MySQL PROJECT

PIZZA SALES ANALYSIS

BY : PRIOM PATHAK

HELLO!

"I am **Priom Pathak** ,a data analyst. In my role, I've leveraged SQL queries to analyze **Pizza Sales** data. With **MySQL** as my primary tool, I've addressed various sales-related questions, showcasing my expertise in data analysis and MySQL proficiency."

RELATED 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.

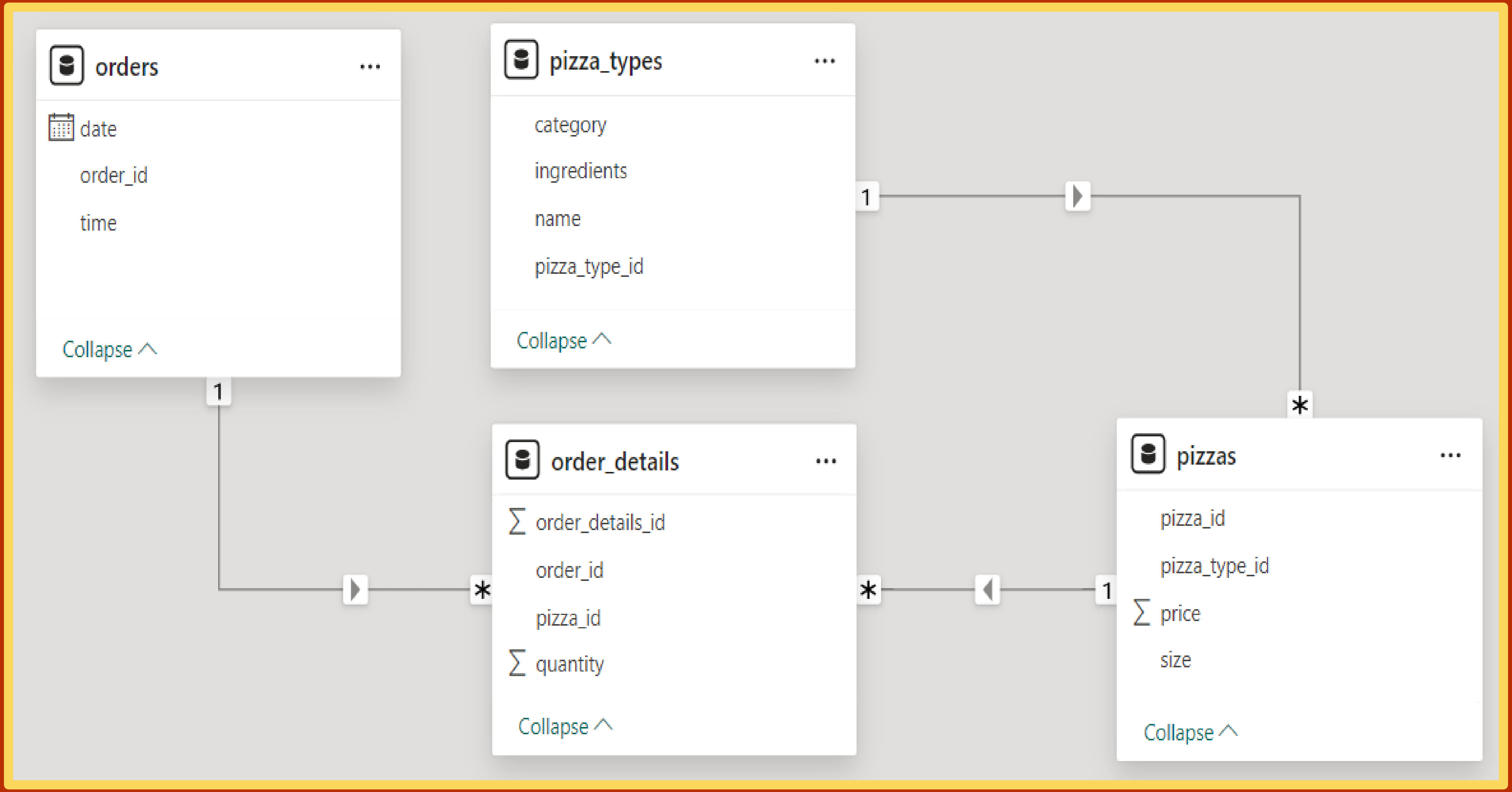
10.Determine the top 3 most ordered pizza types based on revenue.

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

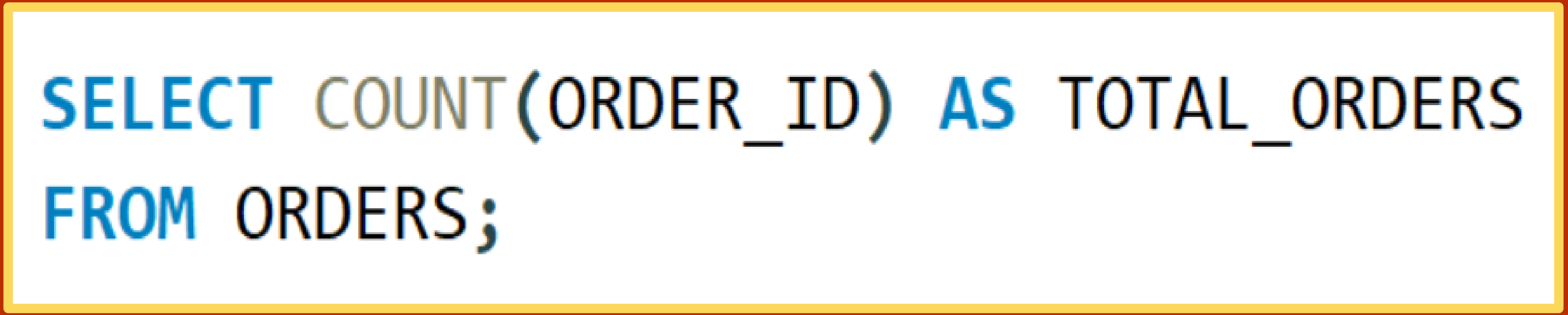
12.Analyze the cumulative revenue generated over time.

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

DATA MODEL

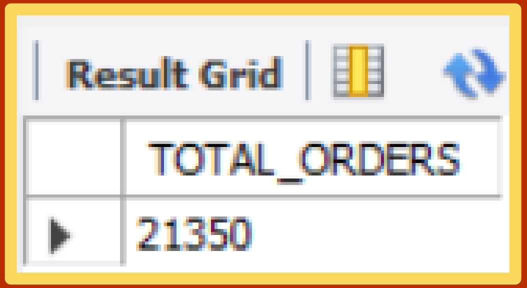

orders
date
order_id
time
Collapse
pizza_types
category
ingredients
name
pizza_type_id
Collapse
order_details
order_details_id
order_id
pizza_id
quantity
Collapse
pizzas
pizza_id
pizza_type_id
price
size
Collapse
1
*
1
*
1
*

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED



```sql
SELECT COUNT(ORDER_ID) AS TOTAL_ORDERS
FROM ORDERS;
```

