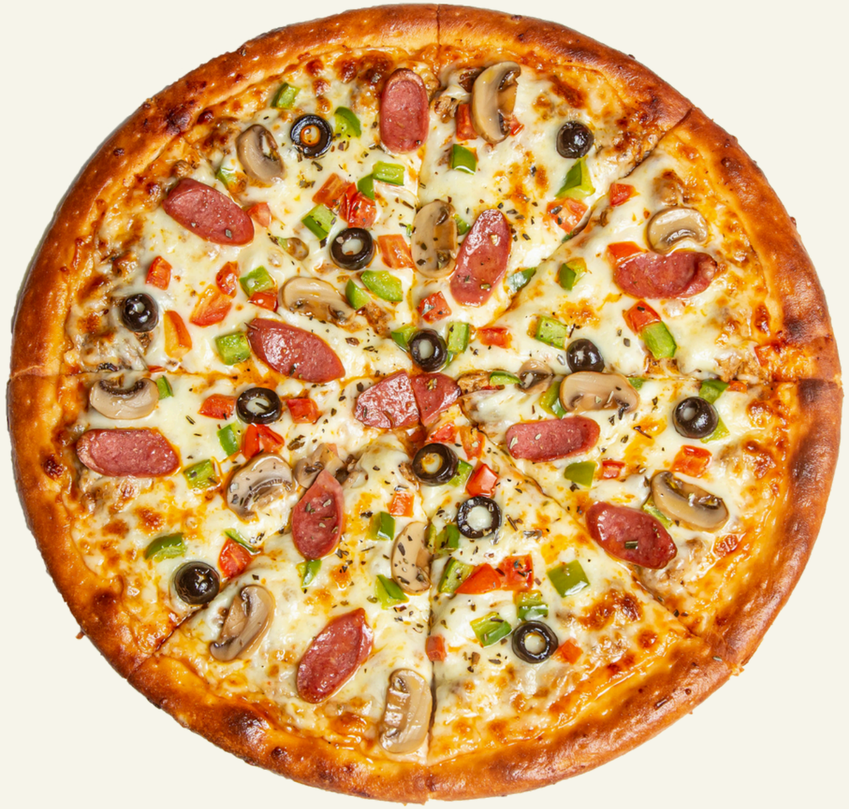
THE WORLD OF PIZZA

Pizza PRESENTATION

– A SLICE OF HAPPINESS IN EVERY BITE –

"TURNING DATA INTO DELICIOUS INSIGHTS WITH SQL"

Pizza sales data provides valuable insights into customer preferences, product performance, and business trends. Through SQL analysis, we can uncover which pizzas sell the most, peak order times, and overall revenue contribution — helping make data-driven business decisions.



This project uses SQL to explore and visualize pizza sales data. By applying queries such as joins, aggregations, and time-based analysis, we discover key insights that can improve marketing, pricing, and inventory strategies.

- PIZZA CONNECTS PEOPLE THROUGH FLAVOR -

PROJECT TITLE

Title: Pizza Sales SQL Analysis

Subtitle: Data Analysis using PostgreSQL

- FROM HUMBLE FLATBREAD TO GLOBAL DELIGHT -

- A SLICE FOR EVERYONE -

PROJECT OVERVIEW

This project analyzes pizza sales data using SQL to extract meaningful business insights such as total revenue, top-selling pizzas, and sales trends over time.

GOALS:

- Practice SQL joins, grouping, and aggregation
- Perform real-world business analysis
- Build a data-driven report

Retrieve the total number of orders placed.

```sql
select
    COUNT(order_id) as total_count
from
    orders1
```

	total_count bigint
1	21350

Calculate the total revenue generated from pizza sales.

```sql
SELECT
    ROUND(SUM(p.price * od.quantity), 2) AS total_revenue
FROM
    order_detail od
    JOIN pizza p ON od.pizza_id = p.pizza_id;
```

	total_revenue numeric
1	817860.05

Identify the highest-priced pizza.

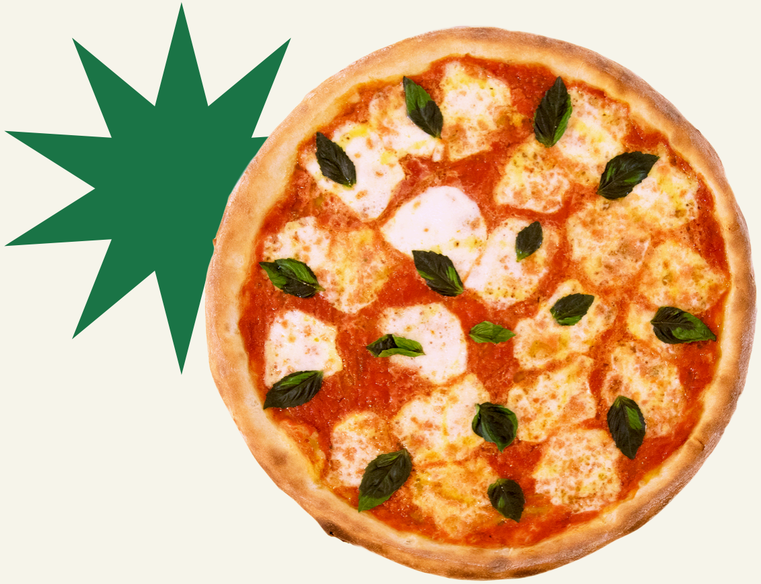
```sql
select
    pt.pizza_name,
    p.sizes,
    p.price
from
    pizzas_type pt
    join pizza p on p.pizza_type_id = pt.pizza_type_id
order by
    price desc
limit
    1;
```

