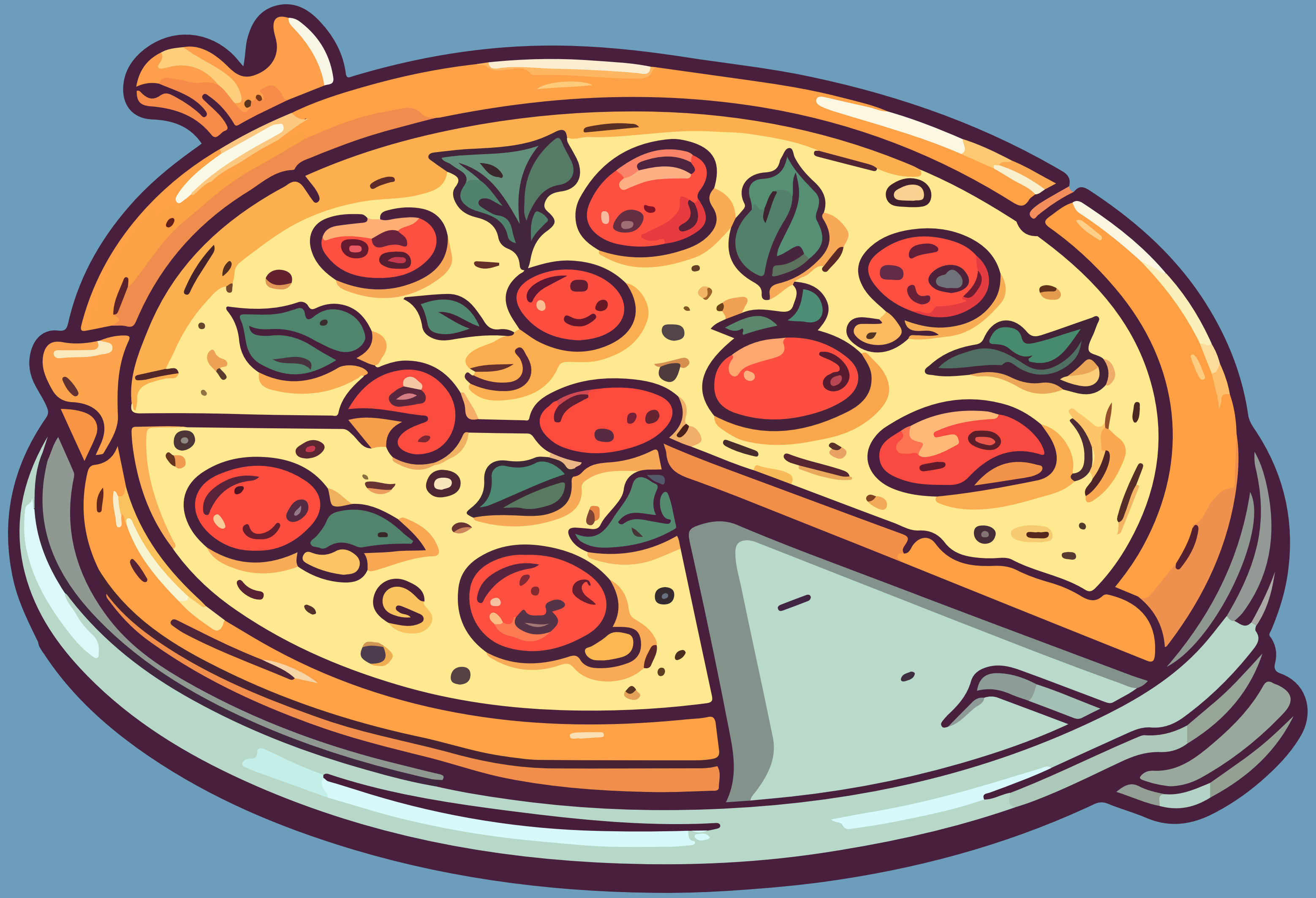
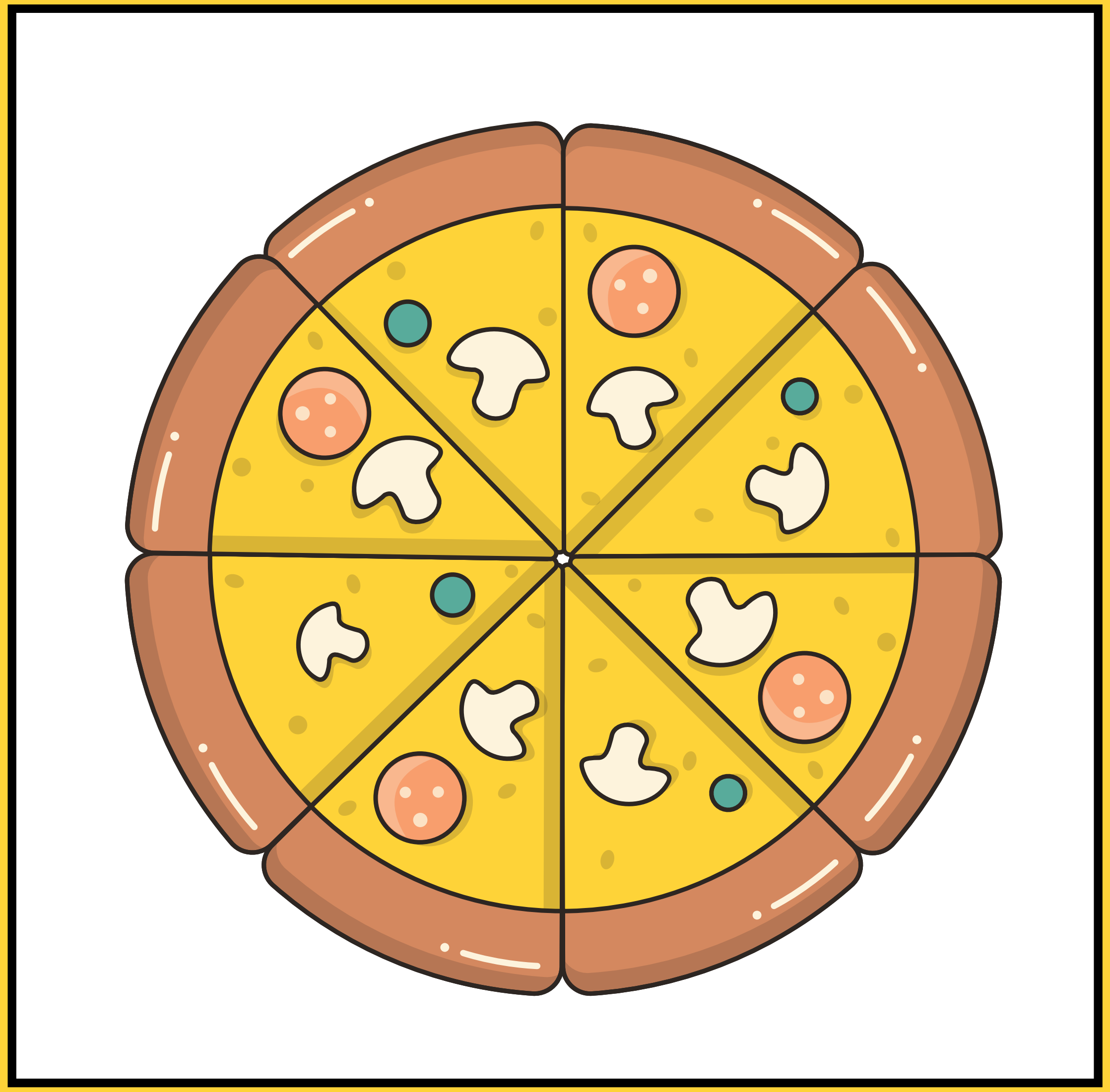
SQL-PROJECT ON PIZZA SALES



