PizzaHut Sales Analysis using SQL

A SQL-based Data Analysis Project

Transforming PizzaHut sales data into meaningful business insights using SQL.

Diya Marvaniya
Computer Engineering Graduate | Aspiring Data Analyst
SQL • Data Analysis

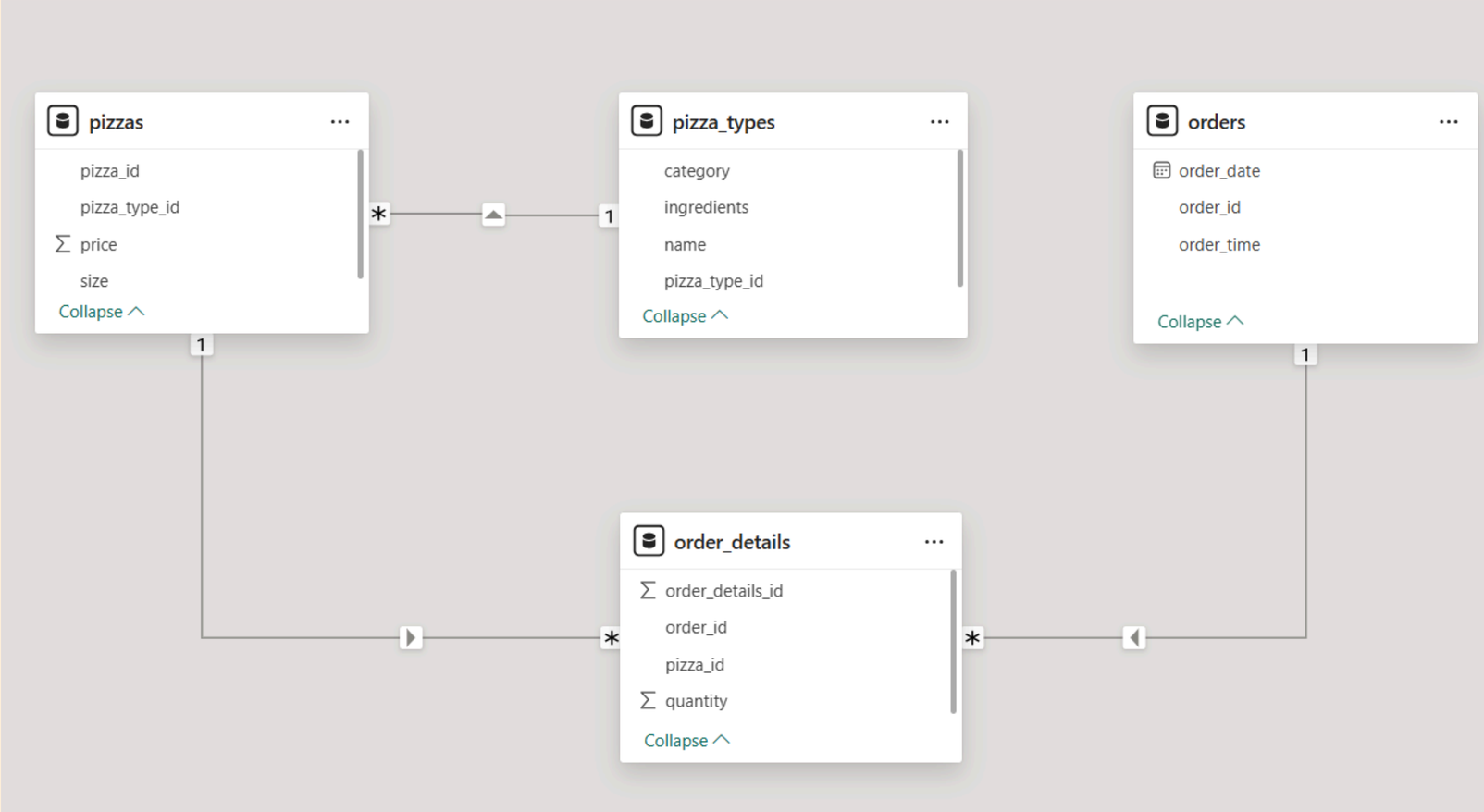
Project Overview

- This project analyzes PizzaHut sales data using SQL.
- Objective is to extract business insights from raw data.
- Used SQL queries to answer real-world business questions.