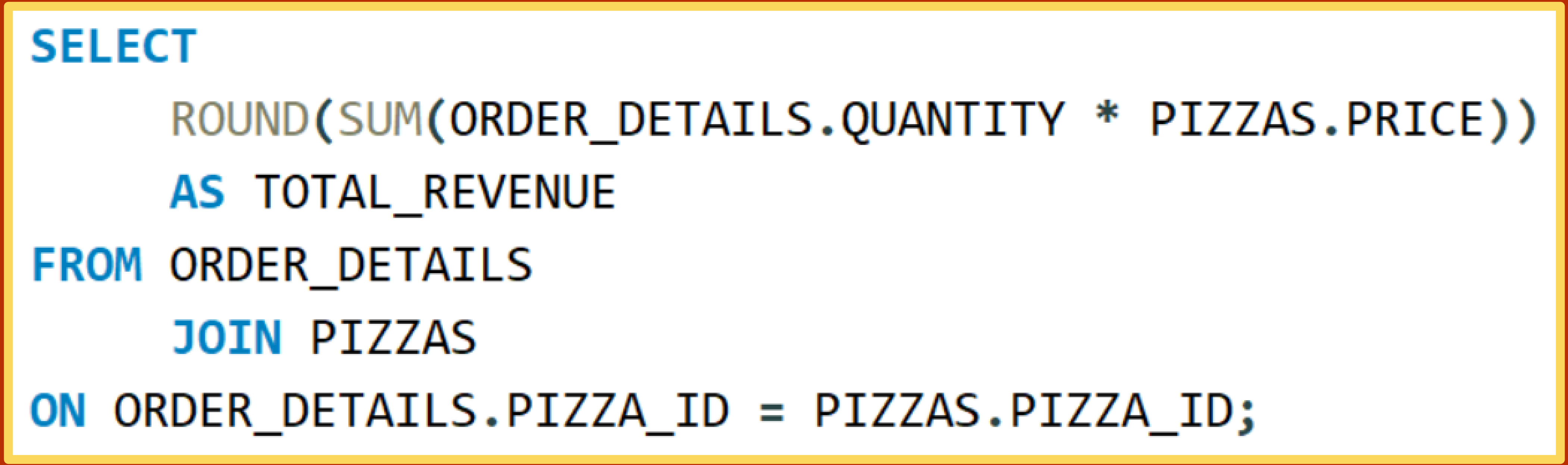
INPUT



```
TOTAL_ORDERS
21350
```

OUTPUT

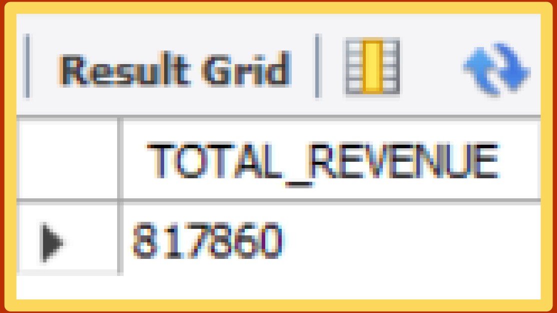
CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES



```
SELECT
    ROUND(SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE))
    AS TOTAL_REVENUE
FROM ORDER_DETAILS
    JOIN PIZZAS
ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID;
```

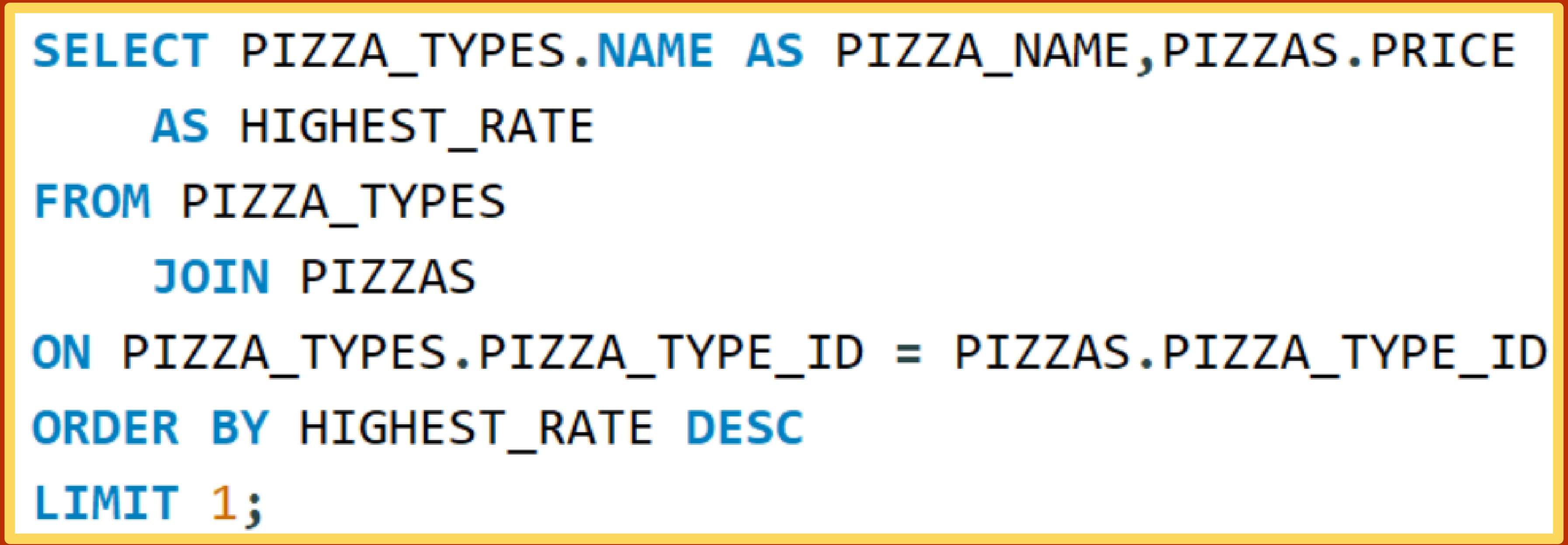
INPUT

OUTPUT



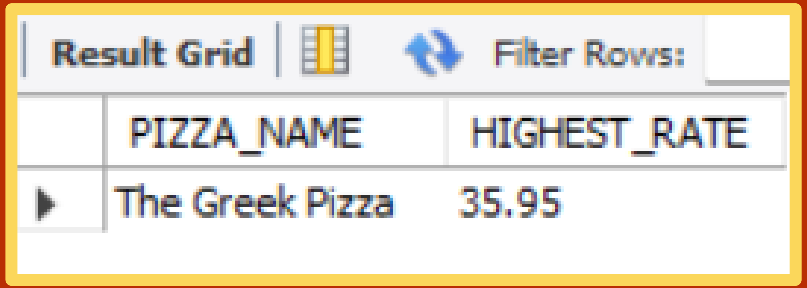
```
TOTAL_REVENUE
817860
```

IDENTIFY THE HIGHEST-PRICED PIZZA



```
SELECT PIZZA_TYPES.NAME AS PIZZA_NAME,PIZZAS.PRICE
    AS HIGHEST_RATE
FROM PIZZA_TYPES
    JOIN PIZZAS
ON PIZZA_TYPES.PIZZA_TYPE_ID = PIZZAS.PIZZA_TYPE_ID
ORDER BY HIGHEST_RATE DESC
LIMIT 1;
```

