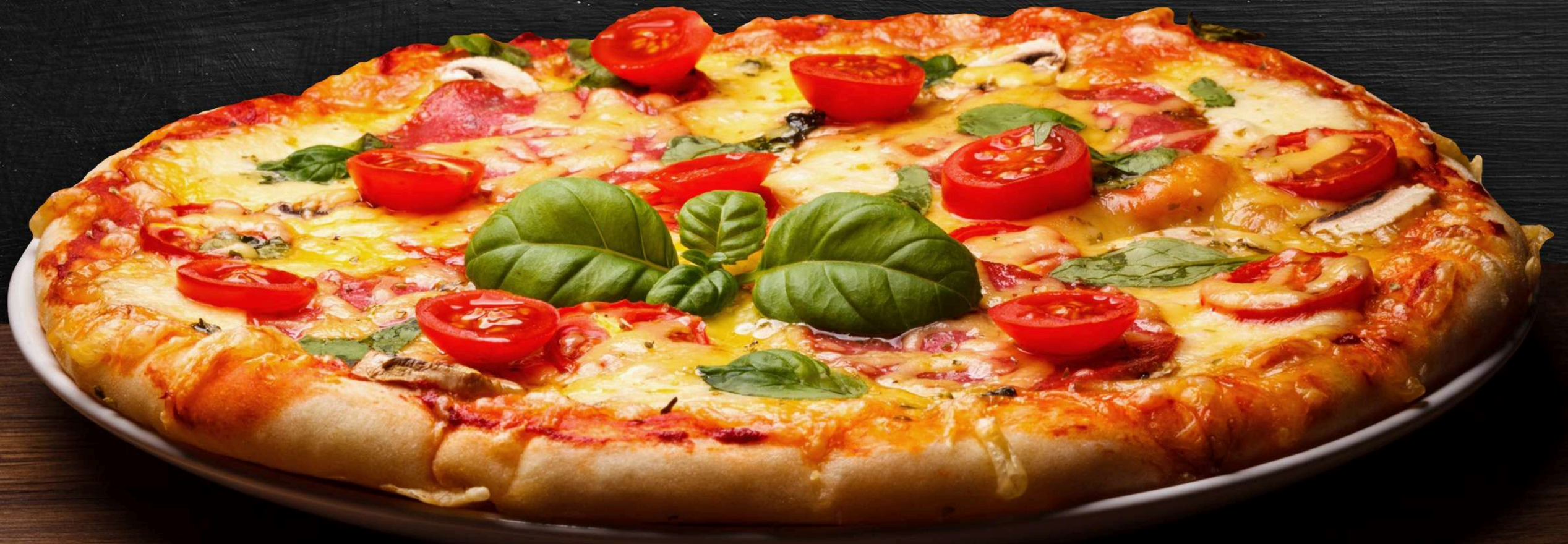
