Ho Chi Minh City University of Technology

Faculty of Computer Science and Engineering

Database System Lab (CO2014)

Assignment Report

Hotel Management Application

Lecturer: Nguyễn Thanh Bình

Group name: Idioka Clerks

Team members: Trương Tân Hào Hiệp – 2011211

Nguyễn Minh Khoa - 2052538

Ho Chi Minh city, 10 December 2022

TABLE OF CONTENTS

1. INTRODUCTION

1.1. Problem

Following the subsiding of COVID-19, the world has seen a resurgence of globalization, one of the most prominent of which is the recovery of tourism. And accommodation has been an irreplaceable part of tourism ever since its conception. An effective, secure and extendable database system plays an important role in the efficient management of facilities, resources and customers for every hotel service in existence. Hospitals used to store their data in traditional file systems with prime examples including Microsoft Excel, Open Office, and Google spreadsheets. The main drawback of a traditional file system is that data definition is part of an application program which works only with specific applications. Files are design-driven, they require a change in design & coding whenever a new kind of data occurs.

For these systems to operate smoothly and successfully, having an efficient and reliable database management system to manage employees is crucial. Therefore, we will design a Hotel's Database system in this assignment.

1.2. Background

A Hotel Database Management System (HDBMS) is a system that helps with the management of a hotel. Customers, services and other data can be accessed by staff without much delay to enable smooth, swift and effective operation of multiple branches. It also lowers the risk of violating different constraints set out by the owner of the system. HDBMS is for computerizing the working environment and is capable of providing easy and effective storage of information related to multiple aspects of running a hotel service.

2. REQUIREMENTS ELICITATION

A resort management company for tenants needs to build a database for the following information:

– Resort has branches, each branch is located in a different province across the country. The system needs to save the address, phone number, contact email address, and pictures of each branch. Each branch has many zones. Zone names (i.e., beach area, garden, etc. or zone A, zone B, zone C…) are placed only in a branch.

– Each branch offers several room types. The number of rooms and the price of each room type may vary from branch to branch. Each type of room will have regulatory information

about: number of beds, size of each bed, maximum number of guests allowed, room area (there may be a difference from reality), list of supplies with number quantity, rent and other description. For example: family room type has 1 bed 1m8, 2 beds 1m, maximum 4 guests, 25m2, 1 refrigerator, 1 television, 1 TV remote, 1 air conditioner, 1 tea table, 4 chairs... price 1 million /day. The information of the same room type will be the same in all branches.

– Each room of the resort needs to have information about the number of rooms, the type of room, in which area, the condition of the supplies in the room (new, scratched, in need of maintenance, need to be replaced, ...). For example: room 101, Ho Chi Minh City branch is a single room type, the garden, the dressing table is scratched in the left corner, etc. The system can always look up the status information of each room at any time (also vacant or rented). Supplies at each branch also need to be managed accurately by the status of each one, the location of the supplies (in which room, or in the stockpile). Each branch is allowed to choose its own supplier for each type of material, each type of material, only one supplier. For example, in the HCMC branch, all tea tables are provided by DP Furniture, but in Da Lat branch, it is managed by TX company. The supplier, including the supplier's name, address, and email contact, will assist with the delivery and maintenance of the supplies. A supplier can cooperate with many different branches, providing different types of supplies.

– Guests can book rooms online on the resort's website. When booking, customers need to provide information: CCCD/ID, full name, phone number, contact email, branch, number of guests, check-in date, check-out date. The system will display the available room types according to the customer's requirements. Next, the customer selects the room type, the desired number of rooms, the amount to be paid will be automatically calculated. After successful booking, the system will respond to customers with booking code, date and time of booking and registered customer information. Guests need to make payment at least 36 hours before check-in date. Reservations will be canceled if payment is not made on time. Guests can cancel the reservation at least 24 hours before the check-in date and receive a full refund of the paid amount. Failure to cancel on time or failure to show up on the date of registration will result in a charge of one day's rental and cancellation of the application. Information on the date and time when guests arrive at check-in and check-out also need to be saved. In addition, the resort also offers a room rental package by year. Each package will have a price, maximum total days rented, maximum number of guests per rental. Customers who buy the package, will be able to stay at any branch at any time of the year,

the room type is arbitrarily selected from the list of available rooms, just within the allowed number of days.

– When a customer has registered for a package, at the time of booking, they will provide the name of the purchased package they want to apply to (1 booking can only choose 1 package). The system will check the registration information, the number of remaining days, and whether the number of guests allowed is valid or not. If it is not valid (number of days exceeded, number of guests exceeded), the customer needs to re-register: one application suitable for package information, another application for out of package requirements. Customer's stay information under package service needs to be saved, similar to normal room rental registration.

– For more convenience, customers can register as a member (login with username, password), to accumulate points and enjoy preferential policies of the resort. Depending on accumulated points, customers are divided into: Potential customers (<50 VND ~ 50 million VND), Loyal customers (<100 VND), VIP customers (<1000 VND) and Super VIP customers (> = 1000 VND).

+ Loyal customers get 10% off when booking.

+ VIP customers get 15% off when booking and 1 extra day when buying service package room rental (reduce 15% and increase the number of days of use).

+ Super VIP customers get 20% off when booking and 2 extra days when buying package room service.

– At each branch, the resort will reserve some space for the service party to rent for business. Site information includes: No. of ground, length, width, location in which area, more description. There are the following types of services: restaurants, spas, convenience stores, souvenirs, bars. The lessee needs to provide information to the resort: business name, store name, representative logo, image, customer service time (there may be many operating hours of the day). The restaurant needs to provide more information about the style of the dish (Vietnamese, European, Chinese, etc.), the maximum number of guests. Spa: the services provided. Souvenirs: goods, brands. The rental price depends on the location and area of the premises and is determined by each branch. A business can lease some premises at one/many branches. The lessee needs to pay the rent one month in advance at the end of the lease year to be able to continue operating for the next year. The system will send a reminder when the rental payment is near due.

– At the end of each month, the company needs to export statistical reports, such as: total monthly revenue of the whole system/ each branch, the total number of tenants of the whole system/ each branch (including the number of guests of package services), the average occupancy rate of the whole system/ each branch.

3. TOOLS

3.1. ERD model design tool

To design our database in this assignment, Draw.io will be our tool of choice. Draw.io is a web-based tool for UML, ERD, and any designed diagram. We can use either the web base or desktop application.

Although Draw.io is a paid service, we don't need premium features, the free version provides us with enough to cover this assignment.

Advantages

– Easily produces good-quality diagrams.
– Has a rich set of predefined shapes for all sorts of different diagramming needs.
– Allows grouping of shapes.
– Smart connectors.
– Integrates with Google Drive.
– Conveniently exports to a variety of formats.
– Allows for collaborative development of diagrams.

3.2. Relational Database Management System (MySQL)

For implementing the database management system, we opt for the MySQL database management system. MySQL is a relational database management system (RDBMS) developed by Oracle that is based on structured query language (SQL). MySQL is one of the most recognizable technologies in the modern big data ecosystem.

Often called the most popular database and currently enjoying widespread, effective use regardless of industry, anyone involved with enterprise data or general IT should at least aim for a basic familiarity with MySQL.

With MySQL, even those new to relational systems can immediately build fast, powerful, and secure data storage systems. MySQL's programmatic syntax and interfaces

are also perfect gateways into the wide world of other popular query languages and structured data stores. The factors that influenced our decision on choosing MySQL:

– Is the most stable and high-speed database.
– Ease of use.
– Completely free because this is open source.
– Community support.
– Works on multiple operating systems: Linux, macOS, Windows, etc.

Advantages

- **Accessibility and Ease of use:** The setting up process is relatively basic and requires less than 30 minutes, source code is completely flexible and entirely free.
- **High Efficiency:** For small to medium data quantities, the database manages to provide responsive interactions.
- **Industry standard:** MySQL is now the industry standard for database systems design and implementation.
- **Security:** Safety is always an extremely important issue in the data industry and MySQL ensures security standards.

