SQL PROJECT

PIZZA SALES

By Mohit Kewat

Hello, my name is Mohit Kewat. In this project, I have utilized **SQL queries** to address various questions related to **pizza sales.**

QUESTIONS RELATED TO PIZZA SALES.

- 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.
- Analyze the cumulative revenue generated over time.

1. Retrieve the total number of orders placed.

Query

```
SELECT COUNT(ORDER_ID) AS TOTAL_ORDERS FROM order_details;
```

Output

Result Grid

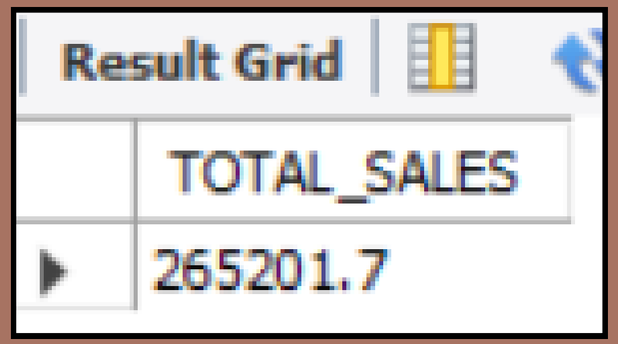
	TOTAL_ORDERS
▶	15768

2. Calculate the total revenue generated from pizza sales.

Query

```
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

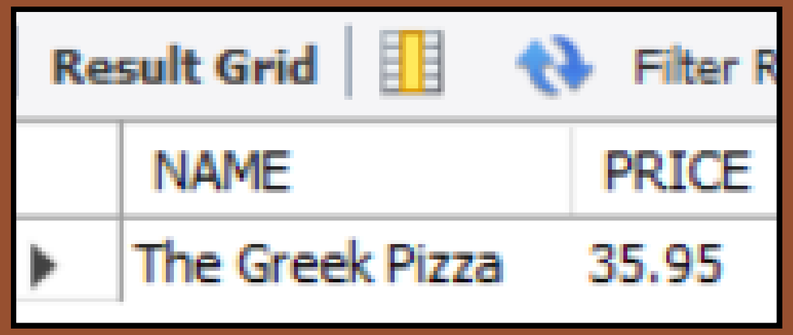


3. Identify the highest-priced pizza.

Query

```
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

4. Identify the most common pizza size ordered.

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

Result Grid | Filter Rows:

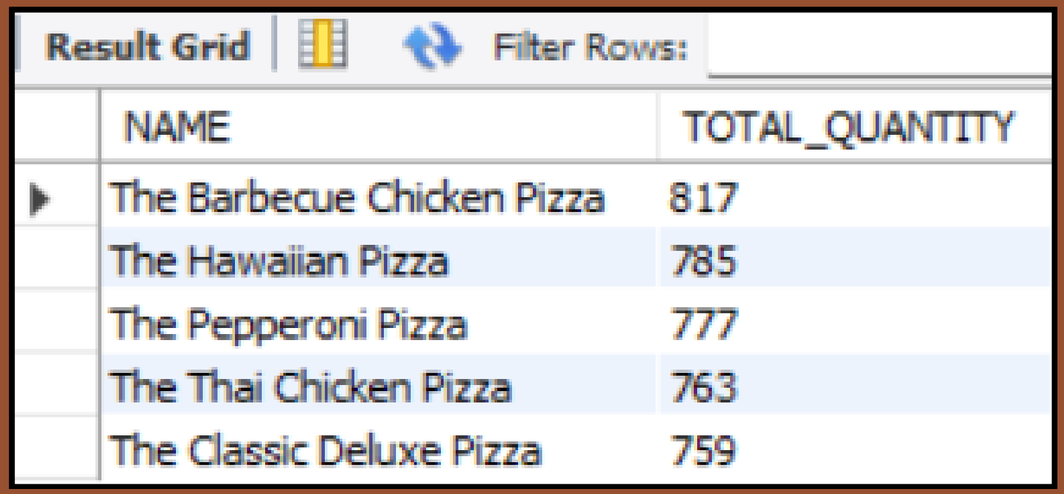
SIZE	ORDER_COUNT
L	6031
M	4960
S	4580
XL	185
XXL	12