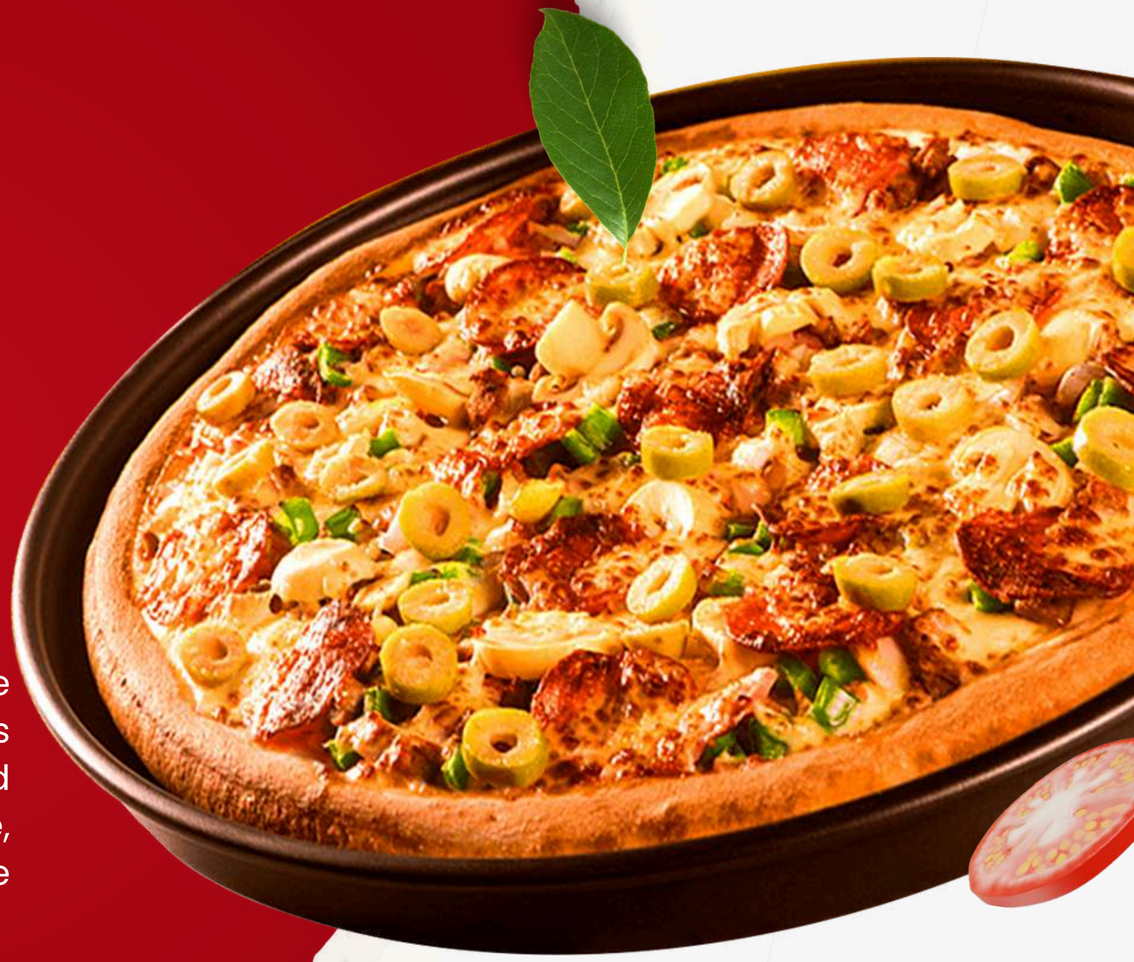
PIZZA HUT SQL SALES REPORT