pizza_name text	sizes text	price numeric
The Greek Pizza	XXL	35.95

Identify the most common pizza size ordered.

```sql
select
    p.sizes,
    COUNT(od.order_details_id) as order_count
from
    order_detail od
    join pizza p on p.pizza_id = od.pizza_id
group by
    p.sizes
order by
    order_count desc
limit
    1;
```

	sizes text	order_count bigint
1	L	18526



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

```sql
select
    pt.pizza_name as pizza_name,
    SUM(od.quantity) as total_quantity
from
    order_detail od
    join pizza p on p.pizza_id = od.pizza_id
    join pizzas_type pt on p.pizza_type_id = pt.pizza_type_id
group by
    pt.pizza_name
order by
    total_quantity desc
limit
    5;
```

	pizza_name text	total_quantity bigint
1	The Classic Deluxe Pizza	2453
2	The Barbecue Chicken Pizza	2432
3	The Hawaiian Pizza	2422
4	The Pepperoni Pizza	2418
5	The Thai Chicken Pizza	2371

- A GLOBAL FAVORITE -

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

```sql
select
    pt.category,
    SUM(od.quantity) as total_quantity
from
    order_detail od
    join pizza p on p.pizza_id = od.pizza_id
    join pizzas_type pt on p.pizza_type_id = pt.pizza_type_id
group by
    pt.category
order by
    total_quantity desc;
```

category text	total_quantity bigint
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

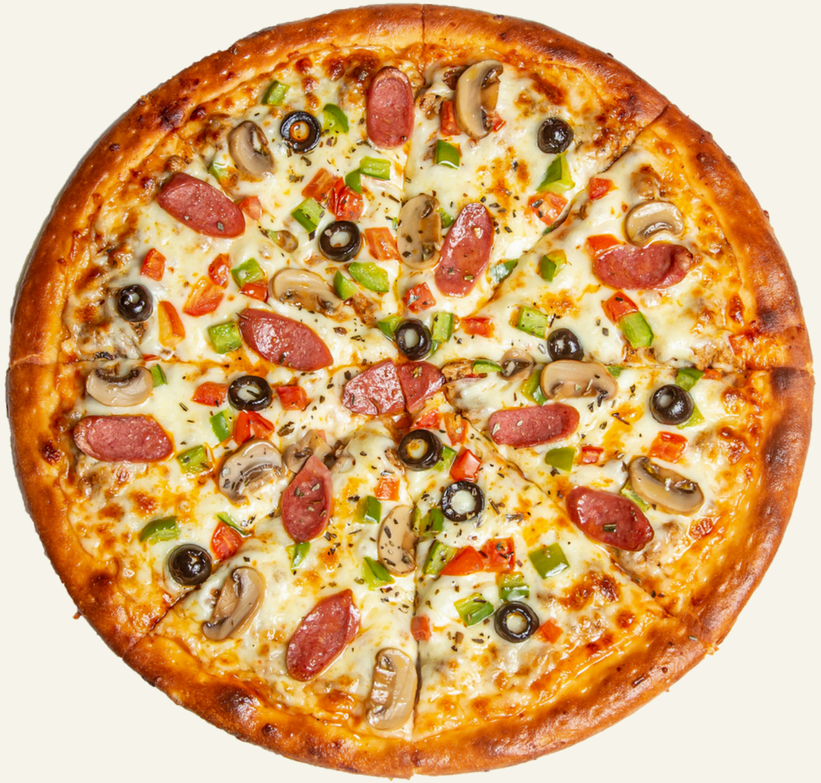
Join relevant tables to find the category-wise distribution of pizzas.

```sql
select
    pt.category,
    COUNT(distinct p.pizza_id) as totaL_pizza
from
    pizza p
    join pizzas_type pt on p.pizza_type_id = pt.pizza_type_id
group by
    pt.category
order by
    total_pizza desc;
```

category text	total_pizza bigint
Veggie	27
Classic	26
Supreme	25
Chicken	18

Determine the distribution of orders by hour of the day.

```sql
select
    EXTRACT(
        hour
        from
            order_time
    ) as order_hour,
    COUNT(order_id) as total_order
from
    orders1
group by
    EXTRACT(
        hour
        from
            order_time
    )
order by
    order_hour;
```