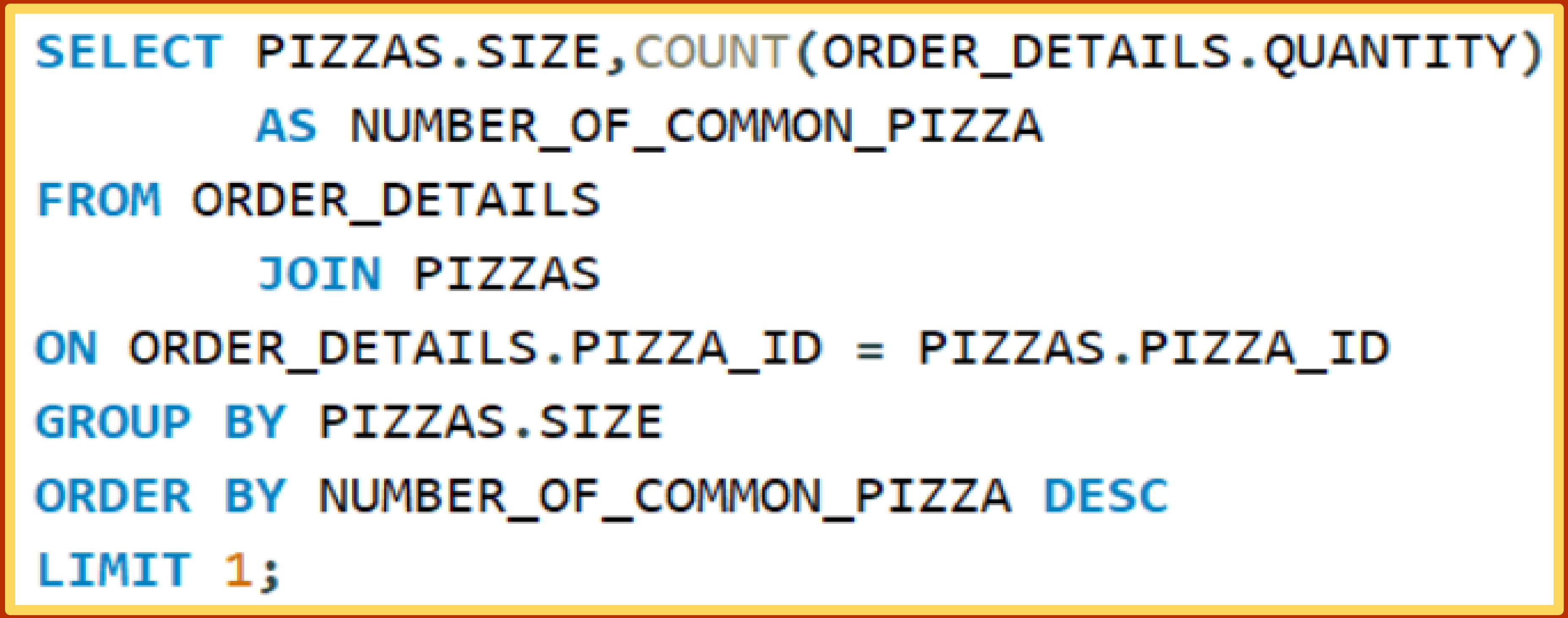
INPUT



PIZZA_NAME	HIGHEST_RATE
The Greek Pizza	35.95

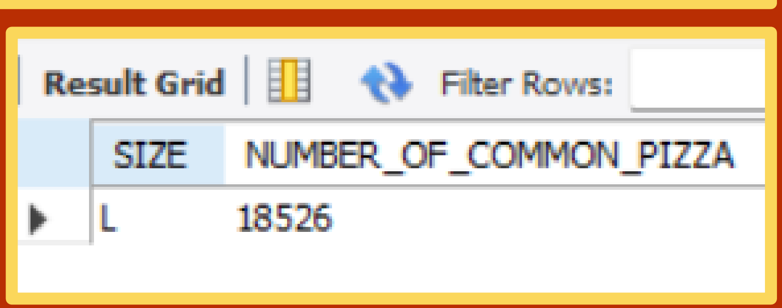
OUTPUT

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED



```
SELECT PIZZAS.SIZE,COUNT(ORDER_DETAILS.QUANTITY)
       AS NUMBER_OF_COMMON_PIZZA
FROM ORDER_DETAILS
       JOIN PIZZAS
ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID
GROUP BY PIZZAS.SIZE
ORDER BY NUMBER_OF_COMMON_PIZZA DESC
LIMIT 1;
```

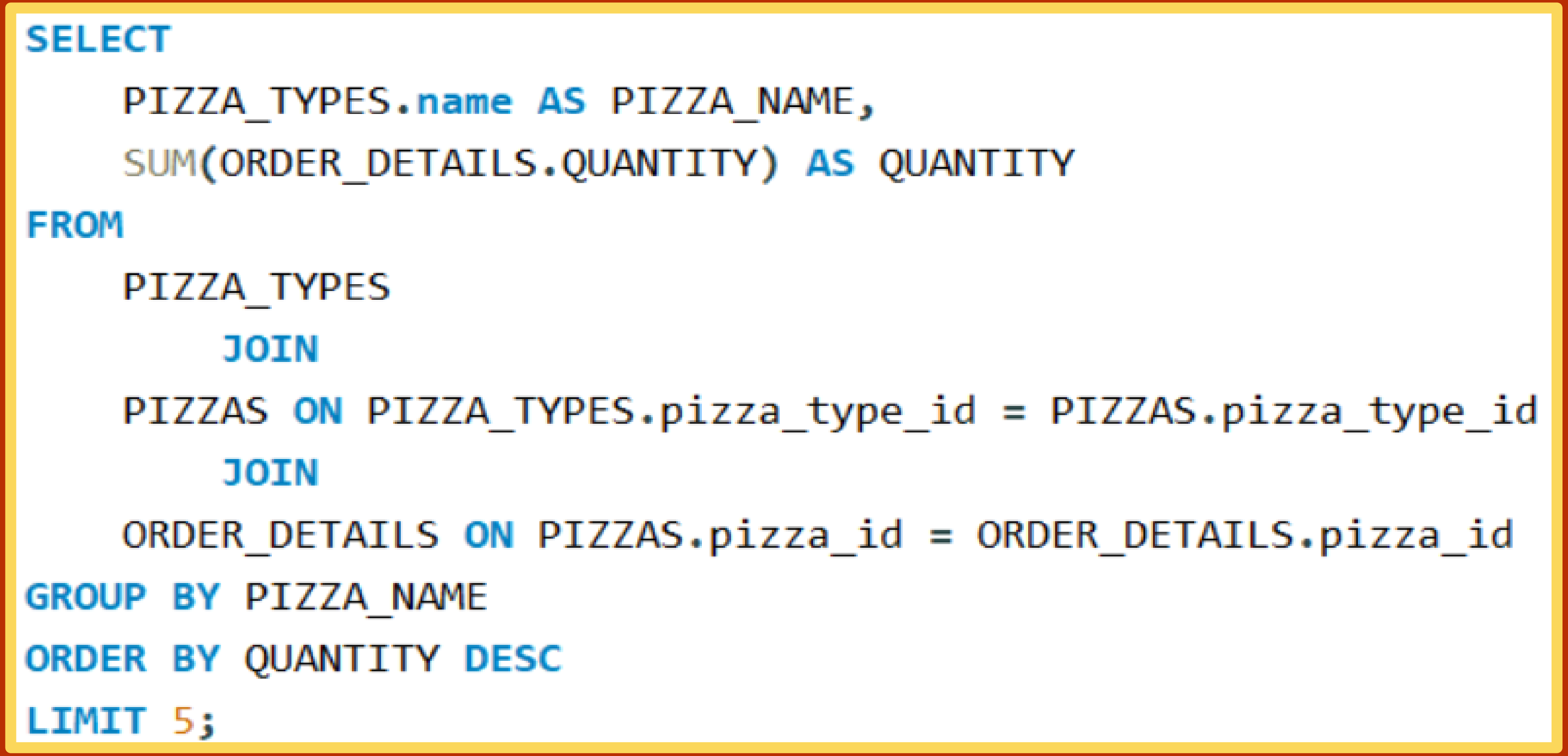
INPUT



```
SIZE   NUMBER_OF_COMMON_PIZZA
L      18526
```

