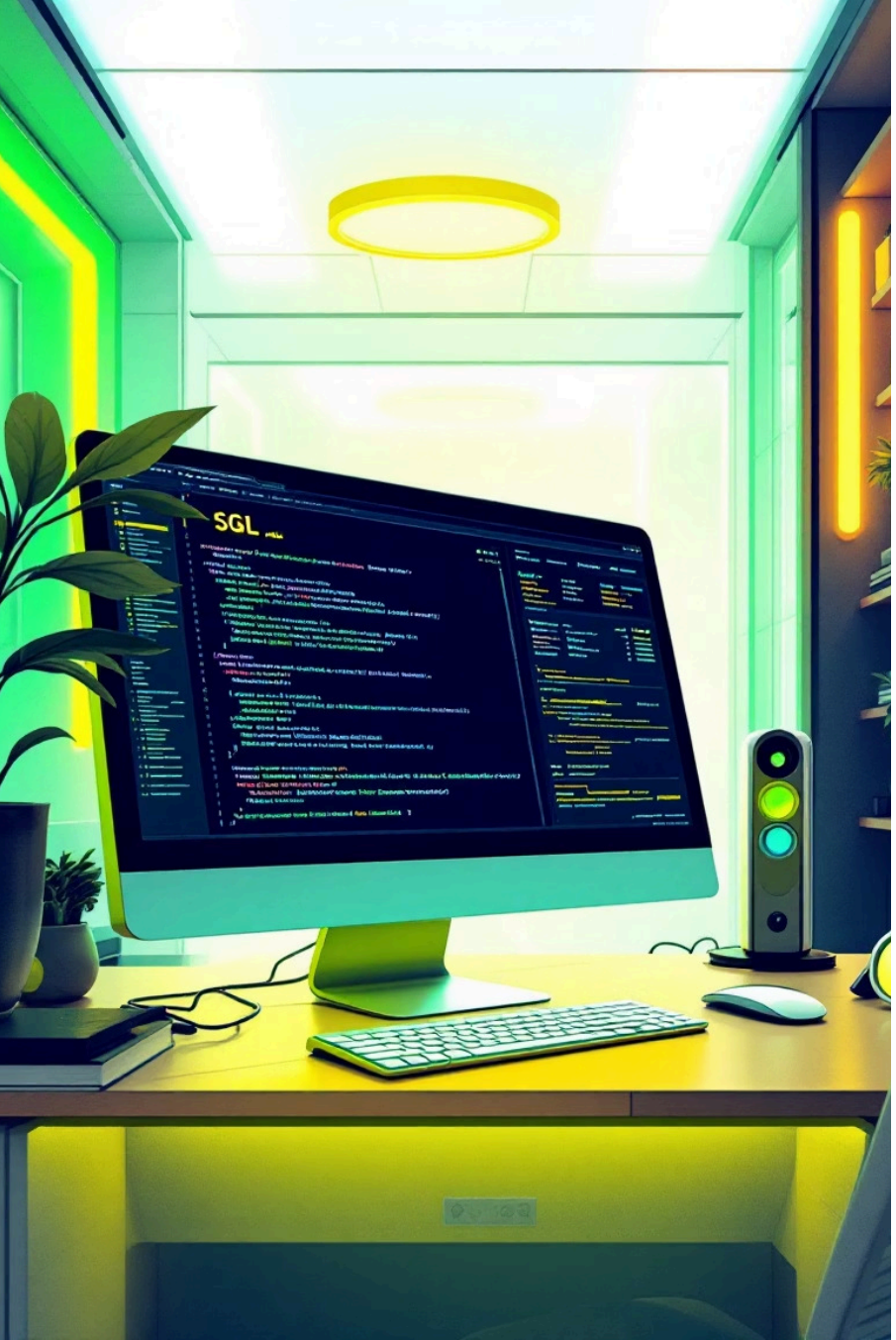


SQL Data Analysis for Zomato

A comprehensive project demonstrating advanced SQL problem-solving skills through analysis of food delivery data. From database setup to complex business insights, this project showcases 20 real-world solutions.

