**Kathmandu University**

**Department of Computer Science and Engineering**

**Dhulikhel, Kavre**



**A Project Report**

**on**

**“RESTAURANT DBMS”**

**[Course Code: COMP232]**

**(For partial fulfillment of II Year/ II Semester in Computer Engineering)**

**Submitted By**

Sailesh Dahal (10)

Sarayu Gautam (14)

Awan Shrestha (45)

Sabil Shrestha (51)

**Submitted To**

**Mr. Santosh Khanal**

Assistant Professor

**Department of Computer Science and Engineering**

**Submission Date**

22nd July, 2019

# List of figures

[Figure 1: Cart 36](file:///D:\ACADEMICS\dbms\DBMSRESTAURANT.docx#_Toc14762901)

[Figure 2: Menu 36](file:///D:\ACADEMICS\dbms\DBMSRESTAURANT.docx#_Toc14762902)

[Figure 3: Signup page 37](file:///D:\ACADEMICS\dbms\DBMSRESTAURANT.docx#_Toc14762903)

[Figure 4: Signup page 37](file:///D:\ACADEMICS\dbms\DBMSRESTAURANT.docx#_Toc14762904)

[Figure 5: orders page 38](file:///D:\ACADEMICS\dbms\DBMSRESTAURANT.docx#_Toc14762905)

[Figure 6: Tables before normalization 38](file:///D:\ACADEMICS\dbms\DBMSRESTAURANT.docx#_Toc14762906)

[Figure 7: Tables after normalization 39](file:///D:\ACADEMICS\dbms\DBMSRESTAURANT.docx#_Toc14762907)

[Figure 8: ER Diagram before normalization 40](#_Toc14762908)

[Figure 9: ER Diagram after normalization 41](file:///D:\ACADEMICS\dbms\DBMSRESTAURANT.docx#_Toc14762909)

# Introduction

## Background

This project is a web-based Restaurant Database Management System which is built using MySQL and Node.js. This project mainly focuses on handling various data related to restaurant management. These include user data, import data, data related to food order, menu contents, restaurant data, data related to reservation and home delivery, and financial data of restaurant.

This project was built with a concept to enhance and digitalize restaurant management so that the data is stored in efficient format which preserves all the data that are generated in the restaurant in comparison to the old traditional way when data used to be stored manually in file system or paper note books.

## Objectives

1. To understand the concept of database management system.
2. To enhance our skill on database management system.
3. To create a database management system to manage a restaurant and its internal affairs.
4. To create a simple web application to perform and demonstrate the basic operations on the database including create, read, update and delete.

# Synopsis

The database represents a Restaurant that has tables, staffs, stock, import, menu, system of home delivery and reservation, and food order. There are total 24 tables among which 21 are entity tables and 3 are relationship tables. This project contains two types of users: Restaurant staff and customer.

The restaurant staffs belong to different category as per their role such as “Manager”, “Chef”, “Waiter”. The manager of the restaurant acts as the head of the restaurant who manages other staffs, decides menu of the restaurant, can update food items and their price, manages imports, issues bill for orders, handle reservations and homedelivery. The kitchen staffs can read orders and prepare order accordingly. They can also view current pantry status, and manage the imported goods. The serving staffs such as “Waiter” can take orders and insert them into the system so that kitchen staffs can view those orders.

The customers are those who can request for a reservation or place a home delivery request. In both cases, the customers must be logged in to the system providing their basic information. While making a reservation, the customer must provide date and time on which date they want to reserve a table in the restaurant. They should also specify the number of persons they are reserving for. While placing a home delivery request, a customer selects food items with their quantity and places the order.

A particular import is made from a particular company whose contact information is stored in the system. An import may contain many items that are stored as import details. Each import detail fills the stock of the same name with updated date of the import.

A menu has its start date and its name. Only a particular menu can be activated at a time. A menu can be discarded for a time and can be re activated as per the seasonal or any other requirements. A menu represents collection of food items which are sorted according to their category. A food item may or may not be available at a time.

# Design and Implementation

## System Requirement Specification

1. **Programming Language**: JavaScript
2. **Framework**: Node.js
3. **Database Management System**: MySQL
4. **Tools Used**: phpMyAdmin, MySQL Workbench, Visual Studio Code

## List of Entities with justification

1. **contact\_info**

This entity stores the contact information of users, restaurant and import company so as to avoid redundancy caused due to the repetition of similar type of information in multiple tables.

1. **users**

This entity stores the information regarding the authentication credentials of the different type of users who need to sign in and signup in the system for different purposes.

1. **staff**

This entity is a child of user entity that stores information particularly about staffs working in the restaurant.

1. **staff\_category**

This entity is for categorizing staffs based on their role and salary. No staffs with a category not stored in the system can exist.

1. **customer**

This entity is the child of user entity for representing a customer. This customer can make a reservation or request for a home delivery.

1. **restaurant**

This entity represents the restaurant and provides information about total number of staffs and tables present in the restaurant along with its capacity.

1. **import\_company**

This entity represents the company from which imports to the restaurant is made. Total transaction and remaining transaction to that company is stored.

1. **import**

This entity represents an import made at a particular date from a company which is identified by its unique bill number.

1. **import\_detail**

This is a weak entity of parent entity import. This entity is responsible for storing details of goods imported in a particular import.

1. **import\_type**

This entity categorizes the imported goods into different type on the basis of their storage and use. This entity also has a measuring unit for a type of import.

1. **stock**

This entity represents the stock of a particular good present in the restaurant. The stock is filled whenever an import is made.

1. **food\_category**

This entity categorizes food item prepared in the restaurant.

1. **food\_item**

This entity represents a particular food item that is prepared in the restaurant. It has its price and belongs to a food category.

1. **menu**

This entity represents the restaurant menu which is currently active.

1. **menu\_content**

This entity is a weak entity of menu, which represents the food item present in the menu and its availability.

1. **restaurant\_table**

This entity stores the number and vacancy of a particular table in the restaurant.

1. **reservation**

This entity represents a table reservation made by a customer for a certain date and time for given number of persons.

1. **home\_delivery**

This entity represents a particular home delivery made by a customer.

1. **food\_order**

This entity represents a particular food order placed whether by a homedelivery of by a waiter.

1. **order\_item**

This entity is weak entity of food\_order and consists information about a particular ordered item along with its quantity.

1. **bill**

This entity represents the bill issued for a particular order made at a particular date which includes total price of the order.

1. **order\_relates\_staff**

This is a relationship table which shows which staff took which order.

1. **order\_relates\_table**

This is a relationship table which shows which order went to which table

1. **order\_relates\_home\_delivery**

This is a relationship table which distinguishes home delivered ordered from normal orders.

## List of attributes with justification

1. **contact\_info**
   1. **contact\_info\_id**

This is the PRIMARY KEY which is VARCHAR (100) needed to represent contact information of a particular user, restaurant or an import company.

* 1. **name**

This represents name of user or a restaurant or an import company which is a VARCHAR (100) and this cannot be null.

* 1. **email**

This represents email of user or a restaurant or an import company which is a VARCHAR (100) and this cannot be null.

* 1. **address**

This represents address of user or a restaurant or an import company which is a VARCHAR (150) and this cannot be null.

* 1. **phone**

This represents phone number of a user or a restaurant or an import company which is a VARCHAR (30) and this cannot be null.

1. **users**
   1. **user\_id**

This is the PRIMARY KEY which is VARCHAR (100) needed to represent contact information of a particular user. It is also a FOREIGN KEY of parent table **contact\_info.**

* 1. **username**

This is a username which is VARCHAR (50), used by users for login and signup in the system.

* 1. **password**

This is the hashed password used for logging into the system.

1. **staff**
   1. **staff\_id**

This is the PRIMARY KEY which is VARCHAR (100) needed to represent contact information of a particular staff. It is also a FOREIGN KEY of parent table **contact\_info.**

* 1. **staff\_category**

This is a FOREIGN KEY VARCHAR (100) of parent table **staff\_category** which provides information about the category to which the particular staff belongs.

* 1. **last\_paid\_date**

This is a TIMESTAMP used for determining the last paid date of a staff.

* 1. **joined\_date**

This is a TIMESTAMP stored as a DEFAULT TIMESTAMP of the time when the staff is created.

1. **staff\_category**
   1. **staff\_category**

This is a VARCHAR (100) that determines to which category a staff belongs.

* 1. **salary**

This is a DOUBLE (10,2) that determines the salary of a staff belonging to particular category.

1. **customer**
   1. **customer\_id**

This is the PRIMARY KEY which is VARCHAR (100) needed to represent contact information of a particular customer. It is also a FOREIGN KEY of parent table **contact\_info.**

1. **restaurant**
   1. **restaurant\_id**

This is the PRIMARY KEY which is VARCHAR (100) needed to represent contact information of a particular restaurant. It is also a FOREIGN KEY of parent table **contact\_info.**

* 1. **total\_staff**

This is an INTEGER (10) used to store total staff working in the restaurant.

