TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC THÀNH PHỐ HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN HỌC KỲ I

NĂM HỌC 2022-2023

MÔN: CÁC HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

QUẢN LÝ KHO PHÂN PHỐI NGUYÊN VẬT LIỆU CỦA CỬA HÀNG THỨC ĂN NHANH

Giảng viên hướng dẫn: Th.s Nguyễn Thanh Trung

Sinh viên thực hiện:

Phạm Gia Khương – 19DH110076

Nguyễn Văn Bình Minh – 19DH110596

Trần Văn Minh – 19DH110060

Thành phố Hồ Chí Minh, tháng 12 năm 2022

TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC THÀNH PHỐ HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN HỌC KỲ I

NĂM HỌC 2022-2023

MÔN: CÁC HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

QUẢN LÝ KHO PHÂN PHỐI NGUYÊN VẬT LIỆU CỦA CỬA HÀNG THỨC ĂN NHANH

Giảng viên hướng dẫn: Th.s Nguyễn Thanh Trung

Sinh viên thực hiện:

Phạm Gia Khương – 19DH110076

Nguyễn Văn Bình Minh – 19DH110596

Trần Văn Minh – 19DH110060

Thành phố Hồ Chí Minh, tháng 12 năm 2022

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG

LỜI MỞ ĐẦU

Ngày nay Công nghệ thông tin đã phát triển với tốc độ nhanh chóng. Công nghệ thông tin đã được áp dụng trong nhiều lĩnh vực như nghiên cứu khoa học, phát triển kinh tế, các loại hình nghệ thuật khác nhau. Thế giới xích lại gần nhau hơn nhờ Công nghệ thông tin. Tất cả các nước đều đang cố gắng làm chủ kiến thức và tìm cách áp dụng thành tựu của Công nghệ thông tin vào mọi ngành nghề kinh tế - xã hội của đất nước.

Do vậy, công tác quản lý kho hàng/kho nguyên vật liệu là công tác không thể thiếu của tất cả các tổ chức kinh tế. Sự ra đời của các sản phẩm phần mềm đặc biệt là các phần mềm ứng dụng như quản lý kho hàng/kho nguyên vật liệu trong vài năm gần đây mang lại nhiều thuận lợi trong công tác quản lý hàng hóa tránh sự nhầm lẫn, thất thu, mất mát. Tuy nhiên bên cạnh những tiện lợi mà các chương trình này mang lại, vẫn còn nhiều khó khăn, nhược điểm cần được khắc phục. Nhược điểm của các chương trình còn nhiều lý do như: Bản thân các nhà lập trình còn hạn chế về trình độ cũng như kinh nghiệm làm phần mềm.

Thông tin về các mặt hàng được biến đổi hàng ngày mà sổ sách không thể cập nhật những thông tin đó một cách chính xác được. Lý do trên cho thấy việc xây dựng một hệ thống thông tin quản lý kinh doanh trên máy tính, đáp ứng nhanh và hiệu quả các yêu cầu tập hợp hàng nhập, hàng xuất, tra cứu, tìm kiếm, thống kê số lược hàng một cách chính xác và nhanh chóng.

Mặc dù rất cố gắng để hoàn thành công việc, xong thời gian có hạn và kinh nghiệm kiến thức chưa nhiều nên việc phân tích thiết kế còn có nhiều thiếu xót cần được bổ xung.Vì vậy, nhóm thực hiện đề tài mong nhận được ý kiến đóng góp của thầy cô và bạn bè để đề tài ngày càng hoàn thiện hơn. Cuối cùng thay cho lời kết, chúng em xin chân thành cảm ơn GV Ths Nguyễn Thanh Trung đã tận tình giúp đỡ, hướng dẫn, sửa chữa trong suốt quá trình khảo sát thiết kế đề tài này.

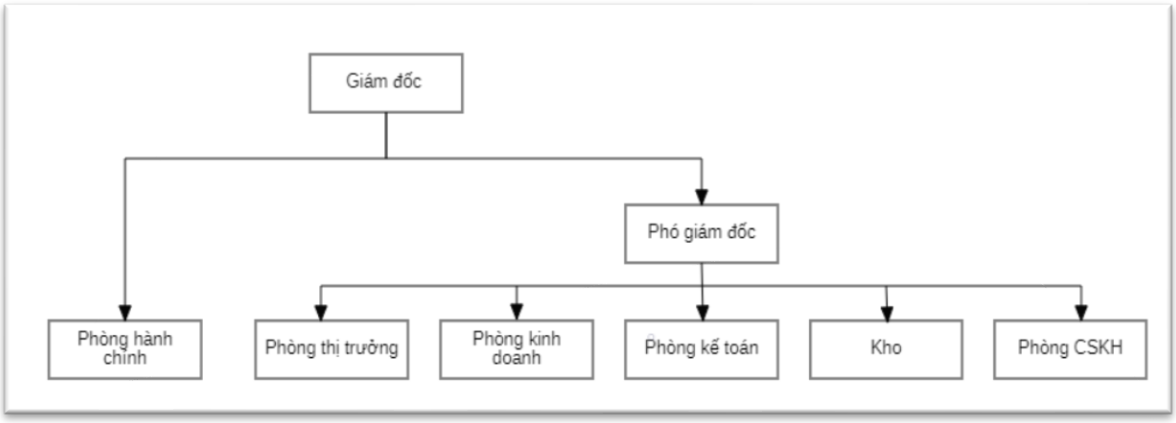
CHƯƠNG 1. KHẢO SÁT NGHIỆP VỤ ĐỀ TÀI

1.1. KHẢO SÁT NGHIỆP VỤ

1.1.1. Tổng quan đề tài

- Kho nguyên vật liệu là một bộ phận không thể thiếu của công ty hay cửa hàng để phục vụ các loại hình dịch vụ đáp ứng cho việc kinh doanh của doanh nghiệp một cách nhanh chóng, tiện lợi.
- Công việc quản lý kho nguyên vật liệu tương đối phức tạp. Đã áp dụng công nghệ thông tin nhưng chỉ dừng lại ở việc sử dụng công cụ đơn thuần là Excel. Hiện nay phương pháp quản lý hàng hóa các kho nguyên vật liệu tại các doanh nghiệp vừa và nhỏ chủ yếu dựa vào phương pháp thủ công:
  - ✓ Hầu hết các hồ sơ tài liệu của kho đều được ghi chép, lưu trữ bằng giấy tờ. Vì vậy lượng sổ sách lưu trữ là rất lớn, cồng kềnh.
  - ✓ Khi quản lý hàng hóa số lượng lớn sẽ bị khó khăn, tốn nhiều thời gian, ùn tắc và khó tránh khỏi sai lệch, phải sử dụng nguồn nhân lực lớn đòi hỏi chi phí cao.
  - ✓ Việc tìm kiếm, kiểm tra thông tin về một loại hàng hóa sẽ rất mất thời gian vì phải tìm kiếm trực tiếp qua sổ sách.
  - ✓ Việc thêm bớt hay chỉnh sửa sẽ gặp nhiều khó khăn vì khi thay đổi sẽ phải thay đổi hồ sơ cũ bằng hồ sơ mới. Khó khăn trong việc sao lưu khi khặp các sự cố về chất lượng giấy hoặc hỏa hoạn.

1.1.2. Cơ cấu tổ chức của hệ thống



Hình 1: Cơ cấu tổ chức của hệ thống

- Giám đốc: là người lãnh đạo, quản lý và giám sát mọi hoạt động chung của công ty.
- Phó giám đốc: là người hỗ trợ cho Giám đốc và chịu trách nhiệm trước các nhiệm vụ được giao.
- Phòng hành chính: tổ chức cán bộ quản lý; sắp xếp lao động, tuyển dụng lao động; quản lý tài chính công ty.
- Phòng thị trưởng: lên kế hoạch và thực hiện công tác tiêu thụ sản phẩm.
- Phòng kinh doanh: bán sản phẩm; tư vấn về việc nghiên cứu và phát triển sản phẩm; mở rộng thị trường; xây dựng mối quan hệ với khách hàng.
- Phòng kế toán: quản lý về tài sản, nguồn vốn, thu chi và hệ thống hóa các số liệu .

1.1.3. Hiện trạng hoạt động của hệ thống

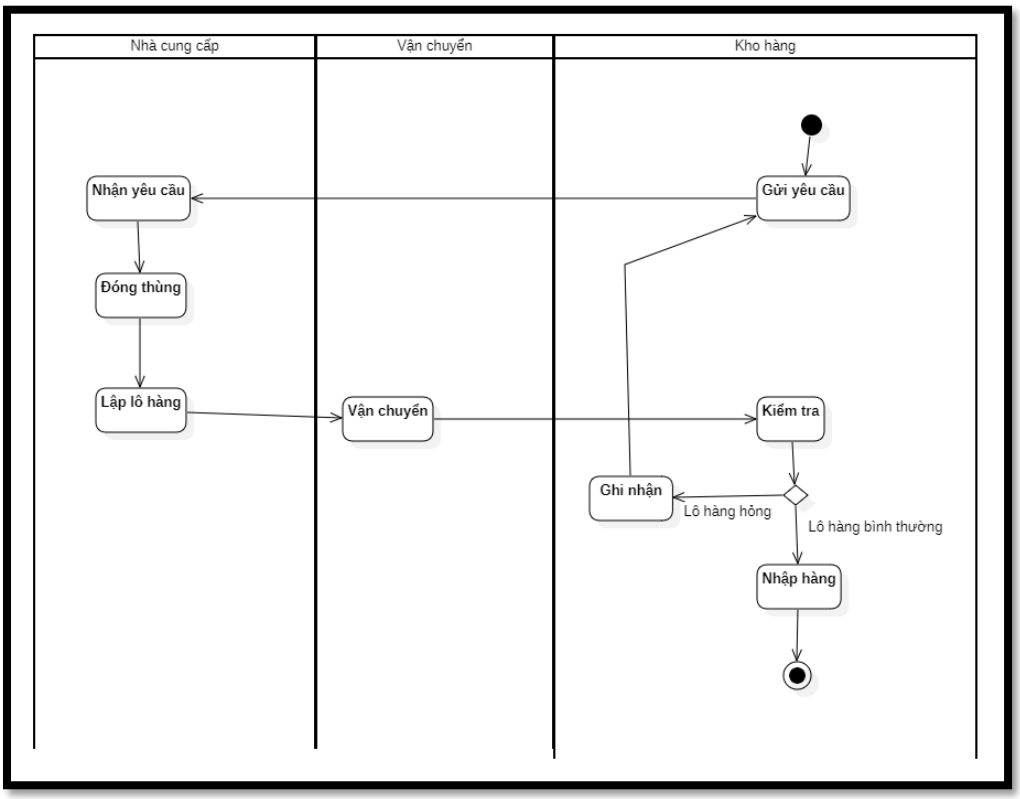
- Công ty hiện nhập sổ, chứng từ thủ công qua giấy.
- Nhập liệu sử dụng Excel để lưu trữ thông tin, sổ sách giấy tờ.
- Việc lưu trữ tốn nhiều thời gian để nhập liệu và rời rạc theo từng file.
- Công ty chưa có biện pháp back up dữ liệu cụ thể.

