Hot & Delicious

Pizza

SQL Pizza Sales Project!

HELLO

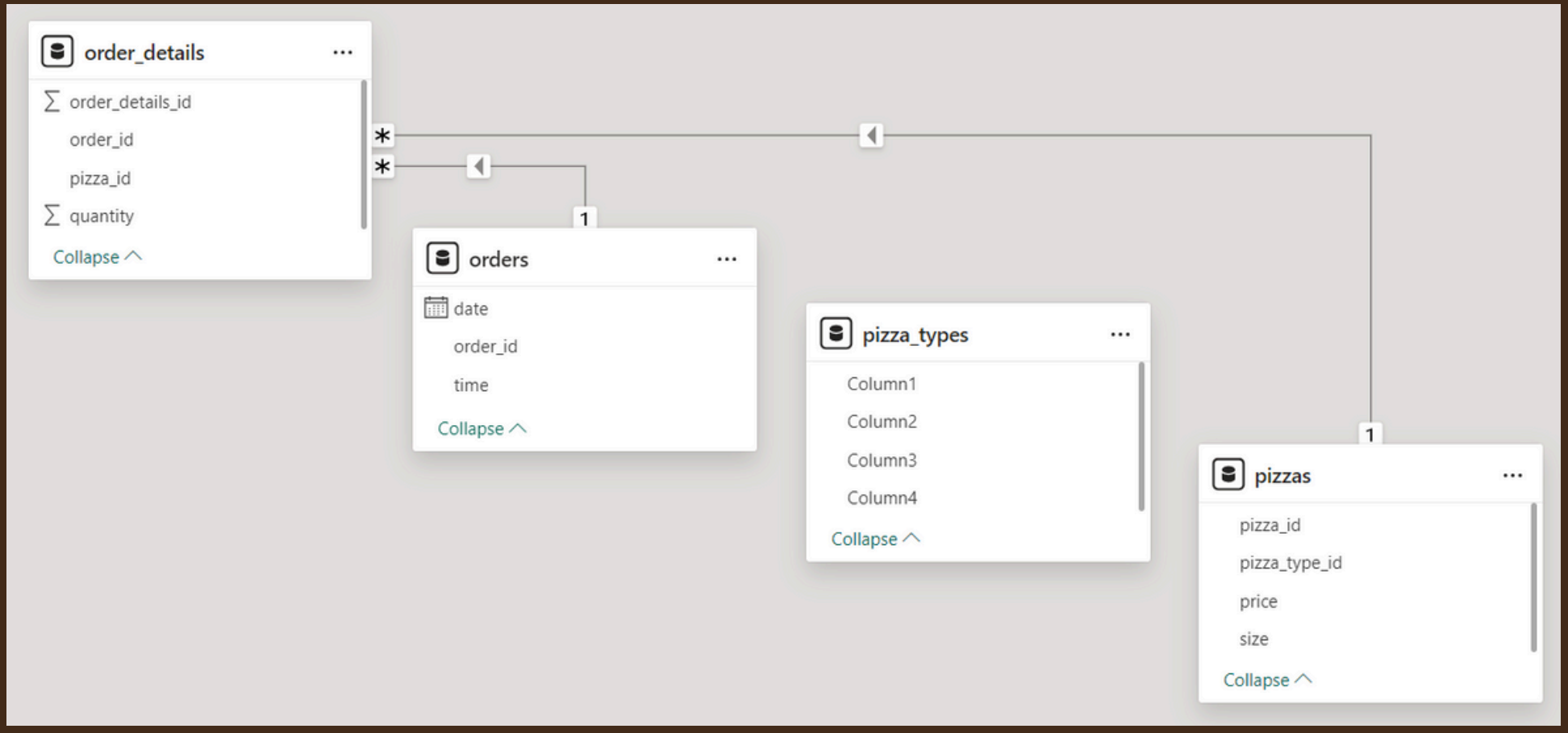
My name is Saumya Singh, in this project i have utilized sql queries to solve questions that were related to pizza sales. The primary objective was to analyze data that help improve sales strategies and customer satisfaction. By leveraging various SQL techniques such as joins, subqueries, and aggregate functions, I was able to identify trends, peak sales periods, and customer preferences. . This project not only enhanced my technical skills but also provided a deeper understanding of data-driven decision-making in the food industry.

This is the model view of the pizza sales data depicted how tables were connected to each other.

The 'Orders' table was linked to the 'Customers' table via the 'CustomerID' field, enabling a detailed view of which customer placed each order. Similarly, the 'Orders' table was connected to the 'Pizzas' table through the 'PizzaID' field, providing insights into which types of pizzas were most popular among different customer segments.

Moreover, the 'Ingredients' table was related to the 'Pizzas' table, showcasing the composition of each pizza and allowing for an analysis of ingredient usage trends. This interconnected structure facilitated comprehensive data analysis, enabling to identify patterns, optimize inventory, and tailor marketing strategies more effectively.

By leveraging these relational connections, the pizza sales data model offered a robust framework for understanding and improving the business's performance.



Retrieve the total numbers of orders placed.

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

Result Grid

	total_orders
▶	21350

Calculate the total revenue generated from the pizza sales

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

Result Grid

	total_revenue
▶	817860.05

Identify the highest-priced pizza.

```sql
select max(price) from pizzas;

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 Rov

name	price
The Greek Pizza	35.95

Identify the most common pizza size ordered.

```sql
SELECT
    pizzas.size,
    COUNT(orders_details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;
```

Result Grid | Filter

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(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;
```

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(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
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.

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

Result Grid

hour	order_coun
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.

```sql
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_order_per_day
FROM
    (SELECT
        orders.order_date, SUM(orders_details.quantity) AS quantity
    FROM
        orders
    JOIN orders_details ON orders.order_id = orders_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

Result Grid | Filter Rows

	avg_pizza_order_per_day
▶	138

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

```sql
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;
```

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.

```
select pizza_types.category, round(sum(orders_details.quantity*pizzas.price) /
(Select round(sum(orders_details.quantity * pizzas.price),2) as total_revenue
from orders_details join pizzas
on orders_details.pizza_id = pizzas.pizza_id)* 100, 2) 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.category order by revenue desc;
```

Result Grid Filter

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Analyze the cumulative revenue generated over time.

```sql
select order_date,
round(sum(revenue) over (order by order_date),2) as cum_revenue
from
(select orders.order_date,
round(sum(orders_details.quantity * pizzas.price),2) as revenue
from orders_details join pizzas
ON orders_details.pizza_id = pizzas.pizza_id
join orders ON orders.order_id = orders_details.order_id
group by orders.order_date) as sales;
```

Result Grid | Filter Row

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	21526.4
2015-01-10	23990.35
2015-01-11	25862.65
2015-01-12	27781.7
2015-01-13	29831.3
2015-01-14	32358.7
2015-01-15	34343.5

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 rn
from
(select pizza_types.category, 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.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
The Spicy Italian Pizza	34831.25
The Italian Supreme Pizza	33476.75
The Sicilian Pizza	30940.5
The Four Cheese Pizza	32265.70000000065
The Mexicana Pizza	26780.75
The Five Cheese Pizza	26066.5