PROJECT

PIZZA SALES ANALYSIS -- SQL

An SQL-based analysis of pizza sales data to understand revenue patterns, customer demand, and top-performing products.

PROJECT DESCRPTION

- THIS PROJECT ANALYZES PIZZA SALES DATA USING SQL TO UNCOVER BUSINESS INSIGHTS AND SUPPORT DATA-DRIVEN DECISIONS. IT FOCUSES ON EVALUATING SALES PERFORMANCE, REVENUE TRENDS, AND PRODUCT-LEVEL ANALYSIS.

- TOP-PERFORMING PIZZA TYPES AND CATEGORIES WERE IDENTIFIED TO PROVIDE ACTIONABLE BUSINESS INSIGHTS.

- THE PROJECT DEMONSTRATES THE ABILITY TO TRANSFORM RAW DATA INTO STRATEGIC INSIGHTS, SUPPORTING SALES PLANNING AND INVENTORY MANAGEMENT.

- IT HIGHLIGHTS STRONG ANALYTICAL THINKING, SQL PROFICIENCY, AND PROBLEM-SOLVING SKILLS IN A REAL-WORLD BUSINESS CONTEXT.

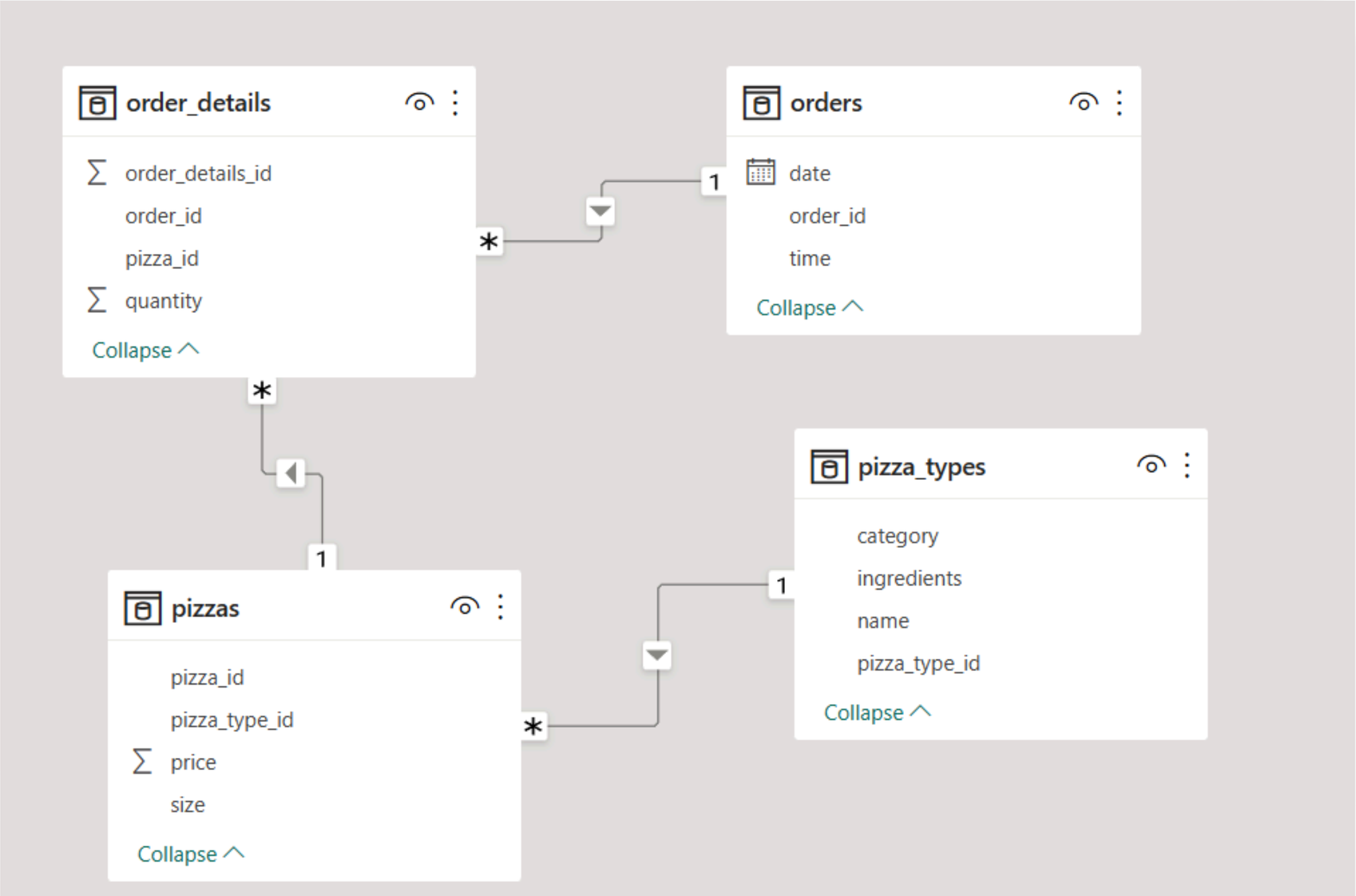
PROJECT
OBJECTIVE

The objective of this project is to analyze pizza sales data to understand overall sales performance and identify top-selling pizzas and categories. It aims to examine customer ordering patterns and determine key trends in demand. Additionally, the project calculates the revenue contribution of each pizza type and analyzes cumulative revenue over time. This analysis helps in uncovering actionable insights to guide business decisions, optimize inventory, and improve sales strategy.