1.1.4. Đánh giá hiện trạng hệ thống cũ và yêu cầu của hệ thống mới

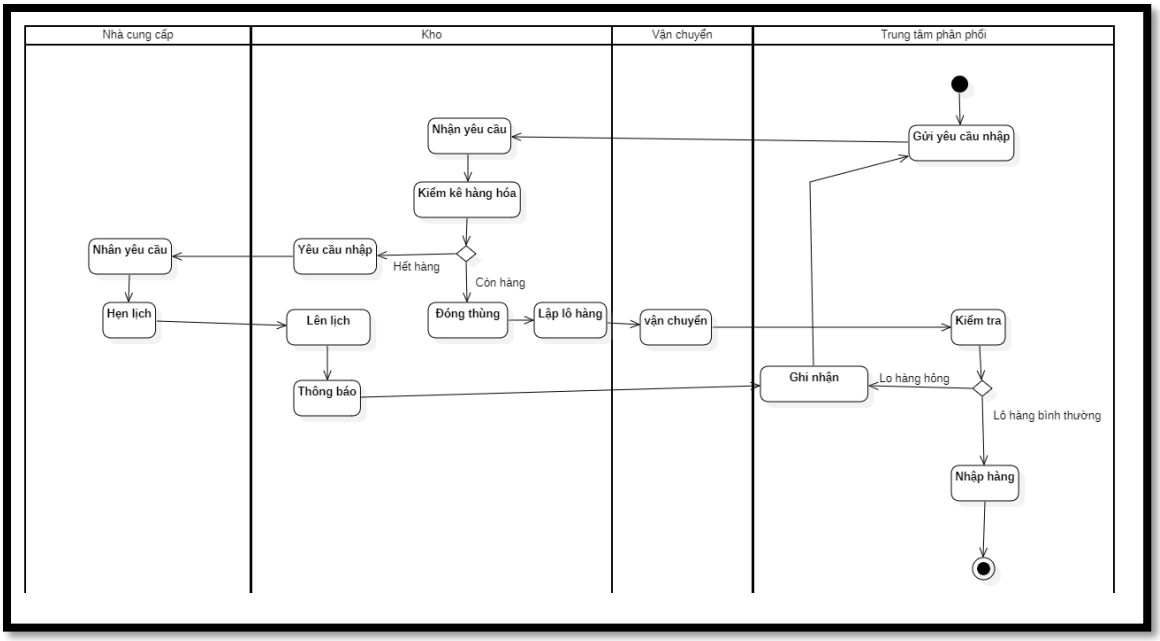
- Đề xuất sử dụng hệ quản trị csdl MSSQL và thiết kế một hệ thống hỗ trợ việc nhập xuất, lưu kho và báo cáo thông qua một phần mềm chính.
- Cải thiện tốc độ và chính xác trong nhập xuất.
- Có thể back up dữ liệu dễ dàng.
- Tuy nhiên cần chuyển đổi dữ liệu từ hệ thống cũ sang hệ thống mới cũng như huấn luyện nhân viên sử dụng phần mềm.

1.2. MÔ TẢ NỘI DUNG GIẢI QUYẾT CỦA ĐỀ TÀI

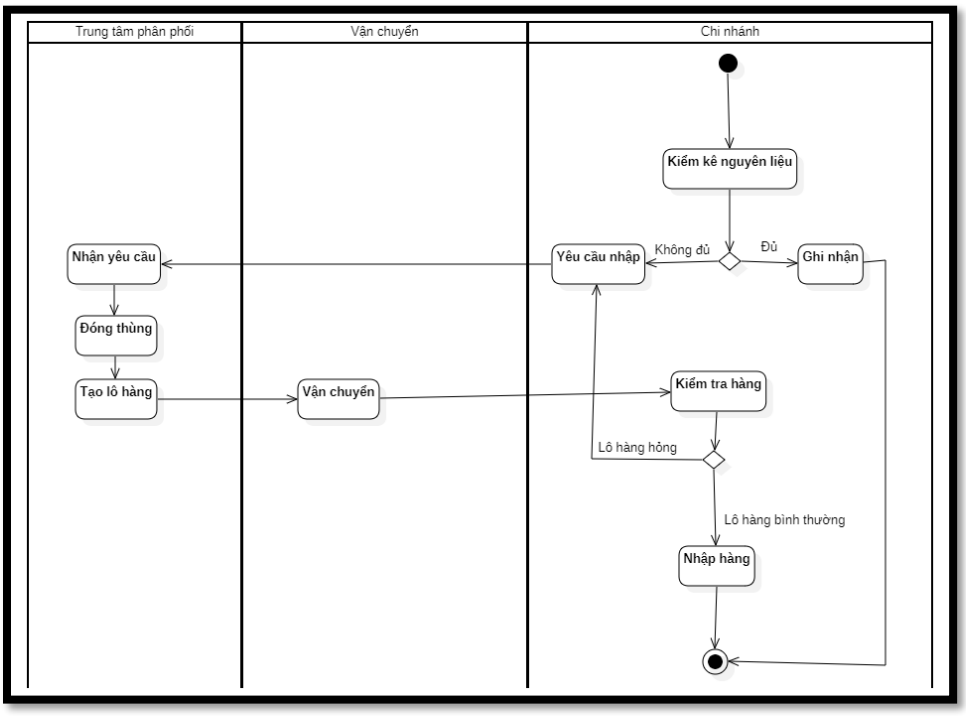
1.2.1. Các quy trình, nghiệp vụ



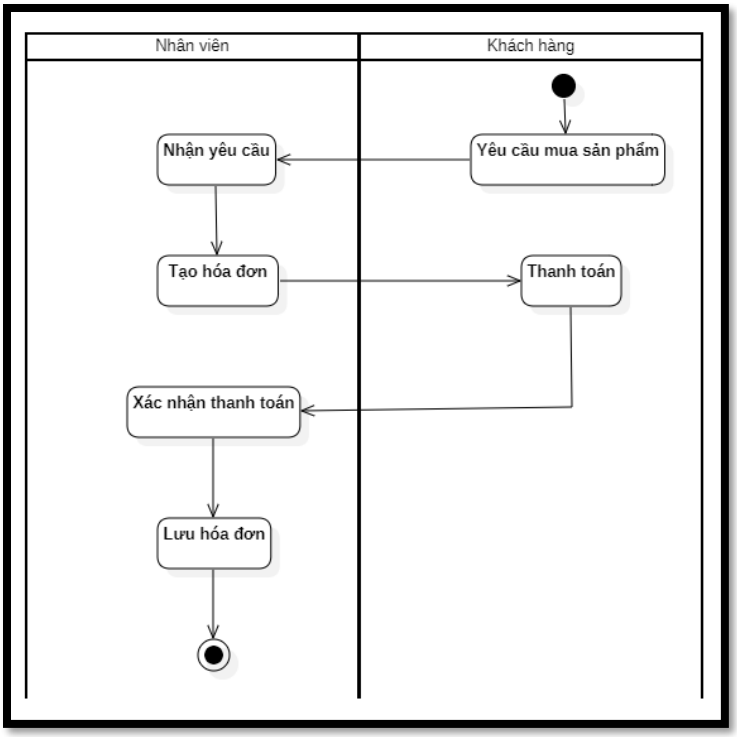
Hình 2: Quy trình cung cấp nguyên liệu



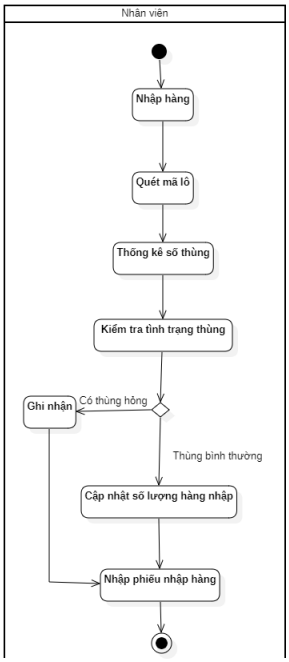
Hình 3: Quy trình phân phối nguyên liệu



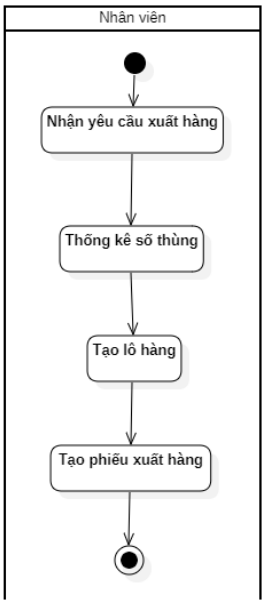
Hình 4: Quy trình nhập hàng chi nhánh



Hình 5: Quy trình tạo hóa đơn



Hình 6: Quy trình nhập hàng



Hình 7: Quy trình xuất hàng

Hình 8: Quy trình kiểm tra tồn kho

1.2.2. Các chứng từ, báo cáo

PHIẾU NHẬP KHO

Ngày……tháng……năm……

Số phiếu:……

- Họ tên người giao:……- Ngày nhập hàng:……
- Xuất tại:……
- Nhập kho tại cơ sở:……
- Lô hàng:……

STT	Mã lô	Số lượng thùng	Trị giá lô	Thời hạn

Người lập phiếu *(Ký, họ tên)* | Người giao *(Ký, họ tên)* | Quản lý *(Ký, họ tên)*

Hình 9: Biểu mẫu phiếu nhập kho

PHIẾU XUẤT KHO

Ngày……tháng……năm……

Số phiếu:……

- Họ tên người nhận: .....
- Địa chỉ: .....
- Xuất kho tại cơ sở:.....
- Cơ sở yêu cầu nhập:……

STT	Mã lô	Số lượng thùng	Trị giá lô	Thời hạn

Người lập phiếu *(Ký, họ tên)* | Người nhận *(Ký, họ tên)* | Tài xế *(Ký, họ tên)* | Thủ kho *(Ký, họ tên)* | Quản lý *(Ký, họ tên)*

Hình 10: Biểu mẫu phiếu xuất kho

HÓA ĐƠN THANH TOÁN

Số HĐ:……

Ngày in:……..

Nhân viên:……

Tên khách hàng:…

SDT khách hàng:…

STT	Sản phẩm	SL	Đơn giá	Thành tiền
1				
2				

Tổng tiền:

Hình 11: Hóa đơn thanh toán

CHI TIẾT LÔ HÀNG

(Mã :……)

- Tài xế: .....
- Ngày xuất:..... Ngày nhập: .....
- Thời hạn:.....
- Tổng số thùng: .....

