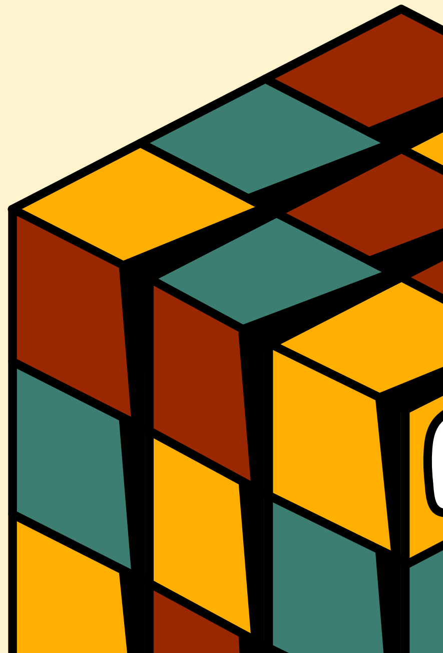
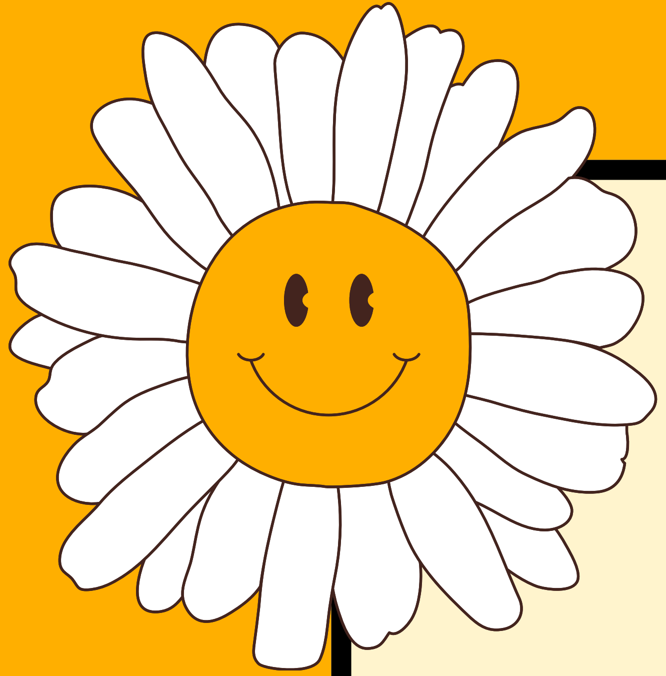
Prepared by Lokesh

