SQL PROJECT ON PIZZA SALES

HELLO!

My name is Rohit Singh. In this project, I utilized SQL queries to analyze and derive insights from pizza sales data. The queries were designed to extract key metrics such as total sales, popular pizza types, customer purchase trends, , sales performance over time and many more. Through this analysis, I aimed to provide valuable insights that could help optimize inventory, improve sales strategies, and enhance customer satisfaction.

VISSION & MISSION

VISSION

To enhance business decision-making and sales growth in the pizza industry through data-driven insights.

MISSION

To analyze pizza sales data using SQL, uncover key insights, and provide actionable recommendations to improve inventory, customer satisfaction, and profitability.

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

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

Result Grid

	total_orders
▶	21350

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```sql
SELECT
    round(SUM(p1.quantity * p2.price), 2) AS Total_revenue
FROM
    orders_details AS p1
        JOIN
    pizzas AS p2 ON p1.pizza_id = p2.pizza_id;
```

Result Grid

	Total_revenue
▶	817860.05

IDENTIFY THE HIGHEST-PRICED PIZZA.

```sql
SELECT
    p1.name, p2.price
FROM
    pizza_types AS p1
        JOIN
    pizzas AS p2 ON p1.pizza_type_id = p2.pizza_type_id
ORDER BY p2.price DESC
LIMIT 1;
```

Result Grid | Filter Ro

	name	price
▶	The Greek Pizza	35.95

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

```sql
SELECT
    p1.size, COUNT(p2.quantity) AS order_count
FROM
    pizzas AS p1
        JOIN
    orders_details AS p2 ON p1.pizza_id = p2.pizza_id
GROUP BY p1.size
ORDER BY order_count DESC
LIMIT 1;
```

Result Grid | Filter

	size	order_count
▶	L	18526

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

```sql
SELECT
    p1.name, SUM(p3.quantity) as pizzas_sold
FROM
    pizza_types AS p1
        JOIN
    pizzas AS p2 ON p2.pizza_type_id = p1.pizza_type_id
        JOIN
    orders_details AS p3 ON p3.pizza_id = p2.pizza_id
GROUP BY p1.name
ORDER BY SUM(p3.quantity) DESC
LIMIT 5;
```

Result Grid | Filter Rows:

name	pizzas_sold
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
    p1.category, SUM(p3.quantity) as pizzas_sold
FROM
    pizza_types AS p1
        JOIN
    pizzas AS p2 ON p2.pizza_type_id = p1.pizza_type_id
        JOIN
    orders_details AS p3 ON p3.pizza_id = p2.pizza_id
GROUP BY p1.category
ORDER BY SUM(p3.quantity) DESC
```

Result Grid | Filter Rows:

pizza_id	pizza_type_id	size	price
bbq_ckn_s	bbq_ckn	S	12.75
bbq_ckn_m	bbq_ckn	M	16.75
bbq_ckn_l	bbq_ckn	L	20.75
cali_ckn_s	cali_ckn	S	12.75
cali_ckn_m	cali_ckn	M	16.75
cali_ckn_l	cali_ckn	L	20.75
ckn_alfredo_s	ckn_alfredo	S	12.75
ckn_alfredo_m	ckn_alfredo	M	16.75
ckn_alfredo_l	ckn_alfredo	L	20.75
ckn_pesto_s	ckn_pesto	S	12.75

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

hour	order_count
9	1
10	8
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399

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

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

Result Grid | Filter Ro

category	pizza_count
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

```sql
SELECT
    ROUND(AVG(pizza_quantity), 0) AS avg_pizza_ordered_perday
FROM
    (SELECT
        p1.order_date, SUM(p2.quantity) AS pizza_quantity
    FROM
        orders AS p1
    JOIN orders_details AS p2 ON p1.order_id = p2.order_id
    GROUP BY p1.order_date) AS order_quantity;
```

Result Grid | Filter Rows:

avg_pizza_ordered_perday
138

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

```sql
SELECT
    p1.name, SUM(p3.quantity * p2.price) AS revenue
FROM
    pizza_types AS p1
        JOIN
    pizzas AS p2 ON p1.pizza_type_id = p2.pizza_type_id
        JOIN
    orders_details AS p3 ON p2.pizza_id = p3.pizza_id
GROUP BY p1.name
ORDER BY revenue DESC
LIMIT 3;
```

Result Grid | Filter Rows:

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

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

```sql
SELECT
    p1.category,
    CONCAT(ROUND(((ROUND(SUM(p3.quantity * p2.price), 0)) / (SELECT
                            SUM(t1.quantity * t2.price)
                        FROM
                            orders_details AS t1
                                JOIN
                            pizzas AS t2 ON t1.pizza_id = t2.pizza_id)) * 100,
                    2),
            '%') AS revenue_percentage
FROM
    pizza_types AS p1
        JOIN
    pizzas AS p2 ON p1.pizza_type_id = p2.pizza_type_id
        JOIN
    orders_details AS p3 ON p2.pizza_id = p3.pizza_id
GROUP BY p1.category
ORDER BY revenue_percentage DESC
;
```

Result Grid | Filter Rows:

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

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
select order_date,
sum(revenue) over(order by order_date) as cum_revenue
from
(SELECT
    p1.order_date, round(SUM(p2.quantity * p3.price),0) AS revenue
FROM
    orders AS p1
        JOIN
    orders_details AS p2 ON p1.order_id = p2.order_id
        JOIN
    pizzas AS p3 ON p2.pizza_id = p3.pizza_id
GROUP BY p1.order_date ) as sales
```

Result Grid | Filter Rows:

order_date	cum_revenue
2015-01-01	2714
2015-01-02	5446
2015-01-03	8108
2015-01-04	9863
2015-01-05	11929
2015-01-06	14358
2015-01-07	16560
2015-01-08	19398
2015-01-09	21525
2015-01-10	23989

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

```
with ranked_pizza as (select name, category, revenue,
row_number() over(partition by category  order by revenue desc) as rank
from
(SELECT
    p1.name, p1.category, round(SUM(p3.quantity * p2.price),0) AS revenue
FROM
    pizza_types AS p1
        JOIN
    pizzas AS p2 ON p1.pizza_type_id = p2.pizza_type_id
        JOIN
    orders_details AS p3 ON p2.pizza_id = p3.pizza_id
GROUP BY p1.name, p1.category) as category_revenue)
select name,category, revenue, rank from ranked_pizza
where rank<=3
```

Result Grid | Filter Rows: | Export:

name	category	revenue	rank
The Thai Chicken Pizza	Chicken	43434	1
The Barbecue Chicken Pizza	Chicken	42768	2
The California Chicken Pizza	Chicken	41410	3
The Classic Deluxe Pizza	Classic	38180	1
The Hawaiian Pizza	Classic	32273	2
The Pepperoni Pizza	Classic	30162	3
The Spicy Italian Pizza	Supreme	34831	1
The Italian Supreme Pizza	Supreme	33477	2
The Sicilian Pizza	Supreme	30940	3
The Four Cheese Pizza	Veggie	32266	1


THANK YOU!