* 1. **capacity**

This is an INTEGER (10) which tells the capacity of the restaurant to accommodate customers.

* 1. **total­\_tables**

This in an INTEGER (10) that stores total tables present in the restaurant.

1. **import\_company**
   1. **import\_company\_id**

This is the PRIMARY KEY which is VARCHAR (100) needed to represent contact information of a particular company. It is also a FOREIGN KEY of parent table **contact\_info.**

* 1. **total\_transactions**

This is DOUBLE (10,2) which stores the total amount of transactions made with that company.

* 1. **remain\_transactions**

This is DOUBLE (10,2) which stores the remaining amount of transactions made with that company.

1. **import**
   1. **import\_company\_id**

This is a VARCHAR (100) FOREIGN KEY of parent table **import\_company** which is needed to determine which import was supplied by which company.

* 1. **bill\_no**

This is INTEGER (10) PRIMARY KEY of import which stores the bill number issued for that particular import.

* 1. **total\_price**

This is DOUBLE (10,2) which stores the total price of that import.

* 1. **import\_date**

This is a TIMESTAMP stored as CURRENT\_TIMESTAMP at the time when the import data is inserted.

1. **import\_detail**
   1. **import\_good**

This is VARCHAR (200) which determines the name of the imported good.

* 1. **import\_type**

This is a VARCHAR (120) which is a FOREIGN KEY of the parent table **import\_type** which determines the type to which that imported good belongs.

* 1. **bill\_no**

This is an INTEGER (10) which is a FOREIGN KEY to the parent table **import** which determines to which import this imported good belongs.

* 1. **quantity**

This is a DOUBLE (10,2) which stores the quantity of particular imported good.

* 1. **price**

This is a DOUBLE (10,2) which stores the price of particular imported good.

1. **import\_type**
   1. **import\_type**

This is a VARCHAR (120) which is PRIMARY KEY used to determine the type of import.

* 1. **measure\_unit**

This is a VARCHAR (10) which determines the unit of measurement of import belonging to a particular import type.

1. **stock**
   1. **stock\_name**

This is a VARCHAR (200) which is a PRIMARY KEY used to determine the name of the stock.

* 1. **type\_of\_stock**

This is a VARCHAR (120) which is a FOREIGN KEY to the parent table import type which determines the type of stock.

* 1. **last\_import­\_date**

This is a TIMESTAMP which is a FOREIGN KEY of parent table **import** which determines the last date in which the stock was filled.

* 1. **quantity**

This is a DOUBLE (10,2) which stores the current quantity of a particular stock.

1. **food\_category**
   1. **food\_category\_name**

This is a VARCHAR (100) which is a PRIMARY KEY of **food\_category** that determines to which category a food item belongs.

1. **food\_item**
   1. **food\_item\_name**

This is a VARCHAR (100) which is a PRIMARY KEY and is used to determine a particular food item.

* 1. **food\_item­­­\_price**

This is a DOUBLE (10,2) which determines price of a particular food item.

* 1. **food\_category\_name**

This is a VARCHAR (100) which is a FOREIGN KEY of parent table **food\_category** which determines the category to which a food item belongs.

1. **menu**
   1. **menu\_name**

This is a VARCHAR (100) which is a PRIMARY KEY used to determine a particular menu.

* 1. **menu\_start\_date**

This is a DATE that represents the starting date of use of a particular menu. It is also a PRIMARY KEY.

* 1. **menu\_end\_date**

This is a DATE that represents the end date of the menu if a menu has been discarded.

* 1. **is\_menu\_active**

This is a BOOLEAN with default value 0 which determines whether a particular menu is currently active or not.

1. **menu\_content**
   1. **menu\_name**

This is a VARCHAR (100) which is a FOREIGN KEY of parent table **menu** which determines the menu to which a particular menu content belongs.

* 1. **food\_item­\_name**

This is a VARCHAR (100) which is a FOREIGN KEY of parent table **food\_item** which is used to determine the name of a particular menu content.

* 1. **is\_food­\_available**

This is a BOOLEAN with default value 1 which is used to determine whether a food item is currently available or not.

1. **restaurant\_table**
   1. **table\_no**

This is an INTEGER (10) which is a PRIMARY KEY which uniquely identifies a table of the restaurant.

* 1. **is\_empty**

This is a BOOLEAN with default value 1 which determines whether a particular table is currently vacant or not.

1. **reservation**
   1. **customer\_id**

This is the FOREIGN KEY of parent table **customer** which is VARCHAR (100) needed to represent contact information of a particular customer who made the reservation.

* 1. **table\_no**

This is the FOREIGN KEY of parent table **restaurant\_table** which is INTEGER (10) which determines the table for which reservation is made.

* 1. **number\_of\_person**

This is an INTEGER (2) to determine the number of persons for whom the reservation is made.

* 1. **reservation\_date**

This is a TIMESTAMP which is stored as CURRENT\_TIMESTAMP at the time on which the reservation was made. This is also a PRIMARY KEY

* 1. **reservation\_fulfilled\_status**

This is a BOOLEAN with default value 0 which determines whether a reservation was fulfilled or not.

* 1. **reserved\_for\_date**

This is a TIMESTAMP which determines for which date a particular reservation was made.

* 1. **reserved\_for\_time**

This is a TIME which determines for which time of day a particular reservation was made.

1. **home\_delivery**
   1. **home\_delivery\_no**

This is a VARCHAR (120) which is a primary key and uniquely determines a particular home delivery.

* 1. **customer\_id**

This is the FOREIGN KEY of parent table **customer** which is VARCHAR (100) needed to represent a particular customer who placed the home delivery request.

* 1. **delivery\_staff\_id**

This is the FOREIGN KEY of parent table **staff** which is VARCHAR (100) needed to represent a particular staff who delivered the delivery.

* 1. **is\_delivered**

This is a BOOLEAN with default value 0 to determine whether a home delivery was delivered or not.

1. **food\_order**
   1. **order\_id**

This is a VARCHAR (120) which is a PRIMARY KEY for uniquely determining a particular order.

* 1. **order\_time**

This is a TIMESTAMP stored as CURRENT\_TIMESTAMP which determines when the order was placed.

1. **order\_item**
   1. **order\_id**

This is a VARCHAR (50) which is a FOREIGN KEY of parent table **food\_order** which determines to which particular order the given order item belongs.

* 1. **food\_item­\_name**

This is a VARCHAR (100) which is a FOREIGN KEY of parent table **food\_item** which determines the name of a particular order item.

* 1. **quantity**

This is an INT (3) which stores the quantity of a particular order item placed in an order.

1. **bill**
   1. **bill\_no**

This is an INTEGER (10) which is a PRIMARY KEY that uniquely identifies a particular bill.

* 1. **order\_id**

This is a VARCHAR (50) which is a FOREIGN KEY of parent table **food\_order** which determines to which particular order the given bill is issued.

* 1. **total\_price**

This is a DOUBLE (10,2) which determines the total price issued in that bill.

* 1. **issue­­\_date**

This is a TIMESTAMP stored as CURRENT\_TIMESTAMP to determine the date in which a particular bill was issued.

## Candidate keys/ Primary keys/ Composite Keys

1. **contact\_info**

CANDIDATE KEY: contact\_info\_id,

PRIMARY KEY: contact\_info\_id

We chose contact\_info\_id as candidate key because throughout this relation only contact\_info\_id is unique among other attributes and it uniquely determines the values of all other attributes present in contact\_info table. This is also a primary key because it is a least subset of candidate key.

1. **users**

CANDIDATE KEY: user\_id

PRIMARY KEY: user\_id

We chose user\_ id as candidate key because throughout this relation only user\_id is unique among other attributes and it uniquely determines the values of all other attributes present in user table. Also, it is foreign key referencing contact\_info\_id in contact\_info table which is always unique.

Here user\_id is also a primary key.

1. **staff**

CANDIDATE KEY: staff\_id, joined\_date

PRIMARY KEY: staff\_id

We chose staff\_id, joined\_date as candidate key because throughout this relation only staff\_id and joined\_date is unique among other attributes and they uniquely determine the values of all other attributes present in staff table.

Also, staff\_id is foreign key referencing user\_id in user table which is always unique. Hence, staff\_id is a primary key.

1. **staff\_category**

CANDIDATE KEY/PRIMARY KEY: staff\_category

We choose staff\_category as candidate key/ primary key because it uniquely determines all the tuples.

1. **customer**

CANDIDATE KEY: customer\_id

PRIMARY KEY: customer\_id

We chose customer\_ id as candidate key because throughout this relation only customer\_id is unique among other attributes and it uniquely determines the values of all other attributes present in customer table. Also, it is foreign key referencing user\_id in users table which is always unique.

