BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

----💻💻💻----

ĐỀ TÀI CUỐI KỲ MÔN HỆ QUẢN TRỊ CSDL

TÊN ĐỀ TÀI: QUẢN LÝ NHÀ SÁCH

GVHD: *TS. Nguyễn Thành Sơn*

Lớp HP: *DBMS330284_04*

Nhóm thực hiện: *Nhóm 5*

Nguyễn Ngọc Hải 23133021

Học kỳ: *1*

Năm học: *2025-2026*

Thành phố Hồ Chí Minh, tháng 09 năm 2025

DANH SÁCH SINH VIÊN NHÓM THỰC HIỆN HỌC KÌ 1 NĂM HỌC 2025 – 2026

Nhóm 5

Đề tài: Quản lý bán sách

23133021	Nguyễn Ngọc Hải

Nhận xét của giảng viên

.....

.....

.....

.....

.....

.....

.....

.....

.....

Ngày …… Tháng …… Năm ……

Giáo viên chấm điểm

MỤC LỤC

LỜI CẢM ƠN

Kính gửi thầy Nguyễn Thành Sơn,

Em xin trân trọng gửi lời cảm ơn đến thầy vì đã tận tình giảng dạy và chia sẻ những kiến thức quý báu trong môn *Hệ quản trị cơ sở dữ liệu*. Qua các bài giảng và hướng dẫn của thầy, em không chỉ được trang bị nền tảng lý thuyết vững chắc mà còn có cơ hội tiếp cận với những ứng dụng thực tế hữu ích cho học tập và công việc sau này.

Trong suốt quá trình thực hiện đồ án, thầy đã luôn quan tâm, định hướng và khích lệ em vượt qua khó khăn để hoàn thành nhiệm vụ. Chính sự tận tâm và nhiệt huyết ấy đã tạo động lực lớn, giúp em có thêm niềm tin và sự nỗ lực để đạt kết quả tốt.

Em xin chân thành cảm ơn và kính chúc thầy luôn dồi dào sức khỏe, thành công trong sự nghiệp giảng dạy, để tiếp tục truyền cảm hứng cho nhiều thế hệ sinh viên sau này.

Trân trọng,

Nguyễn Ngọc Hải

LỜI NÓI ĐẦU

Kính thưa thầy và các bạn,

Lời đầu tiên, em xin được giới thiệu đồ án kết thúc môn học Hệ quản trị cơ sở dữ liệu với đề tài "Xây dựng hệ thống quản lý bán sách". Đây là kết quả của quá trình học tập, tìm hiểu và vận dụng kiến thức đã được học vào một bài toán gần gũi với thực tế.

Trong bối cảnh nhu cầu đọc sách và buôn bán ngày càng phát triển, việc quản lý sách, khách hàng và hóa đơn trở thành một yêu cầu quan trọng. Hệ thống quản lý bán sách được xây dựng nhằm hỗ trợ cửa hàng trong các nghiệp vụ chính như: quản lý thông tin sách và loại sách, lập và quản lý hóa đơn bán hàng, theo dõi chi tiết số lượng và giá trị từng đơn hàng, cũng như tìm kiếm và tra cứu dữ liệu nhanh chóng khi cần thiết.

Bên cạnh đó, hệ thống còn chú trọng đến tính an toàn và hiệu quả thông qua việc phân quyền cho người dùng. Nhân viên có thể thực hiện các nghiệp vụ bán hàng thường ngày, trong khi quản trị viên có quyền quản lý toàn bộ cơ sở dữ liệu.

Thông qua việc thực hiện đề tài, em mong muốn tạo ra một mô hình quản lý đơn giản, dễ sử dụng và có thể đáp ứng được các yêu cầu cơ bản trong hoạt động bán sách.

Trong quá trình thực hiện, mặc dù đã rất cố gắng nhưng khó tránh khỏi những thiếu sót. Em rất mong nhận được sự đóng góp của thầy cô và các bạn để hoàn thiện hơn nữa sản phẩm của mình.

Em xin chân thành cảm ơn!

CHƯƠNG 1: TỔNG QUAN VỀ HỆ THỐNG

1. Đặc tả đề tài

1.1. Mô tả bài toán

Trong bối cảnh xã hội hiện đại, nhu cầu đọc sách ngày càng trở nên phổ biến và đa dạng. Các nhà sách, cửa hàng phát hành hay thư viện tư nhân đều phải đối mặt với khối lượng dữ liệu ngày một lớn, bao gồm hàng nghìn đầu sách khác nhau, nhiều loại hình và phân loại phong phú, cùng với hàng trăm giao dịch phát sinh mỗi ngày. Nếu việc quản lý còn dựa trên phương pháp thủ công như ghi chép sổ sách hay nhập liệu rời rạc bằng bảng tính đơn giản (Excel), người quản lý sẽ gặp nhiều hạn chế như:

- Tốn nhiều thời gian trong việc ghi chép và tìm kiếm thông tin. Ví dụ, khi khách hàng cần biết còn bao nhiêu cuốn *Đắc Nhân Tâm*, nhân viên phải lục tìm trong sổ hoặc bảng tính thủ công.
- Dữ liệu dễ sai sót và trùng lặp, vì không có cơ chế kiểm tra ràng buộc hay đồng bộ giữa các thông tin.
- Khó khăn trong việc theo dõi tồn kho: số lượng sách bán ra, số lượng nhập vào không được cập nhật tự động, dẫn đến tình trạng thiếu hoặc thừa hàng.
- Thiếu tính bảo mật và phân quyền: tất cả nhân viên có thể can thiệp vào dữ liệu, dễ dẫn đến mất mát hoặc chỉnh sửa ngoài ý muốn.

Đề tài *"Xây dựng hệ thống quản lý bán sách"* được đưa ra nhằm giải quyết những khó khăn trên. Mục tiêu là xây dựng một hệ thống cơ sở dữ liệu tập trung, vừa đảm bảo tính chính xác, vừa tự động hóa các nghiệp vụ quản lý, đồng thời cung cấp khả năng tra cứu nhanh chóng và bảo mật dữ liệu thông qua phân quyền người dùng.

Với hệ thống này, người quản lý có thể theo dõi toàn bộ quá trình kinh doanh, từ quản lý danh mục sách, lập hóa đơn cho khách hàng, cập nhật số lượng tồn kho, cho đến việc tra cứu lịch sử giao dịch. Nhờ đó, công tác quản lý được chuyên nghiệp hóa, tiết kiệm thời gian, giảm thiểu sai sót và nâng cao hiệu quả hoạt động kinh doanh của nhà sách.

1.2. Mô tả dữ liệu

Dữ liệu trong hệ thống được thiết kế có cấu trúc chặt chẽ, phản ánh trung thực nghiệp vụ bán sách trong thực tế. Các bảng dữ liệu được xây dựng xoay quanh bốn thực thể chính: Loại sách, Sách, Hóa đơn, và Chi tiết hóa đơn.

- Bảng Loại sách: Giúp phân loại đầu sách theo các nhóm như *Tiểu thuyết, Khoa*

học, Thiếu nhi…. Việc phân loại này không chỉ phục vụ cho công tác quản lý kho, mà còn thuận tiện cho khách hàng khi tra cứu theo thể loại yêu thích.

- Bảng Sách: Lưu trữ thông tin chi tiết của từng cuốn sách bao gồm tên sách, tác giả, loại sách, số lượng tồn kho và giá bán. Đây là bảng dữ liệu trung tâm, liên kết chặt chẽ với các bảng khác.
- Bảng Hóa đơn: Ghi nhận các giao dịch bán hàng, bao gồm ngày lập hóa đơn, tên khách hàng và số điện thoại liên hệ. Thông qua bảng này, nhà sách có thể theo dõi doanh thu theo từng ngày hoặc theo từng khách hàng.
- Bảng Chi tiết hóa đơn: Liên kết một hóa đơn cụ thể với các cuốn sách được bán trong hóa đơn đó, đồng thời lưu số lượng mỗi loại sách. Đây là nơi phản ánh trực tiếp mối quan hệ *nhiều – nhiều* giữa sách và hóa đơn.

Bên cạnh bốn bảng chính, hệ thống còn có các view để hiển thị dữ liệu thân thiện hơn với người dùng, các function để hỗ trợ tính toán (ví dụ: thành tiền, tổng tiền hóa đơn) và tìm kiếm nhanh, cùng các procedure để thực hiện các thao tác thêm, sửa, xóa dữ liệu một cách an toàn. Ngoài ra, trigger được xây dựng để tự động cập nhật số lượng sách trong kho sau mỗi lần phát sinh bán hàng, giúp dữ liệu luôn chính xác và đồng bộ. Việc thiết kế dữ liệu theo mô hình quan hệ chặt chẽ với khóa chính, khóa ngoại, ràng buộc toàn vẹn sẽ đảm bảo rằng hệ thống hạn chế tối đa sai sót, tránh tình trạng dữ liệu mâu thuẫn hoặc bị bỏ sót.

1.3. Mô tả chức năng

Hệ thống quản lý bán sách được xây dựng không chỉ để lưu trữ dữ liệu mà còn hỗ trợ thực hiện đầy đủ các nghiệp vụ cơ bản của một cửa hàng sách. Các chức năng chính bao gồm:

- Quản lý sách: Cho phép thêm mới sách khi có đầu sách mới, cập nhật thông tin sách khi có thay đổi về giá hoặc số lượng, và xóa sách khi không còn kinh doanh. Ngoài ra, hệ thống đảm bảo kiểm tra ràng buộc để tránh nhập sai dữ liệu.
- Quản lý loại sách: Người dùng có thể thêm mới hoặc chỉnh sửa các loại sách, đảm bảo danh mục sách luôn đa dạng và dễ quản lý.
- Quản lý hóa đơn: Hỗ trợ lập hóa đơn mới cho khách hàng, cập nhật thông tin khi có thay đổi, và xóa hóa đơn trong trường hợp cần thiết. Mỗi hóa đơn lưu thông tin khách hàng và ngày lập, giúp thuận tiện cho việc tra cứu và báo cáo doanh thu.

- Quản lý chi tiết hóa đơn: Cho phép thêm sách vào hóa đơn, thay đổi số lượng hoặc xóa nếu nhập sai. Khi phát sinh các thao tác này, hệ thống sẽ tự động cập nhật số lượng tồn kho trong bảng Sách.
- Tìm kiếm và tra cứu: Hệ thống cung cấp nhiều công cụ tìm kiếm như tìm sách theo tên, tác giả, loại; tìm hóa đơn theo khách hàng, số điện thoại hoặc ngày lập. Chức năng này giúp nhân viên nhanh chóng đáp ứng yêu cầu của khách hàng.
- Hiển thị: Thông qua các view, dữ liệu được trình bày rõ ràng và trực quan, ví dụ danh sách sách kèm loại sách, danh sách hóa đơn hoặc chi tiết hóa đơn.
- Phân quyền người dùng: Hệ thống đảm bảo an toàn dữ liệu bằng cách phân chia quyền hạn. Nhân viên có thể thực hiện các nghiệp vụ bán hàng, trong khi quản trị viên có toàn quyền kiểm soát cơ sở dữ liệu.

Với các chức năng này, hệ thống không chỉ đáp ứng nhu cầu quản lý nội bộ mà còn nâng cao trải nghiệm cho khách hàng, giúp cửa hàng hoạt động hiệu quả hơn.

1.4. Về phần giao diện

Mặc dù trọng tâm của đề tài là xây dựng cơ sở dữ liệu, nhưng phần giao diện cũng được quan tâm để minh họa cách hệ thống sẽ được sử dụng trong thực tế. Giao diện được thiết kế với tiêu chí đơn giản – trực quan – dễ thao tác, phù hợp với người dùng phổ thông.

