

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN



Báo cáo Lab 3

Mã hóa dữ liệu sử dụng các thuật toán mã hóa công khai

Môn học: Bảo mật cơ sở dữ liệu

CSC15002_22MMT

Sinh viên:

Nguyễn Hồ Đăng Duy

Phạm Quang Duy

Giảng viên hướng dẫn:

Nguyễn Đình Thúc

Nguyễn Thị Hương

Lê Trọng Anh Tú

Mục lục

1	Thông tin	2
1.1	Thông tin sinh viên	2
1.2	Phân công nhiệm vụ	2
2	Câu a	3
3	Câu b	3
4	Câu c	5
4.1	SP_INS_PUBLIC_NHANVIEN	5
4.2	SP_SEL_PUBLIC_NHANVIEN	6
5	Câu d	7
5.1	Màn hình quản lý đăng nhập	7
5.2	Màn hình quản lý lớp học	9
5.3	Màn hình Danh sách sinh viên	11
5.4	Màn hình thêm sinh viên mới	14
5.5	Màn hình xóa sinh viên	16
5.6	Màn hình điều chỉnh thông tin sinh viên	18
5.7	Màn hình chi tiết điểm	20
5.8	Màn hình thêm / chỉnh sửa điểm	22
6	Câu e	24
6.1	Giới Thiệu	24
6.2	Mục Đích Sử Dụng	24
6.3	Quá Trình Thực Hiện	24
6.3.1	Cài Đặt SQL Server Profiler	24
6.3.2	Cấu Hình SQL Profiler	25
6.3.3	Phân Tích Dữ Liệu	25
6.3.4	Theo dõi truy vấn	25
6.4	Thực hiện với form thêm điểm	26

1 Thông tin

1.1 Thông tin sinh viên

Nhóm gồm có 2 thành viên:

- 22127085 - Nguyễn Hồ Đăng Duy - 22127085@student.hcmus.edu.vn
- 22127088 - Phạm Quang Duy - 22127088@student.hcmus.edu.vn

1.2 Phân công nhiệm vụ

Sinh viên	Các câu đã làm	Tiến độ
Nguyễn Hồ Đăng Duy	Câu a, b, c. Các giao diện và procedure Đăng nhập, Quản lý lớp học, Danh sách sinh viên của lớp	100%
Phạm Quang Duy	Câu e. Các giao diện và procedure Thêm sinh viên mới, Xóa sinh viên, Điều chỉnh thông tin sinh viên, Thêm / chỉnh sửa điểm	100%

Bảng 1: Bảng phân công nhiệm vụ

2 Câu a

Viết script tạo Database có tên **QLSVNhom**

```
1 USE master;
2 GO
3
4 IF DB_ID('QLSVNhom') IS NOT NULL
5 BEGIN
6     DROP DATABASE QLSVNhom
7 END
8 GO
9
10 CREATE DATABASE QLSVNhom;
11 GO
```

3 Câu b

Viết script tạo mới các Table **SINHVIEN**, **NHANVIEN**, **LOP**, **HOCPHAN**, **BANGDIEM** (đã thay đổi SINHVIEN.MASV sang VARCHAR(20) để đồng bộ dữ liệu)

```
1 USE QLSVNhom;
2 GO
3
4 CREATE TABLE SINHVIEN (
5     MASV VARCHAR(20) PRIMARY KEY,
6     HOTEN NVARCHAR(100) NOT NULL,
7     NGAYSINH DATETIME,
8     DIACHI NVARCHAR(200),
9     MALOP VARCHAR(20),
10    TENDN NVARCHAR(100) NOT NULL UNIQUE,
11    MATKHAU VARBINARY(MAX) NOT NULL
12 );
13
14 CREATE TABLE NHANVIEN (
15    MANV VARCHAR(20) PRIMARY KEY,
16    HOTEN NVARCHAR(100) NOT NULL,
17    EMAIL VARCHAR(20),
18    LUONG VARBINARY(MAX),
19    TENDN NVARCHAR(100) NOT NULL UNIQUE,
20    MATKHAU VARBINARY(MAX) NOT NULL,
21    PUBKEY VARCHAR(20)
22 );
23
24 CREATE TABLE LOP (
25    MALOP VARCHAR(20) PRIMARY KEY,
26    TENLOP NVARCHAR(100) NOT NULL,
27    MANV VARCHAR(20)
28 );
29
30 CREATE TABLE HOCPHAN (
31    MAHP VARCHAR(20) PRIMARY KEY,
32    TENHP NVARCHAR(100) NOT NULL,
33    SOTC INT
```

```

34 );
35
36 CREATE TABLE BANGDIEM (
37     MASV VARCHAR(20),
38     MAHP VARCHAR(20),
39     DIEMTHI VARBINARY(MAX),
40     PRIMARY KEY (MASV, MAHP)
41 );
42
43 ALTER TABLE SINHVIEN
44 ADD CONSTRAINT FK_SINHVIEN_LOP
45 FOREIGN KEY (MALOP)
46 REFERENCES LOP(MALOP);
47 GO
48
49 ALTER TABLE LOP
50 ADD CONSTRAINT FK_LOP_NHANVIEN
51 FOREIGN KEY (MANV)
52 REFERENCES NHANVIEN(MANV);
53 GO
54
55 ALTER TABLE BANGDIEM
56 ADD CONSTRAINT FK_BANGDIEM_SINHVIEN
57 FOREIGN KEY (MASV)
58 REFERENCES SINHVIEN(MASV);
59 GO
60
61 ALTER TABLE BANGDIEM
62 ADD CONSTRAINT FK_BANGDIEM_HOCPHAN
63 FOREIGN KEY (MAHP)
64 REFERENCES HOCPHAN(MAHP);
65 GO

```

4 Câu c

4.1 SP_INS_PUBLIC_NHANVIEN

```

1 IF EXISTS (SELECT 1 FROM sys.procedures WHERE name = 'SP_INS_PUBLIC_NHANVIEN')
2 BEGIN
3     DROP PROCEDURE SP_INS_PUBLIC_NHANVIEN;
4 END
5 GO
6
7 CREATE PROCEDURE SP_INS_PUBLIC_NHANVIEN
8     @MANV VARCHAR(20),
9     @HOTEN NVARCHAR(100),
10    @EMAIL VARCHAR(100),
11    @LUONGCB INT,
12    @TENDN NVARCHAR(100),
13    @MK NVARCHAR(100)
14 AS
15 BEGIN
16     SET NOCOUNT ON;
17
18     IF EXISTS (SELECT 1 FROM NHANVIEN WHERE MANV = @MANV)
19     BEGIN
20         RAISERROR('@MANV is exist.', 16, 1);
21         RETURN;
22     END
23
24     DECLARE @HASHED_PASSWORD VARBINARY(100);
25     DECLARE @ENCRYPTED_SALARY VARBINARY(100);
26     DECLARE @SQL NVARCHAR(MAX);
27
28     SET @HASHED_PASSWORD = HASHBYTES('SHA1', @MK);
29
30     IF NOT EXISTS (SELECT 1 FROM sys.asymmetric_keys WHERE name = @MANV)
31     BEGIN
32         SET @SQL = 'CREATE ASYMMETRIC KEY ' + @MANV +
33             ' WITH ALGORITHM = RSA_2048 ENCRYPTION BY PASSWORD = ''' +
34             @MK + '''';
35         EXEC sp_executesql @SQL;
36     END
37
38     DECLARE @ENCRYPTED_LUONG VARBINARY(500);
39     SET @ENCRYPTED_LUONG = ENCRYPTBYASYMKEY(AsymKey_ID(@MANV), CONVERT(VARCHAR,
40         @LUONGCB));
41
42     INSERT INTO NHANVIEN (MANV, HOTEN, EMAIL, LUONG, TENDN, MATKHAU, PUBKEY)
43     VALUES (@MANV, @HOTEN, @EMAIL, @ENCRYPTED_LUONG, @TENDN, @HASHED_PASSWORD,
44         @MANV);
45 END
46 GO

