



PIZZAHUT SALES ANALYSIS



# Project



# PizzaHut Sales Analysis Using SQL

## Project Overview

**Objective:** To analyze PizzaHut's sales data using SQL to gain insights on performance, customer preferences, and revenue.



**Tools:** SQL(MySQL), Excel, CSV files



**Focus Areas:-**

Order volume and timing  
Best-selling pizzas  
Revenue patterns  
Pizza category breakdowns



# Dataset Summary

**orders.csv – order\_id, order\_date, order\_time**

**order\_details.csv – order\_details, order\_id, pizza\_id, quantity**

**pizzas.csv – pizza\_id, pizza\_type\_id, size, price**

**pizza\_types.csv – pizza\_type\_id, name, category, ingredients**





# Basic SQL Analysis

- Total Orders Placed: Query using COUNT(\*) on orders
- Total Revenue: SUM(quantity \* price) after joins
- Highest Priced Pizza: ORDER BY price DESC LIMIT 1
- Most Common Size: GROUP BY size with max count
- Top 5 Most Ordered Pizzas:
- Joined order details with pizza types
- Ranked by quantity ordered

# Intermediate SQL Analysis

- Total Quantity by Category: JOIN & GROUP BY category
- Order Distribution by Hour: EXTRACT(HOUR FROM time)
- Category-wise Pizza Distribution: JOIN pizza tables and group
- Avg Pizzas Per Day: GROUP BY date → AVG(quantity)
- Top 3 Revenue-Generating Pizzas:
- SUM(quantity \* price) → ORDER BY revenue DESC LIMIT 3

# Advanced SQL Analysis

- % Contribution to Revenue (Per Pizza):  
$$\text{(Pizza Revenue / Total Revenue)} * 100$$
- Cumulative Revenue Over Time:  
ORDER BY date +  $\text{SUM(...)} \text{ OVER(ORDER BY date)}$
- Top 3 Pizzas by Revenue in Each Category:  
GROUP BY category and pizza → Window function or subquery for TOP 3



# MySQL Screenshots

```
1 • CREATE DATABASE PizzaHut;
2 • USE PizzaHut;
3 • create table orders (
4     order_id int not null,
5     order_date date not null,
6     order_time time not null,
7     primary key(order_id));
8
9 • create table orders_details (
10    order_details_id int not null,
11    order_id int not null,
12    pizza_id text not null,
13    quantity int not null,
14    primary key(order_details_id));
```



# Retrieve the total number of orders placed.

```
1 -- Retrieve the total number of orders placed.  
2  
3 • select * from orders;  
4  
5 • select count(order_id) from orders;  
6  
7 • select count(order_id) as total_orders from orders;
```

Result Grid	
	total_orders
▶	21350



# Calculate the total revenue generated from pizza sales

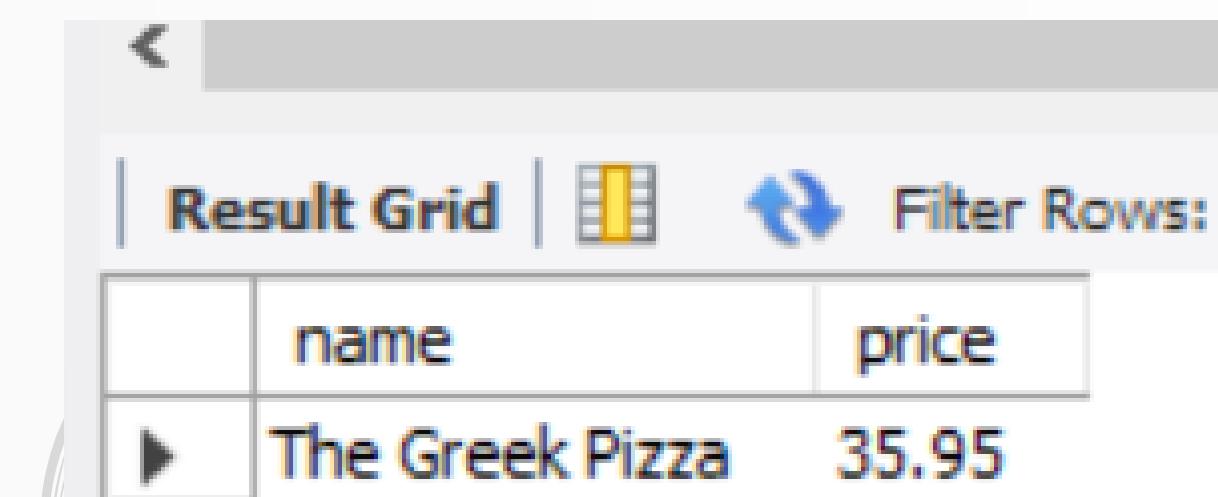
```
23 • SELECT
24     ROUND(SUM(orders_details.quantity * pizzas.price),
25             2) AS total_sales
26 FROM
27     orders_details
28     JOIN
29     pizzas ON pizzas.pizza_id = orders_details.pizza_id;
```

Result Grid	
	total_sales
▶	817860.05



# Identify the highest-priced pizza.

```
12 • SELECT
13      pizza_types.name, pizzas.price
14  FROM
15      pizza_types
16      JOIN
17      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
18  ORDER BY pizzas.price DESC
19  LIMIT 1;
```



	name	price
▶	The Greek Pizza	35.95



# Identify the most common pizza size ordered.

```
24 • SELECT
25     pizzas.size,
26     COUNT(orders_details.order_details_id) AS order_count
27 FROM
28     pizzas
29     JOIN
30     orders_details ON pizzas.pizza_id = orders_details.pizza_id
31 GROUP BY pizzas.size
32 ORDER BY order_count DESC;
```