Disadvantage

– **Debugging:** Compared to the other DBMS, MySQL, it does not handle developing and debugging as well as its counterparts.

– **Weak Enormous-Data Performance:** Although it can manage data in large quantities, MySQL is still not capable of integrating huge and systematic data management such as nationwide supermarket systems, banks, population information management, etc. National numbers …

3.3. MySQL Workbench

MySQL Workbench is a cross-platform GUI client for MySQL database users and administrators. Workbench makes the task easier for database admins by providing important tools to manage databases and users, create databases, run SQL queries, set up and configure servers, and many more.

It is a powerful tool that enables us to visualize modules for creating, executing, and optimizing several queries. So, in this article, I am going to give a tour of MySQL Workbench and show you how to use it.

Key MySQL Workbench Features

– **SQL Development:** This enables you to create and manage connections to database servers. Along with enabling you to configure connection parameters, MySQL Workbench provides the capability to execute SQL queries on the database connections using the built-in SQL Editor

– **Data Modeling (Design):** Enables you to create models of your database schema graphically, reverse and forward engineer between a schema and a live database, and edit all aspects of your database using the comprehensive Table Editor. TheTable Editor provides easy-to-use facilities for editing Tables, Columns, Indexes, Triggers, Partitioning, Options, Inserts and Privileges, Routines and Views

– **Server Administration:** Enables you to administer MySQL server instances by administering users, performing backup and recovery, inspecting audit data, viewing database health, and monitoring the MySQL server performance.

– **Data Migration:** Allows you to migrate from Microsoft SQL Server, Microsoft Access, Sybase ASE, SQLite, SQL Anywhere, PostgreSQL, and other RDBMS tables, objects and data to MySQL. Migration also supports migrating from earlier versions of MySQL to the latest releases.

– **MySQL Enterprise Support:** Support for Enterprise products such as MySQL Enterprise Backup, MySQL Firewall, and MySQL Audit.

4. IMPLEMENTATION

4.1. Create Schema

We first create a schema and access it before implementing the database using MySQL.

```
DROP DATABASE IF EXISTS MYHOTEL;
CREATE DATABASE MYHOTEL;
USE MYHOTEL;
```

4.2. Create Tables

There are 26 tables in total as will be listed below.

4.2.1. Chi nhánh (Mã chi nhánh, tỉnh, địa chỉ, điện thoại, email)

For the MaChiNhanh attribute, the type will be converted to varchar type and appended the prefix ‘CN’ during the insertion phase.

```
-- @block 1 --
CREATE TABLE IF NOT EXISTS CHINHANH (
    MaChiNhanh INT NOT NULL AUTO_INCREMENT,
    Tinh VARCHAR(50) NOT NULL,
    DiaChi VARCHAR(255) NOT NULL,
    DienThoai INT NOT NULL,
    Email VARCHAR(50) NOT NULL,
    CONSTRAINT PK_CHINHANH PRIMARY KEY (MaChiNhanh)
);
```

4.2.2. Hình ảnh chi nhánh (Mã chi nhánh, hình ảnh)

```
-- @block 2 --
CREATE TABLE IF NOT EXISTS HINHANH_CHINHANH (
    HA_MCN VARCHAR(50),
    HinhAnh VARCHAR(255) NOT NULL,
    CONSTRAINT PK_HINHANH PRIMARY KEY (HinhAnh),
    CONSTRAINT FK_HA_MCN FOREIGN KEY (HA_MCN)
        REFERENCES CHINHANH (MaChiNhanh)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

4.2.3. Khu (Mã chi nhánh, tên khu)

```
-- @block 3 --
CREATE TABLE IF NOT EXISTS KHU (
    Khu_MCN VARCHAR(50),
    TenKhu VARCHAR(50) NOT NULL,
    CONSTRAINT PK_KHU PRIMARY KEY (TenKhu),
    CONSTRAINT FK_Khu_MCN FOREIGN KEY (Khu_MCN)
        REFERENCES CHINHANH (MaChiNhanh)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

4.2.4. Loại phòng (Mã loại phòng, tên loại phòng, diện tích, số khách, mô tả khác)

```
-- @block 4 --
CREATE TABLE IF NOT EXISTS LOAIPHONG (
    MaLoaiPhong INT NOT NULL AUTO_INCREMENT,
    TenLoaiPhong VARCHAR(50) NOT NULL,
    DienTich DECIMAL(4 , 1 ) NOT NULL,
    SoKhach INT NOT NULL CHECK (SoKhach BETWEEN 1 AND 10),
    MoTaKhac VARCHAR(255) DEFAULT NULL,
    CONSTRAINT PK_LOAIPHONG PRIMARY KEY (MaLoaiPhong)
);
```

4.2.5. Thông tin giường (Mã loại phòng, kích thước, số lượng)

```
-- @block 5 --
CREATE TABLE IF NOT EXISTS THONGTINGIUONG (
    TTG_MLP INT,
    KichThuoc DECIMAL(2 , 1 ) NOT NULL,
    SoLuong INT NOT NULL DEFAULT 1 CHECK (SoLuong BETWEEN 1 AND 10),
    CONSTRAINT PK_TTG PRIMARY KEY (KichThuoc),
    CONSTRAINT FK_TTG_MLP FOREIGN KEY (TTG_MLP)
        REFERENCES LOAIPHONG (MaLoaiPhong)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

4.2.6. Chi nhánh có loại phòng (Mã loại phòng, mã chi nhánh, giá thuê)

```
-- @block 6 --
CREATE TABLE IF NOT EXISTS CHINHANH_CO_LOAIPHONG (
    Co_MLP INT,
    Co_MCN VARCHAR(50),
    GiaThue INT NOT NULL CHECK (GiaThue > - 1),
    CONSTRAINT FK_Co_MLP FOREIGN KEY (Co_MLP)
        REFERENCES LOAIPHONG (MaLoaiPhong)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_Co_MCN FOREIGN KEY (Co_MCN)
        REFERENCES CHINHANH (MaChiNhanh)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

4.2.7. Phòng (Mã chi nhánh, số phòng, mã loại phòng, tên khu)

```
-- @block 7 --
CREATE TABLE IF NOT EXISTS PHONG (
    Phong_MCN VARCHAR(50),
    SoPhong VARCHAR(3) UNIQUE NOT NULL,
    Phong_TK VARCHAR(50) NOT NULL,
    Phong_MLP INT NOT NULL,
    CONSTRAINT FK_Phong_MCN FOREIGN KEY (Phong_MCN)
        REFERENCES KHU (Khu_MCN)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_Phong_TK FOREIGN KEY (Phong_TK)
        REFERENCES KHU (TenKhu)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_Phong_MLP FOREIGN KEY (Phong_MLP)
        REFERENCES LOAIPHONG (MaLoaiPhong)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT PK_PHONG PRIMARY KEY (SoPhong)
);
```

4.2.8. Loại vật tư (Mã loại vật tư, tên loại vật tư)

For LoatVatTu attribute, the type will be converted to varchar type and appended the prefix 'VT' during the insertion phase.

```
-- @block 8 -- MaLoaiVatTu is of type VT0001
CREATE TABLE IF NOT EXISTS LOAIVATTU (
    MaLoaiVatTu INT(4) ZEROFILL NOT NULL,
    TenLoaiVatTu VARCHAR(50) NOT NULL,
    CONSTRAINT PK_LOAIVATTU PRIMARY KEY (MaLoaiVatTu)
);
```