INTRODUCTION

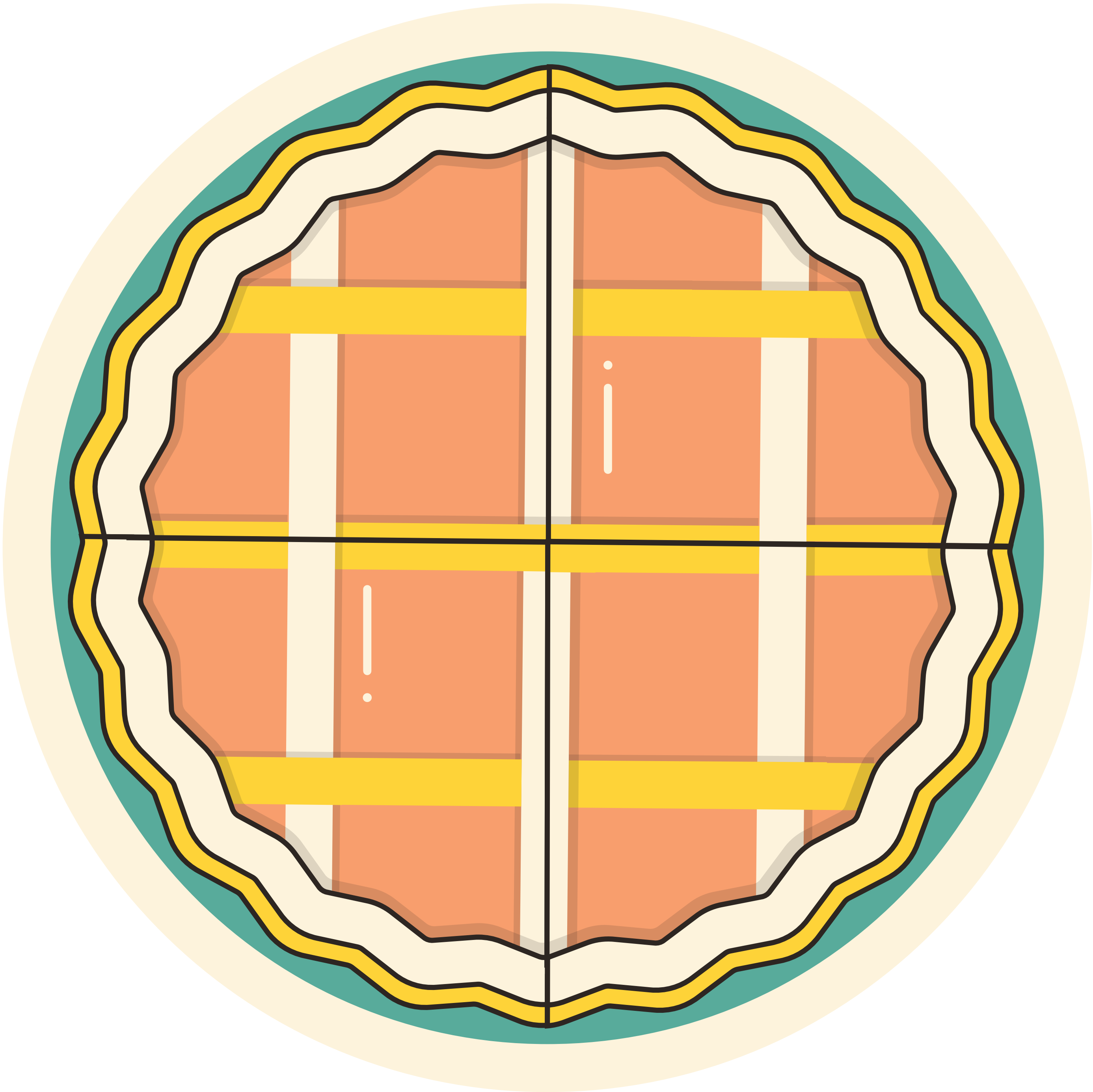
The Pizza Sales SQL Project is a comprehensive data analysis initiative aimed at exploring and understanding the trends, patterns, and factors influencing pizza sales in a hypothetical restaurant or chain. Through this project, we leverage SQL (Structured Query Language) to query, manipulate, and analyze sales data, offering valuable insights to optimize business performance and decision-making processes.

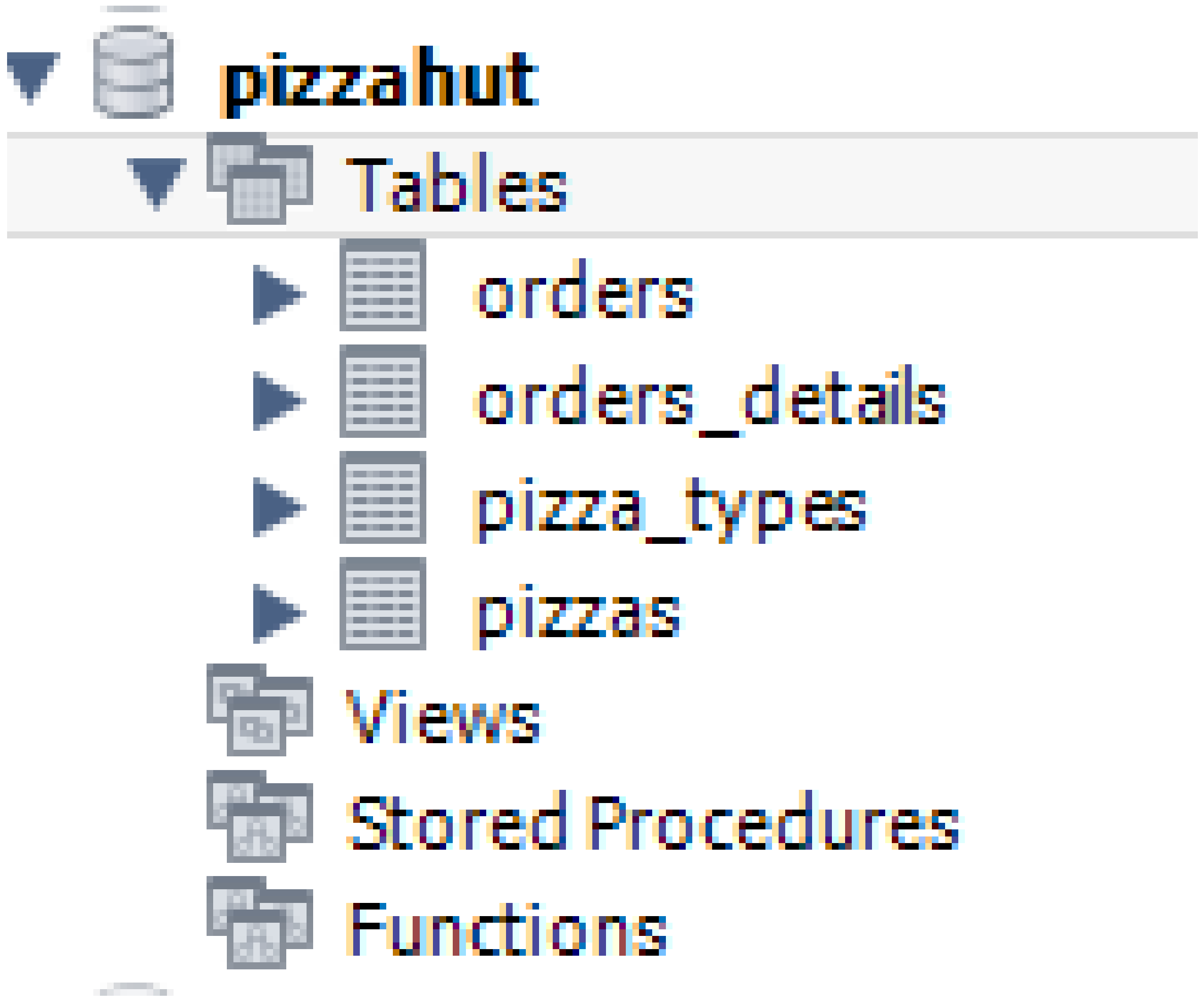


OBJECTIVES

- **Data Exploration: Understand the structure of the dataset, including tables such as orders, customers, pizza types, and toppings.**
- **Sales Analysis: Analyze the sales performance of different pizza types, sizes, and order times (weekdays, weekends, or specific hours).**
- **Customer Insights: Identify key customer demographics and preferences based on their purchase history.**
- **Inventory Management: Track the demand for ingredients and manage stock levels for optimal resource usage.**
- **Revenue Insights: Understand sales growth trends, peak seasons, and the revenue contribution of specific pizza types or branches.**

