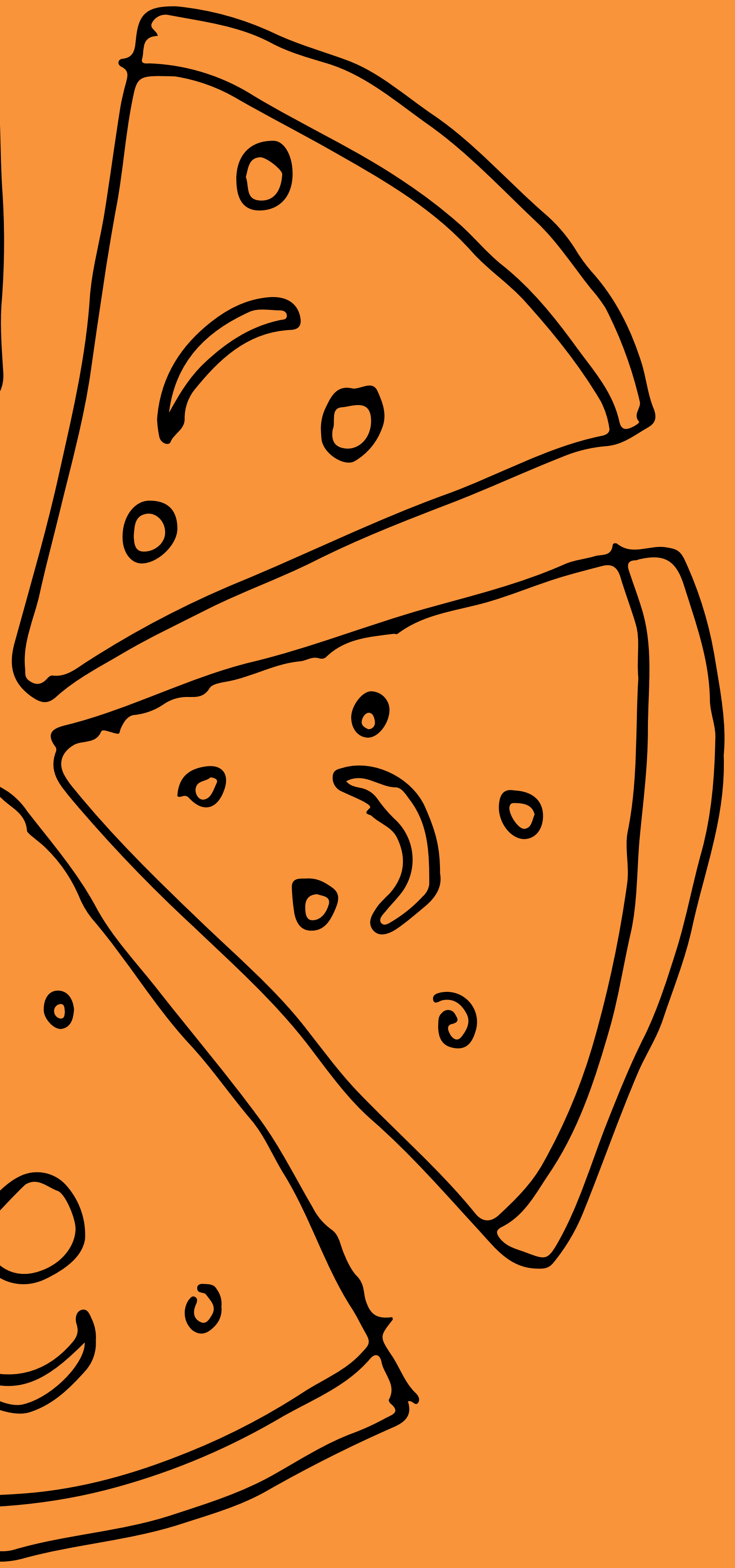
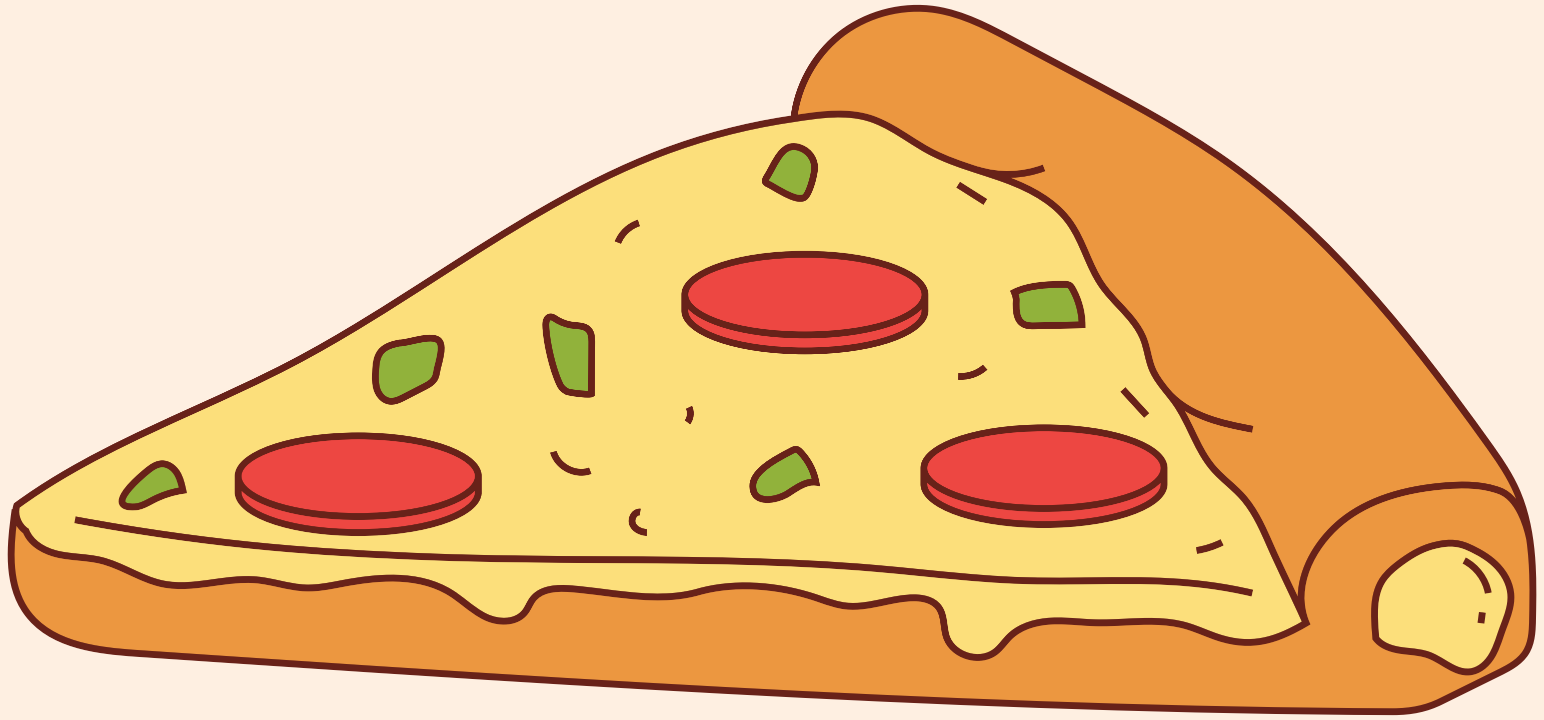


UNVEILING PIZZA PREFERENCES:

A DATA-DRIVEN ANALYSIS WITH SQL

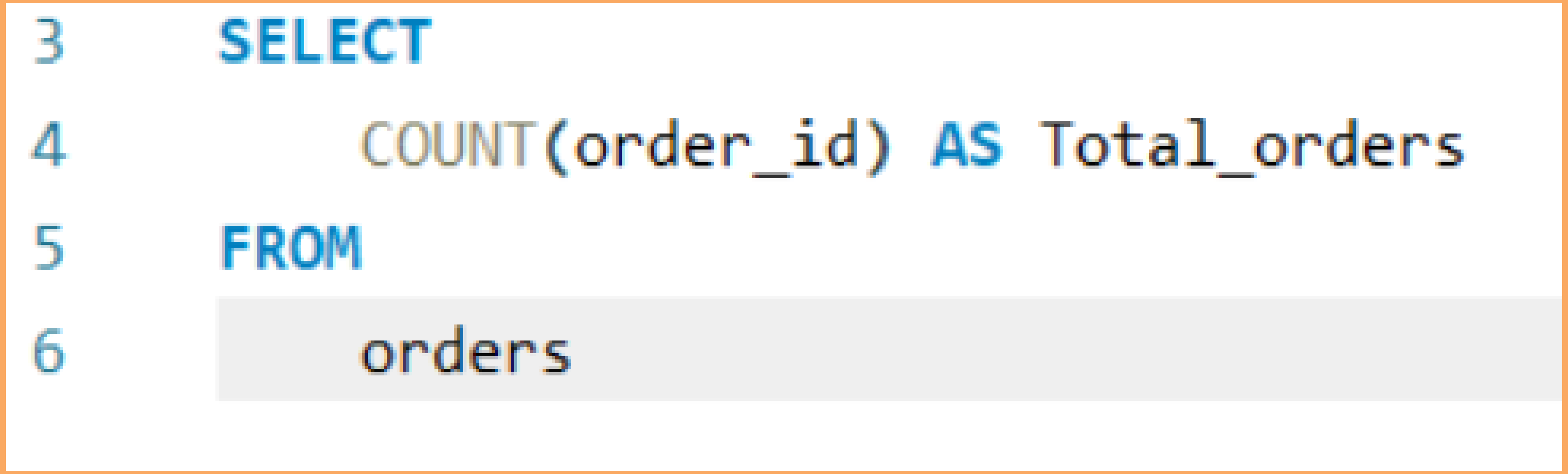
BY TEJINDER SINGH



- This project analyzes a database of pizza orders.
- We'll utilize SQL queries to extract valuable insights.
- Our goal is to understand customer behavior and sales patterns.