4.2.9. Loại vật tư trong loại phòng (Mã loại vật tư, mã loại phòng, số lượng)

```
-- @block 9 --
CREATE TABLE IF NOT EXISTS LOAIVATTU_TRONG_LOAIPHONG (
    Trong_MLVT VARCHAR(6),
    Trong_MLP INT,
    SoLuong INT NOT NULL DEFAULT 1 CHECK (SoLuong BETWEEN 1 AND 20),
    CONSTRAINT FK_Trong_MLVT FOREIGN KEY (Trong_MLVT)
        REFERENCES LOAIVATTU (MaLoaiVatTu)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_Trong_MLP FOREIGN KEY (Trong_MLP)
        REFERENCES LOAIPHONG (MaLoaiPhong)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

4.2.10. Vật tư (Mã chi nhánh, mã loại vật tư, STT vật tư, tình trạng, số phòng)

```
-- @block 10 -- references to chinhanh maybe redundant
CREATE TABLE IF NOT EXISTS VATTU (
    VT_MCN VARCHAR(50),
    VT_MLVT VARCHAR(6),
    SttVatTu INT NOT NULL CHECK (SttVatTu > - 1),
    TinhTrang VARCHAR(50) NOT NULL,
    VT_SP VARCHAR(3),
    CONSTRAINT FK_VT_MLVT FOREIGN KEY (VT_MLVT)
        REFERENCES LOAIVATTU (MaLoaiVatTu)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_VT_MCN_SP FOREIGN KEY (VT_MCN , VT_SP)
        REFERENCES PHONG (Phong_MCN , SoPhong)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT PK_VATTU PRIMARY KEY (SttVatTu)
);
```

4.2.11. Nhà cung cấp (Mã nhà cung cấp, tên nhà cung cấp, email, địa chỉ)

For MaNhaCungCap attribute, the type will be converted to varchar type and appended the prefix 'NCC' during the insertion phase.

```
-- @block 11 -- MaNhaCungCap is of type NCC0001
CREATE TABLE IF NOT EXISTS NHACUNGCAP (
    MaNhaCungCap INT(4) ZEROFILL NOT NULL,
    TenNhaCungCap VARCHAR(50) NOT NULL,
    Email VARCHAR(50),
    DiaChi VARCHAR(255) NOT NULL,
    CONSTRAINT PK_NHACUNGCAP PRIMARY KEY (MaNhaCungCap)
);
```

4.2.12. Cung cấp vật tư (Mã nhà cung cấp, mã loại vật tư, mã chi nhánh)

```
-- @block 12 --
CREATE TABLE IF NOT EXISTS CUNGCAPVATTU (
    CCVT_MNCC VARCHAR(7),
    CCVT_MLVT VARCHAR(6),
    CCVT_MCN VARCHAR(50),
    CONSTRAINT FK_CCVT_MNCC FOREIGN KEY (CCVT_MNCC)
        REFERENCES NHACUNGCAP (MaNhaCungCap)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_CCVT_MLVT FOREIGN KEY (CCVT_MLVT)
        REFERENCES LOAIVATTU (MaLoaiVatTu)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_CCVT_MCN FOREIGN KEY (CCVT_MCN)
        REFERENCES CHINHANH (MaChiNhanh)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

4.2.13. Khách hàng (Mã khách hàng, CCCD/CMND, họ tên, điện thoại, email, username, password, điểm, loại)

```
-- @block 13 -- MaKhachHang is of type KH000001
CREATE TABLE IF NOT EXISTS KHACHHANG (
    MaKhachHang VARCHAR(8) NOT NULL,
    CCCD VARCHAR(12) NOT NULL UNIQUE,
    HoTen VARCHAR(50) NOT NULL,
    Email VARCHAR(50) NOT NULL UNIQUE,
    Username VARCHAR(50) NOT NULL UNIQUE,
    Password VARCHAR(255) NOT NULL,
    Diem INT NOT NULL DEFAULT 0 CHECK (Diem > - 1),
    Loai INT NOT NULL DEFAULT 1 CHECK (Loai BETWEEN 1 AND 4),
    CONSTRAINT PK_KHACHHANG PRIMARY KEY (MaKhachHang)
);
```

4.2.14. Gói dịch vụ (Tên gói, số ngày, số khách, giá)

```
-- @block 14 --
CREATE TABLE IF NOT EXISTS GOIDICHVU (
    TenGoi VARCHAR(50),
    SoNgay INT NOT NULL CHECK (SoNgay BETWEEN 1 AND 100),
    SoKhach INT NOT NULL CHECK (SoKhach BETWEEN 1 AND 10),
    Gia DECIMAL(12 , 1 ) NOT NULL,
    CONSTRAINT PK_GOIDICHVU PRIMARY KEY (TenGoi)
);
```

4.2.15. Hoá đơn gói dịch vụ (mã khách hàng, tên gói, ngày giờ mua, ngày bắt đầu, tổng tiền)

```
-- @block 15 --
CREATE TABLE IF NOT EXISTS HOADONGOIDICHVU (
    HDGDV_MKH VARCHAR(8),
    HDGDV_TG VARCHAR(50),
    NgayGioMua DATETIME NOT NULL,
    NgayBatDau DATETIME NOT NULL,
    TongTien INT NOT NULL,
    SoNgaySuDungConLai INT DEFAULT NULL,
    CONSTRAINT FK_HDGDV_MKH FOREIGN KEY (HDGDV_MKH)
        REFERENCES KHACHHANG (MaKhachHang)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_HDGDV_TG FOREIGN KEY (HDGDV_TG)
        REFERENCES GOIDICHVU (TenGoi)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT PK_HOADONGOIDICHVU PRIMARY KEY (NgayGioMua)
);
```

4.2.16. Đơn đặt phòng (Mã đặt phòng, ngày giờ đặt, số khách, ngày nhận phòng, ngày trả phòng, tình trạng, tổng tiền, mã khách hàng, tên gói dịch vụ)

For MaDatPhong attribute, the type will be converted to varchar type and appended the prefix ‘DP’ during the insertion phase.

```
-- @block 16 --
CREATE TABLE IF NOT EXISTS DONDATPHONG (
    MaDatPhong INT(6) ZEROFILL AUTO_INCREMENT,
    NgayGioDat DATETIME NOT NULL,
    NgayNhanPhong DATETIME NOT NULL,
    NgayTraPhong DATETIME NOT NULL,
    TinhTrang INT NOT NULL CHECK (TinhTrang BETWEEN 0 AND 3),
    TongTien INT NOT NULL DEFAULT 0 CHECK (TongTien > - 1),
    DDP_MKH VARCHAR(8),
    DDP_TG VARCHAR(50),
    SoKhach INT NOT NULL CHECK (SoKhach > - 1),
    CONSTRAINT FK_DDP_MKH FOREIGN KEY (DDP_MKH)
        REFERENCES KHACHHANG (MaKhachHang)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_DDP_TG FOREIGN KEY (DDP_TG)
        REFERENCES GOIDICHVU (TenGoi)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT PK_DONDATPHONG PRIMARY KEY (MaDatPhong)
);
```

4.2.17. Phòng thuê (Mã đặt phòng, mã chi nhánh, số phòng)

```
-- @block 17 --
CREATE TABLE IF NOT EXISTS PHONGTHUE (
    PT_MDP VARCHAR(16),
    PT_MCN VARCHAR(50),
    PT_SP VARCHAR(3) NOT NULL,
    CONSTRAINT FK_PT_MDP FOREIGN KEY (PT_MDP)
        REFERENCES DONDATPHONG (MaDatPhong)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_PT_MCN FOREIGN KEY (PT_MCN)
        REFERENCES PHONG (Phong_MCN)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_PT_SP FOREIGN KEY (PT_SP)
        REFERENCES PHONG (SoPhong)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

4.2.18. Hoá đơn (Mã hoá đơn, thời gian nhận phòng, thời gian trả phòng, mã đặt phòng)

For MaHoaDon attribute, the type will be converted to varchar type and appended the prefix ‘HD’ along with the formatted current date during the insertion phase.