Sample SQL Queries: Zomato Analysis

Explore real-world examples of SQL queries used to extract valuable insights from the Zomato dataset, demonstrating problem-solving across customer behavior, operational efficiency, and market trends.

Top 5 Customers by Total Spending

This query identifies the customers who have spent the most money on orders, helping to pinpoint high-value patrons.

```
SELECT
    c.customer_id,
    c.customer_name,
    SUM(o.order_total) AS total_spent
FROM
    Customers c
JOIN
    Orders o ON c.customer_id = o.customer_id
GROUP BY
    c.customer_id, c.customer_name
ORDER BY
    total_spent DESC
LIMIT 5;
```

Average Delivery Time per Restaurant

Understanding average delivery times for each restaurant can help optimize logistics and improve customer satisfaction.

```
SELECT
    r.restaurant_name,
    AVG(JULIANDAY(o.delivery_time) - JULIANDAY(o.order_time)) AS avg_delivery_minutes
FROM
    Restaurants r
JOIN
    Orders o ON r.restaurant_id = o.restaurant_id
WHERE
    o.delivery_time IS NOT NULL AND o.order_time IS NOT NULL
GROUP BY
    r.restaurant_name
ORDER BY
    avg_delivery_minutes ASC;
```

Most Popular Cuisines by Order Count

This query reveals which cuisines are most frequently ordered, informing menu development and marketing strategies.

```
SELECT
    m.cuisine_type,
    COUNT(o.order_id) AS total_orders
FROM
    Menu_Items m
JOIN
    Order_Items oi ON m.item_id = oi.item_id
JOIN
    Orders o ON oi.order_id = o.order_id
GROUP BY
    m.cuisine_type
ORDER BY
    total_orders DESC
LIMIT 5;
```

20 Business Problems Solved with Zomato SQL Analysis

This project tackles a wide array of business challenges through detailed SQL analysis, providing actionable insights across key operational and strategic areas for Zomato. Each problem uncovers critical information to drive decision-making and optimize performance.

SQL Query Solutions for Business Problems

Here are comprehensive SQL query solutions addressing all 20 identified business problems. Each query is designed to extract actionable insights from Zomato's operational data, providing a robust analytical toolkit for decision-makers.

---

1. Identifying High-Value Customers

This query identifies the customers who have spent the most money on orders, helping to pinpoint high-value patrons for targeted loyalty programs.

```
SELECT
    c.customer_id,
    c.customer_name,
    SUM(o.order_total) AS total_spent
FROM
    Customers c
JOIN
    Orders o ON c.customer_id = o.customer_id
GROUP BY
    c.customer_id, c.customer_name
ORDER BY
    total_spent DESC
LIMIT 5;
```

---

2. Optimizing Restaurant Delivery Times

Understanding average delivery times for each restaurant can help optimize logistics and improve customer satisfaction by identifying bottlenecks.

```
SELECT
    r.restaurant_name,
    AVG((JULIANDAY(o.delivery_time) - JULIANDAY(o.order_time)) * 24 * 60) AS avg_delivery_minutes
FROM
    Restaurants r
JOIN
    Orders o ON r.restaurant_id = o.restaurant_id
WHERE
    o.delivery_time IS NOT NULL AND o.order_time IS NOT NULL
GROUP BY
    r.restaurant_name
ORDER BY
    avg_delivery_minutes ASC;
```

---

3. Understanding Cuisine Popularity

This query reveals which cuisines are most frequently ordered, informing menu development, marketing strategies, and restaurant onboarding decisions.

```
SELECT
    mi.cuisine_type,
    COUNT(oi.order_id) AS total_orders
FROM
    Menu_Items mi
JOIN
    Order_Items oi ON mi.item_id = oi.item_id
JOIN
    Orders o ON oi.order_id = o.order_id
GROUP BY
    mi.cuisine_type
ORDER BY
    total_orders DESC
LIMIT 5;
```

---

4. Analyzing Customer Order Frequency

Identifies customers with the highest number of orders to understand repeat purchase behavior and foster stronger customer relationships.

```
SELECT
    c.customer_id,
    c.customer_name,
    COUNT(o.order_id) AS order_count
FROM
    Customers c
JOIN
    Orders o ON c.customer_id = o.customer_id
GROUP BY
    c.customer_id, c.customer_name
ORDER BY
    order_count DESC
LIMIT 5;
```