Here customer\_id is also a primary key.

1. **restaurant**

CANDIDATE KEY: restaurant\_id

PRIMARY KEY: restaurant\_id

We chose restaurant\_ id as candidate key because throughout this relation only restaurant\_id is unique among other attributes and it uniquely determines the values of all other attributes present in restaurant table. Also, it is foreign key referencing contact\_info\_id in contact\_info table which is always unique.

Here restaurant\_id is also a primary key.

1. **import\_company**

CANDIDATE KEY: import\_company\_id

PRIMARY KEY: import\_company\_id

We chose import\_company\_ id as candidate key because throughout this relation only import\_company\_id is unique among other attributes and it uniquely determines the values of all other attributes present in import\_company table. Also, it is foreign key referencing contact\_info\_id in contact\_info table which is always unique.

Here import\_company\_id is also a primary key.

1. **import**

CANDIDATE KEY: bill\_no, import\_date

PRIMARY KEY: bill\_no

We chose bill\_no, import\_date as candidate key because throughout this relation only bill\_no and import\_date is unique among other attributes and they uniquely determine the values of all other attributes present in import table.

bill\_no is a primary key.

1. **import\_type**

CANDIDATE KEY/PRIMARY KEY: import\_type

We choose import\_type as candidate key/ primary key because it uniquely determines all the tuples.

1. **stock**

CANDIDATE KEY/ PRIMARY KEY: stock\_name

We choose stock\_name as candidate key/ primary key because it uniquely determines all the tuples.

1. **food\_category**

CANDIDATE KEY/ PRIMARY KEY: food\_category\_name

We choose food\_category\_name as candidate key/ primary key because it uniquely determines all the tuples.

1. **food\_item**

CANDIDATE KEY/ PRIMARY KEY: food\_item

We choose food\_item as candidate key/ primary key because it uniquely determines all the tuples.

1. **menu**

CANDIDATE KEY/PRIMARY KEY: menu\_name, menu\_start\_date

We chose menu\_name, menu\_start\_date as candidate key/primary key because throughout this relation only menu\_name and menu\_start\_date is unique among other attributes and they uniquely determine the values of all other attributes present in menu table.

1. **restaurant\_table**

CANDIDATE KEY/ PRIMARY KEY: table\_no

We choose table\_no as candidate key/ primary key because it uniquely determines all the tuples.

1. **reservation**

CANDIDATE KEY/ PRIMARY KEY: reservation\_date

We choose reservation\_date as candidate key/ primary key because it uniquely determines all the tuples.

1. **home\_delivery**

CANDIDATE KEY/ PRIMARY KEY: home\_delivery\_no

We choose home\_delivery\_no as candidate key/ primary key because it uniquely determines all the tuples.

1. **food\_order**

CANDIDATE KEY: order\_id, order\_time

PRIMARY KEY: order\_id

We chose order\_id, order\_time as candidate key because throughout this relation only order\_id and order\_time are unique among other attributes and they uniquely determine the values of all other attributes present in food\_order table.

order\_id is a primary key.

1. **bill**

CANDIDATE KEY: bill\_no, issue\_date

PRIMARY KEY: bill\_no

We chose bill\_no, issue\_date as candidate key because throughout this relation only bill\_no and issue\_date is unique among other attributes and they uniquely determine the values of all other attributes present in bill table.

bill\_no is a primary key.

## Partial dependency/ transitive dependency/functional dependency

1. **contact\_info** (contact\_info\_id, name, address, email, phone)

Keys → contact\_info\_id

F = {contact\_info\_id → name, address, email, phone}

1. **food\_category** (food\_category, food\_category\_name)

Keys → food\_category, food\_category\_name

F = {food\_category → food\_category\_name, food\_category\_name → food\_category}

1. **food\_item** (food\_item\_name, food\_item\_price, food\_category)

Keys → food\_item\_name

F = {food\_item\_name → food\_item\_price, food\_category}

1. **menu** (menu\_name, menu\_start\_date, menu\_end\_date, is\_menu\_active)

Keys → menu\_name, menu\_start\_date

F = {menu\_name, menu\_start\_date → menu\_end\_date, is\_menu\_active}

1. **menu\_content** (menu\_name, food\_item\_name, is\_food\_available)

It is a weak entity having no keys.

F = {food\_item\_name → is\_food\_available}

1. **customer** {customer\_id}

Keys → customer\_id

There is no functional dependency.

1. **manager** (username, password, name)

Keys → username, name

F = {username, name → password, username → password, name → password}

Partial Dependency = {username → password, name → password}

1. **staff\_category** (staff\_category, salary)

Keys → staff\_category

F = {staff\_category → salary}

1. **staff** (staff\_category, last\_paid\_date, staff\_id, joined\_date)

    Key→ staff\_id

    F= {staff\_id staff\_category, last\_paid\_date, joined\_date}

1. **import\_company** (import\_company\_id, total\_transactions, remain\_transactions, purchase type)

    Keys→ import\_company\_id

    F= {import\_company\_id→ total\_transactions, remain\_transactions, purchase\_type}

1. **restaurant** (restaurant\_id, total\_staff, capacity, total\_tables)

    Keys → restaurant\_id

    F= {restaurant\_id → total\_staff, capacity, total\_tables}

1. **restaurant\_table** (table\_no, is\_empty)

    Keys → table\_no

    F= {table\_no → is\_empty}

1. **import** (import\_company\_id, import\_good, bill\_no, quantity, price, import\_date)

    Keys → bill\_no, import\_date

    F= {bill\_no, import\_date → import\_company\_id, import\_good, quantity,

bill\_no → price}

1. **stock** (name, type\_of\_stock, import\_date, quantity)

    Keys → name

    F= {name→ type\_of\_stock, import\_date, quantity}

1. **reservation** (customer\_id, table\_no, number\_of\_person, reservation\_date)

    Keys → customer\_id, reservation\_date

    F = {customer\_id → table\_no, number\_of\_person, reservation\_date → table\_no}

1. **bill** (bill\_no, order\_id, total\_price)

    Keys → bill\_no

    F= {bill\_no → order\_id, total\_price}

1. **home\_delivery** (customer\_id, staff\_id, bill\_no, order date, is\_delivered)

    Keys → customer\_id, order\_date

    F= {customer\_id → staff\_id, bill\_no, order\_date → is\_delivered}

## Normalization

**import** (import\_company\_id, import\_good, bill\_no, quantity, price, import\_date)

Keys → bill\_no, import\_date

F= {bill\_no, import\_date → import\_company\_id, import\_good, quantity,

bill\_no → price}

In this relation, there exists partial dependency, here bill\_no and import\_date which are primary key is related to every item in the relation, but bill\_no being a prime attribute is related to price which is a non-prime attribute. So, we decompose relation import into

**import** (bill\_no, total\_price, import\_date)

**import\_detail** (import\_good, quantity, price)

In similar way we have decomposed relation into following schemas into 2NF form:

1. **restaurant** (restaurant\_id, total staff, capacity, total\_tables)
2. **contact\_info** (contact\_info\_id, name, address, email, phone)
3. **staff\_category** (staff\_category, salary)
4. **staff** (staff\_id, last\_paid\_date, joined\_date)
5. **bill** (bill\_no, total\_price, issue\_date)
6. **menu\_content**(is\_food\_available)
7. **menu** (menu\_name, menu\_start\_date, menu\_end\_date, is\_menu\_active)
8. **food\_category**(food\_category\_name)
9. **food\_item** (food\_item\_name, food\_item\_price)
10. **order\_item**(quantity)
11. **food\_order** (order\_id, order\_time)
12. **restaurant\_table** (table\_no, is\_empty)
13. **reservation** (reservation\_date, number\_of\_person, reservation\_fulfilled\_status, reserved\_for\_date, reserved\_for\_time)
14. **stock** (stock\_name, type\_of\_stock, last\_import\_date, quantity)
15. **import\_type** (import\_type, measure\_unit)
16. **import\_detail** (import\_good, quantity, price)
17. **import** (bill\_no, total\_price, import\_date)
18. **import\_company** (import\_company\_id, total\_transactions, remain\_transactions)
19. **customer** (customer\_id)
20. **users** (user\_id, username, password)
21. **home\_delivery** (home\_delivery\_no, is\_delivered)

# Appendix

## Database scripts

CREATE DATABASE restaurant;

USE restaurant;

CREATE TABLE IF NOT EXISTS contact\_info

(

contact\_info\_id VARCHAR (100),

name VARCHAR (100) NOT NULL,

address VARCHAR (150) NOT NULL,

email VARCHAR (150) NOT NULL,

phone VARCHAR (30) NOT NULL,

PRIMARY KEY (contact\_info\_id)

