PIZZA SALES ANALYSIS USING SQL

CONTACT
9643214191

EMAIL
arorkaran9@gmail.com

Prepared by:
Karan Arora

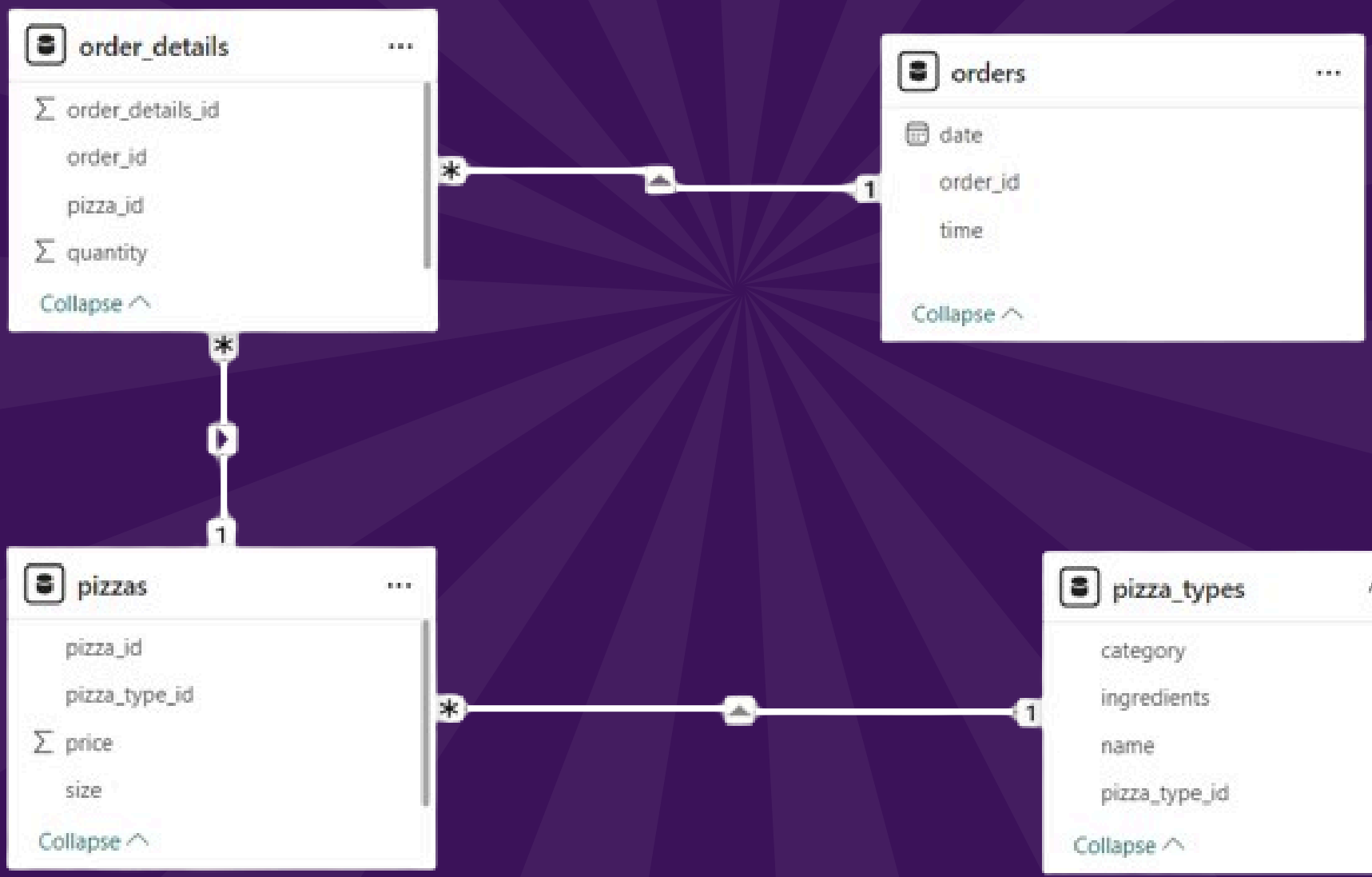
Hello,

I am **Karan Arora**, and I am a **data analyst**. This Pizza Sales Analysis project uses **SQL** to examine pizza sales data, applying Common Table Expressions (CTEs), subqueries, and window functions. The analysis reveals top-selling pizzas, revenue trends, and category-wise sales patterns, providing **valuable insights** to drive business decisions.

PROBLEM STATEMENTS

1. **Retrieve the total number of orders placed.**
2. **Calculate the total revenue generated from pizza sales.**
3. **Identify the highest-priced pizza.**
4. **Identify the most common pizza size ordered.**
5. **List the top 5 most ordered pizza types along with their quantities.**
6. **Join the necessary tables to find the total quantity of each pizza category ordered.**
7. **Determine the distribution of orders by hour of the day.**
8. **Join relevant tables to find the category-wise distribution of pizzas.**
9. **Group the orders by date and calculate the average number of pizzas ordered per day.**
10. **Determine the top 3 most ordered pizza types based on revenue.**
11. **Calculate the percentage contribution of each pizza category type to total revenue.**
12. **Analyze the cumulative revenue generated over time.**
13. **Determine the top 3 most ordered pizza types based on revenue for each pizza category.**

DATA SCHEMA


order_details
order_details_id
order_id
pizza_id
quantity
Collapse
orders
date
order_id
time
Collapse
pizzas
pizza_id
pizza_type_id
price
size
Collapse
pizza_types
category
ingredients
name
pizza_type_id
Collapse
*
1

Retrieve the total number of orders placed.

```sql
SELECT
    COUNT(DISTINCT order_id) AS Total_Orders
FROM
    orders;
```

Result Grid

	Total_Orders
▶	21350

Calculate the total revenue generated from pizza sales.

```sql
SELECT
    ROUND(SUM(o.quantity * p.price),2) as Total_revenue
FROM
    orders_details o
        JOIN
    pizzas p ON o.pizza_id = p.pizza_id;
```

Result Grid

	Total_revenue
▶	817860.05

Identify the highest-priced pizza.

