



LICERIA  
& CO.

# PIZZA SALES

**DATA ANALYSIS USING SQL**

**PRESENTED BY**  
AACHAL JANGAMWAR



# Introduction:

- IN TODAY'S COMPETITIVE FOOD INDUSTRY, DATA PLAYS A CRUCIAL ROLE IN UNDERSTANDING CUSTOMER BEHAVIOR, OPTIMIZING OPERATIONS, AND DRIVING SALES.
- FOR A PIZZA BUSINESS, ANALYZING SALES DATA CAN REVEAL KEY INSIGHTS INTO CUSTOMER PREFERENCES, ORDER TRENDS, AND OPERATIONAL EFFICIENCIES.

# Project Purpose:

THE AIM OF THIS PROJECT IS TO ANALYZE HISTORICAL PIZZA SALES DATA USING SQL TO PROVIDE ACTIONABLE INSIGHTS FOR THE BUSINESS. BY EXAMINING FACTORS SUCH AS POPULAR PIZZA TYPES, SALES PATTERNS, AND CUSTOMER ORDERING HABITS, THE BUSINESS CAN MAKE DATA-DRIVEN DECISIONS TO ENHANCE MARKETING STRATEGIES, IMPROVE INVENTORY MANAGEMENT, AND INCREASE OVERALL PROFITABILITY.

# Dataset Description

## Tables Overview:

- **Orders:** Contains details of customer orders (order ID, order date, customer ID, etc.).
- **Order\_Details:** Information about pizzas in each order (order ID, pizza ID, quantity)
- **Pizzas:** Contains pizza details like pizza ID, size, and type.
- **Pizza\_Types:** Describes types of pizzas (pizza type, ingredients, category).

# Project Scope:

- The dataset contains information on pizza orders, order details, pizza types, and sizes. We will explore and analyze these data points to answer key business questions and provide recommendations for optimizing sales performance.

## SQL Queries

1 Retrieve the total number of orders placed.

```
select * from orders;  
select count(order_id) as total_orders from orders;
```

Result Grid	
	total_orders
▶	21350

2 calculate the total revenue generated from pizza sales.

```
8 • select  
9 round(sum(order_details.quantity * pizzas.price),2) as total_sales  
10 from  
11 order_details  
12 join  
13 pizzas on pizzas.pizza_id = order_details.pizza_id  
14
```

Result Grid	
	total_sales
▶	817860.05

### 3 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;
```

Result Grid			Filter Row
	name	price	
▶	The Greek Pizza	35.95	

### 4 Identify the most common pizza size ordered.

```
28 • select quantity, count(order_details_id)
29 from order_details group by quantity;
30
31 • SELECT
32     pizzas.size, COUNT(order_details.order_details_id) as order_count
33 FROM
34     pizzas
35     JOIN
36     order_details ON pizzas.pizza_id = order_details.pizza_id
37 GROUP BY pizzas.size
38 ORDER BY order_count DESC;
```

Result Grid			Filter Row
	size	order_count	
▶	L	18526	
	M	15385	
	S	14137	
	XL	544	
	XXL	28	

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

```
42 • SELECT
43     pizza_types.name, SUM(order_details.quantity) AS quantity
44 FROM
45     pizza_types
46     JOIN
47     pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
48     JOIN
49     order_details ON order_details.pizza_id = pizzas.pizza_id
50 GROUP BY pizza_types.name
51 ORDER BY quantity DESC
52 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	

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

```
56 • SELECT
57     pizza_types.category,
58     SUM(order_details.quantity) AS quantity
59 FROM
60     pizza_types
61     JOIN
62     pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
63     JOIN
64     order_details ON order_details.pizza_id = pizzas.pizza_id
65 GROUP BY pizza_types.category
66 ORDER BY quantity DESC;
```

Result Grid			Filter Rows:
	category	quantity	
▶	Classic	14888	
	Supreme	11987	
	Veggie	11649	
	Chicken	11050	



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

```
70 • select hour(order_time), count(order_id) as order_count from orders
71 group by hour(order_time);
```

	hour(order_time)	order_count
▶	11	1231
	12	2520
	13	2455
	14	1472
	15	1468
	16	1920
	17	2336
	18	2399

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

```
75 • select category, count(name) from pizza_types
76 group by category;
```

	category	count(name)
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

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

```
92 • SELECT
93     pizza_types.name,
94     sum(order_details.quantity * pizzas.price) AS revenue
95 FROM
96     pizza_types
97     JOIN
98     pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
99     JOIN
00     order_details ON order_details.pizza_id = pizzas.pizza_id
01     group by pizza_types.name order by revenue desc 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	41409.5

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

```
105 • SELECT
106     pizza_types.category,
107     (SUM(order_details.quantity * pizzas.price) / (SELECT
108         ROUND(SUM(order_details.quantity * pizzas.price),
109             2) AS total_sales
110     FROM
111         order_details
112         JOIN
113         pizzas ON pizzas.pizza_id = order_details.pizza_id)) * 100 AS revenue
114 FROM
115     pizza_types
116     JOIN
117     pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
118     JOIN
119     order_details ON order_details.pizza_id = pizzas.pizza_id
120 GROUP BY pizza_types.category
121 ORDER BY revenue DESC;
```

Result Grid			Filter Rows:
	category	revenue	
▶	Classic	26.90596025566967	
	Supreme	25.45631126009862	
	Chicken	23.955137556847287	
	Veggie	23.682590927384577	

## 11 analyze the cumulative revenue generated over time.

```
125 • select order_date,  
126       sum(revenue) over(order by order_date) as cum_revenue  
127 from  
128 (select orders.order_date,  
129       sum(order_details.quantity * pizzas.price) as revenue  
130 from order_details join pizzas  
131 on order_details.pizza_id = pizzas.pizza_id  
132 join orders  
133 on orders.order_id = order_details.order_id  
134 group by orders.order_date) as sales;
```

Result Grid			Filter Rows:
	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	
	2015-01-06	14358.5	
	2015-01-07	16560.7	
	2015-01-08	19399.05	

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

```
137 • select name, revenue from  
138 (select category, name, revenue,  
139       rank() over(partition by category order by revenue desc) as rn  
140 from  
141 (select pizza_types.category, pizza_types.name,  
142       sum((order_details.quantity) * pizzas.price ) as revenue  
143 from pizza_types join pizzas  
144 on pizza_types.pizza_type_id = pizzas.pizza_type_id  
145 join order_details  
146 on order_details.pizza_id = pizzas.pizza_id  
147 group by pizza_types.category, pizza_types.name) as a) as b  
148 where rn <= 3;
```

Result Grid			Filter Rows:
	name	revenue	
▶	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 Pepperoni Pizza	30161.75	
	The Spicy Italian Pizza	34831.25	
	The Italian Supreme Pizza	33476.75	

*Thank  
you!*