- Giao diện quản lý sách: Hiển thị danh sách sách dưới dạng bảng, có cột tên sách, tác giả, loại, số lượng và giá bán. Người dùng có thể tìm kiếm sách theo từ khóa, thêm mới, chỉnh sửa hoặc xóa trực tiếp.
- Giao diện quản lý loại sách: Hiển thị danh sách các loại sách dưới dạng bảng, có cột mã loại sách, tên loại sách. Người dùng có thể tìm kiếm loại sách theo từ khóa, thêm mới, chỉnh sửa hoặc xóa trực tiếp.
- Giao diện quản lý hóa đơn: Cho phép tạo hóa đơn mới bằng cách nhập thông tin khách hàng, ngày lập và thêm sách từ danh mục. Các sách được chọn sẽ tự động tính thành tiền và tổng tiền hóa đơn.
- Giao diện chi tiết hóa đơn: Hiển thị rõ ràng từng sách trong hóa đơn, số lượng và giá bán. Người dùng có thể cập nhật hoặc xóa từng dòng chi tiết.
- Phân quyền giao diện: Khi đăng nhập, nhân viên chỉ nhìn thấy các chức năng liên quan đến bán hàng, trong khi quản trị viên có thể truy cập toàn bộ chức năng, bao gồm quản lý sách, loại sách và phân quyền người dùng.

Giao diện tuy đơn giản nhưng thể hiện rõ cách mà cơ sở dữ liệu được khai thác trong thực tế, đảm bảo tính trực quan và dễ áp dụng. Đây cũng là minh chứng cho thấy việc thiết kế cơ sở dữ liệu tốt sẽ giúp xây dựng ứng dụng thân thiện, hiệu quả hơn.

CHƯƠNG 2 : THIẾT KẾ CƠ SỞ DỮ LIỆU

2.1. Thiết kế cơ sở dữ liệu mức quan niệm

Từ mô tả về dữ liệu cần có ở phần mô tả của bài toán ta hình thành được sơ đồ thực thể kết hợp (ERD).

2.2. Thiết kế cơ sở dữ liệu mức logic

2.2.1. Từ sơ đồ thực thể kết hợp (ERD), ta có các lược đồ quan hệ

- Sach(MaSach, TenSach, TacGia, SoLuong, GiaBan, MaLoaiSach)
- LoaiSach(MaLoaiSach, TenLoaiSach)
- ChiTietHoaDon(MaSach, MaHoaDon, SoLuong)
- HoaDon(MaHoaDon, NgayLapHoaDon, TenKhachHang, SoDienThoai)

2.2.2. Mô tả các thuộc tính, kiểu dữ liệu và các ràng buộc

Sach

	Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
1	MaSach	INT	Mã sách	PRIMARY KEY
2	TenSach	NVARCHAR(200)	Tên sách	NOT NULL
3	TacGia	NVARCHAR(100)	Tác giả	NOT NULL
4	SoLuong	INT	Số lượng	(SoLuong >= 0)
5	GiaBan	DECIMAL(10,2)	Giá bán	CHECK (GiaBan >= 0)
6	MaLoaiSach	INT	Mã loại sách	NOT NULL, FOREIGN KEY (MaLoaiSach) REFERENCES LoaiSach(MaLoaiSach)

LoaiSach

	Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
1	MaLoaiSach	INT	Mã Loại Sách	PRIMARY KEY
2	TenLoaiSach	NVARCHAR(100)	Tên loại sách	NOT NULL

ChiTietHoaDon

PRIMARY KEY (MaHoaDon, MaSach)

	Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
1	MaHoaDon	INT	Mã môn học	NOT NULL, FOREIGN KEY (MaHoaDon) REFERENCES HoaDon(MaHoaDon),

2	MaSach	INT	Tên môn học	NOT NULL, FOREIGN KEY (MaSach) REFERENCES Sach(MaSach)
3	SoLuong	INT	Mã giảng viên	CHECK (SoLuong > 0)

HoaDon

	Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
1	MaHoaDon	INT	Mã hóa đơn	PRIMARY KEY
2	NgayLapHoaDon	DATE	Ngày lập hóa đơn	NOT NULL
3	TenKhachHang	NVARCHAR(100)	Tên khách hàng	NOT NULL
4	SDT	NVARCHAR(20)	Số điện thoại	

a. Cài đặt cơ sở dữ liệu

Bảng sách

```
CREATE TABLE Sach (
    MaSach INT PRIMARY KEY IDENTITY(1,1),
    TenSach NVARCHAR(200) NOT NULL,
    MaLoaiSach INT NOT NULL,
    TacGia NVARCHAR(100) NOT NULL,
    SoLuong INT CHECK (SoLuong >= 0),
    GiaBan DECIMAL(10,2) CHECK (GiaBan >= 0),
    FOREIGN KEY (MaLoaiSach) REFERENCES LoaiSach(MaLoaiSach)
);
```

Bảng loại sách

```
CREATE TABLE LoaiSach (
    MaLoaiSach INT PRIMARY KEY IDENTITY(1,1),
    TenLoaiSach NVARCHAR(100) NOT NULL
);
```

Bảng chi tiết hóa đơn

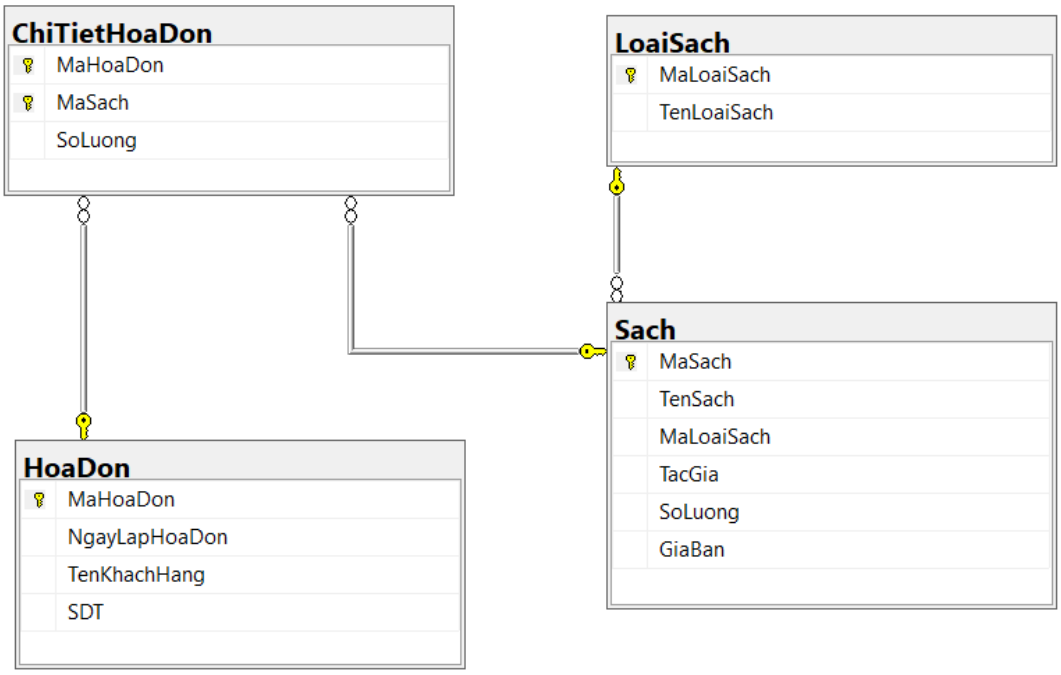
```
CREATE TABLE ChiTietHoaDon (
```

```
    MaHoaDon INT,
    MaSach INT,
    SoLuong INT CHECK (SoLuong > 0),
    PRIMARY KEY (MaHoaDon, MaSach),
    FOREIGN KEY (MaHoaDon) REFERENCES HoaDon(MaHoaDon),
    FOREIGN KEY (MaSach) REFERENCES Sach(MaSach)
);
```

Bảng hóa đơn

```
CREATE TABLE HoaDon (
    MaHoaDon INT PRIMARY KEY IDENTITY(1,1),
    NgayLapHoaDon DATE NOT NULL,
    TenKhachHang NVARCHAR(100) NOT NULL,
    SDT NVARCHAR(20)
);
```

Sơ đồ quan hệ:



b. Các trigger

TRIGGER: Tự động trừ số lượng sách khi thêm/sửa/xóa chi tiết hóa đơn

```
-- Xóa trigger cũ nếu tồn tại
IF EXISTS (SELECT * FROM sys.triggers WHERE name =
'tr_CapNhatSoLuongSach_ChiTietHoaDon')
BEGIN
    DROP TRIGGER tr_CapNhatSoLuongSach_ChiTietHoaDon;
END
GO

-- Tạo trigger mới
CREATE TRIGGER tr_CapNhatSoLuongSach_ChiTietHoaDon
ON ChiTietHoaDon
AFTER INSERT, UPDATE, DELETE
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @ErrorMessage NVARCHAR(4000);

    BEGIN TRY
        -- XỬ LÝ INSERT (Thêm chi tiết hóa đơn)
        IF EXISTS (SELECT * FROM inserted) AND NOT EXISTS (SELECT * FROM
deleted)
        BEGIN
            -- Cập nhật số lượng sách (trừ đi số lượng bán)
            UPDATE Sach
            SET SoLuong = Sach.SoLuong - i.SoLuong
            FROM Sach
            INNER JOIN inserted i ON Sach.MaSach = i.MaSach;

            -- Kiểm tra số lượng sau khi trừ có âm không
            IF EXISTS (
                SELECT 1
```

```
                FROM Sach s
                INNER JOIN inserted i ON s.MaSach = i.MaSach
                WHERE s.SoLuong < 0
            )
            BEGIN
                -- Lấy thông tin sách có số lượng âm để thông báo
                DECLARE @TenSach NVARCHAR(200), @SoLuongConLai INT, @SoLuongBan
INT;

                SELECT TOP 1
                    @TenSach = s.TenSach,
                    @SoLuongConLai = s.SoLuong + i.SoLuong, -- Số lượng trước
khi trừ
                    @SoLuongBan = i.SoLuong
                FROM Sach s
                INNER JOIN inserted i ON s.MaSach = i.MaSach
                WHERE s.SoLuong < 0;

                -- Rollback và thông báo lỗi
                SET @ErrorMessage = N'Không đủ số lượng sách "' + @TenSach +
N'". ' +
                                    N'Số lượng còn lại: ' + CAST(@SoLuongConLai
AS NVARCHAR(10)) + N', ' +
                                    N'số lượng muốn bán: ' + CAST(@SoLuongBan AS
NVARCHAR(10)) + N'.';
                RAISERROR(@ErrorMessage, 16, 1);
                RETURN;
            END
        END

        -- XỬ LÝ UPDATE (Sửa chi tiết hóa đơn)
        ELSE IF EXISTS (SELECT * FROM inserted) AND EXISTS (SELECT * FROM
deleted)
        BEGIN
            -- Hoàn lại số lượng cũ (cộng lại số lượng đã trừ trước đó)
            UPDATE Sach
            SET SoLuong = Sach.SoLuong + d.SoLuong
            FROM Sach
            INNER JOIN deleted d ON Sach.MaSach = d.MaSach;

            -- Trừ đi số lượng mới
            UPDATE Sach
            SET SoLuong = Sach.SoLuong - i.SoLuong
            FROM Sach
            INNER JOIN inserted i ON Sach.MaSach = i.MaSach;

            -- Kiểm tra số lượng sau khi cập nhật có âm không
            IF EXISTS (
                SELECT 1
                FROM Sach s
                INNER JOIN inserted i ON s.MaSach = i.MaSach
                WHERE s.SoLuong < 0
            )
            BEGIN
                -- Lấy thông tin sách có số lượng âm để thông báo
                DECLARE @TenSachUpdate NVARCHAR(200), @SoLuongConLaiUpdate INT,
@SoLuongBanUpdate INT;

                SELECT TOP 1
                    @TenSachUpdate = s.TenSach,
                    @SoLuongConLaiUpdate = s.SoLuong + i.SoLuong, -- Số lượng
sau khi hoàn lại cũ
                    @SoLuongBanUpdate = i.SoLuong
                FROM Sach s
                INNER JOIN inserted i ON s.MaSach = i.MaSach
```

```
                INNER JOIN deleted d ON s.MaSach = d.MaSach
                WHERE s.SoLuong < 0;

                -- Rollback và thông báo lỗi
                SET @ErrorMessage = N'Không đủ số lượng sách "' +
@TenSachUpdate + N'". ' +
                                    N'Số lượng có thể bán: ' +
CAST(@SoLuongConLaiUpdate AS NVARCHAR(10)) + N', ' +
                                    N'số lượng muốn cập nhật: ' +
CAST(@SoLuongBanUpdate AS NVARCHAR(10)) + N'.';
                RAISERROR(@ErrorMessage, 16, 1);
                RETURN;
            END
        END

        -- XỬ LÝ DELETE (Xóa chi tiết hóa đơn)
        ELSE IF EXISTS (SELECT * FROM deleted) AND NOT EXISTS (SELECT * FROM
inserted)
        BEGIN
            -- Hoàn lại số lượng đã bán (cộng lại vào kho)
            UPDATE Sach
            SET SoLuong = Sach.SoLuong + d.SoLuong
            FROM Sach
            INNER JOIN deleted d ON Sach.MaSach = d.MaSach;
        END

    END TRY
    BEGIN CATCH
        -- Xử lý lỗi
        DECLARE @ErrorNumber INT = ERROR_NUMBER();
        DECLARE @ErrorSeverity INT = ERROR_SEVERITY();
        DECLARE @ErrorState INT = ERROR_STATE();
        DECLARE @ErrorProcedure NVARCHAR(128) = ERROR_PROCEDURE();
        DECLARE @ErrorLine INT = ERROR_LINE();
        DECLARE @ErrorMessageCatch NVARCHAR(4000) = ERROR_MESSAGE();

        -- Re-raise lỗi với thông tin chi tiết
        SET @ErrorMessage = N'Lỗi trong trigger cập nhật số lượng sách: ' +
@ErrorMessageCatch;
        RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState);
    END CATCH
END;
GO
```

