**TRƯỜNG ĐẠI HỌC KINH TẾ - ĐẠI HỌC ĐÀ NẴNG**

🙜🙜🙜🕮🙜🙜🙜



**QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**HỆ THÔNG QUẢN LÝ NHÂN SỰ**

Lớp : 48K14.2

Nhóm : 48K142.06

Giảng viên hướng dẫn : Cao Thị Nhâm

Thành viên nhóm : Nguyễn Ngọc Oanh (Leader)

Nguyễn Thị Thu Giang

Trần Thị Xuân Mai

Trần Bảo Thiên Anh

**Đà Nẵng,** **11/2024**

**MỤC LỤC**

[**I.** **THIẾT KẾ CƠ SỞ DỮ LIỆU** 3](#_Toc183799021)

[1. Chuẩn hóa 3](#_Toc183799022)

[2. Thiết kế cơ sở dữ liệu mức logic (dạng chuẩn 2NF) 3](#_Toc183799023)

[3. Thiết kế chi tiết các bảng 4](#_Toc183799024)

[**II.** **XÂY DỰNG CƠ SỞ DỮ LIỆU** 5](#_Toc183799025)

[**III.** **XÂY DỰNG DỮ LIỆU VÀ MODULE ĐỂ XỬ LÝ DỮ LIỆU** 7](#_Toc183799026)

[1. Module sinh dữ liệu dump cho từng bảng 7](#_Toc183799027)

[2. Module xử lý dữ liệu 11](#_Toc183799028)

[3. Thay đổi trong thiết kế 17](#_Toc183799029)

[**IV.** **Xây dựng cơ chế bảo mật phù hợp cho cơ sở dữ liệu** 18](#_Toc183799030)

[1. Thiết lập cơ chế authentication 18](#_Toc183799031)

[2. Tạo và phân quyền cho người dùng 18](#_Toc183799032)

[3. Mã hóa các dữ liệu nhạy cảm 46](#_Toc183799033)

[**V.** **Xây dựng cơ chế backup dữ liệu tự động cho cơ sở dữ liệu** 50](#_Toc183799034)

[**VI.** **PHƯƠNG ÁN GIẢI QUYẾT VẤN ĐỀ DỮ LIỆU RẤT LỚN KHÔNG ĐỦ LƯU TRỮ TRÊN MỘT Ổ ĐĨA** 60](#_Toc183799035)

[**VII.** **ĐỀ RA VÀ THỰC HIỆN PHƯƠNG ÁN ĐỂ HẠN CHẾ TẤN CÔNG SQL INJECTION** 70](#_Toc183799036)

[1. Thủ tục thêm mới dữ liệu: 70](#_Toc183799037)

[2. Thủ tục truy xuất dữ liệu 71](#_Toc183799038)

[3. Thủ tục cập nhật dữ liệu 72](#_Toc183799039)

[4. Xóa dữ liệu bằng thủ tục 73](#_Toc183799040)

## **THIẾT KẾ CƠ SỞ DỮ LIỆU**

### Chuẩn hóa

NHANVIEN (**MaNV**, HoTen, NgaySinh, SDT, GioiTinh, DiaChi, *MaCV*)

CHUCVU (**MaCV**, TenCV)

TAIKHOAN (**MaTK**, TenDangNhap, MatKhau, *MaNV*)

CHAMCONG (**MaCC**, Ngay, GioVao, GioRa, *MaNV*)

LUONG (**MaLuong**, LuongTheoGio, Thuong, Phat, *MaNV)*

### Thiết kế cơ sở dữ liệu mức logic (dạng chuẩn 2NF)

A diagram of a computer

Description automatically generated

### Thiết kế chi tiết các bảng

#### Bảng CHUCVU

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| 1 | MaCV | VARCHAR(20) | Primary Key | Mã chức vụ |
| 2 | TenCV | NVARCHAR(50) | Not null | Tên chức vụ |

#### Bảng NHANVIEN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| 1 | MaNV | VARCHAR(20) | Primary key | Mã nhân viên |
| 2 | HoTen | NVARCHAR(50) | Not null | Họ tên |
| 3 | NgaySinh | DATE | Not null | Ngày sinh |
| 4 | SDT | CHAR(10) | Not null | Số điện thoại |
| 5 | GioiTinh | NVARCHAR(10) |  | Giới tính |
| 6 | DiaChi | NVARCHAR(70) |  | Địa chỉ |
| 7 | MaCV | VARCHAR(20) | Foreign Key | Mã chức vụ |

#### Bảng TAIKHOAN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| 1 | MaTK | VARCHAR(20) | Primary Key | Mã tài khoản |
| 2 | TenDangNhap | VARCHAR(50) | Unique, Not null | Tên đăng nhập |
| 3 | MatKhau | VARCHAR(50) | Not null | Mật khẩu |
| 4 | MaNV | VARCHAR(20) | Foreign Key | Mã nhân viên |

#### Bảng LUONG

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| 1 | MaLuong | VARCHAR(20) | Primary key | Mã lương |
| 2 | LuongTheoGio | FLOAT | >0 | Lương 1 giờ làm |
| 3 | Thuong | FLOAT | >=0 | Thưởng |
| 4 | Phat | FLOAT | >=0 | Phạt |
| 5 | MaNV | VARCHAR(20) | Foreign Key | Mã nhân viên |

#### Bảng CHAMCONG

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên cột** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| 1 | MaCC | VARCHAR(20) | Primary Key | Mã chấm công |
| 2 | MANV | VARCHAR(20) | Foreign Key | Mã nhân viên |
| 3 | Ngay | Date | Not null | Ngày |
| 4 | GioVao | Time | Not null | Giờ vào |
| 5 | GioRa | Time | Not null | Giờ ra |

## **XÂY DỰNG CƠ SỞ DỮ LIỆU**

1. **Script tạo bảng**

CREATE DATABASE NHANSU

USE NHANSU

--Tạo bảng CHUCVU

CREATE TABLE CHUCVU (

MaCV VARCHAR(20),

TenCV NVARCHAR(50) NOT NULL,

PRIMARY KEY (MaCV)

)

--Tạo bảng NHANVIEN

CREATE TABLE NHANVIEN (

MaNV VARCHAR(20),

HoTen NVARCHAR(50) NOT NULL,

NgaySinh DATE NOT NULL,

GioiTinh NVARCHAR(10),

DiaChi NVARCHAR(70),

SDT CHAR(10) NOT NULL,

MaCV VARCHAR(20),

PRIMARY KEY (MaNV),

FOREIGN KEY (MaCV) REFERENCES CHUCVU(MaCV)

)

