BONG

PIZZA SALES ANALYSIS USING SQL

Exploring sales, revenue, and customer trends with SQL queries

By

Sampa Barui

PROJECT OVERVIEW

This project analyzes BongPizza’s sales data using SQL to uncover insights about customer preferences, sales trends, and revenue patterns.

Objectives:

- Retrieve key metrics (orders, revenue, best-sellers).
- Analyze order timing and category-wise sales.
- Identify performance drivers for decision-making.

DATABASE SCHEMA

Tables Used in BongPizza Sales Analysis

1. orders

- Columns: order_id, date, time.

2. order_details

- Columns: order_details_id, order_id, pizza_id, quantity

3. pizzas

- Columns: pizza_id, pizza_type_id, size, price

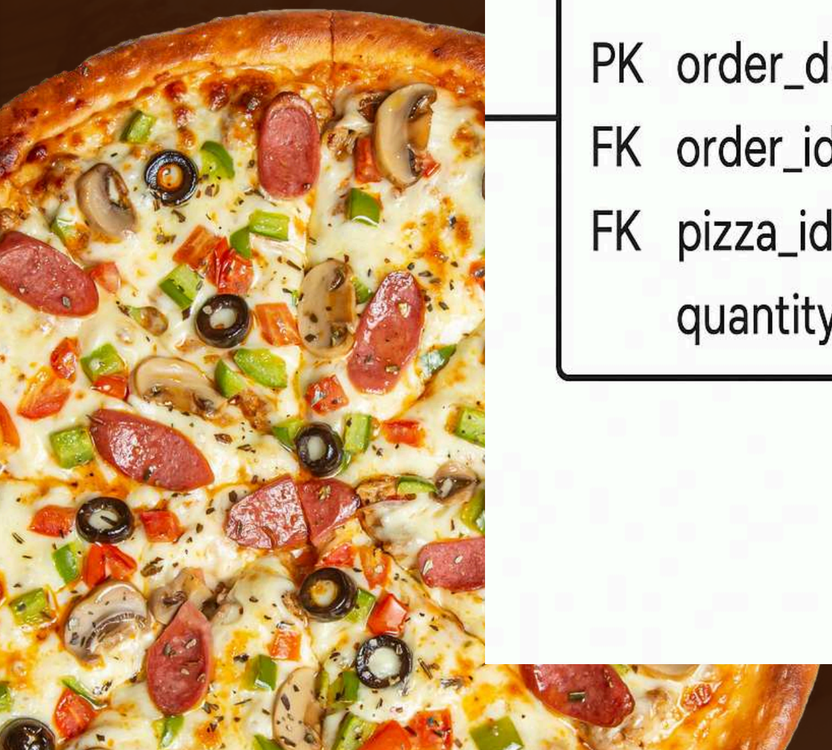
4. pizza_types

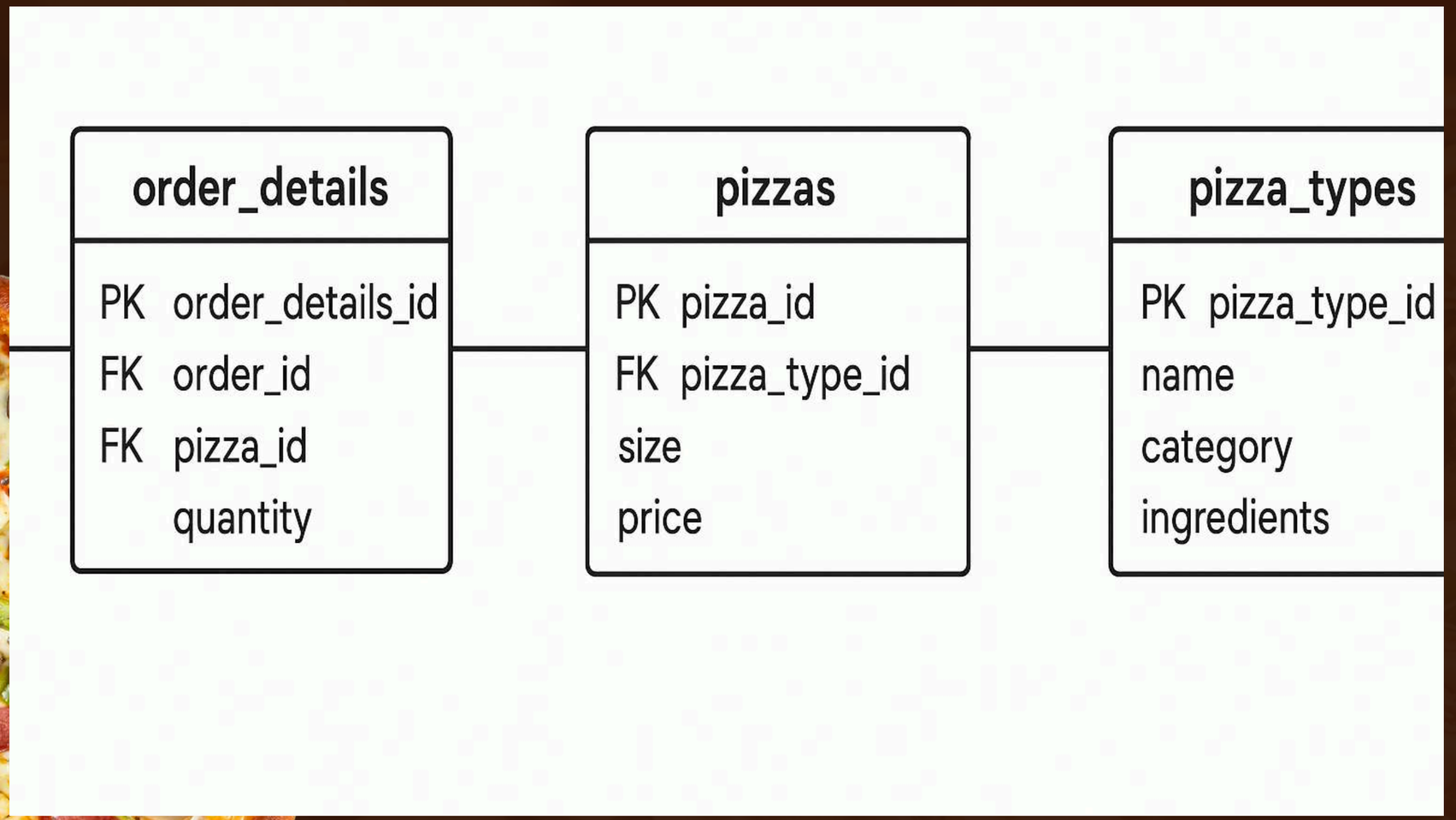
- Columns: pizza_type_id, name, category, ingredients

Primary & Foreign Keys :

- **orders.order_id → PK**
- **order_details.order_id → FK**
- **order_details.pizza_id → FK**
- **pizzas.pizza_type_id → FK**

RELATIONSHIP DIAGRAM





Retrieve the total number of orders placed.

```sql
SELECT
    COUNT(order_id) AS Total_Orders
FROM
    bongpizza.orders;
```

Output

Result Grid

	Total_Orders
▶	21350

Calculate the total revenue generated from pizza sales.