c. Các view

View hiển thị thông tin sách kết hợp với loại sách

```
create view vw_DanhSachSach
as
select
    MaSach as [Mã Sách],
    TenSach as [Tên Sách],
    TenLoaiSach as [Tên Loại Sách],
    TacGia as [Tác Giả],
    SoLuong as [Số Lượng],
    GiaBan as [Giá Bán]
from Sach s
inner join LoaiSach ls on s.MaLoaiSach = ls.MaLoaiSach;
go
```

View hiển thị thông tin loại sách

```
create view vw_DanhSachLoaiSach
as
select
    MaLoaiSach as [Mã Loại Sách],
    TenLoaiSach as [Tên Loại Sách]
from LoaiSach;
go
```

View cho ComboBox loại sách (không có alias)

```
create view vw_LoaiSach
as
select
    MaLoaiSach,
    TenLoaiSach
from LoaiSach;
go
```

View hiển thị thông tin hóa đơn

```
create view vw_DanhSachHoaDon
as
select
    MaHoaDon as [Mã Hóa Đơn],
    NgayLapHoaDon as [Ngày Lập Hoá Đơn],
    TenKhachHang as [Tên Khách Hàng],
    SDT as [Số Điện Thoại]
from HoaDon;
go
```

View hiển thị chi tiết hóa đơn

```
create view vw_ChiTietHoaDon
as
select
    ct.MaHoaDon as [Mã Hóa Đơn],
    ct.MaSach as [Mã Sách],
    s.TenSach as [Tên Sách],
    s.TacGia as [Tác Giả],
    ct.SoLuong as [Số Lượng],
    s.GiaBan as [Giá Bán],
    dbo.fn_TinhThanhTien(ct.SoLuong, s.GiaBan) as [Thành Tiền]
from ChiTietHoaDon ct
inner join Sach s on ct.MaSach = s.MaSach;
go
```

CHƯƠNG 3 : CÀI ĐẶT CHỨC NĂNG

3.1. Kết nối cơ sở dữ liệu

```
4.  private string connectString = @"Data Source=LAPTOP-
    57QKKMG0\SQLEXPRESS;Initial Catalog=QLBanSach;Integrated
    Security=True;Encrypt=True;TrustServerCertificate=True";
5.
6.  public DataTable execQuery(string query)
7.  {
8.
9.      DataTable data = new DataTable();
10.     using (SqlConnection con = new SqlConnection(connectString))
11.     {
12.         con.Open();
13.         SqlCommand cmd = new SqlCommand(query, con);
14.
15.         SqlDataAdapter adapter = new SqlDataAdapter(cmd);
16.
17.         adapter.Fill(data);
18.
19.         con.Close();
20.
21.     }
22.     return data;
23. }
24.
25. public int execNonQuery(string query)
26.
27. {
28.
29.     int data = 0;
30.     using (SqlConnection con = new SqlConnection(connectString))
31.     {
32.
33.         con.Open();
34.
35.         SqlCommand cmd = new SqlCommand(query, con);
36.
37.         data = cmd.ExecuteNonQuery();
38.
39.         con.Close();
40.
41.     }
42.     return data;
43. }
44.
45.
46.
47. public object execScaler(string query)
48. {
49.
50.     object data = 0;
51.
52.     using (SqlConnection con = new SqlConnection(connectString))
53.     {
54.
55.         con.Open();
56.
57.         SqlCommand cmd = new SqlCommand(query, con);
58.
59.         data = cmd.ExecuteScalar();
60.
61.         con.Close();
```

```
62.
63.     }
64.     return data;
65. }
```

3.2. Thủ tục

Thêm Sách (UPSERT) với validation

```
create proc proc_ThemSach
@tenSach nvarchar(256), @maLoaiSach int, @tacGia nvarchar(256), @soLuong int,
@giaBan float
as
begin
	-- Validation dữ liệu đầu vào
	if @tenSach is null or LTRIM(RTRIM(@tenSach)) = ''
	begin
		RAISERROR(N'Tên sách không được để trống!', 16, 1);
		return;
	end

	if @maLoaiSach is null or @maLoaiSach <= 0
	begin
		RAISERROR(N'Mã loại sách không hợp lệ!', 16, 1);
		return;
	end

	if @tacGia is null or LTRIM(RTRIM(@tacGia)) = ''
	begin
		RAISERROR(N'Tác giả không được để trống!', 16, 1);
		return;
	end

	if @soLuong is null or @soLuong < 0
	begin
		RAISERROR(N'Số lượng phải lớn hơn hoặc bằng 0!', 16, 1);
		return;
	end

	if @giaBan is null or @giaBan <= 0
	begin
		RAISERROR(N'Giá bán phải lớn hơn 0!', 16, 1);
		return;
	end

	-- Kiểm tra loại sách có tồn tại không
	if not exists(select 1 from LoaiSach where MaLoaiSach = @maLoaiSach)
	begin
		RAISERROR(N'Loại sách không tồn tại!', 16, 1);
		return;
	end

	-- Trim dữ liệu trước khi xử lý
	set @tenSach = LTRIM(RTRIM(@tenSach));
	set @tacGia = LTRIM(RTRIM(@tacGia));

	-- Kiểm tra xem sách đã tồn tại chưa (dựa trên tên sách và tác giả)
	if exists(select 1 from Sach where TenSach = @tenSach and TacGia =
@tacGia)
	begin
		-- Nếu đã tồn tại thì cộng thêm số lượng và cập nhật giá
		update Sach
		set MaLoaiSach = @maLoaiSach,
```

```
                    SoLuong = SoLuong + @soLuong,  -- Cộng thêm số lượng
                    GiaBan = @giaBan
            where TenSach = @tenSach and TacGia = @tacGia;
      end
      else
      begin
            -- Nếu chưa tồn tại thì thêm mới
            insert into Sach(TenSach, MaLoaiSach, TacGia, SoLuong, GiaBan)
            values(@tenSach, @maLoaiSach, @tacGia, @soLuong, @giaBan);
      end
end;
go
```

Xử lý bên C#

```
private void btnSachThem_Click(object sender, EventArgs e)
{
    // Validate dữ liệu đầu vào
    if (string.IsNullOrWhiteSpace(txtSachTenSach.Text))
    {
        MessageBox.Show("Vui lòng nhập tên sách!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        txtSachTenSach.Focus();
        return;
    }

    if (maSachLoaiSach <= 0)
    {
        MessageBox.Show("Vui lòng chọn loại sách!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        cbSachLoaiSach.Focus();
        return;
    }

    if (string.IsNullOrWhiteSpace(txtSachTacGia.Text))
    {
        MessageBox.Show("Vui lòng nhập tác giả!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        txtSachTacGia.Focus();
        return;
    }

    if (numSachSoLuong.Value <= 0)
    {
        MessageBox.Show("Số lượng phải lớn hơn 0!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        numSachSoLuong.Focus();
        return;
    }

    if (numSachGiaBan.Value <= 0)
    {
        MessageBox.Show("Giá bán phải lớn hơn 0!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        numSachGiaBan.Focus();
        return;
    }

    // Kiểm tra trùng sách trước khi thêm
    string newTenSach = txtSachTenSach.Text.Trim();
    string newTacGia = txtSachTacGia.Text.Trim();
    int newMaLoaiSach = maSachLoaiSach;
```

```
    string checkQuery = "SELECT COUNT(*) FROM Sach WHERE " +
                        "UPPER(LTRIM(RTRIM(TenSach))) = UPPER(N'" +
newTenSach.Replace("'", "''") + "') AND " +
                        "UPPER(LTRIM(RTRIM(TacGia))) = UPPER(N'" +
newTacGia.Replace("'", "''") + "') AND " +
                        "MaLoaiSach = " + newMaLoaiSach;

    try
    {
        var checkResult = dataProvider.execScaler(checkQuery);
        int duplicateCount = Convert.ToInt32(checkResult ?? 0);

        if (duplicateCount > 0)
        {
            MessageBox.Show("Sách với tên, tác giả và loại sách này đã tồn
tại!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
            txtSachTenSach.Focus();
            return;
        }

        StringBuilder query = new StringBuilder("EXEC proc_ThemSach");
        query.Append(" @tenSach = N'" + newTenSach.Replace("'", "''") + "'");
        query.Append(",@maLoaiSach = " + newMaLoaiSach);
        query.Append(",@tacGia = N'" + newTacGia.Replace("'", "''") + "'");
        query.Append(",@soLuong = " + numSachSoLuong.Value);
        query.Append(",@giaBan = " + numSachGiaBan.Value);

        dataProvider.execNonQuery(query.ToString());
        loadDgSach();
        MessageBox.Show("Thêm sách thành công!", "Thành Công",
MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
    }
    catch (Exception ex)
    {
        MessageBox.Show("Thêm sách không thành công! " + ex.Message, "Lỗi",
MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Cập nhật sách với validation

```
create proc proc_CapNhatSach
@maSach int, @tenSach nvarchar(256), @maLoaiSach int, @tacGia nvarchar(256),
@soLuong int, @giaBan float
as
begin
      -- Validation dữ liệu đầu vào
      if @maSach is null or @maSach <= 0
      begin
            RAISERROR(N'Mã sách không hợp lệ!', 16, 1);
            return;
      end

      if @tenSach is null or LTRIM(RTRIM(@tenSach)) = ''
      begin
            RAISERROR(N'Tên sách không được để trống!', 16, 1);
            return;
      end

      if @maLoaiSach is null or @maLoaiSach <= 0
      begin
            RAISERROR(N'Mã loại sách không hợp lệ!', 16, 1);
            return;
```

```
	end

	if @tacGia is null or LTRIM(RTRIM(@tacGia)) = ''
	begin
		RAISERROR(N'Tác giả không được để trống!', 16, 1);
		return;
	end

	if @soLuong is null or @soLuong < 0
	begin
		RAISERROR(N'Số lượng phải lớn hơn hoặc bằng 0!', 16, 1);
		return;
	end

	if @giaBan is null or @giaBan <= 0
	begin
		RAISERROR(N'Giá bán phải lớn hơn 0!', 16, 1);
		return;
	end

	-- Kiểm tra sách có tồn tại không
	if not exists(select 1 from Sach where MaSach = @maSach)
	begin
		RAISERROR(N'Không tìm thấy sách cần cập nhật!', 16, 1);
		return;
	end

	-- Kiểm tra loại sách có tồn tại không
	if not exists(select 1 from LoaiSach where MaLoaiSach = @maLoaiSach)
	begin
		RAISERROR(N'Loại sách không tồn tại!', 16, 1);
		return;
	end

	-- Trim dữ liệu trước khi cập nhật
	set @tenSach = LTRIM(RTRIM(@tenSach));
	set @tacGia = LTRIM(RTRIM(@tacGia));

