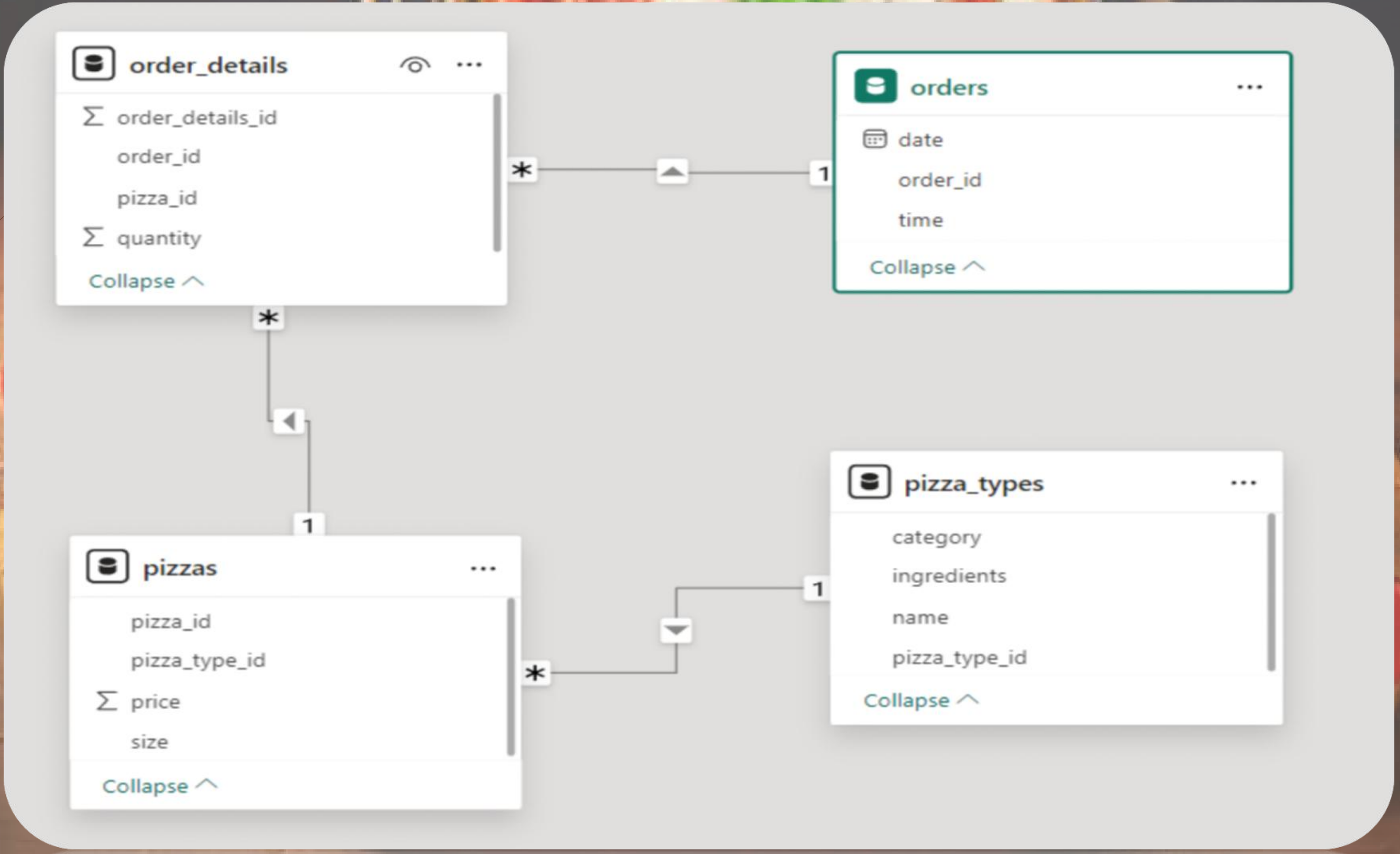
PIZZA SALES ANALYSIS (SQL)

HELLO!

My Name is Akash Kundu. This project belong to pizza sales analysis in which I use MySQL to solve the problems and find the insights.

Schema Design



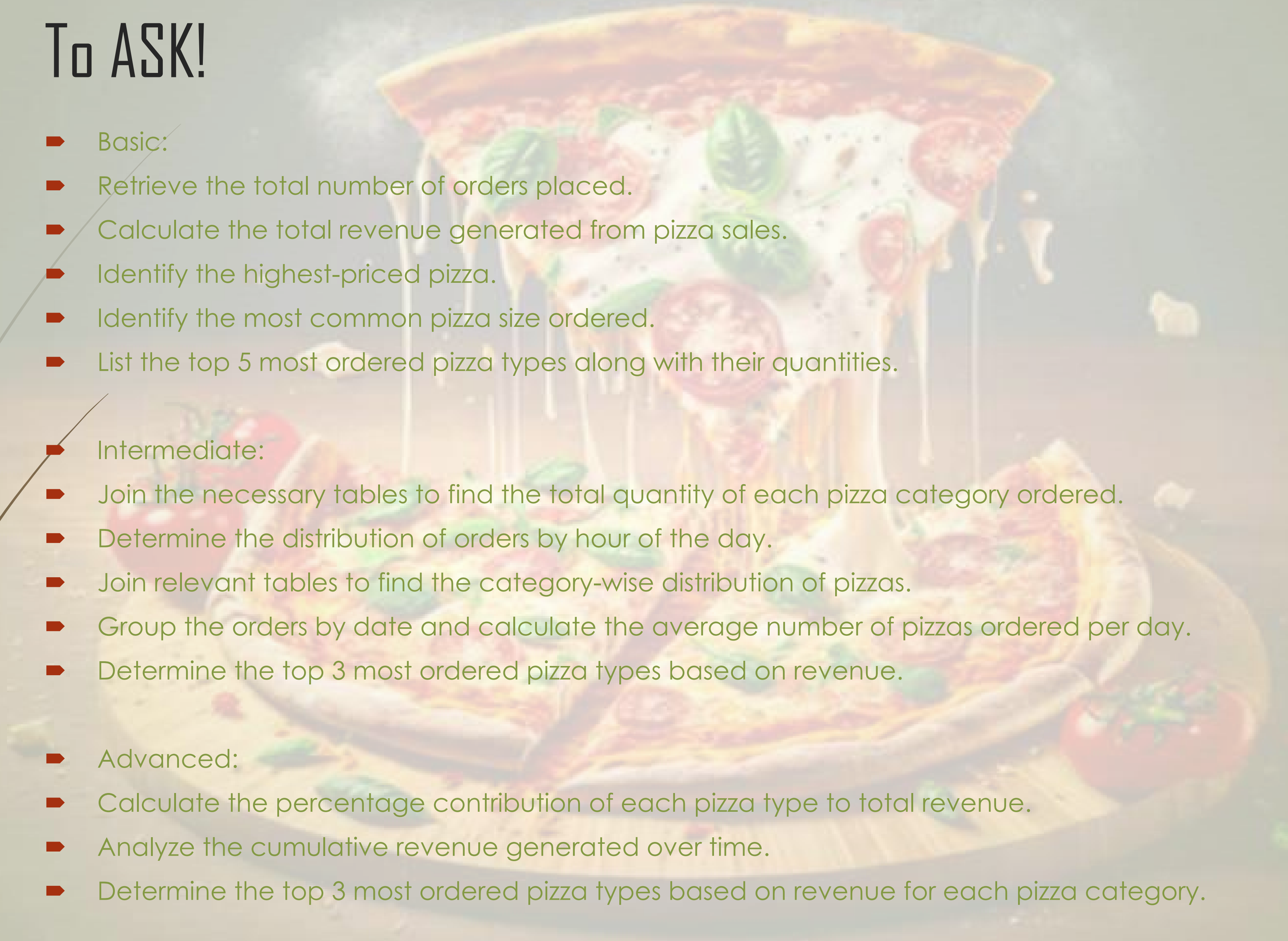
order_details
- order_details_id
- order_id
- pizza_id
- quantity

