TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC THÀNH PHỐ HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN

HỌC PHẦN HỌC KỲ II
NĂM HỌC 2022-2023

CÁC HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

ỨNG DỤNG SQL SEVER HỆ THỐNG BÁN HÀNG CỦA NHÀ HÀNG

GV hướng dẫn: ThS. Trần Thị Thanh Thảo

Sinh viên thực hiện:
20DH110921 – Vũ Đức Dũng
20DH111274 – Nguyễn Thị Hoàng Thơ
20DH111302 – Hoàng Phú

Thành phố Hồ Chí Minh, tháng 5/2023

MỤC LỤC

DANH MỤC BẢNG

GIỚI THIỆU

Ý nghĩa của đề tài

Với sự phát triển vượt bậc của công nghệ số 4.0 trong các ngành nói chung và ngành thực phẩm nói riêng, cho thấy sự cạnh tranh trong ngành ngày càng gay gắt. Các nhà hàng phần lớn đều sử dụng những hệ thống khác nhau để quản lý thông tin của nhà hàng một cách hiệu quả.

Trên thị trường hiện nay sẽ có nhiều hệ thống quản lý dữ liệu chung (bản miễn phí lẫn bản trả phí), thậm chí cụ thể là hệ thống quản lý dữ liệu của nhà hàng /quán ăn. Tuy nhiên, do các hệ thống này được tạo theo một mẫu chung, có thể sử dụng đại trà nên chúng có thể sẽ không đáp ứng hết nhu cầu đặc biệt của một nhà hàng. Hơn nữa, việc sử dụng một hệ thống quản lý dữ liệu được xây dựng riêng cho nhà hàng sẽ giúp nhà hàng nắm quyền kiểm soát hoàn toàn các dữ liệu của mình.

Tóm lại, một hệ thống quản lý nhà hàng, quán ăn là một hướng đi phù hợp với xu hướng, và cũng tạo cơ hội tốt cho nhóm chúng em được thực hành các kiến thức và kỹ năng tương ứng. Do đó, chúng em chọn “Xây dựng cơ sở dữ liệu hệ thống quản lý nhà hàng Kitchen of Men” làm đề tài của mình.

Mục tiêu của đề tài

- Phân tích các yêu cầu của doanh nghiệp (Nhà hàng Kitchen Of Men).
- Thiết kế dữ liệu phù hợp với yêu cầu của doanh nghiệp.
- Thiết kế xử lý phù hợp với yêu cầu của doanh nghiệp.
- Xây dựng cơ sở dữ liệu và trang web dựa trên các thiết kế.

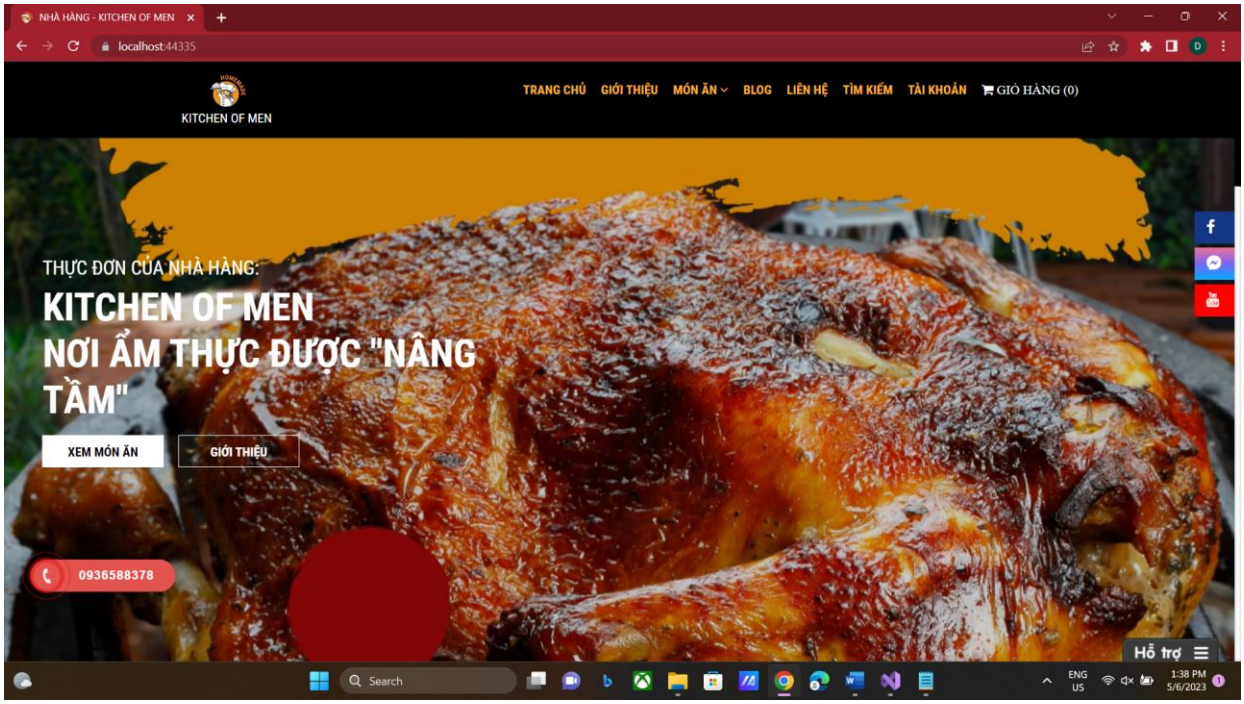
Chương 1 KHẢO SÁT HIỆN TRẠNG VÀ XÁC ĐỊNH YÊU CẦU

1.1 Khảo sát nghiệp vụ

1.1.1 Đặc điểm của nhà hàng

Đặc điểm của nhà hàng: Xác định loại hình nhà hàng và mô hình kinh doanh, bao gồm quy mô, kiểu dáng, phong cách phục vụ, và mục tiêu khách hàng:

- Nhà hàng loại hình phục vụ tại chỗ và online trực tuyến đặt hàng qua website.
- Mô hình kinh doanh chuỗi nhà hàng có nhiều chi nhánh.
- Quy mô phục vụ: Tối đa 50 khách trong 1 thời điểm.
- Phong cách phục vụ: Hiện đại và đơn giản.
- Mục tiêu khách hàng: Mọi lứa tuổi, mọi giới tính



Hình 1 hình ảnh trang website của nhà hàng

1.1.2 Quản lý thực đơn

Nhà hàng sẽ cho đầu bếp lên thực đơn và menu cho nhà hàng, sau đó sẽ bán và phục vụ theo menu đầu bếp đã chuẩn bị.

1.1.3 Quy trình đặt hàng và quản lý kho

1. Kiểm tra hàng hóa, cập nhật số liệu kho tồn nguyên liệu.
2. Nguyên liệu nào sắp hết sẽ lên đơn đặt từ nhà cung cấp.
3. Nhận hàng từ nhà cung cấp, kiểm tra chất lượng và nhập kho.

1.1.4 Quản lý nhân sự

1. Tuyển dụng nhân viên và đào tạo.
2. Sắp xếp lịch làm phù hợp để nhà hàng luôn đủ nhân viên để phục vụ khách.
3. Trưởng quản lý nhân sự sẽ giải quyết các vấn để với nhân viên như lương thưởng hoặc nghỉ phép.

1.1.5 Quản lý hoạt động nhà hàng

1.1.5.1 Quy trình chuẩn bị món ăn

- Nhà hàng Kitchen of men đã khảo sát quy trình chuẩn bị món ăn để đảm bảo sự hiệu quả và chất lượng trong quá trình nấu ăn. Họ đã phân tích các bước chuẩn bị từ việc kiểm tra nguyên liệu, rửa sạch, cắt và chế biến món ăn. Họ cũng xác định các thời gian nấu nướng và công thức phù hợp cho mỗi món để đảm bảo nhất quán và chất lượng cao.

1.1.5.2 Quy trình phục vụ khách hàng

- Nhà hàng Kitchen of men đã tiến hành khảo sát quy trình phục vụ khách hàng để cải thiện trải nghiệm của khách hàng. Họ đã đặt câu hỏi về thời gian chờ đợi, sự chuyên nghiệp của nhân viên phục vụ, sự tận tâm và thân thiện. Dựa trên kết quả khảo sát, nhà hàng đã cải thiện quy trình ghi nhận đơn hàng, thời gian phục vụ và đào tạo nhân viên về kỹ năng giao tiếp và dịch vụ khách hàng.

1.1.6 Yêu cầu chức năng

- Hệ thống quản lý nhà hàng cần đáp ứng một số yêu cầu chức năng để hỗ trợ hoạt động của nhà hàng. Dưới đây là một số yêu cầu chức năng quan trọng:

1. Quản lý thực đơn: Hệ thống cần cho phép quản lý thực đơn, bao gồm việc thêm, sửa đổi và xóa món ăn, đặc điểm, giá cả và thông tin liên quan. Nó nên có khả năng cập nhật thực đơn theo thời gian, quản lý món ăn theo danh mục và hiển thị thông tin chi tiết về mỗi món.

2. Quản lý đặt hàng và kho: Hệ thống cần hỗ trợ quản lý đặt hàng nguyên liệu từ các nhà cung cấp, theo dõi lượng tồn kho, và tạo ra các phiếu nhập/xuất kho. Nó cũng nên cung cấp thông báo khi lượng tồn kho dưới mức tối thiểu và tính toán tồn kho cũng như tiêu thụ nguyên liệu theo mức độ sử dụng.

3. Quản lý nhân viên: Hệ thống cần hỗ trợ quản lý thông tin nhân viên, bao gồm thông tin cá nhân, lịch làm việc, công việc, và lương bổng. Nó nên có khả năng tạo và xem lịch làm việc, quản lý đăng ký và chấm công, và tính toán lương và phụ cấp.

4. Quản lý hóa đơn và thanh toán: Hệ thống cần cho phép tạo và quản lý hóa đơn cho khách hàng, bao gồm các món ăn đã đặt, số lượng, giá cả, và thuế. Nó nên hỗ trợ nhiều phương thức thanh toán như tiền mặt, thẻ, và ví điện tử, và cung cấp tính năng tính toán tự động và in hóa đơn.

5. Báo cáo và phân tích: Hệ thống cần có khả năng tạo ra các báo cáo về doanh thu, lợi nhuận, tỷ suất hủy đặt bàn

1.1.7 Yêu cầu phi chức năng

1.1.7.1 Bảo mật

- Đảm bảo tính bảo mật của hệ thống với người ngoài hệ thống
- Giới hạn các thông tin người dùng được truy cập theo vai trò của người dùng
- Giới hạn các giao tác người dùng có thể thực hiện theo vai trò của người dùng

1.1.7.2 Hiệu suất

- Thời gian phản hồi, thời gian xử lý và thời gian trả kết quả nhanh

1.1.7.3 Tiện dụng

- Cung cấp đầy đủ các chức năng mà người dùng yêu cầu
- Dễ học cách sử dụng
- Hạn chế lỗi nhập liệu

1.1.7.4 An toàn

- Hoạt động ổn định, chính xác
- Có thể khôi phục hệ thống, khôi phục dữ liệu sau các sự cố

Chương 2 CƠ SỞ LÝ THUYẾT

2.1 Ngôn Ngữ Transact SQL

Transact-SQL (T-SQL) là một ngôn ngữ lập trình mở rộng của SQL (Structured Query Language), được sử dụng để tương tác và quản lý cơ sở dữ liệu trong hệ thống quản lý cơ sở dữ liệu quan hệ (RDBMS) của Microsoft, chẳng hạn như SQL Server.

Dưới đây là một số điểm nổi bật về ngôn ngữ Transact-SQL:

1. Làm việc với cơ sở dữ liệu: T-SQL cho phép bạn tạo, sửa đổi và xóa cơ sở dữ liệu, bảng, chế độ xem, thủ tục lưu trữ, hàm, chỉ mục và nhiều đối tượng khác trong cơ sở dữ liệu.

2. Truy vấn dữ liệu: Bằng cách sử dụng câu lệnh SELECT, bạn có thể truy vấn dữ liệu từ bảng và chọn các cột, áp dụng điều kiện, sắp xếp, nhóm và tính toán trên dữ liệu.

3. Xử lý dữ liệu: T-SQL cung cấp các câu lệnh để thêm, cập nhật và xóa dữ liệu từ bảng. Bạn có thể sử dụng các biểu thức và hàm tính toán để thực hiện các phép tính và biến đổi dữ liệu.

4. Điều khiển luồng chương trình: Bằng cách sử dụng câu lệnh điều kiện IF, LOOP và WHILE, bạn có thể thực hiện các lệnh SQL dựa trên các điều kiện hoặc lặp lại các lệnh cho đến khi một điều kiện được thỏa mãn.

5. Gọi thủ tục lưu trữ và hàm: T-SQL cho phép bạn tạo và gọi các thủ tục lưu trữ và hàm để thực hiện các tác vụ phức tạp. Điều này giúp tái sử dụng mã và tăng tính module của ứng dụng.

6. Xử lý lỗi: T-SQL cung cấp cơ chế để xử lý lỗi và ngoại lệ trong quá trình thực thi mã. Bạn có thể sử dụng câu lệnh TRY...CATCH để bắt và xử lý các ngoại lệ xảy ra trong quá trình thực hiện.

Transact-SQL cung cấp một cú pháp mạnh mẽ và nhiều tính năng để làm việc với cơ sở dữ liệu quan hệ. Nó là một phần quan trọng trong việc phát triển ứng dụng dựa trên SQL Server và được sử dụng rộng rãi trong Trong lĩnh vực phát triển ứng dụng và quản lý cơ sở dữ liệu, Transact-SQL được sử dụng rộng rãi trong hệ thống quản lý cơ sở dữ liệu quan hệ SQL Server của Microsoft. SQL Server là một trong những hệ quản lý cơ sở dữ liệu phổ biến nhất trên thế giới và được triển khai trong nhiều môi trường doanh nghiệp.

Transact-SQL là ngôn ngữ chính được sử dụng để phát triển các ứng dụng và thao tác với cơ sở dữ liệu trong SQL Server. Nhờ tính linh hoạt và khả năng mở rộng, T-SQL có thể được sử dụng để xây dựng các ứng dụng quản lý cơ sở dữ liệu phức tạp, viết các truy vấn phức tạp để trích xuất thông tin từ cơ sở dữ liệu, và thực hiện các tác vụ quản lý dữ liệu như xử lý nâng cao, cung cấp bảo mật và tối ưu hóa truy vấn.

Bên cạnh việc sử dụng T-SQL trong SQL Server, ngôn ngữ này cũng có thể được sử dụng trong các hệ thống quản lý cơ sở dữ liệu quan hệ khác như Azure SQL Database và Azure Synapse Analytics.

Do đó, sự phổ biến của SQL Server và tính linh hoạt của Transact-SQL đã làm cho ngôn ngữ này trở thành một công cụ quan trọng trong việc phát triển ứng dụng và quản lý cơ sở dữ liệu trong nhiều môi trường doanh nghiệp và dự án phức tạp.

2.2 Quản lý giao tác, giao dịch

Trong SQL Server, quản lý giao tác và giao dịch là một phần quan trọng để đảm bảo tính toàn vẹn và nhất quán của dữ liệu trong cơ sở dữ liệu. Giao tác là một chuỗi các thao

tác cơ sở dữ liệu (như thêm, cập nhật hoặc xóa dữ liệu) được nhóm lại và xem như một đơn vị làm việc duy nhất. Quản lý giao tác giúp đảm bảo rằng các thao tác được thực hiện thành công hoặc được hoàn tác nếu có lỗi xảy ra.

Dưới đây là một số khái niệm và cú pháp quan trọng liên quan đến quản lý giao tác trong SQL Server:

- Giao tác (Transaction): Một giao tác là một tập hợp các thao tác cơ sở dữ liệu được thực hiện như một đơn vị duy nhất. Một giao tác phải có tính ACID (Atomicity, Consistency, Isolation, Durability) để đảm bảo tính toàn vẹn của dữ liệu.

