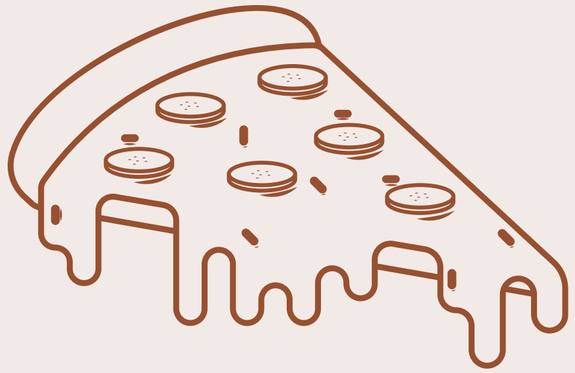


SQL Projects for Pizza Sales

A Comprehensive Analysis of Pizza Sales Data Using SQL

Tushar Soni | Indore, India

About Me & Project Overview

About Me:

- Tushar Soni
- Indore, India
- Email: tushar2000soni@gmail.com
- LinkedIn: linkedin.com/in/tusharsoni2024/
- GitHub: github.com/Tusharsoni69
  (Link to project repository)
- HackerRank: hackerrank.com/profile/tushar_2800

Introduction to Project:

Project Overview:
The goal of this project is to analyze pizza sales data using SQL to uncover key insights into sales trends, revenue generation, and customer preferences. This analysis is broken down into basic, intermediate, and advanced SQL queries to progressively extract deeper insights.

PROJECT OVERVIEW

- **Objective:**

To leverage SQL queries in analyzing pizza sales data to gain actionable insights.

- **Scope:**

This project involves basic retrievals, intermediate data joins, and advanced analytical queries to explore sales patterns and customer behaviors.

- **Outcome:**

The project provides a detailed understanding of pizza sales trends, helping in identifying top-selling pizzas, peak ordering times, and revenue distribution.

DATABASE STRUCTURE

Tables Involved:

- Orders: Contains details of each order placed.
- Order_Details: Holds specific details about pizzas in each order.
- Pizzas: Information about the pizzas, including size and price.
- Pizza_Types: Categorizes pizzas by type and ingredients.

Key Columns:

- order_id: Unique identifier for orders.
- pizza_id: Unique identifier for pizzas.
- pizza_type: Type/category of pizza.
- price: Cost of the pizza.
- size: Size of the pizza (e.g., Small, Medium, Large).
- quantity: Number of pizzas ordered.
- date and time: When the order was placed.

Relationships:

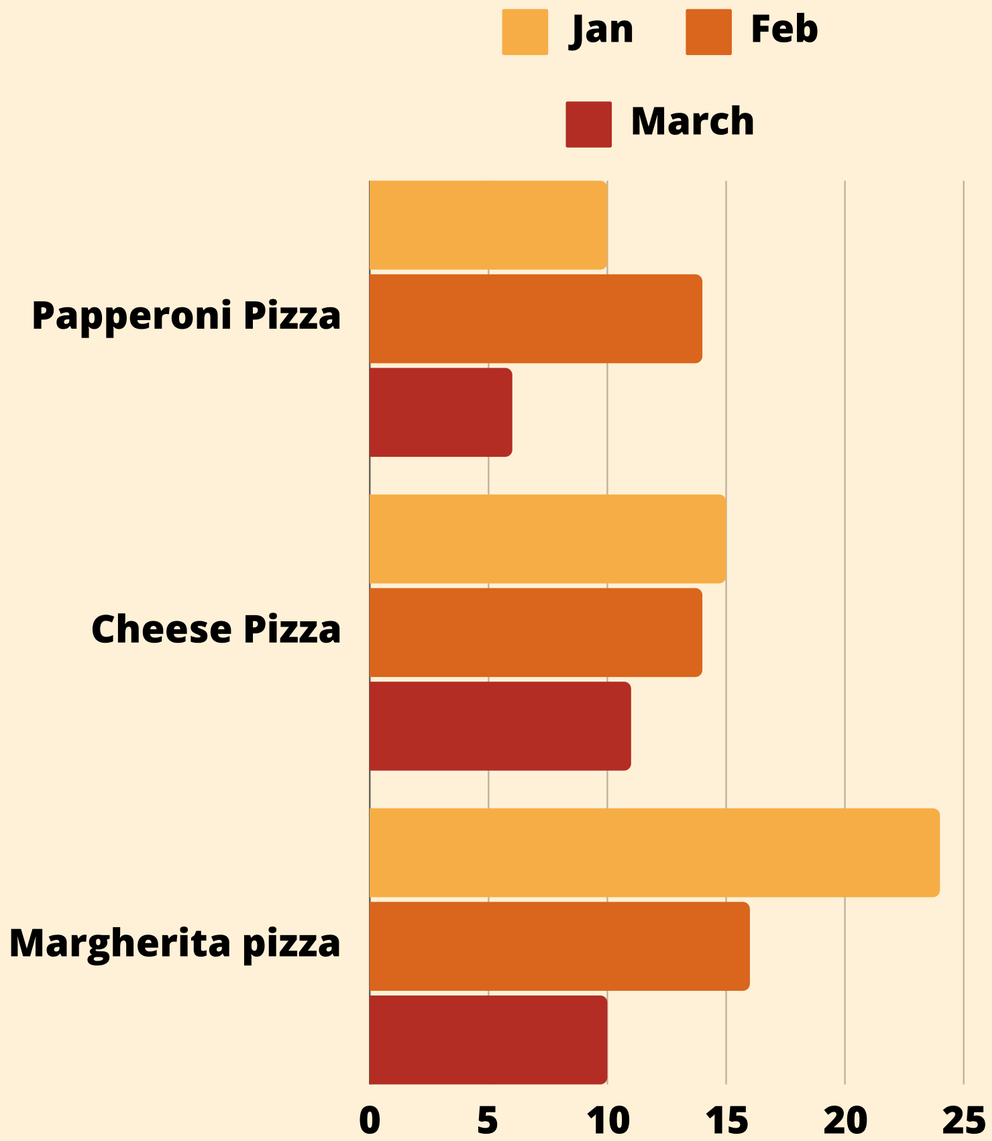
- Orders → Order_Details: One-to-many relationship (one order can have multiple pizza details).
- Order_Details → Pizzas: Many-to-one relationship (each detail links to a specific pizza).
- Pizzas → Pizza_Types: Many-to-one relationship (each pizza is categorized under one type).

