Restaurant Food Ordering System Using SQLite Database Project Report

Student Name: Yasaswini Sure

1. Introduction

The aim of this project was to design and implement a complete relational database that replicates the operational workflow of a modern online food-ordering platform. To achieve this, I developed a fully functional Restaurant Food Ordering System using SQLite as the backend database engine and Python for automated data generation. The intention was not only to satisfy the assignment requirements such as incorporating all four data measurement scales and demonstrating foreign and compound keys but also to build a realistic dataset that supports meaningful analytical tasks.

The simulated environment models the interactions between restaurants, customers, delivery partners, menu catalogues, and order-level transactions. In total, the system consists of 12 restaurants, 600 customers, 80 delivery partners, 267 menu items, 1200 orders, 3607 order items, and 1095 payment records. These volumes were intentionally selected to mirror the scale of real food-delivery platforms, ensuring diversity in behavioural patterns while keeping the system computationally efficient.

Screenshots taken from DB Browser for SQLite (Figures 1- 5) demonstrate that the database was successfully created, populated, and validated. The following sections describe the schema design choices, the data generation approach, and the insights derived from exploratory SQL analyses.

2. Database Schema Design

The final schema consists of seven interconnected tables, each representing a key component of the food-ordering workflow. The design follows relational database principles, ensuring normalisation, referential integrity, and analytical usability.

Core Tables

- **restaurants** - Stores restaurant identity, cuisine, location, and activity status.
- **customers** - Captures customer details, registration dates, and their city of residence.
- **delivery_partners** - Represents riders along with their vehicle type and ordinal rating.
- **menu_items** - Contains the full menu catalogue, mapped to restaurants via foreign keys.
- **orders** - Records each order placed, including timestamps, totals, and delivery details.
- **order_items** - A compound-key table linking individual menu items to each order.
- **payments** - Logs payment outcomes, modes, and timestamps tied to completed orders.

Use of Data Types To fulfil the assignment requirements:

- **Nominal data**: names, categories, cities, vehicle types
- **Ordinal data**: rider ratings (1-5), loyalty levels
- **Interval data**: order and payment timestamps
- **Ratio data**: prices, totals, subtotals, discount values

Design Rationale

The schema is structured to maintain 3NF compliance, eliminate redundancy, and preserve data integrity using foreign keys and CHECK constraints. This organisation supports a wide range of analytical operations such as menu-level item frequency, restaurant revenue tracking, customer behaviour analysis, and delivery partner performance evaluation.

Figure 1 summarises the table structures and record counts, reflecting successful creation and population of each entity.

3. Data Generation and Validation

All data were programmatically generated using Python's built-in libraries such as random, datetime, and SQLite's connector. The goal was to create a dataset that behaves like real customer activity while avoiding privacy concerns.

Realism in Data Generation

To simulate true platform behaviour, several strategies were used:

- Cuisine styles were chosen based on common restaurant categories.
- Customer names were constructed using combinations of first and last names.
- Delivery partner ratings follow a skew typical in gig-economy platforms.
- Menu pricing patterns varied logically by cuisine (e.g., desserts cheaper than mains).
- Order timestamps were distributed across realistic daily activity windows.
- Discounts, fees, and refund probabilities were incorporated to mimic variability.

Figure 2 displays sample records from the orders table, showing diverse order values, randomised timings, and heterogeneous payment statuses.

Validation Steps

Ensuring correctness was a core part of development. Several validation checks were used:

- Foreign key enforcement enabled (PRAGMA foreign_keys = ON).
- Cross-table row counts checked for consistency (e.g., every order has at least one item).
- CHECK constraints prevented negative prices or invalid ratings.
- Payment entries were generated only for valid order outcomes (Paid or Refunded).
- Manual inspection of sample rows confirmed logical distribution and integrity.

Figures 3-5 showcase the outputs of validation queries and confirm that the relationships between entities operate as intended.

4. Analytical Queries and Insights

A series of SQL queries were executed to examine ordering patterns, restaurant performance, and menu-level behaviour. These analyses demonstrate the practical value of the system and confirm the reliability of the generated data.

4.1 Orders per Restaurant (Figure 3)

Order distribution varies noticeably between restaurants. Pizza Hub records the highest number of orders (115), suggesting strong menu appeal or favourable location. This variation aligns with real-world marketplace dynamics and confirms correct linking between restaurant IDs and orders.

