PIZZAHUT SALES ANALYSIS USING SQL

Prepared by - Maira Waseem

Project Overview

This project analyzes a Pizza Sales dataset. It explores sales by category, price, quantity, and hours, providing detailed insights. The analysis highlights key findings that offer valuable information about pizza sales performance.

Go Ahead To Explore More !

ANALYSIS QUESTIONS :

1.Retrieve the total number of orders placed.

2.Calculate the total revenue generated from pizza sales.

3.Identify the highest-priced pizza.

4.Identify the most common pizza size ordered.

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

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

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

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

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

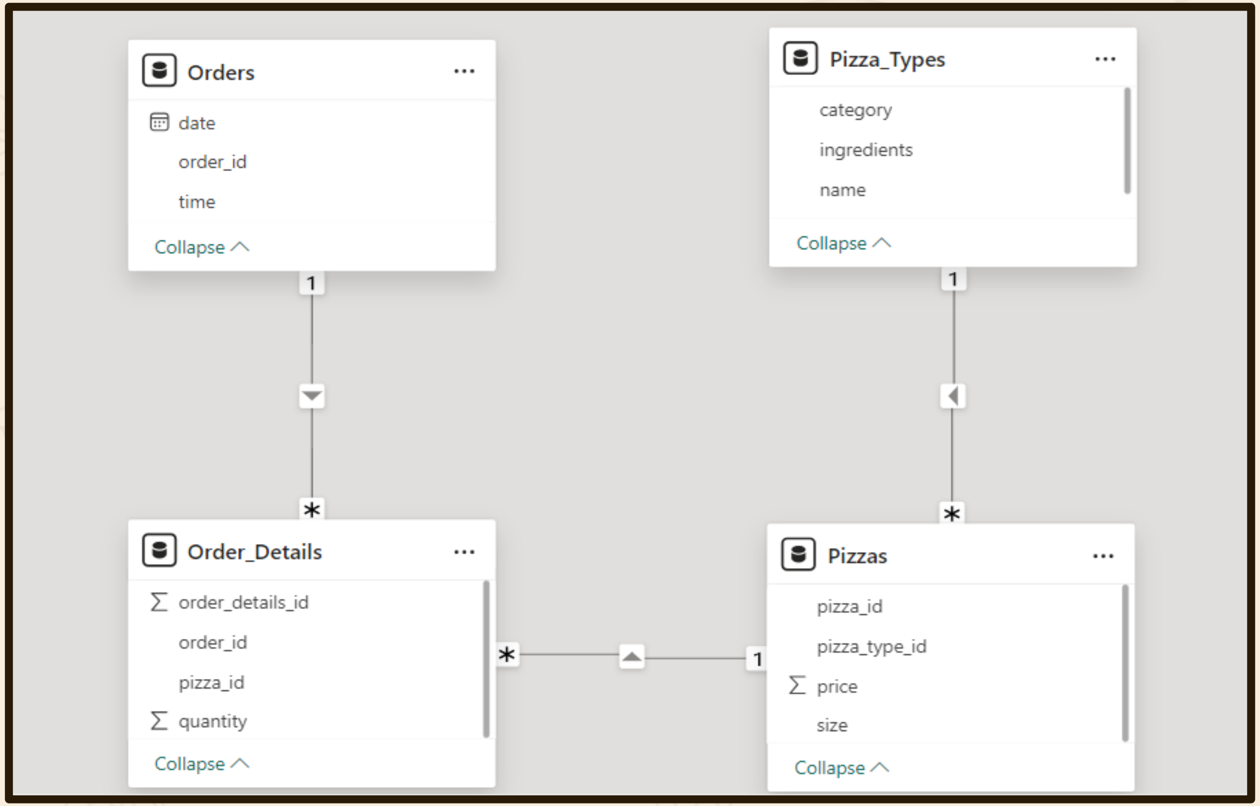
10.Determine the top 3 most ordered pizza types based on revenue.

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

12.Analyze the cumulative revenue generated over time.

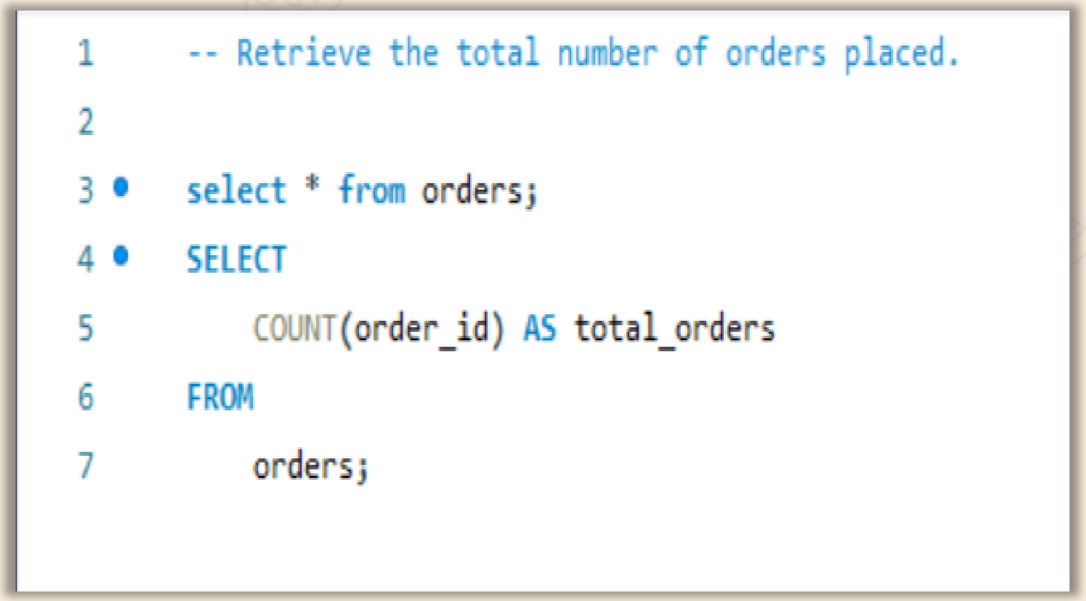
13.Determine the top 3 most ordered pizza types based on revenue for each pizza category.