PIZZERIA
& CO.

PIZZA SALES

SQL PROJECT

PRESENTED BY
TUSHAR SONI



LEVELS

01. Basic

- Simple retrievals like total orders, revenue, highest-priced pizza.
- Suitable for understanding the foundational data.

02. Intermediate

- Joining tables, analyzing orders by time, and calculating averages.
- Ideal for exploring data relationships and trends.

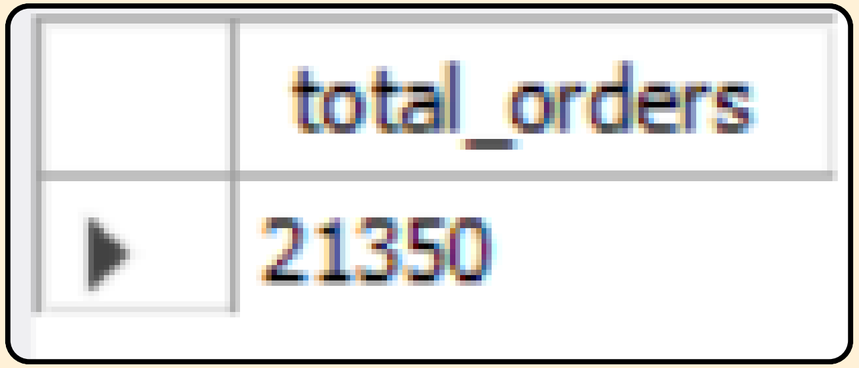
03. Advanced

- Complex queries involving cumulative revenue and percentage contributions.
- Focuses on deeper insights and strategic decisions.

Basic:

1. Retrieve the total number of orders placed.

```sql
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```



total_orders
21350

2. Calculate the total revenue generated from pizza sales.

```sql
SELECT
    ROUND(SUM(p.price * od.quantity), 2) AS revenue
FROM
    pizzas AS p
        JOIN
    order_details AS od ON p.pizza_id = od.pizza_id;
```

	revenue
▶	817860.05

3. Identify the highest-priced pizza.