orders
- date
- order_id
- time

pizzas
- pizza_id
- pizza_type_id
- price
- size

pizza_types
- category
- ingredients
- name
- pizza_type_id

To ASK!

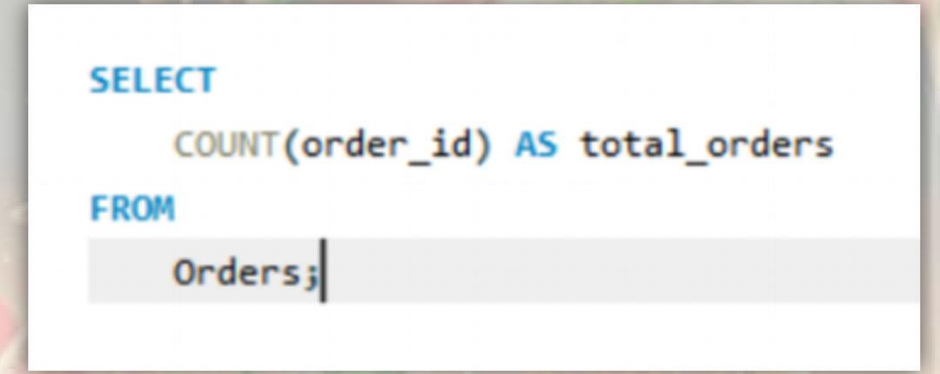


- Basic:
- Retrieve the total number of orders placed.
- Calculate the total revenue generated from pizza sales.
- Identify the highest-priced pizza.
- Identify the most common pizza size ordered.
- List the top 5 most ordered pizza types along with their quantities.

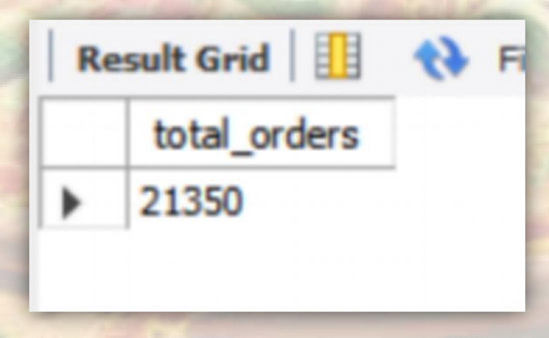
- Intermediate:
- Join the necessary tables to find the total quantity of each pizza category ordered.
- Determine the distribution of orders by hour of the day.
- Join relevant tables to find the category-wise distribution of pizzas.
- Group the orders by date and calculate the average number of pizzas ordered per day.
- Determine the top 3 most ordered pizza types based on revenue.

- Advanced:
- Calculate the percentage contribution of each pizza type to total revenue.
- Analyze the cumulative revenue generated over time.
- Determine the top 3 most ordered pizza types based on revenue for each pizza category.

Q1. Retrieve the total number of orders placed.

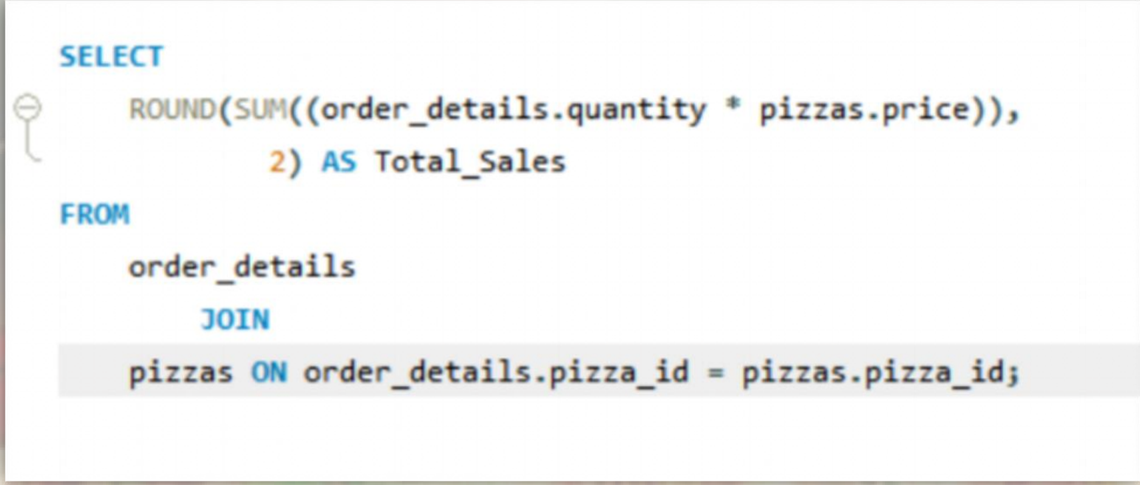


```
SELECT
    COUNT(order_id) AS total_orders
FROM
    Orders;
```

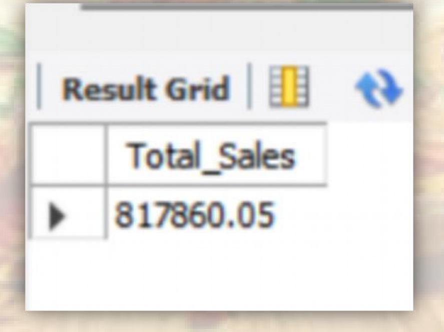


```
total_orders
21350
```

Q2. Calculate the total revenue generated from pizza sales.

