# SQL PROJECT BASED ON PIZZA SALES

PRESENTED BY: LINU LALACHAN

# INTRODUCTION

Hello! I’m Linu Lalachan. In this project, I've utilized SQL queries to address questions concerning pizza sales.

# QUESTIONS REGARDING PIZZA SALES

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.
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.

# QUESTIONS REGARDING PIZZA SALES

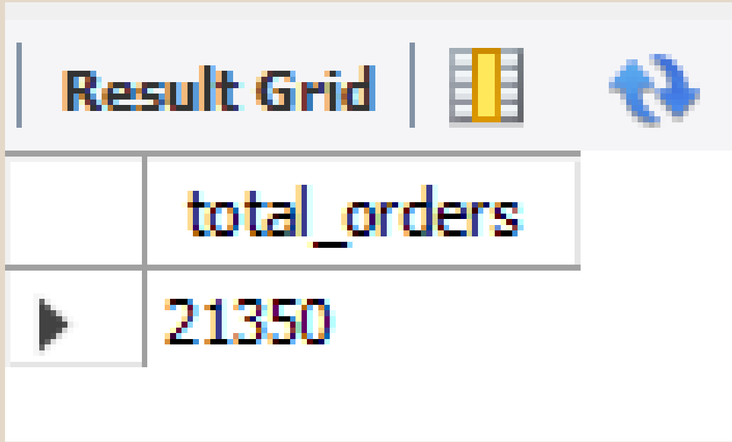
8. Join relevant tables to find the category-wise distribution of pizzas.
9. Group the orders by date and calculate the average number of pizzas ordered per day.
10. Determine the top 3 most ordered pizza types based on revenue.
11. Calculate the percentage contribution of each pizza type to total revenue.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

# 1. Retrieve the total number of orders placed.

INPUT/QUERY:

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

OUTPUT/RESULT:



# 2. Calculate the total revenue generated from pizza sales.

## INPUT/QUERY:

```
SELECT
    ROUND(SUM(order_details_id.order_quantity * pizzas.price),
            2) AS total_sales
FROM
    order_details_id
        JOIN
    pizzas ON order_details_id.pizza_id = pizzas.pizza_id;
```

## OUTPUT/RESULT:

Result Grid

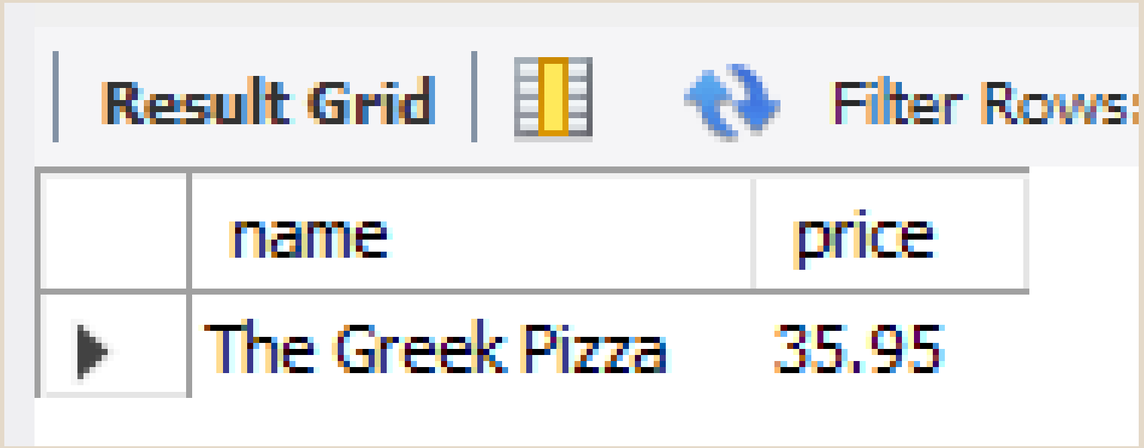
| | total_sales |
|---|---|
| ▶ | 739478.75 |

