**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN HỌC PHẦN: HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**ĐỀ TÀI: QUẢN LÝ BỆNH VIỆN**

**Giáo viên hướng dẫn: ThS. Nguyễn Ngọc Ân**

**Sinh viên thực hiện:**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Mã sv** | **Họ và tên** | **Lớp** |
| 1 | 1771020064 | Nguyễn Thị Ngọc Ánh | CNTT 17-14 |
| 2 | 1771020244 | Nguyễn Thu Hằng | CNTT 17-14 |
| 3 | 1771020675 | Nguyễn Thị Thanh Trà | CNTT 17-14 |

**Hà Nội, năm 2025**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN HỌC PHẦN: HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**ĐỀ TÀI: QUẢN LÝ BỆNH VIỆN**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Mã Sinh Viên | Họ và Tên | Ngày Sinh | Điểm | |
| Bằng Số | Bằng Chữ |
| 1 | 1771020064 | Nguyễn Thị Ngọc Ánh | 20/11/2005 |  |  |
| 2 | 1771020244 | Nguyễn Thu Hằng | 20/02/2005 |  |  |
| 3 | 1771020675 | Nguyễn Thị Thanh Trà | 24/11/2005 |  |  |

**CÁN BỘ CHẤM THI**

**Hà Nội, năm 2025**

**LỜI NÓI ĐẦU**

Trong bối cảnh công nghệ thông tin ngày càng phát triển, việc ứng dụng các hệ quản trị cơ sở dữ liệu (HQTCSDL) vào quản lý và vận hành bệnh viện đã trở thành một yêu cầu cấp thiết. Quản lý bệnh viện là một lĩnh vực phức tạp, đòi hỏi sự chính xác, nhanh chóng và hiệu quả trong việc xử lý lượng lớn thông tin liên quan đến bệnh nhân, hồ sơ bệnh án, lịch trình khám chữa bệnh, dược phẩm, nhân sự và nhiều khía cạnh khác.

Đề tài "Quản lý Bệnh viện " tập trung vào việc nghiên cứu và xây dựng một hệ thống quản lý thông tin toàn diện cho bệnh viện, sử dụng các công cụ và kỹ thuật của HQTCSDL.

Chúng tôi chọn nghiên cứu và phát triển đề tài: “Quản lí bệnh viện” để có thể đóng góp vào việc nâng cao chất lượng quản lý và chăm sóc sức khỏe tại các bệnh viện, mang lại lợi ích thiết thực cho cả bệnh nhân và đội ngũ y tế.

**MỤC LỤC**

[**CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI** 8](#_Toc193296413)

[**1.1. Giới thiệu tổng quan về đề tài** 8](#_Toc193296414)

[**1.2. Mục đích và lý do chọn đề tài** 8](#_Toc193296415)

[**CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ CƠ SỞ DỮ LIỆU** 9](#_Toc193296416)

[**2.1. Xác định các yêu cầu của hệ thống** 9](#_Toc193296417)

[*2.1.1. Yêu cầu chức năng* 9](#_Toc193296418)

[*2.1.2. Yêu cầu phi chức năng* 10](#_Toc193296419)

[*2.1.3. Các thực thể chính trong hệ thống* 10](#_Toc193296420)

[**2.2. Thiết kế mô hình dữ liệu (ERD)** 10](#_Toc193296421)

[**2.3. Chuyển đổi mô hình dữ liệu sang mô hình vật lý (bảng, khóa, ràng buộc)** 12](#_Toc193296422)

[**CHƯƠNG 3. TRIỂN KHAI CƠ SỞ DỮ LIỆU** 15](#_Toc193296423)

[**3.1. Tạo cơ sở dữ liệu và các bảng với các định nghĩa khóa, ràng buộc và các mối quan hệ** 15](#_Toc193296424)

[**3.2. Nhập dữ liệu mẫu cho các bảng trên** 16](#_Toc193296425)

[**3.3. Truy vấn cơ bản (SELECT, INSERT, UPDATE, DELETE)** 20](#_Toc193296426)

[**3.4. Truy vấn nâng cao (INNER JOIN, GROUP BY, HAVING, SUBQUERY)** 22](#_Toc193296427)

[**CHƯƠNG 4: LẬP TRÌNH T-SQL** 23](#_Toc193296428)

[**4.1. Tạo 7-10 view từ cơ bản đến nâng cao** 23](#_Toc193296429)

[**4.2. Tạo 7-10 index cần thiết cho các bảng** 27](#_Toc193296430)

[**4.3. Xây dựng 10 Stored Procedure(không tham số, có tham số, có OUTPUT)** 30](#_Toc193296431)

[**4.4. Tạo 10 function (trả về kiểu vô hướng, bảng, biến bảng)** 35](#_Toc193296432)

[**4.5. Tạo 7- 10 trigger để kiểm soát dữ liệu** 40](#_Toc193296433)

[**CHƯƠNG 5 : BẢO MẬT VÀ QUẢN TRỊ** 45](#_Toc193296434)

[**5.1. Tạo người dùng** 45](#_Toc193296435)

[**5.2. Thiết lập các quyền truy vập và phân quyền người dùng** 46](#_Toc193296436)

[**5.3. Quản lí sao lưu và phục hồi dữ liệu** 47](#_Toc193296437)

[**CHƯƠNG 6: KẾT LUẬN** 49](#_Toc193296438)

[**6.1. Nhận xét về quá trình thực hiện bài tập lớn** 49](#_Toc193296439)

[**6.2. Đánh giá ưu và nhược điểm của hệ thống** 49](#_Toc193296440)

[**6.3. Đề xuất các hướng phát triển tiếp theo** 50](#_Toc193296441)

[**DANH MỤC TÀI LIỆU THAM KHẢO** 52](#_Toc193296442)

**BẢNG CÁC TỪ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| **STT** | **TỪ VIẾT TẮT** | **VIẾT ĐẦY ĐỦ** |
| 1 | DBMS | Database Management System |
| 2 | HQTCSDL | Hệ quản trị cơ sở dữ liệu |

**CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI**

## **1.1. Giới thiệu** **tổng quan về đề tài**

- Quản lý bệnh viện là một bài toán quan trọng trong lĩnh vực y tế, giúp nâng cao hiệu quả hoạt động của bệnh viện, tối ưu hóa quy trình khám chữa bệnh và quản lý dữ liệu bệnh nhân. Trong môn học Hệ quản trị cơ sở dữ liệu (Database Management System - DBMS), đề tài này tập trung vào việc xây dựng một hệ thống cơ sở dữ liệu nhằm lưu trữ, quản lý và truy xuất thông tin một cách hiệu quả.

- Đề tài Quản lý bệnh viện trong môn học Hệ quản trị cơ sở dữ liệu không chỉ giúp sinh viên hiểu rõ cách thiết kế, triển khai và quản lý cơ sở dữ liệu mà còn có tính ứng dụng thực tế cao trong lĩnh vực y tế. Hệ thống này giúp tối ưu hóa quy trình quản lý bệnh viện, nâng cao hiệu suất làm việc và cải thiện trải nghiệm của bệnh nhân.

## **1.2. Mục đích và lý do chọn đề tài**

* Mục đích nghiên cứu:
* Xây dựng hệ thống cơ sở dữ liệu hỗ trợ quản lý bệnh viện, bao gồm các thành phần như: bệnh nhân, bác sĩ, đơn thuốc, phòng bệnh, lịch khám, hóa đơn thanh toán...
* Tối ưu hóa quá trình lưu trữ và truy xuất thông tin, giúp giảm thiểu sai sót trong quản lý hồ sơ y tế.
* Ứng dụng các kỹ thuật của hệ quản trị cơ sở dữ liệu để đảm bảo tính bảo mật, toàn vẹn và hiệu suất cao.
* Lý do chọn đề tài:
* Bệnh viện là một môi trường có lượng dữ liệu lớn, phức tạp, đòi hỏi hệ thống quản lý hiệu quả để hỗ trợ hoạt động chăm sóc sức khỏe.
* Đề tài này giúp áp dụng các kiến thức về hệ quản trị cơ sở dữ liệu vào thực tế, từ thiết kế mô hình quan hệ, tối ưu hóa truy vấn đến đảm bảo an toàn dữ liệu.
* Nâng cao hiểu biết về việc triển khai các hệ thống thông tin y tế, đáp ứng nhu cầu chuyển đổi số trong lĩnh vực y tế hiện nay.

**CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ CƠ SỞ DỮ LIỆU**

## **2.1. Xác định các yêu cầu của hệ thống**

*2.1.1. Yêu cầu chức năng*

Hệ thống cần đáp ứng các chức năng sau:

1. Quản lý bác sĩ:

* Lưu trữ thông tin bác sĩ: mã bác sĩ, họ tên, địa chỉ, số điện thoại.
* Quản lý chuyên khoa của bác sĩ: mỗi bác sĩ thuộc ít nhất một chuyên khoa.
* Theo dõi mối quan hệ giữa bác sĩ và bệnh nhân.

1. Quản lý chuyên khoa:

* Lưu trữ thông tin chuyên khoa: mã chuyên khoa, tên chuyên khoa.
* Gán bác sĩ vào chuyên khoa phù hợp.

1. Quản lý bệnh nhân:

* Lưu trữ thông tin bệnh nhân: mã bệnh nhân, họ tên, địa chỉ, số chứng minh thư (hoặc CCCD).
* Theo dõi lịch sử điều trị của bệnh nhân (bác sĩ phụ trách, ngày giờ điều trị, liệu pháp điều trị).

1. Quản lý y tá:

* Lưu trữ thông tin y tá: mã y tá, họ tên, địa chỉ, số điện thoại.
* Theo dõi việc chăm sóc bệnh nhân của từng y tá theo ngày, bao gồm nội dung chăm sóc (uống thuốc, tiêm, truyền…).

1. Quản lý mỗi quan hệ giữa bác sĩ, bệnh nhân và y tá:

* Mỗi bác sĩ có thể điều trị nhiều bệnh nhân hoặc không có bệnh nhân nào.
* Mỗi bệnh nhân được theo dõi bởi một bác sĩ.
* Mỗi bệnh nhân có thể được nhiều y tá chăm sóc, và mỗi y tá có thể chăm sóc nhiều bệnh nhân.
* Lưu lại lịch sử chăm sóc bệnh nhân của y tá theo từng ngày.

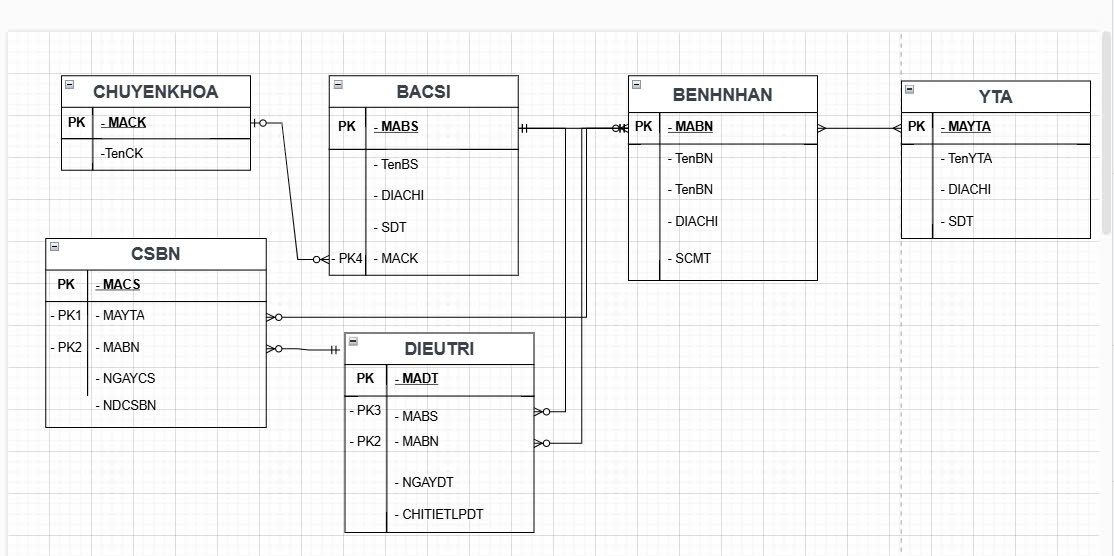
*2.1.2. Yêu cầu phi chức năng*

* Bảo mật: Chỉ có những người có quyền mới được truy cập và chỉnh sửa thông tin.
* Toàn vẹn dữ liệu: Đảm bảo dữ liệu không bị trùng lặp hoặc sai lệch.
* Hiệu suất: Truy xuất dữ liệu nhanh chóng khi cần xem thông tin bệnh nhân, bác sĩ, y tá.
* Dễ sử dụng: Giao diện trực quan, thân thiện với người dùng (nếu có hệ thống quản lý đi kèm).
* Linh hoạt: Hệ thống có thể mở rộng để thêm nhiều tính năng khác như quản lý thuốc, phòng bệnh…

*2.1.3. Các thực thể chính trong hệ thống*

* Bác sĩ (Doctor): mã bác sĩ, họ tên, địa chỉ, số điện thoại.
* Chuyên khoa (Specialty): mã chuyên khoa, tên chuyên khoa.
* Bệnh nhân (Patient): mã bệnh nhân, họ tên, địa chỉ, số chứng minh thư.
* Y tá (Nurse): mã y tá, họ tên, địa chỉ, số điện thoại.
* Điều trị (Treatment): mã bác sĩ, mã bệnh nhân, ngày giờ, liệu pháp điều trị.
* Chăm sóc (Care): mã y tá, mã bệnh nhân, ngày tháng, nội dung chăm sóc.