	-- Cập nhật dữ liệu
	update Sach
	set TenSach = @tenSach, MaLoaiSach = @maLoaiSach, TacGia = @tacGia,
SoLuong = @soLuong, GiaBan = @giaBan
	where MaSach = @maSach;
end;
go
```

Xử lý bên C#

```
private void btnSachSua_Click(object sender, EventArgs e)
{
    // Validate dữ liệu đầu vào
    if (string.IsNullOrWhiteSpace(txtSachTenSach.Text))
    {
        MessageBox.Show("Vui lòng nhập tên sách!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        txtSachTenSach.Focus();
        return;
    }

    if (maSachLoaiSach <= 0)
    {
        MessageBox.Show("Vui lòng chọn loại sách!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
```

```
        cbSachLoaiSach.Focus();
        return;
    }

    if (string.IsNullOrWhiteSpace(txtSachTacGia.Text))
    {
        MessageBox.Show("Vui lòng nhập tác giả!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        txtSachTacGia.Focus();
        return;
    }

    if (numSachSoLuong.Value < 0)
    {
        MessageBox.Show("Số lượng phải lớn hơn hoặc bằng 0!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        numSachSoLuong.Focus();
        return;
    }

    if (numSachGiaBan.Value <= 0)
    {
        MessageBox.Show("Giá bán phải lớn hơn 0!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        numSachGiaBan.Focus();
        return;
    }

    // Kiểm tra trùng sách trước khi gọi stored procedure
    string newTenSach = txtSachTenSach.Text.Trim();
    string newTacGia = txtSachTacGia.Text.Trim();
    int newMaLoaiSach = maSachLoaiSach;

    string checkQuery = "SELECT COUNT(*) FROM Sach WHERE " +
                        "UPPER(LTRIM(RTRIM(TenSach))) = UPPER(N'" +
newTenSach.Replace("'", "''") + "') AND " +
                        "UPPER(LTRIM(RTRIM(TacGia))) = UPPER(N'" +
newTacGia.Replace("'", "''") + "') AND " +
                        "MaLoaiSach = " + newMaLoaiSach + " AND " +
                        "MaSach != " + selectedMaSachSach;

    try
    {
        var checkResult = dataProvider.execScaler(checkQuery);
        int duplicateCount = Convert.ToInt32(checkResult ?? 0);

        if (duplicateCount > 0)
        {
            MessageBox.Show("Sách với tên, tác giả và loại sách này đã tồn
tại!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
            txtSachTenSach.Focus();
            return;
        }

        // Nếu không trùng thì thực hiện cập nhật (sử dụng selectedMaSachSach)
        StringBuilder query = new StringBuilder("EXEC proc_CapNhatSach");
        query.Append(" @maSach = " + selectedMaSachSach);
        query.Append(",@tenSach = N'" + newTenSach.Replace("'", "''") + "'");
        query.Append(",@maLoaiSach = " + newMaLoaiSach);
        query.Append(",@tacGia = N'" + newTacGia.Replace("'", "''") + "'");
        query.Append(",@soLuong = " + numSachSoLuong.Value);
        query.Append(",@giaBan = " + numSachGiaBan.Value);

        int result = dataProvider.execNonQuery(query.ToString());
```

```
        if (result > 0)
        {
            loadDgSach();
            MessageBox.Show("Cập nhật sách thành công!", "Thành Công",
MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        }
        else
        {
            MessageBox.Show("Cập nhật sách không thành công!", "Lỗi",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Cập nhật sách không thành công! " + ex.Message, "Lỗi",
MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Xóa sách với kiểm tra ràng buộc

```
create proc proc_XoaSach
    @MaSach INT
as
begin
    -- Kiểm tra sách có tồn tại không
    if not exists(select 1 from Sach where MaSach = @MaSach)
    begin
        RAISERROR(N'Không tìm thấy sách cần xóa!', 16, 1);
        return;
    end

    -- Kiểm tra có chi tiết hóa đơn nào đang sử dụng sách này không
    if exists(select 1 from ChiTietHoaDon where MaSach = @MaSach)
    begin
        RAISERROR(N'Không thể xóa sách này vì đã có trong hóa đơn!', 16, 1);
        return;
    end

    -- Nếu không có ràng buộc thì mới xóa
    delete from Sach
    where MaSach = @MaSach;
end;
go
```

Xử lý bên C#

```
private void btnSachXoa_Click(object sender, EventArgs e)
{
    DialogResult check = MessageBox.Show("Bạn có chắc muốn xóa sách " +
txtSachTenSach.Text + " ?",
                                         "Cảnh Báo",
                                         MessageBoxButtons.YesNo,
                                         MessageBoxIcon.Question);

    if (check == DialogResult.Yes)
    {
        try
        {
            StringBuilder query = new StringBuilder("EXEC proc_XoaSach");
            query.Append(" @maSach = " + selectedMaSachSach);
```

```
            dataProvider.execNonQuery(query.ToString());

            loadDgSach();
            MessageBox.Show("Xóa sách thành công!", "Thành Công",
MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        }
        catch (Exception ex)
        {
            MessageBox.Show("Xóa sách không thành công! " + ex.Message, "Lỗi",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
}
```

Thêm loại sách (UPSERT) với validation

```
create proc proc_ThemLoaiSach
@tenLoaiSach nvarchar(256)
as
begin
	-- Validation dữ liệu đầu vào
	if @tenLoaiSach is null or LTRIM(RTRIM(@tenLoaiSach)) = ''
	begin
		select 0 as Result, N'Tên loại sách không được để trống!' as
Message;
		return;
	end

	-- Trim dữ liệu
	set @tenLoaiSach = LTRIM(RTRIM(@tenLoaiSach));

	-- Kiểm tra xem loại sách đã tồn tại chưa
	if exists(select 1 from LoaiSach where TenLoaiSach = @tenLoaiSach)
	begin
		-- Nếu đã tồn tại thì không làm gì (hoặc có thể thông báo)
		select 0 as Result, N'Loại sách đã tồn tại' as Message;
	end
	else
	begin
		-- Nếu chưa tồn tại thì thêm mới
		insert into LoaiSach(TenLoaiSach)
		values (@tenLoaiSach);
		select 1 as Result, N'Thêm loại sách thành công' as Message;
	end
end;
go
```

Xử lý bên C#

```
private void btnLoaiSachThem_Click(object sender, EventArgs e)
{
    // Validate dữ liệu đầu vào
    if (string.IsNullOrWhiteSpace(txtLoaiSachTenLoaiSach.Text))
    {
        MessageBox.Show("Vui lòng nhập tên loại sách!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        txtLoaiSachTenLoaiSach.Focus();
        return;
    }

    StringBuilder query = new StringBuilder("EXEC proc_ThemLoaiSach");
```

```
    query.Append(" @tenLoaiSach = N'" + txtLoaiSachTenLoaiSach.Text.Trim() +
"'");

    try
    {
        // Vì stored procedure trả về result set, dùng execQuery thay vì
execNonQuery
        var result = dataProvider.execQuery(query.ToString());

        loadDgLoaiSach();
        loadcbSachLoaiSach();

        if (result.Rows.Count > 0)
        {
            string message = result.Rows[0]["Message"].ToString();
            MessageBox.Show(message, "Thông Báo", MessageBoxButtons.OK,
MessageBoxIcon.Information);
        }
        else
        {
            MessageBox.Show("Thêm/cập nhật loại sách thành công!", "Thành
Công", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Thêm loại sách không thành công! " + ex.Message,
"Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Cập nhật loại sách với validation

```
create proc proc_CapNhatLoaiSach
@maLoaiSach int, @tenLoaiSach nvarchar(256)
as
begin
	-- Validation dữ liệu đầu vào
	if @maLoaiSach is null or @maLoaiSach <= 0
	begin
		RAISERROR(N'Mã loại sách không hợp lệ!', 16, 1);
		return;
	end

	if @tenLoaiSach is null or LTRIM(RTRIM(@tenLoaiSach)) = ''
	begin
		RAISERROR(N'Tên loại sách không được để trống!', 16, 1);
		return;
	end

	if not exists(select 1 from LoaiSach where MaLoaiSach = @maLoaiSach)
	begin
		RAISERROR(N'Không tìm thấy loại sách cần cập nhật!', 16, 1);
		return;
	end

	-- Trim dữ liệu trước khi cập nhật
	set @tenLoaiSach = LTRIM(RTRIM(@tenLoaiSach));

	-- Kiểm tra trùng tên với loại sách khác (case-insensitive)
	if exists(select 1 from LoaiSach where UPPER(TenLoaiSach) =
UPPER(@tenLoaiSach) and MaLoaiSach != @maLoaiSach)
	begin
```

```
            RAISERROR(N'Tên loại sách đã tồn tại!', 16, 1);
            return;
        end

        -- Cập nhật dữ liệu
        update LoaiSach
        set TenLoaiSach = @tenLoaiSach
        where MaLoaiSach = @maLoaiSach;
end;
go
```

Xử lý bên C#

```
private void btnLoaiSachSua_Click(object sender, EventArgs e)
{
    // Validate dữ liệu đầu vào
    if (string.IsNullOrWhiteSpace(txtLoaiSachTenLoaiSach.Text))
    {
        MessageBox.Show("Vui lòng nhập tên loại sách!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        txtLoaiSachTenLoaiSach.Focus();
        return;
    }

    // Kiểm tra trùng tên trước khi gọi stored procedure
    string newName = txtLoaiSachTenLoaiSach.Text.Trim();
    string checkQuery = "SELECT COUNT(*) FROM LoaiSach WHERE
UPPER(LTRIM(RTRIM(TenLoaiSach))) = UPPER(N'" + newName + "') AND MaLoaiSach !=
" + selectedMaLoaiSachLoaiSach;

    try
    {
        var checkResult = dataProvider.execScaler(checkQuery);
        int duplicateCount = Convert.ToInt32(checkResult ?? 0);

        if (duplicateCount > 0)
        {
            MessageBox.Show("Tên loại sách đã tồn tại!", "Lỗi",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            txtLoaiSachTenLoaiSach.Focus();
            return;
        }

        // Nếu không trùng thì thực hiện cập nhật (sử dụng
selectedMaLoaiSachLoaiSach thay vì maLoaiSachLoaiSach)
        StringBuilder query = new StringBuilder("EXEC proc_CapNhatLoaiSach");
        query.Append(" @tenLoaiSach = N'" + newName + "'");
        query.Append(",@maLoaiSach = " + selectedMaLoaiSachLoaiSach);

        int result = dataProvider.execNonQuery(query.ToString());

        if (result > 0)
        {
            loadDgLoaiSach();
            loadDgSach();
            loadcbSachLoaiSach();
            MessageBox.Show("Cập nhật loại sách thành công!", "Thành Công",
MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        }
        else
        {
            MessageBox.Show("Cập nhật loại sách không thành công!", "Lỗi",
MessageBoxButtons.OK, MessageBoxIcon.Error);
```

```
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Cập nhật loại sách không thành công! " + ex.Message,
"Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Xóa loại sách với kiểm tra ràng buộc

```
create proc proc_XoaLoaiSach
    @maLoaiSach int
as
begin
    -- Kiểm tra loại sách có tồn tại không
    if not exists(select 1 from LoaiSach where MaLoaiSach = @maLoaiSach)
    begin
        RAISERROR(N'Không tìm thấy loại sách cần xóa!', 16, 1);
        return;
    end

    -- Kiểm tra có sách nào đang sử dụng loại sách này không
    if exists(select 1 from Sach where MaLoaiSach = @maLoaiSach)
    begin
        RAISERROR(N'Không thể xóa loại sách này vì đang có sách sử dụng!', 16,
1);
        return;
    end

    -- Nếu không có ràng buộc thì mới xóa
    delete from LoaiSach
    where MaLoaiSach = @maLoaiSach;
end;
go
```

