Introduction
Pizza Sales Project using MySQL

- **Analyze pizza sales data using MySQL to identify key business challenges and develop actionable solutions.**

Objective

Business Problems Addressed:

- Retrieve the total number of orders placed.
- Calculate the total revenue generated from pizza sales.
- Identify the highest-priced pizza.
- Identify the most common pizza size ordered.
- List the top 5 most ordered pizza types along with their quantities.
- Join the necessary tables to find the total quantity of each pizza category ordered.

Objective

Business Problems Addressed:

- Determine the distribution of orders by hour of the day.
- Join relevant tables to find the category-wise distribution of pizzas.
- Group the orders by date and calculate the average number of pizzas ordered per day.
- Determine the top 3 most ordered pizza types based on revenue.
- Calculate the percentage contribution of each pizza type to total revenue.
- Analyze the cumulative revenue generated over time.
- Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```sql
-- Retrieve the total number of orders placed

select count(order_id) as total_order from orders;
```

Result Grid

	total_order
▶	21350

```sql
-- Calculate the total revenue generated from pizza sales.

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

Result Grid

	total_revenue
	817860.05

```sql
-- Identify the highest-priced pizza.

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

Result Grid | Filter

name	price
The Greek Pizza	35.95

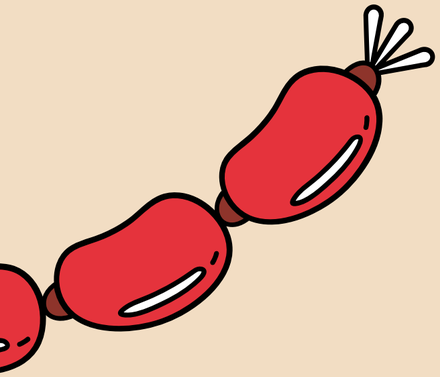
```sql
-- Identify the most common pizza size ordered.

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

esult Grid | Filter Rows

size	ordered_pizza_count
L	18526

```sql
-- List the top 5 most ordered pizza types along with their quantities.
SELECT
    pizza_types.name AS pizza_name,
    SUM(order_details.quantity) AS total_qty
FROM
    pizza_types
        INNER JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        INNER JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_name
ORDER BY total_qty DESC
LIMIT 5;
```

pizza_name	total_qty
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371



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

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

category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

```sql
-- Determine the distribution of orders by hour of the day.

SELECT
    HOUR(order_time) AS hour, COUNT(order_id) AS count
FROM
    orders
GROUP BY hour
ORDER BY count DESC;
```

hour	count
12	2520
13	2455
18	2399
17	2336
19	2009
16	1920
20	1642

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

SELECT
    category, COUNT(name) AS total_pizza
FROM
    pizza_types
GROUP BY category
ORDER BY total_pizza DESC;
```

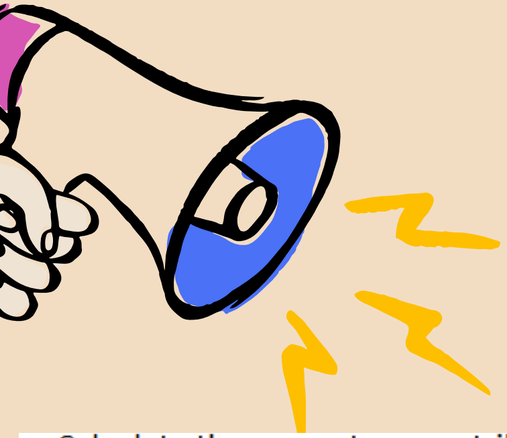
category	total_pizza
Supreme	9
Veggie	9
Classic	8
Chicken	6

```sql
-- Group the orders by date and calculate the average number of pizzas ordered per day.

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

avg_pizzas_ordered
138

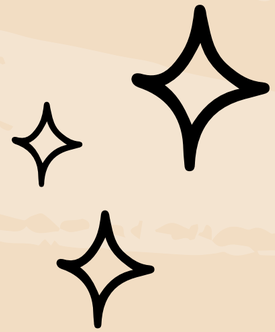
```sql
-- Determine the top 3 most ordered pizza types based on revenue.

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

pizza_type	total_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.

SELECT
    pizza_types.category AS category,
    ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
                ROUND(SUM(order_details.quantity * pizzas.price),
                        0) AS total_sales
            FROM
                pizzas
                    INNER JOIN
                order_details ON pizzas.pizza_id = order_details.pizza_id) * 100,
        2) AS revenue
FROM
    pizza_types
        INNER JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        INNER JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY category
ORDER BY revenue DESC;
```

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68



```sql
-- Analyze the cumulative revenue generated over time.

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
    orders
        INNER JOIN
    order_details ON orders.order_id = order_details.order_id
        INNER JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
GROUP BY orders.order_date) as sales;
```

order_date	cum_revenue
2015-01-01	2713.850000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55

```sql
-- 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 pizza_types.category, pizza_types.name, sum(order_details.quantity*pizzas.price) as revenue
from pizza_types inner join pizzas on pizza_types.pizza_type_id = pizzas.pizza_type_id inner 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;
```

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

Thank You!

Email
dubey.srk@gmail.com

linkedin
https://www.linkedin.com/in/saurabh-kumar-dubey-8437b630/

Github
https://github.com/sdubey0402

Phone Number
8104346108