```

4.2 SP_SEL_PUBLIC_NHANVIEN

```

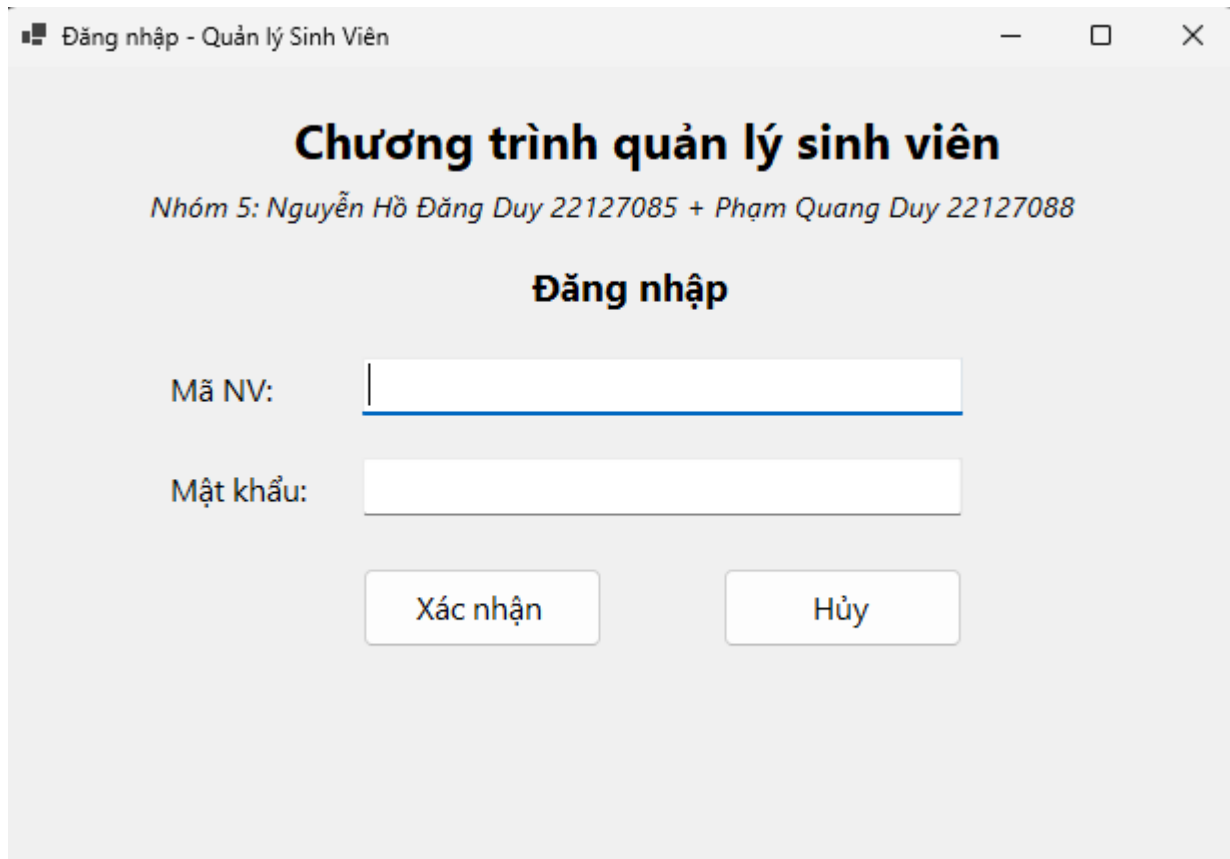
1 IF EXISTS (SELECT 1 FROM sys.procedures WHERE name = 'SP_SEL_PUBLIC_NHANVIEN')
2 BEGIN
3     DROP PROCEDURE SP_SEL_PUBLIC_NHANVIEN;
4 END
5 GO
6
7 CREATE PROCEDURE SP_SEL_PUBLIC_NHANVIEN
8     @TENDN NVARCHAR(100),
9     @MK NVARCHAR(100)
10 AS
11 BEGIN
12     SET NOCOUNT ON;
13     DECLARE @HASHED_PASSWORD VARBINARY(100);
14     DECLARE @MANV VARCHAR(20);
15
16     SET @HASHED_PASSWORD = HASHBYTES('SHA1', @MK);
17     SELECT @MANV = MANV FROM NHANVIEN WHERE TENDN = @TENDN AND MATKHAU =
18     @HASHED_PASSWORD;
19     IF @MANV IS NULL
20     BEGIN
21         RAISERROR('Incorrect TENDN or MK', 16, 1);
22         RETURN;
23     END
24
25     IF NOT EXISTS (SELECT 1 FROM sys.asymmetric_keys WHERE name = @MANV)
26     BEGIN
27         RAISERROR('RSA key doesnt exist', 16, 1);
28         RETURN;
29     END
30
31     SELECT
32         MANV,
33         HOTEN,
34         EMAIL,
35         CONVERT(INT, CONVERT(VARCHAR, DECRYPTBYASYMKEY(ASYMKEY_ID(@MANV), LUONG,
36         @MK))) AS LUONGCB
37     FROM NHANVIEN
38     WHERE TENDN = @TENDN AND MATKHAU = @HASHED_PASSWORD;
39 END
40 GO

```

5 Câu d

Chương trình được viết bằng C# trên Visual Studio Developer 2022

5.1 Màn hình quản lý đăng nhập



Hình 1: Giao diện đăng nhập

- Người dùng được yêu cầu nhập vào **Mã nhân viên** và **Mật khẩu** đã được lưu trong cơ sở dữ liệu.
- Khi nhấn vào nút **Xác nhận** thì chương trình sẽ chạy procedure **SP_LOGIN_NHANVIEN** để kiểm tra. Nếu MANV và MK đúng thì chuyển sang giao diện Quản lý lớp học. Nếu không thì sẽ hiện MessageBox "Đăng nhập không thành công! Kiểm tra lại tài khoản hoặc mật khẩu."
- Nút **Hủy** để xóa hết các thông tin người dùng vừa nhập.