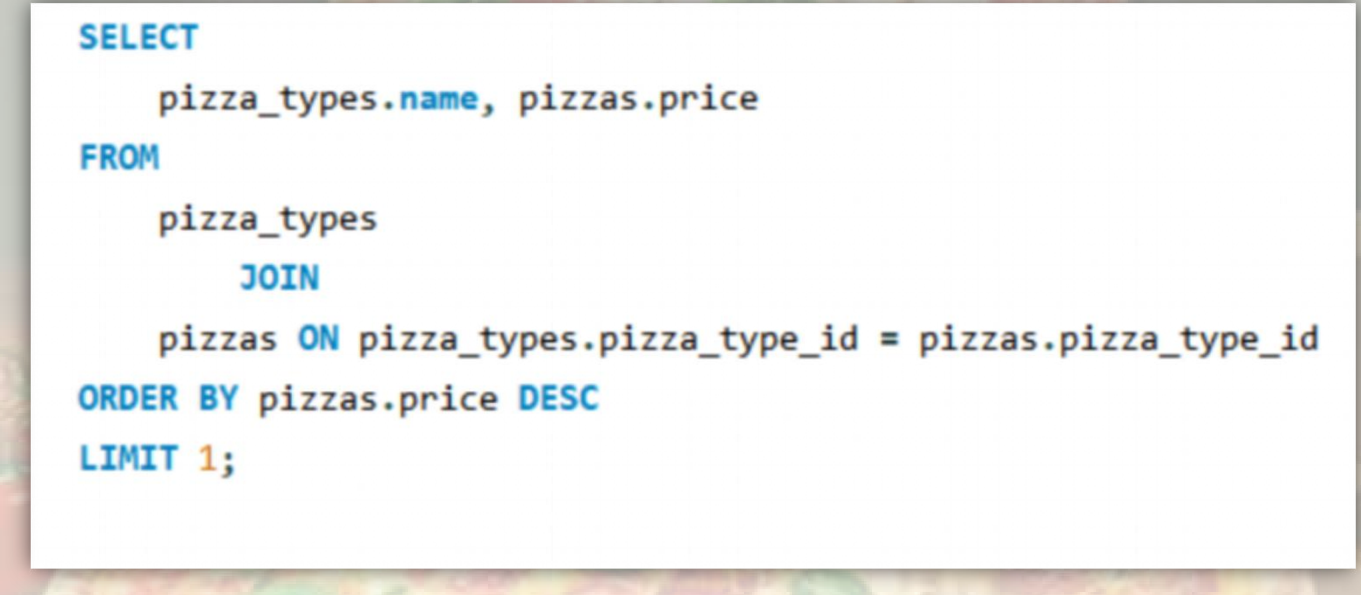


```
SELECT
    ROUND(SUM((order_details.quantity * pizzas.price)),
            2) AS Total_Sales
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id;
```

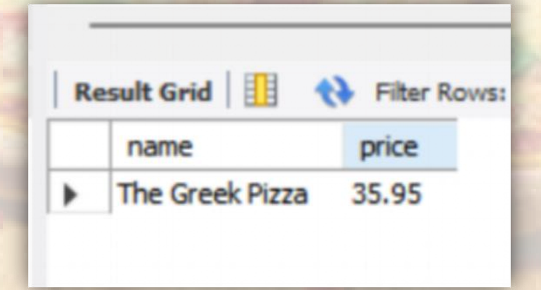


```
Total_Sales
817860.05
```

Q3. Identify the highest-priced pizza.

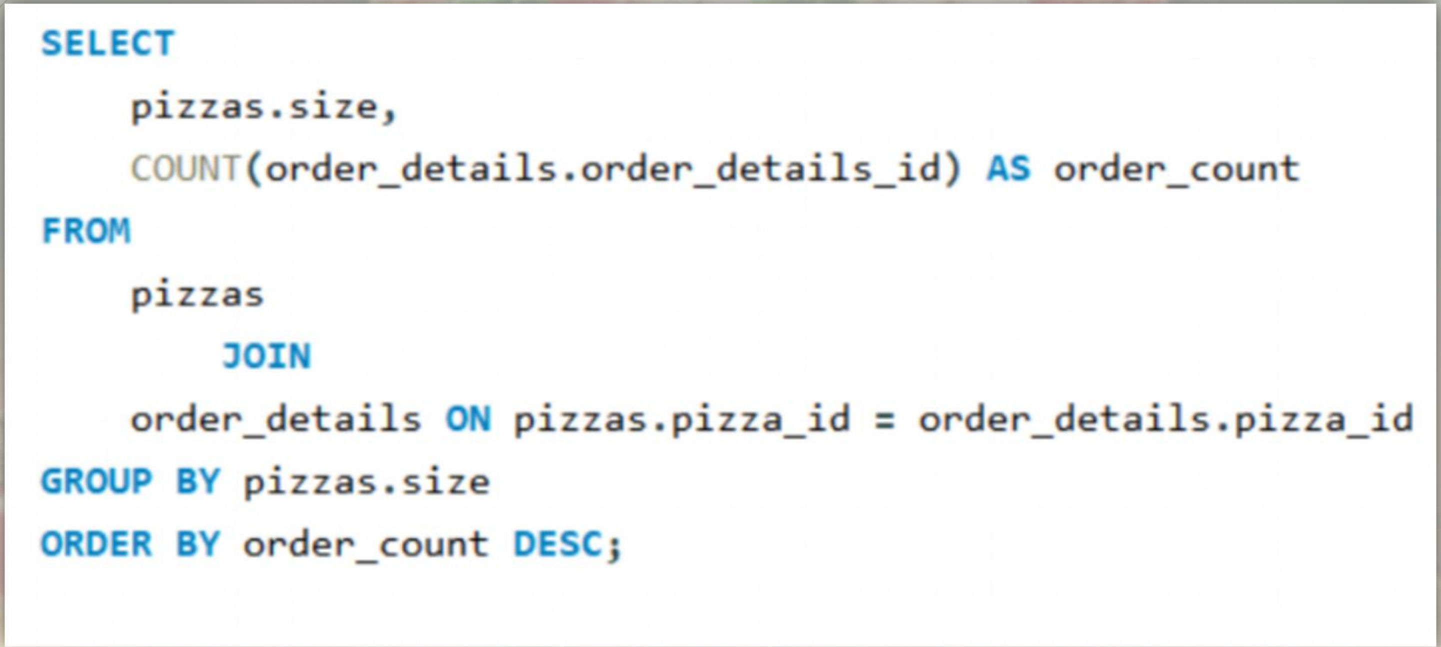


```
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

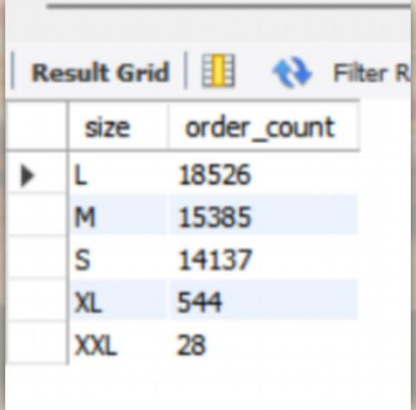


