**ASSIGNMENT**

Student name: Pham Ngoc Hoa | SE05740

Student name: nguyen hai nam | se05817

Teacher: nguyen quynh chi

DBI202 – DATABASE SYSTEM OF DORM FPT UNIVERSITY

March 20, 2018

# TABLE OF CONTENTS

## INTRODUCE THE PROBLEM…………………………………………………………………………………………….

### Describe the problem……………………………………………………………………………………………………….

### Management objectives…………………………………………………………………………………………………..

## entity – relationship – er…………………………………………………………………………………………

### difinITION entity – attributE…………………………………………………………………………………………..

### set-up entity – relationship…………………………………………………………………………………………….

## data dictionary………………………………………………………………………………………………………….

### DATABASE AND TABLE……………………………………………………………………………………….………………….

## ENTITY RELATIONSHIP DIAGRAM (ERD)…………………………………………………………………………

### guard………………………………………………………………………………………………………………………………….

### student……………………………………………………………………………………………………………………………...

### panalize………………………………………………………………………………………………………………………………

### room…………………………………………………………………………………………………………………………………..

### items……………………………………………………………………………………………………………………………………

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

## SQL COMMAND

### QUERY USING ORDER BY………………………………………………………………………………………………………

### QUERY USING INNER JOIN……………………………………………………………………………………………………

### QUERY USING AGGREGATE FUNCTIONS…………………………………………………………………………………

### QUERY USING THE GROUP BY AND HAVING CLAUSES……………………………………………………………

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

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

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

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

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

# INTRODUCE THE PROBLEM

## Describe the problem

Nowadays, Students in FPT University almost live in five Dom, there are A, B, C, D, F and have some guard to manager it. But, it is still have not database system for manage dorm. After the actual survey, the result are as follows:

* Student live in room of dom, each room have one or more student, however at one point an student is only live in one room.
* The head of each room is an student who is the head of the department. And every student is managed by an guard.
* Guard records information about students, including: student code, first name, last name, email, phone, gender, course, roles. Character of the student are “M”(Boy) of ”F”(Girl). Phone number can not greater than 11 digits.
* Gender of student have to equal gender of room.
* Guard also needs to manager number penalize to calculate fines by day or month. Calculate the amount of money owed by the student due to violation of regulations.
* Each penalty include: name of student is fined, type of penalize and quantity.
* Student can register live in a room when number of bed is empty, that student have to pay for register. Check Price are “1” (have paid) or “0” (not paid).
* Information for register include: room code, student code, date that student was check – in.
* Each guard can manager one or more room and room can managed one or more guard.
* Room code is combination by Name Dome and Floor and number of room and number of bed. Example: a2055, it mean dome a, floor 2, room 05, bed 5.
* Information of room include: room code, location, number bed, gender, status bed.
* Location of the room are “Trái” (Left) or “Phải” (Right).
* Status bed of room are “1” (This bed is empty, student can register live in this bed) or ”0” (This bed is registered, student can not register in this bed).
* Guard also need to manager facilities in FPT University to call fixer to fix the rooms are broken to student continue to live safe.
* Items of room can need fix example: light bulb, table,… ect. Each item inclue: Item code, name of things need fix, content of device and price of that device.
* Fix detail include: room code, item code, name of fixer because have only 2 fixer is “Hùng” and “Hưng” so we can’t create new object fixer, day reported malfunction, day fix malfunction, quantity the same thing is broken.
* Guard manager every room in common dom, so manager room has a attribute is name room and every room in dom have to common guard.
* Information of guard include: guard code, first name of guard, last name of guard, email of guard, phone of guard.
* Information of manager room include: name of dom, Room ID , Guard Id, price have to pay for each semester.

**Request:**

* Daily, guard need to caculate the total sutdent fined, caculate the total money.
* Guard need check day student check-in, check out, day report item broken.
* Monthly, guard need to count and display student not pay money for room.
* Monthly, guard need cacule total amount collected.
* Guard can check information of student.
* Guard can check room is emty for student register.
* Guard can check item broken in dom, in room to call fixer.
* Check student register in room valid or invalid
* Check room valid in condition special when student register in room.

## Management objectives

* Manage student and change of students.
* Manage penalize and calculate money fined.
* Manage room for student register.
* Manage items broken to fix.

**Important output**

* Total money for fined of each student.
* Count number room is empty.
* Count number room have items broken.

# entity – relationship – er

## difinITION entity – attributE

Base on the problem description and management objectives, we can present several entities and attributes of the entity as follow:

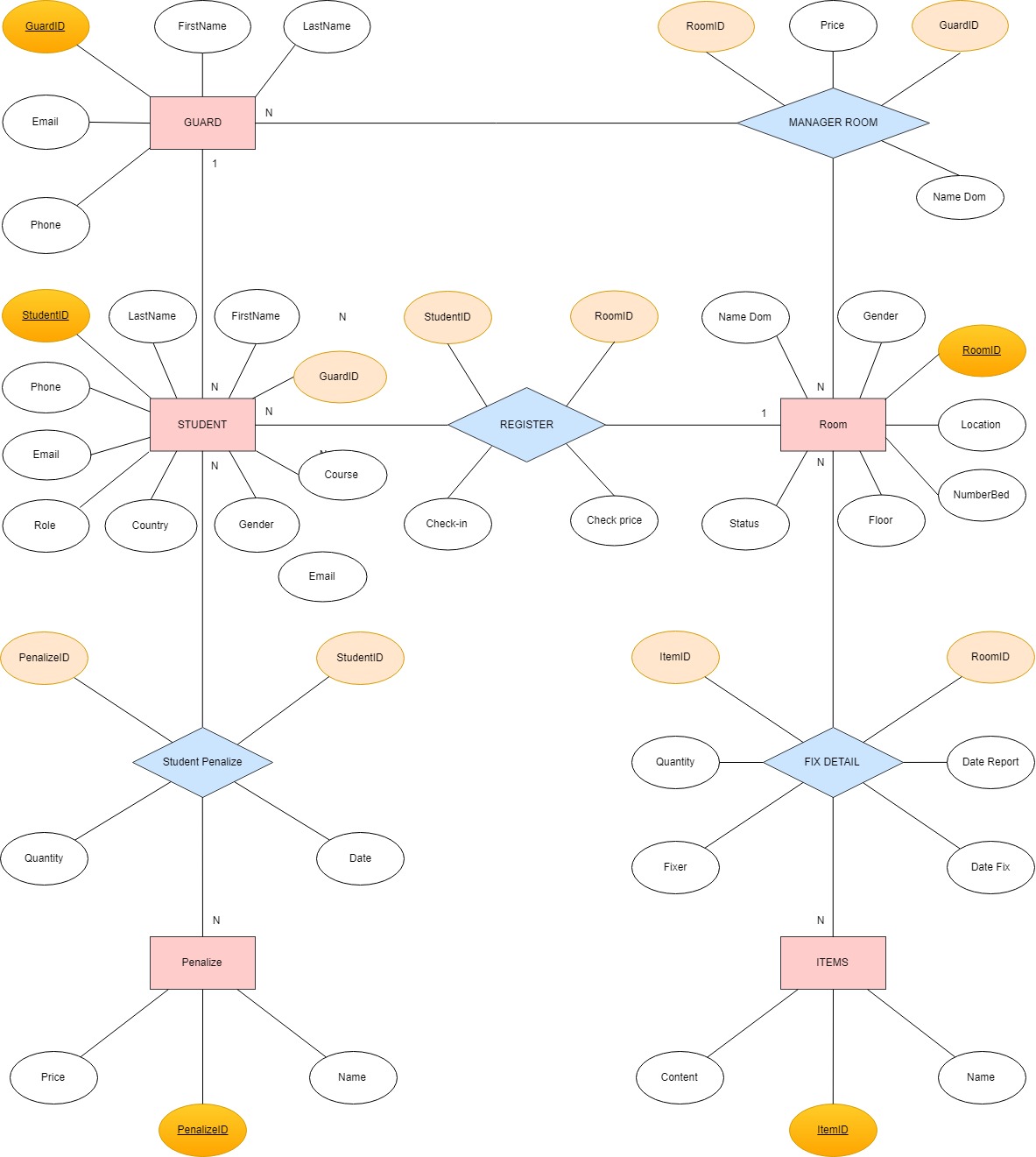
* Student: **StudentID, GuardID,** First Name, Last Name, Email, Phone, Gender, Course, Role, Country.
* Register: **RoomID, StudentID,** Check-In, CheckPrice.
* Room: **RoomID** ,Name Dom, Floor, Location, Number Bed, Gender, Status.
* Fix Detail: **RoomID, ItemID,** Fixer, Date Report, Date Dix, Quantity.
* Items: **ItemID**, Name, Content, Price.
* Student Penalize: **StudentID, PenalizeID,** Date, Quantity.
* Penalize: **PenalizeID**, Name, Price.
* Guard: **GuardID*,*** First Name, Last Name, Email, Phone.
* Manager room: **RoomID**, **GuardID,** Name Dom, Price.

## set-up entity – relationship

\* Some symbols used in the model

**Attibute**

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



## LINK: https://goo.gl/U4x3jo

# data dictionary

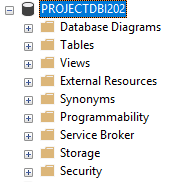
Just for example on some tables (other table are similar, you have to define all the tables in your database). Note: to run the query you have to define the table 1 first then go to the side tables much

## database and table

### cREATE DATABASE **PROJECTDBI202**

--create database

CREATE DATABASE PROJECTDBI202



### **Create table GUARD**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| GuardID | Int |  |  | unique |
| First Name | Nvarchar(45) |  |  | Not null |
| Last name | Nvarchar(45) |  |  | Not null |
| Email | Nvarchar(255) |  | @fpt.edu.vn | Not null |
| Phone | Varchar(11) |  | 0 to 9 | Unique |

***Code:***

--create table guard

CREATE TABLE GUARD(

GuardID INT PRIMARY KEY,

[First Name] NVARCHAR(45) NOT NULL,

[Last Name] NVARCHAR(45) NOT NULL,

Email NVARCHAR(255) CHECK(Email LIKE '%@fpt.edu.vn')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:***

| GuardID | First Name | Last Name | Email | Phone |
| --- | --- | --- | --- | --- |
| 1 | Minh | Nguyễn Quang | [minhnq@fpt.edu.vn](mailto:minhnq@fpt.edu.vn) | 01652035388 |
| 2 | Việt | Nguyễn Khánh | [vietnk@fpt.edu.vn](mailto:vietnk@fpt.edu.vn) | 09496352142 |
| 3 | Hạ | Võ Nhật | [havn@fpt.edu.vn](mailto:havn@fpt.edu.vn) | 0956484512 |
| 4 | Hiếu | Từ Khắc | [hieutk@fpt.edu.vn](mailto:hieutk@fpt.edu.vn) | 0162354896 |

