# PizzaHut

## An SQL Project

TASTE

THE

BEST

FROM

THE

OVEN

SQL PROJECT

BY - PARAMJEET SAINI

Mrparamjeetsaini257@gmail.com

https://github.com/Paramjeet-Saini

## PROJECT INFORMATION

This project presents a comprehensive SQL- based analysis of PizzaHut's operations and sales data.

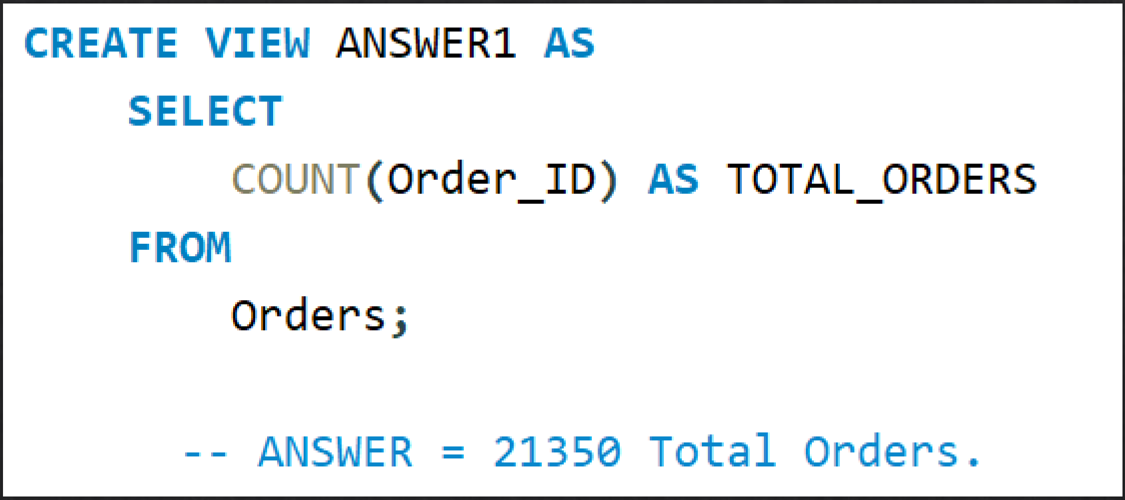
The objective is to explore various aspects of business ranging from order patterns to revenue trends and product sales. By writing and executing SQL queries on a structured database. Using a dataset modelled on real-world scenarios, this project answers multiple analytical questions to uncover insight that can help optimize decision making.

## Questions-

- 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.
- 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.
- 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.

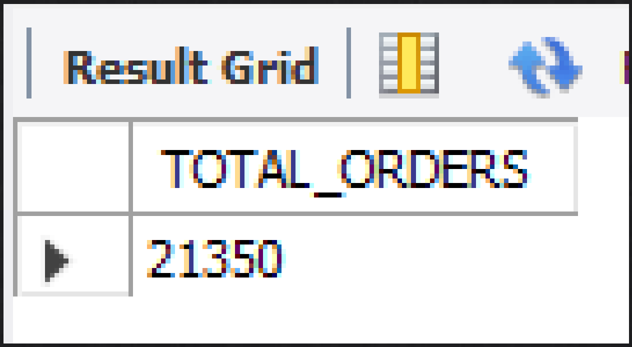
Answer -



```
CREATE VIEW ANSWER1 AS
    SELECT
        COUNT(Order_ID) AS TOTAL_ORDERS
    FROM
        Orders;

      -- ANSWER = 21350 Total Orders.
```

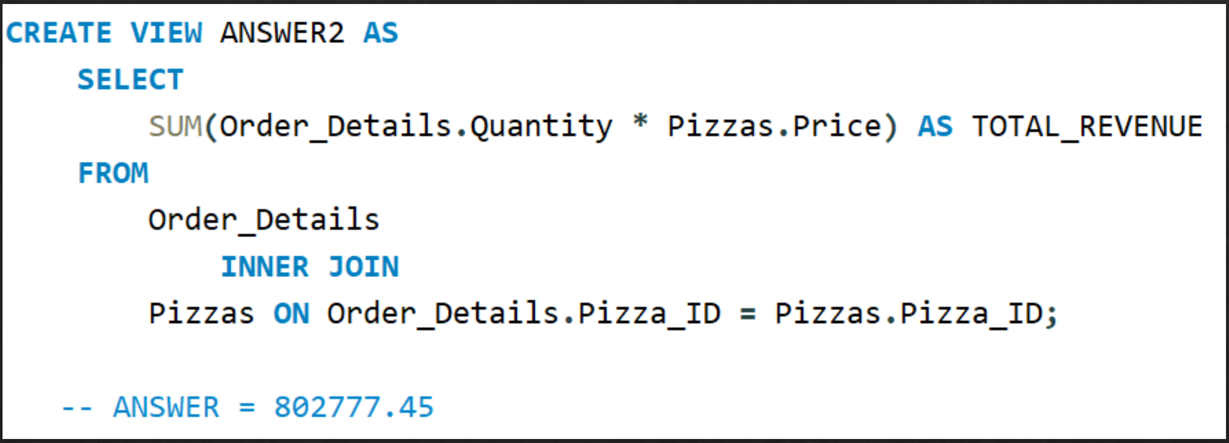
Result -



```
TOTAL_ORDERS
21350
```

## Q2. Calculate the total revenue generated from pizza sales.

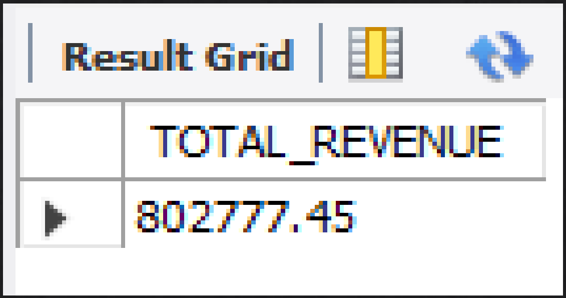
Answer -



```
CREATE VIEW ANSWER2 AS
    SELECT
        SUM(Order_Details.Quantity * Pizzas.Price) AS TOTAL_REVENUE
    FROM
        Order_Details
            INNER JOIN
        Pizzas ON Order_Details.Pizza_ID = Pizzas.Pizza_ID;

   -- ANSWER = 802777.45
```

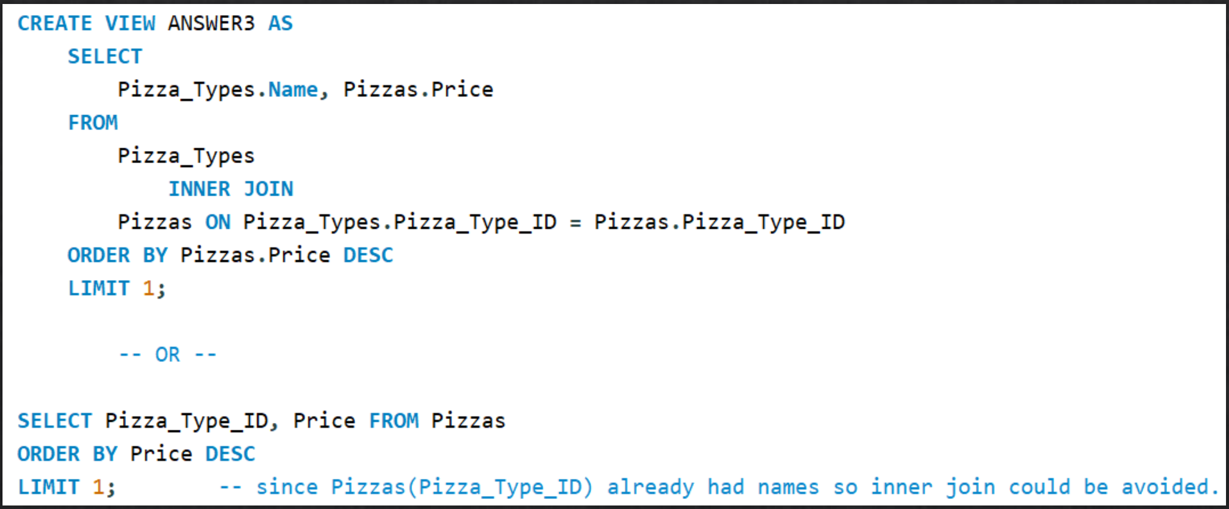
Result -



```
TOTAL_REVENUE
802777.45
```