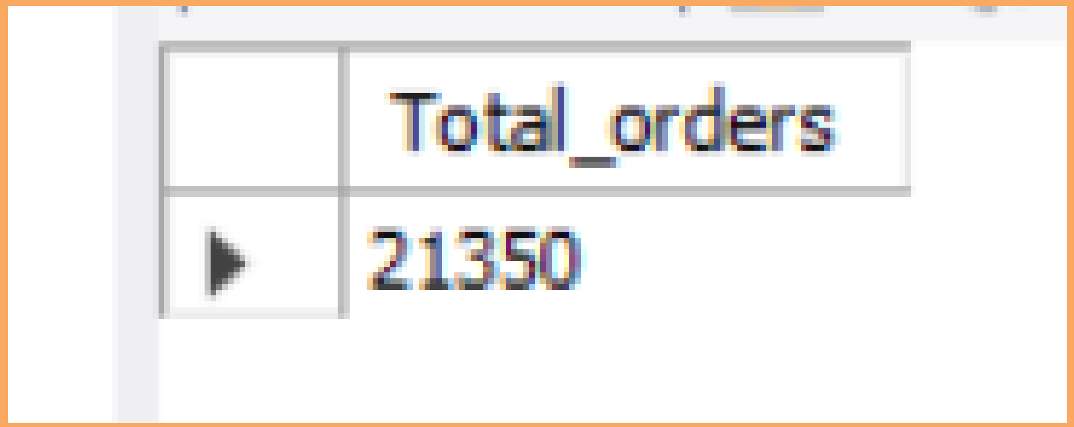
RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED

CODE



```sql
SELECT
    COUNT(order_id) AS Total_orders
FROM
    orders
```

OUTPUT

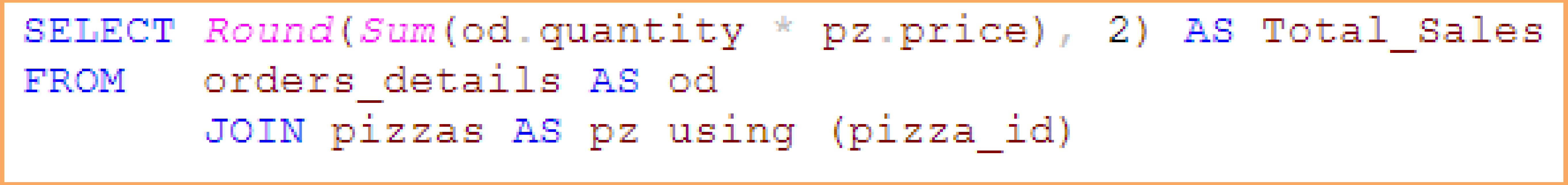


	Total_orders
▶	21350

THIS SLIDE DISPLAYS THE TOTAL NUMBER OF ORDERS PLACED IN OUR DATASET. THIS PROVIDES A STARTING POINT FOR UNDERSTANDING OVERALL CUSTOMER DEMAND.

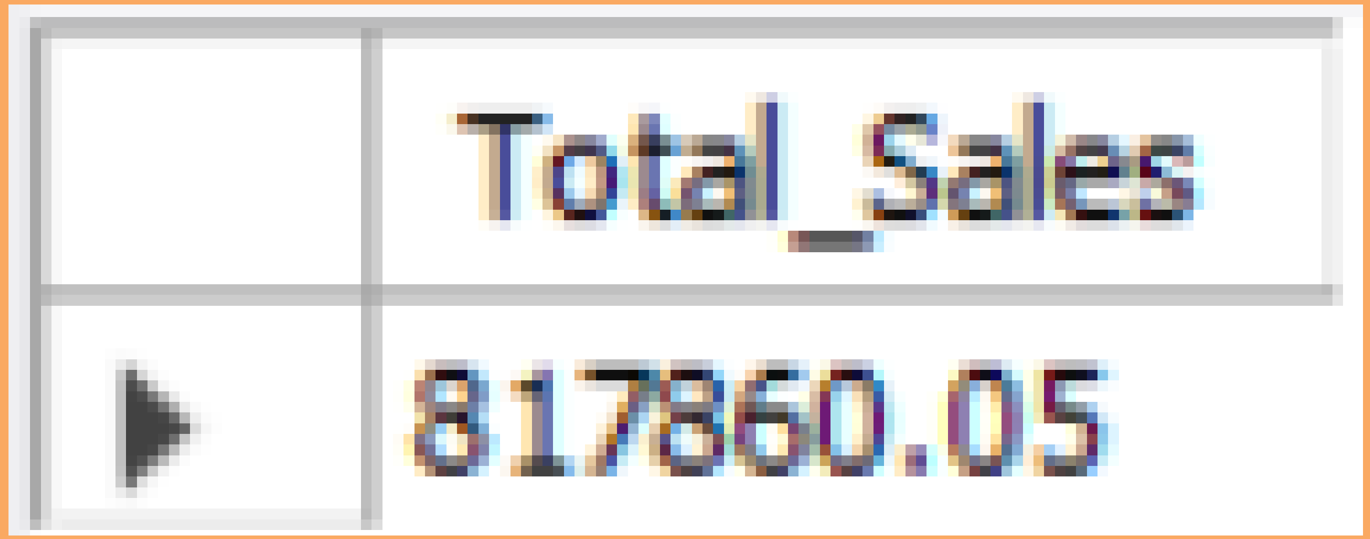
CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

CODE



```sql
SELECT Round(Sum(od.quantity * pz.price), 2) AS Total_Sales
FROM   orders_details AS od
       JOIN pizzas AS pz using (pizza_id)
```

OUTPUT

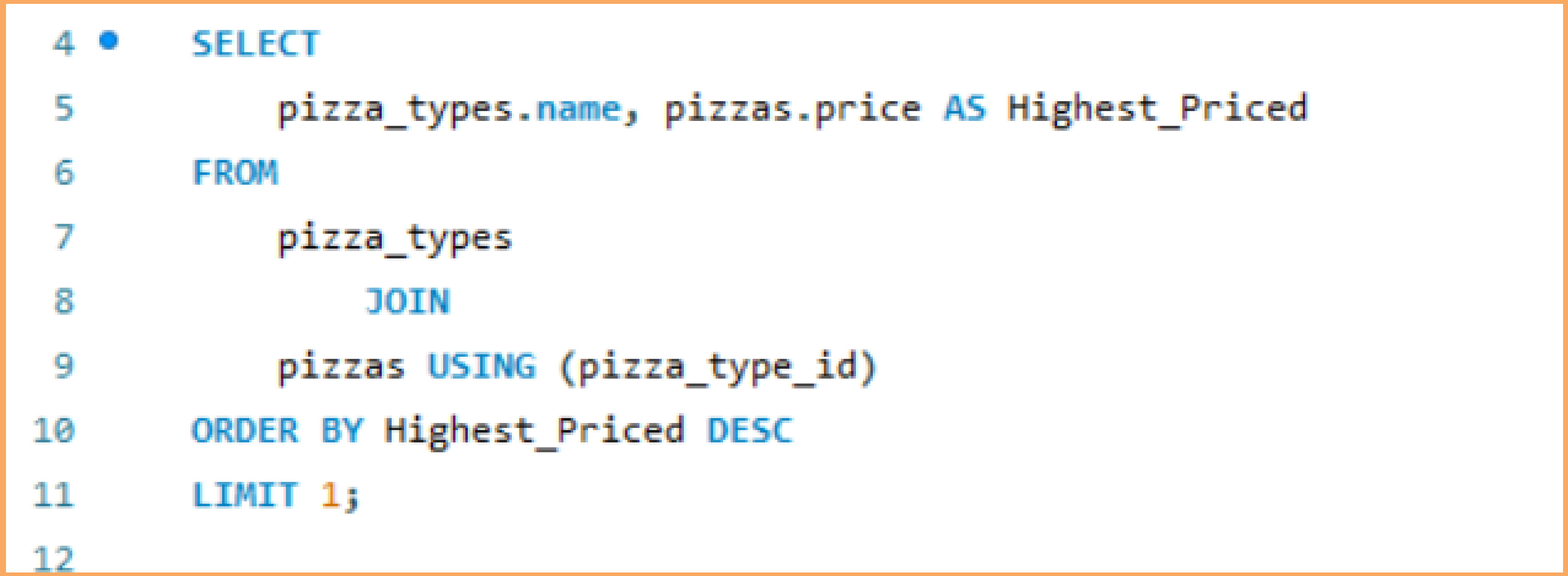


	Total_Sales
▶	817860.05

HERE, WE CALCULATE THE TOTAL REVENUE GENERATED SPECIFICALLY FROM PIZZA SALES. THIS HELPS US GAUGE THE SIGNIFICANCE OF PIZZA WITHIN THE OVERALL BUSINESS.

IDENTIFY THE HIGHEST-PRICED PIZZA.

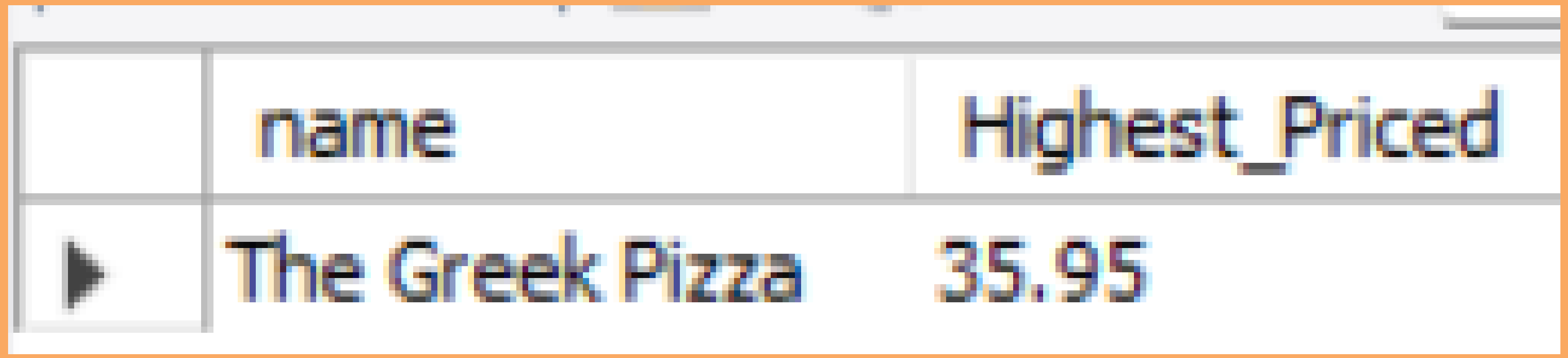
CODE



```sql
SELECT
    pizza_types.name, pizzas.price AS Highest_Priced
FROM
    pizza_types
        JOIN
    pizzas USING (pizza_type_id)
ORDER BY Highest_Priced DESC
LIMIT 1;