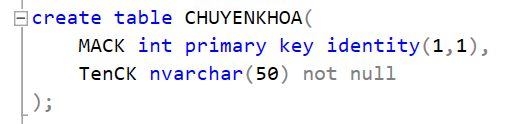
## **2.2. Thiết kế mô hình dữ liệu (ERD)**

****

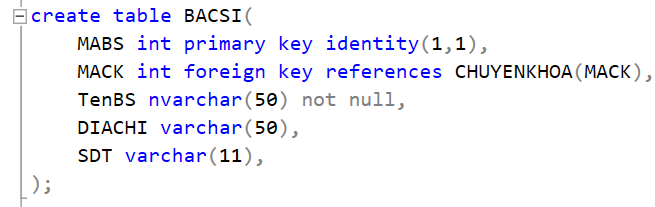
* Các thực thể (bảng) chính trong hệ thống
* CHUYENKHOA (Chuyên khoa)
* MACK (Khóa chính): Mã chuyên khoa.
* TenCK: Tên chuyên khoa.
* BACSI (Bác sĩ)
* MABS (Khóa chính): Mã bác sĩ.
* TenBS: Tên bác sĩ.
* DIACHI: Địa chỉ của bác sĩ.
* SDT: Số điện thoại của bác sĩ.
* MACK (Khóa ngoại): Mã chuyên khoa của bác sĩ.
* BENHNHAN (Bệnh nhân)
* MABN (Khóa chính): Mã bệnh nhân.
* TenBN: Tên bệnh nhân.
* DIACHI: Địa chỉ bệnh nhân.
* SCMT: Số chứng minh thư (hoặc CCCD) của bệnh nhân.
* YTA (Y tá)
* MAYTA (Khóa chính): Mã y tá.
* TenYTA: Tên y tá.
* DIACHI: Địa chỉ của y tá.
* SDT: Số điện thoại của y tá.
* Các bảng quan hệ (liên kết thực thể)
* DIEUTRI (Điều trị)
* MAYTA (Khóa chínhMADT (Khóa chính): Mã điều trị.
* MABS (Khóa ngoại): Mã bác sĩ điều trị bệnh nhân.
* MABN (Khóa ngoại): Mã bệnh nhân được điều trị.
* NGAYDT: Ngày điều trị.
* CHITIETLPDT: Chi tiết liệu pháp điều trị.
* CSBN (Chăm sóc bệnh nhân)
* MACS (Khóa chính): Mã chăm sóc bệnh nhân.
* MAYTA (Khóa ngoại): Mã y tá thực hiện chăm sóc.
* MABN (Khóa ngoại): Mã bệnh nhân được chăm sóc.
* NGAYCS: Ngày chăm sóc bệnh nhân.
* NDCSBN: Nội dung chăm sóc (uống thuốc, tiêm, truyền dịch…).
* Các mối quan hệ trong hệ thống
* Bác sĩ thuộc một hoặc nhiều chuyên khoa.
* Bệnh nhân được một bác sĩ điều trị, nhưng một bác sĩ có thể điều trị nhiều bệnh nhân.
* Bệnh nhân có thể được nhiều y tá chăm sóc theo từng ngày, và một y tá cũng có thể chăm sóc nhiều bệnh nhân.
* Điều trị lưu lại thông tin bác sĩ điều trị bệnh nhân, thời gian điều trị và chi tiết liệu pháp.
* Chăm sóc bệnh nhân ghi lại lịch sử chăm sóc bệnh nhân của các y tá, ngày chăm sóc và nội dung chăm sóc.

2.3. Chuyển đổi mô hình dữ liệu sang mô hình vật lý (bảng, khóa, ràng buộc)

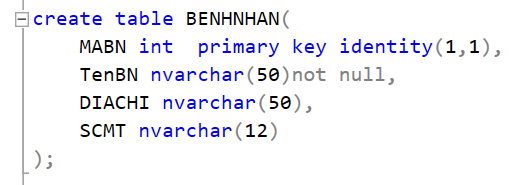
* Bảng CHUYENKHOA (Chuyên khoa)



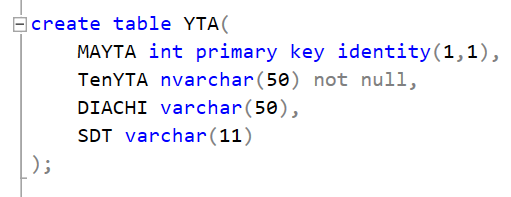
* Bảng BACSI (Bác sĩ)



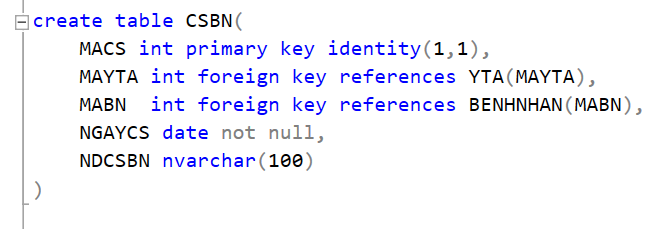
* Bảng BENHNHAN (Bệnh nhân)



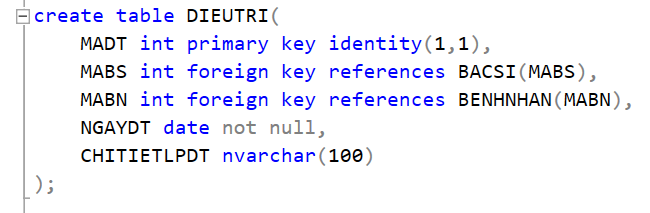
* Bảng YTA (Y tá)



* Bảng CSBN (Chăm sóc bệnh nhân)



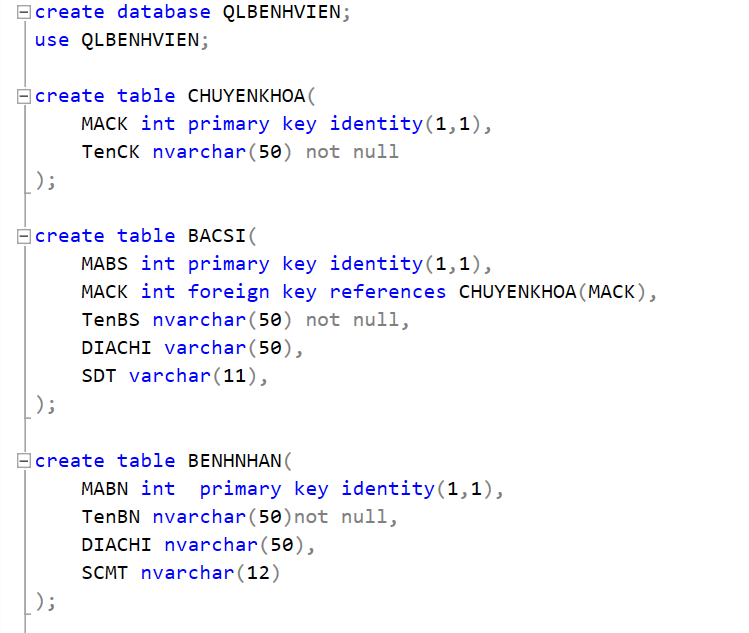
* Bảng DIEUTRI (Điều trị)

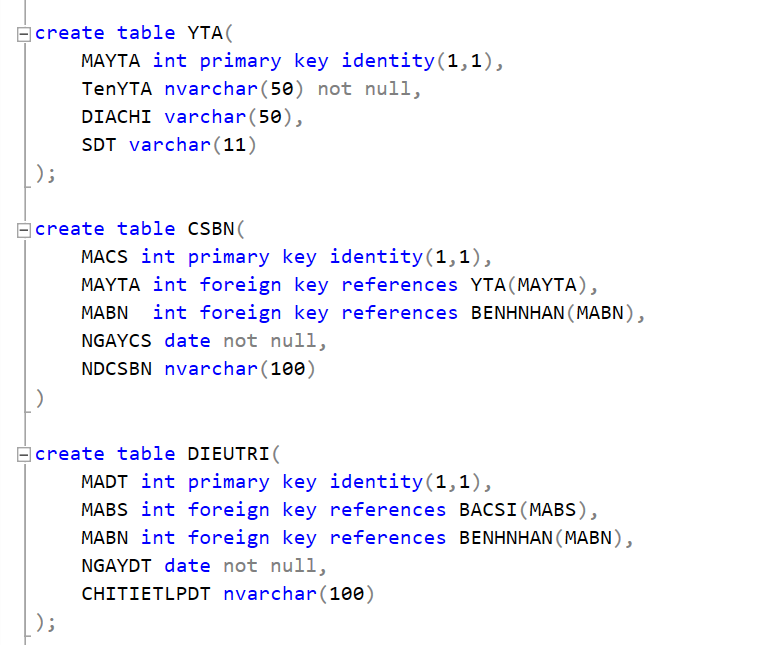


* Tóm tắt quan hệ giữa các bảng
* BACSI liên kết với CHUYENKHOA qua MACK.
* BENHNHAN liên kết với BACSI và YTA qua bảng CSBN.
* BENHNHAN liên kết với DIEUTRI để lưu thông tin điều trị của các bác sĩ.
* CSBN là bảng trung gian giữa BENHNHAN và YTA.
* DIEUTRI là bảng trung gian giữa BENHNHAN và BACSI.
* Ràng buộc và đặc điểm
* Khóa chính (PRIMARY KEY): Được sử dụng để đảm bảo mỗi bảng có một cột hoặc nhóm cột định danh duy nhất.
* Khóa ngoại (FOREIGN KEY): Được sử dụng để liên kết giữa các bảng, đảm bảo tính toàn vẹn dữ liệu.
* Ràng buộc NOT NULL: Ngăn chặn giá trị NULL ở những cột bắt buộc.
* Ràng buộc UNIQUE: Đảm bảo không có giá trị trùng lặp, ví dụ như số CMND của bệnh nhân.

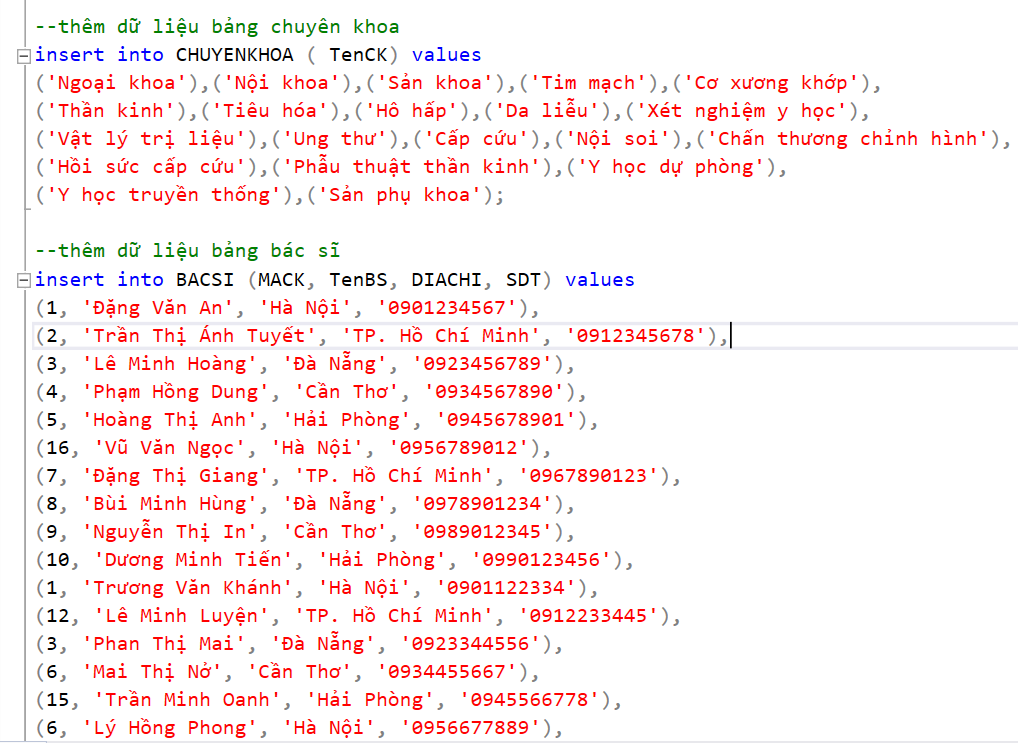
**CHƯƠNG 3. TRIỂN KHAI CƠ SỞ DỮ LIỆU**

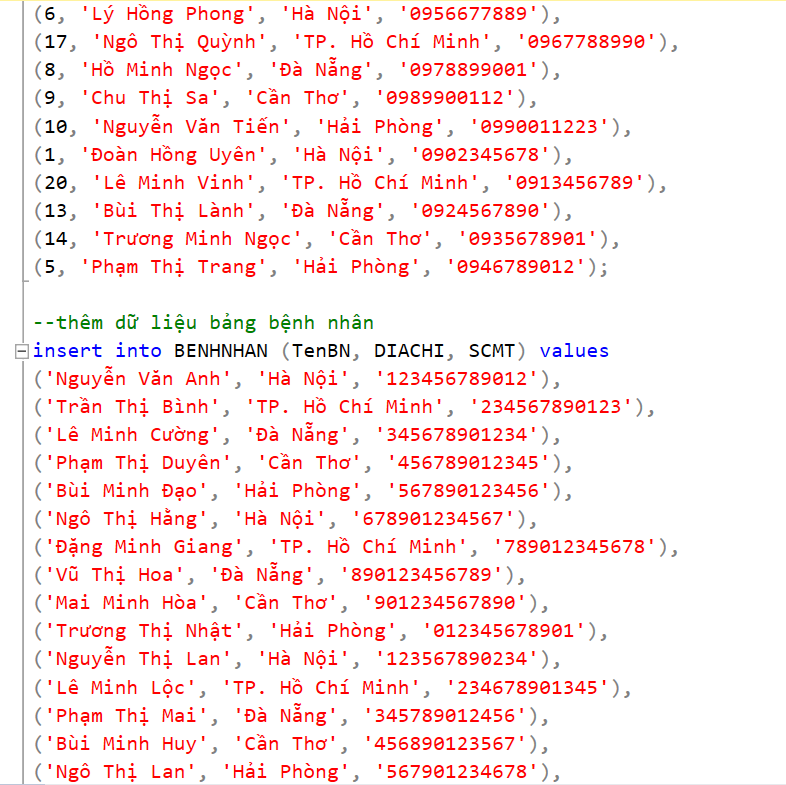
## **3.1. Tạo cơ sở dữ liệu và các bảng với các định nghĩa khóa, ràng buộc và các mối quan hệ**