# Q3. Identify the highest-priced pizza.

## Answers -

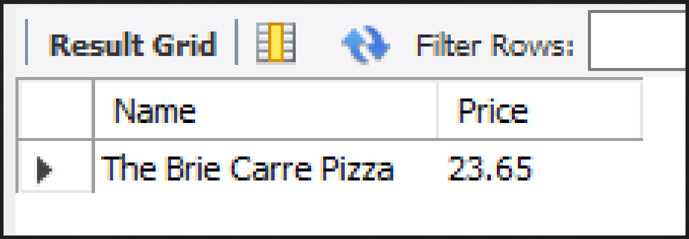


```sql
CREATE VIEW ANSWER3 AS
    SELECT
        Pizza_Types.Name, Pizzas.Price
    FROM
        Pizza_Types
            INNER JOIN
        Pizzas ON Pizza_Types.Pizza_Type_ID = Pizzas.Pizza_Type_ID
    ORDER BY Pizzas.Price DESC
    LIMIT 1;

        -- OR --

SELECT Pizza_Type_ID, Price FROM Pizzas
ORDER BY Price DESC
LIMIT 1;        -- since Pizzas(Pizza_Type_ID) already had names so inner join could be avoided.
```

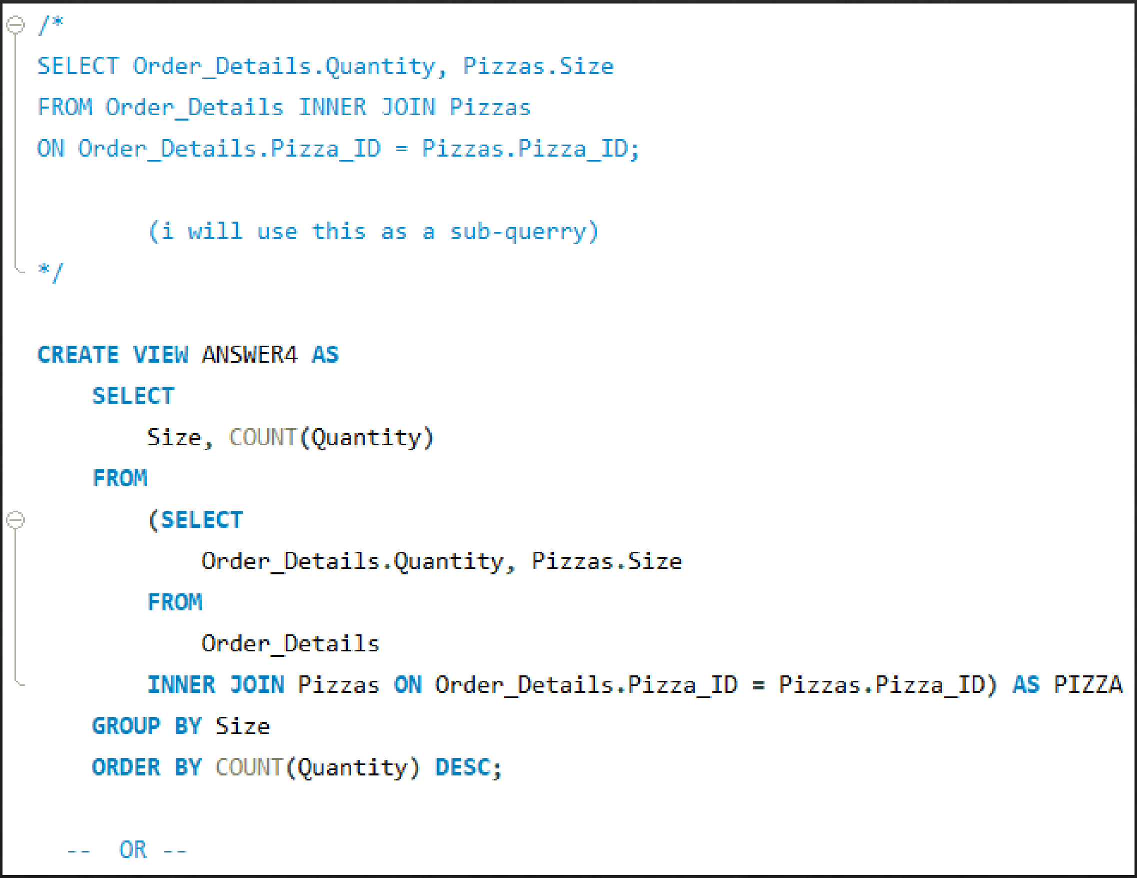
## Result-



| Name | Price |
|---|---|
| The Brie Carre Pizza | 23.65 |

## Q4. Identify the most common pizza size ordered.

Answer -

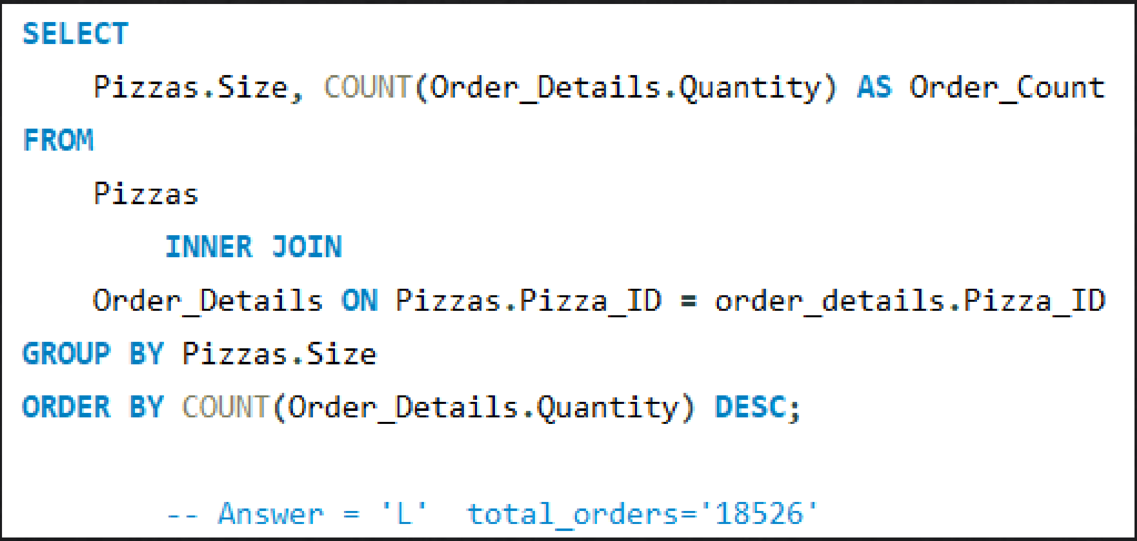


```
/*
SELECT Order_Details.Quantity, Pizzas.Size
FROM Order_Details INNER JOIN Pizzas
ON Order_Details.Pizza_ID = Pizzas.Pizza_ID;

        (i will use this as a sub-querry)
*/

CREATE VIEW ANSWER4 AS
    SELECT
        Size, COUNT(Quantity)
    FROM
        (SELECT
            Order_Details.Quantity, Pizzas.Size
        FROM
            Order_Details
        INNER JOIN Pizzas ON Order_Details.Pizza_ID = Pizzas.Pizza_ID) AS PIZZA
    GROUP BY Size
    ORDER BY COUNT(Quantity) DESC;

  --  OR --
```

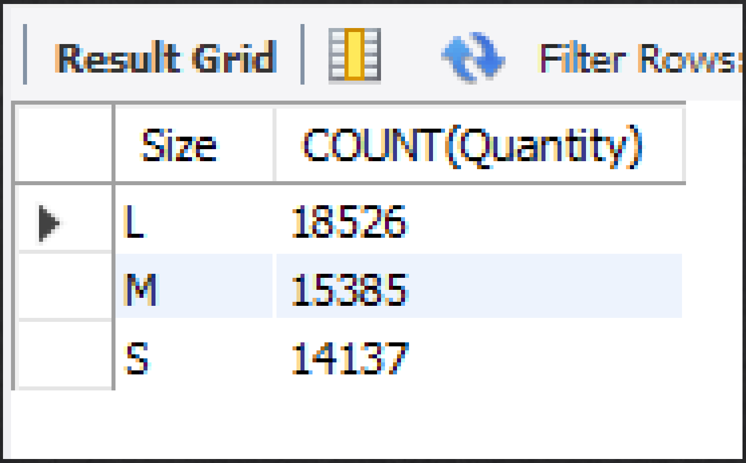


```
SELECT
    Pizzas.Size, COUNT(Order_Details.Quantity) AS Order_Count
FROM
    Pizzas
        INNER JOIN
    Order_Details ON Pizzas.Pizza_ID = order_details.Pizza_ID
GROUP BY Pizzas.Size
ORDER BY COUNT(Order_Details.Quantity) DESC;


        -- Answer = 'L'  total_orders='18526'
```

