## SQL Pizza Sales Analysis Project

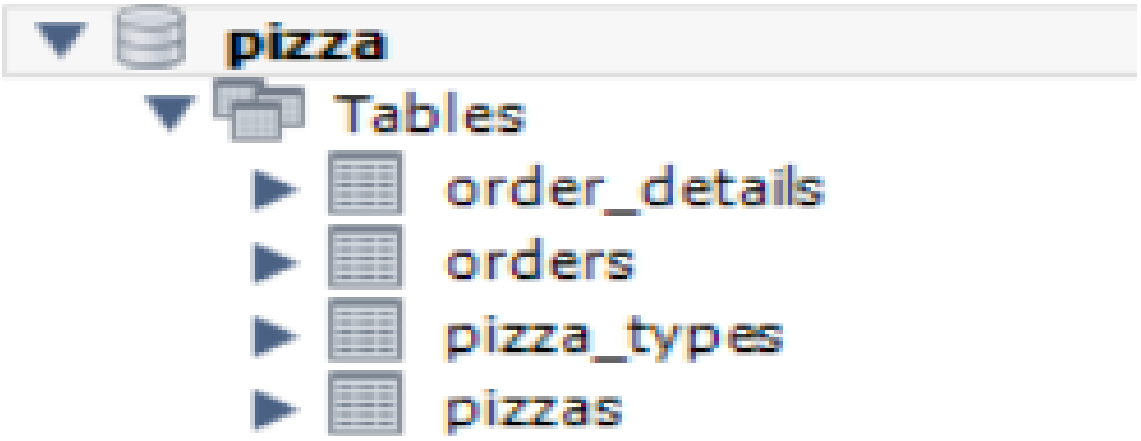
```
CREATE DATABASE pizza;

USE pizza;

CREATE TABLE orders(
order_id int not null,
order_date date not null,
order_time time not null,
primary key(order_id));


CREATE TABLE order_details(
order_details_id int not null,
order_id int not null,
pizza_id text not null,
quantity int not null,
primary key(order_details_id));
```

Here I am creating one pizza name database. And in this database I am adding 4 tables – order_details, orders, pizzas, pizza_type.




- Below are the 13 questions that I analyze through SQL.

```sql
-- Retrive the total number of orders placed.
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| total_orders |
| --- |
| 21350 |

```sql
-- calculate the total revenue from pizza sales.
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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| Total_revenue |
| --- |
| 55113.1 |

```sql
-- Identify the highest-priced pizza.
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;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

| name | price |
| --- | --- |
| The Greek Pizza | 35.95 |

```sql
-- 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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| size | order_count |
|---|---|
| L | 1256 |
| M | 1021 |
| S | 971 |
| XL | 40 |

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

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

| name | Quantity |
|---|---|
| The Pepperoni Pizza | 195 |
| The Thai Chicken Pizza | 164 |
| The Barbecue Chicken Pizza | 162 |
| The California Chicken Pizza | 161 |
| The Classic Deluxe Pizza | 152 |

```sql
-- Join the necessary tables to find the total quantity of each pizza category ordered.
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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| category | Quantity |
|---|---|
| Classic | 992 |
| Supreme | 825 |
| Veggie | 805 |
| Chicken | 731 |

```sql
-- 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);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| hour | order_count |
|---|---|
| 11 | 1231 |
| 12 | 2520 |
| 13 | 2455 |
| 14 | 1472 |
| 15 | 1468 |
| 16 | 1920 |
| 17 | 2336 |
| 18 | 2399 |
| 19 | 2009 |
| 20 | 1642 |

```sql
-- Join relavant tables to find the category-wise distribution of pizzas.
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| category | COUNT(name) |
|---|---|
| Chicken | 6 |
| Classic | 8 |
| Supreme | 9 |
| Veggie | 9 |

```sql
-- Group the orders by date and calculate the average number of pizzas ordered per day.
SELECT
    ROUND(AVG(quantity), 0) AS avg_pizza_ordered_per_day
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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| avg_pizza_ordered_per_day |
|---|
| 134 |

```sql
-- Determin 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 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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

| name | revenue |
|---|---|
| The Thai Chicken Pizza | 3003 |
| The Barbecue Chicken Pizza | 2841.5 |
| The California Chicken Pizza | 2756.75 |

```sql
-- 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 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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| category | revenue |
|---|---|
| Classic | 26.48 |
| Supreme | 25.64 |
| Veggie | 24.5 |
| Chicken | 23.38 |

```sql
-- Analyze the cumulative revenue generated over time.
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 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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| order_date | CUM_revenue |
|---|---|
| 2015-01-01 | 2713.8500000000004 |
| 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 |

```sql
-- 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 rn
    FROM (
        SELECT pizza_types.category,
               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.category, pizza_types.name
    ) AS a
) AS b
WHERE rn <= 3;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| name | revenue |
|---|---|
| The Thai Chicken Pizza | 3003 |
| The Barbecue Chicken Pizza | 2841.5 |
| The California Chicken Pizza | 2756.75 |
| The Pepperoni Pizza | 2454 |
| The Classic Deluxe Pizza | 2367 |
| The Greek Pizza | 1946 |
| The Italian Supreme Pizza | 2516 |
| The Sicilian Pizza | 2313.5 |
| The Spicy Italian Pizza | 2272.5 |
| The Five Cheese Pizza | 2164.5 |