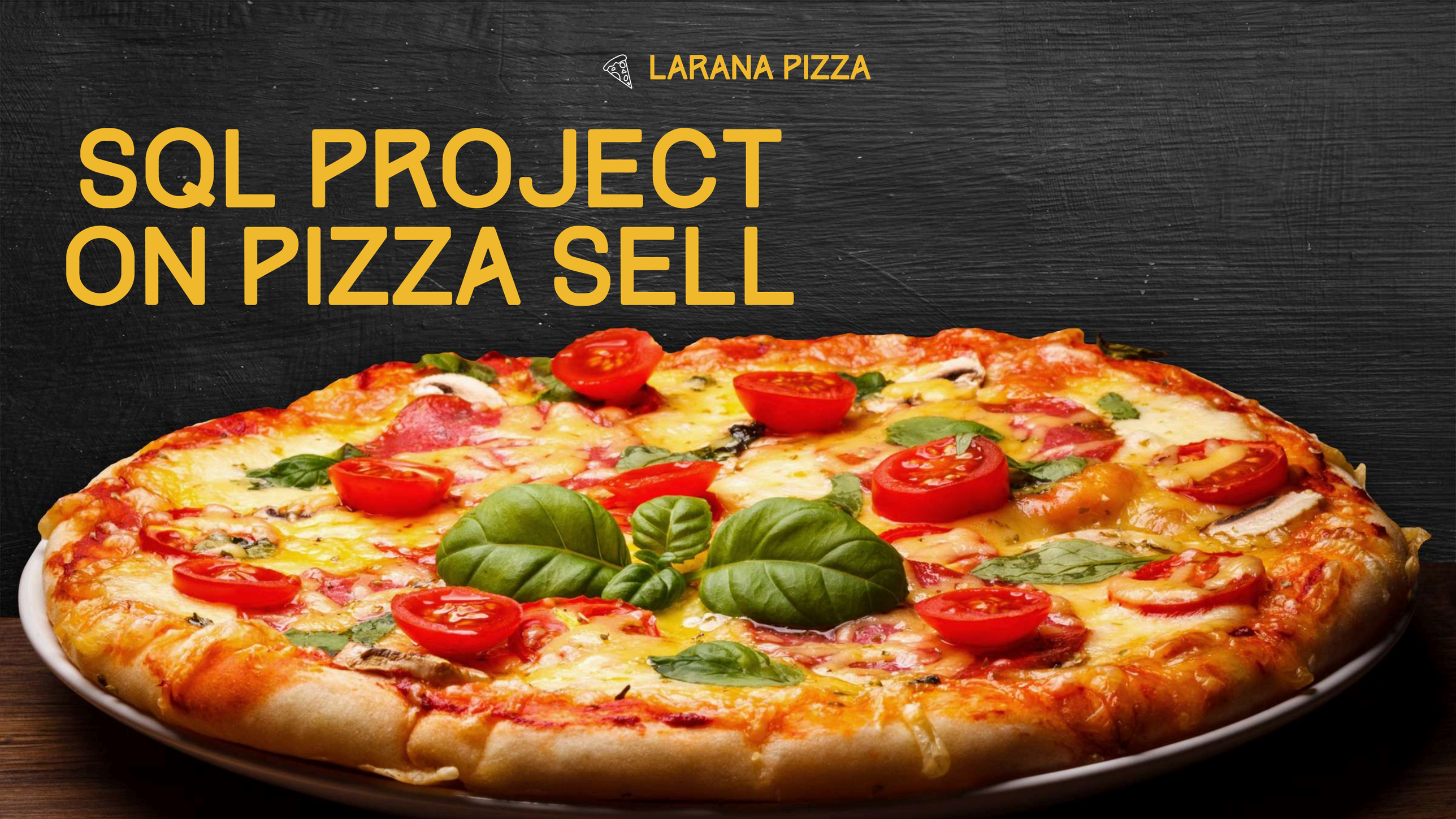
LARANA PIZZA

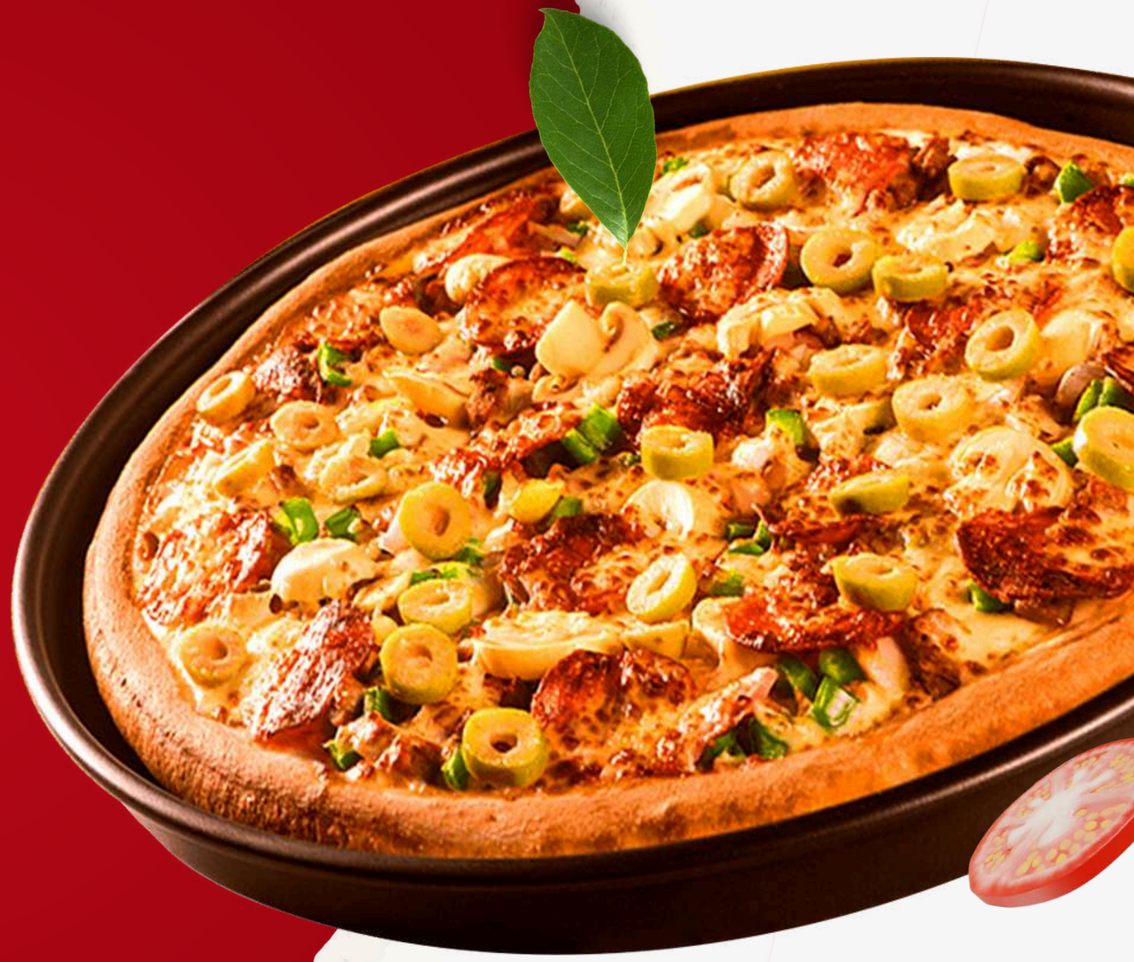
SQL PROJECT ON PIZZA SELL

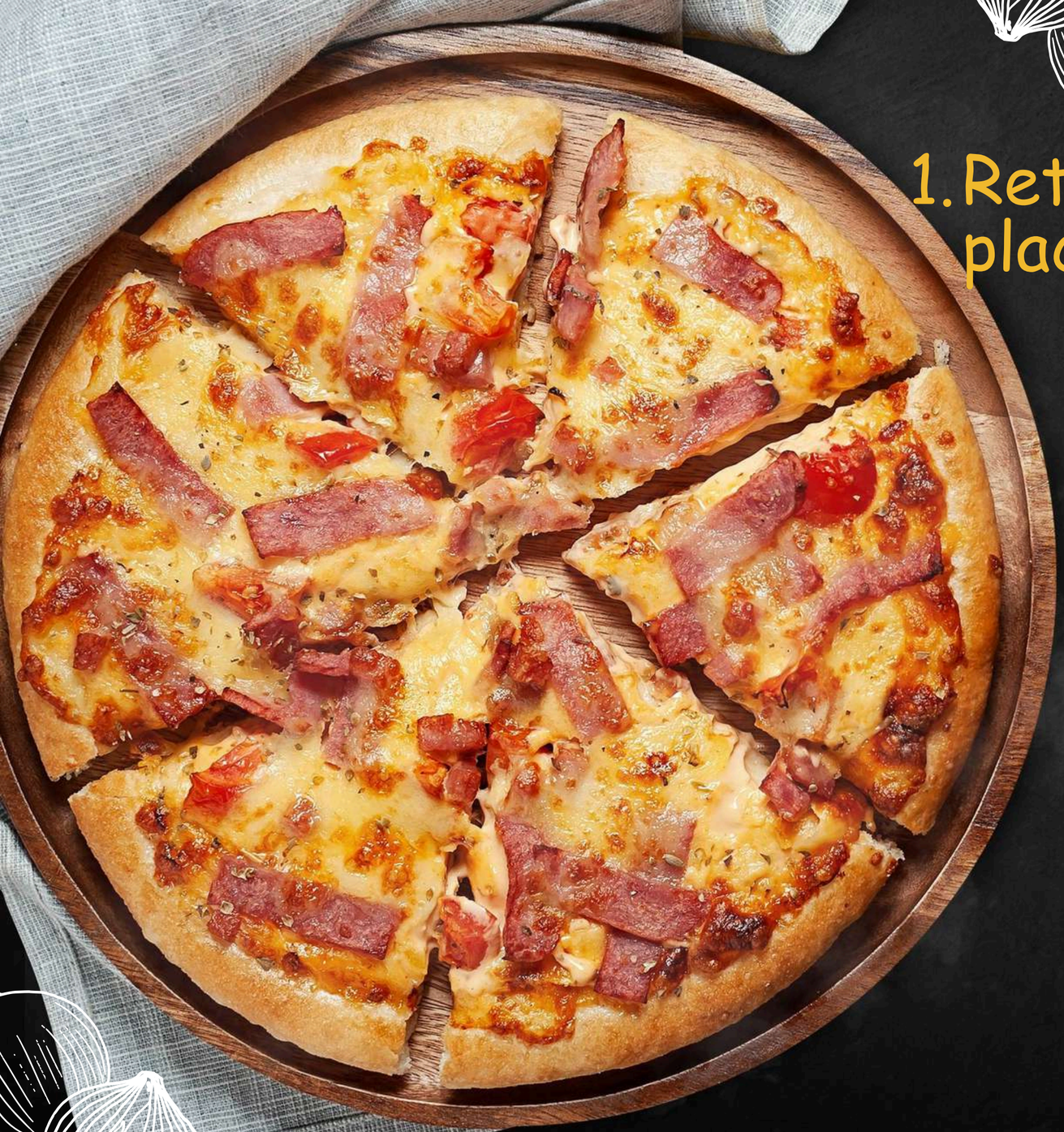


LARANA PIZZA

WELCOME TO
MY SQL PROJECT

I, Aryan Giri from NIT Agartala, [redacted] pleased to upload my SQL project, which focuses on solving 13 analytical problems related to Pizza Hut. This project demonstrates various SQL queries and techniques to analyze sales, customer preferences, order trends, and other key insights. Each problem is accompanied by a structured solution, providing a comprehensive understanding of data manipulation and retrieval. Through this project, I aim to showcase my proficiency in SQL and data analytics by tackling real-world business challenges in the food industry





