TASTE_OF_PIZZAS

P I Z Z A _ S A L E S

Using SQL

ABOUT THIS PROJECT :

This project focuses on analyzing pizza sales data using SQL to uncover key business insights. The main objective was to explore and understand sales performance, customer preferences, and order patterns by writing efficient SQL queries. The dataset includes details such as order dates, pizza types, quantities, prices, and sizes.

Through this analysis, I derived actionable insights that can help improve inventory planning, marketing strategies, and overall decision-making for a pizza business. This project strengthened my SQL skills and deepened my understanding of data-driven business analysis.

USING SQL, I PERFORMED DATA CLEANING, AGGREGATION, FILTERING, AND JOINING OPERATIONS TO ANSWER VARIOUS BUSINESS QUESTIONS, SUCH AS:

- 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.

1. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```
3 •  SELECT
4        ROUND(SUM(orders_details.quantity * pizzas.price),
5                2) AS total_sales
6    FROM
7        orders_details
8            JOIN
9        pizzas ON pizzas.pizza_id = orders_details.pizza_id
```

Result Grid

	total_sales
▶	817860.05

2. IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

```sql
-- Identify the most common pizza size ordered.

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

Result Grid | Filter Rows:

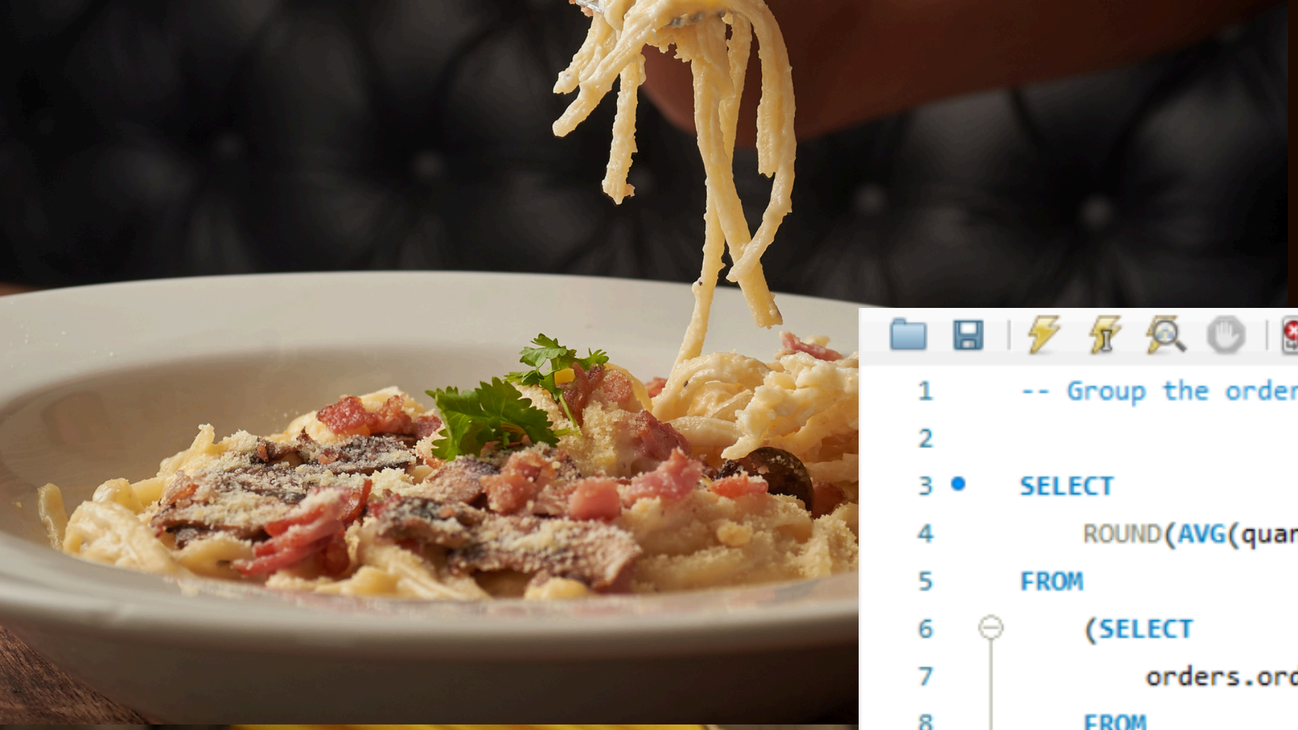
size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