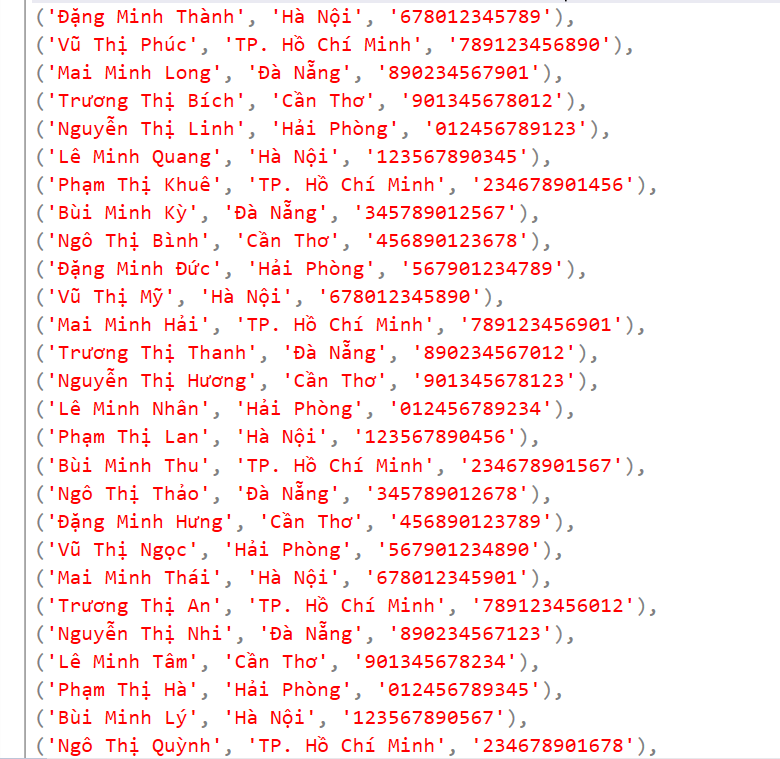
****

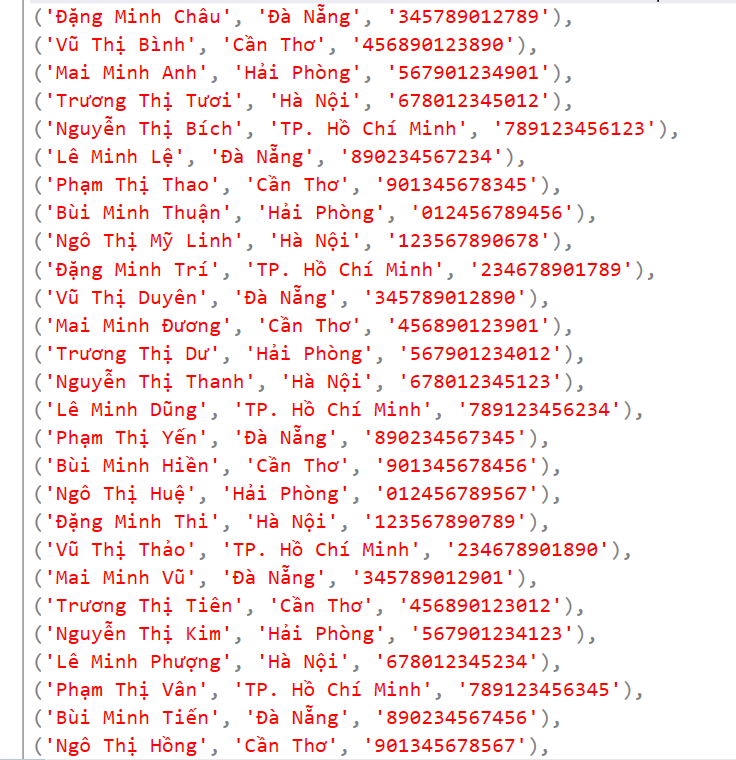
****

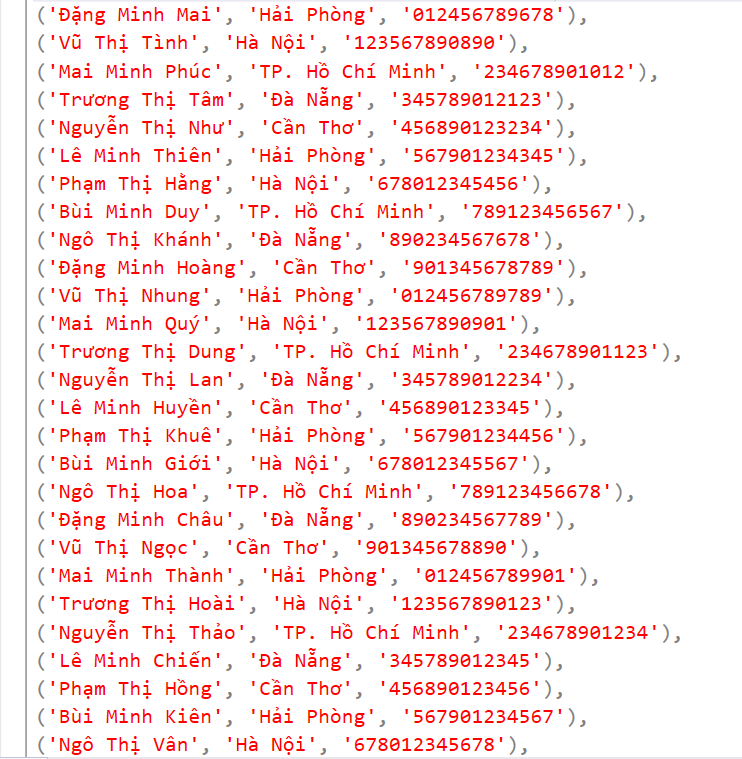
## **3.2. Nhập dữ liệu mẫu cho các bảng trên**