- COMMIT: COMMIT là câu lệnh được sử dụng để lưu trữ các thay đổi từ một giao tác vào cơ sở dữ liệu. Khi COMMIT được thực thi, các thay đổi trong giao tác trước đó sẽ trở thành vĩnh viễn và không thể hoàn tác.

- ROLLBACK: ROLLBACK là câu lệnh được sử dụng để hoàn tác một giao tác và hủy bỏ tất cả các thay đổi trong giao tác đó. Khi ROLLBACK được thực thi, cơ sở dữ liệu sẽ được khôi phục về trạng thái trước khi giao tác bắt đầu.

- SAVEPOINT: SAVEPOINT là một điểm lưu trữ trong một giao tác, cho phép bạn xác định một vị trí trong giao tác để có thể hoàn tác đến đó. Điều này cho phép bạn phân chia một giao tác thành nhiều phần nhỏ và chỉ hoàn tác một phần cụ thể nếu cần thiết.

- BEGIN TRANSACTION: BEGIN TRANSACTION là câu lệnh được sử dụng để bắt đầu một giao tác mới. Sau câu lệnh này, các thao tác cơ sở dữ liệu được thực thi sẽ thuộc về giao tác đó cho đến khi COMMIT hoặc ROLLBACK được gọi.

- SET TRANSACTION: SET TRANSACTION là câu lệnh được sử dụng để cấu hình các thuộc tính của giao tác, Cấp độ cô lập quy định mức độ nào mà các thay đổi từ giao tác hiện tại được nhìn thấy bởi các giao tác khác đang thực thi song song.

Quản lý giao tác và giao dịch trong SQL Server là rất quan trọng để đảm bảo tính toàn vẹn và nhất quán của dữ liệu. Việc sử dụng các câu lệnh COMMIT, ROLLBACK và SAVEPOINT giúp đảm bảo rằng các thay đổi được xử lý một cách an toàn và có thể hoàn tác khi cần thiết.

2.3 Xử lý đồng thời

Xử lý đồng thời trong SQL là quá trình cho phép nhiều truy vấn hoặc giao tác cùng thực thi đồng thời trên cùng một cơ sở dữ liệu. Khi nhiều truy vấn hoặc giao tác được thực thi cùng một lúc, việc quản lý sự đồng thời giữa chúng là cần thiết để đảm bảo tính toàn vẹn và nhất quán của dữ liệu.

SQL Server cung cấp các cơ chế xử lý đồng thời để quản lý việc thực thi các truy vấn và giao tác đồng thời. Dưới đây là một số khái niệm và cơ chế quan trọng liên quan đến xử lý đồng thời trong SQL:

- Cấp độ cô lập (Isolation levels): Cấp độ cô lập xác định mức độ nào mà một giao tác hiện tại có thể nhìn thấy các thay đổi từ các giao tác khác đang được thực thi đồng thời. Các cấp độ cô lập phổ biến bao gồm READ UNCOMMITTED, READ COMMITTED, REPEATABLE READ và SERIALIZABLE.

- Khóa (Locking): SQL Server sử dụng khóa để đảm bảo tính nhất quán của dữ liệu trong quá trình đồng thời. Khóa có thể đặt trên các tài nguyên (bảng, hàng, trang) để ngăn chặn các truy vấn khác từ việc thay đổi hoặc đọc dữ liệu đó trong khi có một giao tác đang thực thi.

- Kiểm soát phiên (Concurrency control): SQL Server sử dụng các thuật toán kiểm soát phiên để quản lý việc đồng thời thực thi các truy vấn và giao tác. Các thuật toán này đảm bảo tính toàn vẹn và nhất quán của dữ liệu trong quá trình đồng thời bằng cách áp dụng các nguyên tắc và quy tắc quản lý khóa.

- Bảng thời gian (Temporal table): SQL Server cung cấp tính năng bảng thời gian, cho phép lưu trữ lịch sử của dữ liệu. Điều này cho phép các truy vấn và giao tác đồng thời truy cập và thay đổi dữ liệu trong quá khứ và hiện tại một cách an toàn.

Ngoài ra, cần thiết lập các chiến lược xử lý đồng thời phù hợp, như sử dụng chỉ mục (indexing) cho các truy vấn thường xuyên được thực hiện, tối ưu hóa câu truy vấn để giảm thiểu thời gian chờ đợi và xung đột, sử dụng các kỹ thuật chia nhỏ (partitioning) để phân tán dữ liệu và giảm đụng độ, và sử dụng các công cụ giám sát và điều chỉnh hiệu suất để theo dõi và tối ưu hóa xử lý đồng thời.

Một số kỹ thuật và công nghệ phổ biến được sử dụng để tăng hiệu suất xử lý đồng thời trong SQL Server bao gồm:

- Parallel Execution: SQL Server có khả năng thực hiện truy vấn và giao tác theo cách đồng thời sử dụng nhiều luồng xử lý. Bằng cách cấu hình phù hợp và sử dụng các lợi ích của đa luồng, bạn có thể tăng hiệu suất xử lý đồng thời.

- In-Memory OLTP: SQL Server cung cấp tính năng In-Memory OLTP cho phép lưu trữ và xử lý dữ liệu trong bộ nhớ để tăng hiệu suất và giảm độ trễ. Điều này đặc biệt hữu ích trong các tình huống đòi hỏi xử lý đồng thời nhanh chóng và có hiệu suất cao.

- Partitioning: Sử dụng kỹ thuật chia nhỏ dữ liệu thành các phân vùng (partitions) để phân tán dữ liệu và giảm đụng độ. Khi dữ liệu được phân vùng, các truy vấn và giao tác có thể được thực thi đồng thời trên các phân vùng khác nhau, tăng hiệu suất và khả năng mở rộng của hệ thống.

- Snapshot Isolation: Snapshot isolation cho phép các giao tác đọc dữ liệu từ một phiên bản được chụp (snapshot) của cơ sở dữ liệu tại thời điểm bắt đầu giao tác. Điều này giúp tránh xung đột đọc và cải thiện hiệu suất đồng thời, nhưng có thể dẫn đến sự mất mát của tính nhất quán.

- Optimistic Concurrency Control (OCC): Sử dụng kiểm soát phiên lạc quan để cho phép nhiều giao tác đọc và ghi dữ liệu đồng thời mà không cần khóa tài nguyên. Trong kỹ thuật này, SQL Server không chặn các giao tác từ việc đọc hoặc ghi dữ liệu, mà sẽ kiểm tra tính nhất quán của dữ liệu khi giao tác hoàn thành. Nếu xảy ra xung đột, giao tác sẽ được hủy hoặc xử lý lại.

- Deadlock Detection: SQL Server có khả năng phát hiện và giải quyết deadlock (tình huống mắc kẹt) trong quá trình xử lý đồng thời. Deadlock xảy ra khi hai hoặc nhiều giao tác đang chờ đợi tài nguyên mà lẫn nhau đang giữ, dẫn đến sự bế tắc. SQL Server sử dụng cơ chế phát hiện deadlock để phát hiện và giải quyết deadlock tự động.

- Thống kê và tối ưu câu truy vấn: SQL Server cung cấp các công cụ để thu thập và duy trì thống kê về cơ sở dữ liệu. Thống kê này giúp trình tối ưu câu truy vấn chọn phương pháp thực thi tối ưu và giảm thời gian chờ đợi và xung đột khi thực thi đồng thời.

Quản lý và xử lý đồng thời trong SQL Server là một nhiệm vụ quan trọng để đảm bảo tính toàn vẹn và hiệu suất của hệ thống. Bằng cách sử dụng các cơ chế và công nghệ phù hợp, bạn có thể đạt được xử lý đồng thời hiệu quả và đáng tin cậy trong môi trường SQL Server.

2.4 Cơ chế phục hồi dữ liệu sau sự cố

SQL Server cung cấp các cơ chế phục hồi dữ liệu sau sự cố để đảm bảo tính toàn vẹn và khả năng phục hồi của cơ sở dữ liệu. Dưới đây là một số cơ chế phục hồi quan trọng trong SQL Server:

- Transaction Log (Bản ghi giao dịch): SQL Server sử dụng bản ghi giao dịch để ghi lại tất cả các hoạt động giao dịch trong cơ sở dữ liệu. Bản ghi giao dịch bao gồm thông tin về các thay đổi dữ liệu và các hoạt động cần thiết để hoàn tất hoặc hủy bỏ giao dịch. Khi xảy ra sự cố, SQL Server có thể sử dụng bản ghi giao dịch để phục hồi dữ liệu và đưa cơ sở dữ liệu về trạng thái đáng tin cậy.

- Checkpoints: SQL Server thực hiện checkpoints để ghi lại dữ liệu từ bộ nhớ đệm vào các tệp dữ liệu vĩnh viễn trên đĩa. Checkpoints giúp đảm bảo rằng các thay đổi dữ liệu đã được ghi lại một cách đầy đủ và an toàn. Khi xảy ra sự cố, SQL Server có thể sử dụng các checkpoint đã ghi lại để phục hồi dữ liệu.

- Transaction Log Backup (Sao lưu bản ghi giao dịch): SQL Server cho phép sao lưu định kỳ của bản ghi giao dịch. Bằng cách sao lưu bản ghi giao dịch, bạn có thể tạo ra các điểm khôi phục để phục hồi dữ liệu đến một thời điểm cụ thể trong quá khứ. Sao lưu bản ghi giao dịch là một phương pháp quan trọng để đảm bảo tính toàn vẹn và khả năng phục hồi của cơ sở dữ liệu.

- Điểm phục hồi (Restore Point): SQL Server cho phép bạn tạo ra các điểm phục hồi để lưu trữ trạng thái của cơ sở dữ liệu tại một thời điểm cụ thể. Điểm phục hồi là các điểm mà bạn có thể phục hồi cơ sở dữ liệu đến sau khi xảy ra sự cố. Điểm phục hồi giúp đảm bảo rằng bạn có thể khôi phục dữ liệu trở lại trạng thái trước khi sự cố xảy ra.

- Mirror Server (Máy chủ phản chiếu): SQL Server hỗ trợ cấu hình máy chủ phản chiếu (mirror server) để tạo ra một bản sao dự phòng của cơ sở dữ liệu chính. Máy chủ phản chiếu duy trì một bản sao chính xác của cơ sở dữ liệu và có thể tiếp tục phục vụ các truy vấn và giao dịch khi máy chủ chính gặp sự cố. Khi máy chủ chính

phục hồi từ sự cố, dữ liệu từ máy chủ phản chiếu có thể được sử dụng để đồng bộ hóa và khôi phục cơ sở dữ liệu chính.

- AlwaysOn Availability Groups: AlwaysOn Availability Groups là một tính năng của SQL Server Enterprise Edition cho phép tạo ra một nhóm các cơ sở dữ liệu phân tán với tính năng đảm bảo tính sẵn sàng cao (high availability). Trong trường hợp xảy ra sự cố với một máy chủ, AlwaysOn Availability Groups tự động chuyển đổi các truy vấn và giao dịch sang một máy chủ khác trong nhóm. Khi sự cố được giải quyết, dữ liệu có thể được đồng bộ và phục hồi trạng thái chính xác.

- Point-in-Time Recovery (Phục hồi theo thời điểm): SQL Server hỗ trợ phục hồi dữ liệu đến một thời điểm cụ thể trong quá khứ sử dụng các bản sao lưu và bản ghi giao dịch. Bằng cách sử dụng phương pháp này, bạn có thể khôi phục dữ liệu đến trạng thái trước khi sự cố xảy ra, giúp đảm bảo tính toàn vẹn và khả năng phục hồi của cơ sở dữ liệu.

- Thiết lập và kiểm tra kế hoạch phục hồi: Quan trọng nhất là thiết lập và kiểm tra kế hoạch phục hồi đầy đủ và chính xác. Kế hoạch này bao gồm việc xác định các bước cụ thể để phục hồi dữ liệu sau sự cố, bao gồm các bước sao lưu, phục hồi từ bản sao lưu, và phục hồi từ bản ghi giao dịch. Ngoài ra, nên thực hiện kiểm tra định kỳ để đảm bảo rằng kế hoạch phục hồi hoạt động như mong đợi và dữ liệu có thể được phục hồi một cách hiệu quả.

- Thiết lập điểm phục hồi và thời gian phục hồi (Recovery Point Objective và Recovery Time Objective): Điểm phục hồi (RPO) là thời điểm cuối cùng mà dữ liệu có thể được phục hồi đến sau khi xảy ra sự cố. Thời gian phục hồi (RTO) là thời gian cần thiết để khôi phục cơ sở dữ liệu sau khi xảy ra sự cố. Thiết lập và tuân thủ RPO và RTO đảm bảo rằng dữ liệu được phục hồi đến một trạng thái gần nhất với sự cố và trong khoảng thời gian hợp lý.

- Monitoring and Alerting (Giám sát và cảnh báo): SQL Server cung cấp các công cụ và tính năng để giám sát hoạt động của cơ sở dữ liệu và phát hiện sớm các vấn đề hoặc sự cố tiềm ẩn. Bằng cách thiết lập giám sát và cảnh báo thích hợp, bạn có thể nhận được thông báo khi có sự cố xảy ra, cho phép bạn nhanh chóng phản ứng và thực hiện các biện pháp phục hồi.

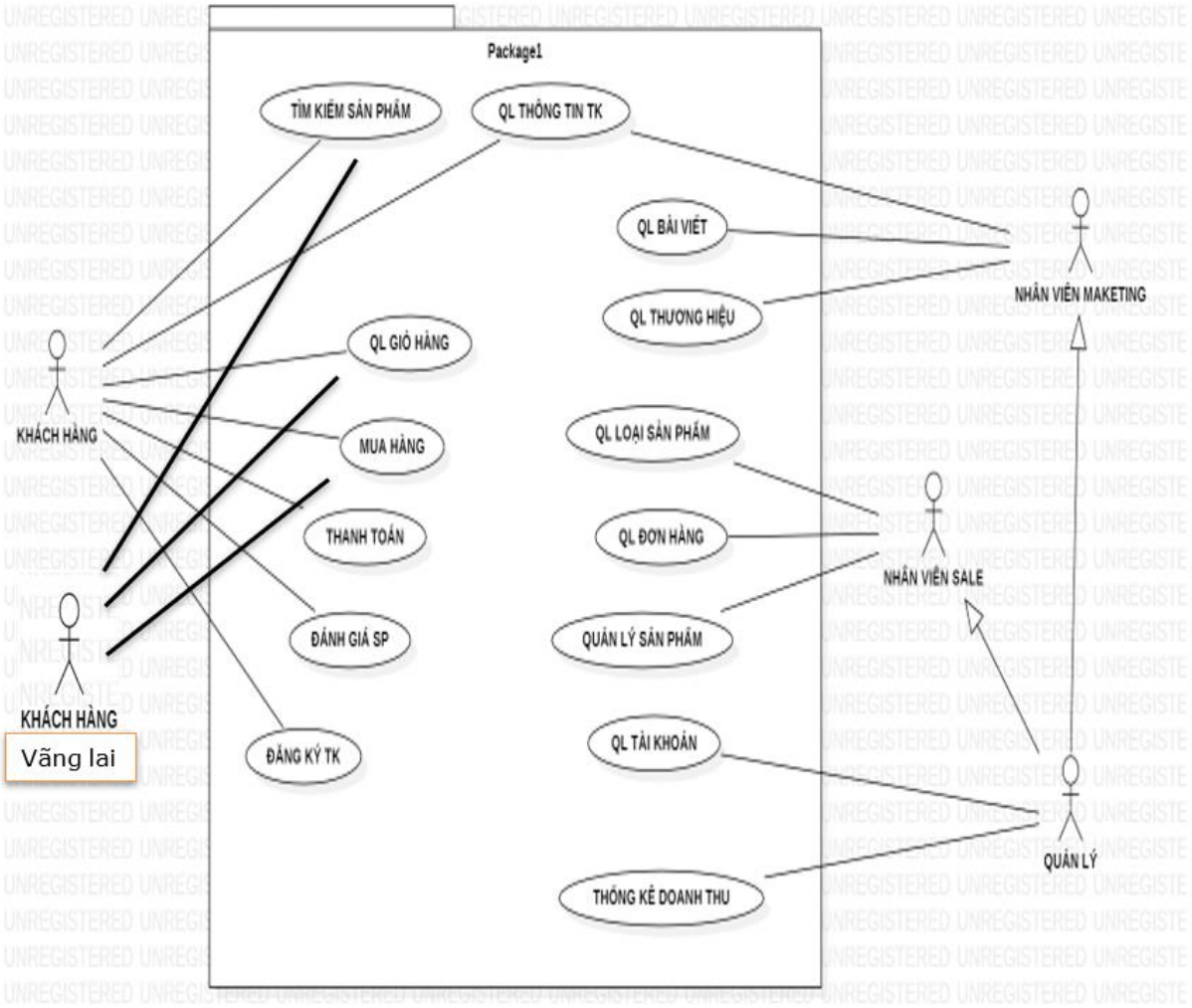
Các cơ chế phục hồi dữ liệu sau sự cố trong SQL Server đảm bảo rằng bạn có khả năng khôi phục dữ liệu một cách an toàn và hiệu quả khi xảy ra sự cố. Việc thiết lập kế hoạch phục hồi, sao lưu định kỳ và kiểm tra kế hoạch sẽ giúp đảm bảo tính toàn vẹn và khả năng phục hồi của cơ sở dữ liệu.

