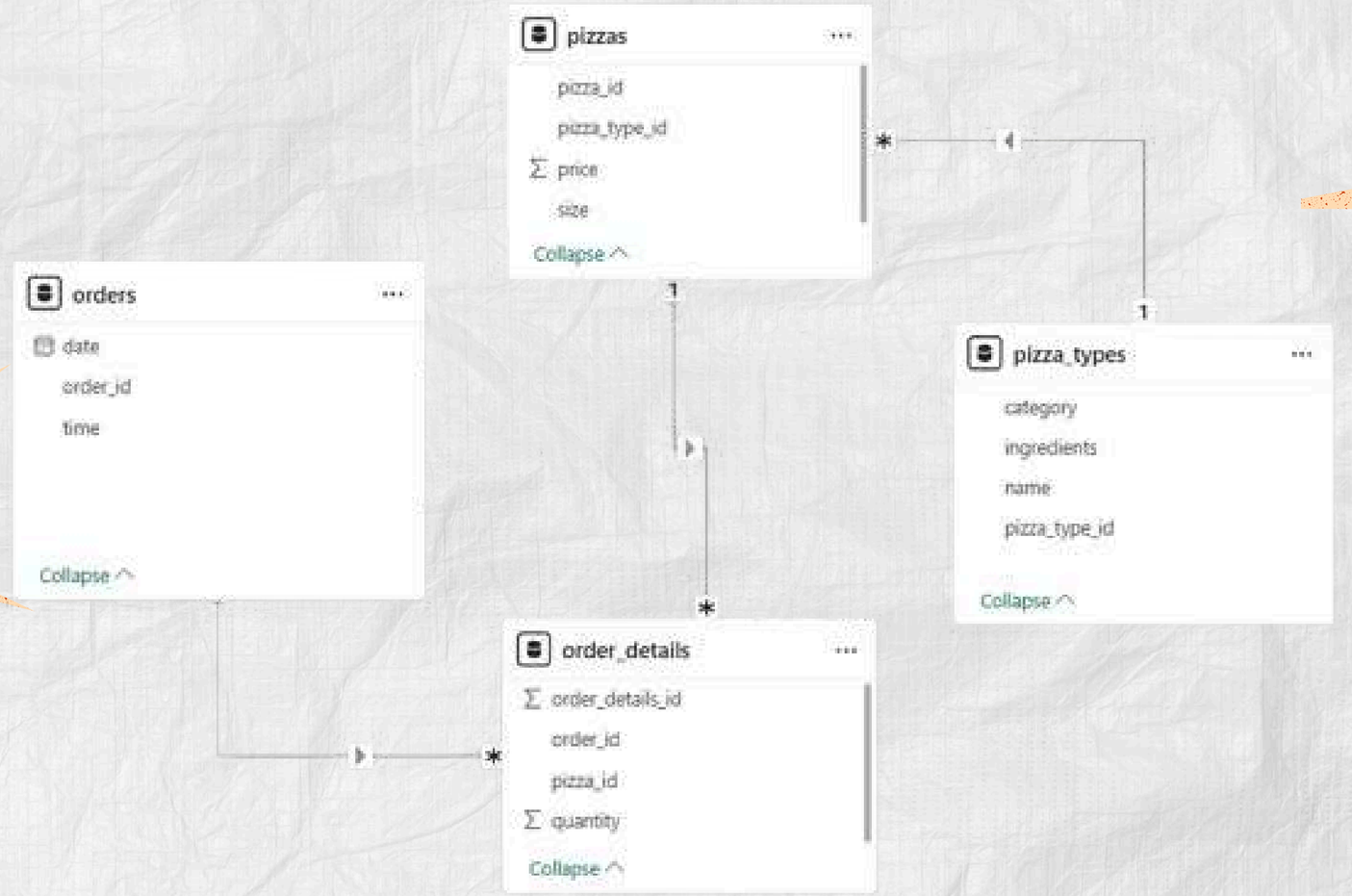
SLICE OF DATA: PIZZA SALES ANALYSIS WITH MYSQL

NEXT >

ABOUT US

My name is Sahil sawal. In this project, I have utilized SQL queries to analyze and answer business-related questions based on pizza sales data. By applying various SQL techniques such as filtering, aggregation, joins, and grouping, I was able to uncover insights into customer preferences, and overall business performance. This analysis helps in understanding which pizzas are most popular, peak ordering times, and key factors that drive sales.

DATABASE SCHEMA

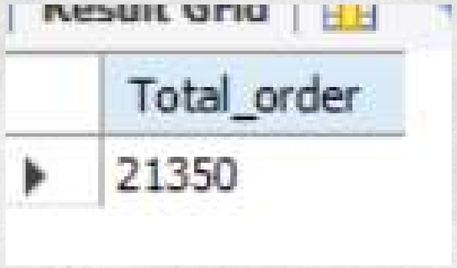


< BACK

NEXT >

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

```
select count(order_id) as Total_order from orders;
```



CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```sql
SELECT
    ROUND(SUM(o.quantity * p.price), 2) AS Total_sales
FROM
    order_detail AS o
        JOIN
    pizzas AS p ON o.Pizza_id = p.pizza_id;
```

	Total_sales
▶	817860.05

< BACK

NEXT >

IDENTIFY THE HIGHEST-PRICED PIZZA.

