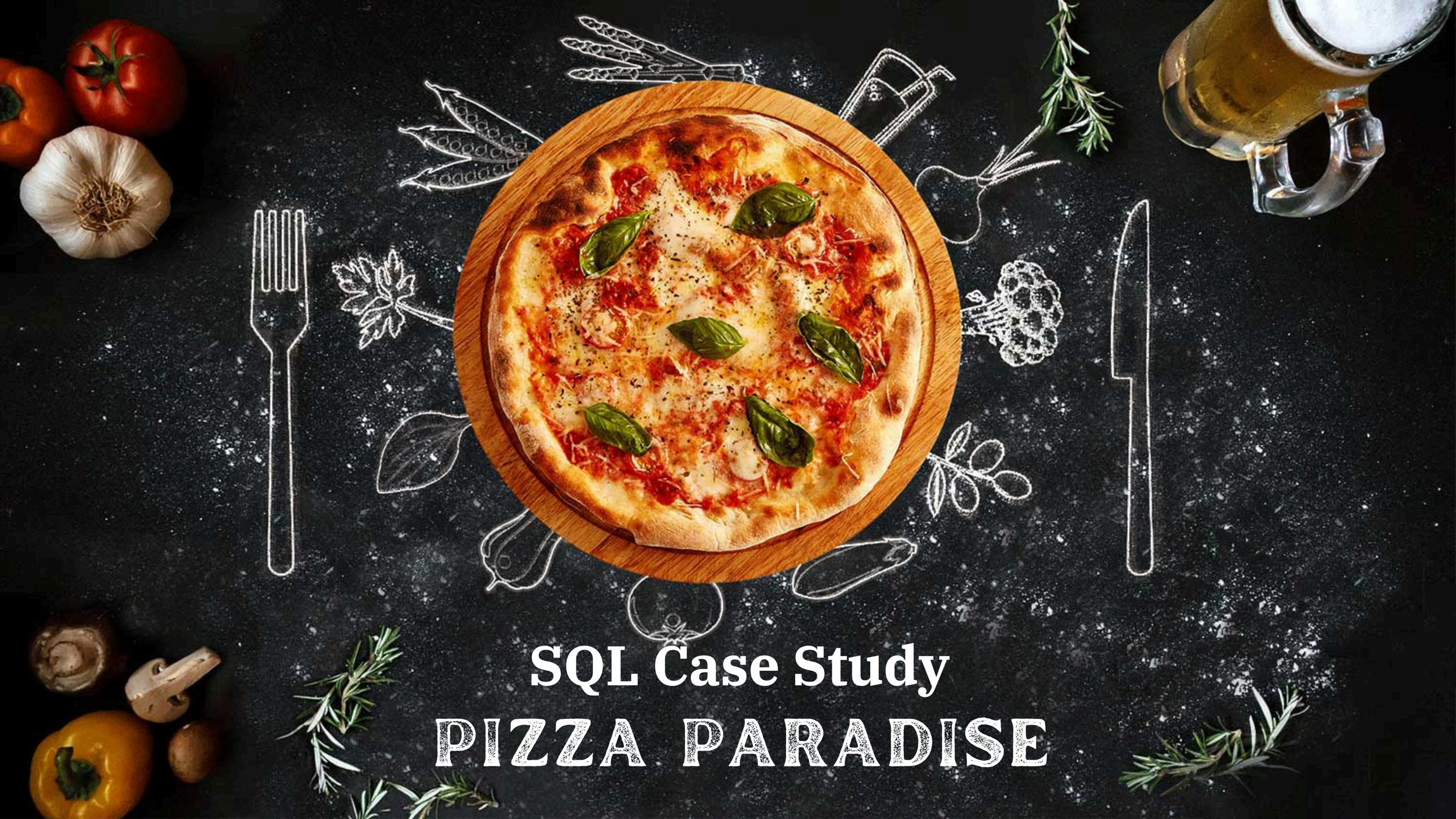

SQL Case Study

PIZZA PARADISE

Table of Contents

INTRODUCTION

“Pizza Paradise” a restaurant established in 2015 has become known for offering a wide variety of pizzas, catering to both vegetarian and non-vegetarian preferences.

“Pizza Paradise” needs assistance to help the restaurant stay afloat. The restaurant has some basic data, but they have no idea how to use their data to help them to run their business.

PROBLEM STATEMENT

This case study is to craft SQL queries of Pizza Paradise's sales performance, like to determine the total sales, which types and sizes of pizzas were most frequently ordered by customers, the quantities sold of each categories of pizzas, the specific times and dates of the orders, and the number of sales per day. This shall provide actionable insights to guide the owner in effectively running and growing their business.

