BORCELLE

RESTAURANT

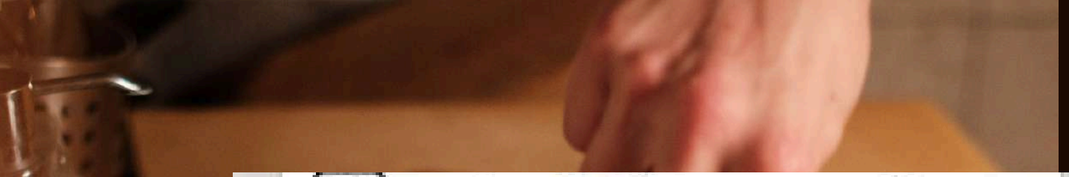
Specialist in Italian Food

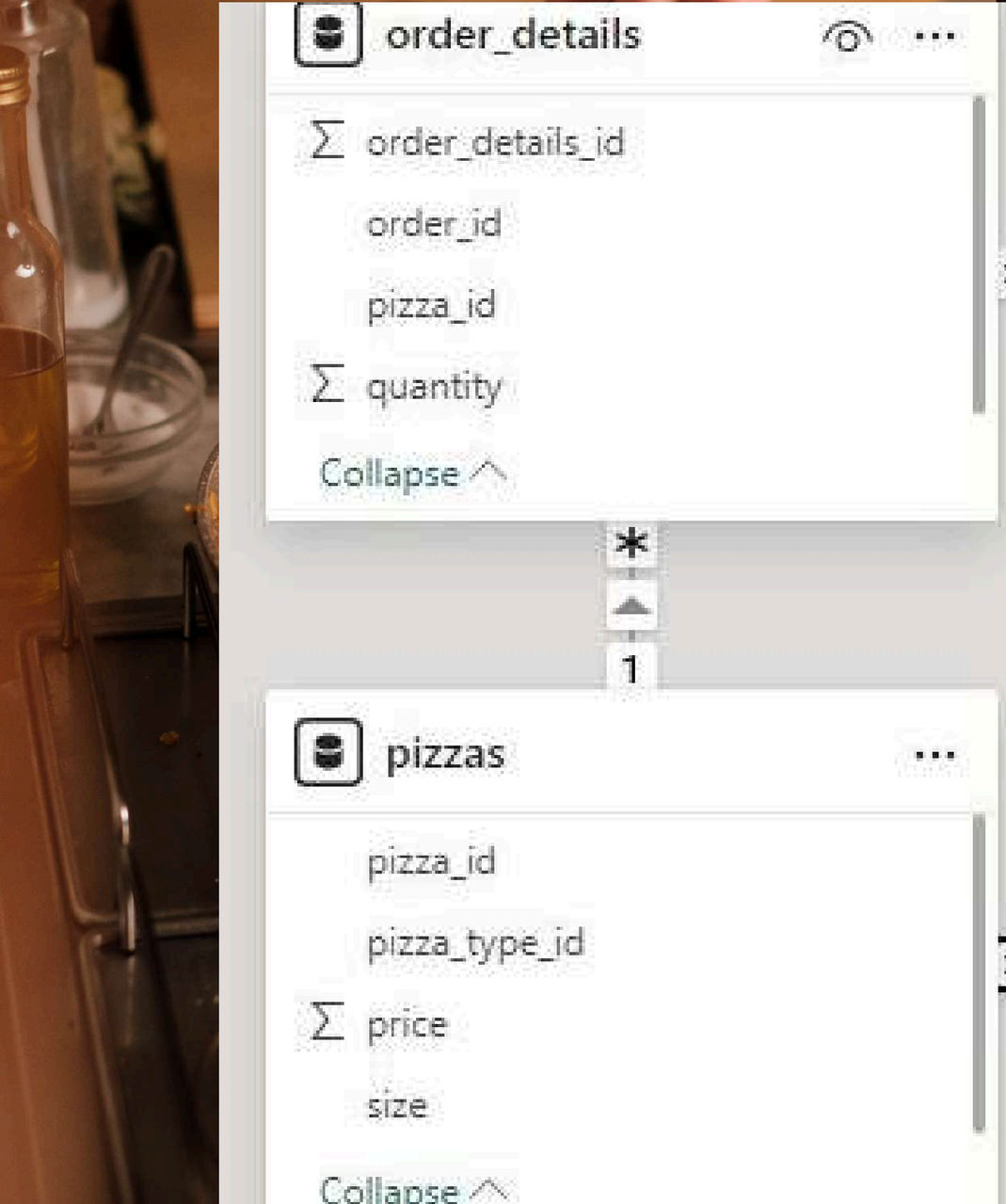
BORCELLE
RESTAURANT

HARSHIT SINGH

In this project, I employed a SQL query to analyze and address questions pertaining to pizza sales.