Chương 3 PHÂN TÍCH – THIẾT KẾ DỮ LIỆU

3.1 Thiết kế cơ sở dữ liệu tập trung cho hệ thống

3.1.1 Phân tích yêu cầu

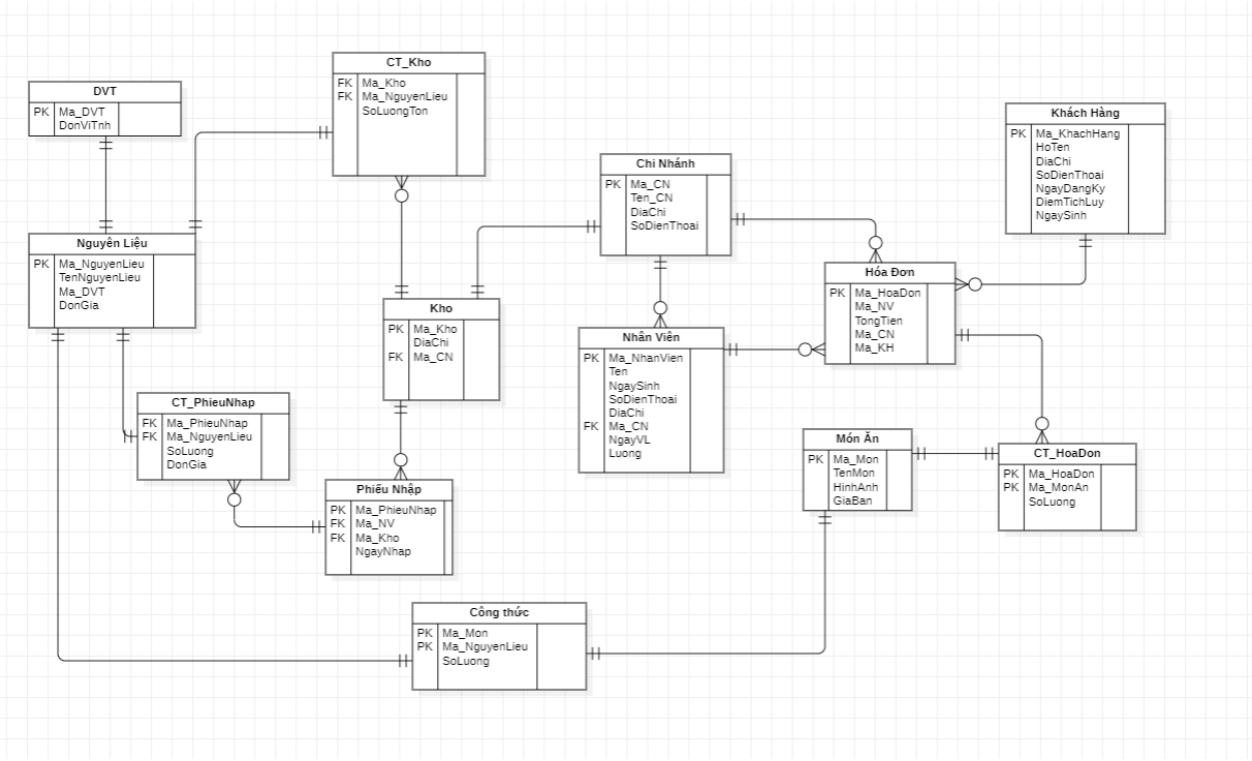
Hình 8 SƠ ĐỒ CHỨC NĂNG



Hình 2 Lược đồ use case về các chức năng

3.1.2 Phân tích dữ liệu

3.1.2.1 Mô hình thực thể kết hợp



Hình 3 Lược đồ ERD

3.1.2.2 Mô hình quan hệ

1. ChiNhanh (**Ma_CN**, TenCN, DiaChi, SoDienThoai)
2. MonAn **(Ma_Mon**, TenMon, HinhAnh, GiaBan)
3. DVT (**Ma_DVT**, DonViTinh)
4. NguyenLieu (**Ma_NguyenLieu**, TenNguyenLieu, **Ma_DVT**, DonGia)
5. CongThuc (**Ma_Mon**, **Ma_NguyenLieu**, SoLuong)
6. KhachHang (**Ma_KhachHang**, HoTen, DiaChi, SoDienThoai, NgayDangKy, DiemTichLuy, NgaySinh, MatKhauKH)
7. NhanVien (**Ma_NhanVien**, Ten, NgaySinh, SoDienThoai, DiaChi, **Ma_CN**, NgayVL, Luong, MatKhauNV)
8. Kho (**Ma_Kho**, DiaChi, **Ma_CN**)

9. CT_Kho (**Ma_Kho**, **Ma_NguyenLieu**, SoLuongTon)
10. PhieuNhap (**Ma_PhieuNhap**, **Ma_NV**, **Ma_Kho**, NgayNhap)
11. CT_PhieuNhap (**Ma_PhieuNhap**, **Ma_NguyenLieu**, SoLuong, DonGia)
12. HoaDon (**Ma_HoaDon**, **Ma_NV**, TongTien, **Ma_CN**, Ma_KH)
13. CT_HoaDon (**Ma_HoaDon**, **Ma_MonAn**, SoLuong)

3.1.2.3 KhachHang

Thuộc tính	Kiểu dữ liệu	Ràng buộc / Miền giá trị
Ma_KhachHang	INT	PRIMARY KEY
HoTen	NVARCHAR (50)	NOT NULL
DiaChi	DATETIME	NOT NULL
SoDienThoai	VARCHAR (20)	NOT NULL
MatKhauKH	NVARCHAR (20)	NOT NULL
NgayDangKy	NVARCHAR (100)	NOT NULL
DiemTichLuy	CHAR (10)	
NgaySinh	DATETIME	

3.1.2.4 ChiNhanh

Thuộc tính	Kiểu dữ liệu	Ràng buộc / Miền giá trị
Ma_CN	CHAR (10)	PRIMARY KEY
TenCN,	VARCHAR (10)	NOT NULL, UNIQUE
DiaChi	VARCHAR (100)	NOT NULL
SoDienThoai	VARCHAR (15)	NOT NULL

3.1.2.5 MonAn

Thuộc tính	Kiểu dữ liệu	Ràng buộc / Miền giá trị

Ma_Mon	INT	PRIMARY KEY, IDENITY (1,1)
TenMo	NVARCHAR (50)	NOT NULL
HinhAnh	NVARCHAR (100)	
GiaBan	MONEY	

3.1.2.6 DVT

Thuộc tính	Kiểu dữ liệu	Ràng buộc / Miền giá trị
Ma_DVT	INT	PRIMARY KEY, IDENITY (1,1)
DonViTinh	VARCHAR (50)	

3.1.2.7 NguyenLieu

Thuộc tính	Kiểu dữ liệu	Ràng buộc / Miền giá trị
Ma_NguyenLieu	INT	PRIMARY KEY, IDENITY (1,1)
TenNguyenLieu	NVARCHAR (100)	NOT NULL
Ma_DVT	INT	FOREIGN KEY, REFERENCES DVT(Ma_DVT)
DonGia	MONEY	NOT NULL

3.1.2.8 NhanVien

Thuộc tính	Kiểu dữ liệu	Ràng buộc / Miền giá trị
Ma_NhanVien	INT	PRIMARY KEY
Ten	NVARCHAR (50)	NOT NULL
NgaySinh	DATETIME	NOT NULL
MatKhauNV	VARCHAR (20)	NOT NULL
SoDienThoai	NVARCHAR (20)	NOT NULL

DiaChi	NVARCHAR (100)	NOT NULL
Ma_CN	CHAR (10)	FOREIGN KEY
NgayVL	DATETIME	NOT NULL
Luong	MONEY	NOT NULL

3.1.2.9 KHO

Thuộc tính	Kiểu dữ liệu	Ràng buộc / Miền giá trị
Ma_Kho	INT	PRIMARY KEY
DiaChi	NVARCHAR (255)	NOT NULL
Ma_CN	CHAR (10)	FOREIGN KEY

3.1.2.10 CT_Kho

Thuộc tính	Kiểu dữ liệu	Ràng buộc / Miền giá trị
Ma_Kho	INT	FOREIGN KEY
Ma_NguyenLieu	INT	FOREIGN KEY
SoLuongTon	INT	NOTNULL

3.1.2.11 PhieuNhap

Thuộc tính	Kiểu dữ liệu	Ràng buộc / Miền giá trị
Ma_PhieuNhap	INT	PRIMARY KEY
Ma_NV	INT	FOREIGN KEY
Ma_Kho	INT	FOREIGN KEY
NgayNhap	DATETIME	NOT NULL

3.1.2.12 CT_PhieuNhap

Thuộc tính	Kiểu dữ liệu	Ràng buộc / Miền giá trị
Ma_PhieuNhap	INT	FOREIGN KEY
Ma_NguyenLieu	INT	FOREIGN KEY
SoLuong	INT	NOT NULL
DonGia	MONEY	NOT NULL

3.1.2.13 HoaDon

Thuộc tính	Kiểu dữ liệu	Ràng buộc / Miền giá trị
Ma_HoaDon	INT	PRIMARY KEY
Ma_NV	INT	FOREIGN KEY
TongTien	MONEY	NOT NULL
Ma_CN	CHAR (10)	FOREIGN KEY
Ma_KH	INT	FOREIGN KEY

3.1.2.14 CT_HoaDon

Thuộc tính	Kiểu dữ liệu	Ràng buộc / Miền giá trị
Ma_HoaDon	INT	FOREIGN KEY
Ma_MonAn	INT	FOREIGN KEY
SoLuong	INT	NOT NULL

3.1.2.15 Chuẩn hóa mô hình quan hệ

3.1.2.16 ChiNhanh

Tập phụ thuộc hàm F = {Ma_CN → TenCN, DiaChi, SoDienThoai;}

➔ Bảng đạt chuẩn **3NF** (Các thuộc tính không khóa phụ thuộc hoàn toàn vào thuộc tính khóa và không phụ thuộc bắc cầu vào thuộc tính không khóa khác; Thuộc tính khóa phụ thuộc vào thuộc tính không khóa)

3.1.2.17 MonAn

Tập phụ thuộc hàm F = {Ma_Mon → TenMon, HinhAnh, Gia Ban;}

➔ Bảng đạt chuẩn **3NF** (Các thuộc tính không khóa phụ thuộc hoàn toàn vào thuộc tính khóa và không phụ thuộc bắc cầu vào thuộc tính không khóa khác; Thuộc tính khóa không phụ thuộc vào thuộc tính không khóa)

3.1.2.18 DVT

Tập phụ thuộc hàm F = {Ma_DVT → DonViTinh}

➔ Bảng đạt chuẩn **3NF** (Các thuộc tính không khóa phụ thuộc hoàn toàn vào thuộc tính khóa và không phụ thuộc bắc cầu vào thuộc tính không khóa khác; Thuộc tính khóa phụ thuộc vào thuộc tính không khóa)

3.1.2.19 NguyenLieu

Tập phụ thuộc hàm F = {Ma_NguyenLieu → TenNguyenLieu, Ma_DVT, DonGia

Ma_DVT -> DonViTinh}

➔ Bảng đạt chuẩn **3NF** (tất cả các tập phụ thuộc hàm non-trivial trong bảng đều phụ thuộc vào khóa chính.)

3.1.2.20 CongThuc

Tập phụ thuộc hàm F = {Ma_Mon, Ma_NguyenLieu → SoLuong}

➔ Bảng đạt chuẩn **3NF** (tất cả các tập phụ thuộc hàm non-trivial trong bảng đều phụ thuộc vào khóa chính)

3.1.2.21 KhachHang

Tập phụ thuộc hàm F = {Ma_KhachHang → HoTen, DiaChi, SoDienThoai, NgayDangKy, DiemTichLuy, NgaySinh}

➔ Bảng đạt chuẩn **3NF** (Các thuộc tính không khóa phụ thuộc hoàn toàn vào thuộc tính khóa và không phụ thuộc bắc cầu vào thuộc tính không khóa khác; Thuộc tính khóa không phụ thuộc vào thuộc tính không khóa)

3.1.2.22 NhanVien

Tập phụ thuộc hàm F = {Ma_NhanVien→ Ten, NgaySinh, SoDienThoai, DiaChi, Ma_CN, NgayVL, Luong

Ma_CN → TenCN, DiaChi, SoDienThoai}

➔ Bảng đạt chuẩn **3NF** (tất cả các tập phụ thuộc hàm non-trivial trong bảng đều phụ thuộc vào khóa chính)

3.1.2.23 Kho

Tập phụ thuộc hàm F = {Ma_Kho → DiaChi, Ma_CN

Ma_CN →TenCN, DiaChi, SoDienThoai}

➔ Bảng đạt chuẩn **3NF** (tất cả các tập phụ thuộc hàm non-trivial trong bảng đều phụ thuộc vào khóa chính)

3.1.2.24 CT_Kho

Tập phụ thuộc hàm F = {Ma_Kho, Ma_NguyenLieu → SoLuongTon}

➔ Bảng đạt chuẩn **3NF** (tất cả các tập phụ thuộc hàm non-trivial trong bảng đều phụ thuộc vào khóa chính)

3.1.2.25 PhieuNhap

- Tập phụ thuộc hàm F = {Ma_PhieuNhap → Ma_NV, Ma_Kho, NgayNhap
- Ma_NV → Ten, NgaySinh, SoDienThoai, DiaChi, Ma_CN, NgayVL, Luong
- Ma_Kho → DiaChi, Ma_CN
- Ma_CN → TenCN, DiaChi, SoDienThoai}

➔Bảng đạt chuẩn **3NF** (tất cả các tập phụ thuộc hàm non-trivial trong bảng đều phụ thuộc vào khóa chính)

3.1.2.26 CT_PhieuNhap

- Tập phụ thuộc hàm F = {Ma_PhieuNhap, Ma_NguyenLieu → SoLuong, DonGia
- Ma_PhieuNhap → Ma_NV, Ma_Kho, NgayNhap
- Ma_NV → Ten, NgaySinh, SoDienThoai, DiaChi, Ma_CN, NgayVL, Luong
- Ma_Kho → DiaChi, Ma_CN
- Ma_CN → TenCN, DiaChi, SoDienThoai
- Ma_NguyenLieu → TenNguyenLieu, Ma_DVT, DonGia
- Ma_DVT → DonViTinh}

➔Bảng đạt chuẩn **3NF** (tất cả các tập phụ thuộc hàm non-trivial trong bảng đều phụ thuộc vào khóa chính)

3.1.2.27 HoaDon

- Tập phụ thuộc hàm F = {Ma_HoaDon → Ma_NV, TongTien, Ma_CN, Ma_KH
- Ma_NV → Ten, NgaySinh, SoDienThoai, DiaChi, Ma_CN, NgayVL, Luong
- Ma_CN → TenCN, DiaChi, SoDienThoai
- Ma_KH →HoTen, DiaChi, SoDienThoai, NgayDangKy, DiemTichLuy, NgaySinh}

➔Bảng đạt chuẩn **3NF** (tất cả các tập phụ thuộc hàm non-trivial trong bảng đều phụ thuộc vào khóa chính)

3.1.2.28 CT_HoaDon

- Tập phụ thuộc hàm F = {Ma_HoaDon, Ma_MonAn → SoLuong
- Ma_HoaDon → Ma_NV, TongTien, Ma_CN, Ma_KH
- Ma_NV → Ten, NgaySinh, SoDienThoai, DiaChi, Ma_CN, NgayVL, Luong
- Ma_CN → TenCN, DiaChi, SoDienThoai
- Ma_KH → HoTen, DiaChi, SoDienThoai, NgayDangKy, DiemTichLuy, NgaySinh
- Ma_MonAn → TenMon, HinhAnh, Gia Ban}

➔Bảng đạt chuẩn **3NF** (tất cả các tập phụ thuộc hàm non-trivial trong bảng đều phụ thuộc vào khóa chính)

➔**Kết luận**: tất cả các bảng trong mô hình quan hệ đã được chuẩn hóa và đạt chuẩn 3NF. Điều này đảm bảo tính chất của cơ sở dữ liệu, giúp giảm thiểu sự lặp lại và tăng tính nhất quán, hiệu quả của quá trình truy vấn và cập nhật dữ liệu.

