PIZZA SALES DATA ANALYSIS

Data -Driven Insights using SQL

Presented By: Pooja Sunil Palde

OBJECTIVE:

To analyze pizza sales data to provide actionable insights for improving operational efficiency and revenue generation.

Sub-Objectives:

- Identify top-selling pizzas and their contribution to revenue.
- Analyze sales trends by time, size, and type.
- Highlight areas for growth and customer preferences.
- Improve decision-making for inventory and marketing strategies.

BACKGROUND AND MOTIVATION :

Background:

Understanding sales patterns is critical in the competitive food industry.
SQL enables efficient data storage, querying, and analysis for actionable insights.

Motivation:

Provide data-driven recommendations to optimize menu offerings and operational strategies.

DATASET DESCRIPTION:

The project uses a pizza sales dataset with the following tables:

- Orders: Contains order information like order ID and timestamp.
- Order Details: Includes pizza ID, quantity, and size.
- Pizzas: Contains pizza details such as name, size, and price.
- Pizza Types: Describes the ingredients of each pizza type.

METHODOLOGY :

1. Data Import and Cleaning: Loaded CSV files into SQL database and addressed inconsistencies.

2. Querying: Used SQL queries to extract meaningful insights.

3. Visualization: Represented findings through charts and graphs.

QUERY 1: RETRIVE THE TOTAL NUMBER OF ORDERS PLACED.

```
3 •    SELECT
4          COUNT(order_id) AS total_orders
5      FROM
6          orders;
```

Result Grid

	total_orders
▶	21350

QUERY 2: CALCULATE THE TOTAL REVENUE GENRATED FROM PIZZA SALES.

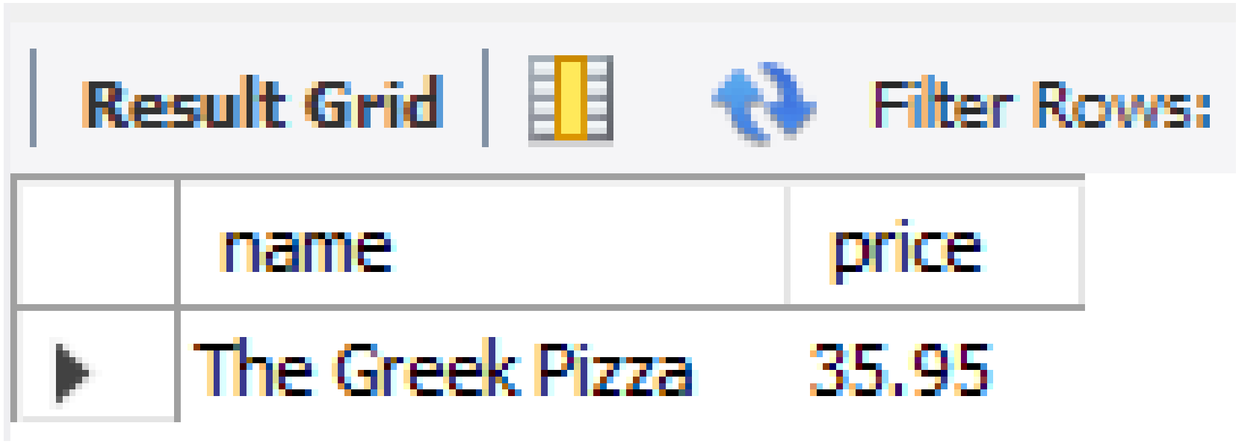
```
SELECT
    SUM(order_details.quantity * pizzas.price) AS total_sales
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
```

Result Grid | Filter

	total_sales
▶	817860.049999993

QUERY 3: IDENTITY 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;
```



	name	price
▶	The Greek Pizza	35.95

QUERY 4: IDENTIFY THE MOST COMMAN 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 I

size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

QUERY 5: JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY.

```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 Rc

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

QUERY 6: DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

```sql
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
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198

QUERY 7: CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.

