



## SQL PIZZA SALES PROJECT

# PIZZA





# ABOUT SQL PROJECT

Conducted a comprehensive sales analysis for a pizza restaurant using SQL. Extracted insights on total revenue, best-selling pizzas, peak order times, and customer ordering behavior. Utilized SQL queries for data cleaning, aggregation, and time-based trend analysis to support business decision-making and optimize sales strategies.

Analyzed pizza sales data using SQL by applying JOINS, GROUP BY, ORDER BY, and subqueries to uncover insights such as top-selling pizzas, peak sales hours, and revenue trends. The project demonstrated strong SQL skills in data aggregation, filtering, and multi-table querying for real-world business analysis.





# OUR BEST SELLER PIZZA

-- Identify the most common pizza size ordered --

```
SELECT
    p.size, COUNT(order_details_id) AS pizza_ordered
FROM
    order_details od
    JOIN
    pizzas p ON p.pizza_id = od.pizza_id
GROUP BY p.size
ORDER BY pizza_ordered DESC;
```

	size	pizza_ordered
►	L	18526
	M	15385
	S	14137
	XL	544
	XXL	28





# Retrieve the total number of orders placed.

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

	total_orders
▶	21350

-- Calculate the total revenue generated from pizza sales.

```
SELECT
    ROUND(SUM(od.quantity * p.price),
          2) AS total_revenue
FROM
    order_details od
    JOIN
    pizzas p ON p.pizza_id = od.pizza_id;
```

	total_revenue
▶	817860.05



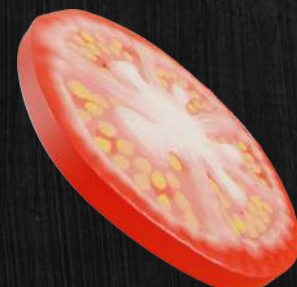




-- Identify the highest-priced pizza --

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

	name	price
▶	The Greek Pizza	35.95





```
-- Identify the most common pizza size ordered --
```

```
SELECT
    p.size, COUNT(order_details_id) AS pizza_ordered
FROM
    order_details od
    JOIN
    pizzas p ON p.pizza_id = od.pizza_id
GROUP BY p.size
ORDER BY pizza_ordered DESC;
```

	size	pizza_ordered
►	L	18526
	M	15385
	S	14137
	XL	544
	XXL	28



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

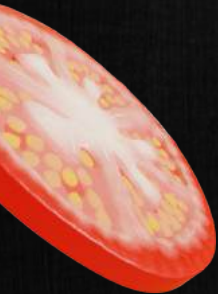
```
SELECT
    pt.name, SUM(od.quantity) AS quantity
FROM
    pizza_types pt
    JOIN
    pizzas p ON p.pizza_type_id = pt.pizza_type_id
    JOIN
    order_details od ON od.pizza_id = p.pizza_id
GROUP BY pt.name
ORDER BY quantity DESC
LIMIT 5;
```

	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

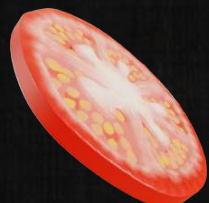
-- Join the necessary tables to find the total quantity of each pizza category ordered --

```
SELECT
    pt.category, SUM(od.quantity) AS total_quantity
FROM
    pizza_types pt
    JOIN
    pizzas p ON p.pizza_type_id = pt.pizza_type_id
    JOIN
    order_details od ON od.pizza_id = p.pizza_id
GROUP BY pt.category
ORDER BY total_quantity DESC;
```

	category	total_quantity
►	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050







```
-- Determine the distribution of orders by hour of the day.
```

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

	hourly_orders	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) as pizza_count
from pizza_types
group by category;
```

	category	pizza_count
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9





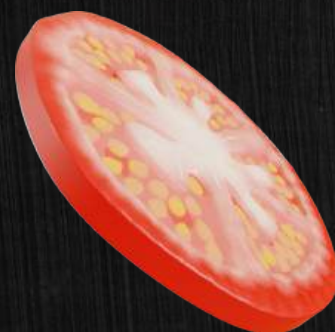


```
-- Group the orders by date and calculate the average number of pizzas ordered per day --
```

```
SELECT
    ROUND(AVG(quantity), 0) as avg_pizzas_ordered_per_day
FROM
    (SELECT
        o.order_date, SUM(od.quantity) AS quantity
    FROM
        orders o
    JOIN order_details od ON od.order_id = o.order_id
    GROUP BY o.order_date) AS order_quantity;
```




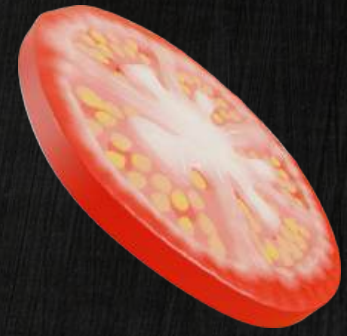
	avg_pizzas_ordered_per_day
▶	138





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