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

Result Grid | Filter Rows:

name	price
The Greek Pizza	35.95

Identify the most common pizza size ordered.

```sql
SELECT
    p.size, SUM(o.quantity) AS Total_orders
FROM
    pizzas p
        JOIN
    orders_details o ON p.pizza_id = o.pizza_id
GROUP BY p.size
ORDER BY COUNT(p.size) DESC
LIMIT 1;
```

Result Grid | Filter Rows:

	size	Total_orders
▶	L	18956

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

```sql
SELECT
    pt.name, SUM(o.quantity) AS total_quantity
FROM
    pizza_types pt
        JOIN
    pizzas p ON pt.pizza_type_id = p.pizza_type_id
        JOIN
    orders_details o ON p.pizza_id = o.pizza_id
GROUP BY pt.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

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

```sql
SELECT
    pt.category, SUM(o.quantity) AS total_quantity
FROM
    pizza_types pt
        JOIN
    pizzas p ON pt.pizza_type_id = p.pizza_type_id
        JOIN
    orders_details o ON p.pizza_id = o.pizza_id
GROUP BY pt.category
order by total_quantity desc
;
```

Result Grid | Filter Rows:

category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hour of the day.

```sql
SELECT
    HOUR(order_time) AS order_hour, count(order_id)
FROM
    orders
GROUP BY order_hour
ORDER BY order_hour;
```

Result Grid | Filter Rows:

order_hour	count(order_id)
9	1
10	8
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

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

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

Result Grid | Filter Rows:

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

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

```sql
WITH daily_totals AS (
    SELECT
        o.order_date AS order_date,
        SUM(od.quantity) AS total_quantity
    FROM
        orders o
    JOIN
        orders_details od ON o.order_id = od.order_id
    GROUP BY
        o.order_date
)
SELECT
    ROUND(AVG(total_quantity), 2) AS average_order_per_day
FROM
    daily_totals;
```

Result Grid | Filter Rows:

	average_order_per_day
▶	138.47

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

```sql
SELECT
    pt.name, SUM(od.quantity * p.price) AS revenue
FROM
    pizzas p
        JOIN
    orders_details od ON p.pizza_id = od.pizza_id
        JOIN
    pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.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

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

```sql
SELECT
    pt.category,
    round((SUM(od.quantity * p.price) / (SELECT
            SUM(od.quantity * p.price)
        FROM
            orders_details od
                JOIN
            pizzas p ON od.pizza_id = p.pizza_id) * 100),2) AS percentage_contribution
FROM
    pizza_types pt
        JOIN
    pizzas p ON pt.pizza_type_id = p.pizza_type_id
        JOIN
    orders_details od ON od.pizza_id = p.pizza_id group by pt.category order by percentage_contribution desc;
```

Result Grid | Filter Rows:

category	percentage_contribution
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Analyze the cumulative revenue generated over time.

```
1
2 •  select order_date,sum(revenue) over(order by order_date) as cum_revenue from
3    (select o.order_date,sum(od.quantity*p.price) as revenue from
4    orders_details od join pizzas p on od.pizza_id=p.pizza_id join
5    orders o on od.order_id=o.order_id group by o.order_date) as sales;
```

Result Grid | Filter Rows:

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

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

```sql
WITH category_sales AS (SELECT pt.category AS pizza_category,pt.name,SUM(od.quantity * p.price) AS revenue
    FROM pizza_types pt JOIN pizzas p ON pt.pizza_type_id = p.pizza_type_id
JOIN orders_details od ON p.pizza_id = od.pizza_id
    GROUP BY pt.category, pt.name
    ORDER BY pt.category ),
ranking_pizza AS (
    SELECT pizza_category,name, RANK() OVER (PARTITION BY pizza_category ORDER BY revenue DESC) AS rnk
    FROM category_sales)
SELECT pizza_category,name,rnk
FROM ranking_pizza
WHERE rnk < 4
ORDER BY pizza_category, rnk;
```

Result Grid | Filter Rows: | Export:

rnk	pizza_category	name
1	Chicken	The Thai Chicken Pizza
2	Chicken	The Barbecue Chicken Pizza
3	Chicken	The California Chicken Pizza
1	Classic	The Classic Deluxe Pizza
2	Classic	The Hawaiian Pizza
3	Classic	The Pepperoni Pizza
1	Supreme	The Spicy Italian Pizza
2	Supreme	The Italian Supreme Pizza
3	Supreme	The Sicilian Pizza
1	Veggie	The Four Cheese Pizza
2	Veggie	The Mexicana Pizza
3	Veggie	The Five Cheese Pizza