DATA MODEL


Orders
date
order_id
time
Collapse
Pizza_Types
category
ingredients
name
Collapse
Order_Details
order_details_id
order_id
pizza_id
quantity
Collapse
Pizzas
pizza_id
pizza_type_id
price
size
Collapse
1
*
1
*
*
1

1.Retrieve The Total Numbers Of Orders Placed.

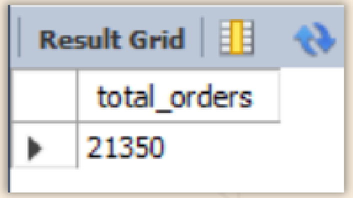
SQL Query :



```
-- Retrieve the total number of orders placed.

select * from orders;
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

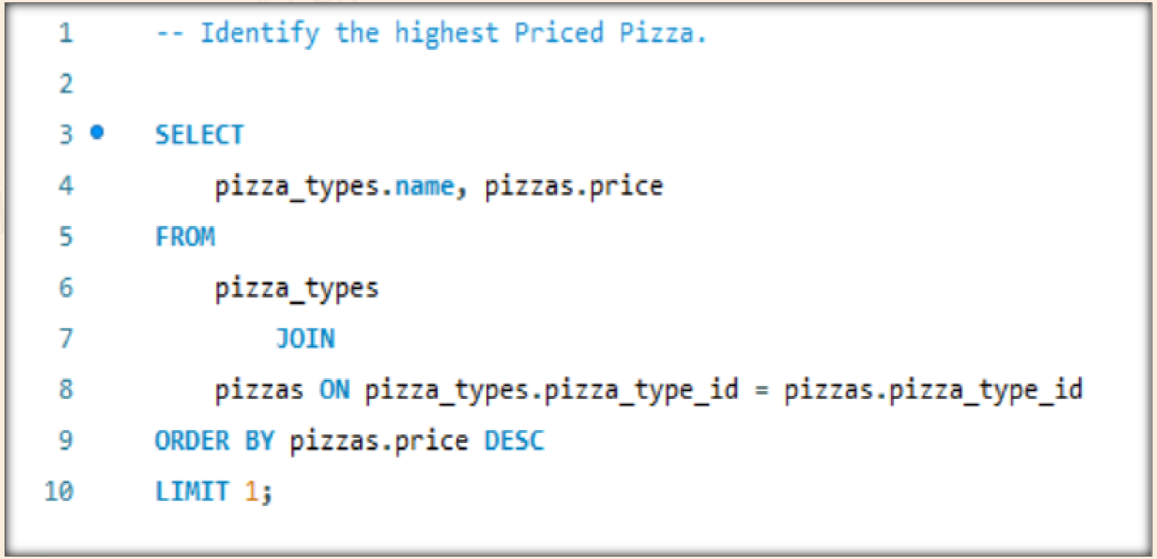
Output :



```
total_orders
21350
```

2.Identify the Highest Priced Pizza.

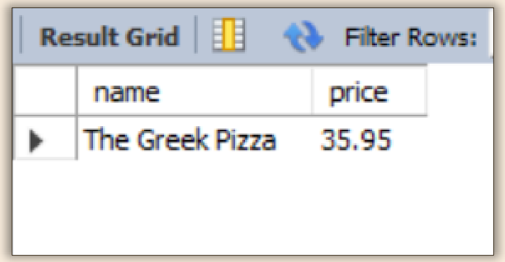
SQL Query :



```
-- 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;
```

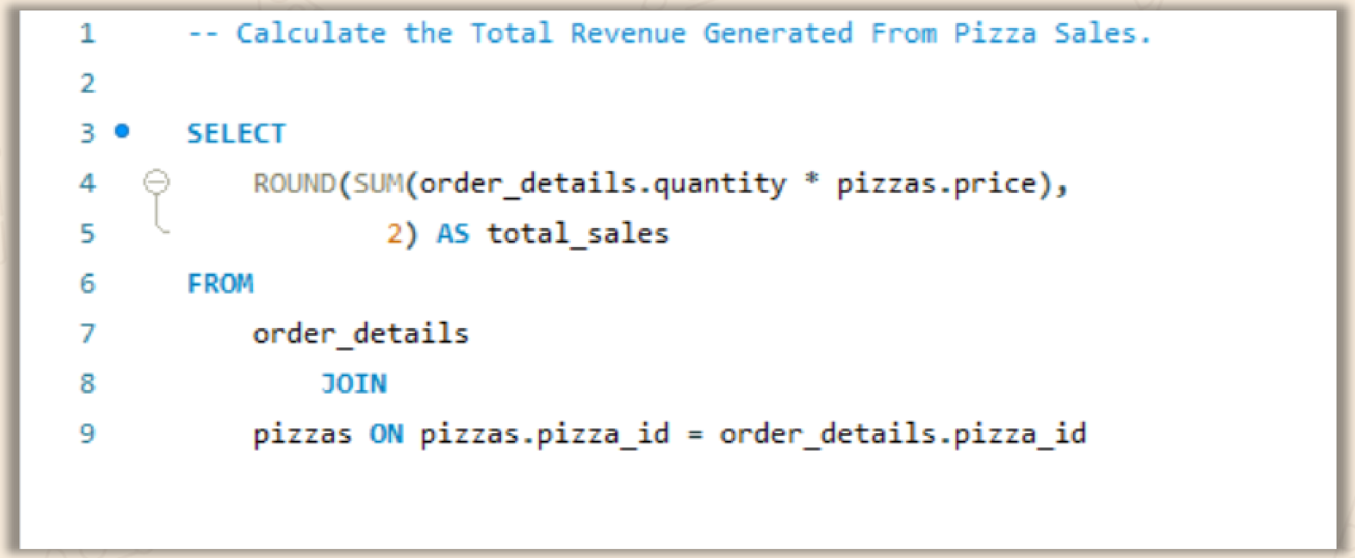
Output :



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

3.Calculate the Total Revenue Generated From the Pizza Sales.

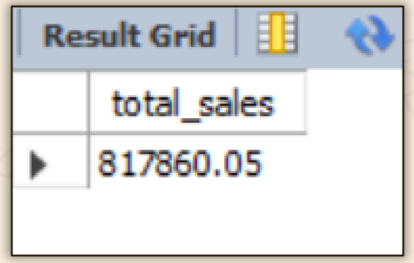
SQL Query :



```
-- Calculate the Total Revenue Generated From Pizza Sales.

