**ASSIGNMENT**

Student name: Trần Xuân Hoàn | HE180030

Student name: nguyễn đình duy |HE180130

Student name: lê tiến đạt |HE181318

Teacher: Hoàng Phan

DBI202 – DATABASE SYSTEM OF DORM FPT UNIVERSITY

March 20, 2024

# Mục lục

## Phát biểu bài toán

### Mô tả bài toán……………………………………………………………………………………………………….………….

### mô tả nghiệp vụ hệ thống…………………………………………………………………………….………………..

## Thực thể, liên kết và ER

### định nghĩa các thực thể và thuộc tính………………………………………………………………………..

### thiết lập mối quan hệ và các thực thể…………………………………………………………………………

## đặc tả dữ liệu

### cơ sở dữ liệu và các bảng…………………………..……………………………………………….………………….

## sơ đồ thực thể liên kết(ERD)

### users……..…………………………………………………………………………………………………………………………….

### grocery……………………………………………………………………………………………………………………………...

### eatery……..……………………………………………………………………………………………………………………………

### motel…………………………………………………………………………………………………………………………………..

### entertainment..…………………………………………………………………………………………………………………

### normal form…………………………………………………………………………………………………………………….

## Các câu truy vấn và lệnh SQL tương ứng

### ORDER BY………………………………………………………………………………………………………………………………

### INNER JOIN……………………………………………………………………………………………………………………………

### AGGREGATE FUNCTIONS……………………………………………………………………………………………………….

### GROUP BY và HAVING CLAUSES……………………………………………….……………………………………………

### SUB-QUERY AS A RELATION………………………………………………………………….……………………………….

### QUERY THAT USES PARTIAL MATCHING IN THE WHERE CLAUSE…………………………………………….

### QUERY THAT USES A SELF-JOIN……………………………………………………………………………………………..

### STORE PROCEDURE……………………………………………………………………………………………………………….

### trigger……………………………………………………………………………………………………………………………….

## KẾT LUẬN

# phát biểu bài toán

## mô tả bài toán

Trước đây, tại Hòa Lạc, khi chúng ta muốn tìm trọ, muốn tìm địa điểm vui chơi giải trí, muốn tìm hàng quán để ăn cơm,.. chúng ta phải đi trên 1 quãng đường rất dài, đi rất mất thời gian thì mới tìm được địa điểm ưng ý. Vì thế, việc ra đời một trang facebook hỗ trợ việc tìm kiếm một cách nhanh chóng hiệu quả, đồng thời cung cấp 1 số dịch vụ cần thiết là một điều tất yếu, từ đó mà trang FU-Hòa Lạc ra đời. Hệ thống cơ sở dữ liệu mà chúng em đang thiết kế là một bản mô phỏng lại trang facebook đó.

Sau một thời gian nghiên cứu tìm hiểu, chúng em có một số kết quả sau đây:

* Người dùng là tất cả cư dân tại Hòa Lạc, bao gồm sinh viên và người dân sinh sống tại đó
* Người dùng khi sử dụng trang FU-Hòa Lạc thường cung cấp số nhà, tên trọ, tên xã (nếu có), số điện thoại khi đặt giao đồ, ngoài ra bao gồm các thông tin về giới tính, căn cước công dân và ngày tháng năm sinh tiện cho công tác quản lý
* Các hàng quán, cơ sở vui chơi giải trí thường bổ sung thông tin về địa chỉ, tên quán, giá cả để người dùng dễ nắm bắt thông tin và lựa chọn
* Thi thoảng sẽ có các cửa hàng thuê tuyển nhân viên là người dùng
* Mỗi trọ sẽ cung cấp thông tin về tên trọ, địa chỉ, số điện thoại chủ sở hữu trọ, loại phòng họ có là chỉ đơn hoặc chỉ có phòng có gác xép, giá thuê trọ
* Các cửa hàng dịch vụ tạp hóa, quán ăn sẽ có danh sách sản phẩm, gọi chung là menu, kèm với giá cả của các sản phẩm
* Người dùng có các hành động đến các cửa hàng, quán ăn để mua đồ, đến các địa điểm dịch vụ giải trí để đặt chỗ, ví dụ như đặt bàn bi a chẳng hạn, đặt bàn lẩu hoặc nướng. Ngoài ra người dùng còn có thể đặt thuê phòng trọ sau khi đã xem phòng trực tiếp và tham khảo các thông tin cả trực tiếp (qua chủ trọ) và gián tiếp (qua thông tin trên giao diện sử dụng)
* Dịch vụ tạp hóa đa dạng, có thể là cửa hàng tạp hóa, cũng có thể là sinh viên pass đồ, thông tin bao gồm tên, địa chỉ, giá sản phẩm mà họ bán hoặc pass lại