Continue on next

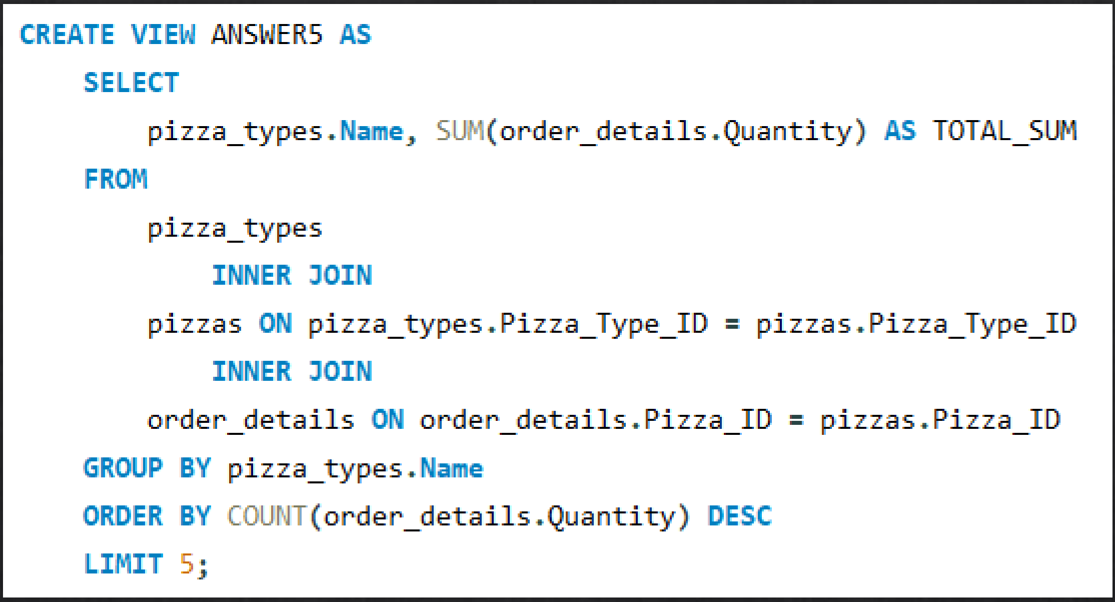
- Result -



| | Size | COUNT(Quantity) |
|---|---|---|
| ▶ | L | 18526 |
| | M | 15385 |
| | S | 14137 |

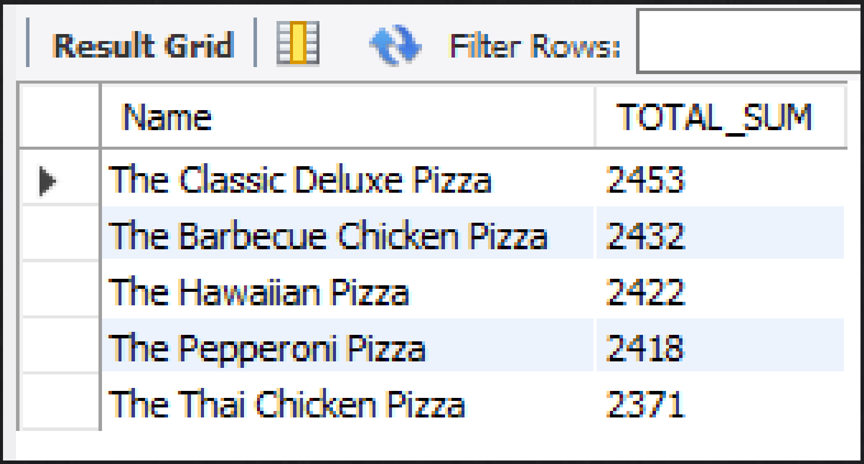
## Q5. List the top 5 most ordered pizza types along with their quantities.

Answer -



```
CREATE VIEW ANSWER5 AS
    SELECT
        pizza_types.Name, SUM(order_details.Quantity) AS TOTAL_SUM
    FROM
        pizza_types
            INNER JOIN
        pizzas ON pizza_types.Pizza_Type_ID = pizzas.Pizza_Type_ID
            INNER JOIN
        order_details ON order_details.Pizza_ID = pizzas.Pizza_ID
    GROUP BY pizza_types.Name
    ORDER BY COUNT(order_details.Quantity) DESC
    LIMIT 5;
```

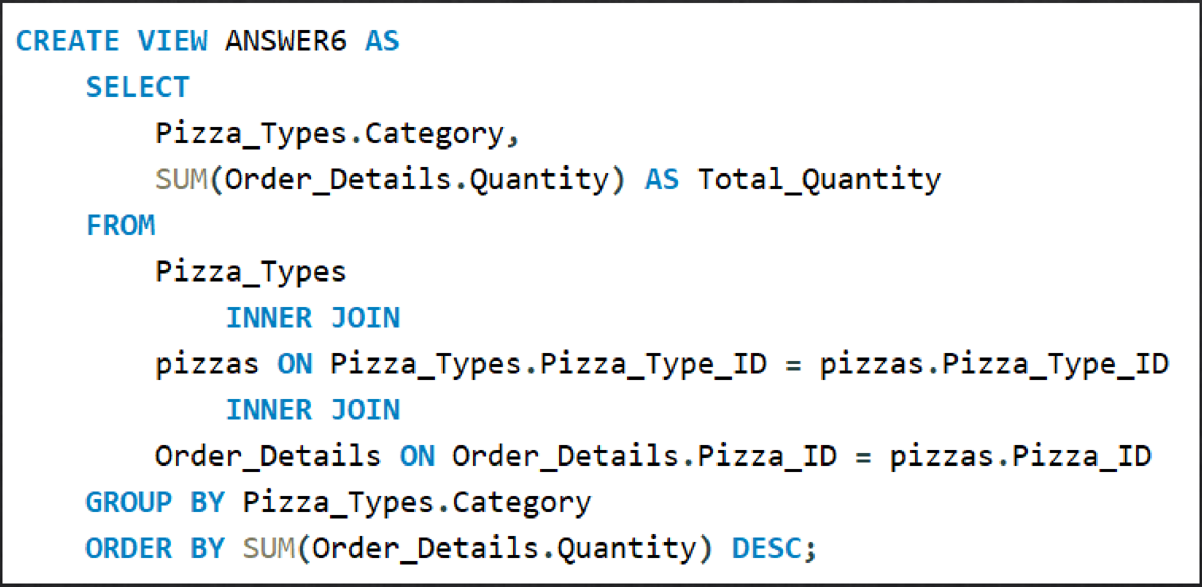
Result -



| Name | TOTAL_SUM |
|---|---|
| The Classic Deluxe Pizza | 2453 |
| The Barbecue Chicken Pizza | 2432 |
| The Hawaiian Pizza | 2422 |
| The Pepperoni Pizza | 2418 |
| The Thai Chicken Pizza | 2371 |

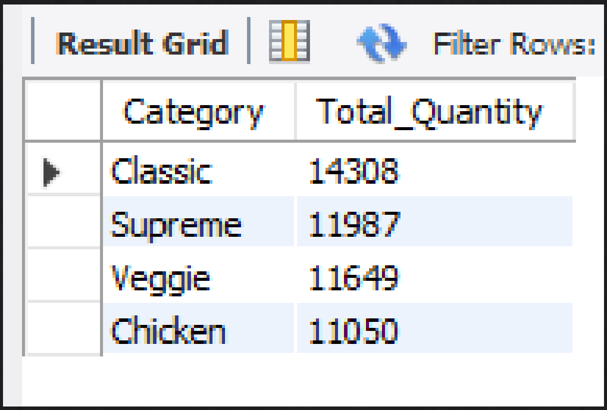
Q6. Join the necessary tables to find the total quantity of each pizza category ordered.