```
-- @block 18 --
CREATE TABLE IF NOT EXISTS HOADON (
    MaHoaDon INT(6) ZEROFILL AUTO_INCREMENT,
    ThoiGianNhanPhong TIME,
    ThoiGianTraPhong TIME,
    HD_MDP VARCHAR(16),
    CONSTRAINT FK_HD_MDP FOREIGN KEY (HD_MDP)
        REFERENCES DONDATPHONG (MaDatPhong)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT PRIMARY KEY (MaHoaDon)
);
```

4.2.19. Doanh nghiệp (Mã doanh nghiệp, tên doanh nghiệp)

For MaDoanhNghiep attribute, the type will be converted to varchar type and appended the prefix ‘DN’ during the insertion phase.

```
-- @block 19 --
CREATE TABLE IF NOT EXISTS DOANHNGHIEP (
    MaDoanhNghiep INT(4) ZEROFILL NOT NULL,
    TenDoanhNghiep VARCHAR(100),
    CONSTRAINT PK_DOANHNGHIEP PRIMARY KEY (MaDoanhNghiep)
);
```

4.2.20. Dịch vụ (Mã dịch vụ, loại dịch vụ, số khách, phong cách, mã doanh nghiệp)

```
-- @block 20 --
CREATE TABLE IF NOT EXISTS DICHVU (
    MaDichVu VARCHAR(6) NOT NULL,
    LoaiDichVu VARCHAR(1) NOT NULL,
    SoKhach INT DEFAULT 0 CHECK (SoKhach > - 1),
    PhongCach VARCHAR(255),
    DV_MDN VARCHAR(6),
    CONSTRAINT FK_DV_MDN FOREIGN KEY (DV_MDN)
        REFERENCES DOANHNGHIEP (MaDoanhNghiep)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT PK_DICHVU PRIMARY KEY (MaDichVu)
);
```

4.2.21. Dịch vụ Spa (Mã dịch vụ, dịch vụ spa)

```
-- @block 21 --
CREATE TABLE IF NOT EXISTS DICHVUSPA (
    DVS_MDV VARCHAR(6) NOT NULL,
    DichVuSpa VARCHAR(255),
    CONSTRAINT FK_DVS_MDV FOREIGN KEY (DVS_MDV)
        REFERENCES DICHVU (MaDichVu)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT PK_DICHVUSPA PRIMARY KEY (DichVuSpa)
);
```

4.2.22. Loại hàng đồ lưu niệm (Mã dịch vụ, loại hàng)

```
-- @block 22 --
CREATE TABLE IF NOT EXISTS LOAIHANGDOLUUNIEM (
    LHDLN_MDV VARCHAR(6) NOT NULL,
    LoaiHang VARCHAR(255),
    CONSTRAINT FK_LHDLN_MDV FOREIGN KEY (LHDLN_MDV)
        REFERENCES DICHVU (MaDichVu)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT PK_LOAIHANGDOLUUNIEM PRIMARY KEY (LoaiHang)
);
```

4.2.23. Thương hiệu đồ lưu niệm (Mã dịch vụ, thương hiệu)

```
-- @block 23 --
CREATE TABLE IF NOT EXISTS THUONGHIEUDOLUUNIEM (
    THDLN_MDV VARCHAR(6) NOT NULL,
    ThuongHieu VARCHAR(100),
    CONSTRAINT FK_THDLN_MDV FOREIGN KEY (THDLN_MDV)
        REFERENCES DICHVU (MaDichVu)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT PK_THUONGHIEUDOLUUNIEM PRIMARY KEY (ThuongHieu)
);
```

4.2.24. Mặt bằng (Mã chi nhánh, STT mặt bằng, chiều dài, chiều rộng, giá thuê, mô tả, mã dịch vụ, tên cửa hàng, logo)

```
-- @block 24 --
CREATE TABLE IF NOT EXISTS MATBANG (
    MB_MCN VARCHAR(50) NOT NULL,
    STTMatBang INT NOT NULL UNIQUE DEFAULT 1 CHECK (STTMatBang BETWEEN 1
AND 50),
    ChieuDai INT NOT NULL,
    ChieuRong INT NOT NULL,
    GiaThue INT NOT NULL CHECK (GiaThue > - 1),
    MoTa VARCHAR(255),
    MB_MDV VARCHAR(6) NOT NULL,
    TenCuaHang VARCHAR(255),
    Logo VARCHAR(255),
    CONSTRAINT FK_MB_MDV FOREIGN KEY (MB_MDV)
        REFERENCES DICHVU (MaDichVu)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_MB_MCN FOREIGN KEY (MB_MCN)
        REFERENCES CHINHANH (MaChiNhanh)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT PK_MATBANG PRIMARY KEY (MB_MCN)
);
```

4.2.25. Hình ảnh cửa hàng (Mã chi nhánh, STT mặt bằng, hình ảnh)

```
-- @block 25 --
CREATE TABLE IF NOT EXISTS HINHANHCUAHANG (
    HACH_MCN VARCHAR(50) NOT NULL,
    HACH_STTMatBang INT NOT NULL DEFAULT 1,
    HinhAnh VARCHAR(255),
    CONSTRAINT FK_HACH_MCN FOREIGN KEY (HACH_MCN)
        REFERENCES MATBANG (MB_MCN)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_HACK_STTMatBang FOREIGN KEY (HACH_STTMatBang)
        REFERENCES MATBANG (STTMatBang)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT PK_HINHANHCUAHANG PRIMARY KEY (HinhAnh)
);
```

4.2.26. Khung giờ hoạt động cửa hàng (Mã chi nhánh, STT mặt bằng, giờ bắt đầu, giờ kết thúc)

```
-- @block 26 --
CREATE TABLE IF NOT EXISTS KHUNGGIOHOATDONG (
    KGHD_MCN VARCHAR(50) NOT NULL,
    KGHD_STTMatBang INT NOT NULL DEFAULT 1,
    GioBatDau TIME,
    GioKetThuc TIME,
    CONSTRAINT FK_KGHD_MCN FOREIGN KEY (KGHD_MCN)
        REFERENCES MATBANG (MB_MCN)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT FK_KGHD_STTMatBang FOREIGN KEY (KGHD_STTMatBang)
        REFERENCES MATBANG (STTMatBang)
        ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT PK_KHUNGGIOHOATDONG PRIMARY KEY (GioBatDau)
);
```

4.3. Insert Data

For each table, at least 4 tuples of data were added to verify the functionality of the tables created. The data corresponding to each table was inserted right after the script for creating that table. Any triggers positioning will be mentioned in the latter parts.

For the enhancement of the reading experience, data insertion scripts will be combined into larger scripts shown below.

4.3.1. Data insertion for table 1

Here the aforementioned changes to MaChiNhanh applied at the end of the insertion.