3.1.3 Ràng buộc trọn vẹn

3.1.3.1 SoDienThoai không được trùng

- Bối cảnh: KhachHang
- Bảng tầm ảnh hưởng:

	Thêm	Xóa	Sửa
KhachHang	+	-	+ (Username)

Bảng 1 Bảng tầm ảnh hưởng R1

3.1.3.2 Sau khi nhập phiếu nhập và chi tiết nhập kho thì tự động cập nhật vào số lượng tồn kho của Chi tiết kho và kho

- Bối cảnh: CT_PhieuKho, PhieuKho, CT_Kho, Kho
- Bảng tầm ảnh hưởng:

	Thêm	Xóa	Sửa
CT_PhieuKho	-	+	- (*)
PhieuKho	-	-	+ (CT_PhieuKho)

Bảng 2 Bảng tầm ảnh hưởng R2

3.1.3.3 MatKhauKH không được để trống

- Bối cảnh: KhachHang
- Bảng tầm ảnh hưởng:

	Thêm	Xóa	Sửa

KhachHang	+	-	+ (MatKhauKH)

Bảng 3 Bảng tầm ảnh hưởng R2

3.1.3.4 Không được xóa món ăn khi món ăn đó có công thức liên kết

- Bối cảnh: MonAn, CongThuc
- Bảng tầm ảnh hưởng:

	Thêm	**Xóa**	**Sửa**
MonAn	-	+	- (*)
CongThuc	-	-	+ (Ma_MonAn)

Bảng 4 Bảng tầm ảnh hưởng R4

3.1.3.5 Số lượng sản phẩm trong chi tiết hóa đơn là số âm không

- Bối cảnh: ChiTietHoaDon
- Bảng tầm ảnh hưởng:

	Thêm	**Xóa**	**Sửa**
ChiTietHoaDon	+	-	+ (SoLuong)

Bảng 5 Bảng tầm ảnh hưởng R5

3.1.3.6 Ngày nhập trong phiếu nhập kho không được nhập ngày trong tương lai

- Bối cảnh: PhieuNhap
- Bảng tầm ảnh hưởng:

	Thêm	**Xóa**	**Sửa**
PhieuNhap	+	-	+ (NgayNhap)

Bảng 6 Bảng tầm ảnh hưởng R6

3.1.3.7 Không thể thêm món ăn trong chi tiết hóa đơn nếu nguyên liệu liên kết công thức hết

- Bối cảnh: CT_HoaDon, CT_Kho
- Bảng tầm ảnh hưởng:

	Thêm	Xóa	Sửa
CT_HoaDon	+	-	+ (SoLuongMonAN)

Bảng 7 Bảng tầm ảnh hưởng R7

3.1.3.8 Điểm tích lũy của khách hàng không được âm, nếu Khách hàng nào có trên 1000 điểm thì thành VIP

- Bối cảnh: KhachHang
- Bảng tầm ảnh hưởng:

	Thêm	Xóa	Sửa
KhachHang	+	-	+ (DiemTichLuy, LoaiKhachHang)

Bảng 8 Bảng tầm ảnh hưởng R8

3.1.3.9 Khách hàng phải đủ 16 trở lên mới được thêm mới

- Bối cảnh: KhachHang
- Bảng tầm ảnh hưởng:

	Thêm	Xóa	Sửa
KhachHang	-	-	+ (NgaySinh)

Bảng 9 Bảng tầm ảnh hưởng R9

3.1.3.10 Nhân viên phải đủ 18 trở lên mới được thêm mới

- Bối cảnh: NhanVien
- Bảng tầm ảnh hưởng:

	Thêm	Xóa	Sửa
NhanVien	+	-	+ (NgaySinh)

Bảng 10 Bảng tầm ảnh hưởng R10

3.1.3.11 Đơn giá của nguyên liệu phải lớn hơn 0

- Bối cảnh: NguyenLieu
- Bảng tầm ảnh hưởng:

	Thêm	Xóa	Sửa
NguyenLieu	+	-	+ (DonGia)

Bảng 11 Bảng tầm ảnh hưởng R12

3.1.3.12 Giá bán của món ăn phải lớn hơn 0

- Bối cảnh: MonAn
- Bảng tầm ảnh hưởng:

	Thêm	Xóa	Sửa
MonAn	+	-	+ (GiaBan)

Bảng 12 Bảng tầm ảnh hưởng R13

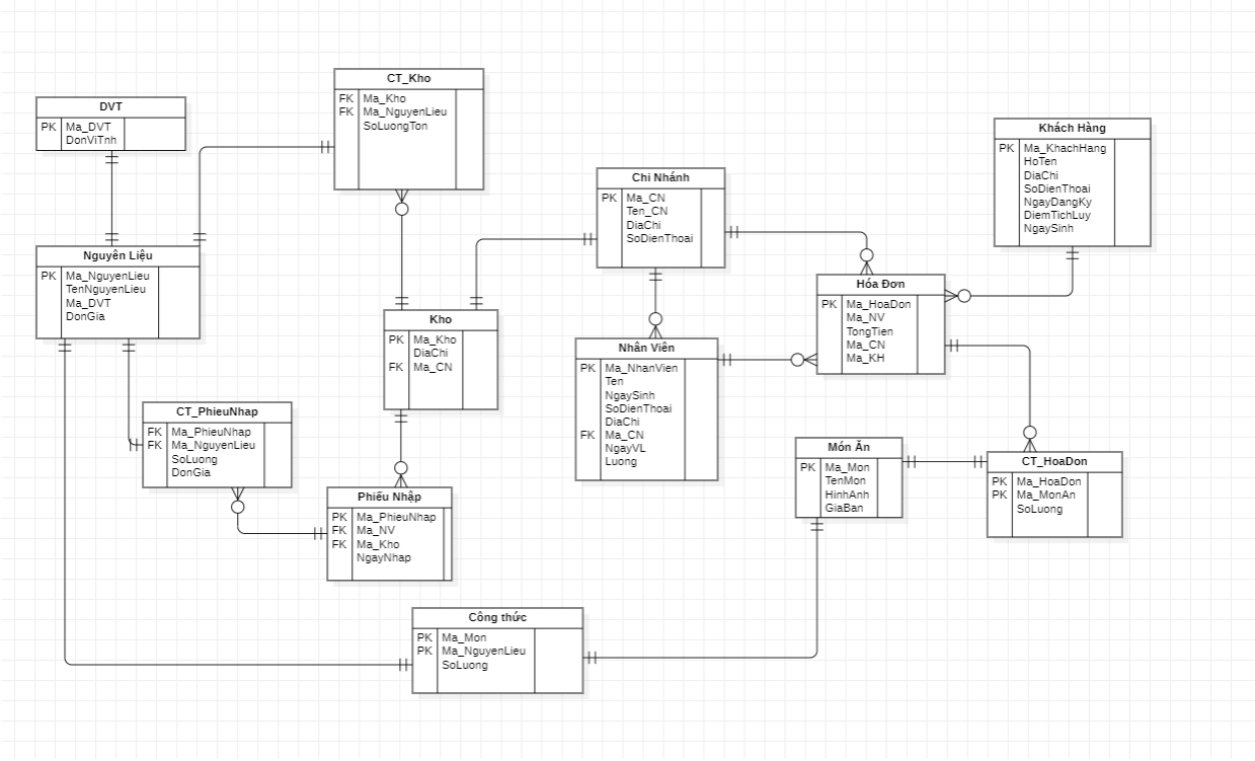
3.1.3.13 Tính lại tổng tiền hóa đơn sau khi thêm, cập nhật, xóa chi tiết hóa đơn

- Bối cảnh: CT_HoaDon, HoaDon
- Bảng tầm ảnh hưởng:

	Thêm	Xóa	Sửa
CT_HoaDon	-	+	+ (Ma_HoaDon, Ma_MonAn, Soluong)
HoaDon	-	-	+ (TongTien)

Chương 4 THIẾT KẾ XỬ LÝ TRONG CLIENT/ SERVER

4.1 Cấu trúc các bảng dữ liệu



Hình 1 Cấu trúc các bảng dữ liệu (Lược đồ ERD)

4.2 Các giao tác

4.2.1 Thêm tài khoản

Client	Server
Yêu cầu server tạo tài khoản và truyền tham số	
	Tạo tài khoản • Đọc SoDienThoai để tạo MatKhauKH • Thêm tài khoản vào bảng KhachHang • Tạo user trong database • Thêm user vào Role thích hợp • Cấp TSK_Client (tài khoản khách) quyền impersonate user
Nhận kết quả giao tác	
Thông báo kết quả giao tác cho người dùng	

Bảng 13 Các bước xử lý giao tác Thêm tài khoản

4.2.2 Xóa tài khoản

Client	Server
Yêu cầu server xóa tài khoản và truyền tham số	
	Xóa tài khoản • Xóa tài khoản khỏi bảng KhachHang • Hủy quyền impersonate user được xóa của TSK_Client (tài khoản khách) • Xóa user trong database
Nhận kết quả giao tác	

Thông báo kết quả giao tác cho người dùng	

Bảng 14 Các bước xử lý giao tác Xóa tài khoản

4.2.3 Cập nhật mật khẩu của tài khoản

Client	Server
Yêu cầu server cập nhật mật khẩu của tài khoản và truyền tham số	
	Cập nhật mật khẩu của tài khoản • Cập nhật mật khẩu của tài khoản trong KhachHang
Nhận kết quả giao tác	
Thông báo kết quả giao tác cho người dùng	

Bảng 15 Các bước xử lý giao tác Cập nhật mật khẩu

4.2.4 Thêm nhân viên

Client	Server
Yêu cầu server thêm nhân viên và truyền tham số	
	Thêm nhân viên • Đọc SoDienThoai để tạo Ma_NV • Thêm nhân viên vào NHANVIEN
Nhận kết quả giao tác	
Thông báo kết quả giao tác cho người dùng	

Bảng 16 Các bước xử lý giao tác Thêm nhân viên

4.2.5 Xóa nhân viên

Client	Server

Yêu cầu server xóa nhân viên và truyền tham số	
	Xóa nhân viên • Xóa nhân viên khỏi NhanVien
Nhận kết quả giao tác	
Thông báo kết quả giao tác cho người dùng	

Bảng 17 Các bước xử lý giao tác Xóa nhân viên

4.2.6 Cập nhật nhân viên

Client	Server
Yêu cầu server cập nhật nhân viên và truyền tham số	
	Cập nhật nhân viên • Cập nhật nhân viên trong NhanVien
Nhận kết quả giao tác	
Thông báo kết quả giao tác cho người dùng	

Bảng 18 Các bước xử lý giao tác Cập nhật nhân viên

4.2.7 Thêm món ăn

Client	Server
Yêu cầu server thêm món ăn và truyền tham số	
	Thêm món ăn • Đọc MonAn để tạo Ma_Mon • Thêm món ăn vào Món Ăn
Nhận kết quả giao tác	
Thông báo kết quả giao tác cho người dùng	

Bảng 19 Các bước xử lý giao tác Thêm món ăn

4.2.8 Xóa món ăn

Client	Server
Yêu cầu server xóa món ăn và truyền tham số	
	Xóa món ăn • Xóa món ăn khỏi MonAn
Nhận kết quả giao tác	
Thông báo kết quả giao tác cho người dùng	

Bảng 20 Các bước xử lý giao tác Xóa món ăn

4.2.9 Cập nhật món ăn

Client	Server
Yêu cầu server cập nhật món ăn và truyền tham số	
	Cập nhật món ăn • Cập nhật món ăn trong MonAn
Nhận kết quả giao tác	
Thông báo kết quả giao tác cho người dùng	

Bảng 21 Các bước xử lý giao tác Cập nhật món ăn

4.2.10 Thêm công thức

Client	Server
Yêu cầu server thêm công thức và truyền tham số	
	Thêm công thức • Thêm công thức vào CongThuc

Nhận kết quả giao tác	
Thông báo kết quả giao tác cho người dùng	

Bảng 22 Các bước xử lý giao tác Thêm công thức

4.2.11 Cập nhật công thức

Client	Server
Yêu cầu server cập nhật công thức và truyền tham số	
	Cập nhật công thức • Cập nhật công thức trong CongThuc
Nhận kết quả giao tác	
Thông báo kết quả giao tác cho người dùng	

Bảng 23 Các bước xử lý giao tác Cập nhật công thức

4.2.12 Nhập Kho

Client	Server
Yêu cầu server thêm số lượng nguyên liệu và truyền tham số	
	Thêm số lượng nguyên liệu theo mã • Đọc bảng CT_PhieuNhap để tạo Ma_PhieuNhap • Thêm phiếu nhập vào PhieuNhap
Nhận kết quả giao tác	
Thông báo kết quả giao tác cho người dùng	

Bảng 24 Các bước xử lý giao tác Nhập Kho

4.2.13 Thu tiền (tạo hóa đơn)

Client	Server

Yêu cầu server kiểm tra mã khách hàng	
	Kiểm tra mã khách hàng Tìm khách hàng trong KhachHang bằng Ma_Khachhang
Nhận kết quả kiểm tra	
Hiện thông tin **khách hàng**	
	Thu tiền ăn • Kiểm tra mã món ăn • Đọc HoaDon để tạo Ma_HoaDon • Thêm hóa đơn vào HoaDon • Tách chuỗi mã món ăn thành các mã món ăn • Lần lượt kiểm tra mã món ăn và thêm món ăn vào chi tiết hóa đơn đã được đặt, tăng tổng tiền của hóa đơn tương ứng •
Nhận kết quả giao tác	
Thông báo kết quả giao tác cho người dùng	

Bảng 25 Các bước xử lý giao tác Thu tiền ăn

4.3 Phân quyền

4.3.1 Các đối tượng phân quyền

- **Quản lý**: các nhân viên quản lý của nhà hàng.
- **Nhân viên kho**: các nhân viên kho của nhà hàng.
- **CSKH**: các nhân viên tư vấn của nhà hàng.
- **Nhân viên phục vụ**: Phục vụ cho khách hàng của nhà hàng.