LARANA PIZZA

1. Retrieve the total number of orders placed.

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

2.CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```
SELECT
ROUND(SUM(pizzas.price * order_details.quantity),
3) AS totel_revenue
FROM
order_details
JOIN
pizzas ON pizzas.pizza_id = order_details.pizza_id;
```

3. Identify the highest-priced pizza.

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

LARANA PIZZA

4. Identify the most common pizza size ordered.

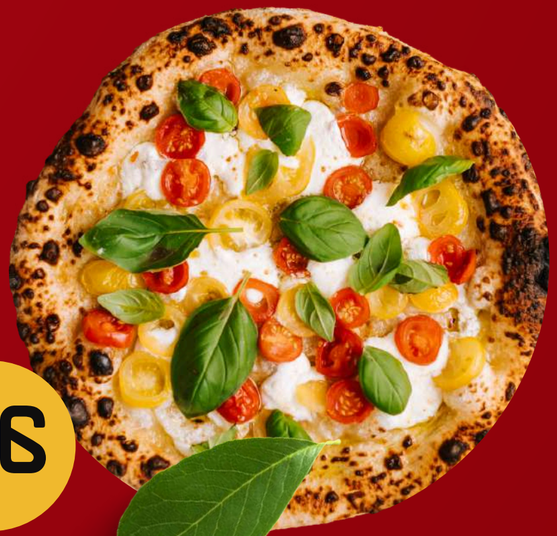
```
SELECT