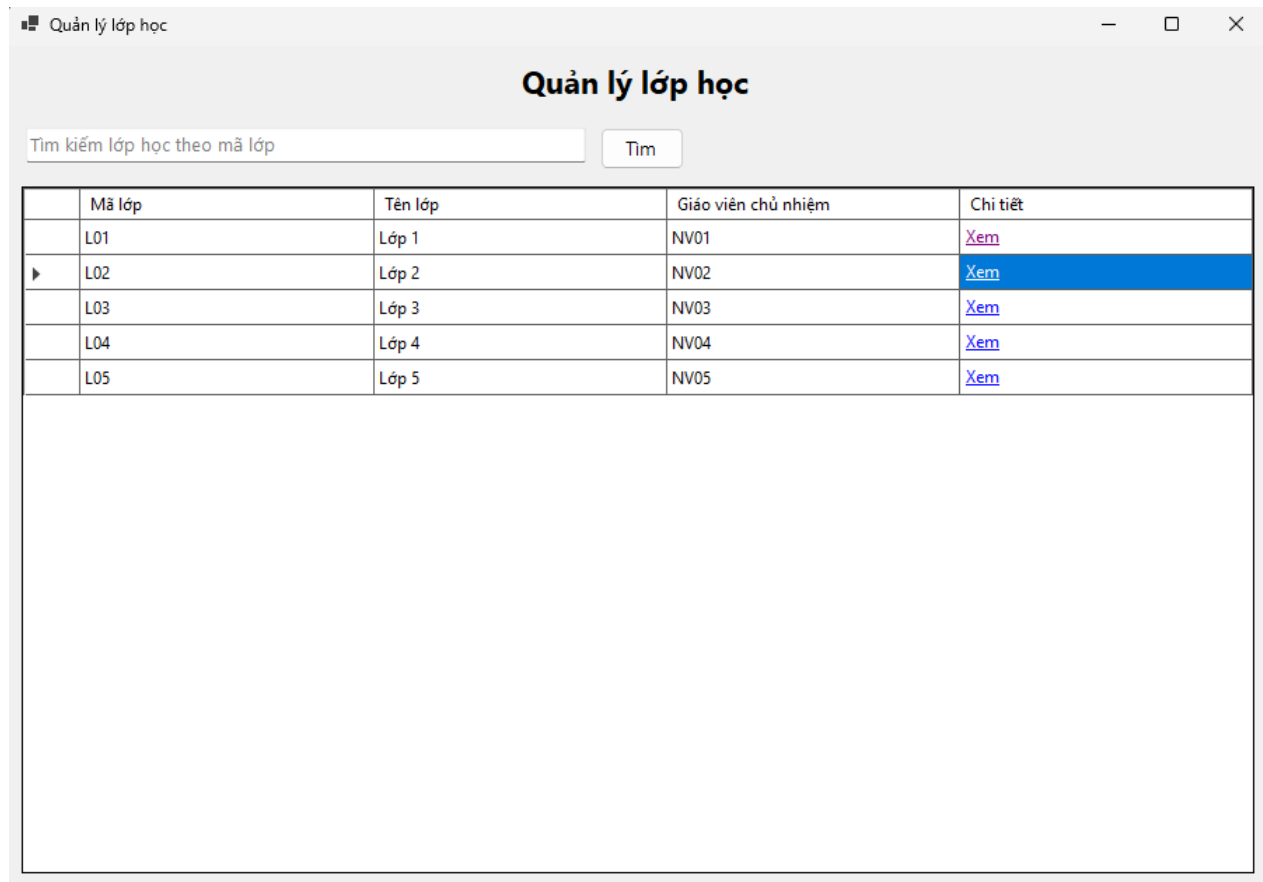
Store procedure **SP_LOGIN_NHANVIEN** nhận vào 2 giá trị là @MANV và @MK. Procedure sẽ hash @MK bằng SHA1 và kiểm tra với giá trị MATKHAU tương ứng với nhân viên @MANV trong table NHANVIEN. Nếu đúng thì sẽ trả về thông tin của nhân viên đó, còn không thì sẽ trả về NULL.


```

1 IF OBJECT_ID('SP_LOGIN_NHANVIEN', 'P') IS NOT NULL
2     DROP PROCEDURE SP_LOGIN_NHANVIEN;
3 GO
4
5 CREATE PROCEDURE SP_LOGIN_NHANVIEN
6     @MANV VARCHAR(20),
7     @MK NVARCHAR(100)
8 AS
9 BEGIN
10     SET NOCOUNT ON;
11
12     DECLARE @HashedPassword VARBINARY(100)
13
14     SELECT @HashedPassword = MATKHAU
15     FROM NHANVIEN
16     WHERE MANV = @MANV
17
18     IF @HashedPassword = HASHBYTES('SHA1', @MK)
19     BEGIN
20         SELECT MANV, HOTEN, EMAIL
21         FROM NHANVIEN
22         WHERE MANV = @MANV
23     END
24     ELSE
25     BEGIN
26         SELECT NULL AS MANV, NULL AS HOTEN, NULL AS EMAIL
27     END
28 END
29 GO

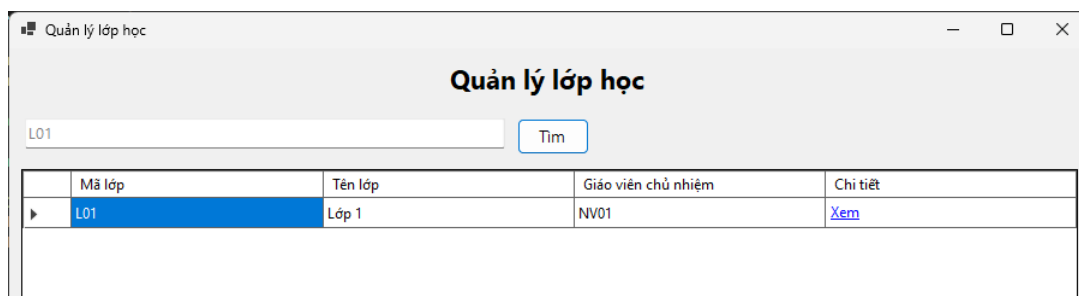
```

5.2 Màn hình quản lý lớp học



Hình 2: Giao diện quản lý lớp học

- Sau khi đăng nhập thành công, chương trình sẽ gọi procedure **SP_GET_CLASSES** để xem toàn bộ danh sách lớp trong cơ sở dữ liệu.
- Khi nhấn vào nút **Xem**, chương trình sẽ chuyển qua giao diện Danh sách sinh viên của lớp.
- Nếu người dùng nhập mã lớp vào thanh tìm kiếm và nhấn **Tìm**. Chương trình sẽ gọi procedure **SP_SEARCH_CLASSES**. Nếu MALOP có tồn tại thì hàm sẽ trả về thông tin lớp cần tìm.



Hình 3: Kết quả khi tìm kiếm lớp

Store procedure SP_GET_CLASSES dùng để trả về toàn bộ danh sách lớp trong cơ sở dữ liệu.

```

1 IF OBJECT_ID('SP_GET_CLASSES', 'P') IS NOT NULL
2 BEGIN
3     DROP PROCEDURE SP_GET_CLASSES;
4 END;
5 GO
6
7 CREATE PROCEDURE SP_GET_CLASSES
8 AS
9 BEGIN
10     SELECT MALOP, TENLOP, MANV
11     FROM LOP;
12 END
13 GO

```

Store procedure SP_SEARCH_CLASSES nhận vào biến @MALOP để thực hiện chức năng tìm kiếm.

```

1 IF OBJECT_ID('SP_SEARCH_CLASSES', 'P') IS NOT NULL
2     DROP PROCEDURE SP_SEARCH_CLASSES;
3 GO
4
5 CREATE PROCEDURE SP_SEARCH_CLASSES
6     @MALOP NVARCHAR(100)
7 AS
8 BEGIN
9     SET NOCOUNT ON;
10    SELECT MALOP, TENLOP, MANV FROM LOP
11    WHERE MALOP = @MALOP;
12 END
13 GO