4.3.2 Phân quyền trên các bảng

Ghi chú: S – Select, I – Insert, U – Update, D – Delete

	Quản lý	Nhân viên kho	CSKH	Phục vụ
KhachHang	S, I, U, D		S	I
ChiNhanh	S, I, U, D			
MonAn	S, I, U, D		S	
DVT	S, I, U, D			
NguyenLieu	S, I, U, D	S, U		
NhanVien	S	S, I, U, D	S	
Kho	S	S, I, U, D		
CT_Kho	S	S, I, U, D		
PhieuNhap	I	S, I, U, D		
CT_PhieuNhap	I	S, I, U, D		
HoaDon	S, I			I
CT_HoaDon	S, I			I

Bảng 26 Phân quyền trên các bảng

4.3.3 Phân quyền trên các giao tác

	Quản lý	Nhân viên kho	CSKH	Phục vụ
Thêm tài khoản	X			
Xóa tài khoản	X			
Cập nhật mật khẩu	X			
Thêm nhân viên	X			
Xóa nhân viên	X			
Cập nhật nhân viên	X			

Thêm món ăn	X	X		
Xóa món ăn	X	X		
Cập nhật món ăn	X	X		
Thêm công thức	X	X	X	
Cập nhật công thức	X	X		
Nhập Kho	X	X		
Thu tiền (Tạo Hóa Đơn)	X		X	X
Thêm khách hàng			X	
Xóa Khách hàng			X	
Cập nhật Khách hàng			X	

Bảng 27 Phân quyền trên các giao tác

4.3.4 Backup

- Full database backup: Mỗi chủ nhật hàng tuần vào lúc 23 giờ.
- Differenitial backup: Mỗi thứ 3, thứ 5, thứ 7 hàng tuần vào lúc 12 giờ
- Logs backups: Mỗi giờ vào phút 5, 15, 25, 35, 45, 55.

Chương 5 TRIỂN KHAI THỰC NGHIỆM

5.1 Phía Server

5.1.1 Cấu trúc các bảng cơ sở dữ liệu

```
CREATE TABLE ChiNhanh (
   Ma_CN char(10) PRIMARY KEY,
   TenCN varchar(50),
   DiaChi varchar(100),
   SoDienThoai nchar(15)
);

CREATE TABLE MonAn (
  Ma_Mon INT IDENTITY(1,1) PRIMARY KEY,
  TenMon NVARCHAR(50),
  HinhAnh NVARCHAR(100),
  GiaBan money
);

CREATE TABLE DVT
(
   Ma_DVT INT IDENTITY(1,1) PRIMARY KEY,
   DonViTinh VARCHAR(50)
);

CREATE TABLE NguyenLieu (
  Ma_NguyenLieu INT IDENTITY(1,1) PRIMARY KEY,
  TenNguyenLieu NVARCHAR(100),
  Ma_DVT INT,
  DonGia money,
  FOREIGN KEY (Ma_DVT) REFERENCES DVT(Ma_DVT)
);


CREATE TABLE CongThuc (
  Ma_Mon INT,
  Ma_NguyenLieu INT,
  SoLuong INT,
  PRIMARY KEY (Ma_Mon, Ma_NguyenLieu),
  FOREIGN KEY (Ma_Mon) REFERENCES MonAn(Ma_Mon),
  FOREIGN KEY (Ma_NguyenLieu) REFERENCES NguyenLieu(Ma_NguyenLieu)
```

```
);

CREATE TABLE KhachHang (
  Ma_KhachHang INT IDENTITY(1,1) PRIMARY KEY,
  HoTen NVARCHAR(50),
  DiaChi NVARCHAR(100),
  SoDienThoai NVARCHAR(20),
  NgayDangKy DateTime,
  DiemTichLuy INT,
  NgaySinh DateTime,
  MatKhauKH VARCHAR(50)
);

CREATE TABLE NhanVien (
  Ma_NhanVien INT IDENTITY(1,1) PRIMARY KEY,
  Ten NVARCHAR(50),
  NgaySinh DateTime,
  SoDienThoai NVARCHAR(20),
  DiaChi NVARCHAR(100),
  Ma_CN char(10),
  NgayVL DateTime,
  Luong money,
  MatKhauNV VARCHAR(50)
  FOREIGN KEY (Ma_CN) REFERENCES ChiNhanh(Ma_CN)
);

CREATE TABLE Kho (
  Ma_Kho INT IDENTITY(1,1) PRIMARY KEY,
  DiaChi NVARCHAR(255),
  Ma_CN char(10),
  FOREIGN KEY (Ma_CN) REFERENCES ChiNhanh(Ma_CN)
);

CREATE TABLE CT_Kho (
  Ma_Kho INT,
  Ma_NguyenLieu INT,
  SoLuongTon INT,
  PRIMARY KEY (Ma_Kho, Ma_NguyenLieu),
  FOREIGN KEY (Ma_Kho) REFERENCES Kho(Ma_Kho),
  FOREIGN KEY (Ma_NguyenLieu) REFERENCES NguyenLieu(Ma_NguyenLieu)
);
```

```
CREATE TABLE PhieuNhap (
  Ma_PhieuNhap INT IDENTITY(1,1) PRIMARY KEY,
  Ma_NV INT,
  Ma_Kho INT,
  NgayNhap DATETIME,
  FOREIGN KEY (Ma_NV) REFERENCES NhanVien(Ma_NhanVien),
  FOREIGN KEY (Ma_Kho) REFERENCES Kho(Ma_Kho)
);

CREATE TABLE CT_PhieuNhap (
  Ma_PhieuNhap INT,
  Ma_NguyenLieu INT,
  SoLuong INT,
  DonGia money,
  PRIMARY KEY (Ma_PhieuNhap, Ma_NguyenLieu),
  FOREIGN KEY (Ma_PhieuNhap) REFERENCES PhieuNhap(Ma_PhieuNhap),
  FOREIGN KEY (Ma_NguyenLieu) REFERENCES NguyenLieu(Ma_NguyenLieu)
);

CREATE TABLE HoaDon (
  Ma_HoaDon INT IDENTITY(1,1) PRIMARY KEY,
  Ma_NV INT,
  TongTien money,
  Ma_CN char(10),
  Ma_KH INT,
  FOREIGN KEY (Ma_NV) REFERENCES NhanVien(Ma_NhanVien),
  FOREIGN KEY (Ma_CN) REFERENCES ChiNhanh(Ma_CN),
  FOREIGN KEY (Ma_KH) REFERENCES KhachHang(Ma_KhachHang)
);

CREATE TABLE CT_HoaDon (
  Ma_HoaDon INT,
  Ma_MonAn INT,
  SoLuong INT,
  PRIMARY KEY (Ma_HoaDon, Ma_MonAn),
  FOREIGN KEY (Ma_HoaDon) REFERENCES HoaDon(Ma_HoaDon),
  FOREIGN KEY (Ma_MonAn) REFERENCES MonAn(Ma_Mon)
);
```

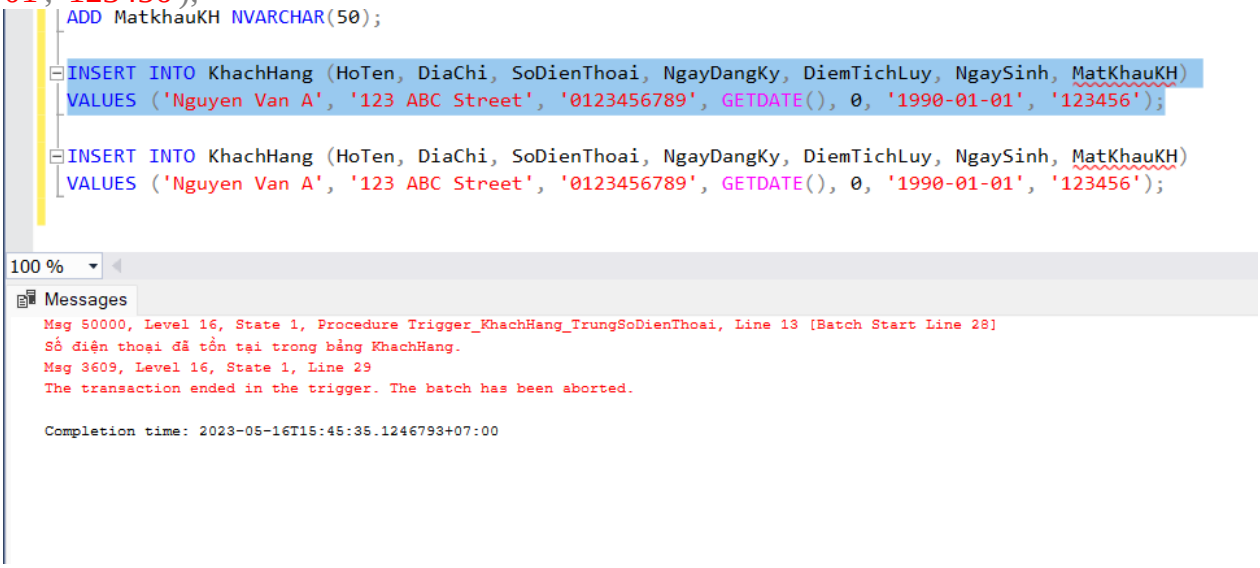
5.1.2 Ràng buộc toàn vẹn

5.1.2.1 SoDienThoai không được trùng

```
CREATE TRIGGER Trigger_KhachHang_TrungSoDienThoai
ON KhachHang
AFTER INSERT, UPDATE
AS
BEGIN
   IF EXISTS (
      SELECT 1
      FROM KhachHang
      GROUP BY SoDienThoai
      HAVING COUNT(*) > 1
   )
   BEGIN
      RAISERROR('Số điện thoại đã tồn tại, vui lòng nhập số khác.', 16, 1);
      ROLLBACK TRANSACTION;
      RETURN;
   END;
END;
```

Ví dụ:

```
INSERT INTO KhachHang (HoTen, DiaChi, SoDienThoai, NgayDangKy,
DiemTichLuy, NgaySinh, MatKhauKH)
VALUES ('Nguyen Van A', '123 ABC Street', '0123456789', GETDATE(), 0, '1990-01-
01', '123456');
```



-Với trigger này, khi nhóm thêm mới 1 khách hàng mới, cập nhật hoặc xóa một KhachHang2, nó sẽ tự động so sánh SoDienThoai đã trùng hay đã đăng ký chưa để số điện thoại không được trùng.

5.1.2.2 Sau khi nhập phiếu nhập và chi tiết nhập kho thì tự động cập nhật vào số lượng tồn kho của Chi tiết kho và kho

```
CREATE TRIGGER tr_AfterInsertPhieuNhap_CT_PhieuNhap
ON CT_PhieuNhap
AFTER INSERT
AS
BEGIN
   SET NOCOUNT ON;

   -- cập nhật in CT_Kho
   UPDATE CT_Kho
   SET CT_Kho.SoLuongTon = CT_Kho.SoLuongTon + inserted.SoLuong
   FROM CT_Kho
   JOIN inserted ON CT_Kho.Ma_NguyenLieu = inserted.Ma_NguyenLieu
   JOIN PhieuNhap ON CT_Kho.Ma_Kho = PhieuNhap.Ma_Kho;

   -- thêm dữ liệu mới CT_Kho khi bên trong kho chưa có
   INSERT INTO CT_Kho (Ma_Kho, Ma_NguyenLieu, SoLuongTon)
   SELECT PhieuNhap.Ma_Kho, inserted.Ma_NguyenLieu, inserted.SoLuong
   FROM inserted
   JOIN CT_PhieuNhap ON CT_PhieuNhap.Ma_PhieuNhap = inserted.Ma_PhieuNhap
   JOIN PhieuNhap ON CT_PhieuNhap.Ma_PhieuNhap = PhieuNhap.Ma_PhieuNhap
   WHERE NOT EXISTS (
      SELECT 1
      FROM CT_Kho
      WHERE CT_Kho.Ma_NguyenLieu = inserted.Ma_NguyenLieu
      AND CT_Kho.Ma_Kho = PhieuNhap.Ma_Kho
   );
END;

-- ví dụ
DECLARE @Ma_PhieuNhap INT;

EXEC sp_ThemPhieuNhap
   @Ma_NV = 1,
   @Ma_Kho = 10,
   @NgayNhap = NULL, -- Cung cấp ngày nhập tự động bằng biến trong stored
procedure
   @Ma_NguyenLieu = 71,
```

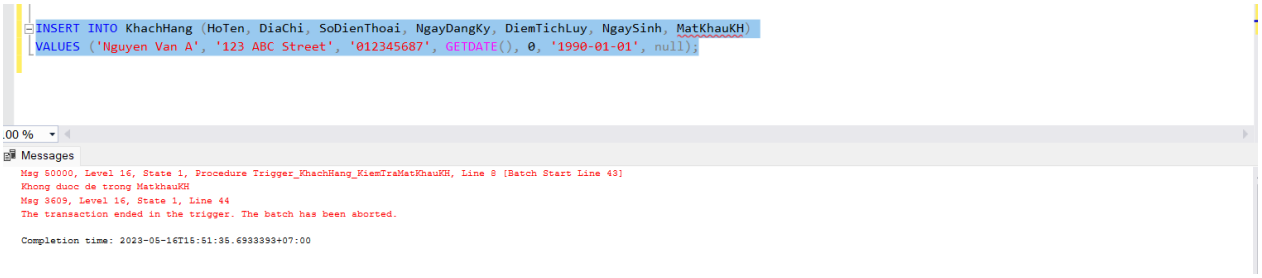
```
    @SoLuong = 10,
    @DonGia = 1000,
    @Ma_PhieuNhap = @Ma_PhieuNhap OUTPUT;

SELECT @Ma_PhieuNhap AS Ma_PhieuNhap;
```

-Với trigger này, khi nhóm thêm mới, cập nhật hoặc xóa một CT_HoaDon, nó sẽ tự động cập nhật lại giá trị SoLuongTon trong bảng CT_Kho theo quy tắc được xác định trong bảng CongThuc.

5.1.2.3 MatKhauKH không được để trống

```
CREATE TRIGGER Trigger_KhachHang_KiemTraMatKhauKH
ON KhachHang
FOR INSERT, UPDATE
AS
BEGIN
   IF EXISTS (SELECT 1 FROM inserted WHERE MatkhauKH IS NULL)
   BEGIN
      RAISERROR ('Khong duoc de trong MatkhauKH', 16, 1)
      ROLLBACK TRANSACTION
   END
END
```

Ví dụ:



-Với trigger này, khi nhóm thêm mới 1 khách hàng mới, cập nhật hoặc xóa một KhachHang2, nó sẽ tự động kiểm tra MatKhauKH không được để trống

5.1.2.4 Không được xóa món ăn khi món ăn liên kết với công thức

```
-- Tạo trigger để kiểm tra ràng buộc toàn vẹn của cơ sở dữ liệu
CREATE TRIGGER Trigger_koduocxoamonan
ON MonAn
AFTER DELETE
AS
BEGIN
```

```
    -- Kiểm tra ràng buộc toàn vẹn
    IF EXISTS (
        SELECT 1
        FROM CongThuc AS CT
        INNER JOIN MonAn AS MA ON CT.Ma_Mon = MA.Ma_Mon
        WHERE MA.Ma_Mon IS NULL
    )
    BEGIN
        -- Nếu tồn tại công thức liên kết với món ăn không tồn tại, hủy bỏ thao tác xóa
        ROLLBACK TRANSACTION;
        RAISERROR('Không thể xóa món ăn có công thức liên kết.', 16, 1);
        RETURN;
    END
END
        --Ví dụ
DELETE FROM MonAn WHERE Ma_Mon = 2;
```

-Với trigger này, khi muốn xóa 1 món ăn thì không thể xóa món ăn đó có công thức liên kết

5.1.2.5 Số lượng món ăn trong chi tiết hóa đơn không được là số âm

```
CREATE TRIGGER Trigger_CT_HoaDon_CheckSoLuong
ON CT_HoaDon
AFTER INSERT, UPDATE
AS
BEGIN
  -- Kiểm tra xem có số lượng sản phẩm trong chi tiết hóa đơn là số âm không
  IF EXISTS (
    SELECT 1
    FROM CT_HoaDon
    WHERE SoLuong < 0
  )
  BEGIN
    RAISERROR('Không thể cập nhật số lượng món ăn.', 16, 1)
    ROLLBACK TRANSACTION
  END
END;
```

-Với trigger này, khi nhóm thêm mới, cập nhật hoặc xóa một CT_HoaDon, nó sẽ tự động kiểm tra số lượng món ăn được thêm không được là số 0.

