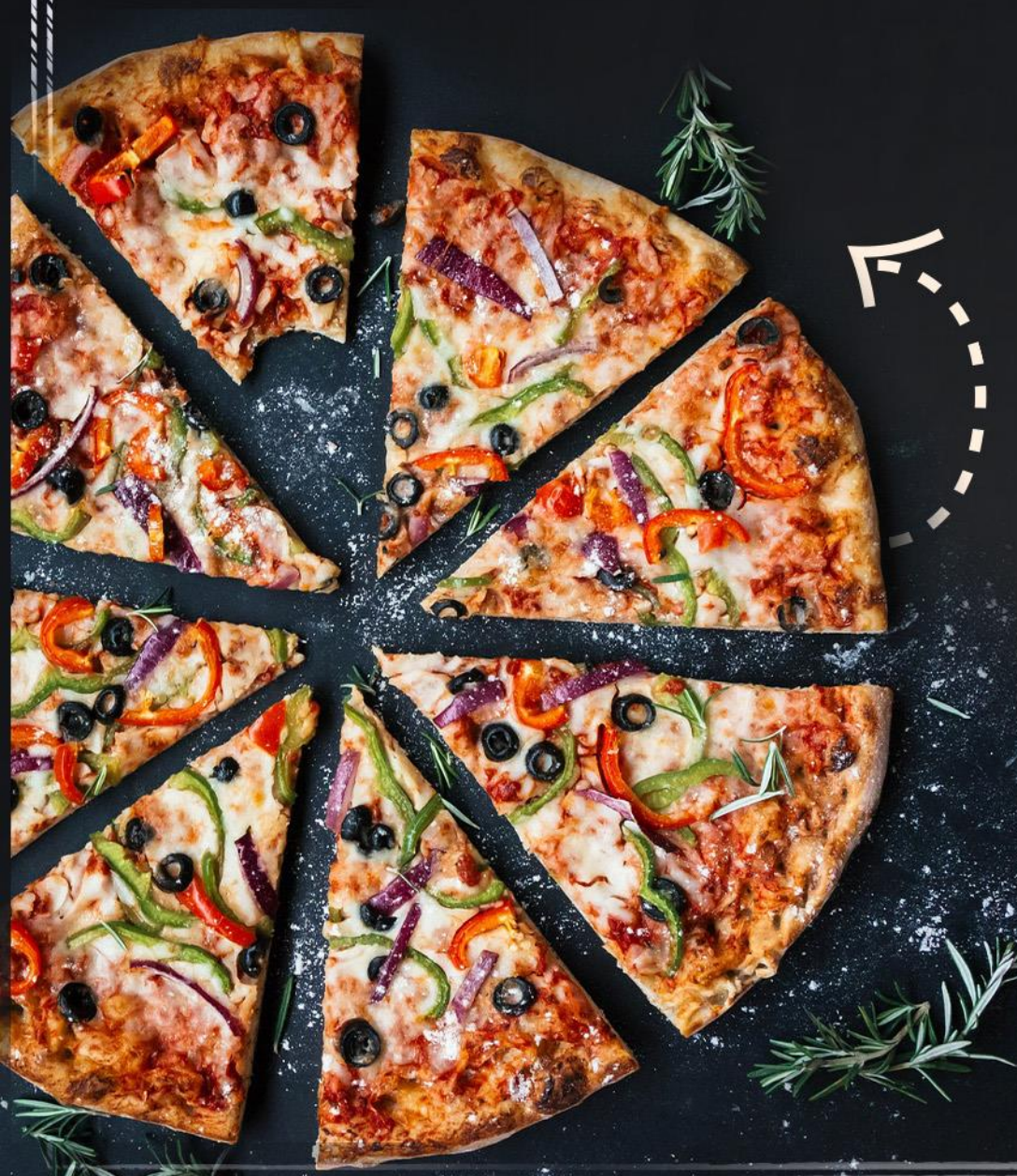
DATA ANALYSIS OF SALES IN

PIZZA HUB

--by using SQL

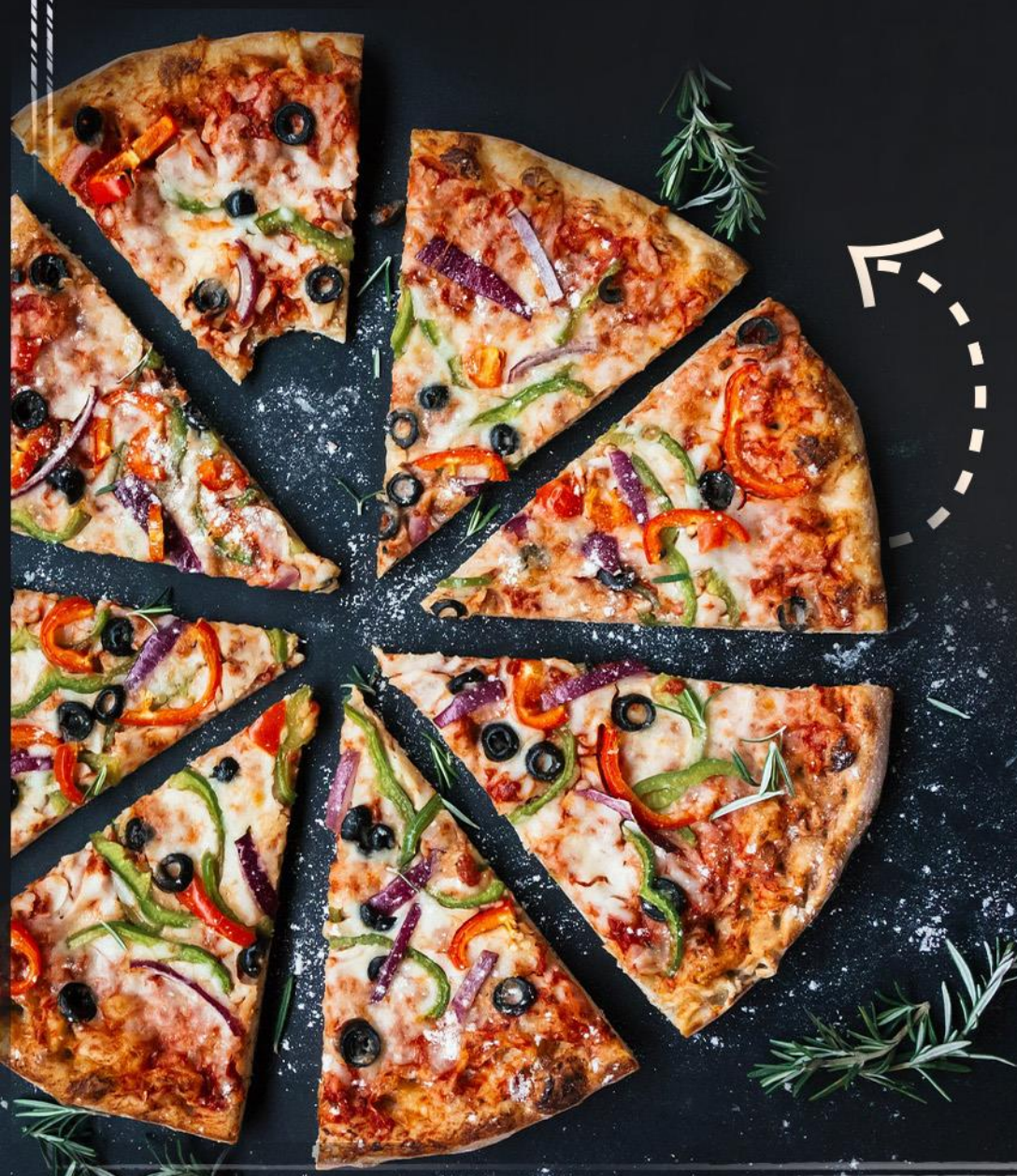
A PROJECT REPORT BY ARPIT SHUKLA



PROJECT OVERVIEW-

This project involves extracting, transforming, and analyzing the pizza sales data from various tables in a MySQL database. By employing SQL joins and functions, and aim to generate comprehensive rep orts and visualizations that will help Pizzahub Brand make data-driven decisions.