```
select pizza_types.category,
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 order by revenue desc;
```

sult Grid | Filter Rows:

category	revenue
Classic	220053.1000000001
Supreme	208196.99999999822
Chicken	195919.5
Veggie	193690.45000000298

QUERY 9: ANALYSE THE CUMULATIVE REVENUE GENERATED OVER TIME.

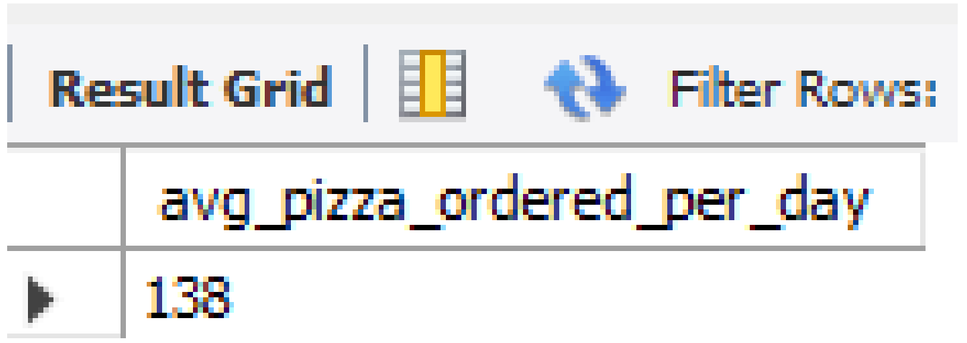
```
select order_date,
sum(revenue )over(order by order_date) as cum_revenue
from
(select orders.order_date,
sum(order_details.quantity*pizzas.price)as revenue
from order_details join pizzas
on order_details.pizza_id=pizzas.pizza_id
join orders
on orders.order_id=order_details.order_id
group by orders.order_date)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

sult Grid | Filter Rows:

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

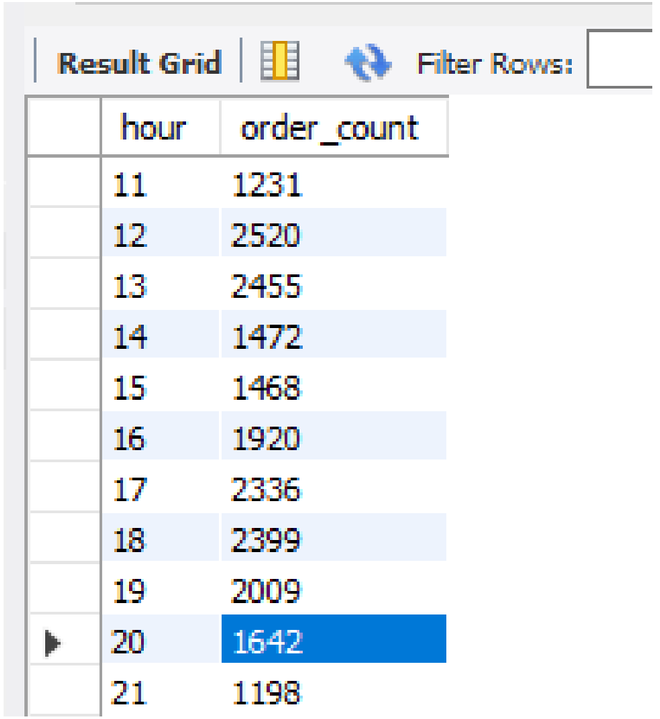
QUERY 10: DETERMINE THE MOST THREE ORDERED PIZZAS TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY.

```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;
```



QUERY 11: DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

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



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

QUERY 12: LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THIER QUANTITY.

```
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
```

ult 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

KEY FINDINGS AND RECOMMENDATIONS

Findings:

- Majority of orders are for large-sized pizzas.
- Revenue is highest during dinner hours (7 PM - 9 PM).
- Some pizza types are underperforming, indicating potential for menu optimization.

Recommendations:

- Increase promotions for popular pizzas during peak hours.
- Consider phasing out or revamping least popular items.
- Optimize inventory to align with sales trends.

conclusion:

SQL effectively analyzed pizza sales data to provide actionable insights.
The project highlights data-driven decision-making's role in improving operational efficiency and profitability.

"This project demonstrates how SQL-based data analysis can transform raw data into valuable business insights."

THANK YOU