pizzas.size, COUNT(order_details.order_details_id) as
order_count
FROM
pizzas
JOIN
order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC
LIMIT 1;
```

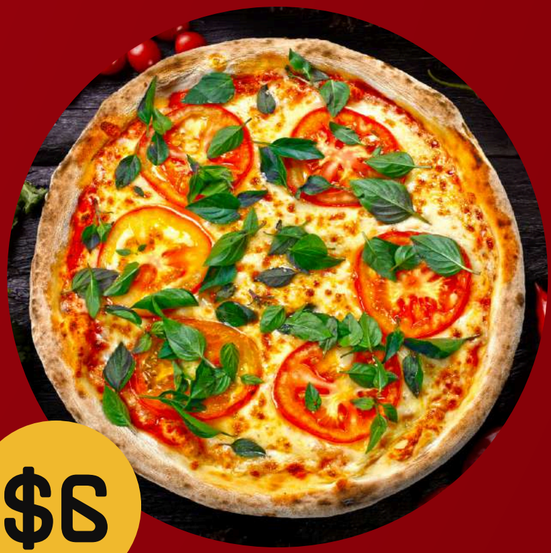
5. List the top 5 most ordered pizza types -- along with their quantities.

LARANA PIZZA

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

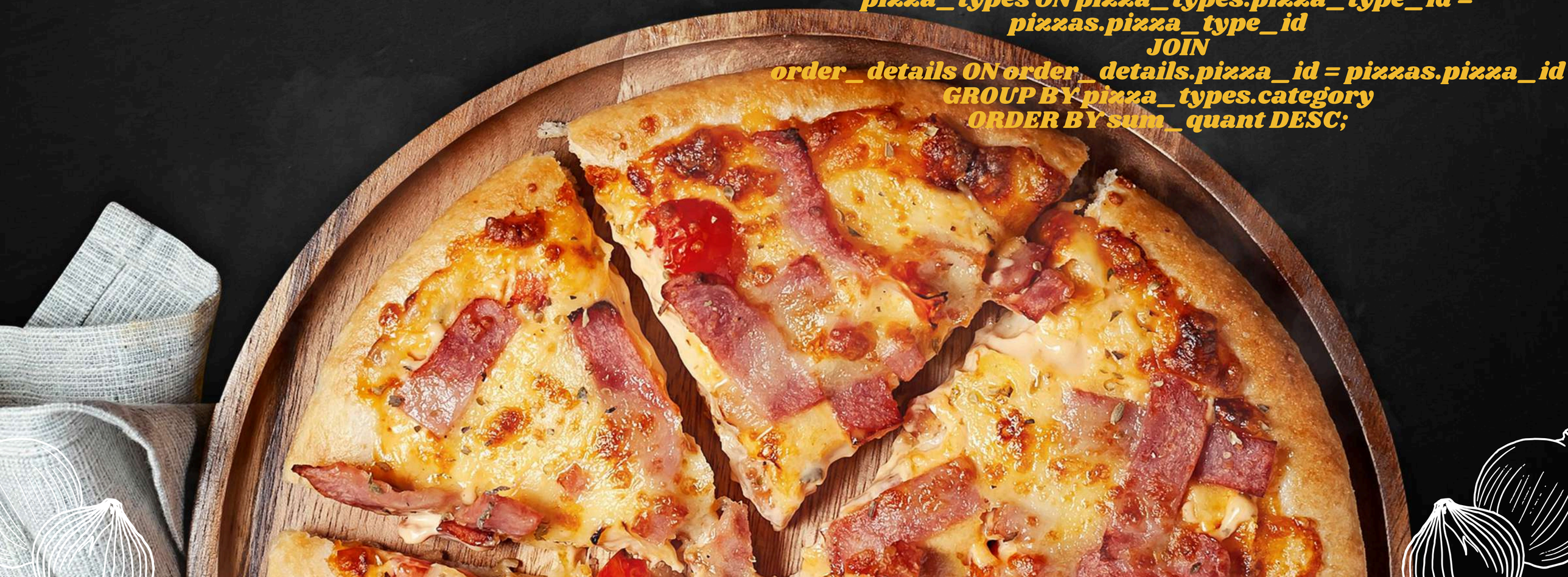
```
SELECT
    pizza_types.name,
    SUM(order_details.quantity) AS sum_quantity
FROM
    pizzas
        JOIN
    pizza_types ON pizza_types.pizza_type_id =
    pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id =
    pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY sum_quantity DESC
LIMIT 5;
```





6. JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

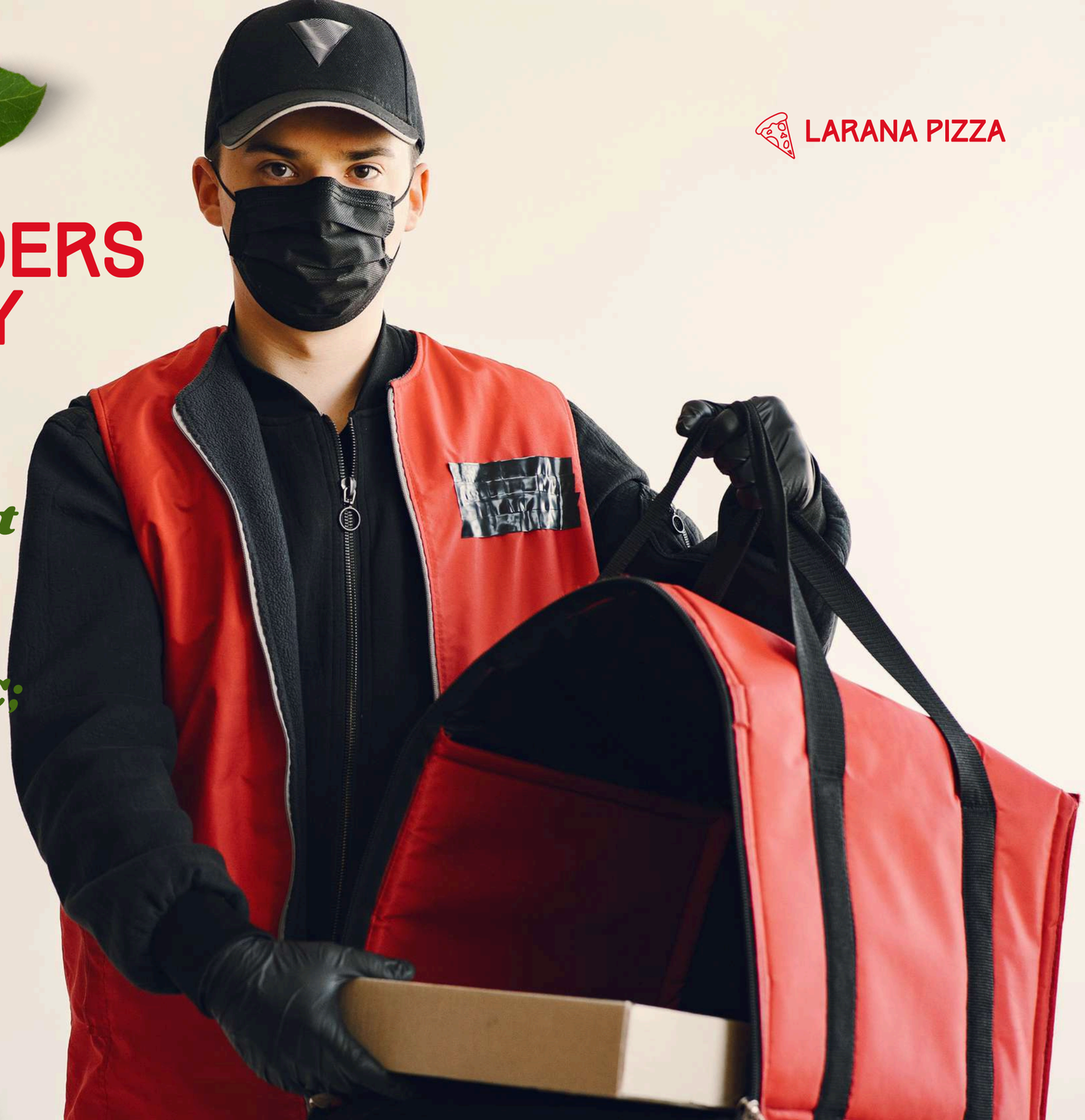
```
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS sum_quant
FROM
    pizzas
        JOIN
    pizza_types ON pizza_types.pizza_type_id =
    pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY sum_quant DESC;
```

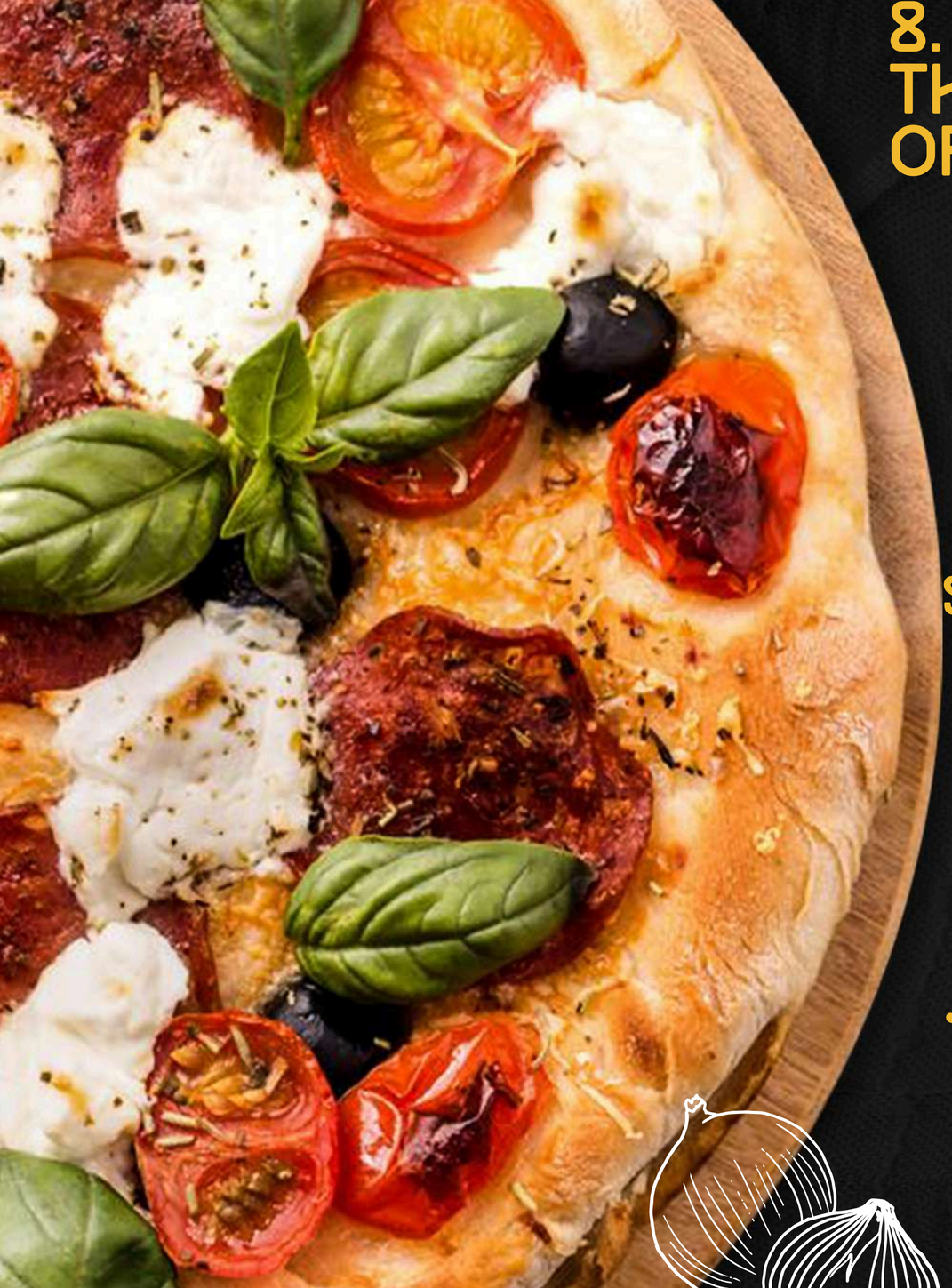


LARANA PIZZA

7. DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY

```
SELECT
HOUR(order_time) AS hour,
COUNT(order_id) AS order_count
FROM
orders
GROUP BY HOUR(order_time)
ORDER BY COUNT(order_id) DESC;
```





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

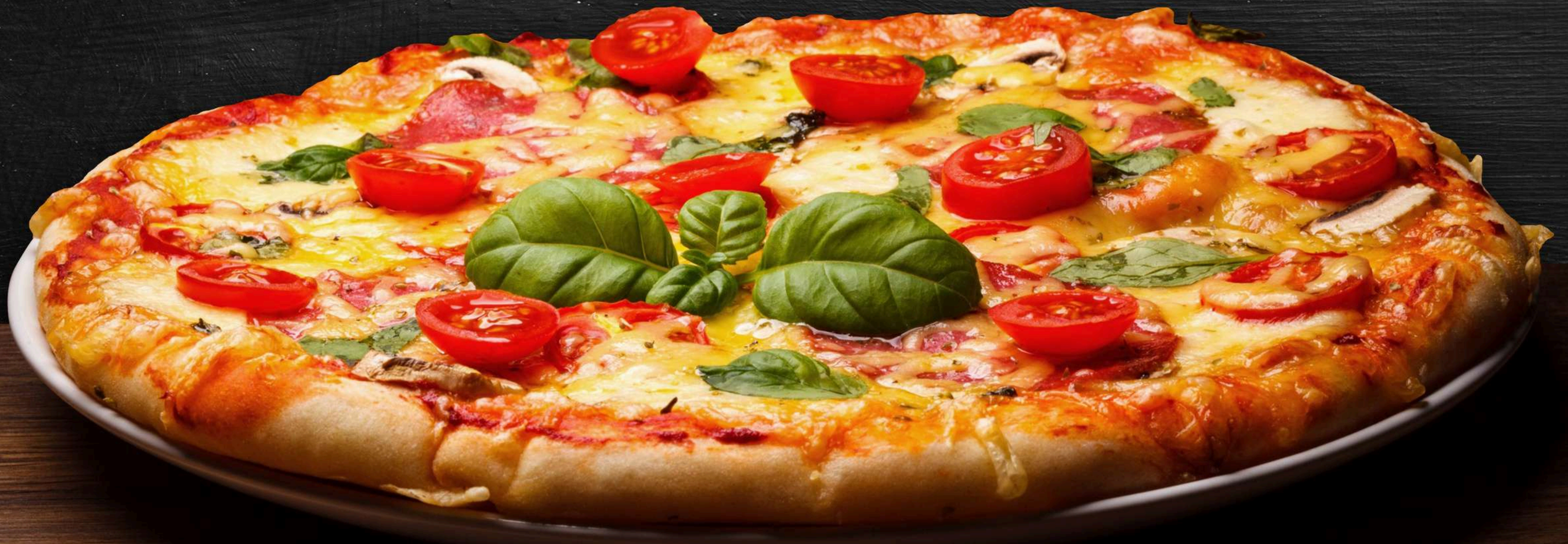
```
SELECT
category, COUNT(name)
FROM
pizza_types
GROUP BY category
ORDER BY COUNT(name) DESC;
```

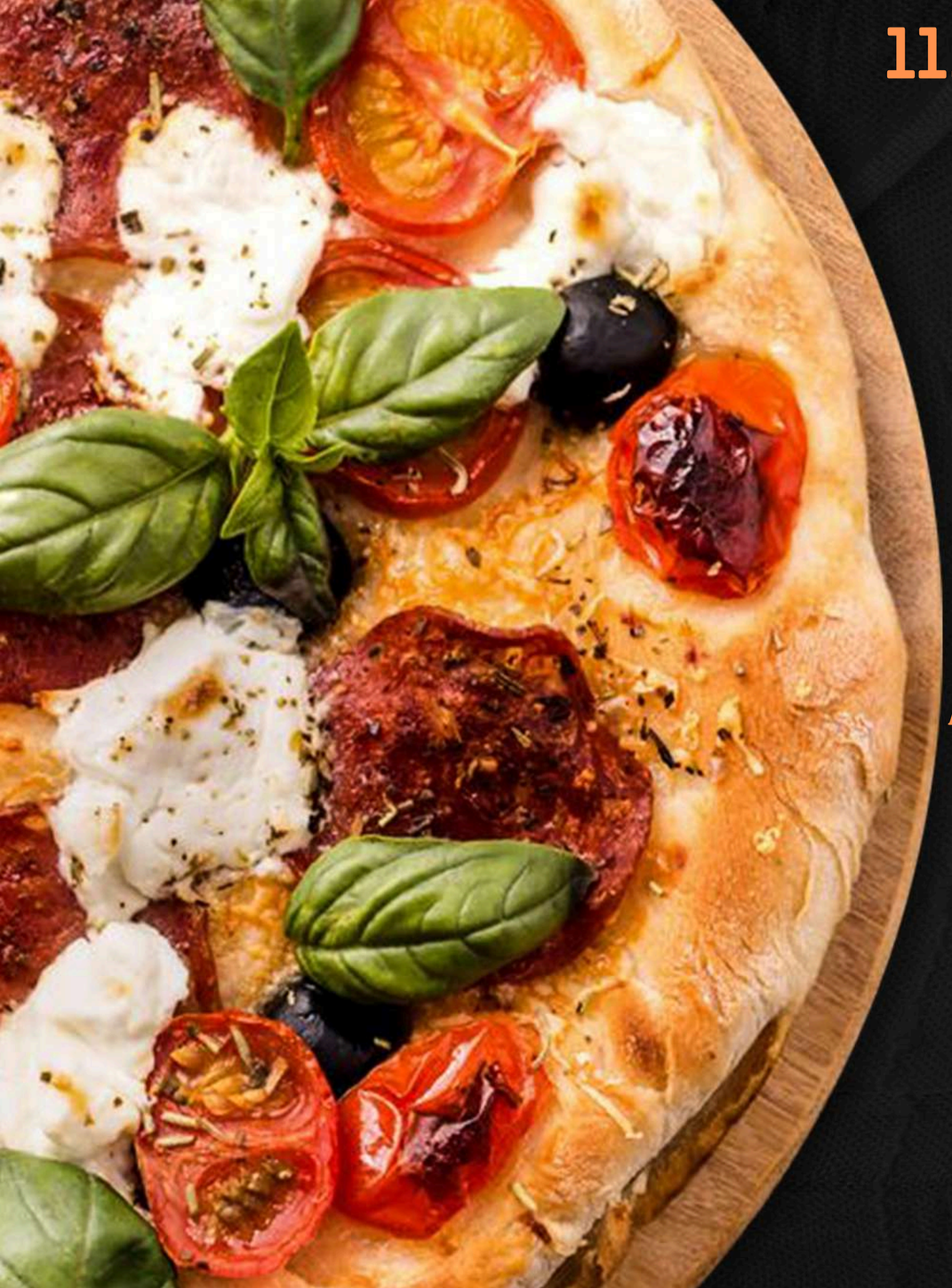
9. GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY

```
SELECT
round( AVG(T_orders),0) as avg_pizza_ordered_per_day
FROM
(SELECT
orders.order_date AS date,
SUM(order_details.quantity) AS T_orders
FROM
orders
JOIN order_details ON orders.order_id = order_details.order_id
GROUP BY date
ORDER BY T_orders DESC) AS order_quantity;
```

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

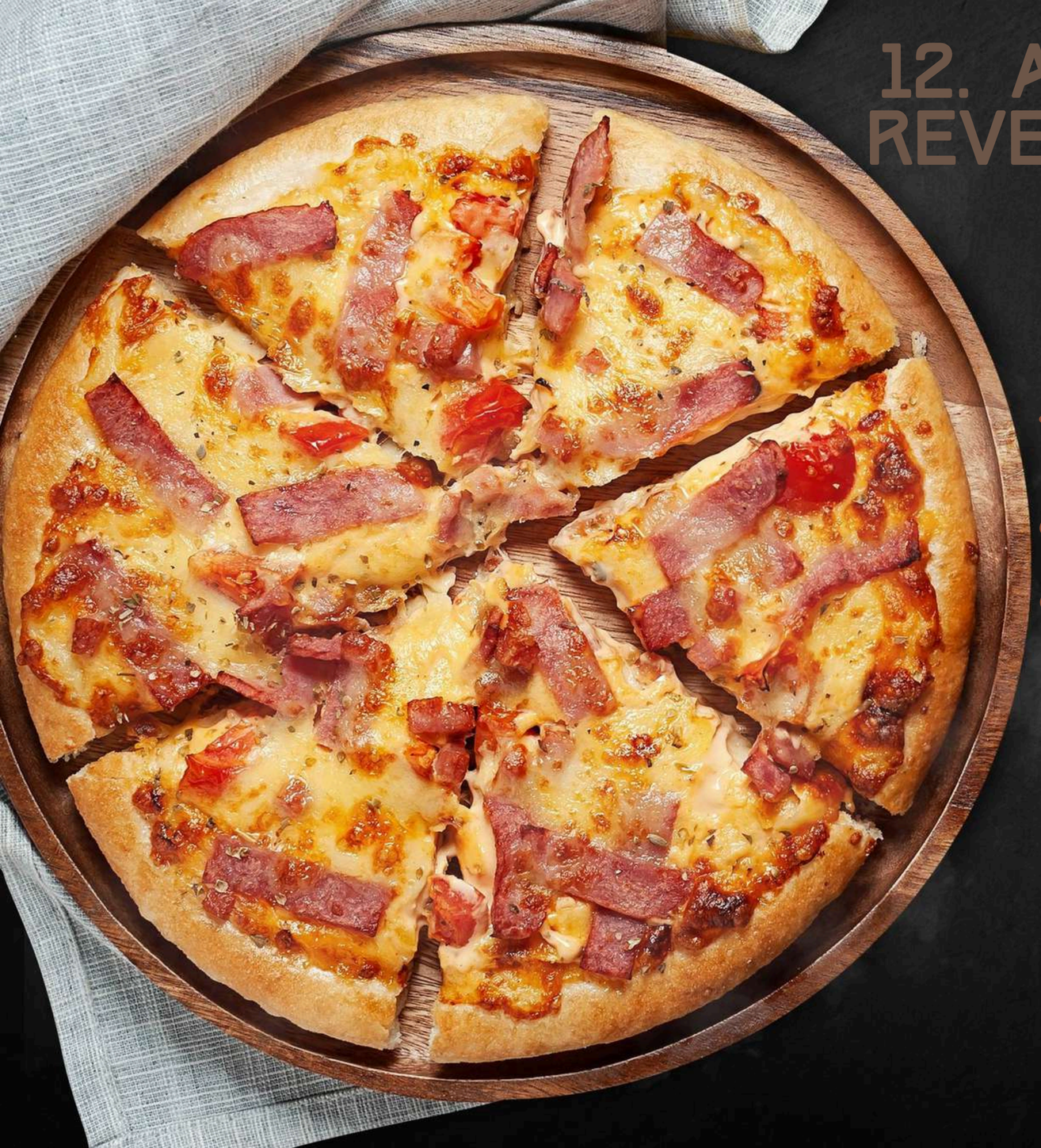
```
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue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