```

OUTPUT

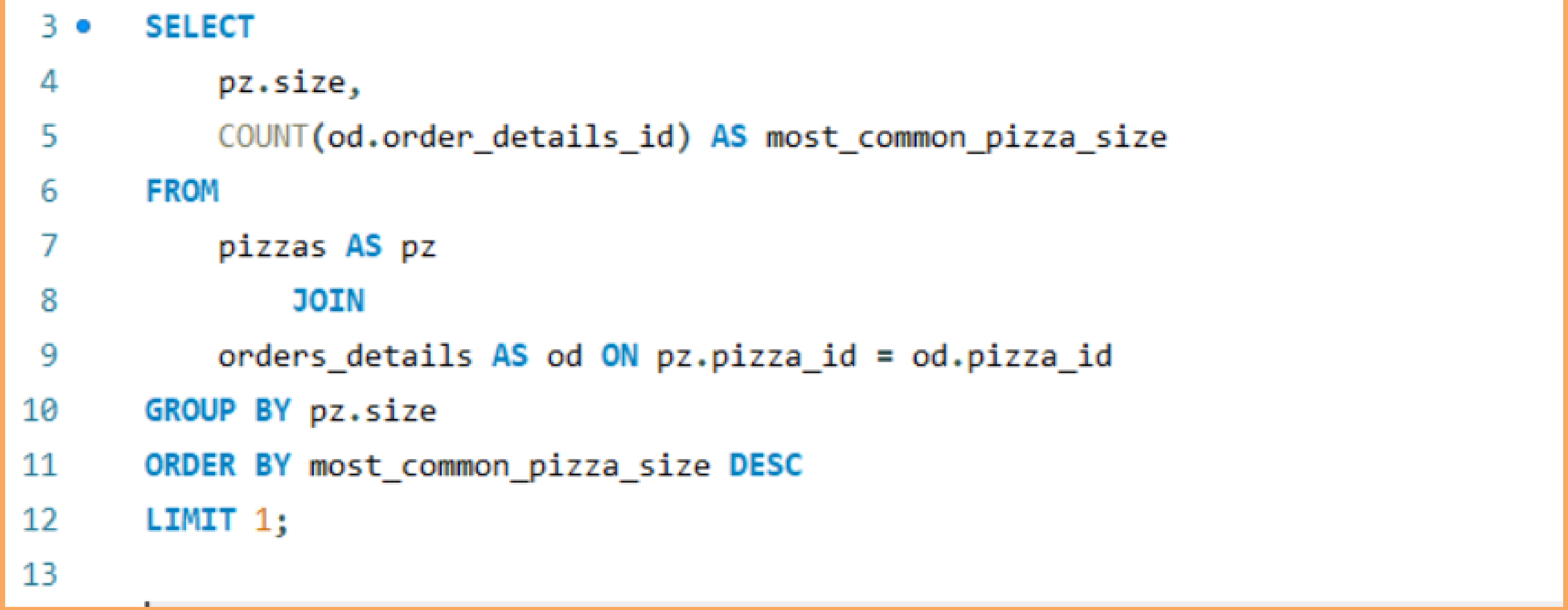


name	Highest_Priced
The Greek Pizza	35.95

THIS SLIDE REVEALS THE PIZZA THAT COMMANDS THE HIGHEST PRICE POINT. THIS COULD BE DUE TO PREMIUM INGREDIENTS OR UNIQUE COMBINATIONS.

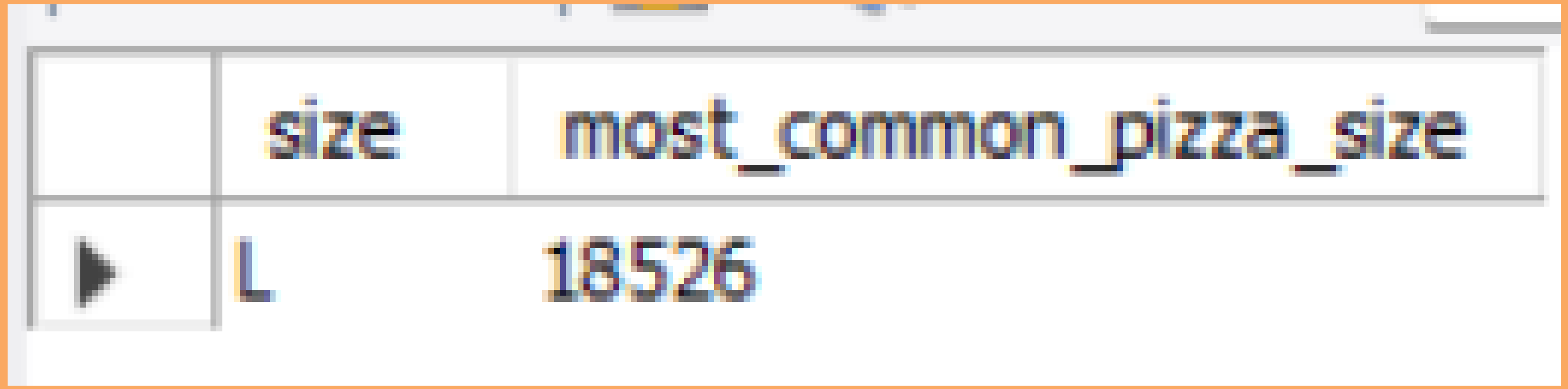
IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

CODE



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

OUTPUT

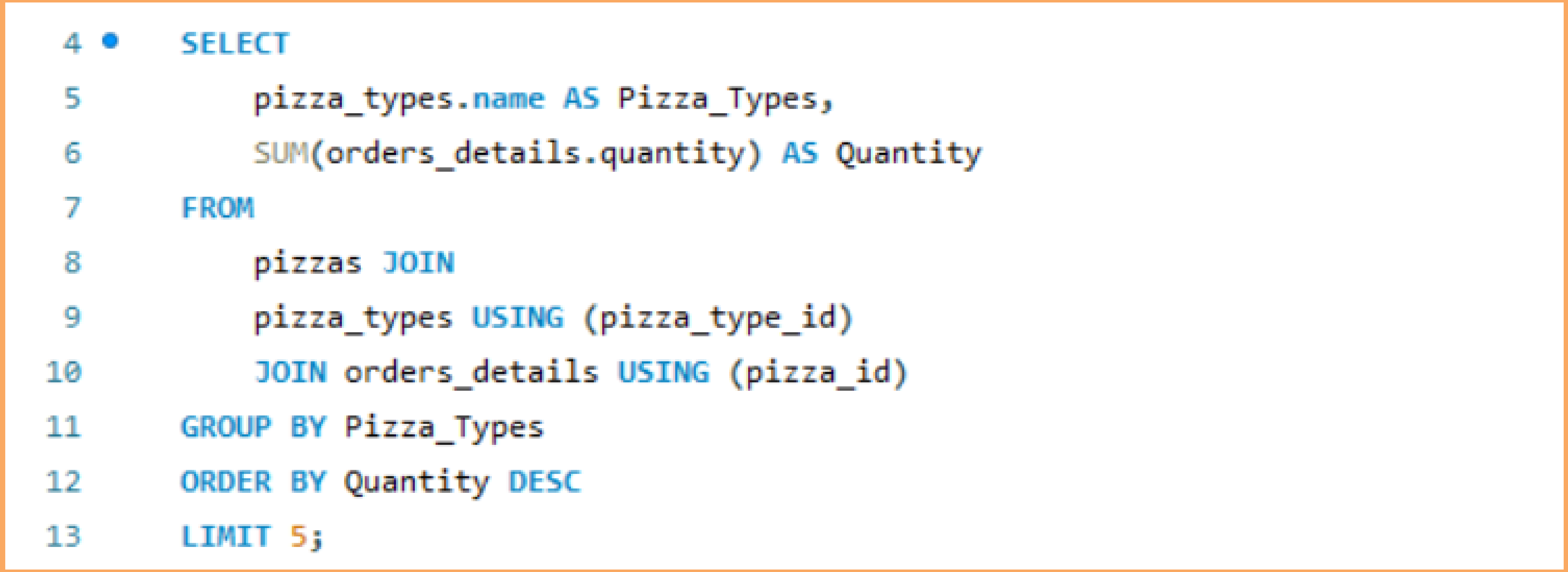


size	most_common_pizza_size
L	18526

WE'LL DETERMINE THE SIZE (SMALL, MEDIUM, LARGE) THAT ENJOYS THE MOST POPULARITY AMONG CUSTOMERS. UNDERSTANDING SIZE PREFERENCES CAN HELP BUSINESSES OPTIMIZE INVENTORY AND PRICING STRATEGIES.

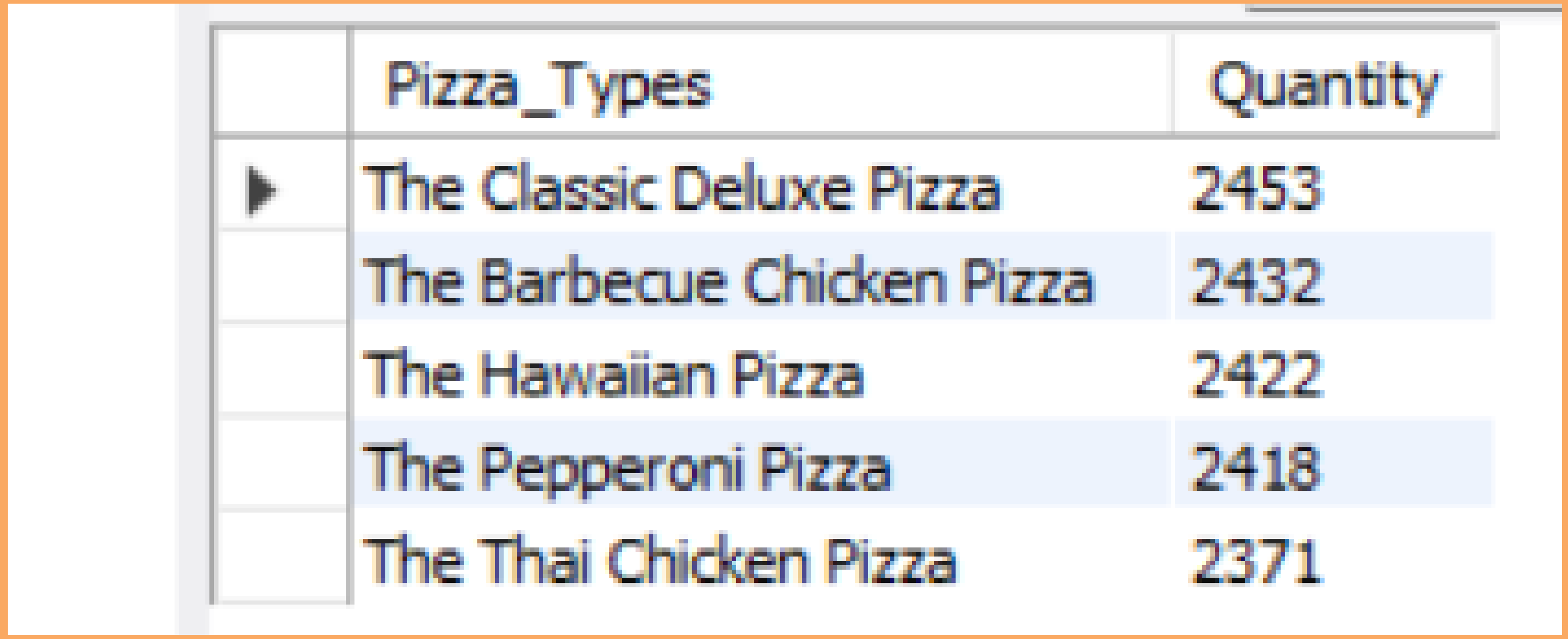
LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.