4.2 Most Frequently Ordered Items (Figure 4)

Popular dishes include Veg Noodles (29 orders) and Tom Yum Soup (27 orders). Item-level popularity tends to correlate with cuisine availability and price sensitivity. This query verifies accuracy in the many-to-one relationship between order_items and menu_items.

4.3 Revenue Summary (Figure 5)

Revenue analysis highlights that Pizza Hub, Dragon Wok, and Bangkok Bites each generated more than £5,000 in total earnings. Average order values fall between £38 and £57, indicating a healthy order composition. The results confirm correct computation of totals using subtotals, discounts, and fees.

These analyses illustrate how the database can support strategic insights such as menu optimisation, dynamic pricing, and restaurant-level forecasting.

5. Discussion

The database fulfils all technical and academic criteria and provides a realistic, extensible foundation for further analysis. While the dataset is synthetic, its structure and variability closely resemble real food-delivery platforms. The relational modelling supports a wide range of analytical approaches, including:

- identifying customer purchasing patterns,
- evaluating high-performing restaurants,
- assessing delivery partner performance,
- modelling demand fluctuations.

One limitation is that the current dataset assumes static menu prices and constant delivery partner availability. Real systems often include dynamic pricing or peak-time adjustments. These aspects could be incorporated in future iterations to enhance realism. A major strength of this project is the complete transparency and control offered by synthetic data generation, which ensures full ethical compliance and avoids the need for anonymisation. The modular schema also enables straightforward extensions such as promotional systems, customer reward models, or delivery-time prediction models.

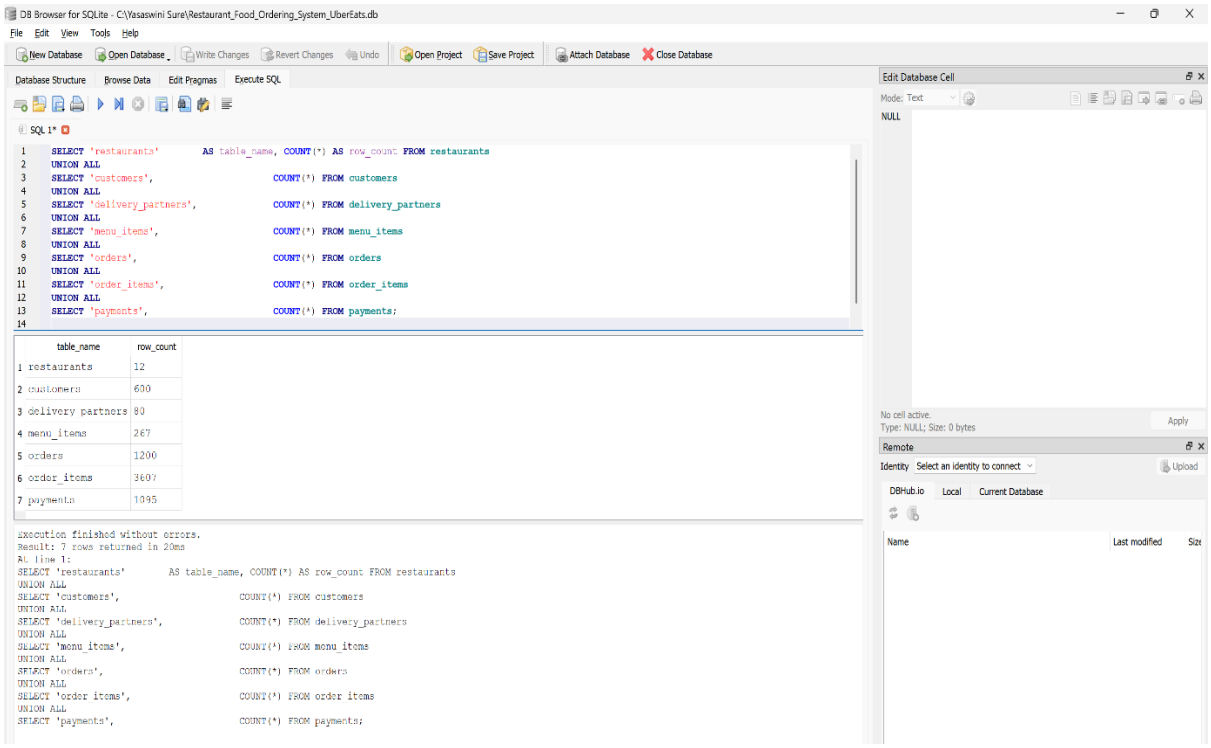
6. Conclusion

This project demonstrates the successful design and implementation of a comprehensive Restaurant Food Ordering System using SQLite and Python. Through careful schema planning, systematic data generation, and rigorous validation, a functional and analytically rich database was produced. With over 7,800 interconnected records, the system captures the essential interactions of a food-delivery ecosystem and supports meaningful SQL-based exploration. The database is academically robust, technically sound, and scalable for future development. It can easily be extended into a dashboard, integrated into machine-learning workflows, or deployed to cloud platforms for real-time analysis. Overall, the project reflects a high standard of database engineering appropriate for MSc Data Science coursework.