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

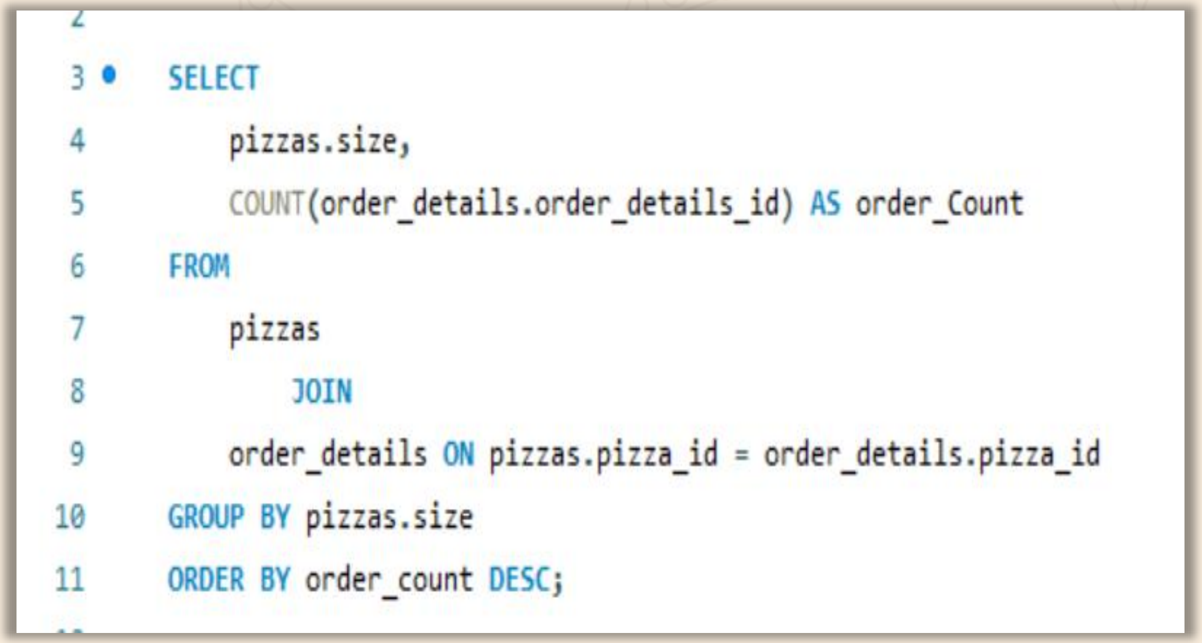
Output :



```
total_sales
817860.05
```

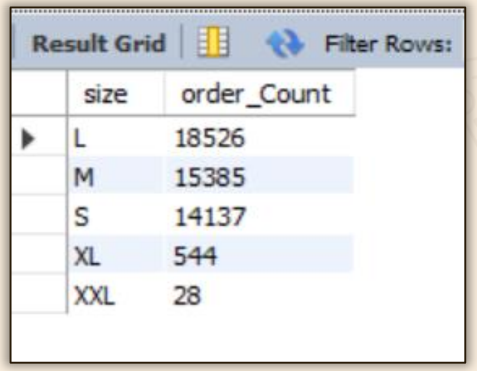
4.Identify the Most Common Pizza Size Ordered.

SQL Query :



```
SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) AS order_Count
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;
```

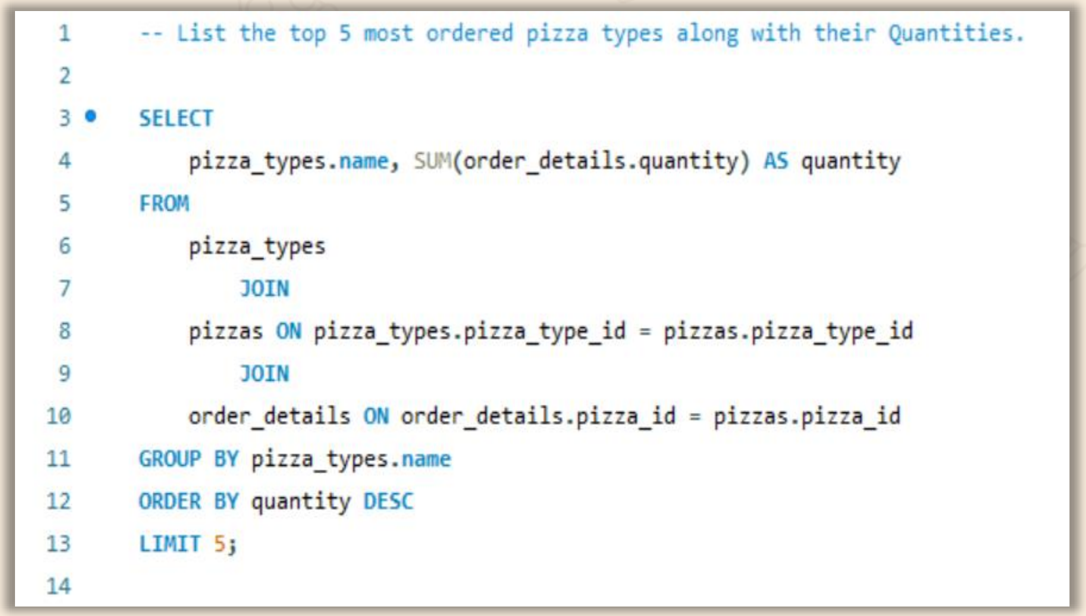
Output :



```
size  order_Count
L     18526
M     15385
S     14137
XL    544
XXL   28
```

5.List the Top 5 most ordered Pizza Type along with their Quantities.

