

# Presentation



By Abhishek Sharma





In this project for pizza sales I am looking to analyze its sales data to gain insights into various aspects of its business performance, including product popularity, peak sales periods, and order trends.

The primary goal of this project is to analyze pizza sales data using SQL to generate actionable insights that can inform strategic decisions such as inventory management, pricing, marketing, and promotion efforts. The dataset includes sales information like pizza types, quantities sold, sales amounts, dates, pizza category and times of orders.



#### Basic:

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.

#### Intermediate:

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

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.

#### Advanced:

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.



### Retrive the total number of orders placed



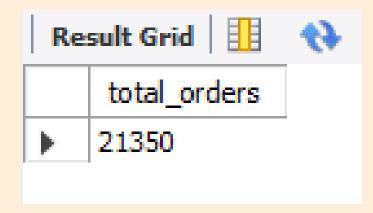


```
SELECT

COUNT(order_id) AS total_orders

FROM

orders;
```



# Calculate the total revenue generated from Pizza sales



```
SELECT

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

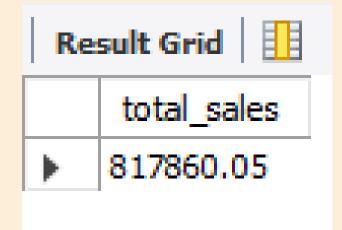
2) AS total_sales

FROM

order_details

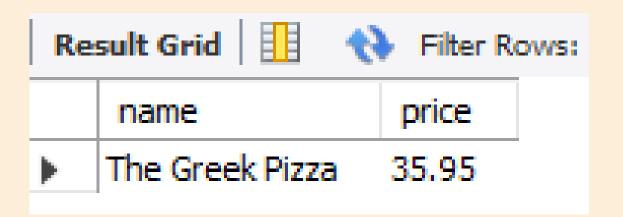
JOIN

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



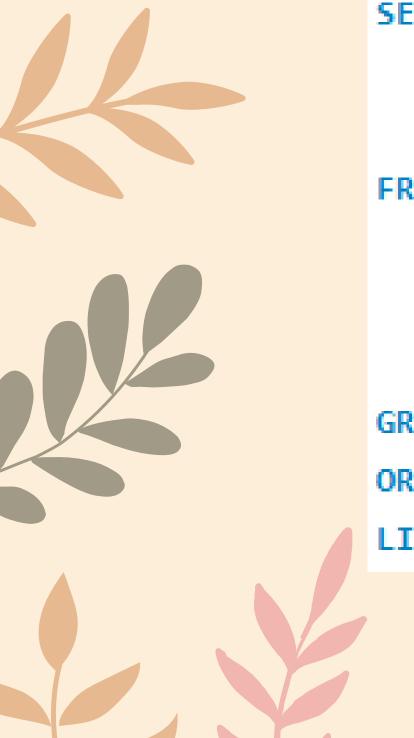
```
Identify the highest-priced pizza.
```

```
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;
```

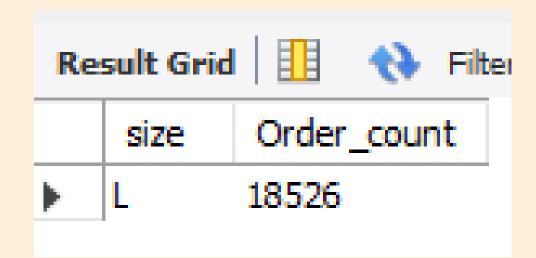




### Identify the most common pizza size ordered.



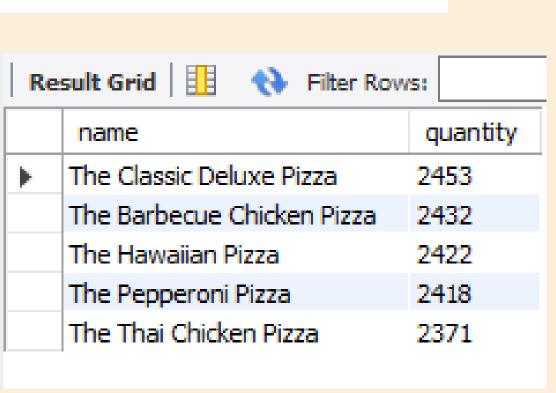
```
SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) AS Order_count
FROM
    pizzas
        JOIN
   order_details ON pizzas.pizza_id = order_details.Pizza_id
GROUP BY pizzas.size
ORDER BY order count DESC
LIMIT 1;
```



### 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;
```





# 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 pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
   order_details ON order_details.Pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity DESC;
```

	category	quantity
<b>&gt;</b>	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

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





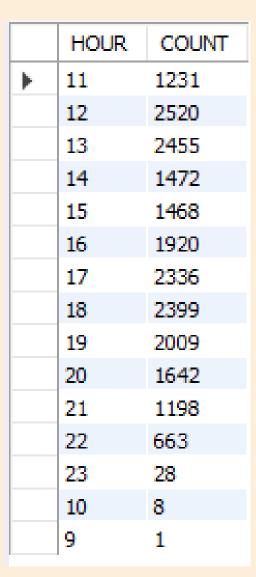
#### SELECT

HOUR(order\_time) AS HOUR, COUNT(order\_id) AS COUNT

FROM

orders

GROUP BY HOUR;





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





```
SELECT

COUNT(pizza_types.name)

FROM

pizza_types

GROUP BY category;
```

	count(pizza_types.name)
•	6
	8
	9
	9
	2

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

```
SELECT

ROUND(AVG(quantity), 0) as avg_pizza_ordered_perday

FROM

(SELECT

orders.order_date, SUM(order_details.quantity) AS quantity

FROM

orders

JOIN order_details ON order_details.order_id = orders.order_id

GROUP BY orders.order_date) AS order_quantity;
```

	avg_pizza_ordered_perday
N	120

### 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 pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
   order_details ON order_details.Pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY Revenue DESC
LIMIT 3;
```

	name	revenue
<b>)</b>	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 NAME,
    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 order_details.Pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY REVENUE DESC;
```



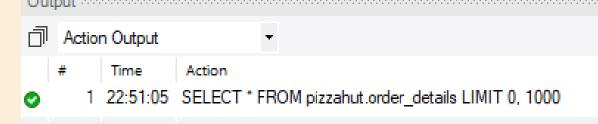
	· — ·	
	NAME	REVENUE
<b>&gt;</b>	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

### Analyze the cumulative revenue generated over time.

```
Select
   order_date,
   ROUND (sum(revenue) over(order by order_date),2) as Cum_revenue
   from
orders.order_date,
   sum(order_details.quantity * pizzas.price) as revenue
  From
   order_details
   Join
   pizzas
   pizzas.pizza_id = order_details.pizza_id
   Join
   Orders
   ON
  Orders.order_id = order_details.order_id
  Group By
   orders.order_date) as Sales;
```



Re	sult Grid 🛮 🔢	Filter Rows:
	order_date	Cum_revenue
<b>&gt;</b>	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



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

```
Select
  Name, Category, 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;
```

Result Grid			
	Name	Category	revenue
<b>)</b>	The Thai Chicken Pizza	Chicken	43434.25
	The Barbecue Chicken Pizza	Chicken	42768
	The Classic Deluxe Pizza	Classic	38180.5
	The Hawaiian Pizza	Classic	32273.25
	The Spicy Italian Pizza	Supreme	34831.25
	The Italian Supreme Pizza	Supreme	33476.75
	The Four Cheese Pizza	Veggie	32265.70000000065
	The Mexicana Pizza	Veggie	26780.75
	•		



