Pizza Sales

2025 March 22

Hemang Taori

Hello, I'm Hemang Taori. I have developed this project to analyze pizza sales for a store, providing insights by addressing key questions that assist the café owner in making informed business decisions.

QUESTIONS

Basic:

Retrieve the total number of orders placed.

Calculate the total revenue generated from pizza sales.

Identify the highest-priced pizza.

Identify the most common pizza size ordered.

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

Intermediate:

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

Determine the distribution of orders by hour of the day.

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

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

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

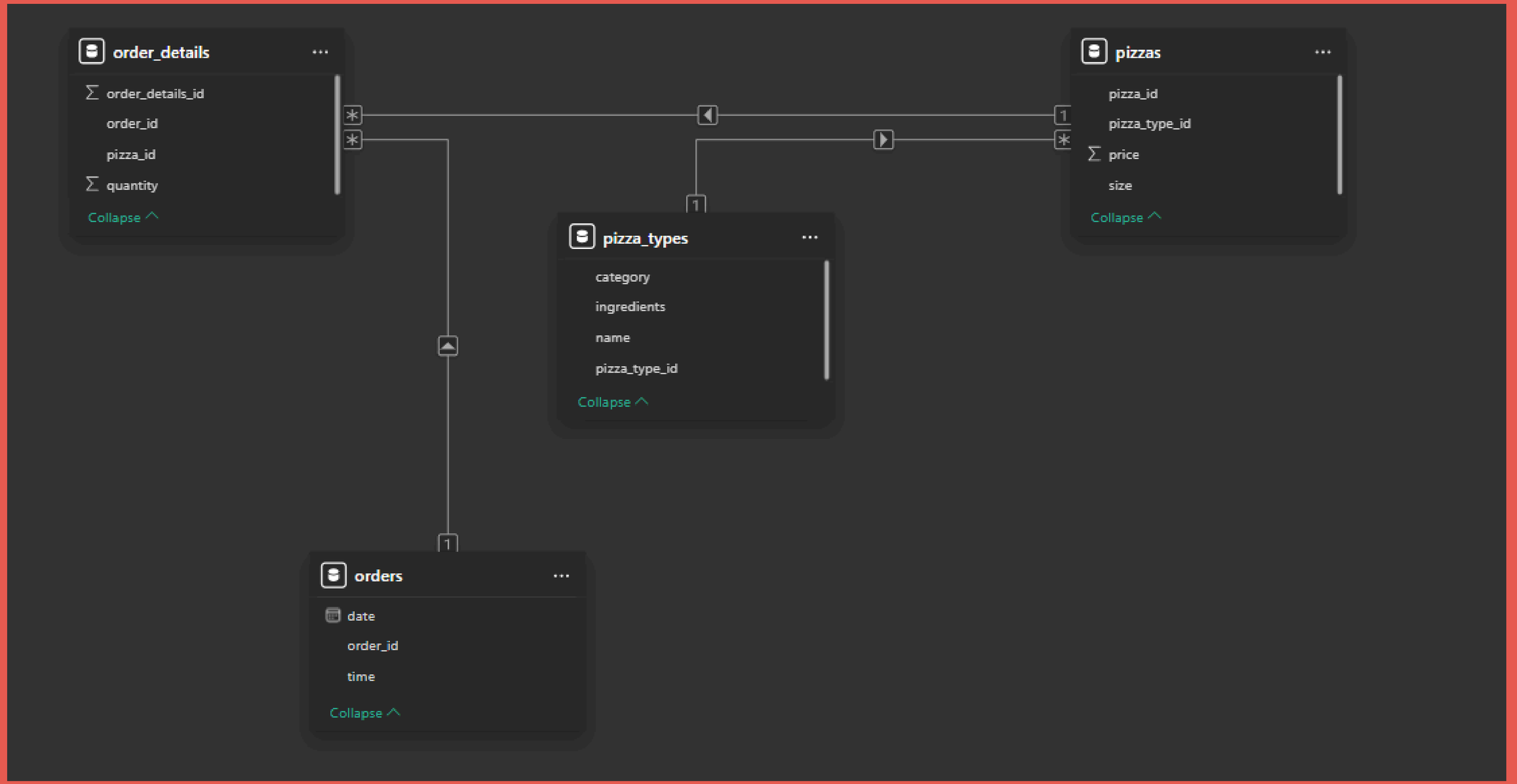
Advanced:

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

Analyze the cumulative revenue generated over time.

