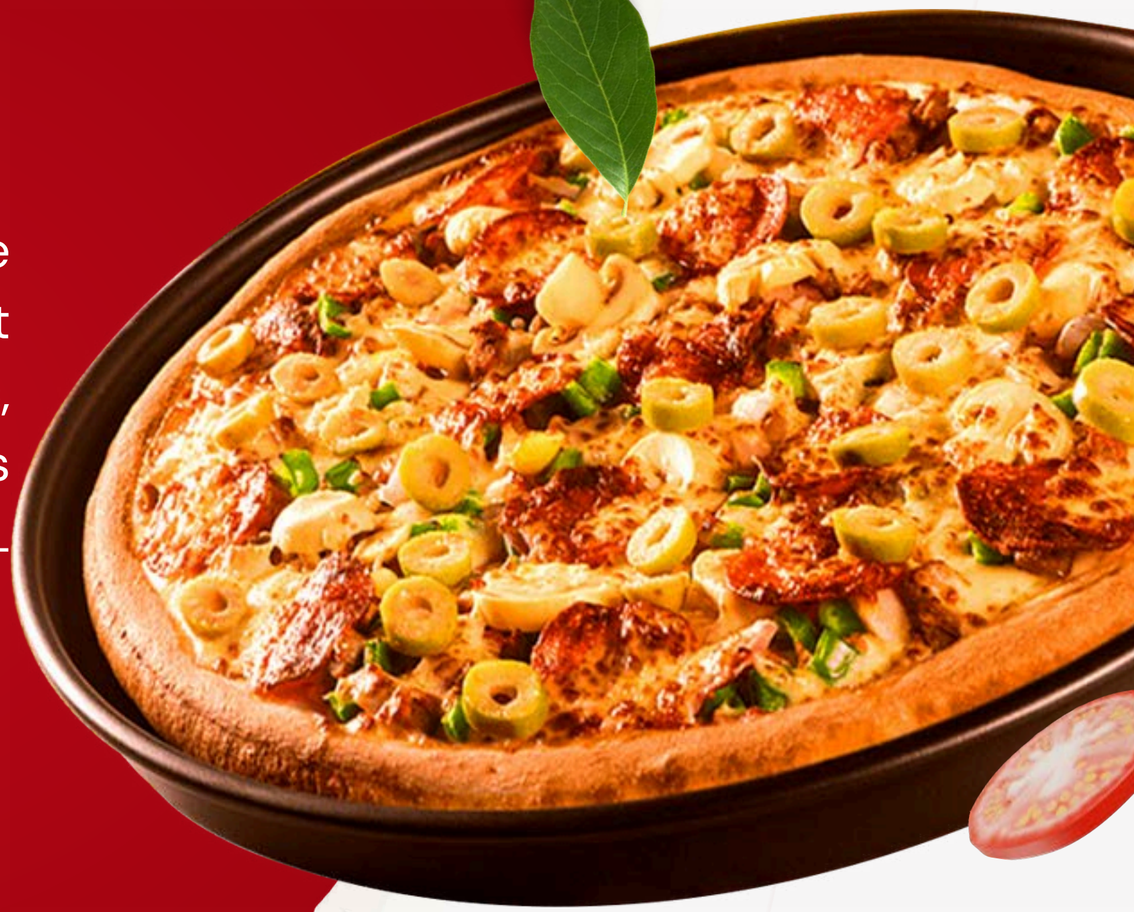
PIZZA SALES ANALYSIS

BY MANISH DIVEKAR

BY MANISH DIVEKAR

INTRODUCTION

In this project, I conducted a comprehensive analysis of pizza sales data to help the restaurant understand customer preferences, revenue patterns, and the performance of different pizzas. The analysis covered basic, intermediate, and advanced SQL queries



BY MANISH DIVEKAR

CHALLENGES

QUERIES INCLUDE

SELECT, GROUP BY, ORDER BY
LIMIT, DESC

QUERIES INCLUDE

JOINS, GROUP BY, ORDER BY,
LIMIT, DESC, sub query

QUERIES INCLUDE

sub query, CTE (COMMON TABLE
EXPRESSION

BASIC ANALYSIS

I started with basic queries to calculate the total number of orders, total revenue, highest-priced pizza, most common pizza size, and top 5 most ordered pizzas.