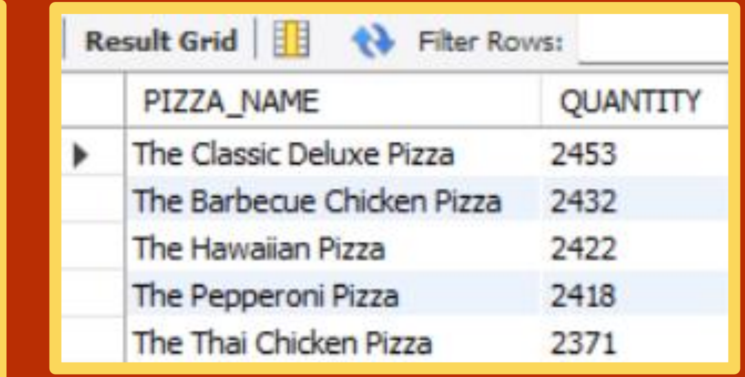
OUTPUT

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



```
SELECT
    PIZZA_TYPES.name AS PIZZA_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 PIZZAS.pizza_id = ORDER_DETAILS.pizza_id
GROUP BY PIZZA_NAME
ORDER BY QUANTITY DESC
LIMIT 5;
```

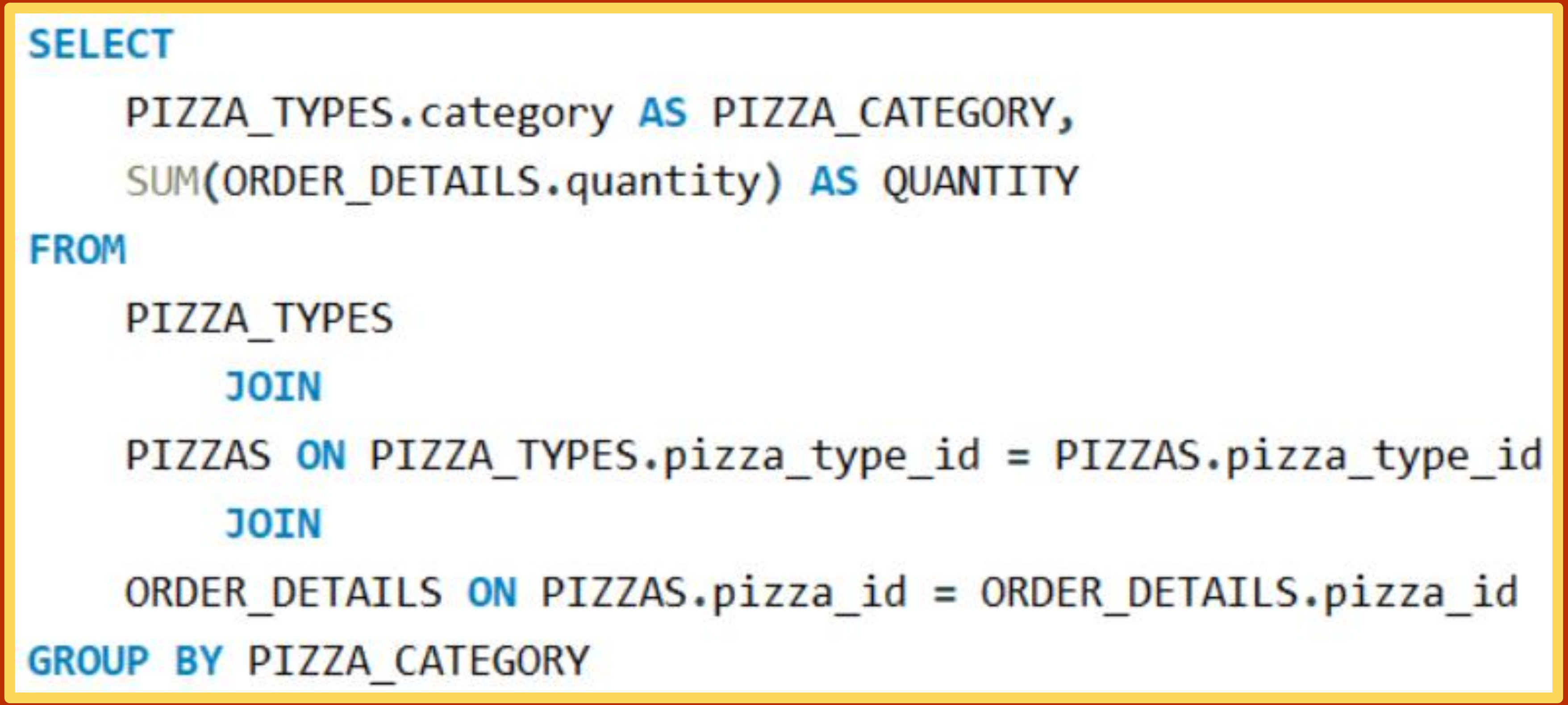
INPUT



PIZZA_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

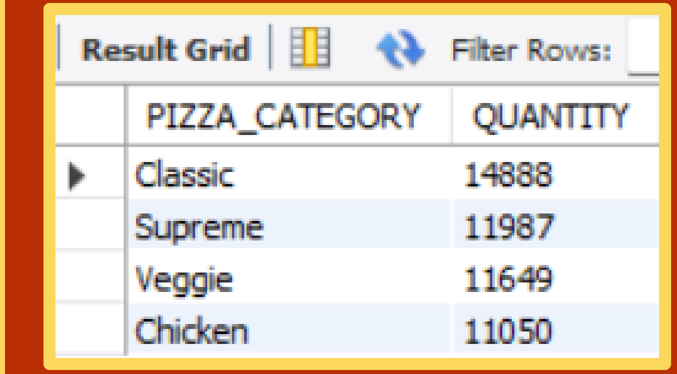
OUTPUT

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



```
SELECT
    PIZZA_TYPES.category AS PIZZA_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 PIZZAS.pizza_id = ORDER_DETAILS.pizza_id
GROUP BY PIZZA_CATEGORY
```