CODE



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

OUTPUT

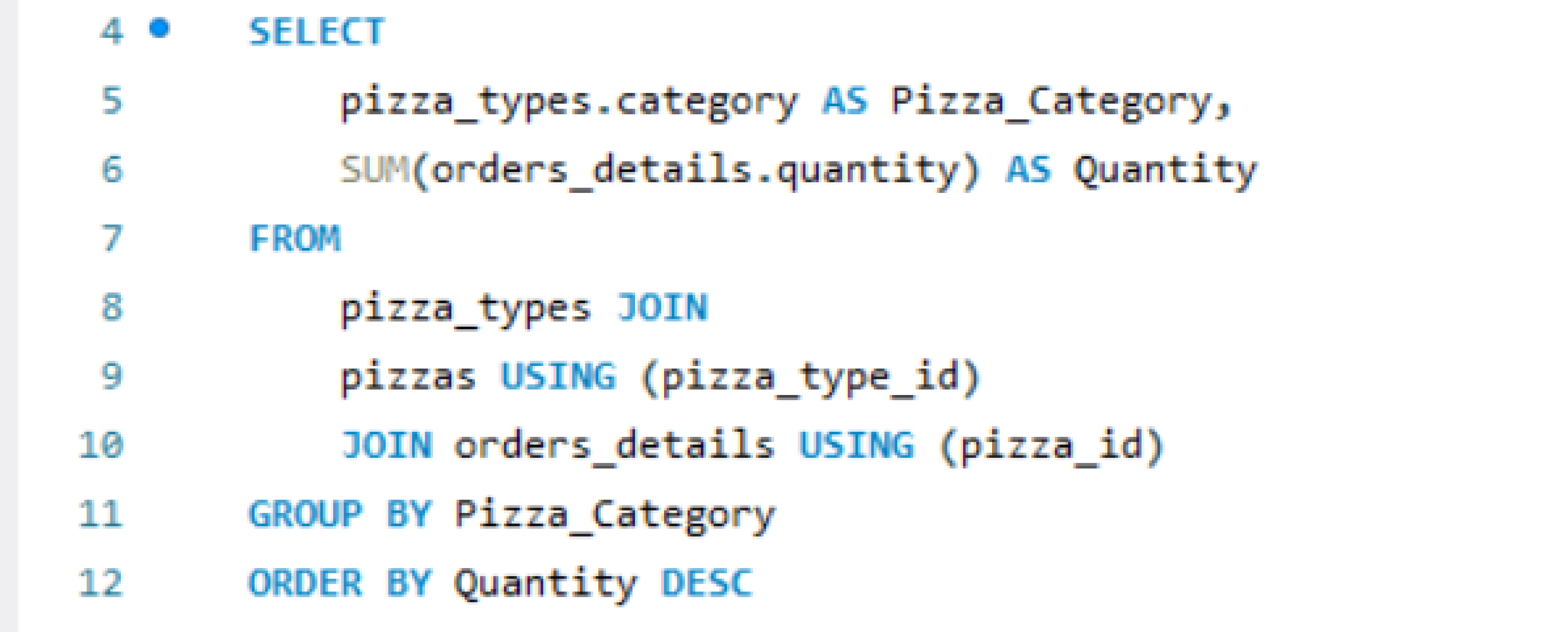


Pizza_Types	Quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

THE TOP 5 PIZZA TYPES IN TERMS OF THE NUMBER OF ORDERS PLACED. THIS REVEALS CUSTOMER FAVORITES AND CAN INFORM FUTURE MARKETING CAMPAIGNS.

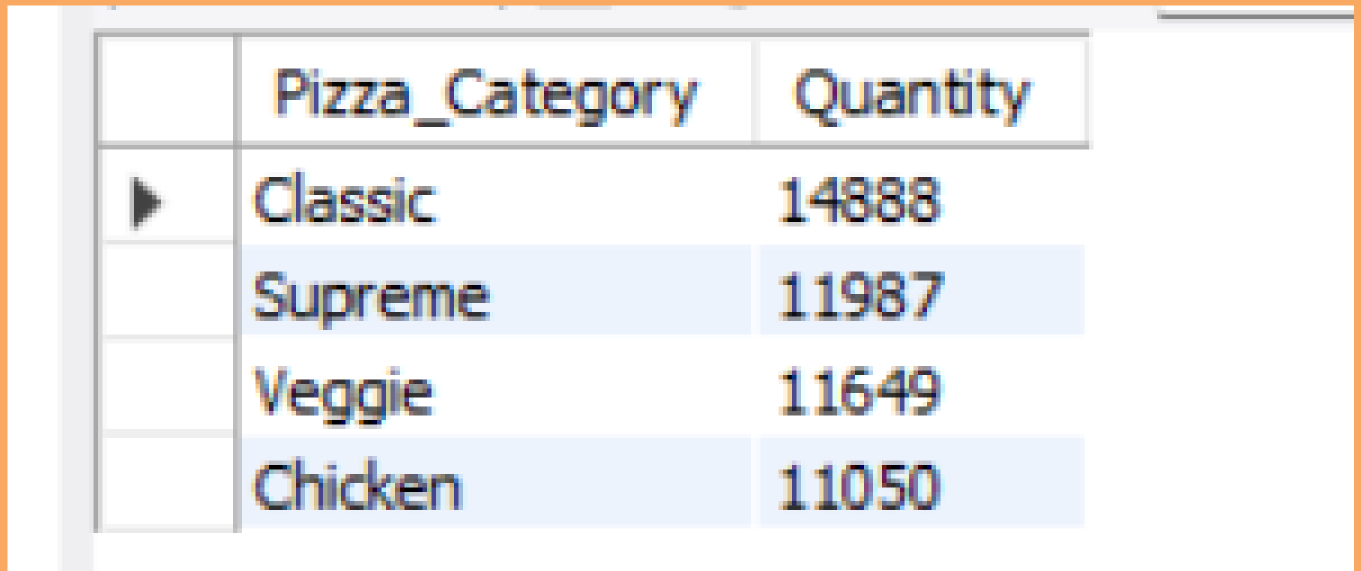
JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

CODE



```sql
SELECT
    pizza_types.category AS Pizza_Category,
    SUM(orders_details.quantity) AS Quantity
FROM
    pizza_types JOIN
    pizzas USING (pizza_type_id)
    JOIN orders_details USING (pizza_id)
GROUP BY Pizza_Category
ORDER BY Quantity DESC
```

OUTPUT

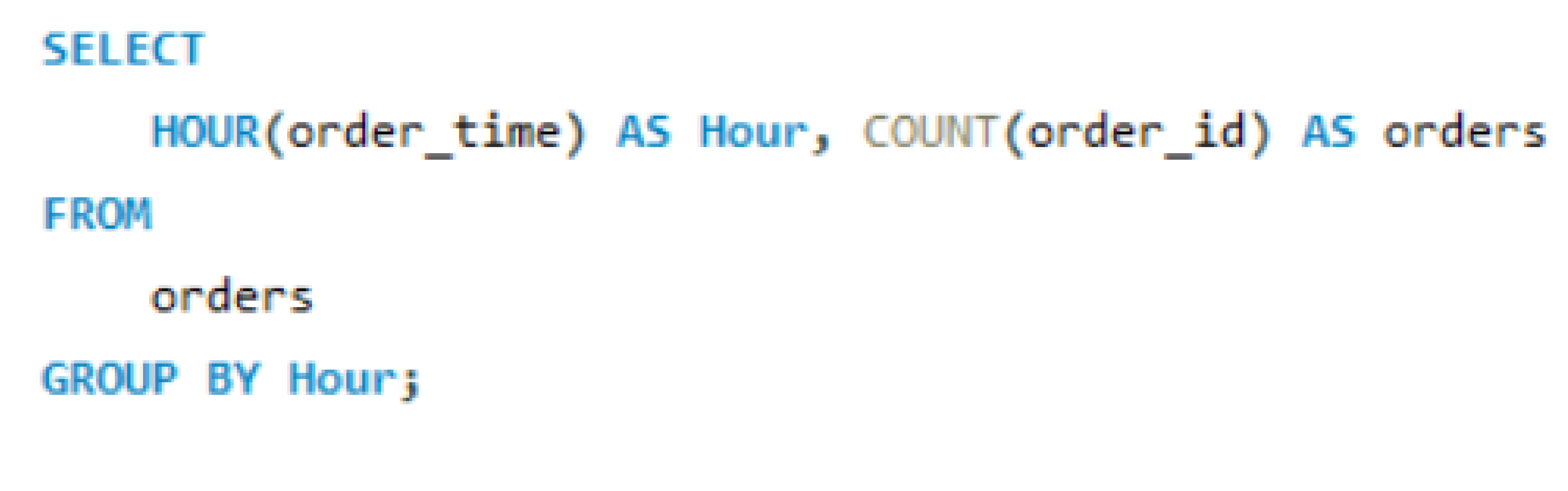


Pizza_Category	Quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

WE'LL JOIN RELEVANT TABLES TO UNCOVER THE TOTAL QUANTITY OF PIZZAS ORDERED WITHIN EACH CATEGORY. THIS HELPS US UNDERSTAND CUSTOMER PREFERENCES ACROSS DIFFERENT PIZZA TYPES.