SQL PORTFOLIO PROJECT

ENHANCING PRODUCT SALES STRATEGY

25 JULY 2024

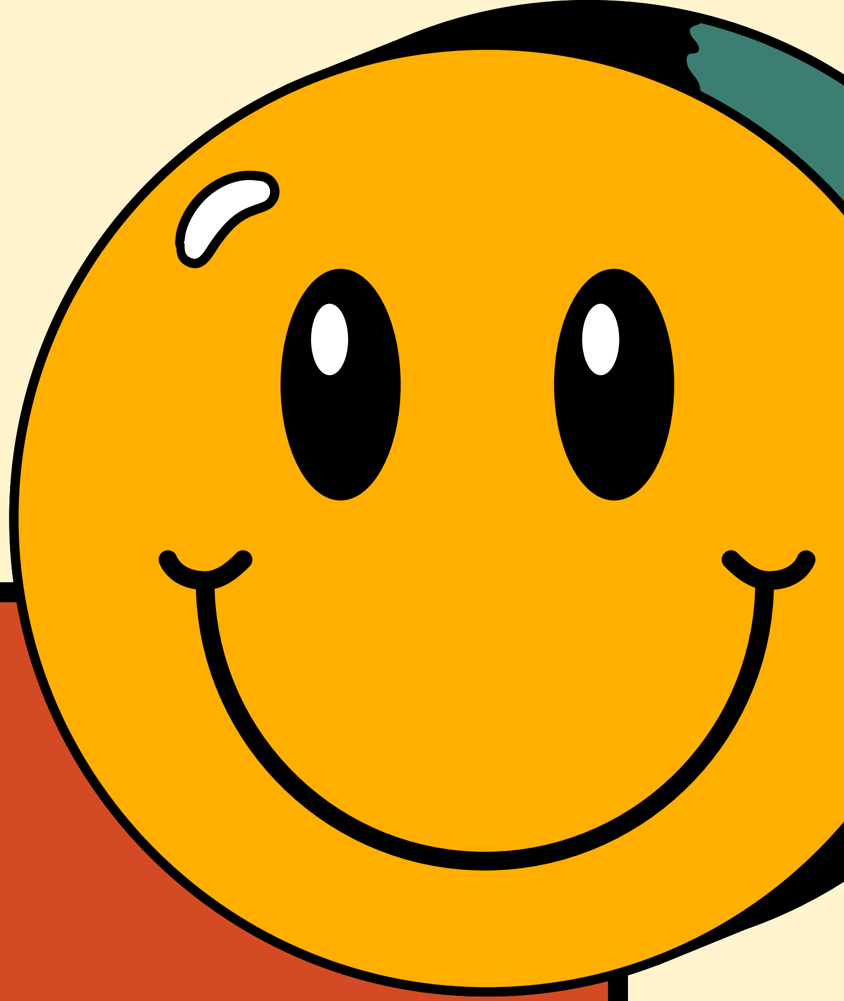




Introduction

In this project, I analyzed a dataset related to pizza sales to uncover insights and trends that can help in decision-making for a pizzeria. By leveraging SQL, I queried the data to understand various aspects such as sales performance, popular pizzas,max profit.

1)retrive the total number of orders placed

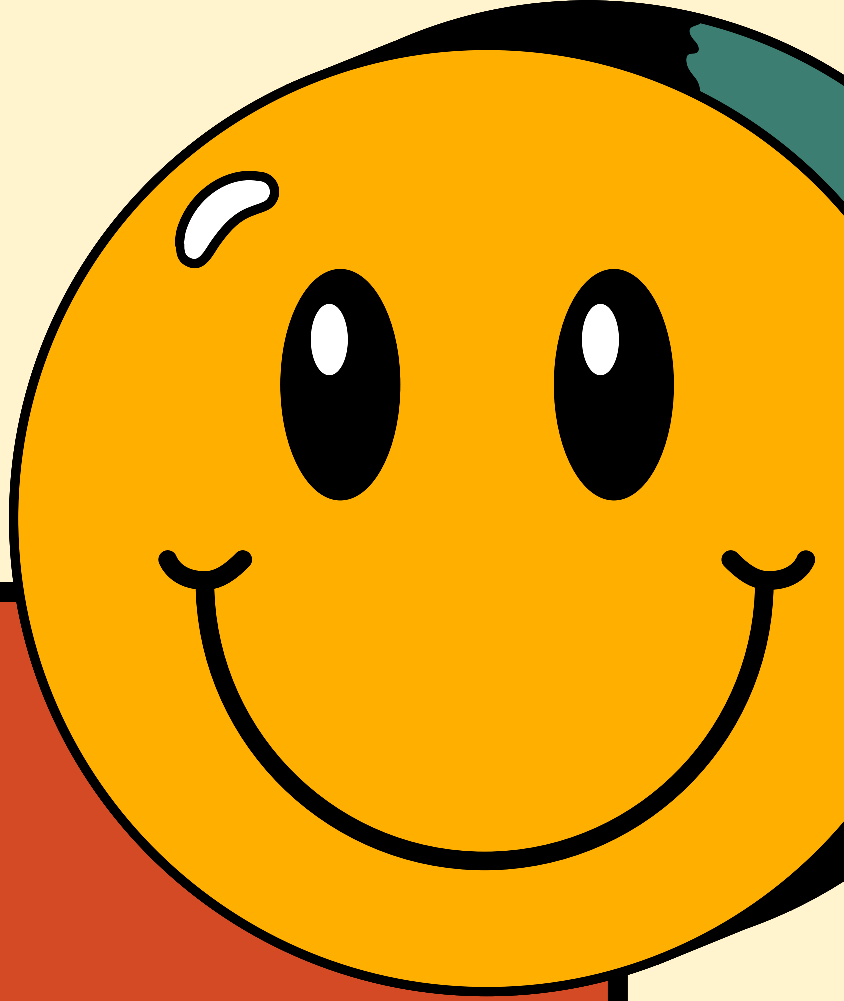


```
SELECT
COUNT(order_id) AS total_orders
FROM
orders;
```