Answer -



```
CREATE VIEW ANSWER6 AS
    SELECT
        Pizza_Types.Category,
        SUM(Order_Details.Quantity) AS Total_Quantity
    FROM
        Pizza_Types
            INNER JOIN
        pizzas ON Pizza_Types.Pizza_Type_ID = pizzas.Pizza_Type_ID
            INNER JOIN
        Order_Details ON Order_Details.Pizza_ID = pizzas.Pizza_ID
    GROUP BY Pizza_Types.Category
    ORDER BY SUM(Order_Details.Quantity) DESC;
```

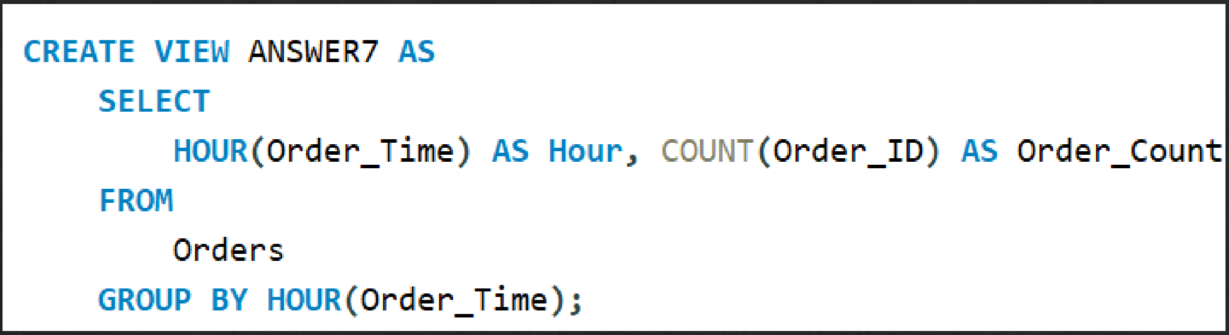
Result -



| Category | Total_Quantity |
|---|---|
| Classic | 14308 |
| Supreme | 11987 |
| Veggie | 11649 |
| Chicken | 11050 |

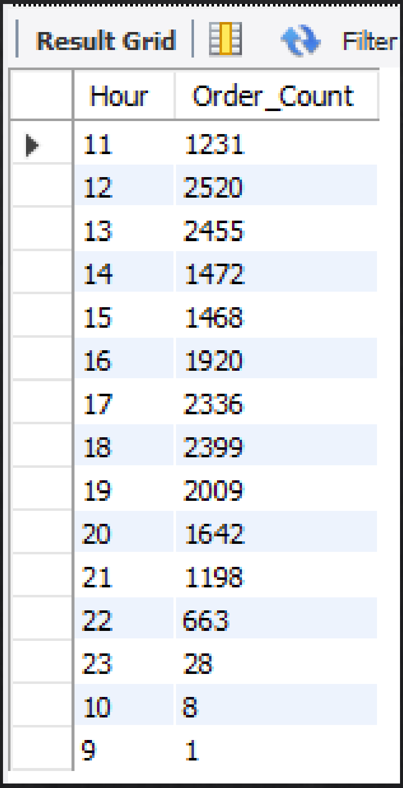
## Q7. Determine the distribution of orders by hour of the day.

Answer -



```
CREATE VIEW ANSWER7 AS
    SELECT
        HOUR(Order_Time) AS Hour, COUNT(Order_ID) AS Order_Count
    FROM
        Orders
    GROUP BY HOUR(Order_Time);
```

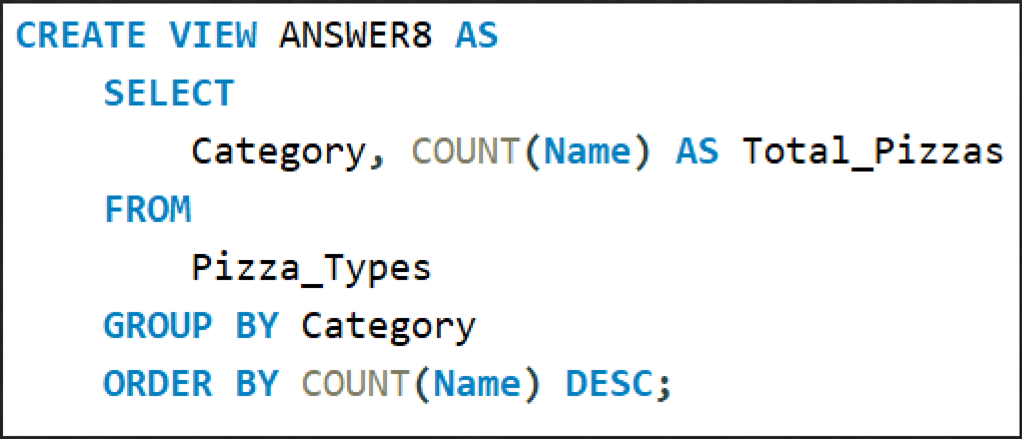
Result -



| 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 |

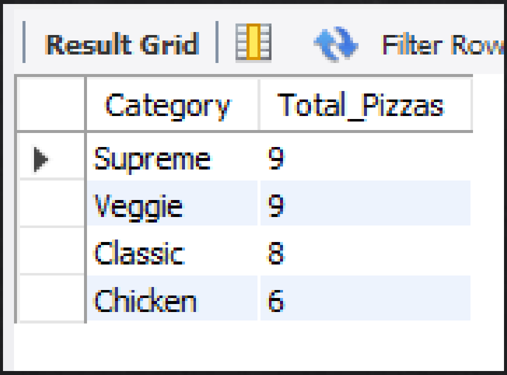
## Q8. Join relevant tables to find the category-wise distribution of pizzas.

Answer -



```
CREATE VIEW ANSWER8 AS
    SELECT
        Category, COUNT(Name) AS Total_Pizzas
    FROM
        Pizza_Types
    GROUP BY Category
    ORDER BY COUNT(Name) DESC;
```

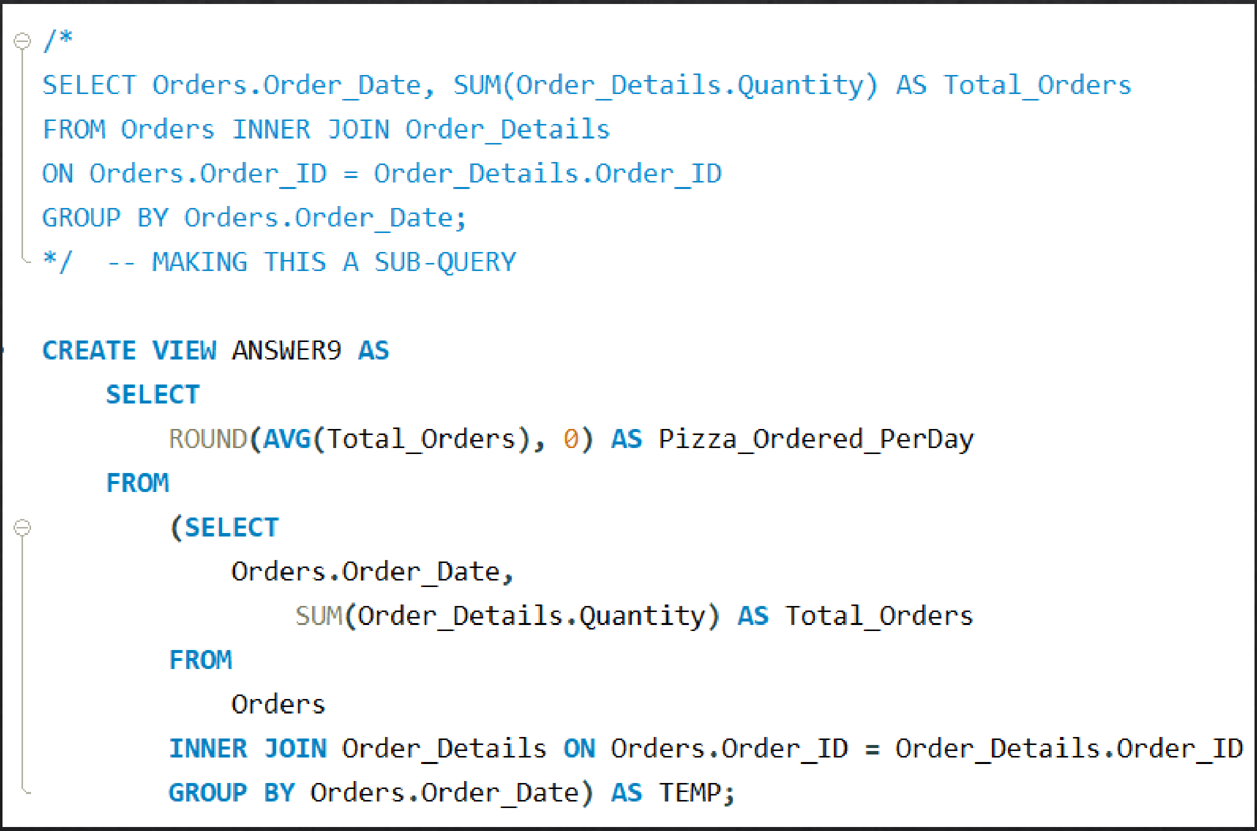
Result -



| Category | Total_Pizzas |
|---|---|
| Supreme | 9 |
| Veggie | 9 |
| Classic | 8 |
| Chicken | 6 |