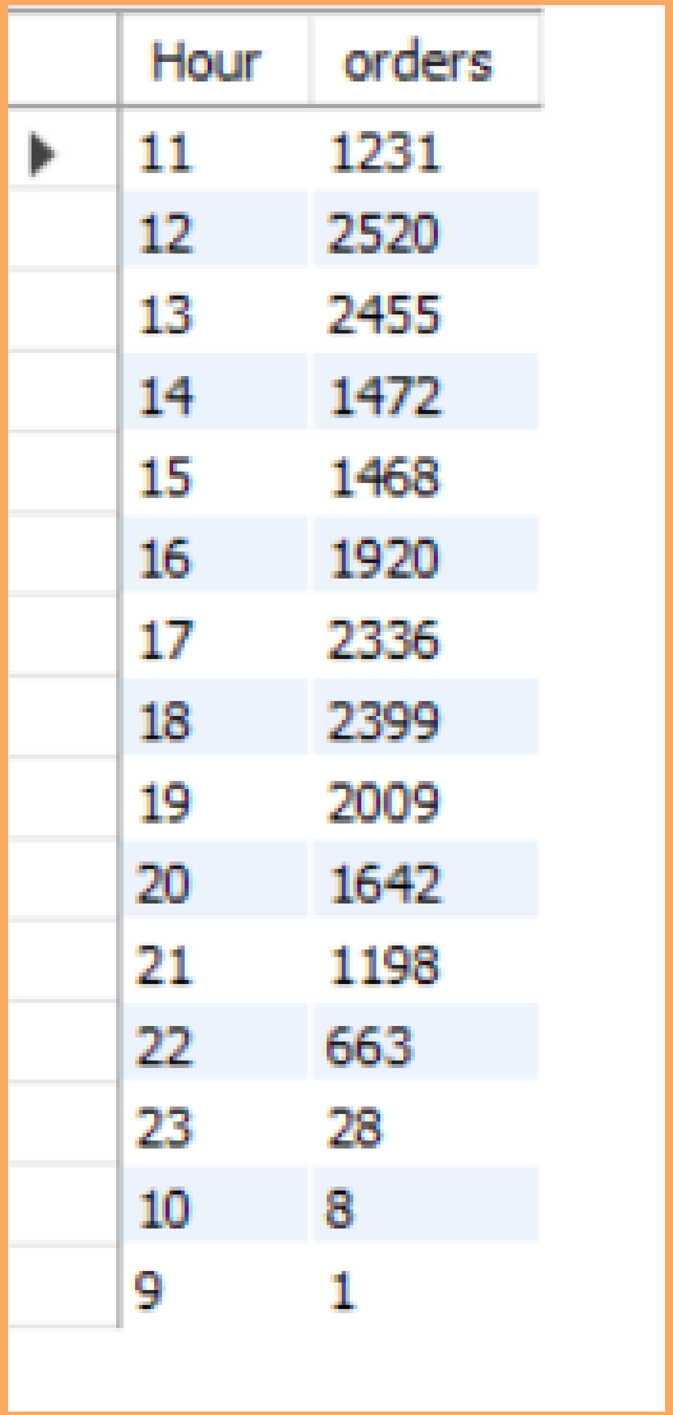
DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

CODE



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

OUTPUT

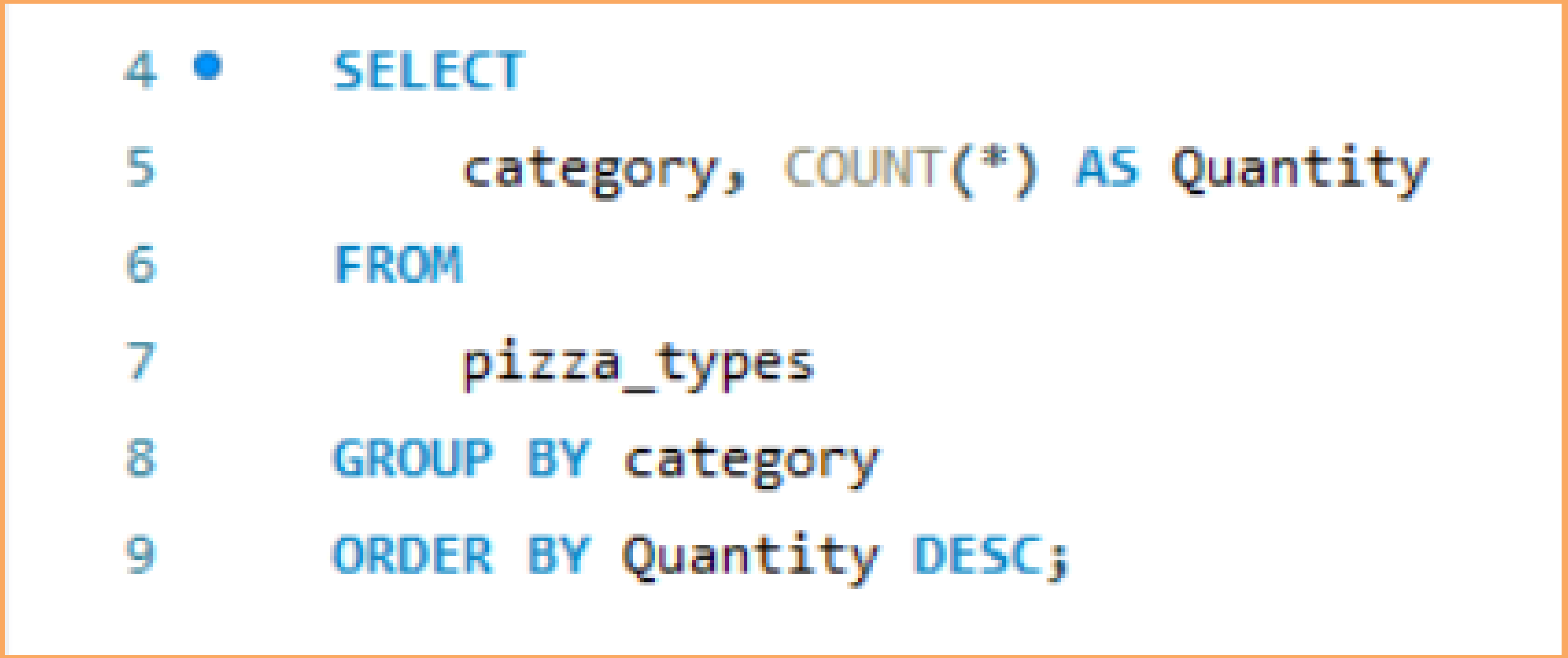


Hour	orders
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

THIS SLIDE ANALYZES ORDER PATTERNS THROUGHOUT THE DAY. WE'LL SEE IF THERE ARE PEAK ORDERING HOURS, WHICH CAN BE CRUCIAL FOR STAFFING AND PROMOTIONAL STRATEGIES.

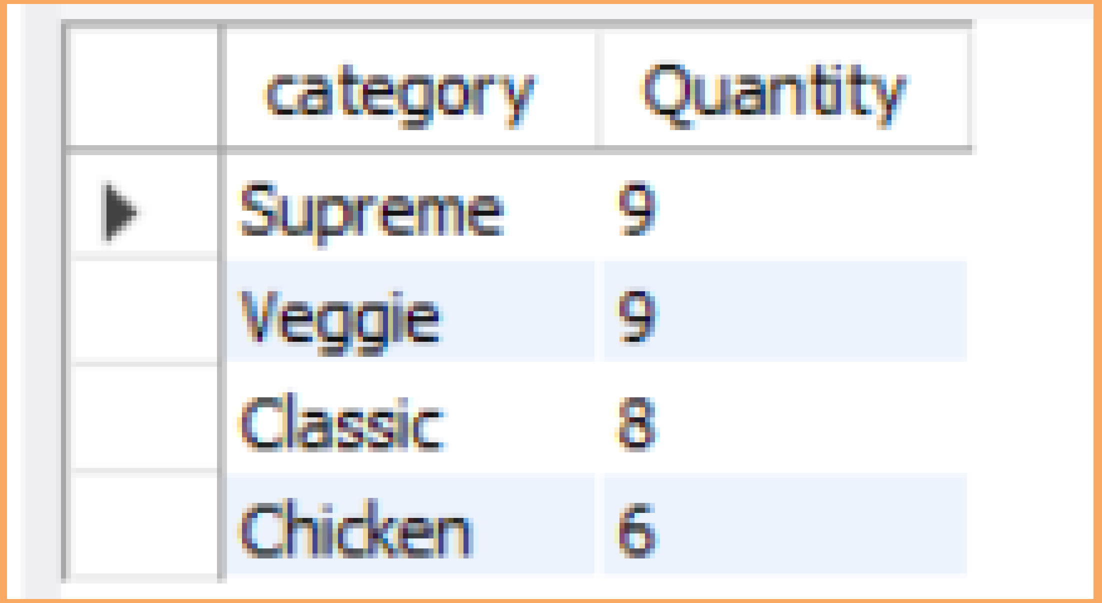
JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS

CODE



```
SELECT
    category, COUNT(*) AS Quantity
FROM
    pizza_types
GROUP BY category
ORDER BY Quantity DESC;
```

OUTPUT

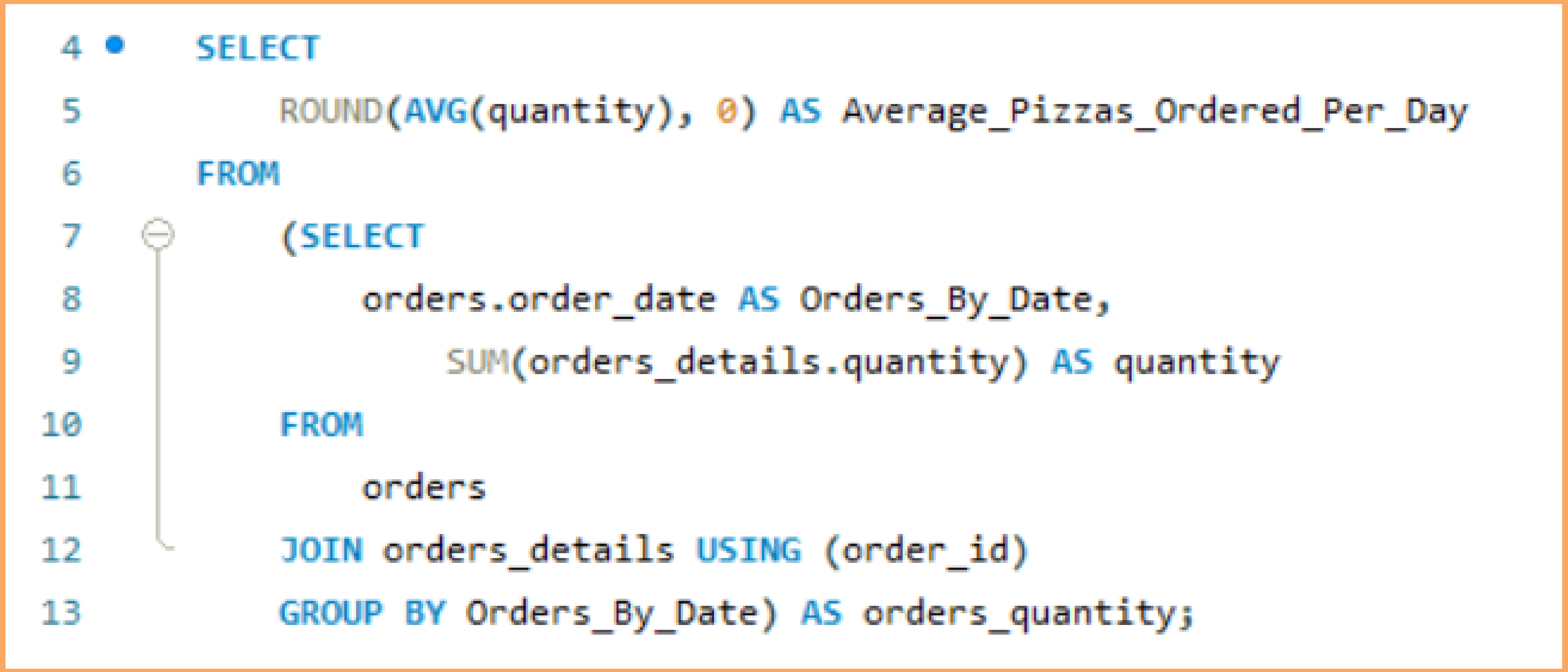


category	Quantity
Supreme	9
Veggie	9
Classic	8
Chicken	6

WE'LL AGAIN LEVERAGE JOINS TO ANALYZE PIZZA DISTRIBUTION WITHIN EACH CATEGORY. THIS PROVIDES A MORE GRANULAR VIEW OF CUSTOMER PREFERENCES ACROSS DIFFERENT PIZZA TYPES.

