SHODWE
Pizza Resto

Home About Contact

# ANALYZING PIZZA SALES PERFORMANCE USING SQL

- WHERE EVERY DATA POINT TELLS A STORY

SHODWE
Pizza Resto

Home About Contact

# ABOUT US

Every data point has a story to tell. This presentation walks through SQL queries that transform raw pizza sales data into meaningful insights. From understanding revenue distribution across categories to identifying the best-selling pizzas, we use SQL to unlock valuable trends and support strategic business decisions.

## 01 Find the total count of orders placed

Query Input

```
SELECT count(order_id) as Total_orders FROM food_db.orders;
```

Query Output

| | Total_orders |
|---|---|
| ▶ | 21350 |

## 02 Compute the total revenue generated from all pizza sales.

Query Input

```
select round(sum(od.quantity*p.price),2) as total_revenue
from order_details od
join pizzas p
on od.pizza_id=p.pizza_id
```

Query Output

| total_revenue |
| --- |
| 817860.05 |

## 03 Find the pizza with the highest price.

Query Input

```
SELECT
  pt.name, p.price
from
  pizza_types pt
join
  pizzas p
on
  pt.pizza_type_id=p.pizza_type_id
order by
  p.price desc
limit 1;
```

Query Output

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

## 04 Determine the most frequently ordered pizza.

Query Input

```
SELECT
    name, COUNT(quantity) as no_of_pizza_sales
FROM
   pizza_types pt
JOIN
   pizzas p ON p.pizza_type_id=pt.pizza_type_id
JOIN
   order_details od ON od.pizza_id=p.pizza_id
GROUP BY name
ORDER BY no_of_pizza_sales DESC
LIMIT 1;
```

Query Output

| name | no_of_pizza_sales |
|---|---|
| The Classic Deluxe Pizza | 2416 |

## 05 Determine the most frequently ordered pizza size.

Query Input

```
SELECT
    p.size,
    COUNT(od.quantity) AS no_of_sales
FROM pizzas p
JOIN order_details od ON p.pizza_id = od.pizza_id
GROUP BY p.size
ORDER BY no_of_sales DESC
LIMIT 1;
```

Query Output

| size | no_of_sales |
|---|---|
| L | 18526 |

## 06 Retrieve the top 5 most ordered pizza types along with their total quantities sold

Query Input

```sql
SELECT
    name, sum(quantity) as quantity
FROM
   pizza_types pt
JOIN
   pizzas p ON p.pizza_type_id=pt.pizza_type_id
JOIN
   order_details od ON od.pizza_id=p.pizza_id
GROUP BY name
ORDER BY quantity DESC
LIMIT 5;
```

Query Output

| 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 |

07

## Join the relevant tables to calculate the total quantity of pizzas ordered for each category

Query Input

```
select
  pt.category, sum(od.quantity) as total_quantity
 from
  pizza_types pt
 join
  pizzas p on p.pizza_type_id=pt.pizza_type_id
 join
  order_details od on od.pizza_id=p.pizza_id
 group by pt.category
 order by total_quantity desc
```

Query Output

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

## 08 Analyze the hourly distribution of orders throughout the day

Query Input

```sql
SELECT
    HOUR(time) AS hour, count(order_id) AS order_count
FROM
    orders
group by hour;
```

Query Output

| 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 |
| 10 | 8 |
| 9 | 1 |

09

## Determine the distribution of pizzas across different categories by joining relevant tables

Query Input

```sql
SELECT
    category, count(name) AS Total
FROM
    pizza_types
    GROUP BY category
    ORDER BY Total DESC
```

Query Output

| category | Total |
|---|---|
| Supreme | 9 |
| Veggie | 9 |
| Classic | 8 |
| Chicken | 6 |

## 10 Calculate the average number of pizzas ordered per day by grouping orders by date.

Query Input

```sql
SELECT
   ROUND(AVG(quantity),0) AS avg_pizza_per_day
FROM
  (SELECT
    date, SUM(quantity) AS quantity
FROM
    orders o
JOIN order_details od ON o.order_id=od.order_id
GROUP BY date) AS order_quantity;
```

Query Output

| avg_pizza_per_day |
| --- |
| 138 |

## 11 Identify the top 3 highest revenue-generating pizza types based on total orders

Query Input

```
SELECT
  name, ROUND(sum(price * quantity),1) as revenue
FROM
  pizza_types pt
JOIN
  pizzas p ON pt.pizza_type_id=p.pizza_type_id
JOIN
  order_details od on od.pizza_id=p.pizza_id
GROUP BY  name
ORDER BY revenue desc
LIMIT 3
```

Query Output

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

## 12 Compute the percentage share of each pizza type in the total revenue

Query Input

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

Query Output

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

## 13 Examine the cumulative revenue growth over time

Query Input

```sql
SELECT
  date, ROUND(sum(revenue) OVER(order by date),2) as cum_revenue
FROM
(SELECT date, sum(price * quantity) AS revenue
FROM order_details od
JOIN pizzas p
ON od.pizza_id=p.pizza_id
JOIN orders o
ON o.order_id=od.order_id
GROUP BY date) AS sales
```

Query Output

| date | cum_revenue |
|---|---|
| 2015-01-01 | 2713.85 |
| 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 | 23990.35 |
| 2015-01-11 | 25862.65 |
| 2015-01-12 | 27781.7 |
| 2015-01-13 | 29831.3 |
| 2015-01-14 | 32358.7 |
| 2015-01-15 | 34343.5 |
| 2015-01-16 | 36937.65 |

## 14 Identify the top 3 highest revenue-generating pizza types within each pizza category

Query Input

```sql
SELECT *
FROM (
    SELECT
        category,
        name,
        revenue,
        RANK() OVER(PARTITION BY category ORDER BY revenue DESC) AS rn
    FROM (
        SELECT
            pt.category,
            pt.name,
            SUM(p.price * od.quantity) AS revenue
        FROM pizza_types pt
        JOIN pizzas p ON pt.pizza_type_id = p.pizza_type_id
        JOIN order_details od ON od.pizza_id = p.pizza_id
        GROUP BY pt.category, pt.name
    ) AS a
) AS b
WHERE rn <= 3;
```

Query Output

| category | name | revenue | rn |
|---|---|---|---|
| Chicken | The Thai Chicken Pizza | 43434.25 | 1 |
| Chicken | The Barbecue Chicken Pizza | 42768 | 2 |
| Chicken | The California Chicken Pizza | 41409.5 | 3 |
| Classic | The Classic Deluxe Pizza | 38180.5 | 1 |
| Classic | The Hawaiian Pizza | 32273.25 | 2 |
| Classic | The Pepperoni Pizza | 30161.75 | 3 |
| Supreme | The Spicy Italian Pizza | 34831.25 | 1 |
| Supreme | The Italian Supreme Pizza | 33476.75 | 2 |
| Supreme | The Sicilian Pizza | 30940.5 | 3 |
| Veggie | The Four Cheese Pizza | 32265.70000000065 | 1 |
| Veggie | The Mexicana Pizza | 26780.75 | 2 |
| Veggie | The Five Cheese Pizza | 26066.5 | 3 |

SHODWE
Pizza Resto

# THANK YOU

- SQL CAN RETRIEVE PIZZA INSIGHTS, BUT ONLY MY STOMACH CAN PROCESS THEM!

-RENUKA KULKARNI