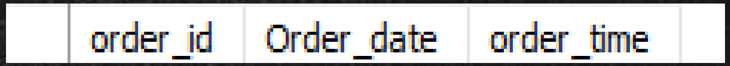
Slice of Data: SQL Solution for Pizza Ordering System

Tables Schema Details

01

Orders Table

It contain details related to Order, related to time. Its schema:

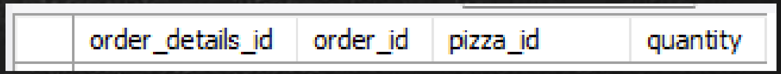


order_id	Order_date	order_time

02

Orders_details

It contain other order details. Its schema:

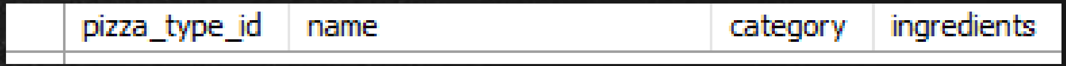


order_details_id	order_id	pizza_id	quantity

03

Pizza_types

It Information about various types and categories of pizza. Its schema:

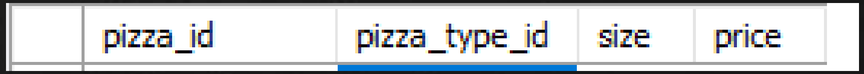


pizza_type_id	name	category	ingredients

04

Pizzas

It contain basic attributes related to pizza like size, price etc. Its schema:



pizza_id	pizza_type_id	size	price

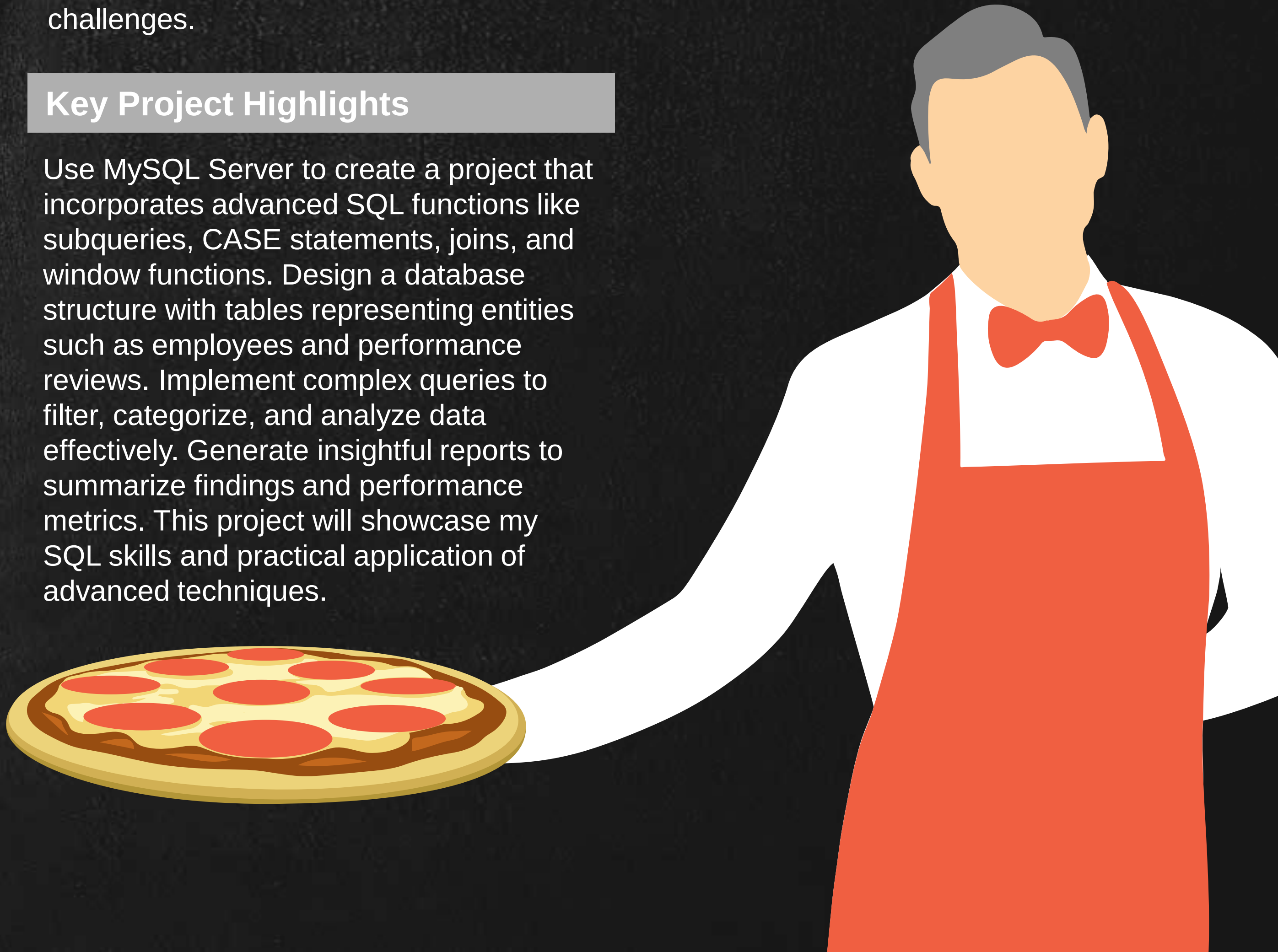
Project Overview

Project Rationale

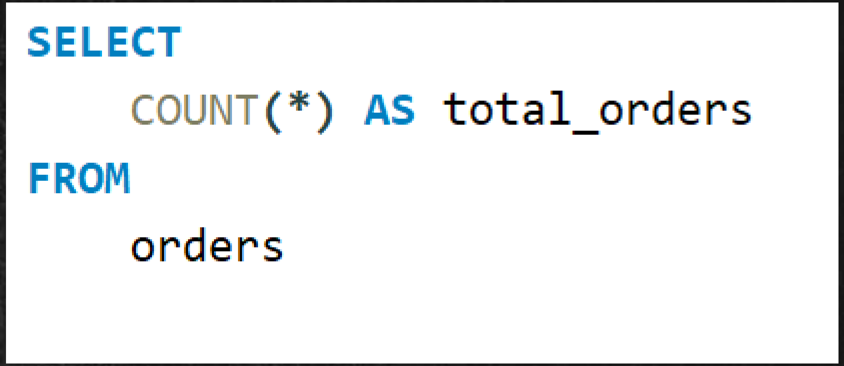
Key questions have been gathered, and SQL Server is being utilized to efficiently explore solutions and address business challenges.

