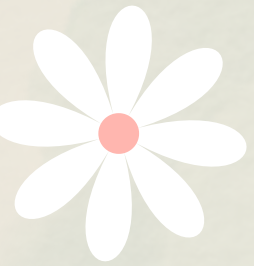# PIZZA SALES ANALYSIS

- Kumar Satyam

PROJECT OVERVIEW

- **Sales Performance: I have calculated total revenue, average order value, total pizzas sold, and total orders. This helps to understand overall sales performance and customer buying habits.**

- **Trends: Analyze daily and monthly trends to see peak sales periods. This can be helpful for staffing and inventory management.**

- **Product Analysis: Identify top-selling pizzas by revenue, quantity, and total orders. You can written queries which categories (e.g., vegetarian, classic) and sizes (small, large) are most popular. This allows to optimize the menu and potentially remove under-performing options.**

CREATE DATABSE

Creating a Database

```sql
create database pizzahut;


CREATE TABLE orders (
      order_id INT NOT NULL,
      order_date DATE NOT NULL,
      order_time TIME NOT NULL,
      PRIMARY KEY (order_id)

);
```
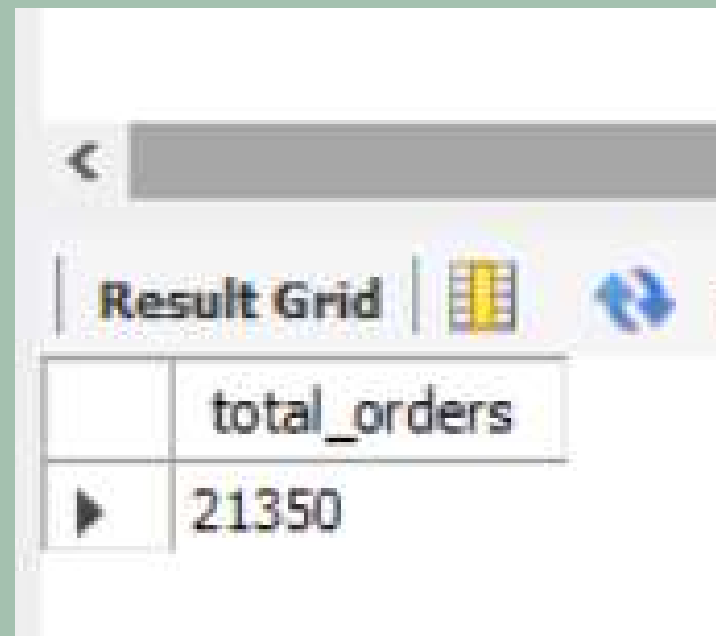
▼ 🗄 **pizzahut**
  ▼ 🗂 Tables
    ▶ ▦ order_details
    ▶ ▦ orders
    ▶ ▦ pizza_types
    ▶ ▦ pizzas
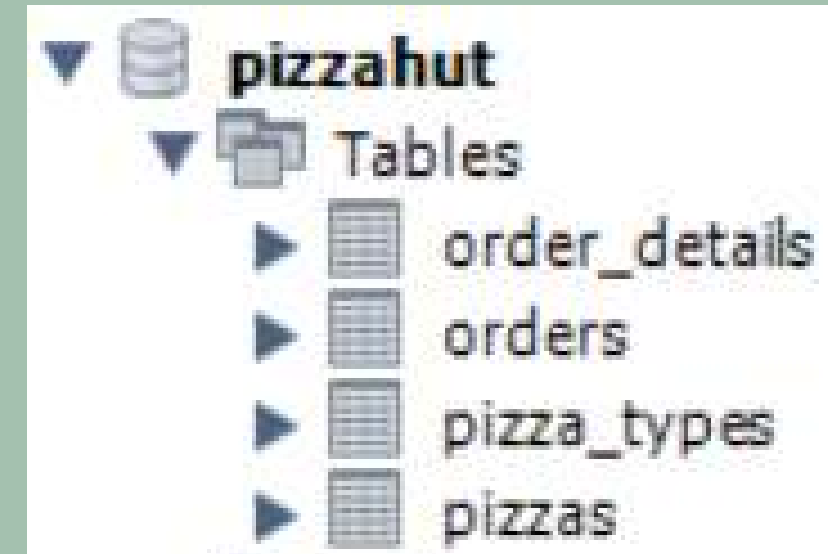
QUERY 1

Retrieve the total number of orders placed.

```sql
select count(order_id) as total_orders from orders;
```
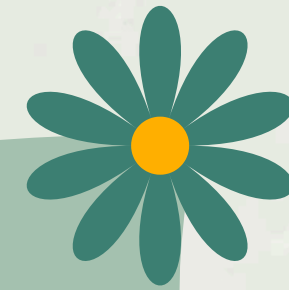
total_orders
21350