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

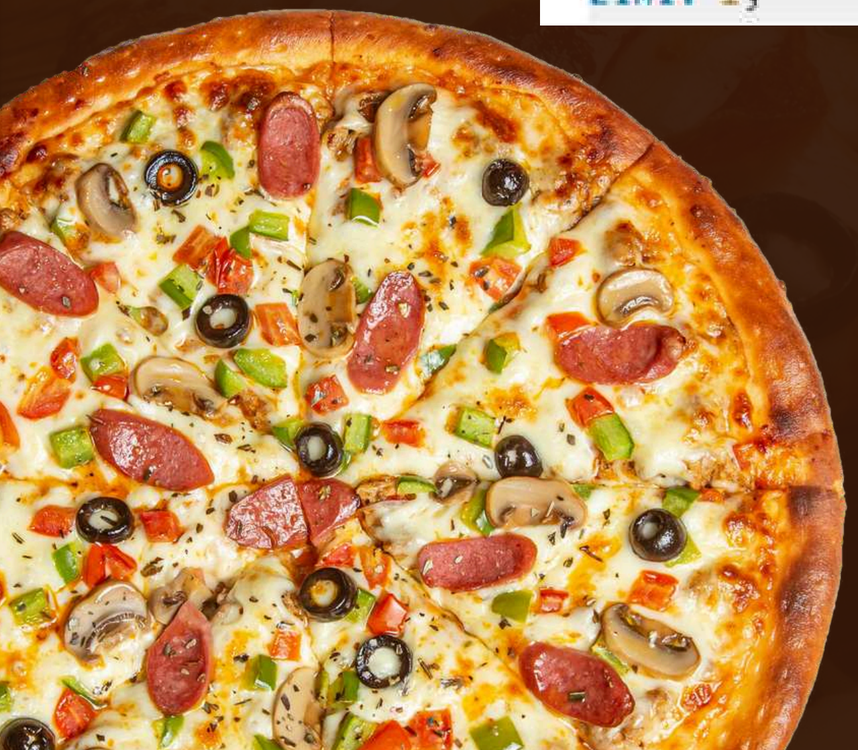
Output

Result Grid

Total_Revenue
44957.15

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;
```



Output

Result Grid | Filter Rows

	name	price
▶	The Greek Pizza	35.95

Identify the most common pizza size ordered

```sql
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	1045
M	839
S	744
XL	34

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

```sql
SELECT
    pizza_types.name, 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.name
ORDER BY Quantity DESC limit 5
;
```

Output

Result Grid | Filter Rows:

name	Quantity
The Pepperoni Pizza	163
The Barbecue Chicken Pizza	131
The California Chicken Pizza	131
The Thai Chicken Pizza	129
The Classic Deluxe Pizza	123

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

```sql
SELECT
    pizza_types.category,
    COUNT(order_details.quantity) AS quantity
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
        JOIN
    pizza_types ON pizza_types.pizza_type_id = pizzas.pizza_type_id
GROUP BY pizza_types.category
ORDER BY quantity DESC
LIMIT 5;
```

Output

Result Grid | Filter

category	quantity
Classic	795
Supreme	658
Veggie	631
Chicken	578

Determine the distribution of orders by hour of the day.

```sql
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

Result 1

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

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

Output

Result Grid | Filter Rows:

category	Category_count
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

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

Output

Result Grid

	average_Qut
▶	133

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

```sql
SELECT
    pizza_types.name,
    ROUND(SUM(order_details.quantity * pizzas.price),
            0) AS revenue
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 revenue
LIMIT 3;
```

Output

Result Grid | Filter Rows:

name	revenue
The Brie Carre Pizza	402
The Calabrese Pizza	682
The Mediterranean Pizza	745

Calculate the percentage contribution of each pizza type to total revenue

```
 select pizza_types.category,round(sum(order_details.quantity*pizzas.price)/
(select ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS Total_Revenue
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id)*100,2) As revenue 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 revenue limit 3;
```

Output

Result Grid | Filter

	category	revenue
▶	Chicken	23.53
	Veggie	24.06
	Supreme	25.74

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(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:

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

Result Grid | Filter Rows:

order_date	cum_revenue
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	44957.15000000001

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((order_details.quantity)*pizzas.price) as revenue 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 ,pizza_types.name  )as a) as b
where rn<=3;
```

Output

Result Grid | Filter Rows:

name	revenue
The Barbecue Chicken Pizza	2334.25
The California Chicken Pizza	2250.25
The Pepperoni Pizza	2076
The Classic Deluxe Pizza	1937.5
The Greek Pizza	1643.5
The Italian Supreme Pizza	2097.75
The Spicy Italian Pizza	1924
The Sicilian Pizza	1898.5
The Five Cheese Pizza	1646.5
The Four Cheese Pizza	1602.8...

KEY INSIGHTS

- Large size pizzas generated most revenue.
- Classic category sold the most units.
- Fridays & weekends had the highest orders.
- BBQ Chicken Pizza was the top-seller across all categories.

RECOMMENDATIONS

- Focus marketing campaigns on top-selling pizzas.
- Offer lunchtime discounts (12–2 PM).
- Increase production for large-sized pizzas.

Seasonal offers during December (sales peak).

CONCLUSION

SQL-based data analysis helped BongPizza uncover key trends, boost operational decisions, and identify customer favorites — demonstrating the power of SQL in real-world business analytics.

THANK YOU

See You Next

sampabarui172@gmail.com