Stt	Thùng	Nguyên liệu	Số lượng	Đơn vị tính	Giá trị

Người lập phiếu
(Ký, họ tên)

Thủ kho
(Ký, họ tên)

Quản lý
(Ký, họ tên)

Hình 12: Chi tiết lô hàng

TỒN KHO

- Cơ sở:.....
- Ngày lập:.....

Stt	Tháng	Năm	Nguyên liệu	Số lượng tồn	Tổng giá trị

Người lập phiếu
(Ký, họ tên)

Thủ kho
(Ký, họ tên)

Quản lý
(Ký, họ tên)

Hình 13: Tồn kho

CHI TIẾT TỒN KHO

- Cơ sở:.....
- Ngày lập:.....
- Tổng số lô tồn:.....

Stt	Lô	Số lượng thùng	Thời hạn	Tình trạng	Tổng giá trị

Người lập phiếu
(Ký, họ tên)

Thủ kho
(Ký, họ tên)

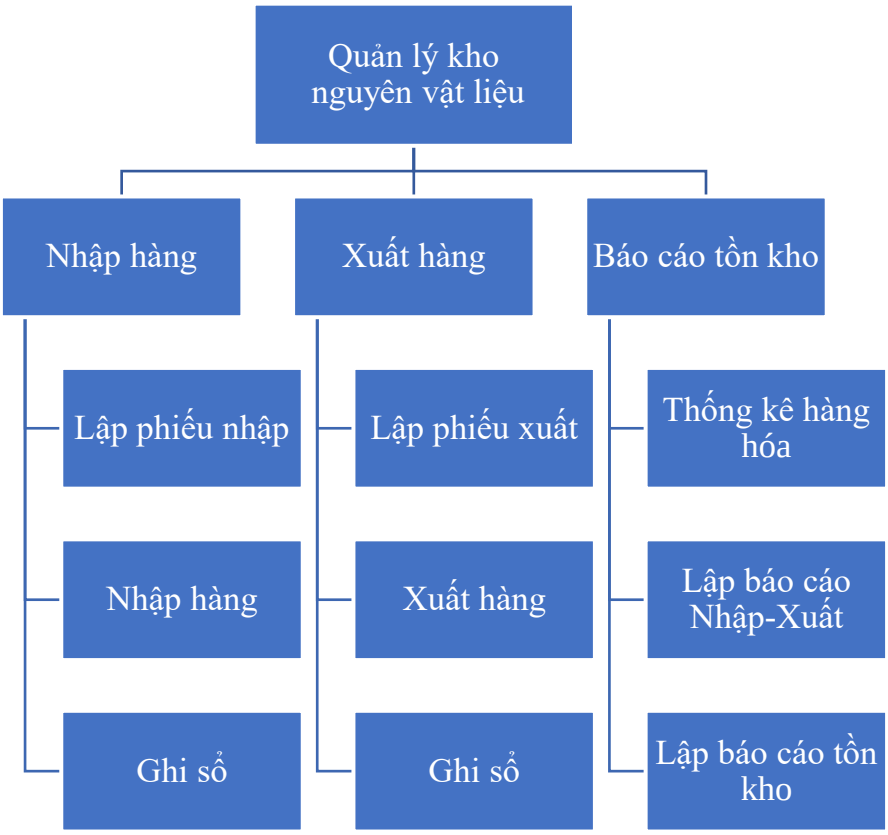
Quản lý
(Ký, họ tên)

Hình 14: Chi tiết tồn kho

CHƯƠNG 2. PHÂN TÍCH HỆ THỐNG

2.1. PHÂN TÍCH CHỨC NĂNG

2.1.1. Sơ đồ chức năng



Hình 15: Sơ đồ chức năng

2.1.2. Mô tả chức năng

❖ Nhập hàng:

- Lập phiếu nhập: tiến hành lập phiếu nhập hàng, phiếu nhập có đầy đủ thông tin, chữ ký trên chứng từ, số lượng hàng hóa, chủng loại, quy cách.
- Nhập hàng: kiểm tra hàng hóa và căn cứ dựa trên phiếu nhập.
- Ghi sổ: đưa thông tin nhập hàng vào sổ để lập báo cáo đối chiếu, chứng thực.

❖ Xuất hàng:

- Lập phiếu xuất: cũng như thủ tục xuất hàng, tiến hành lập phiếu xuất hàng, phiếu xuất có đầy đủ thông tin, chữ ký trên chứng từ, số lượng hàng hóa, chủng loại, quy cách.
- Xuất hàng: xuất hàng theo đúng quy cách, kiểm tar thông tin hàng hóa xuyên suốt trong quá trình xuất hàng căn cứ theo thông tin của phiếu xuất.
- Ghi sổ: đưa thông tin xuất hàng vào sổ để lập báo cáo đối chiếu, chứng thực.

❖ Báo cáo tồn kho:

- Thống kê hàng hóa: kiểm tra số lượng hàng hóa torng kho (theo ngày/tuần/tháng).
- Lập báo cáo nhập-xuất: báo cáo chi tiết thông tin nhập-xuất (số lượng, ngày giờ).
- Lập báo cáo tồn kho: lập báo cáo về lượng hàng còn lại trong kho để các phòng ban nắm được tình hình tử đó đưa ra các hướng kinh doanh.

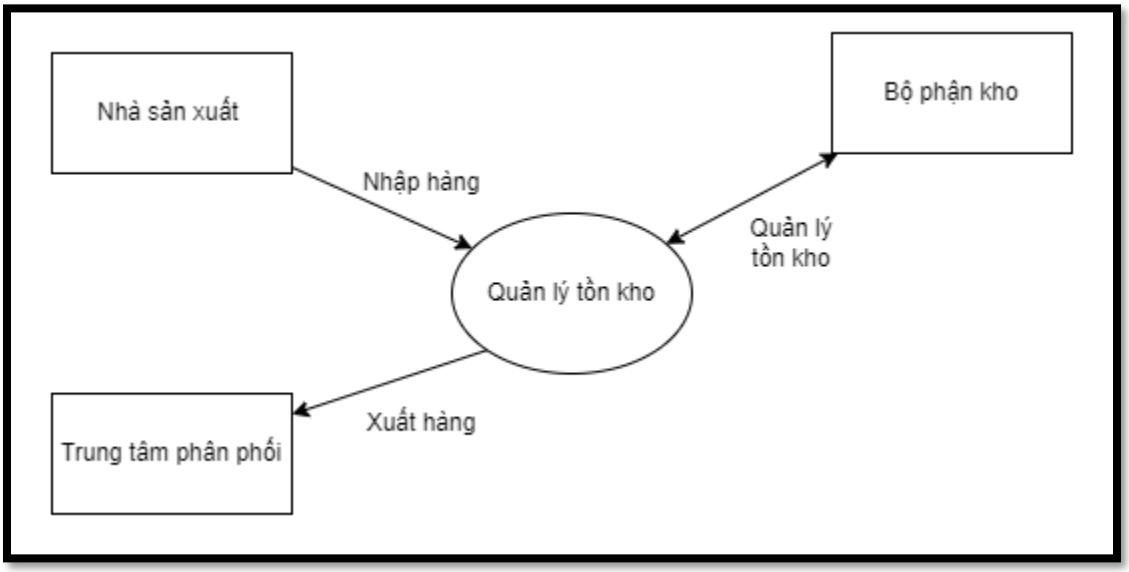
2.1.3. Ràng buộc dữ liệu

- Nhà cung cấp: phải giao đủ số sản phẩm trong thời gian kho yêu cầu.
- Kho: hàng trong kho chỉ được lưu kho trong thời gian quy định.