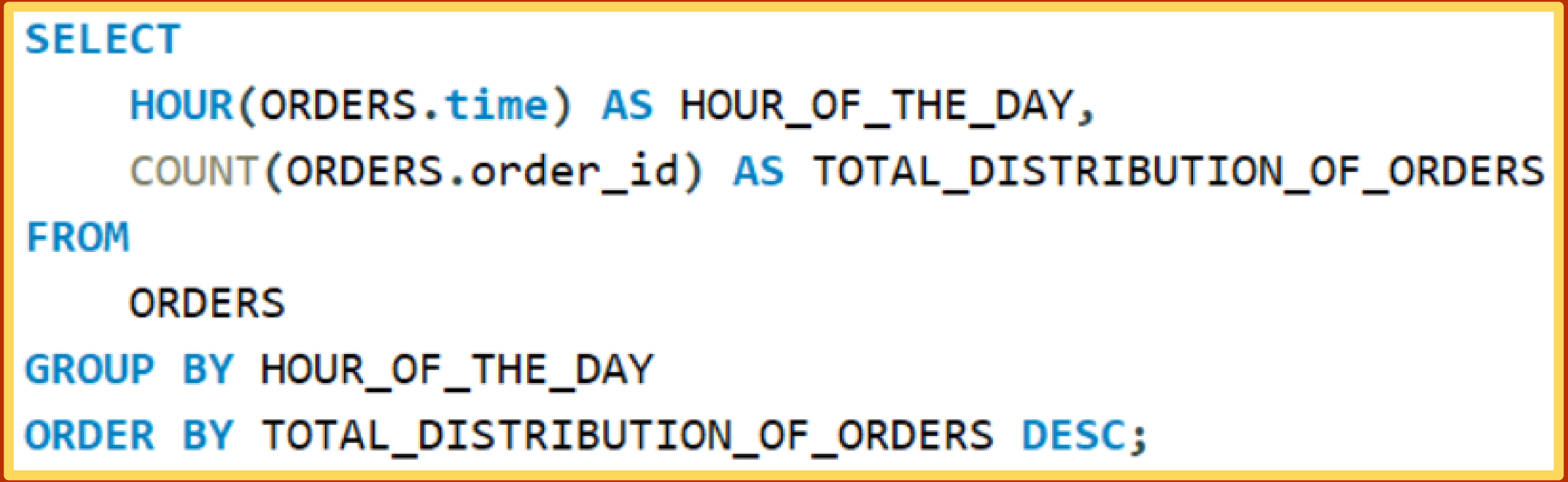
INPUT



PIZZA_CATEGORY	QUANTITY
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

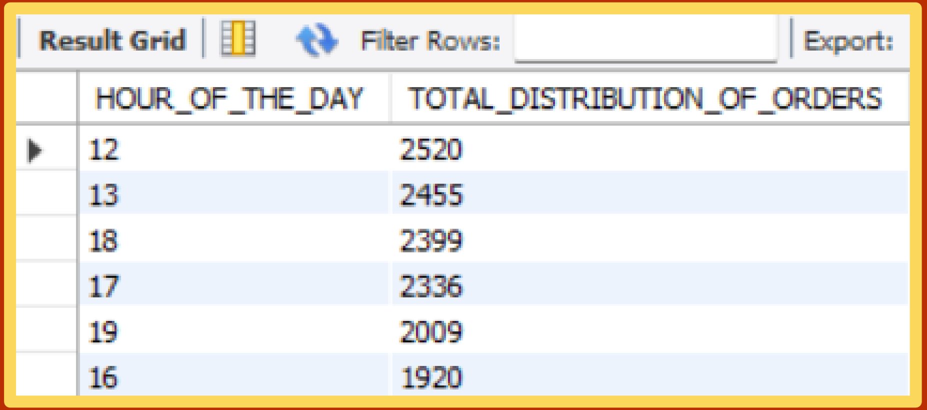
OUTPUT

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY



```
SELECT
    HOUR(ORDERS.time) AS HOUR_OF_THE_DAY,
    COUNT(ORDERS.order_id) AS TOTAL_DISTRIBUTION_OF_ORDERS
FROM
    ORDERS
GROUP BY HOUR_OF_THE_DAY
ORDER BY TOTAL_DISTRIBUTION_OF_ORDERS DESC;
```

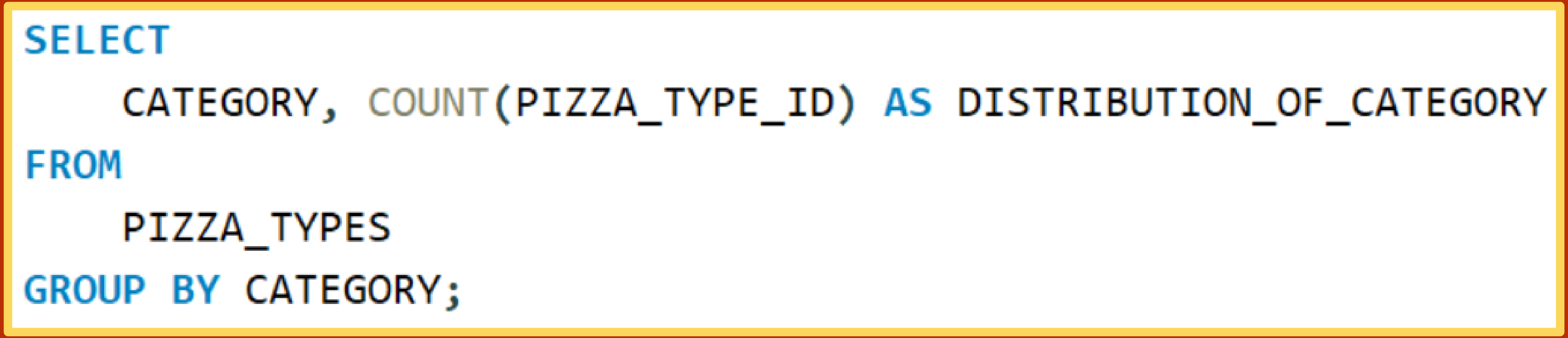
INPUT



HOUR_OF_THE_DAY	TOTAL_DISTRIBUTION_OF_ORDERS
12	2520
13	2455
18	2399
17	2336
19	2009
16	1920

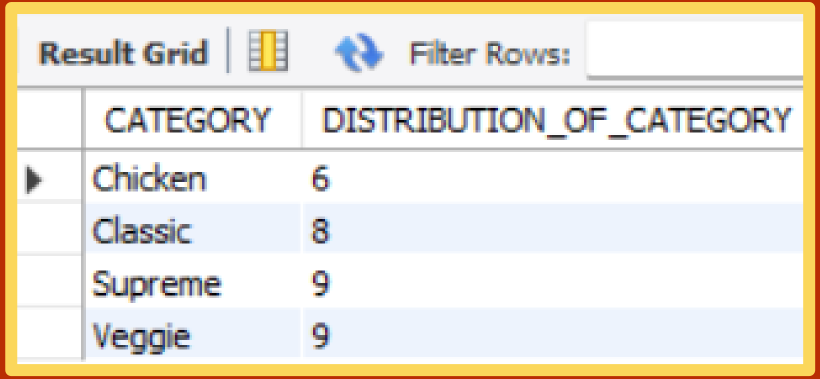
OUTPUT

JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS



```
SELECT
    CATEGORY, COUNT(PIZZA_TYPE_ID) AS DISTRIBUTION_OF_CATEGORY
FROM
    PIZZA_TYPES
GROUP BY CATEGORY;
```

