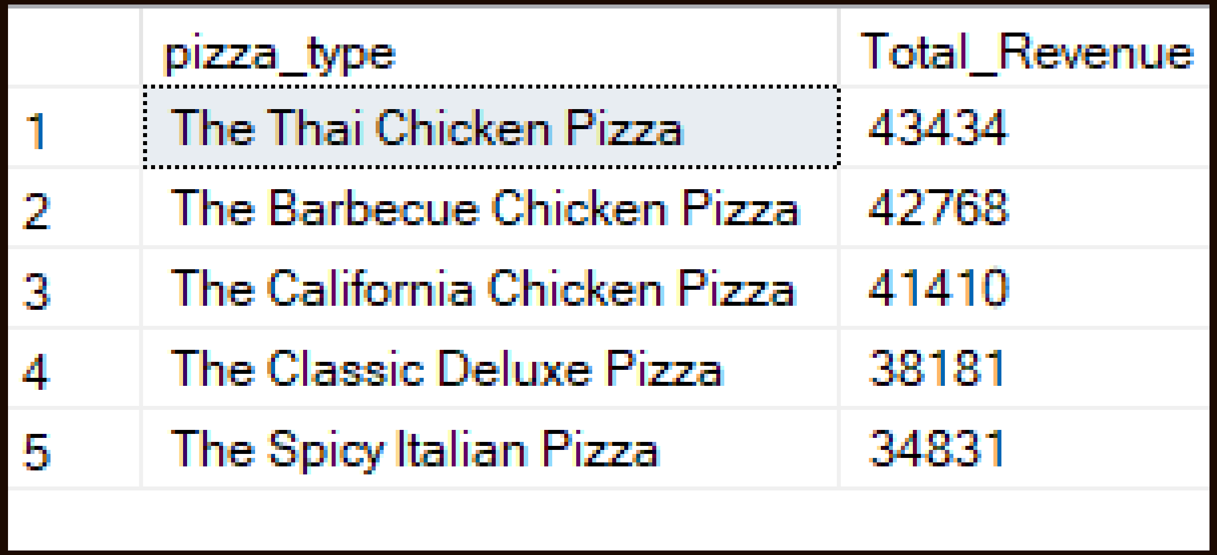


	month	average_quantity
1	April	138
2	August	134
3	December	131
4	February	141
5	January	136
6	July	141
7	June	136

TOP 5 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

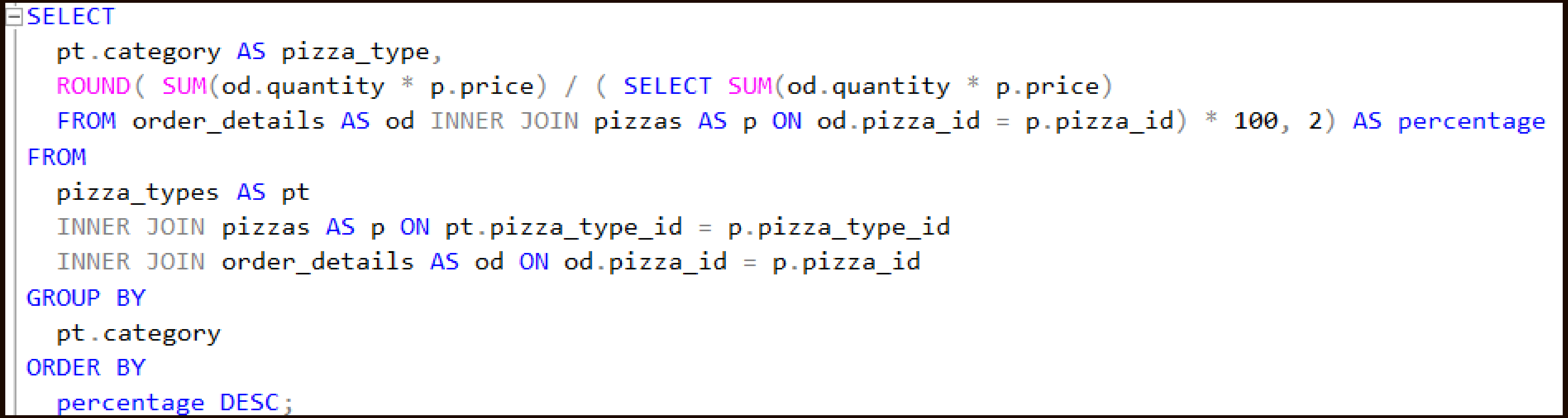


```
SELECT
  TOP (5) pt.name AS pizza_type,
  ROUND( SUM(od.quantity * p.price), 0 ) AS Total_Revenue
FROM
  pizza_types AS pt
  INNER JOIN pizzas AS p ON pt.pizza_type_id = p.pizza_type_id
  INNER JOIN order_details AS od ON od.pizza_id = p.pizza_id
GROUP BY
  pt.name
ORDER BY
  Total_Revenue DESC
```