--Tạo bảng TAIKHOAN

CREATE TABLE TAIKHOAN (

MaTK VARCHAR(20),

TenDangNhap VARCHAR(50) NOT NULL CONSTRAINT UQ\_TenDangNhap\_TK UNIQUE,

MatKhau VARCHAR(50) NOT NULL,

MaNV VARCHAR(20),

PRIMARY KEY (MaTK),

FOREIGN KEY (MaNV) REFERENCES NHANVIEN(MaNV)

)

--Tạo bảng LUONG

CREATE TABLE LUONG (

MaLuong VARCHAR(20),

MaNV VARCHAR(20),

LuongTheoGio FLOAT,

CONSTRAINT CK\_LTG CHECK (LuongTheoGio>0),

Thuong FLOAT,

CONSTRAINT CK\_Thuong CHECK (Thuong>=0),

Phat FLOAT,

CONSTRAINT CK\_Phat CHECK (Phat>=0),

PRIMARY KEY (MaLuong),

FOREIGN KEY (MaNV) REFERENCES NHANVIEN(MaNV)

)

--Tạo bảng CHAMCONG

CREATE TABLE CHAMCONG (

MaCC VARCHAR(20),

MaNV VARCHAR(20),

Ngay DATE NOT NULL,

GioVao TIME NOT NULL,

GioRa TIME NOT NULL,

PRIMARY KEY (MaCC),

FOREIGN KEY (MaNV) REFERENCES NHANVIEN(MaNV)

)

1. **Database Diagrams**

A diagram of a company

Description automatically generated with medium confidence

## **XÂY DỰNG DỮ LIỆU VÀ MODULE ĐỂ XỬ LÝ DỮ LIỆU**

### Module sinh dữ liệu dump cho từng bảng

1. Tạo dữ liệu dump cho bảng CHUCVU

* **Mục đích**: Mục đích của thủ tục ThemChucVu là tạo ra và chèn ngẫu nhiên 1000 chức vụ vào bảng CHUCVU. Mỗi chức vụ sẽ có một mã chức vụ (MaCV) và tên chức vụ (TenCV). Mã chức vụ được tạo dựa trên ký tự đầu tiên của tên chức vụ, theo sau là một chuỗi số.
* **Input:** N/A
* **Output:** 1000 dòng dữ liệu được thêm vào bảng CHUCVU
* **Process:**

+ **Bước 1:** Khởi tạo biến

@i: Được sử dụng làm biến đếm từ 1 đến 1000 (số lượng chức vụ cần tạo).

@TenCV: Lưu trữ tạm thời tên chức vụ được chọn ngẫu nhiên từ danh sách.

@MaCV: Lưu trữ mã chức vụ, được tạo từ ký tự đầu tiên của tên chức vụ và số thứ tự.

**+ Bước 2:** Tạo bảng tạm@DanhSachChucVu

Bảng này lưu danh sách các chức vụ mẫu, bao gồm 5 chức vụ:

Quản lý

Ca trưởng

Nhân viên full-time

Nhân viên part-time

Phụ bếp

**+ Bước 3:** Vòng lặp While

Thủ tục thực hiện lặp từ @i = 1 đến @i = 1000.

Trong mỗi vòng lặp:

* + - * + Chọn ngẫu nhiên tên chức vụ từ bảng tạm @DanhSachChucVu bằng cách tạo số ngẫu nhiên @RandomChucVu.
        + Lấy tên chức vụ tương ứng từ bảng @DanhSachChucVu.
        + Tạo mã chức vụ @MaCV bằng cách kết hợp ký tự đầu của tên chức vụ và chuỗi ‘000’ + số thứ tự (chuỗi gồm 4 chữ số).
        + Thực hiện thêm dữ liệu vào bảng CHUCVU với các giá trị @MaCV và @TenCV.
        + Tăng giá trị của biến @i để chuyển sang vòng lặp tiếp theo.

1. Tạo dữ liệu dump cho bảng NHANVIEN

* **Mục đích:** Thủ tục ThemNhanVien có mục đích tạo ngẫu nhiên và chèn 1000 nhân viên vào bảng NHANVIEN với thông tin cá nhân bao gồm: mã nhân viên, họ tên, ngày sinh, giới tính, địa chỉ, số điện thoại, và mã chức vụ.
* **Input:** N/A
* **Output:** 1000 dòng dữ liệu được thêm vào bảng NHANVIEN
* **Process:**

+ **Bước 1:** Khởi tạo biến

@i: Biến đếm dùng để lặp từ 1 đến 1000 (số lượng nhân viên cần tạo).

@Ho, @HoLot, @Ten: Biến lưu họ, họ lót, và tên của nhân viên.

@DiaChi: Biến lưu địa chỉ của nhân viên.

@SDT: Biến lưu số điện thoại của nhân viên.

@GioiTinh: Biến lưu giới tính của nhân viên.

@MaCV: Biến lưu mã chức vụ của nhân viên.

**+ Bước 2:** Tạo các bảng tạmlưu giá trị ngẫu nhiên

Bảng tạm @DanhSachHo lưu họ, @DanhSachHoLot lưu họ lót, @DanhSachTen lưu tên, @DanhSachDiaChi lưu địa chỉ, và @DanhSachGioiTinh lưu giới tính.

* + Mỗi bảng tạm có cột ID tự động tăng để chọn ngẫu nhiên giá trị dựa

trên số ID.

**+ Bước 3:** Vòng lặp While

Vòng lặp chạy 1000 lần, mỗi lần sẽ thực hiện các công việc sau:

* + Chọn ngẫu nhiên giá trị từ các bảng tạm (họ, họ lót, tên, địa chỉ, giới tính).
  + Chọn ngẫu nhiên MaCV (Mã chức vụ) từ bảng CHUCVU.
  + Tạo ngày sinh ngẫu nhiên cho nhân viên trong độ tuổi từ 22 đến 42.
  + Tạo số điện thoại ngẫu nhiên.
  + Chèn dữ liệu vào bảng NHANVIEN.
  + Tăng giá trị của @i để tiếp tục vòng lặp cho nhân viên tiếp theo.

1. Tạo dữ liệu dump cho bảng LUONG

* **Mục đích:** Thủ tục ThemLuong có mục đích tạo ngẫu nhiên và chèn 1000 dòng lương cho 1000 nhân viên vào bảng LUONG. Mỗi dòng bao gồm mã lương, mã nhân viên, lương theo giờ (dựa trên chức vụ của nhân viên), phần trăm thưởng và phần trăm phạt.
* **Input:** N/A
* **Output:** 1000 dòng dữ liệu được thêm vào bảng LUONG
* **Process:**

+ **Bước 1:** Khởi tạo biến

@i: Biến đếm, bắt đầu từ 1 và tăng dần đến 1000 (tạo 1000 bản ghi).

@MaCV: Biến lưu mã chức vụ của nhân viên.

@LuongTheoGio: Biến lưu mức lương theo giờ, được xác định dựa trên chức vụ.

