## **Design Process and Group Responsibilities**

GROUP: David Xiao, Shadman Rakib, Raymond Li

### **Project Overview**

This document explains how our team planned, designed, and built the database system for a food delivery service. The goal was to create a working PostgreSQL schema that reflects how real delivery apps work — including customers, restaurants, delivery workers, payments, and orders. The final project includes an ER diagram, a relational schema with properly normalized tables, implementation scripts, and a set of documented queries.

### **How We Designed the System**

Our process was collaborative and hands-on. We broke the work into parts and came together regularly to discuss and adjust the design. Here's how we approached it:

#### **1. Figuring Out What We Needed**

We started by thinking about what features are essential in a food delivery app, based on examples like Uber Eats and DoorDash. We talked through the flow of an order: customer to restaurant to delivery, and then listed the major things we’d need to keep track of:

* Customers and their delivery addresses
* Restaurants and their menu items
* Types of cuisine
* Orders and the items in each order
* Payment methods
* Delivery personnel

#### **2. Mapping Out Relationships**

Next, we connected those entities in a visual ER diagram. We made sure to:

* Show that a customer can have many addresses
* Let restaurants offer multiple menu items and cuisines
* Represent the fact that one order comes from one restaurant, for one customer, and is handled by one delivery person
* Include a many-to-many structure where needed (e.g., restaurants and cuisines)

#### **3. Designing the Tables and Keys**

We turned the ERD into real SQL tables. This included:

* Choosing appropriate column types (like SERIAL for IDs and NUMERIC for prices)
* Making sure every table had a clear primary key
* Adding foreign keys to define relationships
* Using ON DELETE CASCADE where appropriate so records wouldn’t break if deleted
* Applying default values (like setting order status to 'Processing') to save time during inserts

#### **4. Writing the SQL Code**

We built the schema in PostgreSQL, carefully ordering the CREATE TABLE statements to avoid issues with foreign keys. We also made sure every constraint, like NOT NULL and foreign keys, was in place.

#### **5. Testing the Database**

To make sure the schema worked as expected, we created sample data and wrote queries to test key functionality, such as:

* Total revenue per day
* Popular menu items
* Workload by delivery personnel
* Revenue grouped by restaurant and date

#### **6. Writing Up the Documentation**

We wanted anyone reading our project to understand how and why everything was built. So we created documentation explaining:

* The logic of each SQL query
* Our design decisions and assumptions
* The role of each table and how it connects to others

### **Assumptions We Made**

Throughout the project, we made a few simplifying assumptions to keep things clear:

* A customer can have multiple addresses (like home and work)
* Menu items can appear in multiple restaurants (like a common franchise item)
* Each order comes from one restaurant
* Each delivery is handled by one person
* Every order is tied to one customer and one restaurant
* Revenue is calculated as: item price + taxes + delivery fee

### **Responsibilities**

Even though we worked together on the sections, each of us took the lead on different parts of the project:

#### **Raymond Li**

* Took the lead on documenting all the entities, their attributes, and how they relate to each other
* Wrote the entity-attribute overview showing primary keys and foreign keys
* Wrote documentation on group process.

#### **Shadman Rakib**

* Designed the visual ER diagram that shows how tables connect
* Helped troubleshoot joins and test data
* Was responsible for structuring and formatting the SQL schema creation file

#### **David Xiao**

* Wrote the SQL code that creates the database tables and constraints
* Created the test queries and made sure they returned accurate and useful results
* Wrote the detailed documentation that explains each query and its purpose

**Design**: Everyone gave input and reviewed the layout of tables and keys

* **Testing**: All of us helped run SQL scripts and tested them in both DB Fiddle and psql
* **Final Docs**: We worked together to organize the files and double-check that everything was included and clearly written

### **What’s Included in Our Project**

* Full SQL schema file: fooddelivery\_schema.sql
* Sample data sets: fooddelivery\_large\_sample.sql and fooddelivery\_sample\_data
* Query SQL code: fooddelivery\_queries.sql
* ER diagram image
* Schema documentation explaining tables and fields
* Query documentation explaining what each test query does