BY ARUNDEEP GANGONI



PIZZA HUT

WELCOME TO PIZZA HUT SQL SALES REPORT

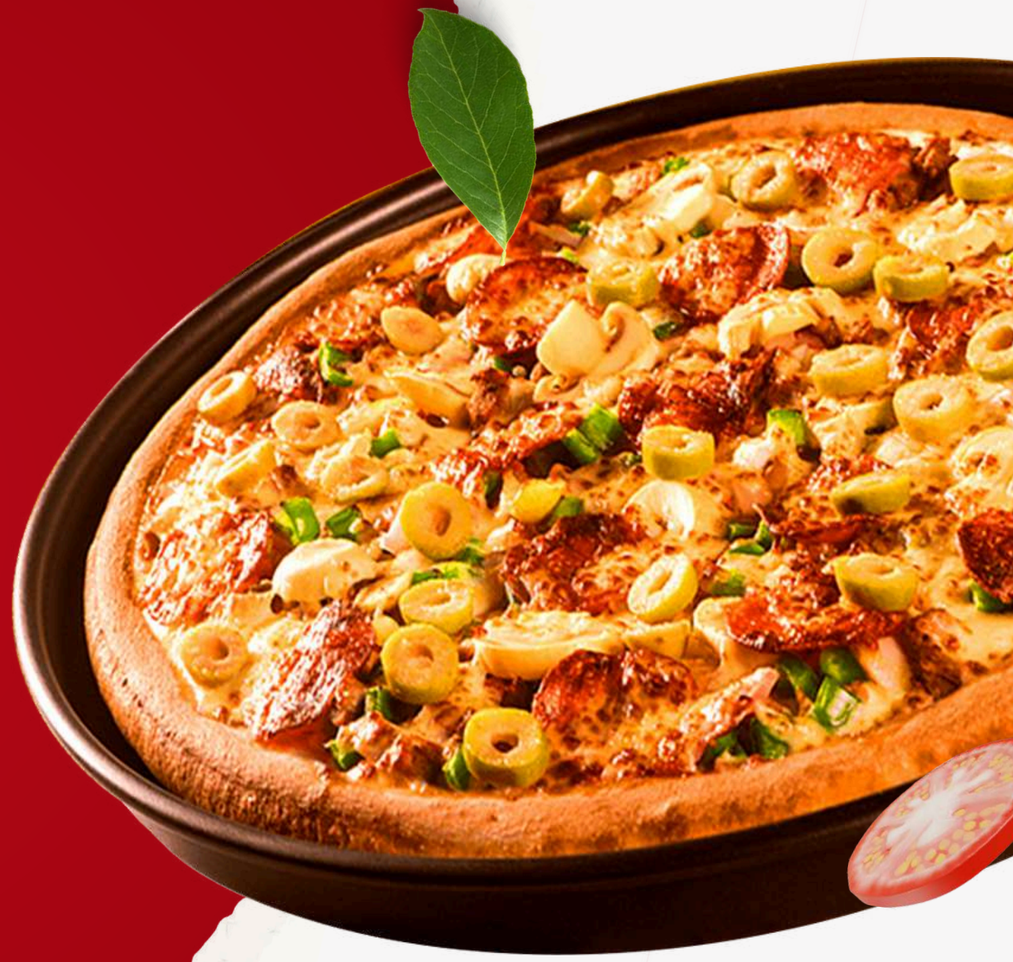


An end-to-end SQL project for Pizza Hut involves creating a comprehensive database to manage pizza orders, order details, revenue, and pizza types. This includes designing tables for customers, orders, pizza types, order items, and sales revenue. SQL queries are used to track customer orders, calculate revenue, manage pizza inventory, and analyze sales trends by pizza type to optimize operations and improve decision-making.

