

# *SQL Project* *On* *Pizza Sales*



***Biplab Bhunia***





Hello

*"In this project, I conducted a comprehensive analysis of **pizza sales** data using SQL. The objective was to extract meaningful insights by crafting and executing SQL queries to answer specific business-related questions."*





# Questions:

## **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.*



# Database Description

Created a database “pizzahut”  
contains 4 tables:

**pizzas,pizza\_type,orders,order\_details.**







Queries ->



***Retrieve the total number of orders placed.***

**SELECT**

COUNT(order\_id) **AS** total\_orders

**FROM** orders;

Result Grid				F
	total_orders			
▶	21350			



***Calculate the total revenue generated from pizza sales.***

**SELECT**

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

**AS** total\_revenue

**FROM** order\_details

**JOIN**

pizzas **ON** order\_details.pizza\_id = pizzas.pizza\_id ;





Result Grid	
	total_revenue
▶	817860.05



# *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 price DESC LIMIT 1;
```



Result Grid     Filter Rows:		
	name	price
▶	The Greek Pizza	35.95



# Identify the most common pizza size ordered.

```
SELECT
    pizzas.size, COUNT(order_details.quantity) 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;
```

Result Grid			Filter R
	size	order_count	
▶	L	18526	





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

```
SELECT
    pizza_types.name , SUM(order_details.quantity) AS quantities
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 quantities DESC LIMIT 5;
```



	name	quantities
▶	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.