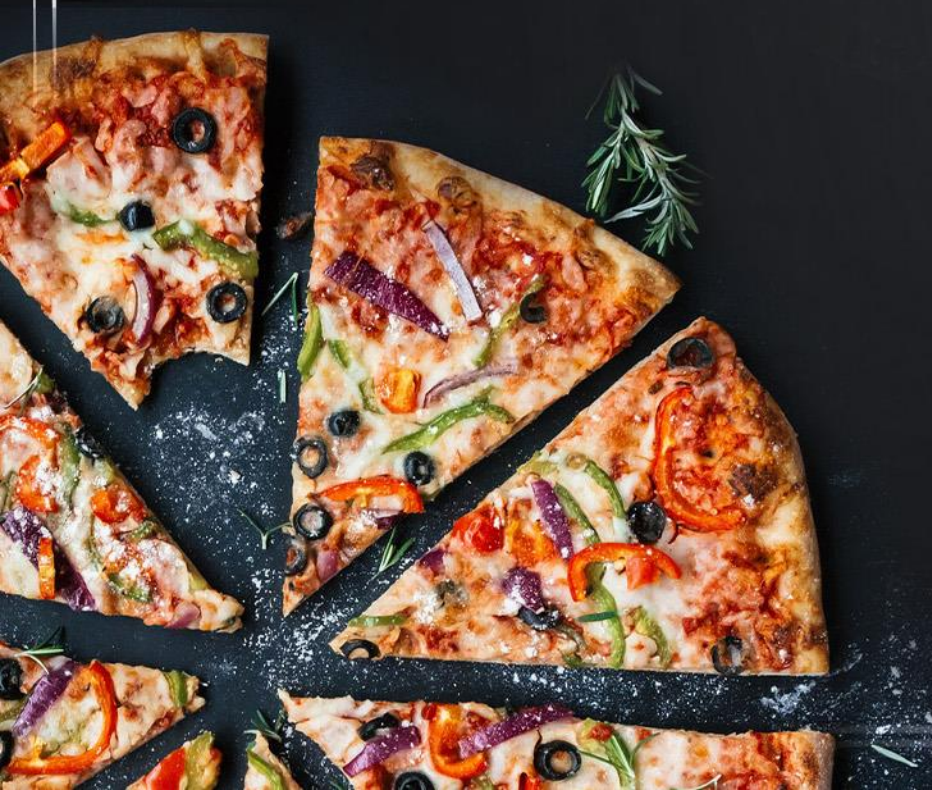
A PROJECT REPORT BY ARPIT SHUKLA



PROJECT OBJECTIVE-

To analyze the sales data of PizzaHub Brand to uncover insights into sales performance, customer preferences, and operational efficiency using MySQL joins and functions to solve some problem statements that they want from us.

A PROJECT REPORT BY ARPIT SHUKLA



TABLES NAME

These are the tables we have in our database.