**+ Bước 2:** Vòng lặp While

Vòng lặp chạy 1000 lần thực hiện các công việc :

* + Truy vấn MaCV (Mã chức vụ) từ bảng NHANVIEN dựa trên MaNV (Mã nhân viên).
  + Xác định mức lương theo giờ dựa trên mã chức vụ (MaCV):
    - Mã bắt đầu bằng 'Q': Quản lý (35,000 đồng/giờ)
    - Mã bắt đầu bằng 'C': Ca trưởng (30,000 đồng/giờ)
    - Mã bắt đầu bằng 'P': Phụ bếp (28,000 đồng/giờ)
    - Mã bắt đầu bằng 'N': Nhân viên full-time/part-time (20,000 đồng/giờ)
    - Nếu mã chức vụ không thuộc các loại trên, lương theo giờ sẽ được tạo ngẫu nhiên trong khoảng từ 50 đến 150 đồng/giờ.
  + Tạo ngẫu nhiên phần trăm thưởng và phần trăm phạt từ 0% đến 30%.
  + Chèn thông tin lương vào bảng LUONG:

MaLuong: Định dạng mã lương là 'L' + MaNV (ví dụ: 'L0001', 'L0002').

MaNV: Mã nhân viên c với định dạng tương ứng như MaNV trong bảng NHANVIEN.

LuongTheoGio: Mức lương theo giờ đã xác định.

Thuong: Phần trăm thưởng, được tạo ngẫu nhiên trong khoảng từ 0% đến 30%.

Phat: Phần trăm phạt, cũng được tạo ngẫu nhiên trong khoảng từ 0% đến 30%.

1. Tạo dữ liệu dump cho bảng CHAMCONG

* **Mục đích:** Thủ tục ThemChamCong có mục đích tạo ngẫu nhiên 1000 bản ghi chấm công cho 1000 nhân viên vào bảng CHAMCONG. Mỗi bản ghi bao gồm mã chấm công, mã nhân viên, ngày chấm công, giờ vào và giờ ra.
* **Input:** N/A
* **Output:** 1000 dòng dữ liệu được thêm vào bảng CHAMCONG
* **Process:**

+ **Bước 1:** Khởi tạo biến

@Ngay: Ngày làm việc của nhân viên, được tạo ngẫu nhiên trong vòng 30 ngày gần nhất.

@GioVao: Giờ vào làm việc, được tạo ngẫu nhiên trong khung 13h30-14h (ca chiều) hoặc 18h-18h30 (ca tối).

@GioRa: Giờ ra làm việc, được tạo ngẫu nhiên trong khoảng từ 16h-22h30.

@i: Biến đếm, bắt đầu từ 1 và tăng dần đến 1000 (tạo 1000 bản ghi).

**+ Bước 2:** Vòng lặp While

Vòng lặp chạy 1000 lần thực hiện các công việc :

* + Tạo ngày chấm công ngẫu nhiên trong 30 ngày gần nhất.
  + Tạo giờ vào ngẫu nhiên bằng hàm NEWID() dựa trên hai ca làm việc:
    - Ca chiều (13h30-14h)
    - Ca tối (18h-18h30)
  + Tạo giờ ra ngẫu nhiên từ 16h-22h30.
  + Chèn dữ liệu chấm công vào bảng CHAMCONG:
    - MaCC: Mã chấm công (định dạng 'CC' + định dạng MaNV, ví dụ: 'CC0001', 'CC0002').
    - MaNV: Mã nhân viên tương ứng trong bảng NHANVIEN (định dạng '0001', '0002').
    - Ngay: Ngày làm việc đã tạo ngẫu nhiên.
    - GioVao: Giờ vào làm việc đã tạo ngẫu nhiên.
    - GioRa: Giờ ra làm việc đã tạo ngẫu nhiên.

1. Tạo dữ liệu dump cho bảng TAIKHOAN

* **Mục đích:** Thủ tục ThemTaiKhoan được sử dụng để tạo ngẫu nhiên 1000 tài khoản cho nhân viên trong bảng TAIKHOAN. Mỗi tài khoản bao gồm mã tài khoản, tên đăng nhập, mật khẩu và mã nhân viên liên kết.
* **Input:** N/A
* **Output:** 1000 dòng dữ liệu được thêm vào bảng TAIKHOAN
* **Process:**

+ **Bước 1:** Khởi tạo biến

@MaNV: Lưu trữ mã nhân viên được chọn ngẫu nhiên từ bảng NHANVIEN.

@i: Biến đếm, bắt đầu từ 1 và tăng dần đến 1000 (tạo 1000 tài khoản).

**+ Bước 2:** Vòng lặp While

Vòng lặp chạy 1000 lần thực hiện các công việc :

* + Chọn ngẫu nhiên mã nhân viên từ bảng NHANVIEN bằng câu lệnh ORDER BY NEWID().
  + Tạo mã tài khoản, tên đăng nhập và mật khẩu theo định dạng cố định.
  + Chèn thông tin tài khoản vào bảng TAIKHOAN:
    - MaTK: Mã tài khoản được tạo dựa trên biến đếm @i, ví dụ: 'TK0001', 'TK0002',...
    - TenDangNhap: Tên đăng nhập có định dạng '2200001', '2200002',... dựa trên giá trị của @i.
    - MatKhau: Mật khẩu có định dạng 'password1', 'password2',... dựa trên giá trị của @i.
    - MaNV: Mã nhân viên được chọn ngẫu nhiên ở bước trước đó.

