Pizza Sales Data Analysis using SQL

BY:CHARAK MADHAV KARLE

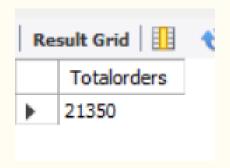
[Technical Stack]

- Database: MySQL
- Key Skills: Complex Joins,
 Aggregation, Time-Series Analysis,
 Window Functions
- Dataset: Pizza Sales (Orders details, pizzas, pizza types,orders)

Project Date: June 2025

Objective: Retrieve the total number of orders placed.

SELECT COUNT(order_id) as Totalorders FROM orders;

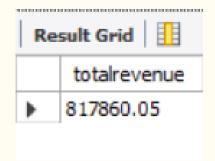


Key Insights & Business Impact Scale of Operations:

• The pizzeria processed 21,350 orders, indicating a high-volume business.

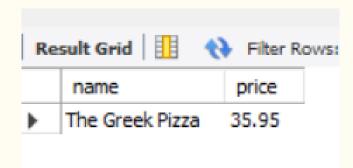
Objective:Calculate the total revenue generated from pizza sales

```
-- Q2Calculate the total revenue generated from pizza sales
SELECT round(SUM(pizzas.price * orders_details.quantity),2)
as totalrevenue
FROM orders_details
JOIN pizzas
ON orders_details.pizza_id=pizzas.pizza_id;
```

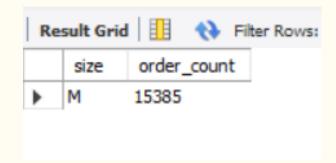


Objective:Identify the highest-priced pizza.

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



Objective:Identify the most common pizza size ordered.



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

```
-- Q5List the top 5 most ordered pizza types along with their quantities

SELECT
    pizza_types.name, SUM(orders_details.quantity) AS quantity

FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON orders_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

Objective: Join the necessary tables to find the total quantity of each pizza category ordered.

```
-- Q6Join the necessary tables to find
-- the total quantity of each pizza category ordered.

SELECT

pizza_types.category, SUM(orders_details.quantity) AS quantity

FROM

pizza_types

JOIN

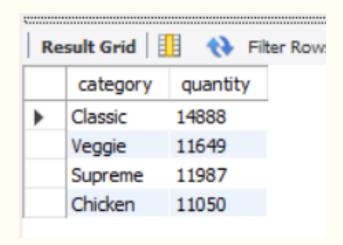
pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id

JOIN

orders_details ON orders_details.pizza_id = pizzas.pizza_id

GROUP BY pizza_types.category

;
```



Objective:Determine the distribution of orders by hour of the day.

```
-- Q7Determine the distribution of orders by hour of the day

SELECT
    HOUR(order_time), COUNT(order_id) AS count_orders
FROM
```

orders
GROUP BY HOUR(order_time);

Re	sult Grid 🔢 🙌	Filter Rows:
	HOUR(order_time)	count_orders
•	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
	-	

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

```
-- Q8Join relevant tables to find the category-wise distribution of pizzas.

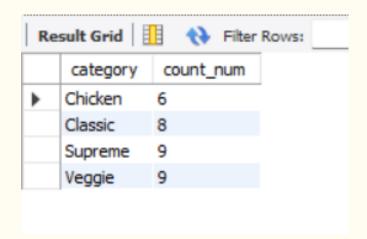
• SELECT

category, COUNT(name) AS count_num

FROM

pizza_types

GROUP BY category;
```



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

```
-- Q9Group the orders by date and calculate the average number of pizzas ordered per day.

SELECT

AVG(sum_num)

FROM

(SELECT

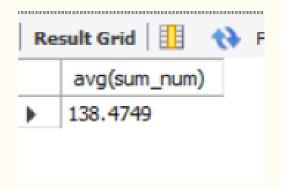
orders.order_date, SUM(orders_details.quantity) AS sum_num

FROM

orders

JOIN orders_details ON orders.order_id = orders_details.order_id

GROUP BY orders.order_date) x;
```



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

-- Q10Determine the top 3 most ordered pizza types based on revenue.

