SQL PROJECT

ON

PIZZA SALES

- Piyush Rajput

INTRODUCTION

"The goal of this project is to analyze pizza sales data to identify trends, understand customer preferences, and optimize inventory management. By utilizing SQL, my aim to provide actionable insights that can drive business growth."

In this project, I have utilized SQL Query to solve questions that were related to Pizza Sales.

1. RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED

Query

Output


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


Result Grid	
	total_orders
▶	21350

2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES

Query

Output


```
SELECT
    round(SUM(od.quantity * p.price),2) AS total_revenue
FROM
    order_details od
        JOIN
    pizzas p ON od.pizza_id = p.pizza_id;
```


	total_revenue
▶	817860.05

3. IDENTIFY THE HIGHEST-PRICED PIZZA

Query

Output

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

Result Grid			Filte
	highest_priced_pizza		
▶	The Greek Pizza		

4. IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

Query

Output

```
SELECT
    p.size, COUNT(od.order_details_id) AS order_count
FROM
    pizzas p
        JOIN
    order_details od ON p.pizza_id = od.pizza_id
GROUP BY p.size
ORDER BY order_count DESC;
```

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

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

Query

Output

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

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

6. JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED

Query

Output

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

category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

7. DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY

Query

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

Output

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

8. JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS

Query

Output

```sql
SELECT
    category, COUNT(name) AS pizza_count
FROM
    pizza_types
GROUP BY category;
```

category	pizza_count
Chicken	6
Classic	8
Supreme	9
Veggie	9

9. GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY

Query

Output

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

pizza_order_count
138

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

Query

Output

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

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

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

Query

Output

```sql
SELECT
    pt.category,
    ROUND(SUM(od.quantity * p.price) / (SELECT
                ROUND(SUM(od.quantity * p.price), 2) AS total_sales
        FROM
                order_details od
                    JOIN
                pizzas p ON od.pizza_id = p.pizza_id) * 100,
        2) AS revenue
FROM
    pizza_types pt
        JOIN
    pizzas p ON p.pizza_type_id = pt.pizza_type_id
        JOIN
    order_details od ON od.pizza_id = p.pizza_id
GROUP BY pt.category
ORDER BY revenue DESC;
```

Result Grid		Filter Rows:
	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

12. ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME

Query

Output

```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 order_details od
join pizzas p on
od.pizza_id = p.pizza_id
join orders o on o.order_id = od.order_id group by o.order_date )as sales;
```

order_date	cum_revenue
2015-01-01	2713.8500000000004
2015-01-02	5445.75
2 2015-01-03 108.15	
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

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

Output

Query

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

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