```
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 order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category;
```



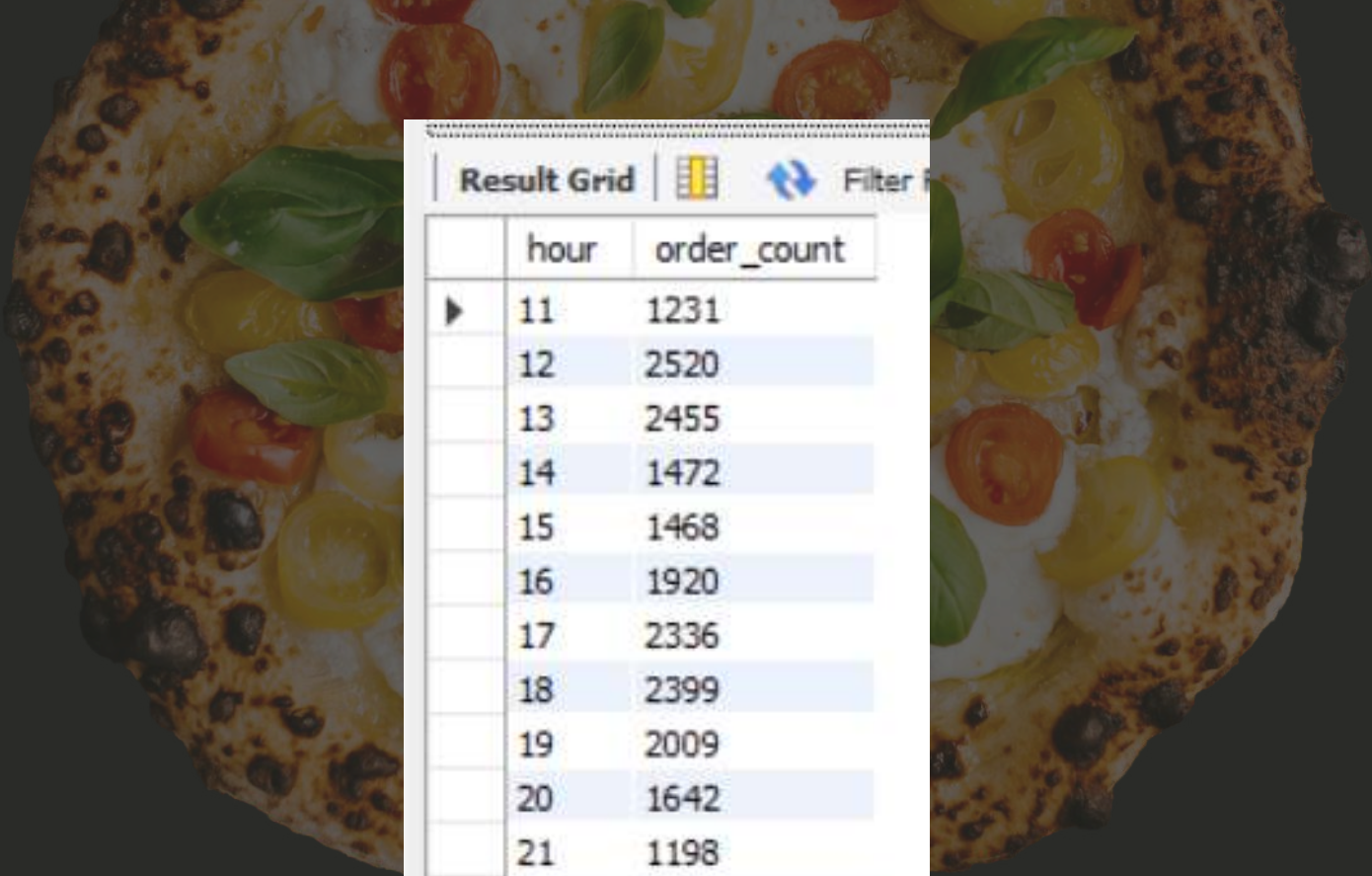
The screenshot shows a database query result grid with the following data:

	category	Quantity
▶	Classic	14888
	Veggie	11649
	Supreme	11987
	Chicken	11050



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

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





Result Grid

	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

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

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

Result Grid     Filter Rows:		
	category	COUNT(name)
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9



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

**SELECT**

**ROUND**(**AVG**(quantity),0) **AS** avg\_pizza\_order\_per\_day

**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 ;

Result Grid



Filter Rows

	avg_pizza_order_per_day
▶	138

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

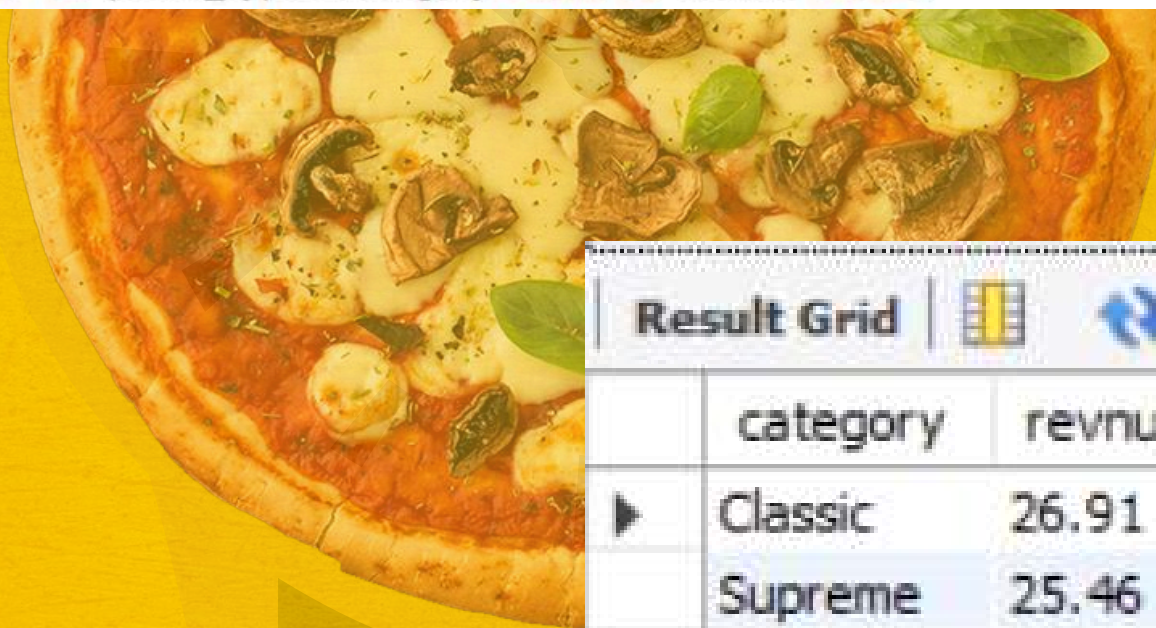
```
SELECT
    pizza_types.name ,
SELECT
    pizza_types.name ,
        SUM(order_details.quantity * pizzas.price) AS revnue
FROM
    pizza_types JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
    JOIN
        order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name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

