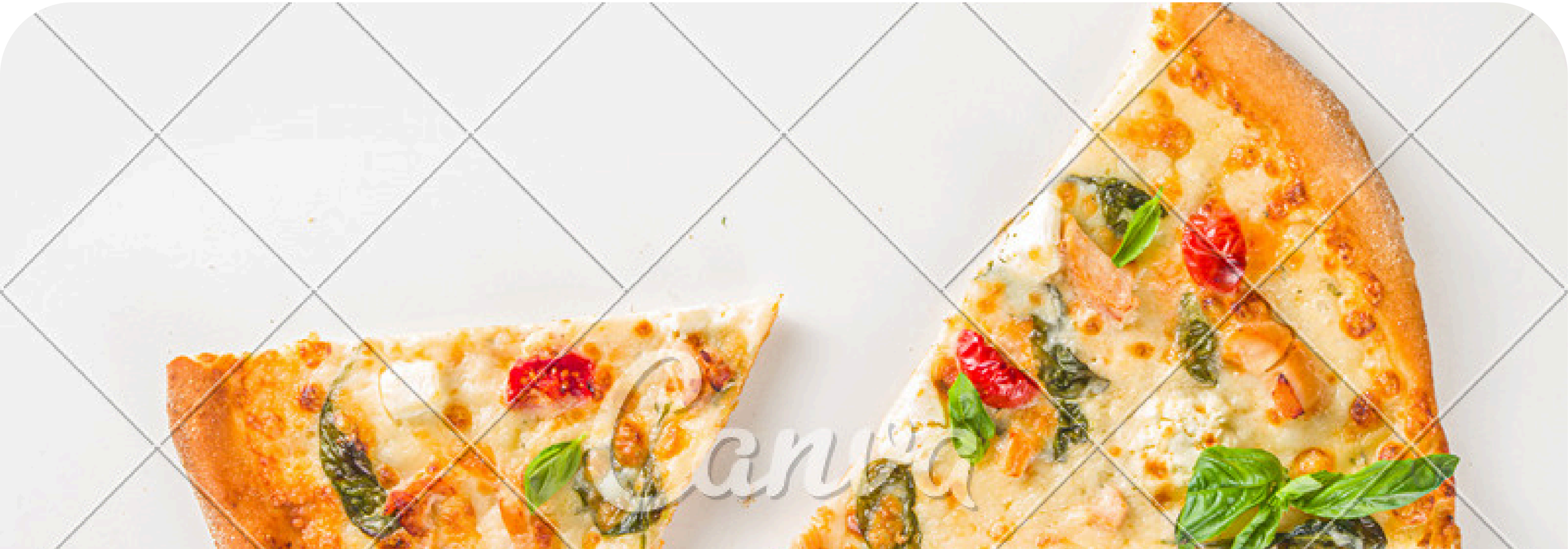
Hello!

My name is Harsh Sharma and in this project i have utilised the SQL queries to solve the questions that were related to Pizza sales!

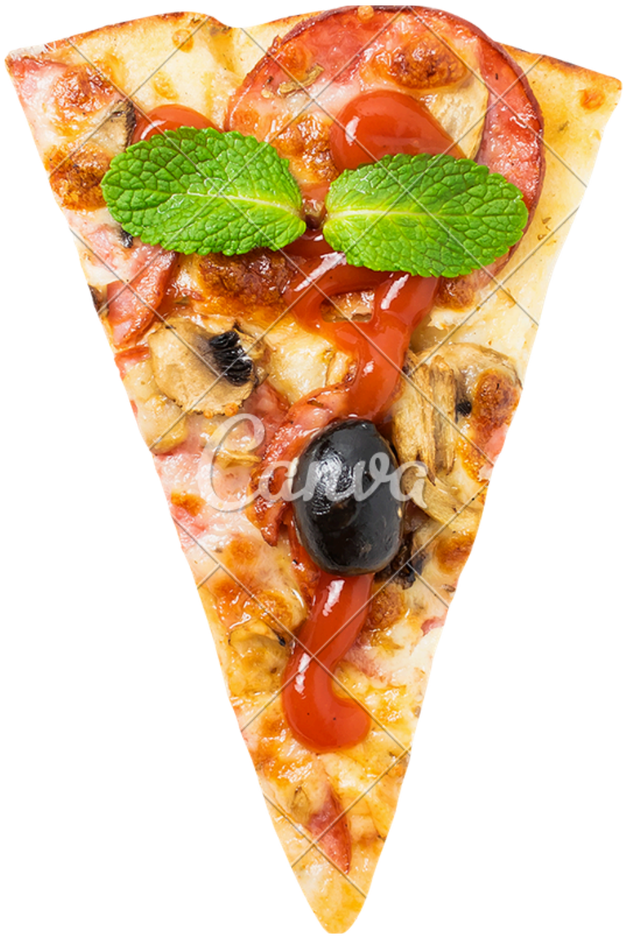


Overview

A pizza sales project using SQL helps analyze sales data from a pizza restaurant chain to gain valuable business insights.