Key Project Highlights

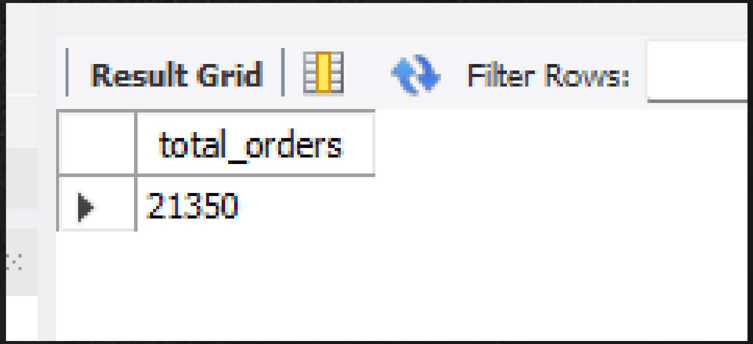
Use MySQL Server to create a project that incorporates advanced SQL functions like subqueries, CASE statements, joins, and window functions. Design a database structure with tables representing entities such as employees and performance reviews. Implement complex queries to filter, categorize, and analyze data effectively. Generate insightful reports to summarize findings and performance metrics. This project will showcase my SQL skills and practical application of advanced techniques.



1. Retrieve the total number of orders placed?

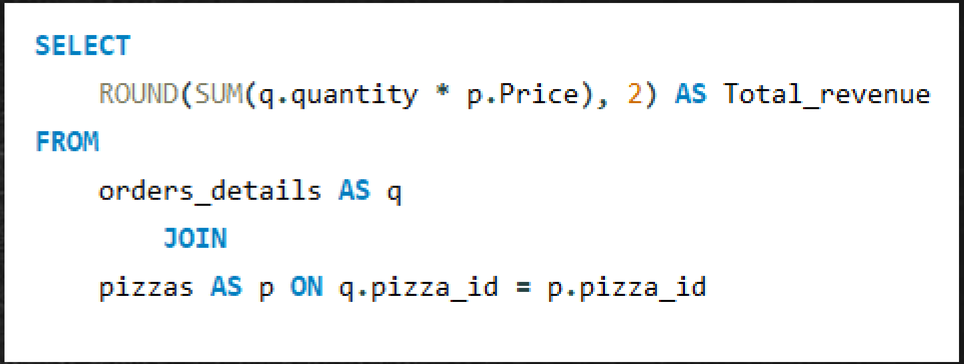


```
SELECT
    COUNT(*) AS total_orders
FROM
    orders
```