DATASET OVERVIEW

Table Name	Description
orders	Contains order date and time
order_details	Records quantity of pizzas ordered
pizzas	Details pizza price and size
pizza_types	Categorizes pizzas by type and name

DATASET
MODEL



BASIC QUESTIONS

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

```
SELECT
    COUNT(order_id) AS total_orders
FROM
    pizzahut.orders;
```

Result Grid

	total_orders
▶	21350

BASIC QUESTIONS

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```
SELECT
    ROUND(SUM(o.quantity * p.price), 2) AS Total_Revenue
FROM
    pizzahut.pizzas p
        JOIN
    pizzahut.order_details o ON p.pizza_id = o.pizza_id;
```

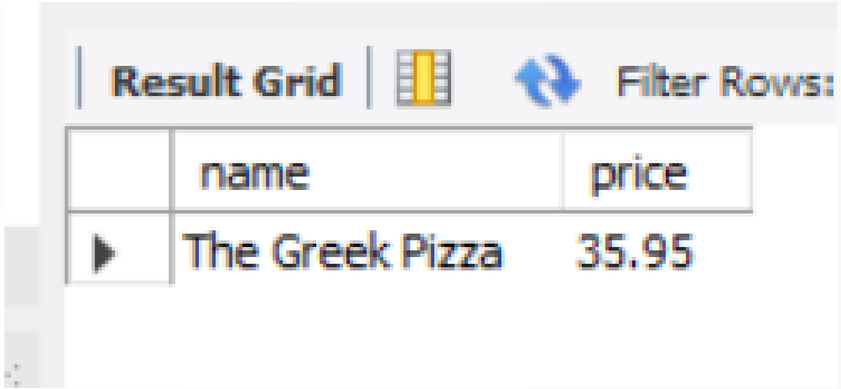
Result Grid | Filter Rows:

	Total_Revenue
▶	817860.05

BASIC QUESTIONS

IDENTIFY THE HIGHEST-PRICED PIZZA.

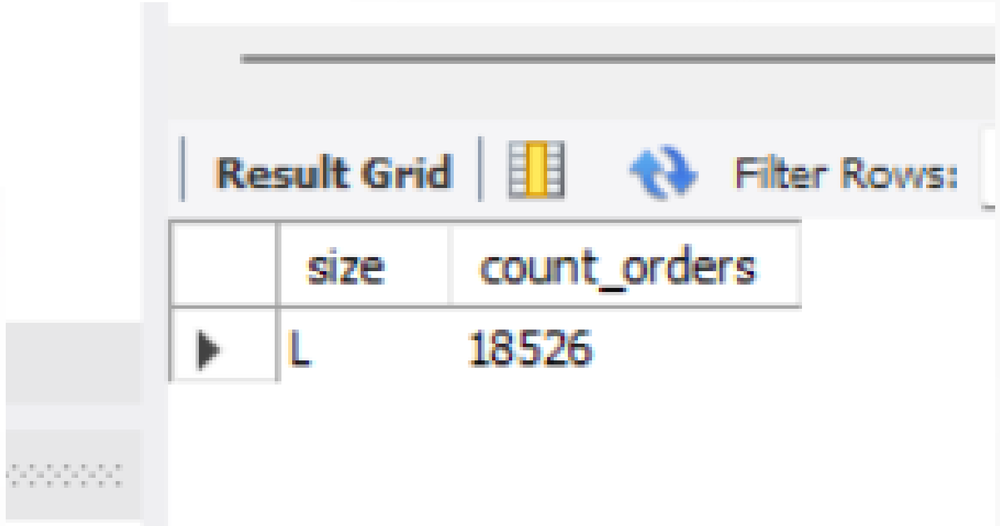
```sql
SELECT
    pt.name, p.price
FROM
    pizzahut.pizzas p
        JOIN
    pizzahut.pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
ORDER BY p.price DESC
LIMIT 1;
```



BASIC QUESTIONS

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