LEVELS OF SCENARIOS

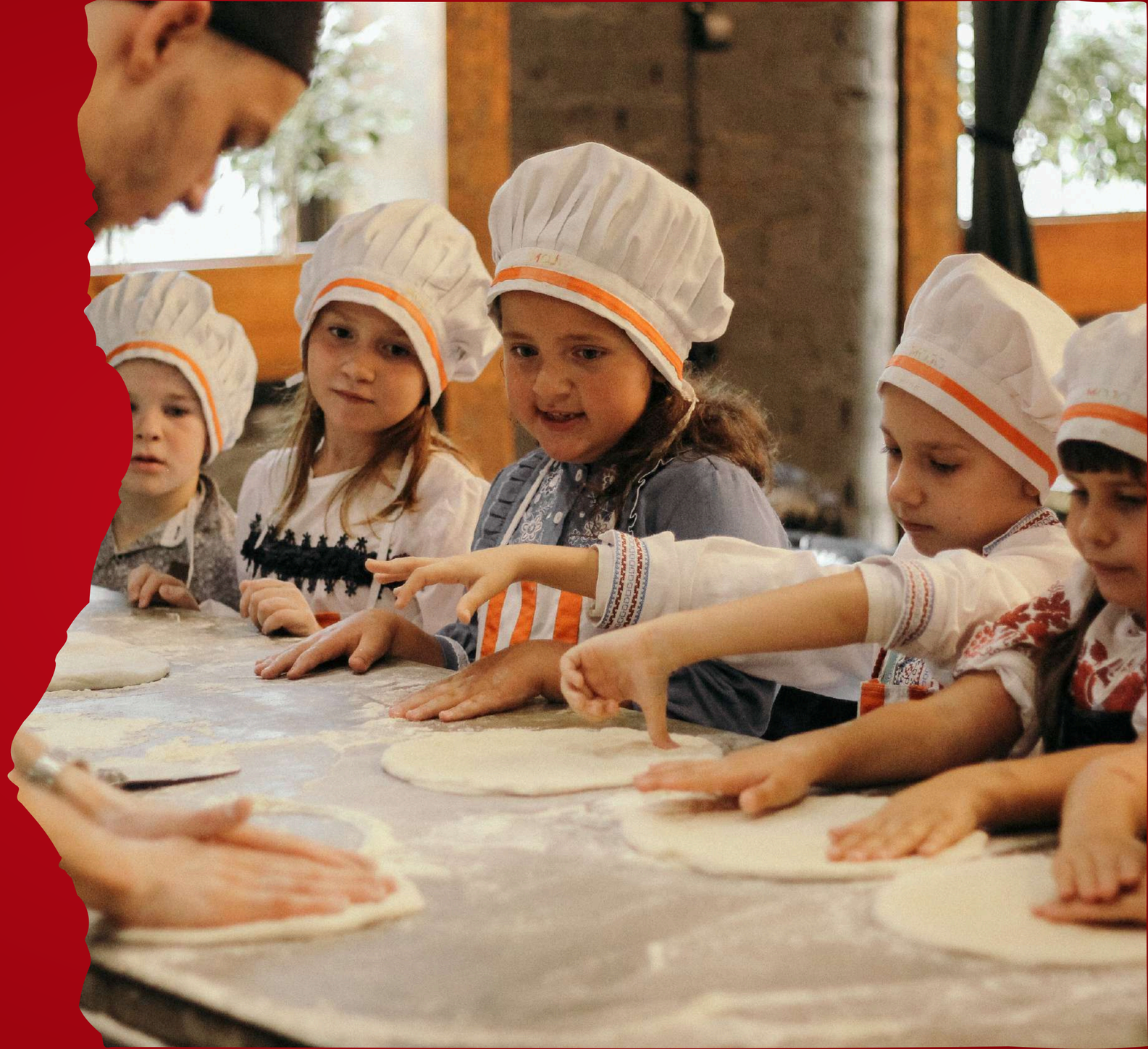
- **Basic Level:** Users begin by creating and managing simple tables such as customers, orders, pizza types, and order details, learning how to organize and store essential data efficiently.
- **Intermediate Level:** Users advance to writing complex SQL queries, including joins to link tables (e.g., orders and pizza types), calculating revenue, and filtering data to retrieve specific order details based on conditions like customer preferences or order status.
- **Advanced Level:** At this stage, users focus on optimizing SQL queries for performance, creating stored procedures to automate order processing, implementing triggers for inventory updates, and performing advanced data analysis like identifying sales trends, customer behavior, and generating detailed reports.

This project helps build a strong SQL foundation while enabling users to tackle more complex data management tasks, enhancing both their technical and problem-solving skills.