5.1.2.6 Ngày nhập trong phiếu nhập kho không được nhập ngày trong tương lai

```
CREATE TRIGGER Trigger_PhieuNhap_KiemTraNgayNhap
ON PhieuNhap
FOR INSERT
AS
BEGIN
  DECLARE @NgayNhap DATE;
  SET @NgayNhap = (SELECT NgayNhap FROM inserted);

  IF @NgayNhap > GETDATE()
  BEGIN
    RAISERROR('NgayNhap cannot be in the future.', 16, 1);
    ROLLBACK TRANSACTION;
  END;
END;
```

-Với trigger này, khi nhóm thêm mới, cập nhật hoặc xóa một Phiếu nhập, nó sẽ tự động kiểm tra ngày nhập không được nhập ngày trong tương lai.

5.1.2.7 Không thể thêm món ăn nếu nguyên liệu liên kết với công thức món ăn đã hết

```
CREATE TRIGGER Trigger_CT_HoaDon_KiemTraSoLuong
ON CT_HoaDon
AFTER INSERT, UPDATE
AS
BEGIN
  DECLARE @MaMonAn INT, @SoLuong INT, @SoLuongTon INT;

  SELECT @MaMonAn = Ma_MonAn, @SoLuong = SoLuong
  FROM inserted;

  SELECT @SoLuongTon = SoLuongTon
  FROM CT_Kho
  WHERE Ma_NguyenLieu = @MaMonAn;

  IF @SoLuong > @SoLuongTon
  BEGIN
    RAISERROR('Nguyên liệu đã hết không thể thêm món ăn.', 16, 1);
```

```
    ROLLBACK TRANSACTION;
  END;
```

-Với trigger này, khi nhóm thêm mới, cập nhật hoặc xóa một CT_HoaDon, nó sẽ tự kiểm tra giá trị SoLuongTon trong bảng CT_Kho theo quy tắc được xác định trong bảng CongThuc.Nếu không còn đủ để tạo món ăn thì sẽ không thêm được món ăn

5.1.2.8 Điểm tích lũy của khách hàng không được âm, nếu Khách hàng nào có trên 1000 điểm thì thành VIP

```
CREATE TRIGGER Trigger_KhachHang_KiemTraDiemTichLuy
ON KhachHang
AFTER INSERT, UPDATE
AS
BEGIN
  DECLARE @MaKH INT, @DiemTichLuy INT;

  SELECT @MaKH = Ma_KhachHang, @DiemTichLuy = DiemTichLuy
  FROM inserted;

  IF @DiemTichLuy < 0
  BEGIN
    RAISERROR('DiemTichLuy không thể âm.', 16, 1);
    ROLLBACK TRANSACTION;
  END;

  IF @DiemTichLuy > 1000
  BEGIN
    -- Update loại thành viên của khách hàng thành "VIP"
    UPDATE KhachHang
    SET LoaiThanhVien = 'VIP'
    WHERE Ma_KhachHang = @MaKH;
  END;
END;
```

-Với trigger này, khi nhóm thêm mới, cập nhật HoaDon, nó sẽ tự cập nhật điểm tích lũy theo quy tắc được xác định trong bảng KhachHang.Nếu điểm khách hàng trên 1000 thì se là vip

5.1.2.9 Khách hàng phải đủ 16 trở lên mới được thêm mới

```
CREATE TRIGGER Trigger_KiemTraTuoiKhachHang
ON KhachHang
```

```
AFTER INSERT, UPDATE
AS
BEGIN
   -- Lấy ngày hiện tại
   DECLARE @NgayHienTai DATE = GETDATE();

   -- Kiểm tra tuổi của khách hàng
   IF EXISTS (
      SELECT 1
      FROM inserted
      WHERE DATEDIFF(YEAR, NgaySinh, @NgayHienTai) < 16
   )
   BEGIN
      -- Nếu có khách hàng dưới 16 tuổi, không cho phép thêm mới hoặc cập nhật thông tin
      RAISERROR('Khách hàng phải đủ 16 tuổi trở lên.', 16, 1);
      ROLLBACK TRANSACTION;
   END;
END;
```

-Với trigger này, khi nhóm thêm mới, cập nhật một khách hàng trong bảng KhachHang.Nếu khách hàng đó trên 16 tuổi mới được đăng ký

5.1.2.10 Nhân viên phải đủ 18 tuổi trở lên mới được làm việc

```
CREATE TRIGGER Trigger_KiemTraTuoiNhanVien
ON NhanVien
AFTER INSERT, UPDATE
AS
BEGIN
   -- Kiểm tra tuổi của nhân viên
   IF EXISTS (
      SELECT 1
      FROM inserted
      WHERE DATEDIFF(YEAR, NgaySinh, GETDATE()) < 18
   )
   BEGIN
      -- Nếu tuổi nhân viên dưới 18, không cho phép thêm hoặc cập nhật thông tin
      RAISERROR('Tuổi nhân viên phải từ 18 trở lên.', 16, 1);
      ROLLBACK TRANSACTION;
   END;
END;
```

-Với trigger này, khi nhóm thêm mới, cập nhật một nhân viên trong bảng Nhân Viên.Nếu nhân viên đó trên 18 tuổi mới được đăng ký

5.1.2.11 Đơn giá của nguyên liệu phải lớn hơn 0

```
-- Trigger để kiểm tra ràng buộc DonGia của Ma_NguyenLieu
CREATE TRIGGER Trigger_KiemTraDonGia
ON NguyenLieu
AFTER INSERT, UPDATE
AS
BEGIN
   -- Kiểm tra ràng buộc DonGia của Ma_NguyenLieu
   IF EXISTS (
      SELECT 1
      FROM NguyenLieu AS nl
      INNER JOIN inserted AS i ON nl.Ma_NguyenLieu = i.Ma_NguyenLieu
      WHERE nl.DonGia < 0
   )
   BEGIN
      -- Nếu DonGia là số âm, hủy bỏ thao tác INSERT hoặc UPDATE
      ROLLBACK;
      RAISERROR ('DonGia phai lon hon hoac bang 0', 16, 1);
      RETURN;
   END;
END;
```

-Với trigger này, khi nhóm thêm mới, cập nhật một nguyên liệu trong bảng nguyên liệu. Giá nguyên liệu đó phải trên 0 mới được thêm vào

5.1.2.12 Giá bán của món ăn phải lớn hơn 0

```
CREATE TRIGGER Trigger_CheckGiaBan
ON MonAn
AFTER INSERT, UPDATE
AS
BEGIN
   IF EXISTS (SELECT * FROM inserted WHERE GiaBan < 0)
   BEGIN
      RAISERROR ('Gía Bán phải lớn 0.', 16, 1);
-- Nếu GiaBan là số âm, hủy bỏ thao tác INSERT hoặc UPDATE
      ROLLBACK TRANSACTION;
      RETURN;
   END
END;;
```

-Với trigger này, khi nhóm thêm mới, cập nhật một món ăn trong bảng món ăn.Giá của món ăn đó phải trên 0 mới được thêm vào.

5.1.2.13 Tính lại tổng tiền hóa đơn sau khi thêm, cập nhật, xóa chi tiết hóa đơn

```
CREATE TRIGGER trg_TinhTongTien_CT_HoaDon
ON CT_HoaDon
AFTER INSERT, UPDATE, DELETE
AS
BEGIN
   SET NOCOUNT ON;

   -- Cập nhật TongTien trong HoaDon khi có thay đổi trong CT_HoaDon
   UPDATE HoaDon
   SET TongTien = (
      SELECT SUM(ct.SoLuong * ma.GiaBan)
      FROM CT_HoaDon AS ct
      JOIN MonAn AS ma ON ct.Ma_MonAn = ma.Ma_Mon
      WHERE ct.Ma_HoaDon = HoaDon.Ma_HoaDon
   )
   WHERE Ma_HoaDon IN (
      SELECT Ma_HoaDon
      FROM inserted
      UNION
      SELECT Ma_HoaDon
      FROM deleted
   );
END
```

-Với trigger này, khi nhóm thêm mới, cập nhật hoặc xóa một CT_HoaDon, nó sẽ tự động cập nhật lại giá trị Tongtien trong bảng HoaDon theo quy tắc được xác định trong bảng Monan.

5.1.2.14 Tự động cập nhật điểm tích lũy cho Khách Hàng mỗi khi Khách hàng có đơn đặt mới

```
CREATE TRIGGER UpdateDiemTichLuy
ON HoaDon
AFTER INSERT
AS
BEGIN
   -- Lấy thông tin về khách hàng và tổng số tiền trong hóa đơn mới
   DECLARE @Ma_KhachHang INT, @TongTien MONEY;
   SELECT @Ma_KhachHang = Ma_KH, @TongTien = TongTien
   FROM inserted;
```

```
    -- Cập nhật điểm tích lũy cho khách hàng
    UPDATE KhachHang
    SET DiemTichLuy = DiemTichLuy + (@TongTien / 10000)
    WHERE Ma_KhachHang = @Ma_KhachHang;
END;
```

5.1.3 Các chức năng

5.1.3.1 Đăng nhập Khách hàng

```
CREATE PROCEDURE DangNhapKH
    @Username1 NVARCHAR(50),
    @Password1 NVARCHAR(50)
AS
BEGIN
    SET NOCOUNT ON;
    -- Kiểm tra sự tồn tại của tài khoản
    IF EXISTS (SELECT * FROM KhachHang WHERE SoDienThoai = @Username1
AND MatkhauKH = @Password1)
    BEGIN
        -- Lấy thông tin nhân viên
        SELECT Ma_KhachHang, HoTen, DiaChi, , NgayDangKy, DiemTichLuy,
NgaySinh FROM KhachHang WHERE SoDienThoai = @Username1;
    END
    ELSE
    BEGIN
        -- Trả về lỗi nếu tài khoản không tồn tại
        RAISERROR ('tài khoản không tồn tại.', 16, 1);
    END
END
--ví dụ
EXEC DangNhapKH @Username = '123456789', @Password = '1234567';
```

5.1.3.2 Tạo tài khoản Khách hàng

```
CREATE PROCEDURE sp_DangKyTaiKhoan
    @HoTen NVARCHAR(50),
    @DiaChi NVARCHAR(100),
    @SoDienThoai NVARCHAR(20),
    @NgayDangKy DATETIME,
    @NgaySinh DATETIME,
    @MatKhau NVARCHAR(50)
```

```
AS
BEGIN
   SET NOCOUNT ON;

   -- Kiểm tra xem khách hàng đã tồn tại chưa
   IF EXISTS (SELECT 1 FROM KhachHang WHERE SoDienThoai = @SoDienThoai)
   BEGIN
      RAISERROR ('Số điện thoại đã được đăng ký!', 16, 1)
      RETURN
   END

   -- Thêm khách hàng mới vào bảng KhachHang
   INSERT INTO KhachHang (HoTen, DiaChi, SoDienThoai, NgayDangKy, NgaySinh,
MatKhauKH)
   VALUES (@HoTen, @DiaChi, @SoDienThoai, @NgayDangKy, @NgaySinh,
@MatKhau)

   SELECT SCOPE_IDENTITY() AS Ma_KhachHang
END
--ví dụ
DECLARE @Ma_KhachHang INT;
DECLARE @NgayDangKy DATETIME;

SET @NgayDangKy = GETDATE();

EXEC sp_DangKyTaiKhoan
   @HoTen = N'Vũ Đức Dũng',
   @DiaChi = N'ok/12/12',
   @SoDienThoai = N'0936588378',
   @NgayDangKy = @NgayDangKy,
   @NgaySinh = '1990-01-01',
   @MatKhau = N'123456789';


SELECT @Ma_KhachHang AS Ma_KhachHang;
```

5.1.3.3 Xóa tài khoản Khách hàng

```
CREATE PROCEDURE sp_XoaTaiKhoanKhachHang
   @SoDienThoai NVARCHAR(20)
AS
BEGIN
   SET NOCOUNT ON;
```

```
   -- Kiểm tra xem Khách hàng tồn tại không
   IF NOT EXISTS (SELECT 1 FROM KhachHang WHERE SoDienThoai =
@SoDienThoai)
   BEGIN
      RAISERROR ('Khách hàng không tồn tại!', 16, 1)
      RETURN
   END

   -- Xóa Khách hàng từ bảng KhachHang
   DELETE FROM KhachHang WHERE SoDienThoai = @SoDienThoai

   SELECT 'Xóa tài khoản Khách hàng thành công' AS ThongBao
END
--- ví dụ
DECLARE @SoDienThoai NVARCHAR(20);

EXEC sp_XoaTaiKhoanKhachHang
   @SoDienThoai=N'0936588378';
```

5.1.3.4 Cập nhật mật khẩu của tài khoản phải nhớ số điện thoại

```
CREATE PROCEDURE sp_CapNhatMatKhauKhachHang
   @SoDienThoai NVARCHAR(20),
   @MatKhau NVARCHAR(50)
AS
BEGIN
   SET NOCOUNT ON;

   -- Kiểm tra xem Khách hàng tồn tại không
   IF NOT EXISTS (SELECT 1 FROM KhachHang WHERE SoDienThoai =
@SoDienThoai)
   BEGIN
      RAISERROR ('Khách hàng không tồn tại!', 16, 1)
      RETURN
   END

   -- Cập nhật mật khẩu Khách hàng
   UPDATE KhachHang
   SET MatKhauKH = @MatKhau
   WHERE SoDienThoai = @SoDienThoai

   SELECT 'Cập nhật mật khẩu Khách hàng thành công' AS ThongBao
```

```
END
------------ví dụ
EXEC sp_CapNhatMatKhauKhachHang
   @SoDienThoai = N'0936588378',
   @MatKhau = N'090909';
```

5.1.3.5 Cập nhật thông tin của tài khoản phải có số điện thoại và mật khẩu

```
CREATE PROCEDURE sp_CapNhatThongTinKhachHang
   @SoDienThoai NVARCHAR(20),
   @MatKhau NVARCHAR(50),
   @HoTen NVARCHAR(50),
   @DiaChi NVARCHAR(100),
   @NgaySinh DATETIME
AS
BEGIN
   SET NOCOUNT ON;

   -- Kiểm tra xem số điện thoại và mật khẩu có chính xác không
   IF NOT EXISTS (SELECT 1 FROM KhachHang WHERE SoDienThoai =
@SoDienThoai AND MatKhauKH = @MatKhau)
   BEGIN
      RAISERROR ('Số điện thoại hoặc mật khẩu không chính xác!', 16, 1)
      RETURN
   END

   -- Cập nhật thông tin khách hàng
   UPDATE KhachHang
   SET HoTen = @HoTen, DiaChi = @DiaChi, NgaySinh = @NgaySinh
   WHERE SoDienThoai = @SoDienThoai;

   SELECT * FROM KhachHang WHERE SoDienThoai = @SoDienThoai;
END;
----ví dụ
EXEC sp_CapNhatThongTinKhachHang
   @SoDienThoai = N'0936588378',
   @MatKhau = N'123456789',
   @HoTen = N'Tên khách hàng mới',
   @DiaChi = N'Địa chỉ khách hàng mới',
   @NgaySinh = '1990-01-01';
```