```
INSERT INTO CHINHANH (Tinh, DiaChi, DienThoai, Email)
VALUES ('Alime','255 Jandex Street, District
13',0901392331,'ChienHugo111@gmail.com'),
    ('Binary','1080 Pixel Street, District
144',0800281220,'DuySocket222@yahoo.com'),
    ('Canary','720 HDMI Street, District
6',0833281520,'LokLok322@hcmut.edu.vn'),
    ('Dharma','480 Low Res Street, District
64',091221120,'ZuDuYu52@icloud.com');
ALTER TABLE CHINHANH
MODIFY COLUMN MaChiNhanh varchar(50);
UPDATE CHINHANH
SET
    MaChiNhanh = CONCAT('CN', MaChiNhanh);
```

4.3.2. Data insertion for table 2 to 7

```
INSERT INTO HINHANH_CHINHANH (HA_MCN, HinhAnh)
VALUES ('CN1', 'youtube.com'),
    ('CN2', 'amazon.com'),
    ('CN3', 'facebook.com'),
    ('CN4', 'shoppe.vn');

INSERT INTO KHU (Khu_MCN, TenKhu)
VALUES ('CN1', 'THANH HOA'),
    ('CN2', 'BRAZIL'),
    ('CN3', 'HAI PHONG'),
    ('CN4', 'HELL');

INSERT INTO LOAIPHONG (TenLoaiPhong, DienTich, SoKhach, MoTaKhac)
VALUES ('PHONG 1', 10.0, 1, 'PHONG CHO NGUOI CO DON'),
    ('PHONG 2',30.5,5,'PHONG CHO 5 AE TRONG 1 CHIEC XE TANK'),
    ('PHONG 3', 60.0, 5, '5 AE NHUNG MA GIAU HON'),
    ('PHONG 4', 100.0, 1, 'PHONG CHO TI PHU');

INSERT INTO THONGTINGIUONG (TTG_MLP, KichThuoc, SoLuong)
VALUES (1, 1.0, 1),
    (2, 2.0, 1),
    (3, 3.0, 1),
    (4, 4.0, 1);

INSERT INTO CHINHANH_CO_LOAIPHONG (Co_MLP, Co_MCN, GiaThue)
VALUES (1, 'CN1', 1000),
    (2, 'CN2', 2000),
    (3, 'CN3', 3000),
    (4, 'CN4', 4000);

INSERT INTO PHONG (Phong_MCN, SoPhong, Phong_TK, Phong_MLP)
VALUES ('CN1', '101', 'THANH HOA', 1),
    ('CN2', '202', 'BRAZIL', 2),
    ('CN3', '303', 'HAI PHONG', 3),
    ('CN4', '404', 'HELL', 4);
```

4.3.3. Data insertion for table 8

Here the aforementioned changes to LoaiVatTu applied at the end of the insertion.

```
INSERT INTO LOAIVATTU (MaLoaiVatTu, TenLoaiVatTu)
VALUES (1, 'Vat tu 1'),
    (2, 'Vat tu 2'),
    (3, 'Vat tu 3'),
    (4, 'Vat tu 4');
ALTER TABLE LOAIVATTU
MODIFY COLUMN MaLoaiVatTu VARCHAR(6);
UPDATE LOAIVATTU
SET
    MaLoaiVatTu = CONCAT('VT', MaLoaiVatTu);
```

4.3.4. Data insertion for table 9 and 10

```
INSERT INTO LOAIVATTU_TRONG_LOAIPHONG(Trong_MLVT, Trong_MLP, SoLuong)
VALUES ('VT0001', 1, DEFAULT),
    ('VT0002', 2, 2),
    ('VT0003', 3, 3),
    ('VT0004', 4, 4);

INSERT INTO VATTU (VT_MCN, VT_MLVT, STTVatTu, TinhTrang, VT_SP)
VALUES ('CN1', 'VT0001', 1, 'TOT', 101),
    ('CN2', 'VT0002', 2, 'TOT', 202),
    ('CN3', 'VT0003', 3, 'TOT', 303),
    ('CN4', 'VT0004', 4, 'TOT', 404);
```

4.3.5. Data insertion for table 11

Here the aforementioned changes to MaNhaCungCap applied at the end of the insertion.

```
INSERT INTO NHACUNGCAP (MaNhaCungCap, TenNhaCungCap, Email, DiaChi)
VALUES (1, 'Hiep', 'tthh@gmail.com', '1 ltk'),
    (2, 'Khoa', 'nmmk@gmail.com', '2 ltk'),
    (3, 'Duy', 'lkd@gmail.com', '3 ltk'),
    (1000, 'Hung', 'tpmh@gmail.com', '4 ltk');
ALTER TABLE NHACUNGCAP
MODIFY COLUMN MaNhaCungCap VARCHAR(7);
UPDATE NHACUNGCAP
SET
    MaNhaCungCap = CONCAT('NCC', MaNhaCungCap);
```

4.3.6. Data insertion for table 12 to 15

```
INSERT INTO CUNGCAPVATTU (CCVT_MNCC, CCVT_MLVT, CCVT_MCN)
VALUES ('NCC0001', 'VT0001', 'CN1'),
    ('NCC0002', 'VT0002', 'CN2'),
    ('NCC0003', 'VT0003', 'CN3'),
    ('NCC1000', 'VT0004', 'CN4');

INSERT INTO CUNGCAPVATTU (CCVT_MNCC, CCVT_MLVT, CCVT_MCN)
VALUES ('NCC0001', 'VT0001', 'CN1'),
    ('NCC0002', 'VT0002', 'CN2'),
    ('NCC0003', 'VT0003', 'CN3'),
    ('NCC1000', 'VT0004', 'CN4');

INSERT INTO GOIDICHVU (TenGoi, SoNgay, SoKhach, Gia)
VALUES ('Goi1', 10, 1, 10000),
    ('Goi2', 25, 2, 20000),
    ('Goi3', 67, 3, 30000),
    ('Goi4', 80, 4, 40000);

VALUES ('KH000001','Goi1','2022-1-1 23:59:59','2022-1-2 23:59:59',10000),
    ('KH000002','Goi2','2022-1-14 23:59:59','2022-1-15 23:59:59',20000),
    ('KH000003','Goi3','2022-2-26 23:59:59','2022-2-27 23:59:59',30000),
    ('KH000004','Goi4','2022-9-7 23:59:59','2022-9-7 23:59:59',40000);

INSERT INTO HOADONGOIDICHVU (HDGDV_MKH, HDGDV_TG, NgayGioMua, NgayBatDau,
TongTien)
VALUES ('KH000001','Goi1','2022-1-1 23:59:59','2022-1-2 23:59:59',10000),
    ('KH000002','Goi2','2022-1-14 23:59:59','2022-1-15 23:59:59',20000),
    ('KH000003','Goi3','2022-2-26 23:59:59','2022-2-27 23:59:59',30000),
    ('KH000004','Goi3','2022-9-7 23:59:59','2022-9-7 23:59:59',40000);
```

4.3.7. Data insertion for table 16

Here the aforementioned changes to MaDatPhong applied at the end of the insertion.

```
INSERT INTO DONDATPHONG (NgayGioDat, NgayNhanPhong, NgayTraPhong, TinhTrang,
TongTien, DDP_MKH, DDP_TG, SoKhach)
VALUES ('2022-02-13 01:51:10','2022-06-12 12:09:20','2022-09-19
11:01:05',0,1500,'KH000001',NULL,2),
    ('2022-03-03 19:03:34','2022-06-24 04:36:37','2022-09-30
00:38:31',1,2300,'KH000002',NULL,4),
    ('2022-04-10 09:00:02','2022-08-18 01:41:43','2022-10-20
07:24:45',2,0,'KH000003','Goi3',8),
    ('2022-05-02 02:27:49','2022-09-08 18:37:16','2022-10-30
18:35:19',3,0,'KH000004','Goi4',6);
ALTER TABLE DONDATPHONG
MODIFY COLUMN MaDatPhong VARCHAR(16);
UPDATE DONDATPHONG
SET
    MaDatPhong = CONCAT('DP', '28112022', MaDatPhong);
```

4.3.8. Data insertion for table 17 to 19

Here the aforementioned changes to MaHoaDon and MaDoanhNghiep applied at the end of the insertion.