Xừ lý bên C#

```
private void btnLoaiSachXoa_Click(object sender, EventArgs e)
{
    DialogResult check = MessageBox.Show("Bạn có chắc muốn xóa loại sách " +
txtLoaiSachTenLoaiSach.Text + " ?",
                                         "Cảnh Báo",
                                         MessageBoxButtons.YesNo,
                                         MessageBoxIcon.Question);

    if (check == DialogResult.Yes)
    {
        try
        {
            StringBuilder query = new StringBuilder("EXEC proc_XoaLoaiSach");
            query.Append(" @maLoaiSach = " + selectedMaLoaiSachLoaiSach);

            dataProvider.execNonQuery(query.ToString());

            loadDgLoaiSach();
            loadcbSachLoaiSach();
            MessageBox.Show("Xóa loại sách thành công!", "Thành Công",
MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        }
        catch (Exception ex)
        {
```

```
            MessageBox.Show("Xóa loại sách không thành công! " + ex.Message,
"Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
}
```

Thêm hóa đơn (UPSERT) với validation

```
create proc proc_ThemHoaDon
@ngayLapHoaDon datetime, @tenKhachHang nvarchar(30), @sdtKhachHang nvarchar(12)
as
begin
	-- Validation dữ liệu đầu vào
	if @ngayLapHoaDon is null
	begin
		select 0 as Result, N'Ngày lập hóa đơn không được để trống!' as
Message;
		return;
	end

	if @tenKhachHang is null or LTRIM(RTRIM(@tenKhachHang)) = ''
	begin
		select 0 as Result, N'Tên khách hàng không được để trống!' as
Message;
		return;
	end

	-- Trim dữ liệu
	set @tenKhachHang = LTRIM(RTRIM(@tenKhachHang));
	if @sdtKhachHang is not null
		set @sdtKhachHang = LTRIM(RTRIM(@sdtKhachHang));

	-- Kiểm tra xem hóa đơn đã tồn tại chưa (dựa trên ngày, tên và SDT khách
hàng)
	if exists(select 1 from HoaDon
			where NgayLapHoaDon = @ngayLapHoaDon
			and TenKhachHang = @tenKhachHang
			and SDT = @sdtKhachHang)
	begin
		-- Nếu đã tồn tại thì cập nhật (trong trường hợp này có thể không
cần cập nhật gì)
		select 0 as Result, N'Hóa đơn đã tồn tại' as Message;
	end
	else
	begin
		-- Nếu chưa tồn tại thì thêm mới
		insert into HoaDon(NgayLapHoaDon, TenKhachHang, SDT)
		values (@ngayLapHoaDon, @tenKhachHang, @sdtKhachHang);
		select 1 as Result, N'Thêm hóa đơn thành công' as Message;
	end
end;
go
```

Xử lý bên C#

```
private void btnHoaDonThem_Click(object sender, EventArgs e)
{
    // Validate dữ liệu đầu vào
    if (string.IsNullOrWhiteSpace(txtHoaDonTenKH.Text))
    {
        MessageBox.Show("Vui lòng nhập tên khách hàng!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
```

```
        txtHoaDonTenKH.Focus();
        return;
    }

    // Kiểm tra trùng hóa đơn trước khi thêm
    DateTime newNgayLap = dateNgayLapHoaDon.Value.Date;
    string newSDT = txtHoaDonSDTKH.Text.Trim();

    // Chỉ kiểm tra trùng nếu có số điện thoại
    if (!string.IsNullOrEmpty(newSDT))
    {
        string checkQuery = "SELECT COUNT(*) FROM HoaDon WHERE " +
                            "CAST(NgayLapHoaDon AS DATE) = '" +
newNgayLap.ToString("yyyy-MM-dd") + "' AND " +
                            "LTRIM(RTRIM(SDT)) = '" + newSDT.Replace("'", "''")
+ "'";

        try
        {
            var checkResult = dataProvider.execScaler(checkQuery);
            int duplicateCount = Convert.ToInt32(checkResult ?? 0);

            if (duplicateCount > 0)
            {
                MessageBox.Show("Đã tồn tại hóa đơn trong ngày " +
newNgayLap.ToString("dd/MM/yyyy") +
                                " cho số điện thoại " + newSDT + "!", "Lỗi",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txtHoaDonSDTKH.Focus();
                return;
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("Lỗi khi kiểm tra trùng hóa đơn! " + ex.Message,
"Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
            return;
        }
    }

    StringBuilder query = new StringBuilder("EXEC proc_ThemHoaDon");
    query.Append(" @ngayLapHoaDon = '" + dateNgayLapHoaDon.Value + "'");
    query.Append(",@tenKhachHang = N'" +
txtHoaDonTenKH.Text.Trim().Replace("'", "''") + "'");
    query.Append(",@sdtKhachHang = '" + newSDT.Replace("'", "''") + "'");

    try
    {
        // Vì stored procedure trả về result set, dùng execQuery thay vì
execNonQuery
        var result = dataProvider.execQuery(query.ToString());

        loadDgHoaDon();

        if (result.Rows.Count > 0)
        {
            string message = result.Rows[0]["Message"].ToString();
            MessageBox.Show(message, "Thông Báo", MessageBoxButtons.OK,
MessageBoxIcon.Information);
        }
        else
        {
            MessageBox.Show("Thêm/cập nhật hóa đơn thành công!", "Thành Công",
MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        }
```

```
    }
    catch (Exception ex)
    {
        MessageBox.Show("Thêm hóa đơn không thành công! " + ex.Message, "Lỗi",
MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Cập nhật hóa đơn với validation

```
create proc proc_CapNhatHoaDon
@maHoaDon int, @ngayLapHoaDon datetime, @tenKhachHang nvarchar(30),
@sdtKhachHang nvarchar(12)
as
begin
	-- Validation dữ liệu đầu vào
	if @maHoaDon is null or @maHoaDon <= 0
	begin
		RAISERROR(N'Mã hóa đơn không hợp lệ!', 16, 1);
		return;
	end

	if @ngayLapHoaDon is null
	begin
		RAISERROR(N'Ngày lập hóa đơn không được để trống!', 16, 1);
		return;
	end

	if @tenKhachHang is null or LTRIM(RTRIM(@tenKhachHang)) = ''
	begin
		RAISERROR(N'Tên khách hàng không được để trống!', 16, 1);
		return;
	end

	-- Kiểm tra hóa đơn có tồn tại không
	if not exists(select 1 from HoaDon where MaHoaDon = @maHoaDon)
	begin
		RAISERROR(N'Không tìm thấy hóa đơn cần cập nhật!', 16, 1);
		return;
	end

	-- Trim dữ liệu
	set @tenKhachHang = LTRIM(RTRIM(@tenKhachHang));
	if @sdtKhachHang is not null
		set @sdtKhachHang = LTRIM(RTRIM(@sdtKhachHang));

	-- Cập nhật dữ liệu
	update HoaDon
	set NgayLapHoaDon = @ngayLapHoaDon, TenKhachHang = @tenKhachHang, SDT =
@sdtKhachHang
	where MaHoaDon = @maHoaDon;
end;
go
```

Xử lý bên C#

```
private void btnHoaDonSua_Click(object sender, EventArgs e)
{
    // Validate dữ liệu đầu vào
    if (string.IsNullOrWhiteSpace(txtHoaDonTenKH.Text))
    {
```

```
        MessageBox.Show("Vui lòng nhập tên khách hàng!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        txtHoaDonTenKH.Focus();
        return;
    }

    // Kiểm tra trùng hóa đơn trước khi gọi stored procedure
    DateTime newNgayLap = dateNgayLapHoaDon.Value.Date;
    string newSDT = txtHoaDonSDTKH.Text.Trim();

    // Chỉ kiểm tra trùng nếu có số điện thoại
    if (!string.IsNullOrEmpty(newSDT))
    {
        string checkQuery = "SELECT COUNT(*) FROM HoaDon WHERE " +
                            "CAST(NgayLapHoaDon AS DATE) = '" +
newNgayLap.ToString("yyyy-MM-dd") + "' AND " +
                            "LTRIM(RTRIM(SDT)) = '" + newSDT.Replace("'", "''")
+ "' AND " +
                            "MaHoaDon != " + selectedMaHoaDonHoaDon;

        try
        {
            var checkResult = dataProvider.execScaler(checkQuery);
            int duplicateCount = Convert.ToInt32(checkResult ?? 0);

            if (duplicateCount > 0)
            {
                MessageBox.Show("Đã tồn tại hóa đơn trong ngày " +
newNgayLap.ToString("dd/MM/yyyy") +
                                " cho số điện thoại " + newSDT + "!", "Lỗi",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txtHoaDonSDTKH.Focus();
                return;
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("Lỗi khi kiểm tra trùng hóa đơn! " + ex.Message,
"Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
            return;
        }
    }

    // Nếu không trùng thì thực hiện cập nhật
    StringBuilder query = new StringBuilder("EXEC proc_CapNhatHoaDon");
    query.Append(" @ngayLapHoaDon = '" + dateNgayLapHoaDon.Value + "'");
    query.Append(",@tenKhachHang = N'" +
txtHoaDonTenKH.Text.Trim().Replace("'", "''") + "'");
    query.Append(",@sdtKhachHang = '" + newSDT.Replace("'", "''") + "'");
    query.Append(",@maHoaDon = " + selectedMaHoaDonHoaDon);

    try
    {
        int result = dataProvider.execNonQuery(query.ToString());

        if (result > 0)
        {
            loadDgHoaDon();
            MessageBox.Show("Cập nhật hóa đơn thành công!", "Thành Công",
MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        }
        else
        {
            MessageBox.Show("Cập nhật hóa đơn không thành công!", "Lỗi",
MessageBoxButtons.OK, MessageBoxIcon.Error);
```

```
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Cập nhật hóa đơn không thành công! " + ex.Message,
"Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Xóa hóa đơn với kiểm tra ràng buộc

```
create proc proc_XoaHoaDon
    @MaHoaDon INT
as
begin
    -- Kiểm tra hóa đơn có tồn tại không
    if not exists(select 1 from HoaDon where MaHoaDon = @MaHoaDon)
    begin
        RAISERROR(N'Không tìm thấy hóa đơn cần xóa!', 16, 1);
        return;
    end

    -- Kiểm tra có chi tiết hóa đơn không
    if exists(select 1 from ChiTietHoaDon where MaHoaDon = @MaHoaDon)
    begin
        RAISERROR(N'Không thể xóa hóa đơn này vì đã có chi tiết hóa đơn!', 16,
1);
        return;
    end

    -- Nếu không có ràng buộc thì mới xóa
    delete from HoaDon
    where MaHoaDon = @MaHoaDon;
end;
go
```

Xử lý bên C#

```
private void btnHoaDonXoa_Click(object sender, EventArgs e)
{
    DialogResult check = MessageBox.Show("Bạn có chắc muốn xóa hóa đơn có mã là
" + selectedMaHoaDonHoaDon + " ?",
                                    "Cảnh Báo",
                                    MessageBoxButtons.YesNo,
                                    MessageBoxIcon.Question);

    if (check == DialogResult.Yes)
    {
        try
        {
            StringBuilder query = new StringBuilder("EXEC proc_XoaHoaDon");
            query.Append(" @maHoaDon = " + selectedMaHoaDonHoaDon);

            dataProvider.execNonQuery(query.ToString());

            loadDgHoaDon();
            MessageBox.Show("Xóa hóa đơn thành công!", "Thành Công",
MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        }
        catch (Exception ex)
        {
```

```
            MessageBox.Show("Xóa hóa đơn không thành công! " + ex.Message,
"Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
}
```

Thêm chi tiết hóa đơn (UPSERT)

```
create proc proc_ThemChiTietHoaDon
@maHoaDon int, @maSach int, @soLuong int
as
begin
    -- Kiểm tra xem chi tiết hóa đơn đã tồn tại chưa
    if exists(select 1 from ChiTietHoaDon where MaHoaDon = @maHoaDon and MaSach
= @maSach)
    begin
        -- Nếu đã tồn tại thì cộng thêm số lượng
        update ChiTietHoaDon
        set SoLuong = SoLuong + @soLuong
        where MaHoaDon = @maHoaDon and MaSach = @maSach;

        select 1 as Result, N'Đã cập nhật số lượng sách trong hóa đơn' as
Message;
    end
    else
    begin
        -- Nếu chưa tồn tại thì thêm mới
        insert into ChiTietHoaDon(MaHoaDon, MaSach, SoLuong)
        values(@maHoaDon, @maSach, @soLuong);

        select 1 as Result, N'Thêm chi tiết hóa đơn thành công' as Message;
    end
end;
go
```