### **Create table Student**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| StudentID | Varchar(7) |  | SE|SB|HExxxxx | PK, Not null |
| GuardID | Int |  |  | FK reference GUARD(GuardID) |
| First Name | Nvarchar(45) |  |  | Not null |
| Last Name | Nvarchar(45) |  |  | Not null |
| Email | Nvarchar(255) |  | @fpt.edu.vn | Not null |
| Phone | Char(11) |  | 0 to 9 | Unique |
| Gender | Char(1) | M | ‘F’ or ‘M’ | Not null |
| Course | Char(3) |  | Kxx | Not null |
| Role | Nvarchar(30) | Thành Viên | ‘Thành Viên’ or ‘Trưởng Phòng’ | Not null |
| Country | Nvarchar(50) | VietNam |  | Not Null |

***Code:***

--create table student

CREATE TABLE STUDENT(

StudentID VARCHAR(7) PRIMARY KEY NOT NULL CHECK(StudentID LIKE 'SE[0-9][0-9][0-9][0-9][0-9]' OR StudentID LIKE 'SB[0-9][0-9][0-9][0-9][0-9]' OR StudentID LIKE 'HE[0-9][0-9][0-9][0-9][0-9]'),

GuardID INT FOREIGN KEY REFERENCES dbo.GUARD(GuardID),

[First Name] NVARCHAR(45) NOT NULL,

[LastName] NVARCHAR(45) NOT NULL,

Email NVARCHAR(255) CHECK(Email LIKE '%@fpt.edu.vn') NOT NULL,

Phone CHAR(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]'),

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

Course CHAR(3) CHECK(Course LIKE 'K[0-9][0-9]') NOT NULL,

[Role] NVARCHAR(30) DEFAULT N'Thành Viên' CHECK(Role IN (N'Thành Viên', N'Trưởng Phòng')) NOT NULL ,

Country NVARCHAR(50) DEFAULT N'Việt Nam' NOT NULL

)

***Example:***

| StudentID | GuardID | First Name | Last Name | Email | Phone | Gender | Course | Role | Contry |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SE06155 | 1 | Tù | Bùi Anh | [tubase06155@gmail.com](mailto:tubase06155@gmail.com) | 01699271212 | M | K12 | Thành Viên | Việt Nam |
| HE13001 | 3 | Công | Lê Thành | [congltthe13001@fpt.edu.vn](mailto:congltthe13001@fpt.edu.vn) | 0969981028 | M | K13 | Thành Viên | Việt Nam |
| SE05627 | 3 | Hiệp | Đỗ Quang | [hiepdq05627/2fpt.edu.vn](mailto:louhoang@gmail.com) | 01649088188 | M | K12 | Trưởng Phòng | Việt Nam |
| SE05740 | 5 | Hòa | Phạm Ngọc | [thaycacac@fpt.edu.vn](mailto:thaycacac@fpt.edu.vn) | 0968038714 | M | K12 | Trưởng Phòng | Campuchia |

### )**Create table ROOM**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| RoomID | Char(5) |  | A|B|C|D|Fxxxxx | Primary key |
| Name Dom | Varchar(6) |  | A|B|C|D|F | Not null |
| Floor | Smallint |  | 1 to 5 | Not null |
| Location | Nvarchar(5) |  | ‘Trái’ or ‘Phải’ | Not null |
| Number Bed | Smallint |  | 1 to 12 | Not null |
| Gender | Char(1) | M | ‘F’ or ‘M’ | Not null |
| Status | Char(1) | 1 | ‘1’ or ‘0’ | Not null |

***Code:***

--create table room

CREATE TABLE ROOM(

RoomID CHAR(5) PRIMARY KEY CHECK(RoomID LIKE '[A-F][1-5][0-9][0-9][0-9]' OR RoomID LIKE '[A-F][1-5][0-9][0-9][0-9][0-9]'),

[Name Dom] VARCHAR(6) CHECK([Name Dom] IN ('A','B','C','D','F')),

[Floor] SMALLINT CHECK([Floor] BETWEEN 1 AND 5) NOT NULL ,

[Location] NVARCHAR(5) CHECK(Location IN (N'Trái', N'Phải')) NOT NULL,

[Number Bed] SMALLINT CHECK([Number Bed] BETWEEN 1 AND 12) NOT NULL,

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

[Status] SMALLINT DEFAULT 1 CHECK(Status BETWEEN 0 AND 1) NOT NULL

)

***Example:***

| RoomID | Name Dom | Floor | Location | Number Bed | Gender | Status |
| --- | --- | --- | --- | --- | --- | --- |
| A1011 | A | 1 | Trái | 1 | M | 1 |
| B3056 | B | 3 | Phải | 6 | F | 0 |
| C4118 | C | 4 | Phải | 8 | M | 1 |
| D2116 | D | 2 | Phải | 6 | M | 0 |