This analysis can identify trends, answer questions, and support data-driven decisions such as:

- **Best-selling pizzas:** Identify the most popular pizzas by quantity sold, revenue generated, or both.
- **Customer behavior:** Analyze average order value, number of pizzas per order, or customer preferences for different pizza sizes and categories



Here is the list of all the question which i have answered in this project through using SQL

```sql
-- Basic:
-- Retrieve the total number of orders placed.
-- Calculate the total revenue generated from pizza sales.
-- Identify the highest-priced pizza.
-- Identify the most common pizza size ordered.
-- List the top 5 most ordered pizza types along with their quantities.


-- Intermediate:
-- Join the necessary tables to find the total quantity of each pizza category ordered.
-- Determine the distribution of orders by hour of the day.
-- Join relevant tables to find the category-wise distribution of pizzas.
-- Group the orders by date and calculate the average number of pizzas ordered per day.
-- Determine the top 3 most ordered pizza types based on revenue.

-- Advanced:
-- Calculate the percentage contribution of each pizza type to total revenue.
-- Analyze the cumulative revenue generated over time.
-- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
```

Creating the database for the project

```sql
CREATE DATABASE PIZZAHUT;
USE PIZZAHUT;

CREATE TABLE ORDERS (
ORDER_ID INT PRIMARY KEY,
ORDER_DATE DATE,
ORDER_TIME TIME
);

CREATE TABLE ORDER_DETAILS (
ORDER_DETAILS_ID INT PRIMARY KEY,
ORDERS_ID INT,
PIZZA_ID TEXT,
QUANTITY INT
);

CREATE TABLE PIZZAS (
PIZZA_ID TEXT PRIMARY KEY,
PIZZA_TYPES_ID TEXT,
SIZE varchar(5),
PRICE float
);

CREATE TABLE PIZZA_TYPES (
PIZZA_TYPE_ID TEXT PRIMARY KEY,
NAME TEXT,
CATEGORY VARCHAR(20),
INGREDIENTS VARCHAR(100)
);
```

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)
                FROM
                    ORDER_DETAILS
                        INNER JOIN
                    PIZZAS ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID)) * 100,
            2) AS PERCENTAGE
FROM
    ORDER_DETAILS
        INNER JOIN
    PIZZAS ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID
        INNER JOIN
    PIZZA_TYPES ON PIZZA_TYPES.PIZZA_TYPE_ID = PIZZAS.PIZZA_TYPE_ID
GROUP BY PIZZA_TYPES.CATEGORY
ORDER BY PERCENTAGE DESC;
```

Result Grid | Filter Rows:

CATEGORY	PERCENTAGE
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

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
        INNER JOIN
    PIZZAS ON PIZZA_TYPES.PIZZA_TYPE_ID = PIZZAS.PIZZA_TYPE_ID
        INNER 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;
```

esult Grid | Filter Rov

CATEGORY	PERCENTAGE
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

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

```sql
SELECT
    ROUND(AVG(quantity), 0) AS avg_pizza_ordered_per_day
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    INNER JOIN order_details ON orders.order_id = order_details.orders_id
    GROUP BY orders.order_date) AS o;
```

sult Grid | Filter Rows

avg_pizza_ordered_per_day
138

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

```sql
SELECT
    PIZZA_TYPES.CATEGORY, SUM(ORDER_DETAILS.QUANTITY)
FROM
    ORDER_DETAILS
        INNER JOIN
    PIZZAS ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID
        INNER JOIN
    PIZZA_TYPES ON PIZZA_TYPES.PIZZA_TYPE_ID = PIZZAS.PIZZA_TYPE_ID
GROUP BY PIZZA_TYPES.CATEGORY;
```

esult Grid | Filter Rows:

CATEGORY	SUM(ORDER_DETAILS.QUANTITY)
Classic	14888
Veggie	11649
Supreme	11987
Chicken	11050

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

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

Result Grid | Filter Rows:

name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

Analyze the cumulative revenue generated over time.

```sql
sELECT ORDER_DATE , SUM(REVENUE)
OVER (ORDER BY ORDER_DATE)
FROM
 (SELECT ORDERS.ORDER_DATE ,
 SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE) AS REVENUE
 FROM
 ORDER_DETAILS
 INNER JOIN PIZZAS ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID
 INNER JOIN
 ORDERS ON ORDERS.ORDER_ID = ORDER_DETAILS.ORDERS_ID
 GROUP BY ORDERS.ORDER_DATE) AS T;
```

