The Great Pizza Analytics Challenge

Transforming IDC Pizza’s raw sales data into actionable insights using SQL

Project Objectives

Database Exploration

• Understand IDC_Pizza tables & relationships

Data Quality

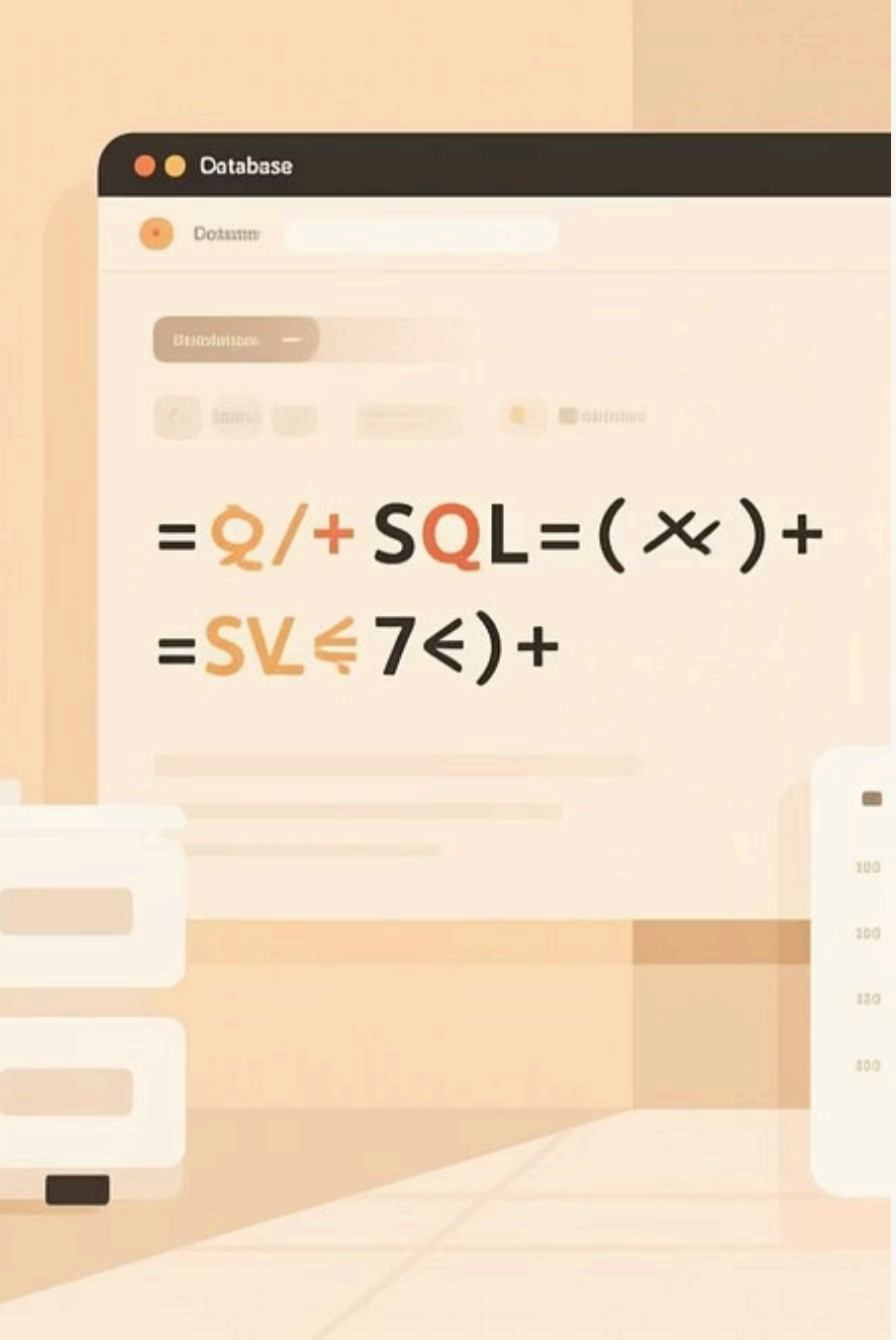
• Identify missing values & clean raw data customer experience through data-driven decision making.