```

Thông tin chi tiết lớp

Thông tin chi tiết lớp L01

Tìm kiếm sinh viên theo mã sinh viên

Tìm

MASV	HOTEN	NGAYSINH	DIACHI	MALOP
SV010	Bùi Bảo Chi	1/7/2003	Quận 11	L01
SV017	Lê Văn Quân	9/3/2004	Bình Tân	L01
SV019	Ngô Văn Quân	4/12/2003	Quận 11	L01
SV026	Phạm Quang Khoa	10/8/2003	Quận 11	L01
SV027	Hoàng Hoài Nam	12/28/2002	Quận 10	L01
SV032	Phạm Minh Nam	11/16/2002	Tân Bình	L01
SV034	Trần Bảo Uyên	10/5/2004	Nhà Bè	L01
SV039	Lê Hoài Nam	5/18/2003	Tân Bình	L01
SV045	Trần Thị Hà	6/26/2002	Bình Chánh	L01
SV047	Đặng Nhật Mai	11/19/2004	Nhà Bè	L01
SV051	Phạm Nhật Hà	9/13/2004	Quận 3	L01
SV052	Hoàng Thị Trang	9/26/2002	Bình Tân	L01
SV056	Đặng Minh Khoa	1/19/2004	Quận 5	L01
SV061	Ngô Đăng Khoa	2/11/2003	Gò Vấp	L01
SV070	Đặng Bảo An	2/8/2004	Quận 4	L01
SV087	Hồ Hữu An	5/20/2003	Gò Vấp	L01
SV090	Đỗ Hoài An	12/16/2002	Quận 2	L01

Thay đổi thông tin

Thêm sinh viên

Xóa sinh viên

Chỉnh sửa điểm

Hình 4: Giao diện Danh sách sinh viên lớp quản lý

- Chương trình sẽ gọi procedure **SP_GET_STUDENTS_BY_CLASS** để có thể hiển thị danh sách sinh viên của từng lớp.
- Nếu lớp đó do nhân viên đã đăng nhập quản lý, màn hình sẽ hiện 4 chức năng ở dưới để có thể thay đổi thông tin của sinh viên trong lớp. Khi nhấn vào thì chương trình sẽ chuyển sang các giao diện tương ứng.
- Nếu nhân viên đã đăng nhập không quản lý lớp đó thì chỉ có thể xem danh sách lớp.
- Nếu người dùng nhập mã sinh viên vào thanh tìm kiếm và nhấn **Tìm**, chương trình sẽ gọi procedure **SP_FIND_STUDENT_IN_CLASS**. Nếu mã sinh viên có tồn tại trong lớp thì sẽ trả về thông tin sinh viên đó, nếu không thì sẽ hiện MessageBox "Không tìm thấy sinh viên nào."

Thông tin chi tiết lớp

Thông tin chi tiết lớp L02

Tim kiếm sinh viên theo mã sinh viên

MASV	HOTEN	NGAYSINH	DIACHI	MALOP
SV002	Nguyễn Hữu Bình	12/12/2004	Quận 6	L02
SV007	Phạm Thị Nam	2/27/2002	Quận 10	L02
SV008	Đỗ Hoài Chi	1/21/2003	Tân Phú	L02
SV016	Bùi Hữu Phương	9/14/2002	Quận 10	L02
SV022	Ngô Bảo Sơn	1/23/2004	Quận 12	L02
SV025	Đỗ Văn Linh	9/20/2003	Quận 10	L02
SV035	Hoàng Hữu Uyên	10/21/2002	Quận 11	L02
SV036	Hồ Nhật Bình	12/10/2003	Quận 8	L02
SV038	Lê Thị Nam	11/21/2003	Nhà Bè	L02
SV040	Trần Hoài Trang	8/17/2002	Thủ Đức	L02
SV044	Đặng Thị Uyên	1/22/2003	Quận 6	L02
SV046	Đặng Thanh Hà	1/28/2003	Thủ Đức	L02
SV049	Đặng Hữu Bình	5/22/2003	Quận 2	L02
SV050	Phạm Bảo Chi	10/4/2002	Quận 11	L02
SV055	Nguyễn Hoài Linh	4/22/2002	Bình Chánh	L02
SV072	Hoàng Đăng Sơn	12/11/2002	Tân Phú	L02
SV073	Bùi Hữu Sơn	3/3/2004	Phú Nhuận	L02

Hình 5: Giao diện Danh sách sinh viên lớp không quản lý

Thông tin chi tiết lớp

Thông tin chi tiết lớp L01

SV017

MASV	HOTEN	NGAYSINH	DIACHI	MALOP
SV017	Lê Văn Quân	9/3/2004	Bình Tân	L01

Thay đổi thông tin Thêm sinh viên Xóa sinh viên Chỉnh sửa điểm

Hình 6: Kết quả tìm kiếm sinh viên bằng mã sinh viên

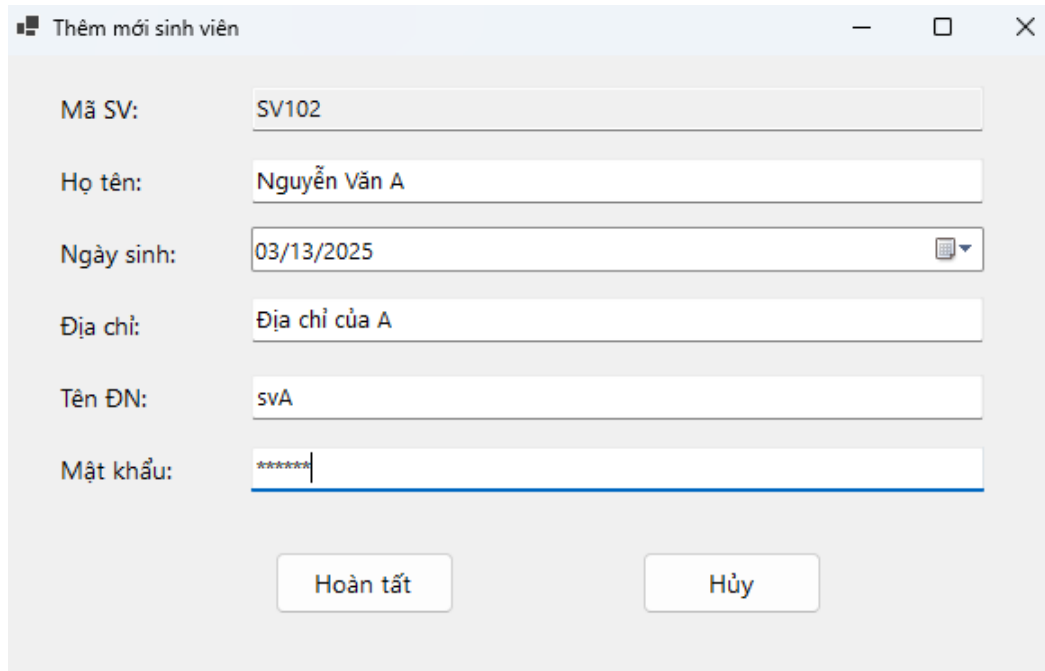
Store procedure SP_GET_STUDENTS_BY_CLASS nhận vào tham số @MALOP để có thể in ra thông tin chi tiết của các sinh viên trong lớp.

```
1 IF OBJECT_ID('SP_GET_STUDENTS_BY_CLASS', 'P') IS NOT NULL
2     DROP PROCEDURE SP_GET_STUDENTS_BY_CLASS;
3 GO
4
5 CREATE PROCEDURE SP_GET_STUDENTS_BY_CLASS
6     @MALOP NVARCHAR(20)
7 AS
8 BEGIN
9     SET NOCOUNT ON;
10    SELECT MASV, HOTEN, NGAYSINH, DIACHI, MALOP
11    FROM SINHVIEN
12    WHERE MALOP = @MALOP;
13 END
14 GO
```