2.2. PHÂN TÍCH DỮ LIỆU

2.2.1. Sơ đồ luồng dữ liệu

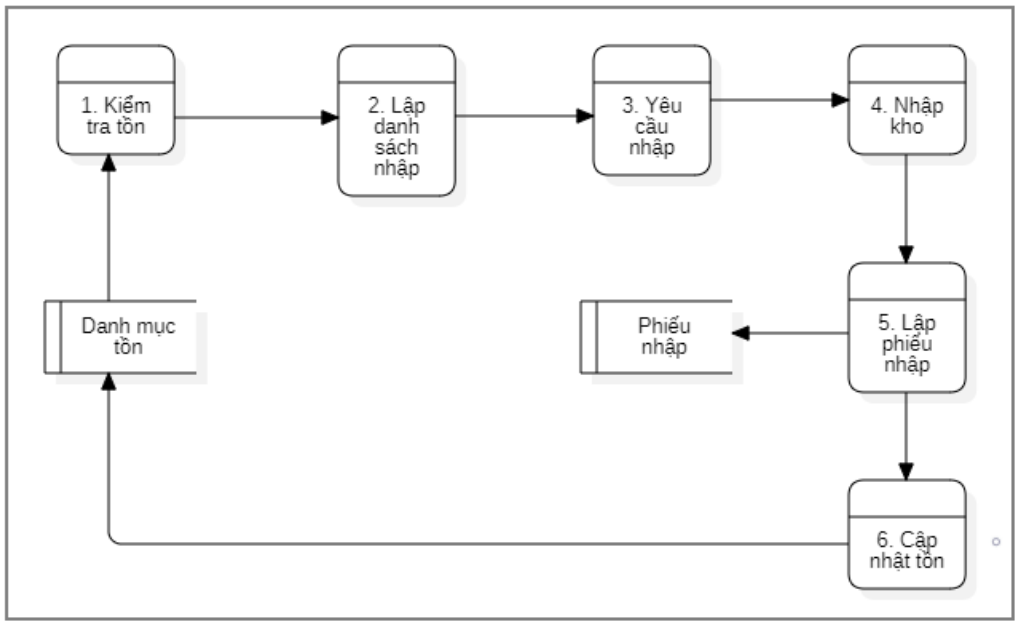
2.2.1.1. DFD mức 0



Hình 16: DFD mức 0

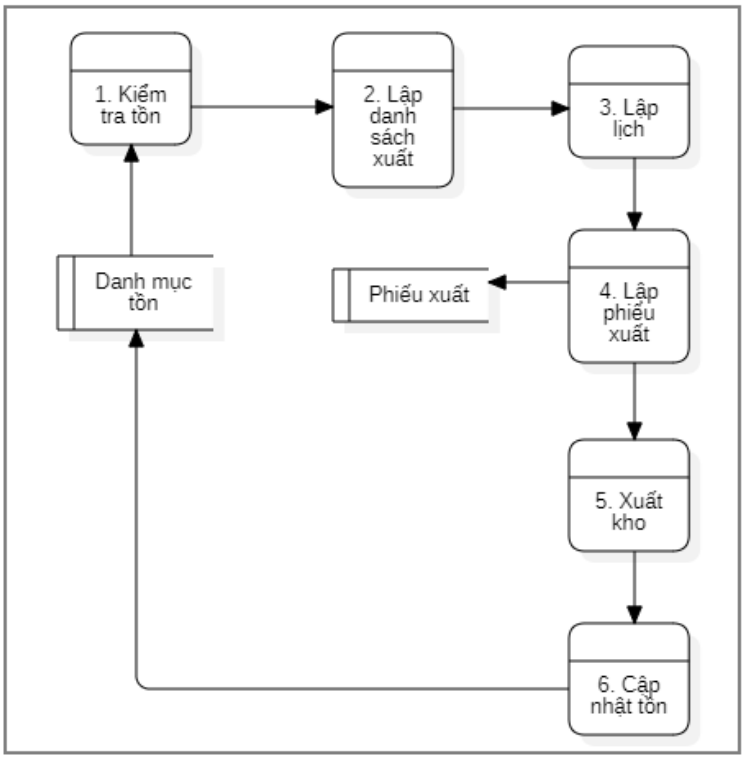
2.2.1.2. DFD mức 1

❖ Nhập kho



Hình 17: DFD nhập kho mức 1

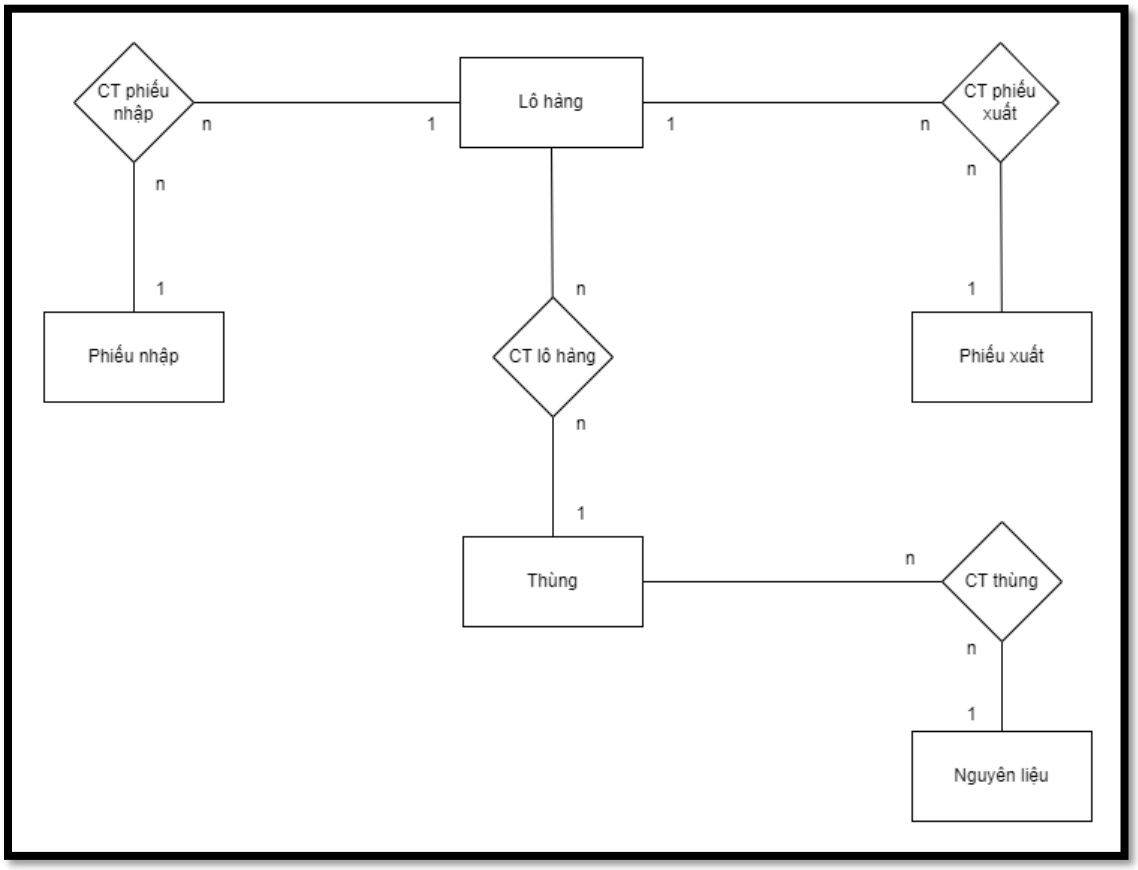
❖ Xuất kho



Hình 18: DFD xuất kho mức 1

2.2.2. Sơ đồ dữ liệu

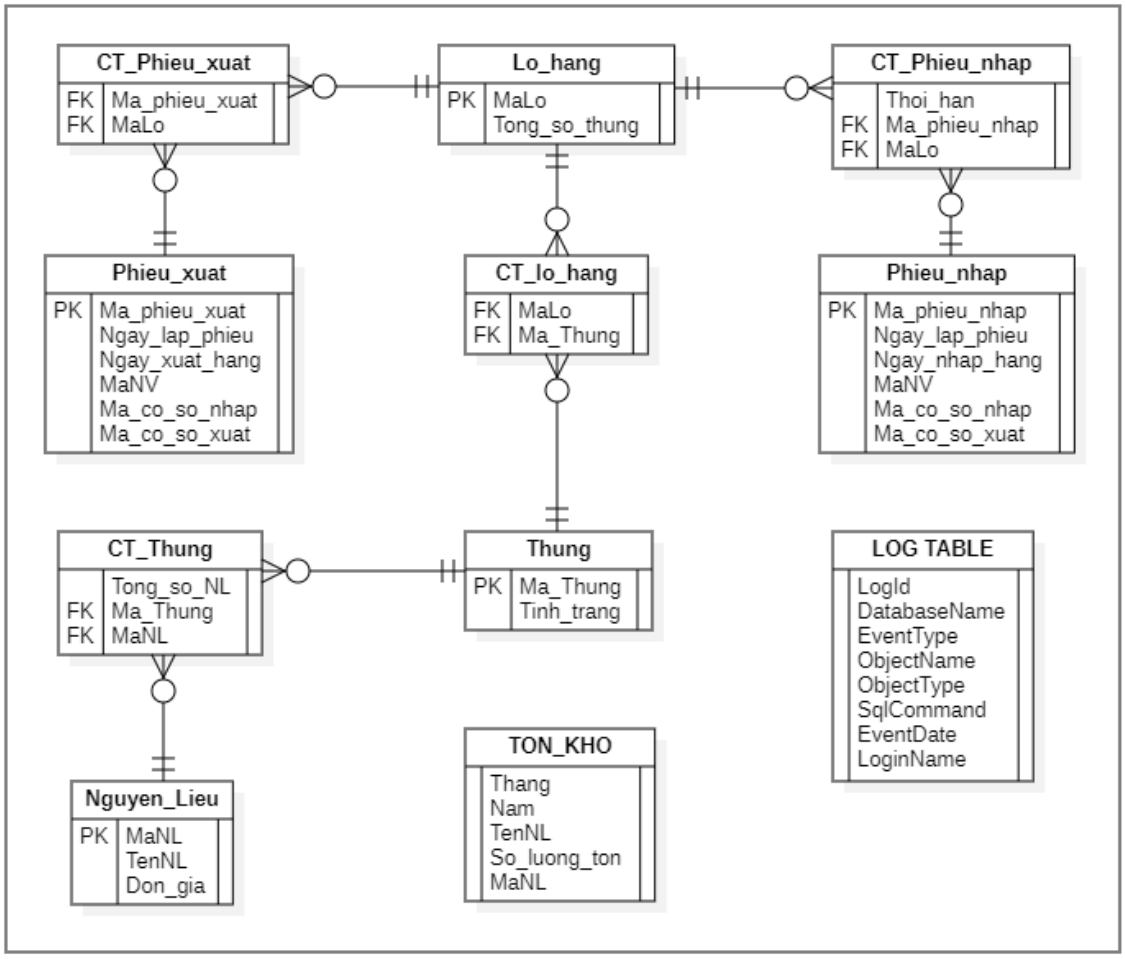
❖ Sơ đồ dữ liệu dạng quan hệ thực thể



Hình 19: ERD dạng quan hệ thực thể

CHƯƠNG 3. THIẾT KẾ HỆ THỐNG

3.1. MÔ HÌNH QUAN HỆ GIỮA CÁC BẢNG



Hình 20: Mô hình quan hệ giữa các bảng

3.2. TỪ ĐIỂN DỮ LIỆU

Bảng 1: Mô tả bảng Lo_hang

Lo_hang			
Stt	Thuộc tính	Kiểu dữ liệu	Ràng buộc
1	MaLo	Char (7)	PK
2	Tong_so_thung	Integer	

Bảng 2: Mô tả bảng Phieu_nhap

Phieu_nhap			
Stt	Thuộc tính	Kiểu dữ liệu	Ràng buộc
1	Ma_phieu_nhap	Char (7)	PK
2	Ngay_lap_phieu	Date	
3	Ngay_nhap_hang	Date	
4	MaNV	Char (7)	
5	Ma_co_so_nhap	Char (7)	
6	Ma_co_so_xuat	Char (7)	

Bảng 3: Mô tả bảng Phieu_xuat

Phieu_xuat			
Stt	Thuộc tính	Kiểu dữ liệu	Ràng buộc
1	Ma_phieu_xuat	Char (7)	PK
2	Ngay_lap_phieu	Date	
3	Ngay_xuat_hang	Date	
4	MaNV	Char (7)	
5	Ma_co_so_nhap	Char (7)	
6	Ma_co_so_xuat	Char (7)	

Bảng 4: Mô tả bảng CT_lo_hang

CT_lo_hang			
Stt	Thuộc tính	Kiểu dữ liệu	Ràng buộc
1	MaLo	Char (7)	FK
2	Ma_Thung	Char (7)	FK

Bảng 5: Mô tả bảng CT_Phieu_nhap

CT_Phieu_nhap			
Stt	Thuộc tính	Kiểu dữ liệu	Ràng buộc
1	Ma_phieu_nhap	Char (7)	FK
2	MaLo	Char (7)	FK
3	Thoi_han	Small Int	

Bảng 6: Mô tả bảng CT_Phieu_xuat

CT_Phieu_xuat			
Stt	Thuộc tính	Kiểu dữ liệu	Ràng buộc
1	Ma_phieu_xuat	Char (7)	FK
2	MaLo	Char (7)	FK

Bảng 7: Mô tả bảng Thung

Thung			
Stt	Thuộc tính	Kiểu dữ liệu	Ràng buộc
1	Ma_Thung	Char (7)	PK
2	Tinh_trang	Nvarchar(12)	

