PROJECT
Portfolio

Name Vishal Kumar Dagur

PIZZA SALES

- USING MYSQL

PROJECT
Portfolio

Home | About | Contact

ABOUT ME

- Aspiring Operations & Data Professional with a strong foundation in SQL and data analysis
- Currently pursuing MBA in Operations Management from Delhi University (SOL)
- Practical experience in procurement, dispatch, and supply chain coordination
- Skilled in MySQL, Excel (Pivot Tables, Macros), and analytical problem-solving
- Passionate about using data to drive insights and optimize operations
- Focused on transitioning into data-driven roles within supply chain and operations analytics

PROJECT
Portfolio

Question | Answer | MySQL

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

```
2
3   SELECT
4       COUNT(order_id) AS total_orders
5   FROM
6       orders;
7
```

Result Grid

total_orders
21350

PROJECT
Portfolio

Question | Answer | MySQL

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES

```sql
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;
```

Result Grid

total_sales
817860.05

PROJECT
Portfolio

Question | Answer | MySQL

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

name	price
The Greek Pizza	35.95

PROJECT
Portfolio

Question | Answer | MySQL

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

```sql
SELECT
    quantity, COUNT(order_details_id)
FROM
    order_details
GROUP BY quantity;

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 Rows: Search

quantity	COUNT(order_details_id)
1	47693
2	903
3	21
4	3

PROJECT
Portfolio

Question | Answer | MySQL

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

```sql
SELECT
    pizza_types.name, SUM(order_details.quantity) AS qty
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 qty DESC
LIMIT 5;
```

Result Grid | Filter Rows:

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

PROJECT
Portfolio

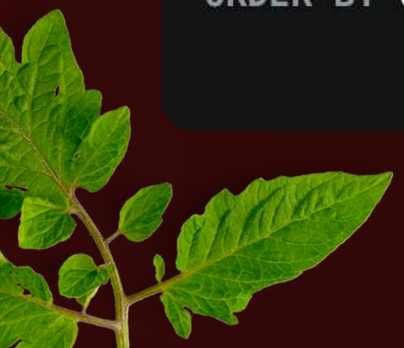
Question | Answer | MySQL

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;
```

Result Grid | Filter Ro

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050



PROJECT
Portfolio

Question | Answer | MySQL

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 Ro

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

PROJECT
Portfolio

Question | Answer | MySQL

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

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

Result Grid | Filter R

category	COUNT(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

PROJECT
Portfolio

Question | Answer | MySQL

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

```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;
```

Result Grid | Filter Rows:

avg_pizza_ordered_per_day
138

PROJECT
Portfolio

Question | Answer | MySQL

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;
```

Result Grid | Filter Rows:

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

PROJECT
Portfolio

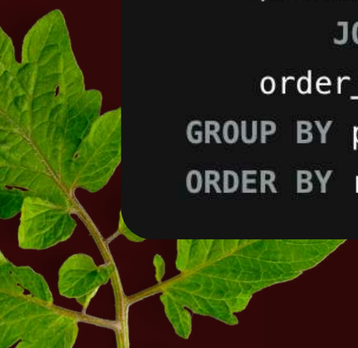
Question | Answer | MySQL

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE

```sql
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) * 100,
            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;
```

Result Grid

	category	revenue
	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68



PROJECT
Portfolio

Question | Answer | MySQL

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;
```

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

PROJECT
Portfolio

Question | Answer | MySQL

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;
```

Result Grid | Filter Rows: Search

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

WHAT I LEARNED

- Strengthened my understanding of SQL fundamentals including JOIN, GROUP BY, ORDER BY, LIMIT, and WHERE clauses.
- Gained hands-on experience in data aggregation, using functions like SUM(), COUNT(), ROUND(), and AVG().
- Learned to extract meaningful business insights from raw data — such as top-selling products, order patterns, and revenue trends.
- Practiced window functions like RANK() and SUM() OVER() to perform advanced analytics.
- Understood the importance of data relationships across multiple tables and how to join them effectively for reporting.
- Developed confidence in handling real-world datasets and generating reports that could support business decisions.

PROJECT
Portfolio

Home About Contact

THANK YOU

FOR ATTENTION

2025 MYSQL PROJECT