# 3. Identify the highest-priced pizza.

## INPUT/QUERY:

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

## OUTPUT/RESULT:




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

# 4. Identify the most common pizza size ordered.

## INPUT/QUERY:

```
SELECT
    pizzas.size,
    COUNT(order_details_id.order_quantity) AS most_common_pizza_size_ordered
FROM
    pizzas
        JOIN
    order_details_id ON pizzas.pizza_id = order_details_id.pizza_id
GROUP BY size order by most_common_pizza_size_ordered  DESC;
```

## OUTPUT/RESULT:

Result Grid | Filter Rows:

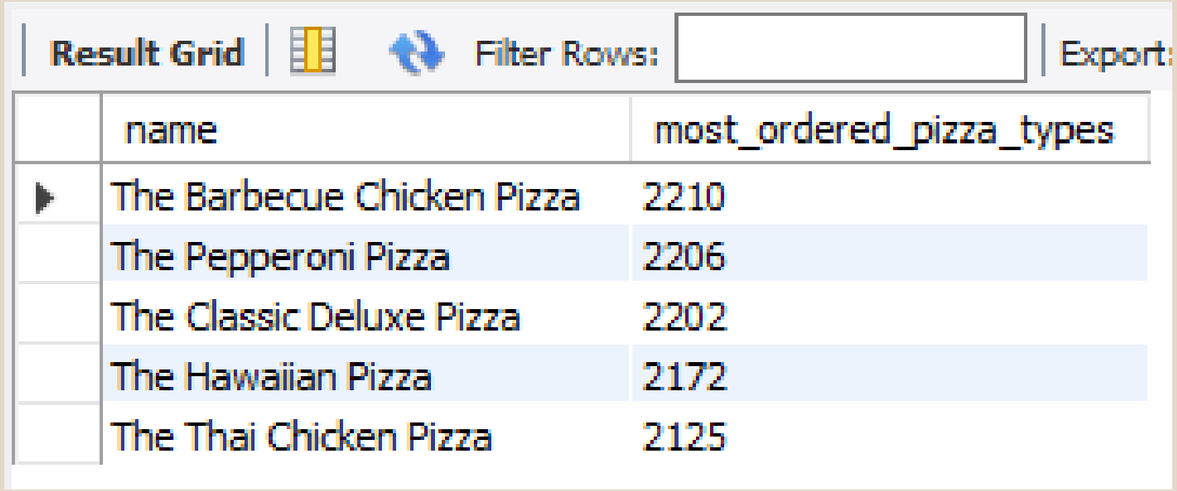
| size | most_common_pizza_size_ordered |
|---|---|
| L | 16730 |
| M | 13926 |
| S | 12774 |
| XL | 501 |
| XXL | 27 |

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

## INPUT/QUERY:

```sql
SELECT
    pizza_types.name,
    SUM(order_details_id.order_quantity) AS most_ordered_pizza_types
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details_id ON order_details_id.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY most_ordered_pizza_types DESC
LIMIT 5;
```