---

5. Assessing Average Order Value

Calculates the average spending per customer to identify opportunities for upselling, cross-selling, and optimizing pricing strategies.

```
SELECT
    c.customer_id,
    c.customer_name,
    AVG(o.order_total) AS avg_order_value
FROM
    Customers c
JOIN
    Orders o ON c.customer_id = o.customer_id
GROUP BY
    c.customer_id, c.customer_name
ORDER BY
    avg_order_value DESC
LIMIT 5;
```

---

6. Measuring Customer Retention

Determines the percentage of customers who have placed more than one order, providing a basic measure of customer loyalty and retention.

```
WITH CustomerOrderCounts AS (
    SELECT
        customer_id,
        COUNT(order_id) AS total_orders
    FROM
        Orders
    GROUP BY
        customer_id
)
SELECT
    (CAST(SUM(CASE WHEN total_orders > 1 THEN 1 ELSE 0 END) AS REAL) * 100.0 / COUNT(customer_id)) AS retention_rate_percentage
FROM
    CustomerOrderCounts;
```

---

7. Evaluating Restaurant Revenue Performance

Ranks restaurants by their total revenue generated, identifying top performers and potential areas for growth or intervention within the platform.

```
SELECT
    r.restaurant_name,
    SUM(o.order_total) AS total_revenue
FROM
    Restaurants r
JOIN
    Orders o ON r.restaurant_id = o.restaurant_id
GROUP BY
    r.restaurant_name
ORDER BY
    total_revenue DESC
LIMIT 5;
```

---

8. Monitoring Restaurant Rating Trends

Identifies restaurants with consistently high or low average ratings, crucial for quality control and addressing customer feedback effectively.

```
SELECT
    r.restaurant_name,
    AVG(ra.rating) AS average_rating
FROM
    Restaurants r
JOIN
    Ratings ra ON r.restaurant_id = ra.restaurant_id
GROUP BY
    r.restaurant_name
ORDER BY
    average_rating DESC
LIMIT 5;
```

---

9. Identifying Menu Item Diversity

Analyzes the number of unique menu items offered by restaurants, highlighting opportunities for menu expansion or simplification to meet market demands.

```
SELECT
    r.restaurant_name,
    COUNT(DISTINCT mi.item_id) AS unique_menu_items
FROM
    Restaurants r
JOIN
    Menu_Items mi ON r.restaurant_id = mi.restaurant_id
GROUP BY
    r.restaurant_name
ORDER BY
    unique_menu_items DESC
LIMIT 5;
```

---

10. Detecting Underperforming Restaurants

Flags restaurants with low order volumes or revenue over a specific period to provide targeted support or re-evaluate partnerships.

```
SELECT
    r.restaurant_name,
    COUNT(o.order_id) AS order_count,
    SUM(o.order_total) AS total_revenue
FROM
    Restaurants r
LEFT JOIN
    Orders o ON r.restaurant_id = o.restaurant_id
WHERE
    o.order_date BETWEEN '2023-01-01' AND '2023-03-31' -- Example quarter
GROUP BY
    r.restaurant_name
HAVING
    COUNT(o.order_id) < 50 -- Example threshold for low orders
ORDER BY
    order_count ASC
LIMIT 5;
```

---

11. Mapping Cuisine Popularity by Location

Understands which cuisines are popular in specific geographic regions, optimizing restaurant placement and localized marketing efforts.

```
SELECT
    c.city,
    mi.cuisine_type,
    COUNT(oi.order_id) AS total_orders
FROM
    Customers c
JOIN
    Orders o ON c.customer_id = o.customer_id
JOIN
    Order_Items oi ON o.order_id = oi.order_id
JOIN
    Menu_Items mi ON oi.item_id = mi.item_id
GROUP BY
    c.city, mi.cuisine_type
ORDER BY
    c.city, total_orders DESC;
```