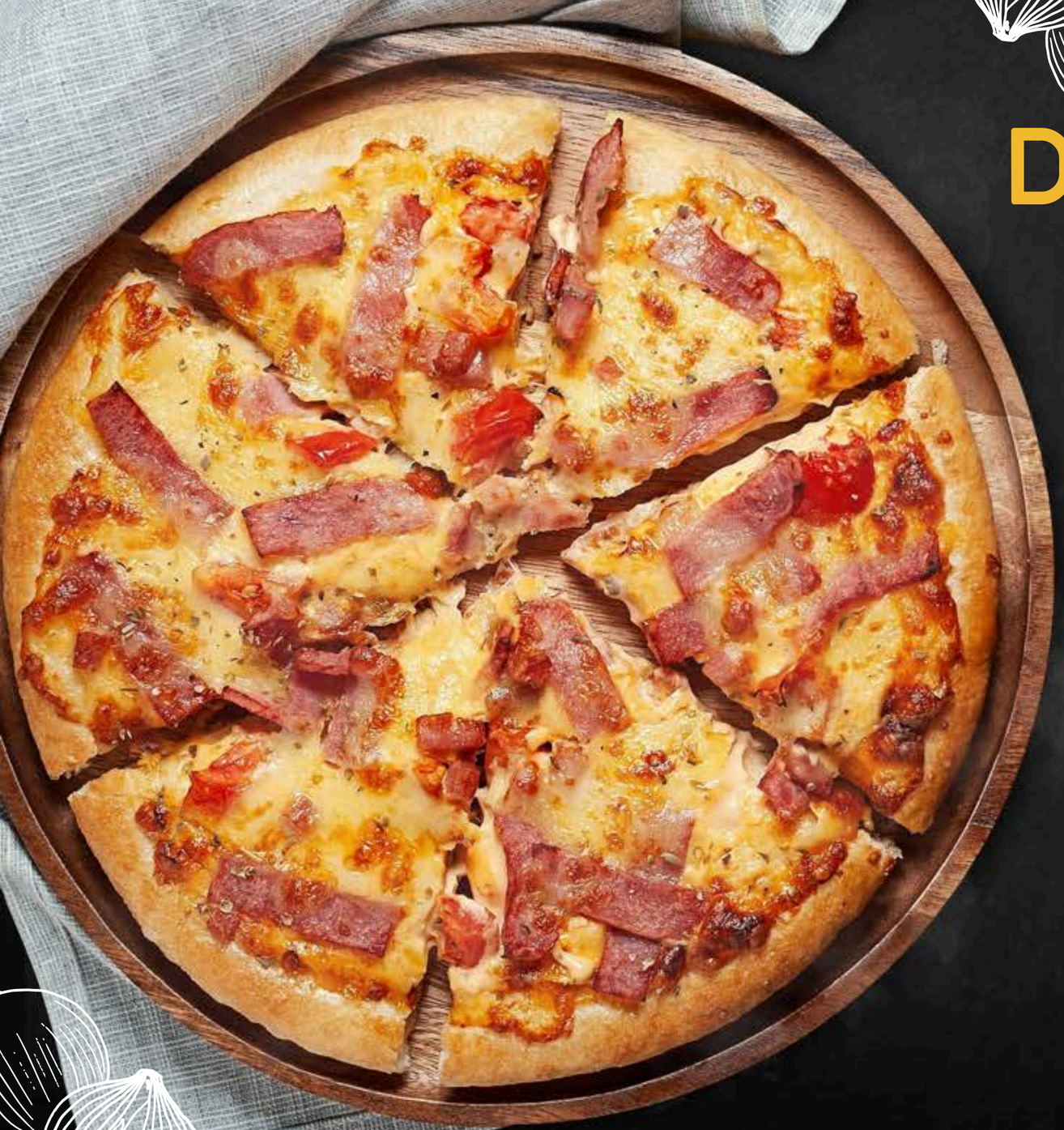
INTERMEDIATE ANALYSIS

I joined tables to determine the quantity of pizzas ordered by category, examined order distribution by hour, and calculated the average number of pizzas ordered per day

ADVANCED ANALYSIS

I used advanced SQL with subqueries and CTEs to:

- Identify the top 3 pizza types by revenue overall and within each category.
- Calculate each pizza's percentage contribution to total revenue.
- Analyze cumulative revenue over time.



