PIZZA SALES ANALYSIS

USING SQL

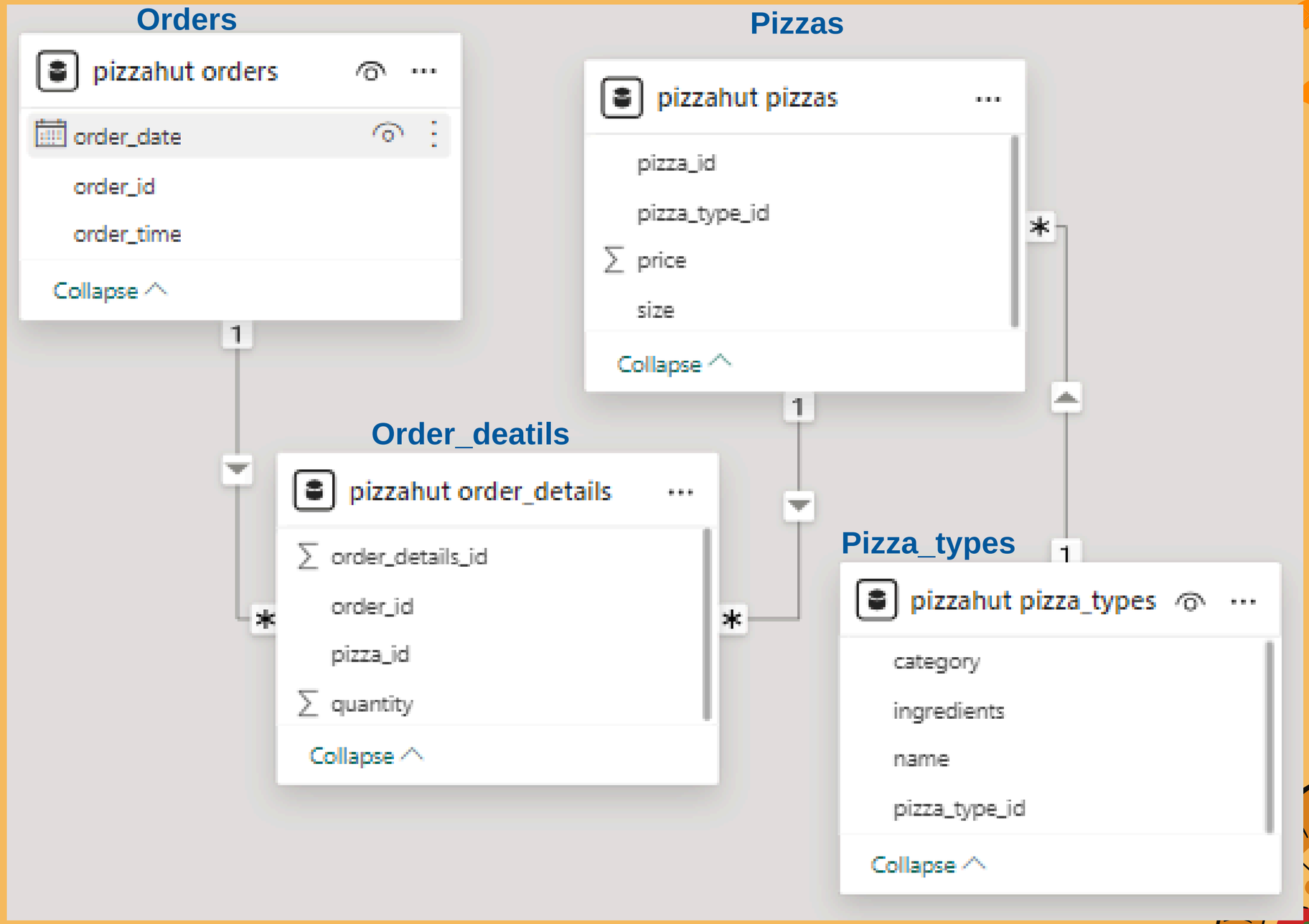
INTRODUCTION

Hello everyone, I'm Vishal Sharma, an aspiring data analyst. In this project, I've conducted a comprehensive pizza sales analysis using SQL queries to tackle key business challenges.Through this analysis, I aim to uncover insights that can inform business decisions and drive growth in the pizza industry."