Result Grid			Filter Rows:	
	name	revnue		
▶	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 ,
    ROUND((SUM(order_details.quantity * pizzas.price) / (SELECT SUM(order_details.quantity * pizzas.price)
FROM
    order_details JOIN pizzas
ON order_details.pizza_id = pizzas.pizza_id) * 100 ),2) AS revnue
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 revnue DESC ;
```

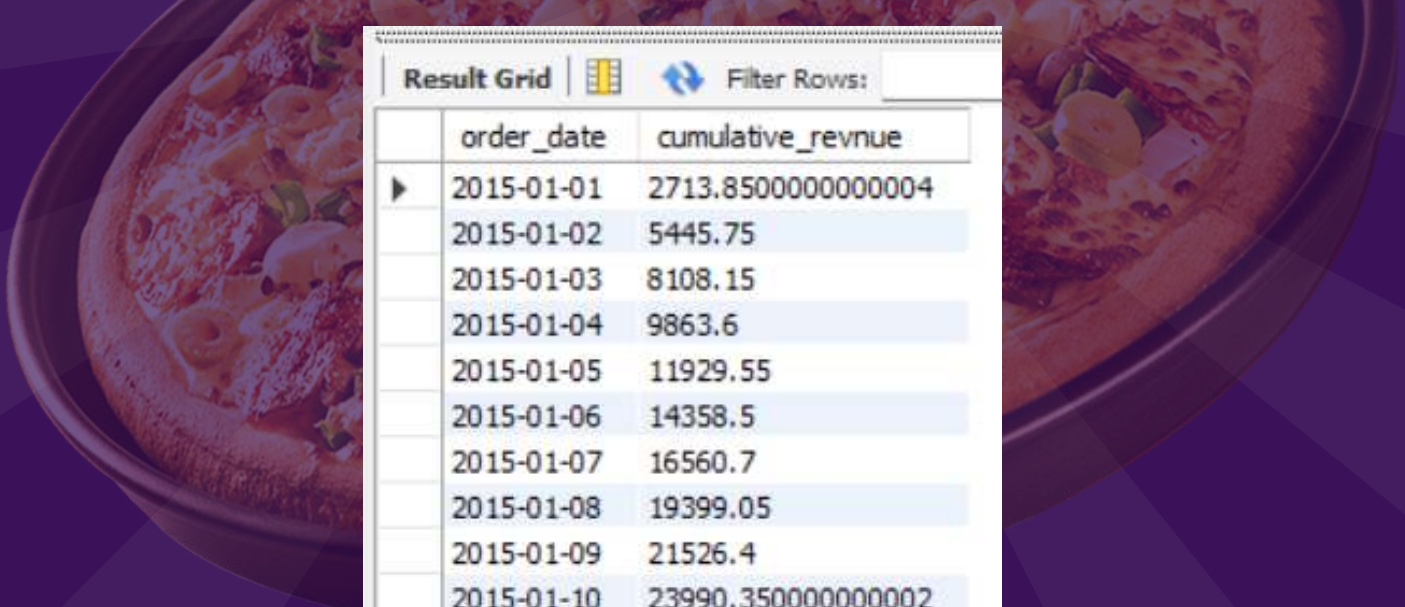


Result Grid			Filter
	category	revnue	
▶	Classic	26.91	
	Supreme	25.46	
	Chicken	23.96	
	Veggie	23.68	



# Analyze the cumulative revenue generated over time.

```
SELECT
    order_date , SUM(revnue) OVER (ORDER BY order_date) AS cumulative_revnue
FROM
    (SELECT orders.order_date , SUM(order_details.quantity * pizzas.price) AS revnue
FROM orders
    JOIN
    order_details ON orders.order_id = order_details.order_id
    JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
GROUP BY orders.order_date ) AS sales
GROUP BY order_date ;
```




Result Grid | Filter Rows:

	order_date	cumulative_revnue
▶	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
	2015-01-09	21526.4
	2015-01-10	23990.350000000002
	2015-01-11	25862.65
	2015-01-12	27781.7
	2015-01-13	29831.300000000003
	2015-01-14	32358.700000000004
	2015-01-15	34343.500000000001
	2015-01-16	36937.650000000001
	2015-01-17	39001.750000000001
	2015-01-18	40978.600000000006



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

```
SELECT name, revnue
FROM
  (SELECT category, name, revnue,
    RANK() OVER(PARTITION BY category ORDER BY revnue DESC) AS Rnk
  FROM
    (SELECT pizza_types.category , pizza_types.name,
      SUM( order_details.quantity * pizzas.price ) AS revnue
    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 Rnk <= 3 ;
```



Result Grid

	name	revnue
▶	The Thai Chicken Pizza	43434.25
	The Barbecue Chicken Pizza	42768
	The California Chicken Pizza	41409.5
	The Classic Deluxe Pizza	38180.5
	The Hawaiian Pizza	32273.2
	The Empress Pizza	30161.75
	The Supreme Pizza	34831.2
	The Italian Supreme Pizza	33476.75
	The Sicilian Pizza	30940.5
	The Four Cheese Pizza	32265.70000000065
	The Mexicana Pizza	26780.75
	The Five Cheese Pizza	26066.5

# *CONCLUSIONS*

*To better analyse the data.*

Increase staffing during peak hours. Promote less popular pizzas with discounts.





# LEARNINGS

***Improve SQL Skills Better  
understanding of data analysis  
technique.***





# Thank You



order now

