

INT2211: Intro to Database

Assignment

Chủ đề: Cơ cấu thành phần của Bệnh Viện

Nội dung bài toán:

Cơ sở dữ liệu quản lý bệnh viện được tạo bằng cơ sở dữ liệu của SQL với mục đích quản lý và tổ chức các thành phần của bệnh viện một cách đầy đủ và chính xác nhất

Mô tả nghiệp vụ của hệ thống:

Bệnh nhân (patients): Là người tham gia điều trị tại bệnh viện, bao gồm các thông tin cơ bản như họ tên, ngày sinh, giới tính, địa chỉ, số điện thoại và thông tin liên lạc khẩn cấp

Bảo hiểm y tế (insurance): Là loại hình bảo hiểm chịu một phần chi phí điều trị tại bệnh viện cho một bệnh nhân cụ thể, bao gồm các thông tin như nhà cung cấp, số hợp đồng, mức bảo hiểm, và ngày hết hạn

Hồ sơ bệnh án (medicalrecords): Ghi lại chi tiết thông tin về bệnh tình của một bệnh nhân, bao gồm chẩn đoán, đơn thuốc, và ngày cập nhật

Bác sĩ (doctors): Là người chịu trách nhiệm điều trị cho bệnh nhân, có chuyên ngành cụ thể và thông tin liên lạc cần thiết

Lịch hẹn (appointments): Là cuộc hẹn giữa bệnh nhân và bác sĩ để thăm khám và đưa ra các phương án điều trị. Bao gồm thời gian, trạng thái và thông tin liên quan

Hóa đơn (billing): Ghi nhận thông tin thanh toán liên quan đến quá trình điều trị của bệnh nhân, bao gồm số tiền, trạng thái thanh toán và liên kết với lịch hẹn

Phản hồi (feedback): Là đánh giá và nhận xét của bệnh nhân sau các cuộc hẹn, bao gồm ngày phản hồi, điểm đánh giá và bình luận chi tiết

Khoa (departments): Là khu vực chuyên trách một lĩnh vực cụ thể trong bệnh viện, có tên gọi và thông tin quản lý

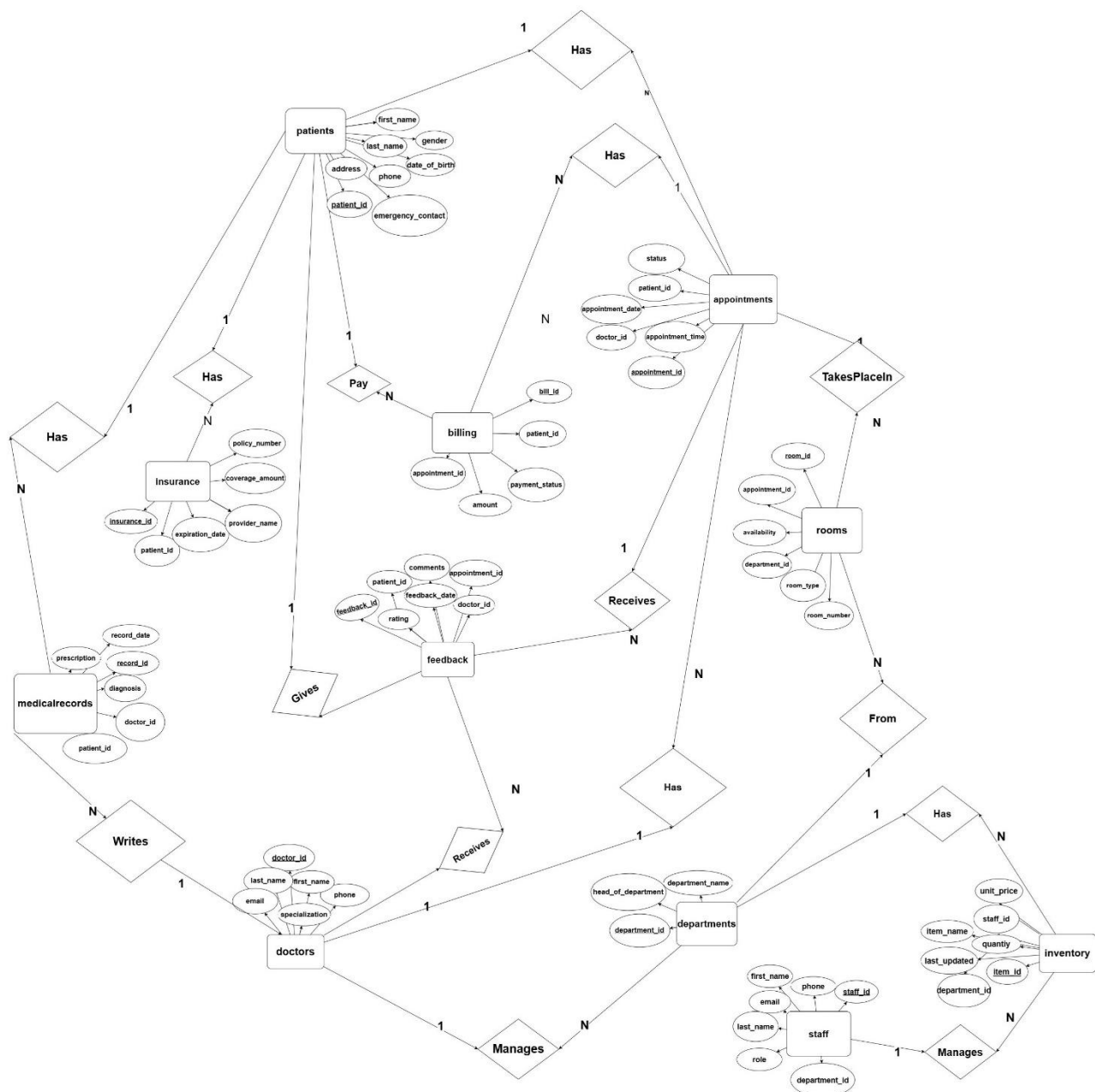
Phòng (rooms): Là phòng thuộc một khoa cụ thể, nơi diễn ra hoạt động khám chữa bệnh, bao gồm loại phòng, số phòng và trạng thái sử dụng