SCHEMA





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

BORCELLE
RESTAURANT

- To retrieve the total number of orders placed

```
1    -- Retrieve the total number of orders placed.
2 •  select count(order_id) as Total_orders from orders;
```

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

	Total_orders
▶	21350

- To calculate the total revenue generated from pizza sales

```sql
-- Calculate the total revenue generated from pizza sales.
SELECT
    round(SUM(pizzas.price * order_details.quantity),2) AS Revenue
FROM
    pizzas
        INNER JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id;
```

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

Revenue
817860.05

BORCELLE
RESTAURANT

- Identify the highest-priced pizza.

Limit to 1000 rows

```sql
-- Identify the highest-priced pizza.
SELECT
    pizza_types.name, pizzas.price
FROM
    pizzas
        INNER JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

name	price
The Greek Pizza	35.95

- Identify the most commonly ordered pizza.

```sql
-- Identify the most common pizza size ordered.
SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) AS order_count
FROM
    pizzas
        INNER JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC
LIMIT 1;
```

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

size	order_count
L	18526

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

```sql
-- List the top 5 most ordered pizza types along with their quantities.
SELECT
    pizza_types.name, SUM(order_details.quantity) AS Qty
FROM
    pizza_types
        INNER JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        INNER JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY qty DESC
LIMIT 5;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

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

- Find the total quantity ordered for each pizza category.

```sql
-- Join the necessary tables to find the total quantity of each pizza category ordered.
SELECT
    pizza_types.category, SUM(order_details.quantity) AS Qty
FROM
    pizza_types
        INNER JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        INNER JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY qty DESC;
```

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

category	Qty
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

BORCELLE
RESTAURANT

- Analyze the distribution of orders by hour throughout the day.

```sql
-- Determine the distribution of orders by hour of the day.
SELECT
    HOUR(order_time), COUNT(order_id) AS order_count
FROM
    orders
GROUP BY HOUR(order_time);
```

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

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

- Group the orders by date and determine the average number of pizzas ordered daily.

```sql
-- Group the orders by date and calculate the average number of pizzas ordered per day.
SELECT
    ROUND(AVG(quantity), 0)
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    INNER JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

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

round(avg(quantity),0)
138

BORCELLE
RESTAURANT

```sql
-- Determine the top 3 most ordered pizza types based on revenue.
SELECT
    pizza_types.name,
    SUM(pizzas.price * order_details.quantity) AS Revenue
FROM
    pizza_types
        INNER JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        INNER JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY Revenue DESC
LIMIT 3;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch ro

name	Revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

- Determine the top 3 pizza types with the highest revenue.

BORCELLE
RESTAURANT

- Determine the revenue percentage contribution of each pizza category.

```sql
-- Calculate the percentage contribution of each pizza category to total revenue.
SELECT
    pizza_types.category,
    ROUND((SUM(pizzas.price * order_details.quantity) / (SELECT
                SUM(pizzas.price * order_details.quantity)
            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 ORDER BY revenue DESC;
```

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

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

BORCELLE

- Examine the trend of cumulative revenue growth over a specified period.

```sql
-- Analyze the cumulative revenue generated over time.
select order_date,sum(revenue) over (order by order_date) as cum_revenue
from
(select orders.order_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.order_date) as sales ;
```

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

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

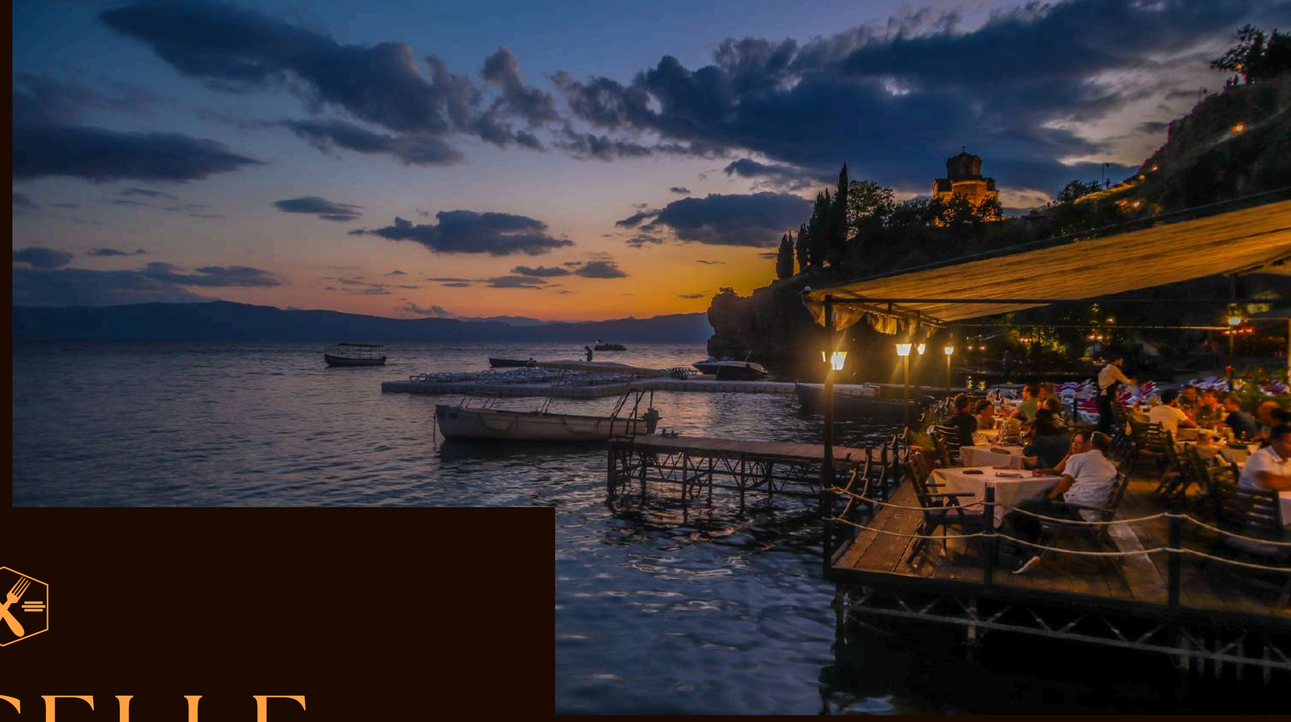
```sql
-- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
select name, revenue from
(select category, name, revenue,
rank() over(partition by category order by revenue desc) as rn
from
(select pizza_types.category, pizza_types.name, sum((order_details.quantity) * 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;
```