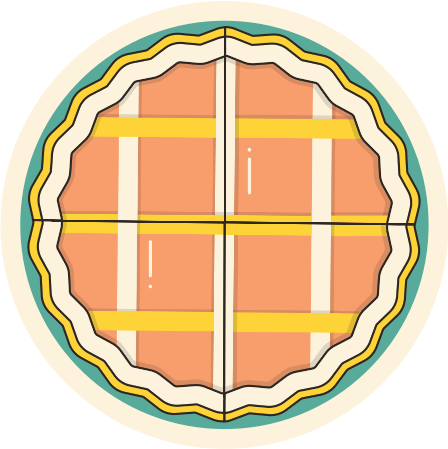
PIZZA SALES DATA

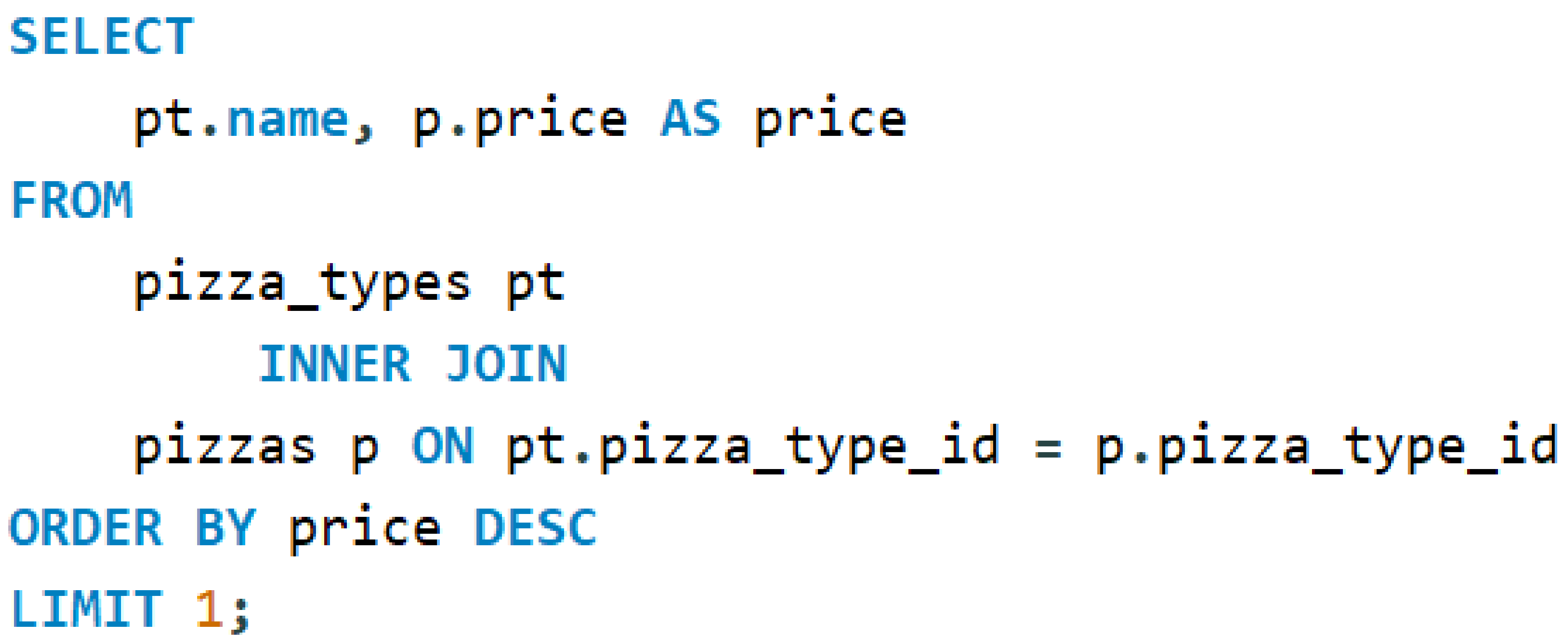




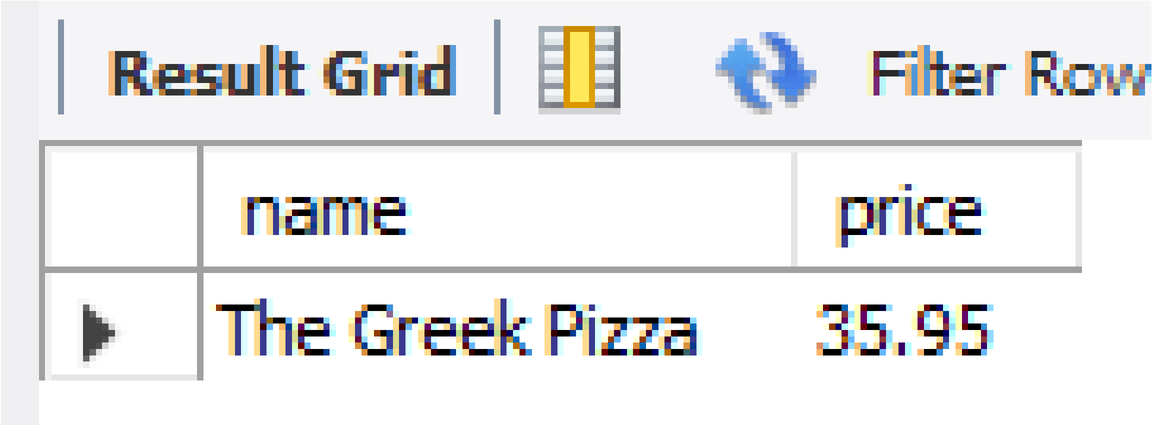
- pizzahut
  - Tables
    - orders
    - orders_details
    - pizza_types
    - pizzas
  - Views
  - Stored Procedures
  - Functions

IDENTIFY THE HIGHEST-PRICED PIZZA.

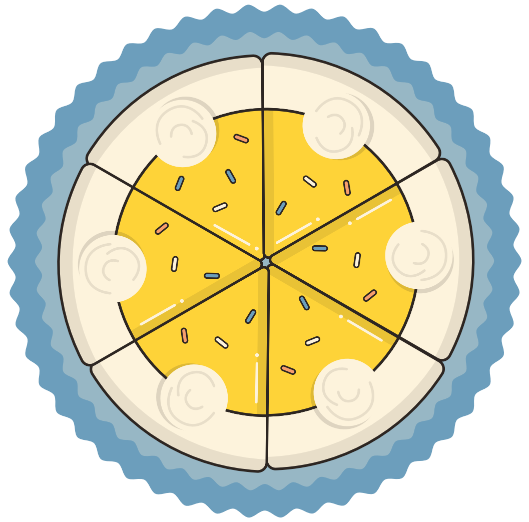




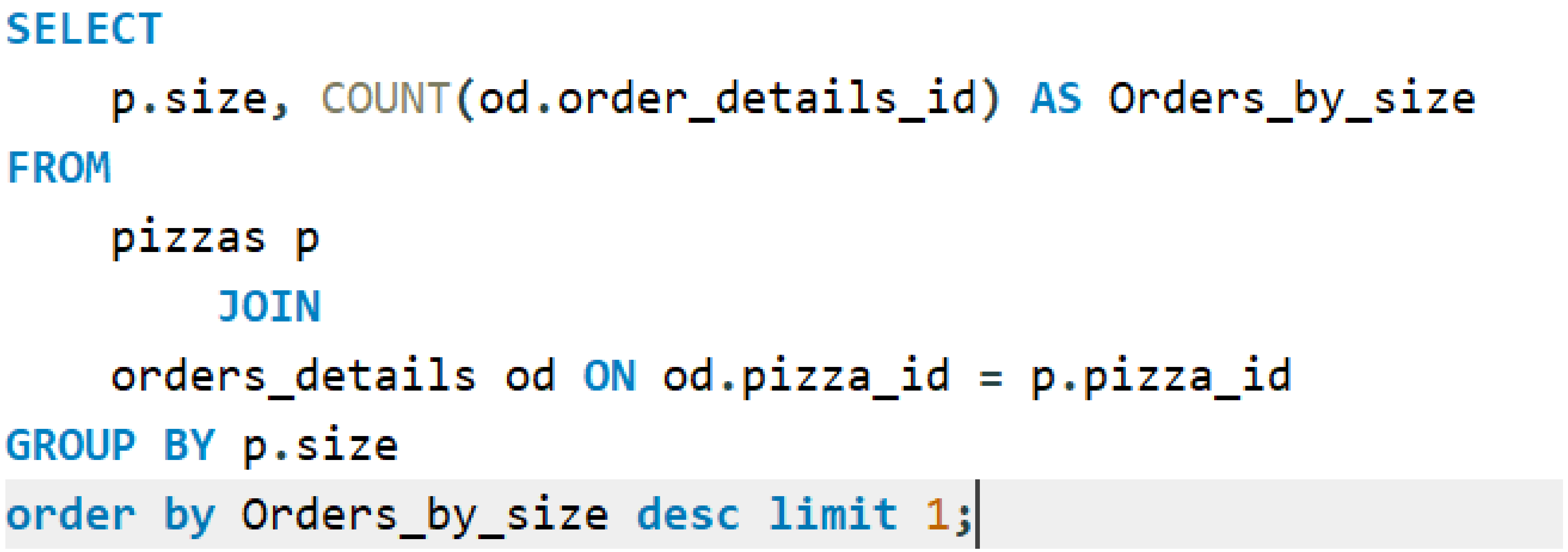
```
SELECT
    pt.name, p.price AS price
FROM
    pizza_types pt
        INNER JOIN
    pizzas p ON pt.pizza_type_id = p.pizza_type_id
ORDER BY price DESC
LIMIT 1;
```



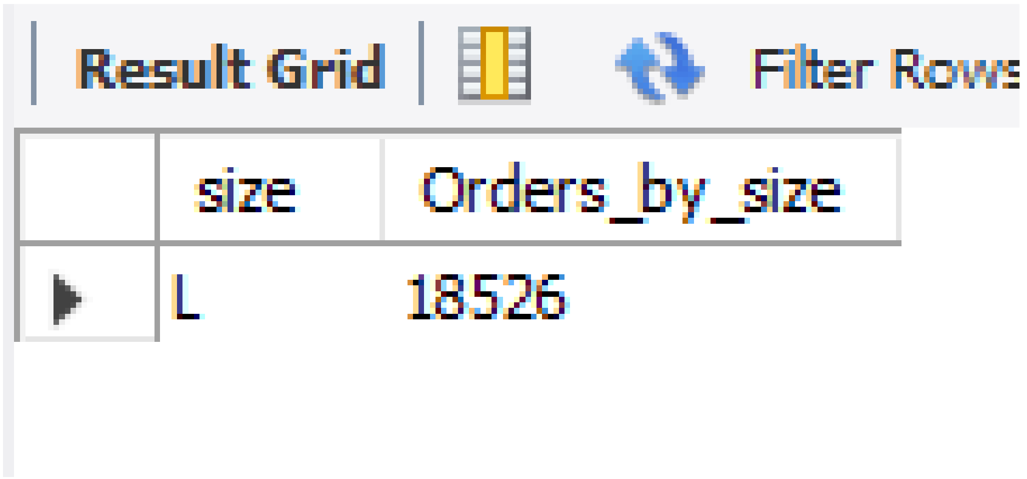
	name	price
▶	The Greek Pizza	35.95



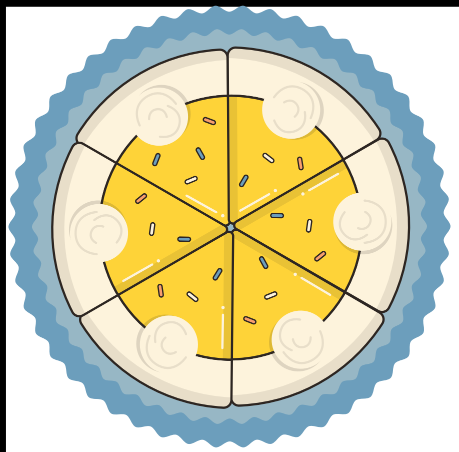
IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.



