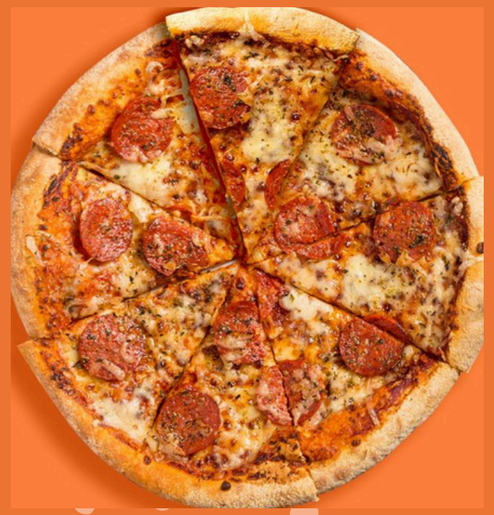
Analyzing Pizza Sales with SQL

In this project, we delve into the world of pizza sales analysis using SQL to uncover valuable insights, to analyze a comprehensive dataset encompassing various aspects of pizza sales, including but not limited to sales figures, most selling, popular toppings.

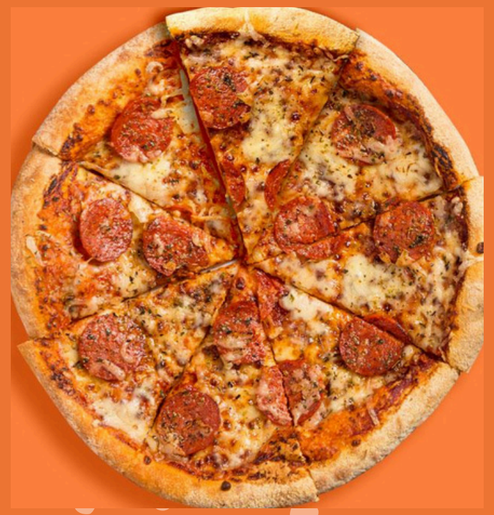


Retrieve the total number of orders placed.

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

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

Calculate the total revenue generated from pizza sales.



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.

```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 top 5 most 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 quantity DESC
LIMIT 5;
```

Join the necessary tables to find the total 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 hour of the day.

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

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

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

```sql
SELECT
    ROUND(AVG(order_quantity.quantity), 0)
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;
```

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

```sql
SELECT
    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.name
ORDER BY revenue DESC
LIMIT 3;
```

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

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

```sql
SELECT
    pizza_types.category,
    ROUND((SUM(order_details.quantity * pizzas.price) /
    (SELECT SUM(order_details.quantity * pizzas.price) AS total_sales
    FROM order_details JOIN pizzas ON pizzas.pizza_id = order_details.pizza_id)) * 100, 2) 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;
```

Analyze the cumulative revenue generated over time.

```sql
SELECT
    sales.order_date,
    SUM(sales.revenue) OVER (ORDER BY sales.order_date) 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;
```