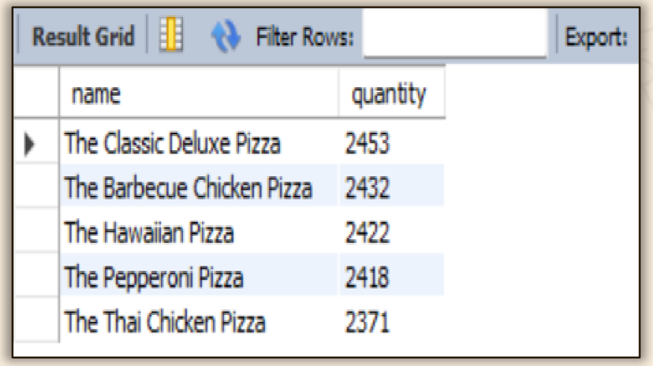
SQL Query :



```sql
-- List the top 5 most ordered pizza types along with their Quantities.

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

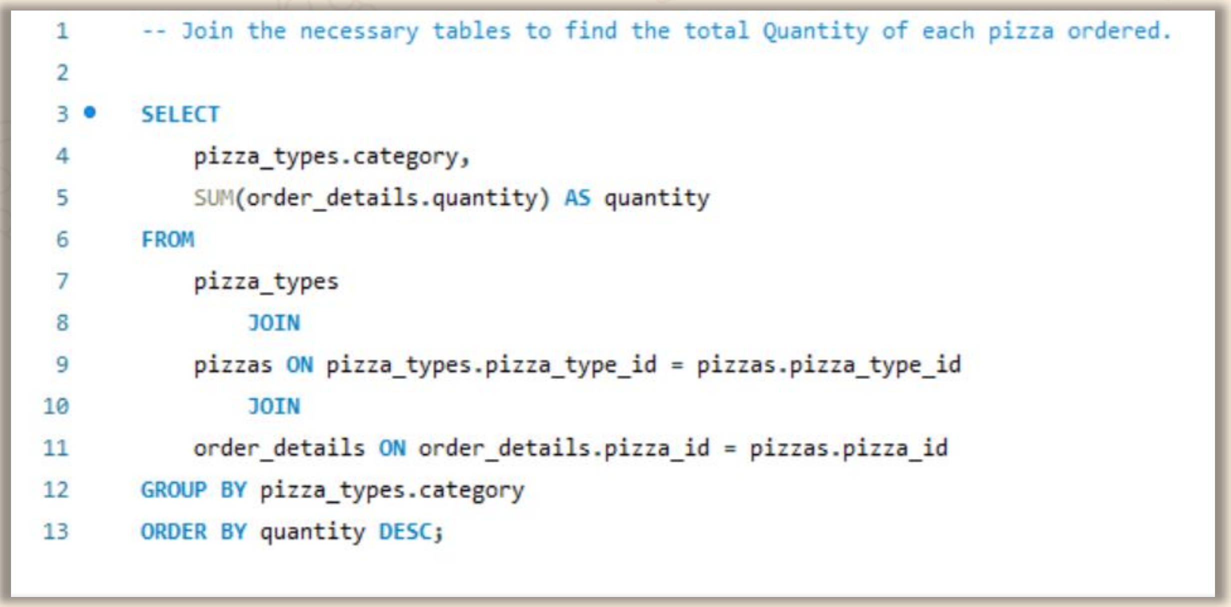
Output :



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

6.Determine the Total Quantity of each Pizza Ordered.

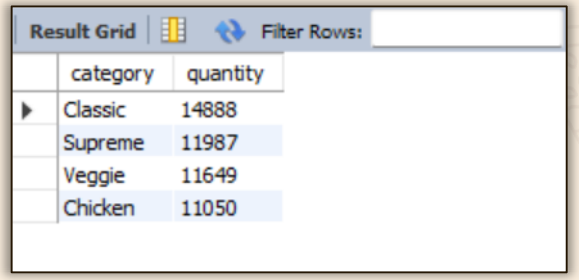
SQL Query :



```sql
-- Join the necessary tables to find the total Quantity of each pizza ordered.

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

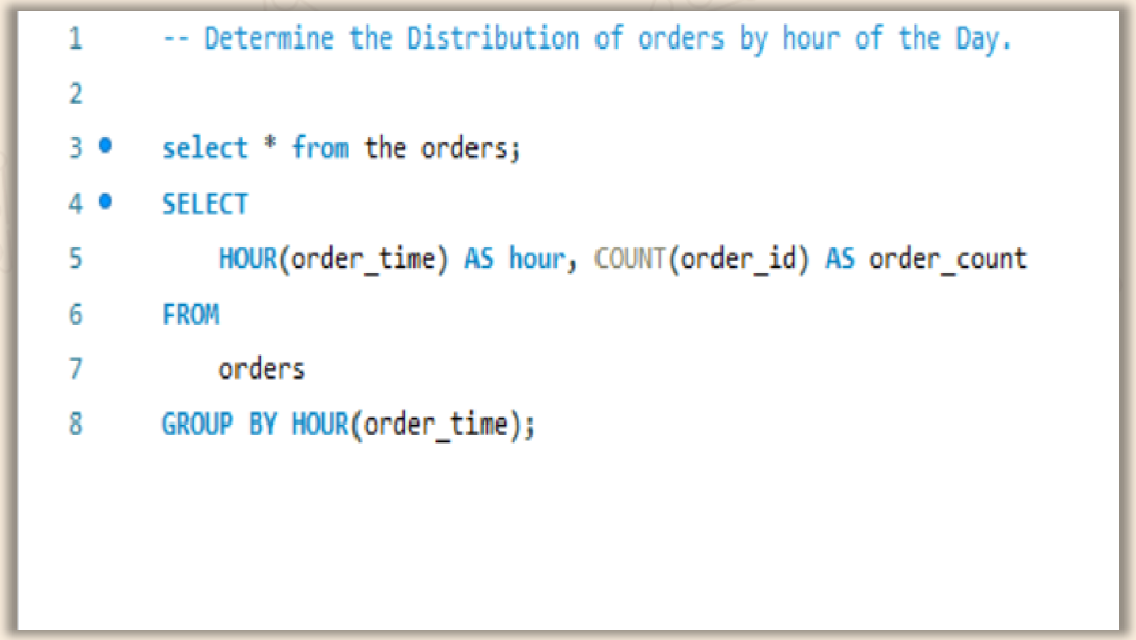
Output :



category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

7.Find the Distribution Of Orders by hour of the Data.