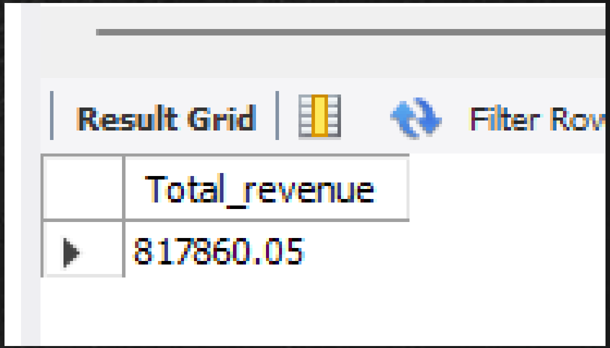


```
total_orders
21350
```

2. Calculate the total revenue generated from pizza sales?

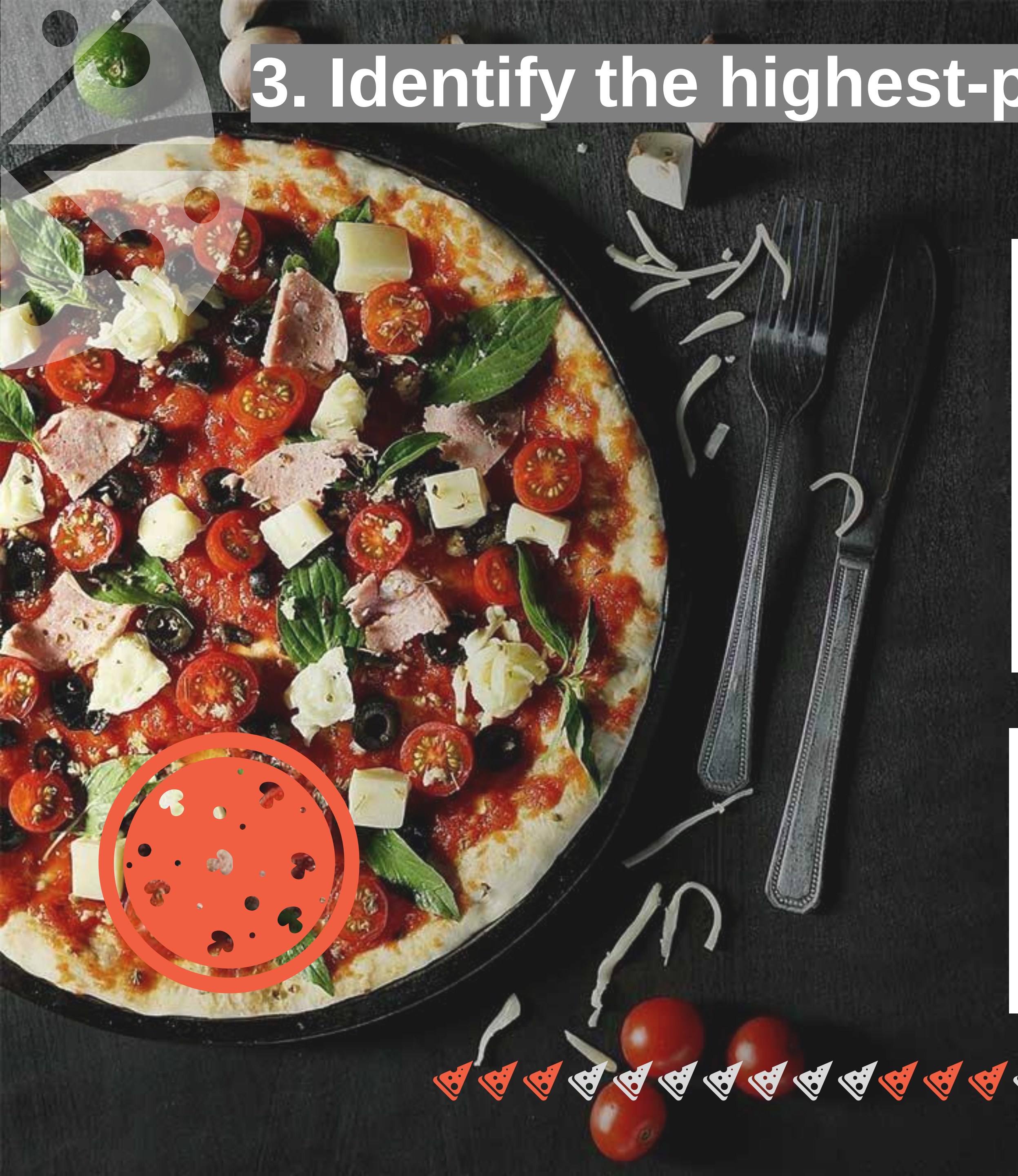


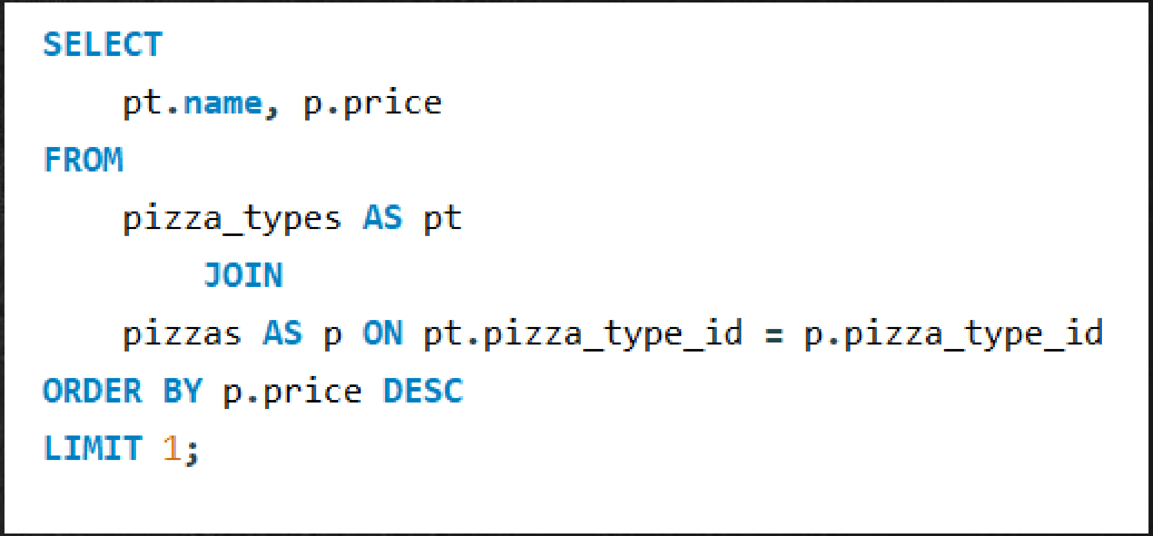
```
SELECT
    ROUND(SUM(q.quantity * p.Price), 2) AS Total_revenue
FROM
    orders_details AS q
        JOIN
    pizzas AS p ON q.pizza_id = p.pizza_id
```



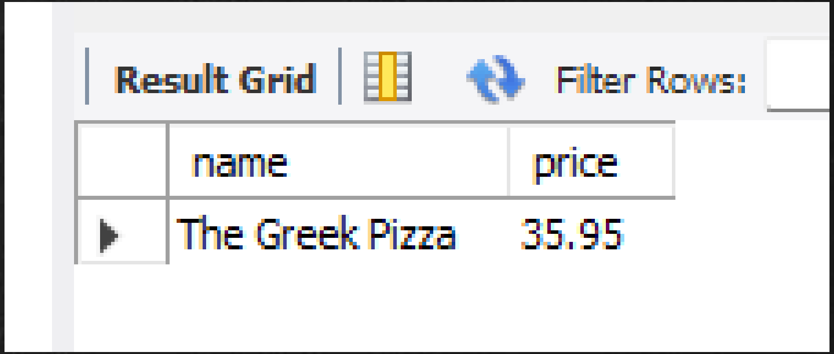
```
Total_revenue
817860.05
```

3. Identify the highest-priced pizza.pizza_types?





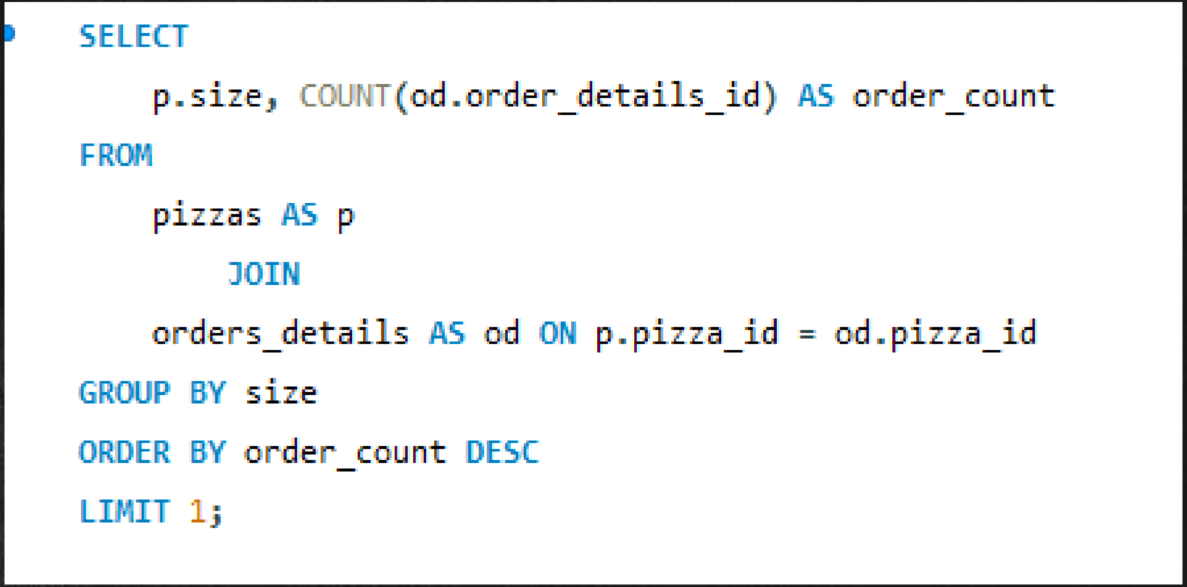
```sql
SELECT
    pt.name, p.price
FROM
    pizza_types AS pt
        JOIN
    pizzas AS p ON pt.pizza_type_id = p.pizza_type_id
ORDER BY p.price DESC
LIMIT 1;
```



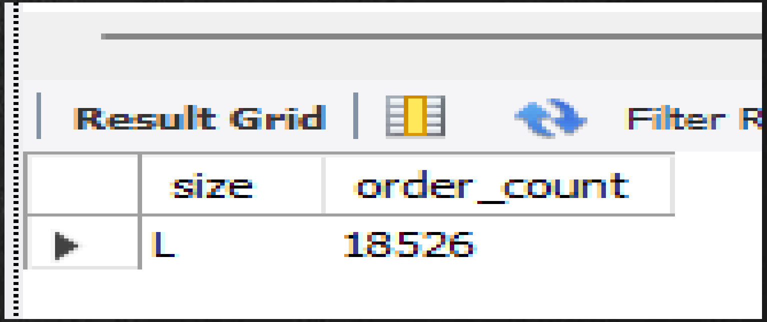
Result Grid | Filter Rows:

	name	price
▶	The Greek Pizza	35.95

4. Identify the most common pizza size ordered?

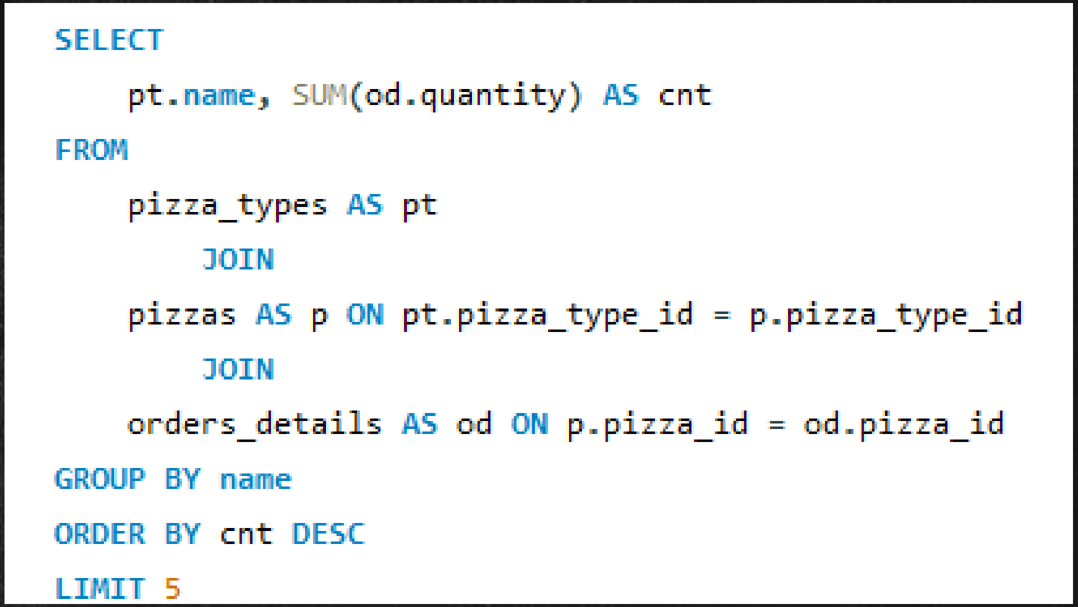


```
SELECT
    p.size, COUNT(od.order_details_id) AS order_count
FROM
    pizzas AS p
        JOIN
    orders_details AS od ON p.pizza_id = od.pizza_id
GROUP BY size
ORDER BY order_count DESC
LIMIT 1;
```

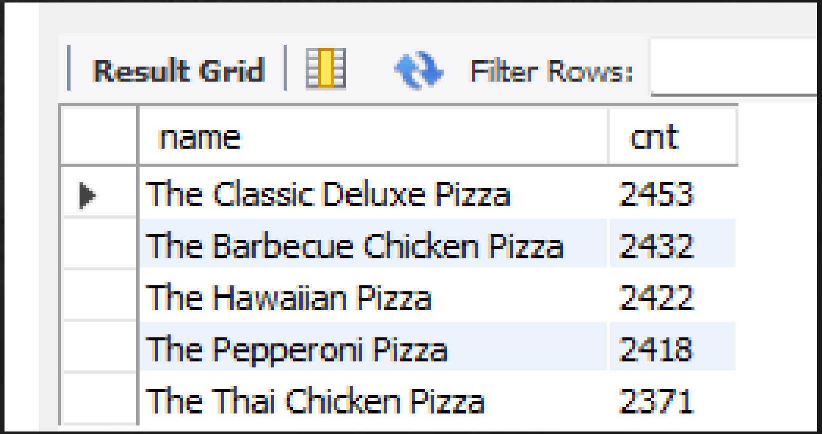


```
size   order_count
L      18526
```

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

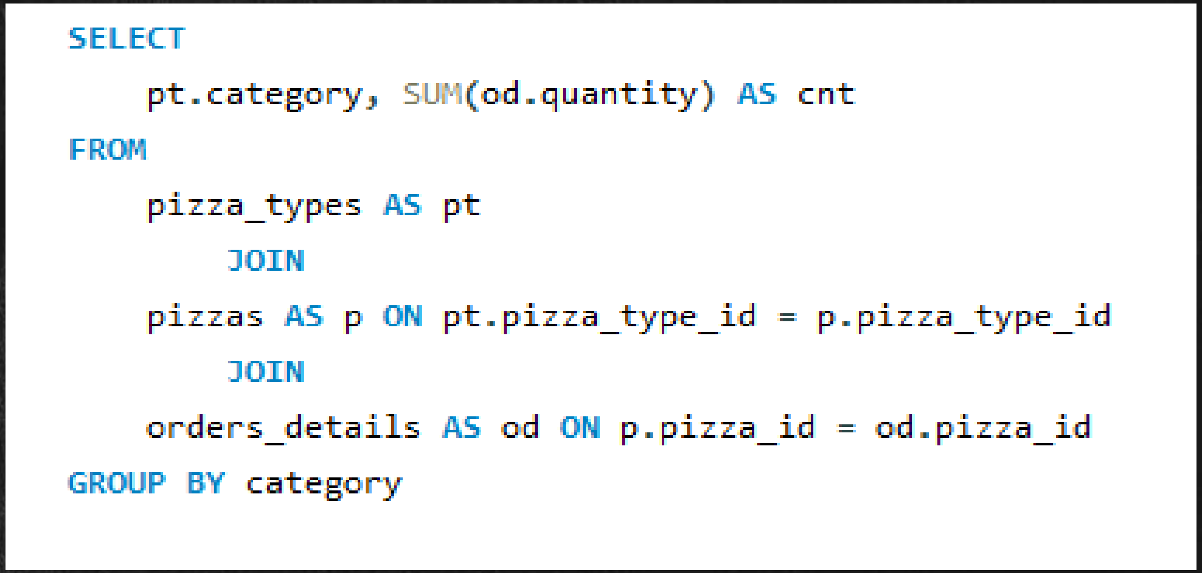


```sql
SELECT
    pt.name, SUM(od.quantity) AS cnt
FROM
    pizza_types AS pt
        JOIN
    pizzas AS p ON pt.pizza_type_id = p.pizza_type_id
        JOIN
    orders_details AS od ON p.pizza_id = od.pizza_id
GROUP BY name
ORDER BY cnt DESC
LIMIT 5
```