```
SELECT
    p.size, COUNT(od.quantity) AS count_orders
FROM
    pizzahut.pizzas p
        JOIN
    pizzahut.order_details od ON p.pizza_id = od.pizza_id
GROUP BY p.size
ORDER BY count_orders DESC
LIMIT 1;
```



BASIC QUESTIONS

LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.

```sql
SELECT
    pt.name, SUM(od.quantity) AS quantities
FROM
    pizzahut.pizzas p
        JOIN
    pizzahut.order_details od ON p.pizza_id = od.pizza_id
        JOIN
    pizzahut.pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.name
ORDER BY quantities DESC
LIMIT 5;
```

Result Grid | Filter Rows:

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

INTERMEDIATE

FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

```sql
SELECT
    pt.category AS pizza_category,
    SUM(od.quantity) AS total_quantity
FROM
    pizzahut.order_details od
        JOIN
    pizzahut.pizzas p ON od.pizza_id = p.pizza_id
        JOIN
    pizzahut.pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.category
ORDER BY total_quantity DESC;
```

Result Grid | Filter Rows:

pizza_category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

INTERMEDIATE

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

```
SELECT
    HOUR(order_time) AS hour,
    COUNT(order_id) AS distribution_of_orders
FROM
    pizzahut.orders
GROUP BY HOUR(order_time);
```

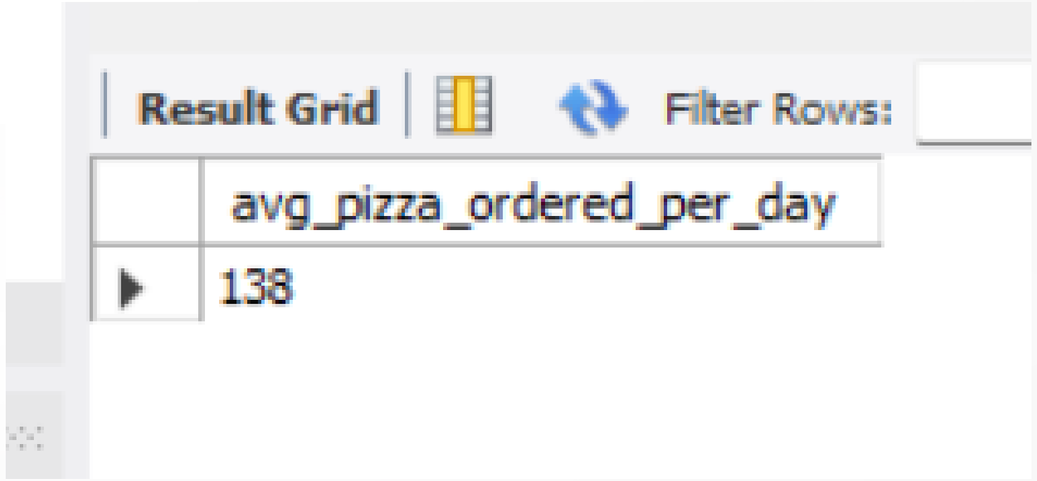
Result Grid | Filter Rows:

hour	distribution_of_orders
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

INTERMEDIATE

CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