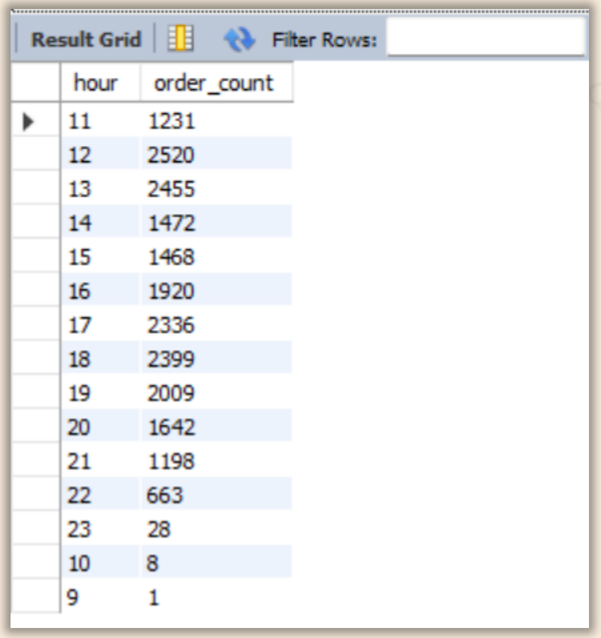
SQL Query :



```
-- Determine the Distribution of orders by hour of the Day.

select * from the orders;
SELECT
    HOUR(order_time) AS hour, COUNT(order_id) AS order_count
FROM
    orders
GROUP BY HOUR(order_time);
```

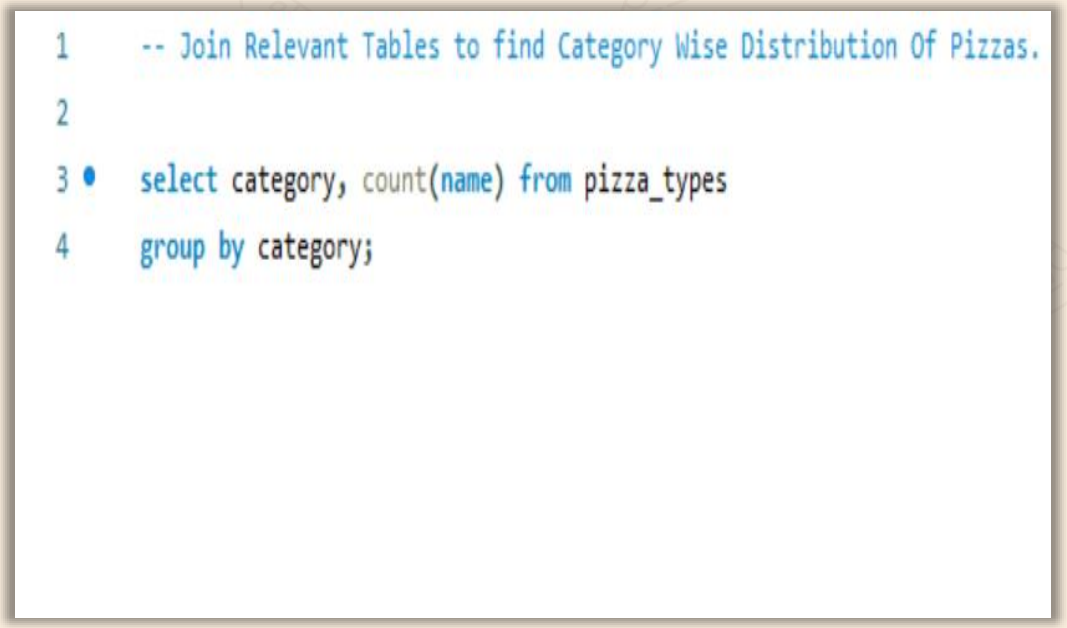
Output :



hour	order_count
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

8.Find the Category Wise Distribution Of Pizzas.

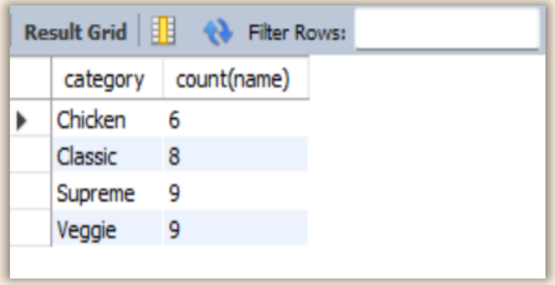
SQL Query :



```
-- Join Relevant Tables to find Category Wise Distribution Of Pizzas.

select category, count(name) from pizza_types
group by category;
```

