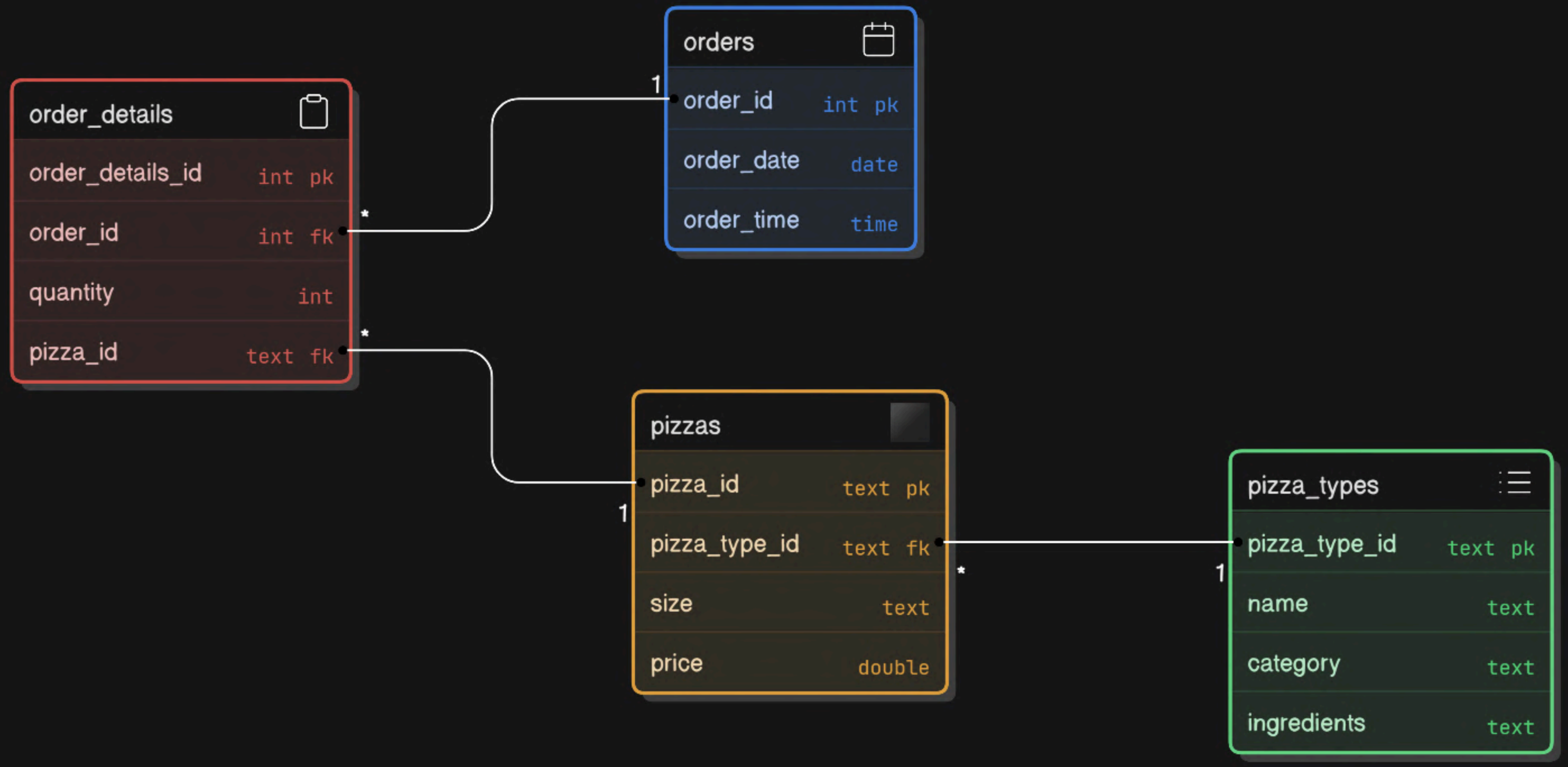
SQL-Driven Pizza Sales Analytics

INTRODUCTION

- I am **Kavya Sharma**, a **second-year BTech CSE student.**
- As part of my learning journey, I developed a project titled **'SQL-Driven Pizza Sales Analytics,'** where **I explored and solved various SQL queries related to pizza sales.**
- This project involved **analyzing orders, revenue, and customer preferences,** showcasing my ability to work with databases and derive meaningful insights.