);

CREATE TABLE IF NOT EXISTS users (

user\_id VARCHAR (100),

username VARCHAR (50),

password VARCHAR (255),

PRIMARY KEY (user\_id),

FOREIGN KEY (user\_id) REFERENCES contact\_info(contact\_info\_id) ON DELETE CASCADE

);

CREATE TABLE IF NOT EXISTS food\_category

(

food\_category\_name VARCHAR (100),

PRIMARY KEY (food\_category\_name)

);

CREATE TABLE IF NOT EXISTS food\_item

(

food\_item\_name VARCHAR (100),

food\_item\_price DOUBLE (10,2),

food\_category\_name VARCHAR (100),

PRIMARY KEY (food\_item\_name),

FOREIGN KEY (food\_category\_name) REFERENCES food\_category(food\_category\_name)

);

CREATE TABLE IF NOT EXISTS menu

(

menu\_name VARCHAR (100),

menu\_start\_date DATE,

menu\_end\_date DATE,

is\_menu\_active BOOLEAN DEFAULT 0,

PRIMARY KEY (menu\_name, menu\_start\_date)

);

CREATE TABLE IF NOT EXISTS menu\_content

(

menu\_name VARCHAR (100),

food\_item\_name VARCHAR (100),

is\_food\_available BOOLEAN DEFAULT 1,

FOREIGN KEY (menu\_name) REFERENCES menu(menu\_name),

FOREIGN KEY (food\_item\_name) REFERENCES food\_item(food\_item\_name)

);

CREATE TABLE IF NOT EXISTS customer (

customer\_id VARCHAR (100) PRIMARY KEY,

FOREIGN KEY (customer\_id) REFERENCES users(user\_id)

);

CREATE TABLE IF NOT EXISTS staff\_category

(

staff\_category VARCHAR (100) PRIMARY KEY,

salary DOUBLE (10,2)

);

CREATE TABLE IF NOT EXISTS staff (

staff\_id VARCHAR (100) PRIMARY KEY,

staff\_category VARCHAR (100),

last\_paid\_date TIMESTAMP,

joined\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP UNIQUE,

FOREIGN KEY (staff\_id) REFERENCES users(user\_id) ON DELETE CASCADE,

FOREIGN KEY (staff\_category) REFERENCES staff\_category(staff\_category) ON DELETE CASCADE);

CREATE TABLE IF NOT EXISTS import\_company

(

import\_company\_id VARCHAR (100) PRIMARY KEY,

total\_transactions DOUBLE (10,2) DEFAULT 0.0,

remain\_transactions DOUBLE (10,2) DEFAULT 0.0,

FOREIGN KEY (import\_company\_id) REFERENCES contact\_info(contact\_info\_id)

);

CREATE TABLE IF NOT EXISTS restaurant

(

restaurant\_id VARCHAR (100) PRIMARY KEY,

total\_staff INTEGER (10),

capacity INTEGER (10),

total\_tables INTEGER (10),

FOREIGN KEY (restaurant\_id) REFERENCES contact\_info(contact\_info\_id) ON DELETE CASCADE

);

CREATE TABLE IF NOT EXISTS restaurant\_table (

table\_no INTEGER (10) PRIMARY KEY,

is\_empty BOOLEAN DEFAULT 1

);

CREATE TABLE IF NOT EXISTS import (

import\_company\_id VARCHAR (100),

bill\_no INTEGER (10),

total\_price DOUBLE (10,2),

import\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP UNIQUE NOT NULL,

PRIMARY KEY (bill\_no),

FOREIGN KEY (import\_company\_id) REFERENCES import\_company(import\_company\_id)

);

CREATE TABLE IF NOT EXISTS import\_type (

import\_type VARCHAR (120) PRIMARY KEY,

measure\_unit VARCHAR (10)

);

CREATE TABLE IF NOT EXISTS import\_detail (

import\_good VARCHAR (200),

import\_type VARCHAR (128),

bill\_no INTEGER (10),

quantity DOUBLE (10,2),

price DOUBLE (10,2),

FOREIGN KEY (bill\_no) REFERENCES import(bill\_no) ON DELETE CASCADE,

FOREIGN KEY (import\_type) REFERENCES import\_type(import\_type)

);

CREATE TABLE IF NOT EXISTS stock (

stock\_name VARCHAR (200) PRIMARY KEY,

type\_of\_stock VARCHAR (120),

last\_import\_date TIMESTAMP,

quantity DOUBLE (10,2),

FOREIGN KEY (last\_import\_date) REFERENCES import(import\_date),

FOREIGN KEY (type\_of\_stock) REFERENCES import\_type(import\_type)

);

CREATE TABLE IF NOT EXISTS reservation

(

customer\_id VARCHAR (100),

table\_no INTEGER (10),

number\_of\_person INTEGER (2),

reservation\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP UNIQUE,

reservation\_fulfilled\_status BOOLEAN DEFAULT false,

reserved\_for\_date TIMESTAMP,

reserved\_for\_time TIME,

PRIMARY KEY (reservation\_date),

FOREIGN KEY (customer\_id) REFERENCES customer(customer\_id),

FOREIGN KEY (table\_no) REFERENCES restaurant\_table(table\_no)

);

CREATE TABLE IF NOT EXISTS food\_order

(

order\_id VARCHAR (120),

order\_time TIMESTAMP DEFAULT CURRENT\_TIMESTAMP UNIQUE,

PRIMARY KEY (order\_id)

);

CREATE TABLE IF NOT EXISTS home\_delivery (

home\_delivery\_no VARCHAR (120),

customer\_id VARCHAR (120),

delivery\_staff\_id VARCHAR (120),

is\_delivered BOOLEAN DEFAULT 0,

PRIMARY KEY (home\_delivery\_no),

FOREIGN KEY (customer\_id) REFERENCES customer(customer\_id),

FOREIGN KEY (delivery\_staff\_id) REFERENCES staff(staff\_id)

);

CREATE TABLE IF NOT EXISTS order\_item (

order\_id VARCHAR (50),

food\_item\_name VARCHAR (100),

quantity INT (3),

FOREIGN KEY (order\_id) REFERENCES food\_order(order\_id),

FOREIGN KEY (food\_item\_name) REFERENCES food\_item(food\_item\_name)

);

CREATE TABLE IF NOT EXISTS bill (

bill\_no INTEGER (10) PRIMARY KEY,

order\_id VARCHAR (120),

total\_price DOUBLE (10,2),

issue\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP UNIQUE,

FOREIGN KEY (order\_id) REFERENCES food\_order(order\_id)

);

CREATE TABLE IF NOT EXISTS order\_relates\_staff (

order\_id VARCHAR (50),

staff\_id VARCHAR (120),

FOREIGN KEY (order\_id) REFERENCES food\_order(order\_id),

FOREIGN KEY (staff\_id) REFERENCES staff(staff\_id)

);

CREATE TABLE IF NOT EXISTS order\_relates\_table (

order\_id VARCHAR (50),

table\_no INT (11),

FOREIGN KEY (order\_id) REFERENCES food\_order(order\_id),

FOREIGN KEY (table\_no) REFERENCES restaurant\_table(table\_no)

);

CREATE TABLE IF NOT EXISTS order\_relates\_home\_delivery (

order\_id VARCHAR (50),

customer\_id VARCHAR (120),

FOREIGN KEY (order\_id) REFERENCES food\_order(order\_id),

FOREIGN KEY (customer\_id) REFERENCES home\_delivery(customer\_id)

);

SELECT user\_id, username, email, name, address, phone FROM users INNER JOIN contact\_info ON users.user\_id=contact\_info.contact\_info\_id WHERE user\_id=?

SELECT \* FROM contact\_info WHERE email=? AND contact\_info\_id!=?

UPDATE contact\_info SET email=?,name=?,address=?,phone=? WHERE contact\_info\_id=?

INSERT INTO contact\_info SET name=?,email=?,phone=?,address=?,contact\_info\_id=?

INSERT INTO import\_company SET total\_transactions=?,remain\_transactions=?, import\_company\_id=?

SELECT import\_company\_id,name,phone,email,address,total\_transactions,remain\_transactions FROM import\_company INNER JOIN contact\_info ON import\_company.import\_company\_id= contact\_info.contact\_info\_id ORDER BY name

SELECT import\_company\_id,name,phone,email,address,total\_transactions,remain\_transactions FROM import\_company INNER JOIN contact\_info ON import\_company.import\_company\_id= contact\_info.contact\_info\_id where import\_company\_id=? ORDER BY name

UPDATE contact\_info SET name=?,address=?,email=?,phone=? WHERE contact\_info\_id=?

UPDATE import\_company SET total\_transactions=?,remain\_transactions=? WHERE import\_company\_id=?

DELETE FROM import\_company WHERE import\_company\_id=?