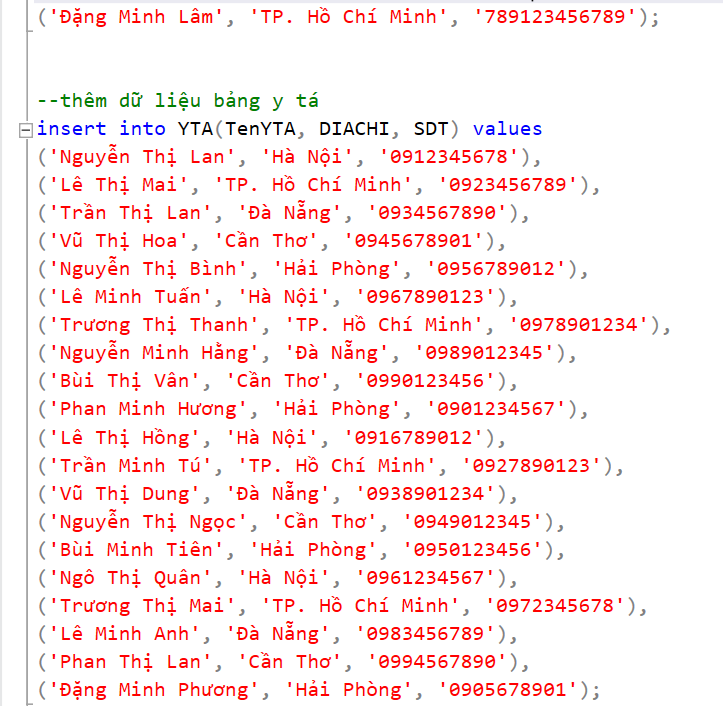
****

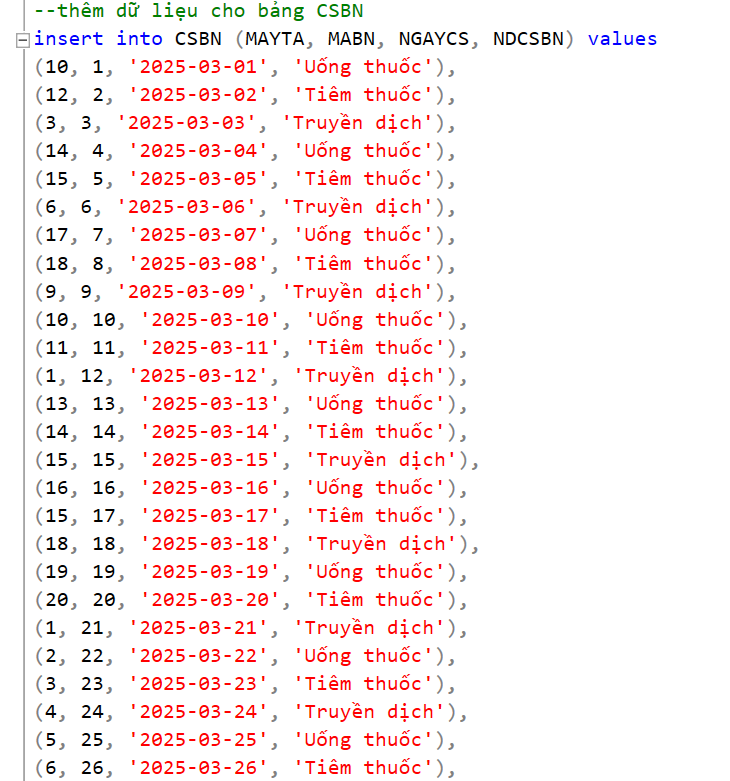
****

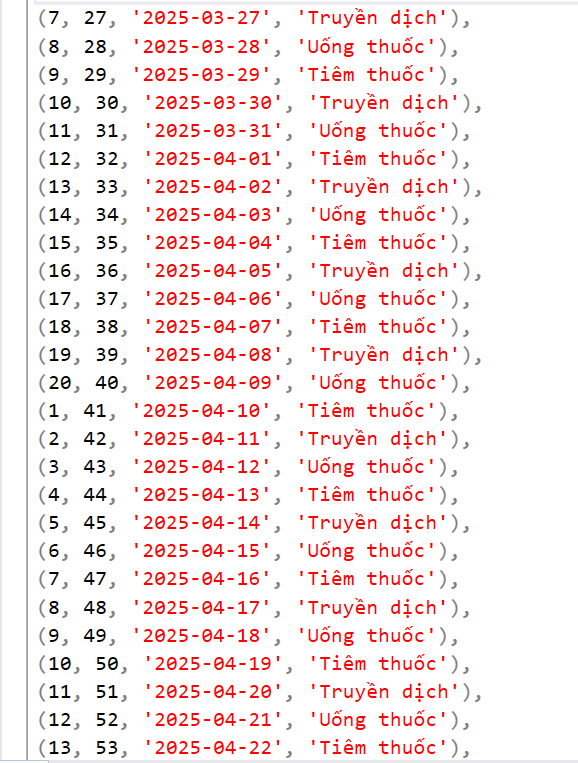
****

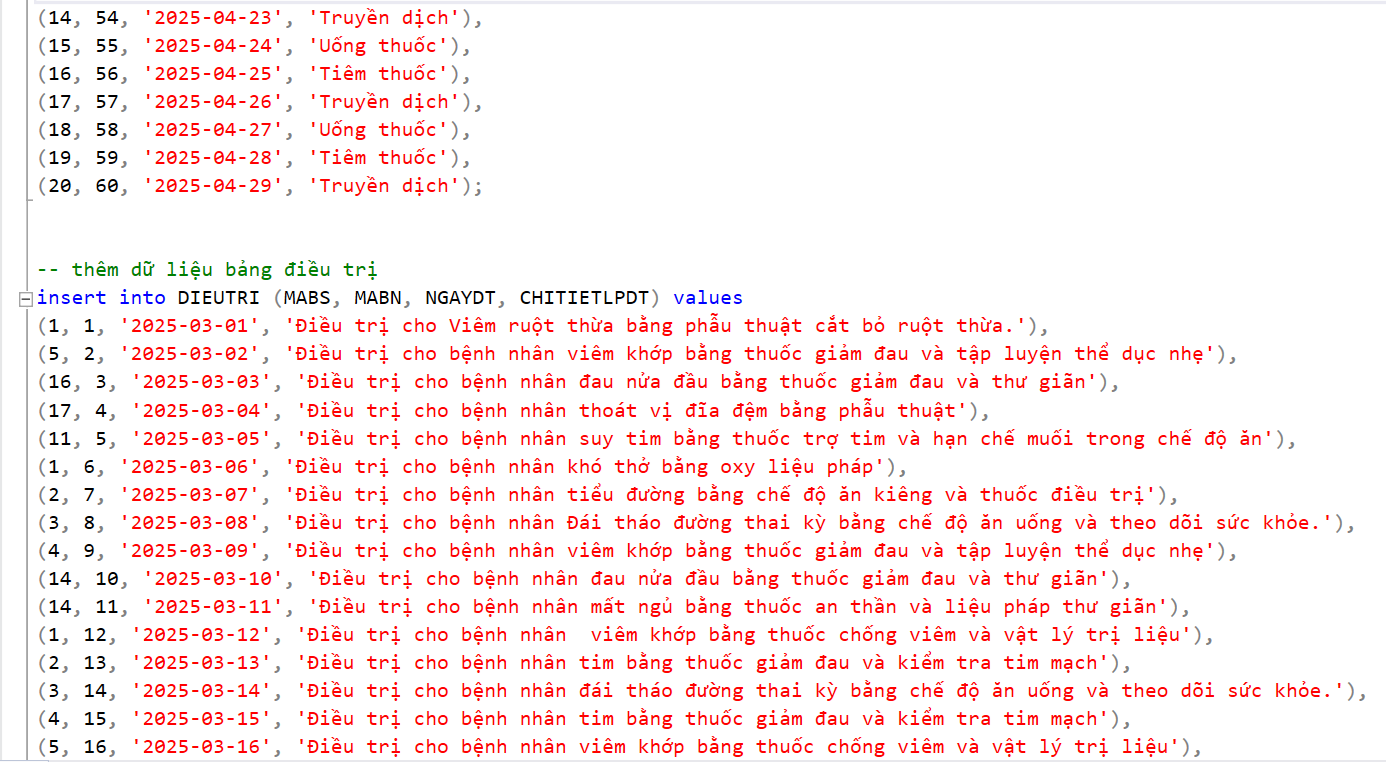
****

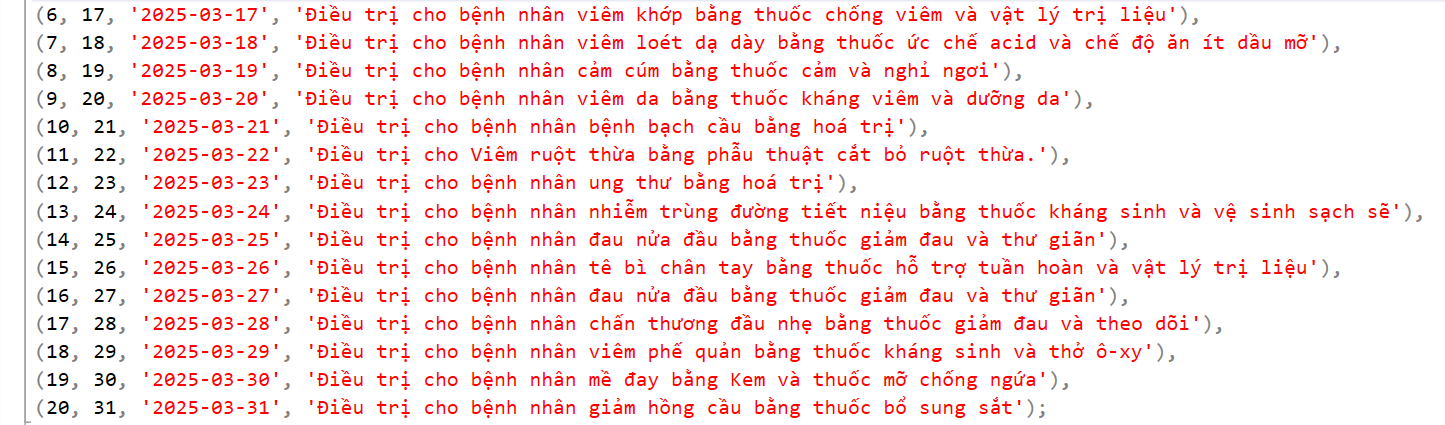
****

****

****

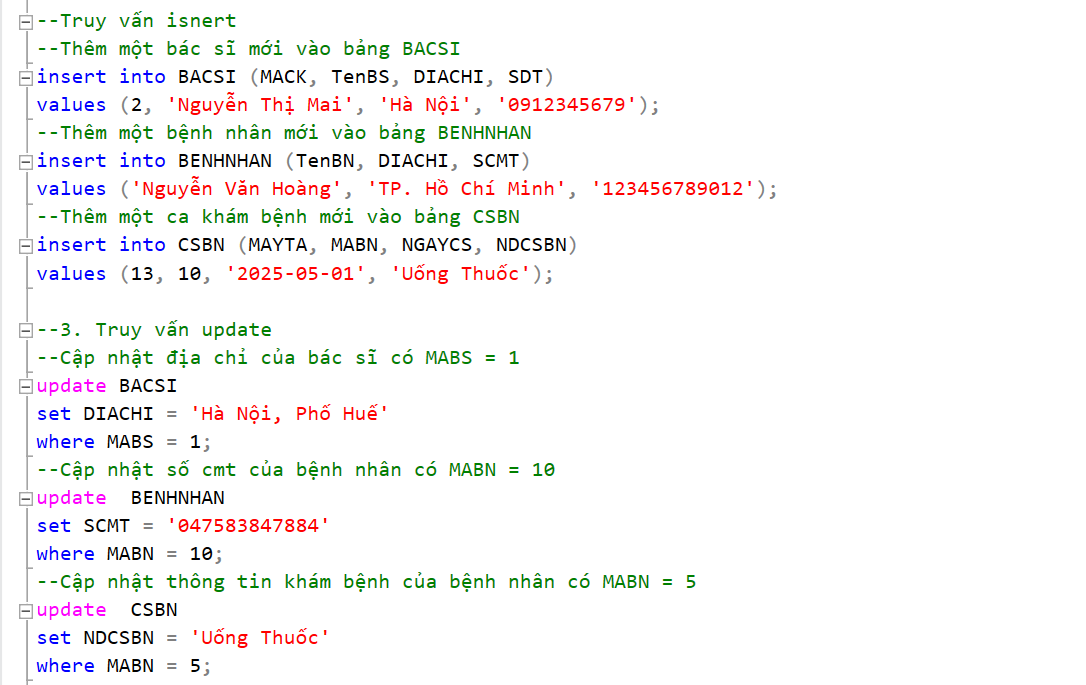
****

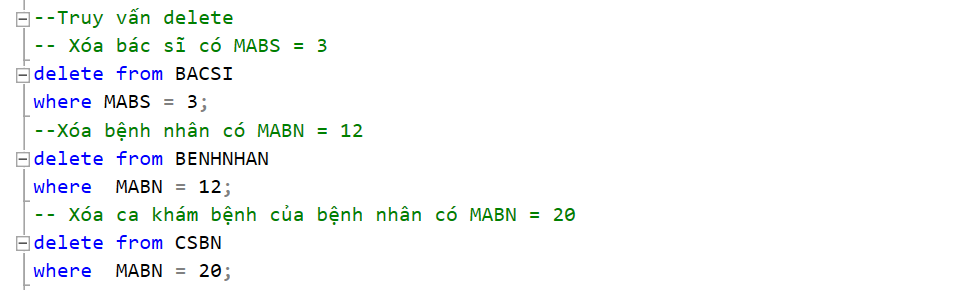
****

****

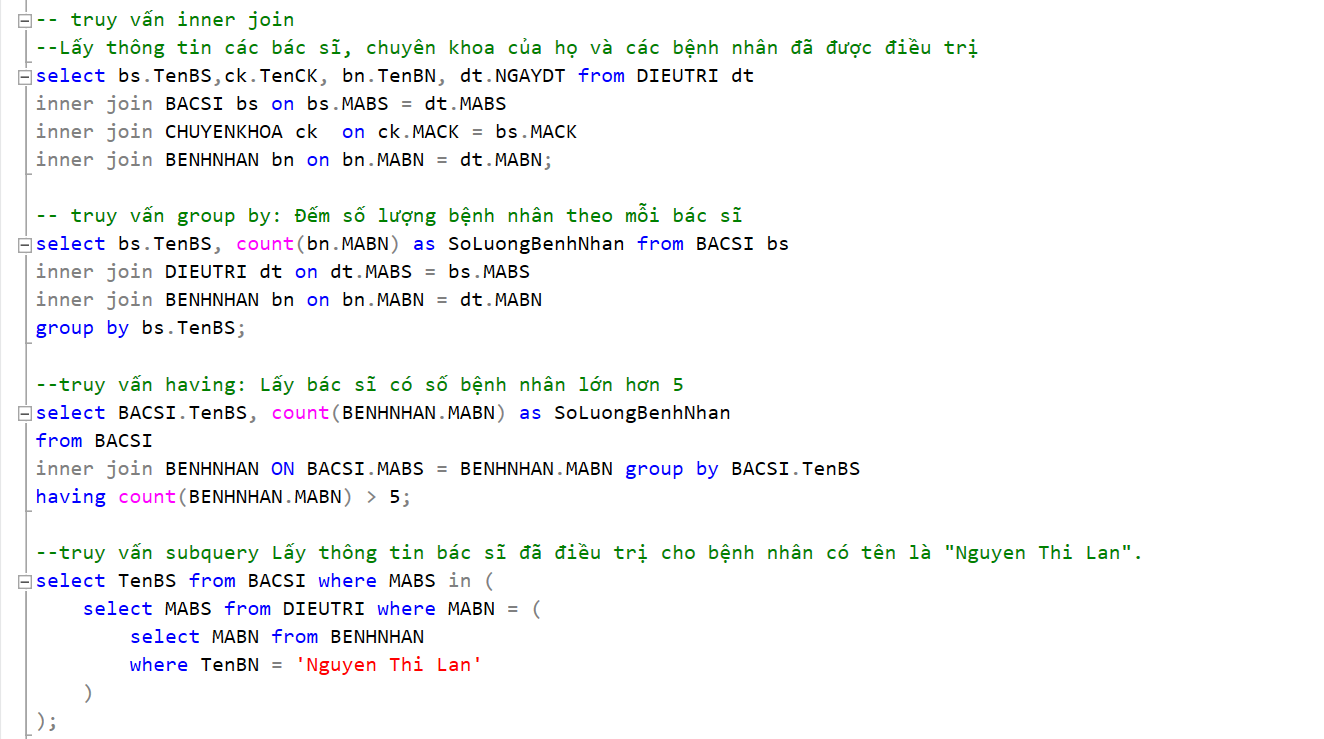
## **3.3. Truy vấn cơ bản (SELECT, INSERT, UPDATE, DELETE)**

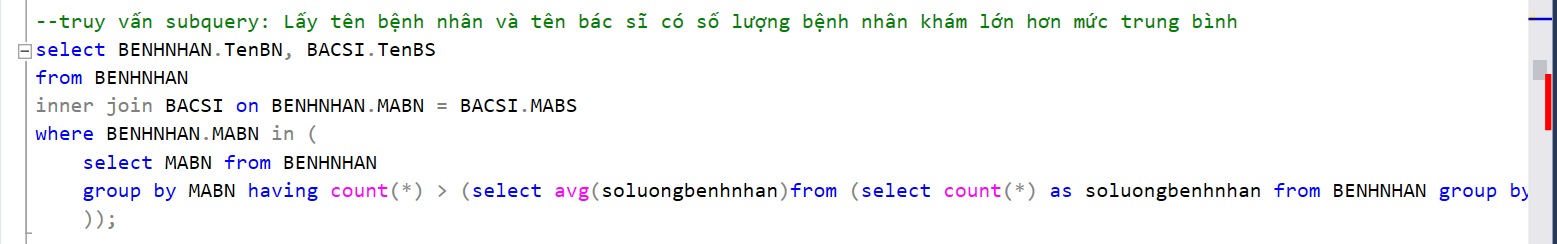
****

****

****

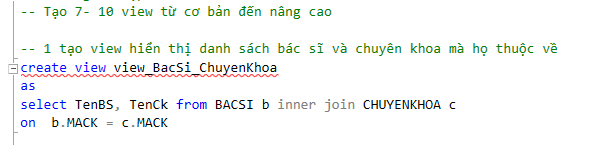
3.4. Truy vấn nâng cao (INNER JOIN, GROUP BY, HAVING, SUBQUERY)

****

****

**CHƯƠNG 4: LẬP TRÌNH T-SQL**

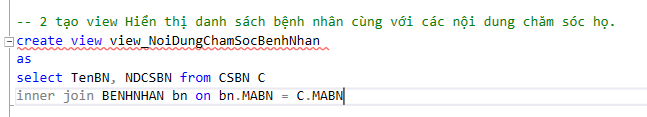
## **4.1. Tạo 7-10 view từ cơ bản đến nâng cao**



Mục đích: Tạo một view để hiển thị tên bác sĩ (TenBS) và tên chuyên khoa (TenCk) mà bác sĩ đó thuộc về.

Giải thích:

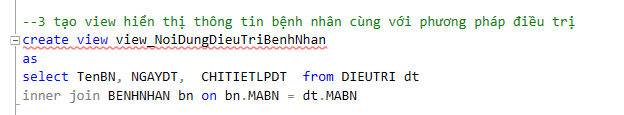
* BACSI và CHUYENKHOA được kết nối thông qua khóa ngoại MACK của cả hai bảng.
* View này giúp dễ dàng theo dõi bác sĩ thuộc chuyên khoa nào mà không cần phải truy vấn dữ liệu từ hai bảng riêng biệt mỗi lần.



Mục đích: Cung cấp thông tin về bệnh nhân và các nội dung chăm sóc bệnh nhân mà họ nhận được.

Giải thích:

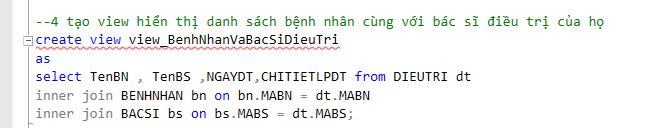
* View này kết nối bảng CSBN (Chăm sóc bệnh nhân) và bảng BENHNHAN để lấy thông tin về bệnh nhân (TenBN) và nội dung chăm sóc (NDCSBN).
* Nó giúp dễ dàng theo dõi các chăm sóc cụ thể mà mỗi bệnh nhân nhận được trong quá trình điều trị.