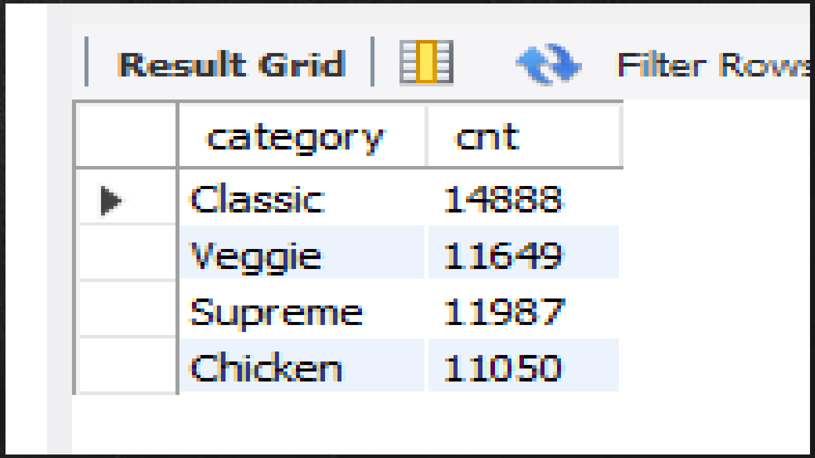


name	cnt
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

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

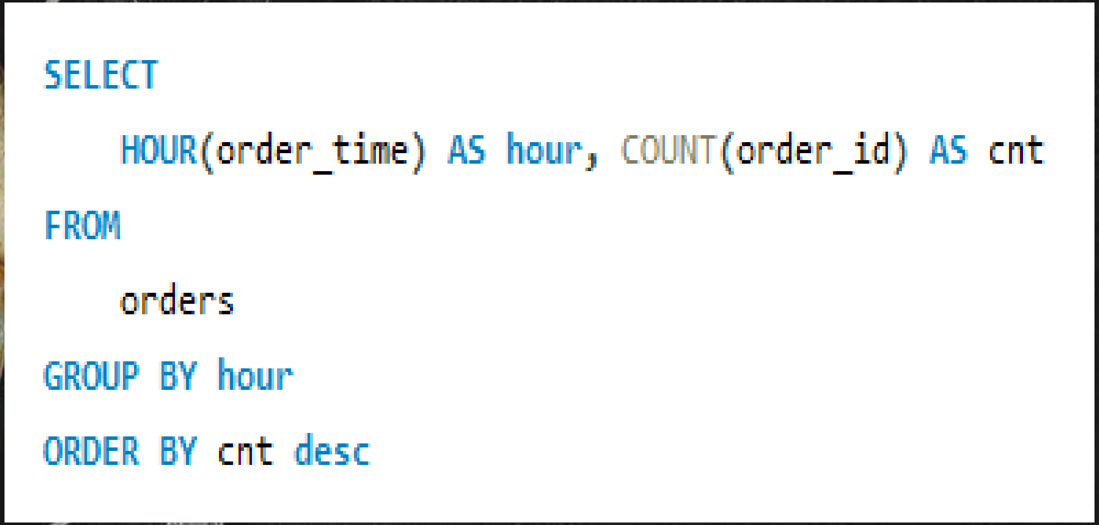


```
SELECT
    pt.category, SUM(od.quantity) AS cnt
FROM
    pizza_types AS pt
        JOIN
    pizzas AS p ON pt.pizza_type_id = p.pizza_type_id
        JOIN
    orders_details AS od ON p.pizza_id = od.pizza_id
GROUP BY category
```

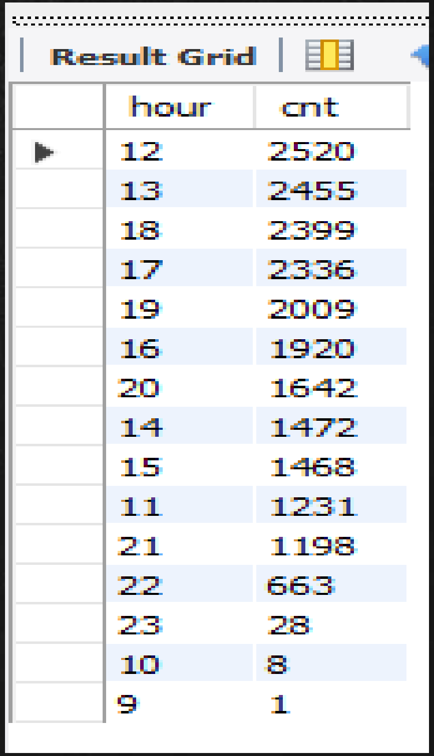


category	cnt
Classic	14888
Veggie	11649
Supreme	11987
Chicken	11050

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

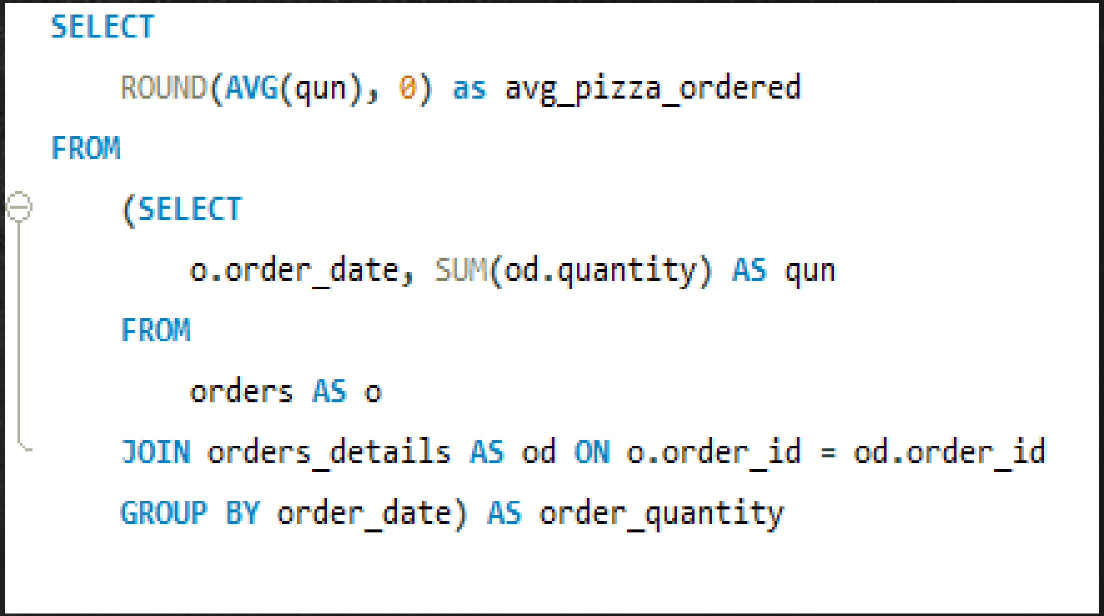


```
SELECT
    HOUR(order_time) AS hour, COUNT(order_id) AS cnt
FROM
    orders
GROUP BY hour
ORDER BY cnt desc
```

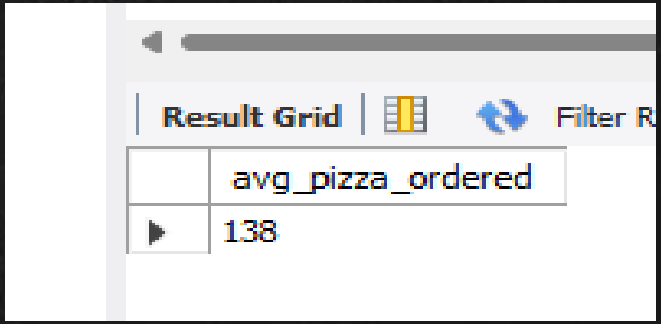


hour	cnt
12	2520
13	2455
18	2399
17	2336
19	2009
16	1920
20	1642
14	1472
15	1468
11	1231
21	1198
22	663
23	28
10	8
9	1

