Made with Perplexity Labs

# Pizza Bite Sales Analysis

## SQL Database Project

Presented by: Tushar Gaur

Made with Perplexity Labs

# Database Structure

## orders

Contains order information with date and time

## order_details

Links orders to specific pizzas with quantities

## pizzas

Contains pizza pricing information by size

## pizza_types

Contains pizza names, categories, and ingredients

## Table Relationships

**orders** → **order_details** (1:many)

**order_details** → **pizzas** (many:1)

**pizzas** → **pizza_types** (many:1)

Made with Perplexity Labs

# Basic SQL Analysis

Total Orders

21,350

Total Revenue

$817,860.05

Total Pizzas Sold

49,574

Most Common Size

Large

38.2%

Top Pizza by Quantity

The Classic Deluxe Pizza

2,453 sold

Average Order Value

‹ 3 / 16 ›

Made with Perplexity Labs

# Intermediate Analysis

## Pizza Category Distribution

| | |
|---|---|
| Classic | 14,888 pizzas (30.0%) |
| Supreme | 11,987 pizzas (24.2%) |
| Veggie | 11,649 pizzas (23.5%) |
| Chicken | 11,050 pizzas (22.3%) |

## Key Insights

- **Peak Ordering Hour:** 12:00 PM (2,520 orders)
- **Second Peak:** 13:00 PM (2,455 orders)
- **Average Pizzas per Day:** 138.5
- **Total Pizzas Sold:** 49,574

Made with Perplexity Labs

# Advanced Revenue Analysis

## Top 3 Pizzas by Revenue

1

### Thai Chicken Pizza

$43,434.25

5.31% of total revenue

2

### Barbecue Chicken Pizza

$42,768.00

5.23% of total revenue

3

### California Chicken Pizza

$41,409.50

5.06% of total revenue

Made with Perplexity Labs

Share

# Data Visualizations

## Pizza Category Distribution

### Category Distribution

Classic: 14,888 (30.0%)

Supreme: 11,987 (24.2%)

Veggie: 11,649 (23.5%)

Chicken: 11,050 (22.3%)

## Revenue by Category

### Revenue Distribution

Classic: 26.91%

Supreme: 25.46%

Chicken: 23.96%

Veggie: 23.68%

## Key Visual Insights

- Classic pizzas dominate sales with 30% market share
- Revenue distribution is more balanced across categories
- Even distribution across Supreme, Veggie, and Chicken categories
- Classic category leads in both volume and revenue

# Sample SQL Queries

## 1. Total Orders Count

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

Result: 21,350 orders

## 2. Total Revenue Calculation

```
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;
```

Result: $817,860.05

## 3. Top Pizza by Quantity

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

Result: The Classic Deluxe Pizza - 2,453 sold

Made with Perplexity Labs

Share

# SQL Query: Total Orders Count

## 1. Retrieve the total number of orders

```sql
SELECT
    COUNT(Order_details_id)
FROM
    order_details AS Total_Order_Placed;
```

**Result:**

> 48,620 total order details

**Explanation:**

This query counts all records in the order_details table, which represents individual pizza orders. Each record represents one pizza type in an order.

Made with Perplexity Labs

Share

# SQL Query: Top Pizza Types

## 5. List the top pizza types by quantity

```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;
```

### Top Results:

- Classic Deluxe Pizza: 2,453 units
- Hawaiian Pizza: 2,422 units

### Explanation:

This query uses JOIN operations to connect three tables and aggregate pizza quantities by type, showing the most popular pizzas.

Made with Perplexity Labs

Share

# SQL Query: Orders by Hour

## 6. Distribution of orders by hour

```sql
SELECT
    HOUR(Order_time) AS Order_per_hour,
    COUNT(order_id) AS Total_order
FROM
    orders
GROUP BY Order_per_hour;
```

**Peak Hours:**

- 12:00 PM: 2,520 orders
- 13:00 PM: 2,455 orders

**Explanation:**

Using the HOUR() function to extract hour from timestamp and GROUP BY to count orders per hour, revealing lunch time as peak ordering period.

Made with Perplexity Labs

Share

# SQL Query: Category Distribution

## 7. Category-wise distribution

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

### Category Counts:

- Supreme: 9 types
- Veggie: 9 types
- Classic: 8 types
- Chicken: 6 types

### Explanation:

Simple GROUP BY query to count pizza types in each category, showing the variety of offerings across different pizza categories.

# SQL Query: Daily Pizza Quantities

## 8. Group orders by date and calculate pizzas per day

```sql
SELECT
    orders.Order_Date AS Per_Day,
    SUM(order_details.Quantity) AS avg_pizza
FROM
    orders
    JOIN order_details ON orders.order_id = order_details.order_id
GROUP BY Per_Day;
```

**Sample Results:**

- 2015-01-01: 162 pizzas
- 2015-01-02: 165 pizzas
- 2015-01-03: 158 pizzas

**Explanation:**

JOIN operation between orders and order_details tables to aggregate daily pizza quantities, useful for understanding daily demand patterns.

# SQL Query: Top 3 by Revenue

## 9. Determine top 3 pizza types based on revenue

```sql
SELECT
    pizza_types.name, SUM(pizzas.price*order_details.Quantity) AS Total_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 Total_revenue DESC
LIMIT 3;
```

### Top 3 Results:

1. Thai Chicken Pizza: $43,434.25
2. Barbecue Chicken Pizza: $42,768.00
3. California Chicken Pizza: $41,409.50

### Explanation:

Complex JOIN across three tables with revenue calculation (price × quantity), sorting by revenue descending and limiting to top 3 results.

Made with Perplexity Labs

Share

# SQL Query: Revenue Percentage by Category

## 10. Calculate percentage contribution of each category to total revenue

```sql
SELECT
    pizza_types.category,
    ROUND(
        SUM(order_details.Quantity * pizzas.price)
        /
        (SELECT SUM(order_details.Quantity * pizzas.price)
         FROM order_details
         JOIN pizzas ON pizzas.pizza_id = order_details.pizza_id
        ) * 100,
        2) AS revenue_percentage
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_percentage DESC;
```

### Results:

- Classic: 26.91%
- Supreme: 25.46%
- Chicken: 23.96%
- Veggie: 23.68%

### Explanation:

Advanced query using subquery to calculate total revenue and percentage contribution by category. Demonstrates window function concepts.

# SQL Query: Cumulative Revenue Analysis

## 11. Analyze 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 daily_revenue
ORDER BY order_date;
```

### Sample Results:

- Day 1: $2,713.85 (cumulative)
- Day 2: $5,445.75 (cumulative)
- Day 3: $8,108.15 (cumulative)

### Explanation:

Advanced SQL using window functions (OVER) with subquery to calculate running totals. Shows progressive revenue growth over time.

# Thank You

## Contact Information

**Email:** *tushargaur3.2002@gamil.com*

**LinkedIn:** *https://www.linkedin.com/in/tushar-gaur-6848041b2/*

This presentation showcased a comprehensive SQL analysis of Pizza Bite's sales data, demonstrating database querying skills from basic operations to advanced window functions and subqueries.

🍕 **Pizza Bite Sales Analysis** 🍕