DELETE FROM contact\_info WHERE contact\_info\_id=?

INSERT INTO import SET import\_company\_id=?,bill\_no=?,total\_price=?,import\_date=?

INSERT INTO import\_detail SET import\_good=?,bill\_no=?,quantity=?,price=?,import\_type=?

SELECT \* FROM stock WHERE stock\_name=?

INSERT INTO stock SET stock\_name=?, type\_of\_stock=?,last\_import\_date=?,quantity=?

UPDATE stock SET type\_of\_stock=?,last\_import\_date=?,quantity=quantity+? WHERE stock\_name=?

SELECT import\_company\_id,import.bill\_no,total\_price,import\_date FROM import ORDER BY import\_date LIMIT ?,20

SELECT import\_company\_id,import.bill\_no,total\_price,import\_date,quantity,price,import\_good FROM import INNER JOIN import\_detail ON import.bill\_no= import\_detail.bill\_no WHERE import.bill\_no=?

SELECT bill\_no,total\_price,import\_date FROM import WHERE import\_company\_id=? ORDER BY import\_date LIMIT ?,20

SELECT quantity,price,import\_good FROM import\_detail WHERE bill\_no=?

DELETE FROM import WHERE bill\_no=?

SELECT user\_id,username,password,email,name,address,phone,staff\_category FROM users INNER JOIN contact\_info ON users.user\_id=contact\_info.contact\_info\_id INNER JOIN staff ON users.user\_id= staff.staff\_id WHERE users.username=? AND staff.staff\_category=?

SELECT user\_id,username,password,email,name,address,phone FROM users INNER JOIN contact\_info ON users.user\_id=contact\_info.contact\_info\_id WHERE username=?

INSERT INTO food\_order SET order\_id=?,order\_time=?

INSERT INTO order\_item SET order\_id=?,food\_item\_name=?, quantity=?

INSERT INTO order\_relates\_staff SET order\_id=?,staff\_id=?

INSERT INTO order\_relates\_table SET order\_id=?,table\_no=?

DELETE FROM food\_order WHERE order\_id=?

SELECT table\_no from restaurant\_table where is\_empty=1 limit 1

INSERT INTO reservation SET customer\_id=?,number\_of\_person=?,table\_no=?,reserved\_for\_date=?,reserved\_for\_time=?

SELECT customer\_id,name,phone,email,address,reservation\_date,reserved\_for\_date,reserved\_for\_time,table\_no,number\_of\_person FROM reservation INNER JOIN users ON reservation.customer\_id= users.user\_id INNER JOIN contact\_info ON users.user\_id= contact\_info.contact\_info\_id ORDER BY reservation\_date

SELECT customer\_id,name,phone,email,address,reservation\_date,reserved\_for\_time,table\_no,number\_of\_person FROM reservation INNER JOIN users ON reservation.customer\_id= users.user\_id INNER JOIN contact\_info ON users.user\_id= contact\_info.contact\_info\_id where reserved\_for\_date =?

SELECT name,phone,email,address,reservation\_date,reserved\_for\_date,reserved\_for\_time,table\_no,number\_of\_person FROM reservation INNER JOIN users ON reservation.customer\_id= users.user\_id INNER JOIN contact\_info ON users.user\_id= contact\_info.contact\_info\_id where customer\_id=?

SELECT customer\_id,name,phone,email,address,reservation\_date,reserved\_for\_date,reserved\_for\_time,number\_of\_person FROM reservation INNER JOIN users ON reservation.customer\_id= users.user\_id INNER JOIN contact\_info ON users.user\_id= contact\_info.contact\_info\_id where table\_no=?

UPDATE reservation SET reserved\_for\_date=?,reserved\_for\_time=?,number\_of\_person=?,reservation\_fulfilled\_status=? WHERE table\_no=? and reservation\_date=?

DELETE FROM reservation WHERE table\_no=? and reservation\_date

INSERT INTO contact\_info SET contact\_info\_id=?,name=?,address=?,email=?,phone=?

INSERT INTO restaurant SET restaurant\_id=?,total\_staff=?,capacity=?,total\_tables=?

INSERT INTO restaurant\_table SET table\_no=?

DELETE FROM contact\_info WHERE contact\_info\_id=?

SELECT \* FROM contact\_info WHERE email=? AND contact\_info\_id!=?

UPDATE contact\_info AS c INNER JOIN restaurant AS r ON c.contact\_info\_id=r.restaurant\_id SET c.email=?,c.name=?,c.address=?,c.phone=?,r.total\_staff=?,r.capacity=?,r.total\_tables=? WHERE r.restaurant\_id=?

SELECT r.restaurant\_id,c.name,c.email,c.address,c.phone,r.total\_staff,r.capacity,r.total\_tables FROM contact\_info AS c INNER JOIN restaurant AS r ON c.contact\_info\_id=r.restaurant\_id WHERE r.restaurant\_id=?

SELECT username,email FROM users INNER JOIN contact\_info ON users.user\_id=contact\_info.contact\_info\_id WHERE username=? OR email=?

INSERT INTO contact\_info SET contact\_info\_id=?,name=?,address=?,email=?,phone=?

INSERT INTO users SET user\_id=?,username=?,password=?

INSERT INTO customer SET customer\_id=?

DELETE FROM contact\_info WHERE contact\_info\_id=?

SELECT username,email,staff\_category FROM users LEFT JOIN contact\_info ON users.user\_id=contact\_info.contact\_info\_id LEFT JOIN staff ON users.user\_id=staff.staff\_id WHERE (username=? OR email=?) OR staff\_category=?

SELECT \* FROM staff\_category WHERE staff\_category=?

INSERT INTO contact\_info SET contact\_info\_id=?,name=?,address=?,email=?,phone=?

INSERT INTO users SET user\_id=?,username=?,password=?

INSERT INTO staff SET staff\_id=?,staff\_category=?

DELETE FROM contact\_info WHERE contact\_info\_id=?

SELECT username,email,staff\_category FROM users LEFT JOIN contact\_info ON users.user\_id=contact\_info.contact\_info\_id LEFT JOIN staff ON users.user\_id=staff.staff\_id WHERE (username=? OR email=?) AND staff\_category=?

SELECT \* FROM staff\_category WHERE staff\_category=?

INSERT INTO contact\_info SET contact\_info\_id=?,name=?,address=?,email=?,phone=?

INSERT INTO users SET user\_id=?,username=?,password=?

INSERT INTO staff SET staff\_id=?,staff\_category=?

DELETE FROM contact\_info WHERE contact\_info\_id=?

SELECT username,email,staff\_category FROM users LEFT JOIN contact\_info ON users.user\_id=contact\_info.contact\_info\_id LEFT JOIN staff ON users.user\_id=staff.staff\_id WHERE (username=? OR email=?) AND staff\_category=?

SELECT \* FROM staff\_category WHERE staff\_category=?

INSERT INTO contact\_info SET contact\_info\_id=?,name=?,address=?,email=?,phone=?

INSERT INTO users SET user\_id=?,username=?,password=?

INSERT INTO staff SET staff\_id=?,staff\_category=?

DELETE FROM contact\_info WHERE contact\_info\_id=?

SELECT \* FROM contact\_info WHERE contact\_info\_id IN(SELECT customer\_id FROM customer)

SELECT contact\_info.\*,staff.staff\_category FROM contact\_info INNER JOIN staff ON contact\_info.contact\_info\_id=staff.staff\_id WHERE staff.staff\_id IN (SELECT staff\_id FROM staff)

UPDATE contact\_info SET name=?,address=?,email=?,phone=? WHERE contact\_info\_id=?

DELETE users.\*,contact\_info.\* FROM users INNER JOIN contact\_info ON users.user\_id=contact\_info.contact\_info\_id WHERE users.user\_id=?

SELECT \* FROM staff\_category

UPDATE staff\_category SET salary=? WHERE staff\_category=?

SELECT COALESCE(SUM(total\_price),0) AS import\_amount, COUNT(bill\_no) AS total\_imports FROM import WHERE YEAR(import\_date)=? AND MONTH(import\_date)=? AND DAY(import\_date)=?

SELECT COALESCE(SUM(total\_price),0) AS import\_amount, COUNT(bill\_no) AS total\_imports FROM import WHERE YEAR(import\_date)=? AND MONTH(import\_date)=?

SELECT COALESCE(SUM(total\_price),0) AS import\_amount, COUNT(bill\_no) AS total\_imports FROM import WHERE YEAR(import\_date)=?

SELECT import\_good,sum(quantity) AS quantity,sum(price) as total\_cost,sum(price)\*100/(select sum(price) from import\_detail) as percentage from import\_detail group by import\_good,import\_type ORDER BY total\_cost DESC

SELECT import\_type,count(import\_good)\*100/(SELECT count(import\_good) from import\_detail) AS percentage,sum(price) as total\_price,sum(quantity) as total\_quantity from import\_detail group by import\_type ORDER BY total\_price

SELECT \* FROM restaurant\_table

UPDATE restaurant\_table SET is\_empty=? WHERE table\_no=?