## mô tả Nghiệp vụ hệ thống

**Request:**

* Hằng ngày sẽ có người dùng đặt món, đặt chỗ tại quán ăn, quán giải trí, đặt mua sản phẩm của bên cung cấp dịch vụ tạp hóa
* Thi thoảng có người dùng ứng tuyển làm việc tại cửa hàng tạp hóa, quán ăn hoặc cửa hàng giải trí
* Thi thoảng có người dùng thuê phòng trọ

**Mục tiêu:**

* Giúp người dùng tìm thông tin nhanh chóng
* Quản lý các bên cung cấp dịch vụ trong khu vực

**Yêu cầu đầu ra:**

* Liệt kê các trọ, các cửa hàng có trong khu vực chỉ định
* Xuất ra giá món ăn người dùng muốn mua của quán ăn
* Đếm hóa đơn mua hàng

# thực thể, liêk kết và ER

## Định nghĩa các thực thể và thuộc tính

Dựa trên các thông tin và mục tiêu mà chương trình đề cập, chúng em có được các bảng và thuộc tính của chúng được thể hiện trong lược đồ sau đây:

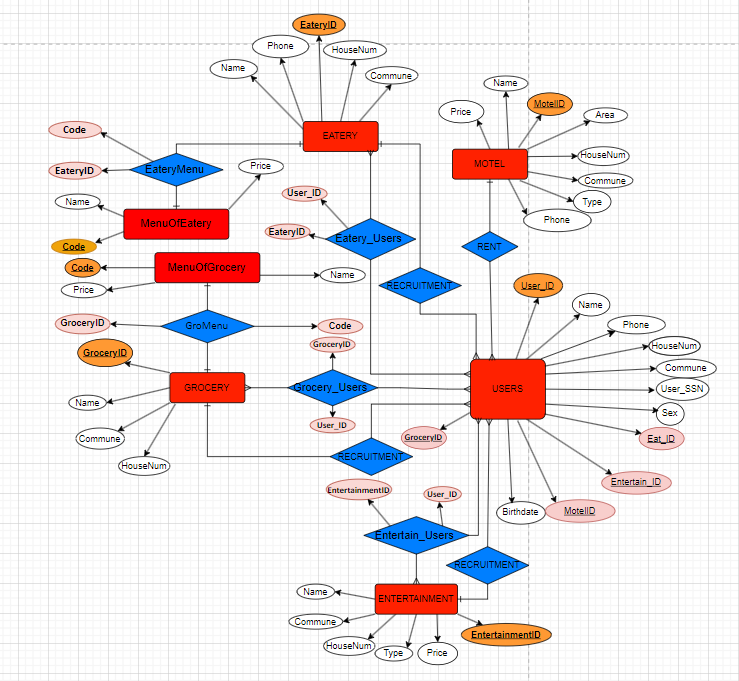
* Users (**User\_ID**, Name, User\_SSN, Phone, Sex, HouseNum, Commune, Birthdate, **GroceryID, EateryID, EntertainmentID, MotelID**)
* Motel (**MotelID**, Name, Phone, HouseNum, Commune, Type, Area, Price)
* Eatery (**EateryID**, Name, Phone, HouseNum, Commune)
* Grocery (**GroceryID**, Name, HouseNum, Commune)
* Entertainment (**EntertainmentID**, Name, HouseNum, Commune, Type, Price)
* MenuOfEatery (**Code**, Name, Price)
* MenuOfGrocery (**Code**, Name, Price)
* EateryMenu (**EateryID, Code**)
* GroMenu (**GroceryID, Code**)
* Eatery\_Users (**User\_ID, EateryID**)
* Grocery\_Users (**User\_ID, GroceryID**)
* Entertain\_Users (**User\_ID, EntertainmentID**)

## set-up entity – relationship

**Các ký hiệu được sử dụng:**

**Attibute**

|  |  |
| --- | --- |
| * Key / identifier attribute |  |
| * Attribute description / description | **ENTITY**  Attribute |
| * Entity | **WEAK ENTIRY** |
| * Weak entity | *Relationship* |
| * Relationship |  |
| * Connectivity (force) = 1 |  |
| * Connectivity = N |  |



## LINK: https://s.net.vn/7Zyl

# Đặc tả dữ liệu

Just for example on some tables (other table are similar, you have to define all the tables in your database). Note: to run the query you have to define the table 1 first then go to the side tables much