2)calculate the total revenue generated from pizza sales

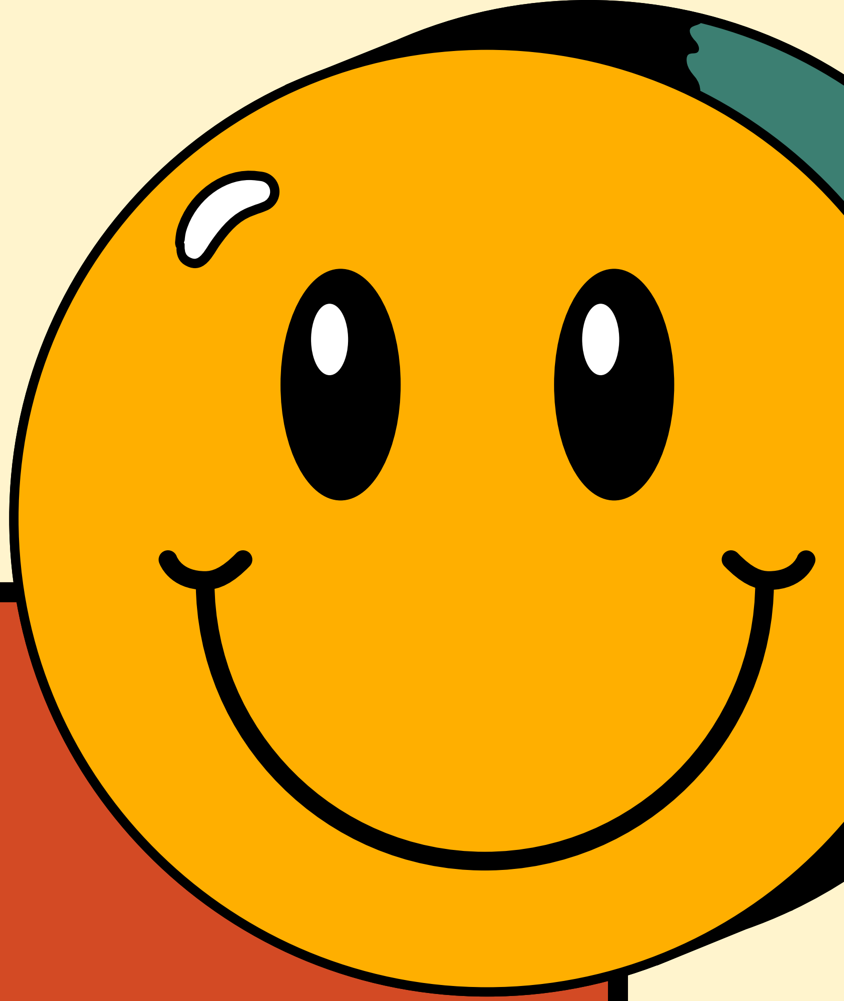
```
SELECT
ROUND(SUM(order_details.quantity * pizzas.price),
2) AS total_sales
FROM
order_details
JOIN
pizzas ON order_details.pizza_id = pizzas.pizza_id;
```