---

12. Identifying Peak Delivery Hours

Determines the busiest times for deliveries, which helps in efficient allocation of delivery personnel and resources to meet demand.

```
SELECT
    STRFTIME('%H', order_time) AS order_hour,
    COUNT(order_id) AS total_orders
FROM
    Orders
GROUP BY
    order_hour
ORDER BY
    total_orders DESC
LIMIT 5;
```

---

13. Analyzing Order Delay Incidents

Pinpoints orders that exceed a defined average delivery time threshold to investigate root causes and implement corrective measures.

```
SELECT
    o.order_id,
    r.restaurant_name,
    o.order_time,
    o.delivery_time,
    (JULIANDAY(o.delivery_time) - JULIANDAY(o.order_time)) * 24 * 60 AS actual_delivery_minutes
FROM
    Orders o
JOIN
    Restaurants r ON o.restaurant_id = r.restaurant_id
WHERE
    o.delivery_time IS NOT NULL
    AND (JULIANDAY(o.delivery_time) - JULIANDAY(o.order_time)) * 24 * 60 > 45 -- Example: orders taking over 45 minutes
ORDER BY
    actual_delivery_minutes DESC
LIMIT 5;
```

---

14. Assessing Delivery Driver Efficiency

Measures the average number of deliveries per driver, or average delivery time, to optimize routing and improve overall delivery network performance.

```
SELECT
    dd.driver_id,
    dd.driver_name,
    COUNT(o.order_id) AS total_deliveries,
    AVG((JULIANDAY(o.delivery_time) - JULIANDAY(o.pickup_time)) * 24 * 60) AS avg_delivery_duration_minutes
FROM
    Delivery_Drivers dd
JOIN
    Orders o ON dd.driver_id = o.driver_id
WHERE
    o.delivery_time IS NOT NULL AND o.pickup_time IS NOT NULL
GROUP BY
    dd.driver_id, dd.driver_name
ORDER BY
    total_deliveries DESC, avg_delivery_duration_minutes ASC
LIMIT 5;
```

---

15. Tracking Monthly/Quarterly Revenue Growth

Monitors revenue trends over time, providing a clear picture of business growth and financial health for strategic planning.

```
SELECT
    STRFTIME('%Y-%m', order_date) AS month,
    SUM(order_total) AS monthly_revenue
FROM
    Orders
GROUP BY
    month
ORDER BY
    month;
```

---

16. Determining Most Popular Order Days

Identifies the days of the week with the highest order volumes, allowing adjustments to staffing, promotions, and restaurant availability.

```
SELECT
    STRFTIME('%w', order_date) AS day_of_week_num, -- 0 for Sunday, 1 for Monday, etc.
    CASE STRFTIME('%w', order_date)
        WHEN '0' THEN 'Sunday'
        WHEN '1' THEN 'Monday'
        WHEN '2' THEN 'Tuesday'
        WHEN '3' THEN 'Wednesday'
        WHEN '4' THEN 'Thursday'
        WHEN '5' THEN 'Friday'
        WHEN '6' THEN 'Saturday'
    END AS day_of_week,
    COUNT(order_id) AS total_orders
FROM
    Orders
GROUP BY
    day_of_week_num, day_of_week
ORDER BY
    total_orders DESC;
```

---

17. Analyzing New Customer Acquisition Trends

Tracks the growth of new customers over specific periods, helping evaluate marketing campaign effectiveness and market penetration.

```
SELECT
    STRFTIME('%Y-%m', customer_since) AS acquisition_month,
    COUNT(customer_id) AS new_customers
FROM
    Customers
GROUP BY
    acquisition_month
ORDER BY
    acquisition_month;
```

---

18. Deconstructing Revenue by Cuisine Type

Breaks down total revenue by different cuisine categories, helping understand which food types are most profitable and popular by monetary value.

```
SELECT
    mi.cuisine_type,
    SUM(oi.order_total) AS total_revenue_by_cuisine
FROM
    Orders o
JOIN
    Order_Items oi ON o.order_id = oi.order_id
JOIN
    Menu_Items mi ON oi.item_id = mi.item_id
GROUP BY
    mi.cuisine_type
ORDER BY
    total_revenue_by_cuisine DESC
LIMIT 5;
```

---

19. Evaluating Promotion Effectiveness

Assesses the impact of discounts and promotional offers on order volume and revenue generation to optimize future campaigns.