Bảng 8: Mô tả bảng CT_Thung

CT_Thung			
Stt	Thuộc tính	Kiểu dữ liệu	Ràng buộc
1	Ma_Thung	Char (7)	FK
2	MaNL	Char (7)	FK
3	Tong_so_NL	Integer	

Bảng 9: Mô tả bảng Nguyen_Lieu

Nguyen_Lieu			
Stt	Thuộc tính	Kiểu dữ liệu	Ràng buộc
1	MaNL	Char (7)	PK
2	TenNL	Nvarchar (30)	
3	Don_Gia	Float	

Bảng 10: Mô tả bảng TON_KHO

TON_KHO			
Stt	Thuộc tính	Kiểu dữ liệu	Ràng buộc
1	Thang	Date	
2	Nam	Date	
3	TenNL	Nvarchar (30)	
4	So_luong_ton	Integer	
5	MaNL	Char (7)	FK

Bảng 11: Mô tả bảng LOG TABLE

LOG TABLE			
Stt	Thuộc tính	Kiểu dữ liệu	Ràng buộc
1	LogId	Int	
2	DatabaseName	Varchar (256)	
3	EventType	Varchar (50)	
4	ObjectNam	Varchar (256)	
5	ObjectType	Varchar (25)	
6	SqlCommand	Varchar (Max)	
7	EventDate	Datetime	
8	LoginName	Varchar (256)	

Bảng 12: Bảng quy định tạo mã

Bảng quy định tạo mã			
Stt	Thuộc tính	Kiểu dữ liệu	Ràng buộc
Stt	Loại	Mã	Chú thích
1	Nguyên Liệu	NL*****	2 kí tự chữ đầu, 5 kí tự số
2	Phiếu Xuất	PX*****	2 kí tự chữ đầu, 5 kí tự số
3	Phiếu Nhập	PN*****	2 kí tự chữ đầu, 5 kí tự số
4	Lô Hàng	LH*****	2 kí tự chữ đầu, 5 kí tự số

3.3. THIẾT KẾ CSDL

❖ Tạo CSDL

Cơ sở dữ liệu sẽ được lưu trong ổ D:/ với tên folder WarehouseDB

Chia thành 3 folder main, subdf, logdf

Folder main sẽ chứa file: warehouseDB.mdf (100MB Growth 10%)

Folder subdf sẽ chứa 2 file:

- warehouseDB_FG1_Dat1_01.ndf (50MB Growth 10%)
- warehouseDB_FG1_Dat1_01_02.ndf (50MB Growth 10%)

Folder logdf sẽ chứa file: warehouseDB_log.ldf (100MB Growth 10MB)

```
CREATE DATABASE WAREHOUSE_MANAGEMENT
On Primary
( Name= warehouseDB_data,
FileName='E:\WarehouseDB\main\warehouseDB.mdf',
Size=100MB,
MaxSize= Unlimited,
FileGrowth=10%
),

FILEGROUP warehouseDB_FG1
 ( NAME = 'warehouseDB_FG1_Dat1',
 FILENAME='E:\WarehouseDB\subdf\warehouseDB_FG1_Dat1_01.ndf',
 SIZE = 50MB,
 MAXSIZE=300MB,
 FILEGROWTH=10%),
 ( NAME = 'warehouseDB_FG1_Dat2',
 FILENAME ='E:\WarehouseDB\subdf\warehouseDB_FG1_Dat1_01_02.ndf',
 SIZE = 50MB,
MAXSIZE=300MB,
 FILEGROWTH=10%)
Log On
( Name= warehouseDB_log,
FileName='E:\WarehouseDB\logdf\warehouseDB_log.ldf',
Size= 50MB,
MaxSize= 100MB,
FileGrowth=10MB
);

use WAREHOUSE_MANAGEMENT
```

Hình 21: Tạo CSDL

❖ Tạo Tables

- Tạo table Lo_hang

```
create table Lo_hang
(
    MaLo char(7) PRIMARY KEY,
    Tong_so_thung integer,
);
```

Hình 22: Tạo table Lo_hang

- Tạo table Thung

```
create table Thung
(
    Ma_Thung char(7) PRIMARY KEY,
    Tinh_trang nvarchar(12) default N'Nguyên kiện'
    CONSTRAINT CHK_Tinh_trang_Thung
    CHECK (Tinh_trang IN ('Nguyên kiện','đã mở','Hỏng'))
);
```

Hình 23: Tạo table Thung

- Tạo table CT_lo_hang

```
create table CT_lo_hang
(
    MaLo char(7) REFERENCES Lo_hang(MaLo),
    Ma_Thung char(7) REFERENCES Thung(Ma_Thung)
);
```

Hình 24: Tạo table CT_lo_hang

- Tạo table Nguyen_Lieu

```
create table Nguyen_Lieu
(
    MaNL char(7) PRIMARY KEY,
    TenNL nvarchar(30),
    Don_gia float,
);
```

Hình 25: Tạo table Nguyen_Lieu

- Tạo table CT_Thung

```
create table CT_Thung
(
    Ma_Thung char(7) REFERENCES Thung(Ma_Thung),
    MaNL char(7) REFERENCES Nguyen_Lieu(MaNL),
    Tong_so_NL integer
);
```

Hình 26: Tạo table CT_Thung

- Tạo table Phieu_nhap

```
create table Phieu_nhap
(
    Ma_phieu_nhap char(7)PRIMARY KEY,
    Ngay_lap_phieu date default GETDATE(),
    Ngay_nhap_hang date default GETDATE(),
    MaNV char(7),
    Ma_co_so_nhap char(7),
    Ma_co_so_xuat char(7)
);
```

Hình 27: Tạo table Phieu_nhap

- Tạo table Phieu_xuat

```
create table Phieu_xuat
(
    Ma_phieu_xuat char(7)PRIMARY KEY,
    Ngay_lap_phieu date default GETDATE(),
    Ngay_xuat_hang date default GETDATE(),
    MaNV char(7),
    Ma_co_so_nhap char(7),
    Ma_co_so_xuat char(7)
);
```

Hình 28: Tạo table Phieu_xuat

- Tạo table Lo_hang

```
create table CT_Phieu_xuat
(
    Ma_phieu_xuat char(7) REFERENCES Phieu_xuat(Ma_phieu_xuat),
    MaLo char(7) REFERENCES Lo_hang(MaLo)
);
```

Hình 29: Tạo table CT_Phieu_xuat

- Tạo table CT_Phieu_nhap

```
create table CT_Phieu_nhap
(
    Ma_phieu_nhap char(7) foreign key REFERENCES Phieu_nhap(Ma_phieu_nhap),
    MaLo char(7) foreign key REFERENCES Lo_hang(MaLo),
    Thoi_han smallint default 5,
    CONSTRAINT CHK_Thoi_han
    CHECK (Thoi_han <= 5)
);
```

Hình 30: Tạo table CT_Phieu_xuat

- Tạo table TON_KHO

```
create table TON_KHO
(
    Thang VARCHAR(5),
    Nam VARCHAR(5),
    Ma_nguyen_lieu CHAR(7),
    Ten_nguyen_lieu NVARCHAR(30),
    So_luong_ton int
);
```

Hình 31: Tạo table TON_KHO

- Alter tables

```
ALTER TABLE NGUYEN_LIEU ADD DEL_STATUS VARCHAR(7) DEFAULT 'ACTIVE' CONSTRAINT CHK_NL_DEL_STATUS CHECK (DEL_STATUS IN('ACTIVE','DELETED'));
ALTER TABLE THUNG ADD DEL_STATUS VARCHAR(7) DEFAULT 'ACTIVE' CONSTRAINT CHK_TH_DEL_STATUS CHECK (DEL_STATUS IN('ACTIVE','DELETED'));
ALTER TABLE LO_HANG ADD DEL_STATUS VARCHAR(7) DEFAULT 'ACTIVE' CONSTRAINT CHK_LH_DEL_STATUS CHECK (DEL_STATUS IN('ACTIVE','DELETED'));
ALTER TABLE PHIEU_NHAP ADD DEL_STATUS VARCHAR(7) DEFAULT 'ACTIVE' CONSTRAINT CHK_PN_DEL_STATUS CHECK (DEL_STATUS IN('ACTIVE','DELETED'));
ALTER TABLE PHIEU_XUAT ADD DEL_STATUS VARCHAR(7) DEFAULT 'ACTIVE' CONSTRAINT CHK_PX_DEL_STATUS CHECK (DEL_STATUS IN('ACTIVE','DELETED'));
```

Hình 32: Alter Tables

- Tạo table LOG TABLE

```
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[ChangeLog](
    [LogId] [INT] IDENTITY(1,1) NOT NULL,
    [DatabaseName] [VARCHAR](256) NOT NULL,
    [EventType] [VARCHAR](50) NOT NULL,
    [ObjectName] [VARCHAR](256) NOT NULL,
    [ObjectType] [VARCHAR](25) NOT NULL,
    [SqlCommand] [VARCHAR](MAX) NOT NULL,
    [EventDate] [DATETIME] NOT NULL,
    [LoginName] [VARCHAR](256) NOT NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY];

ALTER TABLE [dbo].[ChangeLog] ADD  CONSTRAINT [DF_EventsLog_EventDate] DEFAULT (GETDATE()) FOR [EventDate];
```

Hình 33: Tạo table LOG TABLE

❖ Tạo các Index

```
Go
CREATE  INDEX Index_Lohang
On Lo_hang(MaLo)  ;

Go
CREATE  INDEX Index_CT_Lohang
On CT_lo_hang(MaLo, Ma_Thung) ;

Go
CREATE  INDEX Index_Thung
On Thung(Ma_Thung);

Go
CREATE  INDEX Index_CT_Thung
On CT_Thung(Ma_Thung, MaNL);

Go
CREATE  INDEX Index_Nguyen_Lieu
On Nguyen_Lieu(MaNL);

Go
CREATE  INDEX Index_Phieu_nhap
On Phieu_nhap(Ma_phieu_nhap);

Go
CREATE  INDEX Index_Phieu_xuat
On Phieu_xuat(Ma_phieu_xuat);

Go
CREATE  INDEX Index_CT_Phieu_nhap
On CT_Phieu_nhap(Ma_phieu_nhap, MaLo);

Go
CREATE  INDEX Index_CT_Phieu_xuat
On CT_Phieu_xuat(Ma_phieu_xuat, MaLo);
```

