PIZZAHUT
Pizza Resto

PIZZA SALES ANALYSIS

- WHERE EVERY SLICE TELLS A STORY

PIZZAHUT
Pizza Resto

ABOUT SQL PROJECT

PIZZAHUT SALES ANALYSIS

I created a SQL based project inspired by PizzaHut sales data to analyze customer behavior and sales performance. Using MySQL, I wrote and executed several queries to solve real-world business problems, including:

- 🍕 Analyzing total sales and revenue
- 📊 Identifying best-selling pizza types and categories
- 📅 Tracking sales trends by day and time
- 📍 Finding top performing branches and locations
- 🧾 Calculating average order values and quantities
- 🎯 Filtering data using GROUP BY, ORDER BY, and WHERE clauses for deeper insights

QUERY : 1

Retrieve the total number of orders placed

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

Result Grid

	total_orders
▶	21350

QUERY :2

Calculate the total revenue generated from pizza sales

```
SELECT
    ROUND(SUM(orders_details.quantity * pizzas.price),
            2) AS total_sales
FROM
    orders_details
        JOIN
    pizzas ON pizzas.pizza_id = orders_details.pizza_id;
```

Result Grid

	total_sales
▶	817860.05

QUERY :3

Identify the highest-priced pizza

```sql
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;
```

Result Grid | Filter Rows:

name	price
The Greek Pizza	35.95

QUERY :4

Identify the most common pizza size ordered

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

Result Grid

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

QUERY :5

List the top 5 most ordered pizza types along with their quantities

```sql
select pizza_types.name,
sum(orders_details.quantity) as quantity
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join orders_details
on orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.name order by quantity desc limit 5;
```

sult 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

QUERY :6

Join the necessary tables to find the total quantity of each pizza category ordered

```
select pizza_types.category,
sum(orders_details.quantity) as quantity
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join orders_details
on orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.category order by quantity desc;
```

Result Grid

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

QUERY :7

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

hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642

QUERY :8

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

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

Result Grid | Filter Rov

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

QUERY :9

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

```
select avg(quantity) from
(select orders.order_date, sum(orders_details.quantity) as quantity
from orders join orders_details
on orders.order_id = orders_details.order_id
group by orders.order_date) as order_quantity ;
```

Result Grid

	avg(quantity)
▶	138.4749

QUERY :10

Determine the top 3 most ordered pizza types based on revenue

```
select pizza_types.name,
sum(orders_details.quantity * pizzas.price) as revenue
from pizza_types join pizzas
on pizzas.pizza_type_id = pizza_types.pizza_type_id
join orders_details
on orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.name order by revenue desc limit 3;
```

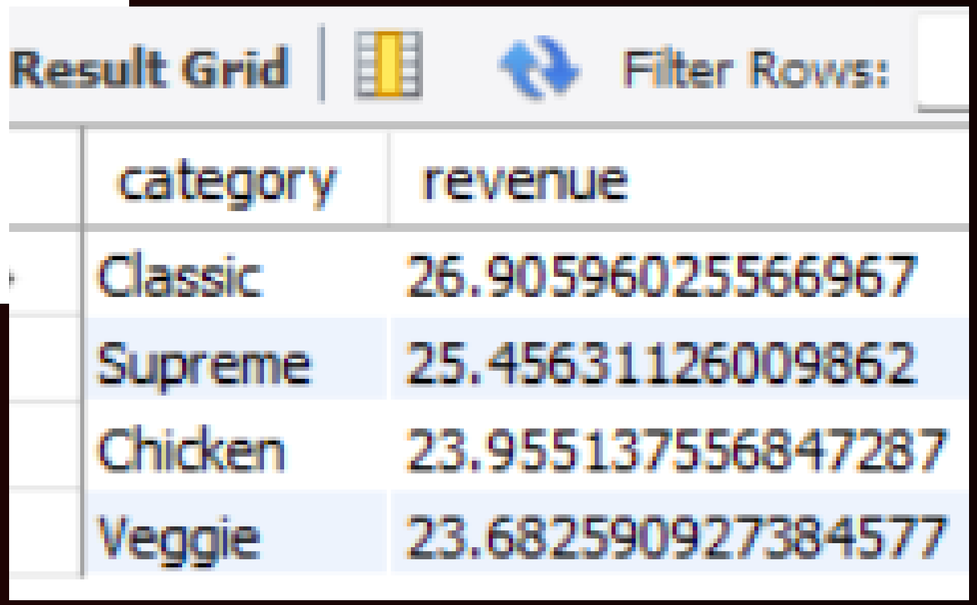
Result Grid | Filter Rows:

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

QUERY :11

Calculate the percentage contribution of each pizza type to total revenue

```
select pizza_types.category,
(sum(orders_details.quantity*pizzas.price) / (select
round(sum(orders_details.quantity * pizzas.price), 2) as total_sales
from orders_details join pizzas
on pizzas.pizza_id = orders_details.pizza_id) )*100 as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join orders_details
on orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.category order by revenue desc;
```



category	revenue
Classic	26.90596025566967
Supreme	25.45631126009862
Chicken	23.955137556847287
Veggie	23.682590927384577

QUERY :12

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(orders_details.quantity * pizzas.price) as revenue
from orders_details join pizzas
on orders_details.pizza_id = pizzas.pizza_id
join orders
on orders.order_id = orders_details.order_id
group by orders.order_date) as sales;
```

Result Grid | Filter Rows:

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

PIZZAHUT
Pizza Resto

THIS PROJECT HELPED ME IMPROVE MY SQL SKILLS USING CONCEPTS LIKE JOINS, AGGREGATION FUNCTIONS, SUBQUERIES, AND DATA-DRIVEN DECISION-MAKING.

THANK YOU FOR WATCHING