```
SELECT

pizza_types.name,

SUM(orders_details.quantity * pizzas.price) AS revenue

FROM

pizza_types

JOIN

pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id

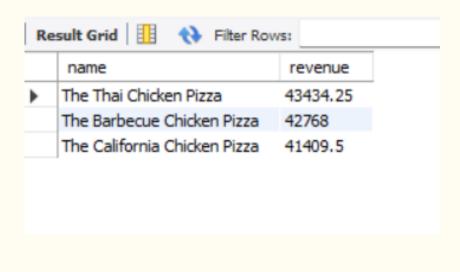
JOIN

orders_details ON orders_details.pizza_id = pizzas.pizza_id

GROUP BY pizza_types.name

ORDER BY revenue DESC

LIMIT 3;
```



Objective:Calculate the percentage contribution of each pizza type to total revenue.

-- Q11Calculate the percentage contribution of each pizza type to total revenue

```
SELECT x.category,

ROUND((x.revenue / (SELECT ROUND(SUM(pizzas.price * orders_details.quantity),2) AS totalrevenue
FROM orders_details JOIN pizzas ON orders_details.pizza_id = pizzas.pizza_id)) * 100,2) AS revenue_percer

FROM(SELECT pizza_types.category,
ROUND(SUM(orders_details.quantity * pizzas.price), 2) AS revenue
FROM

orders_details JOIN pizzas ON orders_details.pizza_id = pizzas.pizza_id
JOIN pizza_types ON pizza_types.pizza_type_id = pizzas.pizza_type_id

GROUP BY pizza_types.category) AS x;
```

category		revenue_percen	
١	Classic	26.91	
	Veggie	23.68	
	Supreme	25.46	
	Chicken	23.96	

Objective:Analyze the cumulative revenue generated over time.

```
-- Q12Analyze the cumulative revenue generated over time.

SELECT order_date,SUM(revenue) over(order by order_date) as cum_rev FROM

(SELECT orders.order_date,
sum(orders_details.quantity*pizzas.price) as revenue
FROM orders_details

JOIN pizzas
ON orders_details.pizza_id=pizzas.pizza_id

JOIN orders
ON orders_details.order_id=orders.order_id

GROUP BY orders.order_date) as x;
```

Res	ult Grid 📗	N Filter Rows:	Export:
	order_date	cum_rev	
•	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	34343.50000000001	
	2015-01-16	36937.65000000001	
	2015-01-17	39001.75000000001	

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

```
SELECT category,name,revenue,rank_ FROM
(SELECT *,rank() over (partition by category order by revenue desc ) as rank_ FROM
(SELECT pizza_types.category,pizza_types.name,
sum(orders_details.quantity*pizzas.price) as revenue
FROM orders_details JOIN pizzas ON orders_details.pizza_id=pizzas.pizza_id
JOIN orders ON orders_details.order_id=orders.order_id JOIN pizza_types
ON pizza_types.pizza_type_id=pizzas.pizza_type_id
GROUP BY pizza_types.category,pizza_types.name )as a ) as b
WHERE rank_<=3;</pre>
```

	category	name	revenue	rank_	
٠	Chicken	The Thai Chicken Pizza	43434.25	1	
	Chicken	The Barbecue Chicken Pizza	42768	2	
	Chicken	The California Chicken Pizza	41409.5	3	
	Classic	The Classic Deluxe Pizza	38180.5	1	
	Classic	The Hawaiian Pizza	32273.25	2	
	Classic	The Pepperoni Pizza	30161.75	3	
	Supreme	The Spicy Italian Pizza	34831.25	1	
	Supreme	The Italian Supreme Pizza	33476.75	2	
	Supreme	The Sicilian Pizza	30940.5	3	
	Veggie	The Four Cheese Pizza	32265.70000000065	1	
	Veggie	The Mexicana Pizza	26780.75	2	
	Veggie	The Five Cheese Pizza	26066.5	3	