### Create TABLE **MANAGER ROOM**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| RoomID | Char(5) |  | A|B|C|D|Fxxxxx | FK reference ROOM(RoomID) |
| GuardID | Int |  |  | FK reference GUARD(GuardID) Not null |
| Name Dom | Char(1) |  | A|B|C|D|F | Not null |
| Price | Varchar(10) |  | %VND | Not null |

***Code:***

--create table manager room

CREATE TABLE [MANAGER ROOM](

RoomID CHAR(5) FOREIGN KEY REFERENCES dbo.ROOM(RoomID) CHECK(RoomID LIKE '[A-F][1-5][0-9][0-9][0-9]' OR RoomID LIKE '[A-F][1-5][0-9][0-9][0-9][0-9]'),

GuardID INT FOREIGN KEY REFERENCES dbo.GUARD(GuardID) NOT NULL,

[Name Dom] CHAR(1) CHECK([Name Dom] IN ('A','B','C','D','F')) NOT NULL,

Price VARCHAR(10) CHECK(Price LIKE '%VND') NOT NULL

)

***Example:***

| RoomID | GuardID | Name Dom | Price |
| --- | --- | --- | --- |
| A1011 | 1 | A | 3200000VND |
| B3056 | 1 | B | 3200000VND |
| C4118 | 1 | C | 2800000VND |
| D2116 | 1 | D | 2800000VND |

### create table register

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| RoomID | Char(5) |  | A|B|C|D|Fxxxxx | FK reference ROOM(RoomID) |
| StudentID | Varchar(7) |  | SE|SB|HExxxxx | FK reference STUDENT (StudentID) |
| Check-in | Datetime |  | < =Getdate() | Not null |
| Check Price | Smallint | 1 | 0 or 1 | Not null |

***Code:***

--create table register

CREATE TABLE REGISTER(

RoomID CHAR(5) FOREIGN KEY REFERENCES dbo.ROOM(RoomID) CHECK(RoomID LIKE '[A-F][1-5][0-9][0-9][0-9]' OR RoomID LIKE '[A-F][1-5][0-9][0-9][0-9][0-9]'),

StudentID VARCHAR(7) FOREIGN KEY REFERENCES dbo.STUDENT(StudentID) CHECK(StudentID LIKE 'SE[0-9][0-9][0-9][0-9][0-9]' OR StudentID LIKE 'SB[0-9][0-9][0-9][0-9][0-9]' OR StudentID LIKE 'HE[0-9][0-9][0-9][0-9][0-9]') NOT NULL,

[Check-in] DATETIME CHECK([Check-in] <= GETDATE()) NOT NULL,

[Check Price] SMALLINT DEFAULT 1 CHECK([Check Price] BETWEEN 0 AND 1) NOT NULL

)

***Example:***

| RoomID | StudentID | Check-in | Check Price |
| --- | --- | --- | --- |
| A1113 | HE13025 | 2018-03-28 | 0 |
| A2021 | HE13027 | 2018-03-28 | 1 |
| C5012 | SE05427 | 2018-03-28 | 1 |
| F1086 | SE05627 | 2018-03-28 | 1 |

### create table **items**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| ItemID | Int |  |  | PK, Not null |
| Name | Nvarchar(50) |  |  | Not null |
| Price | Int |  |  | Not null |
| Content | Nvarchar(255) |  |  | Not null |

***Code:***

--create table items

CREATE TABLE ITEMS(

ItemID INT PRIMARY KEY NOT NULL,

[Name] NVARCHAR(50) NOT NULL,

Price INT NOT NULL,

Content NVARCHAR(255) NOT NULL

)

***Example:***

| ItemID | Name | Price | Content |
| --- | --- | --- | --- |
| 5 | Bóng điện | 120000 | Bóng đèn LED tuýp T8 N01 60/10W nhựa Rạng Đông |
| 6 | Bình Nóng Lạnh | 1800000 | Bình nước nóng 30L Aiston AN30RS-MT |
| 7 | Toilet | 500000 | Toilet KOHLER Highline Classic White |
| 3 | Giường | 3000000 | Giường tầng sắt Ký túc xá FPT – Hòa Phát |

### create table **FIX DEtail**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| RoomID | Char(5) |  | A|B|C|D|Fxxxxx | FK reference ROOM(RoomID) |
| ItemID | Int |  |  | FK reference ITEMS(ItemID) |
| Fixer | Nvarchar(90) |  |  |  |
| Data report | Datetime |  | <= Getdate() | Not null |
| Date Fix | Datetime |  | <= Getdate() |  |
| Quantity | Int |  |  | Not null |

***Code:***

--create table fixdetail

CREATE TABLE [FIX DETAIL](

RoomID CHAR(5) CHECK(RoomID LIKE '[A-F][1-5][0-9][0-9][0-9]' OR RoomID LIKE '[A-F][1-5][0-9][0-9][0-9][0-9]') FOREIGN KEY REFERENCES dbo.ROOM(RoomID),

ItemID INT FOREIGN KEY REFERENCES dbo.ITEMS(ItemID),

Fixer NVARCHAR(90),

[Date Report] DATETIME CHECK([Date Report] <= GETDATE()) NOT NULL,

[Date Fix] DATETIME CHECK([Date Fix]<= GETDATE()),

Quantity INT NOT NULL

)