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

Query

```
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;
```

Output



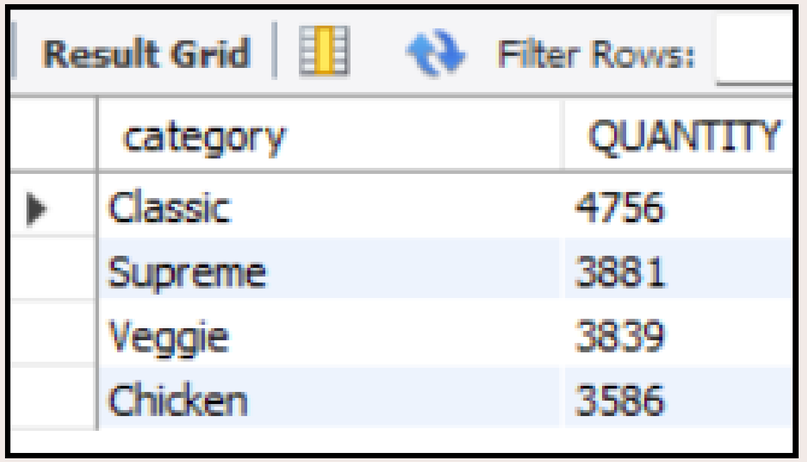
NAME	TOTAL_QUANTITY
The Barbecue Chicken Pizza	817
The Hawaiian Pizza	785
The Pepperoni Pizza	777
The Thai Chicken Pizza	763
The Classic Deluxe Pizza	759

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

Query

```
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	4756
Supreme	3881
Veggie	3839
Chicken	3586

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

Query

```
SELECT HOUR(ORDER_TIME) AS HOUR, COUNT(ORDER_ID) AS ORDER_COUNT
FROM orders
GROUP BY HOUR(ORDER_TIME);
```

Output

Result Grid | Filter

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

Query

```
SELECT CATEGORY, COUNT(NAME)
FROM pizza_types
GROUP BY CATEGORY;
```

Output

Result Grid | Filter Rows:

CATEGORY	COUNT(NAME)
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

Query

```
SELECT ROUND(AVG(QUANTITY),0) AS AVG_PER_DAY_ORDER 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

Result Grid | Filte

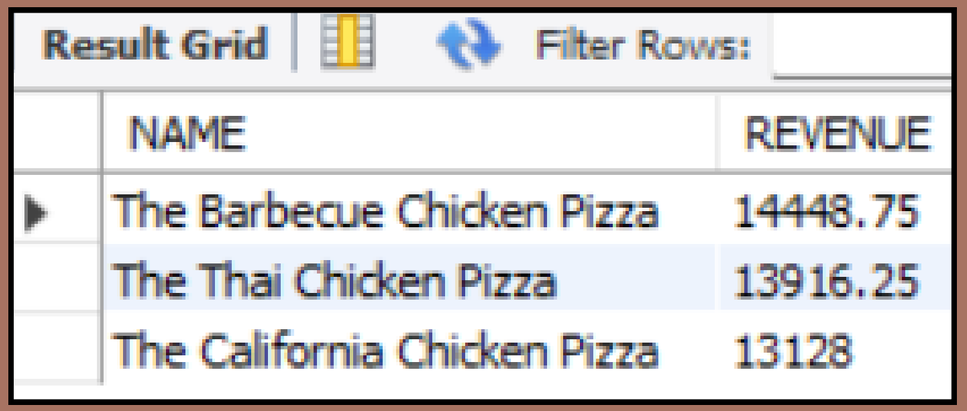
AVG_PER_DAY_ORDER
138

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

Query

```
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 Barbecue Chicken Pizza	14448.75
The Thai Chicken Pizza	13916.25
The California Chicken Pizza	13128

11. Analyze the cumulative revenue generated over time.

Query

```
SELECT ORDER_DATE,
SUM(REVENUE) OVER(ORDER BY ORDER_DATE) AS cumulative_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

Result Grid | Filter Rows: | Expo

ORDER_DATE	cumulative_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
2015-01-18	40978.600000000006
2015-01-19	43365.75000000001
2015-01-20	45763.65000000001
2015-01-21	47804.20000000001

Thank You :)