PIZZA HUT

BASIC LEVEL

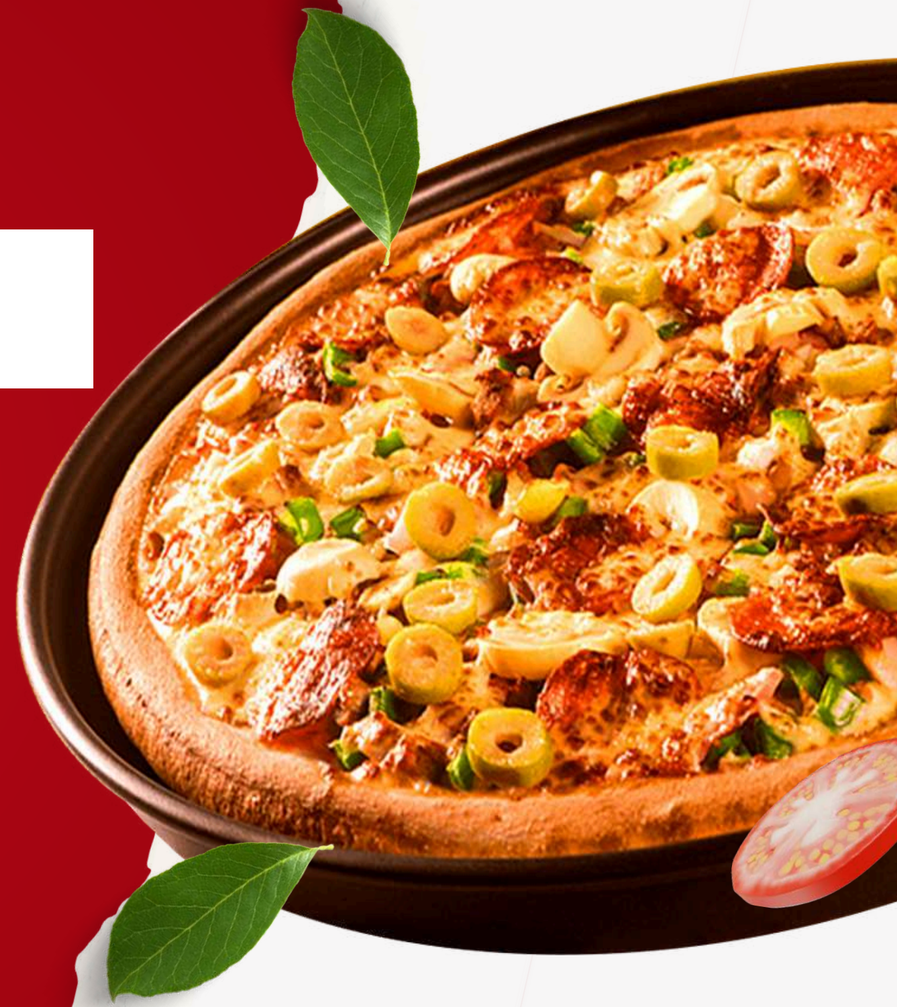


RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED

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

Result Grid

	count(order_id)
▶	21350

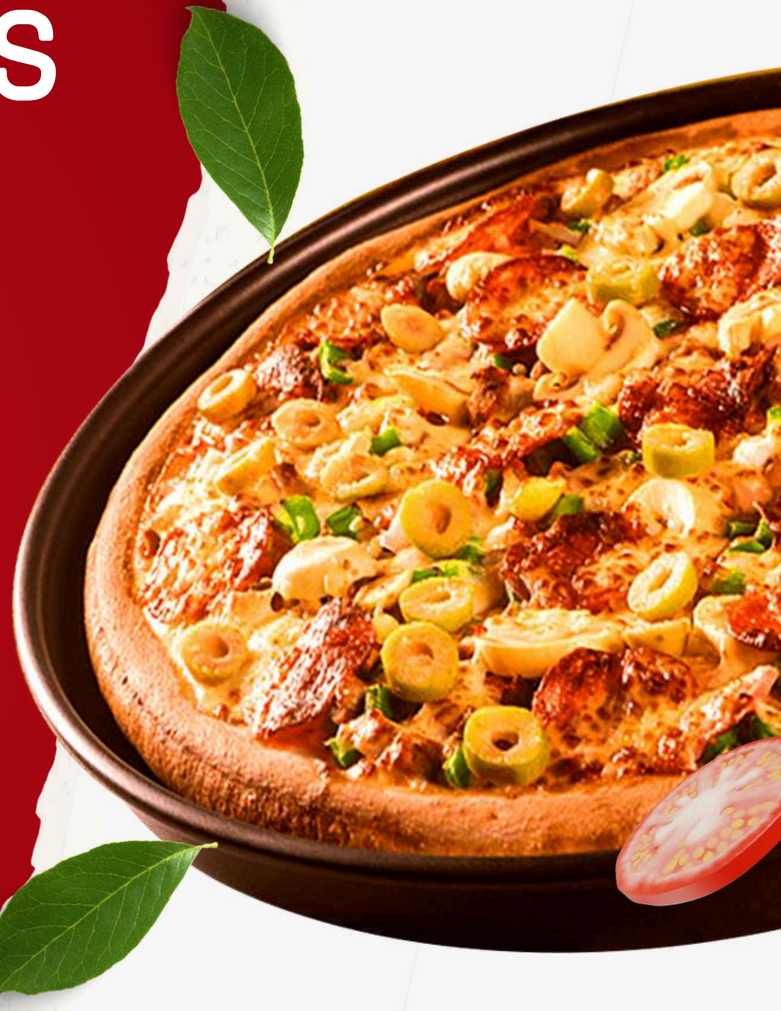


CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES

```sql
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
```

Result Grid

	total_revenue
▶	817860.05



PIZZA HUT