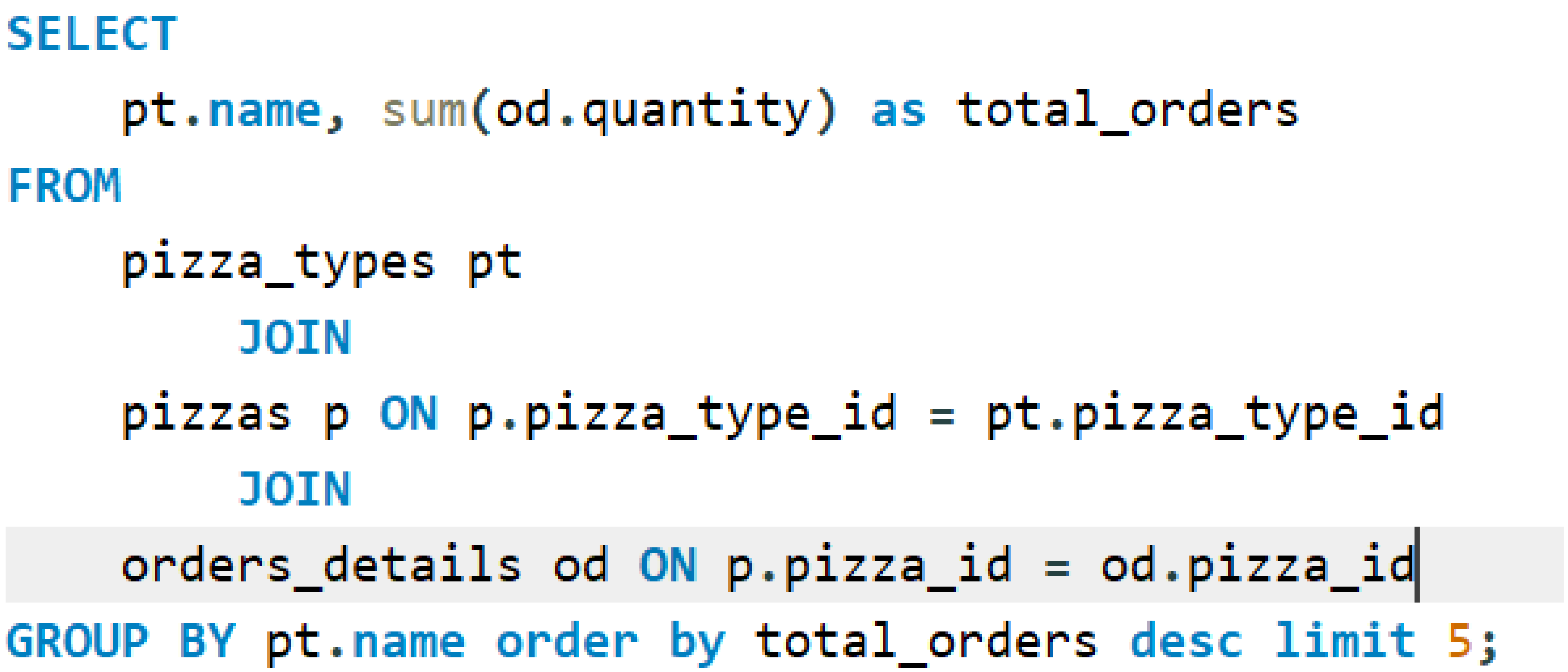
```
SELECT
    p.size, COUNT(od.order_details_id) AS Orders_by_size
FROM
    pizzas p
        JOIN
    orders_details od ON od.pizza_id = p.pizza_id
GROUP BY p.size
order by Orders_by_size desc limit 1;
```



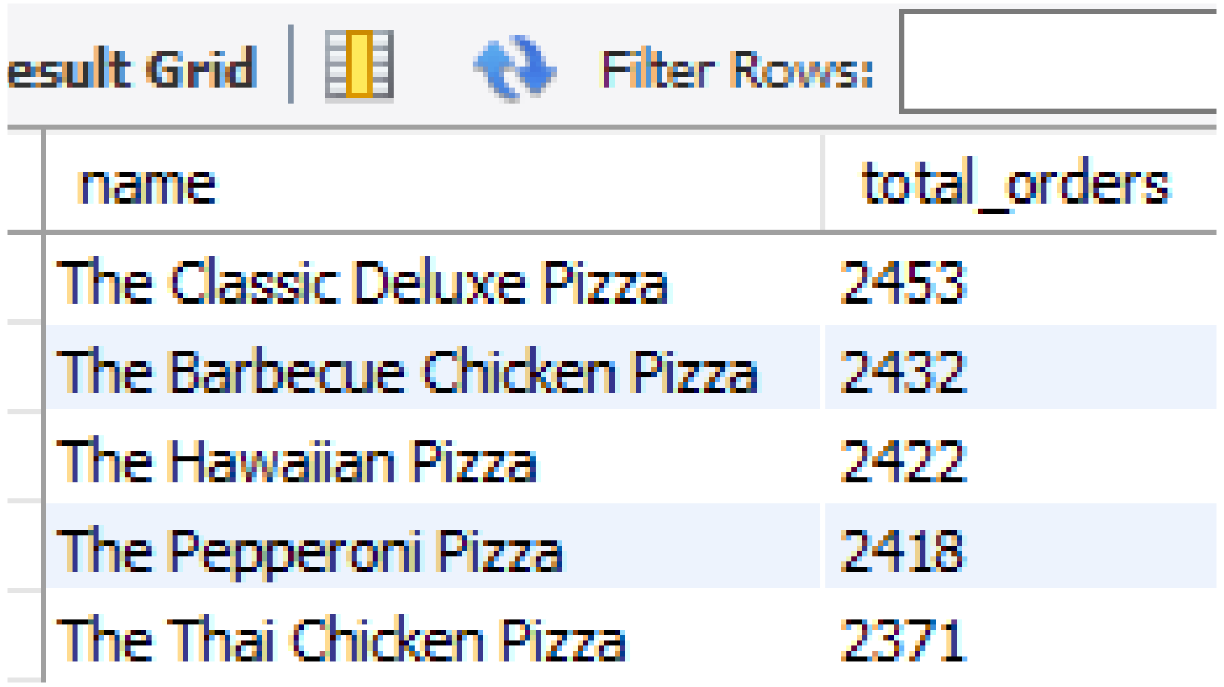
size	Orders_by_size
L	18526



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

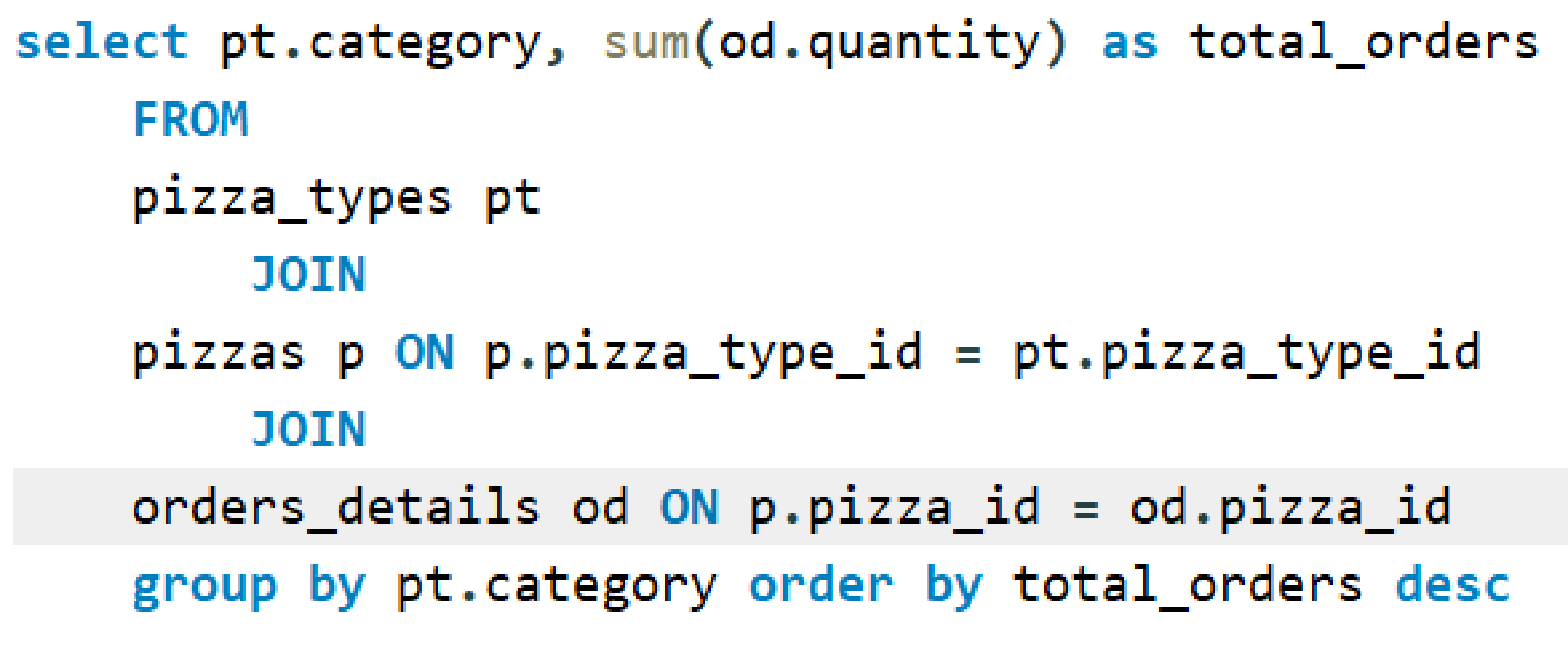


```
SELECT
    pt.name, sum(od.quantity) as total_orders
FROM
    pizza_types pt
        JOIN
    pizzas p ON p.pizza_type_id = pt.pizza_type_id
        JOIN
    orders_details od ON p.pizza_id = od.pizza_id
GROUP BY pt.name order by total_orders desc limit 5;
```