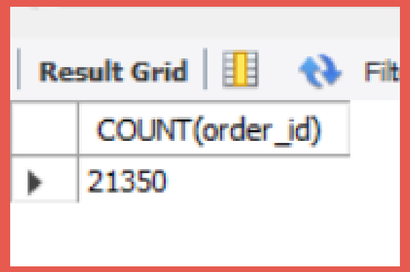
Determine the top 3 most ordered pizza types based on revenue for each pizza category.

SCHEMA


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

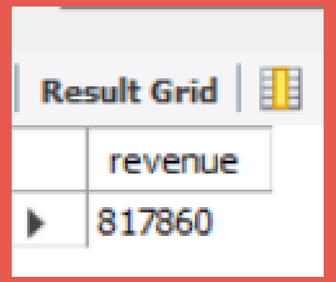
1. RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED

```
SELECT
    COUNT(order_id)
FROM
    orders;
```



2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES

```sql
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price)) AS revenue
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id;
```



3. IDENTIFY THE HIGHEST PRICED PIZZA

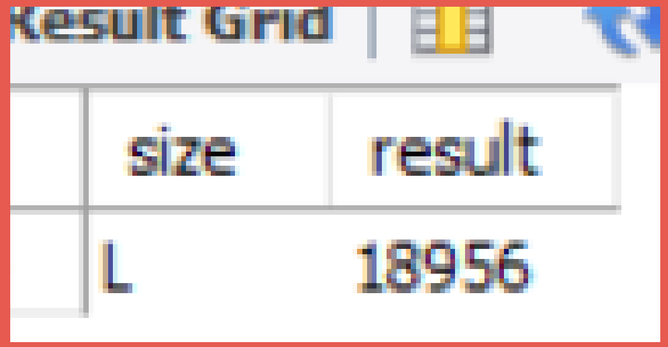
```sql
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

Result Grid | Filter Row

	name	price
▶	The Greek Pizza	35.95

4.IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

```sql
SELECT
    pizzas.size, SUM(order_details.quantity) AS result
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY result DESC
LIMIT 1;
```



5.LIST THE TOP 5 MOST ORDERED PIZZAS ALONG WITH THEIR QUANTITIES

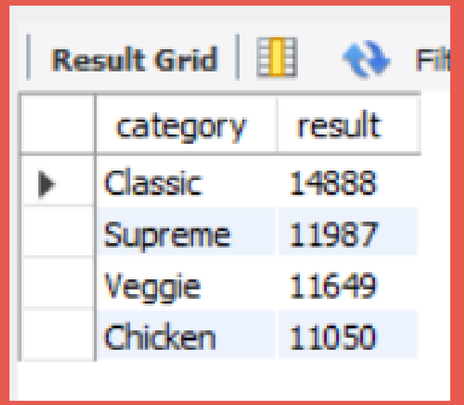
```sql
SELECT
    pizza_types.name, SUM(order_details.quantity) AS result
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY result DESC
LIMIT 5;
```

Result Grid | Filter Rows:

name	result
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

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

```sql
SELECT
    pizza_types.category, SUM(order_details.quantity) AS result
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY result DESC;
```



category	result
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

7. DETERMINE THE DISTRIBUTION OF ORDERS BY THE HOUR OF THE DAY.

