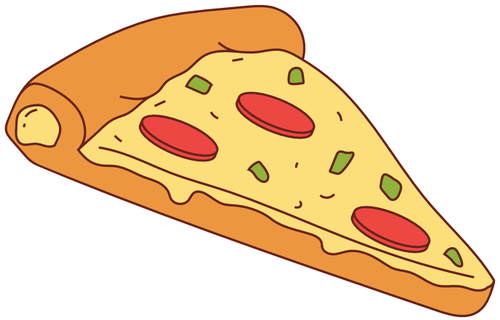
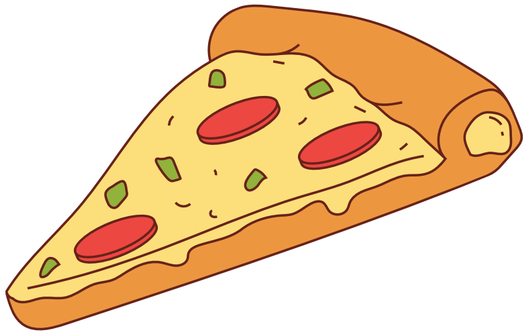
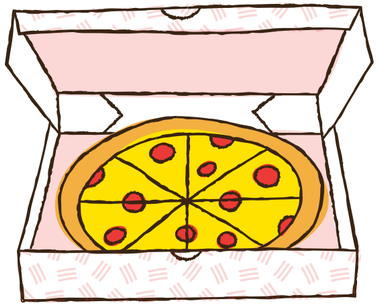
# SQL PROJECT ON PIZZA SALES



# Introduction

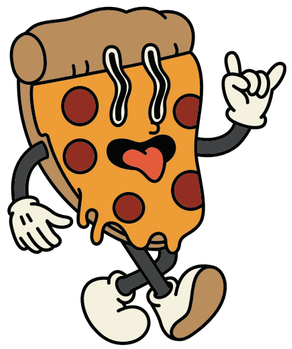


Hello, My name is Siddhi Anil Pawar. and in these project I have Utilized sql queries to solve questions that were related to pizza sales.



# Used Databases

- order_details
- orders
- pizza_types
- pizzas



# List of Questions

## Basic:

1)Retrieve the total number of orders placed.

2)Calculate the total revenue generated from pizza sales.

3)Identify the highest-priced pizza.

4)Identify the most common pizza size ordered.

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

## Intermediate:

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

7)Determine the distribution of orders by hour of the day.

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

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

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

# Advanced:

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

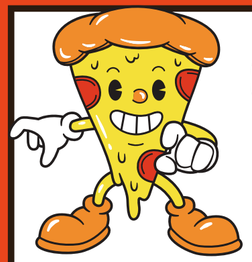
12)Analyze the cumulative revenue generated over time.

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

# Retrieve the total number of orders placed.

```
create database pizzahut;
use pizzahut;
select * from orders;
select count(order_id) as total_orders from orders;
```

| total_orders |
| --- |
| 21350 |



# Calculate the total revenue generated from pizza sales.

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

| | total_revenue |
|---|---|
| ▶ | 817860.05 |

# Identify the highest-priced pizza.

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

| | name | price |
|---|---|---|
| ▶ | The Greek Pizza | 35.95 |

# Identify the most common pizza size ordered.

```sql
SELECT
    p.size, COUNT(od.order_details_id) AS total_sales
FROM
    pizzas p
        JOIN
    order_details od ON p.pizza_id = od.pizza_id
GROUP BY p.size
ORDER BY total_sales DESC
LIMIT 1;
```

| | size | total_sales |
|---|---|---|
| ▶ | L | 18526 |

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

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

| name | total_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.

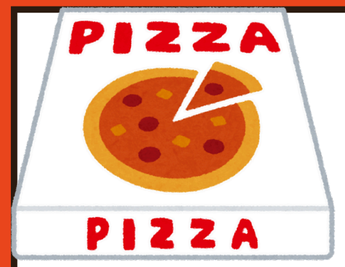
```sql
SELECT pt.category, SUM(od.quantity) AS total_qty
FROM pizza_types pt
JOIN pizzas p ON pt.pizza_type_id = p.pizza_type_id
JOIN order_details od ON od.pizza_id = p.pizza_id
GROUP BY pt.category
ORDER BY total_qty DESC;
```

| | category | total_qty |
|---|---|---|
| ▶ | Classic | 14888 |
| | Supreme | 11987 |
| | Veggie | 11649 |
| | Chicken | 11050 |

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

```
SELECT
    COUNT(order_id) AS total_orders, HOUR(time) AS hours
FROM
    orders
GROUP BY HOUR(time);
```

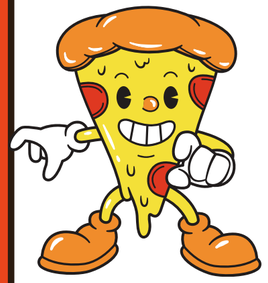
| total_orders | hours |
|---|---|
| 1231 | 11 |
| 2520 | 12 |
| 2455 | 13 |
| 1472 | 14 |
| 1468 | 15 |
| 1920 | 16 |
| 2336 | 17 |




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

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

| | total_pizzas | category |
|---|---|---|
| ▶ | 6 | Chicken |
| | 8 | Classic |
| | 9 | Supreme |
| | 9 | Veggie |



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

```sql
SELECT ROUND(AVG(total_qty), 0) as avg_pizza_order
FROM
(SELECT o.date, SUM(od.quantity) AS total_qty
FROM orders o
JOIN order_details od ON od.order_id = o.order_id
GROUP BY o.date) AS oreder_qty;
```

| | avg_pizza_order |
|---|---|
| ▶ | 138 |

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

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

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

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

| | category | revenue |
|---|---|---|
| ▶ | Classic | 26.91 |
| | Supreme | 25.46 |
| | Chicken | 23.96 |
| | Veggie | 23.68 |

# Analyze the cumulative revenue generated over time.

```
select date,sum(revenue) over(order by date) as cum_revenue from
(select o.date, sum(od.quantity*p.price) as revenue
from order_details od join pizzas p
on od.pizza_id=p.pizza_id join orders o
on o.order_id=od.order_id group by o.date) as sales;
```

| | 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 |

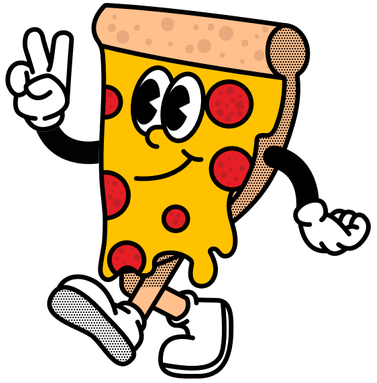
# Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```
select category,name,revenue from
(select category , name , revenue,
rank() over(partition by category order by revenue desc) as rn from
(select pt.name,pt.category, sum(od.quantity*p.price) as revenue
from pizza_types pt join pizzas p on pt.pizza_type_id=p.pizza_type_id
join order_details od on od.pizza_id=p.pizza_id
group by pt.name,pt.category) as rev_cat) as b
where rn <=3;
```

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

# Conclusion

The analysis of pizza sales data using SQL has provided valuable insights into customer preferences, popular toppings, peak ordering times, and revenue trends. Through examining sales transactions, we identified the top-selling pizzas, which can inform marketing strategies and menu optimization. Additionally, analyzing order timestamps revealed peak hours. Overall, this SQL project demonstrates the power of data analysis in informing business decisions within the pizza industry.



# Thank you!!