```
INSERT INTO PHONGTHUE (PT_MDP, PT_MCN, PT_SP)
VALUES ('DP28112022000001', 'CN1', 101),
    ('DP28112022000002', 'CN2', 202),
    ('DP28112022000003', 'CN3', 303),
    ('DP28112022000004', 'CN4', 404);

INSERT INTO HOADON (ThoiGianNhanPhong, ThoiGianTraPhong, HD_MDP)
VALUES ('12:09:20', '11:01:05', 'DP28112022000001'),
    ('04:36:37', '00:38:31', 'DP28112022000002'),
    ('01:41:43', '07:24:45', 'DP28112022000003'),
    ('18:37:16', '18:35:19', 'DP28112022000004');
ALTER TABLE HOADON
MODIFY COLUMN MaHoaDon VARCHAR(16);
UPDATE HOADON
SET
    MaHoaDon = CONCAT('HD', '28112022', MaHoaDon);

INSERT INTO DOANHNGHIEP (MaDoanhNghiep, TenDoanhNghiep)
VALUES (1, 'Google'),
    (2, 'Meta'),
    (3, 'Apple'),
    (4, 'Microsoft');
ALTER TABLE DOANHNGHIEP
MODIFY COLUMN MaDoanhNghiep VARCHAR(6);
UPDATE DOANHNGHIEP
SET
    MaDoanhNghiep = CONCAT('DN', MaDoanhNghiep);
```

Although we used ‘28112022’ in the snippet above when modifying MaHoaDon, replacing it with CURDATE () + 0 will make the field take the date of insertion instead.

4.3.9. Data insertion for table 20 to 26

```
INSERT INTO DICHVU (MaDichVu, LoaiDichVu, SoKhach, PhongCach, DV_MDN)
VALUES ('DVS001', 'S', 1, 'MX', 'DN0003'),
    ('DVS002', 'S', 1, 'NN', 'DN0003'),
    ('DVS003', 'S', 1, 'CD', 'DN0003'),
    ('DVS004', 'S', 1, 'XH', 'DN0003'),
    ('DVM001', 'M', 1, 'MK', 'DN0004'),
    ('DVM002', 'M', 1, 'NV', 'DN0004'),
    ('DVM003', 'M', 1, 'AT', 'DN0004'),
    ('DVM004', 'M', 1, 'CA', 'DN0004');

INSERT INTO DICHVUSPA (DVS_MDV, DichVuSpa)
VALUES ('DVS001', 'MAT XA'),
    ('DVS002', 'NGAM NUOC'),
    ('DVS003', 'CHUOM DA'),
    ('DVS004', 'XONG HOI');

INSERT INTO LOAIHANGDOLUUNIEM (LHDLN_MDV, LoaiHang)
VALUES ('DVM001', 'MOC KHOA'),
    ('DVM002', 'NON VAI'),
    ('DVM003', 'AO THUN'),
    ('DVM004', 'CHUP ANH');

INSERT INTO THUONGHIEUDOLUUNIEM (THDLN_MDV, ThuongHieu)
VALUES ('DVM001', 'GUCCI'),
    ('DVM002', 'DOLCE'),
    ('DVM003', 'COOLMATE'),
    ('DVM004', 'FUMA');

INSERT INTO MATBANG (MB_MCN, STTMatBang, ChieuDai, ChieuRong, GiaThue, MoTa,
MB_MDV, TenCuaHang, Logo)
VALUES ('CN1',1,5,10,1000,'NULL','DVM001','NON SON','A.COM'),
    ('CN2',2,10,15,2000,'NULL','DVM002','PHUC LONG','B.COM'),
    ('CN3',3,15,20,3000,'NULL','DVM003','GOGI','C.COM'),
    ('CN4',4,20,25,4000,'NULL','DVM004','OZ','D.COM');

INSERT INTO HINHANHCUAHANG (HACH_MCN, HACH_STTMatBang, HinhAnh)
VALUES ('CN1', 1, 'a.com'),
    ('CN2', 2, 'b.com'),
    ('CN3', 3, 'c.com'),
    ('CN4', 4, 'd.com');

INSERT INTO KHUNGGIOHOATDONG (KGHD_MCN, KGHD_STTMatBang, GioBatDau, GioKetThuc)
VALUES ('CN1', 1, '7:00:00', '17:00:00'),
    ('CN2', 2, '7:15:00', '17:00:00'),
    ('CN3', 3, '7:30:00', '17:00:00'),
```

4.4. Query, Trigger and Store Procedure

4.4.1. Store Procedure

a) Create a store procedure to compute the current info of all available GoiDichVu of a customer

```
-- @block STORED PROCEDURE 1 --
DELIMITER \\
DROP PROCEDURE IF EXISTS GoiDichVu\\
CREATE PROCEDURE GoiDichVu (IN MaKhachHang VARCHAR(8))
BEGIN
    DECLARE count int DEFAULT 0;
    SET count = (SELECT COUNT(HDGDV_MKH) FROM HOADONGOIDICHVU WHERE HDGDV_MKH
= MaKhachHang);
    IF count > 0 THEN
    SELECT HOADONGOIDICHVU.HDGDV_TG AS 'Tên Gói', GOIDICHVU.SoKhach AS 'Số
Khách', HOADONGOIDICHVU.NgayBatDau AS 'Ngày Bắt đầu',
            ADDDATE(HOADONGOIDICHVU.NgayBatDau, INTERVAL GOIDICHVU.SoNgay
DAY) AS 'Ngày kết thúc', HOADONGOIDICHVU.SoNgaySuDungConLai AS 'Số ngày sử
dụng còn lại'
    FROM (HOADONGOIDICHVU INNER JOIN GOIDICHVU ON HOADONGOIDICHVU.HDGDV_TG =
GOIDICHVU.TenGoi)
    WHERE HOADONGOIDICHVU.HDGDV_MKH = MaKhachHang;
    ELSE SELECT CONCAT('YOUR PARAMETER ',MaKhachHang,' DOES NOT EXIST!') AS
'ERROR';
    END IF;
END\\
```

b) Create a store procedure to calculate the total number of customers of a branch in the given business year

```
-- @block STORED PROCEDURE 2 --
DROP PROCEDURE IF EXISTS ThongKeLuotKhach\\
CREATE PROCEDURE ThongKeLuotKhach (IN MCN VARCHAR(50), IN NamThongKe INT(5))
BEGIN
    SELECT MONTH(NgayNhanPhong) AS 'Tháng', SUM(SoKhach) AS 'Tổng số lượt
khách'
    FROM (SELECT TinhTrang, SoKhach, PT_MCN, NgayNhanPhong FROM DONDATPHONG
INNER JOIN PHONGTHUE ON MaDatPhong = PT_MDP) AS T
    WHERE PT_MCN = MCN AND TinhTrang = 1 AND YEAR(NgayNhanPhong) = NamThongKe
    GROUP BY MONTH(NgayNhanPhong)
    ORDER BY MONTH(NgayNhanPhong);
END\\
DELIMITER ;
```

4.4.2. Trigger

a) Create a trigger to calculate the total cost of GoiDichVu in table 15

```
-- @block TRIGGER 1a --
DELIMITER \\
DROP TRIGGER IF EXISTS update_TongTienGoiDichVu\\
CREATE TRIGGER update_TongTienGoiDichVu
BEFORE INSERT ON HOADONGOIDICHVU FOR EACH ROW
BEGIN
    DECLARE LoaiKH INT;
    DECLARE SoNgayConLai INT;
    DECLARE SoTien INT;
    SET LoaiKH = (SELECT Loai FROM KHACHHANG WHERE KHACHHANG.MaKhachHang =
NEW.HDGDV_MKH);
    SET SoNgayConLai = (SELECT SoNgay FROM GOIDICHVU WHERE GOIDICHVU.TenGoi =
NEW.HDGDV_TG);
    IF LoaiKH = 2 THEN SET SoTien = ((SELECT Gia FROM GOIDICHVU WHERE
GOIDICHVU.TenGoi = NEW.HDGDV_TG) * 9 / 10);
    ELSEIF LoaiKH = 3 THEN SET SoTien = ((SELECT Gia FROM GOIDICHVU WHERE
GOIDICHVU.TenGoi = NEW.HDGDV_TG) * 17 / 20),
                            SoNgayConLai = SoNgayConLai + 1;
    ELSEIF LoaiKH = 4 THEN SET SoTien = ((SELECT Gia FROM GOIDICHVU WHERE
GOIDICHVU.TenGoi = NEW.HDGDV_TG) * 4 / 5),
                            SoNgayConLai = SoNgayConLai + 2;
    ELSE SET SoTien = NEW.TongTien;
    END IF;
    SET NEW.TongTien = SoTien;
    SET NEW.SoNgaySuDungConLai = SoNgayConLai;
END\\
DELIMITER ;
```

b) Create a trigger to calculate the total cost of DonDatPhong in table 16