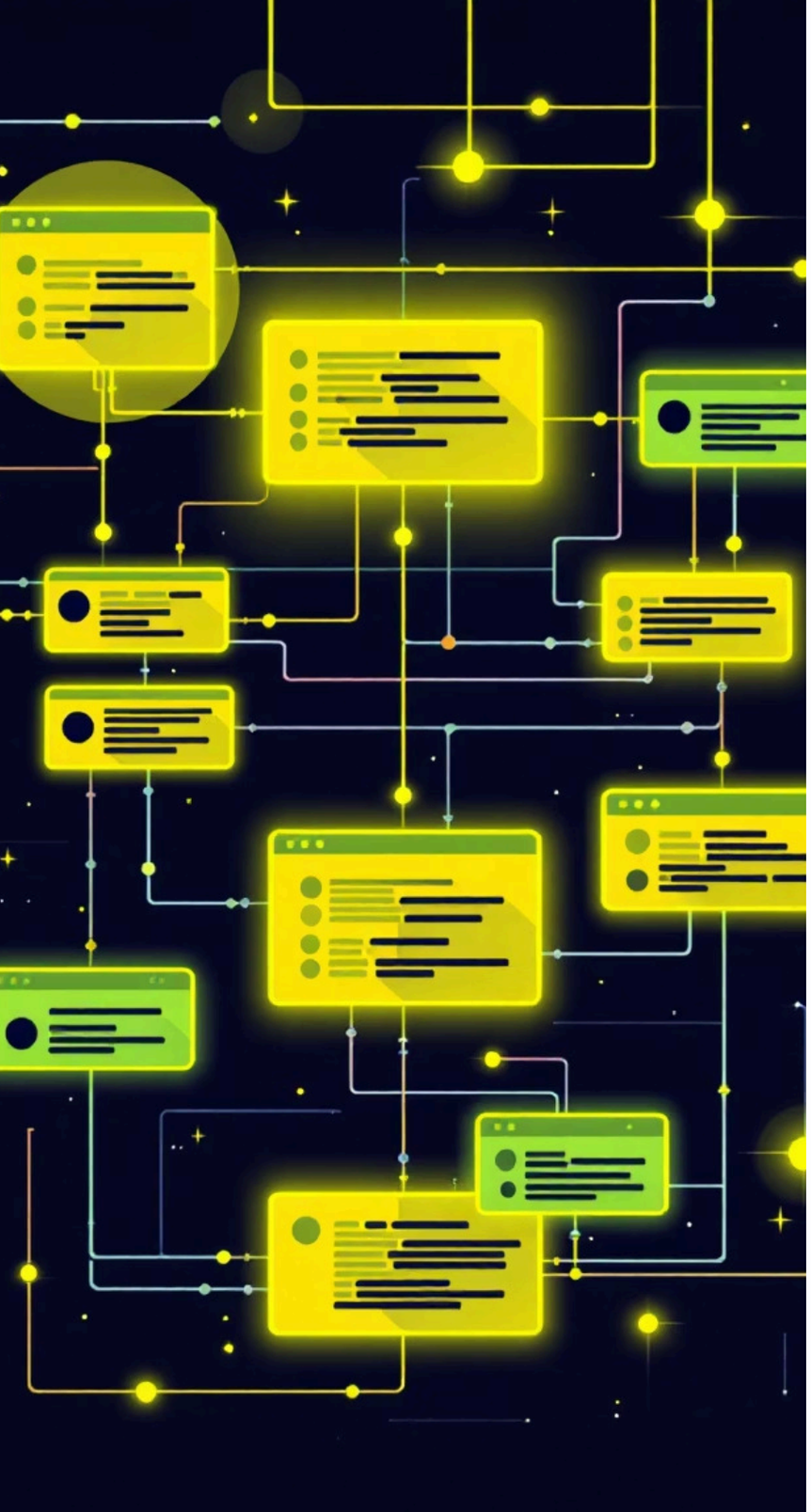
```
SELECT
    p.promotion_name,
    COUNT(o.order_id) AS orders_with_promotion,
    SUM(o.order_total) AS revenue_from_promotion
FROM
    Promotions p
JOIN
    Orders o ON p.promotion_id = o.promotion_id
WHERE
    o.order_date BETWEEN '2023-01-01' AND '2023-03-31' -- Example period
GROUP BY
    p.promotion_name
ORDER BY
    revenue_from_promotion DESC;
```