DESC LIMIT 1;

	size	order_count
▶	L	18526
	M	15385
	S	14137
	XL	544
	XXL	28

	size	order_count
▶	L	18526



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

```
35 •   SELECT
36     pizza_types.name, SUM(orders_details.quantity) AS quantity
37   FROM
38     pizza_types
39       JOIN
40     pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
41       JOIN
42     orders_details ON orders_details.pizza_id = pizzas.pizza_id
43   GROUP BY pizza_types.name
44   ORDER BY quantity DESC
45   LIMIT 5;
```

Result Grid | Filter Rows:

	name	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

```
13 • SELECT
14     pizza_types.category,
15     SUM(orders_details.quantity) AS quantity
16 FROM
17     pizza_types
18     JOIN
19     pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
20     JOIN
21     orders_details ON orders_details.pizza_id = pizzas.pizza_id
22 GROUP BY pizza_types.category
23 ORDER BY quantity DESC;
```

Result Grid | Filter

	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050



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

```
13 • SELECT  
14      HOUR(order_time) AS hour, COUNT(order_id) AS order_count  
15 FROM  
16   orders  
17 GROUP BY HOUR(order_time);
```

Result Grid | Filter R

	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.

```
7 • SELECT  
8     category, COUNT(name)  
9 FROM  
10    pizza_types  
11 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.

```
20 •   SELECT
21       ROUND(AVG(quantity), 0) as avg_pizza_ordered_per_day
22   FROM
23   (SELECT
24       orders.order_date, SUM(orders_details.quantity) AS quantity
25   FROM
26       orders
27   JOIN orders_details ON orders.order_id = orders_details.order_id
28   GROUP BY orders.order_date) AS order_quantity;
```

	avg_pizza_ordered_per_day
▶	138



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

```
11 • SELECT
12     pizza_types.name,
13     SUM(orders_details.quantity * pizzas.price) AS revenue
14 FROM
15     pizza_types
16     JOIN
17     pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
18     JOIN
19     orders_details ON orders_details.pizza_id = pizzas.pizza_id
20 GROUP BY pizza_types.name
21 ORDER BY revenue DESC
22 LIMIT 3;
```

Result Grid | Filter Rows:

	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.

```
34 •   SELECT
35     pizza_types.category,
36     (SUM(orders_details.quantity * pizzas.price) / (SELECT
37         ROUND(SUM(orders_details.quantity * pizzas.price),
38             2) AS total_sales
39
40     FROM
41         orders_details
42         JOIN
43             pizzas ON pizzas.pizza_id = orders_details.pizza_id)) * 100 AS revenue
44
45     FROM
46         pizza_types
47         JOIN
48             pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
49         JOIN
50             orders_details ON orders_details.pizza_id = pizzas.pizza_id
51     GROUP BY pizza_types.category
52
53     ORDER BY revenue DESC;
```

Result Grid | Filter Rows:

	category	revenue
▶	Classic	26.90596025566967
	Supreme	25.45631126009862
	Chicken	23.955137556847287
	Veggie	23.682590927384577

# Analyze the cumulative revenue generated over time.

```
15 • select order_date,  
16   sum(revenue) over(order by order_date) as cum_revenue  
17   from  
18   (select orders.order_date,  
19     sum(orders_details.quantity * pizzas.price) as revenue  
20     from orders_details join pizzas  
21       on orders_details.pizza_id = pizzas.pizza_id  
22     join orders  
23       on orders.order_id = orders_details.order_id  
24     group by orders.order_date) as sales;
```

Result Grid		
	order_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

	order_date	cum_revenue
	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

	order_date	cum_revenue
	2015-01-11	25862.65
	2015-01-12	27781.7
	2015-01-13	29831.30000000003
	2015-01-14	32358.70000000004
	2015-01-15	34343.50000000001

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

```
5 • select name, revenue from
6   (select category, name, revenue,
7    rank() over(partition by category order by revenue desc) as rn
8   from
9   (select pizza_types.category, pizza_types.name,
0    sum((orders_details.quantity) * pizzas.price) as revenue
1   from pizza_types join pizzas
2   on pizza_types.pizza_type_id = pizzas.pizza_type_id
3   join orders_details
4   on orders_details.pizza_id = pizzas.pizza_id
5   group by pizza_types.category, pizza_types.name) as a) as b
6   where rn <=3;
```

Result Grid		
	name	revenue
▶	The Pepperoni Pizza	30161.75
	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	32273.25
	The Spicy Italian Pizza	34831.25
	The Italian Supreme Pizza	33476.75
	The Sicilian Pizza	30940.5
	The Four Cheese Pizza	32265.7000000065



# **Presented by: Moh Ahamad**

## **Aspiring Data Analyst**



Email: work.ahamad925@gmail.com

"SQL projects demonstrate a data analyst's ability to extract, manipulate, and analyze data from real-world databases. They showcase practical skills in deriving actionable insights through structured queries and data-driven thinking."



A close-up photograph of a hand holding a single slice of pizza. The pizza has a golden-brown crust and is topped with melted cheese and various toppings like pepperoni and vegetables. The background is blurred, making the pizza slice the focal point.

Thank  
you!