***Example:***

| RoomID | ItemID | Fixer | Date Report | Date Fix | Quantity |
| --- | --- | --- | --- | --- | --- |
| A1011 | 1 | Hùng | 2018-03-02 | 2018-03-05 | 1 |
| A2021 | 2 | Hải | 2018-03-03 | 2018-03-05 | 2 |
| C3043 | 4 | NULL | 2018-03-02 | NULL | 1 |
| B1124 | 5 | NULL | 2018-03-02 | NULL | 2 |

### Create table **penalize**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| PenalizeID | Int |  |  | PK |
| Name | Nvarchar(255) |  |  | Not null |
| Price | Int |  |  | Not null |

***Code:***

--create table penalize

CREATE TABLE PENALIZE(

PenalizeID INT PRIMARY KEY,

[Name] NVARCHAR(255) NOT NULL,

Price INT NOT NULL

)

***Example:***

| PenalizeID | Name | Price |
| --- | --- | --- |
| 1 | Đi về muộn không phép | 0 |
| 2 | Hút thuốc trong ký túc xá | 50000 |
| 3 | Đun nấu trong ký túc xá | 500000 |
| 4 | Nuôi động vật trong ký túc xá | 500000 |

### create table **student penalize**

| Column Name | Data Type | Default | Check | Key/ Index/ Constraint |
| --- | --- | --- | --- | --- |
| StudentID | Varchar(7) |  | SE|SB|HExxxxx | FK reference STUDENT(StudentID) |
| PenalizeID | Int |  |  | FK reference PENALIZE(PenalizeID) |
| Date | Datetime |  | <= Getdate() | Not null |
| Quantity | Smallint | 1 |  | Not null |

***Code:***

--create table student penalize

CREATE TABLE [STUDENT PENALIZE](

StudentID VARCHAR(7) CHECK(StudentID LIKE 'SE[0-9][0-9][0-9][0-9][0-9]' OR StudentID LIKE 'SB[0-9][0-9][0-9][0-9][0-9]' OR StudentID LIKE 'HE[0-9][0-9][0-9][0-9][0-9]') FOREIGN KEY REFERENCES dbo.STUDENT(StudentID),

PenalizeID INT FOREIGN KEY REFERENCES dbo.PENALIZE(PenalizeID),

[Date] DATETIME NOT NULL CHECK(Date<=GETDATE()),

Quantity SMALLINT DEFAULT 1 NOT NULL

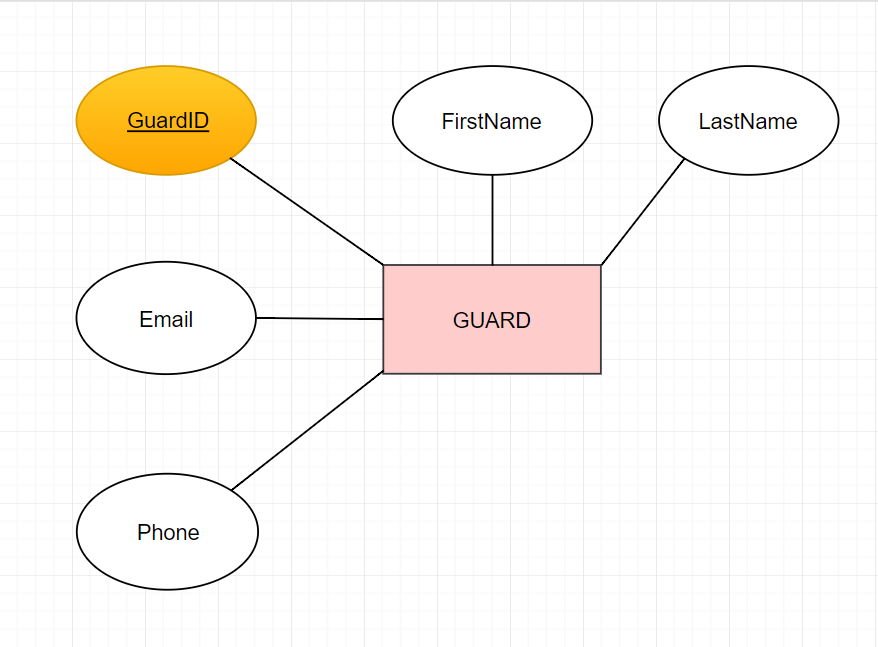
)

***Example:***

| StudentID | PenalizeID | Date | Quantity |
| --- | --- | --- | --- |
| HE13001 | 1 | 2018-03-28 | 1 |
| HE13027 | 3 | 2018-03-28 | 1 |
| SE05580 | 3 | 2018-03-28 | 3 |
| HE13025 | 2 | 2018-03-28 | 2 |

