Where Every Slice is a Taste of Perfection

WELCOME TO PIZZA RESTO

ORDER NOW

Start Your Slide

HELLO!

My name is Roshan Raj Nayak, and I have created a SQL-based project on Pizza Sales Analysis.

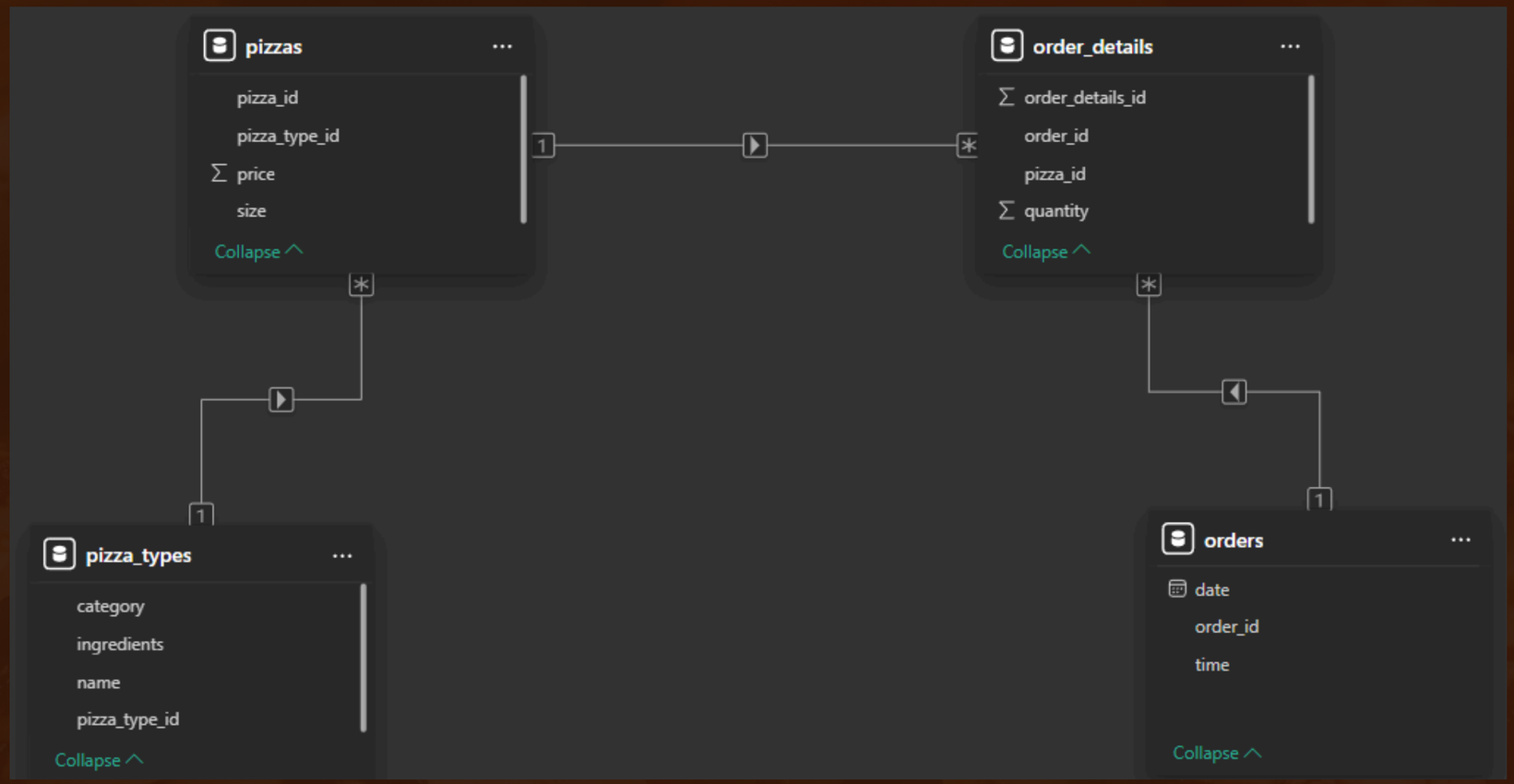
This project focuses on exploring and understanding the sales performance of different pizza categories and types.

Using powerful SQL queries, I extracted, joined, and analyzed data from multiple tables to uncover valuable business insights.

The analysis helps identify top-selling pizzas, revenue trends, and customer preferences across various categories.

Through this project, I aimed to transform raw sales data into meaningful information for better decision-making.

It demonstrates how SQL can be effectively used for data analysis, visualization, and performance evaluation.



The database consists of four main interconnected tables — orders, order_details, pizzas, and pizza_types.

These relationships form the foundation for analyzing pizza sales data, revenue trends, and customer preferences.

Using SQL joins, this model allows detailed insights into sales, pricing, and category performance.

