LARANA PIZZA

LARANA PIZZA

LARANA PIZZA

WELCOME TO LARANA PIZZA

I'm Deepa Nayak, and in this project, I have done comprehensive analysis of pizza sales data using SQL to uncover trends and customer preferences.

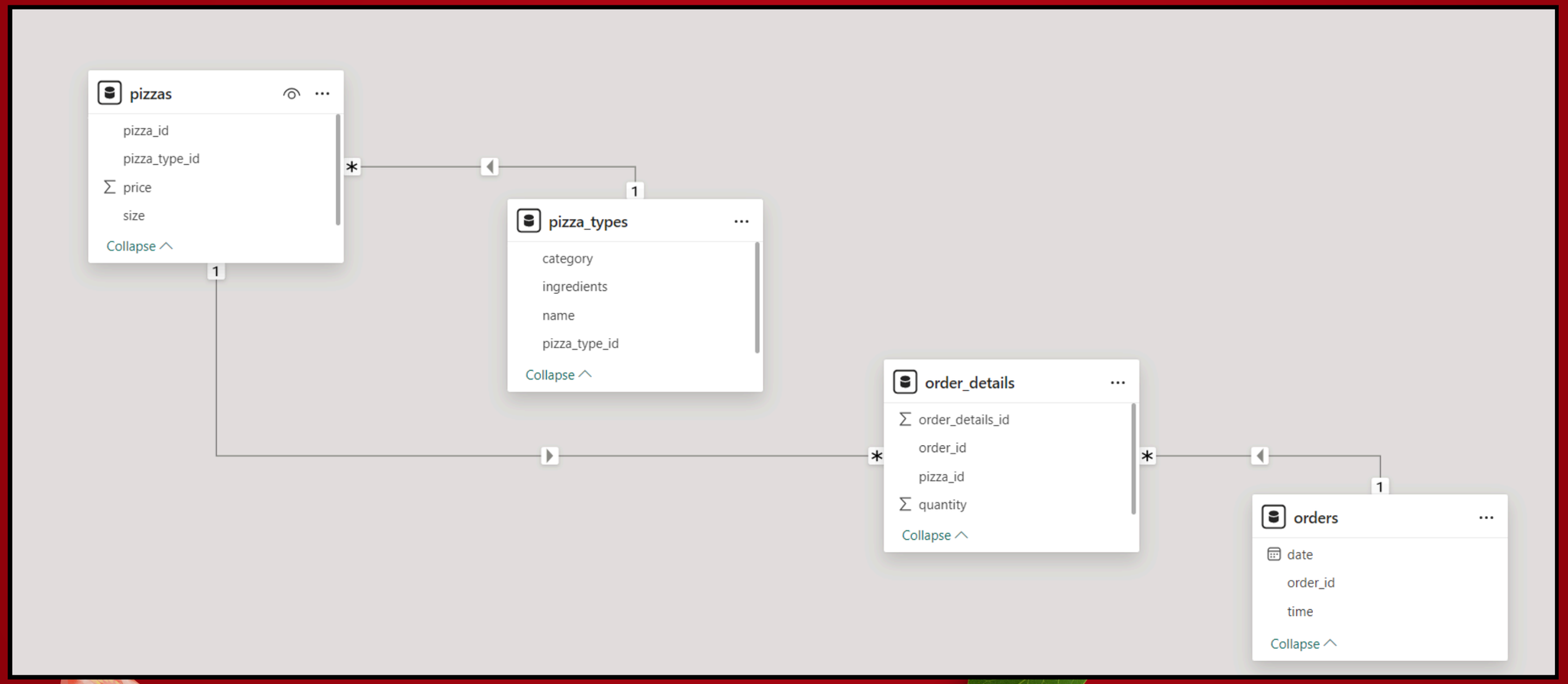
LARANA PIZZA

ANALYTICAL FOCUS

The goal was to identify top-selling pizzas, peak sales times, and actionable insights for optimizing business strategies. SQL queries were used to analyze data across multiple tables, helping inform marketing and decision-making.

LARANA PIZZA

ENTITY-RELATIONSHIP DIAGRAM



LARANA PIZZA

ESSENTIAL SQL QUERIES

Retrieve the total number of orders placed.

```
Select
      count (order_id)
            as [Total_Orders]
                  from orders;
```

Results | Messages

	Total_Orders
1	21350

LARANA PIZZA

Calculate the total revenue generated from pizza sales.

```
from
    order_details
join
    pizzas
on
    pizzas.pizza_id=order_details.pizza_id;
```

Results | Messages

	Total_Sales
1	817860.05

LARANA PIZZA

Identify the highest-priced pizza.

```sql
SELECT TOP 1
    pizza_types.name,
    pizzas.price
FROM
    pizza_types
JOIN
    pizzas
ON
    pizzas.pizza_type_id = pizza_types.pizza_type_id
ORDER BY
    pizzas.price DESC;
```