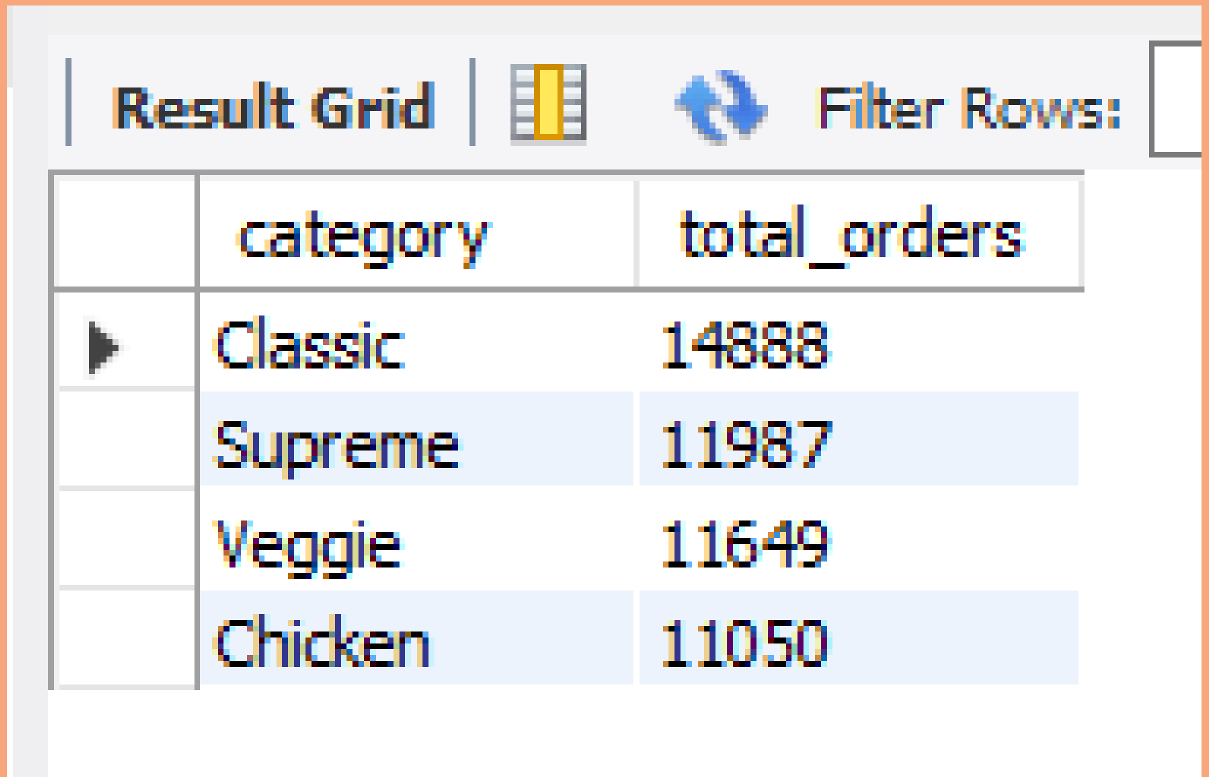


name	total_orders
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

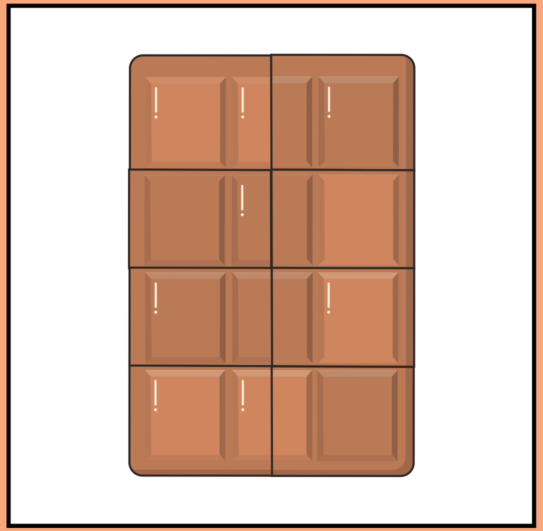
JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

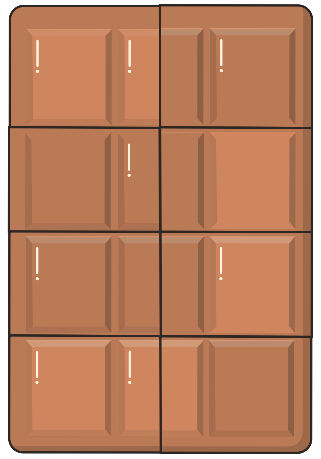


```
select pt.category, sum(od.quantity) as total_orders
    FROM
    pizza_types pt
        JOIN
    pizzas p ON p.pizza_type_id = pt.pizza_type_id
        JOIN
    orders_details od ON p.pizza_id = od.pizza_id
    group by pt.category order by total_orders desc
```

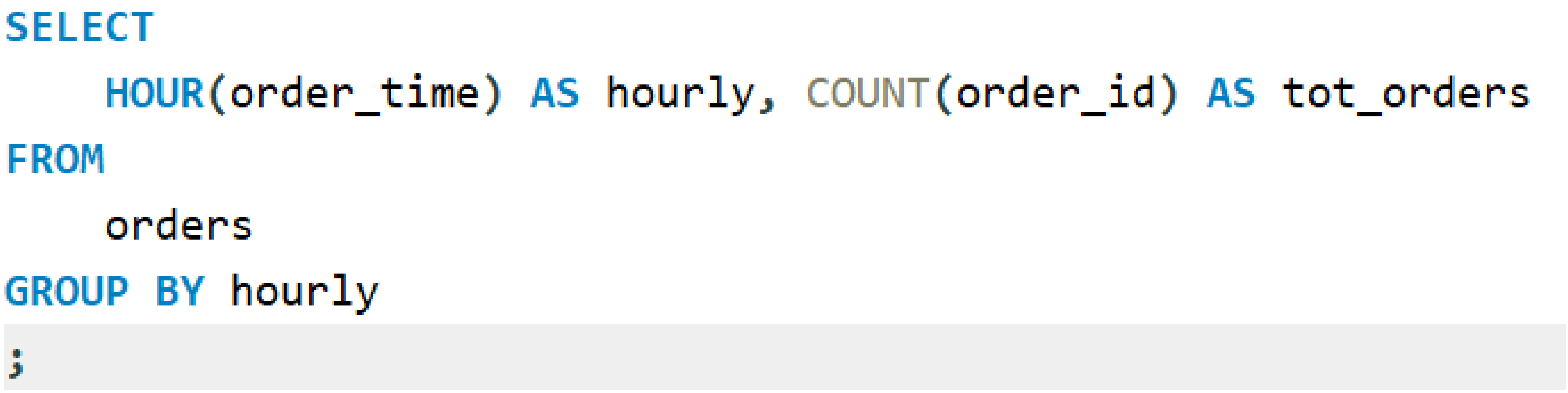


category	total_orders
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

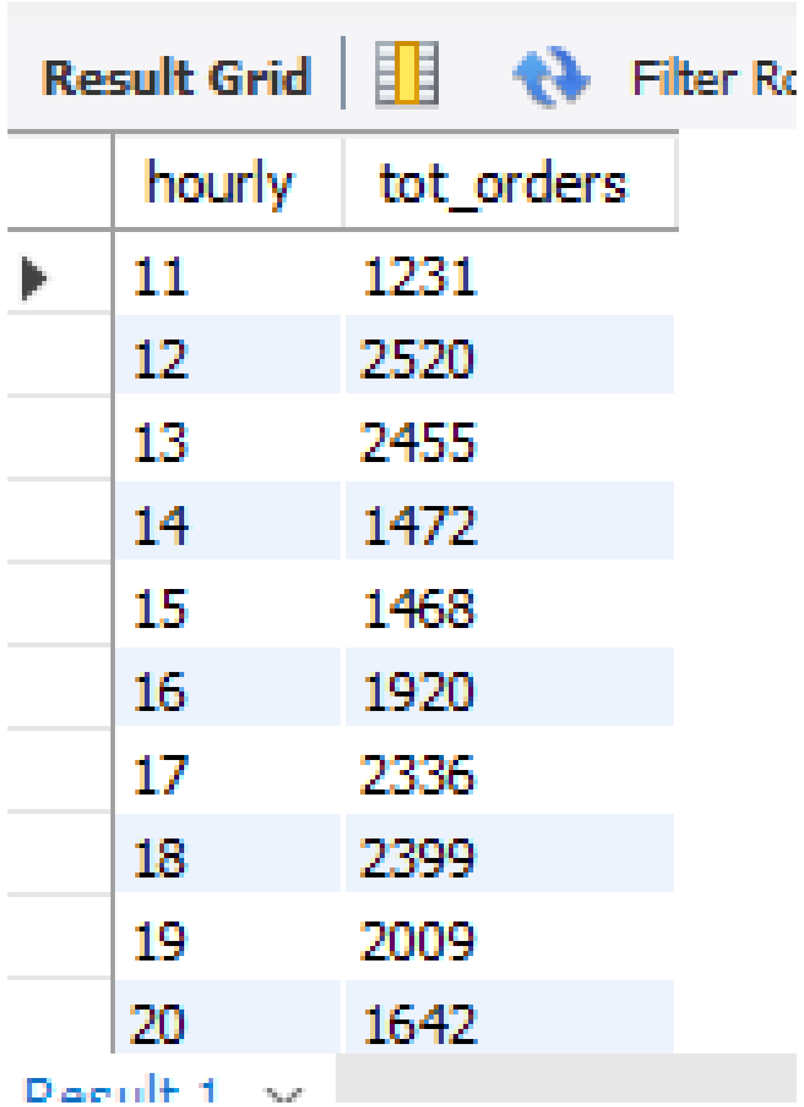




DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.