Sales Analysis

• Analyze pizzas, orders, pricing, and category performance

Real-World Application

• Apply SQL concepts to solve practical business questions



Key Tasks Performed

Database Setup

• Installed IDC_Pizza database and reviewed schema

Data Profiling

• Listed unique categories & inspected missing ingredient/price fields

Product Analysis

• Filtered pizzas by size, price range, and name patterns

Order Insights

• Extracted date-wise and time-based order behaviour

Revenue Calculations

• Computed order values and category-level sales

Advanced Analysis

• Identified unordered pizzas

• Compared price differences using self-joins

Skills Demonstrated

Core SQL Techniques

- Filtering (WHERE, LIKE, IN, BETWEEN)
- Aggregations: SUM, AVG, COUNT
- DISTINCT & COALESCE for cleaning
- GROUP BY & HAVING

Advanced SQL Concepts

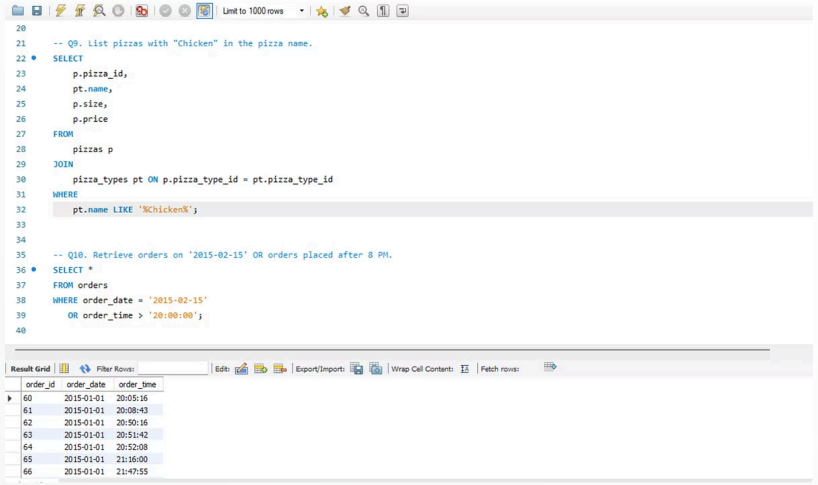
- INNER, LEFT, RIGHT joins
- Self-join for comparative analysis
- Query optimization fundamentals
- Structured analytical problem-solving

```sql
-- Q2. List all unique pizza categories using DISTINCT.
SELECT DISTINCT category from pizza_types;

/* Display pizza_type_id, name, and ingredients, replacing NULL ingredients with "Missing Data".
Show only the first 5 rows.*/

SELECT
    pizza_type_id,
    name,
    IFNULL(ingredients, 'Missing Data') AS ingredients
FROM
    pizza_types
LIMIT 5;

-- Check for pizzas missing a price using IS NULL.
SELECT *
FROM pizzas
WHERE price IS NULL;
```

Phase 1: Foundation & Inspection

Tasks Covered:

- List unique categories
- Replace NULL ingredients
- Identify missing pizza prices
- Inspect table structure



```
-- Q9. List pizzas with "Chicken" in the pizza name.
SELECT
    p.pizza_id,
    pt.name,
    p.size,
    p.price
FROM
    pizzas p
JOIN
    pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
WHERE
    pt.name LIKE '%Chicken%';


-- Q10. Retrieve orders on '2015-02-15' OR orders placed after 8 PM.
SELECT *
FROM orders
WHERE order_date = '2015-02-15'
   OR order_time > '20:00:00';
```

order_id	order_date	order_time
60	2015-01-01	20:05:16
61	2015-01-01	20:08:43
62	2015-01-01	20:50:16
63	2015-01-01	20:51:42
64	2015-01-01	20:52:08
65	2015-01-01	21:16:00
66	2015-01-01	21:47:55

Phase 2: Filtering & Exploration

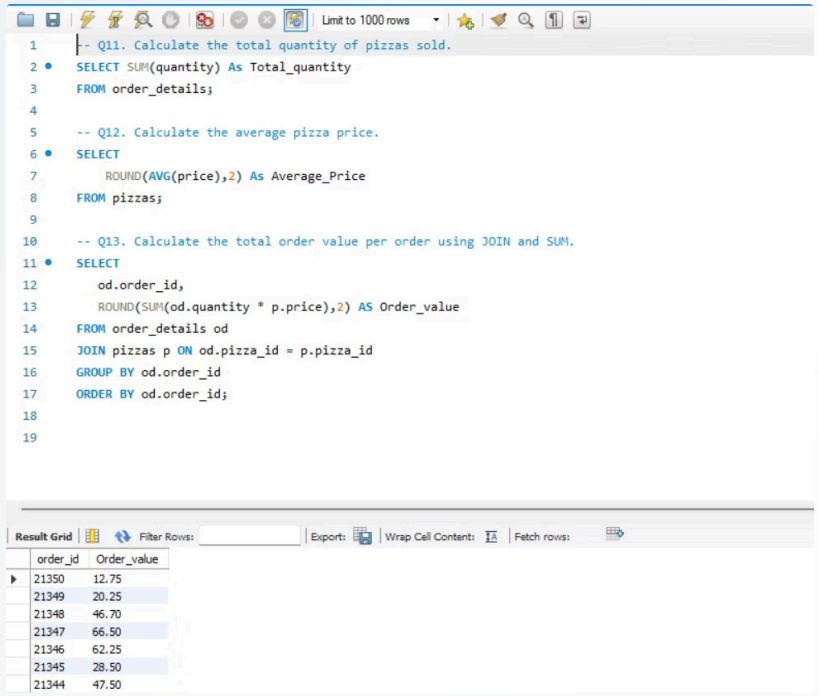
Tasks Covered:

- Orders on specific days
- Price sorting
- Size-based filtering
- Range-based price filtering
- Name contains “Chicken”
- Orders after 8 PM

Phase 3: Sales Performance

Tasks Covered:

- Total pizzas sold
- Average pizza price
- Order value per order
- Category-wise quantity sold



```sql
-- Q11. Calculate the total quantity of pizzas sold.
SELECT SUM(quantity) As Total_quantity
FROM order_details;

-- Q12. Calculate the average pizza price.
SELECT
    ROUND(AVG(price),2) As Average_Price
FROM pizzas;

-- Q13. Calculate the total order value per order using JOIN and SUM.
SELECT
   od.order_id,
   ROUND(SUM(od.quantity * p.price),2) AS Order_value
FROM order_details od
JOIN pizzas p ON od.pizza_id = p.pizza_id
GROUP BY od.order_id
ORDER BY od.order_id;
```

Result Grid

order_id	Order_value
21350	12.75
21349	20.25
21348	46.70
21347	66.50
21346	62.25
21345	28.50
21344	47.50

Phase 3 Continued

Advanced Analytics:

- Categories with >5000 units sold
- Pizzas never ordered
- Price difference across sizes (Self Join)

```sql
-- Sales analysis by category (all included in ONE answer):

-- Total quantity sold per pizza category (SUM,GROUPBY)
SELECT pt.category,SUM(od.quantity) AS quantity_sold
FROM order_details od
JOIN pizzas p
ON od.pizza_id = p.pizza_id
JOIN pizza_types pt
ON p.pizza_type_id = pt.pizza_type_id
group by pt.category;

-- Categories with more than 5,000 pizzas sold
SELECT pt.category,SUM(od.quantity) AS quantity_sold
FROM order_details od
JOIN pizzas p
ON od.pizza_id = p.pizza_id
JOIN pizza_types pt
ON p.pizza_type_id = pt.pizza_type_id
group by pt.category
having quantity_sold > 5000;
```

Result Grid

pizza_type_id	size_1	price_1	size_2	price_2	price_difference
bbq_ckn	L	20.75	M	16.75	-4.00
bbq_ckn	M	16.75	S	12.75	-4.00
bbq_ckn	L	20.75	S	12.75	-8.00
big_meat	L	20.50	M	16.00	-4.50
big_meat	M	16.00	S	12.00	-4.00
big_meat	L	20.50	S	12.00	-8.50
calabrese	L	20.25	M	16.25	-4.00

Key Takeaways

Database Relationships

• Strong understanding of how tables connect

Query Excellence

• Ability to write clean and optimized SQL queries

Data Quality Discipline

• Handling NULLs & cleaning inconsistent values

Actionable Insights

• Extracting trends that impact decision-makin

Independent Problem-Solving

• Solved analytical challenges without guidance

Practice Rigor

• Experienced multi-phase SQL workflow

Project Outcome

A comprehensive SQL-driven analysis pipeline was successfully developed, uncovering key insights across pricing, category performance, and sales behavior. The project strengthened practical data analytics skills by applying real-world SQL techniques—from data cleaning and filtering to complex joins and aggregation-based decision insights.

Lessons Learned

The Foundation of Data Quality

Clean, accurate data is the bedrock of reliable analytics. Identifying and addressing inconsistencies early was crucial for valid insights.

SQL as a Strategic Tool

Beyond querying, SQL proved to be a powerful strategic tool for translating complex business questions into clear, actionable data-driven answers.

Iterative Data Exploration

The project reinforced the value of an iterative approach, allowing for continuous refinement of queries and deeper dives into emerging patterns.

Business Context is Critical

Understanding the intricacies of the pizza business was vital for framing meaningful questions and interpreting findings effectively for strategic decision-making.