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

```
1   -- List the top 5 most ordered pizza types along with their quantities.
2
3   SELECT
4       pizza_types.name, SUM(orders_details.quantity) AS quantity
5   FROM
6       pizza_types
7           JOIN
8       pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9           JOIN
10      orders_details ON orders_details.pizza_id = pizzas.pizza_id
11  GROUP BY pizza_types.name
12  ORDER BY quantity DESC
13  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



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

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

SELECT
    ROUND(AVG(quantity), 0) as avg_sales_per_day
FROM
    (SELECT
        orders.order_date, SUM(orders_details.quantity) AS quantity
    FROM
        orders
    JOIN orders_details ON orders.order_id = orders_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

Result Grid | Filte

avg_sales_per_day
138

5. DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

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

select pizza_types.name,
sum(orders_details.quantity * pizzas.price) as revenue
from pizza_types join pizzas
on pizzas.pizza_type_id = pizza_types.pizza_type_id
join orders_details
on orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.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

6. ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
-- 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,
sum(orders_details.quantity * pizzas.price) as revenue
from orders_details join pizzas
on orders_details.pizza_id = pizzas.pizza_id
join orders
on orders.order_id = orders_details.order_id
group by orders.order_date) as sales ;
```

Result Grid | Filter Rows: | Export:

order_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
2015-01-09	21526.4
2015-01-10	23990.350000000002
2015-01-11	25862.65

7. DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY.

```sql
-- Determine the top 3 most ordered pizza types based on revenue for each pizza category.

select name, revenue, category
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((orders_details.quantity) * pizzas.price) as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join orders_details
on orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.category, pizza_types.name) as a) as b
where rn <= 3;
```

Result Grid | Filter Rows: | Export: | Wrap Cell

name	revenue	category
The Thai Chicken Pizza	43434.25	Chicken
The Barbecue Chicken Pizza	42768	Chicken
The California Chicken Pizza	41409.5	Chicken
The Classic Deluxe Pizza	38180.5	Classic
The Hawaiian Pizza	32273.25	Classic
The Pepperoni Pizza	30161.75	Classic
The Spicy Italian Pizza	34831.25	Supreme
The Italian Supreme Pizza	33476.75	Supreme
The Sicilian Pizza	30940.5	Supreme

Result 5

SQL CONCEPTS LEARNED FROM THIS PROJECT :-

During this project, I strengthened my understanding and practical application of various SQL concepts at basic, intermediate, and advanced levels. Below is a summary of the concepts I explored:

1.Basic SQL Concepts

- SELECT Statements: Used to retrieve specific columns like order count, revenue, and pizza names.
- Aggregate Functions: Applied COUNT(), SUM(), MAX() to calculate total orders, revenue, and highest-priced pizzas.
- GROUP BY & ORDER BY: Grouped data to find popular pizza types and sorted results based on quantity.
- LIMIT: Limited output to show top-selling items like top 5 pizzas.
- WHERE Clauses: Filtered records based on conditions like size or price.

2. Intermediate SQL Concepts

- JOIN Operations: Used INNER JOIN and LEFT JOIN to combine data from multiple tables (orders, pizzas, categories).
- Date and Time Functions: Extracted hour from timestamps using functions like EXTRACT(HOUR FROM ...) to analyze order patterns.
- GROUP BY with JOINs: Grouped joined data to calculate category-wise orders and average pizzas per day.
- Nested Queries: Used subqueries to find top items based on total quantity or revenue.

3. Advanced SQL Concepts

- Window Functions: Used functions like SUM() OVER() to compute cumulative revenue over time.
- Percentage Calculations: Calculated revenue contributions by dividing individual totals by overall revenue.
- Complex Aggregations: Combined multiple groupings and conditions to find top 3 pizzas by revenue in each category.

This hands-on project enhanced my ability to write efficient and meaningful SQL queries, transforming raw data into actionable business insights.

CONTACT :

+91 9925152359

G-MAIL :

Parvashah3183@gmail.com

LINKEDIN :

https://www.linkedin.com/in/parvashah3183

TASTE_OF_PIZZAS

THANK YOU !!