## database and table

### cREATE DATABASE **PROJECTDBI202**

--create database

CREATE DATABASE assignment

A screenshot of a computer

Description automatically generated

### **Create table Eatery**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| EateryID | Varchar(7) |  |  | PK, not null |
| Name | Nvarchar(45) |  |  | Not null |
| HouseNum | Nvarchar(5) |  |  | Not null |
| Commune | Nvarchar(255) |  |  | Not null |
| Phone | Varchar(11) |  | 0 to 9 | Unique |

***Code:***

--create table Eatery

create table Eatery(

EateryID VARCHAR(7) PRIMARY KEY,

[Name] NVARCHAR(45) NOT NULL CHECK ([Name] not LIKE '%[0-9.]%'),

HouseNum NVARCHAR (5) NOT NULL,

Commune NVARCHAR(45) NOT NULL,

Phone VARCHAR(11) UNIQUE CHECK(Phone LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]' OR Phone LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]')

)

***Example:***

| EateryID | Name | Phone | HouseNum | Commune |
| --- | --- | --- | --- | --- |
| 2000001 | Cơm Tún | 0324965781 | 13b | Thạch Hòa |
| 2000002 | Mai Linh | 0926514775 | 124a | Thạch Hòa |
| 2000003 | Nguyễn Tuấn | 0362541897 | 44a | Thạch Hòa |
| 2000004 | Nguyễn Việt | 0922651249 | 2a | Tân Xã |

### **Create table motel**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| MotelID | Varchar(7) |  |  | PK, Not null |
| Name | Nvarchar(45) |  |  | Not null |
| HouseNum | Nvarchar(20) |  |  | Not null |
| Commune | Nvarchar(45) |  |  | Not null |
| type | Nvarchar(20) |  |  | Not null |
| area | Nvarchar(4) |  |  | Not null |
| price | Char(20) |  | %VND | Not null |
| Phone | Char(11) |  | 0 to 9 | Not null |
| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| MotelID | Varchar(7) |  |  | PK, Not null |

***Code:***

create table Motel(

MotelID VARCHAR(7) PRIMARY KEY,

[Name] NVARCHAR(45) NOT NULL CHECK ([Name] not LIKE '%[0-9.]%'),

HouseNum NVARCHAR (20) NOT NULL,

Commune NVARCHAR(45) NOT NULL,

[Type] NVARCHAR(20) NOT NULL,

[Area] NVARCHAR(4) not null,

Price VARCHAR(20) NOT NULL CHECK(Price LIKE '%.000VND') ,

Phone VARCHAR(11) UNIQUE CHECK(Phone LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]' OR Phone LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]')

)

***Example:***

| MotelID | Name | House num | Commune | Type | Area | Price | PricePhone | ID | Name |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1000001 | Tuấn Yến | 255 | Thạch Hòa | Đơn | 20m^2 | 2.500.000 | 0934567890 | 1000001 | Tuấn Yến |
| 1000002 | Ông Bà | 1a | Hạ Bằng | có gác xép | 18m^2 | 1.800.000 | 0945678901 | 1000002 | Ông Bà |
| 1000003 | Happy | 20c | Thạch Hòa | có gác xép | 20m^2 | 1.800.000 | 0956789012 | 1000003 | Happy |
| 1000004 | 2 Anh Cò | 232b | Tân Xã | có gác xép | 25m^2 | 2.000.000VND | 0967890123 | 1000004 | 2 Anh Cò |

### **Create table Entertainment**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| EntertainmentID | Varchar(7) |  |  | PK, Not null |
| Name | Nvarchar(45) |  | Not like  '%[0-9.]%' | Not null |
| HouseNum | Nvarchar(20) |  |  | Not null |
| Commune | Nvarchar(45) |  |  | Not null |
| type | Nvarchar(20) |  |  | Not null |
| price | Char(20) |  | %VND | Not null |
| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |

***Code:***

--create table room

create table Entertainment(

EntertainmentID VARCHAR(7) PRIMARY KEY,

[Name] NVARCHAR(255) NOT NULL CHECK ([Name] not LIKE '%[0-9.]%'),

HouseNum NVARCHAR (10) NOT NULL,

Commune NVARCHAR(45) NOT NULL,

Price VARCHAR(20) CHECK(Price LIKE '%VND') NOT NULL,

[Type] NVARCHAR(20) NOT NULL

)

***Example:***

### A table with numbers and words Description automatically generated

### Create TABLE **Grocery**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| GroceryID | Varchar(7) |  |  | PK, Not null |
| Name | Nvarchar(45) |  | Not like  '%[0-9.]%' | Not null |
| HouseNum | Nvarchar(20) |  |  | Not null |
| Commune | Nvarchar(45) |  |  | Not null |