---

20. Forecasting Peak Order Volume

Predicts hours with the highest order influx to proactively manage restaurant capacity, delivery fleet, and customer service resources.

```
SELECT
    STRFTIME('%Y-%m-%d %H', order_time) AS order_hour_interval,
    COUNT(order_id) AS order_count
FROM
    Orders
GROUP BY
    order_hour_interval
ORDER BY
    order_count DESC
LIMIT 10;
```



Project Architecture

01

Database Setup

Created zomato_db with five interconnected tables: restaurants, customers, riders, orders, and deliveries.

02

Data Import

Inserted sample data across all tables with proper foreign key relationships and constraints.

03

Data Cleaning

Handled null values and ensured data integrity using COALESCE and validation checks.

04

Business Analysis

Solved 20 complex business problems using advanced SQL techniques and window functions.

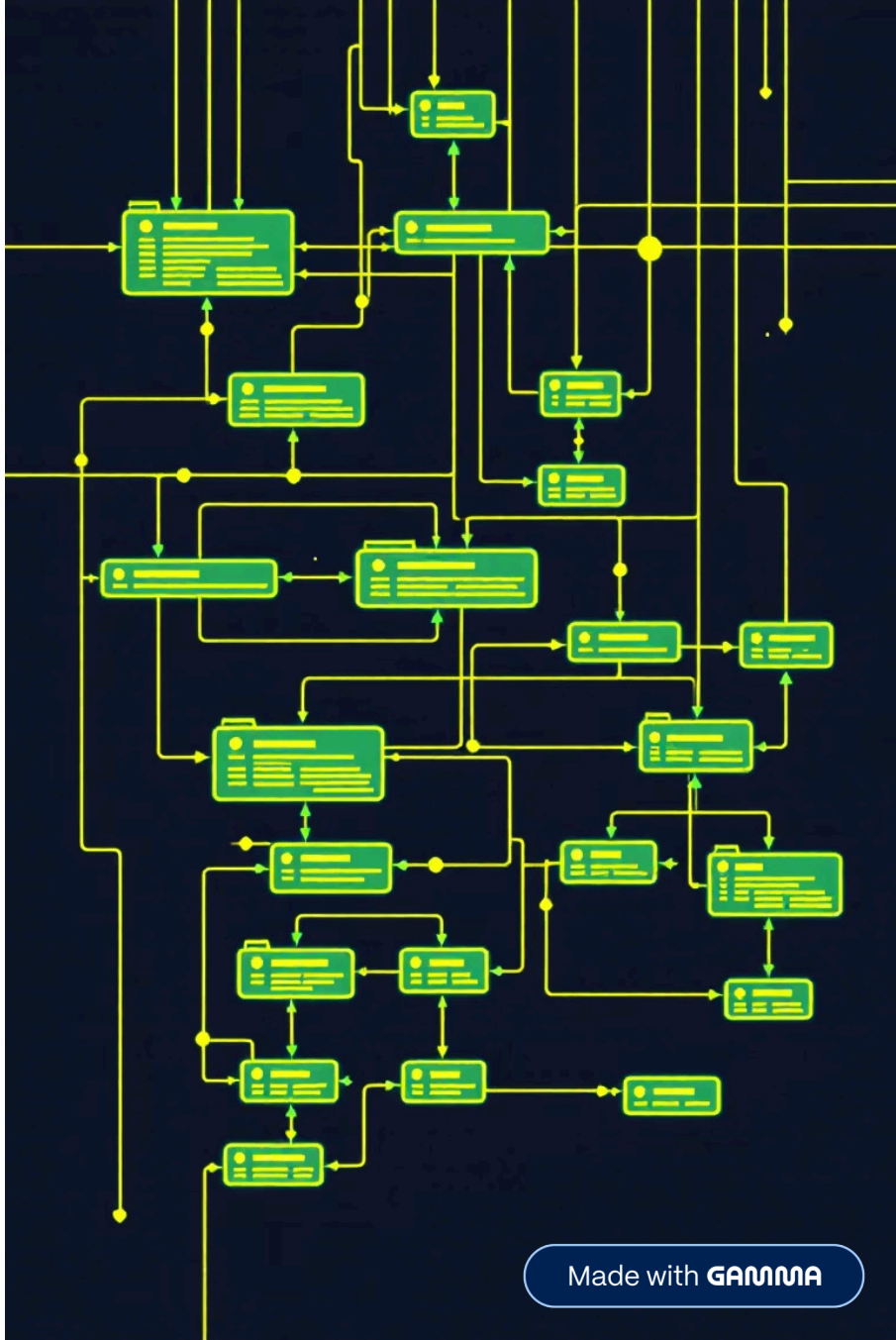
Database Schema

Core Tables

- Restaurants: ID, name, city, opening hours
- Customers: ID, name, registration date
- Riders: ID, name, sign-up date

Transaction Tables

- Orders: ID, customer, restaurant, items, date/time, status, amount
- Deliveries: ID, order, status, time, rider



Customer Insights

Top Dishes Analysis

Identified most frequently ordered dishes by customer "Arjun Mehta" in the last year using DENSE_RANK and date filtering.

High-Value Customers

Listed customers spending over 100K total using aggregation and HAVING clauses for revenue thresholds.

Order Value Analysis

Calculated average order value for customers with 750+ orders to identify premium segments.