ER Diagram for Pizza Sales Schema



QUERIES FOR DATA ANALYSIS

- Retrieve the total number of orders placed.

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

Result Grid

	total_order_number
▶	21350

- **Calculate the total revenue generated from pizza sales.**

sales_project* pizzas pizza_types orders order_details S

Limit to 1000 rows

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

Result Grid

total_sales
817860.05

- 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;
```

Result Grid

	name	price
▶	The Greek Pizza	35.95

- Identify the most common pizza size ordered.

es_project pizzas pizza_types orders order_details SQL Fi

Limit to 1000 rows

```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
LIMIT 1;
```

Result Grid Filt

size	order_count
L	18526

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

```sql
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;
```

Result Grid | Filter Rows:

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

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

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

Result Grid

quantity	category
14888	Classic
11987	Supreme
11649	Veggie
11050	Chicken

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

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

Result Grid | Filter Rows:

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

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

Limit to 1000 rows

```sql
SELECT
    COUNT(pizza_types.name) AS total_pizzas, category
FROM
    pizza_types
GROUP BY category;
```

Result Grid Filter Rows:

total_pizzas	category
6	Chicken
8	Classic
9	Supreme
9	Veggie

- **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 AS per_day,
            SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY per_day) AS avg_order_quantity;
```

Result Grid

per_day	quantity
2015-01-01	162
2015-01-02	165
2015-01-03	158
2015-01-04	106
2015-01-05	125
2015-01-06	147
2015-01-07	138
2015-01-08	173

Result Grid

avg_orders_per_day
138

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

```sql
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS revenue,
    pizza_types.name AS pizza_ordered
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
GROUP BY pizza_ordered
ORDER BY revenue DESC
LIMIT 3;
```

Result Grid Filter Rows:

revenue	pizza_ordered
43434.25	The Thai Chicken Pizza
42768	The Barbecue Chicken Pizza
41409.5	The California Chicken Pizza

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

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

Result Grid | Filter Rows:

revenue_percentage	pizza_category
26.91	Classic
25.46	Supreme
23.96	Chicken
23.68	Veggie

- **Analyze the cumulative revenue generated over time.**

```sql
SELECT dates , sum(revenue) OVER (ORDER BY dates ) AS cumulative_revenue
FROM
(SELECT orders.order_date AS dates,
        round(sum(pizzas.price * order_details.quantity),0) 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 dates ) AS sales ;
```

Result Grid | Filter Rows:

dates	cumulative_revenue
2015-01-01	2714
2015-01-02	5446
2015-01-03	8108
2015-01-04	9863
2015-01-05	11929
2015-01-06	14358
2015-01-07	16560
2015-01-08	19398
2015-01-09	21525
2015-01-10	23989

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

```sql
SELECT order_rank , category , pizza_names , revenue
 FROM
(SELECT category , pizza_names , revenue ,
RANK() OVER(PARTITION BY category ORDER BY  revenue DESC ) AS order_rank
FROM
(SELECT pizza_types.category AS category , pizza_types.name AS pizza_names ,
 round(sum(order_details.quantity * pizzas.price),0) 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 category , pizza_names
 ORDER BY revenue DESC ) AS a ) AS b
 WHERE order_rank <= 3
 ORDER BY order_rank ;
```

Result Grid | Filter Rows: | Export:

order_rank	category	pizza_names	revenue
1	Chicken	The Thai Chicken Pizza	43434
1	Classic	The Classic Deluxe Pizza	38180
1	Supreme	The Spicy Italian Pizza	34831
1	Veggie	The Four Cheese Pizza	32266
2	Chicken	The Barbecue Chicken Pizza	42768
2	Classic	The Hawaiian Pizza	32273
2	Supreme	The Italian Supreme Pizza	33477
2	Veggie	The Mexicana Pizza	26781
3	Chicken	The California Chicken Pizza	41410
3	Classic	The Pepperoni Pizza	30162
3	Supreme	The Sicilian Pizza	30940

THANK YOU FOR ATTENTION