Xử lý bên C#

```
private void btnThem_Click(object sender, EventArgs e)
{
    // Kiểm tra dữ liệu đầu vào
    if (string.IsNullOrEmpty(Sach.Text))
    {
        MessageBox.Show("Vui lòng nhập tên sách!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        Sach.Focus();
        return;
    }

    if (string.IsNullOrEmpty(tacGia.Text))
    {
        MessageBox.Show("Vui lòng nhập tác giả!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        tacGia.Focus();
        return;
    }

    if (numSoLuongSach.Value <= 0)
    {
        MessageBox.Show("Số lượng phải lớn hơn 0!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        numSoLuongSach.Focus();
        return;
```

```
    }

    try
    {
        // Lấy mã sách từ tên sách và tác giả để đảm bảo chính xác
        string getMaSachQuery = "SELECT MaSach FROM Sach WHERE TenSach = N'" +
Sach.Text.Trim().Replace("'", "''") + "' AND TacGia = N'" +
tacGia.Text.Trim().Replace("'", "''") + "'";
        var maSachResult = dataProvider.execScaler(getMaSachQuery);

        if (maSachResult == null)
        {
            MessageBox.Show("Không tìm thấy sách với tên và tác giả này!",
"Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
            Sach.Focus();
            return;
        }

        int maSach = (int)maSachResult;
        int soLuong = (int)numSoLuongSach.Value;

        // Thêm chi tiết hóa đơn
        StringBuilder query = new StringBuilder("EXEC proc_ThemChiTietHoaDon");
        query.Append(" @maHoaDon = " + maHoaDon);
        query.Append(",@maSach = " + maSach);
        query.Append(",@soLuong = " + soLuong);

        var result = dataProvider.execQuery(query.ToString());

        loadChiTietHoaDon(); // Refresh danh sách
        capNhatTongTien(); // Cập nhật tổng tiền

        if (result.Rows.Count > 0)
        {
            string message = result.Rows[0]["Message"].ToString();
            MessageBox.Show(message, "Thông Báo", MessageBoxButtons.OK,
MessageBoxIcon.Information);
        }
        else
        {
            MessageBox.Show("Thêm chi tiết hóa đơn thành công!", "Thành Công",
MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        }

        // Clear form
        Sach.Text = "";
        tacGia.Text = "";
        numSoLuongSach.Value = 1;
    }
    catch (Exception ex)
    {
        MessageBox.Show("Thêm chi tiết hóa đơn không thành công! " +
ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Cập nhật chi tiết hóa đơn

```
create proc proc_CapNhatChiTietHoaDon
@maHoaDon int, @maSach int, @soLuongMoi int
as
begin
    update ChiTietHoaDon
```

```
    set SoLuong = @soLuongMoi
    where MaHoaDon = @maHoaDon and MaSach = @maSach;
end;
go
```

Xử lý bên C#

```
private void btnSua_Click(object sender, EventArgs e)
{
    // Kiểm tra dữ liệu đầu vào
    if (string.IsNullOrEmpty(Sach.Text))
    {
        MessageBox.Show("Vui lòng nhập tên sách!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        Sach.Focus();
        return;
    }

    if (string.IsNullOrEmpty(tacGia.Text))
    {
        MessageBox.Show("Vui lòng nhập tác giả!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        tacGia.Focus();
        return;
    }

    if (numSoLuongSach.Value <= 0)
    {
        MessageBox.Show("Số lượng phải lớn hơn 0!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        numSoLuongSach.Focus();
        return;
    }

    if (selectedMaSach <= 0)
    {
        MessageBox.Show("Vui lòng chọn một dòng để sửa!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    try
    {
        int soLuongMoi = (int)numSoLuongSach.Value;

        // Cập nhật chi tiết hóa đơn sử dụng selectedMaSach
        // Không cần thay đổi sách, chỉ cập nhật số lượng
        StringBuilder query = new StringBuilder("EXEC
proc_CapNhatChiTietHoaDon");
        query.Append(" @maHoaDon = " + maHoaDon);
        query.Append(",@maSach = " + selectedMaSach);
        query.Append(",@soLuongMoi = " + soLuongMoi);

        int result = dataProvider.execNonQuery(query.ToString());

        if (result > 0)
        {
            loadChiTietHoaDon();
            capNhatTongTien(); // Cập nhật tổng tiền
            MessageBox.Show("Cập nhật chi tiết hóa đơn thành công!", "Thành
Công", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        }
        else
```

```
        {
            MessageBox.Show("Cập nhật chi tiết hóa đơn không thành công!",
"Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Cập nhật chi tiết hóa đơn không thành công! " +
ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Xóa chi tiết hóa đơn

```
create proc proc_XoaChiTietHoaDon
@maHoaDon int, @maSach int
as
begin
    delete from ChiTietHoaDon
    where MaHoaDon = @maHoaDon and MaSach = @maSach;
end;
go
```

Xử lý bên C#

```
private void btnXoa_Click(object sender, EventArgs e)
{
    // Kiểm tra có chọn sách để xóa không
    if (string.IsNullOrEmpty(Sach.Text))
    {
        MessageBox.Show("Vui lòng chọn sách để xóa!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    if (selectedMaSach <= 0)
    {
        MessageBox.Show("Vui lòng chọn một dòng để xóa!", "Thông Báo",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    DialogResult check = MessageBox.Show("Bạn có chắc muốn xóa sách '" +
Sach.Text + "' khỏi hóa đơn?",
                                "Cảnh Báo",
                                MessageBoxButtons.YesNo,
                                MessageBoxIcon.Question);

    if (check == DialogResult.Yes)
    {
        try
        {
            // Xóa chi tiết hóa đơn sử dụng selectedMaSach
            StringBuilder query = new StringBuilder("EXEC
proc_XoaChiTietHoaDon");
            query.Append(" @maHoaDon = " + maHoaDon);
            query.Append(",@maSach = " + selectedMaSach);

            dataProvider.execNonQuery(query.ToString());
```

```
            loadChiTietHoaDon();
            capNhatTongTien(); // Cập nhật tổng tiền
            MessageBox.Show("Xóa chi tiết hóa đơn thành công!", "Thành Công",
MessageBoxButtons.OK, MessageBoxIcon.Asterisk);

            // Clear form
            Sach.Text = "";
            tacGia.Text = "";
            numSoLuongSach.Value = 1;
            selectedMaSach = 0; // Reset selectedMaSach
        }
        catch (Exception ex)
        {
            MessageBox.Show("Xóa chi tiết hóa đơn không thành công! " +
ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
}
```

3.3. Hàm

Hàm tính thành tiền cho một chi tiết hóa đơn

```
create function fn_TinhThanhTien(@soLuong int, @giaBan decimal(10,2))
returns decimal(12,2)
as
begin
    declare @thanhTien decimal(12,2);
    set @thanhTien = @soLuong * @giaBan;
    return @thanhTien;
end;
go
```

Hàm tính tổng tiền cho một hóa đơn

```
create function fn_TinhTongTienHoaDon(@maHoaDon int)
returns decimal(15,2)
as
begin
    declare @tongTien decimal(15,2);

    select @tongTien = sum(ct.SoLuong * s.GiaBan)
    from ChiTietHoaDon ct
    inner join Sach s on ct.MaSach = s.MaSach
    where ct.MaHoaDon = @maHoaDon;

    -- Nếu không có chi tiết thì trả về 0
    if @tongTien is null
        set @tongTien = 0;

    return @tongTien;
end;
go
```

Xử lý bên C#

```
private void capNhatTongTien()
{
    try
    {
```

```
        string query = "SELECT dbo.fn_TinhTongTienHoaDon(" + maHoaDon + ") AS
TongTien";
        var result = dataProvider.execScaler(query);

        decimal giaTriTongTien = 0;
        if (result != null && result != DBNull.Value)
        {
            giaTriTongTien = Convert.ToDecimal(result);
        }

        tongTien.Text = "Tổng tiền: " + giaTriTongTien.ToString("N0") + " VNĐ";
    }
    catch (Exception ex)
    {
        tongTien.Text = "Tổng tiền: 0 VNĐ";
        // Có thể log error nếu cần
    }
}
```

Hàm tìm kiếm sách theo tên

```
CREATE FUNCTION fn_TimKiemSachTheoTen(@tenSach NVARCHAR(200))
RETURNS TABLE
AS
RETURN
(
    SELECT
        MaSach as [Mã Sách],
        TenSach as [Tên Sách],
        TenLoaiSach as [Tên Loại Sách],
        TacGia as [Tác Giả],
        SoLuong as [Số Lượng],
        GiaBan as [Giá Bán]
    FROM Sach s
    INNER JOIN LoaiSach ls ON s.MaLoaiSach = ls.MaLoaiSach
    WHERE (@tenSach IS NULL OR @tenSach = '' OR TenSach LIKE N'%' + @tenSach +
'%')
);
GO
```

Xử lý bên C#

```
private void timKiemSachTheoTen()
{
    string tenSach = txtSachTenSach.Text.Trim();

    try
    {
        string query = "SELECT * FROM fn_TimKiemSachTheoTen(N'" +
tenSach.Replace("'", "''") + "')";
        dgSach.DataSource = dataProvider.execQuery(query);
        // KHÔNG cập nhật selectedMaSachSach khi tìm kiếm
        // Chỉ cập nhật khi user thực sự click vào một dòng
    }
    catch (Exception)
    {
        loadDgSach();
    }
}
```

Hàm tìm kiếm sách theo tác giả

```
CREATE FUNCTION fn_TimKiemSachTheoTacGia(@tacGia NVARCHAR(100))
RETURNS TABLE
AS
RETURN
(
    SELECT
        MaSach as [Mã Sách],
        TenSach as [Tên Sách],
        TenLoaiSach as [Tên Loại Sách],
        TacGia as [Tác Giả],
        SoLuong as [Số Lượng],
        GiaBan as [Giá Bán]
    FROM Sach s
    INNER JOIN LoaiSach ls ON s.MaLoaiSach = ls.MaLoaiSach
    WHERE (@tacGia IS NULL OR @tacGia = '' OR TacGia LIKE N'%' + @tacGia + '%')
);
GO
```

Xử lý bên C#

```
private void timKiemSachTheoTacGia()
{
    string tacGia = txtSachTacGia.Text.Trim();

    try
    {
        string query = "SELECT * FROM fn_TimKiemSachTheoTacGia(N'" +
tacGia.Replace("'", "''") + "')";
        dgSach.DataSource = dataProvider.execQuery(query);
        // KHÔNG cập nhật selectedMaSachSach khi tìm kiếm
    }
    catch (Exception)
    {
        loadDgSach();
    }
}
```

Hàm tìm kiếm sách theo loại sách

```
CREATE FUNCTION fn_TimKiemSachTheoLoai(@tenLoaiSach NVARCHAR(100))
RETURNS TABLE
AS
RETURN
(
    SELECT
        MaSach as [Mã Sách],
        TenSach as [Tên Sách],
        TenLoaiSach as [Tên Loại Sách],
        TacGia as [Tác Giả],
        SoLuong as [Số Lượng],
        GiaBan as [Giá Bán]
    FROM Sach s
    INNER JOIN LoaiSach ls ON s.MaLoaiSach = ls.MaLoaiSach
    WHERE (@tenLoaiSach IS NULL OR @tenLoaiSach = '' OR TenLoaiSach =
@tenLoaiSach)
);
GO
```

Xử lý bên C#

```
private void timKiemSachTheoLoai()
{
    string tenLoaiSach = "";

    if (cbSachLoaiSach.SelectedIndex >= 0 &&
        cbSachLoaiSach.SelectedValue != null &&
        !string.IsNullOrEmpty(cbSachLoaiSach.Text))
    {
        tenLoaiSach = cbSachLoaiSach.Text.Replace("'", "''");
    }

    try
    {
        string query = "SELECT * FROM fn_TimKiemSachTheoLoai(N'" + tenLoaiSach
+ "')";
        dgSach.DataSource = dataProvider.execQuery(query);
        // KHÔNG cập nhật selectedMaSachSach khi tìm kiếm
    }
    catch (Exception)
    {
        loadDgSach();
    }
}
```