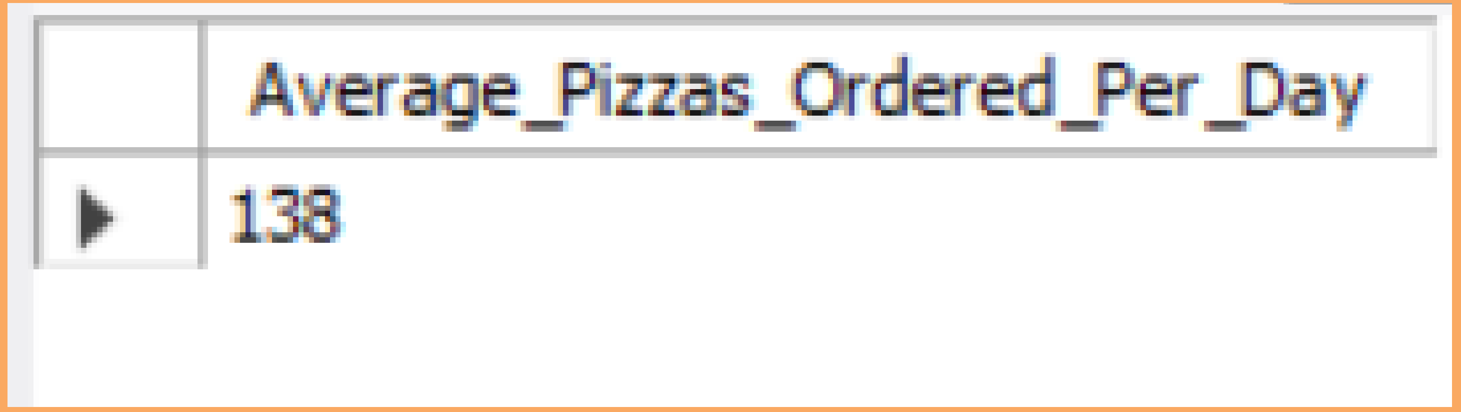
GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

CODE



```
SELECT
    ROUND(AVG(quantity), 0) AS Average_Pizzas_Ordered_Per_Day
FROM
    (SELECT
        orders.order_date AS Orders_By_Date,
            SUM(orders_details.quantity) AS quantity
    FROM
        orders
    JOIN orders_details USING (order_id)
    GROUP BY Orders_By_Date) AS orders_quantity;
```

OUTPUT

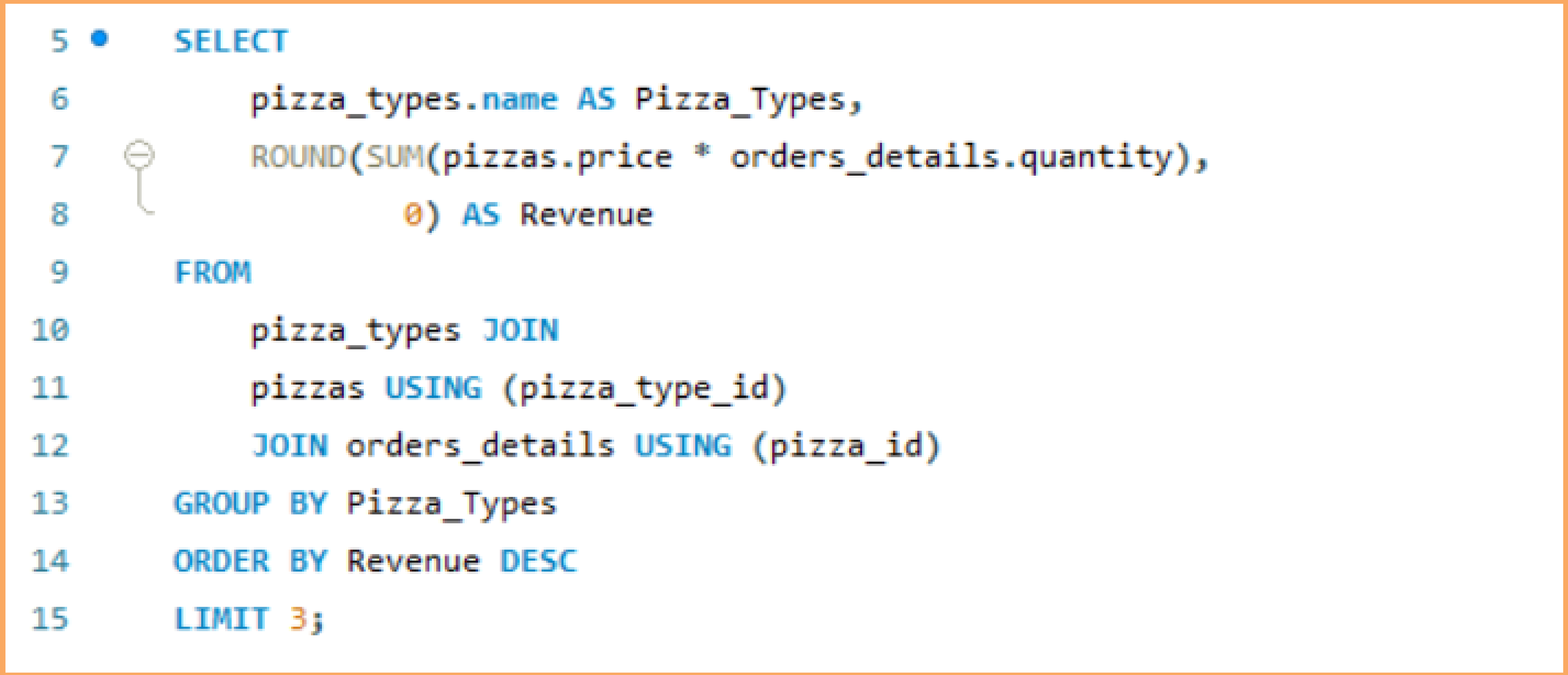


	Average_Pizzas_Ordered_Per_Day
▶	138

THIS SLIDE EXAMINES TRENDS IN THE DAILY AVERAGE NUMBER OF ORDERS PLACED. IDENTIFYING ANY SEASONAL SPIKES OR VARIATIONS CAN BE VALUABLE FOR BUSINESSES

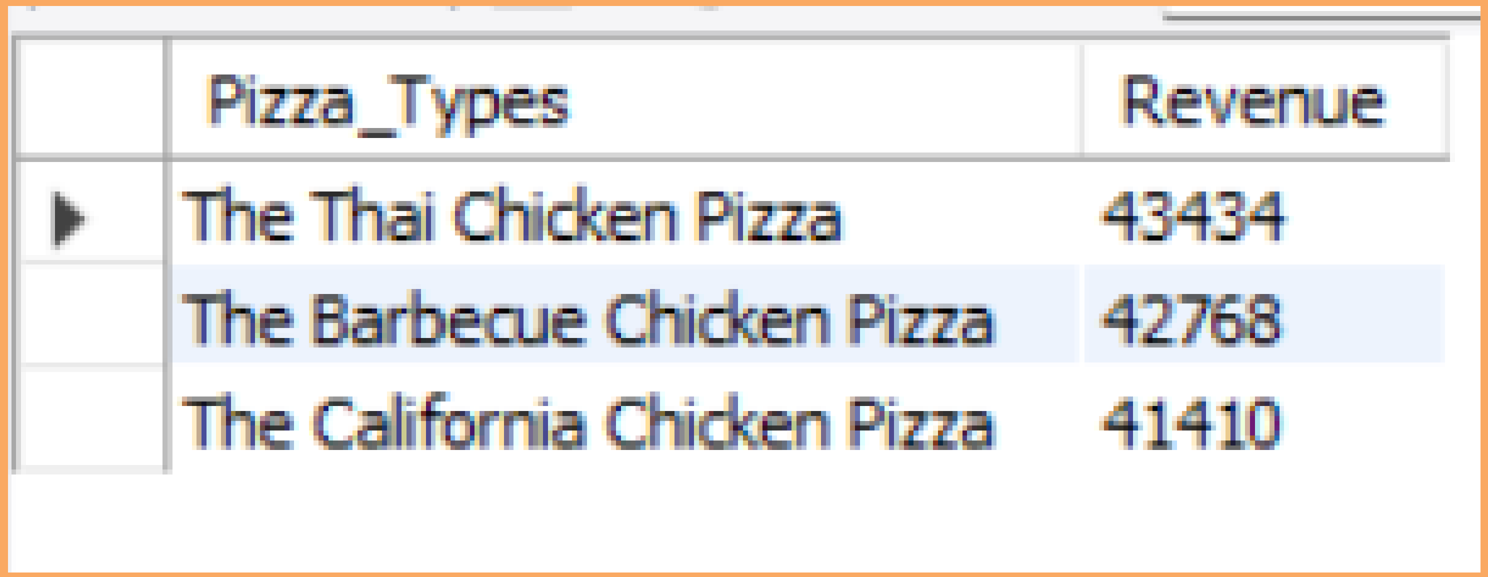
DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

