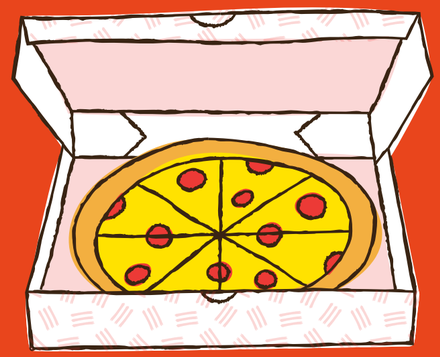
sql mini project

This project focused on designing and executing SQL queries to analyze and manage data for a pizza company. Key tasks included optimizing data retrieval, generating insightful reports, and supporting data-driven decision-making.

-- Retrieve the total number of orders placed.

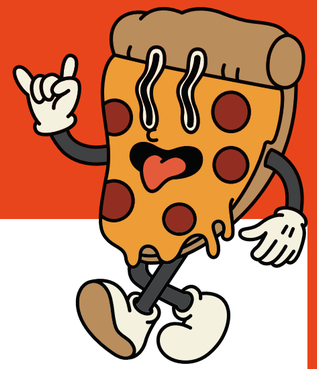
```sql
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

-- Calculate the total revenue generated from pizza sales.



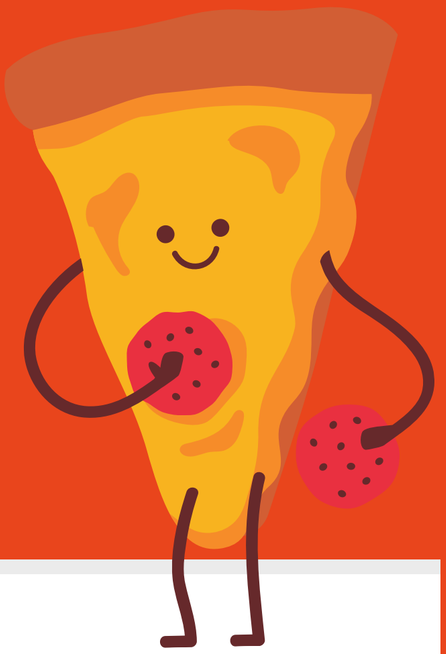
```sql
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS total_revenue
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id;
```

-- 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;
```

-- Identify the most common pizza size ordered.alter



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

-- Identify the most common size orderd.

```sql
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;
```

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

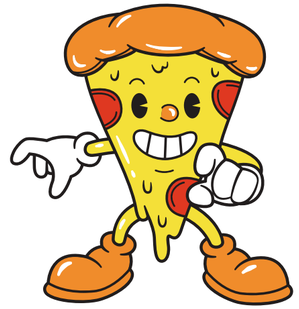
```sql
SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY SUM(order_details.quantity)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

-- join the necessary table to find the toal 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;
```

-- Determine the distribution of orders by hours of the day;

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



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

```sql
select category,count(name)from pizza_types
group by category ;
```

-- group the orders by date and calculate the average numbers of pizza ordered per day ,

```sql
SELECT
    round( AVG(quantity),2)
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;
```

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

```sql
select pizza_types.name,  sum(order_details.quantity*pizzas.price ) 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  sum(order_details.quantity*pizzas.price ) desc limit 3 ;
```

-- calculate the percentage contribution of each pizza typr 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_revenue
                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 SUM(order_details.quantity * pizzas.price) DESC;
```

-- analyze the cummulative 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 orders
join order_details
on order_details.order_id = orders.order_id
join pizzas
on pizzas.pizza_id = order_details.pizza_id
group by orders.order_date ) as sales ;
```

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

```sql
select category, 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 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, pizza_types.name
 order by
 sum(order_details.quantity*pizzas.price)
 desc) as a ) as b
 where rn<=3;
```