SQL PIZZA SALES ANALSYIS

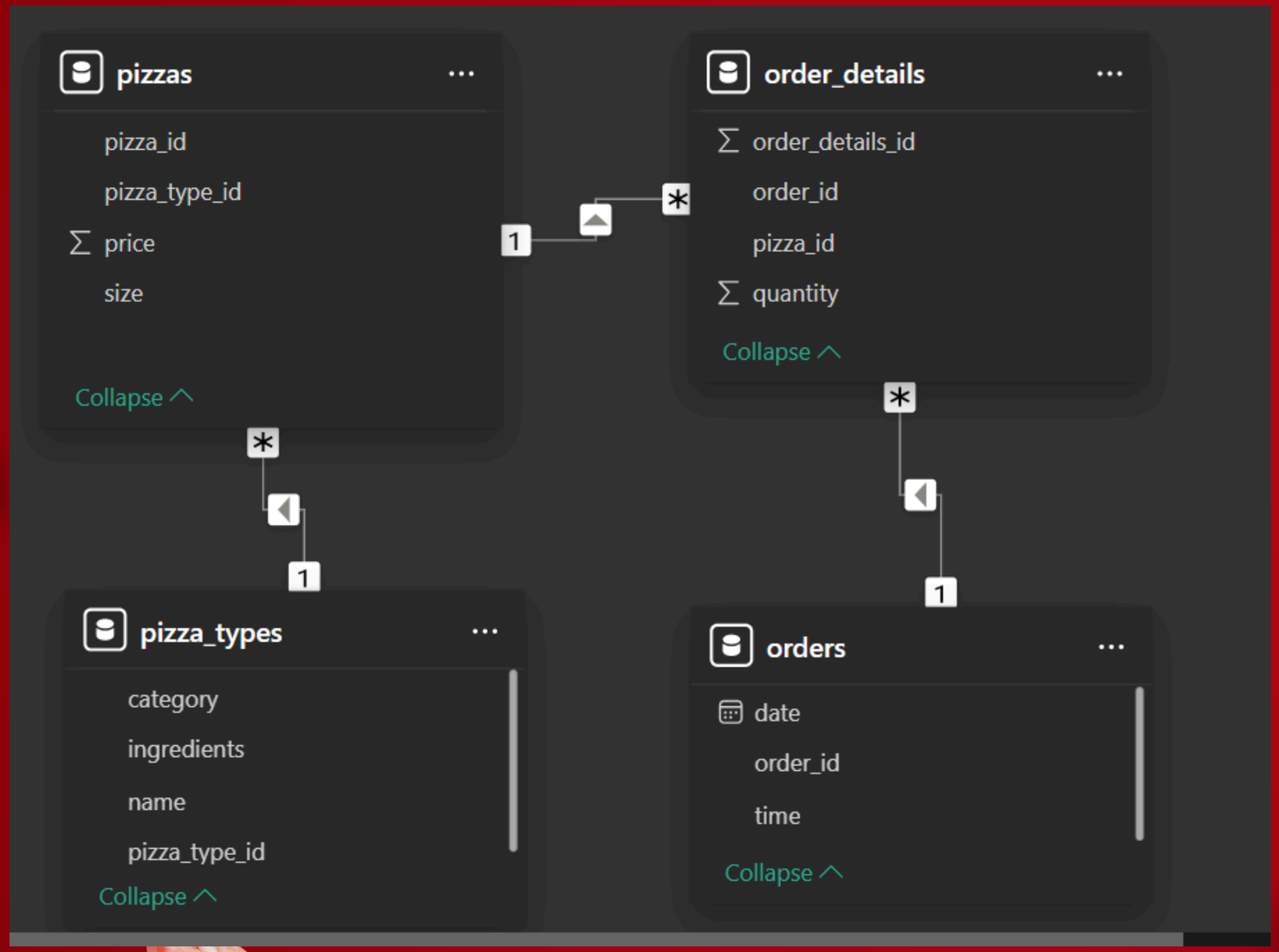
BY KALPANA SINGH

ABOUT THE PROJECT

Hello everyone, My name is kalpana and in this presentation, i will be discussing an analysis of pizza sales using SQL. We'll explore how data related to pizza sales -- such as customer orders, order details, and order quantities can be effectively managed and queried using SQL database.