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

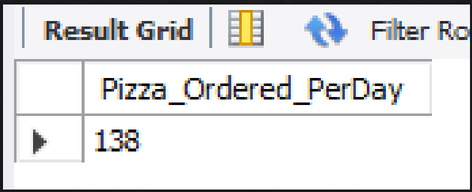
Answer -



```
/*
SELECT Orders.Order_Date, SUM(Order_Details.Quantity) AS Total_Orders
FROM Orders INNER JOIN Order_Details
ON Orders.Order_ID = Order_Details.Order_ID
GROUP BY Orders.Order_Date;
*/  -- MAKING THIS A SUB-QUERY

CREATE VIEW ANSWER9 AS
    SELECT
        ROUND(AVG(Total_Orders), 0) AS Pizza_Ordered_PerDay
    FROM
        (SELECT
            Orders.Order_Date,
                SUM(Order_Details.Quantity) AS Total_Orders
        FROM
            Orders
        INNER JOIN Order_Details ON Orders.Order_ID = Order_Details.Order_ID
        GROUP BY Orders.Order_Date) AS TEMP;
```

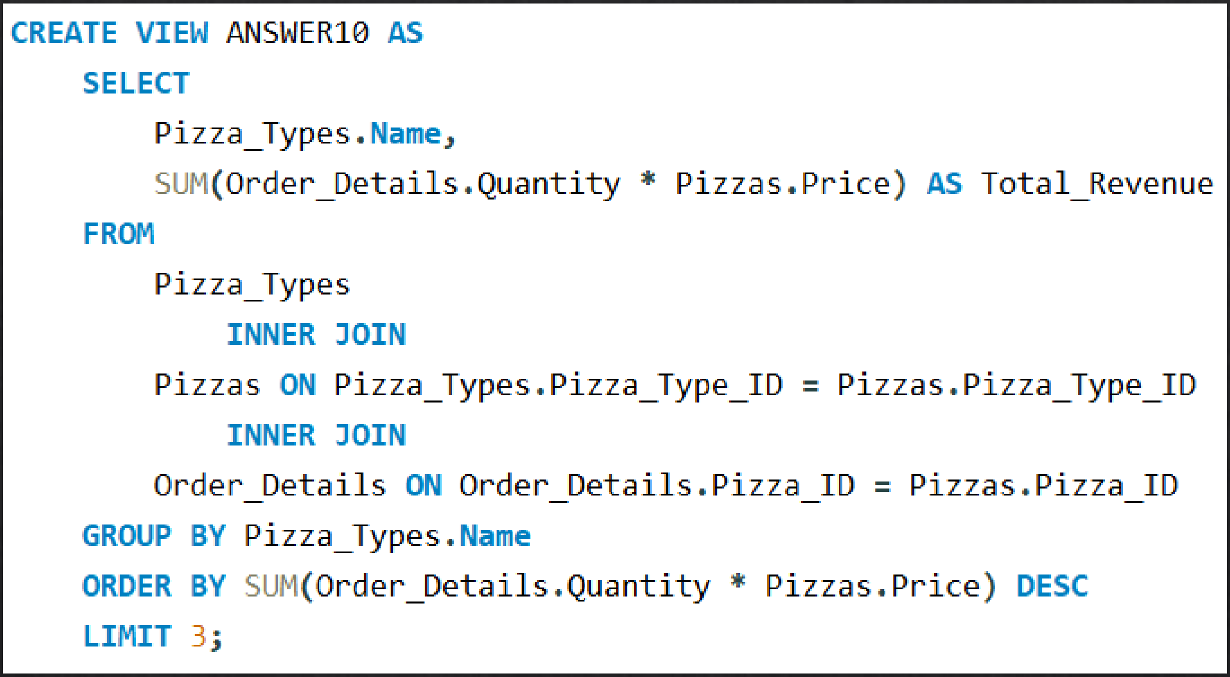
Result -



```
Pizza_Ordered_PerDay
138
```

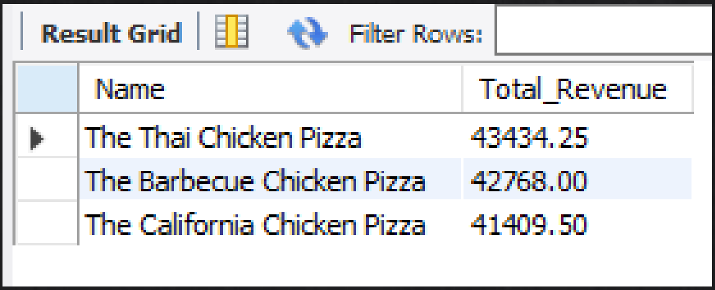
## Q10. Determine the top 3 most ordered pizza types based on revenue.

Answer -



```
CREATE VIEW ANSWER10 AS
    SELECT
        Pizza_Types.Name,
        SUM(Order_Details.Quantity * Pizzas.Price) AS Total_Revenue
    FROM
        Pizza_Types
            INNER JOIN
        Pizzas ON Pizza_Types.Pizza_Type_ID = Pizzas.Pizza_Type_ID
            INNER JOIN
        Order_Details ON Order_Details.Pizza_ID = Pizzas.Pizza_ID
    GROUP BY Pizza_Types.Name
    ORDER BY SUM(Order_Details.Quantity * Pizzas.Price) DESC
    LIMIT 3;
```