Appendix:

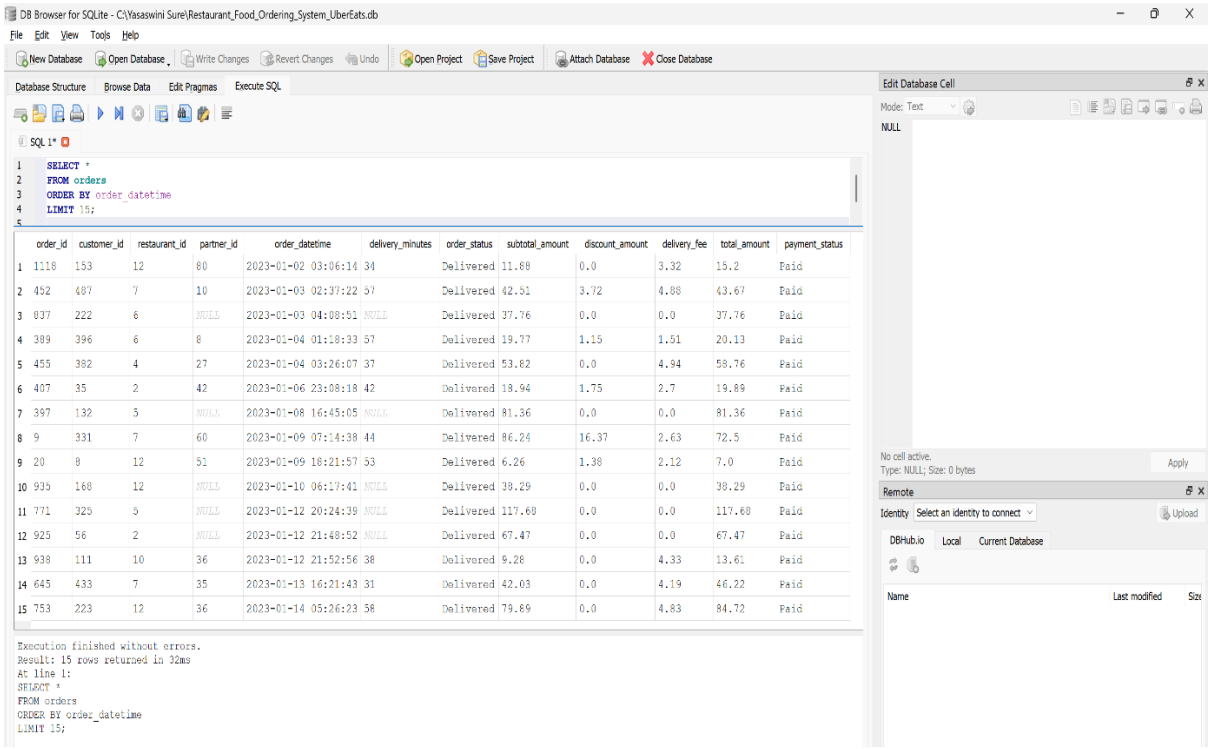
Row Count Query

Figure X: Summary of row counts across all tables confirming successful data generation and population