Results | Messages

	name	price
1	The Greek Pizza	35.9500007629395

LARANA PIZZA

Identify the most common pizza size ordered.

```sql
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;
```

Results | Messages

	size	Order_Count
1	L	18526
2	M	15385
3	S	14137
4	XL	544
5	XXL	28

LARANA PIZZA

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

```sql
SELECT TOP 5
    pizza_types.name,
    SUM(order_details.quantity) AS [Qty]
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
    [Qty] DESC;
```

Results | Messages

	name	Qty
1	The Classic Deluxe Pizza	2453
2	The Barbecue Chicken Pizza	2432
3	The Hawaiian Pizza	2422
4	The Pepperoni Pizza	2418
5	The Thai Chicken Pizza	2371

LARANA PIZZA

ANALYTICAL SQL QUERIES

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;
```

Results | Messages

	category	quantity
1	Classic	14888
2	Supreme	11987
3	Veggie	11649
4	Chicken	11050

LARANA PIZZA

Determine the distribution of orders by hour of the day.

```sql
SELECT
    DATEPART(HOUR, time) AS order_hour,
    COUNT(order_id) AS order_count
FROM
    orders
GROUP BY
    DATEPART(HOUR, time)
ORDER BY
    order_hour ASC;
```

Results | Messages

	order_hour	order_count
1	9	1
2	10	8
3	11	1231
4	12	2520
5	13	2455
6	14	1472
7	15	1468
8	16	1920
9	17	2336
10	18	2399
11	19	2009
12	20	1642
13	21	1198
14	22	663
15	23	28

LARANA PIZZA

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

```
Select category,
    count (name) as [pizza_count]
from
    pizza_types
group by
    category;
```

Results | Messages

	category	pizza_count
1	Chicken	6
2	Classic	8
3	Supreme	9
4	Veggie	9

LARANA PIZZA

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

```
Select avg (quantity) as [Avg_Qty]
from
(Select
    orders.date, sum (order_details.quantity) as [quantity]
from
    order_details
Join
    orders
on
    order_details.order_id=orders.order_id
group by
    orders.date) as [order_quantity];
```

Results | Messages

	Avg_Qty
1	138

LARANA PIZZA

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

```sql
SELECT TOP 3
    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;
```

Results | Messages

	name	Revenue
1	The Thai Chicken Pizza	43434.25
2	The Barbecue Chicken Pizza	42768
3	The California Chicken Pizza	41409.5

LARANA PIZZA

COMPLEX SQL QUERIES

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

```
Select
    pizza_types.category,
    (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 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;
```

Results | Messages

	category	Revenue
1	Classic	26.905960258281 6
2	Supreme	25.456311237244 1
3	Chicken	23.955137556847 3
4	Veggie	23.682591050147 1

LARANA PIZZA

Analyze the cumulative revenue generated over time.

```
Select
    date,
Sum (Revenue) over (order by date) as [cumulative_revenue]
from (Select
    orders.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.date) as [Sales];
```

Results | Messages

	date	cumulative_revenue
1	2015-01-01	2713.85000228882
2	2015-01-02	5445.7500038147
3	2015-01-03	8108.15000724792
4	2015-01-04	9863.60000801086
5	2015-01-05	11929.5500087738
6	2015-01-06	14358.5000114441
7	2015-01-07	16560.700012207
8	2015-01-08	19399.0500183105
9	2015-01-09	21526.4000225067
10	2015-01-10	23990.350025177
11	2015-01-11	25862.6500263214
12	2015-01-12	27781.7000274658
13	2015-01-13	29831.3000278473
14	2015-01-14	32358.7000293732
15	2015-01-15	34343.5000324249
16	2015-01-16	36937.6500339508

LARANA PIZZA

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 [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;
```

Results | Messages

	name	revenue
1	The Thai Chicken Pizza	43434.25
2	The Barbecue Chicken Pizza	42768
3	The California Chicken Pizza	41409.5
4	The Classic Deluxe Pizza	38180.5
5	The Hawaiian Pizza	32273.25
6	The Pepperoni Pizza	30161.75
7	The Spicy Italian Pizza	34831.25
8	The Italian Supreme Pizza	33476.75
9	The Sicilian Pizza	30940.5
10	The Four Cheese Pizza	32265.7010040283
11	The Mexicana Pizza	26780.75
12	The Five Cheese Pizza	26066.5

CONCLUSION AND FUTURE SCOPE

This project provided key insights into pizza sales trends, customer preferences, and peak sales periods using SQL-based analysis. The findings can help optimize business strategies and improve decision-making. Future enhancements could include customer segmentation, predictive analytics, and integration with visualization tools like Power BI for deeper insights.

LARANA PIZZA

FREE DELIVERY SERVICE

Thank you for your time!
Feel free to explore the project on GitHub!

LARANA PIZZA

OUR CONTACT

https://github.com/Deepanayaktangodu

www.linkedin.com/in/deepa-nayak-a825a7175

deepanayak109@gmail.com

LARANA PIZZA

THANK YOU!