Mục đích: Hiển thị thông tin về bệnh nhân cùng với các phương pháp điều trị mà bệnh nhân đang thực hiện.

Giải thích:

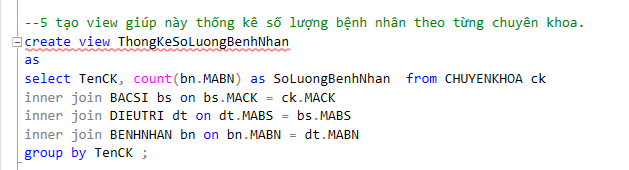
* View này kết nối bảng DIEUTRI (Điều trị) và bảng BENHNHAN để lấy thông tin bệnh nhân (TenBN), ngày điều trị (NGAYDT), và chi tiết phương pháp điều trị (CHITIETLPDT).
* Giúp theo dõi các phương pháp điều trị của bệnh nhân trong suốt quá trình điều trị.



Mục đích**:** Cung cấp thông tin về bệnh nhân và bác sĩ điều trị của họ.

Giải thích:

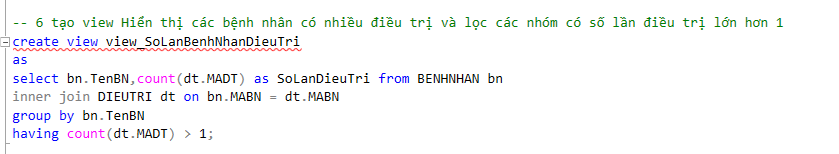
* View này kết nối ba bảng: DIEUTRI (Điều trị), BENHNHAN (Bệnh nhân), và BACSI (Bác sĩ).
* Nó giúp dễ dàng theo dõi bác sĩ điều trị cho bệnh nhân cụ thể vào từng thời điểm, cùng với các chi tiết về phương pháp điều trị.



Mục đích: Thống kê số lượng bệnh nhân theo từng chuyên khoa.

Giải thích:

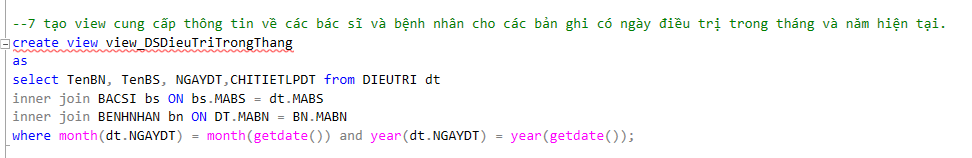
* View này kết nối các bảng CHUYENKHOA, BACSI, DIEUTRI, và BENHNHAN để đếm số lượng bệnh nhân (MABN) cho mỗi chuyên khoa (TenCK).
* Việc nhóm theo TenCK và sử dụng hàm count() giúp thống kê số lượng bệnh nhân trong từng chuyên khoa, hỗ trợ việc quản lý nguồn lực và nhân sự.



Mục đích: Lọc các bệnh nhân có số lần điều trị lớn hơn 1.

Giải thích:

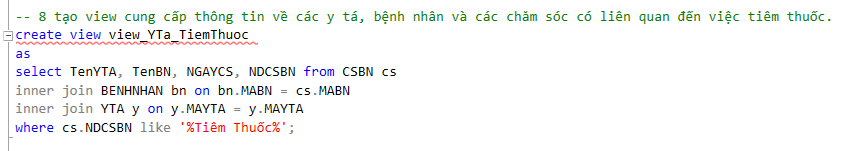
* View này lọc các bệnh nhân có số lần điều trị lớn hơn một lần bằng cách sử dụng hàm count() trong DIEUTRI.
* Lọc với điều kiện having count(dt.MADT) > 1 giúp xác định những bệnh nhân cần điều trị nhiều lần, có thể phục vụ cho việc phân bổ lịch điều trị và tài nguyên.



Mục đích: Hiển thị các bản ghi điều trị của bệnh nhân và bác sĩ trong tháng và năm hiện tại.

Giải thích:

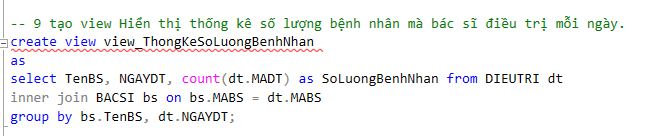
* View này sử dụng hàm month(getdate()) và year(getdate()) để lọc các bản ghi điều trị trong tháng và năm hiện tại.
* Giúp theo dõi các điều trị đang diễn ra trong thời gian thực, phục vụ cho công tác quản lý và điều phối.



Mục đích: Lọc và hiển thị các y tá, bệnh nhân và chăm sóc có liên quan đến việc tiêm thuốc.

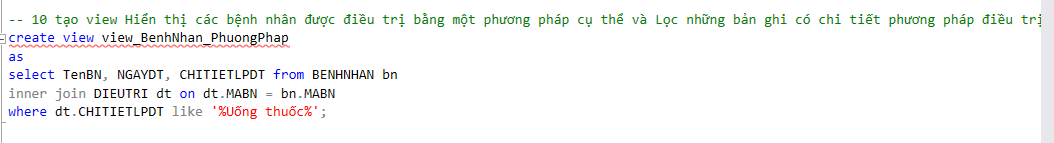
Giải thích:

* View này kết nối ba bảng: CSBN (Chăm sóc bệnh nhân), BENHNHAN (Bệnh nhân), và YTA (Y tá).
* Lọc các chăm sóc có liên quan đến tiêm thuốc bằng cách sử dụng LIKE '%Tiêm Thuốc%' trong trường NDCSBN.
* Giúp theo dõi các y tá thực hiện việc tiêm thuốc cho bệnh nhân.



Mục đích: Thống kê số lượng bệnh nhân mà mỗi bác sĩ điều trị trong một ngày cụ thể.

Giải thích: View này đếm số lượng bệnh nhân mà mỗi bác sĩ điều trị trong từng ngày sử dụng hàm count(), giúp quản lý lịch làm việc và công suất của bác sĩ.



Mục đích: Hiển thị các bệnh nhân được điều trị bằng phương pháp "Uống thuốc".

Giải thích:

* View này lọc các bệnh nhân có phương pháp điều trị chứa cụm từ "Uống thuốc", giúp tìm các bệnh nhân đang sử dụng phương pháp này trong điều trị.
* Tạo điều kiện để phân tích chi tiết phương pháp điều trị, từ đó hỗ trợ các nghiên cứu cải tiến quy trình điều trị.

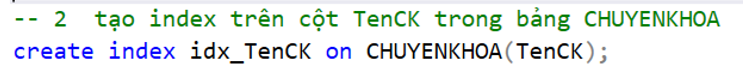
**4.2. Tạo 7-10 index cần thiết cho các bảng**



Mục đích: Tạo chỉ mục trên cột MACK trong bảng CHUYENKHOA.

Tác dụng:

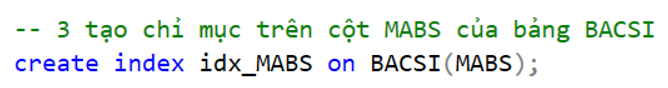
* Cột MACK thường được sử dụng để liên kết với bảng khác (chẳng hạn như bảng BACSI), do đó, việc tạo chỉ mục trên cột này giúp tối ưu hóa các truy vấn sử dụng MACK làm điều kiện lọc hoặc liên kết (join).
* Chỉ mục này sẽ cải thiện tốc độ truy vấn khi bạn tìm kiếm hoặc thực hiện các phép toán trên cột MACK, đặc biệt trong các truy vấn sử dụng JOIN với bảng khác.



Mục đích: Tạo chỉ mục trên cột TenCK của bảng CHUYENKHOA.

Tác dụng:

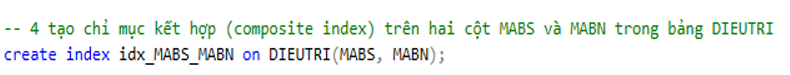
* Cột TenCK có thể được sử dụng trong các truy vấn tìm kiếm các chuyên khoa (chẳng hạn như tìm kiếm theo tên chuyên khoa).
* Việc tạo chỉ mục trên TenCK giúp giảm thời gian tìm kiếm khi có truy vấn lọc hoặc sắp xếp dựa trên cột này.



Mục đích: Tạo chỉ mục trên cột MABS trong bảng BACSI.

Tác dụng:

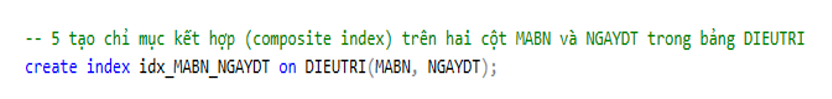
* Cột MABS đại diện cho mã bác sĩ và thường được sử dụng để liên kết với bảng DIEUTRI trong các truy vấn, vì vậy việc tạo chỉ mục trên cột này giúp tăng tốc các truy vấn dựa trên MABS.
* Chỉ mục giúp tối ưu hóa các phép toán tìm kiếm, lọc và JOIN với cột MABS.



Mục đích: Tạo chỉ mục kết hợp trên cột MABS và MABN trong bảng DIEUTRI.

Tác dụng:

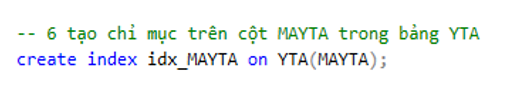
* Chỉ mục kết hợp này sẽ cải thiện hiệu suất các truy vấn sử dụng đồng thời hai cột này, chẳng hạn như các truy vấn JOIN giữa bảng BACSI và BENHNHAN.
* Khi các cột được sử dụng trong cùng một truy vấn, ví dụ như khi cần tìm tất cả các điều trị của bác sĩ đối với một bệnh nhân nhất định, chỉ mục này giúp truy vấn diễn ra nhanh hơn.



Mục đích: Tạo chỉ mục kết hợp trên cột MABN và NGAYDT trong bảng DIEUTRI.

Tác dụng:

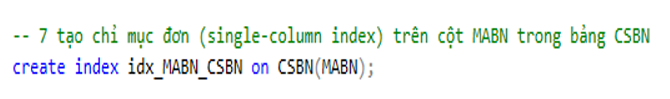
* Chỉ mục này sẽ cải thiện hiệu suất khi truy vấn tìm kiếm điều trị của bệnh nhân (MABN) theo ngày điều trị (NGAYDT).
* Đặc biệt, nếu hệ thống cần thống kê hoặc tìm kiếm điều trị của bệnh nhân trong một khoảng thời gian cụ thể, chỉ mục này sẽ giúp tối ưu hóa quá trình truy vấn.



Mục đích: Tạo chỉ mục trên cột MAYTA trong bảng YTA.

Tác dụng:

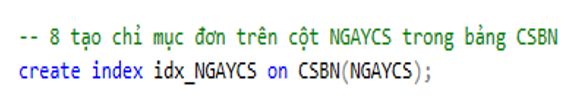
* Cột MAYTA là mã y tá và sẽ được sử dụng trong các truy vấn để liên kết hoặc tìm kiếm thông tin của y tá.
* Việc tạo chỉ mục trên cột này giúp các truy vấn tìm kiếm các y tá trở nên nhanh chóng hơn.



Mục đích: Tạo chỉ mục đơn trên cột MABN trong bảng CSBN.

Tác dụng:

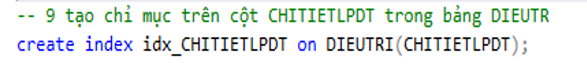
* Cột MABN đại diện cho mã bệnh nhân và được sử dụng trong các truy vấn liên kết giữa bảng CSBN và bảng BENHNHAN.
* Chỉ mục đơn trên MABN giúp tối ưu hóa các truy vấn sử dụng mã bệnh nhân như điều kiện lọc hoặc JOIN



Mục đích: Tạo chỉ mục đơn trên cột NGAYCS trong bảng CSBN.

Tác dụng:

* Cột NGAYCS đại diện cho ngày chăm sóc bệnh nhân, thường được sử dụng trong các truy vấn để tìm kiếm hoặc lọc các chăm sóc trong một ngày cụ thể.
* Chỉ mục này giúp tối ưu hóa các truy vấn tìm kiếm các bản ghi chăm sóc theo ngày.

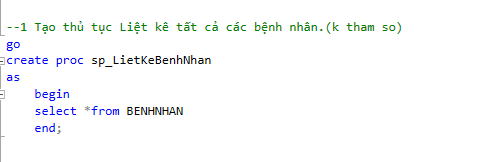


Mục đích: Tạo chỉ mục trên cột CHITIETLPDT trong bảng DIEUTRI.

Tác dụng:

* Cột CHITIETLPDT chứa chi tiết về phương pháp điều trị, và việc tạo chỉ mục trên cột này giúp tối ưu hóa các truy vấn tìm kiếm, lọc hoặc thống kê các bản ghi điều trị dựa trên chi tiết phương pháp điều trị.
* Chỉ mục này rất hữu ích khi cần tìm các bệnh nhân điều trị bằng một phương pháp cụ thể.

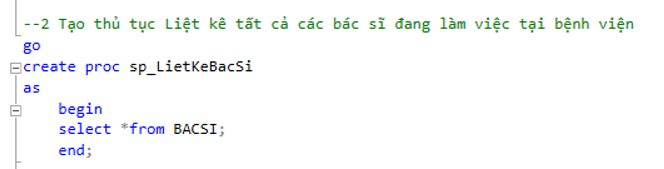
**4.3. Xây dựng 10 Stored Procedure(không tham số, có tham số, có OUTPUT)**



Mục đích: Thủ tục này dùng để liệt kê tất cả các bệnh nhân trong bảng BENHNHAN.