## OUTPUT/RESULT:



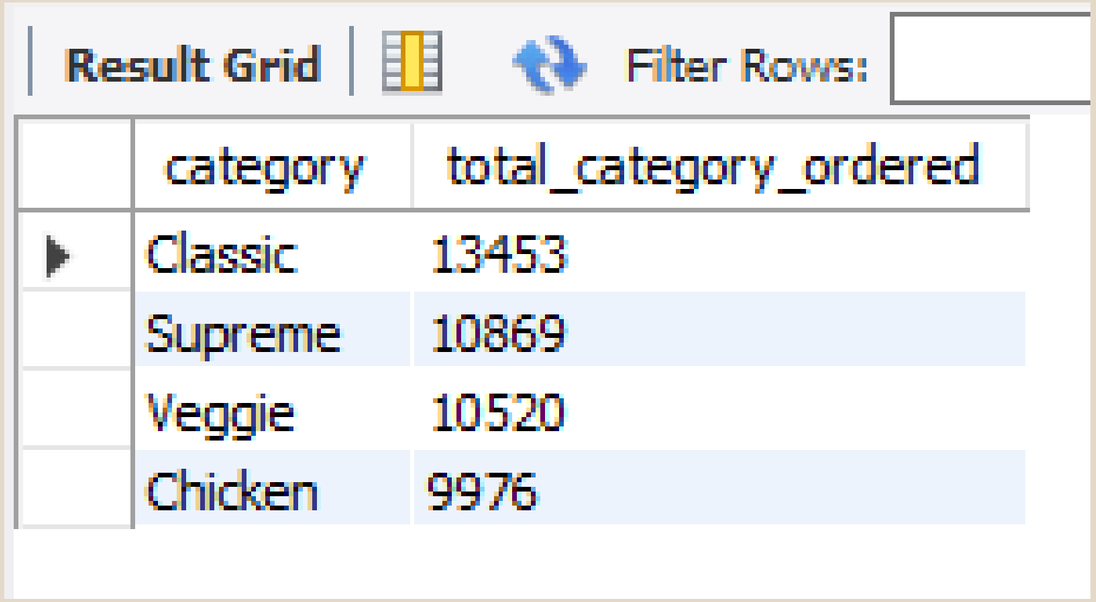

| name | most_ordered_pizza_types |
|---|---|
| The Barbecue Chicken Pizza | 2210 |
| The Pepperoni Pizza | 2206 |
| The Classic Deluxe Pizza | 2202 |
| The Hawaiian Pizza | 2172 |
| The Thai Chicken Pizza | 2125 |

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

## INPUT/QUERY:

```sql
SELECT
    pizza_types.category,
    SUM(order_details_id.order_quantity) AS total_category_ordered
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details_id ON pizzas.pizza_id = order_details_id.pizza_id
GROUP BY category
ORDER BY total_category_ordered DESC;
```

## OUTPUT/RESULT:



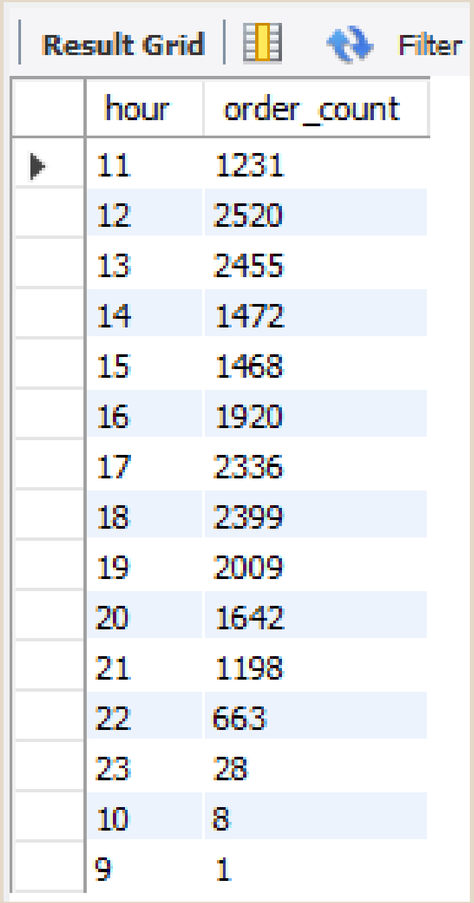

| category | total_category_ordered |
|---|---|
| Classic | 13453 |
| Supreme | 10869 |
| Veggie | 10520 |
| Chicken | 9976 |

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

## INPUT/QUERY:

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

## OUTPUT/RESULT:



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 |
| 21 | 1198 |
| 22 | 663 |
| 23 | 28 |
| 10 | 8 |
| 9 | 1 |