Store procedure SP_FIND_STUDENT_IN_CLASSES nhận vào biến @MASV và @MALOP để tìm kiếm sinh viên trong lớp tương ứng.

```
1 IF OBJECT_ID('SP_FIND_STUDENT_IN_CLASS', 'P') IS NOT NULL
2     DROP PROCEDURE SP_FIND_STUDENT_IN_CLASS;
3 GO
4
5 CREATE PROCEDURE SP_FIND_STUDENT_IN_CLASS
6     @MASV NVARCHAR(20),
7     @MALOP NVARCHAR(20)
8 AS
9 BEGIN
10    SET NOCOUNT ON;
11    SELECT MASV, HOTEN, NGAYSINH, DIACHI, MALOP
12    FROM SINHVIEN
13    WHERE MASV = @MASV AND MALOP = @MALOP;
14 END
15 GO
```

5.4 Màn hình thêm sinh viên mới



Hình 7: Giao diện thêm sinh viên

- Người dùng phải là giáo viên quản lý của lớp đang hiển thị
- Sau khi bật form thì chương trình sẽ chạy procedure **SP_GENERATE_NEW_STUDENT_ID** để tạo ID sinh viên mới nhất. Bằng cách lấy số của sinh viên cuối cùng và cộng thêm 1 hoặc nếu chưa tồn tại sinh viên nào thì ID sẽ là SV001.
- Sau khi điền đầy đủ thông tin, khi nhấn vào nút **Xác nhận** thì chương trình sẽ chạy procedure **SP_INSERT_STUDENT** với các tham số được thêm vào tương ứng dưới database.
- Nếu kiểm tra sự hợp lệ của tên đăng nhập hay mã lớp, Stored procedures sẽ thực hiện thêm sinh viên vào database.
- Nút **Hủy** để xóa hết các thông tin người dùng vừa nhập.

Store procedure **SP_INSERT_STUDENT** nhận vào các giá trị là @MASV, @HOTEN, @NGAYSINH, @DIACHI, @TENDN, @MATKHAU, @MALOP và trả ra @ErrorMessage nếu có lỗi. Procedure sẽ hash @MATKHAU bằng SHA2_256.

```

1 IF OBJECT_ID('SP_GENERATE_NEW_STUDENT_ID', 'P') IS NOT NULL
2     DROP PROCEDURE SP_GENERATE_NEW_STUDENT_ID;
3 GO
4
5 CREATE PROCEDURE SP_GENERATE_NEW_STUDENT_ID
6 AS
7 BEGIN
8     DECLARE @NewID INT;
9     DECLARE @MaxID NVARCHAR(20);

```

```

10     SELECT @MaxID = MAX(MASV) FROM SINHVIEN;
11     IF @MaxID IS NULL
12         SET @NewID = 1;
13     ELSE
14         SET @NewID = CAST(SUBSTRING(@MaxID, PATINDEX('%[0-9]%', @MaxID), LEN(
15     @MaxID)) AS INT) + 1;
16     SELECT 'SV' + RIGHT('000' + CAST(@NewID AS NVARCHAR(10)), 3) AS NewMASV;
17 END
18 GO
19 IF OBJECT_ID('SP_INSERT_STUDENT', 'P') IS NOT NULL
20     DROP PROCEDURE SP_INSERT_STUDENT;
21 GO
22
23 CREATE PROCEDURE SP_INSERT_STUDENT
24     @MASV VARCHAR(20),
25     @HOTEN NVARCHAR(100),
26     @NGAYSINH DATETIME,
27     @DIACHI NVARCHAR(200),
28     @MALOP VARCHAR(20),
29     @TENDN NVARCHAR(100),
30     @MATKHAU NVARCHAR(100),
31     @ErrorMessage NVARCHAR(200) OUTPUT
32 AS
33 BEGIN
34     BEGIN TRY
35         SET NOCOUNT ON;
36
37         IF EXISTS (SELECT 1 FROM SINHVIEN WHERE TENDN = @TENDN)
38             BEGIN
39                 SET @ErrorMessage = 'Ten dang nhap da ton tai!';
40                 RETURN;
41             END
42
43         IF NOT EXISTS (SELECT 1 FROM LOP WHERE MALOP = @MALOP)
44             BEGIN
45                 SET @ErrorMessage = 'Ma lop khong hop le!';
46                 RETURN;
47             END
48
49         DECLARE @EncryptedPassword VARBINARY(MAX);
50         SET @EncryptedPassword = HASHBYTES('SHA2_256', @MATKHAU);
51
52         INSERT INTO SINHVIEN (MASV, HOTEN, NGAYSINH, DIACHI, MALOP, TENDN,
53     MATKHAU)
54         VALUES (@MASV, @HOTEN, @NGAYSINH, @DIACHI, @MALOP, @TENDN,
55     @EncryptedPassword);
56
57         SET @ErrorMessage = '';
58     END TRY
59     BEGIN CATCH
60         SET @ErrorMessage = ERROR_MESSAGE();
61     END CATCH
62 END;
63 GO

```


5.5 Màn hình xóa sinh viên

Hình 8: Giao diện xóa sinh viên

- Người dùng phải là giáo viên quản lý của lớp đang hiển thị
- Sau khi bật form thì chỉ cần nhập mã sinh viên thì chương trình sẽ chạy procedure **SP_FIND_STUDENT_IN_CLASS** để tìm sinh viên và hiển thị thông tin sinh viên đó.
- Sau khi đã hiện đầy đủ thông tin, khi nhấn vào nút **Xác nhận** thì chương trình sẽ hỏi xác nhận lại rồi chạy procedure **SP_DELETE_STUDENT** với các tham số được thêm vào tương ứng dưới database.
- Nếu kiểm tra sự hợp lệ của input, Stored procedures sẽ thực hiện xóa sinh viên vào database.
- Nút **Hủy** để xóa hết các thông tin người dùng vừa nhập.

Store procedure **SP_DELETE_STUDENT** nhận vào các giá trị là @MASV, @MALOP và trả ra @ErrorMessage nếu có lỗi.

```

1 IF OBJECT_ID('SP_DELETE_STUDENT', 'P') IS NOT NULL
2     DROP PROCEDURE SP_DELETE_STUDENT;
3 GO
4
5 CREATE PROCEDURE SP_DELETE_STUDENT
6     @MASV NVARCHAR(20),
7     @MALOP NVARCHAR(20),
8     @ErrorMessage NVARCHAR(200) OUTPUT
9 AS
10 BEGIN
11     SET NOCOUNT ON;
12     IF NOT EXISTS(SELECT 1 FROM SINHVIEN WHERE MASV = @MASV AND MALOP = @MALOP)

```

```

13 BEGIN
14     SET @ErrorMessage = 'Sinh vien khong ton tai trong lop.';
15     RETURN;
16 END
17
18 DELETE FROM BANGDIEM WHERE MASV = @MASV;
19 DELETE FROM SINHVIEN
20 WHERE MASV = @MASV AND MALOP = @MALOP;
21
22 SET @ErrorMessage = '';
23 END
24 GO

```

5.6 Màn hình điều chỉnh thông tin sinh viên

Hình 9: Giao diện sửa thông tin sinh viên

- Người dùng phải là giáo viên quản lý của lớp đang hiển thị
- Sau khi bật form thì chỉ cần nhập mã sinh viên thì chương trình sẽ chạy procedure **SP_FIND_STUDENT_IN_CLASS** để tìm sinh viên và hiển thị thông tin sinh viên đó.
- Sau khi đã hiện đầy đủ thông tin, người dùng có thể chỉnh sửa thông tin so với thông tin hiện tại đang được hiển thị. Khi nhấn vào nút **Xác nhận** thì chương trình sẽ chạy procedure **SP_UPDATE_STUDENT_INFO** với các tham số được thêm vào tương ứng dưới database.
- Nếu kiểm tra sự hợp lệ của input, Stored procedures sẽ thực hiện chỉnh sửa thông tin sinh viên vào database.
- Nút **Hủy** để xóa hết các thông tin người dùng vừa nhập.