BY MANISH DIVEKAR

DATABASE OVERVIEW

Orders
Order_id
Date
Time

Order_details
Order_detais_id
Order_id
Pizza_id
Quantity

Pizzas
Pizza_id
Pizza_type_id
size
Price

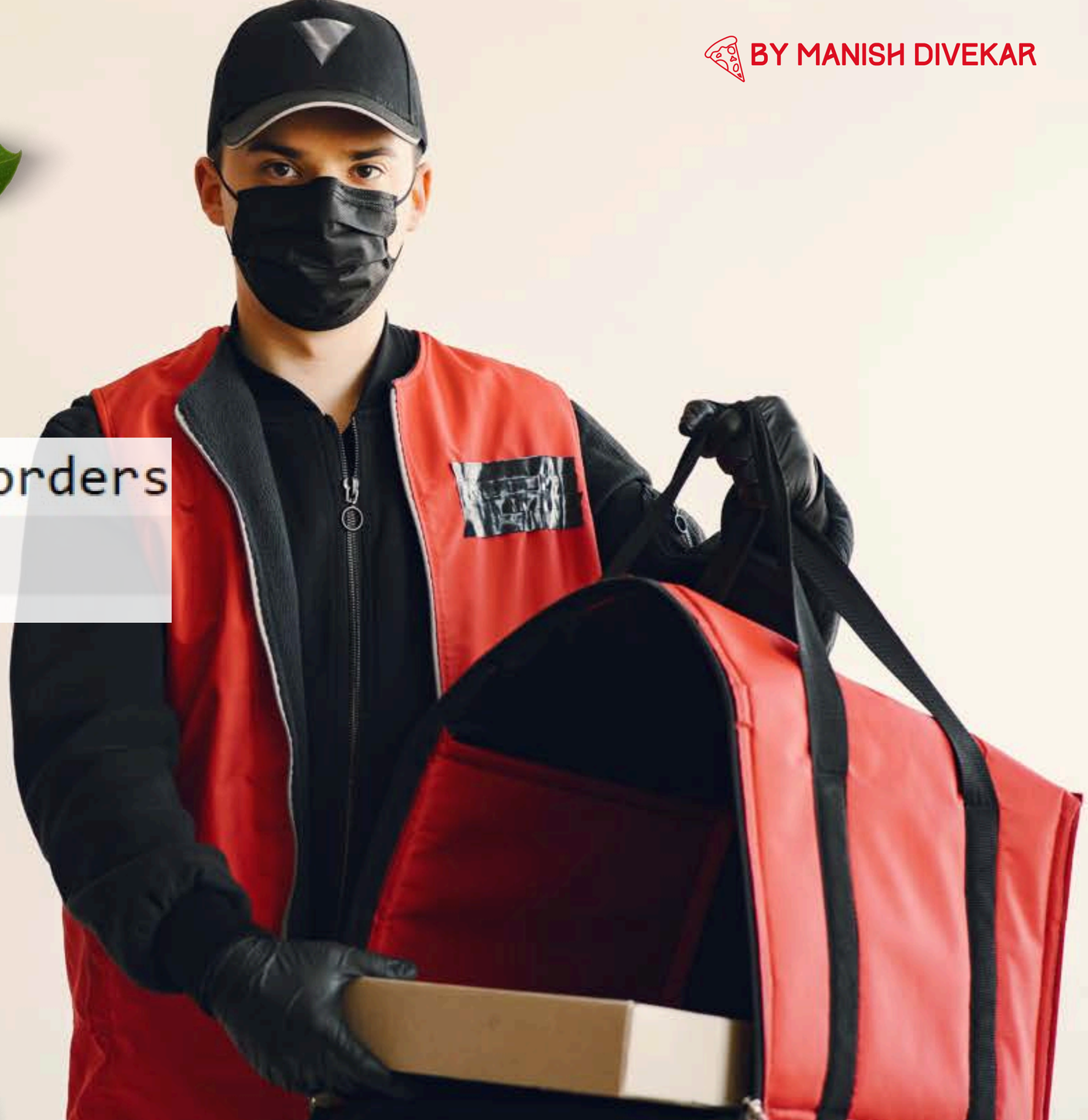
Pizza type
Pizza_type_id
Name
Category
Ingredient

BY MANISH DIVEKAR

1.RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

```sql
select count(*) as total_orders
from orders
```