***Code:***

--create table Grocery

create table Grocery(

GroceryID VARCHAR(7) PRIMARY KEY,

[Name] NVARCHAR(255) NOT NULL CHECK ([Name] not LIKE '%[0-9].%'),

HouseNum NVARCHAR (10) NOT NULL,

Commune NVARCHAR(45) NOT NULL

)

***Example:***

A white rectangular object with black text

Description automatically generated

### create table **USERS**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| User\_ID | Varchar(7) |  |  | PK, Not null |
| MotelID | Varchar(7) |  |  | FK reference Motel(MotelID) |
| GroceryID | Varchar(7) |  |  | FK reference Grocery(GroceryID) |
| EateryID | Varchar(7) |  |  | FK reference Eatery(EateryID) |
| EntertainmentID | Varchar(7) |  |  | FK reference Entertainment(EntertainmentID) |
| Name | Nvarchar(225) |  | Not like  '%[0-9.]%' | Not null |
| Gender | Char(1) | M | ‘F’ or ‘M’ | Not null |
| User\_SSN | Varchar(20) |  |  | Unique,Not null |
| Phone | Nvarchar(11) |  | 0 to 9 | Not null |
| HouseNum | Nvarchar(20) |  |  | Not null |
| Commune | Nvarchar(45) |  |  | Not null |
| Birthdate | Nvarchar(45) | Datetime |  | Not null |

***Code:***

create table Users(

[User\_ID] VARCHAR(7) PRIMARY KEY,

GroceryID VARCHAR(7) foreign key

references Grocery(GroceryID) ,

EateryID VARCHAR(7) foreign key

references Eatery(EateryID) ,

EntertainmentID VARCHAR(7) foreign key

references Entertainment(EntertainmentID),

MotelID VARCHAR(7) foreign key

references Motel(MotelID),

[Name] NVARCHAR(255) NOT NULL CHECK ([Name] not LIKE '%[0-9.]%'),

User\_SSN varchar(20) UNIQUE CHECK(User\_SSN LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),

Phone VARCHAR(11) UNIQUE CHECK(Phone LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]' OR Phone LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),

Sex CHAR(1) DEFAULT 'M' CHECK(Sex IN('F','M')) NOT NULL,

HouseNum NVARCHAR (10),

Commune NVARCHAR(45) NOT NULL,

[Birthdate] date check (Birthdate <= getdate()) not null

)

***Example:***

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Name | User\_SSN | Phone | Sex | HouseNum | Commune | Birthdate | Gro\_ID | Eatery\_ID | IDEntertainmentID | MotelID |
| 0000001 | Nguyễn Thị Hồng Hoa | 0362002001 | 0912326327 | F | FPT | Thạch Hòa | 2001-11-02 | 3000001 | 2000007 | 4000003 | 10000010 |
| 0000002 | Trần Văn Thành | 0361232121 | 0912797289 | M | 255 | Thạch Hòa | 2003-12-31 | 3000003 | 2000006 | 4000001 | 1000001 |
| 0000003 | Nguyễn Phi Long | 036430730009 | 0912556778 | M | FPT | Thạch Hòa | 2003-01-09 | 3000004 | 2000003 | 4000002 | 10000010 |
| 0000004 | Trần Hồng Ngân | 036217044098 | 0912555666 | F | 232b | Tân Xã | 2004-07-21 | 3000005 | 2000002 | 4000006 | 1000004 |
| 0000005 | Vũ Thị Tuyết Nhi | 036220714071 | 0322145678 | F | 14a | Tân Xã | 2005-4-12 | 3000002 | 2000001 | 4000003 | 1000006 |
| 0000006 | Trần Huy Hoàng | 003688910003 | 0948123456 | M | 1a | Hạ Bằng | 2002-6-12 | 3000006 | 2000002 | 4000005 | 1000002 |
| 0000007 | Hán Quang Minh | 036088990934 | 0912444567 | M | 134 | Hạ Bằng | 2002-7-12 | 3000007 | 2000004 | 4000004 | 1000002 |
| 0000008 | Trần Huy Vũ | 037123002474 | 0948123456 | M | 1a | Hạ Bằng | 2002-6-12 | 3000004 | 2000001 | 4000007 | 1000002 |
| 0000009 | Ông Cao Thắng | 036889100300 | 0912345999 | M | 12 | Kim Sơn | 2004-6-10 | 3000006 | 2000005 | 4000002 | 1000007 |
| 0000010 | Lã Quỳnh Anh | 036088901273 | 0949278203 | F | 334 | Thạch Hòa | 2005-10-12 | 3000003 | 2000002 | 4000006 | 1000003 |
| 0000011 | Nguyễn Quế Hoa | 03608890198 | 0844270208 | F | 14 | Thạch Hòa | 2003-4-27 | 3000010 | 2000007 | 4000009 | 1000008 |