### Module xử lý dữ liệu

|  |  |
| --- | --- |
| **Tên module** | **Cách làm** |
| 1. Thủ tục in ra danh sách nhân viên chưa chấm công trong một ngày  * Mục đích: Giúp theo dõi những nhân viên chưa thực hiện chấm công vào ngày được chỉ định, đồng thời xuất kết quả danh sách những nhân viên này kèm mã nhân viên của họ. | * Input: @Ngay * Output: Danh sách nhân viên chưa chấm công cho ngày được chỉ định * Process:  1. Lấy ra tất cả HoTen và MaNV trong bảng NHANVIEN left join bảng CHAMCONG với phép nối bảng là:  * Ngay=@Ngay * NHANVIEN.MaNV=CHAMCONG.MaNV  1. Và điều kiện: WHERE MaCC IS NULL để lọc ra những nhân viên không có bản ghi chấm công trong ngày được chỉ định. 2. Xuất danh sách nhân viên chưa chấm công và (print) in ra kết quả: danh sách nhân viên chưa chấm công cho ngày được cung cấp (@Ngay). |
| 1. Thủ tục cập nhật chức vụ cho nhân viên  * Mục đích: Cập nhật chức vụ của một nhân viên trong bảng NHANVIEN | * Input: @MaNV, @MaCV * Output: N/A * Process:  1. Kiểm tra xem MaNV có tồn tại trong bảng NHANVIEN hay không. Nếu không tìm thấy nhân viên có mã đó → thông báo ‘Nhân viên không tồn tại’ và ngừng thủ tục. 2. Nếu MaNV tồn tại, tiến hành cập nhật bảng NHANVIEN có giá trị MaCV=@MaCV với điều kiện MaNV=@MaNV → In ra thông báo ‘Cập nhật chức vụ thành công’ |
| 3. Thay đổi mật khẩu cho nhân viên  - Mục đích: Cung cấp chức năng bảo mật cho người dùng để thay đổi mật khẩu của mình khi cần (dựa trên tính hợp lệ của mật khẩu hiện tại). | * Input: @TenDangNhap, @MatKhauCu, @MatKhauMoi * Output: In ra kết quả (PRINT):   + Nếu mật khẩu cũ đúng: "Mật khẩu đã được thay đổi."  + Nếu mật khẩu cũ không đúng: "Mật khẩu cũ không chính xác."   * Process:  1. Kiểm tra tên đăng nhập và mật khẩu    1. kiểm tra xem tài khoản với TenDangNhap và MatKhauCu có tồn tại trong bảng TAIKHOAN hay không với điều kiện:   → TenDangNhap = @TenDangNhap  → MatKhau = @MatKhauCu.   * 1. Nếu tài khoản tồn tại và mật khẩu cũ khớp, tiếp hành cập nhật bảng TAIKHOAN có giá trị Matkhau= @Matkhaumoi và điều kiện TenDangNhap= @TenDangNhap   → In thông báo ‘Mật khẩu đã được thay đổi’   * 1. Nếu không tìm thấy tài khoản với mật khẩu cũ khớp → in ra 'Mật khẩu cũ không chính xác.' |
| 4. Tính tổng lương cho nhân viên dựa trên tổng giờ làm.  - Mục đích: Trigger này tự động cập nhật tổng lương (Tongluong) của nhân viên mỗi khi có một bản ghi mới được thêm vào bảng CHAMCONG. Mỗi lần nhân viên chấm công, trigger sẽ tính toán tổng số giờ làm việc và cập nhật tổng lương (Tongluong) dựa trên số giờ làm việc, lương theo giờ, các khoản thưởng và phạt | * Bảng kí sinh: CHAMCONG * Loại: after * Sự kiện: insert * Process:   + - 1. Lấy @MaNV từ bản ghi mới được thêm vào bảng CHAMCONG thông qua bảng inserted       2. Tính tổng giờ làm việc cho nhân viên (Tổng số giờ làm + tổng số phút lẻ còn lại (đổi sang đơn vị giờ) ).       3. Lấy thông tin @luong= LuongTheoGio, @thuong= thuong và @phat = phat từ bảng LUONG   Điều kiện: WHERE MaNV = @MaNV   * + - 1. Set @thuong= @thuong/100 và @phat = @phat/100 để chuyển từ phần trăm thành số thập phân.       2. Cập nhật tổng lương của nhân viên dựa trên giờ làm việc, lương theo giờ, thưởng và phạt.   Điều kiện: WHERE MaNV = @MaNV |

|  |  |
| --- | --- |
| 5.Kiểm tra nhân viên có phần trăm lỗi > 20% khi biết mã nhân viên. Nếu có thì in ra tên nhân viên và đưa ra thông báo 'Cảnh báo đuổi việc'. Ngược lại thì đưa ra thông báo 'Không bị cảnh báo đuổi việc'. | * Input: @manv * Output: @ten, @nx   + Nếu nhân viên có phần trăm lỗi lớn hơn 20% thì in tên của nhân viên và thông báo 'Cảnh báo đuổi việc'   + Nếu không, thì in tên của nhân viên và thông báo 'Không bị cảnh báo đuổi việc' * Process:  1. Nhận tham số đầu vào: @manv (mã nhân viên) 2. Lấy tên nhân viên (HoTen) từ bảng NHANVIEN dựa trên @manv    * + Kết quả HoTen được lưu vào biến đầu ra: @ten. 3. Kiểm tra phần trăm lỗi:    * + Thực hiện truy vấn để kiểm tra giá trị Phat từ bảng LUONG, dựa vào mã nhân viên.      + Sử dụng INNER JOIN giữa bảng LUONG và NHANVIEN để tìm phần trăm lỗi của nhân viên 4. Đưa ra cảnh báo:    * + Nếu phần trăm lỗi (Phat) lớn hơn 20%, set @nx = N'Cảnh báo đuổi việc'.      + Nếu không, set @nx thành N'Không bị cảnh báo đuổi việc'. |
| 6. Trả về các ca làm bất thường (ca làm bất thường: giờ vào - giờ ra <=3),  nếu không có ca nào thì in ra thông báo 'không có ca làm bất thường' | * Input: N/A * Output:   + Nếu có các ca làm bất thường,trả kết quả gồm: MaNV, Ngay, GioVao, GioRa   + Nếu không có ca làm bất thường, in ra thông báo   'Không có ca làm bất thường'   * Process:  1. Khai báo biến: @MaNV, @Ngay, @GioVao, @GioRa 2. Kiểm tra có ca làm bất thường hay không    1. Đếm số ca làm trong bảng CHAMCONG với điều kiện thời gian làm việc <=3 tiếng   → datediff(hour, @Giovao, @Giora) <=3   * 1. Nếu tồn tại trên > 1 ca làm bất thường thì trả về thông tin gồm: Mã nhân viên, Ngày, Giờ Vào, Giờ Ra   2. Nếu không có ca làm bất thường nào → in ra thông báo "Không có ca làm bất thường". |
| 7. Khi xóa những tài khoản không đăng nhập liên tục trong 2 tuần thì  thực hiện thao tác: Cập nhật tài khoản là QUIT, thay vì xóa. | * Bảng: TAIKHOAN * Loại: Instead of * Sự kiện: Delete * Process:  1. Khai báo @MaNV, @dangnhap 2. Lấy ra MaNV trong bảng inserted → @MaNV 3. Lấy ra ngày đăng nhập gần nhất (Max(Ngay)) trong bảng CHAMCONG với điều kiện MaNV=@MaNV 4. Kiểm tra thời gian đăng nhập kể từ lần đăng nhập cuối cùng.    1. Nếu **khoảng thời gian ≥ 14 ngày**, nghĩa là nhân viên không đăng nhập trong 2 tuần, thì sẽ cập nhật bảng TAIKHOAN có giá trị TenDangnhap=’QUIT’ với điều kiện MaNV=@MaNV    2. Ngược lại, in thông báo 'Vẫn đăng nhập đều' |
| 8. Cập nhật giờ ra cho nhân viên trong chấm công | * Input: @MaNV, @Ngay, @GioRa * Output:   + Nếu nhân viên đã có dữ liệu chấm công cho ngày đã làm việc thì in thông báo ‘Cập nhật giờ ra thành công’   + Nếu nhân viên không có dữ liệu chấm công cho ngày đã làm việc thì in thông báo ‘Không tìm thấy dữ liệu chấm công cho nhân viên trong ngày làm việc này * Process:  1. Nhận 3 tham số đầu vào: @MaNV, @Ngay, @GioRa 2. Kiểm tra xem có bản ghi nào trong bảng CHAMCONG với mã nhân viên và ngày được cung cấp hay không.  * Nếu bản ghi tồn tại, cập nhật giờ ra: thực hiện câu lệnh UPDATE để cập nhật giờ ra cho nhân viên, đưa ra thông báo 'Cập nhật giờ ra thành công'. * Nếu không tìm thấy bản ghi, in thông báo 'Không tìm thấy dữ liệu chấm công cho nhân viên trong ngày này.' |
| 9. Thêm nhân viên mới với kiểm tra trùng mã nhân viên.  - Mục đích: Dùng để thêm mới một nhân viên vào bảng NHANVIEN. Trước khi thêm, thủ tục kiểm tra xem mã nhân viên đã tồn tại hay chưa để tránh trường hợp thêm trùng mã. | * Input: @MaNV, @HoTen, @NgaySinh, @GioiTinh, @DiaChi, @SDT, @MaCV * Output: N/A * Process:  1. Nhận tham số các tham số đầu vào: @MaNV, @HoTen, @NgaySinh, @GioiTinh, @DiaChi, @SDT, @MaCV 2. Kiểm tra xem mã nhân viên đã tồn tại trong bảng NHANVIEN hay chưa.  * Nếu mã nhân viên đã tồn tại, in ra thông báo 'Mã nhân viên đã tồn tại'. * Nếu không tồn tại, thực hiện câu lệnh INSERT để thêm nhân viên mới vào bảng NHANVIEN và in ra thông báo 'Nhân viên đã được thêm thành công |
| 10. Thêm xử lý khi nhân viên đi làm muộn, về sớm ca chiều (13h30 – 14h). | - Input: @MaNV, @Ngay, @GioRa, @Gi,oVao  - Output: N/A  - Process:   * Khai báo các biến giờ vào và giờ ra chuẩn   cho hai ca làm việc:  + Ca 1: Giờ vào: 14h - Giờ ra: 17h30  + Ca 2: Giờ vào: 18h30h - Giờ ra: 22h30   * Kiểm tra giờ làm việc:   + Nếu giờ vào thuộc ca 1, kiểm tra xem nhân viên có đi làm muộn hoặc về sớm hay không. Nếu có, cộng mức phạt vào biến @Phat  + Nếu giờ vào thuộc ca 2, làm tương tự như ca 1.  + Nếu giờ vào không thuộc ca nào, in ra thông báo rằng ‘Giờ làm việc không thuộc ca làm việc quy định’ và dừng thực hiện   * Cập Nhật Mức Phạt:   + UPDATE để cập nhật giá trị mức phạt:  set Phat = Phat + @Phat vào bảng LUONG cho nhân viên với MaNV = @MaNV  + In thông báo kết quả: Thông báo về việc chấm công thành công cùng với mức phạt |