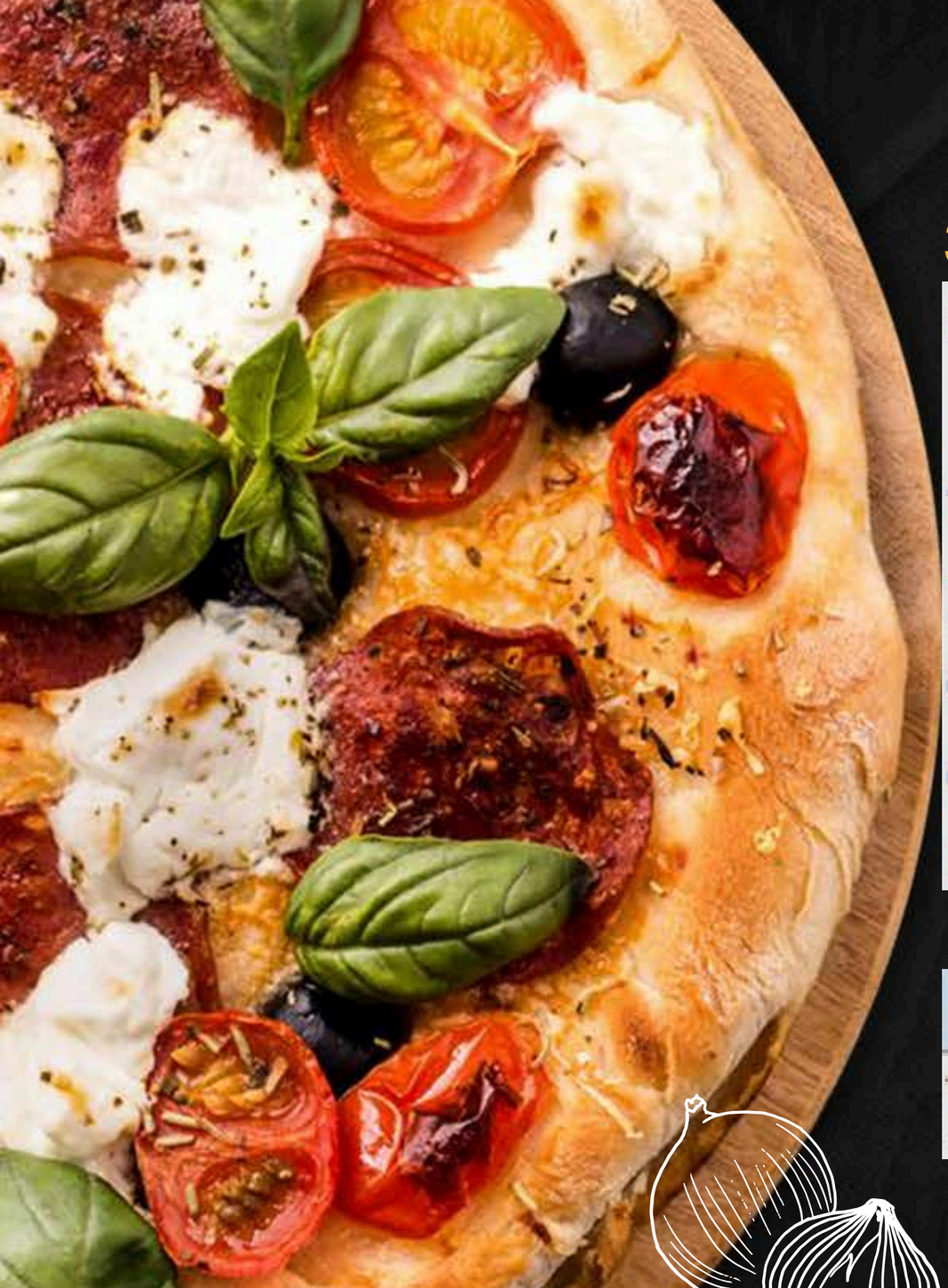
total_orders
21350



2.Calculate the total revenue generated from pizza sales.

```sql
select round(sum(p.price * od.quantity),2) as total_revenue
from pizzas p
join order_details od
on p.pizza_id = od.pizza_id
```

total_revenue
817860.05



BY MANISH DIVEKAR

3.Identify the highest-priced pizza.

```sql
select pt.name,
       p.price
from pizzas p
join pizza_types pt
on p.pizza_type_id = pt.pizza_type_id
order by price desc
limit 1
```

name	price
The Greek Pizza	35.95

BY MANISH DIVEKAR

4.Identify the most common pizza size ordered.

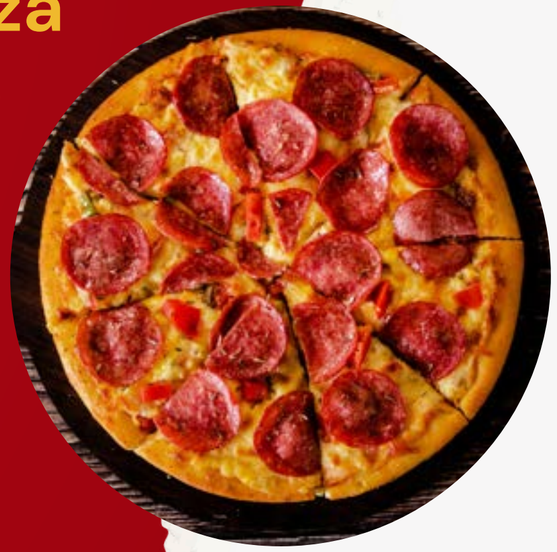
```sql
select size,
       count(od.order_details_id) as order_count
from order_details od
join pizzas p
on p.pizza_id = od.pizza_id
group by size
```

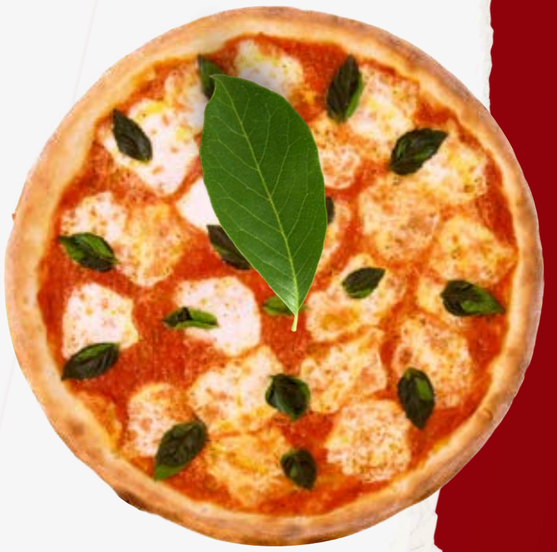
size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