## Screenshots

## UI Screenshots

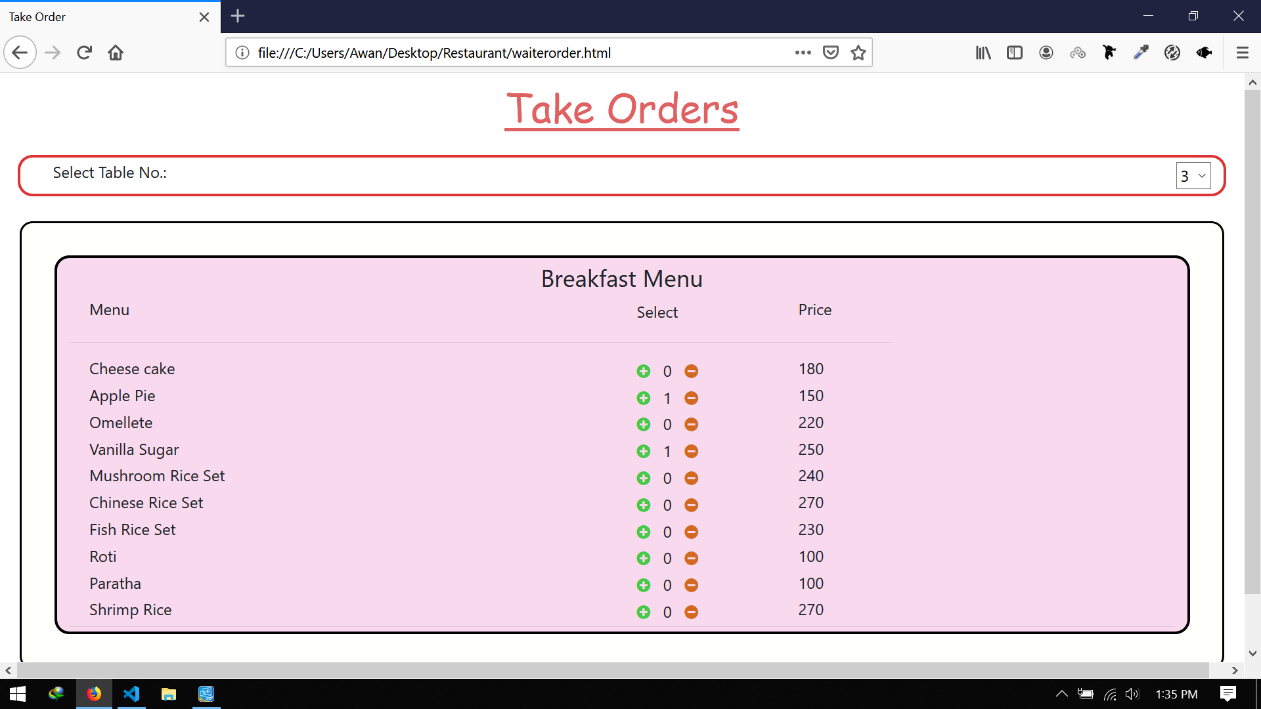
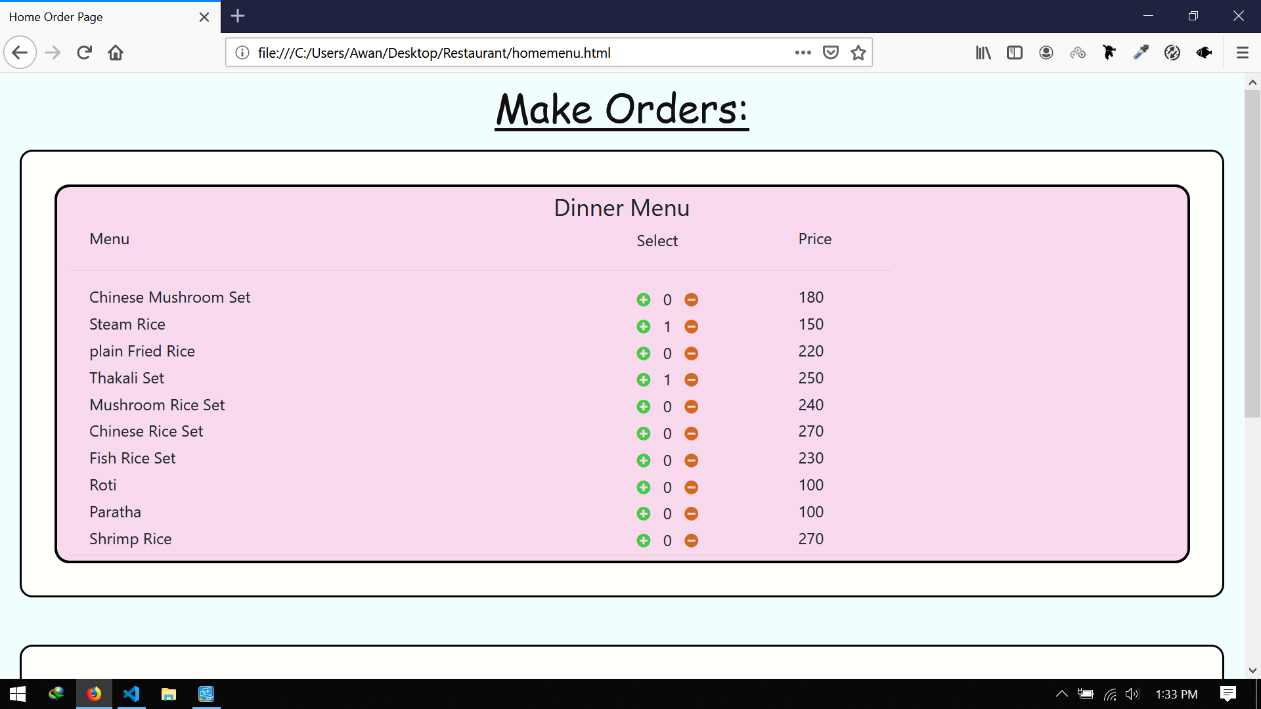
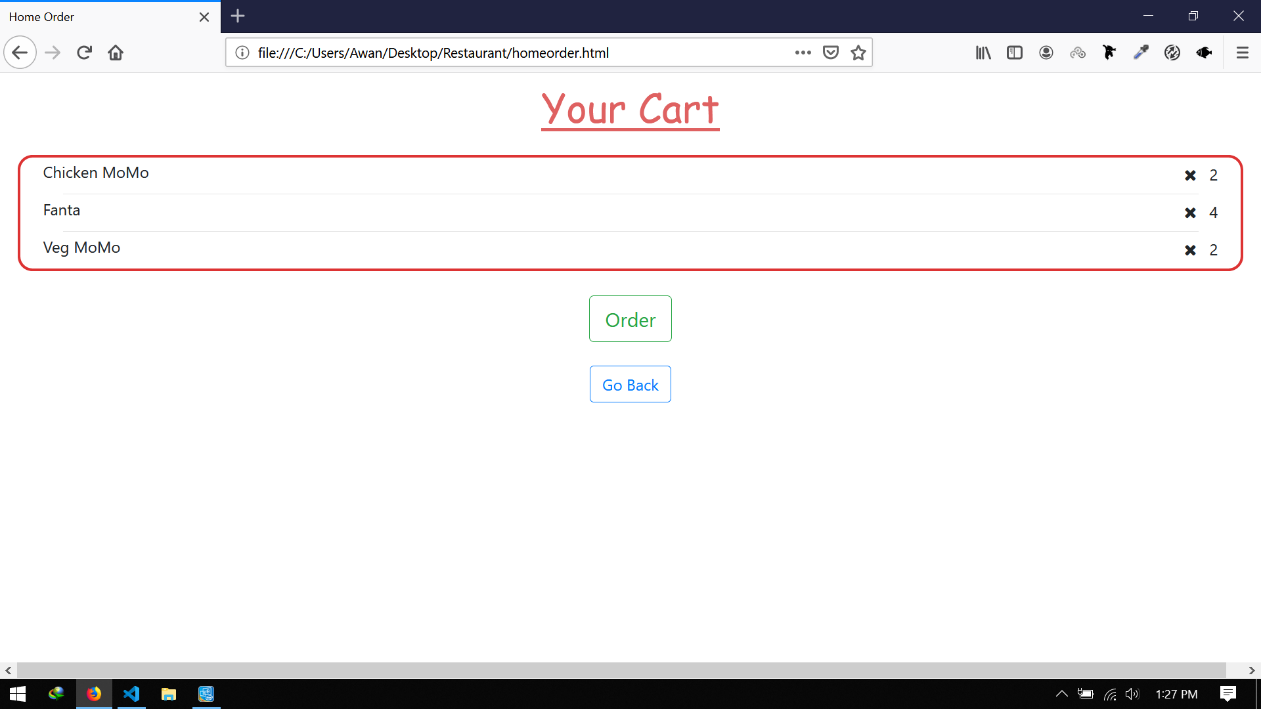
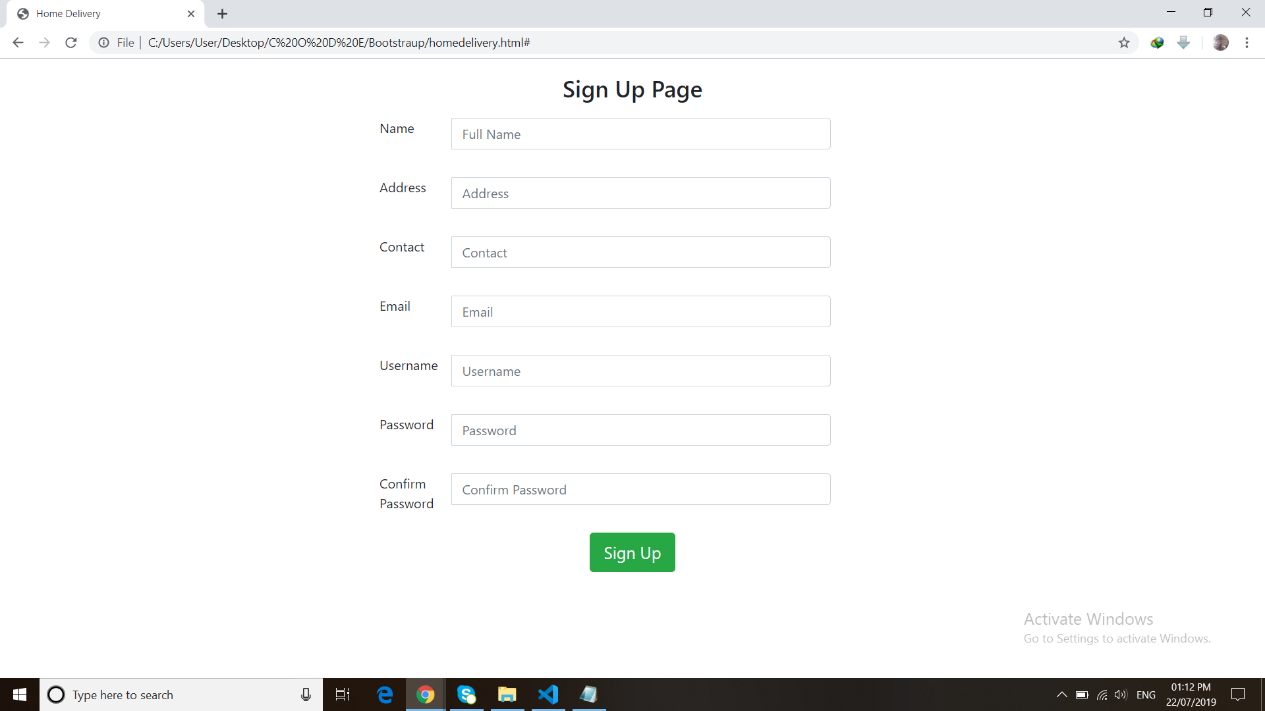
****