### Thay đổi trong thiết kế

* + Trong bảng **LUONG**, bổ sung thêm cột ‘Tongluong’ với kiểu dữ liệu Float để lưu trữ tổng lương của mỗi nhân viên sau khi tính toán từ giờ làm, lương theo giờ, thưởng và phạt.
  + Cơ sở dữ liệu sau khi thay đổi:

NHANVIEN (**MaNV**, HoTen, NgaySinh, SDT, GioiTinh, DiaChi, *MaCV*)

CHUCVU (**MaCV**, TenCV)

TAIKHOAN (**MaTK**, TenDangNhap, MatKhau, *MaNV*)

CHAMCONG (**MaCC**, Ngay, GioVao, GioRa, *MaNV*)

LUONG (**MaLuong**, LuongTheoGio, Thuong, Phat, Tonglunog *MaNV)*

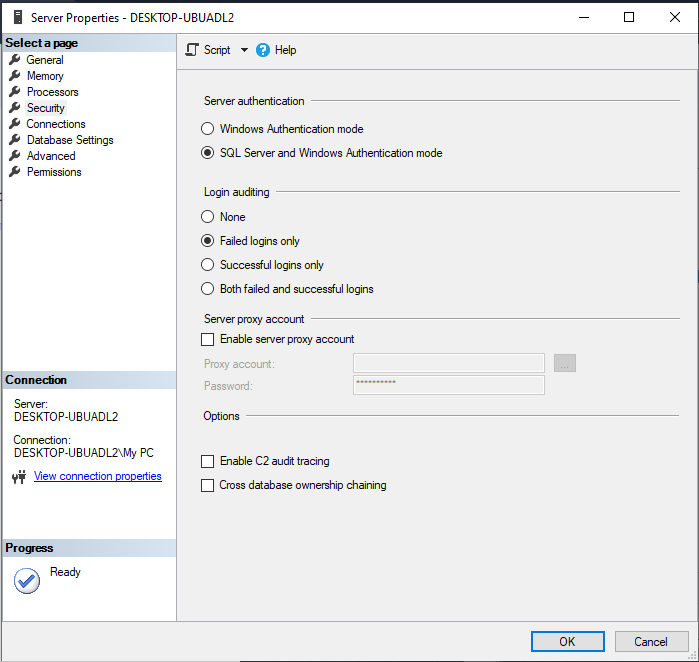
* + Database Diagrams:

A screenshot of a computer

Description automatically generated

## **Xây dựng cơ chế bảo mật phù hợp cho cơ sở dữ liệu**

### Thiết lập cơ chế authentication

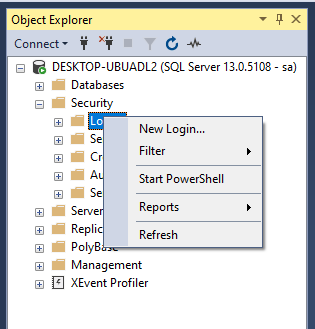


Hình 1: Thiết lập cơ chế authentication

### Tạo và phân quyền cho người dùng

**a. Tạo tài khoản cho người dùng**

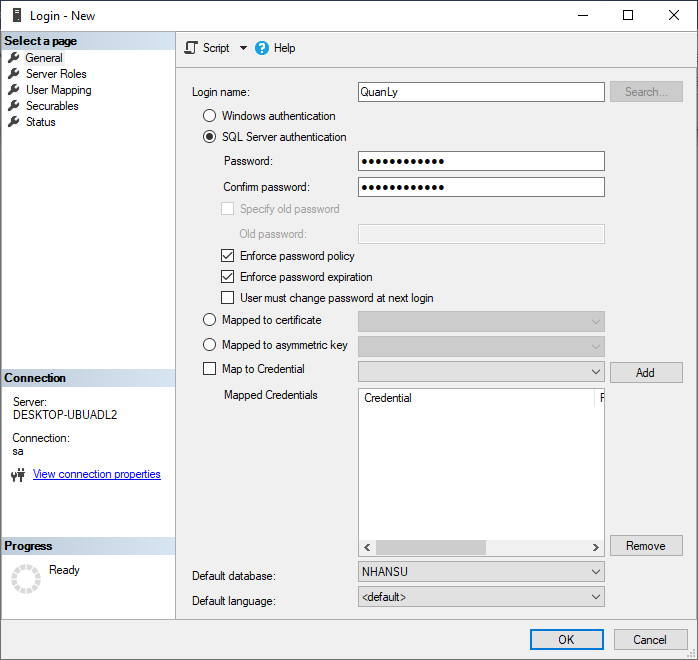
1. Mở **SQL Server Management Studio** (SSMS) và kết nối với server quản trị máy (sa)
2. Mở rộng mục **Security** và click chuột phải vào **Logins**, sau đó chọn **New Login** để tạo một tài khoản Quản Lý



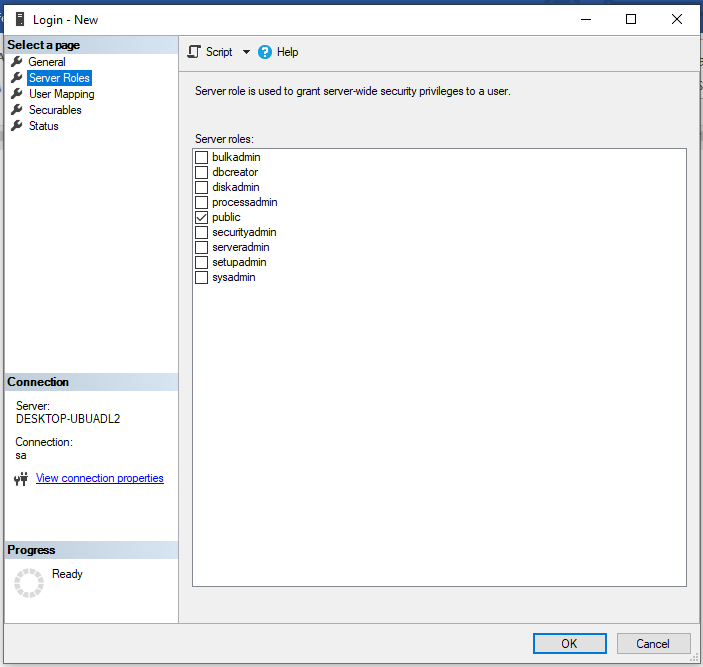
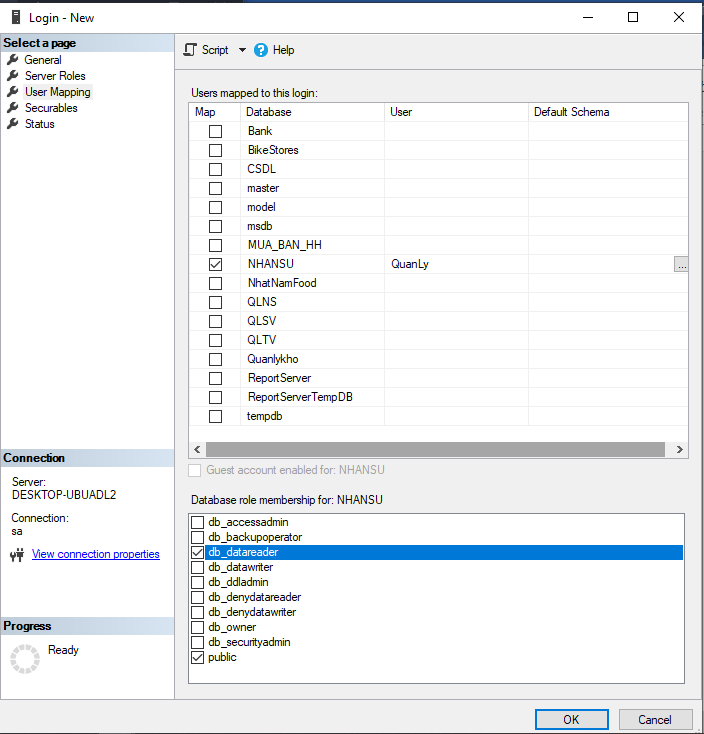
Hình 2.Tạo login cho Quản Lý

1. Trong cửa sổ **Login - New**, chọn **SQL Server Authentication,** nhập tên đăng nhập của người dùng trong ô **Login name** và mật khẩu

Với: user: QuanLy - password: quanlynhansu

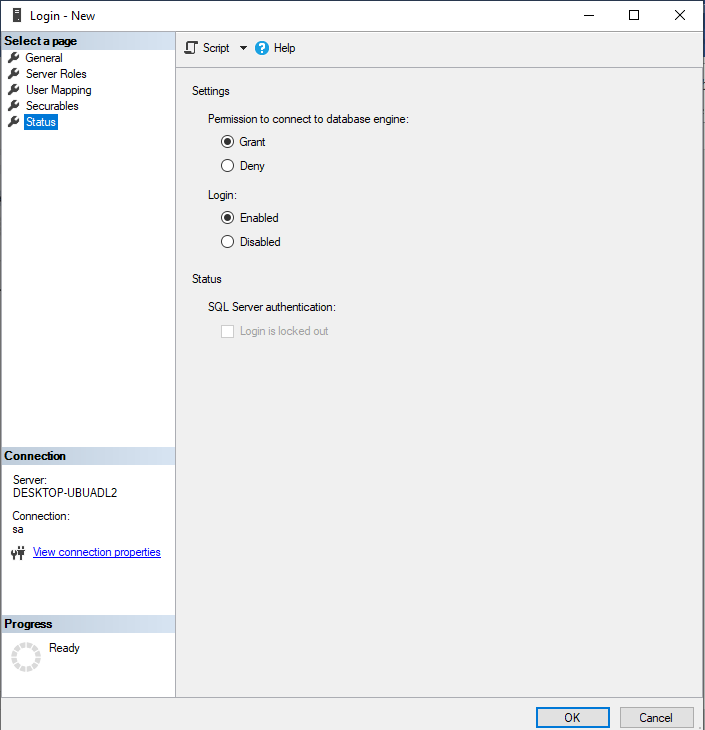
- Set chế default database là **NHANSU** và bỏ chọn ô **User must change password at next login** 

Hình 3: Nhập tên đăng nhập và mật khẩu quản lý

1. Trong chế độ **Server Roles,** set chế độ **public** 
2. Trong chế độ **User Mapping**, chọn **NHANSU** để hướng người dùng **Quản lý** chỉ sử dụng vào database **NHANSU**, đồng thời chọn **db\_datareader** để người dùng Quản lý có thể đọc các dữ liệu trong database.