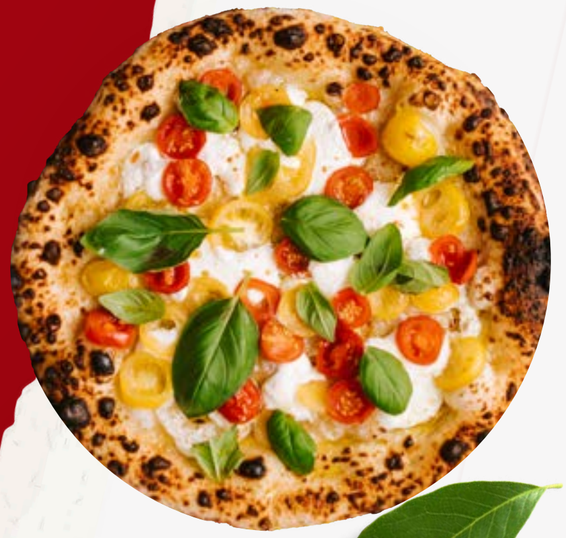
5.List the top 5 most ordered pizza types along with their quantities.

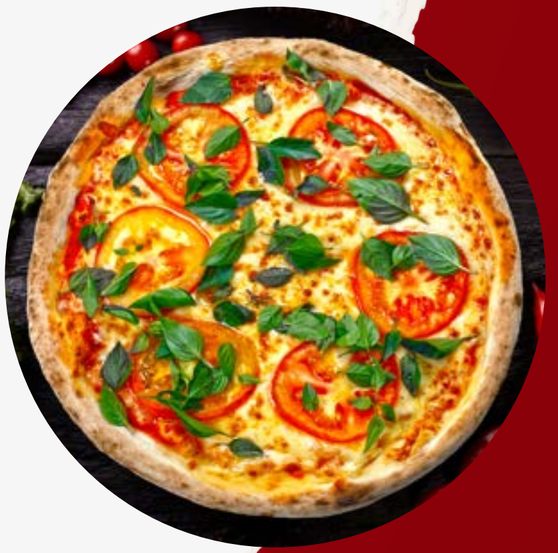
```sql
select name,
        sum(quantity) as total_quantity
from pizza_types pt
join pizzas p
on p.pizza_type_id = pt.pizza_type_id
join order_details od
on p.pizza_id = od.pizza_id
group by name
order by total_quantity desc
limit 5
```