Thiết bị y tế (inventory): Bao gồm các thiết bị, dụng cụ và thuốc hỗ trợ điều trị, được quản lý bởi các khoa và nhân viên có liên quan

Nhân viên (staff): Là những người làm việc tại bệnh viện, đảm nhận các vai trò như thu ngân, quản lý thiết bị, kỹ thuật viên... và thuộc các khoa khác nhau

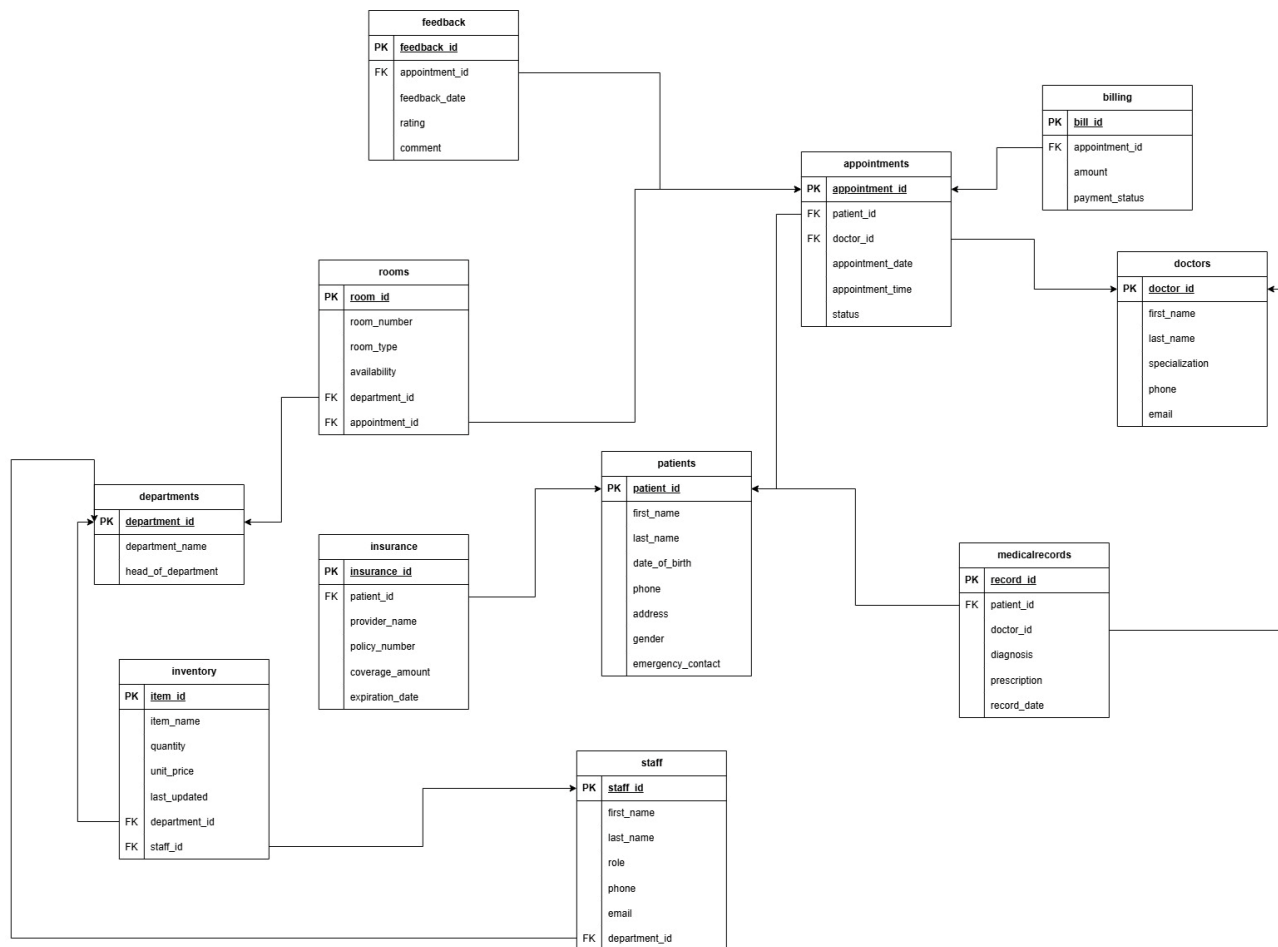
Mô hình ER của hệ thống:

Mô hình ER phù hợp với nhu cầu của một Bệnh viện (rõ hơn trong [ER_Hospital.jpg](#))



Mô hình quan hệ :

Mô hình quan hệ chuẩn hóa 3NF được chuyển từ ER (rõ hơn trong [3NF_Hospital.jpg](#))



Gồm các bảng:

patients(**patient_id**, first_name, last_name, date_of_birth, phone, address, gender, emergency_contact)

doctors(**doctor_id**, first_name, last_name, specialization, phone, email)

medicalrecords(**record_id**, *patient_id*, *doctor_id*, diagnosis, prescription, record_date)

billing(**bill_id**, *appointment_id*, amount, payment_status)

appointments(**appointment_id**, *patient_id*, *doctor_id*, appointment_date, appointment_time, status)

staff(**staff_id**, first_name, last_name, role, phone, email,
department_id)

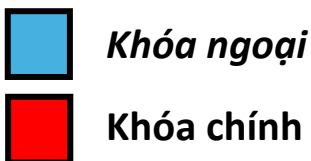
departments(**department_id**, department_name,
head_of_department)

inventory(**item_id**, item_name, quantity, unit_price, last_updated,
department_id, *staff_id*)

insurance(**insurance_id**, *patient_id*, provider_name, policy_number,
coverage_amount, expiration_date)

rooms(**room_id**, room_number, room_type, availability,
department_id, *appointment_id*)

feedback(**feedback_id**, *appointment_id*, feedback_date, rating,
comment)



Đặc tả dữ liệu các bảng:

Patients

Thuộc tính	Kiểu dữ liệu	Độ dài	Ghi chú
patient_id	INT		Khóa chính, tự động tăng
first_name	VARCHAR	50	Tên riêng
last_name	VARCHAR	50	Họ
date_of_birth	DATE		Ngày sinh
phone	VARCHAR	15	Số điện thoại
address	TEXT		Địa chỉ
gender	VARCHAR	10	`Male`, `Female`
emergency_contact	VARCHAR	15	Số điện thoại liên lạc khẩn cấp

Doctors

Thuộc tính	Kiểu dữ liệu	Độ dài	Ghi chú
doctor_id	INT		Khóa chính, tự động tăng
first_name	VARCHAR	50	Tên riêng
last_name	VARCHAR	50	Họ
specialization	VARCHAR	100	Chuyên ngành
phone	VARCHAR	15	Số điện thoại
email	VARCHAR	100	Email