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

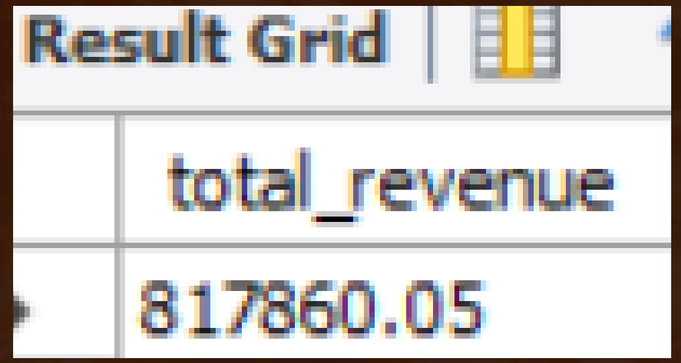
```sql
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

Result Grid

total_orders
21350

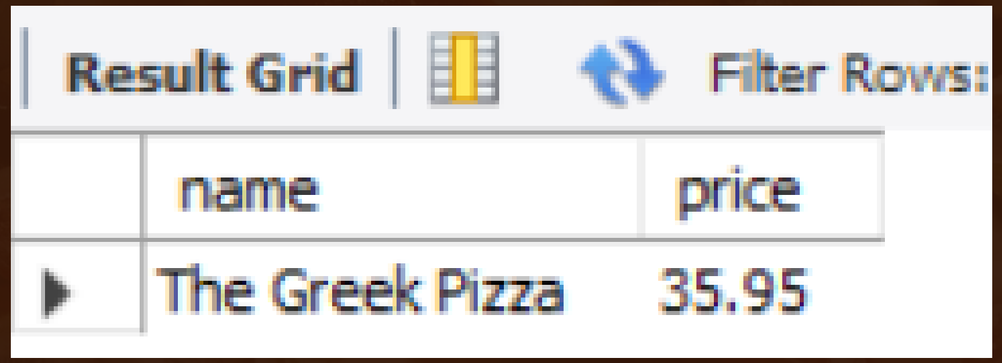
CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```
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;
```



Identify the highest-priced pizza.

```sql
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;
```



IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

```sql
SELECT
    pizzas.size,
    COUNT(order_details.order_detail_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

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

LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.

```sql
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;
```

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

JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

```sql
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

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

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

Result Grid | Filter Rows:

HOUR(order_time)	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.

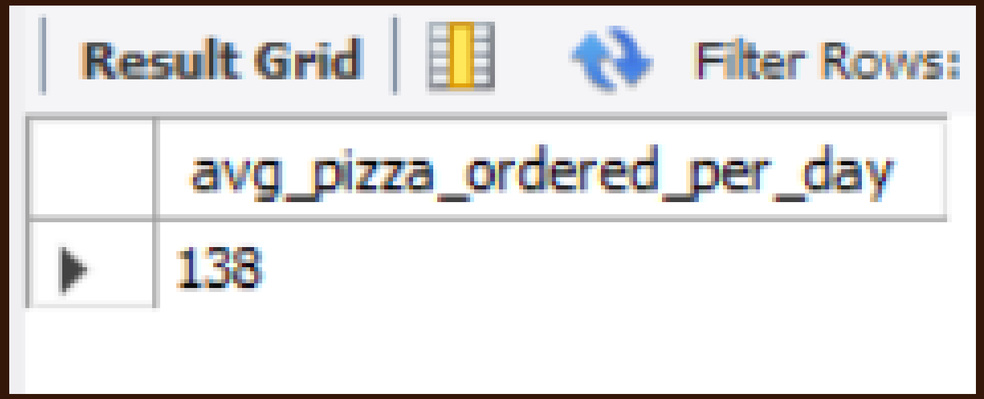
```
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

Result Grid | Filter Rows:

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

GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

```sql
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;
```



DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

```sql
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;
```

Result Grid | Filter Rows:

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.

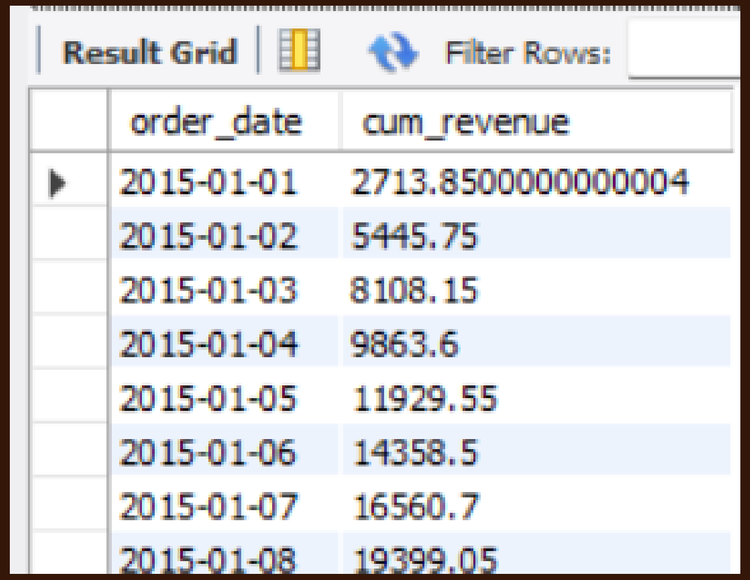
```sql
SELECT
    pizza_types.category,
    ROUND((SUM(order_details.quantity * pizzas.price) / (SELECT
                    ROUND(SUM(order_details.quantity * pizzas.price),
                                2) AS total_revenue
                FROM
                    order_details
                        JOIN
                    pizzas ON pizzas.pizza_id = order_details.pizza_id)) * 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 | Filte

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
SELECT
    sales.order_date,
    SUM(sales.revenue) OVER (ORDER BY sales.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:

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

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY.

```sql
SELECT
    name, revenue
FROM
    (SELECT category, name, revenue,
            RANK() OVER (PARTITION BY category ORDER BY revenue DESC) AS ranking
        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
    ranking <= 3;
```

Result Grid | Filter Rows:

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

THANK YOU FOR ATTENTION