Hình 34: Tạo các Index

3.4. THIẾT KẾ STORED PROCEDURE

❖ Procedure Insert

- Procedure Insert Phieu_xuat

```
CREATE SEQUENCE SEQ_PX_ID MAXVALUE 99999 START WITH 1;

CREATE OR ALTER PROCEDURE PRC_INSERT_PX
                                        @input_ngaylapphieu date,
                                        @input_manv char(7),
                                        @input_cosonhap char(7),
                                        @input_cosoxuat char(7),
                                        @input_ngayxuathang date

AS
DECLARE @ErrorMessage VARCHAR(2000), @id CHAR(7) = concat('PX', DBO.numberZerosString(NEXT VALUE FOR SEQ_PX_ID));
BEGIN
    BEGIN TRAN
        BEGIN TRY
            IF @input_ngaylapphieu IS NULL AND @input_ngayxuathang IS NULL
            BEGIN
                INSERT INTO PHIEU_XUAT (Ma_phieu_xuat,MANV,MA_CO_SO_NHAP,MA_CO_SO_XUAT)
                VALUES(@id,@input_manv,@input_cosonhap,@input_cosoxuat);
            END

            ELSE IF @input_ngaylapphieu IS NOT NULL AND @input_ngayxuathang IS NOT NULL
            BEGIN
                INSERT INTO PHIEU_XUAT (Ma_phieu_xuat,NGAY_LAP_PHIEU,MANV,MA_CO_SO_NHAP,MA_CO_SO_XUAT,NGAY_XUAT_HANG)
                VALUES(@id,@input_ngaylapphieu,@input_manv,@input_cosonhap,@input_cosoxuat,@input_ngayxuathang);
            END

            ELSE IF @input_ngaylapphieu IS NULL AND @input_ngayxuathang IS NOT NULL
            BEGIN
                INSERT INTO PHIEU_XUAT (Ma_phieu_xuat,MANV,MA_CO_SO_NHAP,MA_CO_SO_XUAT,NGAY_XUAT_HANG)
                VALUES(@id,@input_manv,@input_cosonhap,@input_cosoxuat,@input_ngayxuathang);
            END

            ELSE IF @input_ngaylapphieu IS NOT NULL AND @input_ngayxuathang IS NULL
            BEGIN
                INSERT INTO PHIEU_XUAT (Ma_phieu_xuat,NGAY_LAP_PHIEU,MANV,MA_CO_SO_NHAP,MA_CO_SO_XUAT)
                VALUES(@id,@input_ngaylapphieu,@input_manv,@input_cosonhap,@input_cosoxuat);
            END

            ELSE
            BEGIN
                RAISERROR('INSERT UNSUCCESSFULLY',10,1);
            END

            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END
```

Hình 35: Stored Procedure Insert Phieu_xuat

- Procedure Insert Phieu_nhap

```
CREATE SEQUENCE SEQ_PN_ID MAXVALUE 99999 START WITH 1;

CREATE OR ALTER PROCEDURE PRC_INSERT_PN
                                        @input_ngaylapphieu date,
                                        @input_manv char(7),
                                        @input_cosonhap char(7),
                                        @input_cosoxuat char(7),
                                        @input_ngayxuathang date

AS
DECLARE @ErrorMessage VARCHAR(2000), @id CHAR(7) = concat('PX', DBO.numberZerosString(NEXT VALUE FOR SEQ_PX_ID));
BEGIN
    BEGIN TRAN
        BEGIN TRY
            IF @input_ngaylapphieu IS NULL AND @input_ngayxuathang IS NULL
            BEGIN
                INSERT INTO PHIEU_NHAP (Ma_phieu_nhap,MANV,MA_CO_SO_NHAP,MA_CO_SO_XUAT)
                VALUES(@id,@input_manv,@input_cosonhap,@input_cosoxuat);
            END

            ELSE IF @input_ngaylapphieu IS NOT NULL AND @input_ngayxuathang IS NOT NULL
            BEGIN
                INSERT INTO PHIEU_NHAP (Ma_phieu_nhap,NGAY_LAP_PHIEU,MANV,MA_CO_SO_NHAP,MA_CO_SO_XUAT,Ngay_nhap_hang)
                VALUES(@id,@input_ngaylapphieu,@input_manv,@input_cosonhap,@input_cosoxuat,@input_ngayxuathang);
            END

            ELSE IF @input_ngaylapphieu IS NULL AND @input_ngayxuathang IS NOT NULL
            BEGIN
                INSERT INTO PHIEU_NHAP (Ma_phieu_nhap,MANV,MA_CO_SO_NHAP,MA_CO_SO_XUAT,Ngay_nhap_hang)
                VALUES(@id,@input_manv,@input_cosonhap,@input_cosoxuat,@input_ngayxuathang);
            END

            ELSE IF @input_ngaylapphieu IS NOT NULL AND @input_ngayxuathang IS NULL
            BEGIN
                INSERT INTO PHIEU_NHAP (Ma_phieu_nhap,NGAY_LAP_PHIEU,MANV,MA_CO_SO_NHAP,MA_CO_SO_XUAT)
                VALUES(@id,@input_ngaylapphieu,@input_manv,@input_cosonhap,@input_cosoxuat);
            END

            ELSE
            BEGIN
                RAISERROR('INSERT UNSUCCESSFULLY',10,1);
            END

            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END
```

Hình 36: Stored Procedure Insert Phieu_nhap

- Procedure Insert Lo_hang

```
--16 LO HANG
CREATE SEQUENCE SEQ_LH_ID MAXVALUE 99999 START WITH 1;

--Insert Lohang
CREATE  or ALTER PROCEDURE PRC_INSERT_LH
                                        @input_tongthung INT

AS
DECLARE @ErrorMessage VARCHAR(2000), @id CHAR(7) = concat('LH', DBO.numberZerosString(NEXT VALUE FOR SEQ_LH_ID))
BEGIN
    BEGIN TRAN
        BEGIN TRY
            INSERT INTO LO_HANG (MaLo,TONG_SO_THUNG)
            VALUES(@id,@input_tongthung);
            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 37: Stored Procedure Insert Lo_hang

- Procedure Insert Thùng

```
--17 THUNG
CREATE SEQUENCE SEQ_TH_ID MAXVALUE 99999 START WITH 1;
--Insert Thung
CREATE  or ALTER  PROCEDURE PRC_INSERT_TH
                                        @input_tinhtrang varchar

AS
DECLARE @ErrorMessage VARCHAR(2000), @id CHAR(7) = concat('TH', DBO.numberZerosString(NEXT VALUE FOR SEQ_TH_ID))
BEGIN
    BEGIN TRAN
        BEGIN TRY
            INSERT INTO THUNG (TINH_TRANG)
            VALUES(@input_tinhtrang);
            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 38: Stored Procedure Insert Thung

- Procedure Insert CT_Lo_hang

```
e Snapshots rt CT lo hang
CREATE  or ALTER  PROCEDURE PRC_INSERT_CTLH
                                            @input_malo char(7),
                                            @input_mathung char(7)

AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            INSERT INTO CT_LO_HANG
            VALUES(@input_malo,@input_mathung);
            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 39: Stored Procedure Insert CT_Lo_hang

- Procedure Insert Phieu_nhap

```
--19 Insert CT nhap hang
create or ALTER PROCEDURE PRC_INSERT_CTPN
                                        @input_mapn char(7),
                                        @input_malo char(7),
                                        @input_thoihan SMALLINT

AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            IF (@input_thoihan > 5)
            BEGIN
                RAISERROR('INSERT UNSUCCESSFULLY',10,1)
            END
            ELSE
            BEGIN
                INSERT INTO CT_PHIEU_NHAP
                VALUES(@input_mapn,@input_malo,@input_thoihan);
            END
            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 40: Stored Procedure Insert CT_Phieu_nhap

- Procedure Insert Ct_Phieu_xuat

```
--20 Insert CT XUAT hang
CREATE  or ALTER  PROCEDURE PRC_INSERT_CTPX
                                        @input_mapx char(7),
                                        @input_malo char(7)

AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            INSERT INTO CT_PHIEU_XUAT
            VALUES(@input_mapx,@input_malo);;
            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 41: Stored Procedure Insert CT_Phieu_xuat

- Procedure Insert Nguyen_Lieu

```
--23 NGUYEN LIEU
CREATE SEQUENCE SEQ_NL_ID MAXVALUE 99999 START WITH 1;
--Insert nguyen lieu
CREATE  or ALTER  PROCEDURE PRC_INSERT_NL
                                        @input_tennl nvarchar,
                                        @input_dongia float

AS
DECLARE @ErrorMessage VARCHAR(2000), @id CHAR(7) = concat('NL', DBO.numberZerosString(NEXT VALUE FOR SEQ_NL_ID))
BEGIN
    BEGIN TRAN
        BEGIN TRY
            INSERT INTO NGUYEN_LIEU(MaNL,TENNL,DON_GIA)
            VALUES(@id,@input_tennl,@input_dongia);
            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 42: Stored Procedure Insert Nguyen_Lieu

- Procedure Insert CT_Thung

```
--25 Insert chi tiet THUNG
CREATE  or ALTER  PROCEDURE PRC_INSERT_CTTHUNG
                                        @input_mathung char(7),
                                        @input_manl char(7),
                                        @input_soluong INT

AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            INSERT INTO CT_THUNG
            VALUES(@input_mathung,@input_manl,@input_soluong);
            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 43: Stored Procedure Insert CT_Thung

❖ Procedures Update

- Procedure Update Phieu_nhap

```
--UPDATE PhieuNhap
CREATE OR ALTER PROCEDURE PRC_UPDATE_PHIEUNHAP @input_mapn CHAR(7), @input_ngaynhap DATE
AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            IF @input_ngaynhap IS NULL
            BEGIN
                UPDATE PHIEU_NHAP
                SET  NGAY_NHAP_HANG = GETDATE()
                WHERE MA_PHIEU_NHAP = @input_mapn;
            END

            ELSE
            BEGIN
                UPDATE PHIEU_NHAP
                SET  NGAY_NHAP_HANG = @input_ngaynhap
                WHERE MA_PHIEU_NHAP = @input_mapn;
            END
            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 44: Stored Procedure Update Phieu_nhap

- Procedure Update CT_PhieuNhap_ Thoihan

```
--UPDATE CT_PN thoihan
create or ALTER PROCEDURE PRC_UDATE_THOI_HAN_CTPN
                                        @input_mapn char(7),
                                        @input_malo char(7),
                                        @input_thoihan INT

AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            IF @input_thoihan > 5
                BEGIN
                    RAISERROR('UPDATE UNSUCCESSFULLY, thoi han phai <= 5',10,1);
                END
                ELSE
                BEGIN
                    UPDATE CT_PHIEU_NHAP
                    SET THOI_HAN = @input_thoihan
                    WHERE MA_PHIEU_NHAP LIKE @input_mapn AND MALO LIKE @input_malo;
                END
            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 45: Stored Procedure Update CT_PhieuNhap_ Thoihan

- Procedure Update Phieu_xuat