Hello!

Welcome to my Pizza Sales Analysis project. Through SQL queries, joins, ranks, and functions, I've explored pizza sales data to uncover insights into consumption patterns, most common pizza types, peak sales time, revenue details and much more. Join me as we slice through the data to reveal the secrets of pizza sales!

Mapping Pizza Sales Insights



Retrieve the total number of orders placed.

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

	total_orders
▶	21350

Calculate the total revenue generated from pizza sales.

```
3    SELECT
4        ROUND(SUM(order_details.quantity * pizzas.price),
5                2) AS total_sales
6    FROM
7        order_details
8            JOIN
9        pizzas USING (pizza_id);
```

	total_sales
▶	817860.05

Identify the highest-priced pizza.

```
 3 •    SELECT
 4          pizza_types.name, pizzas.price
 5      FROM
 6          pizza_types
 7              JOIN
 8          pizzas USING (pizza_type_id)
 9      ORDER BY price DESC
10      LIMIT 1
```

	total_sales
▶	817860.05

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 USING (pizza_id)
GROUP BY pizzas.size
ORDER BY order_count DESC
LIMIT 1
```

	size	order_count
▶	L	18526

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

```sql
SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas USING (pizza_type_id)
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY 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

Join the necessary tables to find the total quantity of each pizza category ordered.

```sql
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
```

	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

Determine the distribution of orders by hour of the day.

```
3 •   SELECT
4         HOUR(order_time), COUNT(order_id) AS order_count
5     FROM
6         orders
7     GROUP BY HOUR(order_time)
8
```

HOUR(order_time)	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

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

```
3 •    select category, count(name) from pizza_types
4      group by category
```

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

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

```sql
SELECT
    ROUND(AVG(quantity), 0) AS average_orders_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
```

	average_orders_per_day
▶	138

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

```
3  •  SELECT
4         pizza_types.name,
5         SUM(order_details.quantity * pizzas.price) AS revenue
6     FROM
7         pizza_types
8             JOIN
9         pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
10            JOIN
11        order_details ON order_details.pizza_id = pizzas.pizza_id
12    GROUP BY pizza_types.name
13    ORDER BY revenue DESC
14    LIMIT 3
```

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

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

```sql
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 order_details.pizza_id = pizzas.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
```

	category	revenue
▶	Classic	26.91
	Veggie	23.68
	Supreme	25.46
	Chicken	23.96

Analyze the cumulative revenue generated over time.

```sql
select order_date, sum(revenue) over(order by order_date) as cum_revenue from
(select orders.order_date, round(sum(order_details.quantity*pizzas.price),2) 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 order_date) as sales
```

order_date	cum_revenue
2015-01-01	2713.85
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

order_date	cum_revenue
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.399999999998
2015-01-10	23990.35
2015-01-11	25862.649999999998
2015-01-12	27781.699999999997

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

```sql
select name,revenue,rankk from
(select category, name, revenue, rank() over(partition by category order by revenue desc) as rankk from

(SELECT
    pizza_types.category,
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas USING (pizza_type_id)
        JOIN
    order_details USING (pizza_id)
GROUP BY category , name) as a) as b
where rankk<=3
```

	name	revenue	rankk
	The Thai Chicken Pizza	43434.25	1
	The Barbecue Chicken Pizza	42768	2
	The California Chicken Pizza	41409.5	3
	The Classic Deluxe Pizza	38180.5	1
	The Hawaiian Pizza	32273.25	2
	The Pepperoni Pizza	30161.75	3

As we wrap up, I want to express my gratitude for your time and attention. This project has been a fascinating exploration of pizza sales insights using SQL.
If you have any questions or would like to delve deeper into the analysis, please don't hesitate to reach out.

Thank you again for being a part of this journey!
Vishal Sharma