CODE



```
SELECT
    pizza_types.name AS Pizza_Types,
    ROUND(SUM(pizzas.price * orders_details.quantity),
            0) AS Revenue
FROM
    pizza_types JOIN
    pizzas USING (pizza_type_id)
    JOIN orders_details USING (pizza_id)
GROUP BY Pizza_Types
ORDER BY Revenue DESC
LIMIT 3;
```

OUTPUT

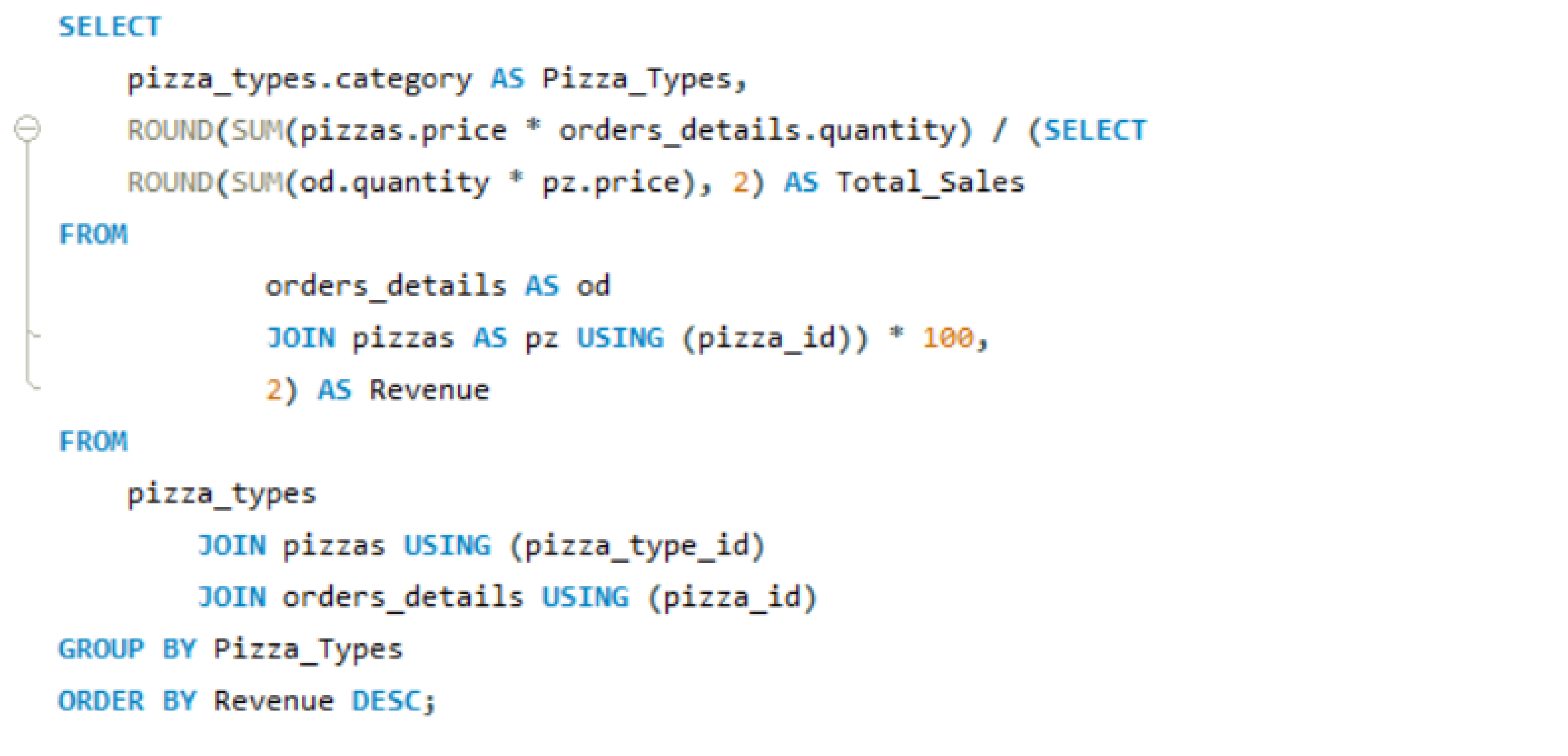


Pizza_Types	Revenue
The Thai Chicken Pizza	43434
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41410

WE SHIFT OUR FOCUS TO REVENUE INSTEAD OF QUANTITY. THIS REVEALS THE PIZZA TYPES THAT CONTRIBUTE THE MOST TO THE BUSINESS'S BOTTOM LINE, POTENTIALLY DUE TO HIGHER PRICES OR LARGER ORDER SIZES.

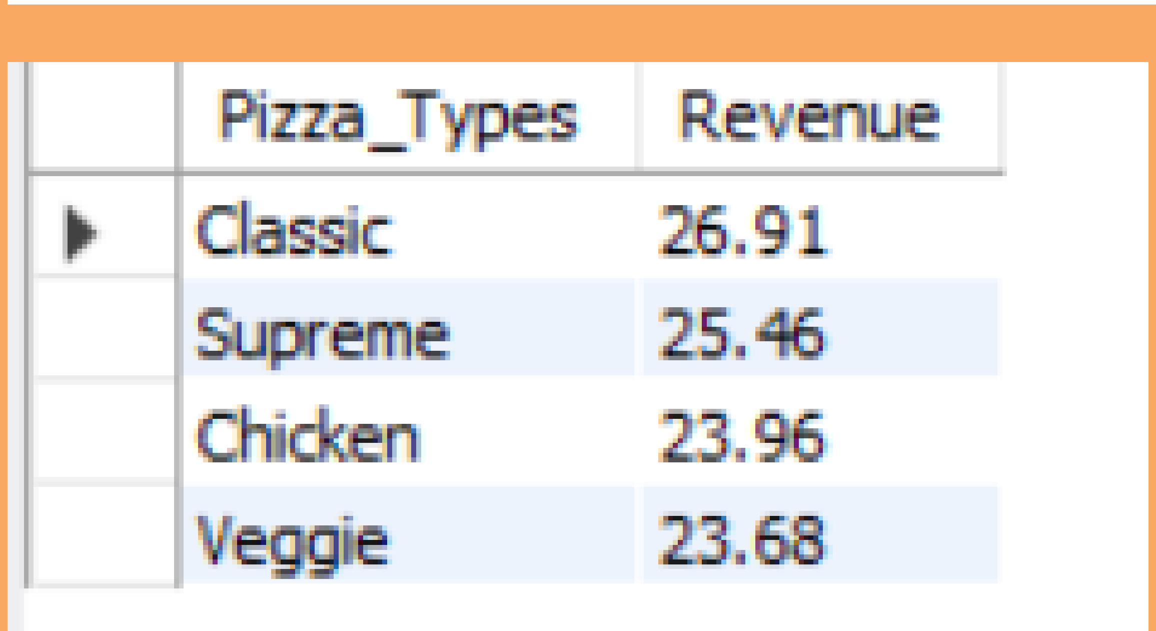
CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.

CODE



```
SELECT
    pizza_types.category AS Pizza_Types,
    ROUND(SUM(pizzas.price * orders_details.quantity) / (SELECT
    ROUND(SUM(od.quantity * pz.price), 2) AS Total_Sales
FROM
            orders_details AS od
            JOIN pizzas AS pz USING (pizza_id)) * 100,
            2) AS Revenue
FROM
    pizza_types
        JOIN pizzas USING (pizza_type_id)
        JOIN orders_details USING (pizza_id)
GROUP BY Pizza_Types
ORDER BY Revenue DESC;
```

OUTPUT

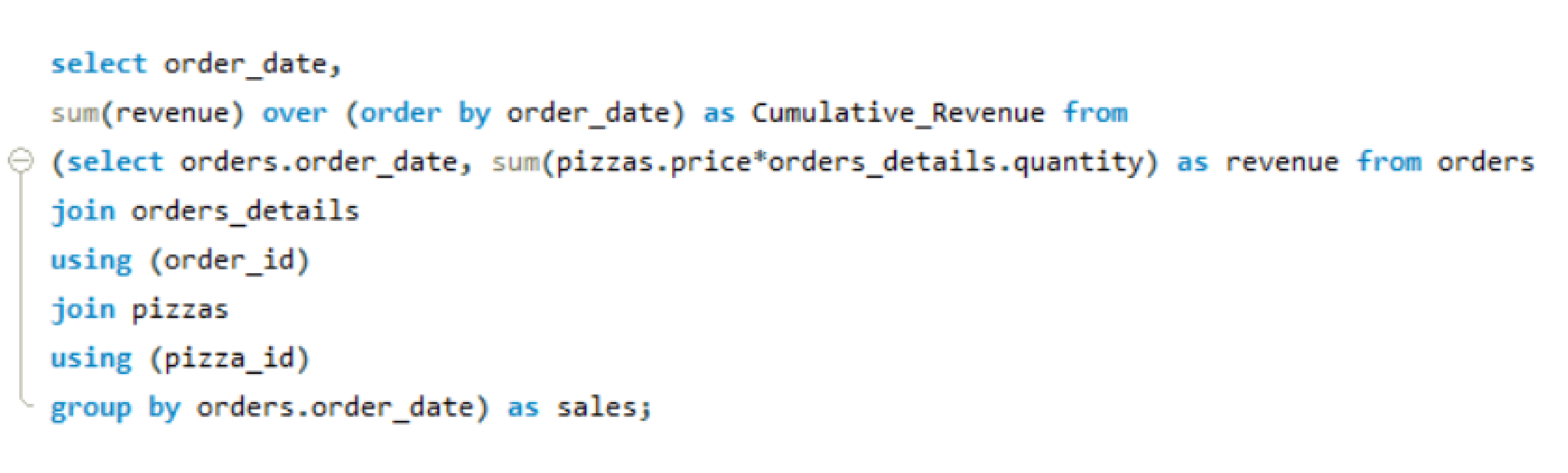


Pizza_Types	Revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

THIS SLIDE DIVES DEEPER INTO REVENUE CONTRIBUTION. WE'LL SEE THE PERCENTAGE SHARE OF EACH PIZZA TYPE'S REVENUE COMPARED TO THE TOTAL PIZZA REVENUE. THIS HELPS US IDENTIFY PIZZAS THAT ARE HIGH PERFORMERS FINANCIALLY.