TABLES

- orders
- order_details
- pizzas
- pizza_types

Field Types

#	Field	Schema	Table	Type
1	order_id	pizzaparadise	orders	INT
2	order_date	pizzaparadise	orders	DATE
3	order_time	pizzaparadise	orders	TIME

Field Types

#	Field	Schema	Table	Type
1	order_details_id	pizzaparadise	order_details	INT
2	order_id	pizzaparadise	order_details	INT
3	pizza_id	pizzaparadise	order_details	TEXT
4	quantity	pizzaparadise	order_details	INT

Field Types

#	Field	Schema	Table	Type
1	pizza_id	pizzaparadise	pizzas	TEXT
2	pizza_type_id	pizzaparadise	pizzas	TEXT
3	size	pizzaparadise	pizzas	TEXT
4	price	pizzaparadise	pizzas	DOUBLE

Field	Schema	Table	Type
pizza_type_id	pizzaparadise	pizza_types	TEXT
name	pizzaparadise	pizza_types	TEXT
category	pizzaparadise	pizza_types	TEXT
ingredients	pizzaparadise	pizza_types	TEXT

TABLE DETAILS

- orders

  The orders table captures order_id, order_date and order_time for the all the orders that has been placed by customers.

- order_details

  The order_details table captures order_id, pizza_id and quantity for each pizza.

- pizzas

  The pizzas table captures pizza_id, pizza_type_id, size and price for each pizza that has been ordered.

- pizza_types

  The pizza_types table captures pizza_type_id, name, category, and ingredients that each pizza contains.

Case Study Questions and Answers

Problem Statement 1

1. Retrieve the total number of orders placed.

```
select * from orders;
select count(order_id) as total_orders from orders;
```

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

total_orders
21350

Problem Statement 2

2. Calculate the total revenue generated 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

total_revenue
817860.05

Problem Statement 3

3. 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:

name	price
The Greek Pizza	35.95

Problem Statement 4

4. Identify the most common pizza size ordered.

```sql
SELECT
    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:

size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

Problem Statement 5

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

```sql
SELECT
    pizza_types.name,
    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.name
ORDER BY total_quantity DESC
LIMIT 5;
```

Result Grid | Filter Rows:

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

Problem Statement 6

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;
```

Result Grid | Filter Rows:

category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Problem Statement 7

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

```
select hour(order_time) as hours, count(order_id) as order_count from orders
group by hours;
```

Result Grid | Filter Rows:

hours	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

Problem Statement 8

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

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

Result Grid | Filter Rows:

category	count(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

Problem Statement 9

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

```
SELECT
    ROUND(AVG(total_quantity), 0) AS avg_pizza_ordered_per_day
FROM
    (SELECT
        orders.order_date,
            SUM(order_details.quantity) AS total_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:

avg_pizza_ordered_per_day
138

Problem Statement 10

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

```sql
SELECT
    pizza_types.name,
    SUM(pizzas.price * order_details.quantity) 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;
```

Result Grid | Filter Rows:

name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

Problem Statement 11

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

```
SELECT
    pizza_types.category,
    ROUND((SUM(pizzas.price * order_details.quantity) / (SELECT
                    ROUND(SUM(pizzas.price * order_details.quantity),
                                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
LIMIT 3;
```

Result Grid | Filter Rows:

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96

Problem Statement 12

12. Analyze the cumulative revenue generated over time.

```sql
select order_date,
sum(revenue) over(order by order_date) 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 orders.order_id = order_details.order_id
group by orders.order_date) as sales;
```

Result Grid | Filter Rows:

order_date	cumulative_revenue
2015-09-18	596598.6000000003
2015-09-19	598885.1500000004
2015-09-20	600714.1500000004
2015-09-21	602845.6000000003
2015-09-22	605016.0000000003
2015-09-23	607179.0000000003
2015-09-26	609425.8500000003
2015-09-27	611740.5500000003
2015-09-28	613775.8500000003
2015-09-29	616537.9000000004
2015-09-30	618735.9500000004
2015-10-01	621938.1000000004
2015-10-02	624012.9500000004
2015-10-03	626413.9000000004
2015-10-04	628556.1000000003
2015-10-06	630772.0500000003
2015-10-07	632864.4000000003
2015-10-08	634840.2500000002
2015-10-09	637352.8500000002
2015-10-10	639663.0500000002
2015-10-11	641579.3000000002
2015-10-13	643905.2500000001
2015-10-14	646051.6000000001
2015-10-15	650371.8

Problem Statement 13

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

```sql
select name, revenue from
(select category, name, revenue,
rank() over(partition by category order by revenue desc) as new_rank 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 new_table) as new2_table2
where new_rank<=3;
```

Result Grid | Filter Rows:

name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5
The Classic Deluxe Pizza	38180.5
The Hawaiian Pizza	32273.25
The Pepperoni Pizza	30161.75
The Spicy Italian Pizza	34831.25
The Italian Supreme Pizza	33476.75
The Sicilian Pizza	30940.5
The Four Cheese Pizza	32265.70000000065
The Mexicana Pizza	26780.75
The Five Cheese Pizza	26066.5


THANK YOU