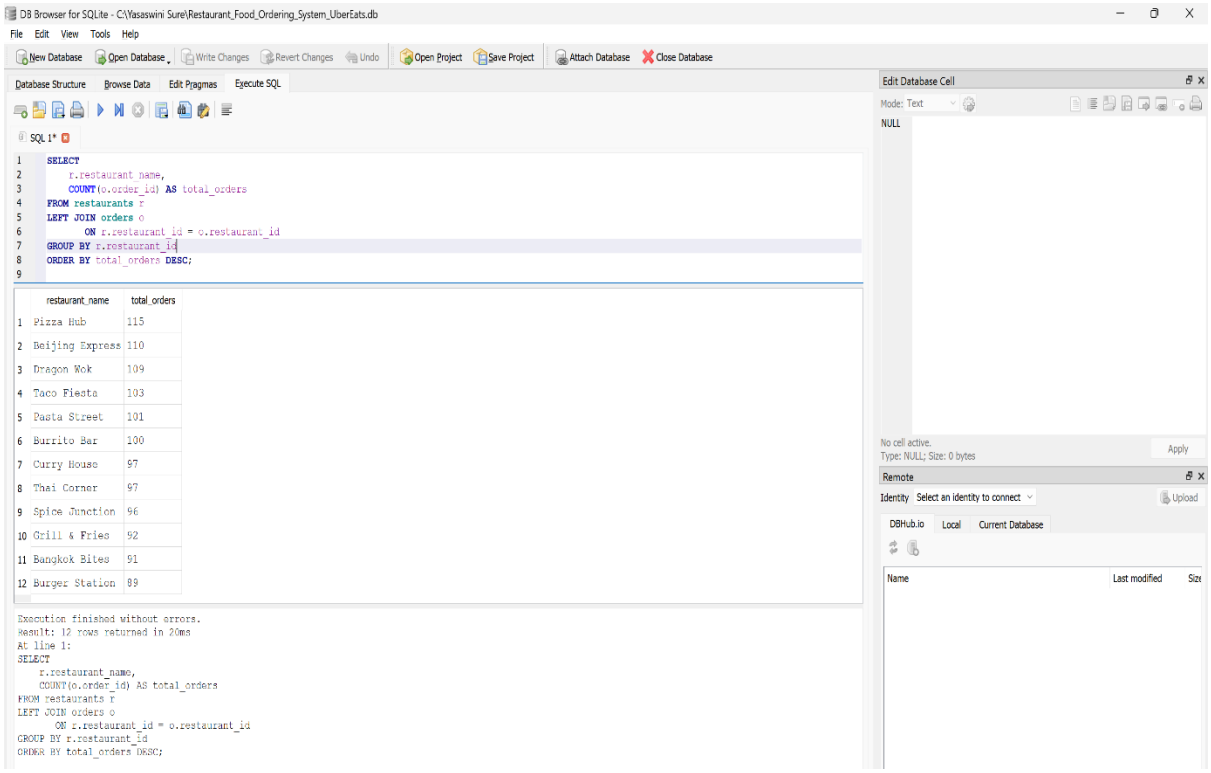
Orders Table Preview

Figure X: Sample of the earliest generated orders showing timestamps, delivery details, and payment statuses.



