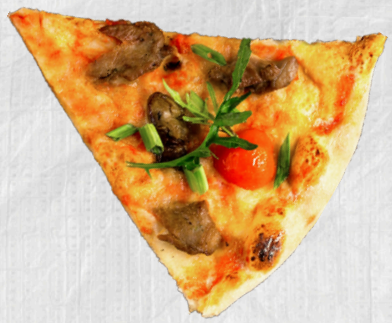
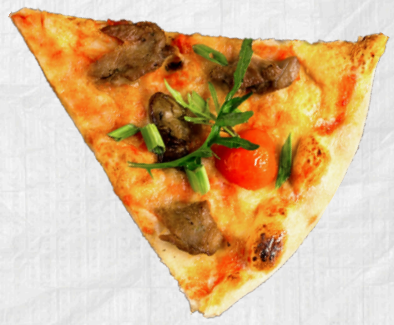
DELICIOUS PIZZA

PIZZA SALES REPORT ANALYSIS USING SQL

Unlocking Business Insights from Pizza Sales Data Through Structured Query Language (SQL)

ORDER NOW ›





Hello!

I’m Siddique, a passionate in Data Analytics .

I have created this SQL-based project to analyze pizza sales data and generate actionable insights for business decision-making.

This project reflects my ability to work with real-world datasets, perform aggregations, join multiple tables, and extract meaningful patterns using SQL.

ORDER NOW ›

```sql
-- Retrieve the total number of orders placed

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

Limit to 1000 rows

Result Grid

total_orders
21350

ORDER NOW ›

```sql
-- Calculate the total revenue generated from pizza sales.

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

Result Grid

total_revenue
817860.05

ORDER NOW ›

```sql
-- Identify the highest-priced pizza.

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

Result Grid | Filter Rows:

	name	price
▶	The Greek Pizza	35.95

ORDER NOW

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

select quantity, count(order_details_id)
from order_details
group by quantity;
```

Result Grid | Filter Rows:

quantity	count(order_details_id)
1	47693
2	903
3	21
4	3

ORDER NOW ›

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

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
ORDER BY order_count DESC;
```

Result Grid | Filter Rows:

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

ORDER NOW ›

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

SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantity
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 quantity DESC
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

Result 1

ORDER NOW ›

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

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;
```

Result Grid | Filter Rows:

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

ORDER NOW

```sql
-- Determine the distribution of orders by hour of the day.
SELECT
    HOUR(order_time) AS hour, COUNT(order_id) AS order_count
FROM
    orders
GROUP BY hour;
```

Result Grid | Filter Rows:

hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468

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

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

Result Grid | Filter Rows:

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

ORDER NOW ›

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

select round(avg(quantity),0) as avg_pizza_ordered_per_day from
(select orders.order_date, sum(order_details.quantity) as quantity from orders
join order_details on
orders.order_id = order_details.order_id
group by orders.order_date) as order_quantity;
```

Result Grid | Filter Rows:

	avg_pizza_ordered_per_day
▶	138

ORDER NOW ›

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

ORDER N

```sql
-- Calculate the percentage contribution of each pizza type to total revenue.
SELECT
    pizza_types.category,
    ROUND(SUM(order_details.quantity * pizzas.price) /
        (SELECT SUM(order_details.quantity * pizzas.price)
         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;
```

Result Grid | Filter Rows:

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

ORDER NOW

```sql
-- Analyze the cumulative revenue generated over time.

select order_date,round(sum(revenue) over (order by order_date),0) as cumulative_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:

order_date	cumulative_revenue
2015-01-01	2714
2015-01-02	5446
2015-01-03	8108
2015-01-04	9864
2015-01-05	11930

ORDER NOW ›

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

select category,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;
```

Result Grid | Filter Rows: | Export:

category	name	revenue
Chicken	The Thai Chicken Pizza	43434.25
Chicken	The Barbecue Chicken Pizza	42768
Chicken	The California Chicken Pizza	41409.5
Classic	The Classic Deluxe Pizza	38180.5
Classic	The Hawaiian Pizza	32273.25

ORDER NOW ›