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

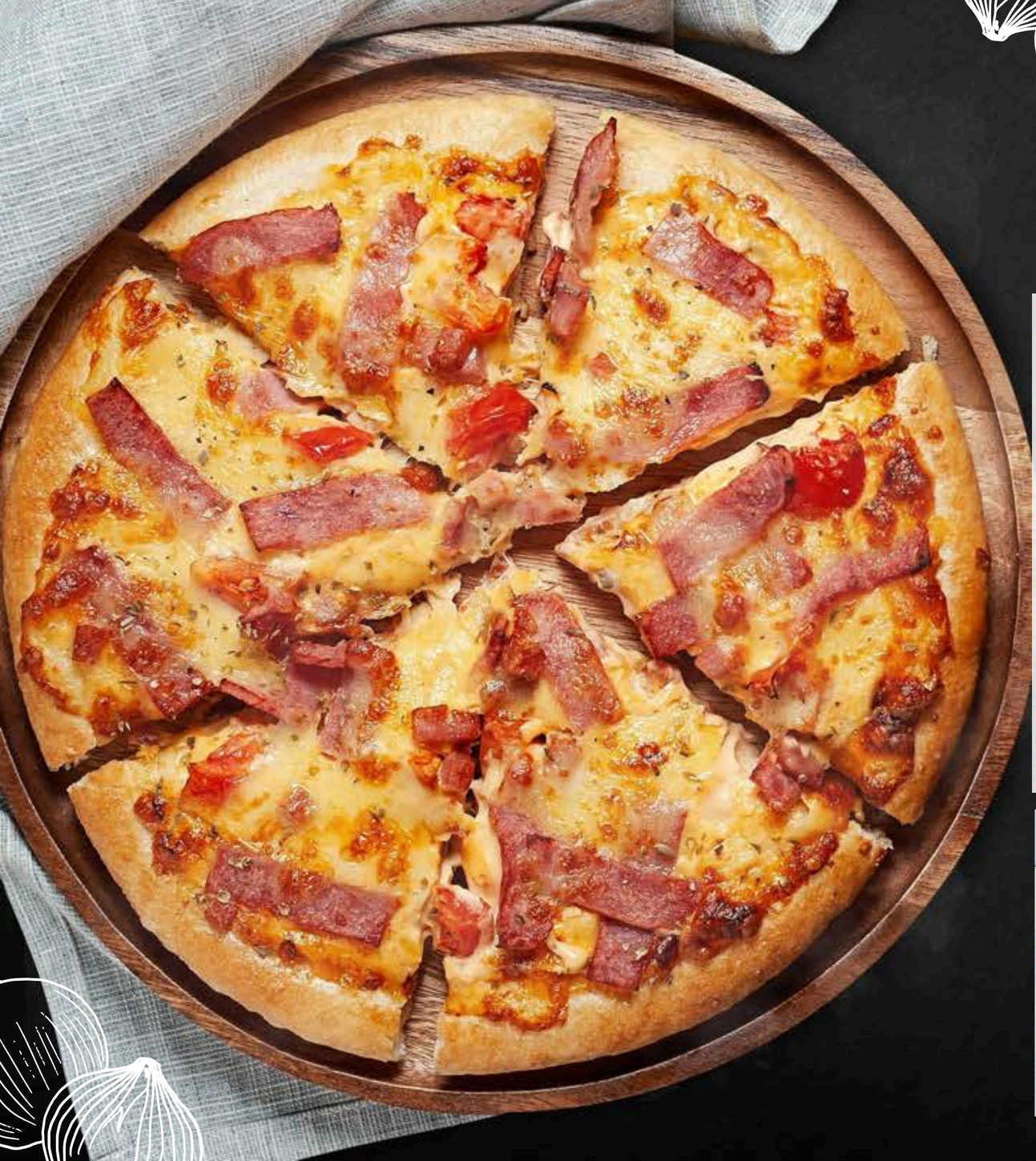








BY MANISH DIVEKAR



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

```sql
select category,
       sum(quantity) as quantity
from order_details od
join pizzas pz
on   pz.pizza_id = od.pizza_id
join pizza_types pt
on   pt.pizza_type_id = pz.pizza_type_id
group by category
```