Tools & Skills Used

- SQL(Joins, Aggregation, Subqueries)
- GROUP BY, ORDER BY, Window Functions
- Aggregate Functions (SUM, COUNT, AVG)
- Data Analysis & Logical Thinking

Dataset Description



Problem Statement / Business Questions

- What is the total revenue generated?
- Which pizza category sells the most?
- What are the top 5 best-selling pizzas?
- What is the average order value?
- Which time/day has maximum orders?
- Why should customers choose us over competitors?

Query 1. Retrieve the total number of order placed

```
select count(order_id)Total_orders from orders
```

Result Grid

	Total_orders
▶	21350

Query 2. Calculate the total revenue generated from pizza sales

```sql
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

Query 3. Identify the highest-prized 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

Query 4. Identify the most common pizza size ordered

```
SELECT
    pizzas.size, COUNT(order_details.order_details_id) 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

Query 5. List the top 5 most ordered pizza types along with their quantity

```sql
SELECT
    pizza_types.name,
    SUM(order_details.quantity) most_ordered_pizzas_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 most_ordered_pizzas_quantity DESC
LIMIT 5;
```

Result Grid | Filter Rows: | Export:

name	most_ordered_pizzas_quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

Query 6. Join all 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

Query 7. Determine the distribution of orders by hours 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 Row

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

2009

Query 8. 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

Query 9. Group the orders by date and calculate the average number of pizzas order per day

```sql
SELECT
    ROUND(AVG(total_orders), 0) Average_pizzas_order_per_day
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) total_orders
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS data;
```

Result Grid | Filter Rows:

Average_pizzas_order_per_day
138

Query 10. Determine the top 3 most ordered pizza types based on revenue

```sql
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) 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

Query 11. 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

Query 12. Analyze the cumulative revenue generated over time

```
select order_date, sum(revenue) over (order by order_date) as cum_revenue from
(select orders.order_date, round(sum(order_details.quantity * pizzas.price),2) 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
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

Query 13. Determine the top 2 most ordered pizza types based on revenue for each pizza category

```
select category, name, revenue, rnk from
 (select category, name, revenue, rank() over(partition by category order by revenue desc) as rnk
 from
(select pizza_types.category, pizza_types.name,  round(sum(order_details.quantity * pizzas.price),2) as revenue
from order_details join pizzas
     on order_details.pizza_id = pizzas.pizza_id
     join pizza_types
     on pizza_types.pizza_type_id = pizzas.pizza_type_id
     group by pizza_types.category, pizza_types.name) as data) as b where rnk <=2;
```

Result Grid | Filter Rows: | Export:

category	name	revenue	rnk
Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768	2
Classic	The Classic Deluxe Pizza	38180.5	1
Classic	The Hawaiian Pizza	32273.25	2
Supreme	The Spicy Italian Pizza	34831.25	1
Supreme	The Italian Supreme Pizza	33476.75	2
Veggie	The Four Cheese Pizza	32265.7	1
Veggie	The Mexicana Pizza	26780.75	2

Key Insights / Findings

- Classic category contributes the highest sales.
- Large size pizzas generate maximum revenue.
- Peak orders observed during evening hours.
- Top 5 pizzas contribute significant portion of total revenue.
- Total revenue generated from pizza sales.
- Average pizza order per day.
- Which pizza category needs improve marketing.

Conclusion & Learnings

Conclusion

- Successfully analyzed PizzaHut sales data using SQL.
- Derived meaningful business insights from raw sales data.
- Answered key business questions related to revenue and sales performance.

Key Learnings

- Strengthened understanding of SQL joins, aggregations, and subqueries.
- Gained confidence in writing optimized SQL queries.
- Improved data analysis and problem-solving skills.

Thank you

Thank you

- Thank you for taking the time to review this project.
- Open to feedback and suggestions.
- Open to data analyst and data science internship and entry-level opportunities.

Contact

- Diya Marvaniya
- **LinkedIn**: www.linkedin.com/in/diya-marvaniya-932088381
- **GitHub**: github.com/diyamarvaniya08
- **Email:** diyamarvaniya.18@gmail.com