Phân tích:

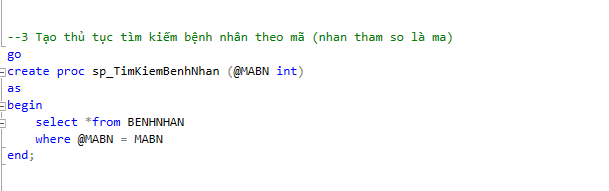
* Đây là thủ tục không có tham số, sẽ trả về toàn bộ dữ liệu từ bảng BENHNHAN.
* Thường được sử dụng để kiểm tra và lấy danh sách bệnh nhân trong cơ sở dữ liệu.



Mục đích: Liệt kê danh sách tất cả các bác sĩ trong bảng BACSI.

Phân tích:

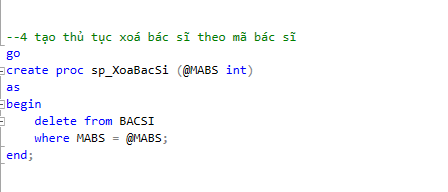
* Thủ tục không có tham số, trả về toàn bộ dữ liệu từ bảng BACSI.
* Sử dụng khi cần lấy thông tin về tất cả các bác sĩ trong hệ thống.



Mục đích: Tìm kiếm thông tin bệnh nhân dựa trên mã bệnh nhân MABN.

Phân tích:

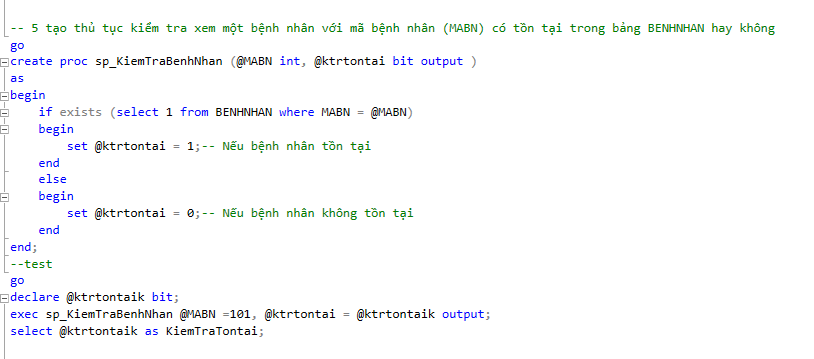
* Thủ tục này có tham số đầu vào là @MABN, dùng để tra cứu một bệnh nhân cụ thể.
* Thích hợp khi bạn cần truy xuất thông tin chi tiết về một bệnh nhân dựa vào mã bệnh nhân.



Mục đích: Xóa bác sĩ khỏi hệ thống dựa trên mã bác sĩ MABS.

Phân tích:

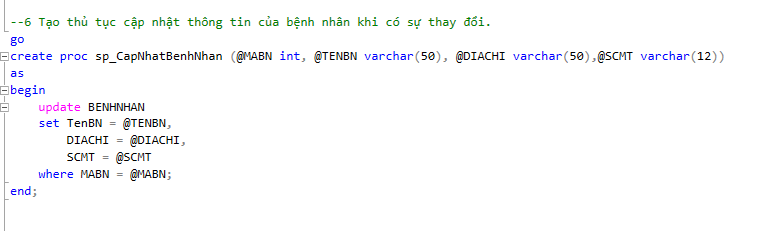
* Thủ tục có tham số @MABS là mã bác sĩ, giúp xóa một bác sĩ cụ thể khỏi bảng BACSI.
* Thường sử dụng khi cần loại bỏ bác sĩ khỏi hệ thống, ví dụ khi bác sĩ nghỉ việc hoặc chuyển công tác.



Mục đích: Kiểm tra xem một bệnh nhân có tồn tại trong hệ thống hay không.

Phân tích:

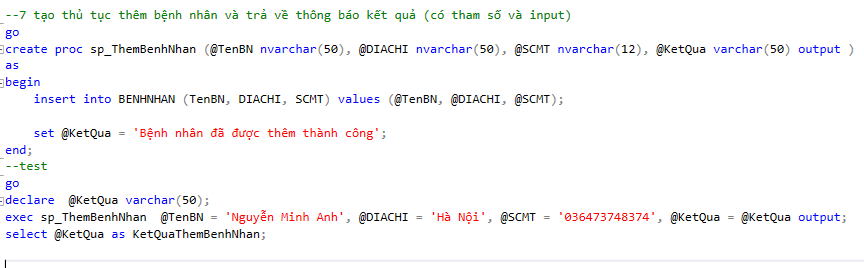
* Thủ tục này có tham số đầu vào @MABN và tham số đầu ra @ktrtontai (kiểu bit).
* Trả về 1 nếu bệnh nhân tồn tại và 0 nếu không tồn tại.
* Thường dùng để xác thực dữ liệu trước khi thực hiện các thao tác khác như sửa, xóa hoặc thêm bệnh nhân.



Mục đích: Cập nhật thông tin của bệnh nhân khi có sự thay đổi.

Phân tích:

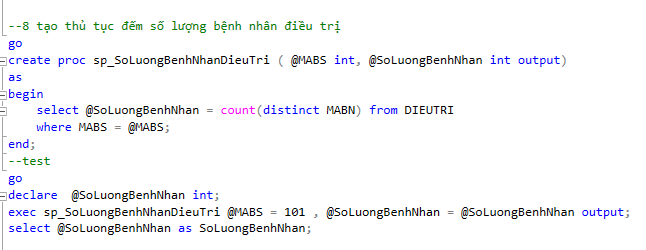
* Thủ tục có tham số đầu vào: mã bệnh nhân @MABN, tên bệnh nhân @TENBN, địa chỉ @DIACHI, và số chứng minh thư @SCMT.
* Cập nhật các thông tin liên quan đến bệnh nhân trong bảng BENHNHAN.



Mục đích: Thêm một bệnh nhân mới và trả về kết quả thông báo.

Phân tích:

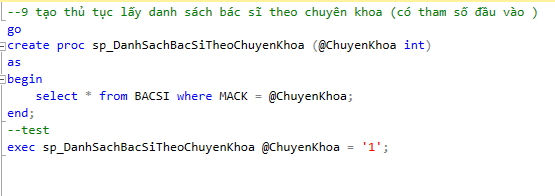
* Thủ tục có tham số đầu vào để nhận thông tin bệnh nhân mới và tham số đầu ra @KetQua để trả về thông báo kết quả.
* Đây là một thủ tục rất quan trọng trong việc thêm dữ liệu vào cơ sở dữ liệu và cung cấp phản hồi cho người dùng.



Mục đích: Đếm số lượng bệnh nhân đã được bác sĩ điều trị.

Phân tích:

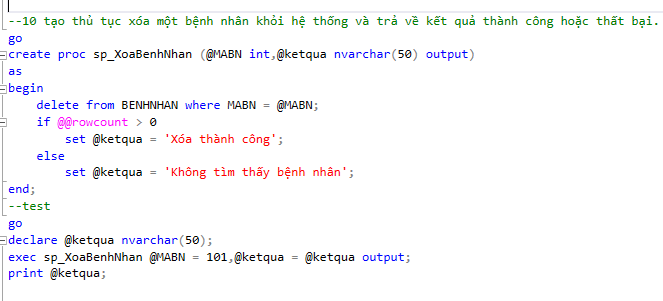
* Thủ tục này nhận mã bác sĩ @MABS và trả về số lượng bệnh nhân điều trị bởi bác sĩ đó.
* Đây là thủ tục hữu ích cho việc thống kê số lượng bệnh nhân được điều trị bởi từng bác sĩ trong bệnh viện.



Mục đích: Liệt kê các bác sĩ theo mã chuyên khoa.

Phân tích:

* Thủ tục này nhận tham số đầu vào @ChuyenKhoa để tra cứu các bác sĩ làm việc trong chuyên khoa cụ thể.
* Đây là thủ tục hữu ích khi bạn muốn tìm kiếm bác sĩ theo lĩnh vực chuyên môn của họ.



Mục đích: Xóa một bệnh nhân khỏi hệ thống và trả về thông báo kết quả.

Phân tích:

* Thủ tục này nhận tham số @MABN là mã bệnh nhân và @ketqua là thông báo kết quả.
* Trả về thông báo "Xóa thành công" nếu bệnh nhân được xóa hoặc "Không tìm thấy bệnh nhân" nếu không có bệnh nhân tương ứng.

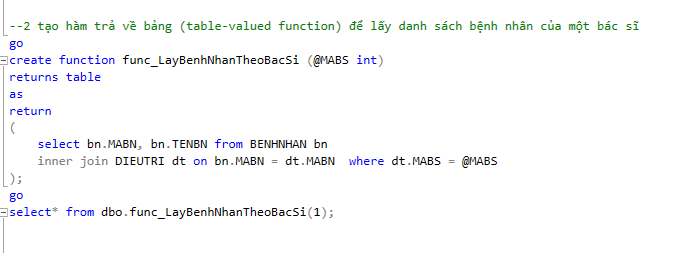
4.4. Tạo 10 function (trả về kiểu vô hướng, bảng, biến bảng)



Mục đích: Kiểm tra xem một bệnh nhân có đang trong danh sách điều trị hay không.

Phân tích:

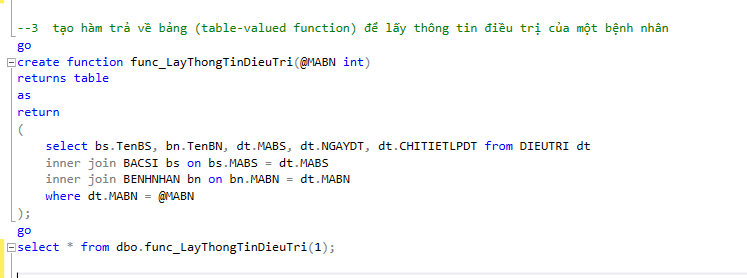
* Hàm này trả về kiểu dữ liệu bit (0 hoặc 1), nếu bệnh nhân có trong bảng DIEUTRI (bảng điều trị) thì trả về 1, ngược lại trả về 0.
* Sử dụng trong các tình huống khi cần xác định xem bệnh nhân đã được điều trị hay chưa.



Mục đích: Trả về danh sách bệnh nhân của một bác sĩ cụ thể.

Phân tích:

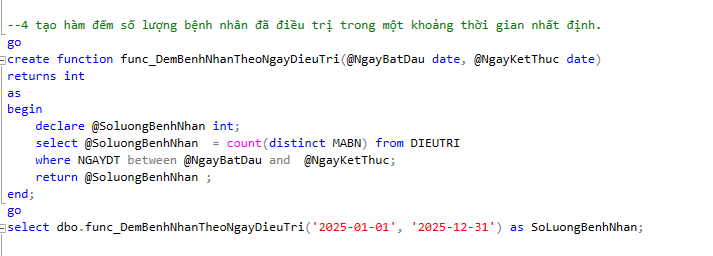
* Hàm trả về một bảng với các bệnh nhân mà bác sĩ có mã MABS đã điều trị, kết hợp bảng BENHNHAN và DIEUTRI.
* Hàm này rất hữu ích khi bạn cần lấy danh sách bệnh nhân của từng bác sĩ.



Mục đích: Trả về thông tin điều trị của một bệnh nhân, bao gồm tên bác sĩ, tên bệnh nhân, ngày điều trị và chi tiết liệu trình.

Phân tích:

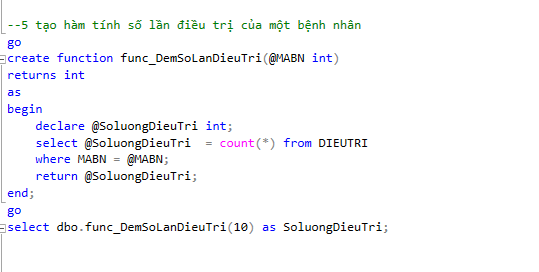
* Hàm này kết hợp bảng DIEUTRI, BACSI và BENHNHAN để trả về các thông tin liên quan đến quá trình điều trị của một bệnh nhân.
* Sử dụng khi cần tra cứu lịch sử điều trị của một bệnh nhân.



Mục đích: Đếm số lượng bệnh nhân điều trị trong một khoảng thời gian xác định.

Phân tích:

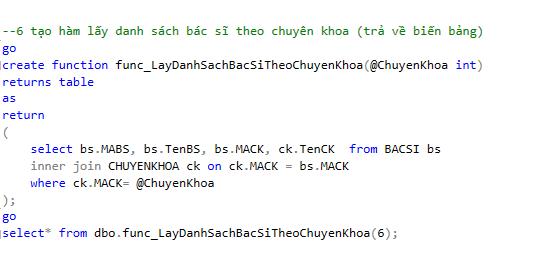
* Hàm trả về số lượng bệnh nhân được điều trị trong khoảng thời gian từ @NgayBatDau đến @NgayKetThuc.
* Thích hợp để thống kê số lượng bệnh nhân điều trị trong một giai đoạn cụ thể.



Mục đích: Tính số lần điều trị của một bệnh nhân.

Phân tích:

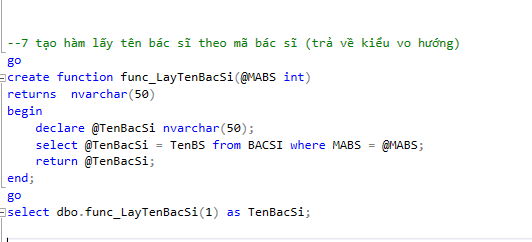
* Hàm này trả về số lần bệnh nhân với mã @MABN đã được điều trị trong hệ thống.
* Dễ dàng sử dụng khi bạn muốn biết bệnh nhân đã điều trị bao nhiêu lần trong hệ thống.