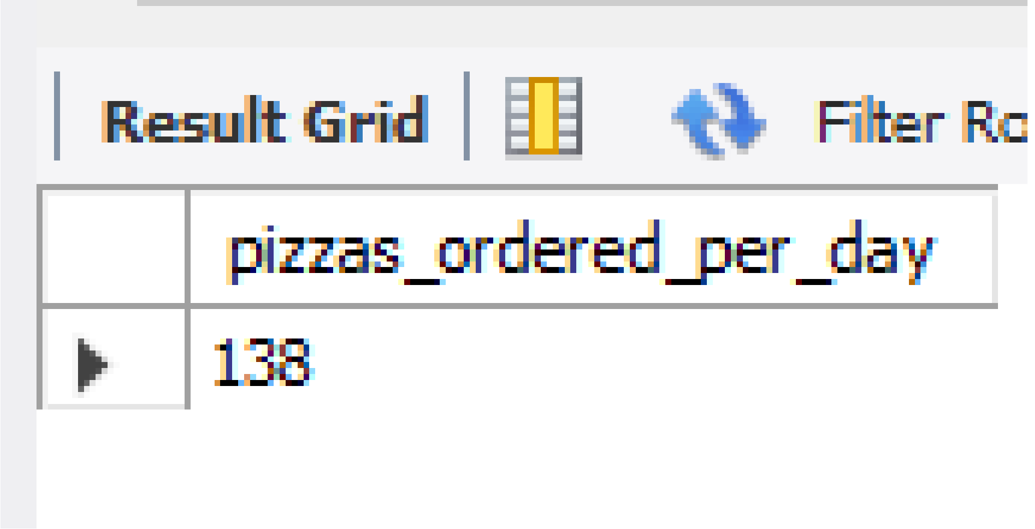
```
SELECT
    HOUR(order_time) AS hourly, COUNT(order_id) AS tot_orders
FROM
    orders
GROUP BY hourly
;
```



hourly	tot_orders
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642

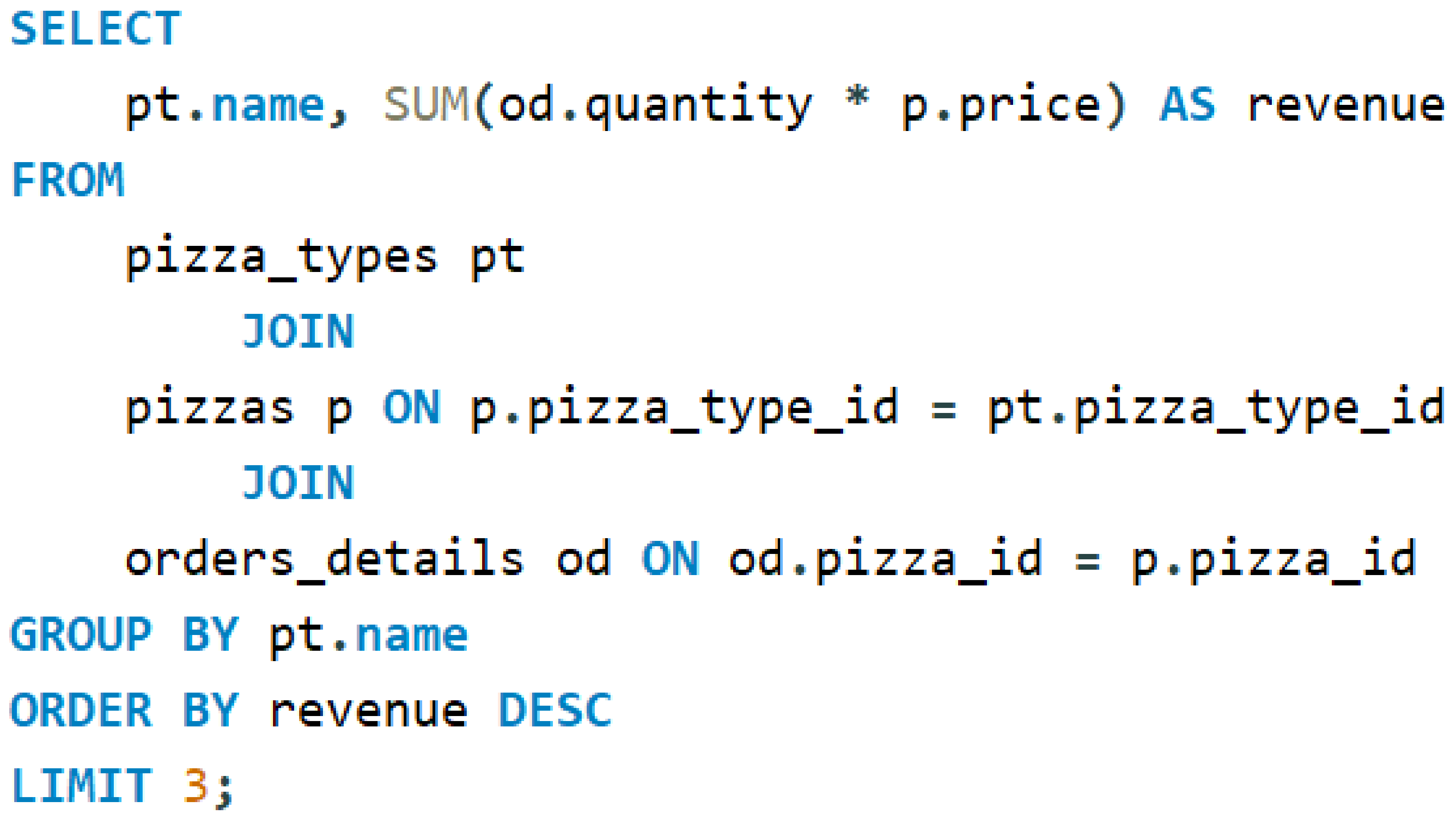
GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

```
SELECT
    ROUND(AVG(quantity), 0) AS pizzas_ordered_per_day
FROM
    (SELECT
        o.order_date AS date, SUM(od.quantity) AS quantity
    FROM
        orders o
    JOIN orders_details od ON o.order_id = od.order_id
    GROUP BY date) AS orders_perdate
```

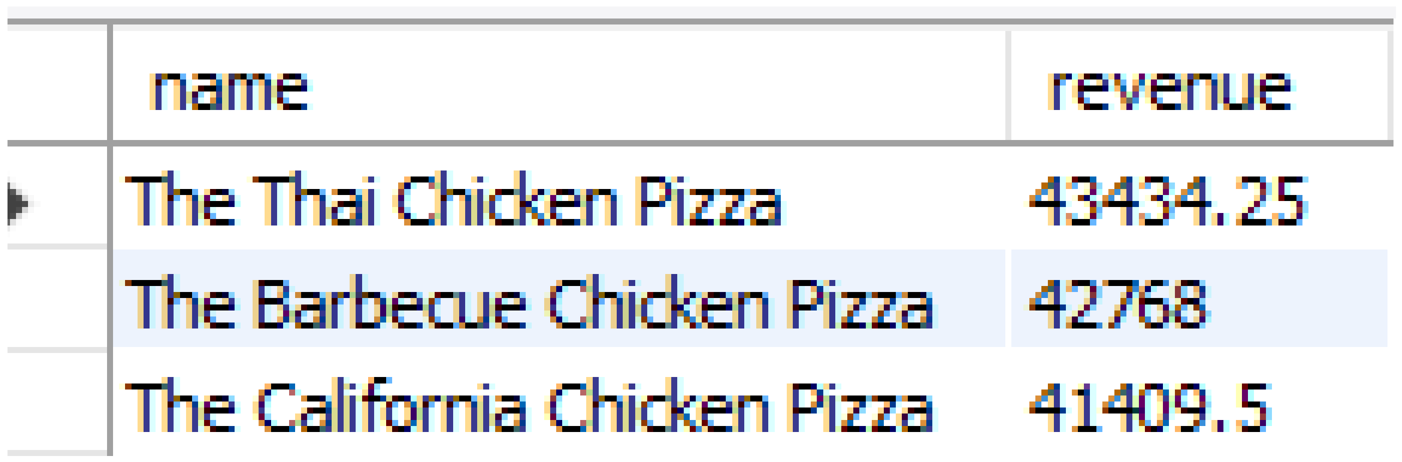


	pizzas_ordered_per_day
▶	138

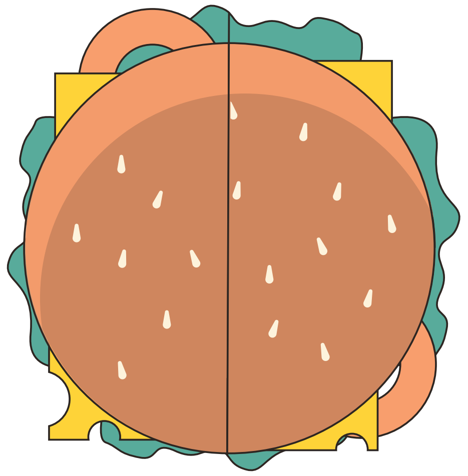
DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.



