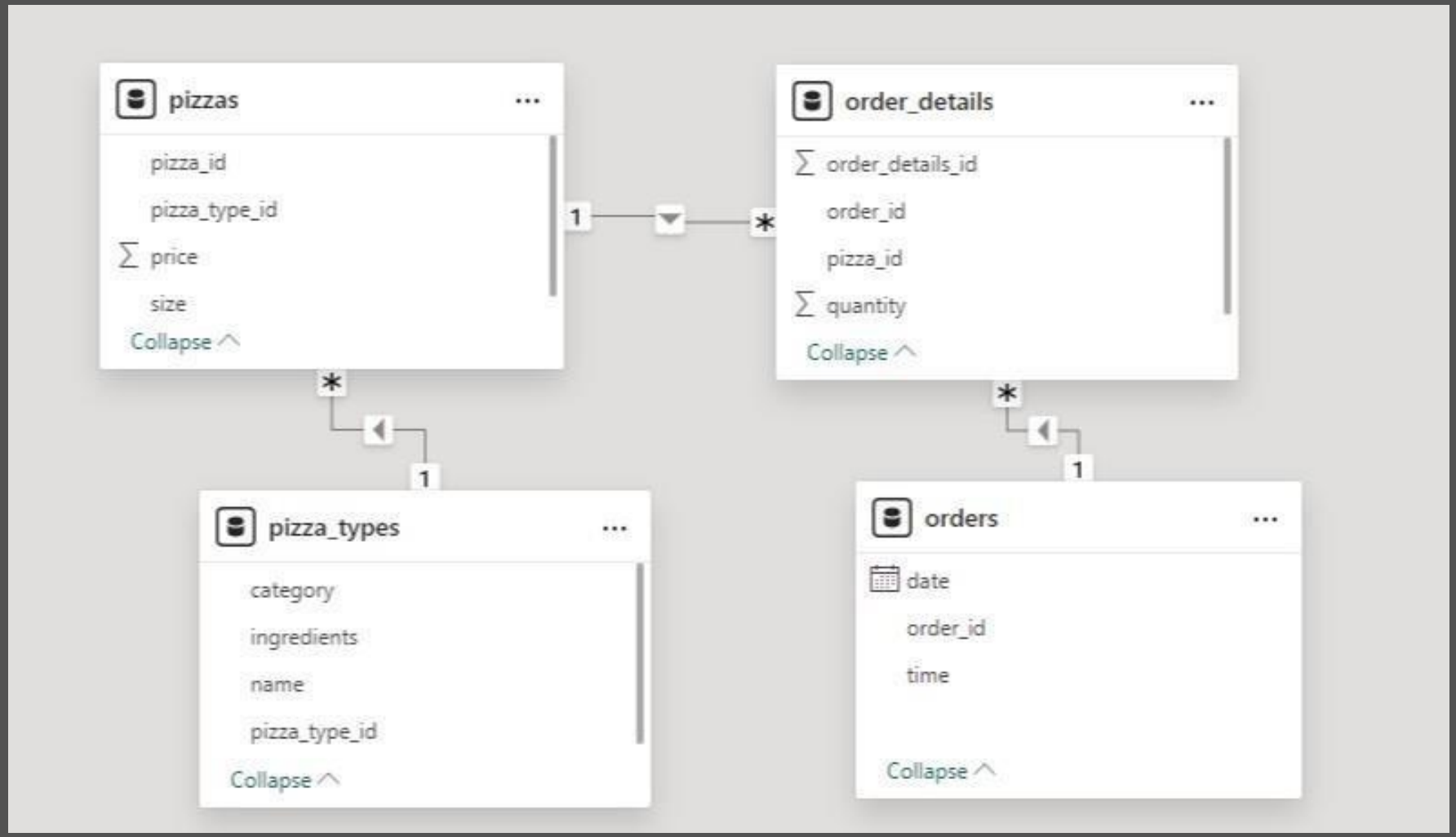
PIZZA SALES ANALYSIS

Using SQL

By Devika Lahari

Model View: Representing Relationships Between Tables



Questions

1. Retrieve the total number of orders placed
2. Calculate the total revenue generated from pizza sales
3. Identify the highest priced pizza
4. Identify the most common pizza size ordered
5. Top 5 most ordered pizza types along with their quantities
6. Join the necessary tables to find the total quantities of of each pizza category ordered
7. Detemine the distri. of orders by hour of the day
8. Find the category wise distribution of pizzas
9. Group the orders by date and calculate average number of pizzas ordered per day
10. Top 3 most ordered pizza types based on revenue
11. calculate the % contri. of eac h pizza type to total revenue
12. Analyze the cumulative revenue generated over time
13. Top 3 most ordered pizza types based on revenue for each pizza category

Objective

The objective of this project is to analyze pizza sales data to identify trends and provide actionable insights that can help to increase sales and aim to uncover key metrics and patterns within the sales data by leveraging SQL queries in MySQL.