category	quantity
Classic	14888
Veggie	11649
Supreme	11987
Chicken	11050

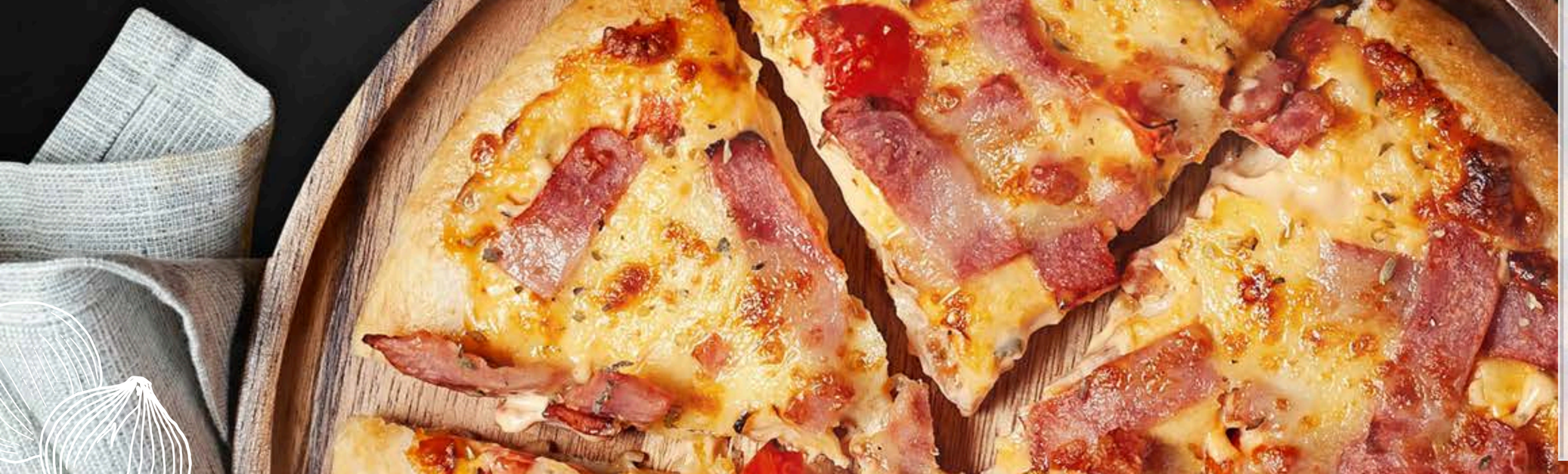
BY MANISH DIVEKAR

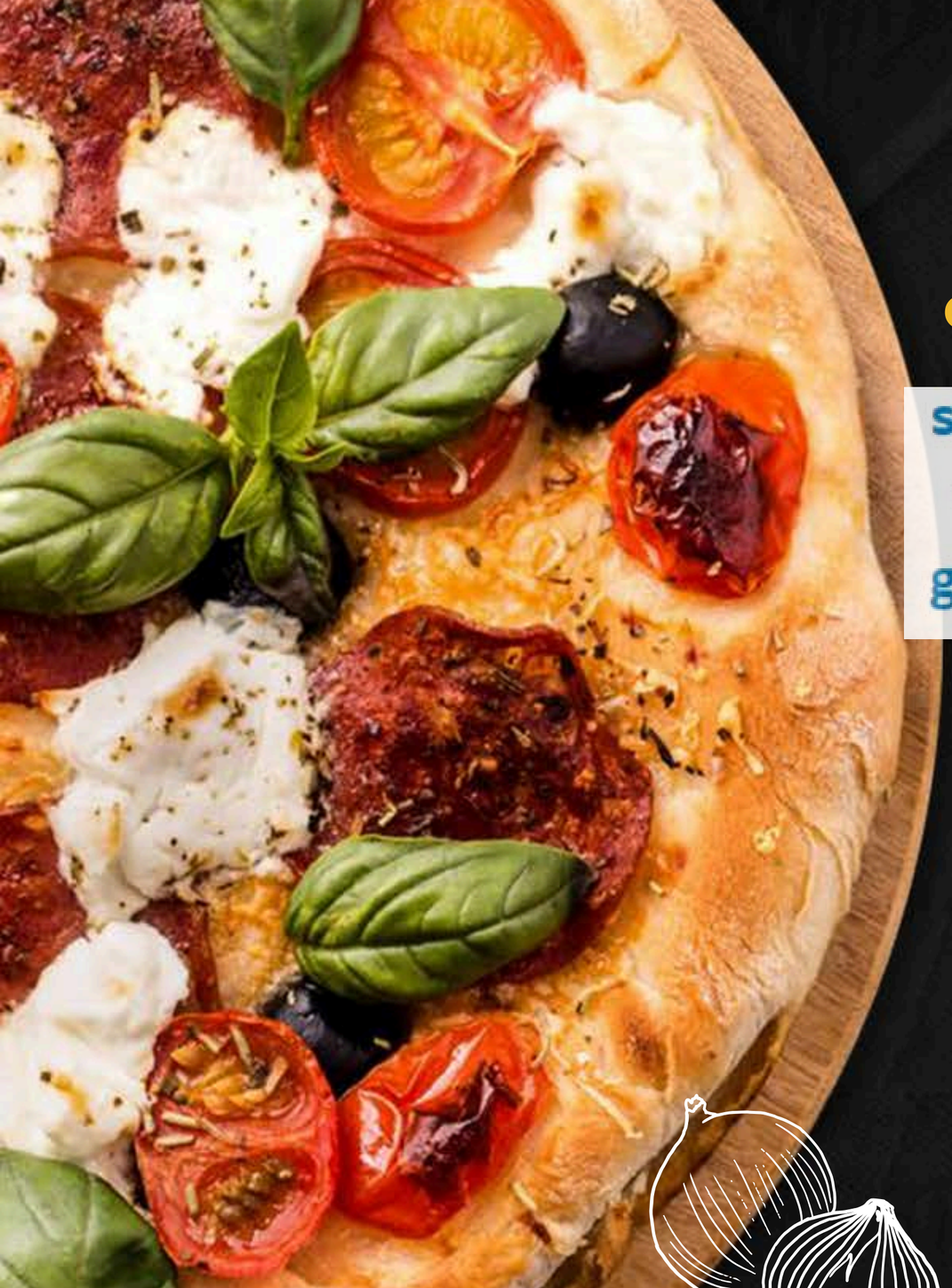
7.Determine the distribution of orders by hour of the day.

```sql
select hour(time) as hours,
       count(order_id) as  order_count
from orders
group by hours
```

★★★★★

hours	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.

```
select category, count(name) as piza_count
 from pizza_types
group by category
```

category	piza_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.

```sql
select round(avg(quantity),0) as avg_pizzas_ord_per_day from
(select date, sum(quantity)  as quantity
from orders
join order_details
on order_details.order_id = orders.order_id
group by date) as order_quantity
```