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

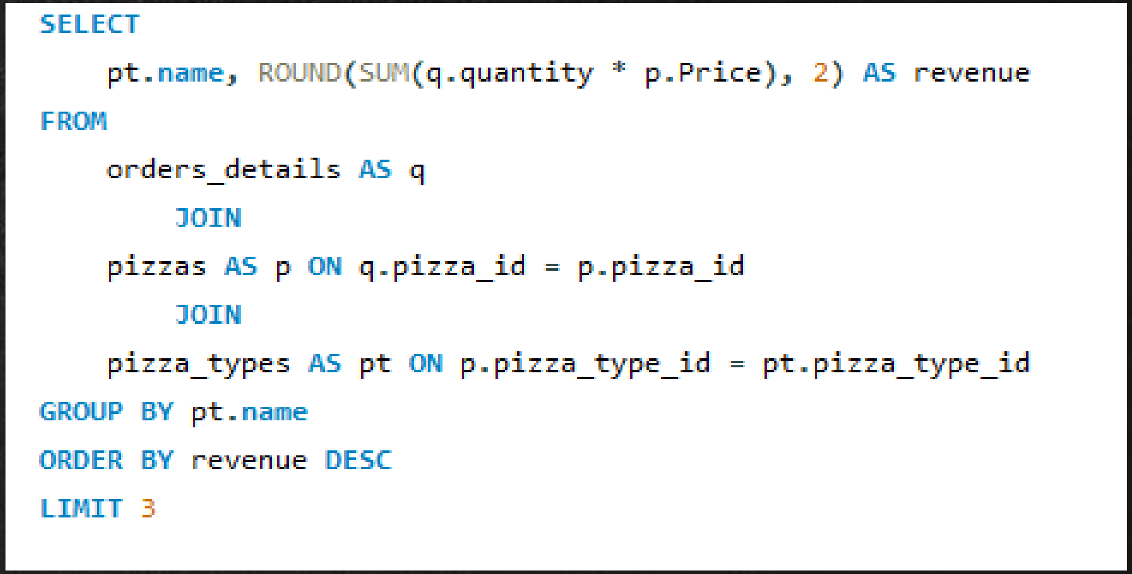


```
SELECT
    ROUND(AVG(qun), 0) as avg_pizza_ordered
FROM
    (SELECT
        o.order_date, SUM(od.quantity) AS qun
    FROM
        orders AS o
    JOIN orders_details AS od ON o.order_id = od.order_id
    GROUP BY order_date) AS order_quantity
```

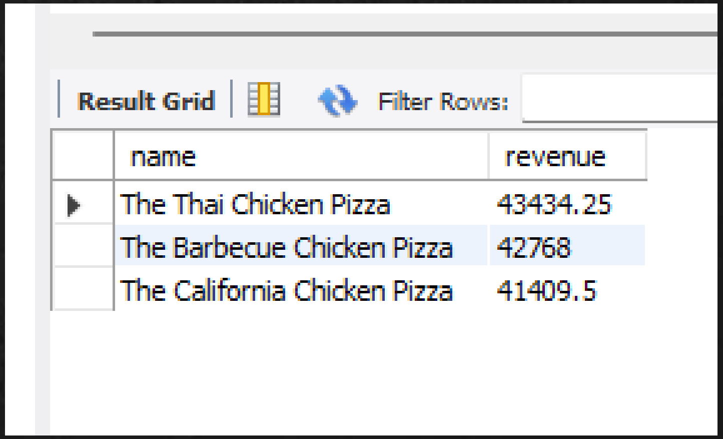


```
avg_pizza_ordered
138
```

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

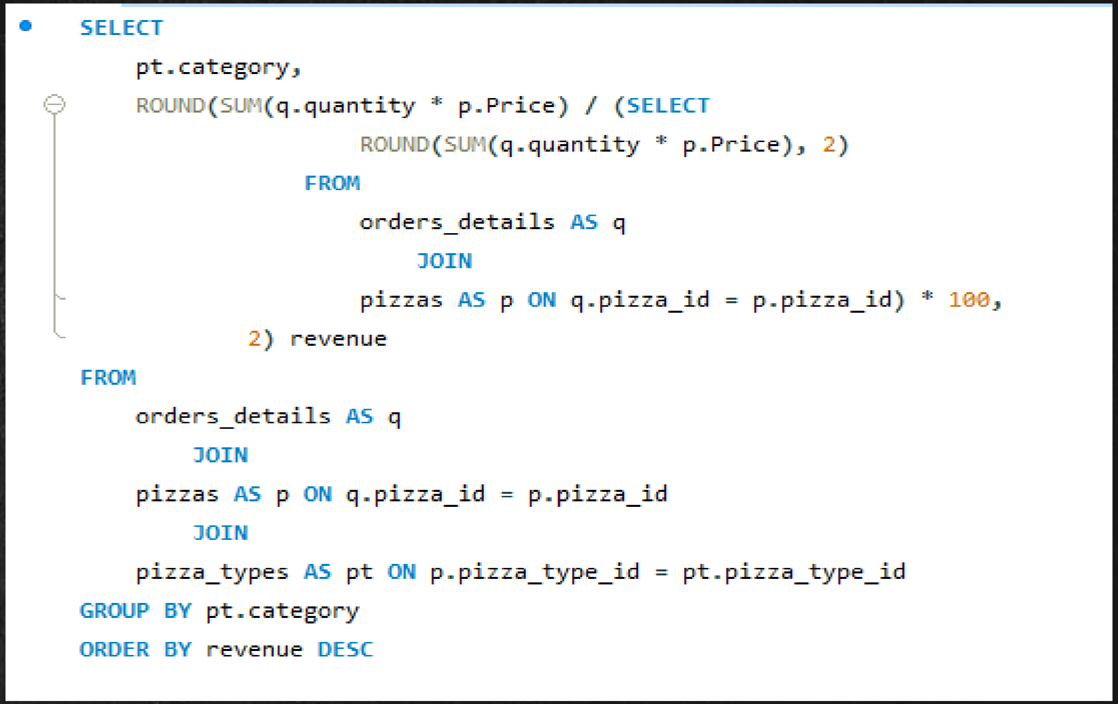


```
SELECT
    pt.name, ROUND(SUM(q.quantity * p.Price), 2) AS revenue
FROM
    orders_details AS q
        JOIN
    pizzas AS p ON q.pizza_id = p.pizza_id
        JOIN
    pizza_types AS pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.name
ORDER BY revenue DESC
LIMIT 3
```