Mục đích: Trả về danh sách bác sĩ thuộc một chuyên khoa cụ thể.

Phân tích:

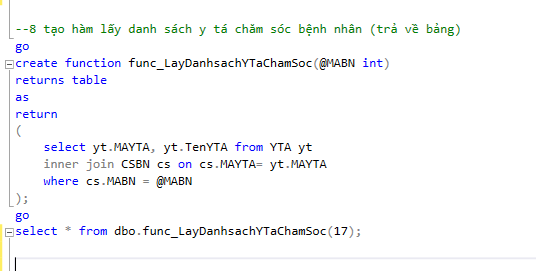
* Hàm này giúp tra cứu bác sĩ làm việc trong một chuyên khoa (@ChuyenKhoa), kết hợp bảng BACSI và CHUYENKHOA.
* Hữu ích cho việc tìm kiếm bác sĩ theo chuyên môn hoặc chuyên khoa.



Mục đích: Lấy tên của bác sĩ dựa trên mã bác sĩ.

Phân tích:

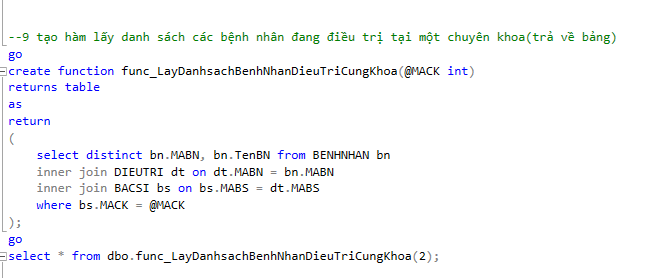
* Hàm này trả về tên của bác sĩ có mã @MABS.
* Dễ dàng sử dụng để lấy tên bác sĩ khi chỉ có mã bác sĩ.



Mục đích: Trả về danh sách y tá chăm sóc bệnh nhân cụ thể.

Phân tích:

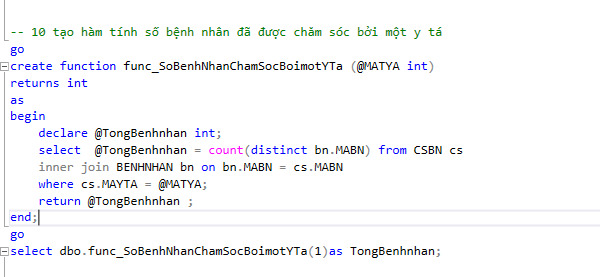
* Hàm này kết hợp bảng YTA và CSBN để lấy danh sách y tá chăm sóc bệnh nhân có mã @MABN.
* Thích hợp để xác định các y tá liên quan đến một bệnh nhân.



Mục đích: Lấy danh sách bệnh nhân đang điều trị tại một chuyên khoa cụ thể.

Phân tích:

* Hàm này trả về danh sách bệnh nhân đang điều trị tại chuyên khoa có mã @MACK, thông qua bảng DIEUTRI và BACSI.
* Rất hữu ích cho việc theo dõi bệnh nhân của một chuyên khoa.

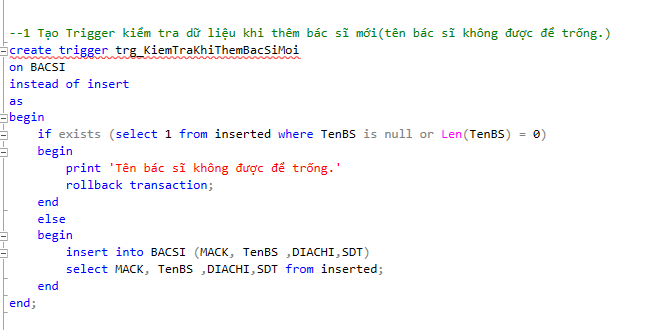


Mục đích: Đếm số bệnh nhân đã được chăm sóc bởi một y tá cụ thể.

Phân tích:

* Hàm trả về số lượng bệnh nhân đã được chăm sóc bởi y tá có mã @MATYA, thông qua bảng CSBN.
* Thủ tục này hữu ích khi cần thống kê số bệnh nhân mà một y tá đã chăm sóc

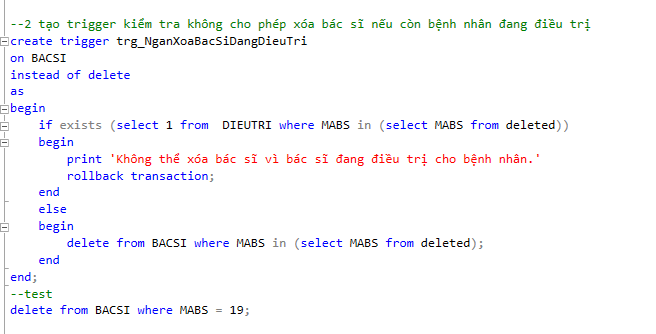
**4.5. Tạo 7- 10 trigger để kiểm soát dữ liệu**



Mục đích: Kiểm tra xem khi thêm bác sĩ mới vào bảng BACSI, tên bác sĩ (TenBS) có bị trống hay không. Nếu tên bác sĩ trống, giao dịch bị rollback.

Phân tích:

* Trigger này sử dụng INSTEAD OF INSERT để thay thế thao tác INSERT mặc định và thực hiện kiểm tra tên bác sĩ trước khi thêm vào bảng.
* Nếu tên bác sĩ bị trống hoặc có độ dài bằng 0, trigger sẽ in ra thông báo và hủy giao dịch.
* Điều này đảm bảo rằng chỉ những bác sĩ có tên hợp lệ mới có thể được thêm vào bảng.



Mục đích: Ngăn không cho phép xóa bác sĩ nếu bác sĩ đó vẫn còn bệnh nhân đang được điều trị.

Phân tích:

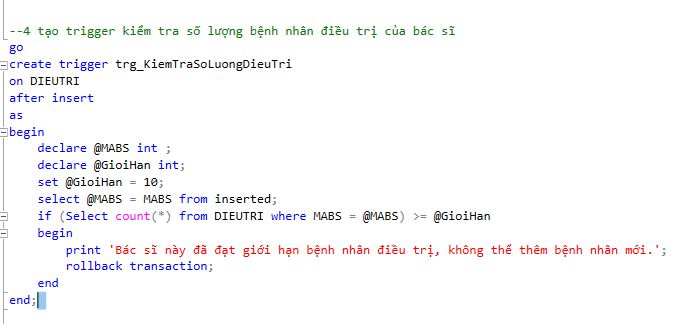
* Trigger này sử dụng INSTEAD OF DELETE để thay thế thao tác xóa và thực hiện kiểm tra trước khi xóa bác sĩ.
* Nếu bác sĩ trong bảng BACSI đang điều trị cho bệnh nhân (kiểm tra trong bảng DIEUTRI), trigger sẽ không cho phép xóa và hủy giao dịch.
* Điều này giúp duy trì tính toàn vẹn của dữ liệu, không cho phép xóa bác sĩ đang có mối quan hệ với bệnh nhân.



Mục đích: Ngăn không cho phép cập nhật tên bác sĩ nếu bác sĩ đó đã có bệnh nhân.

Phân tích:

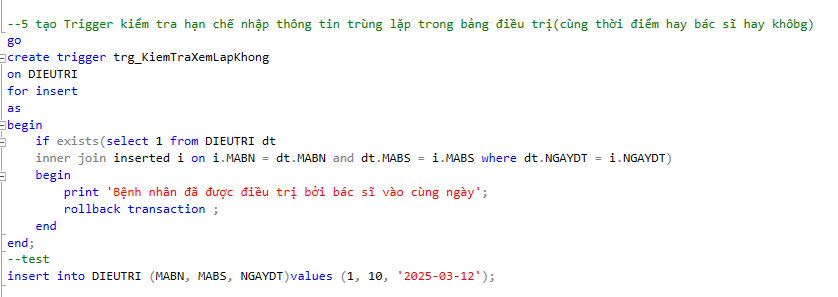
* Trigger này sử dụng INSTEAD OF UPDATE để thay thế thao tác cập nhật.
* Nếu bác sĩ trong bảng BACSI đã điều trị cho bệnh nhân (dựa vào bảng DIEUTRI), trigger sẽ không cho phép cập nhật tên bác sĩ và hủy giao dịch.
* Điều này đảm bảo rằng các bác sĩ có bệnh nhân đang điều trị không thể thay đổi tên, tránh gây rối loạn trong dữ liệu



Mục đích: Kiểm tra xem bác sĩ đã đạt giới hạn số lượng bệnh nhân điều trị hay chưa trước khi thêm bệnh nhân mới vào bảng DIEUTRI.

Phân tích:

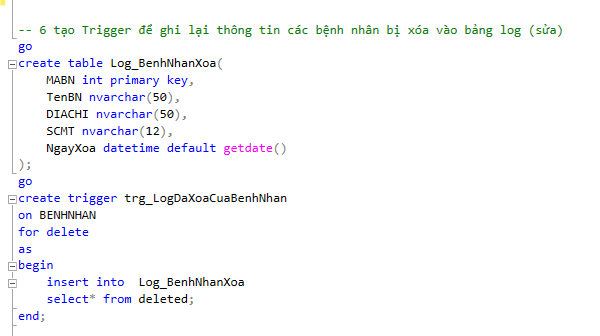
* Trigger này sử dụng AFTER INSERT để thực thi sau khi thêm dữ liệu vào bảng DIEUTRI.
* Nếu bác sĩ đã điều trị đủ số lượng bệnh nhân (giới hạn là 10 bệnh nhân), trigger sẽ hủy giao dịch và không cho phép thêm bệnh nhân mới.
* Điều này giúp quản lý số lượng bệnh nhân mỗi bác sĩ có thể điều trị, tránh quá tải.



Mục đích: Ngăn không cho phép bệnh nhân được điều trị bởi cùng một bác sĩ vào cùng một ngày.

Phân tích:

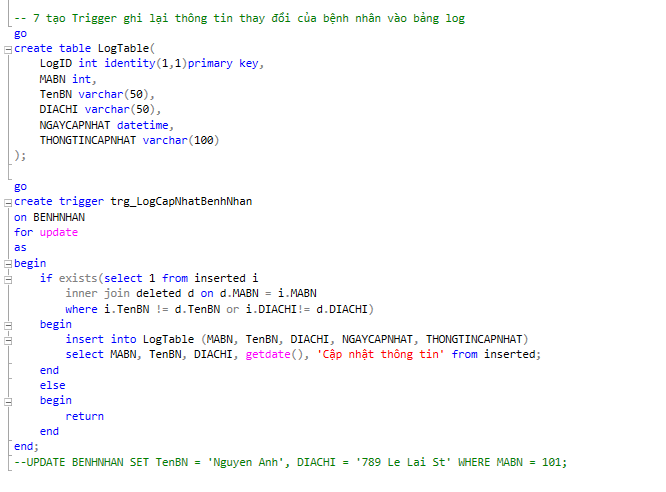
* Trigger này sử dụng FOR INSERT để kiểm tra sự trùng lặp trong bảng DIEUTRI trước khi thêm một bản ghi mới.
* Nếu bệnh nhân và bác sĩ đã có lịch điều trị vào cùng ngày, trigger sẽ hủy giao dịch và không cho phép thêm bản ghi mới.
* Điều này đảm bảo rằng không có tình trạng trùng lặp lịch điều trị của bác sĩ.



Mục đích: Ghi lại thông tin của bệnh nhân bị xóa vào bảng log Log\_BenhNhanXoa.

Phân tích:

* Trigger này sử dụng FOR DELETE để thực thi khi một bản ghi trong bảng BENHNHAN bị xóa.
* Thông tin của bệnh nhân bị xóa sẽ được lưu vào bảng Log\_BenhNhanXoa để đảm bảo dữ liệu lịch sử.
* Điều này giúp bạn theo dõi thông tin các bệnh nhân đã bị xóa và phục vụ cho các yêu cầu báo cáo hoặc khôi phục sau này.



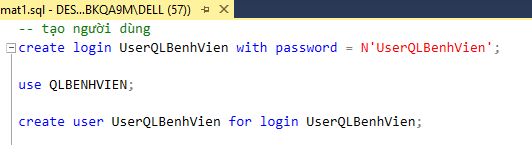
Mục đích: Ghi lại các thay đổi về thông tin bệnh nhân vào bảng log LogTable.

Phân tích:

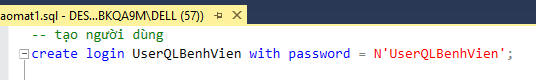
* Trigger này sử dụng FOR UPDATE để theo dõi các thay đổi trong bảng BENHNHAN.
* Nếu có sự thay đổi trong các trường như TenBN hoặc DIACHI, trigger sẽ ghi lại thông tin vào bảng LogTable.
* Điều này giúp duy trì lịch sử thay đổi của bệnh nhân, phục vụ cho các nhu cầu quản lý hoặc audit.

**CHƯƠNG 5 : BẢO MẬT VÀ QUẢN TRỊ**

**5.1. Tạo người dùng**

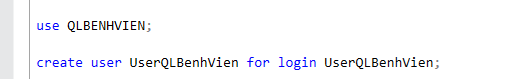


Tạo login: Lệnh create login dùng để tạo một login (tài khoản đăng nhập) trong SQL Server. Login là đối tượng quản lý quyền truy cập vào SQL Server, nhưng chưa có quyền truy cập vào một cơ sở dữ liệu cụ thể.



