

PIZZA SALES SQL PROJECT

The objective of this project is to analyze a fictional pizza restaurant's sales data using SQL to extract meaningful business insights. By writing and executing SQL queries, we aim to:

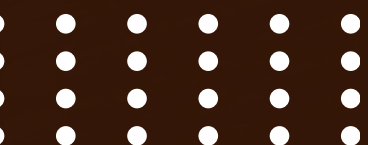
Understand customer ordering patterns

Identify top-performing pizzas by revenue and quantity

Analyze sales trends by time and category

Provide data-driven recommendations to optimize operations and menu strategy

This project showcases how SQL can be used effectively to turn raw transactional data into actionable insights for business decision-making.



QUESTIONS FROM THE ANALYSIS



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.


```
1  -- Retrieve the total number of orders placed.  
2      I  
3 • select count(order_id) as total_orders from orders;
```

Result Grid			 Filter Rows:	
	total_orders			
▶	21350			

```
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
```

	total_sales
▶	817860.05


```
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 Rows:		Exports	Wrap Cell Content:
	name	price				
▶	The Greek Pizza	35.95				

```
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
```

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


```
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;
```

	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

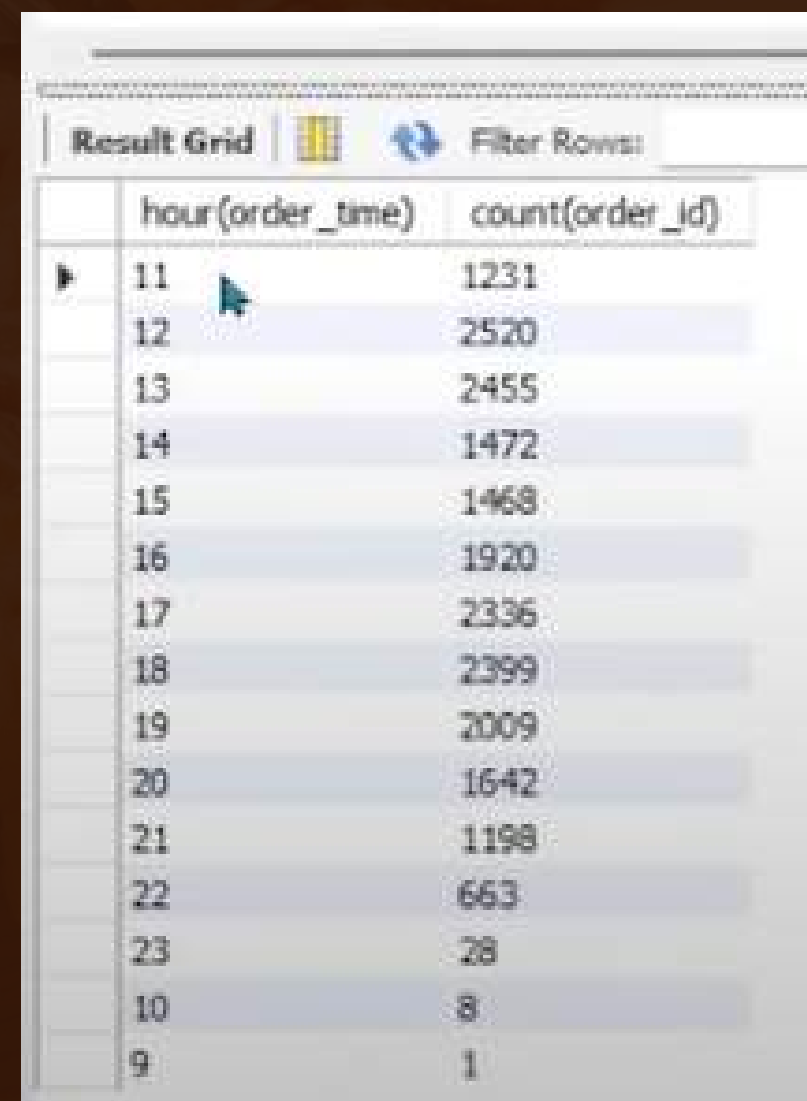

```
select pizza_types.category,  
sum(order_details.quantity) as quantity  
from pizza_types join pizzas  
on pizza_types.pizza_type_id = pizzas.pizza_type_id  
join order_details  
on order_details.pizza_id = pizzas.pizza_id  
group by pizza_types.category order by quantity desc;
```



The screenshot shows a database interface with a 'Result Grid' tab. The grid displays the results of the SQL query, with columns 'category' and 'quantity'. The rows are sorted in descending order of quantity. The 'Supreme' category is highlighted with a mouse cursor.

	category	quantity
	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050


```
select hour(order_time), count(order_id) from orders  
group by hour(order_time);
```



The screenshot shows a database query result grid. The grid has two columns: 'hour(order_time)' and 'count(order_id)'. The data is sorted by hour in descending order, from 11 down to 9. The counts for each hour are: 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), and 9 (1). A mouse cursor is hovering over the row for hour 11.

	hour(order_time)	count(order_id)
▶	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


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

I

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


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

Result Grid		Filter Rows
	avg_pizza_ordered_per_day	
▶	138	


```
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
	The Thai Chicken Pizza	43434.25
	The Barbecue Chicken Pizza	42768
	The California Chicken Pizza	41409.5


```

SELECT
    pizza_types.category,
    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;

```

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68


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

	order_date	cum_revenue
▶	2015-01-01	2713.850000000000004
	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


```

SELECT name, 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;

```

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



**THANK YOU
FOR ATTENTION**