5.1.3.6 Xem danh sách tài khoản khách hàng

```
CREATE PROCEDURE LayDanhsachKH
AS
BEGIN
   SELECT * FROM KhachHang;
END;

EXEC LayDanhsachKH;
```

5.1.3.7 Đăng nhập Nhân Viên

```
CREATE PROCEDURE DangNhapNV
   @Username NVARCHAR(50),
   @Password NVARCHAR(50)
AS
BEGIN
   SET NOCOUNT ON;

   -- Kiểm tra sự tồn tại của tài khoản
   IF EXISTS (SELECT * FROM NhanVien WHERE SoDienThoai = @Username AND
MatKhauNV = @Password)
   BEGIN
      -- Lấy thông tin nhân viên
      SELECT Ma_NhanVien, Ten, Ma_CN FROM NhanVien WHERE SoDienThoai =
@Username;
   END
   ELSE
   BEGIN
      -- Trả về lỗi nếu tài khoản không tồn tại
      RAISERROR ('tài khoản không tồn tại.', 16, 1);
   END
END
--ví dụ
EXEC DangNhapNV @Username = '123456789', @Password = '1234567';
```

5.1.3.8 Tạo tài khoản Nhân Viên

```
CREATE PROCEDURE sp_DangKyTaiKhoanNhanVien
   @Ten NVARCHAR(50),
   @NgaySinh DATETIME,
   @SoDienThoai NVARCHAR(20),
   @DiaChi NVARCHAR(100),
```

```
    @Ma_CN CHAR(10),
    @NgayVL DATETIME,
    @Luong MONEY,
    @MatKhau NVARCHAR(50)
AS
BEGIN
    SET NOCOUNT ON;

    -- Kiểm tra xem Nhân viên đã tồn tại chưa
    IF EXISTS (SELECT 1 FROM NhanVien WHERE SoDienThoai = @SoDienThoai)
    BEGIN
        RAISERROR ('Số điện thoại đã được đăng ký!', 16, 1)
        RETURN
    END

    -- Thêm Nhân viên mới vào bảng NhanVien
    INSERT INTO NhanVien (Ten, NgaySinh, SoDienThoai, DiaChi, Ma_CN, NgayVL,
Luong, MatKhauNV)
    VALUES (@Ten, @NgaySinh, @SoDienThoai, @DiaChi, @Ma_CN, @NgayVL,
@Luong, @MatKhau)

    SELECT SCOPE_IDENTITY() AS Ma_NhanVien
END
---- ví dụ
DECLARE @Ma_NhanVien INT;
DECLARE @NgayVL DATETIME;

SET @NgayVL = GETDATE();
EXEC sp_DangKyTaiKhoanNhanVien
    @Ten = N'Hoàng Tho',
    @NgaySinh = '1990-01-01',
    @SoDienThoai = N'0936588378',
    @DiaChi = N'Địa chỉ nhân viên',
    @Ma_CN = 'ABC123',
    @NgayVL = @NgayVL,
    @Luong = 10000000,
    @MatKhau = N'123456789';

SELECT @Ma_NhanVien AS Ma_NhanVien;
```

5.1.3.9 Cập nhật mật khẩu nhân viên

```
CREATE PROCEDURE sp_CapNhatMatKhauNhanVien
   @SoDienThoai NVARCHAR(20),
   @MatKhau NVARCHAR(50)
AS
BEGIN
   SET NOCOUNT ON;

   -- Kiểm tra xem Nhân viên tồn tại không
   IF NOT EXISTS (SELECT 1 FROM NhanVien WHERE SoDienThoai =
@SoDienThoai)
   BEGIN
      RAISERROR ('Nhân viên không tồn tại!', 16, 1)
      RETURN
   END

   -- Cập nhật mật khẩu Nhân viên
   UPDATE NhanVien
   SET MatKhauNV = @MatKhau
   WHERE SoDienThoai = @SoDienThoai

   SELECT 'Cập nhật mật khẩu Nhân viên thành công' AS ThongBao
END
---ví dụ
EXEC sp_CapNhatMatKhauNhanVien
   @SoDienThoai = N'0936588378',
   @MatKhau = N'1234456';
```

5.1.3.10 Cập nhật thông tin nhân viên với yêu cầu phải biết số điện thoại và mật khẩu

```
CREATE PROCEDURE sp_CapNhatThongTinNhanVien
   @SoDienThoai NVARCHAR(20),
   @MatKhau NVARCHAR(50),
   @Ten NVARCHAR(50),
   @NgaySinh DATETIME,
   @DiaChi NVARCHAR(100)
AS
BEGIN
   SET NOCOUNT ON;

   -- Kiểm tra xem số điện thoại và mật khẩu có chính xác không
```

```
    IF NOT EXISTS (SELECT 1 FROM NhanVien WHERE SoDienThoai =
@SoDienThoai AND MatKhauNV = @MatKhau)
    BEGIN
        RAISERROR ('Số điện thoại hoặc mật khẩu không chính xác!', 16, 1)
        RETURN
    END

    -- Cập nhật thông tin nhân viên
    UPDATE NhanVien
    SET Ten = @Ten, NgaySinh = @NgaySinh, DiaChi = @DiaChi
    WHERE SoDienThoai = @SoDienThoai;

    SELECT * FROM NhanVien WHERE SoDienThoai = @SoDienThoai;
END;
--------------ví dụ
EXEC sp_CapNhatThongTinNhanVien
    @SoDienThoai = N'0987654321',
    @MatKhau = N'password123',
    @Ten = N'Tên nhân viên mới',
    @NgaySinh = '1990-01-01',
    @DiaChi = N'Địa chỉ nhân viên mới';
```

5.1.3.11 Xem thông tin nhân viên

```
CREATE PROCEDURE sp_XemThongTinNhanVien
    @SoDienThoai NVARCHAR(20)
AS
BEGIN
    SET NOCOUNT ON;

    -- Kiểm tra xem số điện thoại có tồn tại không
    IF NOT EXISTS (SELECT 1 FROM NhanVien WHERE SoDienThoai =
@SoDienThoai)
    BEGIN
        RAISERROR ('Nhân viên không tồn tại!', 16, 1);
        RETURN;
    END;
    -- Lấy thông tin của Nhân viên
    SELECT Ma_NhanVien, Ten, NgaySinh, SoDienThoai, DiaChi, Ma_CN, NgayVL,
Luong
    FROM NhanVien
    WHERE SoDienThoai = @SoDienThoai;
END;
```

```
-----ví dụ
EXEC sp_XemThongTinNhanVien
   @SoDienThoai = N'0936588378';
```

5.1.3.12 Xem danh sách nhân viên

```
CREATE PROCEDURE sp_XemDanhSachNhanVien
AS
BEGIN
   SET NOCOUNT ON;

   -- Lấy danh sách nhân viên
   SELECT Ma_NhanVien, Ten, NgaySinh, SoDienThoai, DiaChi, Ma_CN, NgayVL,
Luong
   FROM NhanVien;
END;
---------------ví dụ
exec sp_XemDanhSachNhanVien
```

5.1.3.13 Xem danh sách món ăn

```
CREATE PROCEDURE sp_XemDanhSachMonAn
AS
BEGIN
   SET NOCOUNT ON;

   -- Lấy danh sách món ăn
   SELECT * FROM MonAn;
END;
----ví dụ
EXEC sp_XemDanhSachMonAn;
```

5.1.3.14 Xem thông tin món ăn thông qua tên món ăn

```
CREATE PROCEDURE sp_XemThongTinMonAn
   @TenMon NVARCHAR(50)
AS
BEGIN
   SET NOCOUNT ON;

   -- Kiểm tra xem món ăn có tồn tại hay không
   IF NOT EXISTS (SELECT 1 FROM MonAn WHERE TenMon = @TenMon)
   BEGIN
      SELECT 'Món ăn không tồn tại' AS ThongBao;
```

```
        RETURN;
    END

    -- Lấy thông tin món ăn dựa trên tên món
    SELECT * FROM MonAn WHERE TenMon = @TenMon;
END;
------ví dụ
EXEC sp_XemThongTinMonAn @TenMon = N'Tên món';
```

5.1.3.15 Thêm món ăn mới và công thức liên kết

```
CREATE PROCEDURE sp_ThemMonAn
    @TenMon NVARCHAR(50),
    @HinhAnh NVARCHAR(100),
    @GiaBan MONEY,
    @MaNguyenLieu INT,
    @SoLuong INT
AS
BEGIN
    SET NOCOUNT ON;

    -- Thêm món ăn vào bảng MonAn
    INSERT INTO MonAn (TenMon, HinhAnh, GiaBan)
    VALUES (@TenMon, @HinhAnh, @GiaBan);

    -- Lấy mã món ăn vừa thêm
    DECLARE @Ma_Mon INT;
    SET @Ma_Mon = SCOPE_IDENTITY();

    -- Thêm công thức liên kết vào bảng CongThuc
    INSERT INTO CongThuc (Ma_Mon, Ma_NguyenLieu, SoLuong)
    VALUES (@Ma_Mon, @MaNguyenLieu, @SoLuong);

    SELECT @Ma_Mon AS MaMonAn; -- Trả về mã món ăn vừa thêm
END;
-----ví dụ
DECLARE @MaNguyenLieu INT = 1;
DECLARE @SoLuong INT = 2;

EXEC sp_ThemMonAn
    @TenMon = N'Món A',
    @HinhAnh = N'hinhanh.jpg',
    @GiaBan = 10000,
```

```
    @MaNguyenLieu = @MaNguyenLieu,
    @SoLuong = @SoLuong;
```

5.1.3.16 Thêm nguyên liệu mới

```
CREATE PROCEDURE sp_ThemNguyenLieu
    @TenNguyenLieu NVARCHAR(100),
    @MaDVT INT,
    @DonGia MONEY
AS
BEGIN
    SET NOCOUNT ON;

    -- Kiểm tra xem đơn vị tính có tồn tại không
    IF NOT EXISTS (SELECT 1 FROM DVT WHERE Ma_DVT = @MaDVT)
    BEGIN
        RAISERROR ('Đơn vị tính không tồn tại!', 16, 1)
        RETURN
    END

    BEGIN TRY
        -- Thêm nguyên liệu mới vào bảng NguyenLieu
        INSERT INTO NguyenLieu (TenNguyenLieu, Ma_DVT, DonGia)
        VALUES (@TenNguyenLieu, @MaDVT, @DonGia)

        -- Lấy thông tin nguyên liệu vừa thêm
        DECLARE @MaNguyenLieu INT;
        SET @MaNguyenLieu = SCOPE_IDENTITY();

        -- Kiểm tra xem nguyên liệu đã được thêm thành công hay không
        IF EXISTS (SELECT 1 FROM NguyenLieu WHERE Ma_NguyenLieu =
@MaNguyenLieu)
        BEGIN
            SELECT @MaNguyenLieu AS Ma_NguyenLieu, TenNguyenLieu, Ma_DVT,
DonGia
            FROM NguyenLieu
            WHERE Ma_NguyenLieu = @MaNguyenLieu;

            PRINT 'Thêm nguyên liệu thành công!'
        END
        ELSE
        BEGIN
            PRINT 'Thêm nguyên liệu thất bại!'
```

```
    END
  END TRY
  BEGIN CATCH
    PRINT 'Thêm nguyên liệu thất bại!'
  END CATCH
END

---------- ví dụ
EXEC sp_ThemNguyenLieu
  @TenNguyenLieu = N'gà ác',
  @MaDVT = 11,
  @DonGia = 10000;
```

5.1.3.17 Thêm phiếu nhập và chi tiết phiếu nhập

```
CREATE PROCEDURE sp_ThemPhieuNhap
  @Ma_NV INT,
  @Ma_Kho INT,
  @NgayNhap DATETIME,
  @Ma_NguyenLieu INT,
  @SoLuong INT,
  @DonGia MONEY,
  @Ma_PhieuNhap INT OUTPUT
AS
BEGIN
  SET NOCOUNT ON;

  DECLARE @NgayHienTai DATETIME;
  SET @NgayHienTai = GETDATE();

  -- Thêm phiếu nhập mới vào bảng PhieuNhap
  INSERT INTO PhieuNhap (Ma_NV, Ma_Kho, NgayNhap)
  VALUES (@Ma_NV, @Ma_Kho, @NgayHienTai)

  -- Lấy mã phiếu nhập vừa được thêm vào
  SET @Ma_PhieuNhap = SCOPE_IDENTITY()

  -- Thêm chi tiết phiếu nhập vào bảng CT_PhieuNhap
  INSERT INTO CT_PhieuNhap (Ma_PhieuNhap, Ma_NguyenLieu, SoLuong, DonGia)
  VALUES (@Ma_PhieuNhap, @Ma_NguyenLieu, @SoLuong, @DonGia)

  -- Hiển thị thông báo thành công và trả về thông tin phiếu nhập
```

```
    SELECT 'Thêm phiếu nhập thành công.' AS ThongBao,
        @Ma_PhieuNhap AS Ma_PhieuNhap,
        @Ma_NV AS Ma_NV,
        @Ma_Kho AS Ma_Kho,
        @NgayHienTai AS NgayNhap,
        @Ma_NguyenLieu AS Ma_NguyenLieu,
        @SoLuong AS SoLuong,
        @DonGia AS DonGia
END

----------------------ví dụ
DECLARE @Ma_PhieuNhap INT;

EXEC sp_ThemPhieuNhap
    @Ma_NV = 1,
    @Ma_Kho = 10,
    @NgayNhap = NULL, -- Cung cấp ngày nhập tự động bằng biến trong stored
procedure
    @Ma_NguyenLieu = 71,
    @SoLuong = 10,
    @DonGia = 1000,
    @Ma_PhieuNhap = @Ma_PhieuNhap OUTPUT;

SELECT @Ma_PhieuNhap AS Ma_PhieuNhap;
```

5.1.3.18 Thêm hóa đơn mới

```
CREATE PROCEDURE sp_ThemHoaDon
    @Ma_NV INT,
    @Ma_CN CHAR(10),
    @Ma_KH INT
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @TongTien MONEY = 0; -- Giá trị mặc định là 0

    INSERT INTO HoaDon (Ma_NV, TongTien, Ma_CN, Ma_KH )
    VALUES (@Ma_NV, @TongTien, @Ma_CN, @Ma_KH );

    SELECT H.Ma_HoaDon, H.Ma_NV, H.TongTien, H.Ma_CN, H.Ma_KH, N.Ten AS
TenNV, C.TenCN, KH.HoTen AS HoTenKH
    FROM HoaDon H
```

```
    INNER JOIN NhanVien N ON H.Ma_NV = N.Ma_NhanVien
    INNER JOIN ChiNhanh C ON H.Ma_CN = C.Ma_CN
    INNER JOIN KhachHang KH ON H.Ma_KH = KH.Ma_KhachHang
    WHERE H.Ma_HoaDon = SCOPE_IDENTITY();
END
---ví dụ
EXEC sp_ThemHoaDon
    @Ma_NV = 1,
    @Ma_CN = 'CN1',
    @Ma_KH = 9;
```

5.1.3.19 Thêm chi tiết hóa đơn

```
CREATE PROCEDURE sp_ThemCT_HoaDon
    @Ma_HoaDon INT,
    @Ma_MonAn INT,
    @SoLuong INT
AS
BEGIN
    -- Kiểm tra nếu Ma_MonAn và Ma_HoaDon đã tồn tại trong bảng CT_HoaDon
    IF EXISTS (
        SELECT 1
        FROM CT_HoaDon
        WHERE Ma_MonAn = @Ma_MonAn AND Ma_HoaDon = @Ma_HoaDon
    )
    BEGIN
        -- Cập nhật SoLuong mới bằng cách cộng dồn với SoLuong cũ
        UPDATE CT_HoaDon
        SET SoLuong = SoLuong + @SoLuong
        WHERE Ma_MonAn = @Ma_MonAn AND Ma_HoaDon = @Ma_HoaDon
    END
    ELSE
    BEGIN
        -- Thêm bản ghi mới vào bảng CT_HoaDon
        INSERT INTO CT_HoaDon (Ma_HoaDon, Ma_MonAn, SoLuong)
        VALUES (@Ma_HoaDon, @Ma_MonAn, @SoLuong)
    END
END

----ví dụ
EXEC sp_ThemCT_HoaDon @Ma_HoaDon = 1, @Ma_MonAn = 23, @SoLuong = 5;
```