Hàm tìm kiếm loại sách theo tên

```
CREATE FUNCTION fn_TimKiemLoaiSachTheoTen(@tenLoaiSach NVARCHAR(100))
RETURNS TABLE
AS
RETURN
(
    SELECT
        MaLoaiSach as [Mã Loại Sách],
        TenLoaiSach as [Tên Loại Sách]
    FROM LoaiSach
    WHERE (@tenLoaiSach IS NULL OR @tenLoaiSach = '' OR TenLoaiSach LIKE N'%' +
@tenLoaiSach + '%')
);
GO
```

Xử lý bên C#

```
private void timKiemLoaiSachTheoTen()
{
    string tenLoaiSach = txtLoaiSachTenLoaiSach.Text.Trim();

    try
    {
        string query = "SELECT * FROM fn_TimKiemLoaiSachTheoTen(N'" +
tenLoaiSach.Replace("'", "''") + "')";
        DataTable dt = dataProvider.execQuery(query);
        dgLoaiSach.DataSource = dt;
    }
    catch (Exception)
    {
        loadDgLoaiSach();
    }
}
```

Hàm tìm kiếm hóa đơn theo tên khách hàng

```
CREATE FUNCTION fn_TimKiemHoaDonTheoTenKH(@tenKhachHang NVARCHAR(100))
RETURNS TABLE
AS
RETURN
(
    SELECT
        MaHoaDon as [Mã Hóa Đơn],
        NgayLapHoaDon as [Ngày Lập Hoá Đơn],
        TenKhachHang as [Tên Khách Hàng],
        SDT as [Số Điện Thoại]
    FROM HoaDon
    WHERE (@tenKhachHang IS NULL OR @tenKhachHang = '' OR TenKhachHang LIKE
N'%' + @tenKhachHang + '%')
);
GO
```

Xử lý bên C#

```
private void timKiemHoaDonTheoTenKH()
{
    string tenKH = txtHoaDonTenKH.Text.Trim();

    try
    {
        string query = "SELECT * FROM fn_TimKiemHoaDonTheoTenKH(N'" +
tenKH.Replace("'", "''") + "')";
        DataTable dt = dataProvider.execQuery(query);
        dgHoaDon.DataSource = dt;
    }
    catch (Exception)
    {
        loadDgHoaDon();
    }
}
```

Hàm tìm kiếm hóa đơn theo số điện thoại

```
CREATE FUNCTION fn_TimKiemHoaDonTheoSDT(@sdt NVARCHAR(20))
RETURNS TABLE
AS
RETURN
(
    SELECT
        MaHoaDon as [Mã Hóa Đơn],
        NgayLapHoaDon as [Ngày Lập Hoá Đơn],
        TenKhachHang as [Tên Khách Hàng],
        SDT as [Số Điện Thoại]
    FROM HoaDon
    WHERE (@sdt IS NULL OR @sdt = '' OR SDT LIKE '%' + @sdt + '%')
);
GO
```

Xử lý bên C#

```
private void timKiemHoaDonTheoSDT()
{
    string sdt = txtHoaDonSDTKH.Text.Trim();
```

```
    try
    {
        string query = "SELECT * FROM fn_TimKiemHoaDonTheoSDT('" +
sdt.Replace("'", "''") + "')";
        DataTable dt = dataProvider.execQuery(query);
        dgHoaDon.DataSource = dt;

        // KHÔNG cập nhật selectedMaHoaDonHoaDon khi tìm kiếm
    }
    catch (Exception)
    {
        loadDgHoaDon();
    }
}
```

Hàm tìm kiếm hóa đơn theo ngày lập

```
CREATE FUNCTION fn_TimKiemHoaDonTheoNgay(@ngayLap DATE)
RETURNS TABLE
AS
RETURN
(
    SELECT
        MaHoaDon as [Mã Hóa Đơn],
        NgayLapHoaDon as [Ngày Lập Hoá Đơn],
        TenKhachHang as [Tên Khách Hàng],
        SDT as [Số Điện Thoại]
    FROM HoaDon
    WHERE (@ngayLap IS NULL OR CAST(NgayLapHoaDon AS DATE) = @ngayLap)
);
GO
```

Xử lý bên C#

```
private void timKiemHoaDonTheoNgay()
{
    DateTime ngayLap = dateNgayLapHoaDon.Value.Date;

    try
    {
        string query = "SELECT * FROM fn_TimKiemHoaDonTheoNgay('" +
ngayLap.ToString("yyyy-MM-dd") + "')";
        DataTable dt = dataProvider.execQuery(query);
        dgHoaDon.DataSource = dt;
    }
    catch (Exception)
    {
        loadDgHoaDon();
    }
}
```

CHƯƠNG 4 : PHÂN QUYỀN

4.1. Tạo role

-Hệ thống gồm 2 role là admin và nhân viên

- Tạo role cho admin và nhân viên

```
IF NOT EXISTS (SELECT 1 FROM sys.database_principals WHERE type = 'R' AND name
= 'app_admin')
    CREATE ROLE app_admin;
IF NOT EXISTS (SELECT 1 FROM sys.database_principals WHERE type = 'R' AND name
= 'app_employee')
    CREATE ROLE app_employee;
GO
```

4.2. Tạo login, user và gán quyền

```
CREATE LOGIN admin01 WITH PASSWORD = '1';
CREATE USER  admin01 FOR LOGIN admin01;
ALTER ROLE app_admin ADD MEMBER admin01;

CREATE LOGIN nv01 WITH PASSWORD = '2';
CREATE USER  nv01 FOR LOGIN nv01;
ALTER ROLE app_employee ADD MEMBER nv01;
```

4.3. Thêm quyền cho các role

Quyền cho ADMIN: toàn quyền trong schema dbo

```
GRANT CONTROL ON SCHEMA::dbo TO app_admin;    -- bao gồm toàn bộ quyền trên mọi
object dbo
GRANT EXECUTE TO app_admin;                   -- chạy mọi proc/UDF trong DB
GO
```

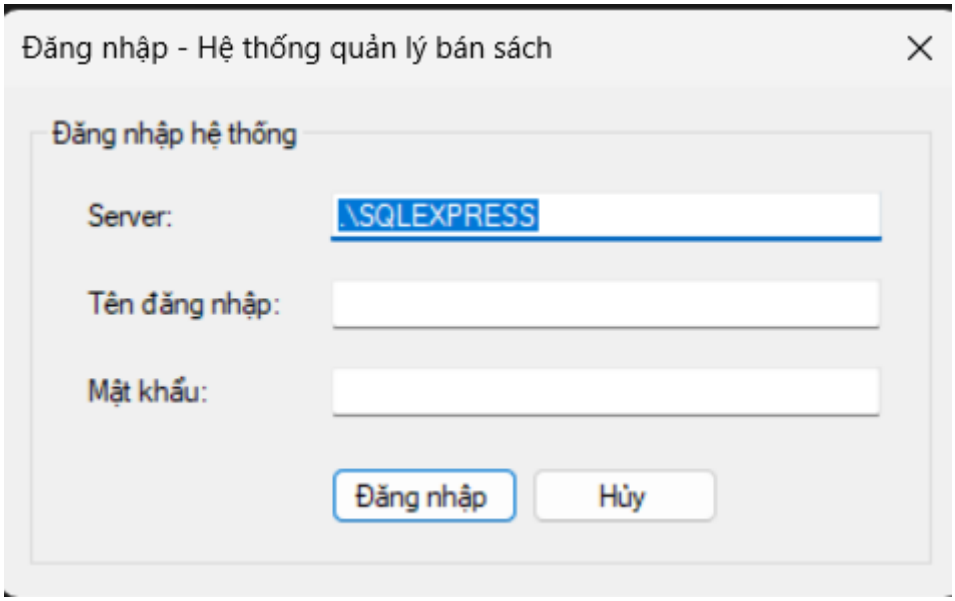
Quyền cho NHÂN VIÊN

```
-- Quyền đọc bảng chính
GRANT SELECT ON OBJECT::dbo.Sach            TO app_employee;
GRANT SELECT ON OBJECT::dbo.LoaiSach        TO app_employee;
GRANT SELECT ON OBJECT::dbo.HoaDon          TO app_employee;
GRANT SELECT ON OBJECT::dbo.ChiTietHoaDon   TO app_employee;

-- Quyền thao tác nghiệp vụ bán hàng
GRANT INSERT, UPDATE, DELETE ON OBJECT::dbo.HoaDon        TO app_employee;
GRANT INSERT, UPDATE, DELETE ON OBJECT::dbo.ChiTietHoaDon TO app_employee;



-- Cho phép dùng các TVF tìm kiếm
IF OBJECT_ID('dbo.fn_TimKiemSachTheoTen',       'IF') IS NOT NULL GRANT SELECT
ON OBJECT::dbo.fn_TimKiemSachTheoTen       TO app_employee;
IF OBJECT_ID('dbo.fn_TimKiemSachTheoTacGia',    'IF') IS NOT NULL GRANT SELECT
ON OBJECT::dbo.fn_TimKiemSachTheoTacGia    TO app_employee;
IF OBJECT_ID('dbo.fn_TimKiemSachTheoLoai',      'IF') IS NOT NULL GRANT SELECT
ON OBJECT::dbo.fn_TimKiemSachTheoLoai      TO app_employee;
IF OBJECT_ID('dbo.fn_TimKiemLoaiSachTheoTen',   'IF') IS NOT NULL GRANT SELECT
ON OBJECT::dbo.fn_TimKiemLoaiSachTheoTen   TO app_employee;
IF OBJECT_ID('dbo.fn_TimKiemHoaDonTheoTenKH',   'IF') IS NOT NULL GRANT SELECT
ON OBJECT::dbo.fn_TimKiemHoaDonTheoTenKH   TO app_employee;
```