### create table **Menu of eatery**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| Code | varchar(7) |  |  | Primary key |
| name | nVarchar(255) |  | Not like  '%[0-9.]%' | Not null |
| price | Varchar(20) |  | %VND | Not null |
| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |

***Code:***

--create table room

create table MenuOfEatery(

Code varchar(7) primary key,

[Name] NVARCHAR(255) NOT NULL CHECK ([Name] not LIKE '%[0-9.]%'),

Price VARCHAR(20) NOT NULL CHECK(Price LIKE '%VND')

)

***Example:***

A table with numbers and letters

Description automatically generated

### create table Menu of grocery

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| Code | varchar(7) |  |  | Primary key |
| name | nVarchar(255) |  | Not like  '%[0-9.]%' | Not null |
| price | Varchar(20) |  | %VND | Not null |
| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| Code | varchar(7) |  |  | Primary key |
| name | nVarchar(255) |  | Not like  '%[0-9.]%' | Not null |

***Code:***

--create table room

create table MenuOfGrocery(

Code varchar(7) primary key,

[Name] NVARCHAR(255) NOT NULL CHECK ([Name] not LIKE '%[0-9.]%'),

Price VARCHAR(20) CHECK(Price LIKE '%VND') NOT NULL

)

***Example:***

***A table with numbers and text

Description automatically generated***

### Create table **eATERYMENU**

| 1. Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| EateryID | Varchar(7) |  |  | FK reference Eatery(EateryID) |
| Code | Varchar(7) |  |  | FK reference MenuOfEatery (Code) |

***Code:***

create table EateryMenu(

EateryID VARCHAR(7) foreign key

references Eatery(EateryID),

Code VARCHAR(7) foreign key

references MenuOfEatery(Code)

)

***Example:***

| EateryID | Code |
| --- | --- |
| 2000001 | 0000101 |
| 2000002 | 0000102 |
| 2000003 | 0000103 |
| 2000004 | 0000104 |

### create table **GRomenu**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| GroceryID | Varchar(7) |  |  | FK reference Grocery (**GroceryID**) |
| Code | Varchar(7) |  |  | FK reference MenuOfGrocery(Code) |

***Code:***

create table GroMenu(

GroceryID VARCHAR(7) foreign key

references Grocery(GroceryID),

Code VARCHAR(7) foreign key

references MenuOfGrocery(Code)

)

***Example:***

| GroceryID | Code |
| --- | --- |
| 3000001 | 0000200 |
| 3000001 | 0000201 |
| 3000001 | 0000202 |
| 3000002 | 0000204 |

### Create TABLE **grocery\_USers**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| [User\_ID] | Varchar(7) |  |  | FK references Users([User\_ID]) |
| GroceryID | Varchar(7) |  |  | FK references Grocery(GroceryID) |

***Code:***

create table Grocery\_Users(

[User\_ID] VARCHAR(7) foreign key

references Users([User\_ID]),

GroceryID VARCHAR(7) foreign key

references Grocery(GroceryID)

)

***Example:***

| [User\_ID] | GroceryID |
| --- | --- |
| 0000001 | 3000001 |
| 0000002 | 3000001 |
| 0000003 | 3000002 |
| 0000004 | 3000003 |

### create table Eatery\_Users

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| [User\_ID] | Varchar(7) |  |  | FK references Users([User\_ID]) |
| EateryID | Varchar(7) |  |  | FK references Eatery(EateryID) |

***Code:***

create table Eatery\_Users(

[User\_ID] VARCHAR(7) foreign key

references Users([User\_ID]),

EateryID VARCHAR(7) foreign key

references Eatery(EateryID)

)

***Example:***

| [User\_ID] | EateryID |
| --- | --- |
| 0000001 | 4000001 |
| 0000002 | 4000001 |
| 0000003 | 4000001 |
| F1086 | 0000004 |

### create table **Entertain\_Users**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| [User\_ID] | Varchar(7) |  |  | FK references Users([User\_ID]) |
| EntertainmentID | Varchar(7) |  |  | FK references Entertainment(EntertainmentID)  ) |