Through this analysis, we'll uncover valuable insights into sales patterns, customer preferences, and operational efficiencies, all of which can help drive business decisions and strategies for the pizza industry.

DATA MODEL


pizzas
pizza_id
pizza_type_id
price
size
Collapse
order_details
order_details_id
order_id
pizza_id
quantity
Collapse
pizza_types
category
ingredients
name
pizza_type_id
Collapse
orders
date
order_id
time
Collapse
1
*
1
*
*
1

APPROACH

1) Table Joins: Integrate relevant tables to derive insights such as pizza categories, sizes, and order timestamps.

2) Aggregations and calculations: Usr SQL function like SUM(), COUNT(), AVG() AND PERCENTAGE() to calculate totals, averages and distributions.

3)Filtering and Ranking: Identify top- performing pizzas and categories and ranking techniques.

4) Temporal Analysis: Analysis order patterns and revenue distribution over time to identify trends.

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED

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

Result Grid | Filter Rows:

Total_orders
21350

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES

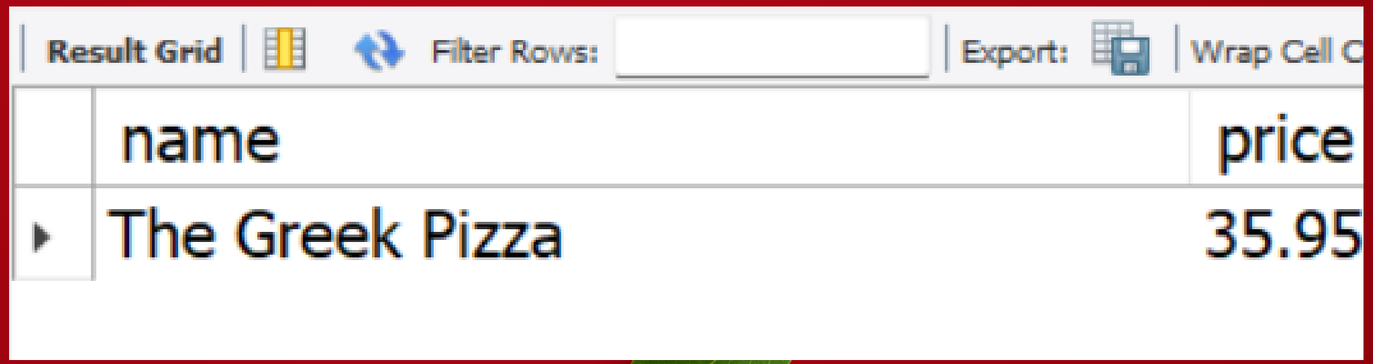
```sql
SELECT
      ROUND(sum(order_details.quantity * pizzas.price),
            2) as total_sales
FROM order_details
JOIN pizzas
ON pizzas.pizza_id = order_details.pizza_id;
```

Result Grid | Filter Ro

total_sales
817860.05

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;
```



name	price
The Greek Pizza	35.95

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

```sql
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

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
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

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 Rows:

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

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(order_time);
```

Result Grid | Filter Rows:

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

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;
```

Result Grid | Filter Rows:

avg_pizza_ordered_per_da
138

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: | Export: | Wrap Cell Content:

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_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;
```

Result Grid | Filter Rows:

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,
      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	2713.85
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED O 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(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: | Wrap Cell Content:

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

KEY TAKEAWAYS:

1) XXL Pizza Size Underperformance:

Out of 21,350 orders, XXL pizzas were ordered only 28 times. It's recommended to discontinue this size and focus on more popular sizes.

2) Classic Pizza Popularity:

Among the four pizza categories, customers prefer the "Classic" category the most, making it a key area to focus on.

3) Order Timing Trend:

There are very few orders between 9 AM and 11 AM, with peak sales starting from 11 PM, likely due to people being busy during the day.

4) Chicken Pizza Category:

The chicken category has only six pizza types, leading to fewer sales. Expanding the variety of chicken pizzas, like other categories (supreme, veggies, classic), could boost sales.

5) Average Orders:

With an average of 138 pizzas sold per order, further advertising, discounts, or coupons could help increase this number and attract more customers.

THANK YOU!