```
--UPDATE PHIEUXUAT
CREATE OR ALTER PROCEDURE PRC_UPDATE_PHIEUXUAT @input_mapx CHAR(7), @input_ngayxuat DATE
AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
                IF @input_ngayxuat IS NULL
                BEGIN
                    UPDATE PHIEU_XUAT
                    SET  NGAY_XUAT_HANG = GETDATE()
                    WHERE MA_PHIEU_XUAT = @input_mapx;
                END
                ELSE
                BEGIN
                    UPDATE PHIEU_XUAT
                    SET  NGAY_XUAT_HANG = @input_ngayxuat
                    WHERE MA_PHIEU_XUAT = @input_mapx;
                END
            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 46: Stored Procedure Update Phieu_xuat

- Procedure Update Thung

```
--UPDATE THUNG
CREATE OR ALTER PROCEDURE PRC_UPDATE_THUNG @input_mathung CHAR(7),@input_tinhtrang VARCHAR
AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            UPDATE THUNG
            SET
                TINH_TRANG = @input_tinhtrang
            WHERE MA_THUNG = @input_mathung;
            COMMIT
        END TRY
        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 47: Stored Procedure Update Thung

❖ Procedure Delete

- Procedure Delete Nguyen_Lieu

```
--DELETE NGUYEN_LIEU
CREATE OR ALTER PROCEDURE PRC_DEL_NL @input_manl CHAR(7)
AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            UPDATE NGUYEN_LIEU
            SET DEL_STATUS = 'DELETED'
            WHERE MANL LIKE @input_manl;
            COMMIT
        END TRY
        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 48: Stored Procedure Delete Nguyen_Lieu

❖ Procedure Delete Thung

```
--DELETE THUNG
CREATE OR ALTER PROCEDURE PRC_DEL_TH @input_mathung CHAR(7)
AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            UPDATE THUNG
            SET DEL_STATUS = 'DELETED'
            WHERE MA_THUNG LIKE @input_mathung;
            COMMIT
        END TRY
        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 49: Stored Procedure Delete Thung

❖ Procedure Delete Lo_hang

```
--DELETE LO_HANG
CREATE OR ALTER PROCEDURE PRC_DEL_LH @input_malo CHAR(7)
AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            UPDATE LO_HANG
            SET DEL_STATUS = 'DELETED'
            WHERE MALO LIKE @input_malo;
            COMMIT
        END TRY
        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 50: Stored Procedure Delete Lo_hang

❖ Procedure Delete Phieu_nhap

```
--DELETE PHIEU_NHAP
CREATE OR ALTER PROCEDURE PRC_DEL_PN @input_mapn CHAR(7)
AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            UPDATE PHIEU_NHAP
            SET DEL_STATUS = 'DELETED'
            WHERE MA_PHIEU_NHAP LIKE @input_mapn;
            COMMIT
        END TRY
        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 51: Stored Procedure Delete Phieu_nhap

❖ Procedure Delete Phieu_xuat

```
--DELETE PHIEU_XUAT
CREATE OR ALTER PROCEDURE PRC_DEL_PX @input_mapx CHAR(7)
AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            UPDATE PHIEU_XUAT
            SET DEL_STATUS = 'DELETED'
            WHERE MA_PHIEU_XUAT LIKE @input_mapx;
            COMMIT
        END TRY
        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 52: Stored Procedure Delete Phieu_xuat

3.5. THIẾT KẾ TRIGGER

❖ Trigger Update Thùng bị hỏng

```
CREATE OR ALTER TRIGGER TRIG_UPDATE_THUNG
ON THUNG
AFTER UPDATE
AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            UPDATE CT_THUNG
            SET TONG_SO_NL = 0
            FROM Thung JOIN CT_Thung
            ON Thung.Ma_Thung = CT_Thung.Ma_Thung
            WHERE THUNG.TINH_TRANG LIKE 'Hỏng';
        COMMIT
        END TRY
        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 53: Trigger Update Thùng hỏng

❖ Trigger xem sơ bộ Tồn kho

```
create or ALTER PROCEDURE PRC_XEM_TON_KHO
AS
SET NOCOUNT ON;
BEGIN
    BEGIN TRY
        SELECT *
        FROM TON_KHO with (nolock)
        ORDER BY THANG,NAM;
    END TRY
    BEGIN CATCH
        ROLLBACK
        DECLARE @ErrorMessage VARCHAR(2000)
        SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
        RAISERROR(@ErrorMessage, 16, 1)
    END CATCH
END;
```

Hình 54: Trigger xem sơ bộ Tồn kho

❖ Trigger Update Tồn kho sau khi Insert Phiếu nhập

```
CREATE OR ALTER TRIGGER TRIG_UPDATE_TONKHO_PN
ON CT_Phieu_nhap
AFTER INSERT
AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            exec PRC_UPDATE_TON_KHO;
        COMMIT
        END TRY
        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 55: Trigger Update Tồn kho sau khi Insert Phiếu nhập

❖ Trigger Update Tồn kho sau khi Insert Phiếu xuất

```
CREATE OR ALTER TRIGGER TRIG_UPDATE_TONKHO_PX
ON CT_Phieu_xuat
AFTER INSERT
AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            exec PRC_UPDATE_TON_KHO;
        COMMIT
        END TRY
        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 56: Trigger Update Tồn kho sau khi Insert Phiếu xuất

❖ Trigger Update Tồn kho

```
create or ALTER PROCEDURE PRC_UPDATE_TON_KHO
AS
BEGIN
    SET TRANSACTION ISOLATION LEVEL REPEATABLE READ
    BEGIN TRAN
        BEGIN TRY
            DELETE FROM TON_KHO;

            INSERT INTO TON_KHO
            SELECT * FROM V_TON_KHO
            waitfor delay '00:00:10'
            COMMIT
        END TRY
        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 57: Trigger Update Tồn kho

❖ Trigger Xem báo cáo Tồn kho

```
create or ALTER PROCEDURE PRC_BAOCAO_TON_KHO
AS
SET NOCOUNT ON;
BEGIN
    BEGIN TRY
        SELECT *
        FROM TON_KHO
        ORDER BY THANG,NAM;
    END TRY
    BEGIN CATCH
        ROLLBACK
        DECLARE @ErrorMessage VARCHAR(2000)
        SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
        RAISERROR(@ErrorMessage, 16, 1)
    END CATCH
END;
```

Hình 58: Trigger Xem báo cáo Tồn kho

❖ Trigger Logging

```
CREATE TRIGGER [backup_objects]
ON DATABASE
FOR CREATE_PROCEDURE,
    ALTER_PROCEDURE,
    DROP_PROCEDURE,
    CREATE_TABLE,
    ALTER_TABLE,
    DROP_TABLE,
    CREATE_FUNCTION,
    ALTER_FUNCTION,
    DROP_FUNCTION,
    CREATE_VIEW,
    ALTER_VIEW,
    DROP_VIEW
AS

SET NOCOUNT ON

DECLARE @data XML
SET @data = EVENTDATA()

INSERT INTO changelog(databasename, eventtype,
    objectname, objecttype, sqlcommand, loginname)
VALUES(
@data.value('(/EVENT_INSTANCE/DatabaseName)[1]', 'varchar(256)'),
@data.value('(/EVENT_INSTANCE/EventType)[1]', 'varchar(50)'),
@data.value('(/EVENT_INSTANCE/ObjectName)[1]', 'varchar(256)'),
@data.value('(/EVENT_INSTANCE/ObjectType)[1]', 'varchar(25)'),
@data.value('(/EVENT_INSTANCE/TSQLCommand)[1]', 'varchar(max)'),
@data.value('(/EVENT_INSTANCE/LoginName)[1]', 'varchar(256)')
);

ENABLE TRIGGER [backup_objects] ON DATABASE;
```

Hình 59: Trigger Logging

3.6. THIẾT KẾ FUNCTION

- Function tạo mã

```
--FUNCTIONS
--5 NUMBERS
CREATE or alter FUNCTION numberZerosString(
 @n as bigint
) RETURNS char(5)
AS
BEGIN
    declare @returnValue char(5);
    set @returnValue = cast(format(@n,'00000')as char(5));
    RETURN @returnValue
END;
```

Hình 60: Funtion tạo mã

3.7. THIẾT KẾ GIAO TÁC (TRANSACTION)

- Tình huống có nhiều User cùng insert 1 bảng.

Xử lý bằng cách cho Transaction trong procedure insert và rollback khi deadlock.

VD: Transaction cho INSERT LO_HANG

```
--Insert Lohang
CREATE  or ALTER PROCEDURE PRC_INSERT_LH
                                        @input_tongthung INT

AS
DECLARE @ErrorMessage VARCHAR(2000), @id CHAR(7) = concat('LH', DBO.numberZerosString(NEXT VALUE FOR SEQ_LH_ID))
BEGIN
    BEGIN TRAN
        BEGIN TRY
            INSERT INTO LO_HANG (MaLo,TONG_SO_THUNG)
            VALUES(@id,@input_tongthung);
            COMMIT
        END TRY

        BEGIN CATCH
            ROLLBACK
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 61: Transaction cho Insert Lo_hang

- Tình huống nhiều User cùng Update 1 bảng 1 lúc

Xử lý bằng cách cho Transaction trong Procedure Update

VD: Transaction cho UPDATE THUNG