5.1.3.20 Cập nhật hóa đơn

```
CREATE PROCEDURE sp_Sua_CT_HoaDon
   @Ma_HoaDon INT,
   @Ma_MonAn INT,
   @SoLuong INT
AS
BEGIN
   SET NOCOUNT ON;

   -- Kiểm tra xem Ma_HoaDon và Ma_MonAn đã tồn tại trong CT_HoaDon hay không
   IF EXISTS (
      SELECT 1
      FROM CT_HoaDon
      WHERE Ma_HoaDon = @Ma_HoaDon
         AND Ma_MonAn = @Ma_MonAn
   )
   BEGIN
      -- Kiểm tra nếu SoLuong mới là 0 thì xóa bản ghi trong CT_HoaDon
      IF @SoLuong = 0
      BEGIN
         DELETE FROM CT_HoaDon
         WHERE Ma_HoaDon = @Ma_HoaDon
            AND Ma_MonAn = @Ma_MonAn;
      END
      ELSE
      BEGIN
         -- Sửa đổi SoLuong trong CT_HoaDon
         UPDATE CT_HoaDon
         SET SoLuong = @SoLuong
         WHERE Ma_HoaDon = @Ma_HoaDon
            AND Ma_MonAn = @Ma_MonAn;
      END
   END
END
----------ví dụ
EXEC sp_Sua_CT_HoaDon @Ma_HoaDon = 1, @Ma_MonAn = 23, @SoLuong = 0;
```

5.1.3.21 Xóa chi tiết hóa đơn

```
CREATE PROCEDURE sp_XoaCT_HoaDon
   @Ma_HoaDon INT,
```

```
    @Ma_MonAn INT
AS
BEGIN
    -- Kiểm tra sự tồn tại của bản ghi trong bảng CT_HoaDon dựa trên Ma_HoaDon
và Ma_MonAn
    IF EXISTS (
        SELECT 1
        FROM CT_HoaDon
        WHERE Ma_HoaDon = @Ma_HoaDon AND Ma_MonAn = @Ma_MonAn
    )
    BEGIN
        -- Xóa bản ghi trong bảng CT_HoaDon dựa trên Ma_HoaDon và Ma_MonAn
        DELETE FROM CT_HoaDon
        WHERE Ma_HoaDon = @Ma_HoaDon AND Ma_MonAn = @Ma_MonAn

        -- Thông báo xóa thành công
        PRINT N'Xóa CT_HoaDon thành công.'
    END
    ELSE
    BEGIN
        -- Thông báo xóa không thành công khi bản ghi không tồn tại
        PRINT N'Xóa CT_HoaDon không thành công. Ma_HoaDon hoặc Ma_MonAn
không hợp lệ.'
    END
END
----ví dụ
EXEC sp_XoaCT_HoaDon @Ma_HoaDon = 1, @Ma_MonAn = 21
```

5.1.3.22 Tính tiền cho một hóa đơn và hiện thị thông tin khách hàng của hóa đơn đó

```
CREATE FUNCTION fn_TinhTongTienVaThongTinKhachHang
(
    @MaHoaDon INT
)
RETURNS TABLE
AS
RETURN
(
    SELECT HD.Ma_HoaDon, HD.TongTien, KH.HoTen, KH.DiaChi, KH.SoDienThoai
    FROM HoaDon AS HD
    INNER JOIN KhachHang AS KH ON HD.Ma_KH = KH.Ma_KhachHang
    WHERE HD.Ma_HoaDon = @MaHoaDon
);
```

```
--- ví dụ
SELECT *
FROM dbo.fn_TinhTongTienVaThongTinKhachHang(1); -- Thay đổi giá trị 1 thành mã hóa đơn cần tìm
```

5.1.3.23 Xem tổng số lượng tồn của nguyên liệu ở các kho

```
CREATE VIEW View_NguyenLieu_ChiNhanh AS
SELECT NL.TenNguyenLieu, NL.DonGia, DVT.DonViTinh, CT.SoLuongTon,
CN.TenCN, K.DiaChi
FROM NguyenLieu NL
INNER JOIN DVT ON NL.Ma_DVT = DVT.Ma_DVT
INNER JOIN CT_Kho CT ON NL.Ma_NguyenLieu = CT.Ma_NguyenLieu
INNER JOIN Kho K ON CT.Ma_Kho = K.Ma_Kho
INNER JOIN ChiNhanh CN ON K.Ma_CN = CN.Ma_CN;

---ví dụ
SELECT * FROM View_NguyenLieu_ChiNhanh;
```

5.1.3.24 Hiển thị thông tin về khách hàng và số lượng đơn hàng đã đặt

```
CREATE VIEW View_KhachHang_DonHang AS
SELECT KH.Ma_KhachHang, KH.HoTen, KH.DiaChi, KH.SoDienThoai,
KH.DiemTichLuy, COUNT(HD.Ma_HoaDon) AS SoLuongDonHang,
SUM(HD.TongTien) AS TongTienDonHang
FROM KhachHang KH
LEFT JOIN HoaDon HD ON KH.Ma_KhachHang = HD.Ma_KH
GROUP BY KH.Ma_KhachHang, KH.HoTen, KH.DiaChi, KH.SoDienThoai,
KH.DiemTichLuy;
---ví dụ
SELECT * FROM View_KhachHang_DonHang;
```

5.1.3.25 Hiển thị thông tin món ăn được mua nhiều nhất và số lượng đã bán

```
CREATE VIEW View_MonAn_BestSelling AS
SELECT TOP 1 M.Ma_Mon, M.TenMon, M.GiaBan, SUM(CT.SoLuong) AS
SoLuongBan, SUM(CT.SoLuong * M.GiaBan) AS TongTienBan
FROM MonAn M
JOIN CT_HoaDon CT ON M.Ma_Mon = CT.Ma_MonAn
GROUP BY M.Ma_Mon, M.TenMon, M.GiaBan
ORDER BY SUM(CT.SoLuong) DESC;
```

```
---ví dụ
SELECT * FROM View_MonAn_BestSelling;
```

5.1.3.26 Hiện thị thông tin của các chi nhánh và doanh thu của chi nhánh đó

```
CREATE VIEW View_ChiNhanh_TongTienBan AS
SELECT CN.Ma_CN, CN.TenCN, CN.DiaChi, CN.SoDienThoai, SUM(HD.TongTien)
AS TongTienBan
FROM ChiNhanh CN
LEFT JOIN HoaDon HD ON CN.Ma_CN = HD.Ma_CN
GROUP BY CN.Ma_CN, CN.TenCN, CN.DiaChi, CN.SoDienThoai;

-----ví dụ
SELECT * FROM View_ChiNhanh_TongTienBan;
```

5.1.4 Phân quyền

5.1.4.1 Tạo login mặc định

```
IF EXISTS(SELECT * FROM master.dbo.syslogins WHERE NAME =
'DungTho_Client')
        DROP LOGIN DungTho_Client
CREATE LOGIN DungTho_Client WITH PASSWORD = '123456'
GO
CREATE USER DTClient FOR LOGIN DungTho_Client
GO
```

5.1.4.2 Tạo Roles

```
CREATE ROLE DT_QuanLy
GO
CREATE ROLE DT_NhanVienKho
GO
CREATE ROLE DT_NhanVienCSKH
GO
CREATE ROLE DT_NhanVienPV
GO
CREATE ROLE DT_KhachHang
GO
```

5.1.4.3 Cấp quyền cho các Roles

```
GRANT CONTROL ON DATABASE::BAOCAOCHCSDL_FOOD TO DT_QuanLy
```

```
GRANT SELECT ON KhachHang TO DTClient
----------gắn role cho user
ALTER USER DTClient WITH DEFAULT_SCHEMA = dbo;
ALTER ROLE DT_QuanLy ADD MEMBER DTClient;


ALTER USER DT1Client WITH DEFAULT_SCHEMA = dbo;
ALTER ROLE DT_NhanVienKho ADD MEMBER DTClient;
------ phân quyền
-- Phân quyền cho vai trò DT_QuanLy
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.ChiNhanh TO DT_QuanLy
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.MonAn TO DT_QuanLy
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.NhanVien TO DT_QuanLy
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.Kho TO DT_QuanLy
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.PhieuNhap TO DT_QuanLy
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.CT_PhieuNhap TO
DT_QuanLy
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.Kho TO DT_QuanLy
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.CT_Kho TO DT_QuanLy
-- Phân quyền cho các vai trò còn lại
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.MonAn TO
DT_NhanVienKho
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.NguyenLieu TO
DT_NhanVienKho
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.CongThuc TO
DT_NhanVienKho
-- Tiếp tục phân quyền cho các vai trò khác

-- Phân quyền cho vai trò DT_KhachHang
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.KhachHang TO
DT_KhachHang
GRANT SELECT ON dbo.HoaDon TO DT_KhachHang
GRANT SELECT ON dbo.CT_HoaDon TO DT_KhachHang

-- Phân quyền cho vai trò DT_NhanVienPV
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.HoaDon TO
DT_NhanVienPV
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.CT_HoaDon TO
DT_NhanVienPV
-- Tiếp tục phân quyền cho các vai trò khác

-- Phân quyền cho vai trò DT_NhanVienCSKH
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.KhachHang TO
DT_NhanVienCSKH
GRANT SELECT ON dbo.HoaDon TO DT_NhanVienCSKH
GRANT SELECT ON dbo.CT_HoaDon TO DT_NhanVienCSKH
-- Tiếp tục phân quyền cho các vai trò khác

-- Phân quyền cho vai trò DT_NhanVienKho
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.NguyenLieu TO
DT_NhanVienKho
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.PhieuNhap TO
DT_NhanVienKho
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.CT_PhieuNhap TO
DT_NhanVienKho
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.Kho TO DT_NhanVienKho
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.CT_Kho TO

DT_NhanVienKho

--Phân quyền cho các proc
GRANT EXECUTE ON sp_CapNhatThongTinNhanVien TO DT_QuanLy;
GRANT EXECUTE ON sp_XemDanhSachMonAn TO DT_QuanLy;
GRANT EXECUTE ON sp_XemThongTinMonAn TO DT_QuanLy;
GRANT EXECUTE ON sp_ThemMonAn TO DT_QuanLy;
GRANT EXECUTE ON sp_ThemNguyenLieu TO DT_QuanLy;
GRANT EXECUTE ON sp_ThemPhieuNhap TO DT_NhanVienKho;
GRANT EXECUTE ON sp_ThemCT_PhieuNhap TO DT_NhanVienKho;
GRANT EXECUTE ON sp_ThemHoaDon TO DT_NhanVienPV;
GRANT EXECUTE ON sp_ThemCT_HoaDon TO DT_NhanVienPV;
GRANT EXECUTE ON sp_Sua_CT_HoaDon TO DT_NhanVienPV;
GRANT EXECUTE ON sp_XoaCT_HoaDon TO DT_QuanLy;
GRANT EXECUTE ON DangNhapLogin TO DT_QuanLy;
GRANT EXECUTE ON DangNhapLogin TO DT_NhanVienKho;
GRANT EXECUTE ON DangNhapLogin TO DT_NhanVienCSKH;
GRANT EXECUTE ON DangNhapLogin TO DT_KhachHang;
GRANT EXECUTE ON LayDanhsachKH TO DT_NhanVienPV;
GRANT EXECUTE ON sp_DangKyTaiKhoan TO DT_QuanLy;
GRANT EXECUTE ON sp_XoaTaiKhoanKhachHang TO DT_QuanLy;
GRANT EXECUTE ON sp_CapNhatMatKhauNhanVien TO DT_QuanLy;
GRANT EXECUTE ON sp_XemThongTinNhanVien TO DT_QuanLy;
GRANT EXECUTE ON sp_XemDanhSachNhanVien TO DT_QuanLy;
GRANT EXECUTE ON sp_CapNhatThongTinKhachHang TO DT_NhanVienCSKH;
```

5.1.5 Xử lý đồng thời (Khi cập nhật mật khẩu của Khách hàng thì sẽ lock bảng KhachHang để đảm bảo không ai khác cập nhật cùng lúc được)

```
BEGIN TRANSACTION;

BEGIN TRY
   -- Bắt đầu transaction và bắt đầu thực hiện các thao tác

   -- Lock bảng KhachHang để đảm bảo không có ai khác cập nhật cùng lúc
   -- Chế độ XLOCK sẽ khóa bảng và chỉ cho phép xem dữ liệu
   SELECT * FROM KhachHang WITH (XLOCK, ROWLOCK) WHERE SoDienThoai
= N'3443755275';

   -- Thực hiện cập nhật mật khẩu khách hàng
   EXEC sp_CapNhatMatKhauKhachHang
      @SoDienThoai = N'3443755275',
      @MatKhau = N'090909';

   -- Kết thúc transaction và xác nhận các thay đổi
   COMMIT TRANSACTION;
   SELECT 'Cập nhật mật khẩu khách hàng thành công' AS ThongBao;
END TRY
BEGIN CATCH
   -- Xử lý lỗi nếu có
   ROLLBACK TRANSACTION;
   SELECT ERROR_MESSAGE() AS ErrorMessage;
END CATCH;
```

Chương 6 KẾT LUẬN

6.1 Kết quả đạt được

6.1.1 Phía server

- Xây dựng cấu trúc dữ liệu phù hợp với yêu cầu của người dùng.
- Xây dựng các ràng buộc toàn vẹn nhằm giảm thiểu sai sót nhập liệu của người dùng.
- Xây dựng các stored procedure và function đáp ứng yêu cầu truy vấn và giao tác của người dùng, đảm bảo việc xử lý giao tác được thực hiện nhanh và có thể xử lý đồng thời.
- Phân quyền trên database.

- Lập lịch backup tự động cho database.
- Xây dựng thành công mô hình phân tán theo trạm chi nhánh.

6.1.2 Phía client

Xây dựng trang web phục vụ đầy đủ các chức năng tùy loại người dùng yêu cầu.

6.2 Kết quả chưa đạt được

- Mã hóa dữ liệu nhằm tăng tính bảo mật.
- Giao diện chưa thực sự bắt mắt.

6.3 Hướng phát triển trong tương lai

- Mã hóa dữ liệu để tăng tính bảo mật.
- Thiết kế giao diện bắt mắt hơn.

TÀI LIỆU THAM KHẢO

[1] Microsoft, Microsoft SQL documentation
https://docs.microsoft.com/vi-vn/sql/?view=sql-server-ver15

[2] Microsoft, ASP.NET documentation
https://docs.microsoft.com/vi-vn/aspnet/core/?view=aspnetcore-5.0

[3] Stack Exchange, stackoverflow.com
https://stackoverflow.com/

BẢNG PHÂN CÔNG

	Vũ Đức Dũng	Nguyễn Thị Hoàng Thơ	Hoàng Phú
Khảo sát nghiệp vụ	X	X	X
Xác định các yêu cầu cho đề tài	X		
Thiết kế lược đồ Use case		X	
Phân tích dữ liệu	X		X
Phân tích ràng buộc toàn vẹn	X	X	X
Thiết kế các bước thực hiện giao tác	X		X
Thiết kế phân quyền	X	X	X
Xây dựng lịch backup		X	
Xây dựng các ràng buộc toàn vẹn và các giao tác trên server	X	X	X
Phân quyền trên server	X		
Lập lịch backup tự động		X	
Xây dựng các chức năng và giao diện cho client	X		X
Test hệ thống	X	X	
Viết báo cáo	X		X