SQL PROJECT ON PIZZA SALES

WELCOME TO

SQL PROJECT ON PIZZA SALES

🔍 In this project, I performed an in-depth analysis of a pizza sales dataset using MySQL. The dataset included information on customer orders, pizza types, sizes, prices, and quantities sold.

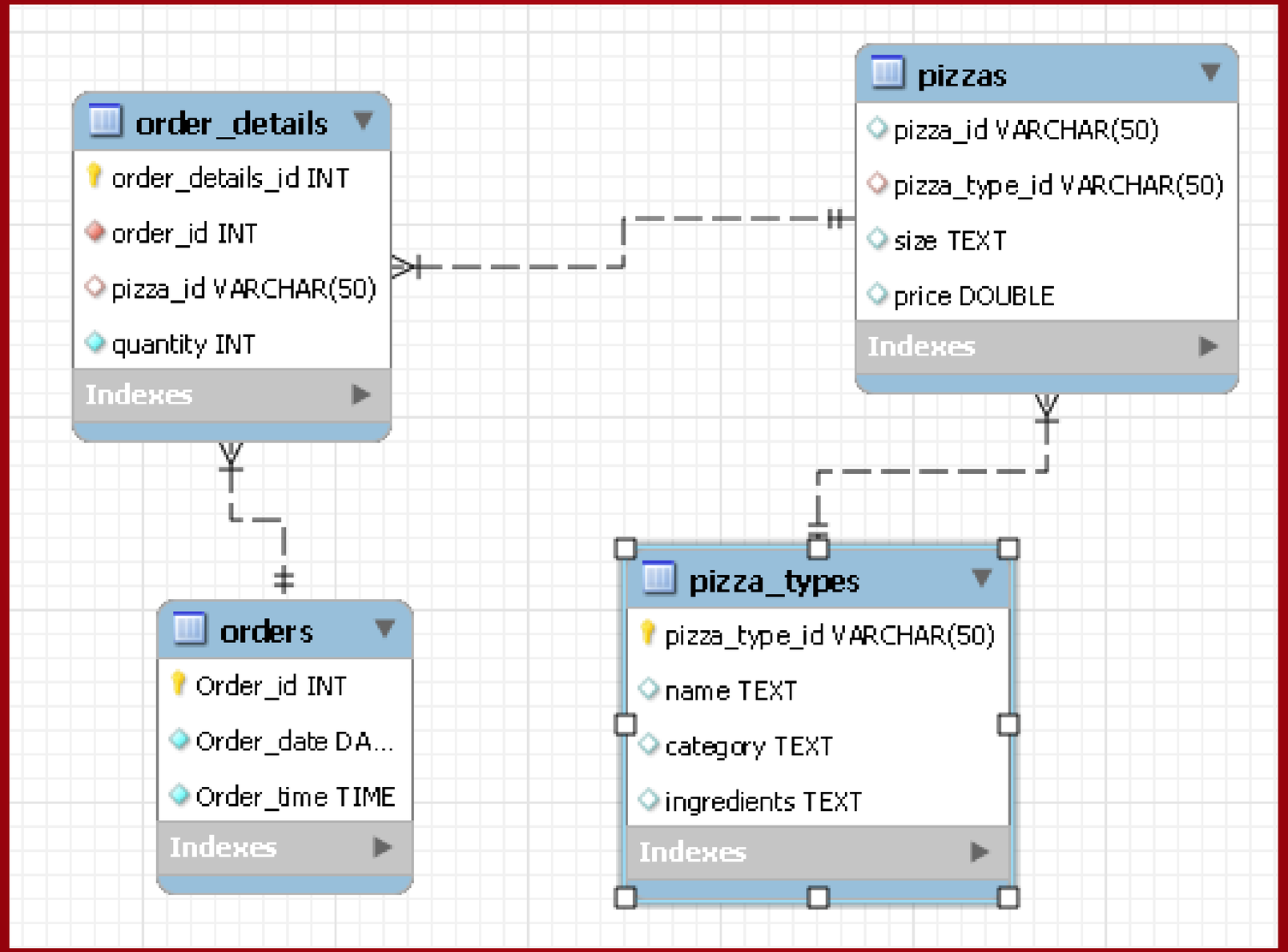
The main objective was to extract actionable business insights using SQL queries, categorized into three levels:

- Basic Analysis: Understanding order volume, revenue, and popular pizzas.
- Intermediate Analysis: Exploring time-based trends, category performance, and daily averages.
- Advanced Analysis: Calculating revenue contribution by category, cumulative revenue, and top-performing pizzas using advanced SQL techniques like window functions and ranking.

This project helped me strengthen my knowledge of:

- Joins
- Aggregations
- Grouping & Filtering
- Window Functions
- Subqueries & CTEs

EER Diagram of Pizza Sales Database Schema



Q1: RETRIEVE THE TOTAL NUMBERS OF ORDERS PLACED.

QUERY:

```
SELECT
    COUNT(Order_id) AS Total_orders
FROM
    orders;
```

RESULT:

	Total_orders
▶	21350

Q2: CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

QUERY:

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

RESULT:

	Total_sales
▶	817860.05

Q3: IDENTIFY THE HIGHEST-PRICED PIZZA.

QUERY:

```
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:

	name	price
▶	The Greek Pizza	35.95

Q4: IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

QUERY:

```
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:

	size	Order_count
▶	L	18526
	M	15385
	S	14137

Q5: LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.

QUERY:

```
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:

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

Q8: JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

QUERY:

```
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS Total_quantity_ordered
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;
```

RESULT:

	category	Total_quantity_ordered
▶	Classic	14888
	Veggie	11649
	Supreme	11987
	Chicken	11050

Q7: DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

QUERY:

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

RESULT:

hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399

Q8: JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

QUERY:

```
SELECT
    category, COUNT(name) AS Types
FROM
    pizza_types
GROUP BY category;
```

RESULT:

	category	Types
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

Q9:GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

QUERY:

```
SELECT
    ROUND(AVG(quantity), 0) AS Order_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:

	Avg_order_per_day
▶	138

Q10:DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

QUERY:

```
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 pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```

RESULT:

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

Q11:CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.

QUERY:

```
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:

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

Q12:ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

QUERY:

```
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:

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

Q13:DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY.

QUERY:

```
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:

name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The Classic Deluxe Pizza	38180.5
The Hawaiian Pizza	32273.25
The Spicy Italian Pizza	34831.25
The Italian Supreme Pizza	33476.75
The Four Cheese Pizza	32265.700000000
The Mexicana Pizza	26780.75

THANK YOU!