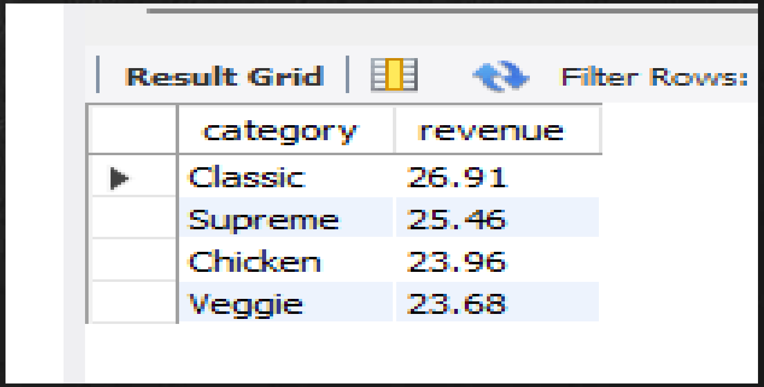


name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

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

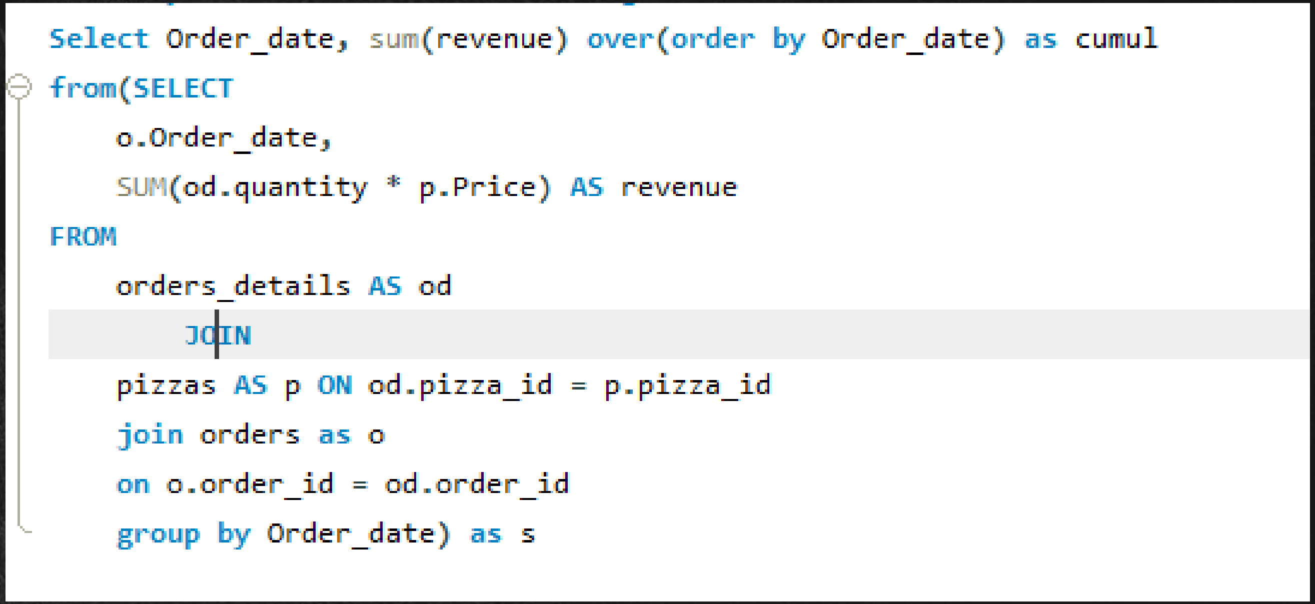


```
SELECT
    pt.category,
    ROUND(SUM(q.quantity * p.Price) / (SELECT
                ROUND(SUM(q.quantity * p.Price), 2)
            FROM
                orders_details AS q
                    JOIN
                pizzas AS p ON q.pizza_id = p.pizza_id) * 100,
        2) revenue
FROM
    orders_details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

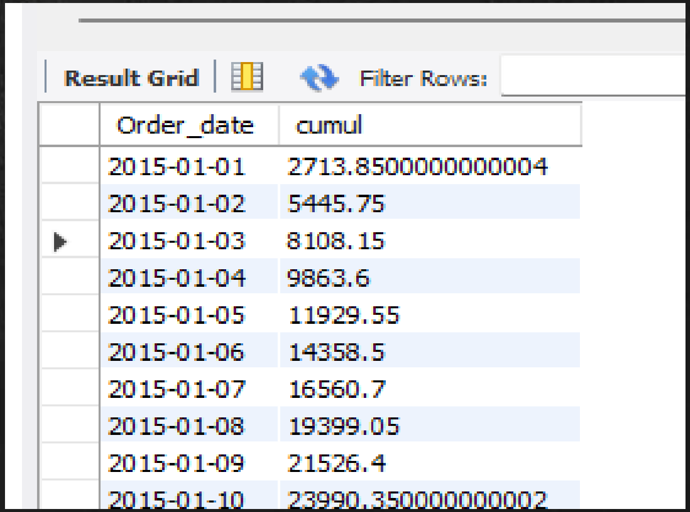


category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

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

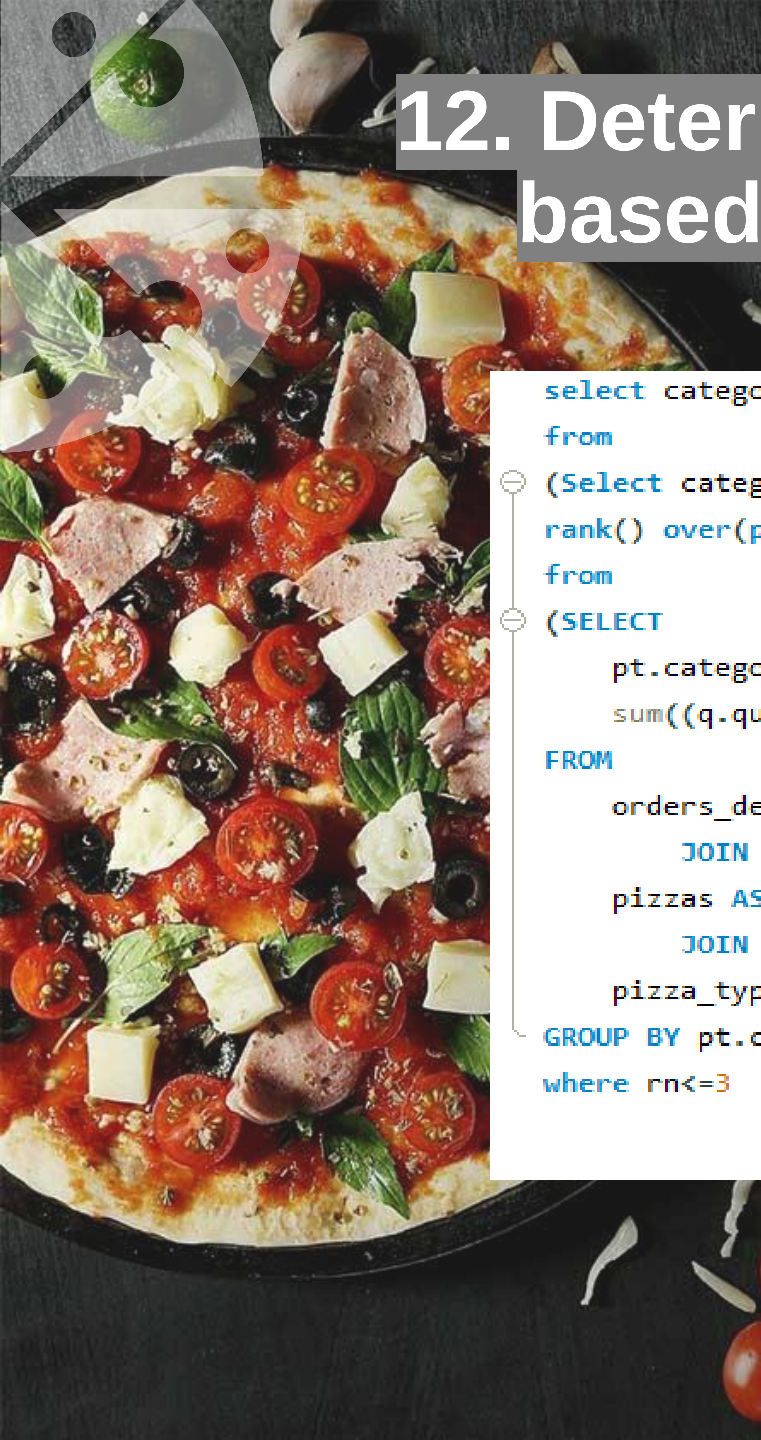


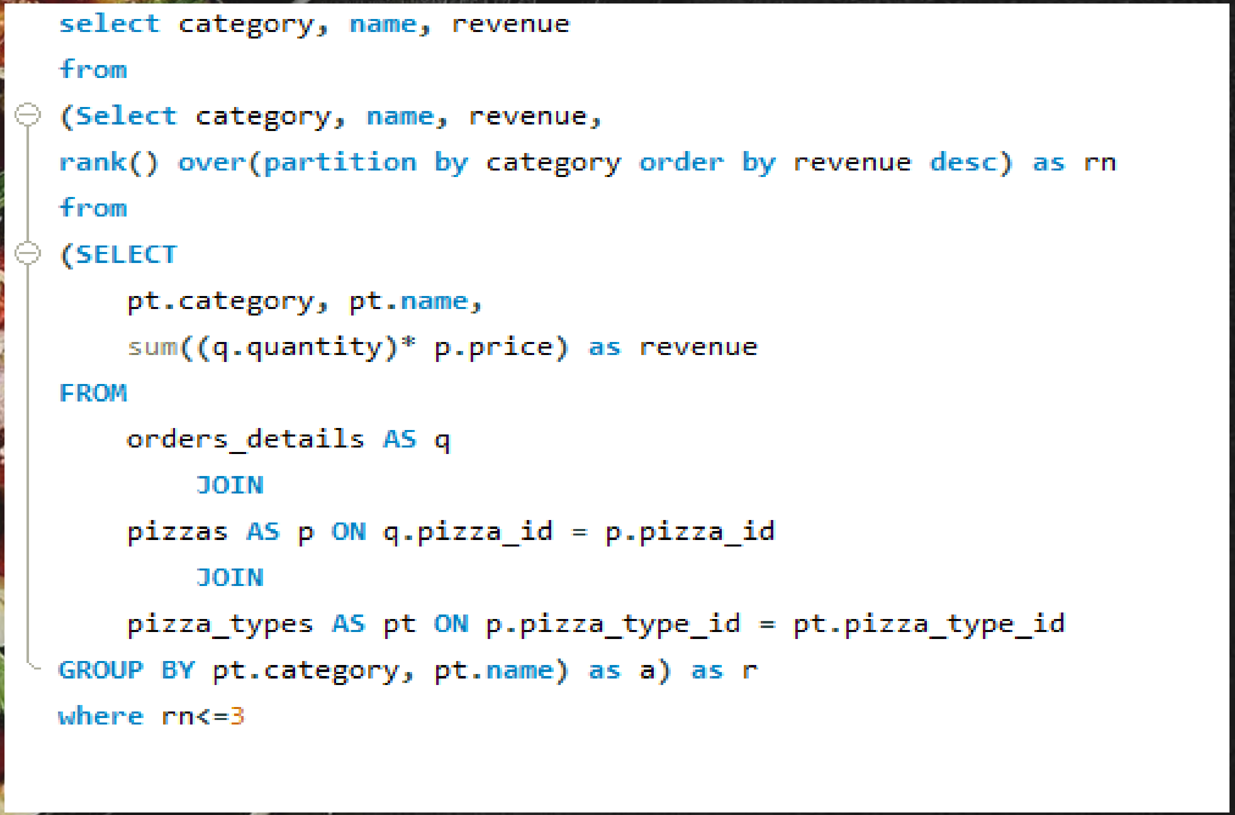
```
Select Order_date, sum(revenue) over(order by Order_date) as cumul
from(SELECT
    o.Order_date,
    SUM(od.quantity * p.Price) AS revenue
FROM
    orders_details AS od