Medicalrecords

Thuộc tính	Kiểu dữ liệu	Độ dài	Ghi chú
record_id	INT		Khóa chính, tự động tăng
patient_id	INT		Khóa ngoại tới `patients(patient_id)`
doctor_id	INT		Khóa ngoại tới `doctors(doctor_id)`
diagnosis	TEXT		Chẩn đoán
prescription	TEXT		Đơn thuốc
record_date	DATE		Ngày tạo hồ sơ

Appointments

Thuộc tính	Kiểu dữ liệu	Độ dài	Ghi chú
appointment_id	INT		Khóa chính, tự động tăng
patient_id	INT		Khóa ngoại tới `patients(patient_id)`
doctor_id	INT		Khóa ngoại tới `doctors(doctor_id)`
appointment_date	DATE		Ngày hẹn
appointment_time	TIME		Giờ hẹn
status	VARCHAR	20	`Scheduled`, `Completed`, `Cancelled`

Billing

Thuộc tính	Kiểu dữ liệu	Độ dài	Ghi chú
bill_id	INT		Khóa chính, tự động tăng
appointment_id	INT		Khóa ngoại tới `appointments(appointment_id)`
amount	DECIMAL	10,2	Tổng tiền
payment_status	VARCHAR	20	`Paid`, `Unpaid`, `Pending`

Inventory

Thuộc tính	Kiểu dữ liệu	Độ dài	Ghi chú
item_id	INT		Khóa chính, tự động tăng
item_name	VARCHAR	100	Tên thiết bị
quantity	INT		Số lượng
unit_price	DECIMAL	10,2	Giá mỗi đơn vị
last_updated	DATETIME		Thời gian cập nhật cuối
department_id	INT		Khóa ngoại tới `departments(department_id)`
staff_id	INT		Khóa ngoại tới `staff(staff_id)`

Staff

Thuộc tính	Kiểu dữ liệu	Độ dài	Ghi chú
staff_id	INT		Khóa chính, tự động tăng
first_name	VARCHAR	50	Tên riêng
last_name	VARCHAR	50	Họ
role	VARCHAR	50	Vai trò công việc (ví dụ: Thu ngân)
phone	VARCHAR	15	Số điện thoại
email	VARCHAR	100	Email
department_id	INT		Khóa ngoại tới `departments(department_id)`

Departments

Thuộc tính	Kiểu dữ liệu	Độ dài	Ghi chú
department_id	INT		Khóa chính, tự động tăng
department_name	VARCHAR	100	Tên khoa
head_of_department	VARCHAR	100	Trưởng khoa

Insurance

Thuộc tính	Kiểu dữ liệu	Độ dài	Ghi chú
insurance_id	INT		Khóa chính, tự động tăng
patient_id	INT		Khóa ngoại tới `patients(patient_id)`
provider_name	VARCHAR	100	Tên nhà cung cấp bảo hiểm
policy_number	VARCHAR	50	Số hợp đồng bảo hiểm
coverage_amount	DECIMAL	10,2	Số tiền bảo hiểm chi trả
expiration_date	DATE		Ngày hết hạn bảo hiểm

Rooms

Thuộc tính	Kiểu dữ liệu	Độ dài	Ghi chú
room_id	INT		Khóa chính, tự động tăng
room_number	VARCHAR	20	Số phòng
room_type	VARCHAR	50	Loại phòng (ví dụ: Phòng đơn, phòng đôi)
availability	BOOLEAN		DEFAULT TRUE
department_id	INT		Khóa ngoại tới `departments(department_id)`
appointment_id	INT		Khóa ngoại tới `appointments(appointment_id)`

Feedback

Thuộc tính	Kiểu dữ liệu	Độ dài	Ghi chú
feedback_id	INT		Khóa chính, tự động tăng
appointment_id	INT		Khóa ngoại tới 'appointments(appointment_id)'
feedback_date	DATE		Ngày phản hồi
rating	INT		Điểm đánh giá (thang 1-5)
comment	TEXT		Nhận xét chi tiết

Các ràng buộc dữ liệu:

Patients và insurance: Mối quan hệ thông qua trường "patient_id". Mỗi Bệnh nhân (Patient) sẽ có một bảo hiểm y tế (insurance) tương ứng

Patients và Appointments: Mối quan hệ thông qua trường "patient_id". Mỗi bệnh nhân có thể có nhiều lịch hẹn khám (Appointments), mỗi lịch hẹn phải gắn liền với một bệnh nhân cụ thể