Figure : Cart

Figure : Menu

Figure : Signup page

Figure : Signup page

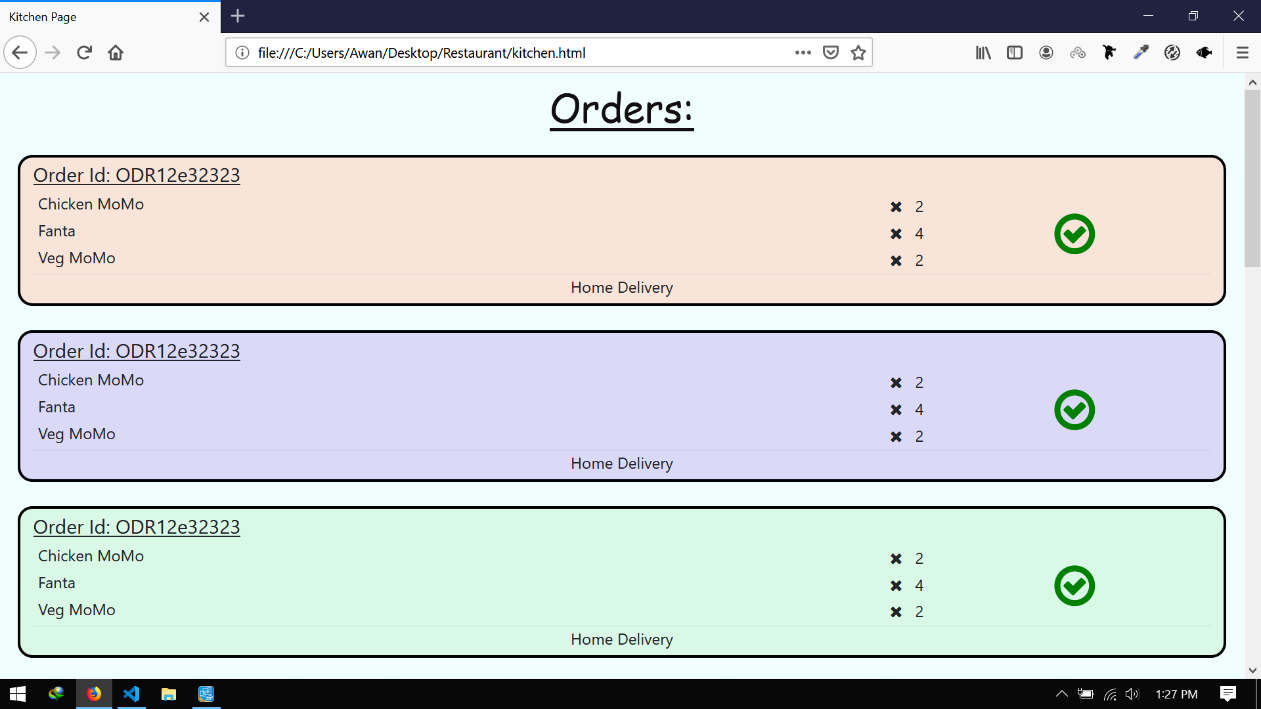
****

Figure : orders page

## Tables and ER diagrams

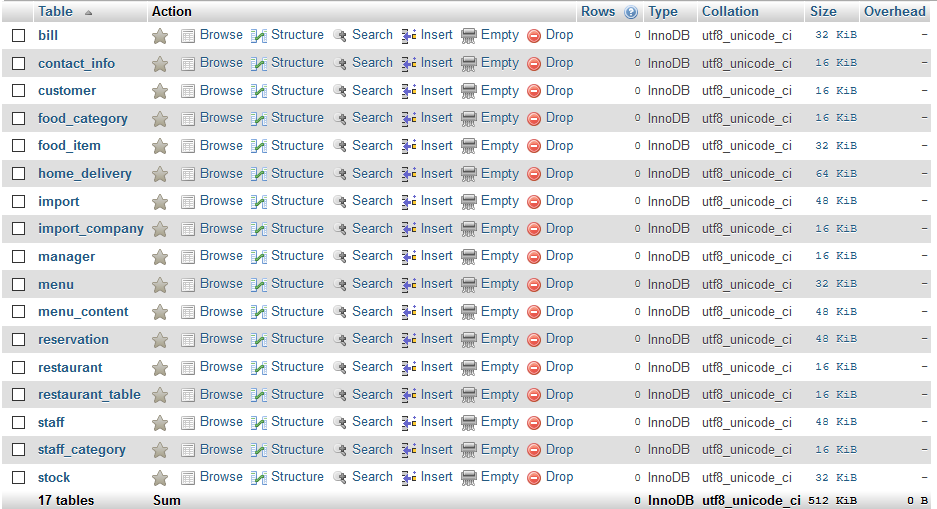


Figure : Tables before normalization

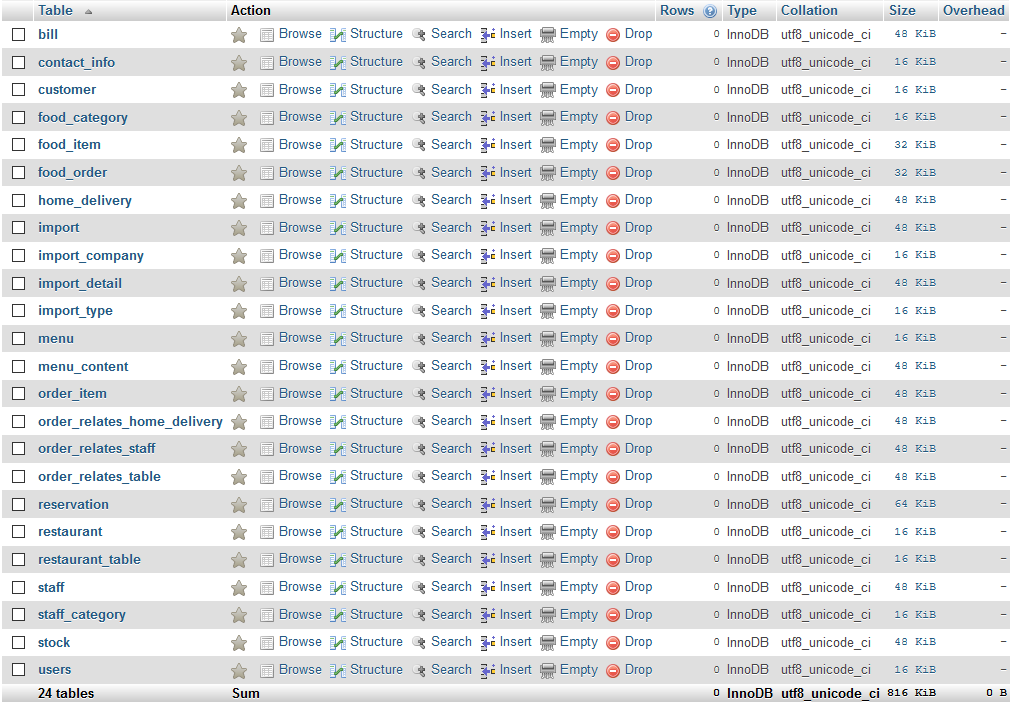
****

Figure : Tables after normalization

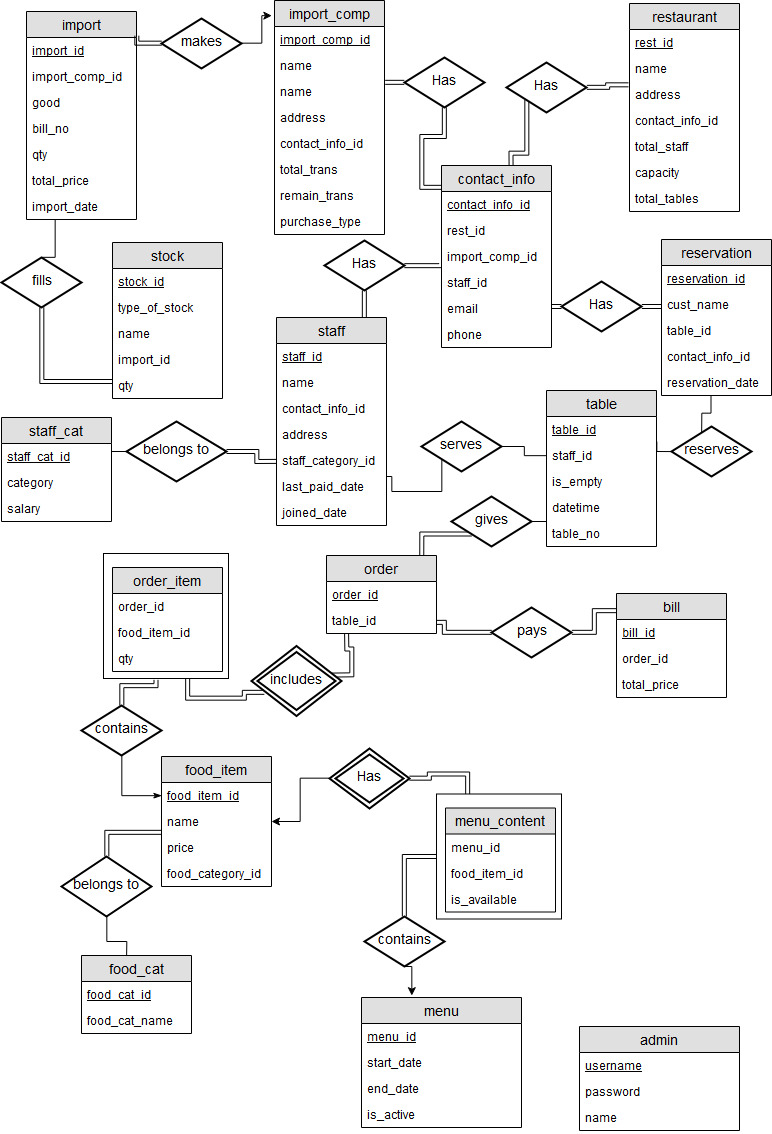
****

Figure 8: ER Diagram before normalization

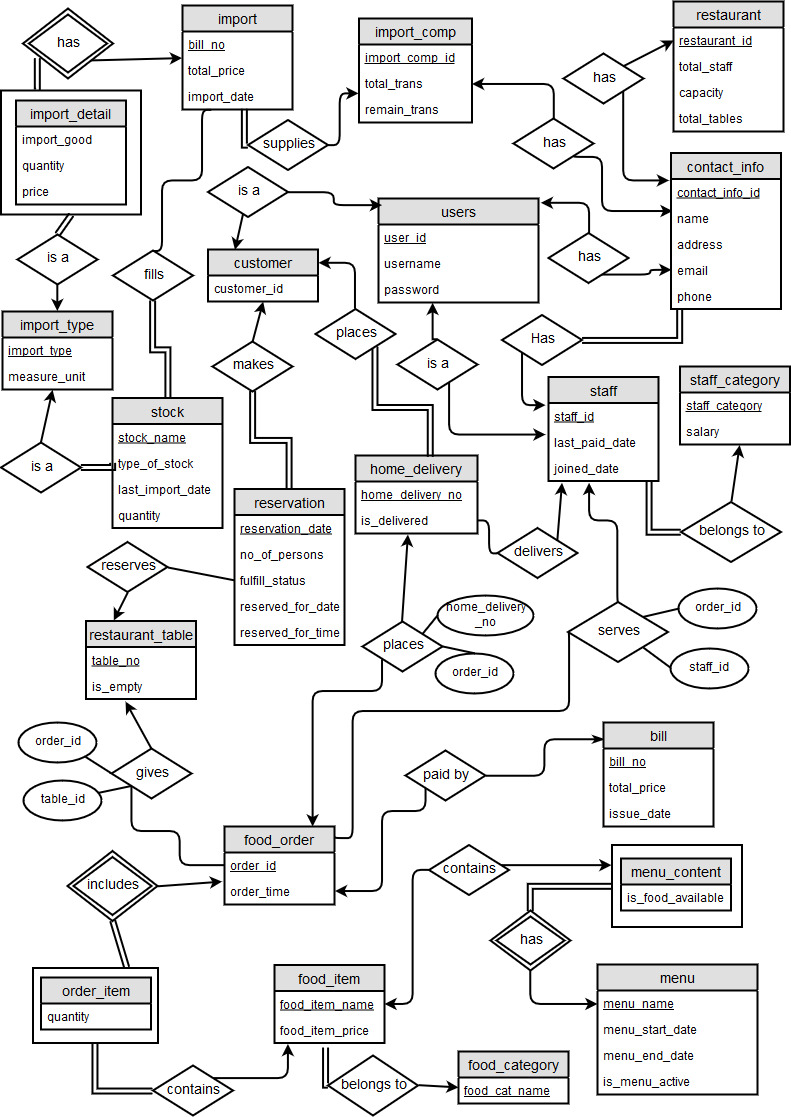


Figure : ER Diagram after normalization

Table of Contents

[List of figures ii](#_Toc14765549)

[Chapter 1: Introduction 1](#_Toc14765550)

[1.1. Background 1](#_Toc14765551)

[1.2. Objectives 1](#_Toc14765552)

[Chapter 2: Synopsis 2](#_Toc14765553)

[Chapter 3: Design and Implementation 3](#_Toc14765554)

[3.1. System Requirement Specification 3](#_Toc14765555)

[3.2. List of Entities with justification 3](#_Toc14765556)

[3.3. List of attributes with justification 6](#_Toc14765557)

[3.4. Candidate keys/ Primary keys/ Composite Keys 14](#_Toc14765558)

[3.5. Partial dependency/ transitive dependency/functional dependency 18](#_Toc14765559)

[3.6. Normalization 21](#_Toc14765560)

[Chapter 4: Appendix 23](#_Toc14765561)

[4.1. Database scripts 23](#_Toc14765562)

[4.2. Screenshots 36](#_Toc14765563)

[4.2.1. UI Screenshots 36](#_Toc14765564)

[4.2.2. Tables and ER diagrams 38](#_Toc14765565)