01: orders

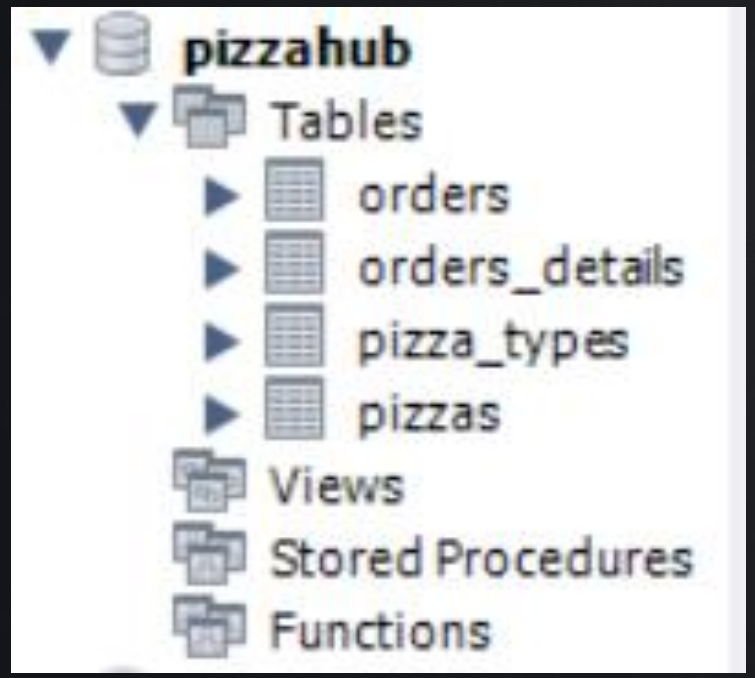
02: orders_details

03: pizza_types

04: pizzas

A PROJECT REPORT BY ARPIT SHUKLA

BASIC STRUCTURE AND SCHEMA:




```
1  create database pizzahub;
2  create table orders(
3  order_id int not null,
4  order_date date not null,
5  order_time time not null,
6  primary key(order_id));
7
8  create table orders_details(
9  order_details_id int not null,
10 order_id int not null,
11 pizza_id text not null,
12 quantity int not null,
13 primary key(order_details_id));
```

QUESTION 1: -- Retrieve the total number of orders placed.

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

Result Grid

	total_orders
▶	21350

QUESTION 2:

-- Calculate the total revenue generated from pizza sales.

```
2 •  SELECT
3        ROUND(SUM(pizzas.price * orders_details.quantity),
4                2) AS total_sales
5    FROM
6        pizzas
7            JOIN
8        orders_details ON pizzas.pizza_id = orders_details.pizza_id
```

Result Grid

	total_sales
▶	817860.05

QUESTION 3: -- Identify the highest-priced pizza.

```
2 •  SELECT
3        pizzas.price, pizza_types.name
4    FROM
5        pizzas
6            JOIN
7        pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
8    ORDER BY price DESC
9    LIMIT 1;
```

Result Grid | Filter Rows:

	price	name
▶	35.95	The Greek Pizza

QUESTION 4: -- Identify the most common pizza size ordered.

```
2 •  SELECT
3        pizzas.size,
4        COUNT(orders_details.order_details_id) AS order_count
5    FROM
6        pizzas
7            JOIN
8        orders_details ON pizzas.pizza_id = orders_details.pizza_id
9    GROUP BY pizzas.size
10   ORDER BY order_count DESC;
```

Result Grid | Filte

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

QUESTION 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;
```

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

QUESTION 6:

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

```sql
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

QUESTION 7:

-- Determine the distribution of orders by hour of the day.

```
2 •  select hour(order_time) as hour,count(order_id) as orders from orders
3    group by hour(order_time);
```

Result Grid

hour	orders
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663

QUESTION 8:

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

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

Result Grid | Filter R

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

QUESTION 9:

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

```
2  •  SELECT
3         ROUND(AVG(quantity), 0)
4     FROM
5         (SELECT
6             orders.order_date, SUM(orders_details.quantity) AS quantity
7         FROM
8             orders
9         JOIN orders_details ON orders.order_id = orders_details.order_id
10        GROUP BY orders.order_date) AS order_quantity;
```

Result Grid | Filter Ro

	ROUND(AVG(quantity), 0)
▶	138

QUESTION 10: -- Determine the top 3 most ordered pizza types based on revenue.

```
3 •  select pizza_types.name,
4    sum(orders_details.quantity * pizzas.price) as revenue
5    from pizza_types join pizzas
6    on pizzas.pizza_type_id = pizza_types.pizza_type_id
7    join orders_details
8    on orders_details.pizza_id = pizzas.pizza_id
9    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

QUESTION 11:

-- Calculate the percentage contribution of each pizza type to total revenue.