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

```
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 order_details.pizza_id = pizzas.pizza_id) * 100,
    2) AS revenue_percent
FROM
    pizzas
        JOIN
    pizza_types 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_percent DESC;
```



12. ANALYZE THE CUMALATICVE REVENUE GENERATED OVER TIME

```
select order_date,
round(sum(revenue) over(order by
order_date),2)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 ;
```

LARANA PIZZA

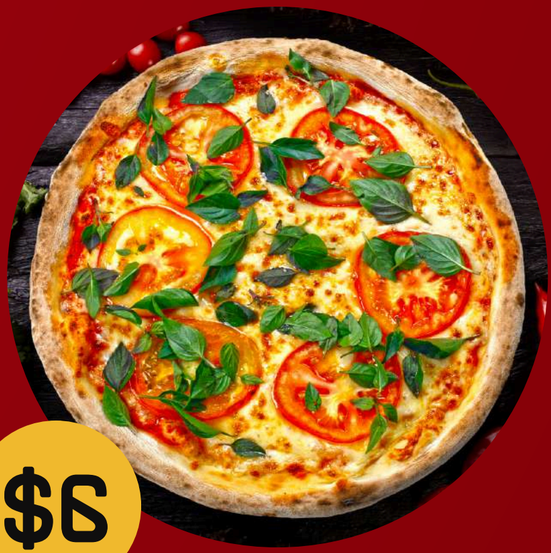
13. DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPESBASED ON REVENUE FOR EACH PIZZA CATEGORY

```
select category,name,round(revenue,2) as
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 order_details join pizzas
on order_details.pizza_id =
pizzas.pizza_id
join pizza_types
on pizza_types.pizza_type_id =
pizzas.pizza_type_id
group by
pizza_types.category,pizza_types.name)
as a) as b where rn <=3;
```

$6

$6

$6



$6

This concludes my SQL project, where I have addressed 13 analytical problems related to Pizza Hut with structured queries and insights. Through this work, I aimed to showcase effective data analysis techniques that can be applied to real-world business scenarios. I hope this project serves as a valuable reference for understanding SQL-based problem-solving in analytics