3)to identify the highest prized pizza



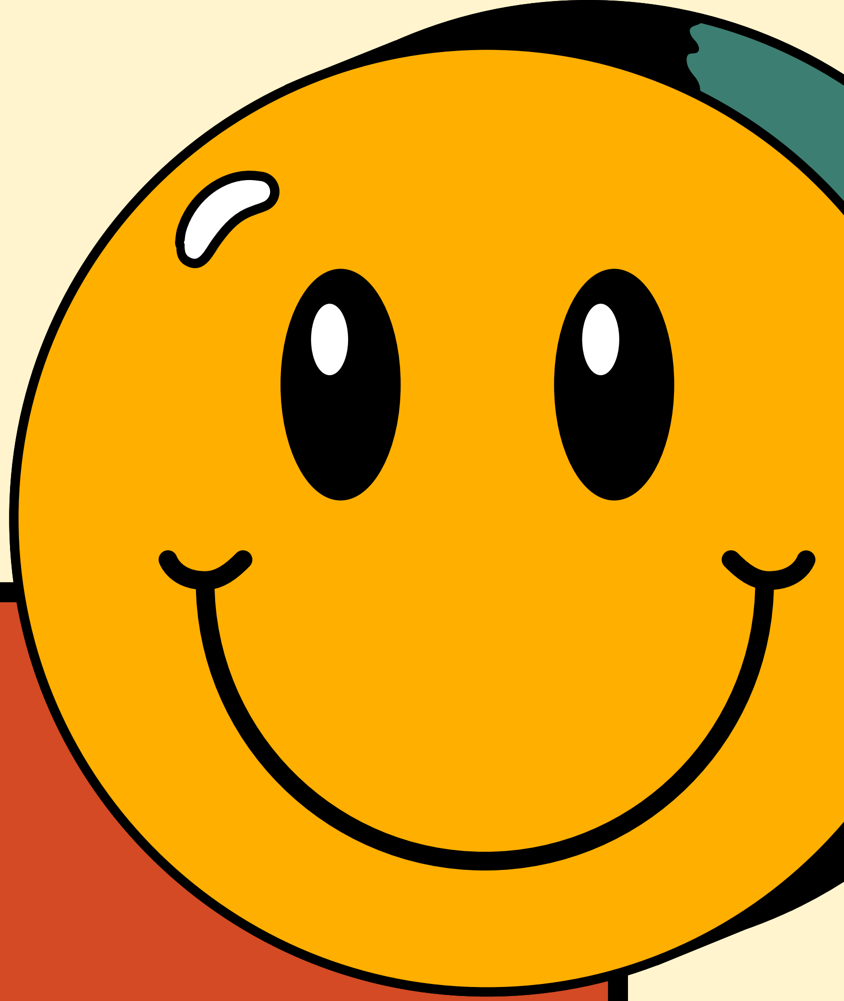
```
SELECT
pizza_types.name, pizzas.price AS highest_price
FROM
pizza_types
JOIN
pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

4)identify the most common pizza size ordered



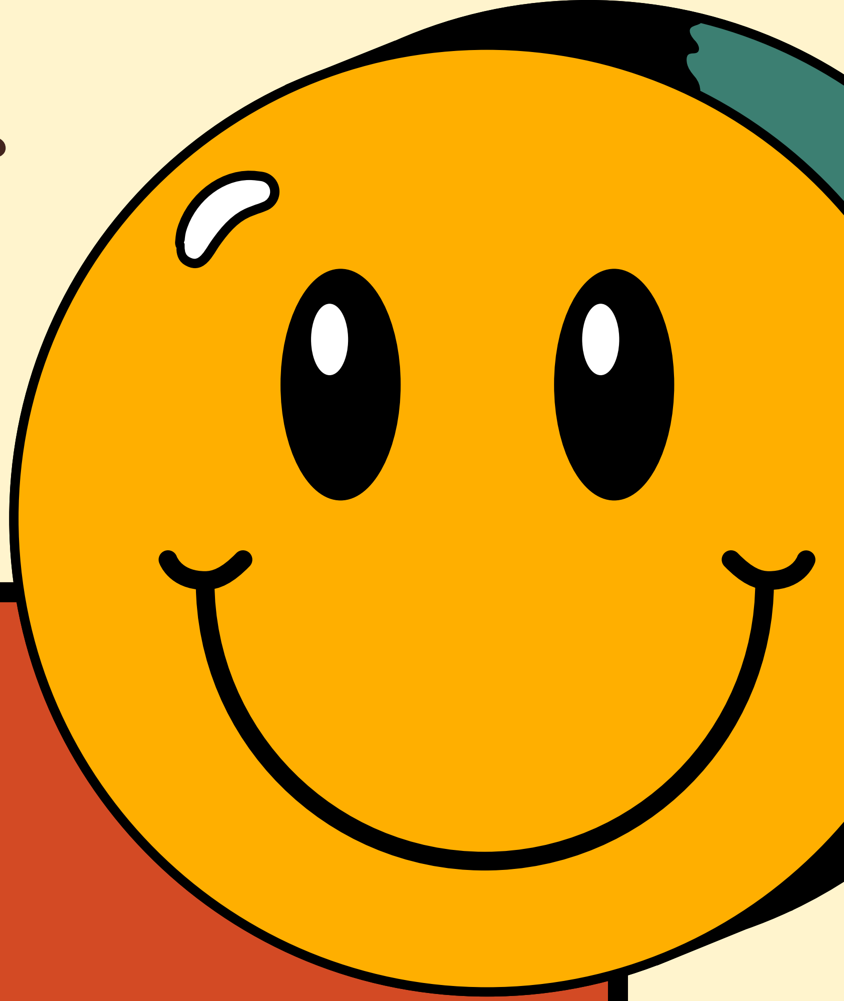
```
SELECT
pizzas.size,
COUNT(order_details.order_details_id) AS order_count
FROM
order_details
JOIN
pizzas ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC
LIMIT 1;
```

5)list the 5 most ordered pizza types along with their quantities