Trong lệnh này, tạo login UserQLBenhVien với mật khẩu là UserQLBenhVien. Mật khẩu này có thể được thay đổi sau này. Chú ý rằng mật khẩu được định nghĩa dưới dạng N'...' để đảm bảo rằng các ký tự đặc biệt hoặc ngôn ngữ không bị sai lệch.

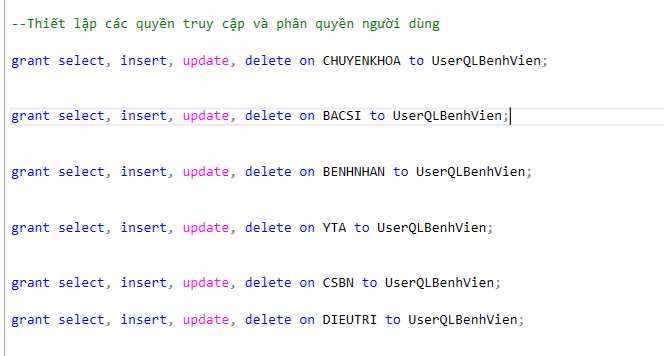
* Chuyển vào cơ sở dữ liệu: Lệnh use QLBENHVIEN; chuyển tới cơ sở dữ liệu QLBENHVIEN, nơi người dùng sẽ làm việc. Lệnh này thiết lập ngữ cảnh của cơ sở dữ liệu mà các thao tác tiếp theo sẽ được thực hiện.
* Tạo user trong cơ sở dữ liệu: Tiếp theo, lệnh create user tạo một user cụ thể cho login đã tạo trước đó. Trong trường hợp này, người dùng UserQLBenhVien sẽ được liên kết với login UserQLBenhVien.



Việc này đảm bảo rằng người dùng có thể đăng nhập vào cơ sở dữ liệu QLBENHVIEN và thực hiện các thao tác trong cơ sở dữ liệu này.

**5.2. Thiết lập các quyền truy vập và phân quyền người dùng**

Phần này liên quan đến việc cấp quyền truy cập cho người dùng đã tạo. Trong SQL Server, bạn có thể phân quyền chi tiết cho người dùng đối với các bảng, views, hoặc các đối tượng khác trong cơ sở dữ liệu. Quyền truy cập có thể bao gồm các hành động như SELECT, INSERT, UPDATE, DELETE, v.v.

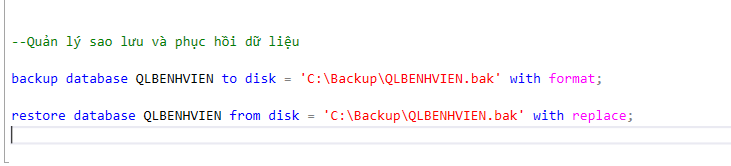


* Phân quyền cho người dùng: Các lệnh grant được sử dụng để cấp quyền cho người dùng UserQLBenhVien đối với các bảng trong cơ sở dữ liệu. Các quyền bao gồm:
* SELECT: Quyền đọc dữ liệu từ bảng.
* INSERT: Quyền thêm mới dữ liệu vào bảng.
* UPDATE: Quyền cập nhật dữ liệu trong bảng.
* DELETE: Quyền xóa dữ liệu từ bảng.

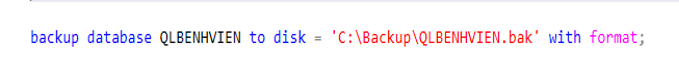
Ví dụ, lệnh grant select, insert, update, delete on CHUYENKHOA to UserQLBenhVien; cấp quyền đầy đủ (SELECT, INSERT, UPDATE, DELETE) cho người dùng UserQLBenhVien trên bảng CHUYENKHOA. Các lệnh tương tự được thực hiện cho các bảng khác như BACSI, BENHNHAN, YTA, CSBN, và DIEUTRI.

* Lợi ích của việc phân quyền chi tiết: Việc phân quyền cho từng bảng hoặc đối tượng giúp bảo vệ dữ liệu và kiểm soát việc người dùng có thể thực hiện những thao tác nào. Điều này rất quan trọng trong các ứng dụng thực tế, nơi dữ liệu nhạy cảm cần được bảo vệ.

## **5.3. Quản lí sao lưu và phục hồi dữ liệu**

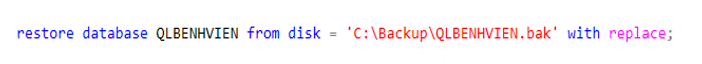


**Sao lưu cơ sở dữ liệu:** Lệnh backup database được sử dụng để sao lưu toàn bộ cơ sở dữ liệu QLBENHVIEN vào một file cụ thể. Trong trường hợp này, file sao lưu sẽ được lưu tại C:\Backup\QLBENHVIEN.bak



Tùy chọn with format đảm bảo rằng file sao lưu sẽ được tạo mới mỗi lần chạy lệnh sao lưu, giúp tránh việc ghi đè dữ liệu cũ.

Phục hồi cơ sở dữ liệu: Lệnh restore database phục hồi cơ sở dữ liệu từ file sao lưu. Trong trường hợp này, cơ sở dữ liệu QLBENHVIEN sẽ được phục hồi từ file sao lưu QLBENHVIEN.bak.



Tùy chọn with replace cho phép phục hồi cơ sở dữ liệu và thay thế cơ sở dữ liệu hiện tại, điều này rất hữu ích khi bạn muốn khôi phục một bản sao lưu vào một cơ sở dữ liệu đã có sẵn.

Lợi ích của sao lưu và phục hồi: Sao lưu dữ liệu định kỳ giúp đảm bảo rằng bạn có thể khôi phục lại dữ liệu trong trường hợp có sự cố, chẳng hạn như lỗi phần cứng, mất dữ liệu, hoặc sự cố phần mềm. Điều này giúp bảo vệ dữ liệu quan trọng và đảm bảo tính liên tục của các dịch vụ.

**CHƯƠNG 6: KẾT LUẬN**

6.1. Nhận xét về quá trình thực hiện bài tập lớn

Mục tiêu: Phân tích và nhận xét về quá trình thực hiện dự án xây dựng hệ thống quản lý bệnh viện, từ việc thiết kế cơ sở dữ liệu đến triển khai và kiểm thử hệ thống.

Phân tích: Lập kế hoạch và thiết kế cơ sở dữ liệu: Quá trình thực hiện bài tập lớn bắt đầu từ việc hiểu rõ yêu cầu của hệ thống quản lý bệnh viện, bao gồm các tính năng cần thiết như quản lý bác sĩ, bệnh nhân, các khoa, điều trị, và các thông tin liên quan đến chẩn đoán và điều trị. Sau đó, cơ sở dữ liệu được thiết kế để đáp ứng các yêu cầu này, bao gồm việc tạo ra các bảng và mối quan hệ giữa chúng như bảng BacSi, BenhNhan, DieuTri, ChuyenKhoa, v.v.

Xây dựng các chức năng và quyền truy cập: Sau khi thiết kế cơ sở dữ liệu, việc phân quyền cho người dùng và đảm bảo bảo mật thông qua việc cấp quyền truy cập hợp lý cho từng đối tượng là một phần quan trọng. Quá trình này giúp hệ thống đảm bảo tính bảo mật và tính toàn vẹn của dữ liệu trong khi cung cấp quyền truy cập phù hợp cho từng nhóm người dùng.

Kiểm thử và triển khai: Sau khi xây dựng cơ sở dữ liệu và các chức năng, hệ thống sẽ được kiểm thử để đảm bảo các thao tác như thêm, sửa, xóa, và truy vấn dữ liệu hoạt động chính xác. Việc kiểm thử cũng bao gồm kiểm tra các chức năng sao lưu và phục hồi dữ liệu để đảm bảo hệ thống có thể duy trì và bảo vệ dữ liệu trong trường hợp có sự cố.

Nhận xét: Quá trình thực hiện bài tập lớn có thể gặp phải một số thử thách trong việc quản lý và thiết kế các mối quan hệ giữa các bảng, đặc biệt là khi có nhiều bảng liên quan đến nhau (như bảng bác sĩ và bệnh nhân). Tuy nhiên, qua quá trình thực hiện, người làm có thể rút ra kinh nghiệm về cách xây dựng một hệ thống cơ sở dữ liệu logic và dễ duy trì. Các bước phân quyền và kiểm thử hệ thống là rất quan trọng để đảm bảo tính bảo mật và hiệu quả hoạt động của hệ thống.

6.2. Đánh giá ưu và nhược điểm của hệ thống

Mục tiêu: Đánh giá tổng thể hệ thống quản lý bệnh viện, làm rõ những ưu điểm và nhược điểm của hệ thống trong việc đáp ứng nhu cầu quản lý bệnh viện.

Phân tích:

* Ưu điểm của hệ thống:
* Tính bảo mật cao: Hệ thống quản lý bệnh viện cho phép phân quyền rõ ràng cho từng nhóm người dùng (bác sĩ, y tá, nhân viên quản trị). Điều này giúp bảo vệ dữ liệu quan trọng của bệnh nhân và đảm bảo rằng mỗi người chỉ có quyền truy cập vào thông tin cần thiết.
* Quản lý dữ liệu hiệu quả: Hệ thống cơ sở dữ liệu có thể giúp tổ chức và lưu trữ thông tin bệnh viện một cách khoa học, dễ dàng truy xuất và cập nhật. Các thông tin về bệnh nhân, bác sĩ, các khoa, và các lịch sử điều trị được quản lý tập trung.
* Sao lưu và phục hồi dữ liệu: Hệ thống đã được tích hợp các tính năng sao lưu và phục hồi dữ liệu, giúp bảo vệ dữ liệu quan trọng trong trường hợp có sự cố xảy ra.
* Tính linh hoạt: Hệ thống có thể dễ dàng mở rộng và điều chỉnh khi có nhu cầu thay đổi, thêm tính năng mới hoặc điều chỉnh các quyền truy cập.
* Nhược điểm của hệ thống:
* Cần kỹ năng chuyên môn để vận hành: Hệ thống yêu cầu người sử dụng có một mức độ hiểu biết về cơ sở dữ liệu và các chức năng của hệ thống. Những người không có kinh nghiệm có thể gặp khó khăn trong việc sử dụng hệ thống.
* Khả năng bảo trì và nâng cấp: Với việc sử dụng hệ quản trị cơ sở dữ liệu SQL Server, việc bảo trì và nâng cấp hệ thống yêu cầu các chuyên gia có kiến thức vững về cơ sở dữ liệu và các công cụ hỗ trợ.
* Chi phí triển khai: Việc triển khai và duy trì hệ thống có thể đòi hỏi các chi phí đầu tư ban đầu và chi phí duy trì trong dài hạn, đặc biệt nếu sử dụng các công cụ hoặc phần mềm trả phí.

6.3. Đề xuất các hướng phát triển tiếp theo

Mục tiêu: Đưa ra các hướng phát triển trong tương lai để cải thiện và mở rộng hệ thống quản lý bệnh viện, đáp ứng nhu cầu ngày càng cao của ngành y tế.

Phân tích:

* Tích hợp công nghệ AI trong quản lý bệnh viện: Việc tích hợp các công nghệ trí tuệ nhân tạo (AI) vào hệ thống quản lý bệnh viện có thể giúp dự đoán và phân tích các xu hướng sức khỏe của bệnh nhân, tối ưu hóa lịch trình điều trị và giảm thiểu lỗi trong chẩn đoán. Các mô hình học máy có thể giúp phân tích các dữ liệu lớn, giúp bác sĩ đưa ra quyết định chính xác hơn.
* Mở rộng hệ thống để hỗ trợ đa cơ sở: Hệ thống hiện tại chỉ phục vụ cho một cơ sở y tế duy nhất. Trong tương lai, có thể mở rộng hệ thống để hỗ trợ nhiều cơ sở bệnh viện, giúp quản lý thông tin giữa các bệnh viện và các cơ sở y tế con cùng một hệ thống dữ liệu. Điều này giúp đơn giản hóa công tác quản lý và cải thiện dịch vụ chăm sóc sức khỏe.
* Phát triển ứng dụng di động: Để nâng cao tính tiện lợi, có thể phát triển một ứng dụng di động cho bệnh nhân và bác sĩ. Ứng dụng này có thể giúp bệnh nhân đặt lịch hẹn, xem thông tin về lịch sử điều trị, kết quả xét nghiệm, và thậm chí giao tiếp trực tiếp với bác sĩ thông qua tin nhắn hoặc video call.
* Bảo mật và tuân thủ quy định về bảo vệ dữ liệu: Với sự gia tăng của các quy định bảo vệ dữ liệu, đặc biệt là dữ liệu y tế, hệ thống cần phải tuân thủ nghiêm ngặt các quy định về bảo mật và quyền riêng tư của người dùng. Cần tích hợp các phương thức bảo mật tiên tiến như mã hóa dữ liệu, xác thực đa yếu tố và giám sát an ninh thường xuyên.
* Ứng dụng công nghệ blockchain: Công nghệ blockchain có thể được tích hợp để đảm bảo tính minh bạch và bảo mật cho hồ sơ y tế. Dữ liệu bệnh nhân có thể được ghi vào blockchain để đảm bảo rằng dữ liệu không thể bị thay đổi hoặc xóa một cách trái phép.

**DANH MỤC TÀI LIỆU THAM KHẢO**

1. Nguyễn Hồng Sơn (2007), *Giáo trình hệ thống Mạng máy tính CCNA* (Semester 1), NXB Lao động xã hội.
2. Phạm Quốc Hùng (2017), *Đề cương bài giảng Mạng máy tính*, Đại học SPKT Hưng Yên.