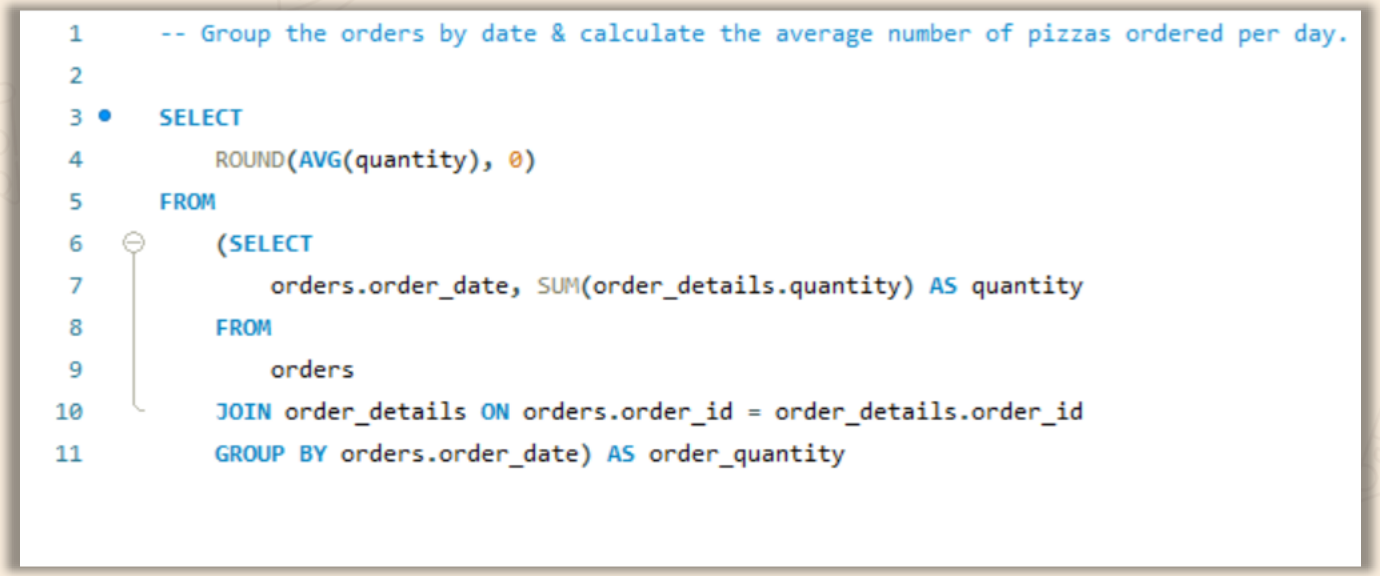
Output :



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

9.Group the Orders By Date & Calculate the Average Number Of Pizzas Ordered Per Day.

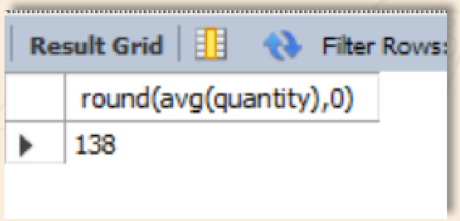
SQL Query :



```sql
-- Group the orders by date & calculate the average number of pizzas ordered per day.

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

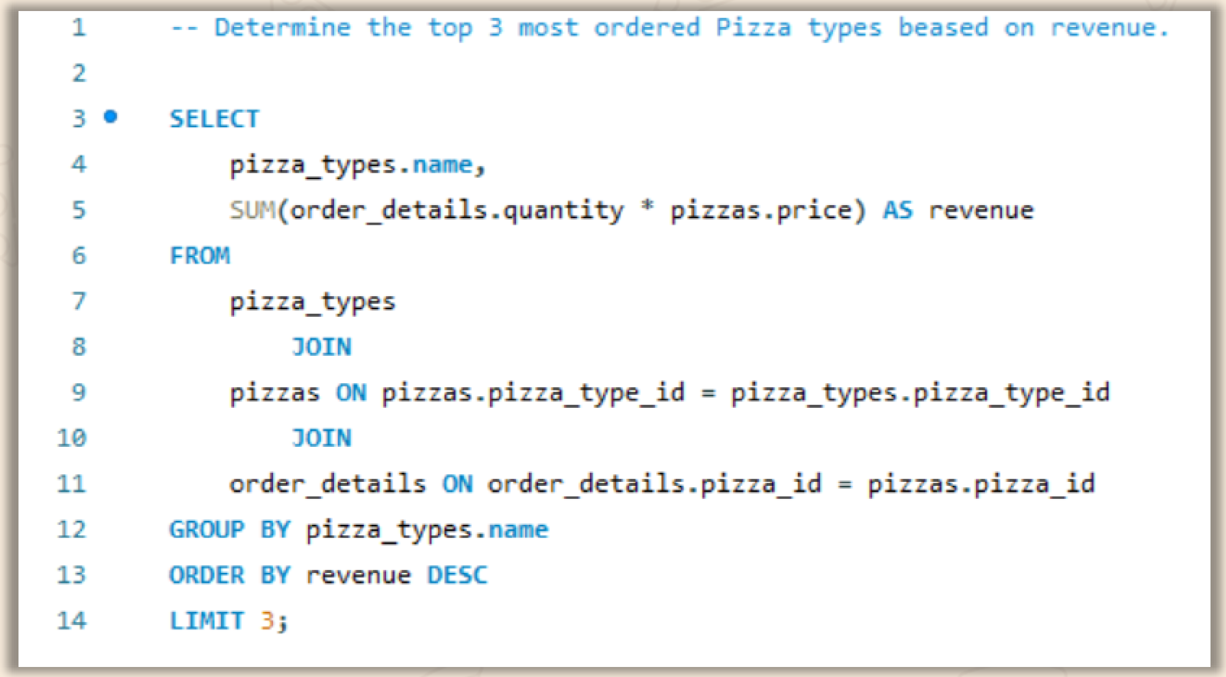
Output :



```
round(avg(quantity),0)
138
```

10.Determine the Top 3 most Pizzas Ordered Based On Revenue.

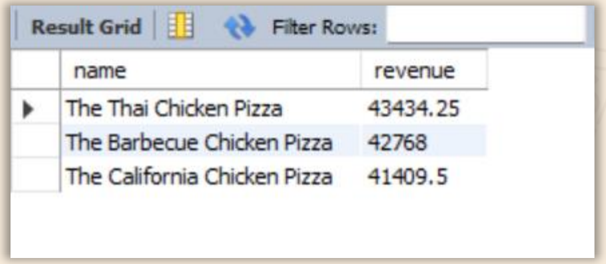
SQL Query :



```sql
-- Determine the top 3 most ordered Pizza types beased on revenue.