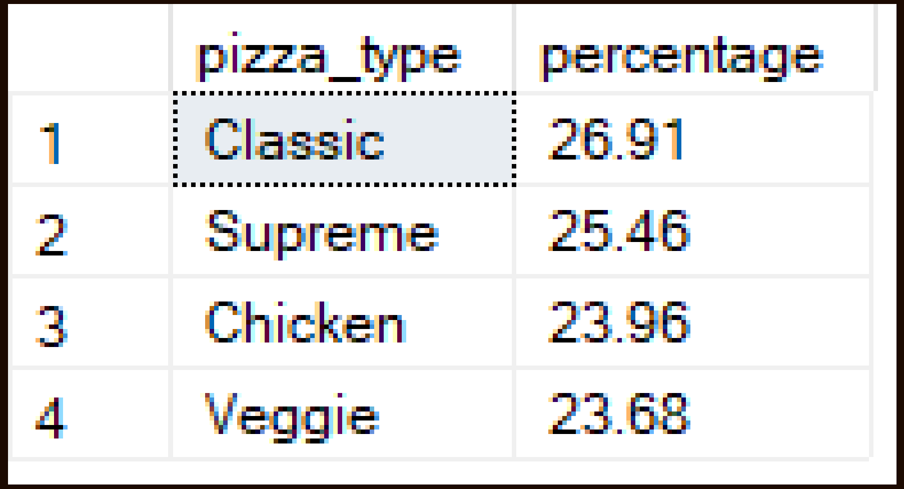


	pizza_type	Total_Revenue
1	The Thai Chicken Pizza	43434
2	The Barbecue Chicken Pizza	42768
3	The California Chicken Pizza	41410
4	The Classic Deluxe Pizza	38181
5	The Spicy Italian Pizza	34831

THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.

