SQL PROJECT ON PIZZA SALES ANALYSIS

EXPLORING PIZZA SALES DATA USING SQL

INTRODUCTION

Overview:

- Introduction to the SQL project on pizza sales analysis.
- Brief description of the project's objectives and significance.

Purpose:

- To analyze pizza sales data using SQL queries.
- To gain insights into customer preferences, sales trends, and business strategies optimization.

Objectives:

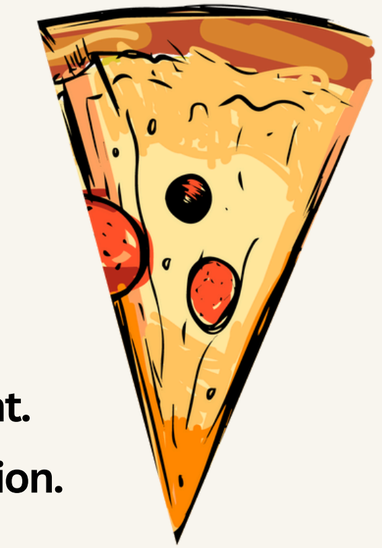
1.Analyze sales data: Extracting meaningful insights from the dataset to inform decision-making.

2.Identify popular pizza types: Determining which pizza types are most frequently ordered by customers.

3. Explore customer preferences: Analyzing toppings, sizes, and other factors to understand customer preferences.

4.Identify sales trends: Identifying patterns and trends in sales data to forecast future demand and optimize business operations.

DATA DESCRIPTION



- The dataset used for this project contains records of pizza orders from a fictional pizza restaurant.
- The dataset includes columns such as Date, Pizza Type, Quantity , Price, and Customer Information.
- The dataset consists of 1000 entries, spanning a period of one year.
- Prior to analysis, the dataset underwent preprocessing steps to remove duplicate entries and handle missing values.

Retrieve the total number of orders placed.

```sql
SELECT COUNT(Order_Id) AS Total_Orders
 FROM Orders;
```

Result Grid

	Total_Orders
▶	21350

Calculate the total revenue generated from pizza sales.

```sql
SELECT
    ROUND(SUM(orders_details.quantity * pizzas.price),
            2) AS Total_Sales
FROM
    orders_details
        JOIN
    pizzas ON pizzas.pizza_id = orders_details.Pizza_Id
```

Result Grid

	Total_Sales
▶	817860.05

Identify the highest-priced pizza.

```sql
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;
```

Result Grid | Filter Rows

name	price
The Greek Pizza	35.95

Identify the most common pizza size ordered

```sql
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

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

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

```sql
SELECT pizza_types.name, SUM(orders_details.quantity) AS quantity
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.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

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

```sql
SELECT pizza_types.category, SUM(orders_details.quantity) AS quantity
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
ORDER BY quantity DESC;
```

Result Grid

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hour of the day.

```sql
SELECT HOUR(order_time) AS order_hour, COUNT(order_id) AS order_count
FROM orders
GROUP BY HOUR(order_time);
```

Result Grid | Filter Rows

order_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

Identify Join relevant tables to find the category-wise distribution of pizzas. highest-priced pizza.

```
SELECT category, COUNT(name) AS name_count
FROM pizza_types
GROUP BY category;
```

Result Grid | Filter Ro

category	name_count
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

```sql
SELECT
    ROUND(AVG(quantity), 0) AS avg_quantity
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
LIMIT 0 , 1000;
```

Result Grid

avg_quantity
138

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

```sql
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

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

```sql
SELECT
    pizza_types.category,
    ROUND((SUM(orders_details.quantity * pizzas.price) / (SELECT SUM(orders_details.quantity * pizzas.price)
    FROM
    orders_details JOIN pizzas ON pizzas.pizza_id = orders_details.pizza_id)) * 100, 2) AS percentage_of_total_sales
FROM pizza_types
JOIN pizzas ON pizza_types.pizza_type_id = pizzas.
JOIN orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY percentage_of_total_sales DESC
LIMIT 0, 1000;
```

Result Grid | Filter Rows:

category	percentage_of_total_sales
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Analyze the cumulative revenue generated over time.

```sql
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:

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
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
2015-01-22	50300.90000000001
2015-01-23	52724.600000000006
2015-01-24	55013.850000000006
2015-01-25	56631.40000000001
2015-01-26	58515.80000000001
2015-01-27	61043.85000000001
2015-01-28	63050.85000000001

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

```sql
SELECT name,revenue 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:

name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5
The Classic Deluxe Pizza	38180.5
The Hawaiian Pizza	32273.25
The Pepperoni Pizza	30161.75
The Spicy Italian Pizza	34831.25
The Italian Supreme Pizza	33476.75
The Sicilian Pizza	30940.5
The Four Cheese Pizza	32265.70000000065
The Mexicana Pizza	26780.75
The Five Cheese Pizza	26066.5