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

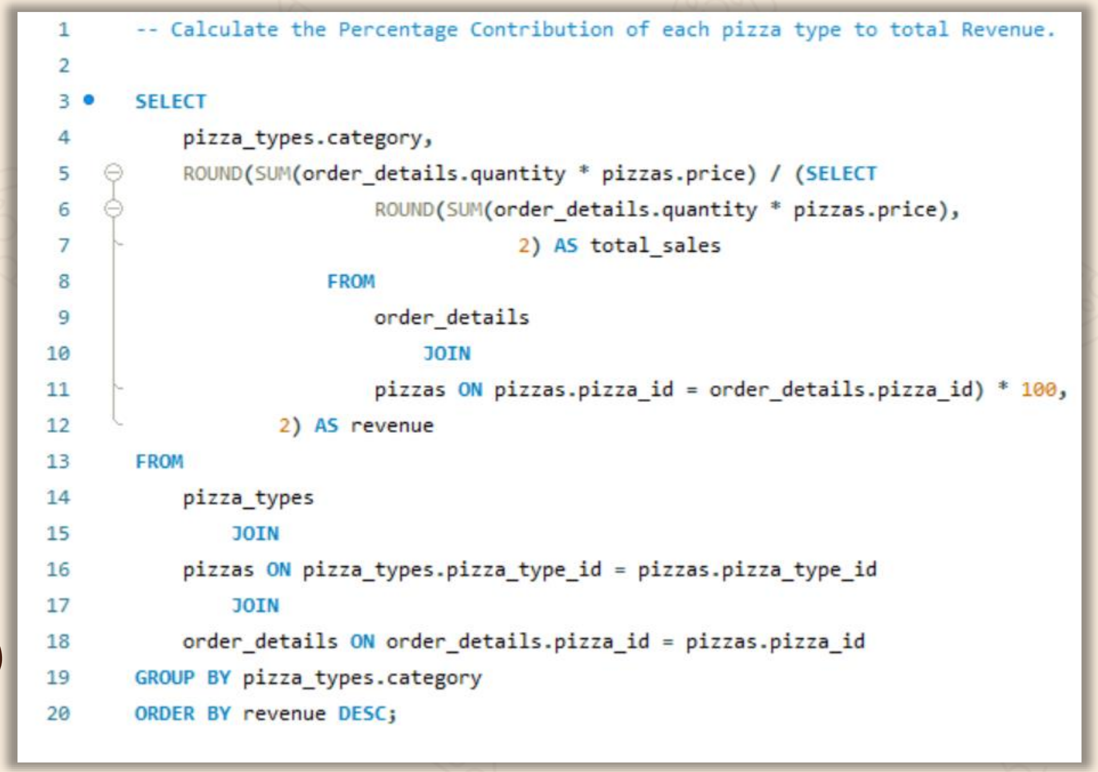
Output :



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

11.Calculate the Percentage Contribution of each Pizza Type to total Revenue.

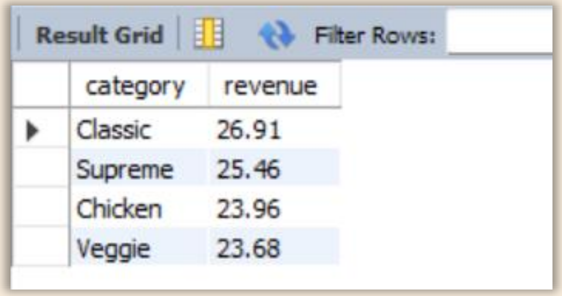
SQL Query :



```sql
-- Calculate the Percentage Contribution of each pizza type to total Revenue.

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

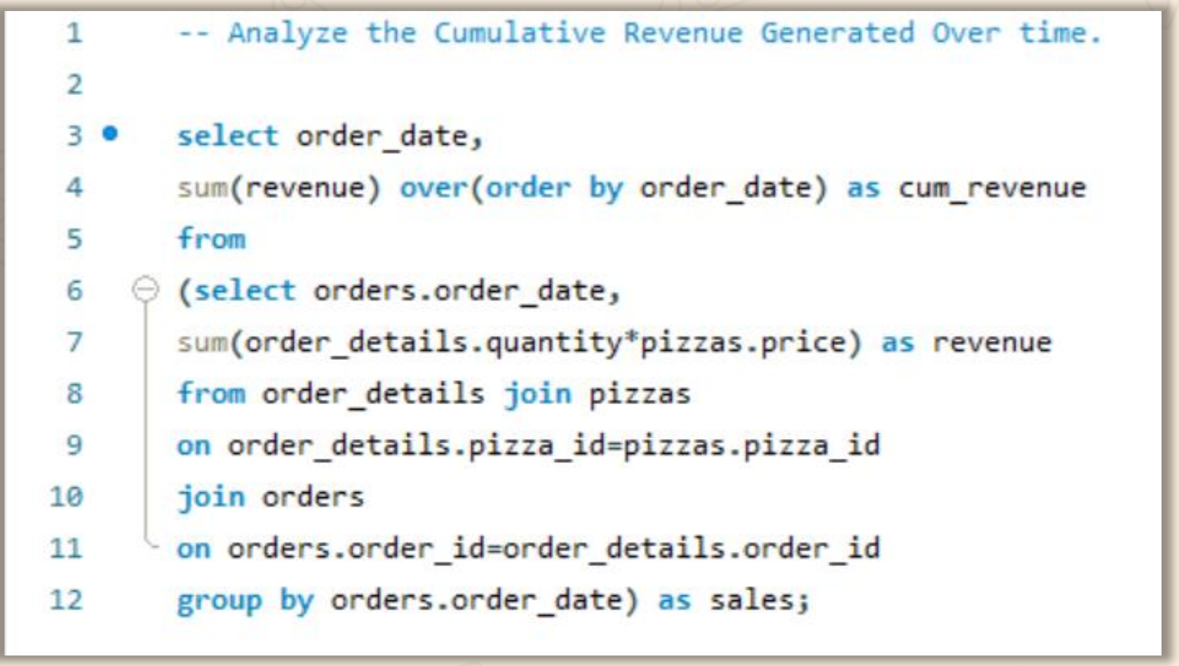
Output :



category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

12.Analyze the Cumulative Revenue Generated Over Time.

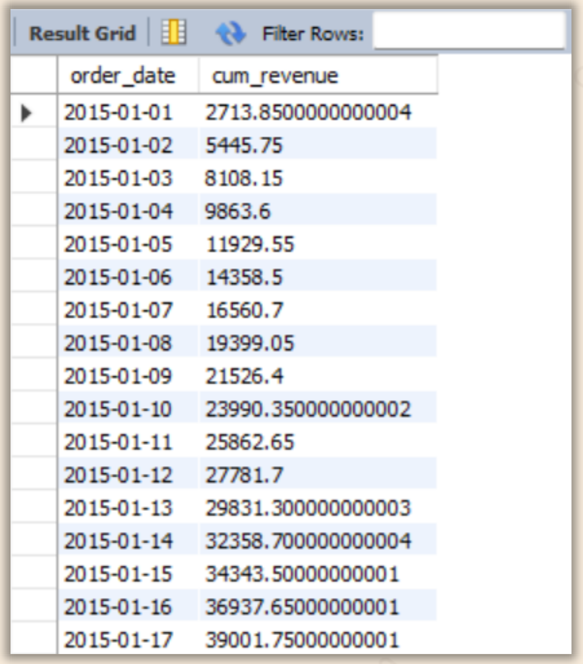
SQL Query :