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

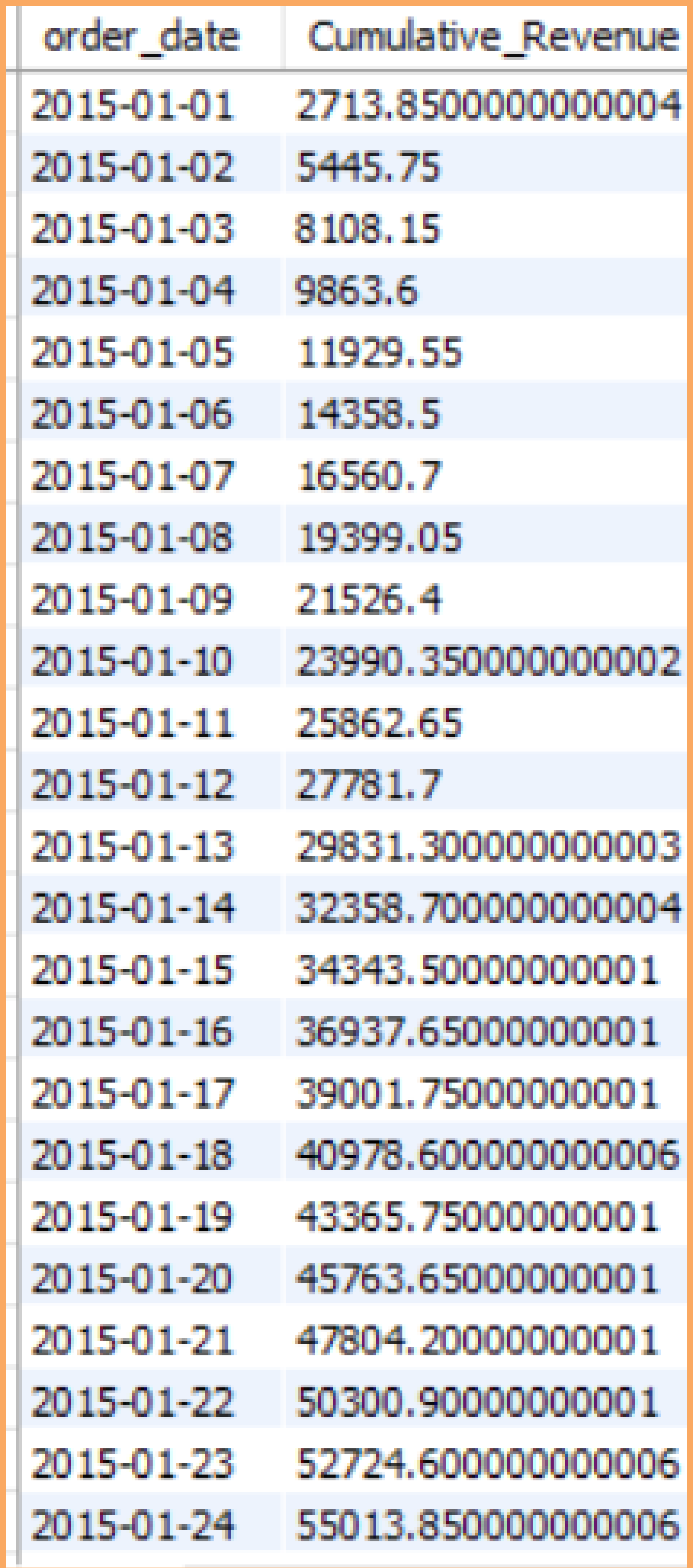
CODE



```
select order_date,
sum(revenue) over (order by order_date) as Cumulative_Revenue from
(select orders.order_date, sum(pizzas.price*orders_details.quantity) as revenue from orders
join orders_details
using (order_id)
join pizzas
using (pizza_id)
group by orders.order_date) as sales;
```

THIS SLIDE UNVEILS THE OVERALL REVENUE GROWTH PATTERN OVER TIME. WE CAN IDENTIFY PERIODS OF HIGH GROWTH OR STAGNATION, WHICH CAN INFORM BUSINESS DECISIONS LIKE PROMOTIONS OR MENU CHANGES.

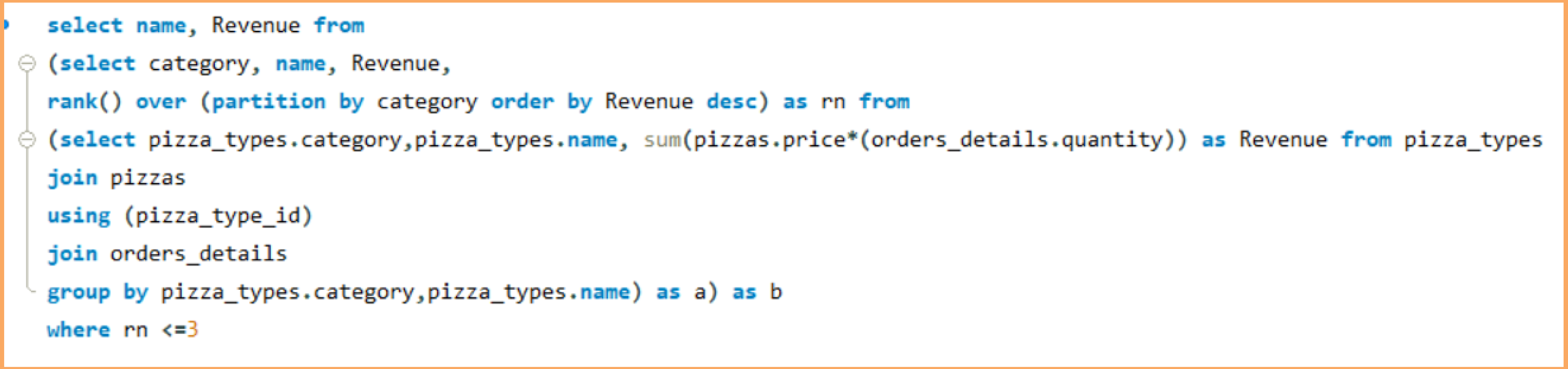
OUTPUT



order_date	Cumulative_Revenue
2015-01-01	2713.8500000000004
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	23990.350000000002
2015-01-11	25862.65
2015-01-12	27781.7
2015-01-13	29831.300000000003
2015-01-14	32358.700000000004
2015-01-15	34343.50000000001
2015-01-16	36937.65000000001
2015-01-17	39001.75000000001
2015-01-18	40978.600000000006
2015-01-19	43365.75000000001
2015-01-20	45763.65000000001
2015-01-21	47804.20000000001
2015-01-22	50300.90000000001
2015-01-23	52724.600000000006
2015-01-24	55013.850000000006

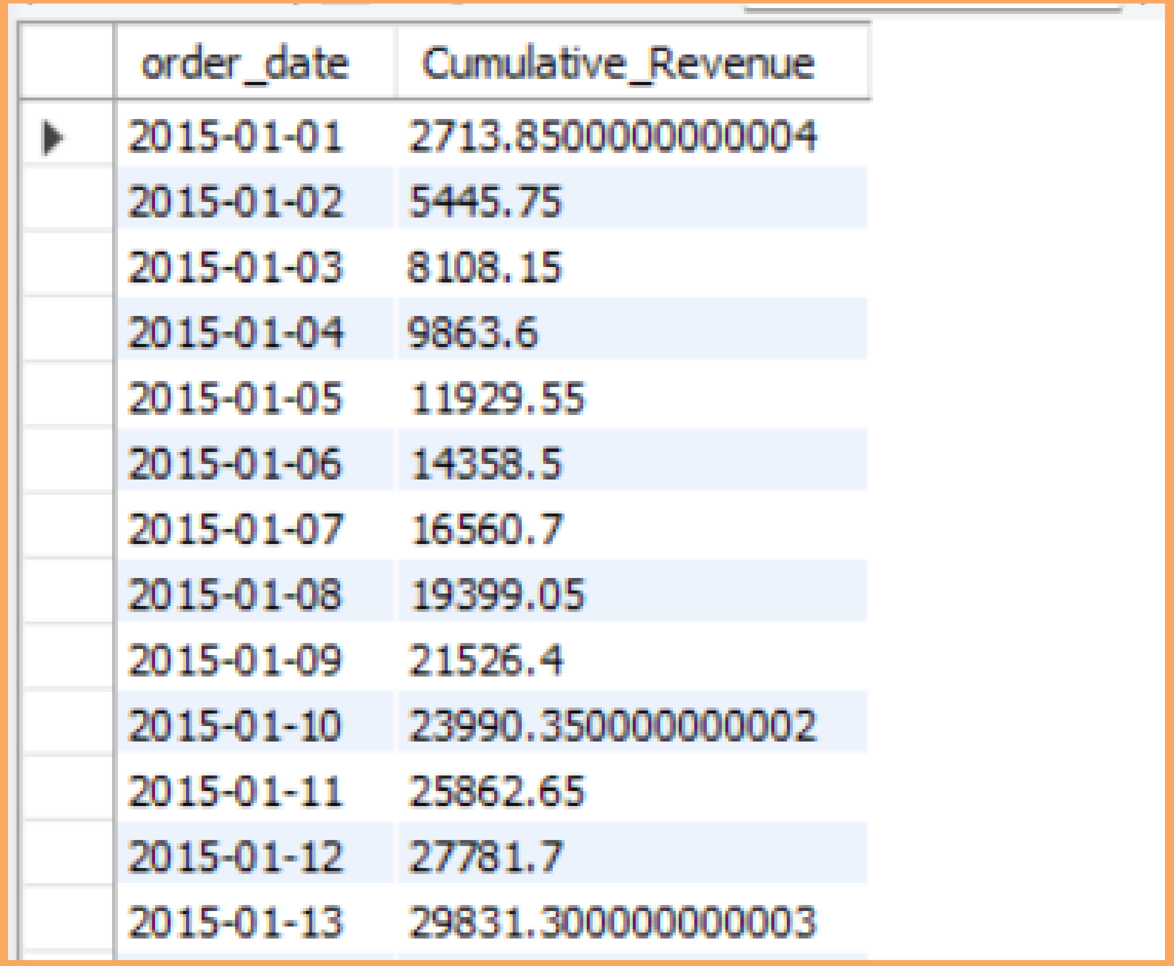
DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY.

CODE



```
select 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(pizzas.price*(orders_details.quantity)) as Revenue from pizza_types
join pizzas
using (pizza_type_id)
join orders_details
group by pizza_types.category,pizza_types.name) as a) as b
where rn <=3
```

OUTPUT



order_date	Cumulative_Revenue
2015-01-01	2713.8500000000004
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	23990.350000000002
2015-01-11	25862.65
2015-01-12	27781.7
2015-01-13	29831.300000000003

WE'LL USE JOINS TO ANALYZE CATEGORY-SPECIFIC REVENUE LEADERS. THIS REVEALS THE TOP-SELLING PIZZAS WITHIN EACH CATEGORY, PROVIDING VALUABLE INSIGHTS FOR TARGETED MARKETING AND INVENTORY MANAGEMENT.

Thank You