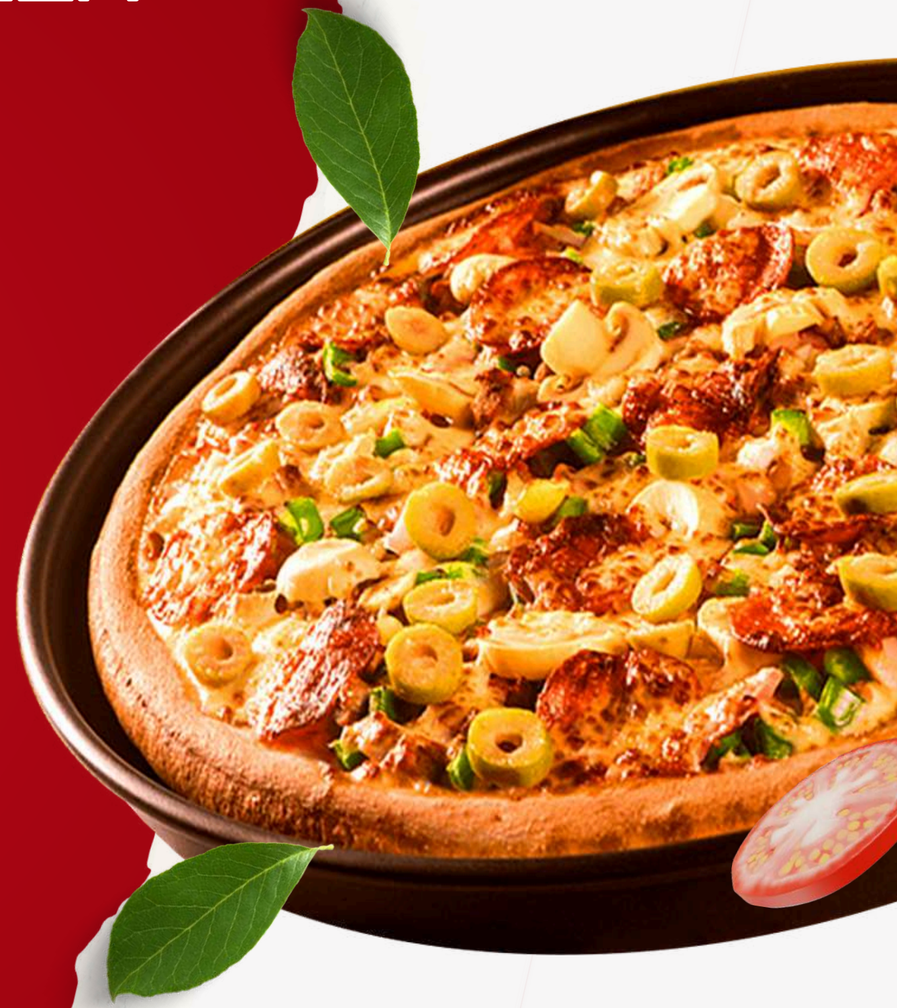
INTERMEDIATE LEVEL

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

```sql
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
```

Result Grid | Filter Rows:

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050



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

```sql
SELECT
    ROUND(AVG(quantity), 0) as avg_qty_pizzas_ordered_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 total_quantity
```