```
   name              price
▶  The Greek Pizza   35.95
```

Q4. Identify the most common pizza size ordered.

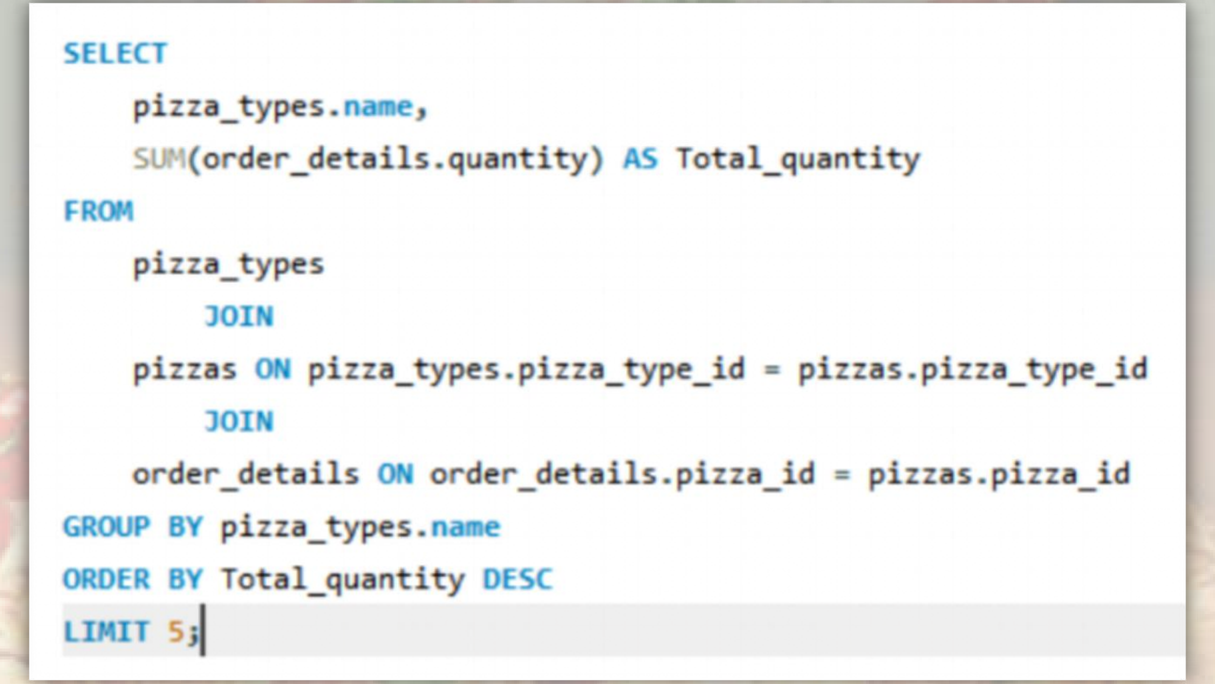


```sql
SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;
```

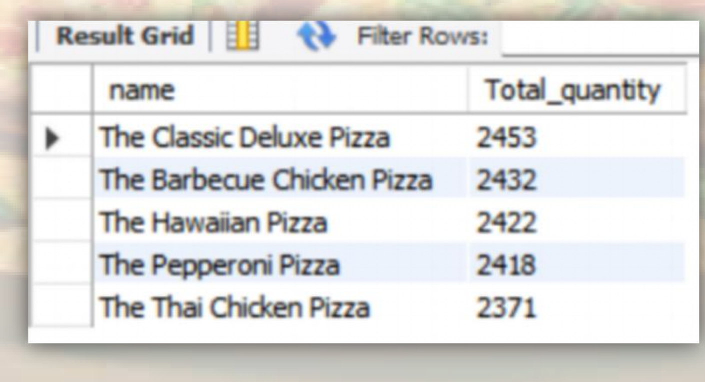


size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

Q5. List the top 5 most ordered pizza types along with their quantities.

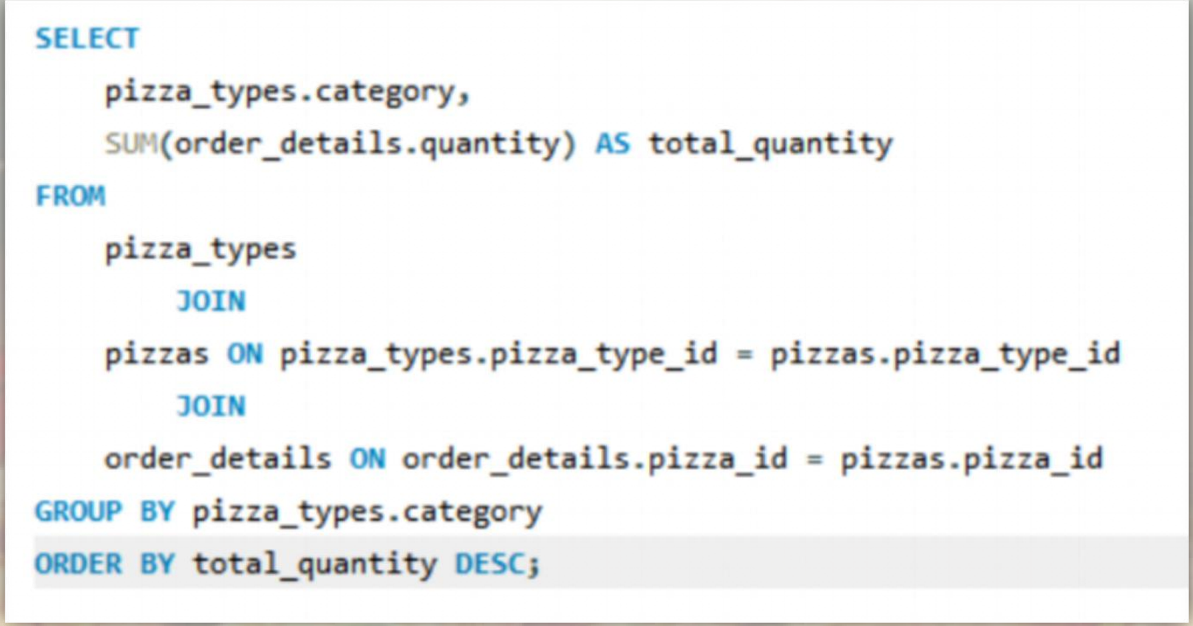