```
3 •  select pizza_types.category,
4    round(sum(orders_details.quantity * pizzas.price)/(select round(sum(orders_details.quantity * pizzas.price),2) as total_sales
5    from orders_details join
6    pizzas on pizzas.pizza_id = orders_details.pizza_id)*100,2) as revenue
7    from pizza_types join pizzas
8    on pizzas.pizza_type_id = pizza_types.pizza_type_id
9    join orders_details
10   on orders_details.pizza_id = pizzas.pizza_id
11   group by pizza_types.category order by revenue desc;
```

Result Grid

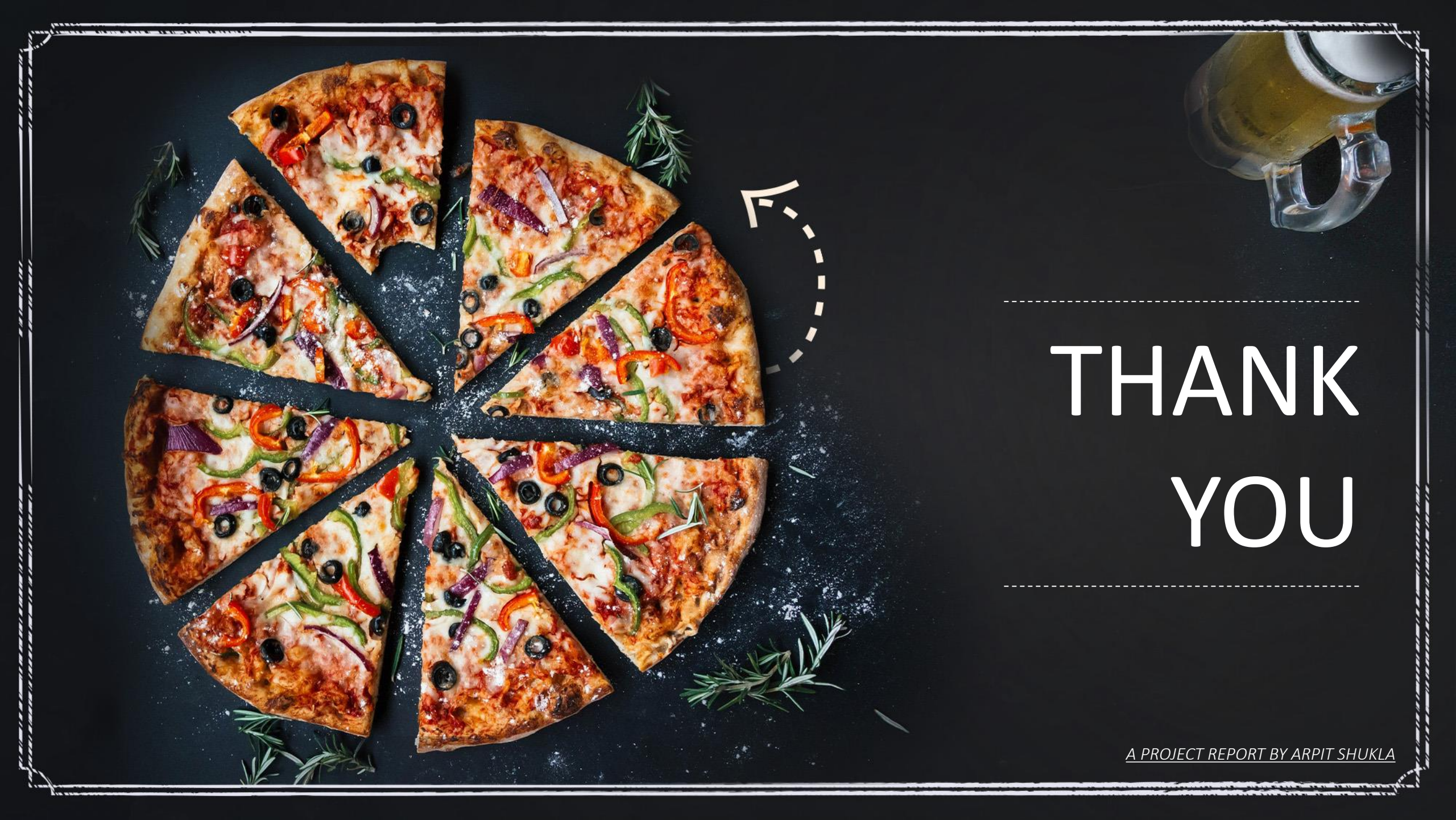
category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

QUESTION 12: -- Analyze the cumulative revenue generated over time.

```
3   select order_date,
4   sum(revenue) over(order by order_date) as cum_revenue
5   from
6   (select orders.order_date,
7   sum(orders_details.quantity*pizzas.price) as revenue
8   from orders_details join pizzas
9   on orders_details.pizza_id=pizzas.pizza_id
10  join orders on orders.order_id= orders_details.order_id
11  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
2015-01-11	25862.65
2015-01-12	27781.7



THANK YOU

A PROJECT REPORT BY ARPIT SHUKLA