```sql
SELECT
    ROUND(AVG(quantity), 0) AS avg_pizza_ordered_per_day
FROM
    (SELECT
        o.order_date, SUM(od.quantity) AS quantity
    FROM
        pizzahut.orders o
    JOIN pizzahut.order_details od ON o.order_id = od.order_id
    GROUP BY o.order_date) AS order_quantity;
```



INTERMEDIATE

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

```sql
SELECT
    pt.pizza_type_id,
    pt.name,
    SUM(od.quantity * p.price) AS Total_revenue
FROM
    pizzahut.pizza_types pt
        JOIN
    pizzahut.pizzas p ON pt.pizza_type_id = p.pizza_type_id
        JOIN
    pizzahut.order_details od ON p.pizza_id = od.pizza_id
GROUP BY pt.pizza_type_id , pt.name
ORDER BY total_revenue DESC
LIMIT 3;
```

Result Grid | Filter Rows: | Export:

pizza_type_id	name	Total_revenue
thai_ckn	The Thai Chicken Pizza	43434.25
bbq_ckn	The Barbecue Chicken Pizza	42768
cali_ckn	The California Chicken Pizza	41409.5

ADVANCED

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.

```sql
SELECT
    pt.category,
    CONCAT(ROUND((SUM(od.quantity * p.price) / (SELECT
                            ROUND(SUM(o.quantity * p.price), 2) AS Total_Revenue
                        FROM
                            pizzahut.pizzas p
                                JOIN
                            pizzahut.order_details o ON p.pizza_id = o.pizza_id)) * 100,
                    2),
            '%') AS revenue
FROM
    pizzahut.pizza_types pt
        JOIN
    pizzahut.pizzas p ON pt.pizza_type_id = p.pizza_type_id
        JOIN
    pizzahut.order_details od ON p.pizza_id = od.pizza_id
GROUP BY pt.category;
```

Result Grid | Filter Rows:

category	revenue
Classic	26.91%
Veggie	23.68%
Supreme	25.46%
Chicken	23.96%

ADVANCED

TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY.

```sql
with cte as(
select pt.category,pt.name,sum(od.quantity*p.price) as revenue,
rank() over( partition by category order by sum(od.quantity*p.price) desc) as rn
from pizzahut.pizza_types pt
join pizzahut.pizzas p
on pt.pizza_type_id=p.pizza_type_id
join pizzahut.order_details od
on od.pizza_id=p.pizza_id
group by pt.category,pt.name
)
select category,name,revenue
from cte
where rn<=3
```

Result Grid | Filter Rows: | Export: | Wrap C

category	name	revenue
Chicken	The Thai Chicken Pizza	43434.25
Chicken	The Barbecue Chicken Pizza	42768
Chicken	The California Chicken Pizza	41409.5
Classic	The Classic Deluxe Pizza	38180.5
Classic	The Hawaiian Pizza	32273.25
Classic	The Pepperoni Pizza	30161.75
Supreme	The Spicy Italian Pizza	34831.25
Supreme	The Italian Supreme Pizza	33476.75
Supreme	The Sicilian Pizza	30940.5
Veggie	The Four Cheese Pizza	32265.70000000065
Veggie	The Mexicana Pizza	26780.75
Veggie	The Five Cheese Pizza	26066.5

ADVANCED

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
select order_date, sum(revenue) over(order by order_date) as cum_revenue
from
(
select o.order_date, sum(od.quantity*p.price) as revenue
from pizzahut.orders o
join pizzahut.order_details od
on o.order_id=od.order_id
join pizzahut.pizzas p
on od.pizza_id=p.pizza_id
group by o.order_date) as Sales;
```

order_date	cum_revenue
2015-01-01	2713.850000000004
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

THANK YOU