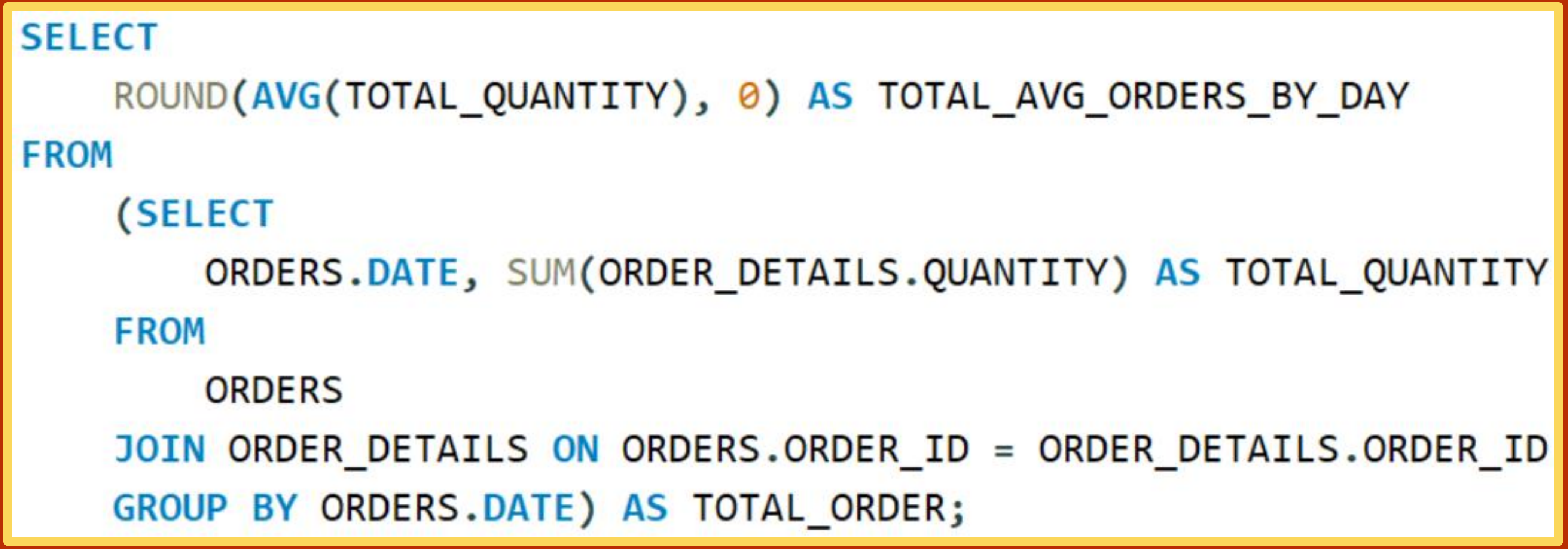
INPUT



CATEGORY	DISTRIBUTION_OF_CATEGORY
Chicken	6
Classic	8
Supreme	9
Veggie	9

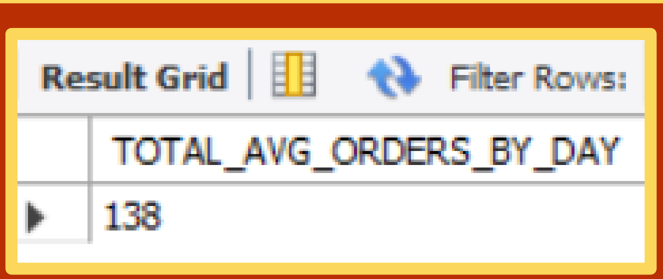
OUTPUT

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



```sql
SELECT
    ROUND(AVG(TOTAL_QUANTITY), 0) AS TOTAL_AVG_ORDERS_BY_DAY
FROM
    (SELECT
        ORDERS.DATE, SUM(ORDER_DETAILS.QUANTITY) AS TOTAL_QUANTITY
    FROM
        ORDERS
    JOIN ORDER_DETAILS ON ORDERS.ORDER_ID = ORDER_DETAILS.ORDER_ID
    GROUP BY ORDERS.DATE) AS TOTAL_ORDER;
```

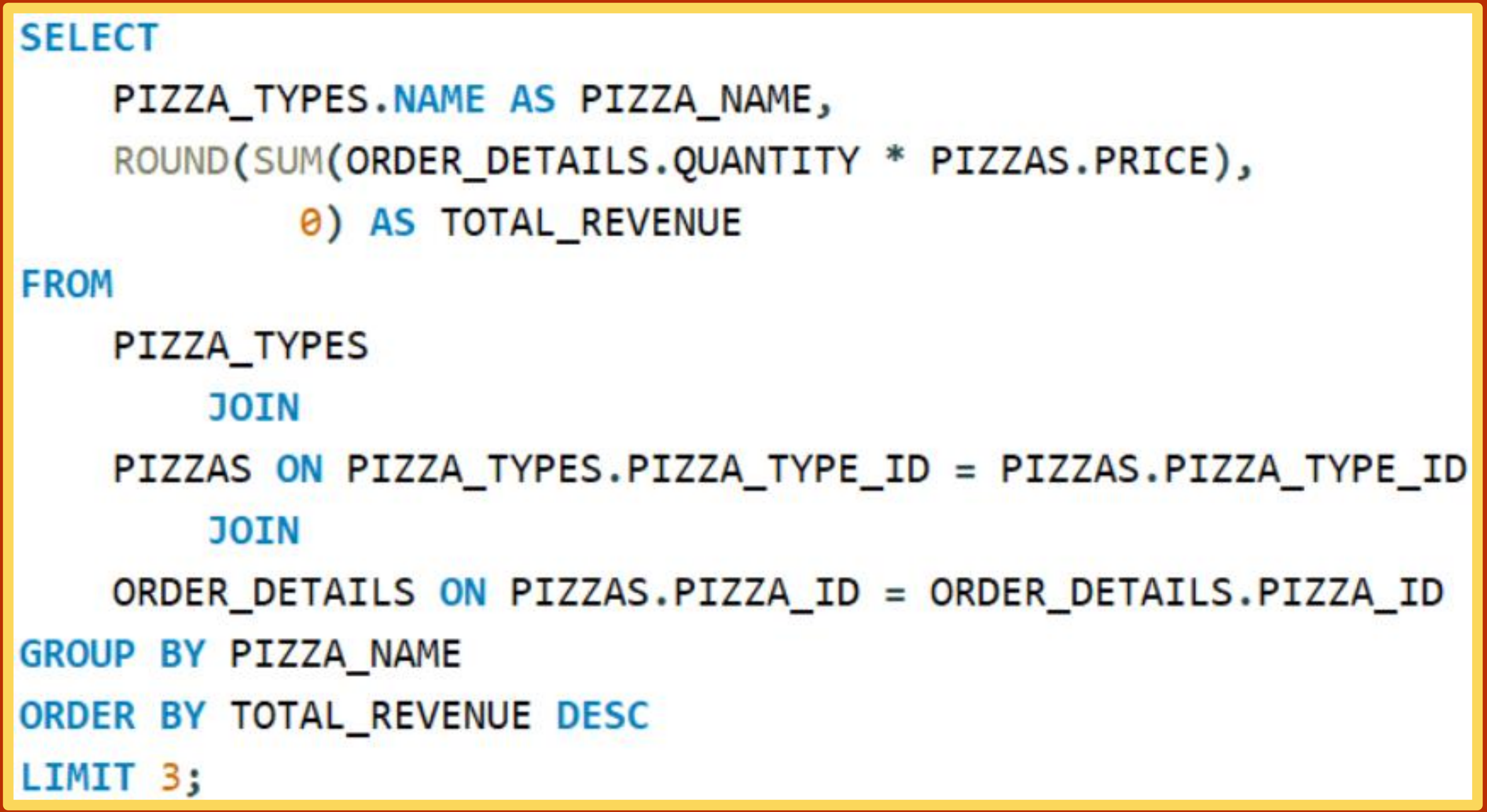
INPUT



```
TOTAL_AVG_ORDERS_BY_DAY
138
```

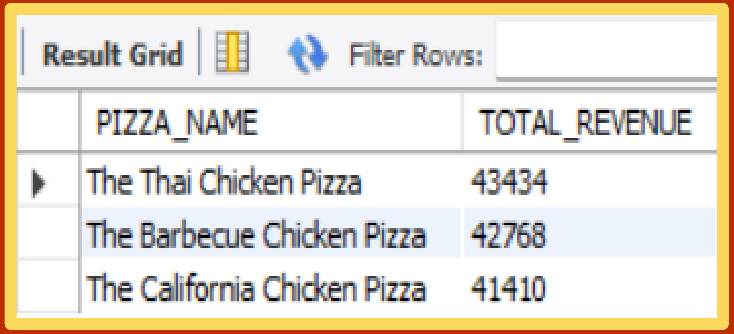
OUTPUT

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE



```
SELECT
    PIZZA_TYPES.NAME AS PIZZA_NAME,
    ROUND(SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE),
            0) AS TOTAL_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_NAME
ORDER BY TOTAL_REVENUE DESC
LIMIT 3;
```