Doctors và Appointments: Mối quan hệ thông qua trường "doctor_id". Một bác sĩ có thể thực hiện nhiều cuộc hẹn khám (Appointments), mỗi lịch hẹn chỉ gắn với một bác sĩ duy nhất

Appointments và Billing: Mối quan hệ thông qua trường “appointment_id”. Mỗi lịch hẹn (Appointments) có thể nhận một hoặc không có phản hồi (Feedback) từ bệnh nhân, và mỗi phản hồi chỉ thuộc về một lịch hẹn

Departments và Rooms: Mối quan hệ thông qua trường “department_id”. Mỗi phòng (Rooms) thuộc về một khoa (Departments) cụ thể. Một khoa có thể có nhiều phòng.

Departments và Inventory: Mối quan hệ thông qua trường “department_id”. Mỗi thiết bị y tế hoặc vật tư (Inventory) phải thuộc về một khoa (Departments). Một khoa có thể quản lý nhiều thiết bị y tế.

Staff và Inventory: Mối quan hệ thông qua trường “staff_id”. Mỗi thiết bị y tế hoặc vật tư (Inventory) có thể được quản lý bởi một nhân viên (Staff), và một nhân viên có thể quản lý nhiều thiết bị.

Departments và Staff: Mối quan hệ thông qua trường “department_id”. Mỗi nhân viên (Staff) thuộc về một khoa (Departments), và một khoa có thể có nhiều nhân viên.

Medicalrecords và Patients/Doctors: Mối quan hệ thông qua các trường “patient_id” và “doctor_id”. Mỗi hồ sơ bệnh án (Medicalrecords) phải liên kết với một bệnh nhân (Patients) và một bác sĩ (Doctors). Một bệnh nhân có thể có nhiều hồ sơ, và một bác sĩ có thể tạo hồ sơ cho nhiều bệnh nhân.

Rooms và Appointments: Mối quan hệ thông qua trường “appointment_id”. Một phòng (Rooms) có thể được sử dụng cho một lịch hẹn cụ thể, nhưng không bắt buộc. Một phòng có thể được đặt cho nhiều lịch hẹn khác nhau vào các thời điểm khác nhau.

Cài đặt trong SQL:

Tạo ra các bảng với mã script **CreateDB.sql**

```
CREATE DATABASE hospital;

USE hospital;

CREATE TABLE Patients (
    patient_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    date_of_birth DATE NOT NULL,
    phone VARCHAR(15),
    address TEXT,
    gender VARCHAR(10),
    emergency_contact VARCHAR(15)
);

CREATE TABLE Doctors (
    doctor_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    specialization VARCHAR(50),
    phone VARCHAR(15),
    email VARCHAR(100)
);

CREATE TABLE Departments (
    department_id INT AUTO_INCREMENT PRIMARY KEY,
    department_name VARCHAR(50) NOT NULL,
    head_of_department INT,
    FOREIGN KEY (head_of_department) REFERENCES
Doctors(doctor_id)
);
```

```
CREATE TABLE Appointments (  
    appointment_id INT AUTO_INCREMENT PRIMARY KEY,  
    patient_id INT NOT NULL,  
    doctor_id INT NOT NULL,  
    appointment_date DATE NOT NULL,  
    appointment_time TIME NOT NULL,  
    status VARCHAR(20),  
    FOREIGN KEY (patient_id) REFERENCES Patients(patient_id),  
    FOREIGN KEY (doctor_id) REFERENCES Doctors(doctor_id)  
);  
  
CREATE TABLE MedicalRecords (  
    record_id INT AUTO_INCREMENT PRIMARY KEY,  
    patient_id INT NOT NULL,  
    doctor_id INT NOT NULL,  
    diagnosis TEXT,  
    prescription TEXT,  
    record_date DATE NOT NULL,  
    FOREIGN KEY (patient_id) REFERENCES Patients(patient_id),  
    FOREIGN KEY (doctor_id) REFERENCES Doctors(doctor_id)  
);  
  
CREATE TABLE Billing (  
    bill_id INT AUTO_INCREMENT PRIMARY KEY,  
    patient_id INT NOT NULL,  
    appointment_id INT NOT NULL,  
    amount DECIMAL(10, 2) NOT NULL,  
    payment_status VARCHAR(20),  
    FOREIGN KEY (patient_id) REFERENCES Patients(patient_id),
```

```
        FOREIGN KEY (appointment_id) REFERENCES  
Appointments(appointment_id)
```

```
);
```

```
CREATE TABLE Rooms (  
    room_id INT AUTO_INCREMENT PRIMARY KEY,  
    room_number VARCHAR(10) NOT NULL,  
    room_type VARCHAR(20),  
    availability BOOLEAN DEFAULT TRUE,  
    department_id INT,  
    appointment_id INT
```

```
);
```

```
CREATE TABLE Staff (  
    staff_id INT AUTO_INCREMENT PRIMARY KEY,  
    first_name VARCHAR(50) NOT NULL,  
    last_name VARCHAR(50) NOT NULL,  
    role VARCHAR(50) NOT NULL,  
    phone VARCHAR(15),  
    email VARCHAR(100),  
    department_id INT,  
    FOREIGN KEY (department_id) REFERENCES  
Departments(department_id)
```

```
);
```

```
CREATE TABLE Inventory (  
    item_id INT AUTO_INCREMENT PRIMARY KEY,  
    item_name VARCHAR(100) NOT NULL,  
    quantity INT NOT NULL,  
    unit_price DECIMAL(10, 2),  
    last_updated DATE NOT NULL,  
    department_id INT,
```

```
        staff_id INT
    );

CREATE TABLE Insurance (
    insurance_id INT AUTO_INCREMENT PRIMARY KEY,
    patient_id INT NOT NULL,
    provider_name VARCHAR(100) NOT NULL,
    policy_number VARCHAR(50),
    coverage_amount DECIMAL(10, 2),
    expiration_date DATE NOT NULL,
    FOREIGN KEY (patient_id) REFERENCES Patients(patient_id)
);

CREATE TABLE Feedback (
    feedback_id INT AUTO_INCREMENT PRIMARY KEY,
    patient_id INT NOT NULL,
    doctor_id INT,
    appointment_id INT,
    feedback_date DATE NOT NULL,
    rating INT CHECK (rating BETWEEN 1 AND 5),
    comments TEXT,
    FOREIGN KEY (patient_id) REFERENCES Patients(patient_id),
    FOREIGN KEY (doctor_id) REFERENCES Doctors(doctor_id),
    FOREIGN KEY (appointment_id) REFERENCES
Appointments(appointment_id)
);
```

Tạo ra các ràng buộc qua mã **Constraints.sql**:

+Các ràng buộc đặc biệt:

***Đảm bảo rằng email của mỗi bác sĩ là duy nhất**

```
ALTER TABLE Doctors  
ADD CONSTRAINT unique_email UNIQUE (email);
```

***Đảm bảo rằng "head_of_department" phải là một bác sĩ hợp lệ.**

```
ALTER TABLE Departments  
ADD CONSTRAINT fk_department_head FOREIGN KEY  
(head_of_department) REFERENCES Doctors(doctor_id);
```

***Tránh việc một bệnh nhân có nhiều lịch hẹn trùng lặp với cùng một bác sĩ vào cùng thời gian**

```
ALTER TABLE Appointments  
ADD CONSTRAINT unique_appointment UNIQUE (patient_id,  
doctor_id, appointment_date, appointment_time);
```

***Tránh trùng lặp email giữa các nhân viên**

```
ALTER TABLE Staff  
ADD CONSTRAINT unique_email UNIQUE (email);
```


***Đảm bảo dữ liệu tồn kho chính xác**

```
ALTER TABLE Inventory  
  
ADD CONSTRAINT fk_inventory_departments FOREIGN KEY  
(department_id) REFERENCES Departments(department_id),  
  
ADD CONSTRAINT fk_inventory_staff FOREIGN KEY  
(staff_id) REFERENCES Staff(staff_id),  
  
ADD CONSTRAINT check_quantity CHECK (quantity >= 0),
```

***Đảm bảo tính chính xác của thông tin bảo hiểm**

```
ALTER TABLE Insurance  
  
ADD CONSTRAINT check_coverage_amount CHECK  
(coverage_amount >= 0),  
  
ADD CONSTRAINT check_expiration_date CHECK  
(expiration_date IS NOT NULL);
```

+Các ràng buộc cơ bản

Thêm 5 mẫu vào mỗi bảng qua mã **Insert.sql**

Mã **queries.sql** thỏa mãn các điều kiện trả lời các truy vấn:

+Truy vấn sử dụng **INNER JOIN**

-- a. Truy vấn sử dụng INNER JOIN: Lấy tất cả các lịch hẹn cùng thông tin chi tiết của bệnh nhân và bác sĩ

```
SELECT
    a.appointment_id,
    p.first_name AS patient_name,
    d.first_name AS doctor_name,
    a.appointment_date,
    a.appointment_time,
    a.status
FROM
    Appointments a
INNER JOIN
    Patients p ON a.patient_id = p.patient_id
INNER JOIN
    Doctors d ON a.doctor_id = d.doctor_id;
```

+Truy vấn sử dụng **OUTER JOIN**

-- b. Truy vấn sử dụng OUTER JOIN: Lấy tất cả các phòng kèm thông tin chi tiết của khoa, bao gồm các khoa không có phòng

```
SELECT
    r.room_id,
    r.room_number,
    r.room_type,
    r.availability,
    d.department_name
```

```
FROM
    Rooms r
RIGHT OUTER JOIN
    Departments d ON r.department_id =
d.department_id;
```

+Truy vấn sử dụng subquery trong **WHERE**

-- c. Truy vấn sử dụng subquery trong WHERE: Lấy tất cả các bác sĩ đã từng điều trị ít nhất một bệnh nhân với lịch hẹn đã hoàn thành

```
SELECT
    d.doctor_id,
    d.first_name,
    d.last_name
FROM
    Doctors d
WHERE
    d.doctor_id IN (
        SELECT
            a.doctor_id
        FROM
            Appointments a
        WHERE
            a.status = 'Completed'
    );
```

+Truy vấn sử dụng subquery trong **FROM**

-- d. Truy vấn sử dụng subquery trong FROM: Lấy tổng số bệnh nhân mà mỗi bác sĩ đã điều trị

```
SELECT
    doc.first_name AS doctor_name,
    COUNT(*) AS total_patients
FROM
    (SELECT
        a.doctor_id, p.patient_id
    FROM
        Appointments a
    INNER JOIN
        Patients p ON a.patient_id = p.patient_id
    GROUP BY
        a.doctor_id, p.patient_id) AS
patient_summary
INNER JOIN
    Doctors doc ON patient_summary.doctor_id =
doc.doctor_id
GROUP BY
    doc.first_name;
```

+Truy vấn sử dụng **GROUP BY** và các hàm tổng hợp

-- e. Truy vấn sử dụng GROUP BY và các hàm tổng hợp:
Lấy tổng doanh thu và số lượng lịch hẹn của từng bác sĩ

```
SELECT
    d.first_name AS doctor_name,
    COUNT(a.appointment_id) AS total_appointments,
    SUM(b.amount) AS total_revenue
FROM
    Doctors d
INNER JOIN
    Appointments a ON d.doctor_id = a.doctor_id
INNER JOIN
    Billing b ON a.appointment_id = b.appointment_id
WHERE
    b.payment_status = 'Paid'
GROUP BY
    d.first_name;
```

Tạo một transaction có tác dụng đặt phòng bệnh trong mã
Transaction.sql

```
-- Bắt đầu một giao dịch (transaction)

START TRANSACTION;

-- Bước 1:Cập nhật trạng thái phòng (đổi trạng thái
phòng từ có sẵn ->không có sẵn)

UPDATE Rooms

SET availability = FALSE

WHERE room_id = 1 and department_id = 1;

-- Bước 2:Cập nhật trạng thái thanh toán hóa đơn (đổi
trạng thái từ Pending ->Paid)

UPDATE Billing

SET payment_status = 'Paid'

WHERE bill_id = 2;

-- Nếu cả hai bước trên đều thực hiện thành công, xác
nhận giao dịch (commit)

COMMIT;

-- Nếu xảy ra lỗi ở bất kỳ bước nào, hủy bỏ giao dịch
(rollback) để đảm bảo tính toàn vẹn dữ liệu

ROLLBACK;
```