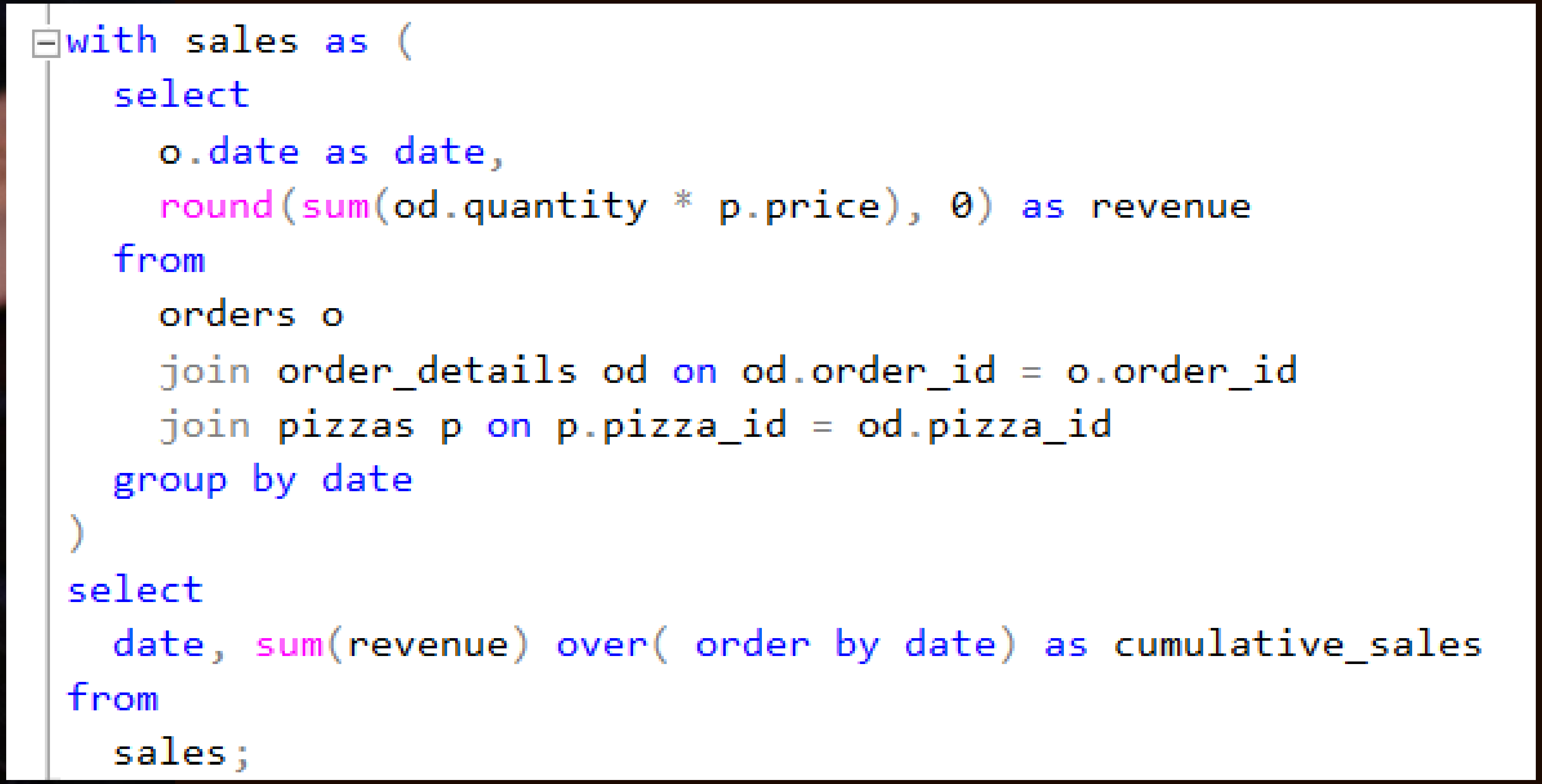


```
SELECT
  pt.category AS pizza_type,
  ROUND( SUM(od.quantity * p.price) / ( SELECT SUM(od.quantity * p.price)
  FROM order_details AS od INNER JOIN pizzas AS p ON od.pizza_id = p.pizza_id) * 100, 2) AS percentage
FROM
  pizza_types AS pt
  INNER JOIN pizzas AS p ON pt.pizza_type_id = p.pizza_type_id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

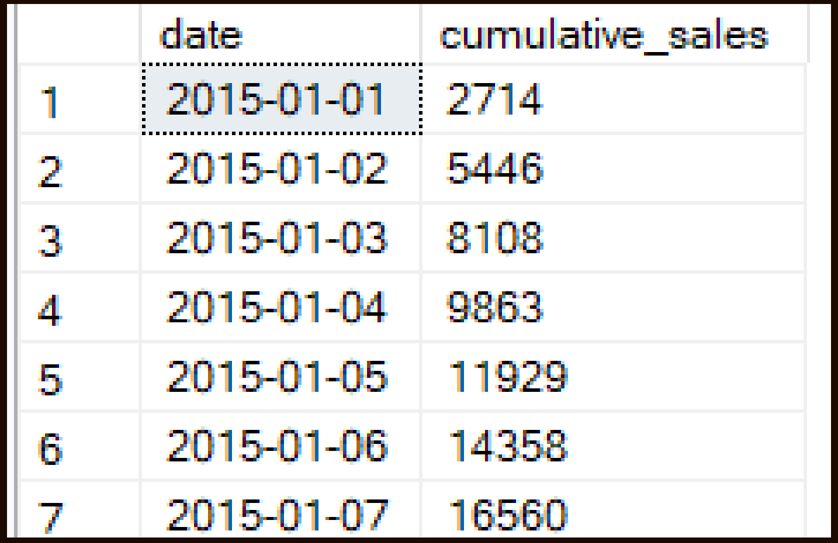


	pizza_type	percentage
1	Classic	26.91
2	Supreme	25.46
3	Chicken	23.96
4	Veggie	23.68

CUMULATIVE REVENUE GENERATED OVER TIME.

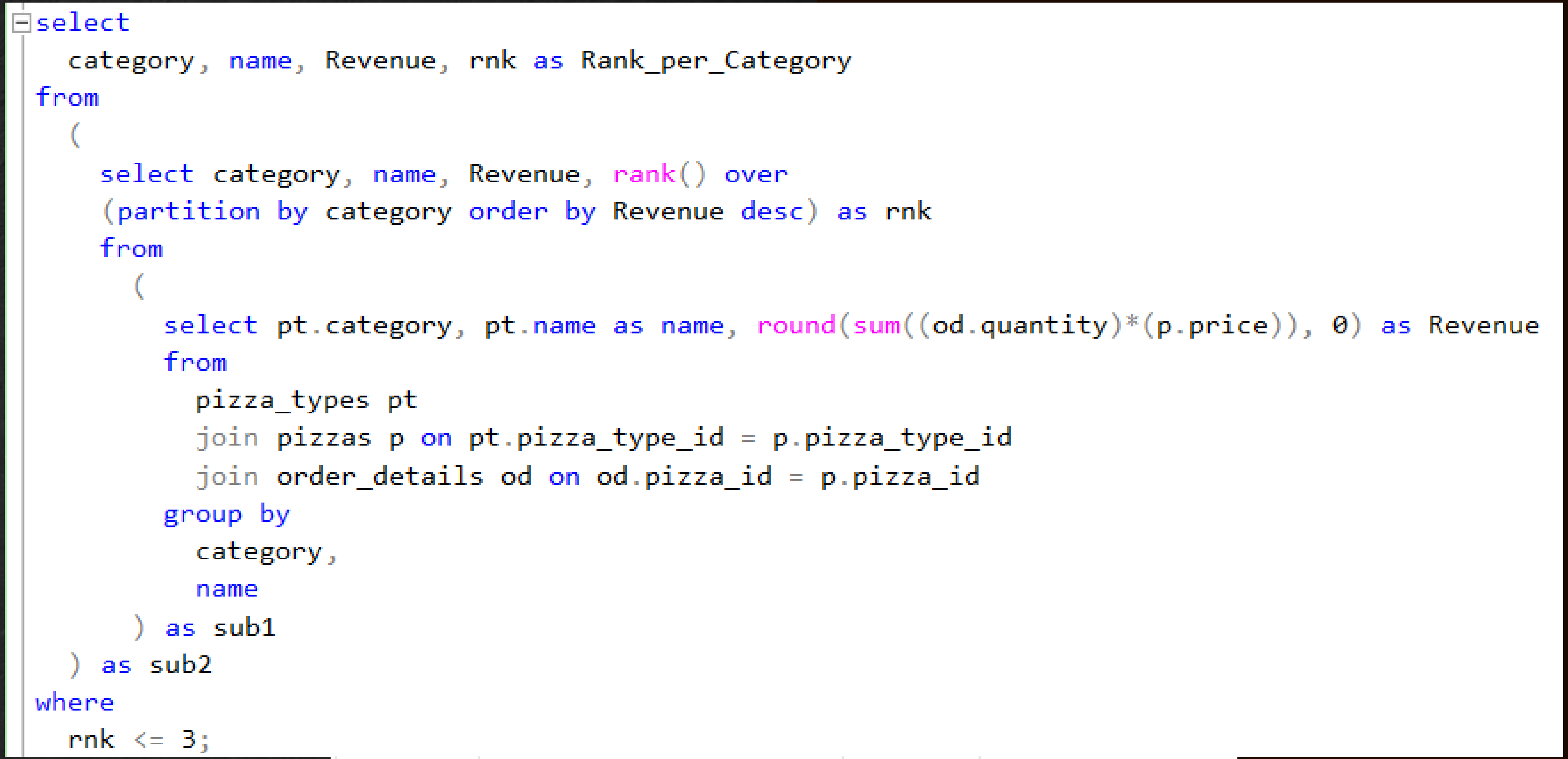