        JOIN
    pizzas AS p ON od.pizza_id = p.pizza_id
    join orders as o
    on o.order_id = od.order_id
    group by Order_date) as s
```



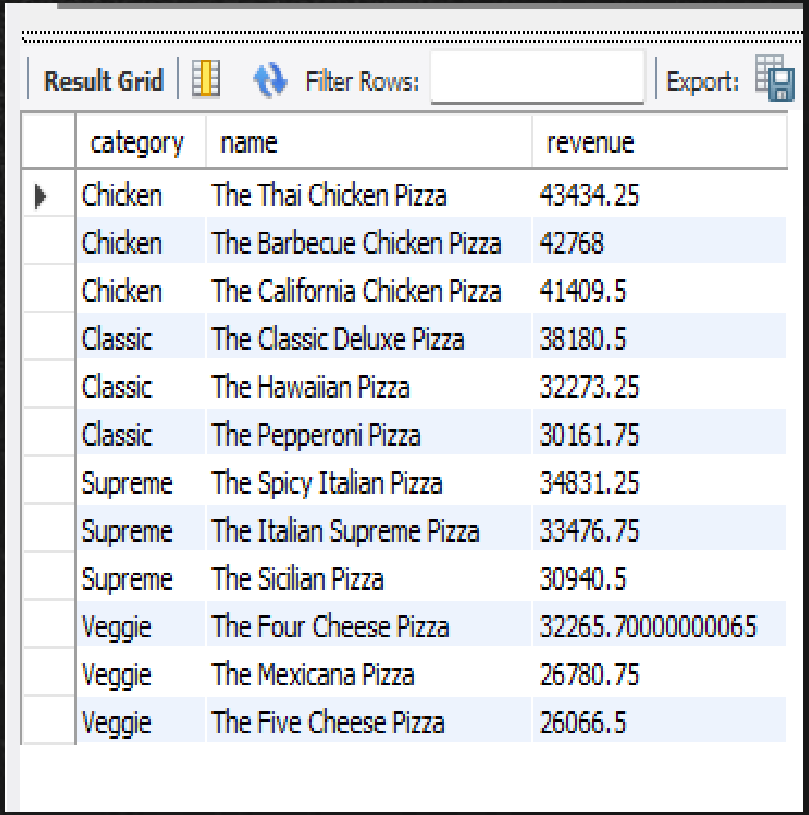
Order_date	cumul
2015-01-01	2713.8500000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.4
2015-01-10	23990.350000000002

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





```
select category, name, revenue
from
(Select category, name, revenue,
rank() over(partition by category order by revenue desc) as rn
from
(SELECT
    pt.category, pt.name,
    sum((q.quantity)* p.price) as revenue
FROM
    orders_details AS q
        JOIN
    pizzas AS p ON q.pizza_id = p.pizza_id
        JOIN
    pizza_types AS pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.category, pt.name) as a) as r
where rn<=3
```



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	32265.7000000065
Veggie	The Mexicana Pizza	26780.75
Veggie	The Five Cheese Pizza	26066.5

THANK YOU