Tạo một trigger để thêm hồ sơ y tế sau khi cuộc hẹn được hoàn thành trong mã **Trigger.sql**

```
-- Tạo trigger để thêm hồ sơ y tế sau khi cuộc hẹn
được hoàn thành

DELIMITER $$

CREATE TRIGGER after_appointment_completed
AFTER UPDATE ON Appointments
FOR EACH ROW
BEGIN
    -- Kiểm tra nếu trạng thái cuộc hẹn được cập nhật
    thành 'Completed'
    IF NEW.status = 'Completed' THEN
        -- Thêm bản ghi mới vào bảng MedicalRecords
        INSERT INTO MedicalRecords (patient_id,
            doctor_id, diagnosis, prescription, record_date)
            VALUES (NEW.patient_id, NEW.doctor_id, 'Chưa
            cập nhật', 'Chưa cập nhật', NOW());
        END IF;
    END$$

DELIMITER ;
```

Tạo một procedure có tác dụng đặt lịch hẹn với bác sĩ qua mã

Procedure.sql

```
DELIMITER $$
```

```
CREATE PROCEDURE BookAppointment (
```

```
    IN patientID INT, -- Tham số đầu vào :ID của bệnh nhân
```

```
    IN doctorID INT, -- ID của bác sĩ
```

```
    IN appointmentDate DATE, -- Ngày hẹn
```

```
    IN appointmentTime TIME, -- Giờ hẹn
```

```
    OUT resultMessage VARCHAR(255) -- Tham số đầu ra  
:Thông điệp kết quả
```

```
)
```

```
BEGIN
```

```
    DECLARE doctorExists INT; -- Biến lưu trạng thái  
kiểm tra sự tồn tại của bác sĩ
```

```
    DECLARE patientExists INT; --
```

```
_____ của bệnh nhân
```

```
    DECLARE appointmentConflict INT; -- Biến kiểm tra  
xung đột lịch hẹn
```

```
-- Bắt đầu khối có nhãn
```

```
proc: BEGIN
```

```
-- Kiểm tra xem bác sĩ có tồn tại không
```

```
SELECT COUNT(*) INTO doctorExists
```

```
FROM doctors
```



```
WHERE doctor_id = doctorID; -- Kiểm tra ID
bác sĩ trong bảng doctors
```

```
IF doctorExists = 0 THEN -- Nếu bác sĩ không
tồn tại
```

```
SET resultMessage = 'Ko tồn tại bác sĩ';
-- Gán thông báo lỗi
```

```
LEAVE proc; -- Thoát khối
```

```
END IF;
```

```
-- Kiểm tra xem bệnh nhân có tồn tại không
```

```
SELECT COUNT(*) INTO patientExists
```

```
FROM patients
```

```
WHERE patient_id = patientID; -- Kiểm tra ID
bệnh nhân trong bảng patients
```

```
IF patientExists = 0 THEN -- Nếu bệnh nhân
không tồn tại
```

```
SET resultMessage = 'Bệnh nhân ko tồn
tại'; -- Gán thông báo lỗi
```

```
LEAVE proc; -- Thoát khỏi khối
```

```
END IF;
```

```
-- Kiểm tra xung đột lịch hẹn với bác sĩ
```

```
SELECT COUNT(*) INTO appointmentConflict
```

```
FROM appointments
```

```
WHERE doctor_id = doctorID -- Kiểm tra ID bác
sĩ
```

```

        AND appointment_date = appointmentDate --
Cùng ngày hẹn

        AND appointment_time = appointmentTime; --
Cùng giờ hẹn

    IF appointmentConflict > 0 THEN -- Nếu có
xung đột lịch hẹn

        SET resultMessage = 'Lịch bị trùng, bác
sĩ được chỉ định đang tạm vắng'; -- Gán thông báo lỗi

    ELSE

        -- Thêm lịch hẹn mới vào bảng
appointments

        INSERT INTO appointments (patient_id,
doctor_id, appointment_date, appointment_time,
status)

        VALUES (patientID, doctorID,
appointmentDate, appointmentTime, 'Scheduled'); --
Trạng thái mặc định là Scheduled

        SET resultMessage = 'Đặt hẹn thành công';
-- Gán thông báo thành công

    END IF;

    END proc; -- Kết thúc khối có nhãn
END$$

DELIMITER ;

```

Kết luận:

Hệ thống có thể hỗ trợ người bệnh đặt khám, giúp bệnh viện lưu trữ và quản lý dữ liệu thông tin người khám chữa bệnh.

Thành viên:

- Vũ Đức Huy (23020380)
- Vũ Đức Minh (23020401)
- Hoàng Minh Quyền (23020421)
- Phạm Bảo Lăng (23020391)