Time-Based Analysis

Peak Time Slots

Identified busiest 2-hour intervals using EXTRACT and FLOOR functions on order timestamps.

1

Weekly Patterns

2

Analyzed order frequency by day of week to identify peak days for each restaurant.

Monthly Trends

3

Compared month-over-month sales using LAG window function to track growth patterns.

Seasonal Demand

4

Tracked item popularity across seasons (Spring, Summer, Winter) to identify demand spikes.

Restaurant Performance

Revenue Ranking

Ranked restaurants by total revenue within each city using PARTITION BY and RANK functions for last year's data.

Popular Dishes by City

Identified most ordered dish in each city based on order count, revealing regional preferences.

Cancellation Rates

Compared order cancellation rates between 2023 and 2024 using CTEs to calculate year-over-year changes.

20

Business Problems

Solved using SQL

5

Core Tables

In database

Delivery Operations

1

Undelivered Orders

Found orders placed but not delivered using LEFT JOIN and NULL checks across restaurants.

2

Rider Efficiency

Calculated average delivery times using EXTRACT(EPOCH) to convert time differences to minutes.

3

Rating System

Assigned 5-star (under 15 min), 4-star (15-20 min), or 3-star (over 20 min) ratings based on delivery speed.

4

Monthly Earnings

Calculated rider earnings at 8% of order amounts, grouped by month for compensation tracking.

Advanced SQL Techniques

Window Functions

Leveraged RANK, DENSE_RANK, LAG, and PARTITION BY for complex rankings and comparisons across data segments.