ORDER_DATE	SUM(REVENUE) OVER (ORDER BY ORDER_DATE)
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
2015-01-11	25862.65
2015-01-12	27781.7
2015-01-13	29831.300000000003
2015-01-14	32358.700000000004
2015-01-15	34343.50000000001
2015-01-16	36937.65000000001
2015-01-17	39001.75000000001
2015-01-18	40978.600000000006
2015-01-19	43365.75000000001
2015-01-20	45763.65000000001
2015-01-21	47804.20000000001
2015-01-22	50390.00000000001

Result 1

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

```
SELECT CATEGORY , NAME , REVENUE , RN
FROM
(SELECT CATEGORY , NAME, REVENUE,
 ROW_NUMBER() OVER ( PARTITION BY CATEGORY ORDER BY REVENUE DESC ) AS RN
 FROM
 (SELECT PIZZA_TYPES.CATEGORY , PIZZA_TYPES.NAME ,
 ROUND(SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE),2) AS REVENUE
 FROM  PIZZA_TYPES
INNER JOIN
PIZZAS ON PIZZA_TYPES.PIZZA_TYPE_ID = PIZZAS.PIZZA_TYPE_ID
INNER JOIN
ORDER_DETAILS ON PIZZAS.PIZZA_ID = ORDER_DETAILS.PIZZA_ID
 GROUP BY PIZZA_TYPES.CATEGORY, PIZZA_TYPES.NAME) AS A)AS B WHERE RN <4;
```

sult Grid | Filter Rows: | Export:

CATEGORY	NAME	REVENUE	RN
Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768	2
Chicken	The California Chicken Pizza	41409.5	3
Classic	The Classic Deluxe Pizza	38180.5	1
Classic	The Hawaiian Pizza	32273.25	2
Classic	The Pepperoni Pizza	30161.75	3
Supreme	The Spicy Italian Pizza	34831.25	1
Supreme	The Italian Supreme Pizza	33476.75	2
Supreme	The Sicilian Pizza	30940.5	3
Veggie	The Four Cheese Pizza	32265.7	1
Veggie	The Mexicana Pizza	26780.75	2
Veggie	The Five Cheese Pizza	26066.5	3

FIND 2ND HIGHEST PRICED PIZZA NAME

```
SELECT
    PIZZA_TYPES.NAME, SUM(PIZZAS.PRICE)
FROM
    PIZZAS
        INNER JOIN
    PIZZA_TYPES ON PIZZAS.PIZZA_TYPE_ID = PIZZA_TYPES.PIZZA_TYPE_ID
GROUP BY PIZZA_TYPES.NAME
ORDER BY SUM(PIZZAS.PRICE) DESC
LIMIT 1 OFFSET 1;
```

Result Grid | Filter Rows:

NAME	SUM(PIZZAS.PRICE)
The Italian Vegetables Pizza	50.5

FIND COUNT OF “M” SIZED PIZZAIZZA NAME

```
SELECT
    COUNT(PIZZAS.SIZE)
FROM
    PIZZAS
        INNER JOIN
    ORDER_DETAILS ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID
WHERE
    PIZZAS.SIZE = 'M';
```

Result Grid | Filte

COUNT(PIZZAS.SIZE)
15385

FIND 3RD HIGHEST SIZED PIZZA BASED ON REVENUE

```
SELECT
    PIZZAS.SIZE,
    ROUND(SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE),
            2) AS REVENUE
FROM
    ORDER_DETAILS
        INNER JOIN
    PIZZAS ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID
GROUP BY PIZZAS.SIZE
ORDER BY REVENUE DESC
LIMIT 1 OFFSET 2;
```

Result Grid

SIZE	REVENUE
S	178076.5

DETERMINE TOP 2 ORDERED PIZZA CATEGORY BASED ON REVENUE

```sql
SELECT
    PIZZA_TYPES.CATEGORY,
    ROUND(SUM(ORDER_DETAILS.QUANTITY * PIZZAS.PRICE),
            0) AS REVENUE
FROM
    ORDER_DETAILS
        INNER JOIN
    PIZZAS ON ORDER_DETAILS.PIZZA_ID = PIZZAS.PIZZA_ID
        INNER JOIN
    PIZZA_TYPES ON PIZZA_TYPES.PIZZA_TYPE_ID = PIZZAS.PIZZA_TYPE_ID
GROUP BY PIZZA_TYPES.CATEGORY
ORDER BY REVENUE DESC
LIMIT 2;
```

Result Grid | Filter

CATEGORY	REVENUE
Classic	220053
Supreme	208197

Thank you
all!