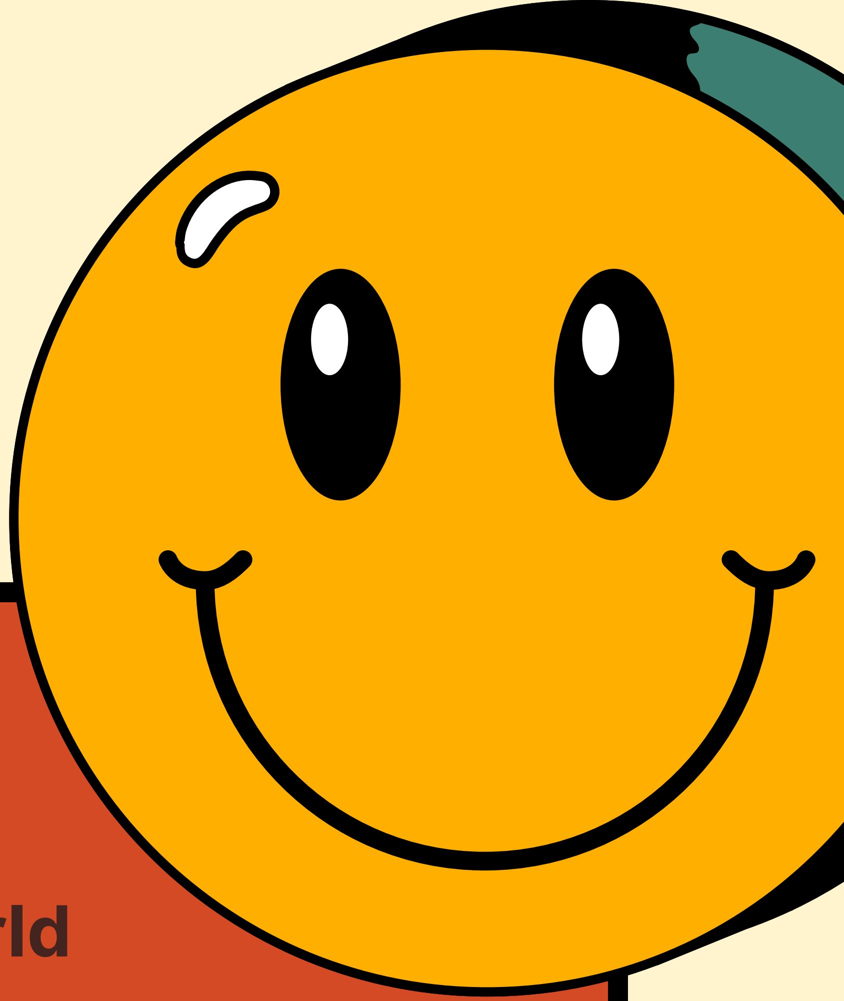
```
SELECT
pizza_types.name,
SUM(order_details.quantity) AS
total_quantity
FROM
pizza_types
JOIN
pizzas ON pizzas.pizza_type_id =
pizza_types.pizza_type_id
JOIN
order_details ON order_details.pizza_id
= pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY total_quantity DESC
LIMIT 5;
```

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



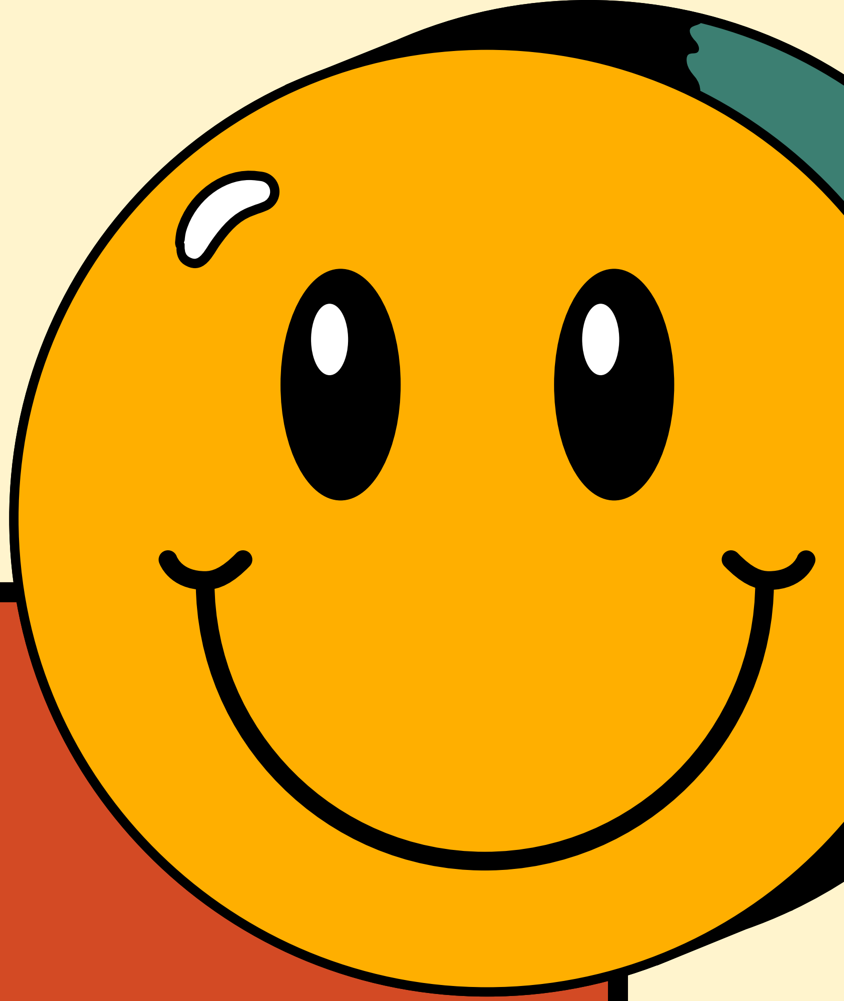
```sql
SELECT
pizza_types.category,
