PIZZA SALE

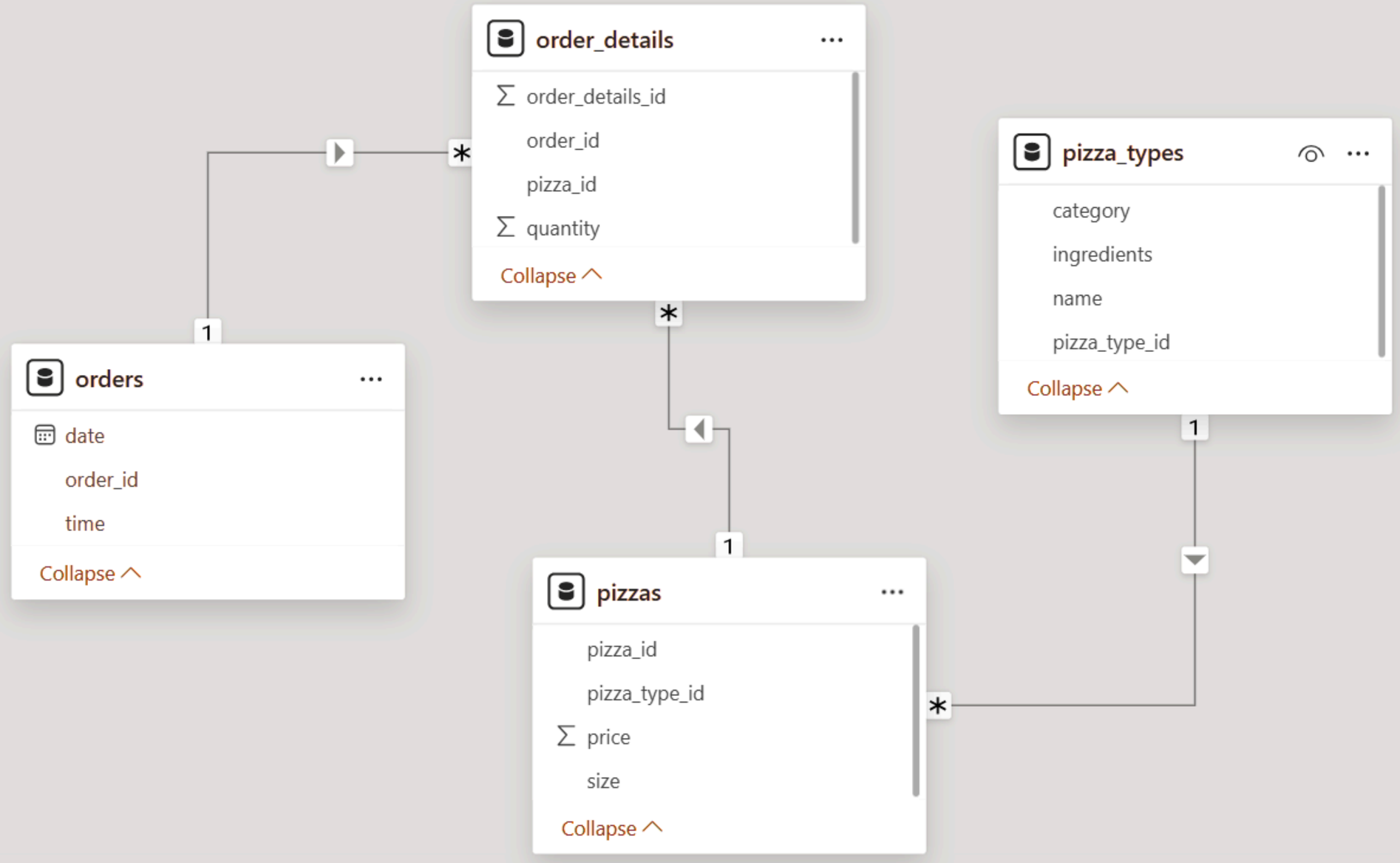
Introduction

Hey , My name is Priyanshi and in this project I have utilized Sql queries to solve question that are related to pizza sale