```
-- @block TRIGGER 1b --
DELIMITER \\
DROP TRIGGER IF EXISTS update_TongTienDonDatPhong\\
CREATE TRIGGER update_TongTienDonDatPhong
BEFORE INSERT ON DONDATPHONG FOR EACH ROW
BEGIN
    DECLARE LoaiKH INT;
    DECLARE SoNgayConLai INT;
    DECLARE SoTien INT;
    SET LoaiKH = (SELECT Loai FROM KHACHHANG WHERE KHACHHANG.MaKhachHang =
NEW.DDP_MKH);
    IF NEW.DDP_TG IS NULL THEN
        IF LoaiKH = 2 THEN SET SoTien = NEW.TongTien * 9 / 10;
        ELSEIF LoaiKH = 3 THEN SET SoTien = NEW.TongTien * 17 / 20;
        ELSEIF LoaiKH = 4 THEN SET SoTien = NEW.TongTien * 4 / 5;
        ELSE SET SoTien = NEW.TongTien;
        END IF;
    ELSE
        SET SoTien = 0;
        UPDATE HOADONGOIDICHVU
        SET SoNgaySuDungConLai = SoNgaySuDungConLai - TIMESTAMPDIFF(DAY,
NEW.NgayNhanPhong, NEW.NgayTraPhong)
        WHERE HOADONGOIDICHVU.HDGDV_TG = NEW.DDP_TG AND
HOADONGOIDICHVU.HDGDV_MKH = NEW.DDP_MKH;
    END IF;
    SET NEW.TongTien = SoTien;
END\\
DELIMITER ;
```

c) Create a trigger to calculate the new Diem of KhachHang in table 13

```
-- @block TRIGGER 1c1 --
DELIMITER \\
DROP TRIGGER IF EXISTS update_Diem_GDV\\
CREATE TRIGGER update_Diem_GDV
AFTER INSERT ON HOADONGOIDICHVU FOR EACH ROW
BEGIN
    DECLARE DiemThem INT;
    SET DiemThem = floor(NEW.TongTien/1000);
    UPDATE KHACHHANG
    SET Diem = Diem + DiemThem
    WHERE MaKhachHang = NEW.HDGDV_MKH;
END\\
DELIMITER ;
```

```
-- @block TRIGGER 1c2 --
DELIMITER \\
DROP TRIGGER IF EXISTS update_Diem_DDP\\
CREATE TRIGGER update_Diem_DDP
AFTER INSERT ON DONDATPHONG FOR EACH ROW
BEGIN
    DECLARE DiemThem INT;
    IF NEW.TinhTrang = 1 THEN
        SET DiemThem = floor(NEW.TongTien/1000);
        UPDATE KHACHHANG
        SET Diem = Diem + DiemThem
        WHERE MaKhachHang = NEW.DDP_MKH;
    END IF;
END\\
DELIMITER ;
```

```
-- @block TRIGGER 1c3 --
DELIMITER \\
DROP TRIGGER IF EXISTS update_Diem_DDP_thanhtoan\\
CREATE TRIGGER update_Diem_DDP_thanhtoan
AFTER UPDATE ON DONDATPHONG FOR EACH ROW
BEGIN
    DECLARE DiemThem INT;
    IF NEW.TinhTrang <> OLD. TinhTrang AND NEW.TinhTrang = 1 THEN
        SET DiemThem = floor(NEW.TongTien/1000);
        UPDATE KHACHHANG
        SET Diem = Diem + DiemThem
        WHERE MaKhachHang = NEW.DDP_MKH;
    END IF;
END\\
DELIMITER ;
```

d) Create a trigger to update the new Loai of KhachHang in table 13

```
-- @block TRIGGER 1d --
DELIMITER \\
DROP TRIGGER IF EXISTS update_LoaiKhachHang\\
CREATE TRIGGER update_LoaiKhachHang
BEFORE UPDATE ON KHACHHANG FOR EACH ROW
BEGIN
    IF NEW.Diem < 50 THEN SET NEW.Loai = 1;
    ELSEIF NEW.Diem < 100 THEN SET NEW.Loai = 2;
    ELSEIF NEW.Diem < 1000 THEN SET NEW.Loai = 3;
    ELSE SET NEW.Loai = 4;
    END IF;
END\\
DELIMITER ;
```

e) Create a trigger constraint to prohibit any case of overlapping GoiDichVu

```
-- @block TRIGGER 2 --
DELIMITER \\
DROP TRIGGER IF EXISTS constraint_OverlappingPackage\\
CREATE TRIGGER constraint_OverlappingPackage
BEFORE INSERT ON HOADONGOIDICHVU FOR EACH ROW PRECEDES
update_TongTienGoiDichVu
BEGIN
    DECLARE msg VARCHAR(255);
    SET msg = "OVERLAPPING RENTAL PACKAGE PURCHASE!";
    IF (EXISTS(SELECT * FROM HOADONGOIDICHVU WHERE HDGDV_TG = NEW.HDGDV_TG))
THEN
        IF (NEW.NgayBatDau BETWEEN (SELECT NgayBatDau FROM HOADONGOIDICHVU
WHERE HDGDV_TG = NEW.HDGDV_TG AND HDGDV_MKH = NEW.HDGDV_MKH)
            AND (SELECT ADDDATE(NgayBatDau, INTERVAL 1 YEAR) FROM
HOADONGOIDICHVU WHERE HDGDV_TG = NEW.HDGDV_TG AND HDGDV_MKH = NEW.HDGDV_MKH))
        THEN SIGNAL sqlstate '03000' SET message_text = msg;
        END IF;
    END IF;
END\\
DELIMITER ;
```

4.5. Delete Tables and Schema

We drop any pre-created tables before creating new ones to avoid conflict.