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

## INPUT/QUERY:

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

## OUTPUT/RESULT:

Result Grid | Filter Rows:

| category | categorywise_distributionof_pizzas |
|---|---|
| Chicken | 6 |
| Classic | 8 |
| Supreme | 9 |
| Veggie | 9 |

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

## INPUT/QUERY:

```sql
SELECT
    AVG(total_quantity) AS avg_pizzas_ordered_in_a_day
FROM
    (SELECT
        order_date,
            SUM(order_details_id.order_quantity) AS total_quantity
    FROM
        order_details_id
    JOIN orders ON order_details_id.order_id = orders.order_id
    GROUP BY order_date) AS quantity;
```

## OUTPUT/RESULT:

Result Grid | Filter Rows:

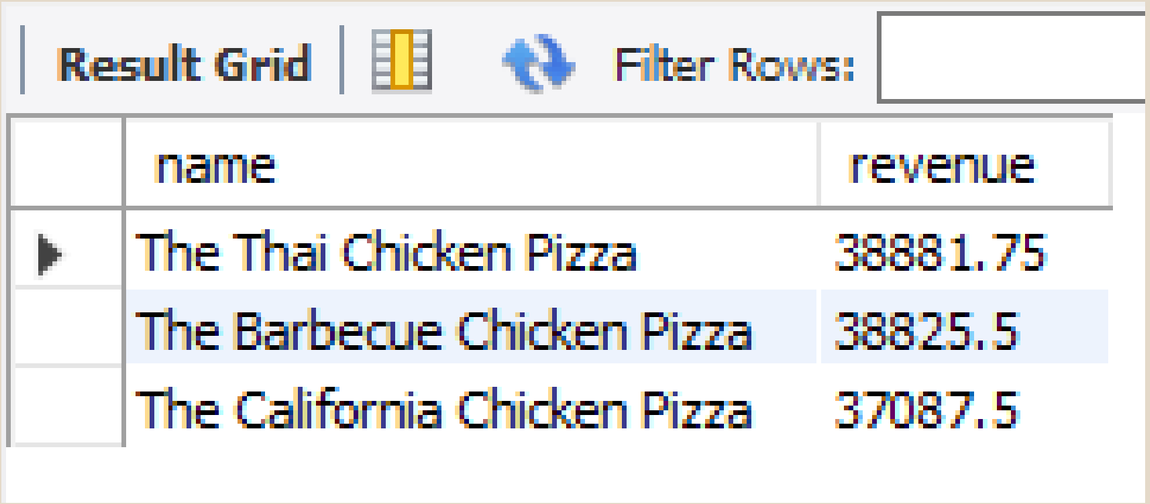
| avg_pizzas_ordered_in_a_day |
| --- |
| 138.3272 |

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

## INPUT/QUERY:

```
SELECT
    name,
    SUM(order_details_id.order_quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details_id ON order_details_id.pizza_id = pizzas.pizza_id
GROUP BY name
ORDER BY revenue DESC
LIMIT 3;
```