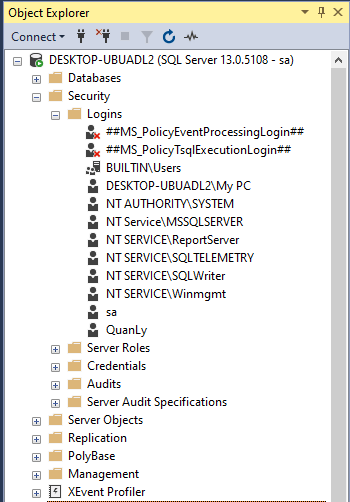
Hình 4: Thiết lập người dùng chỉ sử dụng database NHANSU

1. Trong chế độ **Status**, thiết lập chế độ **Grant** trong **Setting** và thiết lập **Login** trong chế độ bật chế độ **Enabled**

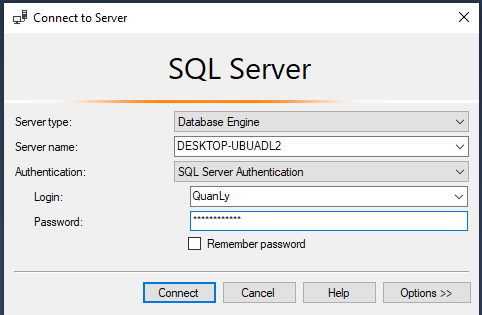


Hình 5: Thiết lập chế độ status cho người dùng

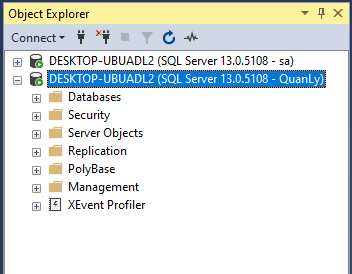
1. Nhấn chọn **OK** - Hoàn thành việc tạo login cho người Quản Lý



1. Đăng nhập SQL Authentication với user QuanLy



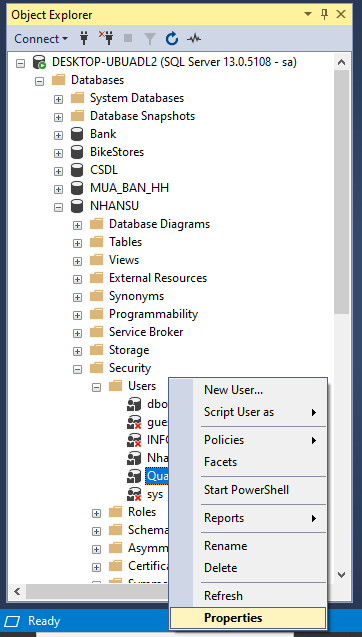
Hình 6 : Màn hình đăng nhập cho người dùng Quản Lý



Hình 7: Màn hình tạo người dùng Quản lý thành công

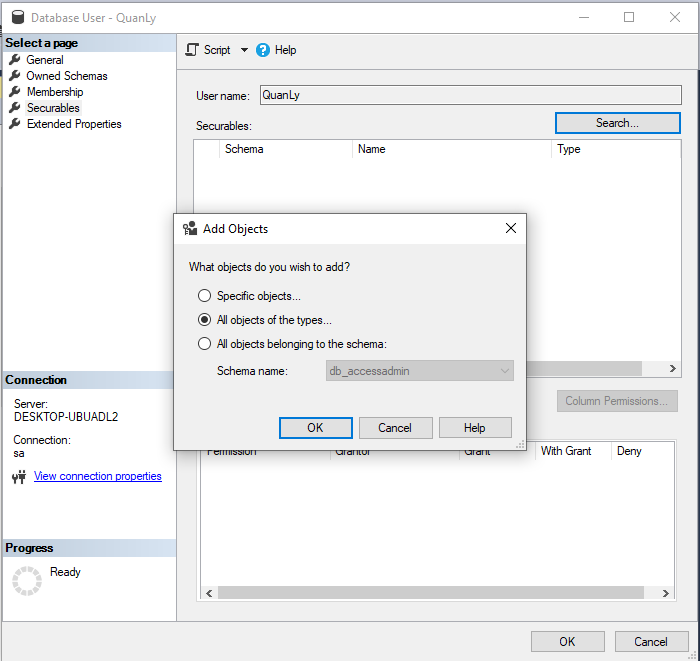
**b. Phân quyền cho người dùng Quản lý**