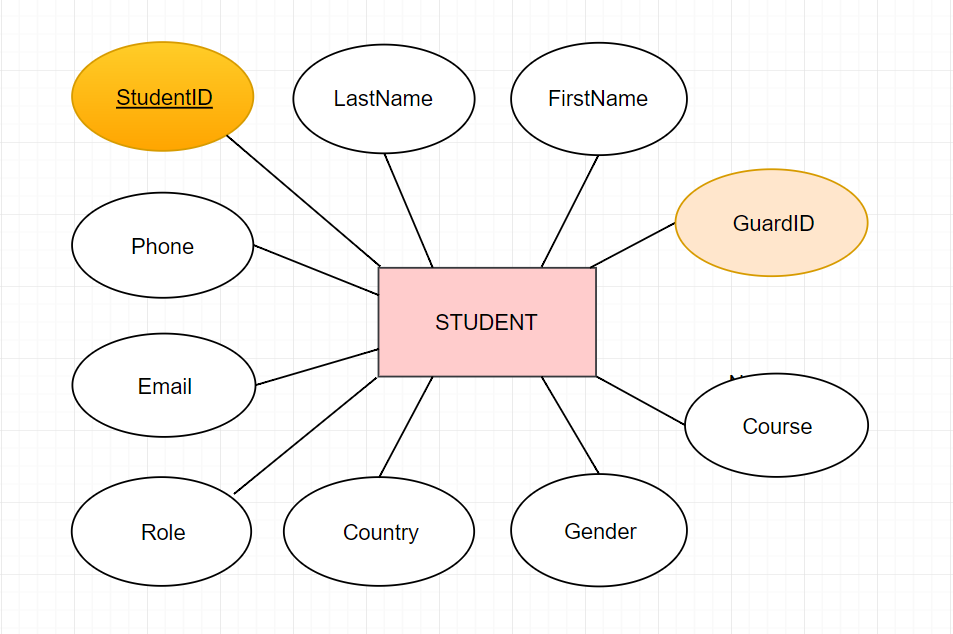
# IV. entity relationship diagram (erd)

## guard



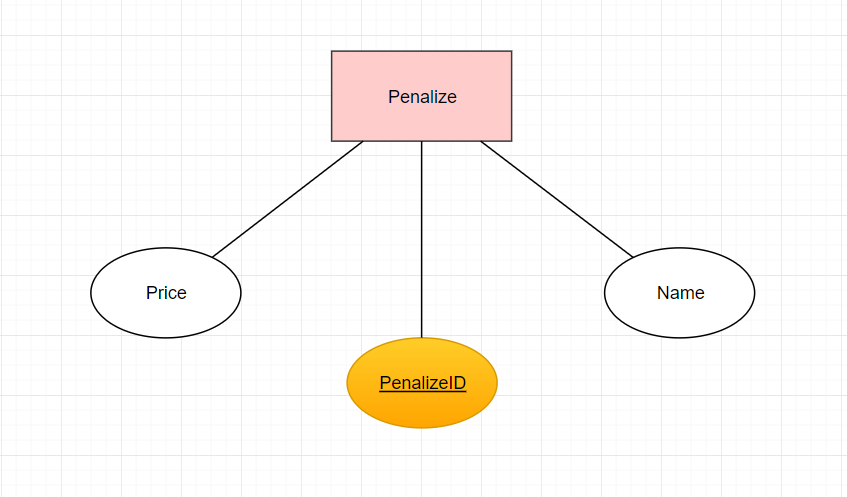
|  |  |
| --- | --- |
| This is the Guard entity, the root of whole diagram. Tournament entity has 5 attributes.  The attribute GuardID also is the primary key of this entity. Each guard has Name and contact. Name is FirstName and LastName. Guard’s contact is Email and Phone. |  |

## student



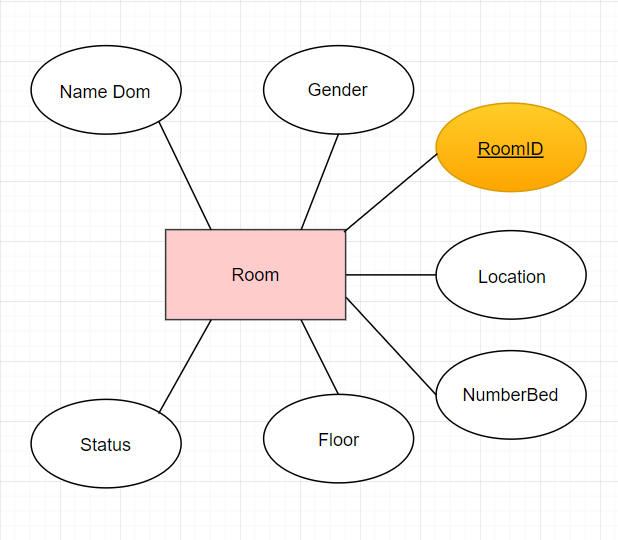
|  |  |
| --- | --- |
| This is the Student entity. This has 10 attributes.  Each student have StudentID is Primary key. Contaxt of Student has Phone number as Phone, Email. Student’s Information has FirstName, LastName Role, Country, Gender, Course. Last, each Student is Guarded by Guards so this have GuardID is foriegn key |  |
|  |  |