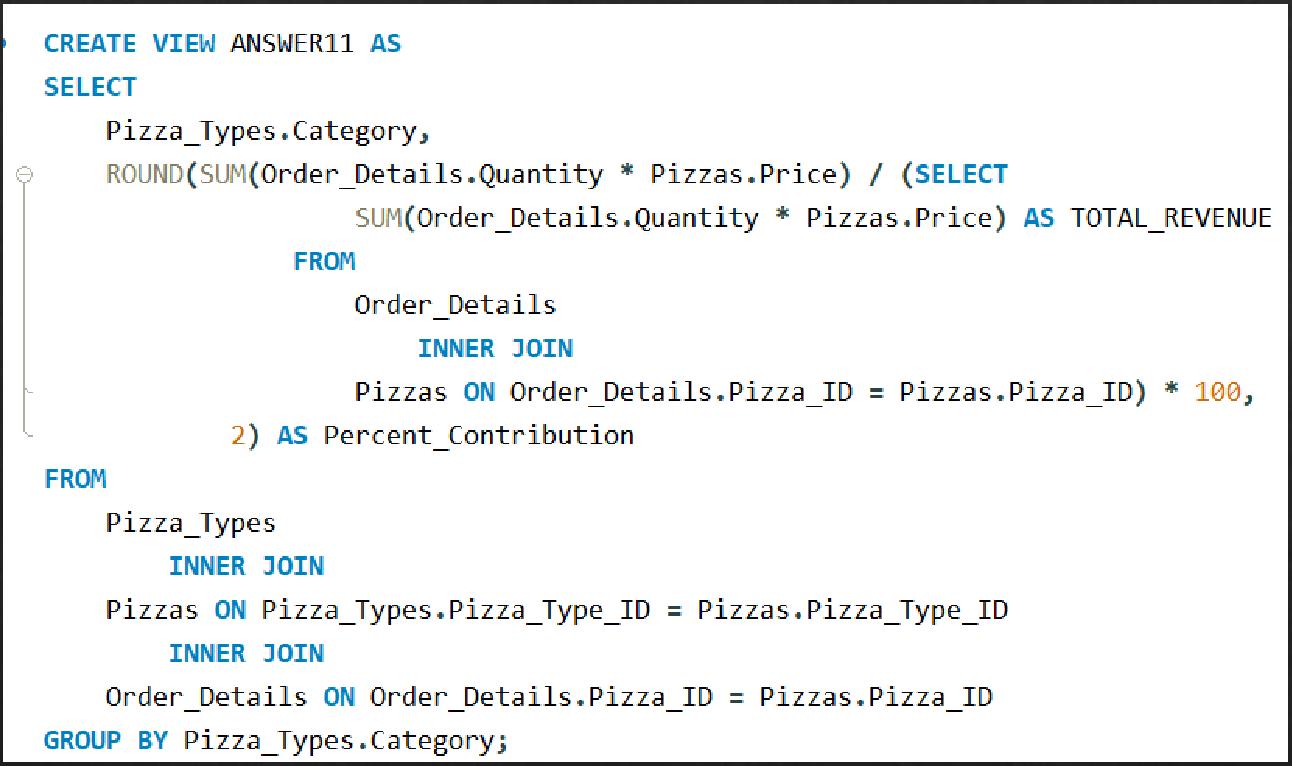
Result -



| Name | Total_Revenue |
|---|---|
| The Thai Chicken Pizza | 43434.25 |
| The Barbecue Chicken Pizza | 42768.00 |
| The California Chicken Pizza | 41409.50 |

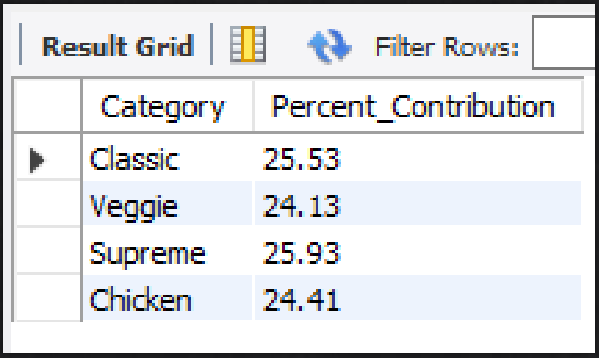
## Q11. Calculate the percentage contribution of each pizza type to total revenue.

Answer -



```
CREATE VIEW ANSWER11 AS
SELECT
    Pizza_Types.Category,
    ROUND(SUM(Order_Details.Quantity * Pizzas.Price) / (SELECT
                SUM(Order_Details.Quantity * Pizzas.Price) AS TOTAL_REVENUE
            FROM
                Order_Details
                    INNER JOIN
                Pizzas ON Order_Details.Pizza_ID = Pizzas.Pizza_ID) * 100,
            2) AS Percent_Contribution
FROM
    Pizza_Types
        INNER JOIN
    Pizzas ON Pizza_Types.Pizza_Type_ID = Pizzas.Pizza_Type_ID
        INNER JOIN
    Order_Details ON Order_Details.Pizza_ID = Pizzas.Pizza_ID
GROUP BY Pizza_Types.Category;
```

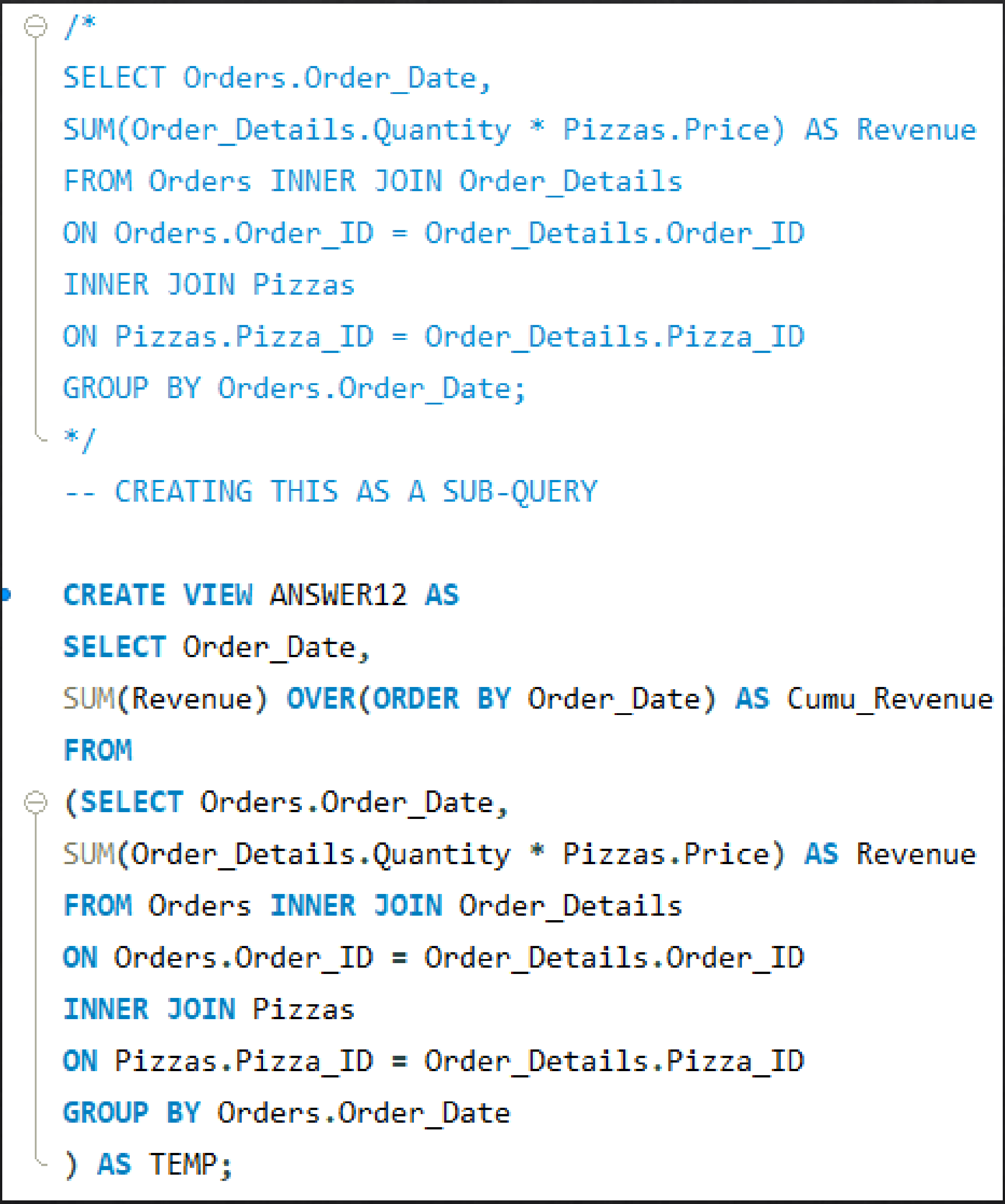
Result -



| Category | Percent_Contribution |
|---|---|
| Classic | 25.53 |
| Veggie | 24.13 |
| Supreme | 25.93 |
| Chicken | 24.41 |