SUM(order_details.quantity) AS total_quantity
FROM
pizza_types
JOIN
pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN
order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY total_quantity DESC;
```

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



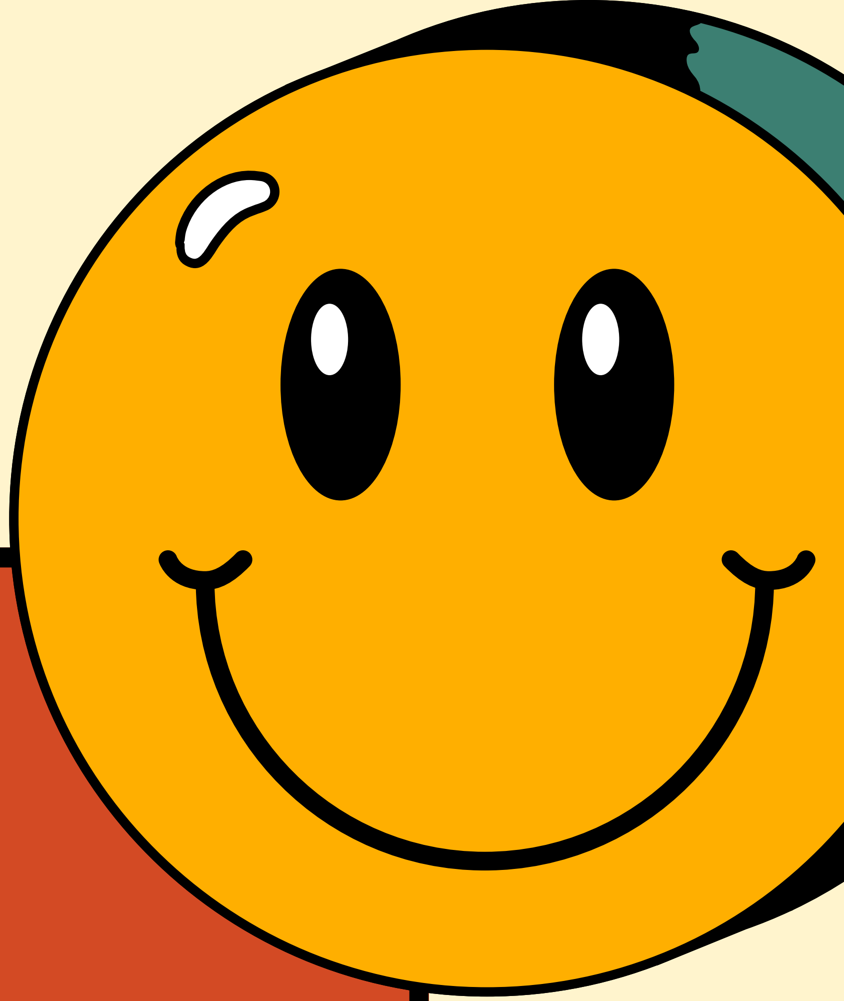
```
SELECT
HOUR(order_time) AS hours, COUNT(order_id) AS count_orderId
FROM
orders
GROUP BY hours
ORDER BY count_orderId DESC;
```

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



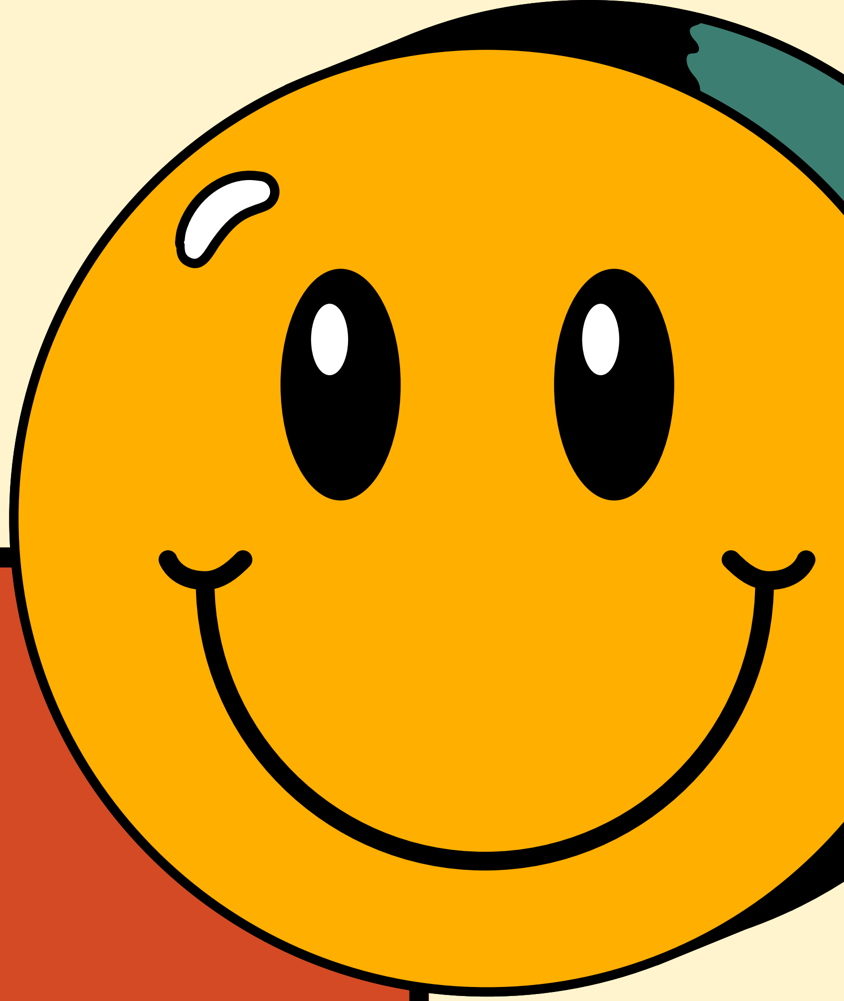
```
SELECT