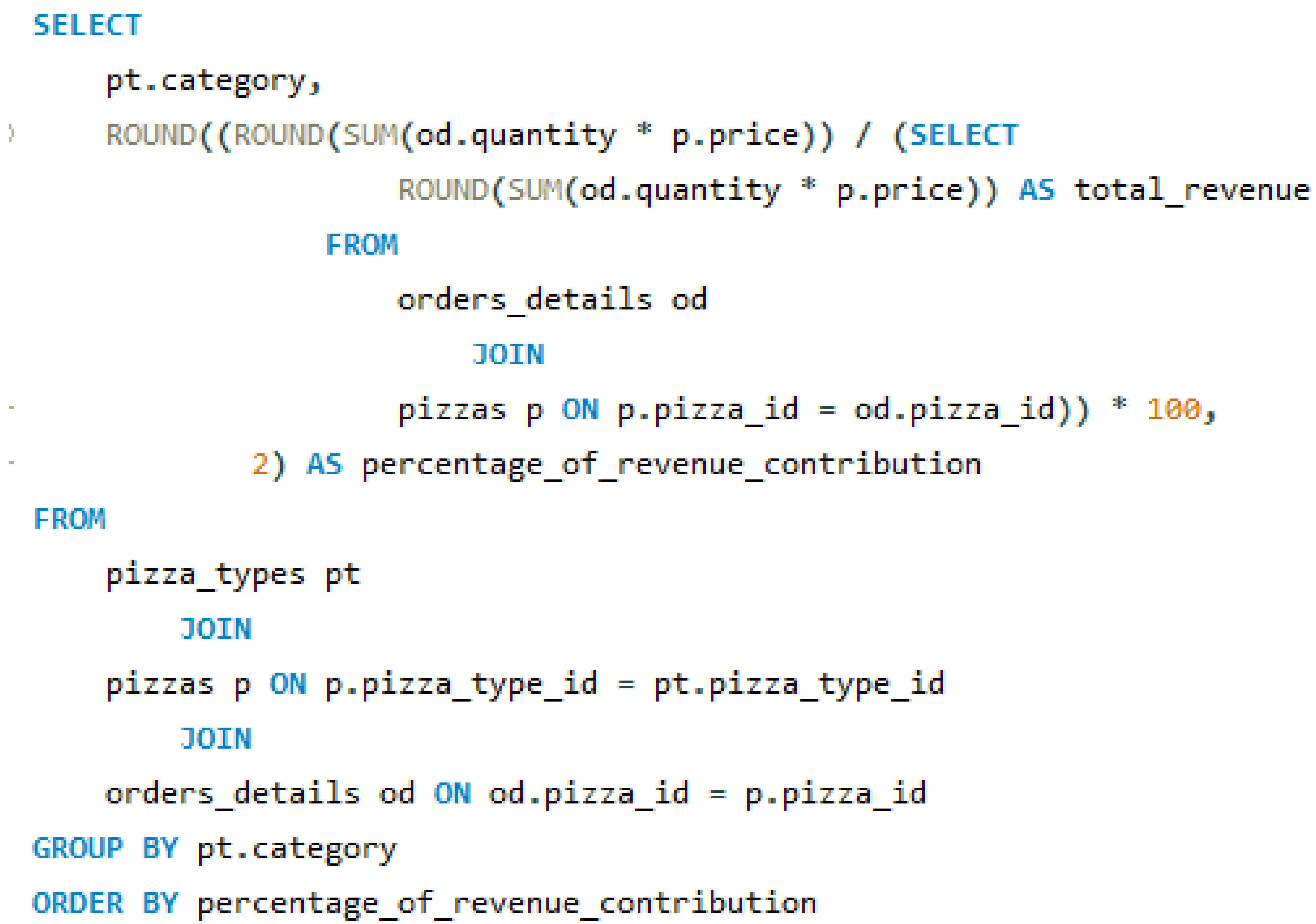
```
SELECT
    pt.name, SUM(od.quantity * p.price) AS revenue
FROM
    pizza_types pt
        JOIN
    pizzas p ON p.pizza_type_id = pt.pizza_type_id
        JOIN
    orders_details od ON od.pizza_id = p.pizza_id
GROUP BY pt.name
ORDER BY revenue DESC
LIMIT 3;
```



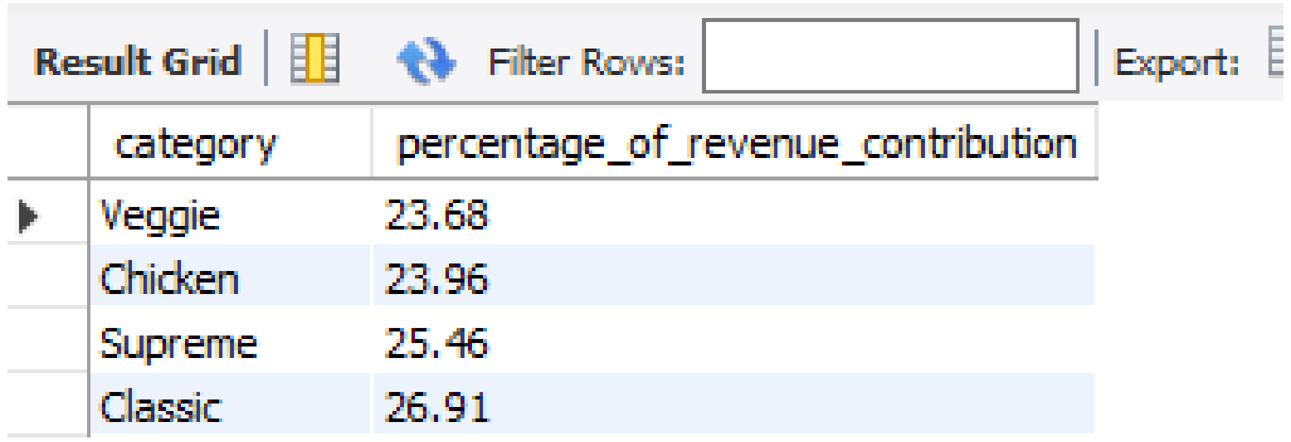
name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5



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

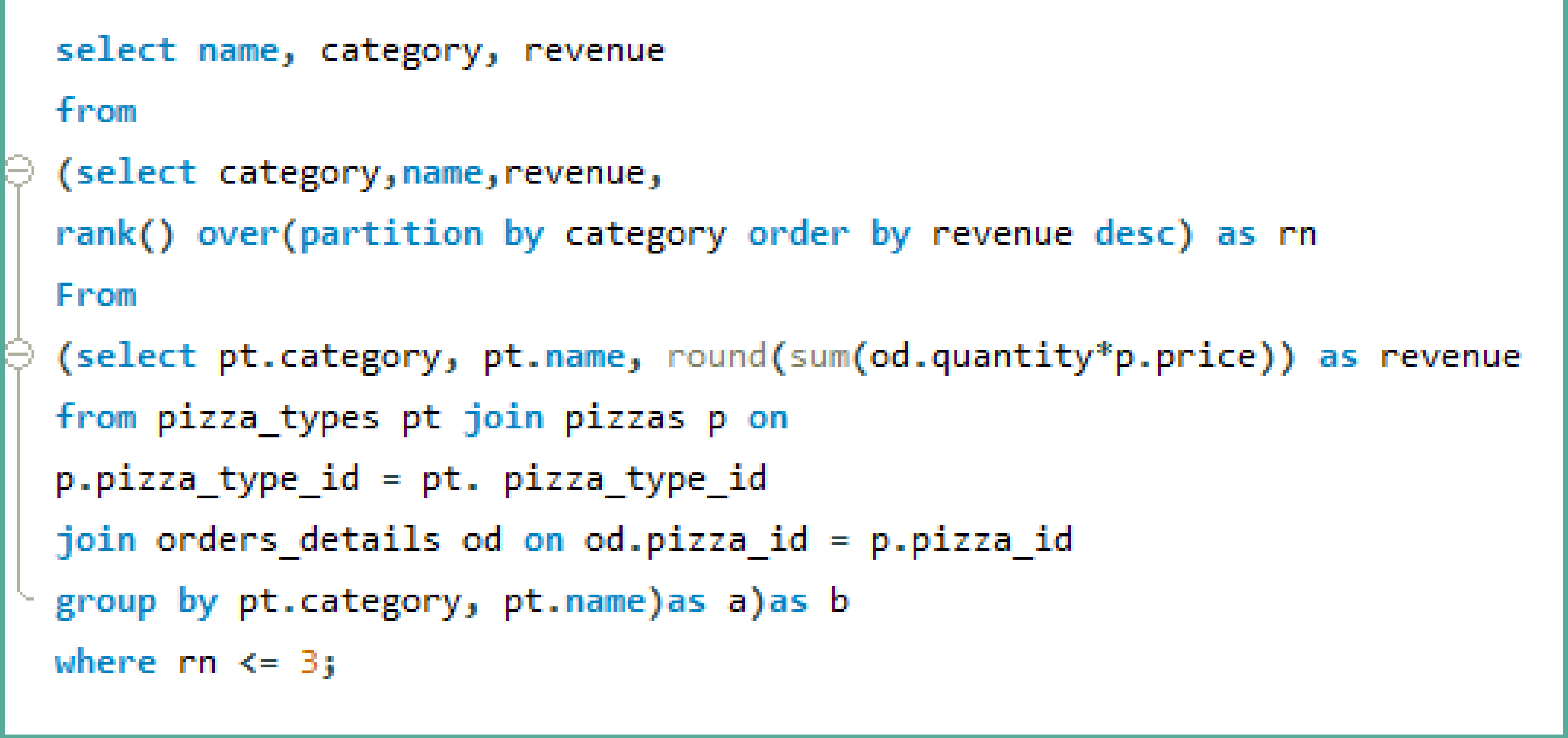


```
SELECT
    pt.category,
    ROUND((ROUND(SUM(od.quantity * p.price)) / (SELECT
                ROUND(SUM(od.quantity * p.price)) AS total_revenue
            FROM
                orders_details od
                    JOIN
                pizzas p ON p.pizza_id = od.pizza_id)) * 100,
        2) AS percentage_of_revenue_contribution
FROM
    pizza_types pt
        JOIN
    pizzas p ON p.pizza_type_id = pt.pizza_type_id
        JOIN
    orders_details od ON od.pizza_id = p.pizza_id
GROUP BY pt.category
ORDER BY percentage_of_revenue_contribution
```