```
SELECT
    pt.name, ROUND(SUM(od.quantity * p.price), 2) AS revenue
FROM
    pizza_types pt
    JOIN
    pizzas p ON p.pizza_type_id = pt.pizza_type_id
    JOIN
    order_details od ON od.pizza_id = p.pizza_id
GROUP BY pt.name
ORDER BY revenue DESC
LIMIT 3;
```

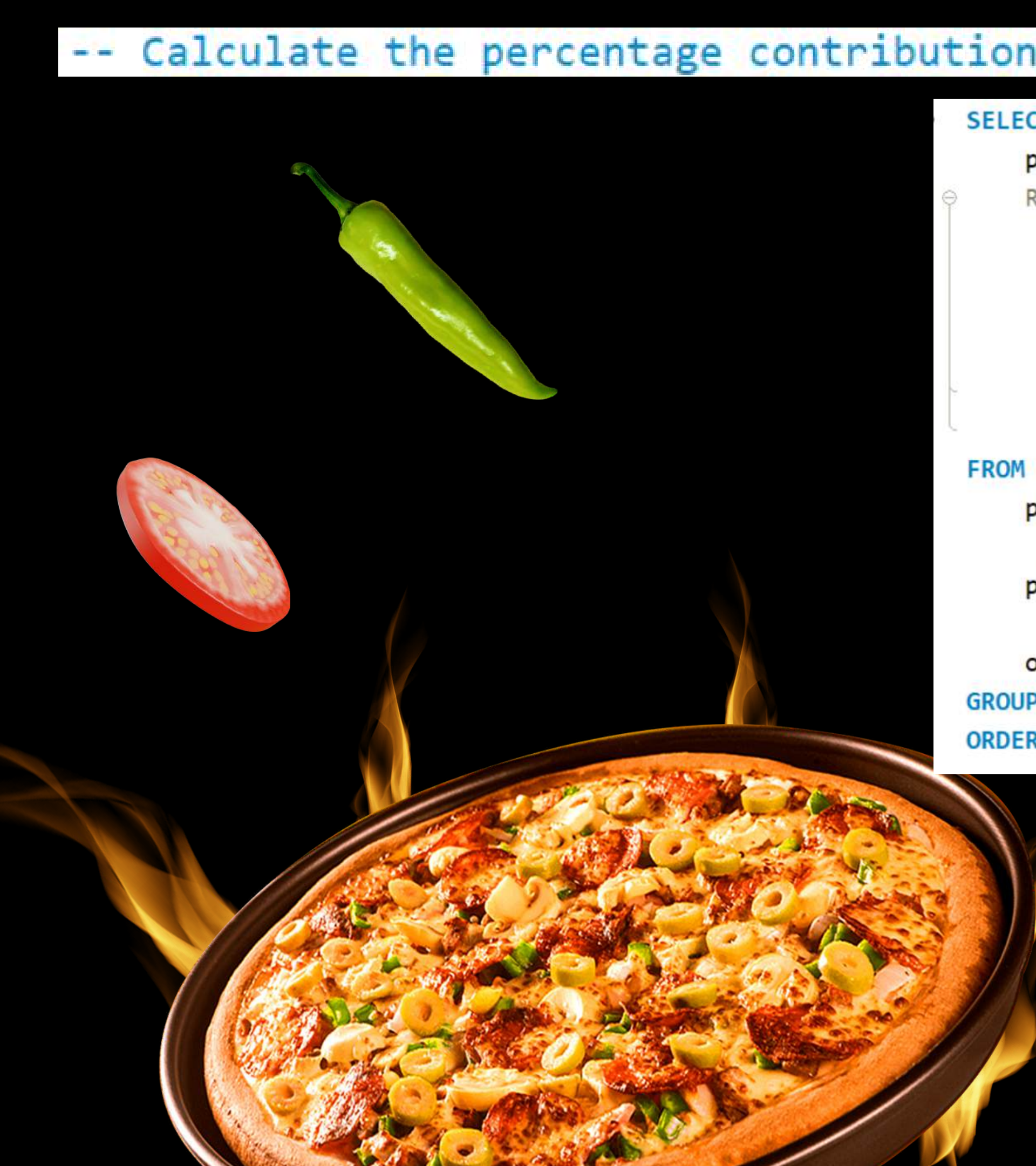


	name	revenue
▶	The Thai Chicken Pizza	43434.25
	The Barbecue Chicken Pizza	42768
	The California Chicken Pizza	41409.5



-- Calculate the percentage contribution of each pizza type to total revenue --

```
SELECT
  pt.category,
  ROUND(SUM(od.quantity * p.price) / (SELECT
    ROUND(SUM(od.quantity * p.price), 2) AS total_revenue
    FROM
      order_details od
      JOIN
        pizzas p ON p.pizza_id = od.pizza_id) * 100,
    2) AS revenue
FROM
  pizza_types pt
  JOIN
    pizzas p ON p.pizza_type_id = pt.pizza_type_id
  JOIN
    order_details od ON od.pizza_id = p.pizza_id
GROUP BY pt.category
ORDER BY revenue;
```





	category	revenue
▶	Veggie	23.68
	Chicken	23.96
	Supreme	25.46
	Classic	26.91



-- Analyze the cumulative revenue generated over time.

```
select order_date, sum(revenue) over (order by order_date) as cum_revenue
from
(select o.order_date, round(sum(od.quantity * p.price), 2) as revenue
from order_details od
join pizzas p
on p.pizza_id = od.pizza_id
join orders o
on o.order_id = od.order_id
group by o.order_date) as sales;
```



	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





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

```
select name,revenue from
(select category,name,revenue,
rank() over(partition by category order by revenue desc) as rn
from
(select pt.category,pt.name,sum((od.quantity) * p.price) as revenue
from pizza_types pt
join pizzas p
on p.pizza_type_id = pt.pizza_type_id
join order_details od
on od.pizza_id = p.pizza_id
group by pt.category,pt.name) as a) as b
where rn <= 3;
```

	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
	The Sicilian Pizza	30940.5
	The Four Cheese Pizza	32265.700000000065
	The Mexicana Pizza	26780.75
	The Five Cheese Pizza	26066.5





Thank You!

Thank you for your guidance and support. Your help made a big difference in completing this project and showcasing my SQL skills effectively. I truly appreciate it!

This project showcased my SQL skills through the use of JOINS, GROUP BY, ORDER BY, and subqueries to analyze real-world pizza sales data. It demonstrated my ability to extract actionable insights, perform complex data manipulations, and write efficient queries for business decision-making.

