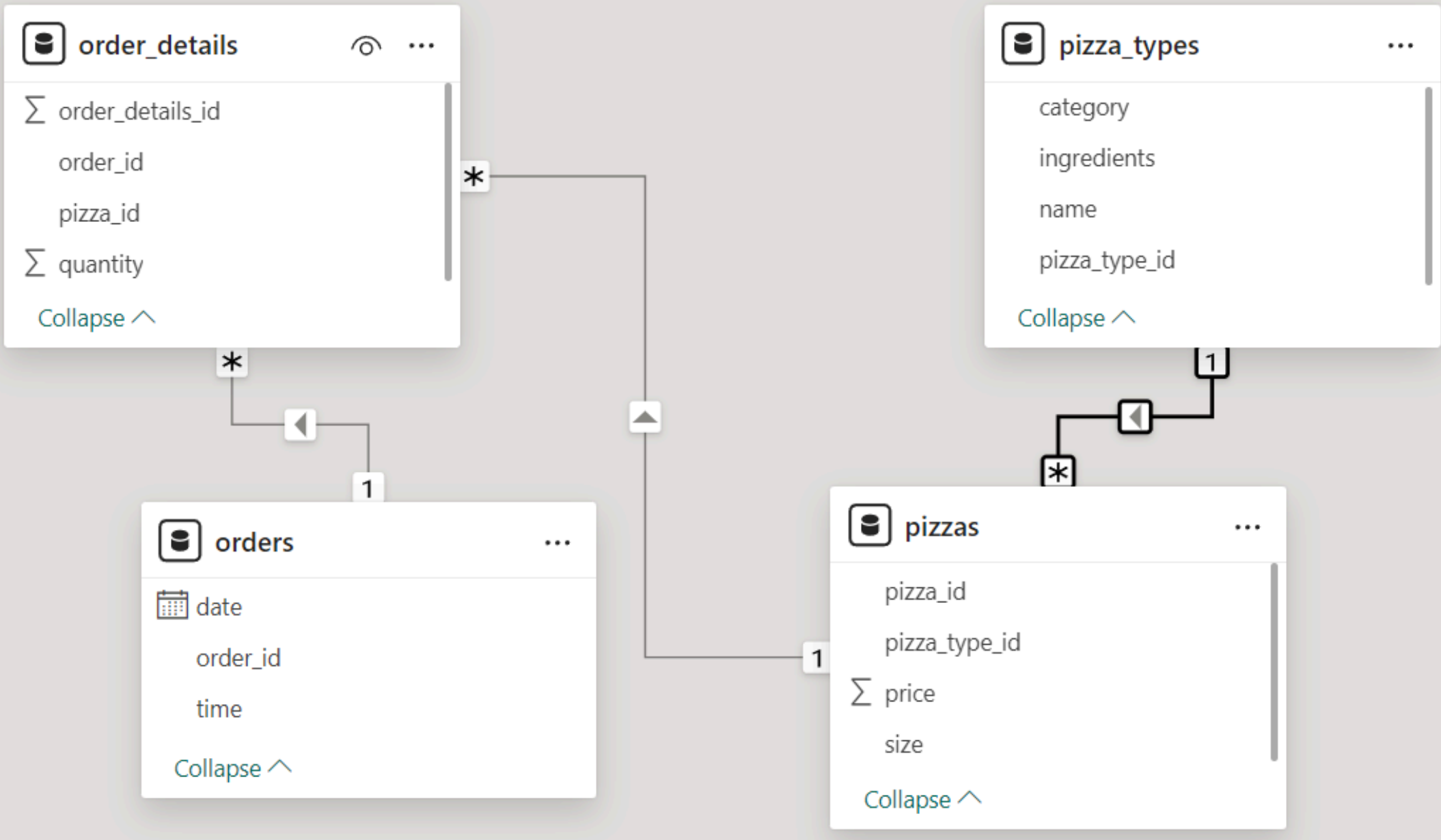
SQL

PIZZA

SALES ANALYSIS

In this project, I have utilized SQl to solve queries that are related to Pizza sales.

Data Model


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

BASIC

Q1. Retrieve the total number of orders placed.

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

	total_orders
▶	21350

Q2. Calculate the total revenue generated from pizza sales.

```
SELECT
    ROUND(SUM(pizzas.price * order_details.quantity),
            2) AS total_revenue
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id;
;
```

	total_revenue
▶	817860.05

Q3. 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;
```

	name	price
▶	The Greek Pizza	35.95

Q4. 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 ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;
```

	size	order_count
▶	L	18526
	M	15385
	S	14137
	XL	544
	XXL	28

Q5. find the category-wise distribution of pizzas.

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

size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

INTERMIDIATE

Q6. Join the necessary tables to find the total quantity of each pizza category ordered.

```sql
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS total_quantity
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 total_quantity DESC;
```

	category	total_quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

Q7. Determine the distribution of orders by hour of the day.

```sql
SELECT
    HOUR(order_time) AS hour, COUNT(order_id) as order_count
FROM
    orders
GROUP BY hour;
```

	category	total_quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

Q8. List the top 5 most ordered pizza types along with their quantities.

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

Q9. Group the orders by date and calculate the average number of pizzas ordered per day.

```sql
SELECT
    ROUND(AVG(quantity), 0) AS avg_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;
```

	avg_orders_per_day
▶	138

Q10. Determine the top 3 most ordered pizza types based on revenue.

```sql
SELECT
    pizza_types.name,
    sum(pizzas.price * order_details.quantity) AS revenue
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.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;
```

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

ADVANCE

Q11. 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)
         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
        )
    ) * 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;
```

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

Q12. 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 orders.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

Q13. Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```sql
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;
```

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