SQL PROJECT ON PIZZA SALES

WELCOME TO PIZZA SALES SQL PROJECT

Description

This project focuses on analyzing a pizza sales dataset using SQL to extract meaningful business insights. The objective of this project is to understand sales performance, customer demand, and revenue trends by writing efficient SQL queries.

WELCOME TO PIZZA SALES SQL PROJECT

Description

This project analyzes a pizza sales dataset collected from Kaggle using SQL. The data consists of four related tables (pizza, pizza_type, orders & order_details), which were joined as required to perform meaningful analysis.

WELCOME TO PIZZA SALES SQL PROJECT

Description

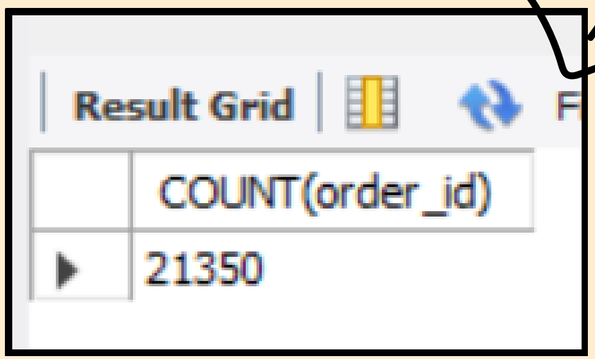
The objective of this project is to apply SQL concepts to solve real-world business questions related to sales performance, revenue, and customer demand. Key SQL techniques such as **joins, aggregate functions, GROUP BY, subqueries, and window functions** have been used throughout the project.

The following sections present the SQL queries along with their outputs, demonstrating how each analytical question was solved.

1.

Retrieve the total number of orders placed.

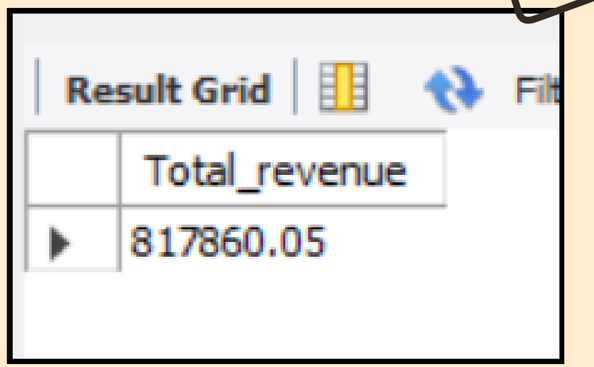
```
1
2 • SELECT COUNT(order_id) FROM orders;
3
```



2.

Calculate the total revenue generated from pizza sales.

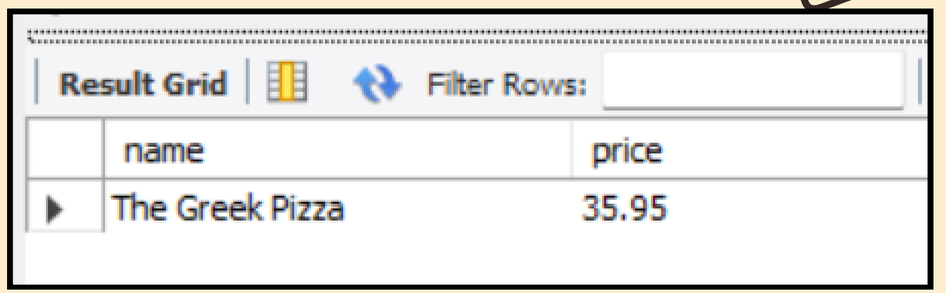
```
1
2  SELECT ROUND(SUM(price*quantity), 2) AS Total_revenue
3  FROM pizzas AS a
4  JOIN order_details AS b
5  ON a.pizza_id = b.pizza_id;
6
```



3.

Identify the highest-priced pizza.

```
2 • SELECT name, price
3   FROM pizza_types AS a
4   JOIN pizzas AS b
5   ON a.pizza_type_id = b.pizza_type_id
6   ORDER BY price DESC
7   LIMIT 1;
```



4.