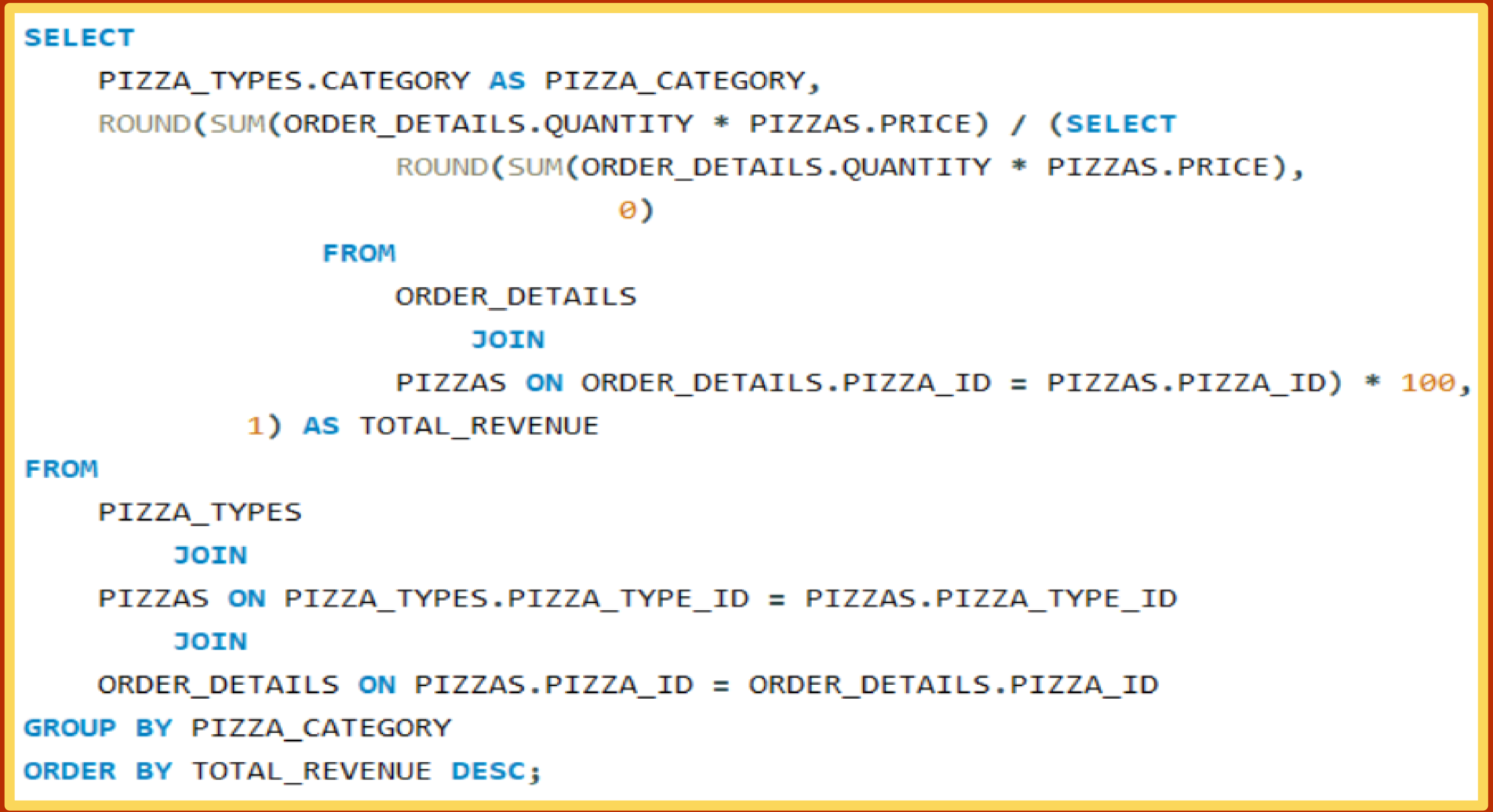
INPUT



PIZZA_NAME	TOTAL_REVENUE
The Thai Chicken Pizza	43434
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41410

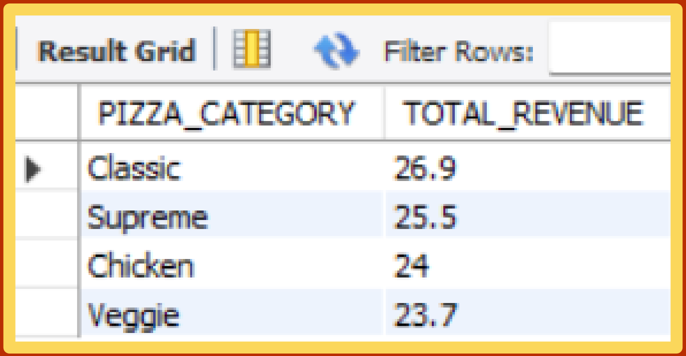
OUTPUT

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE



```
SELECT
    PIZZA_TYPES.CATEGORY AS PIZZA_CATEGORY,
    ROUND(SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE) / (SELECT
                    ROUND(SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE),
                                0)
                FROM
                    ORDER_DETAILS
                        JOIN
                    PIZZAS ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID) * 100,
            1) AS TOTAL_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_CATEGORY
ORDER BY TOTAL_REVENUE DESC;
```