QUERY 2

Calculate the total revenue generated
from pizza sales.

```sql
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
```

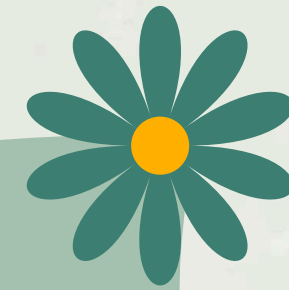| total_sales |
| --- |
| 817860.05 |

Result Grid

QUERY 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;
```

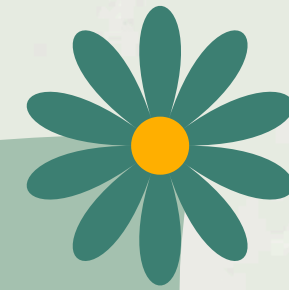| Result Grid	Filter Rows:
name	price
The Greek Pizza	35.95

QUERY 4

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;
```

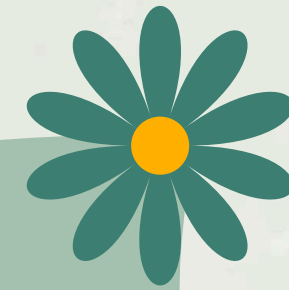| size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

QUERY 5

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;
```

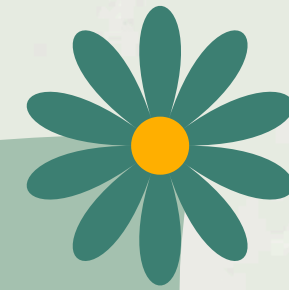| Result Grid	Filter Rows:	
	name	quantity
▶	The Classic Deluxe Pizza	2453
	The Barbecue Chicken Pizza	2432
	The Hawaiian Pizza	2422
	The Pepperoni Pizza	2418
	The Thai Chicken Pizza	2371

QUERY 6

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;
```

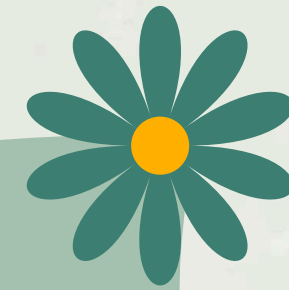| Result Grid		Fil
	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

QUERY 7

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(order_time);
```

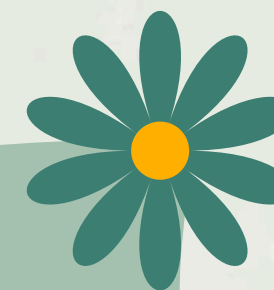| hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336

QUERY 8

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

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

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

QUERY 9

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

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

Result Grid	Filter Rows:
avg_pizza_ordered_per_day
138

# QUERY 10

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

```sql
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) 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.name
ORDER BY revenue DESC
LIMIT 3;
```

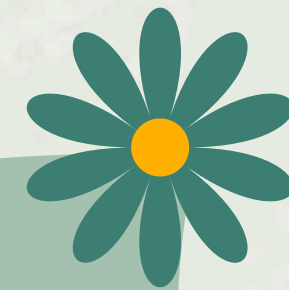|	name	revenue
▶	The Thai Chicken Pizza	43434.25
	The Barbecue Chicken Pizza	42768
	The California Chicken Pizza	41409.5

QUERY 11
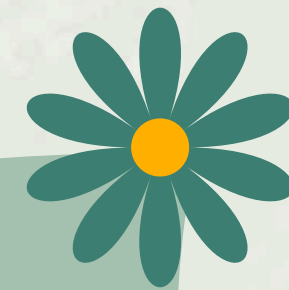
Calculate the percentage contribution of each pizza type to total revenue.

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

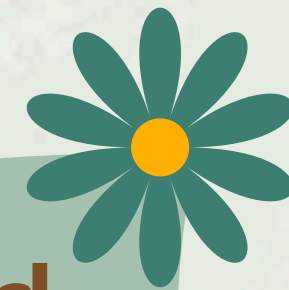|	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

QUERY 12

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,
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;
```

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

QUERY 13

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
The Spicy Italian Pizza	34831.25

CONCLUSION

In conclusion, this analysis of pizza sales data can provide valuable insights into customer behavior and sales trends. We identified peak sales periods, top-performing pizzas, and customer preferences for size and category. This information can be used to optimize the menu offerings, improve staffing and inventory management, and ultimately boost sales and customer satisfaction.

Additionally, further analysis can be conducted to explore factors influencing sales, such as marketing campaigns, competitor activity, and economic conditions.

THANK YOU