## panalize

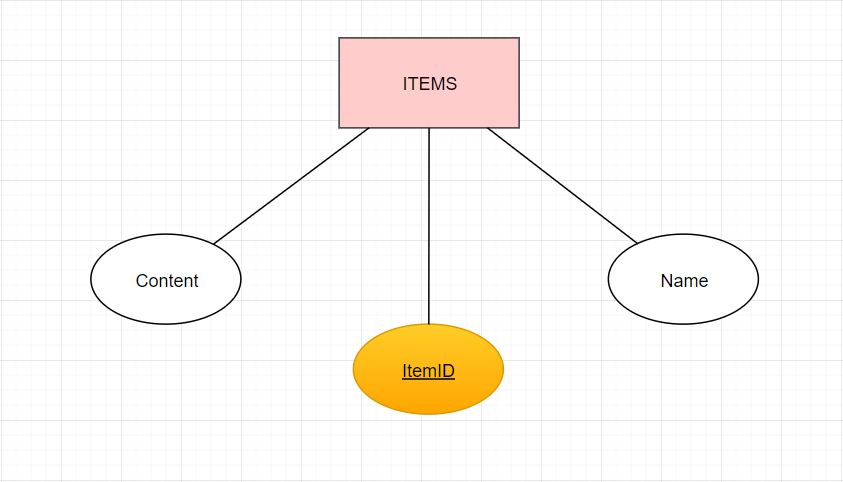


|  |  |
| --- | --- |
| Student could violate the rule do need have Penalize Entity. It Know as Name of rule be violate. PenalizeID is Primary key to insert to database. The last, Price is penaties. |  |
|  |  |

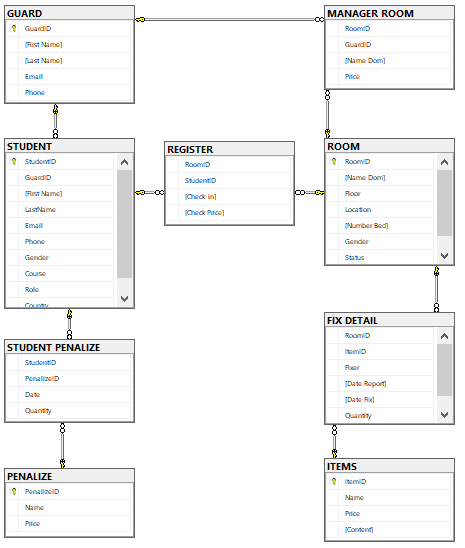
## room



|  |  |
| --- | --- |
| Room entity have 7 attributes.  To find this ROOM need information is NameDom, Floor, Location of room. Gender to know gender of Student this room, Check NumberBed to know Number of Student this Room. Status as (Full or not), The Last, RoomID to insert to database. |  |
| items |  |
|  |  |



|  |  |
| --- | --- |
| Each Student as a room hava many ITEM so this is ITEM entity.  ITEM has 3 attributes. This is ItemID is Primary key to mark many the same Items in different room, Name of Item and Content as (need fix or not). |  |
| 3 normal form  |  |  | | --- | --- | |  |  | | BEFORE | AFTER |   Because, the DomName and Floor can get by RoomID (ex: a room’s name A207. A207 the first word mean this room from DomA. A207 the first number mean this room from second Floor). We can get Gender of each room by Student’s gender. So we can drop entities to get 3NF. full diagram |  |
|  |  |



# V. sql command

I using Microsoft SQL Server 2016, this server build intelligent, mission-critical applications using a scalable, hybrid database platform that has everything built in—from in-memory performance and advanced security to in-database analytics.

## query using order by

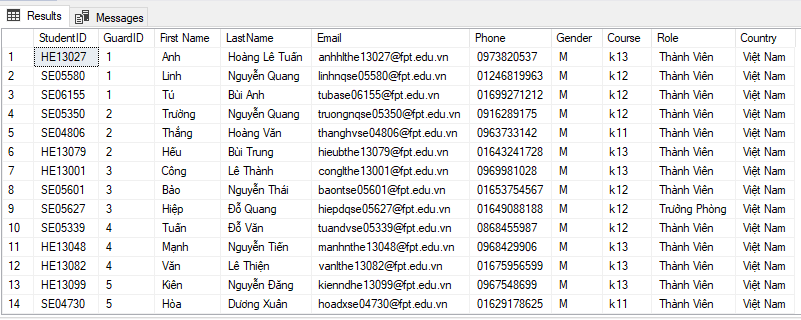
***Code:***

SELECT \*

FROM dbo.STUDENT

ORDER BY GuardID ASC

***Result:***

******

We use query containing ORDER BY to sort the list ascending or descending by the values of a domain. SELECT \* FROM STUDENT command give us all record in Student table and then sort the records ascending by GuardID which is id of guard

## query using inner join

***Code:***

SELECT s.StudentID, s.LastName +' '+ s.[First Name] AS 'Full Name', r.[Check Price]

FROM dbo.STUDENT s

INNER JOIN dbo.REGISTER r

ON r.StudentID = s.StudentID

WHERE r.[Check Price] = 1

***Result:***

|  |  |
| --- | --- |
| **In this query, we use INNER JOIN command to find out who is the students who have paid tuition. Focus on details, we inner join Player table with Student table, choose the rows adapts condition that the Check Price must equal to number writen. Then we select columns StudentID and Full name from student table and Check Price form Register table where the Check Price, which is position name is equal to 1.** |  |

## query using aggregate functions

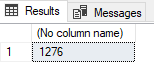
***Code:***

SELECT COUNT(r.RoomID)

FROM dbo.ROOM r

WHERE r.Status = 0

***Result:***



We count number of the room is empty of all dom from room table. We have to use function COUNT() with parameter is RoomID and condition is Status = 0 which is count the number bed is empty.