```
with sales as (
  select
    o.date as date,
    round(sum(od.quantity * p.price), 0) as revenue
  from
    orders o
    join order_details od on od.order_id = o.order_id
    join pizzas p on p.pizza_id = od.pizza_id
  group by date
)
select
  date, sum(revenue) over( order by date) as cumulative_sales
from
  sales;
```

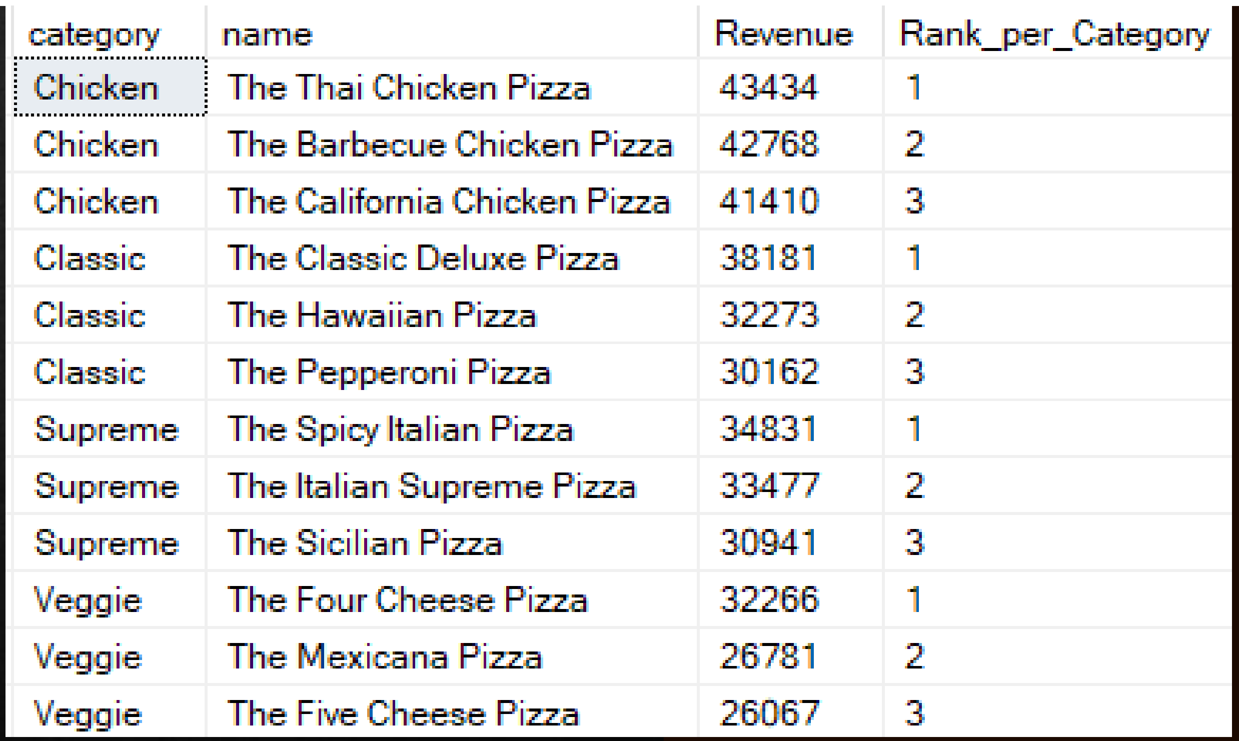


	date	cumulative_sales
1	2015-01-01	2714
2	2015-01-02	5446
3	2015-01-03	8108
4	2015-01-04	9863
5	2015-01-05	11929
6	2015-01-06	14358
7	2015-01-07	16560

TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY.