```sql
SELECT
    pizza_types.name,
    SUM(order_details.quantity) AS Total_quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY Total_quantity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

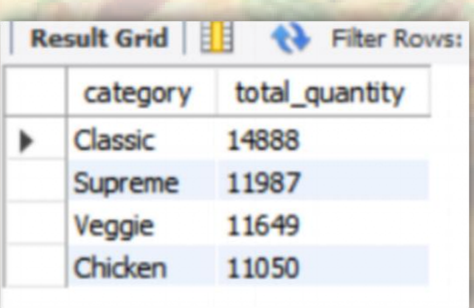


name	Total_quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

Q6. Join the necessary tables to find the total quantity of each pizza category ordered

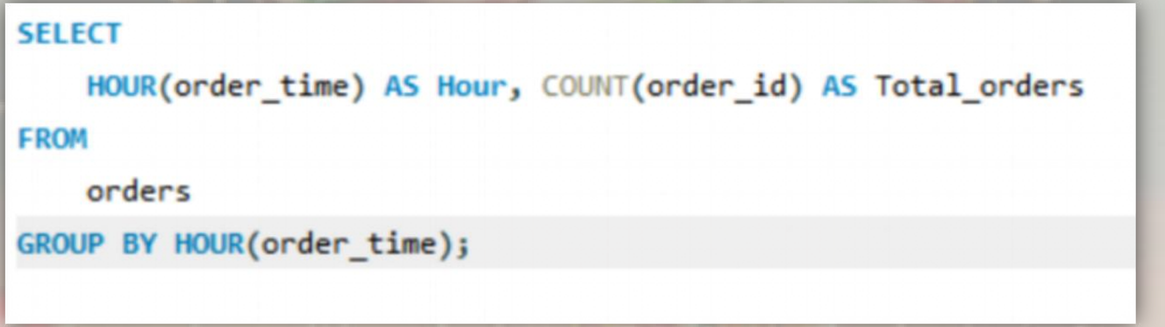


```
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS total_quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY total_quantity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

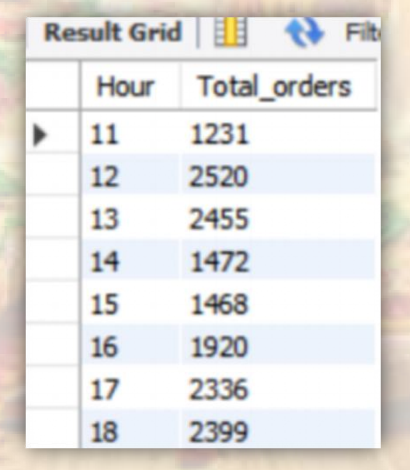


category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Q7. Determine the distribution of orders by hour of the day.

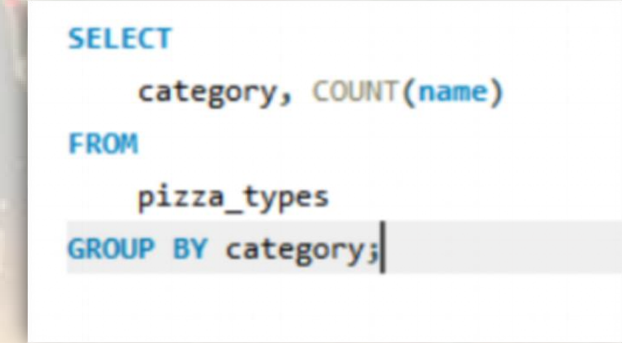