***Code:***

create table Entertain\_Users(

[User\_ID] VARCHAR(7) foreign key

references Users([User\_ID]),

EntertainmentID VARCHAR(7) foreign key

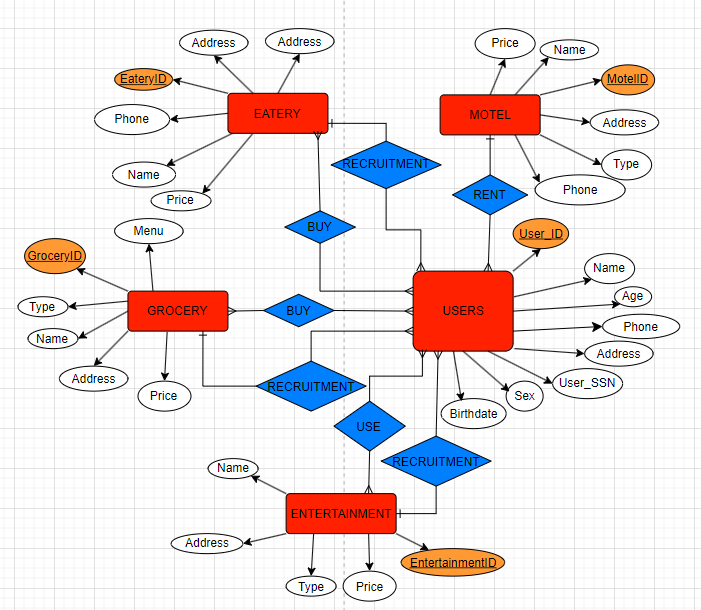
references Entertainment(EntertainmentID)

)

***Example:***

| [User\_ID] | EntertainmentID |
| --- | --- |
| 0000001 | 4000001 |
| 0000002 | 4000001 |
| 0000004 | 4000007 |
| 0000005 | 4000003 |

# IV. Sơ đồ thực thể Liên kết (ERD)



## EATERY

A diagram of a company

Description automatically generated

|  |  |
| --- | --- |
| Đây là thực thể eatery, thực thể eatery có 5 thuộc tính tính. Thực thể có EateryID làm khóa chính. Mỗi một Eatery(hàng ăn) có các thông tin như HouseNum(Số nhà), Commune (Thôn), Phone(sđt) để liên hệ, Name(tên) |  |

## Grocery

A diagram of a grocery store

Description automatically generated

|  |  |
| --- | --- |
| Đây là thực thể Grocery, thực thể Grocery có 5 thuộc tính tính. Thực thể có GroceryID làm khóa chính. Mỗi một Grocery(hàng tạp hóa) có các thông tin như HouseNum(Số nhà), Commune (Thôn), Name(tên) |  |
|  |  |

## EnTERTAINMENT

A diagram of a company

Description automatically generated

|  |  |
| --- | --- |
| đây là thực thể Entertainment, thực thể Entertainment có 5 thuộc tính tính. Tương tự như thực thể Grocery, Thực thể có EntertainmentID làm khóa chính. Mỗi một Entertainment (Khu giải trí) có các thông tin như HouseNum(số nhà) và Commune(tên xã), Type để phân loại các hình thức, Name(tên), cuối cùng là Price(giá) để ước lượng giá tiền |  |
|  |  |

## MOTEL

A diagram of a motel

Description automatically generated

|  |  |
| --- | --- |
| Kế đến là thực thể Motel(trọ), thực thể Motel có 8 thuộc tính . Thực thể có MotelID làm khóa chính. Mỗi một Motel (trọ) có các thông tin như HouseNum(số nhà) và Commune(tên xã), Type để phân loại các hình thức, Name(tên), Price(giá) để ước lượng giá tiền |  |
| Users |  |
|  |  |

A diagram of a user flow

Description automatically generated

Thực thể quan trọng nhất, users(người dùng) gồm User\_ID làm khóa chính. GroceryID, EateryID, EntertainmentID, MotelID lần lượt là khóa ngoại của các thực thể đã nêu trên. Sau đó là Name(tên)User\_SSN(số cccd), Phone(sđt), Sex(giới tính), HouseNum(số nhà) và Commune(tên xã), cuối cùng là Birthdate(ngày sinh)

|  |  |
| --- | --- |
|  |  |
| menu of grocery and menu of eatery  |  |  | | --- | --- | |  |  | |  |  |   Hai Thực thể MenuOfGrocery và MenuOfEatery đều như nhau ở chỗ là có 2 thuộc tính nhưng khác ở chỗ MenuOfGrocery có các thuộc tính Code(mã của vật phẩm) và Price(giá trị của đồ vật), còn MenuOfEatery Có Code(mã của đồ ăn) và Name(tên của thực phẩm).  Normal form Các bảng trên chúng em đã chuyển đổi hoàn toàn về dạng chuẩn 3NF. full diagram |  |
|  |  |