Identify the most common pizza size ordered.

```sql
SELECT size, COUNT(size) AS common_ordered
FROM pizzas AS a
JOIN order_details AS b
ON a.pizza_id = b.pizza_id
GROUP BY size
ORDER BY common_ordered DESC
LIMIT 1;
```

5.

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

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

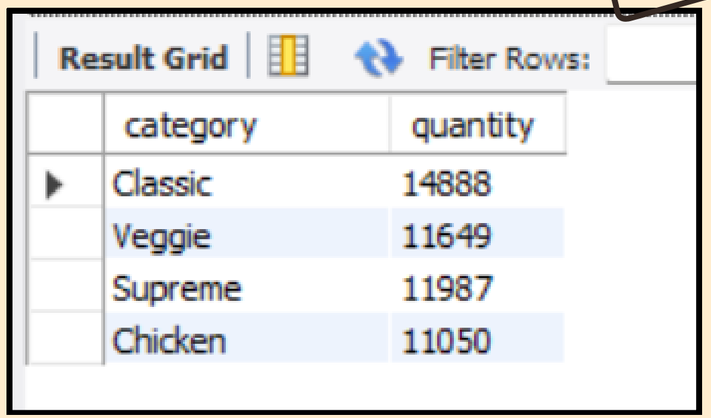
Result Grid | Filter Rows:

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.

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

```
1 •  SELECT category, SUM(quantity) AS quantity
2    FROM pizza_types AS a
3    JOIN pizzas AS b
4    ON a.pizza_type_id = b.pizza_type_id
5    JOIN order_details AS c
6    ON b.pizza_id = c.pizza_id
7    GROUP BY category;
```



Result Grid | Filter Rows:

	category	quantity
▶	Classic	14888
	Veggie	11649
	Supreme	11987
	Chicken	11050

7.

Determine the distribution of orders by hour of the day.

```sql
SELECT HOUR(order_time) AS time, COUNT(order_id) as orders_per_hr
FROM orders
GROUP BY time;
```

Result Grid | Filter Rows:

time	orders_per_hr
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

8.

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

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

Result Grid | Filter Rows:

Name	Revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

9.

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

```
1 •  SELECT Category, COUNT(name) as Distribution
2    FROM pizza_types AS a
3    JOIN pizzas AS b
4    ON a.pizza_type_id = b.pizza_type_id
5    JOIN order_details AS c
6    ON b.pizza_id = c.pizza_id
7    GROUP BY category;
```

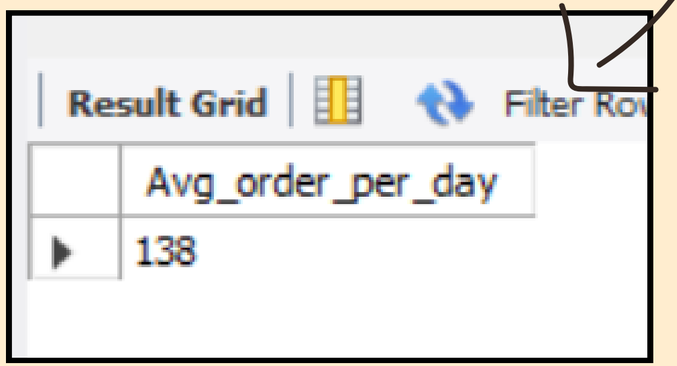
Result Grid | Filter Rows:

Category	Distribution
Classic	14579
Veggie	11449
Supreme	11777
Chicken	10815

10.

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

```sql
SELECT ROUND(AVG(Quantity), 0) AS Avg_order_per_day
FROM
(SELECT order_date, SUM(quantity) Quantity
FROM orders AS a
JOIN order_details AS b
ON a.order_id = b.order_id
GROUP BY order_date) AS order_quantity;
```




THANK YOU