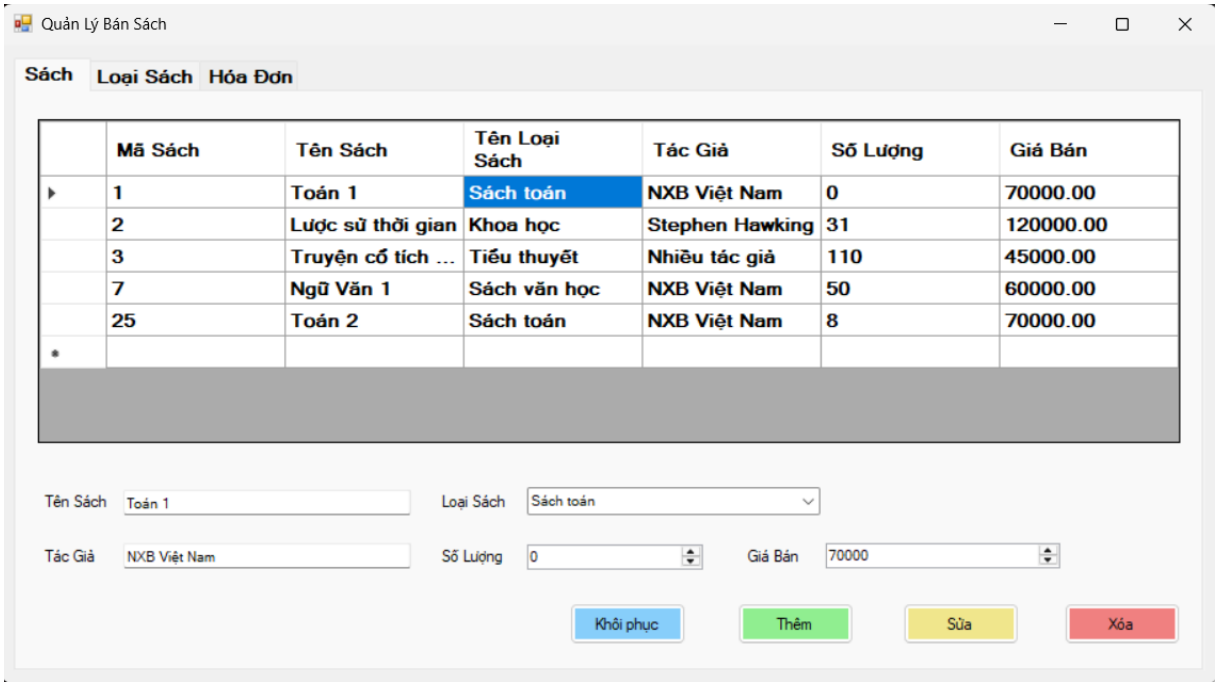
```sql
IF OBJECT_ID('dbo.fn_TimKiemHoaDonTheoSDT',     'IF') IS NOT NULL GRANT SELECT
ON OBJECT::dbo.fn_TimKiemHoaDonTheoSDT     TO app_employee;
IF OBJECT_ID('dbo.fn_TimKiemHoaDonTheoNgay',    'IF') IS NOT NULL GRANT SELECT
ON OBJECT::dbo.fn_TimKiemHoaDonTheoNgay    TO app_employee;
IF OBJECT_ID('dbo.fn_TimKiemSachTongHop',       'IF') IS NOT NULL GRANT SELECT
ON OBJECT::dbo.fn_TimKiemSachTongHop       TO app_employee;
IF OBJECT_ID('dbo.fn_TimKiemHoaDonTongHop',     'IF') IS NOT NULL GRANT SELECT
ON OBJECT::dbo.fn_TimKiemHoaDonTongHop     TO app_employee;

-- Cho phép chạy scalar function tính toán
IF OBJECT_ID('dbo.fn_TinhThanhTien',        'FN') IS NOT NULL GRANT EXECUTE ON
OBJECT::dbo.fn_TinhThanhTien        TO app_employee;
IF OBJECT_ID('dbo.fn_TinhTongTienHoaDon',   'FN') IS NOT NULL GRANT EXECUTE ON
OBJECT::dbo.fn_TinhTongTienHoaDon   TO app_employee;

-- (Tùy chọn) Cấp quyền đọc các VIEW nếu tồn tại
IF OBJECT_ID('dbo.vw_DanhSachSach',   'V') IS NOT NULL GRANT SELECT ON
OBJECT::dbo.vw_DanhSachSach   TO app_employee;
IF OBJECT_ID('dbo.vw_DanhSachHoaDon', 'V') IS NOT NULL GRANT SELECT ON
OBJECT::dbo.vw_DanhSachHoaDon TO app_employee;
IF OBJECT_ID('dbo.vw_ChiTietHoaDon',  'V') IS NOT NULL GRANT SELECT ON
OBJECT::dbo.vw_ChiTietHoaDon  TO app_employee;
IF OBJECT_ID('dbo.vw_DanhSachLoaiSach','V') IS NOT NULL GRANT SELECT ON
OBJECT::dbo.vw_DanhSachLoaiSach TO app_employee;
IF OBJECT_ID('dbo.vw_LoaiSach',       'V') IS NOT NULL GRANT SELECT ON
OBJECT::dbo.vw_LoaiSach       TO app_employee;
GO
```

CHƯƠNG 5 : CÀI ĐẶT GIAO DIỆN

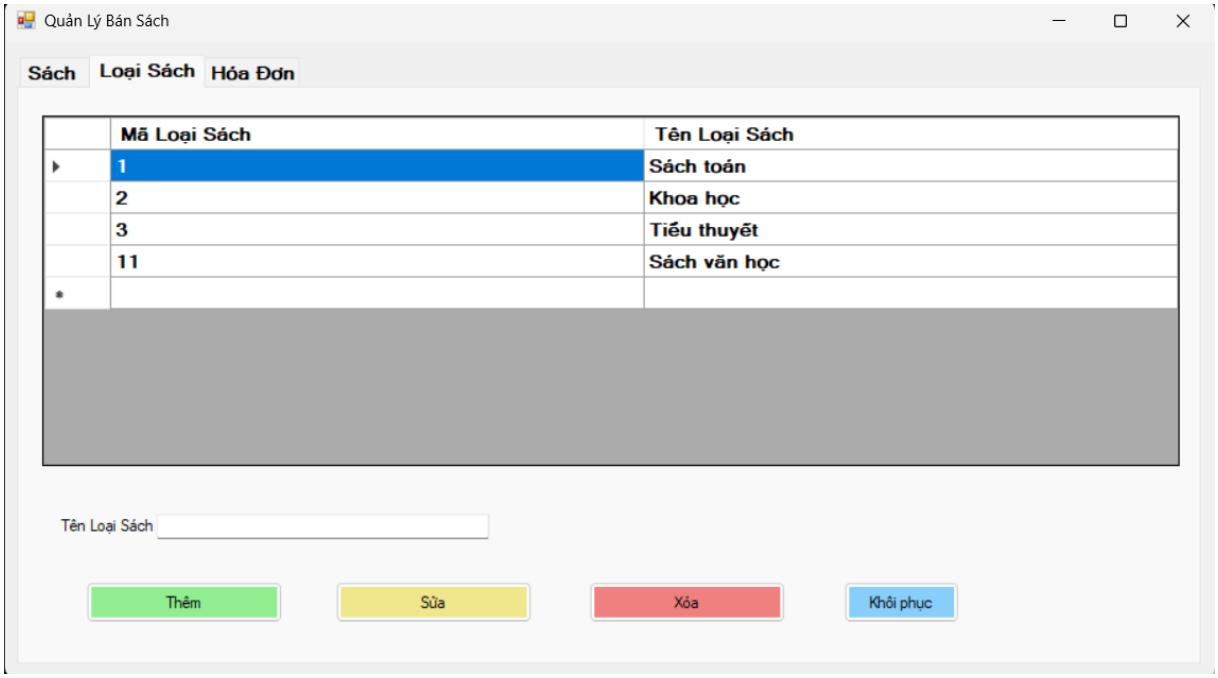
5.1. Form đăng nhập



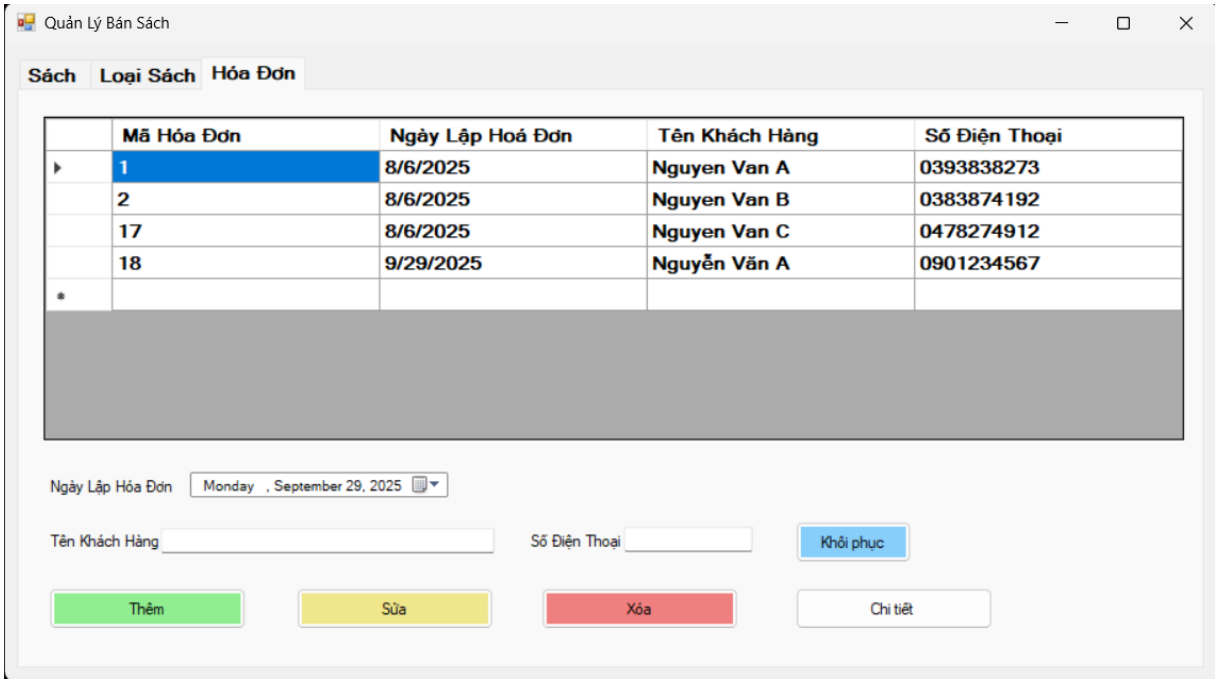
5.2. Tab sách



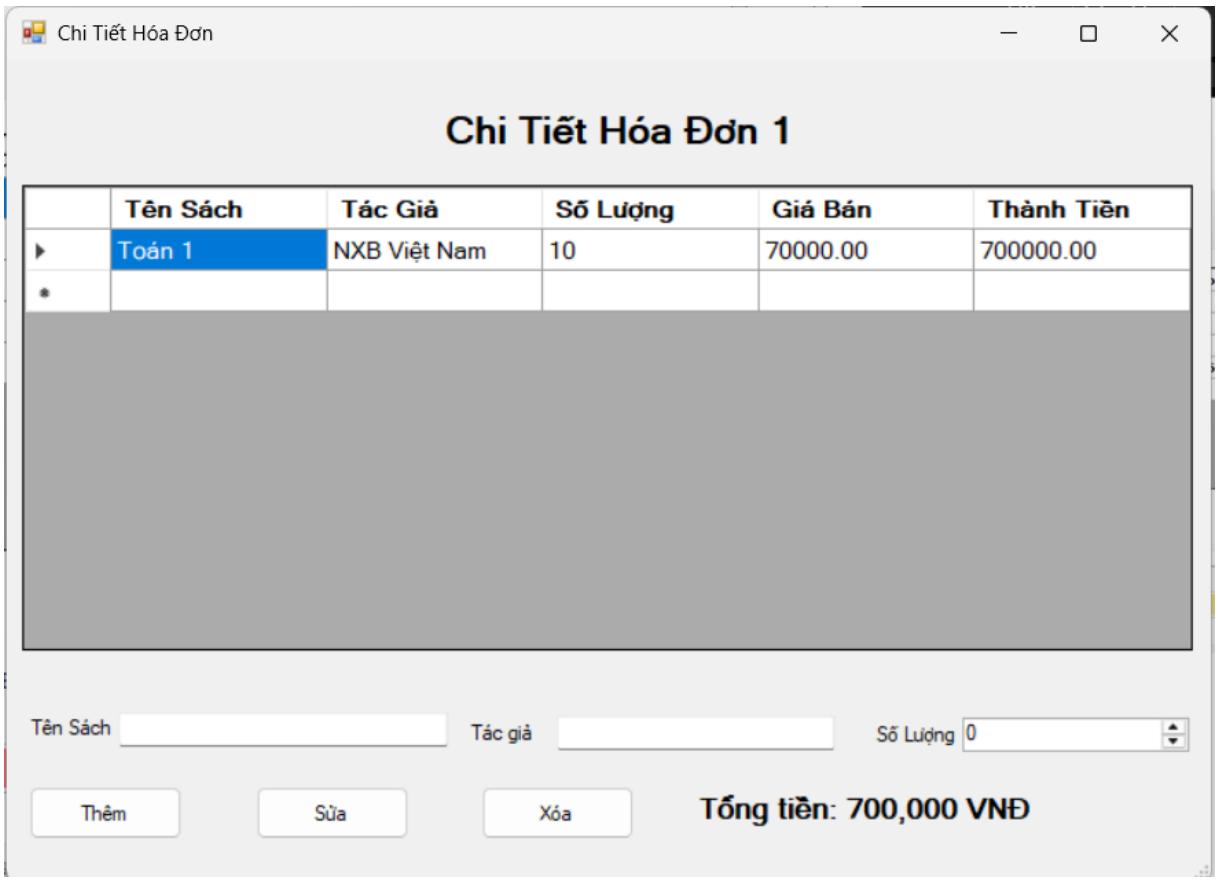
5.3. Tab loại sách



5.4. Tab hóa đơn



5.5. Form chi tiết hóa đơn


Chi Tiết Hóa Đơn
Chi Tiết Hóa Đơn 1
Tên Sách
Tác Giả
Số Lượng
Giá Bán
Thành Tiền
Toán 1
NXB Việt Nam
10
70000.00
700000.00
Tên Sách
Tác giả
Số Lượng
0
Thêm
Sửa
Xóa
Tổng tiền: 700,000 VNĐ