```
SELECT
    HOUR(order_time) AS Hour, COUNT(order_id) AS Total_orders
FROM
    orders
GROUP BY HOUR(order_time);
```

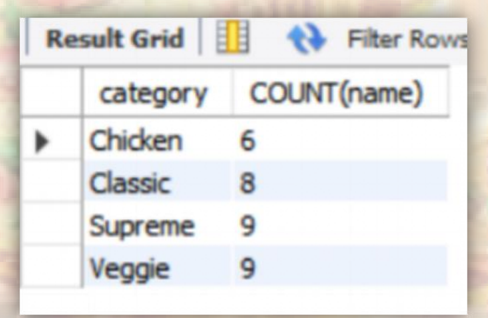


Hour	Total_orders
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399

Q8. Join relevant tables to find the category-wise distribution of pizzas.

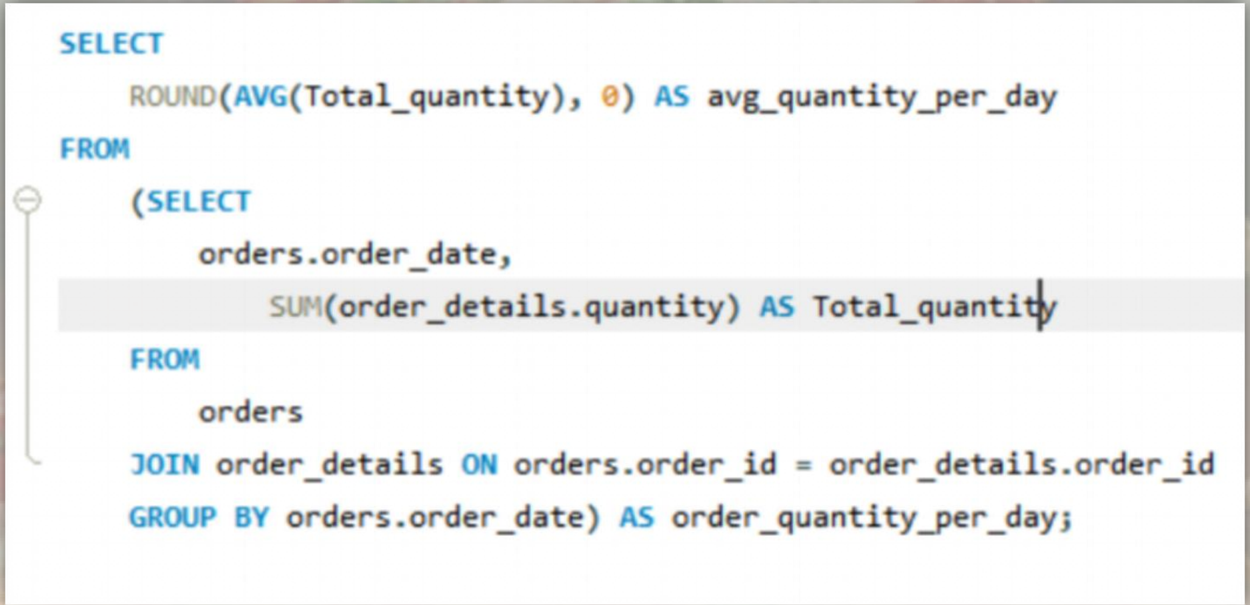


```
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

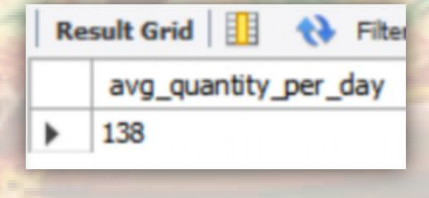


```
   category  COUNT(name)
▶  Chicken   6
   Classic   8
   Supreme   9
   Veggie    9
```