category, COUNT(name)
FROM
pizza_types
GROUP BY category;
```

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



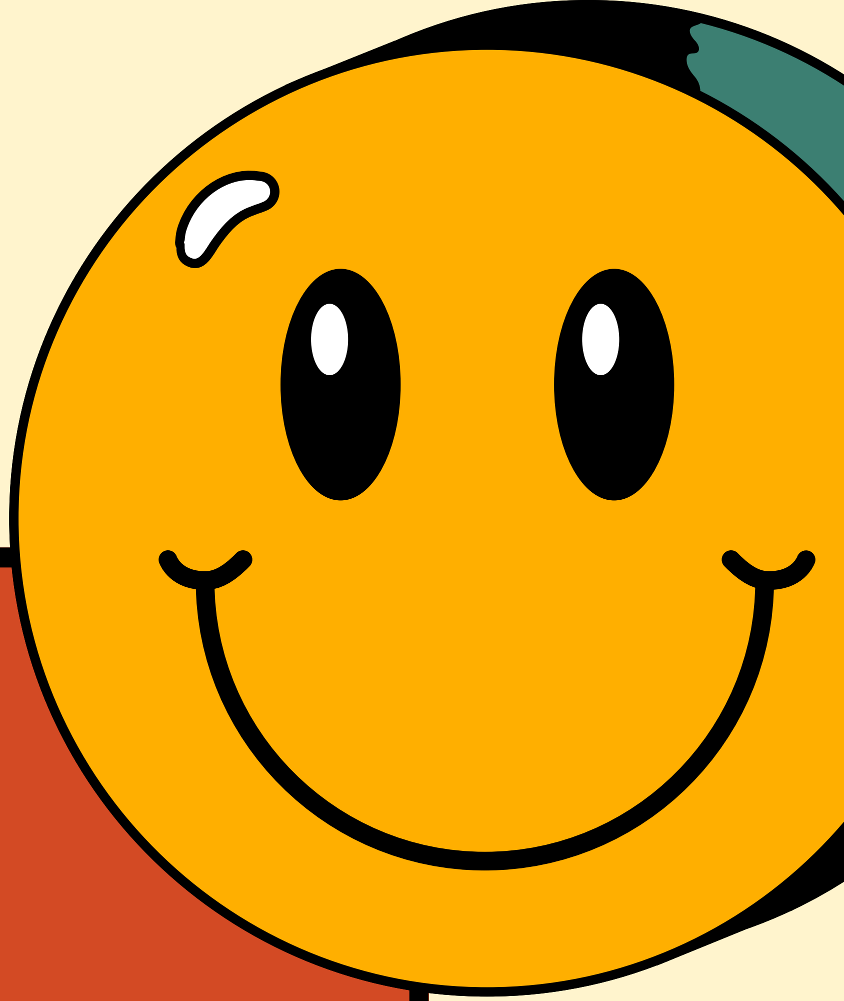
```
SELECT
AVG(quantity)
FROM
(SELECT
orders.order_date, SUM(order_details.quantity) AS quantity
FROM
order_details
JOIN orders ON orders.order_id = order_details.order_id
GROUP BY orders.order_date) AS order_quantity;
```

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
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 revenue DESC
LIMIT 3;
```