```sql
SELECT
    HOUR(orders.order_time) AS hours,
    COUNT(orders.order_id) AS count
FROM
    orders
GROUP BY hours;
```

Result Grid

hours	count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009

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

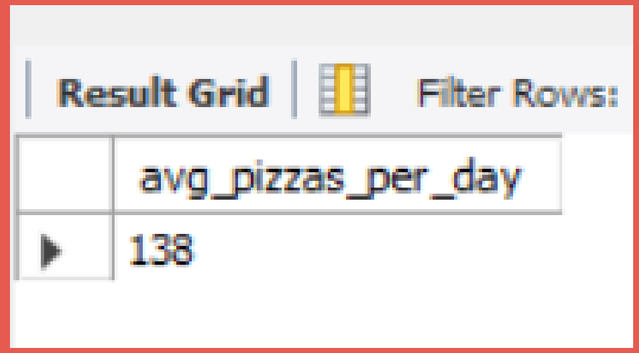
```sql
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category
ORDER BY COUNT(name) DESC;
```

Result Grid | Filter Rows

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

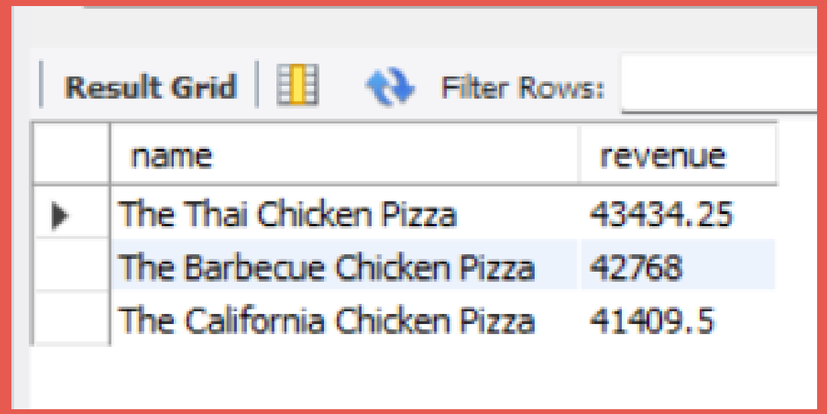
9. GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

```
With a as (select
date(orders.order_date) as date, sum(order_details.quantity) as revenue
from orders join order_details
on orders.order_id = order_details.order_id
group by date)
select round(avg(revenue),0) as avg_pizzas_per_day from a;
```



10.DETERMINE THE TOP 3 PIZZAS ORDERED BY REVENUE

```sql
SELECT
    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 pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```



11. CALCULATE THE PERCENTAGE CONTRIBUTED OF EACH PIZZA CATEGORY TO THE TOTAL REVENUE

```sql
SELECT
    pizza_types.category,
    SUM(order_details.quantity * pizzas.price * 100) / (SELECT
            SUM(order_details.quantity * pizzas.price)
        FROM
            order_details
                JOIN
            pizzas ON order_details.pizza_id = pizzas.pizza_id) AS a
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY a DESC;
```

Result Grid | Filter Rows:

category	a
Classic	26.905960255669893
Supreme	25.45631126009906
Chicken	23.955137556847493
Veggie	23.682590927384418

12. ANALYZE THE CUMMALTIVE REVENUE GENERATED OVER TIME

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

from orders join order_details
on orders.order_id = order_details.order_id

join pizzas
on order_details.pizza_id = pizzas.pizza_id

group by orders.order_date) as sales;
```

Result Grid | Filter Rows:

order_date	cum_rev
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

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

```sql
with ranked_pizzas as
(
select
pizza_types.category, pizza_types.name, sum(order_details.quantity * pizzas.price) as net_profit,
rank() over(partition by pizza_types.category order by sum(order_details.quantity * pizzas.price) desc) as rank_num

from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id

join order_details
on pizzas.pizza_id = order_details.pizza_id

group by pizza_types.category, pizza_types.name
)

select category, name, net_profit
from ranked_pizzas
where rank_num<=3
order by category, net_profit desc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Co

category	name	net_profit
Chicken	The Thai Chicken Pizza	43434.25
Chicken	The Barbecue Chicken Pizza	42768
Chicken	The California Chicken Pizza	41409.5
Classic	The Classic Deluxe Pizza	38180.5
Classic	The Hawaiian Pizza	32273.25
Classic	The Pepperoni Pizza	30161.75
Supreme	The Spicy Italian Pizza	34831.25
Supreme	The Italian Supreme Pizza	33476.75
Supreme	The Sicilian Pizza	30940.5

THANK YOU !!!