```
-- Analyze the Cumulative Revenue Generated Over time.

select order_date,
sum(revenue) over(order by order_date) as cum_revenue
from
(select orders.order_date,
sum(order_details.quantity*pizzas.price) as revenue
from order_details join pizzas
on order_details.pizza_id=pizzas.pizza_id
join orders
on orders.order_id=order_details.order_id
group by orders.order_date) as sales;
```

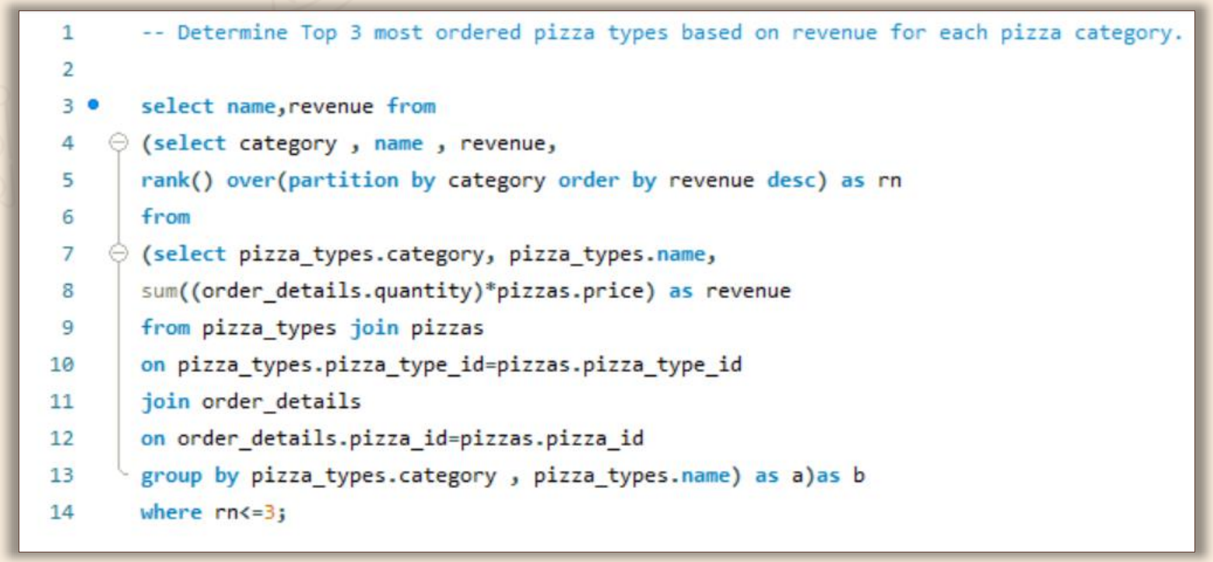
Output :



order_date	cum_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

13.Determine the Top 3 most Ordered Pizza Types Based on Revenue For Each Category.

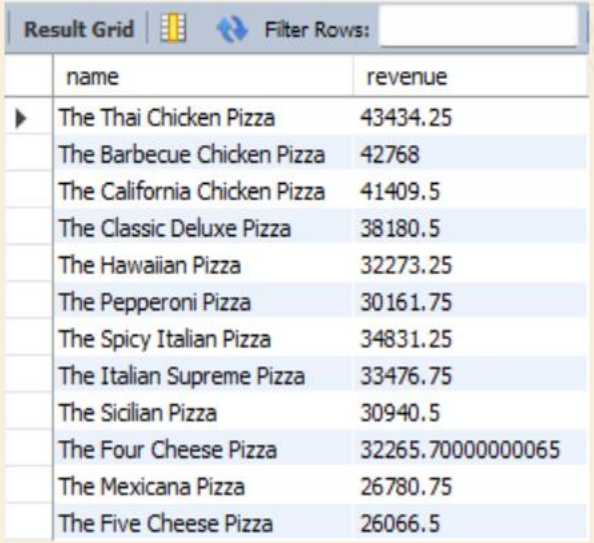
SQL Query :



```sql
-- Determine Top 3 most ordered pizza types based on revenue for each pizza category.

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((order_details.quantity)*pizzas.price) as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id=pizzas.pizza_type_id
join order_details
on order_details.pizza_id=pizzas.pizza_id
group by pizza_types.category , pizza_types.name) as a)as b
where rn<=3;
```

Output :



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

Conclusion:

This project was a blend of theoretical knowledge and hands-on experience in efficiently analyzing a pizza sales dataset. By exploring various aspects of the dataset, such as pizza types, order IDs, order dates and times, and order quantities, we gained valuable insights.
Delving deeper into the world of SQL, this project enhanced my analytical skills and allowed me to apply them to real-world scenarios.

THANKS!

Any Queries?

www.linkedin.com/in/maira-waseem-764b68301