PIZZAHUT

SQL PROJECT: BUSINESS INSIGHTS FOR PIZZAHUT

A DATA-DRIVEN APPROACH TO UNDERSTANDING SALES PATTERNS

HARSHAL PATIL

31/05/2025

TOOL USED: MYSQL WORKBENCH

PROJECT OVERVIEW AND PROJECT OBJECTIVES

ANALYZED PIZZAHUT'S ORDER DATA TO EXTRACT MEANINGFUL BUSINESS INSIGHTS.

USED MYSQL TO WRITE AND EXECUTE SQL QUERIES.

FOCUSED ON REAL-WORLD BUSINESS QUESTIONS UNDER BASIC, INTERMEDIATE, AND ADVANCED LEVELS.

IMPROVE BUSINESS UNDERSTANDING THROUGH SQL.

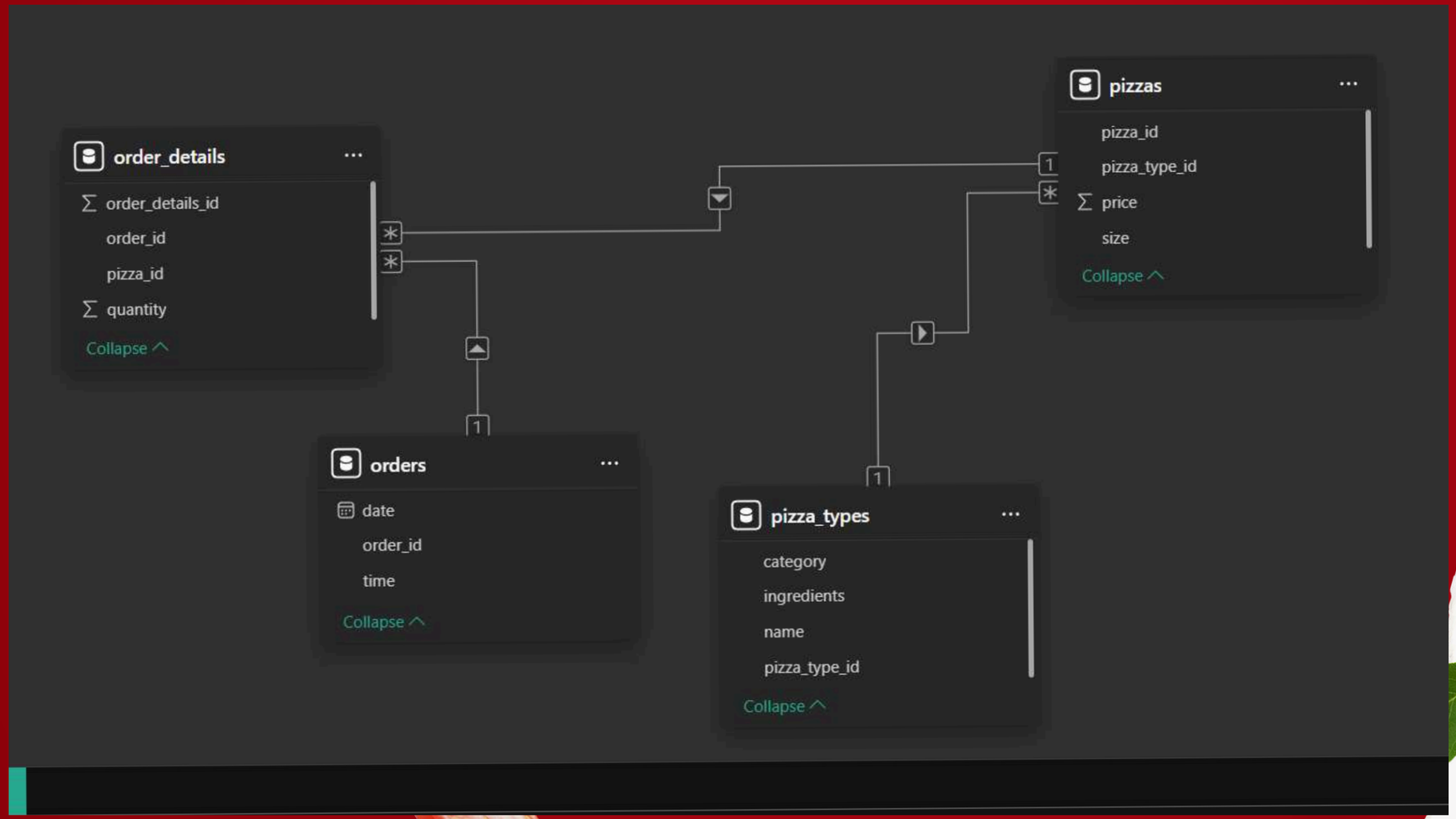
SOLVE PRACTICAL BUSINESS PROBLEMS USING STRUCTURED QUERIES.

GENERATE INSIGHTS RELATED TO SALES, CUSTOMER BEHAVIOR, AND PIZZA PERFORMANCE.