```
DROP TABLE IF EXISTS chinhanh;
DROP TABLE IF EXISTS hinhanh_chinhanh;
DROP TABLE IF EXISTS khu;
DROP TABLE IF EXISTS loaiphong;
DROP TABLE IF EXISTS thongtingiuong;
DROP TABLE IF EXISTS chinhanh_co_loaiphong;
DROP TABLE IF EXISTS phong;
DROP TABLE IF EXISTS loaivattu;
DROP TABLE IF EXISTS loaivattu_trong_loaiphong;
DROP TABLE IF EXISTS vattu;
DROP TABLE IF EXISTS nhacungcap;
DROP TABLE IF EXISTS cungcapvattu;
DROP TABLE IF EXISTS khachhang;
DROP TABLE IF EXISTS goidichvu;
DROP TABLE IF EXISTS hoadongoidichvu;
DROP TABLE IF EXISTS dondatphong;
DROP TABLE IF EXISTS phongthue;
DROP TABLE IF EXISTS hoadon;
DROP TABLE IF EXISTS doanhnghiep;
DROP TABLE IF EXISTS dichvu;
DROP TABLE IF EXISTS dichvuspa;
DROP TABLE IF EXISTS loaihangdoluuniem;
DROP TABLE IF EXISTS thuonghieudoluuniem;
DROP TABLE IF EXISTS matbang;
DROP TABLE IF EXISTS hinhanhcuahang;
DROP TABLE IF EXISTS khunggiohoatdong;
```

5. APPLICATION

Instead of developing an application, we chose to create a local website for internal staff to manage the hotel business. Each manager can register using the tool we provided and login to the server.

5.1. Overview

We use ***USBWebserver*** to deploy our website. ***USBWebserver*** is a combination of popular webserver software: ***Apache***, ***MySQL***, ***PHP*** and ***phpMyAdmin***. With USBWebserver it is possible to develop and show your PHP websites everywhere and anytime. The main advantage of USBWebserver is that you can use it from USB or a regular CD. Combined with Visual Studio Code to implement the source code of .php and .css file.

Figure 1. USBWebserver Interface.

After installing USBWebserver, we can access to phpMyAdmin whch is written in PHP, and is a MySQL administration tools, especially for web hosting services. Then we can upload our database to phpMyAdmin.

After successfully upload our database to phpMyAdmin, we create accounts for ***sManager***. Then we implement the code in two .php files and one .css file to create a basic interface and link the user data to our database in phpMyAdmin.

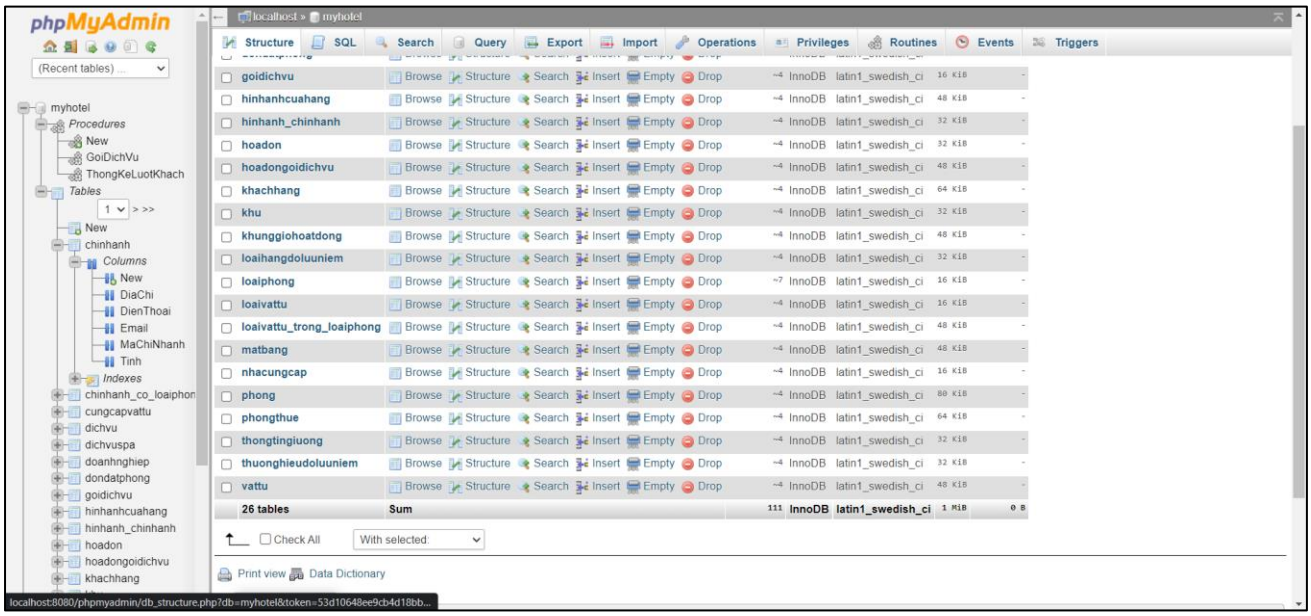

Figure 2. phpMyAdmin page after upload database.

5.2. Product

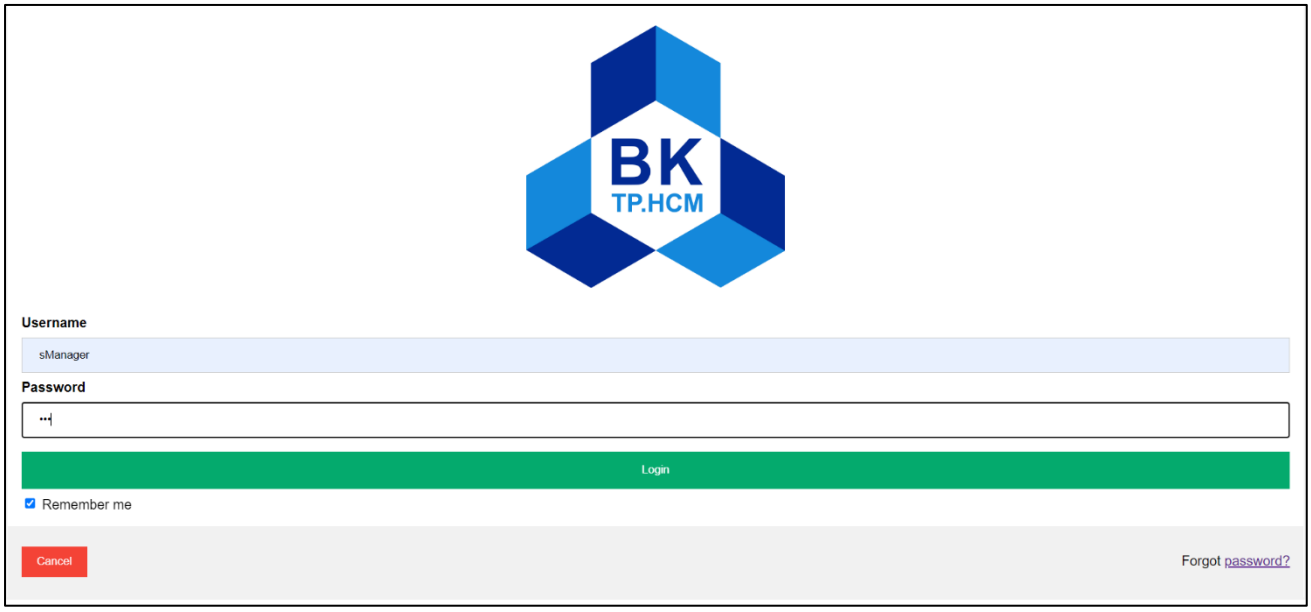

Figure 3. Login page.

Login with any username and password is acceptable at this phase due to the fact that we did not implement an account login information check algorithm yet.

Figure 4. Home page.

Figure 5. When clicked on the Tra cứu khách hàng button.

Figure 6. Result of finding customer with given name.

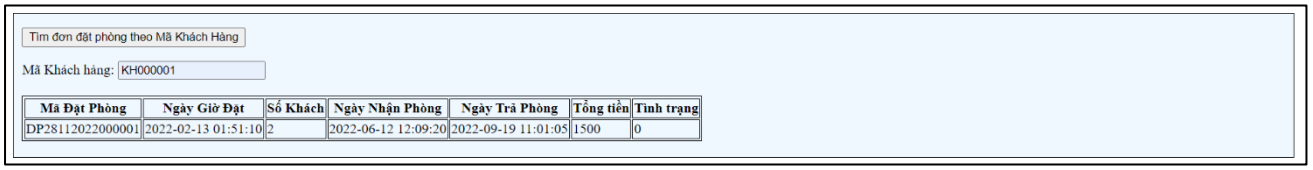

Figure 7. Result of finding DonDatPhong of customer with given customer ID.

Figure 8. Insert a new LoaiPhong.

Figure 9. Show to total amount of customers of a branch in the given business year.

6. CONCLUSION

In this Assignment, we improved the database model from the previous assignment in many aspects. We remodel the schema and use normalization to complete it. The implementation is now having some improvements with Example Query Command, Trigger and Stored Procedure which could bring more convenient to the users and become more applicable.

In the implementation phase, we learn the way developers connect the database to a server to let people access and use it in a meaningful way. Thanks to that, we can gain knowledge about PHP which is a popular legacy language for web development.

However, there are still many drawbacks that can be improved in our implementation. For the database manipulation, we could create more stored procedures, functions and more triggers to make the database easier to use. And for the website, we need to improve a lot. The website has not had the function for the user to create new account, and there could be more information to show to the users. Besides, the user interface is still very basic and is not attractive to the user. It also cannot be used by other roles but only the hotel manager.