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



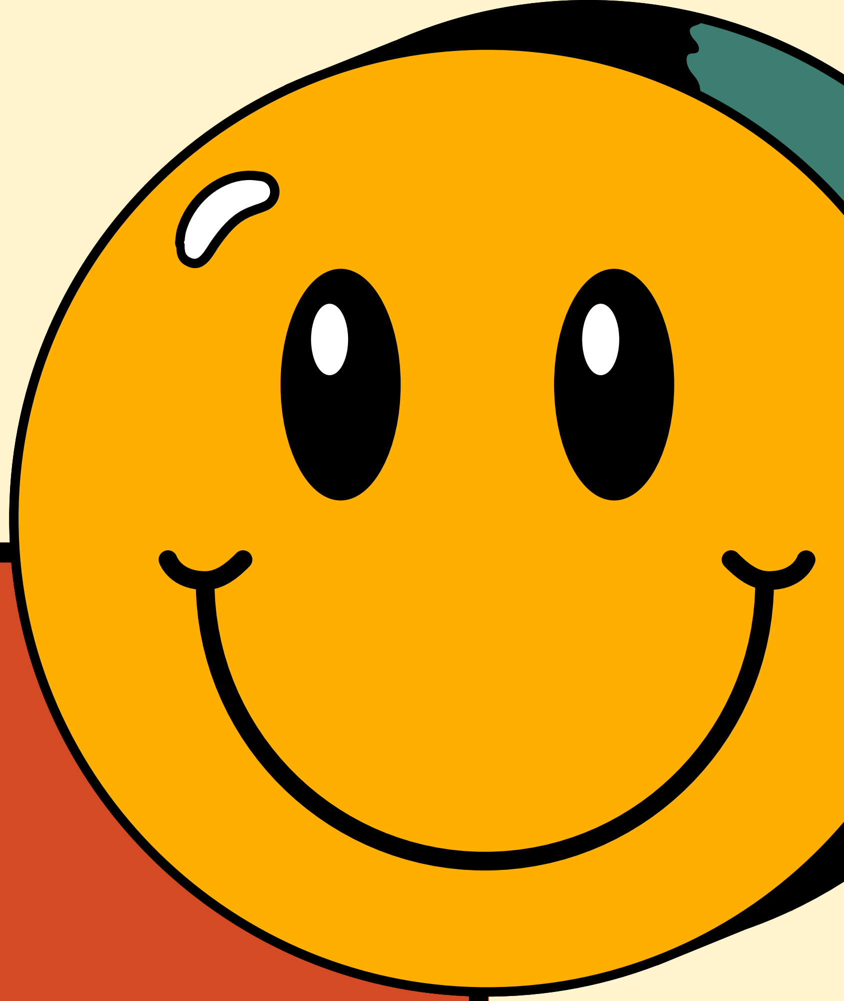
```
SELECT
pizza_types.category,
(SUM(order_details.quantity * pizzas.price)) / (SELECT
ROUND(SUM(order_details.quantity * pizzas.price),
2) AS total_sales
FROM
order_details
JOIN
pizzas ON order_details.pizza_id = pizzas.pizza_id)*100 AS revenue
FROM
pizza_types
JOIN
pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN
order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC;
```

12)Analyze the cumulative revenue generated over time

```
select order_date,sum(revenue) over( rows between unbounded preceding and
current row ) as cumulative_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
order_details.order_id=orders.order_id group by orders.order_date) as sales;
```

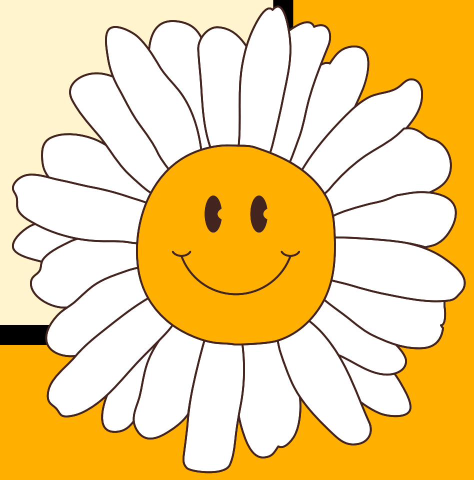
Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```
SELECT
name,
revenue
FROM (
SELECT
category,
name,
revenue,
RANK() OVER(PARTITION BY category ORDER BY revenue DESC) AS rnk
FROM (
SELECT
pizza_types.category,
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 pizzas.pizza_id = order_details.pizza_id
GROUP BY
pizza_types.category,
pizza_types.name
) AS pizza_table
) AS b
WHERE rnk <= 3;
```



Conclusion

This project not only showcases my proficiency in SQL but also my ability to apply data-driven approaches to real-world business problems.



Thank You