Tool :- PostgreSQL

QUESTIONS :-

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 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.**

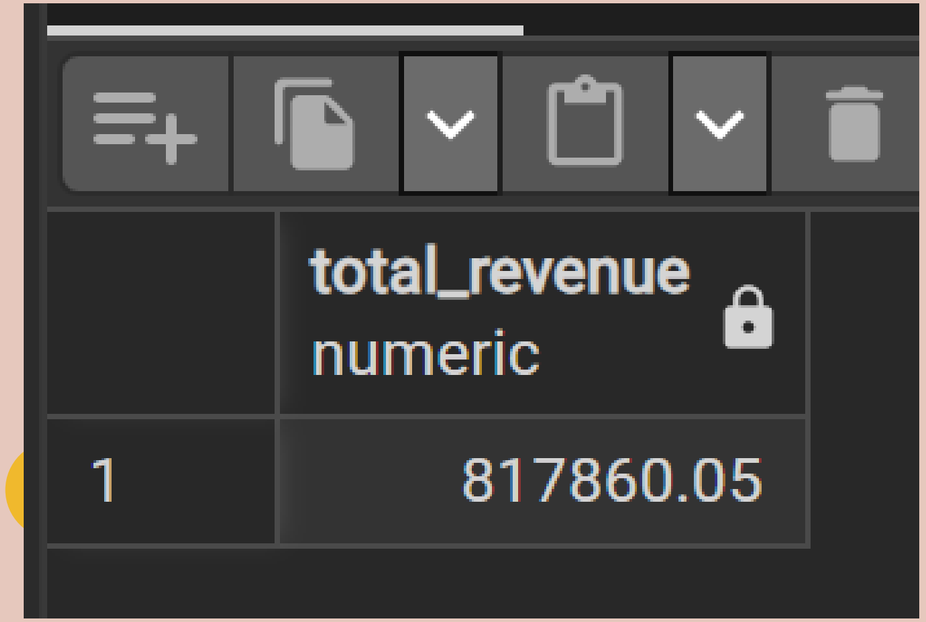

order_details
order_details_id
order_id
pizza_id
quantity
Collapse
orders
date
order_id
time
Collapse
pizza_types
category
ingredients
name
pizza_type_id
Collapse
pizzas
pizza_id
pizza_type_id
price
size
Collapse
1
*
1
*
1
*

```sql
-- Retrive the total number of orders placed

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

	total_orders bigint
1	21350

```sql
-- Calculate the total revenue generated from pizza sales.
select round(sum(order_details.quantity * pizzas.price),2) as Total_Revenue
from order_details join pizzas
on pizzas.pizza_id = order_details.pizza_id
```



	total_revenue numeric
1	817860.05

```sql
-- Identify the highest-priced pizza.
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;
```

	name text	price numeric (5,2)
	The Greek Pizza	35.95

```sql
-- Identify the most common pizza size ordered.

select pizzas.size, count(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 limit 1;
```

	size text	order_count bigint
1	L	18526

```sql
-- List the top 5 most ordered pizza types along with their quantities.
select pizza_types.name, 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.name order by quantity desc limit 5;
```

	name text	quantity bigint
1	The Classic Deluxe Pizza	2453
2	The Barbecue Chicken P...	2432
3	The Hawaiian Pizza	2422
4	The Pepperoni Pizza	2418
5	The Thai Chicken Pizza	2371

```sql
-- 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;
```

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

```sql
-- Determine the distribution of orders by hour of the day

select extract(hour from time) as hour , count(order_id) as order_count
from orders
group by hour
order by order_count desc;
```

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