Q9. Group the orders by date and calculate the average number of pizzas ordered per day.

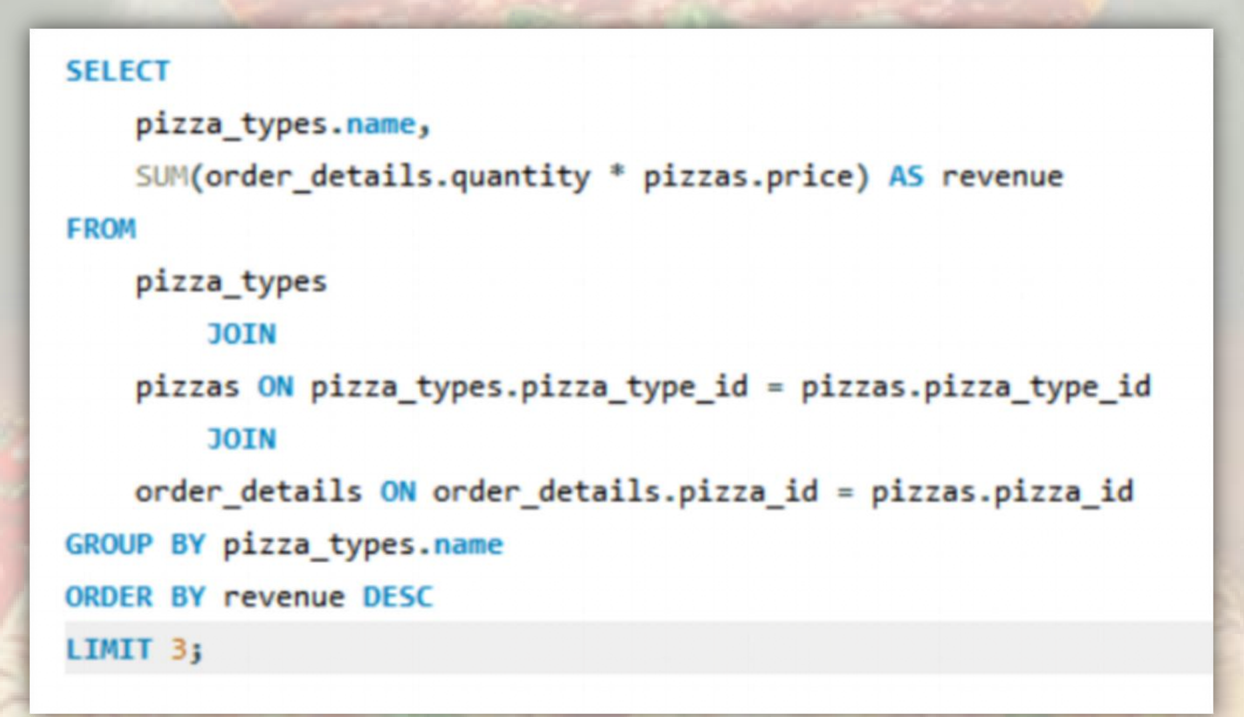


```
SELECT
    ROUND(AVG(Total_quantity), 0) AS avg_quantity_per_day
FROM
    (SELECT
        orders.order_date,
            SUM(order_details.quantity) AS Total_quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity_per_day;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

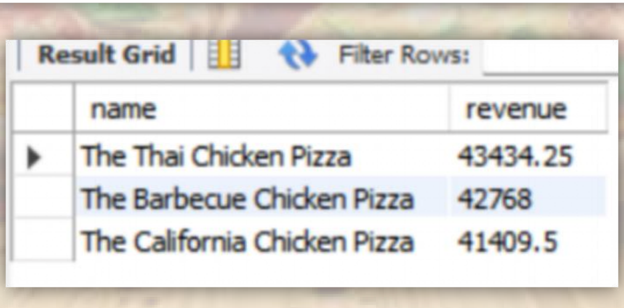


	avg_quantity_per_day
▶	138

Q10. Determine the top 3 most ordered pizza types based on revenue.

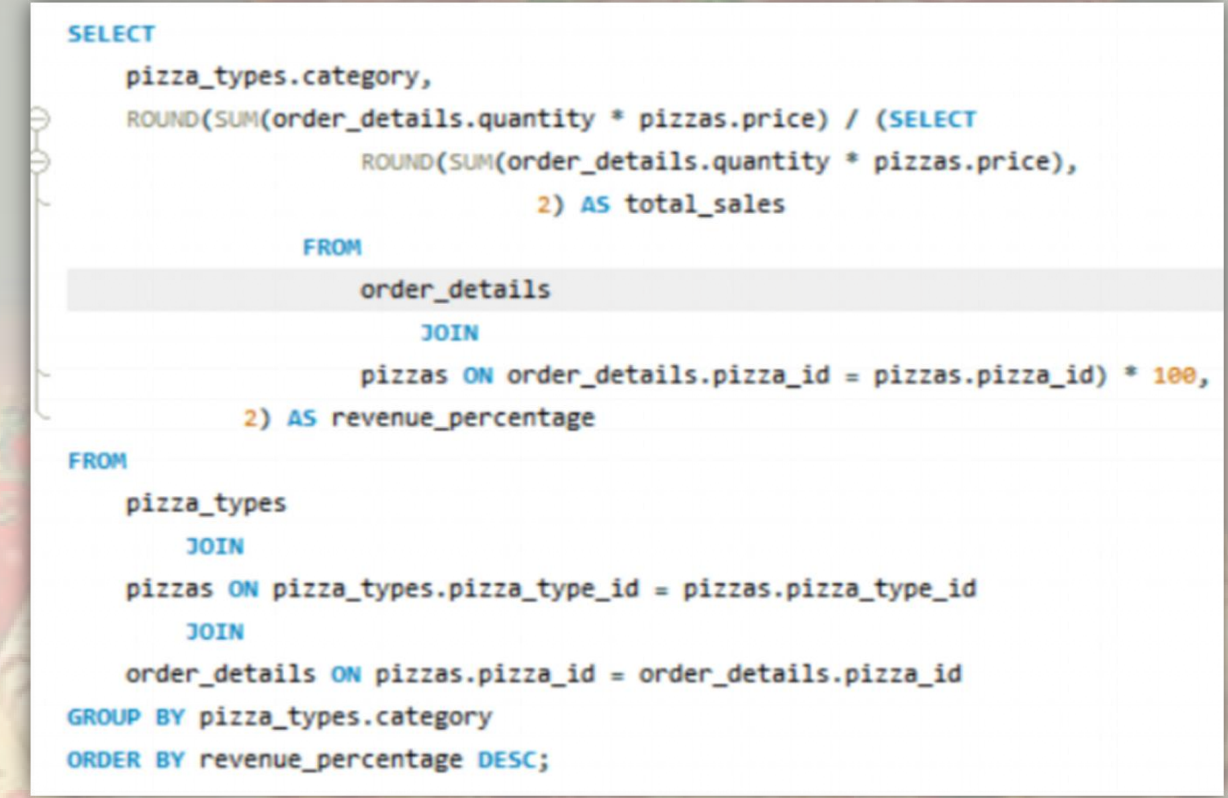


