



# SQL PROJECT ON PIZZA SELLS





# INTRODUCTION

## HELLO!

My name is Rishav Kumar. I have utilised SQL query to solve questions that were related to pizza sells

“This project demonstrates my ability to analyze real-world datasets using SQL. I explored sales trends, customer preferences, and revenue drivers in a pizza business context.”

Title: Pizza Sales Insights using SQL

Subtitle: Exploratory Data Analysis Project

Technical skills: SQL, Python, Excel, Power BI, Machine Learning

Role: Aspiring Data Analyst | SQL Enthusiast | MSc Mathematics







# OBJECTIVES:

- Understand pizza sales performance using SQL
- Derive actionable insights to aid business decisions
- Techniques used: Aggregation, Joins, Grouping, Time Analysis



# QUESTIONS:

- Q1. Retrieve the total number of orders placed.
- Q2. Calculate the total revenue generated from pizza sales.
- Q3. Identify the highest-priced pizza.
- Q4. Identify the most common pizza size ordered.
- Q5. List the top 5 most ordered pizza types along with their quantities.
- Q6. Join the necessary tables to find the total quantity of each pizza category ordered.
- Q7. Determine the distribution of orders by hour of the day.
- Q8. Join relevant tables to find the category-wise distribution of pizzas.
- Q9. Group the orders by date and calculate the average number of pizzas ordered per day.
- Q10. Determine the top 3 most ordered pizza types based on revenue.
- Q11. Calculate the percentage contribution of each pizza type to total revenue.
- Q12. Analyze the cumulative revenue generated over time.
- Q13. Determine the top 3 most ordered pizza types based on revenue for each pizza category.



# 1. RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

```
SELECT count(order_id) AS total_orders FROM orders;
```

## 2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

**SELECT**

`ROUND(SUM(order_details.quantity * pizzas.price),2)`

**AS** total\_sales

**FROM**

`order_details`

**JOIN**

`pizzas ON pizzas.pizza_id = order_details.pizza_id;`



### 3. IDENTIFY THE HIGHEST-PRICED PIZZA.

```
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;
```

## 4. IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

```
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;
```



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

```
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;
```



## 6. JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

```
SELECT
    pizza_types.category,
    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
        pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity DESC;
```



## 7. DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

**SELECT**

**HOUR(order\_time) AS hour,**

**COUNT(order\_id) AS order\_count**

**FROM**

**orders**

**GROUP BY HOUR(order\_time);**



## 8. JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

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



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

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



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

```
SELECT
    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
        pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```



## 11. 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_sales  
       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  
   pizzas.pizza_id = order_details.pizza_id  
GROUP BY pizza_types.category  
ORDER BY revenue DESC;
```



## 12. ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```
select order_date, sum(revenue) over(order by order_date)
as cumulative_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;
```



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

```
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;
```



# TOOLS & SKILLS USED

\$6

- MySQL
- Data Cleaning & Joins
- Data Aggregation

\$6

## Connect With Me

- [<https://www.linkedin.com/in/rishav-kumar-seth/>]
- [<https://github.com/rishavkumarseth2507>]
- [[rishavkumarseth2507@gmail.com](mailto:rishavkumarseth2507@gmail.com)]

\$6

LARANA PIZZA





THANK YOU!