Common Table Expressions

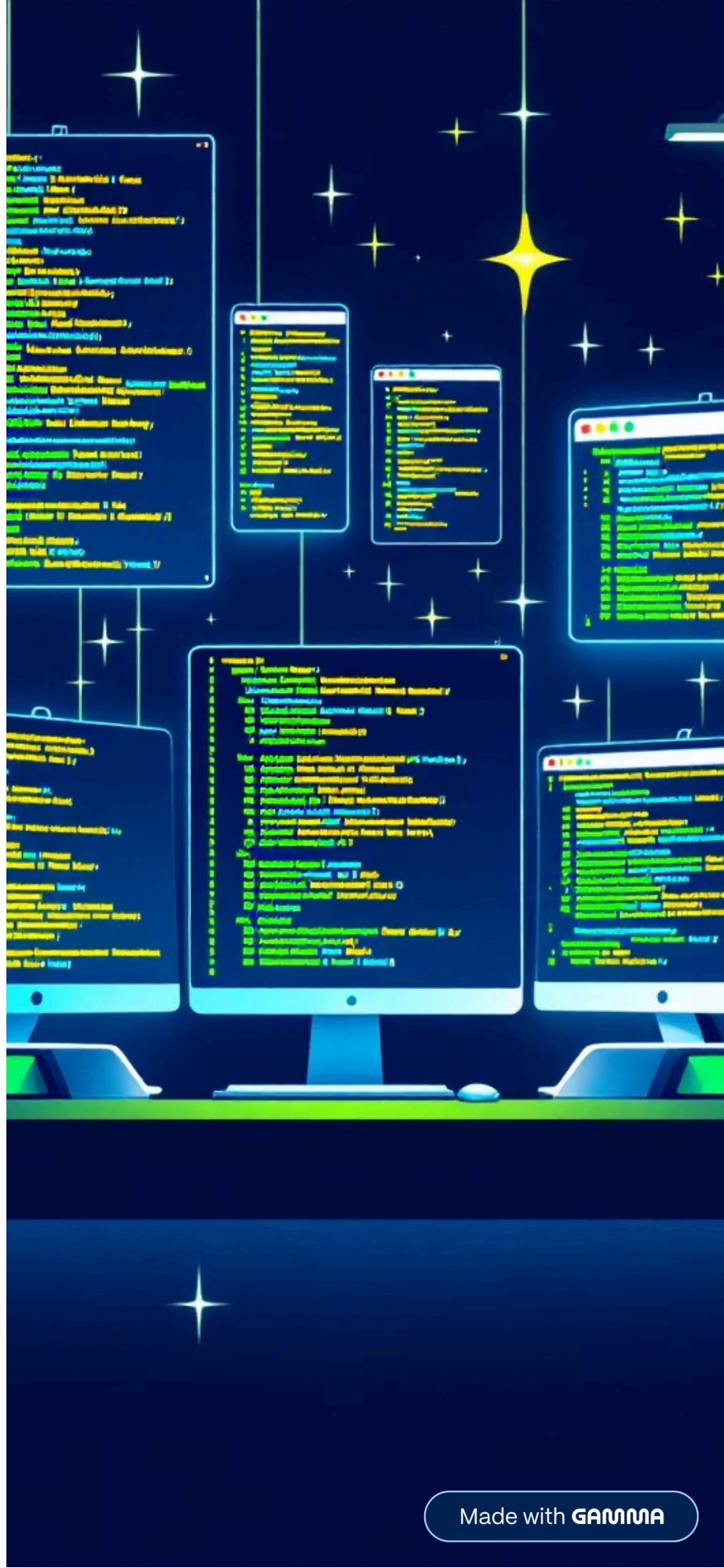
Used CTEs to break down complex queries into readable, maintainable components for multi-step analysis.

Advanced Joins

Applied LEFT JOIN, INNER JOIN, and subqueries to combine data across multiple tables efficiently.

Date Manipulation

Extracted year, month, day components and calculated intervals using EXTRACT and date arithmetic functions.



Customer Segmentation & Growth

Gold vs Silver Customers

Segmented customers based on spending above or below average order value (AOV). Gold customers exceed AOV; Silver customers fall below.

- Total orders per segment
- Total revenue per segment
- Lifetime value calculation

Restaurant Growth Ratio

Calculated monthly growth using LAG function to compare current month orders against previous month for delivered orders only.

Customer Churn

Identified customers who ordered in 2023 but not in 2024 using NOT IN subquery.

Key Takeaways

Comprehensive Analysis

Demonstrated ability to solve 20 diverse business problems from customer behavior to operational efficiency.

Advanced SQL Mastery

Showcased expertise in window functions, CTEs, complex joins, and date manipulation techniques.

Real-World Application

Provided actionable insights for food delivery operations including revenue, delivery performance, and customer segmentation.

Notice: All customer names and data are computer-generated for educational purposes only. This project does not represent real Zomato data.