Result Grid | Filter Rows:

name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5
The Classic Deluxe Pizza	38180.5
The Hawaiian Pizza	32273.25
The Pepperoni Pizza	30161.75
The Spicy Italian Pizza	34831.25
The Italian Supreme Pizza	33476.75
The Sicilian Pizza	30940.5
The Four Cheese Pizza	32265.70000000065
The Mexicana Pizza	26780.75
The Five Cheese Pizza	26066.5

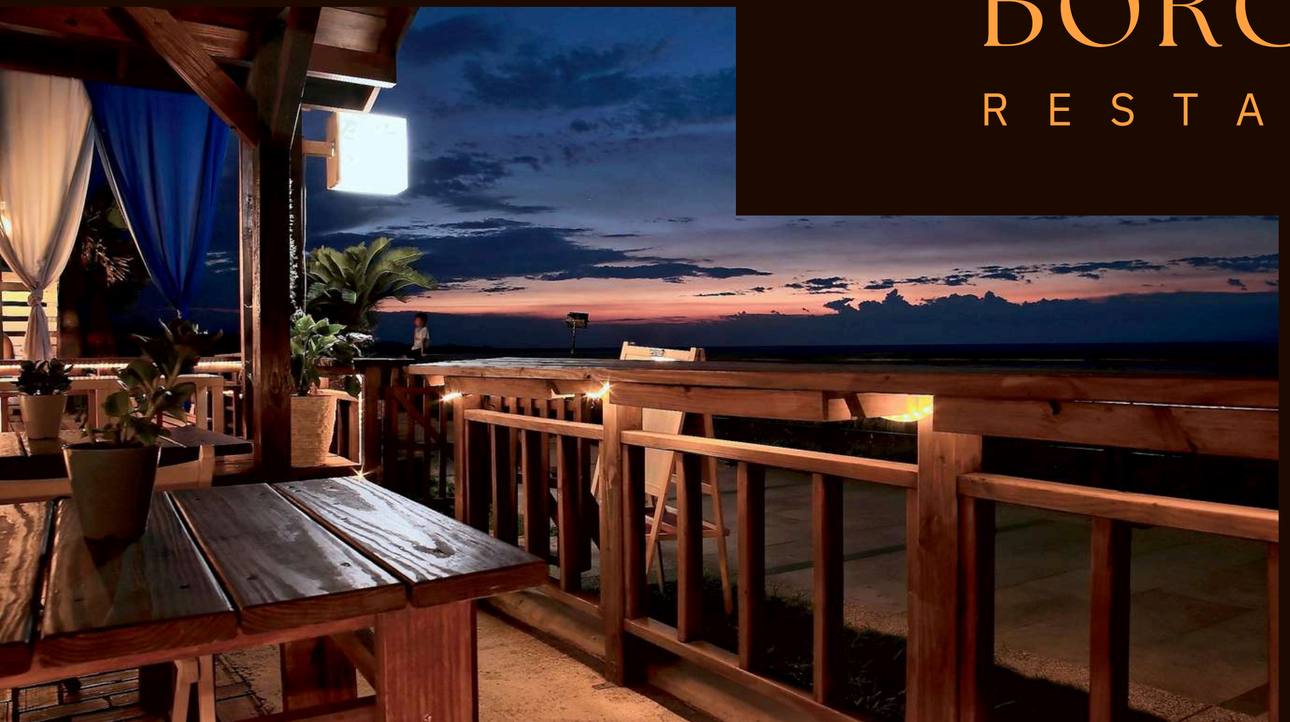
Result 1

- Rank the top 3 pizza types by revenue for each category.



BORCELLE

RESTAURANT



BORCELLE
RESTAURANT

CONTACT US

HARSHIT SINGH

+91 9811068834

singhharshit15@gmail.com

BORCELLE

RESTAURANT

THANK YOU