DATASET OVERVIEW

TABLES USED: ORDERS, ORDER_DETAILS, PIZZAS, PIZZA_TYPES, PIZZA_CATEGORIES

DATA TYPES: ORDERS, PIZZA PRICES, PIZZA CATEGORIES, SIZES, ORDER TIMESTAMPS, ETC.



TOTAL ORDERS

SHOWS TOTAL NUMBER OF ORDERS PLACED, INDICATING BUSINESS VOLUME. HELPS ASSESS PIZZAHUT'S OPERATIONAL THROUGHPUT.

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

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

total_orders
21350

TOTAL REVENUE FROM PIZZA SALES

DISPLAYS THE TOTAL REVENUE GENERATED FROM ALL PIZZAS SOLD. KEY INDICATOR OF FINANCIAL PERFORMANCE.

```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 | Filter Rows: | Export: | Wrap Cell Content:

total_revenue
817860.05

HIGHEST PRICED PIZZA

IDENTIFIES THE MOST EXPENSIVE PIZZA, USEFUL FOR PRICING STRATEGY AND PREMIUM PRODUCT POSITIONING.

```sql
-- Problem 3: Identify the highest-priced pizza.
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: | Export: | Wrap Cell Content:

name	price
The Greek Pizza	35.95

MOST COMMON PIZZA SIZE ORDERED HIGHLIGHTING CUSTOMER SIZE PREFERENCE, CRUCIAL FOR INVENTORY AND PACKAGING DECISIONS.

```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: | Export: | Wrap Cell Content:

size	order_count
L	18526
M	15385
S	14137
XL	544

TOP 5 MOST ORDERED PIZZA TYPES
LISTED THE BEST-SELLING PIZZAS BY QUANTITY.
GUIDES MARKETING AND MENU ENGINEERING EFFORTS.

```sql
-- Problem 5: List the top 5 most ordered pizza types along with their quantities.
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: | Export: | Wrap Cell Content:

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

Top 5 Most Ordered Pizza Types

```sql
-- problem 6: Join 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: | Export: | Wrap Cell Content:

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Orders by Hour of Day

Analyzes when customers order most. Helps optimize staffing and promotional timings.

```sql
-- Problem 7: Determine the distribution of orders by hour of the day.
SELECT
    HOUR(order_time), COUNT(order_id)
FROM
    orders
GROUP BY HOUR(order_time);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

hour(order_time)	count(order_id)
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

Category-wise Pizza Distribution

Shows how pizza types are distributed across categories.
Useful for inventory and supply chain alignment.

```sql
-- Problem 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: | Export: | Wrap Cell Content:

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

Average Pizzas Ordered per Day

Calculated daily average order volume. Useful for demand forecasting and operational planning.

```sql
-- Problem 9: Group the orders by date and calculate the average number of pizzas ordered per day.

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: | Export: | Wrap Cell Content:

avg_pizza_ordered_per_day
138

Pizza Types by Revenue

Highlighted which specific pizzas bring in the most money, guiding strategic promotions and upsells.

```sql
-- Problem 10: Determine the top 3 most ordered pizza types based on revenue.

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

Revenue Contribution by Pizza Type

Percentage share of each pizza type in total revenue. Helps identify high-ROI items.

```sql
-- problem 11: Calculate the percentage contribution of each pizza type to total revenue.

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: | Export: | Wrap Cell Content:

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

CUMULATIVE REVENUE OVER TIME

```sql
-- Problem 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,
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: | Export: | Wrap Cell Content:

order_date	cum_revenue
2015-02-03	77925.90000000002
2015-02-04	80159.80000000002
2015-02-05	82375.60000000002
2015-02-06	84885.55000000002
2015-02-07	87123.20000000001
2015-02-08	89158.20000000001
2015-02-09	91353.55000000002
2015-02-10	93410.05000000002
2015-02-11	95870.05000000002
2015-02-12	98028.85000000002
2015-02-13	100783.35000000002
2015-02-14	103102.50000000001
2015-02-15	105243.75000000001

CONCLUSION

- SUCCESSFULLY ANSWERED 17 REAL-WORLD BUSINESS QUESTIONS USING SQL.
- DEMONSTRATED SKILL IN QUERYING, JOINING, AND AGGREGATING DATA.
- GAINED PRACTICAL KNOWLEDGE FOR RETAIL/F&B ANALYTICS.

FINAL BUSINESS INSIGHTS SUMMARY

-
- PRODUCT FOCUS: HIGH-DEMAND PIZZAS AND TOP REVENUE GENERATORS SHOULD BE PRIORITIZED IN MARKETING AND INVENTORY.
- OPERATIONAL TIMING: ORDER TIMING INSIGHTS CAN ENHANCE LABOR PLANNING AND PROMOTIONAL TIMING.
- CATEGORY OPTIMIZATION: KNOWING WHICH CATEGORIES GENERATE THE MOST REVENUE HELPS IN DESIGNING FUTURE OFFERINGS.
- DATA-DRIVEN DECISIONS: SQL-BASED INSIGHTS ARE POWERFUL FOR REAL-TIME BUSINESS STRATEGY REFINEMENT.

THANK YOU!