```sql
-- Join relevant tables to find the category-wise
-- distribution of pizzas.
select category, count(name) from pizza_types
group by category;
```

	category text	count bigint
1	Supreme	9
2	Chicken	6
3	Classic	8
4	Veggie	9

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

select round(avg(quantity),0) as Avg_pizzas_ordered_per_day
from (select orders.date ,
sum(order_details.quantity) as quantity
from orders join order_details
on orders.order_id = order_details.order_id
group by orders.date) as order_quantity;
```

	avg_pizzas_ordered_per_day numeric
1	138

```sql
-- Determine the top 3 most ordered pizza types based on revenue.
select 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 limit 3;
```

	name text	revenue numeric
1	The Thai Chicken Pizza	43434.25
2	The Barbecue Chicken Pizza	42768.00
3	The California Chicken Pizza	41409.50

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

with PizzaRevenue AS (select pizza_types.category,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.category),
TotalRevenue as (select sum(revenue) as Total_Revenue
from PizzaRevenue)

select
p.category,round((p.revenue/t.Total_Revenue)*100,2) as percentage_contribution
from PizzaRevenue p, TotalRevenue t
order by percentage_contribution desc;
```

	category text	percentage_contribution numeric
1	Classic	26.91
2	Supreme	25.46
3	Chicken	23.96
4	Veggie	23.68

```sql
-- Analyze the cumulative revenue generated over time.
CREATE VIEW total_revenue_view AS
SELECT
    pizza_types.category AS pizza_category,
    pizza_types.name AS pizza_type,
    orders.date AS order_date,
    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
JOIN orders
    ON orders.order_id = order_details.order_id
GROUP BY pizza_types.category, pizza_types.name, orders.date;


   SELECT
    order_date,
    SUM(daily_revenue) OVER (ORDER BY order_date) AS cumulative_revenue
FROM (
    SELECT
        order_date,
        SUM(revenue) AS daily_revenue
    FROM total_revenue_view
    GROUP BY order_date
) AS daily_totals
ORDER BY order_date;
```

	order_date date	cumulative_revenue numeric
1	2015-01-01	2713.85
2	2015-01-02	5445.75
3	2015-01-03	8108.15
4	2015-01-04	9863.60
5	2015-01-05	11929.55
6	2015-01-06	14358.50
7	2015-01-07	16560.70
8	2015-01-08	19399.05
9	2015-01-09	21526.40
10	2015-01-10	23990.35
11	2015-01-11	25862.65
12	2015-01-12	27781.70
13	2015-01-13	29831.30
14	2015-01-14	32358.70
15	2015-01-15	34343.50

349	2015-12-21	801288.65
350	2015-12-22	803171.60
351	2015-12-23	805415.90
352	2015-12-24	807553.75
353	2015-12-26	809196.80
354	2015-12-27	810615.80
355	2015-12-28	812253.00
356	2015-12-29	813606.25
357	2015-12-30	814944.05
358	2015-12-31	817860.05

```sql
-- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
WITH ranked_pizzas AS (
    SELECT
        pizza_category,
        pizza_type,
        SUM(revenue) AS total_revenue,
        RANK() OVER (PARTITION BY pizza_category ORDER BY SUM(revenue) DESC) AS rank
    FROM total_revenue_view
    GROUP BY pizza_category ,pizza_type
)
SELECT *
FROM ranked_pizzas
WHERE rank <= 3;
```

	pizza_category text	pizza_type text	total_revenue numeric	rank bigint
	Chicken	The Thai Chicken Pizza	43434.25	1
	Chicken	The Barbecue Chicken Piz…	42768.00	2
	Chicken	The California Chicken Piz…	41409.50	3
	Classic	The Classic Deluxe Pizza	38180.50	1
	Classic	The Hawaiian Pizza	32273.25	2
	Classic	The Pepperoni Pizza	30161.75	3
	Supreme	The Spicy Italian Pizza	34831.25	1
	Supreme	The Italian Supreme Pizza	33476.75	2
	Supreme	The Sicilian Pizza	30940.50	3
0	Veggie	The Four Cheese Pizza	32265.70	1
1	Veggie	The Mexicana Pizza	26780.75	2
2	Veggie	The Five Cheese Pizza	26066.50	3

OUR CONTACT

Gmail Id :-
priyanshi1652001@gmail.com

GitHub link

Linkedin Id

THANK YOU!