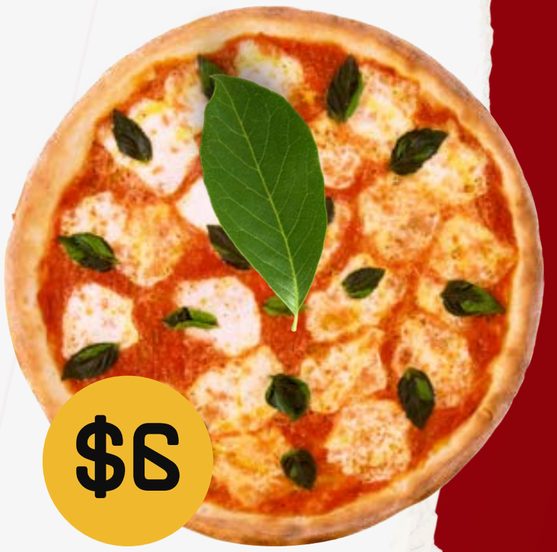
avg_pizzas_ord_per_day
138

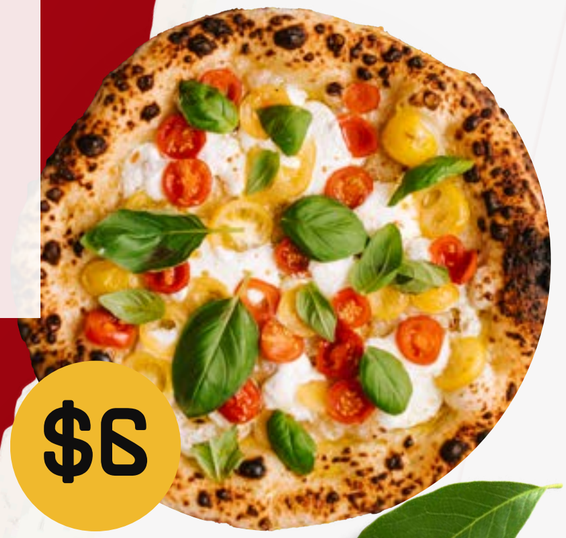
BY MANISH DIVEKAR

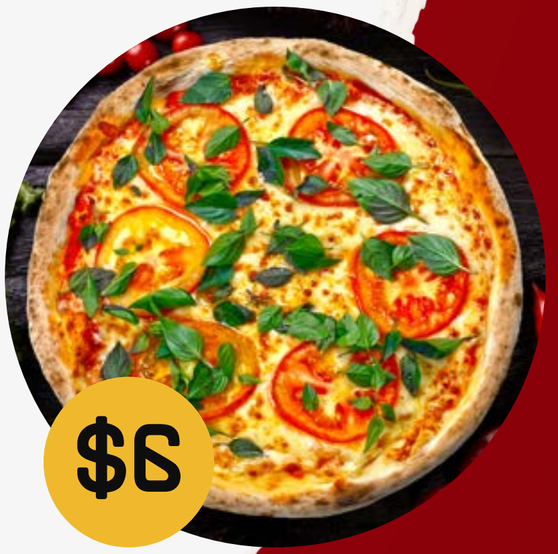
10. Determine the top 3 most ordered pizza types based on revenue.

```sql
select pt.name,
      round(sum(p.price * od.quantity),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 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







ADVANCE

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

```sql
select pt.category,
       round((sum(od.quantity * p.price) / (select sum(od.quantity * p.price) from pizzas p
                                            join order_details od
                                            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
```

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

12.Analyze the cumulative revenue generated over time.

```sql
select date,
sum(revenue) over(order by date) as cum_revenue
from
(SELECT o.date,
        SUM(od.quantity * p.price) AS revenue
FROM orders o
JOIN order_details od ON o.order_id = od.order_id
JOIN pizzas p ON p.pizza_id = od.pizza_id
GROUP BY o.date) as sale
```

date	cum_revenue
2015-01-01	2713.850000000004
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
2015-01-13	29831.300000000003
2015-01-14	32358.700000000004
2015-01-15	34343.50000000001
2015-01-16	36937.65000000001
2015-01-17	39001.75000000001
2015-01-18	40978.60000000006

13.Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```
with sales as
(select category,
        name,
        sum(p.price * od.quantity) as revenue,
        ROW_NUMBER() OVER (PARTITION BY pt.category
                           ORDER BY SUM(od.quantity * p.price) DESC
) AS rnk
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 category, name)

select *
from sales
where rnk <=3
order by category, revenue  desc
```

Result

category	name	revenue	rnk
Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768	2
Chicken	The California Chicken Pizza	41409.5	3
Classic	The Classic Deluxe Pizza	38180.5	1
Classic	The Hawaiian Pizza	32273.25	2
Classic	The Pepperoni Pizza	30161.75	3
Supreme	The Spicy Italian Pizza	34831.25	1
Supreme	The Italian Supreme Pizza	33476.75	2
Supreme	The Sicilian Pizza	30940.5	3
Veggie	The Four Cheese Pizza	32265.70000000065	1
Veggie	The Mexicana Pizza	26780.75	2
Veggie	The Five Cheese Pizza	26066.5	3

Key Insights

These analyses revealed which pizza types were most popular, which sizes customers preferred, and the peak hours for orders providing actionable insights to optimize menu offerings and marketing strategies.

Conclusion

Overall, this project demonstrates my ability to write efficient SQL queries, join multiple tables, perform time based and category based analyses, and translate findings into business insights.

THANK YOU!