Store procedure **SP_UPDATE_STUDENT_INFO** nhận vào 6 giá trị là @MASV, @HOTEN, @NGAYSINH, @DIACHI, @OLD_MALOP, @NEW_MALOP và trả ra @ErrorMessage nếu có lỗi.

```

1 IF OBJECT_ID('SP_UPDATE_STUDENT_INFO', 'P') IS NOT NULL
2     DROP PROCEDURE SP_UPDATE_STUDENT_INFO;
3 GO
4
5 CREATE PROCEDURE SP_UPDATE_STUDENT_INFO
6     @MASV NVARCHAR(20),
7     @HOTEN NVARCHAR(100),
8     @NGAYSINH DATETIME,
9     @DIACHI NVARCHAR(200),

```

```

10  @OLD_MALOP NVARCHAR(20),
11  @NEW_MALOP NVARCHAR(20),
12  @ErrorMessage NVARCHAR(200) OUTPUT
13  AS
14  BEGIN
15      SET NOCOUNT ON;
16      IF NOT EXISTS(SELECT 1 FROM SINHVIEN WHERE MASV = @MASV AND MALOP =
@OLD_MALOP)
17          BEGIN
18              SET @ErrorMessage = 'Sinh vien khong ton tai trong lop hien tai';
19              RETURN;
20          END
21
22      IF @NEW_MALOP <> @OLD_MALOP
23          BEGIN
24              IF NOT EXISTS(SELECT 1 FROM LOP WHERE MALOP = @NEW_MALOP)
25                  BEGIN
26                      SET @ErrorMessage = 'Ma lop moi khong ton tai.';
27                      RETURN;
28                  END
29          END
30
31      UPDATE SINHVIEN
32      SET HOTEN = @HOTEN,
33          NGAYSINH = @NGAYSINH,
34          DIACHI = @DIACHI,
35          MALOP = @NEW_MALOP
36      WHERE MASV = @MASV AND MALOP = @OLD_MALOP;
37
38      SET @ErrorMessage = '';
39  END
40  GO

```

5.7 Màn hình chi tiết điểm

Mã SV	Họ Tên	Mã HP	Tên Học Phần	Điểm
SV010	Bùi Bảo Chi	HP01	Cấu trúc dữ liệu và g...	10.00
SV010	Bùi Bảo Chi	HP04	Triết học Mác - Lênin	6.00
SV019	Ngô Văn Quân	HP02	Mạng máy tính	8.00
SV026	Phạm Quang Khoa	HP02	Mạng máy tính	9.00

Chỉnh sửa điểm

Hình 10: Giao diện chi tiết điểm của lớp

- Người dùng phải là giáo viên quản lý của lớp đang hiển thị
- Khi truy cập vào form thì chương trình sẽ chạy procedure **SP_GET_SCORES_BY_CLASS** với các tham số là mã lớp hiện tại, public key và mật khẩu để giải mã điểm. Danh sách toàn bộ điểm của các sinh viên trong lớp đó sẽ hiện lên.
- Khi nhấn vào **Chỉnh sửa điểm**, chương trình sẽ mở ra giao diện để nhân viên thêm / chỉnh sửa điểm cho sinh viên cụ thể.

Store procedure **SP_GET_SCORES_BY_CLASS** nhận vào các giá trị là @MALOP, @MK, @PUBKEY và trả ra @ErrorMessage nếu có lỗi. Điểm sẽ được giải mã.

```

1 IF OBJECT_ID('SP_GET_SCORES_BY_CLASS', 'P') IS NOT NULL
2     DROP PROCEDURE SP_GET_SCORES_BY_CLASS;
3 GO
4
5 CREATE PROCEDURE SP_GET_SCORES_BY_CLASS
6     @MALOP VARCHAR(20),
7     @PUBKEY VARCHAR(20),
8     @MK NVARCHAR(20),
9     @ErrorMessage NVARCHAR(200) OUTPUT
10 AS
11 BEGIN

```

```

12 SET NOCOUNT ON;
13 IF NOT EXISTS (SELECT 1 FROM LOP WHERE MALOP = @MALOP)
14 BEGIN
15     SET @ErrorMessage = 'Ma lop khong ton tai!';
16     RETURN;
17 END
18
19 IF NOT EXISTS (SELECT 1 FROM sys.asymmetric_keys WHERE name = @PUBKEY)
20 BEGIN
21     SET @ErrorMessage = 'Khoa giai ma khong hop le!';
22     RETURN;
23 END
24
25 SELECT
26     S.MASV,
27     S.HOTEN,
28     B.MAHP,
29     H.TENHP,
30     CAST(
31         DecryptByAsymKey(AsymKey_ID(@PUBKEY), B.DIEMTHI, @MK)
32         AS DECIMAL(18,2)) AS DIEM
33 FROM BANGDIEM B
34 JOIN SINHVIEN S ON B.MASV = S.MASV
35 JOIN HOCPHAN H ON B.MAHP = H.MAHP
36 WHERE S.MALOP = @MALOP;
37 END
38 GO

```

5.8 Màn hình thêm / chỉnh sửa điểm

Hình 11: Giao diện sửa điểm sinh viên

- Người dùng phải là giáo viên quản lý của lớp đang hiển thị
- Sau khi chọn chỉnh sửa và đã nhập đầy đủ thông tin để sửa điểm trong form điểm chi tiết, người dùng nhấn vào nút **Xác nhận** thì chương trình sẽ chạy procedure **SP_INS_DIEM** với các tham số được thêm vào tương ứng dưới database.
- Với sinh viên đã có điểm ở học phần đó, điểm sẽ được cập nhật mới. Còn với sinh viên chưa có điểm ở học phần đó, điểm sẽ được thêm vào.
- Nếu kiểm tra sự hợp lệ của input, Stored procedures sẽ thực hiện thêm / cập nhật điểm sinh viên vào database.
- Nút **Hủy** để xóa hết các thông tin người dùng vừa nhập.