```
--UPDATE THUNG
CREATE OR ALTER PROCEDURE PRC_UPDATE_THUNG @input_mathung CHAR(7),@input_tinhtrang VARCHAR
AS
BEGIN
    BEGIN TRAN
        BEGIN TRY
            UPDATE THUNG
            SET
                TINH_TRANG = @input_tinhtrang
            WHERE MA_THUNG = @input_mathung;
            COMMIT
        END TRY
        BEGIN CATCH
            ROLLBACK
            DECLARE @ErrorMessage VARCHAR(2000)
            SELECT @ErrorMessage = 'Lỗi: ' + ERROR_MESSAGE()
            RAISERROR(@ErrorMessage, 16, 1)
        END CATCH
END;
```

Hình 62: Transaction cho Update Thung

3.8. VẤN ĐỀ XỬ LÝ ĐỒNG THỜI

Các vấn đề:

- LOST DATA:

Mô tả: 1 transaction cập nhật tồn kho đang diễn ra thì 1 transaction cũng cập nhật tồn kho và dẫn tới việc thất thoát dữ liệu:

Bảng 13: Transaction LOST DATA

T1	T2
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ BEGIN TRAN DELETE FROM TON_KHO; waitfor delay '00:00:10'	
	SET TRANSACTION ISOLATION LEVEL REPEATABLE READ BEGIN TRAN DELETE FROM TON_KHO; INSERT INTO TON_KHO SELECT * FROM V_TON_KHO COMMIT
INSERT INTO TON_KHO SELECT * FROM V_TON_KHO COMMIT	

Giải pháp:

Sử dụng mức cô lập REPEATABLE READ để việc UPDATE diễn ra tuần tự giữa các TRANSACTIONS.

- Giải quyết được 3 vấn đề: Lost Updated, Dirty Read và Unrepeatable Read
- Chưa giải quyết được vấn đề Phantom, do vẫn cho phép insert những dòng dữ liệu thỏa điều kiện thiết lập Slock.
- Slock được giữ đến hết giao dịch sẽ cản trở việc cập nhật dữ liệu của các giao dịch khác tuy nhiên vẫn việc delay sẽ không ảnh hưởng quy trình.

- DIRTY READ:

Mô tả: 1 transaction cập nhật tồn kho đang diễn ra thì 1 transaction khác cần xem gấp dữ liệu mà không cần tính chính xác.

Bảng 14: Transaction DIRTY DEAD

T1	T2
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ BEGIN TRAN DELETE FROM TON_KHO; waitfor delay '00:00:10'	
	SELECT * FROM TON_KHO with (nolock) ORDER BY THANG,NAM;
INSERT INTO TON_KHO SELECT * FROM V_TON_KHO COMMIT	

Giải pháp:

Sử dụng READ UNCOMMITED/ WITH (NOLOCK) để xem dữ liệu bẩn nhưng sẽ giúp việc truy suất diễn ra nhanh chóng.

- Giải quyết vấn đề Lost Updated

Có khả năng xảy ra 3 vấn đề của truy xuất đồng thời: Dirty Read, Unrepeatable Read, Phantom tuy nhiên sẽ không ảnh hưởng đến quy trình.

- COMMITED READ:

Mô tả: 1 transaction cập nhật tồn kho đang diễn ra thì 1 transaction khác cần xem dữ liệu chính xác để lập báo cáo.

Bảng 15: Transaction COMMITED READ

T1	T2
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ BEGIN TRAN DELETE FROM TON_KHO; waitfor delay '00:00:10' INSERT INTO TON_KHO SELECT * FROM V_TON_KHO COMMIT	
	SET TRANSACTION ISOLATION LEVEL READ COMMITTED SELECT * FROM TON_KHO ORDER BY THANG,NAM;

Giải pháp:

Sử dụng READ COMMITED mặc định của SQL server để xem dữ liệu đã commited.

- Giải quyết vấn đề Dirty Read, hỗ trợ việc báo cáo.
- Tuy phải chờ các transaction T1 xử lý xong thì T2 mới xử lý nhưng do tính chất nghiệp vụ báo cáo nên việc delay giữa các transactions là chấp nhận được.

3.9 . PHÂN QUYỀN

- Tạo User

```
CREATE LOGIN BINHMINH WITH PASSWORD = 'BINHMINH123';
CREATE LOGIN GIAKHUONG WITH PASSWORD = 'GIAKHUONG123';
CREATE LOGIN VANMINH WITH PASSWORD = 'VANMINH123';

CREATE USER BINHMINH FOR LOGIN BINHMINH;
CREATE USER GIAKHUONG FOR LOGIN GIAKHUONG;
CREATE USER VANMINH FOR LOGIN VANMINH;
```

Hình 63: Tạo User

- Gán quyền Admin

```
CREATE ROLE ROLE_ADMIN;
--GRANT TABLE
GRANT SELECT,UPDATE,INSERT,DELETE ON CT_LO_HANG TO ROLE_ADMIN;
GRANT SELECT,UPDATE,INSERT,DELETE ON CT_PHIEU_NHAP TO ROLE_ADMIN;
GRANT SELECT,UPDATE,INSERT,DELETE ON CT_PHIEU_XUAT TO ROLE_ADMIN;
GRANT SELECT,UPDATE,INSERT,DELETE ON CT_THUNG TO ROLE_ADMIN;
GRANT SELECT,UPDATE,INSERT,DELETE ON LO_HANG TO ROLE_ADMIN;
GRANT SELECT,UPDATE,INSERT,DELETE ON NGUYEN_LIEU TO ROLE_ADMIN;
GRANT SELECT,UPDATE,INSERT,DELETE ON PHIEU_NHAP TO ROLE_ADMIN;
GRANT SELECT,UPDATE,INSERT,DELETE ON PHIEU_XUAT TO ROLE_ADMIN;
GRANT SELECT,UPDATE,INSERT,DELETE ON THUNG TO ROLE_ADMIN;
GRANT SELECT,UPDATE,INSERT,DELETE ON TON_KHO TO ROLE_ADMIN;
--GRANT VIEW
GRANT SELECT ON NHAP_KHO TO ROLE_ADMIN;
GRANT SELECT ON XUAT_KHO TO ROLE_ADMIN;
GRANT SELECT ON V_TON_KHO TO ROLE_ADMIN;
--GRANT SCHEMA
GRANT EXECUTE ON SCHEMA:: DBO TO ROLE_ADMIN;
```

Hình 64: Gán quyền Admin

- Gán quyền Quản lý

```
CREATE ROLE ROLE_MANAGER;
--GRANT TABLE
GRANT SELECT,UPDATE,INSERT ON CT_LO_HANG TO ROLE_MANAGER;
GRANT SELECT,UPDATE,INSERT ON CT_PHIEU_NHAP TO ROLE_MANAGER;
GRANT SELECT,UPDATE,INSERT ON CT_PHIEU_XUAT TO ROLE_MANAGER;
GRANT SELECT,UPDATE,INSERT ON CT_THUNG TO ROLE_MANAGER;
GRANT SELECT,UPDATE,INSERT ON LO_HANG TO ROLE_MANAGER;
GRANT SELECT,UPDATE,INSERT ON NGUYEN_LIEU TO ROLE_MANAGER;
GRANT SELECT,UPDATE,INSERT ON PHIEU_NHAP TO ROLE_MANAGER;
GRANT SELECT,UPDATE,INSERT ON PHIEU_XUAT TO ROLE_MANAGER;
GRANT SELECT,UPDATE,INSERT ON THUNG TO ROLE_MANAGER;
GRANT SELECT,UPDATE,INSERT ON TON_KHO TO ROLE_MANAGER;
--GRANT VIEW
GRANT SELECT ON NHAP_KHO TO ROLE_MANAGER;
GRANT SELECT ON XUAT_KHO TO ROLE_MANAGER;
GRANT SELECT ON V_TON_KHO TO ROLE_MANAGER;
--GRANT SCHEMA
GRANT EXECUTE ON SCHEMA:: DBO TO ROLE_MANAGER;
```

Hình 65: Gán quyền Quản lý

- Gán quyền Nhân viên

```
CREATE ROLE ROLE_STAFF;
--GRANT TABLE
GRANT SELECT,UPDATE,INSERT ON CT_LO_HANG TO ROLE_STAFF;
GRANT SELECT,UPDATE,INSERT ON CT_PHIEU_NHAP TO ROLE_STAFF;
GRANT SELECT,UPDATE,INSERT ON CT_PHIEU_XUAT TO ROLE_STAFF;
GRANT SELECT,UPDATE,INSERT ON CT_THUNG TO ROLE_STAFF;
GRANT SELECT,UPDATE,INSERT ON LO_HANG TO ROLE_STAFF;
GRANT SELECT,UPDATE,INSERT ON NGUYEN_LIEU TO ROLE_STAFF;
GRANT SELECT,UPDATE,INSERT ON PHIEU_NHAP TO ROLE_STAFF;
GRANT SELECT,UPDATE,INSERT ON PHIEU_XUAT TO ROLE_STAFF;
GRANT SELECT,UPDATE,INSERT ON THUNG TO ROLE_STAFF;
GRANT SELECT,UPDATE,INSERT ON TON_KHO TO ROLE_STAFF;
--GRANT VIEW
GRANT SELECT ON NHAP_KHO TO ROLE_STAFF;
GRANT SELECT ON XUAT_KHO TO ROLE_STAFF;
GRANT SELECT ON V_TON_KHO TO ROLE_STAFF;
--GRANT SCHEMA
GRANT EXECUTE ON SCHEMA:: DBO TO ROLE_STAFF;
--REVOKE
REVOKE EXECUTE ON DBO.PRC_DEL_LH FROM ROLE_STAFF;
REVOKE EXECUTE ON DBO.PRC_DEL_NL FROM ROLE_STAFF;
REVOKE EXECUTE ON DBO.PRC_DEL_PN FROM ROLE_STAFF;
REVOKE EXECUTE ON DBO.PRC_DEL_PX FROM ROLE_STAFF;
REVOKE EXECUTE ON DBO.PRC_DEL_TH FROM ROLE_STAFF;
REVOKE EXECUTE ON DBO.PRC_BAOCAO_TON_KHO FROM ROLE_STAFF;
----------------------GRANT ROLE
ALTER ROLE ROLE_ADMIN ADD MEMBER VANMINH;
ALTER ROLE ROLE_MANAGER ADD MEMBER GIAKHUONG;
ALTER ROLE ROLE_STAFF ADD MEMBER BINHMINH;
```

Hình 66: Gán quyền Nhân viên

KẾT LUẬN

- Kết quả đạt được:
  - Quản lý thông tin kho.
  - Nhập hàng.
  - Xuất hàng.
  - Kiểm kê, báo cáo và thống kê hàng hóa (tồn kho).
- Hướng phát triển mở rộng ứng dụng trong tương lai
  - Nâng cấp cơ sở dữ liệu sao cho phù hợp nhất với thực tế và có thể dễ dàng sử dụng, không tốn nhiều bộ nhớ lưu trữ.
  - Nâng cấp bảo mật hệ thống cơ sở dữ liệu.
  - Kết nối dữ liệu, xây dựng một phần mềm hoàn chỉnh.

TÀI LIỆU THAM KHẢO

CREATE TRIGGER . (2022, 15 12). Được truy lục từ Microsoft: https://learn.microsoft.com/en-us/sql/t-sql/statements/create-trigger-transact-sql?view=sql-server-ver16

Huyền, T. (2020, 10 22). *Chỉ mục (INDEX) trong SQL*. Được truy lục từ Quantrimang: https://quantrimang.com/hoc/chi-muc-index-trong-sql-162405

Rakesh. (2021, 02 08). *Functions In SQL Server*. Được truy lục từ C#Corner : https://www.c-sharpcorner.com/UploadFile/b926a6/function-operation-in-sql-database/

Stored Procedure in SQL Server. (không ngày tháng). Được truy lục từ javaTpoint: https://www.javatpoint.com/stored-procedure-in-sql-server

Trung, N. T. (2022). Hướng dẫn thực hiện đề tài môn Hệ quản trị cơ sở dữ liệu.

PHỤ LỤC:

STT	MSSV	Họ và tên	Nội dung thực hiện	Trưởng nhóm
1	19DH110076	Phạm Gia Khương	Chương 1, 2, 3	X
2	19DH110596	Nguyễn Văn Bình Minh	Chương 1, 2, 3	
3	19DH110060	Trần Văn Minh	Chương 1, 2, 3	