```sql
SELECT
    pt.name, p.price
FROM
    pizzas AS p
        JOIN
    pizza_types AS pt
    ON p.pizza_type_id = pt.pizza_type_id
ORDER BY p.price DESC
LIMIT 1;
```

name	price
The Greek Pizza	35.95

4. Identify the most common pizza size ordered.

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

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

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

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

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

Intermediate:

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

```sql
SELECT
    SUM(od.quantity) AS total_quantity, pt.category
FROM
    pizzas AS p
        JOIN
    pizza_types AS pt ON p.pizza_type_id = pt.pizza_type_id
        JOIN
    order_details AS od ON p.pizza_id = od.pizza_id
GROUP BY pt.category
ORDER BY total_quantity DESC;
```

total_quantity	category
14888	Classic
11987	Supreme
11649	Veggie
11050	Chicken

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

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

hour_time	total
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

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

```sql
SELECT
    category, COUNT(name) as total
FROM
    pizza_types
GROUP BY category;
```

category	total
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

```sql
with order_table as (
SELECT
    orders.order_date, SUM(order_details.quantity)
    as total_order
FROM orders JOIN order_details
ON orders.order_id = order_details.order_id
GROUP BY orders.order_date
)
select round(avg(total_order))
 as avg_pizza_ordered_per_day
from order_table;
```

avg_pizza_ordered_per_day
138

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

```sql
SELECT
    pt.name, SUM(p.price * od.quantity) AS revenue
FROM
    pizzas p
        INNER JOIN
    order_details od ON p.pizza_id = od.pizza_id
        INNER JOIN
    pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.name
ORDER BY revenue DESC LIMIT 3;
```

name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

Advanced:

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

```sql
SELECT
 pt.category,
 ROUND((((SUM(p.price * od.quantity) /
 (SELECT SUM(p.price * od.quantity)
             FROM pizzas p JOIN  order_details od
             ON p.pizza_id = od.pizza_id)) * 100)),
         2) AS PERCENTAGE
 FROM pizzas p
     INNER JOIN
 order_details od ON p.pizza_id = od.pizza_id
     INNER JOIN
 pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
 GROUP BY pt.category;
```

category	PERCENTAGE
Classic	26.91
Veggie	23.68
Supreme	25.46
Chicken	23.96

2. Analyze the cumulative revenue generated over time.

```sql
with commulative_table as (
SELECT o.order_date, round(sum(od.quantity*p.price),2)
as revenue FROM order_details od
join pizzas p
on p.pizza_id=od.pizza_id
join orders o
on o.order_id=od.order_id
group by o.order_date
order by o.order_date)
select *, round(sum(revenue) over(order by order_date),2)
as cumm_revenue from commulative_table ;
```

order_date	revenue	cumm_revenue
2015-01-01	2713.85	2713.85
2015-01-02	2731.9	5445.75
2015-01-03	2662.4	8108.15
2015-01-04	1755.45	9863.6
2015-01-05	2065.95	11929.55
2015-01-06	2428.95	14358.5
2015-01-07	2202.2	16560.7
2015-01-08	2838.35	19399.05
2015-01-09	2127.35	21526.4
2015-01-10	2463.95	23990.35
2015-01-11	1872.3	25862.65
2015-01-12	1919.05	27781.7
2015-01-13	2049.6	29831.3

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

```sql
with cte as (
select category, name , revenue,
 rank() over(partition by category order by revenue desc)
 as ranks from
(select pizza_types.name, pizza_types.category,
  round(sum(pizzas.price*order_details.quantity),2)
  as revenue from pizzas join pizza_types
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.name, pizza_types.category
order by revenue desc) as a )
select * from cte
where ranks<=3;
```

category	name	revenue	ranks
Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768	2
Chicken	The California Chicken Pizza	41409.5	3
Classic	The Classic Deluxe Pizza	38180.5	1
Classic	The Hawaiian Pizza	32273.25	2
Classic	The Pepperoni Pizza	30161.75	3
Supreme	The Spicy Italian Pizza	34831.25	1
Supreme	The Italian Supreme Pizza	33476.75	2
Supreme	The Sicilian Pizza	30940.5	3
Veggie	The Four Cheese Pizza	32265.7	1
Veggie	The Mexicana Pizza	26780.75	2
Veggie	The Five Cheese Pizza	26066.5	3

CONCLUSION & FURTHER RESOURCES

- Summary:

This project has successfully analyzed pizza sales data to extract valuable insights into sales patterns, revenue generation, and customer preferences.

- Future Work:

Future analysis could involve integrating more data points, such as customer feedback, and applying machine learning models for predictive analytics.

- Connect with Me:
  - LinkedIn: linkedin.com/in/tusharsoni2024/
  - GitHub: github.com/Tusharsoni69 (Link to project repository)
  - Email: tushar2000soni@gmail.com
  - HackerRank: hackerrank.com/profile/tushar_2800

PIZZA SALES SQL PROJECT

Tushar Soni

PROJECT LEAD

GET IN TOUCH WITH US

tushar2000soni@gmail.com

linkedin.com/in/tusharsoni2024/

github.com/Tusharsoni69

THANK YOU!

Thank you for viewing my presentation.
I appreciate your time and look forward to connecting with you.