A computer screen shot of a computer

Description automatically generated

# V. Lệnh truy vấn SQL

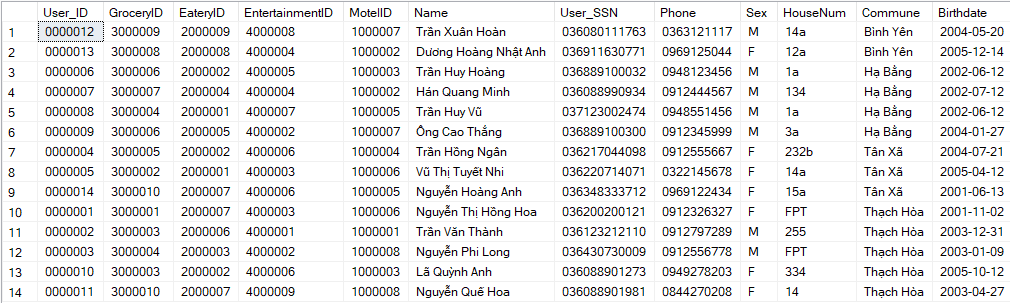
Trong chương trình lần này, chúng em sử dụng SQL Server 2019, là một hệ thống thông minh có những chức năng tiện lợi, hỗ trợ nhiều tác vụ để xây dựng và quản lý cơ sở dữ liệu một cách hiệu quả.

## query using order by

***Code:***

select \*from dbo.Users

order by Commune



***Result:***

Chúng em sử dụng truy vấn chứa ORDER BY để sắp xếp danh sách tăng dần hoặc giảm dần theo giá trị của miền. Lệnh select \*from dbo.Users cung cấp cho chúng ta tất cả bản ghi trong bảng Users và sau đó sắp xếp các bản ghi tăng dần theo Commune.

## query using inner join

***Code:***

select e.EateryID, e.Name, e.HouseNum, e.Commune,

e.Phone, em.Code

from dbo.Eatery e join dbo.EateryMenu em on e.EateryID=em.EateryID

order by e.EateryID

***Result:***



|  |  |
| --- | --- |
| **Trong truy vấn này chúng em sử dụng lệnh INNER JOIN để tìm những Eatery cùng với EateryMenu tương ứng dựa trên điều kiện kết nối và cuối cùng sắp xếp kết quả dựa theo EateryID.** |  |

## query using aggregate functions

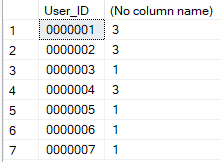
***Code:***

select gu.User\_ID ,count(gu.GroceryID)

from dbo.Grocery\_Users gu

group by gu.User\_ID

***Result:***



Chúng em đếm số lượng Grocery mà mỗi User sử dụng. Chúng em sử dụng function COUNT()

với biến GroceryID để thực hiện việc đếm.

***Code:***

select Top 2 e.\*

from Entertainment e

order by Name

***Result:***



Hiển thị một ví dụ khác về truy vấn sử dụng hàm tổng hợp, đó là lệnh TOP. Chúng em sử dụng lệnh này để tìm ra 2 dịch vụ Entertainment và sắp xếp chúng theo Name.

## query using the group by and having clauses

***Code:***

SELECT Commune, COUNT(\*) AS NumberOfGroceries

FROM Grocery

GROUP BY Commune

HAVING COUNT(\*) > 1;

***Result:***



Chọn cột "Commune" từ bảng "Grocery" và đếm số lượng hàng hóa trong mỗi khu phố. Kết quả của hàm đếm được gán vào một cột mới có tên là "NumberOfGroceries". Nhóm các hàng dữ liệu theo giá trị của cột "Commune". Điều này có nghĩa là các hàng dữ liệu với cùng một giá trị "Commune" sẽ được nhóm lại với nhau. Lọc các nhóm dựa trên số lượng hàng hóa trong mỗi nhóm. Chỉ các nhóm có số lượng hàng hóa lớn hơn 1 mới được chấp nhận. Tóm lại, câu lệnh trên trả về số lượng cửa hàng tạp hóa trong mỗi xã, nhưng chỉ hiển thị những xã có ít nhất hai cửa hàng tạp hóa.