## Q12. Analyze the cumulative revenue generated over time.

Answer -

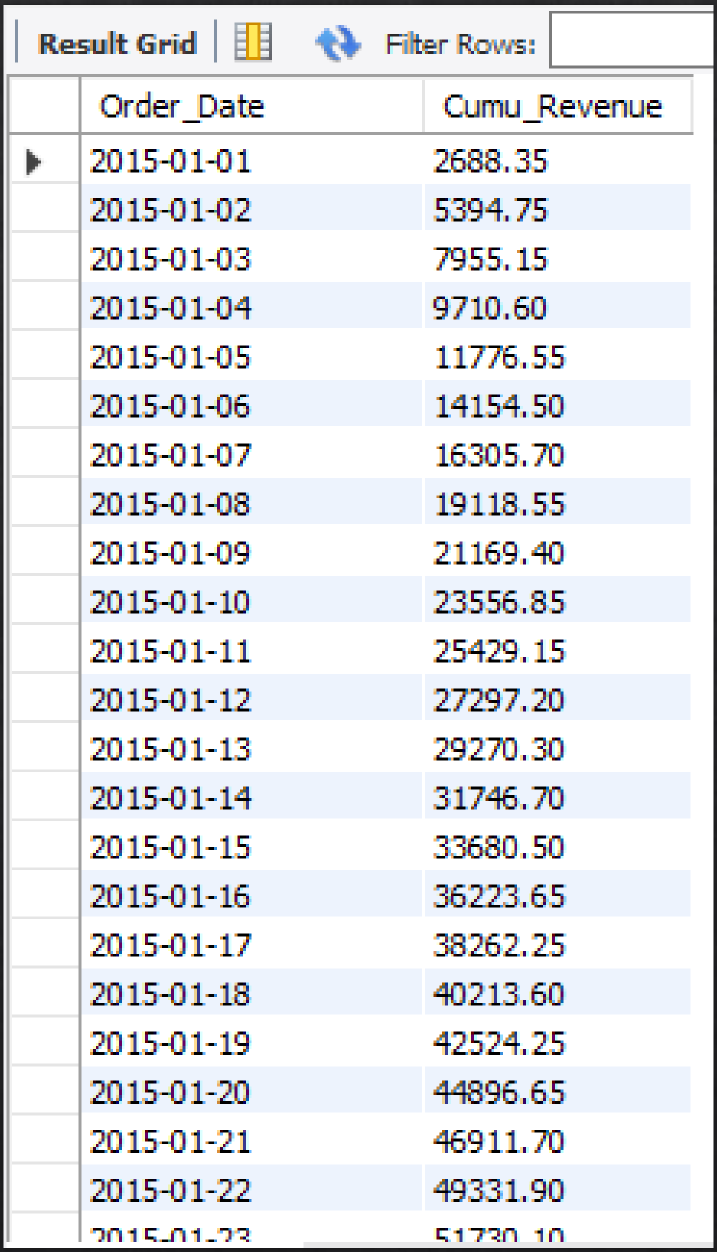


```sql
/*
SELECT Orders.Order_Date,
SUM(Order_Details.Quantity * Pizzas.Price) AS Revenue
FROM Orders INNER JOIN Order_Details
ON Orders.Order_ID = Order_Details.Order_ID
INNER JOIN Pizzas
ON Pizzas.Pizza_ID = Order_Details.Pizza_ID
GROUP BY Orders.Order_Date;
*/
-- CREATING THIS AS A SUB-QUERY

CREATE VIEW ANSWER12 AS
SELECT Order_Date,
SUM(Revenue) OVER(ORDER BY Order_Date) AS Cumu_Revenue
FROM
(SELECT Orders.Order_Date,
SUM(Order_Details.Quantity * Pizzas.Price) AS Revenue
FROM Orders INNER JOIN Order_Details
ON Orders.Order_ID = Order_Details.Order_ID
INNER JOIN Pizzas
ON Pizzas.Pizza_ID = Order_Details.Pizza_ID
GROUP BY Orders.Order_Date
) AS TEMP;
```

Continue on next

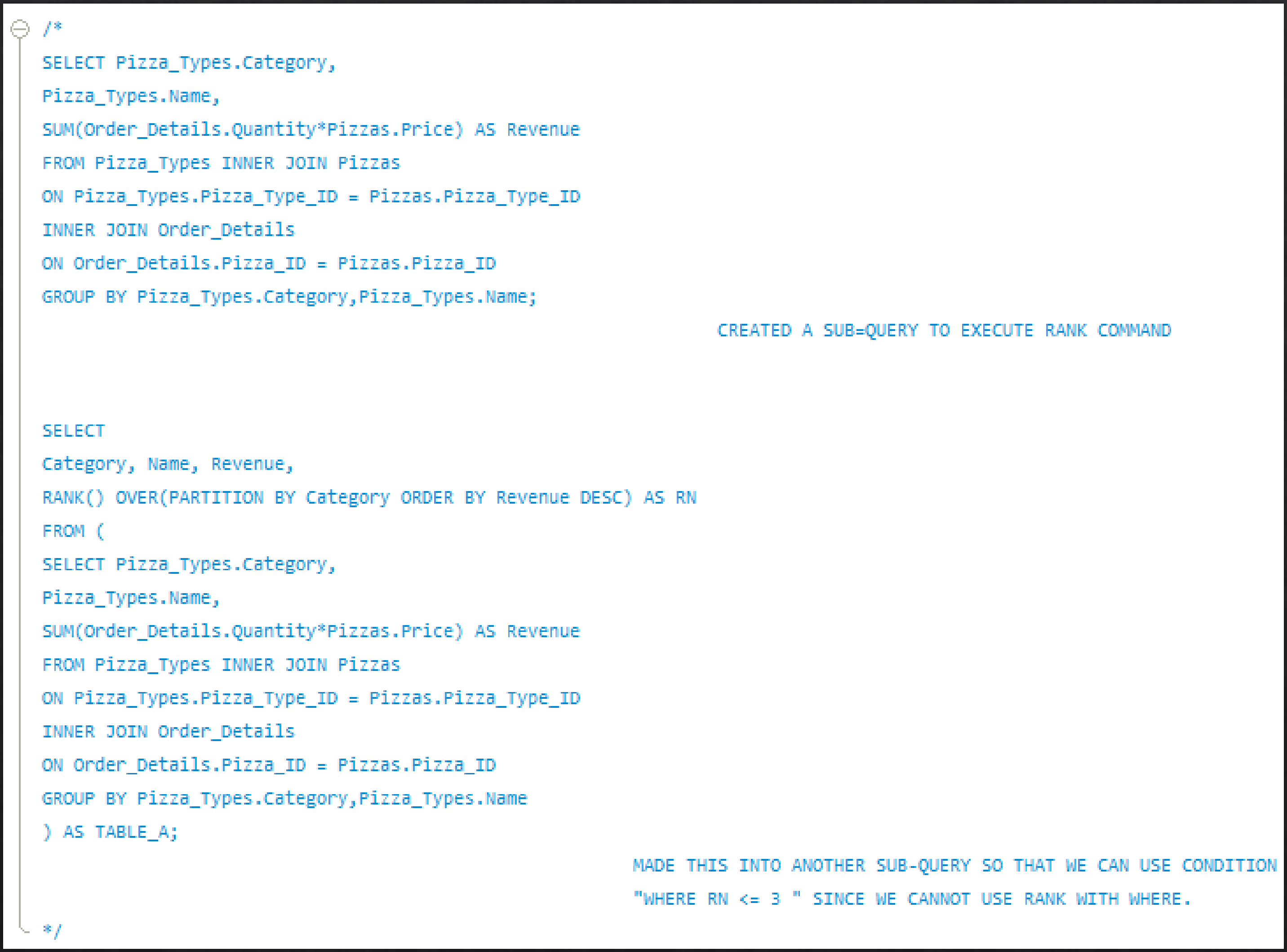
Result -



| Order_Date | Cumu_Revenue |
|---|---|
| 2015-01-01 | 2688.35 |
| 2015-01-02 | 5394.75 |
| 2015-01-03 | 7955.15 |
| 2015-01-04 | 9710.60 |
| 2015-01-05 | 11776.55 |
| 2015-01-06 | 14154.50 |
| 2015-01-07 | 16305.70 |
| 2015-01-08 | 19118.55 |
| 2015-01-09 | 21169.40 |
| 2015-01-10 | 23556.85 |
| 2015-01-11 | 25429.15 |
| 2015-01-12 | 27297.20 |
| 2015-01-13 | 29270.30 |
| 2015-01-14 | 31746.70 |
| 2015-01-15 | 33680.50 |
| 2015-01-16 | 36223.65 |
| 2015-01-17 | 38262.25 |
| 2015-01-18 | 40213.60 |
| 2015-01-19 | 42524.25 |
| 2015-01-20 | 44896.65 |
| 2015-01-21 | 46911.70 |
| 2015-01-22 | 49331.90 |
| 2015-01-23 | 51730.10 |

