# PIZZA SALES- SQL PROJECT

Welcome to pizza sales analysis presentation! Using SQL, I have delved the queries across three levels:

**Basic**: We'll cover total orders, revenue, the highest-priced pizza, the most common size, and the top 5 pizza types by quantity.

**Intermediate**: We've examined quantities by category, hourly order distribution, category-wise pizza distribution, average daily orders, and the top 3 pizza types by revenue.

**Advanced**: We'll explore each pizza type's revenue contribution, cumulative revenue over time, and the top 3 pizza types by revenue within each category.

# Retrieve the total number of orders placed

```sql
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

Result Grid | Filter Rows: | Export:

| | total_orders |
|---|---|
| ▶ | 21350 |

# Calculate the total revenue generated from pizza sales.

```sql
select
sum(order_details.quantity * pizzas.price) as total_sales
from order_details join pizzas
on pizzas.pizza_id = order_details.pizza_id;

-- roundoff

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 | Filter Rows: | Export:

| 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 | Filter Rows: | Export:

| name | price |
|---|---|
| The Greek Pizza | 35.95 |

# Identify the most common pizza size ordered.

```
SELECT
    quantity, COUNT(order_details_id) FROM order_details
GROUP BY quantity;
 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;
```

Result Grid | Filter Rows: | Export:

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

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

```
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: | Export:

| 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
    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: | Export:

| category | quantity |
|---|---|
| Classic | 14888 |
| Supreme | 11987 |
| Veggie | 11649 |
| Chicken | 11050 |

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

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

Result Grid | Filter Rows: | Export:

| hour | order_count |
|---|---|
| 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 |

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

```
select category , count(name) from pizza_types
group by category
```

Result Grid | Filter Rows: | Export:

| category | count(name) |
|---|---|
| Chicken | 6 |
| Classic | 8 |
| Supreme | 9 |
| Veggie | 9 |

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

```sql
SELECT
    round(AVG(quantity),0) as average_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 order_quantity;
```

Result Grid | Filter Rows: | Export:

| | average_pizzas_ordered_per_day |
|---|---|
| ▶ | 138 |

# 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;
```

Result Grid | Filter Rows:

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

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

```sql
SELECT
    pizza_types.category,
    (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) 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 | 26.9059602556696 7 |
| Supreme | 25.45631126009862 |
| Chicken | 23.955137556847287 |
| Veggie | 23.682590927384577 |

# Analyze the cumulative revenue generated over time.

```sql
select order_date,
sum(revenue) over (order by order_date) as cumulative_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;
```

Result Grid | Filter Rows:

| order_date | cumulative_revenue |
|---|---|
| 2015-01-01 | 2713.850000000004 |
| 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 |
| 2015-01-08 | 19399.05 |
| 2015-01-09 | 21526.4 |
| 2015-01-10 | 23990.350000000002 |

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

```
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;
```

Result Grid | Filter Rows: | Exp

| 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 |

# Thank You!

**Email**
akankshalanghani13@gmail.com

**LinkedIn**
@AakankshaLanghani

**GitHub**
https://github.com/AakankshaLanghani