## query that uses a sub-query as a relation

***Code:***

select moe.Code, moe.Name, moe.Price

from dbo.MenuOfEatery moe

where moe.Name not in (

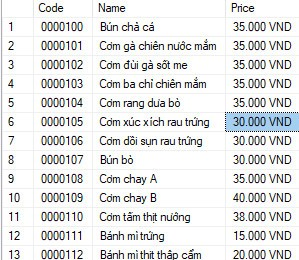
select moe1.Name

from dbo.MenuOfEatery moe1 join dbo.EateryMenu em1 on em1.Code=moe1.Code

where moe.Price=35000

)

***Result:***



## query that uses partial matching in the where clause

***Code:***

select \*

from dbo.Grocery g

where g.GroceryID like '%1%' or g.GroceryID like N'%1'

***Result:***



Câu lệnh trên dùng để hiện thị ra những Grocery có GroceryID có chứa số ‘1’ hoặc có số ‘1’ ở cuối

## query that uses a self – join

***Code:***

select u.User\_ID, u.Name, u.Phone, u.Sex, U.Birthdate from Users u join Motel m

on u.MotelID = m.MotelID

where u.MotelID = '1000006'***Result:***

******

|  |  |
| --- | --- |
| Câu lệnh querry trên tương tự ví dụ sử dung INNER JOIN. |  |

## store procedure

***Code:***

CREATE PROC Information\_Users

@mscccd VARCHAR(12)

AS

BEGIN

SELECT u.User\_ID, u.[Name], u.Phone, u.User\_SSN, e.EateryID, e.Name

FROM dbo.Users u, dbo.Eatery e, dbo.Eatery\_Users ea

WHERE u.User\_SSN = @mscccd

and u.User\_ID = ea.User\_ID

and e.EateryID = ea.EateryID

END

EXEC dbo.Information\_Users @mscccd = '036200200121'***Result:***



Nhập vào User\_SSN, in ra thông tin của user, tên là gì, số điện thoại, mã số, ID của cửa hàng và tên của hàng

## trigger

***Code:***

CREATE TRIGGER trg\_CheckEntertainmentType

ON Entertainment

AFTER INSERT, UPDATE

AS

BEGIN

IF EXISTS (

SELECT 1

FROM inserted

WHERE [Type] NOT IN ('Billiard', 'Net', 'Golf', 'Karaoke')

)

BEGIN

RAISERROR('Invalid entertainment type. Allowed types are: Billiard, Net, Golf, Karaoke', 16, 1)

ROLLBACK TRANSACTION

END

END

# Mục đích của trigger này là ngăn không cho người dùng chèn hoặc cập nhật dữ liệu trong bảng Entertainment với giá trị Type không hợp lệ. Bất cứ khi nào có lệnh INSERT hoặc UPDATE với giá trị Type khác 'Billiard', 'Net', 'Golf' hoặc 'Karaoke', trigger sẽ hiển thị thông báo lỗi và huỷ bỏ giao dịch, đảm bảo tính toàn vẹn dữ liệu và giá trị Type chỉ chấp nhận các giá trị đã cho phép.

# VII. CONCLUSION

- Sau khi thiết kế và tạo xong cơ sở dữ liệu cho trang FU-Hòa Lạc, một số kết luận quan trọng có thể rút ra như sau:

1, Cơ sở dữ liệu đã được thiết kế và triển khai thành công, bao gồm các bảng chính như Users, Eatery, Motel, Entertainment, Grocery, và các bảng liên quan khác để quản lý menu, mối quan hệ giữa người dùng và các cơ sở kinh doanh.

2, Cơ sở dữ liệu này giúp lưu trữ và quản lý thông tin một cách có hệ thống, đảm bảo tính toàn vẹn và nhất quán của dữ liệu.

3, Việc sử dụng cơ sở dữ liệu sẽ hỗ trợ tốt cho hoạt động của trang FU-Hòa Lạc, cho phép người dùng dễ dàng tìm kiếm và đặt hàng/đặt chỗ tại các cơ sở kinh doanh trong khu vực Hòa Lạc.

4. Cơ sở dữ liệu cũng giúp quản lý thông tin của người dùng và các cơ sở kinh doanh một cách hiệu quả, đồng thời có thể mở rộng và bổ sung thêm tính năng trong tương lai.

5, Việc triển khai cơ sở dữ liệu này đáp ứng được nhu cầu ban đầu của trang FU-Hòa Lạc, giúp tăng cường trải nghiệm người dùng và nâng cao hiệu quả hoạt động của trang.

6, Tuy nhiên, cần có sự đánh giá và bảo trì thường xuyên để đảm bảo hoạt động ổn định, hiệu suất cao và an toàn dữ liệu cho cơ sở dữ liệu này.

7, Tóm lại, việc thiết kế và triển khai cơ sở dữ liệu cho trang FU-Hòa Lạc là một bước quan trọng, giúp tối ưu hóa quá trình quản lý thông tin và cung cấp dịch vụ cho người dùng, đồng thời tạo tiền đề vững chắc cho sự phát triển và mở rộng của trang trong tương lai

The End .