Result Grid | Filter Rows:

	avg_qty_pizzas_ordered_per_day
▶	138

PIZZA HUT

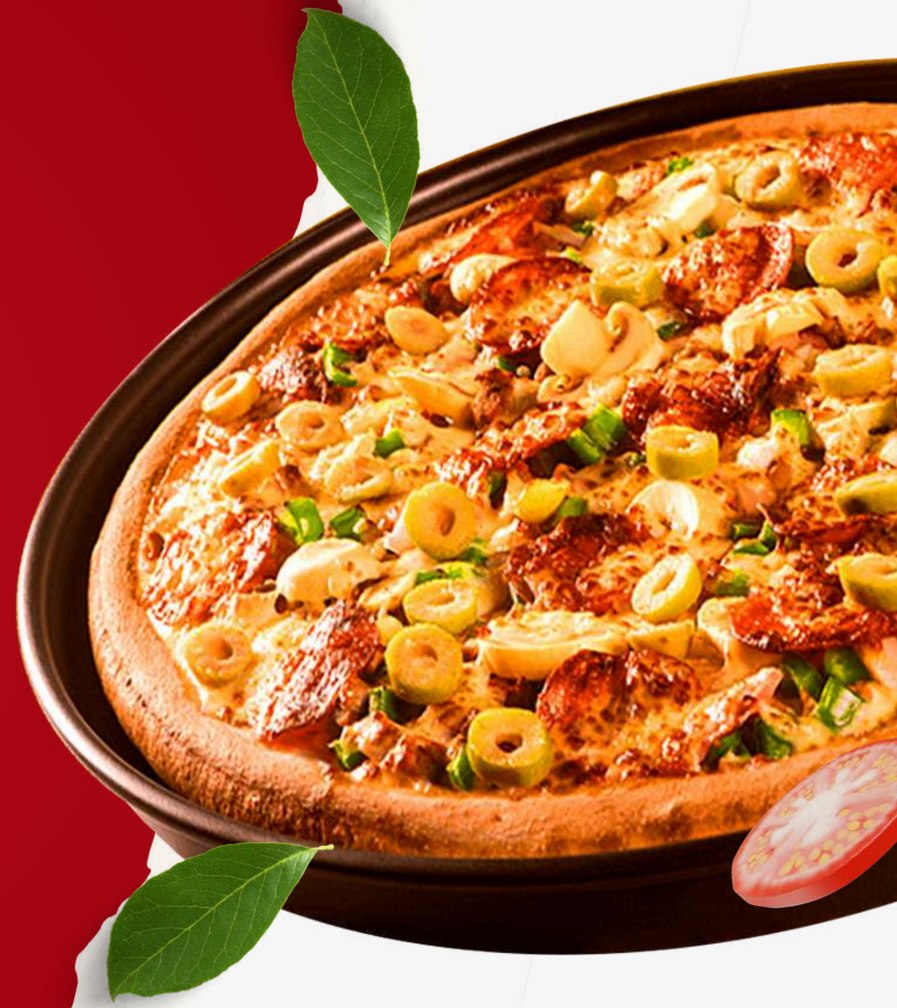
ADVANCE LEVEL

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE

```sql
SELECT
    pizza_types.category,
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) / (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) * 100 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
ORDER BY revenue DESC
```

Result Grid | Filter Rows:

category	revenue
Classic	26.905960255669658
Supreme	25.45631126009884
Chicken	23.955137556847287
Veggie	23.682590927384215