## OUTPUT/RESULT:



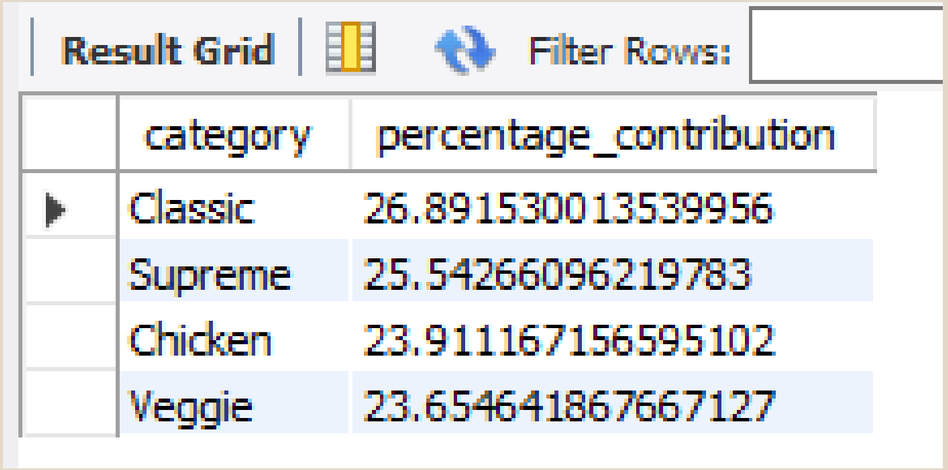

| name | revenue |
|---|---|
| The Thai Chicken Pizza | 38881.75 |
| The Barbecue Chicken Pizza | 38825.5 |
| The California Chicken Pizza | 37087.5 |

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

## INPUT/QUERY:

```
SELECT
    pizza_types.category,
    (SUM(order_details_id.order_quantity * pizzas.price) / (SELECT
            ROUND(SUM(order_details_id.order_quantity * pizzas.price),
                    2)
        FROM
            order_details_id
                JOIN
            pizzas ON order_details_id.pizza_id = pizzas.pizza_id)) * 100 AS percentage_contribution
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details_id ON order_details_id.pizza_id = pizzas.pizza_id
GROUP BY category
ORDER BY percentage_contribution DESC;
```

## OUTPUT/RESULT:



Result Grid | Filter Rows:

| category | percentage_contribution |
|---|---|
| Classic | 26.891530013539956 |
| Supreme | 25.54266096219783 |
| Chicken | 23.911167156595102 |
| Veggie | 23.6546418676671 27 |

# 12. Analyze the cumulative revenue generated over time.

## INPUT/QUERY:

```
select order_date,sum(total_revenue_generated_in_a_day) over(order by order_date) as cum_revenue from
(select order_date,round(sum(order_quantity*price),2) as total_revenue_generated_in_a_day
from order_details_id join pizzas
on order_details_id.pizza_id = pizzas.pizza_id
join orders
on order_details_id.order_id = orders.order_id
group by order_date) as cum_sales;
```

## OUTPUT/RESULT:

Result Grid | Filter Rows:

| order_date | cum_revenue |
|---|---|
| 2015-01-01 | 2713.85 |
| 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 | 21526.399999999998 |
| 2015-01-10 | 23990.35 |
| 2015-01-11 | 25862.649999999998 |
| 2015-01-12 | 27781.699999999997 |
| 2015-01-13 | 29831.299999999996 |
| 2015-01-14 | 32358.699999999997 |
| 2015-01-15 | 34343.5 |
| 2015-01-16 | 36937.65 |
| 2015-01-17 | 39001.75 |
| 2015-01-18 | 40978.6 |
| 2015-01-19 | 43365.75 |

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

## INPUT/QUERY:

```
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_quantity*price) as revenue
from order_details_id join pizzas
on order_details_id.pizza_id = pizzas.pizza_id
join pizza_types
on pizza_types.pizza_type_id = pizzas.pizza_type_id
group by pizza_types.category,pizza_types.name order by revenue) as sales) as pizza_sales
where rn<=3;
```

## OUTPUT/RESULT:

Result Grid | Filter Rows: | Export:

| category | name | revenue |
|---|---|---|
| Chicken | The Thai Chicken Pizza | 38881.75 |
| Chicken | The Barbecue Chicken Pizza | 38825.5 |
| Chicken | The California Chicken Pizza | 37087.5 |
| Classic | The Classic Deluxe Pizza | 34221.5 |
| Classic | The Hawaiian Pizza | 28926 |
| Classic | The Pepperoni Pizza | 27514.5 |
| Supreme | The Spicy Italian Pizza | 31558.25 |
| Supreme | The Italian Supreme Pizza | 30433.25 |
| Supreme | The Sicilian Pizza | 27751.25 |
| Veggie | The Four Cheese Pizza | 29039.55000000055 |
| Veggie | The Mexicana Pizza | 24124 |
| Veggie | The Five Cheese Pizza | 24031.5 |