Store procedure **SP_INS_DIEM** nhận vào các giá trị là **@MASV**, **@MAHP**, **@DIEMTHI**, **@PUBKEY** và trả ra **@ErrorMessage** nếu có lỗi. Điểm sẽ được mã hóa.

```

1 IF OBJECT_ID('SP_INS_DIEM', 'P') IS NOT NULL
2     DROP PROCEDURE SP_INS_DIEM;
3 GO
4
5 CREATE PROCEDURE SP_INS_DIEM
6     @MASV VARCHAR(20),
7     @MAHP VARCHAR(20),
8     @DIEMTHI DECIMAL(4,2),
9     @PUBKEY VARCHAR(20),

```

```

10      @ErrorMessage NVARCHAR(200) OUTPUT
11 AS
12 BEGIN
13     BEGIN TRY
14         IF EXISTS (SELECT 1 FROM BANGDIEM WHERE MASV = @MASV AND MAHP = @MAHP)
15             BEGIN
16                 UPDATE BANGDIEM
17                     SET DIEMTHI = ENCRYPTBYASYMKEY(AsymKey_ID(@PUBKEY), CAST(@DIEMTHI AS
VARBINARY))
18                     WHERE MASV = @MASV AND MAHP = @MAHP;
19                 SET @ErrorMessage = '';
20             END
21         ELSE
22             BEGIN
23                 INSERT INTO BANGDIEM (MASV, MAHP, DIEMTHI)
24                     VALUES (@MASV, @MAHP, ENCRYPTBYASYMKEY(AsymKey_ID(@PUBKEY), CAST(
@DIEMTHI AS VARBINARY)));
25             END
26             SET @ErrorMessage = '';
27         END
28     END TRY
29     BEGIN CATCH
30         SET @ErrorMessage = ERROR_MESSAGE();
31     END CATCH
32 END;

```


6 Câu e

6.1 Giới Thiệu

SQL Server Profiler là một công cụ giúp theo dõi, ghi nhận và phân tích hoạt động của SQL Server. Công cụ này hữu ích trong việc giám sát hiệu suất truy vấn, phát hiện và khắc phục sự cố.

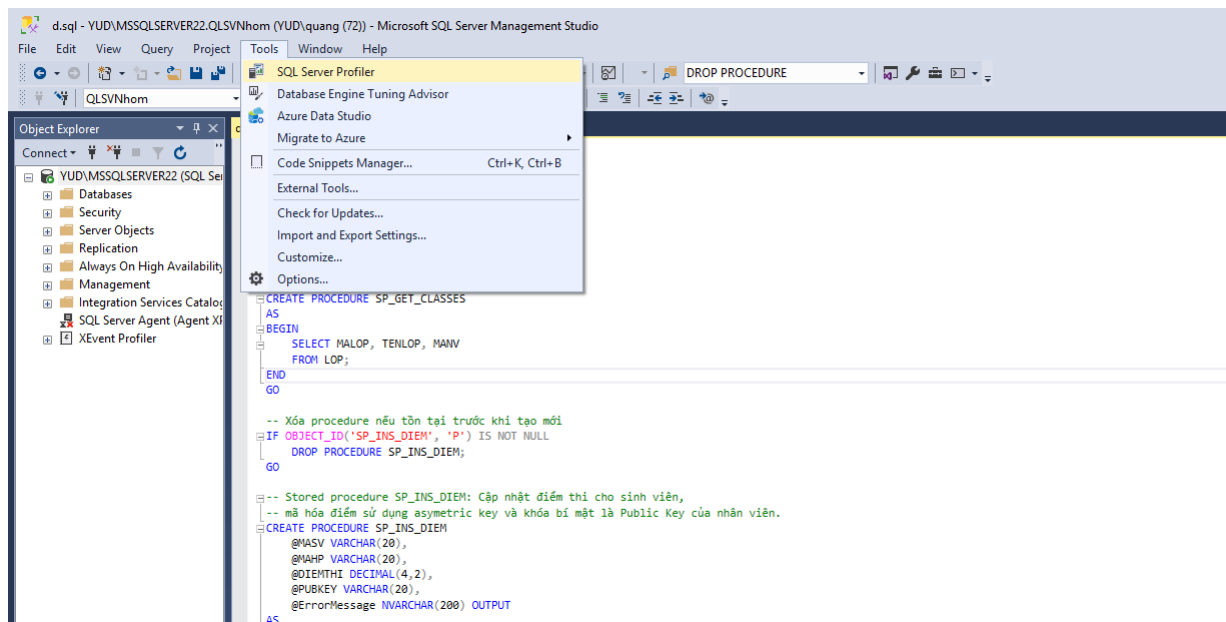
6.2 Mục Đích Sử Dụng

- Theo dõi lịch sử truy vấn từ người dùng.
- Phát hiện các truy vấn chạy chậm để tối ưu hiệu suất.
- Kiểm tra các lệnh SQL gây tắc nghẽn.
- Ghi lại hoạt động truy vấn để phân tích bảo mật.