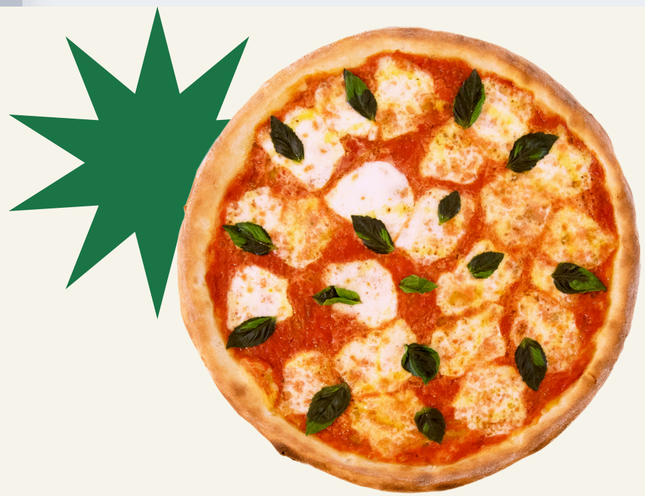


order_hour numeric	total_order bigint
9	1
10	8
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28

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

```sql
select
    ROUND(AVG(daily_pizza), 0) as avg_pizzas_per_day
from
    (
        select
            o.order_date,
            SUM(od.quantity) as daily_pizza
        from
            orders1 o
            join order_detail od on o.order_id = od.order_id
        group by
            o.order_date
    ) as a;
```

avg_pizzas_per_day numeric
138



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

```sql
select
    pt.category,
    ROUND(
        SUM(p.price * od.quantity) / (
            select
                ROUND(SUM(p.price * od.quantity), 2) as total_sales
            from
                order_detail od
                join pizza p on p.pizza_id = od.pizza_id
        ) * 100,
        0
    ) as revenue
from
    order_detail od
    join pizza p on p.pizza_id = od.pizza_id
    join pizzas_type pt on p.pizza_type_id = pt.pizza_type_id
group by
    pt.category
order by
    revenue desc;
```

category text	revenue numeric
Classic	27
Supreme	25
Chicken	24
Veggie	24

Analyze the cumulative revenue generated over time.

```sql
select
    order_date,
    SUM(total_revenue) over (
        order by
            order_date
    ) as cum_revenue
from
    (
        select
            o.order_date,
            SUM(p.price * od.quantity) as total_revenue
        from
            order_detail od
            join pizza p on p.pizza_id = od.pizza_id
            join orders1 o on o.order_id = od.order_id
        group by
            o.order_date
    ) as sales;
```

order_date date	cum_revenue numeric
2015-01-01	2713.85
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.60
2015-01-05	11929.55
2015-01-06	14358.50
2015-01-07	16560.70
2015-01-08	19399.05
2015-01-09	21526.40
2015-01-10	23990.35
2015-01-11	25862.65
2015-01-12	27781.70
2015-01-13	29831.30
2015-01-14	32358.70
2015-01-15	34343.50

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

```sql
select
    category, pizza_name,revenue
from(
     select
        category, pizza_name, revenue,
            RANK() over (partition by category order by revenue desc) as rank
        from(
              select
                    pt.pizza_name, pt.category,
                    SUM(p.price * od.quantity) as revenue
                from
                    order_detail od
                    join pizza p on p.pizza_id = od.pizza_id
                    join pizzas_type pt on p.pizza_type_id = pt.pizza_type_id
                group by
                    pt.pizza_name,
                    pt.category) as a) as b
where rank <= 3;
```

category text	pizza_name text	revenue numeric
Chicken	The Thai Chicken Pizza	43434.25
Chicken	The Barbecue Chicken Pizza	42768.00
Chicken	The California Chicken Pizza	41409.50
Classic	The Classic Deluxe Pizza	38180.5
Classic	The Hawaiian Pizza	32273.25
Classic	The Pepperoni Pizza	30161.75
Supreme	The Spicy Italian Pizza	34831.25
Supreme	The Italian Supreme Pizza	33476.75
Supreme	The Sicilian Pizza	30940.50
Veggie	The Four Cheese Pizza	32265.70
Veggie	The Mexicana Pizza	26780.75
Veggie	The Five Cheese Pizza	26066.5

Project Summary

This SQL project analyzed pizza sales data to uncover patterns in customer behavior, product performance, and revenue generation.

Using PostgreSQL, multiple queries involving joins, aggregations, and time-based analysis helped extract valuable insights from raw data.

Final Insight

SQL-based analysis provided actionable insights into product demand, customer timing patterns, and profitability — helping the business make data-driven decisions for marketing and inventory management.

-Data may be numbers, but with SQL, it tells a delicious story — one slice at a time. -

ONE SLICE AT A TIME

MORE THAN A MEAL

Thank you

– PIZZA IS LIFE, SHARED ONE SLICE AT A TIME –