PIZZA SALES

welcome to

sql project

Hello everyone, my name is Nikita Shinde. In this project, I have utilized SQL queries to analyze and answer various business-related questions based on pizza sales data. The objective is to extract valuable insights such as sales trends, customer preferences, and product performance to support data-driven decision-making.

INTRODUCTION

- SQL (Structured Query Language) is used to manage and manipulate databases.
- It helps to retrieve, insert, update, and delete data efficiently.
- This project focuses on learning SQL through Question–Answer based queries.
- It improves practical understanding of database concepts.

SQL Commands Used :-

- SELECT – Retrieve data from tables.
- INSERT – Add new records.
- UPDATE – Modify existing data.
- DELETE – Remove records.
- WHERE, ORDER BY, GROUP BY – For filtering and sorting

o

➢ RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED ?

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

Output :-

Result Grid

	total_orders
▶	21350

Calculate the total revenue generated from pizza sales ?

```
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
```

Output :-

Result Grid

	total_sales
▶	817860.05

➢ 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;
```

Output :-

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 ASC;
```

Output :-

Result Grid | Filter

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

along with their quantities?

```sql
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;
```

Output :-

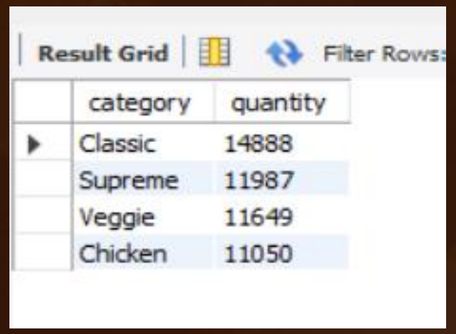
Result Grid | Filter Rows:

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;
```

Output :-



Result Grid | Filter Rows:

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hour of the day?

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

Output :-

	HOUR(order_time)	order_count
▶	11	1231
	12	2520
	13	2455
	14	1472
	15	1468
	16	1920
	17	2336
	18	2399
	19	2009
	20	1642
	21	1198

➢ Join relevant tables to find the category-wise distribution of pizzas?

```
select category , count(name) from pizza_types
group by category;
```

Output :-

	category	count(name)
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

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;
```

Output :-

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

Group the orders by date and calculate the average number of pizzas ordered per day?

```
SELECT
    ROUND(AVG(quantity), 1)
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;
```

Output :-

	ROUND(AVG(quantity), 1)
▶	138.5

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) *
        2) 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.category
ORDER BY revenue DESC;
```

Output :-

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

Determine the top 3 most ordered pizza types based on revenue 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;
```

Output :-

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

Analyze the cumulative revenue generated over time ?

```sql
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;
```

Output :-

order_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
2015-01-20	45763.65000000001
2015-01-21	47804.20000000001
2015-01-22	50300.90000000001

BENEFITS

- Easy to understand SQL concepts.
- Practical and exam-oriented.
- Helpful for beginners and engineering students.
- Can be extended with advanced queries.

CONCLUSION

- SQL is an essential skill for database management.
- Learning through Question–Answer format makes it more effective.
- This project builds a strong foundation in SQL.
- Future scope includes advanced queries and real databases.

THANK YOU !