6.3 Quá Trình Thực Hiện

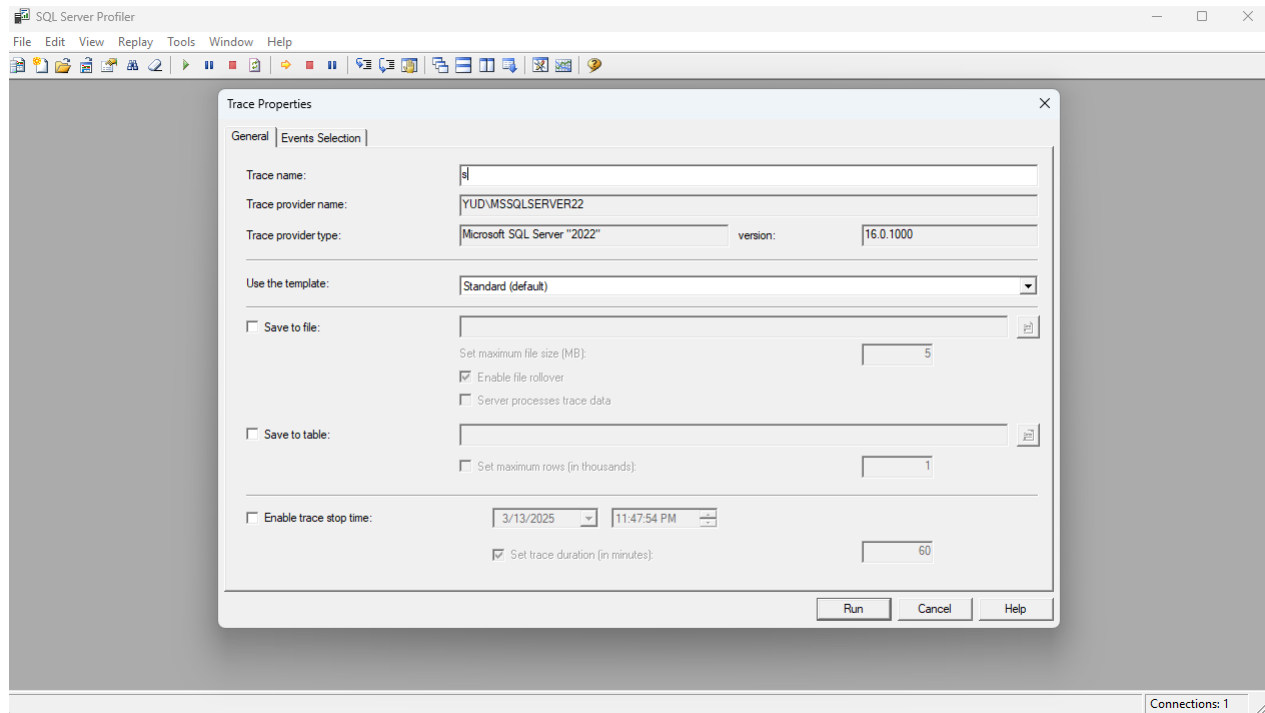
6.3.1 Cài Đặt SQL Server Profiler

1. Mở SQL Server Management Studio (SSMS).
2. Chuyển đến Tools → SQL Server Profiler.
3. Kết nối đến SQL Server.



Hình 12: Kết nối SQL Profiler

6.3.2 Cấu Hình SQL Profiler



Hình 13: Cấu hình Tracer

6.3.3 Phân Tích Dữ Liệu

1. Mở file .trc hoặc dùng truy vấn SQL:

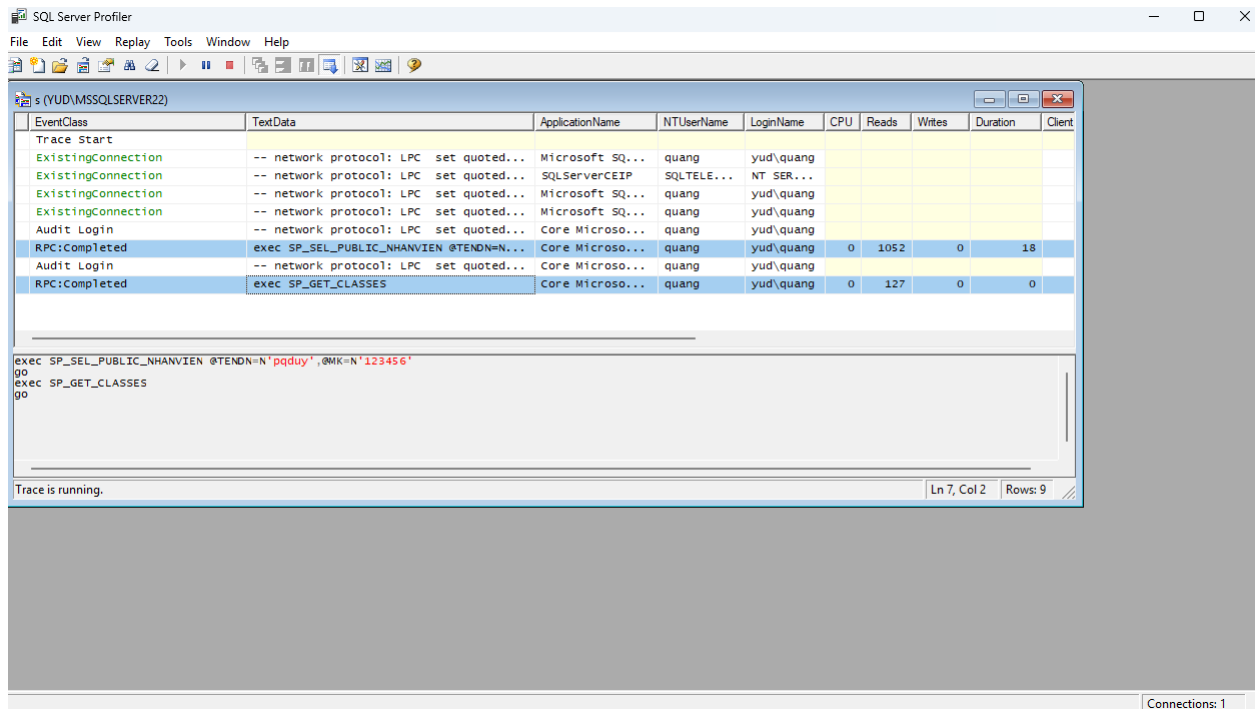
```
SELECT * FROM user_query_log_table;
```

2. Xem truy vấn chạy lâu nhất:

```
SELECT Duration, TextData FROM user_query_log_table ORDER BY Duration DESC;
```

6.3.4 Theo dõi truy vấn

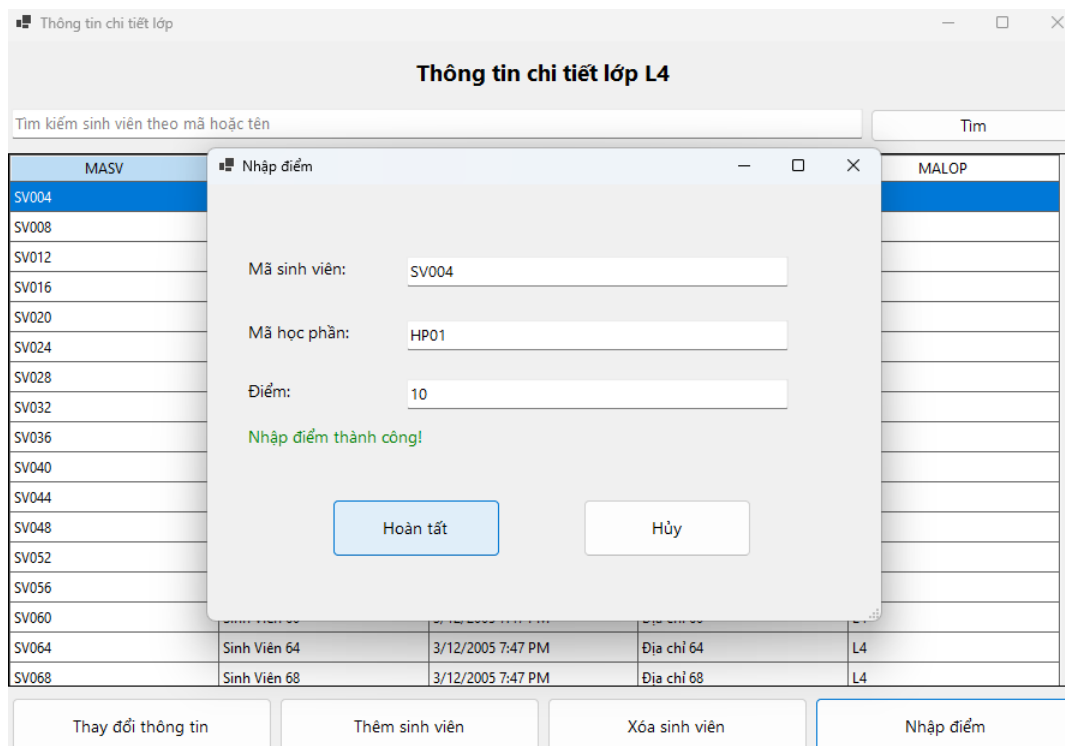
Khi thực hiện thao tác trên app, có tương tác với database, mọi thao tác sẽ được lưu lại trong file tracer đó. Ví dụ dưới là cho việc login của nhân viên.



Hình 14: Tracing

6.4 Thực hiện với form thêm điểm

Khi bật form, nhập dữ liệu và xác nhận, đoạn SQL Script sẽ được thực thi



Hình 15: Tracing

Sau đó đoạn SQL script đó cũng sẽ được lưu vào tracer

RPC:Completed	exec sp_reset_connection	Core Microso...	quang	yud\
Audit Login	-- network protocol: LPC set quoted...	Core Microso...	quang	yud\
RPC:Completed	exec SP_FIND_STUDENT_IN_CLASS @MASV=...	Core Microso...	quang	yud\
RPC:Completed	declare @p5 nvarchar(200) set @p5=N...	Core Microso...	quang	yud\

```
declare @p5 nvarchar(200)
set @p5=N''
exec SP_INS_DIEM @MASV=N'SV004',@MAHP=N'HP01',@DIEMTHI=10,@PUBKEY=N'NV04',@ErrorMessage=@p5 output
select @p5
```

Hình 16: Tracing