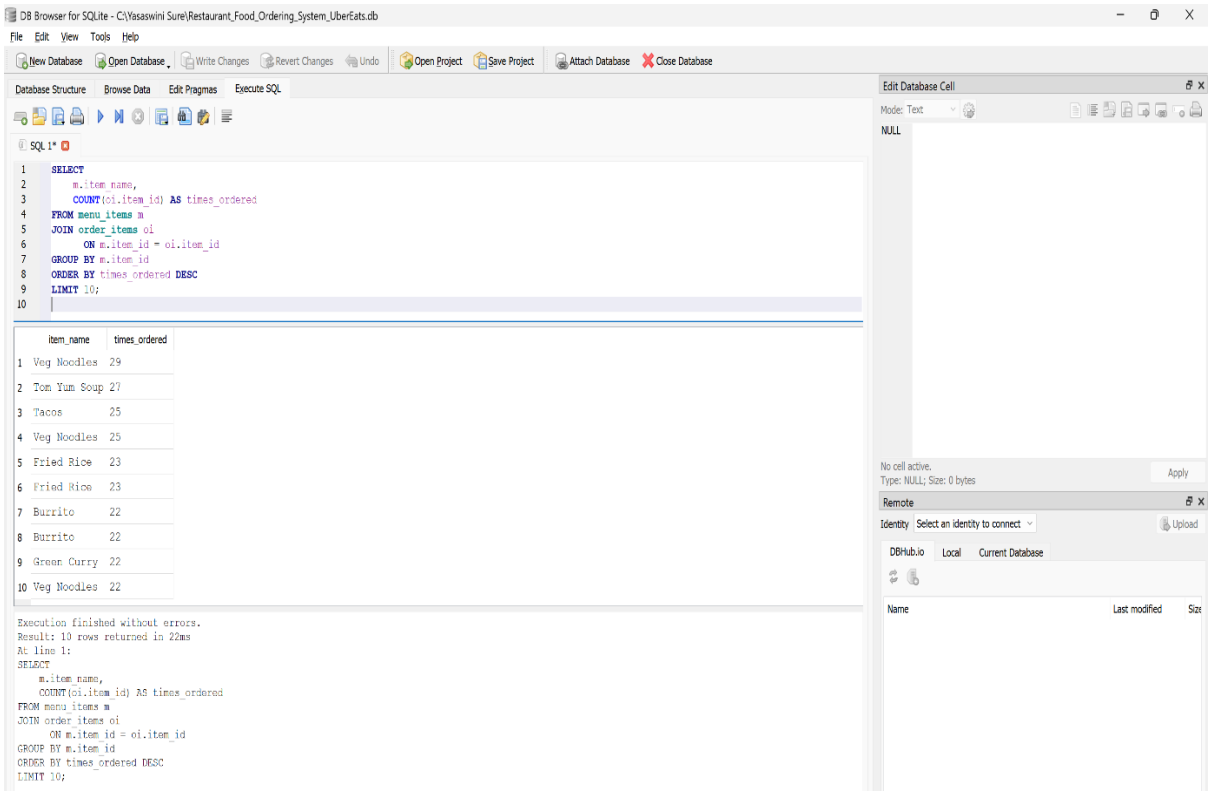
Orders per Restaurant

Figure X: Restaurant-level demand analysis showing total orders received by each restaurant partner.



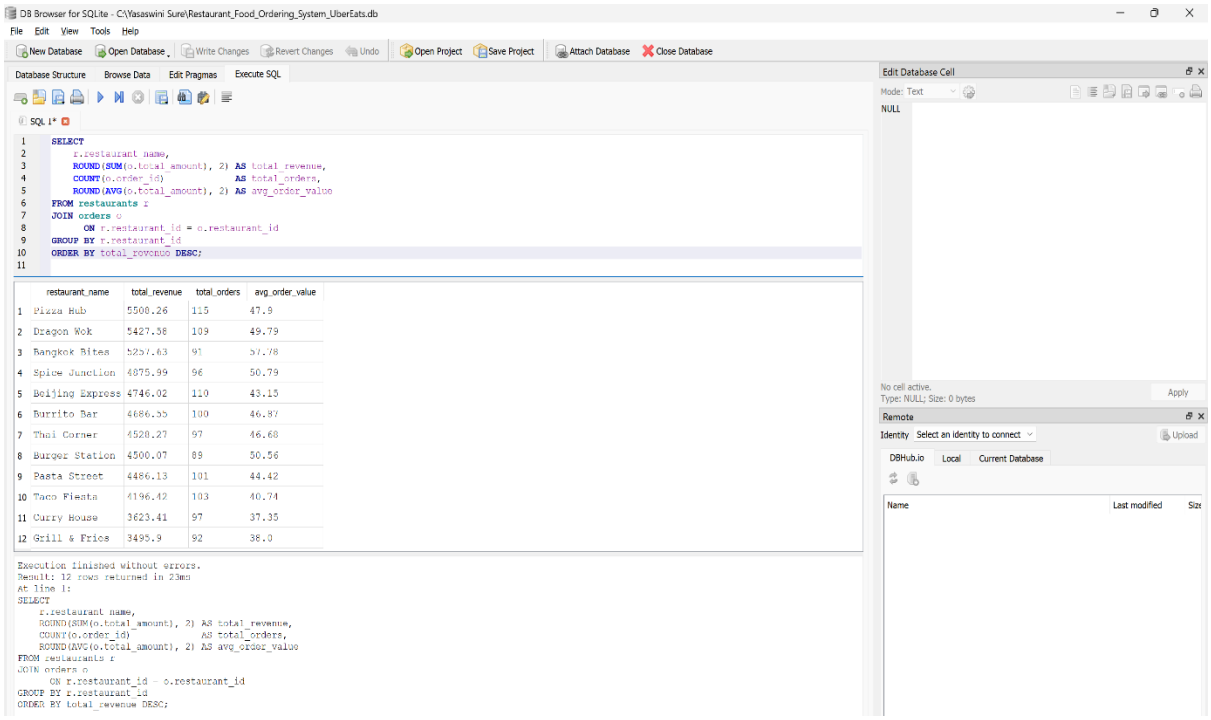
Most Ordered Items

Figure X: Top ten most frequently ordered menu items across the platform.



Revenue Analysis

Figure X: Revenue breakdown for each restaurant comparing total revenue, order volume, and average order value.



GitHub Link: https://github.com/yashumcu0304/SQLite-Project/blob/main/Restaurant_Food_Ordering_System.ipynb