```
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue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

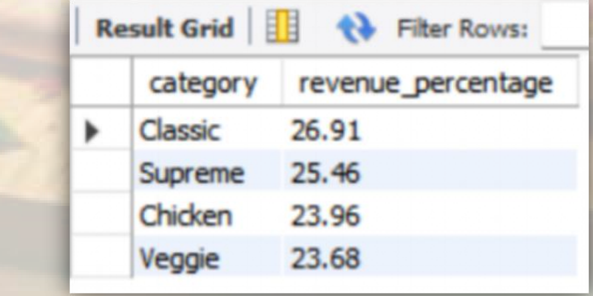


```
  name                           revenue
▶ The Thai Chicken Pizza         43434.25
  The Barbecue Chicken Pizza     42768
  The California Chicken Pizza   41409.5
```

Q11. Calculate the percentage contribution of each pizza type to total revenue.

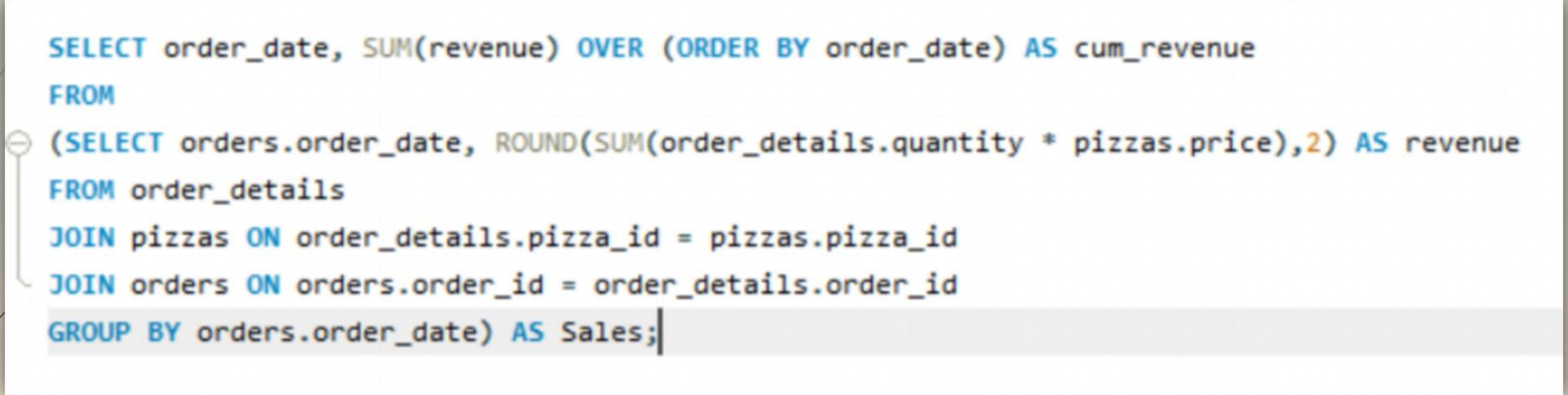


```
SELECT
    pizza_types.category,
    ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
                    ROUND(SUM(order_details.quantity * pizzas.price),
                                2) AS total_sales
                FROM
                    order_details
                        JOIN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

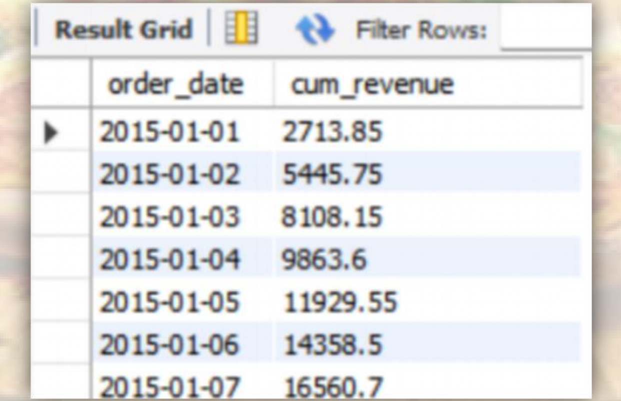


category	revenue_percentage
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Q12. Analyze the cumulative revenue generated over time.

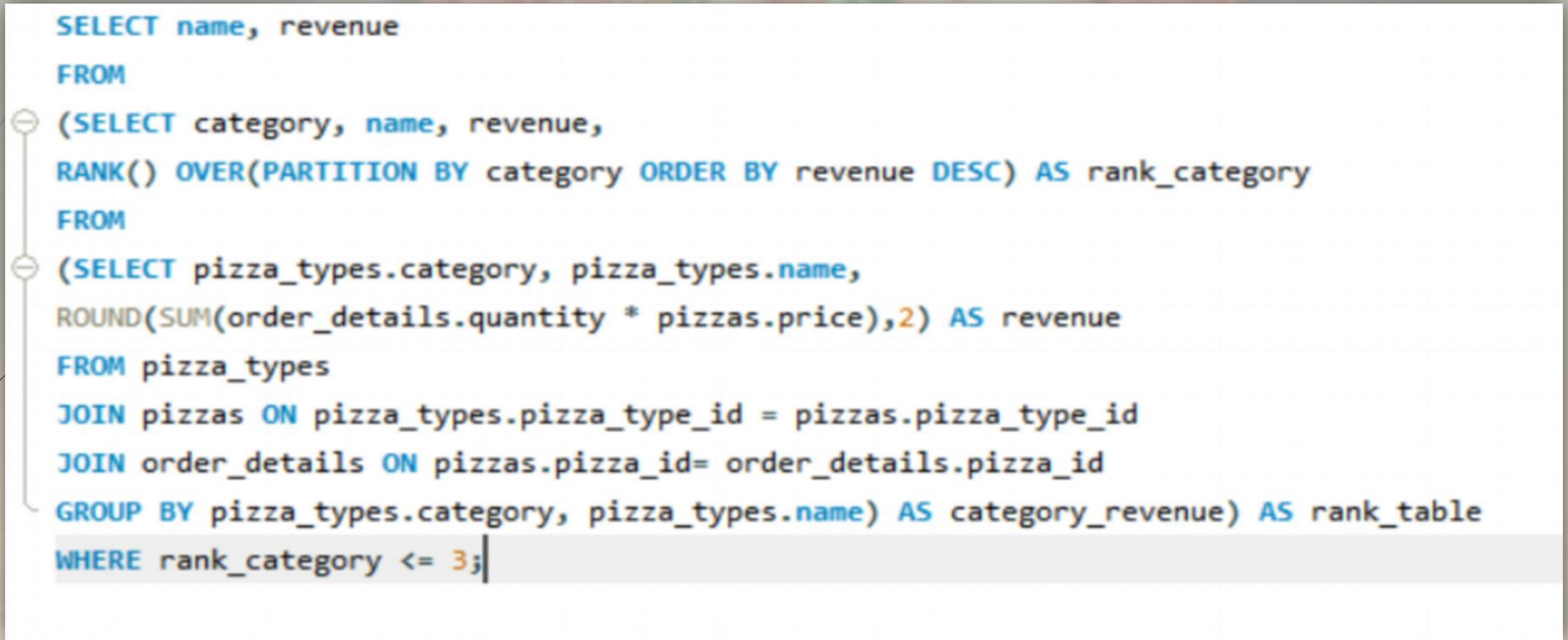


```
SELECT order_date, SUM(revenue) OVER (ORDER BY order_date) AS cum_revenue
FROM
(SELECT orders.order_date, ROUND(SUM(order_details.quantity * pizzas.price),2) AS revenue
FROM order_details
JOIN pizzas ON order_details.pizza_id = pizzas.pizza_id
JOIN orders ON orders.order_id = order_details.order_id
GROUP BY orders.order_date) AS Sales;
```

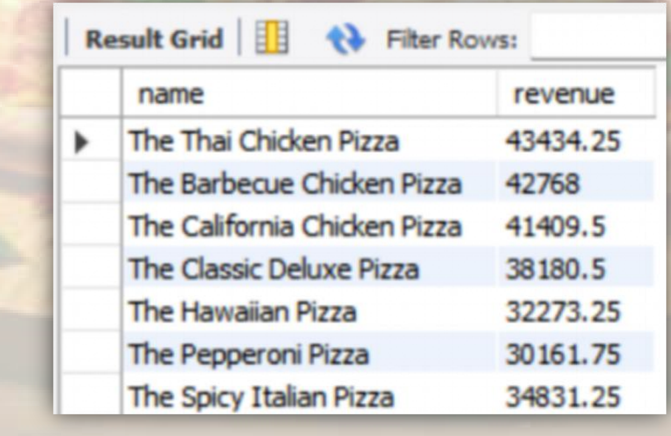


order_date	cum_revenue
2015-01-01	2713.85
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7

Q13. Determine the 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 rank_category
FROM
(SELECT pizza_types.category, pizza_types.name,
ROUND(SUM(order_details.quantity * pizzas.price),2) AS revenue
FROM pizza_types
JOIN pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN order_details ON pizzas.pizza_id= order_details.pizza_id
GROUP BY pizza_types.category, pizza_types.name) AS category_revenue) AS rank_table
WHERE rank_category <= 3;
```



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

THANK YOU!