Tools Used

- Database Management System: MySQLQuery
- Tool: MySQL Workbench

Methodology

Database Setup:

- Created a database named "pizzahut" and Imported tables (orders,pizzas, order_details, pizza_types) into the database.

Data Exploration and Cleaning:

- Explored the structure and contents of each table to understand the relationships and data points.Ensured data integrity by checking for null values, duplicates, and any inconsistencies.

Querying and Analysis:

- SQL queries were written to extract meaningful insights from the data. These queries addressed various aspects of pizza sales, including total orders, revenue generation, popular pizza types, and order distribution.

Overview

In this project, addressed several critical questions related to pizza sales. These questions are organized under different categories to provide a structured analysis:

Sales Volume and Revenue

- Retrieve the total number of orders placed:
- Calculate the total revenue generated from pizza sales:

Product Analysis

- Identify the highest-priced pizza:
- Identify the most common pizza size ordered:
- List the 5 most ordered pizza types along with their quantities:

Category and Time Analysis

- *Join the necessary tables and find the total quantity of each pizza categoryordered*
- *Determine the distribution of orders by hour of the day*
- *Join the relevant tables and find the category-wise distribution of pizzas*

Trend and Performance Analysis

- *Group the orders by date and calculate the average number of pizzas orderedper day*
- *Determine the top 3 most ordered pizza types based on revenue*
- *Calculate the percentage contribution of each pizza type to total revenueAnalyze the*
- *cumulative revenue generated over time*
- *Determine the top 3 most ordered pizza types based on revenue for each pizzacategory*

Retrieve the total number of orders placed

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

Result Grid

	total_orders
▶	21350

Calculate total revenue generated from pizza sales

```sql
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price)) AS total_sales
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
```

Result Grid

total_sales
817860

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;
```

Result Grid | Filter Rows:

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 5 most ordered pizza types along with their quantities.

```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
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
LIMIT 5;
```

Result Grid

	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

Determine the distribution orders by hour of theday

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

Result Grid | Filte

hour	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 the relative 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 average number of pizzas ordered per day

```sql
SELECT
    ROUND(AVG(quantity), 0) as avg_pizza_ordered_per_day
FROM
    (SELECT
        orders.date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.date) AS order_quantity
```

Result Grid | Filter Rows:

	avg_pizza_ordered_per_day
▶	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:

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

Calculate thepercentage contribution eachpizza 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

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

Analyze the cumulative revenue generated over time

```sql
select date,
sum(revenue) over (order by date) as cum_revenue
from
(select orders.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.date) as sales;
```

Result Grid | Filter Rows:

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
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
2015-01-18	40978.600000000006
2015-01-19	43365.75000000001

Result 1

Determine the top 3 most ordered pizza types based onrevenue 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
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

Insights

- The analysis revealed that a Large(L) size pizza is most commonly ordered.
- The Thai Chicken Pizza($43434.25), Barbecue Chicken Pizza($42768) and California Chicken pizza($41409.5) generate the highest revenue.
- The most ordered pizza types based on quantities are the classic delux pizza(2453),the barbeque chicken pizza(2432) and the Hawaiian pizza(2422)
- The highest-priced pizza is the Greek Pizza ($35.95) is contributing significantly to the revenue.
- The average number of pizzas ordered per day is 138.
- Cumulative revenue trends provide a long-term view of performance.
- Understanding the percentage contribution of each pizza type to total revenue helps in identifying customer preferences.

Conclusion

The analysis of pizza sales data reveals valuable insights for optimizing business operations and driving sales growth. By leveraging MySQL queries, we've identified total orders and revenue, as well as customer preferences and temporal patterns in ordering behavior. These findings inform actionable recommendations for menu optimization, pricing strategies, and resource allocation. Moving forward, continuous monitoring and adaptation based on data-driven insights will be crucial for sustaining competitive advantage and achieving long-term success.

THANKYOU

Devika Lahari