## Q13. Determine the top 3 most ordered pizza types based on revenue for each pizza category.

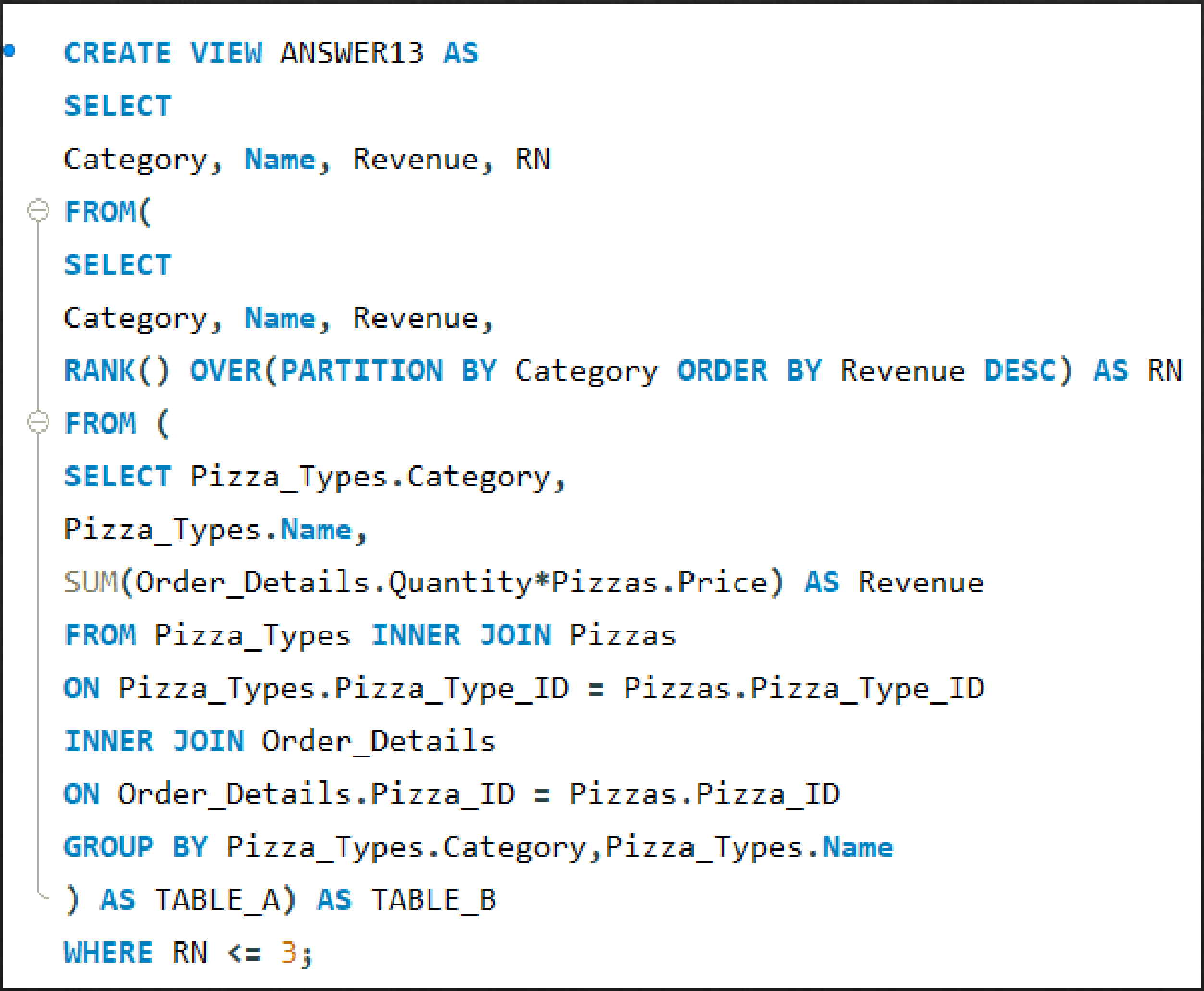
Answer -



```
/*
SELECT Pizza_Types.Category,
Pizza_Types.Name,
SUM(Order_Details.Quantity*Pizzas.Price) AS Revenue
FROM Pizza_Types INNER JOIN Pizzas
ON Pizza_Types.Pizza_Type_ID = Pizzas.Pizza_Type_ID
INNER JOIN Order_Details
ON Order_Details.Pizza_ID = Pizzas.Pizza_ID
GROUP BY Pizza_Types.Category,Pizza_Types.Name;
                                                                CREATED A SUB=QUERY TO EXECUTE RANK COMMAND


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 INNER JOIN Pizzas
ON Pizza_Types.Pizza_Type_ID = Pizzas.Pizza_Type_ID
INNER JOIN Order_Details
ON Order_Details.Pizza_ID = Pizzas.Pizza_ID
GROUP BY Pizza_Types.Category,Pizza_Types.Name
) AS TABLE_A;
                                                        MADE THIS INTO ANOTHER SUB-QUERY SO THAT WE CAN USE CONDITION
                                                        "WHERE RN <= 3 " SINCE WE CANNOT USE RANK WITH WHERE.
*/
```

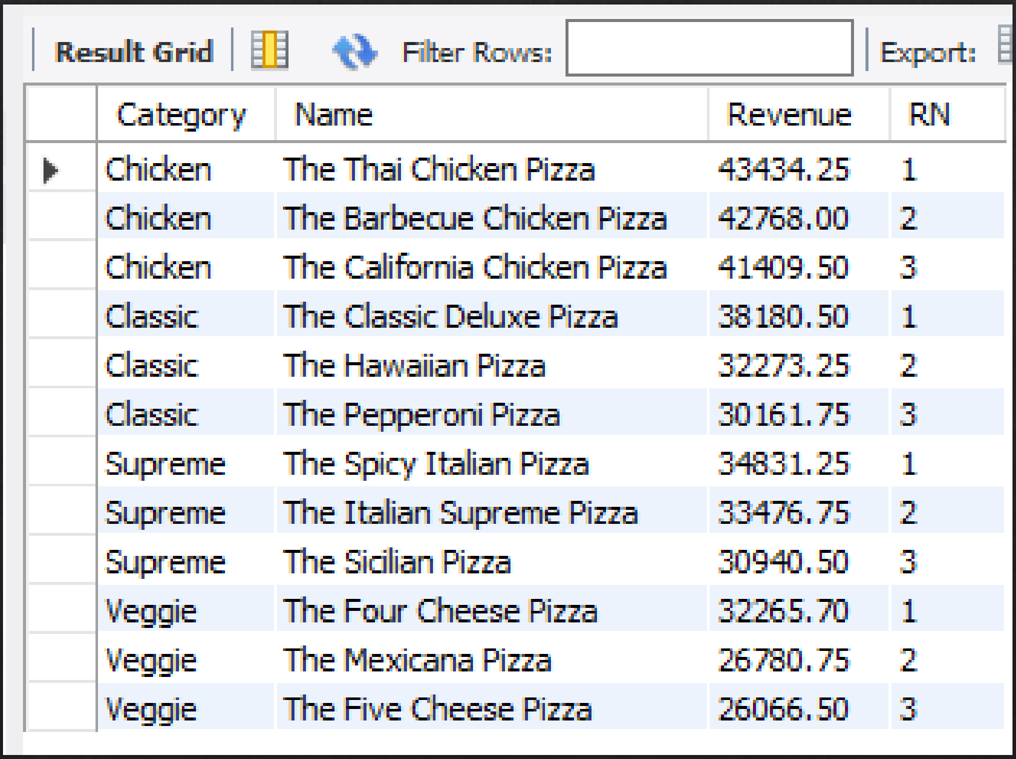
Continue on next ⇨



```
CREATE VIEW ANSWER13 AS
SELECT
Category, Name, Revenue, RN
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 INNER JOIN Pizzas
ON Pizza_Types.Pizza_Type_ID = Pizzas.Pizza_Type_ID
INNER JOIN Order_Details
ON Order_Details.Pizza_ID = Pizzas.Pizza_ID
GROUP BY Pizza_Types.Category,Pizza_Types.Name
) AS TABLE_A) AS TABLE_B
WHERE RN <= 3;
```

Continue on next ⇨

Result -



| | Category | Name | Revenue | RN |
|---|---|---|---|---|
| ▶ | Chicken | The Thai Chicken Pizza | 43434.25 | 1 |
| | Chicken | The Barbecue Chicken Pizza | 42768.00 | 2 |
| | Chicken | The California Chicken Pizza | 41409.50 | 3 |
| | Classic | The Classic Deluxe Pizza | 38180.50 | 1 |
| | Classic | The Hawaiian Pizza | 32273.25 | 2 |
| | Classic | The Pepperoni Pizza | 30161.75 | 3 |
| | Supreme | The Spicy Italian Pizza | 34831.25 | 1 |
| | Supreme | The Italian Supreme Pizza | 33476.75 | 2 |
| | Supreme | The Sicilian Pizza | 30940.50 | 3 |
| | Veggie | The Four Cheese Pizza | 32265.70 | 1 |
| | Veggie | The Mexicana Pizza | 26780.75 | 2 |
| | Veggie | The Five Cheese Pizza | 26066.50 | 3 |