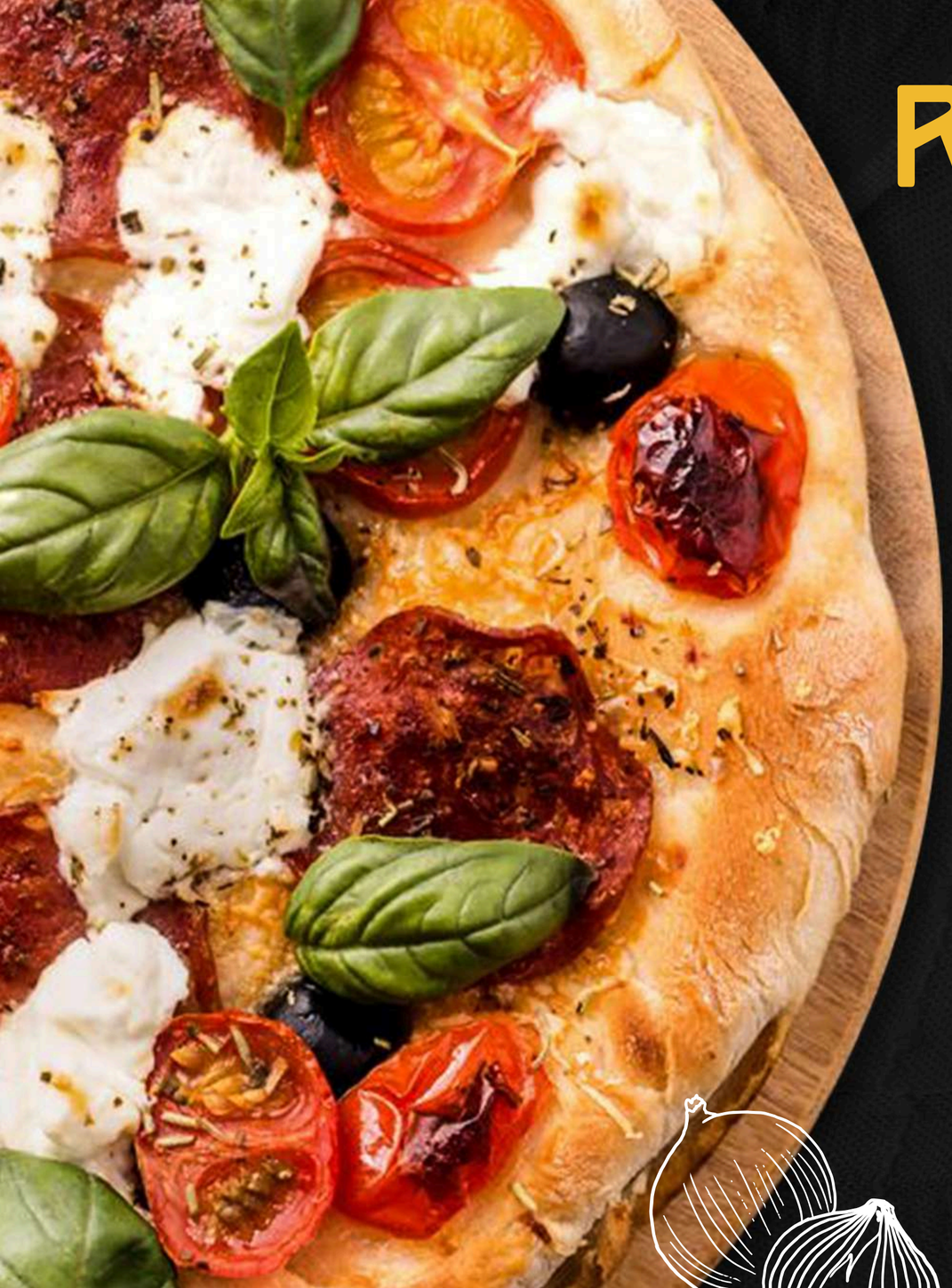
PIZZA HUT

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME

```sql
select order_date, sum(revenue) over(order by order_date) as cummilative_sales
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:

order_date	cummilative_sales
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



RESOURCES

The datasets and a range of SQL questions, covering basic to advanced levels, will be shared in our repository. These resources will help you practice and deepen your SQL knowledge beyond what's covered in this presentation. You can use them to refine your skills and tackle more complex database challenges.

click on below link to access datasets and sql basic to advance level questions

https://github.com/arundeep1212/pizza-hut-sql-project

PIZZA HUT

THANK YOU!