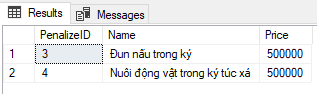
***Code:***

SELECT TOP 2 p.\*

FROM dbo.PENALIZE p

ORDER BY p.Price DESC

***Result:***



Shows another example of query using aggregate functions, which is TOP command.

We use this command to find out top of two penalize in Penalize table and sort them by the price.

## query using the group by and having clauses

***Code:***

WITH t AS (

SELECT r.RoomID,

SUBSTRING(r.RoomID,1,4) AS 'Number Room',

SUBSTRING(r.RoomID,5,5) AS 'Number Bed',

r.[Name Dom], r.FLOOR, r.Status, r.Gender

FROM dbo.ROOM r

)

SELECT t.[Number Room], COUNT(t.[Number Bed]) AS 'Bed empty'

FROM t

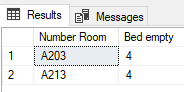
WHERE t.FLOOR = 2 AND t.[Name Dom] = 'A' AND t.Status = 0 AND t.Gender = 'M'

GROUP BY t.[Number Room]

HAVING COUNT(t.[Number Bed]) > 3

ORDER BY t.[Number Room];

***Result:***



Assume that have four boy student want to live together and they require that in floor two in dom a. So, guard have write query display name of room enough that require. Firstly, we have to create table temporary to save RoomID, Number Room, Number Bed, Name Dom, Floor, Status and gender from Room table. Then, we have to use that temporaty table, display number room and count number bed when status equal to 0 meaning is that number bed is empty. Moreover, we have to check that room for men and GROUP BY with Number Room, and HAVING count the number bed have to more than 3.

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

***Code:***

SELECT st.\*

FROM dbo.STUDENT st

WHERE st.StudentID NOT IN (

SELECT s.StudentID

FROM dbo.STUDENT s

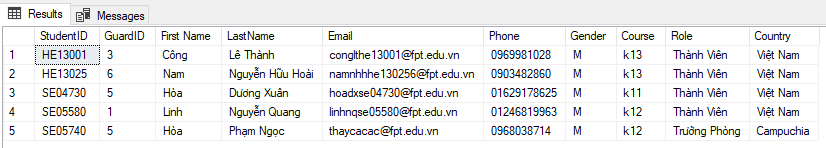
LEFT OUTER JOIN dbo.REGISTER r

ON r.StudentID = s.StudentID

WHERE r.[Check Price] = 1

)

***Result:***



If guard want to display these student doesn’t deposit in the dorm. Firstly, we will return studentID were deposit in the dorm so, Check price equal to 1 from Register table. Then we display student deposit in the dorm by way display information of student if StudentId not is appear studentId we return above.

## query that uses partial matching in the where clause

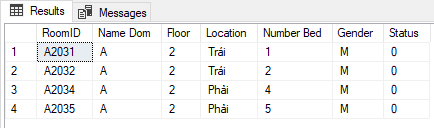
***Code:***

SELECT r.\*

FROM dbo.ROOM r

WHERE r.Status = 0 AND r.RoomID LIKE 'A203%'

***Result:***



Then guard solute issue I talk example uses having and group by, the the student want to register the room that valid that require. So we have to number of bed to student can register, we can uses LIKE combination A205 %, % it’s meaning that any letter.

## query that uses a self – join

***Code:***

SELECT s.StudentID, s.LastName +' '+ s.[First Name] AS 'Full Name', r.[Check Price]

FROM dbo.STUDENT s, dbo.REGISTER r

WHERE r.StudentID = s.StudentID AND r.[Check Price] = 1

***Result:***

|  |  |
| --- | --- |
| This query is similar to the example of query using inner join. |  |

## store procedure

***Code:***

CREATE PROC Information\_Student

@mssv VARCHAR(7)

AS

BEGIN

SELECT s.StudentID, s.LastName +' '+s.[First Name] AS 'Full Name', s.Phone, s.Email, s.Role, r.RoomID, COUNT(sp.PenalizeID) AS 'Number Penalize'

FROM dbo.STUDENT s

LEFT OUTER JOIN dbo.REGISTER r ON r.StudentID = s.StudentID

LEFT OUTER JOIN dbo.[STUDENT PENALIZE] sp ON sp.StudentID = s.StudentID

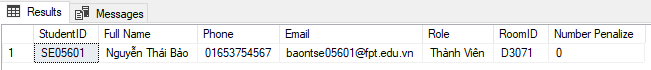
WHERE s.StudentID = @mssv

GROUP BY s.StudentID, s.LastName, s.[First Name], s.Phone, s.Email, s.Role, r.RoomID

END

EXEC dbo.Information\_Student @mssv = 'SE05601'

***Result:***



Enter the Student Code, print the student's information, how many rooms are in the room, and how many times the student owes the student's debt.

## trigger

***Code:***

CREATE TRIGGER Check\_Register

ON dbo.REGISTER

AFTER INSERT , UPDATE

AS

BEGIN

DECLARE @roomid VARCHAR(7)

SELECT @roomid = (SELECT Inserted.RoomID FROM Inserted)

IF (SELECT r.Status FROM dbo.ROOM r WHERE @roomid = r.RoomID) = 1

BEGIN

DELETE dbo.REGISTER WHERE @roomid = RoomID

PRINT 'Not empty'

END

END

A student wants to go somewhere where there must be no one.

THE END