```
select
  category, name, Revenue, rnk as Rank_per_Category
from
  (
    select category, name, Revenue, rank() over
    (partition by category order by Revenue desc) as rnk
    from
      (
        select pt.category, pt.name as name, round(sum((od.quantity)*(p.price)), 0) as Revenue
        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
          category,
          name
      ) as sub1
  ) as sub2
where
  rnk <= 3;
```



category	name	Revenue	Rank_per_Category
Chicken	The Thai Chicken Pizza	43434	1
Chicken	The Barbecue Chicken Pizza	42768	2
Chicken	The California Chicken Pizza	41410	3
Classic	The Classic Deluxe Pizza	38181	1
Classic	The Hawaiian Pizza	32273	2
Classic	The Pepperoni Pizza	30162	3
Supreme	The Spicy Italian Pizza	34831	1
Supreme	The Italian Supreme Pizza	33477	2
Supreme	The Sicilian Pizza	30941	3
Veggie	The Four Cheese Pizza	32266	1
Veggie	The Mexicana Pizza	26781	2
Veggie	The Five Cheese Pizza	26067	3

KEY FINDINGS

The analysis of the pizza store database reveals several key findings.

- The total number of orders placed.
- The total revenue generated from pizza sales.
- The highest-priced pizza.
- The most common pizza size ordered.
- The distribution of orders by hour of the day.
- Category-wise distribution of pizzas.
- The revenue share by pizza category.
- The cumulative revenue over time.
- Rank wise Pizza type for each category.

These insights play a very important role in understaing the business operations and help in decision making in day to day running of the store.

THANK YOU