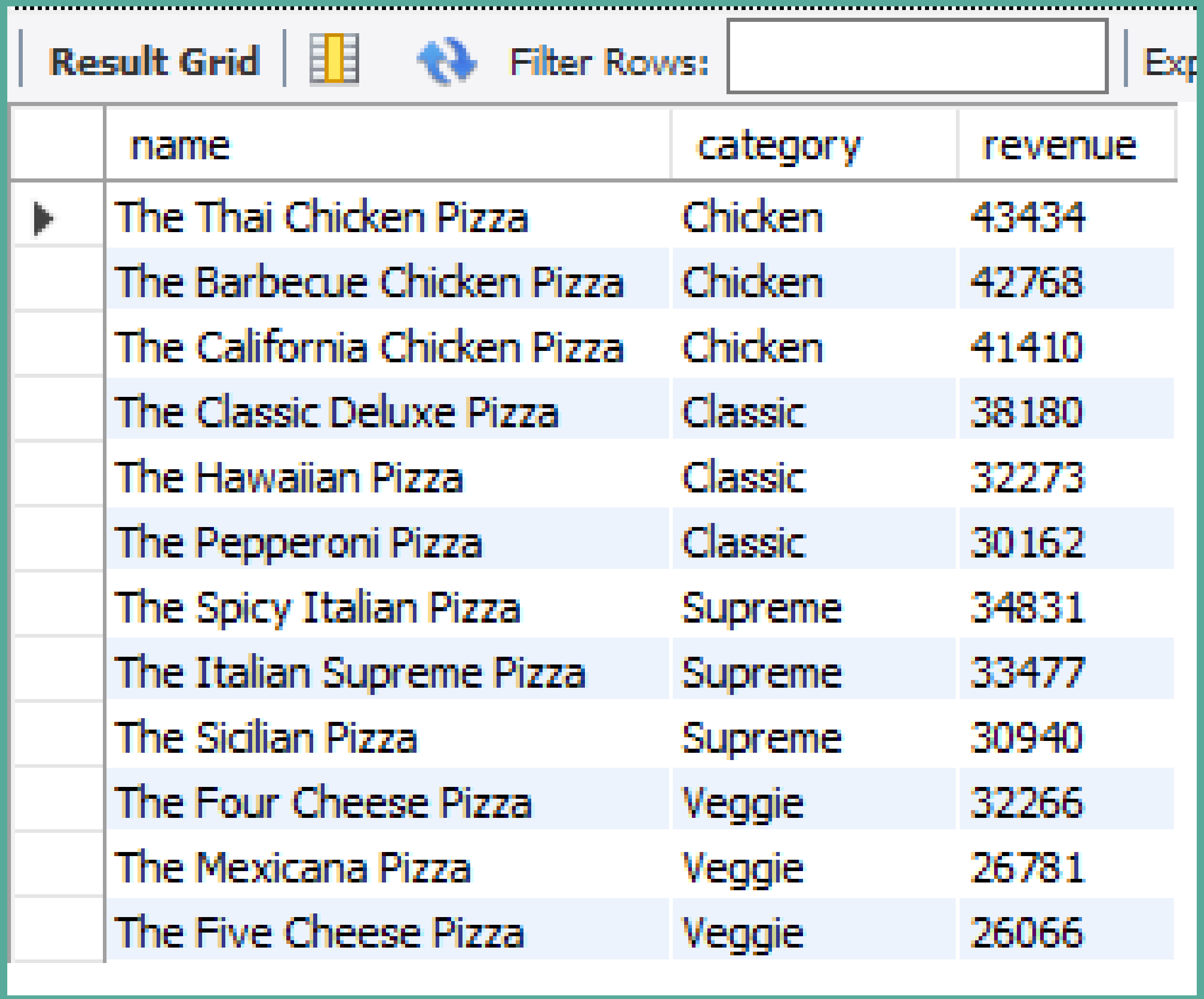


category	percentage_of_revenue_contribution
Veggie	23.68
Chicken	23.96
Supreme	25.46
Classic	26.91

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

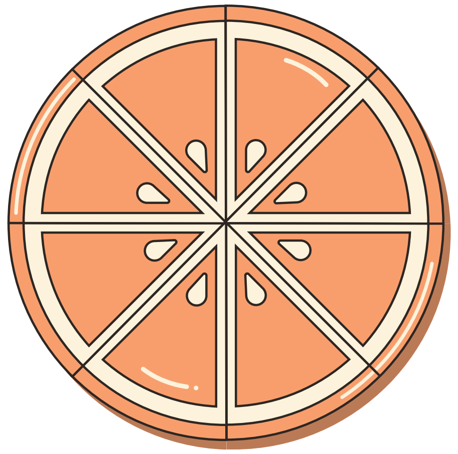


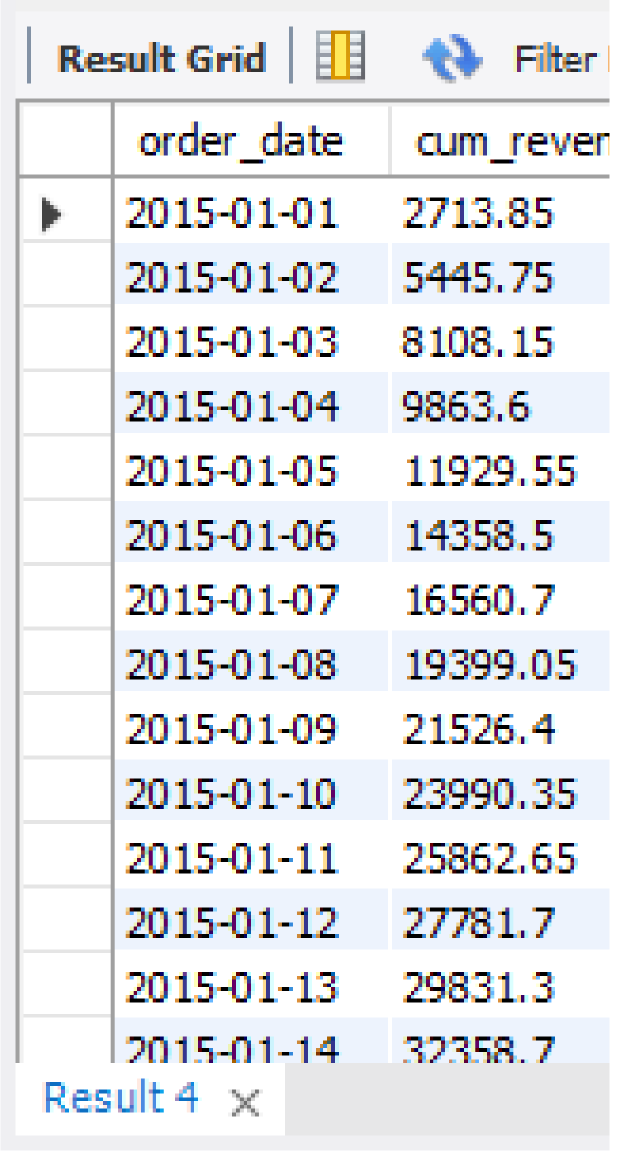
```
select name, category, revenue
from
(select category,name,revenue,
rank() over(partition by category order by revenue desc) as rn
From
(select pt.category, pt.name, round(sum(od.quantity*p.price)) as revenue
from pizza_types pt join pizzas p on
p.pizza_type_id = pt. pizza_type_id
join orders_details od on od.pizza_id = p.pizza_id
group by pt.category, pt.name)as a)as b
where rn <= 3;
```



name	category	revenue
The Thai Chicken Pizza	Chicken	43434
The Barbecue Chicken Pizza	Chicken	42768
The California Chicken Pizza	Chicken	41410
The Classic Deluxe Pizza	Classic	38180
The Hawaiian Pizza	Classic	32273
The Pepperoni Pizza	Classic	30162
The Spicy Italian Pizza	Supreme	34831
The Italian Supreme Pizza	Supreme	33477
The Sicilian Pizza	Supreme	30940
The Four Cheese Pizza	Veggie	32266
The Mexicana Pizza	Veggie	26781
The Five Cheese Pizza	Veggie	26066

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.





	order_date	cum_rever
▶	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

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