```sql
SELECT
    p.name, o.price
FROM
    pizza_types AS p
        JOIN
    pizzas AS o ON p.pizza_type_id = o.pizza_type_id
ORDER BY o.price DESC
LIMIT 5;
```

name	price
The Greek Pizza	35.95
The Greek Pizza	25.5
The Brie Carre Pizza	23.65
The Italian Vegetables Pizza	21
The Barbecue Chicken Pizza	20.75

< BACK

NEXT >

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

```sql
SELECT
    p.size,
    sum(o.Quantity)
FROM
    pizzas AS p
        JOIN
    order_detail AS o ON p.pizza_id = o.Pizza_id
GROUP BY p.size
ORDER BY sum(o.Quantity) DESC;
```

size	sum(o.Quantity)
L	18956
M	15635
S	14403
XL	552
XXL	28

< BACK

NEXT >

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

```sql
SELECT
    p.name, SUM(o.quantity) AS Quantity
FROM
    pizza_types AS p
        JOIN
    pizzas AS q ON p.pizza_type_id = q.pizza_type_id
        JOIN
    order_detail AS o ON o.Pizza_id = q.pizza_id
GROUP BY p.name
ORDER BY SUM(o.quantity) DESC
LIMIT 5;
```

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

< BACK

NEXT >

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

```
SELECT
    p.category, SUM(o.quantity)
FROM
    pizza_types AS p
        JOIN
    pizzas AS q ON p.pizza_type_id = q.pizza_type_id
        JOIN
    order_detail AS o ON o.Pizza_id = q.pizza_id
GROUP BY p.category
ORDER BY SUM(o.quantity) DESC;
```

	category	SUM(o.quantity)
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

< BACK

NEXT >

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

```
SELECT
    HOUR(Order_time), COUNT(Order_id)
FROM
    orders
GROUP BY HOUR(Order_time);
```

HOUR(Order_time)	COUNT(Order_id)
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

< BACK

NEXT >

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

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

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

< BACK

NEXT >

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

```sql
SELECT
    ROUND(AVG(quant), 0)
FROM
    (SELECT
        o.Order_date, SUM(q.quantity) AS quant
    FROM
        orders AS o
    JOIN order_detail AS q ON o.Order_id = q.Order_id
    GROUP BY o.Order_date) AS ordered_quantity;
```

ROUND(AVG(quant), 0)
138

< BACK

NEXT >

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

```sql
SELECT
    p.name, SUM(o.quantity * q.price) AS revenue
FROM
    pizza_types AS p
        JOIN
    pizzas AS q ON p.pizza_type_id = q.pizza_type_id
        JOIN
    order_detail AS o ON o.Pizza_id = q.pizza_id
GROUP BY p.name
ORDER BY revenue DESC
LIMIT 3;
```

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

< BACK

NEXT >

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

```sql
SELECT
    p.category,
    round((SUM(o.Quantity * q.price) / (SELECT
            ROUND(SUM(o.quantity * q.price), 2) AS Total_sales
        FROM
            order_detail AS o
                JOIN
            pizzas AS q ON o.Pizza_id = q.pizza_id)) * 100, 2) AS revenue
FROM
    pizza_types AS p
        JOIN
    pizzas AS q ON p.pizza_type_id = q.pizza_type_id
        JOIN
    order_detail AS o ON o.Pizza_id = q.pizza_id
GROUP BY p.category
ORDER BY revenue DESC;
```

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

< BACK

NEXT >

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
select Order_date,
sum(revenue) over(order by Order_date) as cum_revenue
 from
(select o.Order_date,
sum(q.quantity * p.price) as revenue
from order_detail as q
join pizzas as p
on q.Pizza_id = p.Pizza_id
join orders as o
on o.Order_id = q.Order_id
group by o.Order_date) as sales;
```

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
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

< BACK

NEXT >

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

```sql
select category,name,  revenue from
(select category, name, revenue,
rank() over(partition by category order by revenue) as Ranks
from
(select p.category, p.name,
sum(o.quantity * q.price) as revenue
from pizza_types as p
join pizzas as q
on p.pizza_type_id = q.pizza_type_id
join order_detail as o
on o.Pizza_id = q.pizza_id
group by p.category, p.name) as a) as b
where Ranks <= 3;
```

category	name	revenue
Chicken	The Chicken Pesto Pizza	16701.75
Chicken	The Chicken Alfredo Pizza	16900.25
Chicken	The Southwest Chicken Pizza	34705.75
Classic	The Pepperoni, Mushroom, and Peppers Pizza	18834.5
Classic	The Big Meat Pizza	22968
Classic	The Napolitana Pizza	24087
Supreme	The Brie Carre Pizza	11588.4999999999
Supreme	The Spinach Supreme Pizza	15277.75
Supreme	The Calabrese Pizza	15934.25
Veggie	The Green Garden Pizza	13955.75
Veggie	The Mediterranean Pizza	15360.5
Veggie	The Spinach Pesto Pizza	15596

< BACK

NEXT >

CONCLUSION

IN CONCLUSION, THIS PROJECT DEMONSTRATED HOW SQL CAN BE EFFECTIVELY USED TO ANALYZE PIZZA SALES DATA AND EXTRACT MEANINGFUL BUSINESS INSIGHTS. BY WRITING AND EXECUTING QUERIES, I WAS ABLE TO IDENTIFY TOP-SELLING PIZZAS, POPULAR ORDER TIMES, AND OVERALL SALES PATTERNS. THESE FINDINGS CAN HELP BUSINESSES IMPROVE DECISION-MAKING, OPTIMIZE INVENTORY, AND ENHANCE CUSTOMER SATISFACTION.

< BACK

THANK YOU!

< BACK