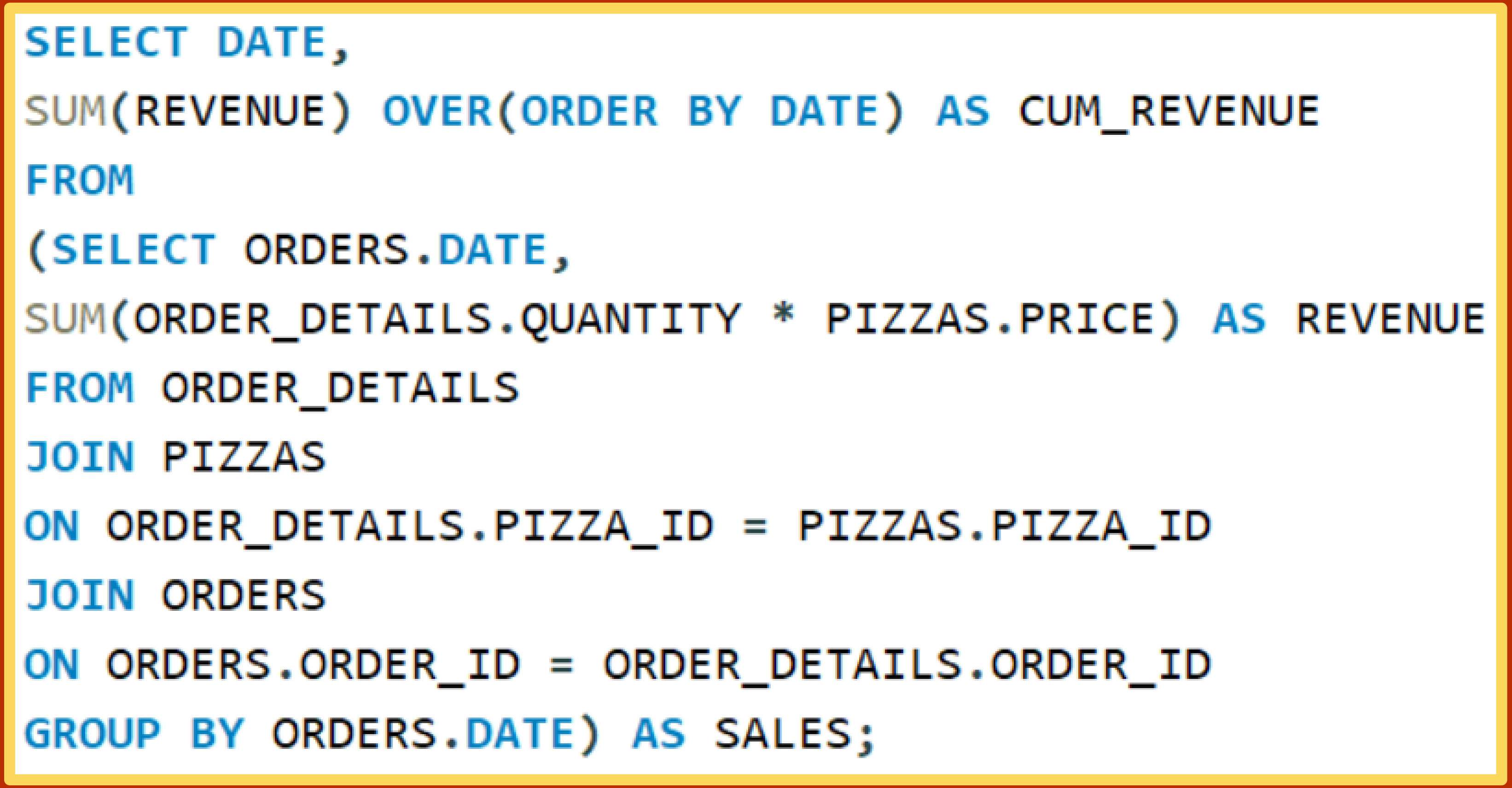
INPUT



PIZZA_CATEGORY	TOTAL_REVENUE
Classic	26.9
Supreme	25.5
Chicken	24
Veggie	23.7

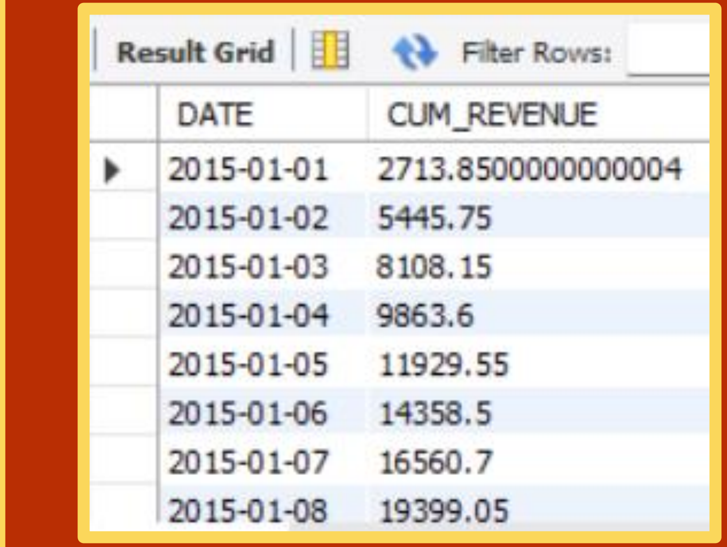
OUTPUT

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME



```
SELECT DATE,
SUM(REVENUE) OVER(ORDER BY DATE) AS CUM_REVENUE
FROM
(SELECT ORDERS.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.DATE) AS SALES;
```

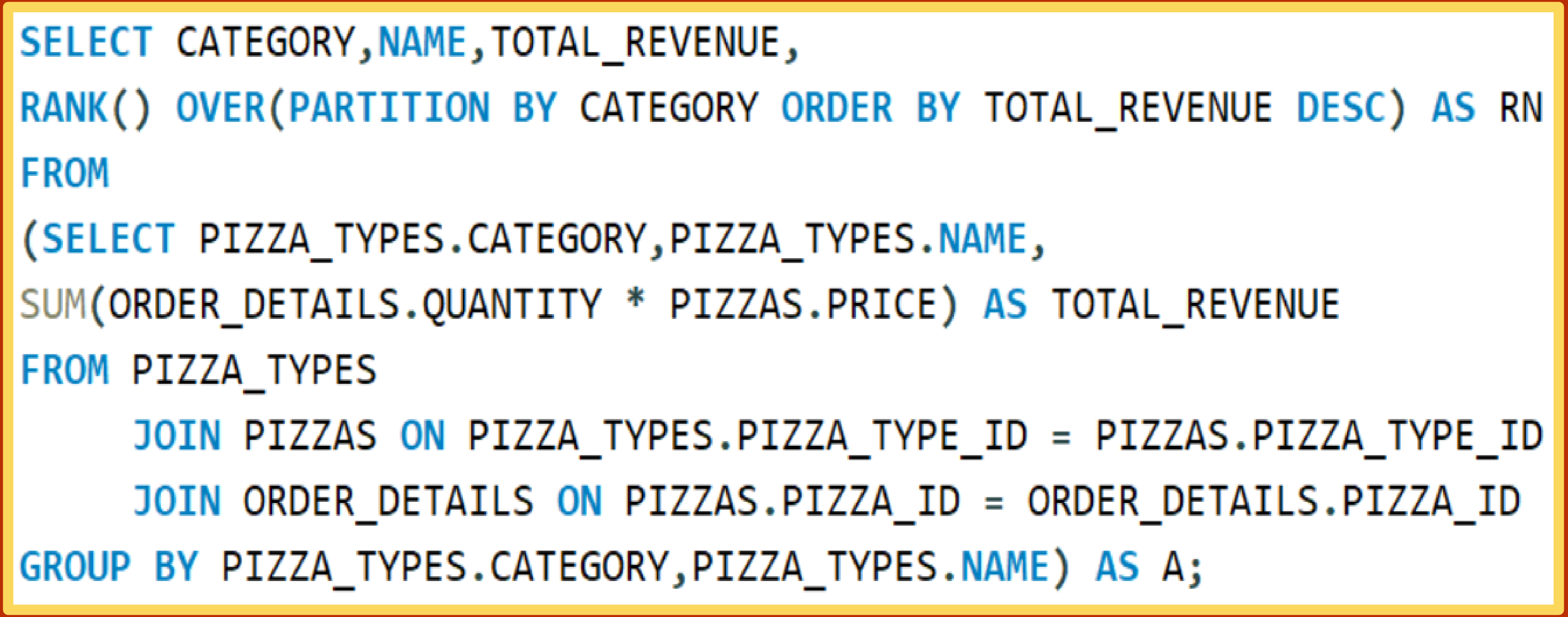
INPUT



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

OUTPUT

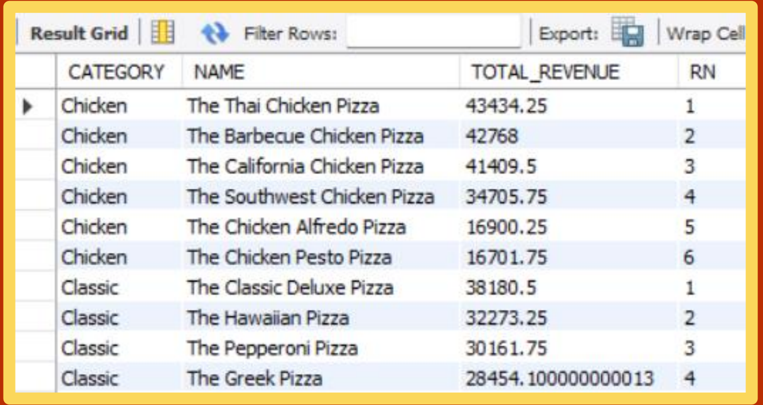
ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME



```
SELECT CATEGORY,NAME,TOTAL_REVENUE,
RANK() OVER(PARTITION BY CATEGORY ORDER BY TOTAL_REVENUE DESC) AS RN
FROM
(SELECT PIZZA_TYPES.CATEGORY,PIZZA_TYPES.NAME,
SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE) AS TOTAL_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 A;
```

INPUT

OUTPUT



CATEGORY	NAME	TOTAL_REVENUE	RN
Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768	2
Chicken	The California Chicken Pizza	41409.5	3
Chicken	The Southwest Chicken Pizza	34705.75	4
Chicken	The Chicken Alfredo Pizza	16900.25	5
Chicken	The Chicken Pesto Pizza	16701.75	6
Classic	The Classic Deluxe Pizza	38180.5	1
Classic	The Hawaiian Pizza	32273.25	2
Classic	The Pepperoni Pizza	30161.75	3
Classic	The Greek Pizza	28454.100000000013	4

CONCLUSION

"In conclusion, the Pizza Sales MySQL project has been a rewarding endeavor, showcasing the power of data analysis in uncovering valuable insights. Leveraging MySQL, we've efficiently managed and manipulated large datasets, facilitating informed decision-making for stakeholders. The project highlights the importance of data-driven strategies in optimizing sales performance and enhancing customer satisfaction. Moving forward, the insights gleaned from this project will serve as a foundation for refining marketing strategies, improving operational efficiency, and driving business growth in the competitive pizza industry."

THANK YOU