1. Chọn **Properties** cho user **Quan Ly**

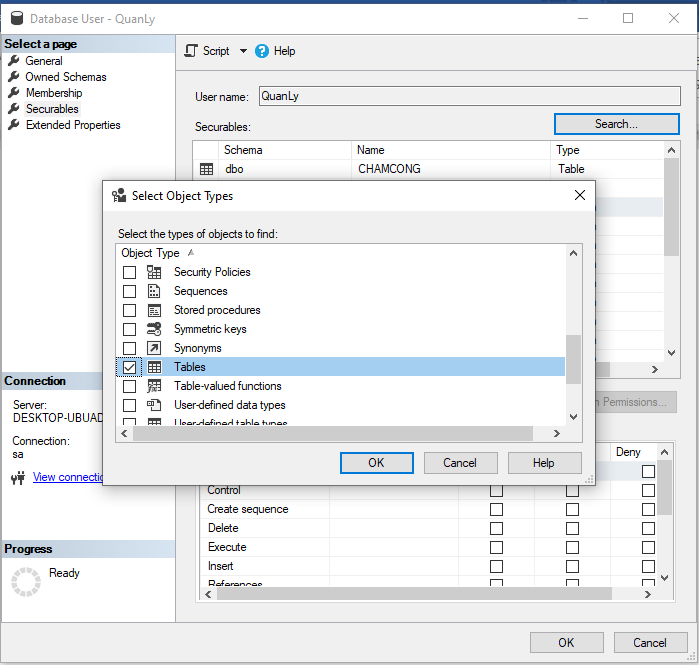


Hình 8: Thiết lập quyền cho Quản Lý

2. Trong cửa sổ **Database User – QuanLy**, chọn **Search**

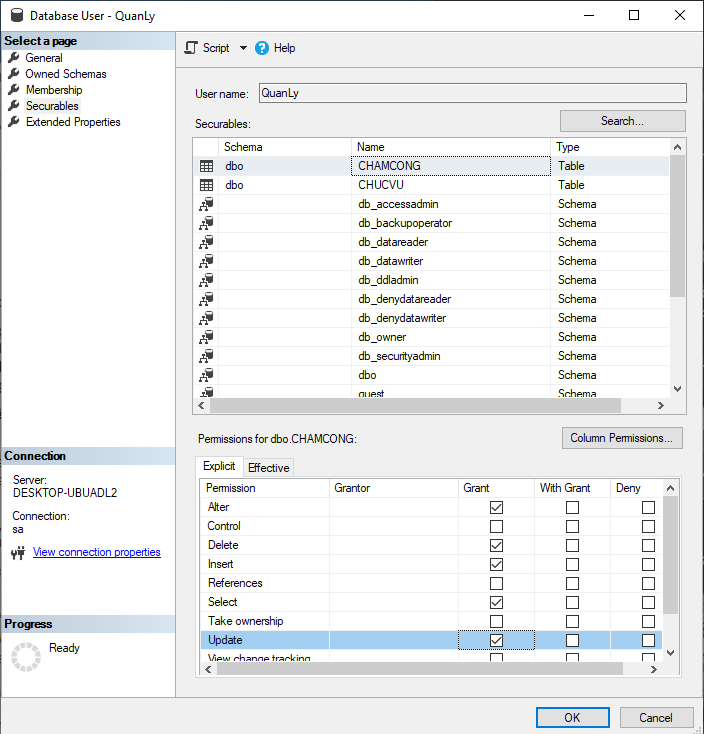


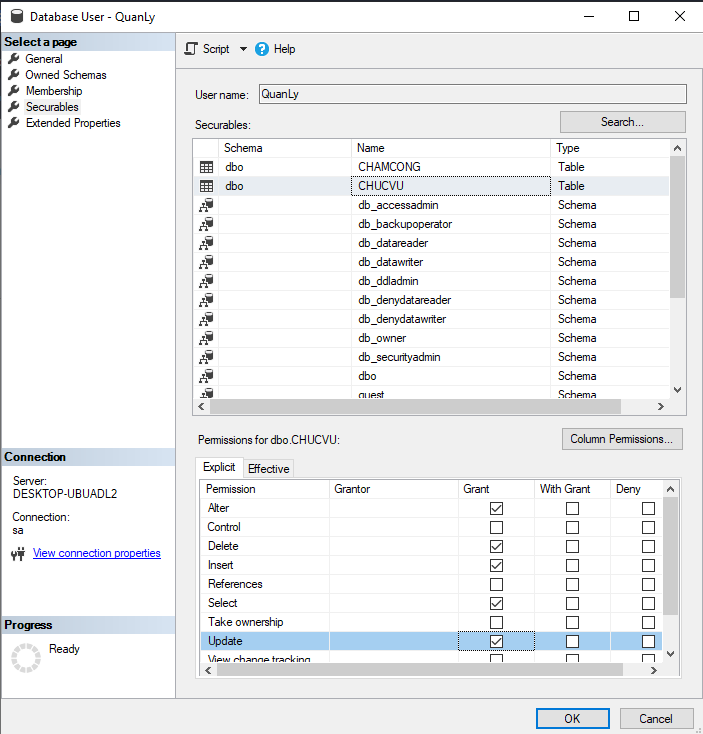
3. Chọn **All objects of the types - chọn tables - OK**

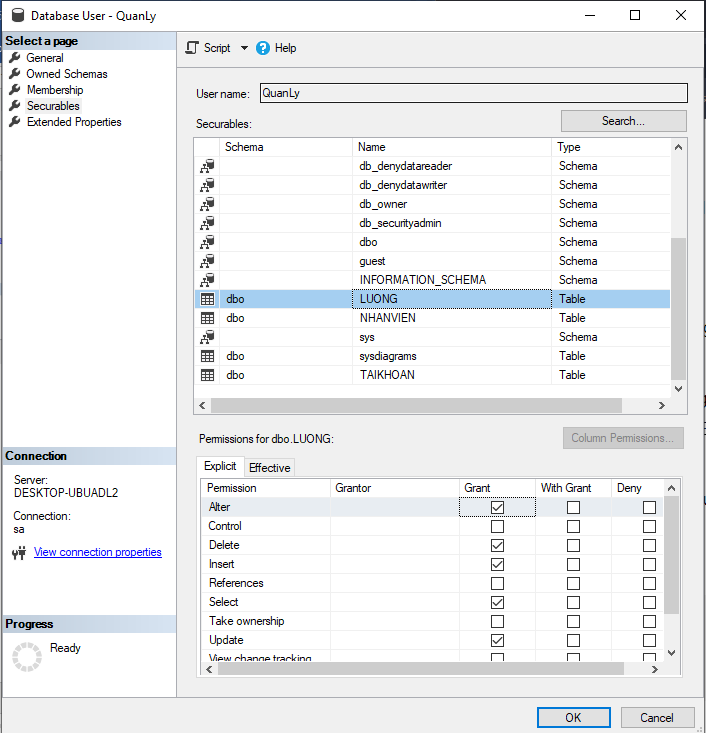


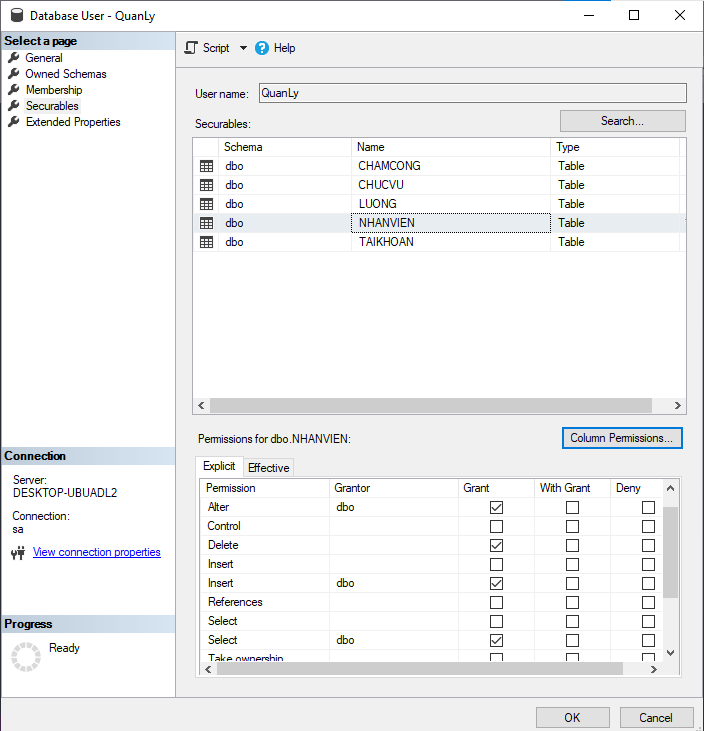
4. Thiết lập quyền cho **Quản Lý:**

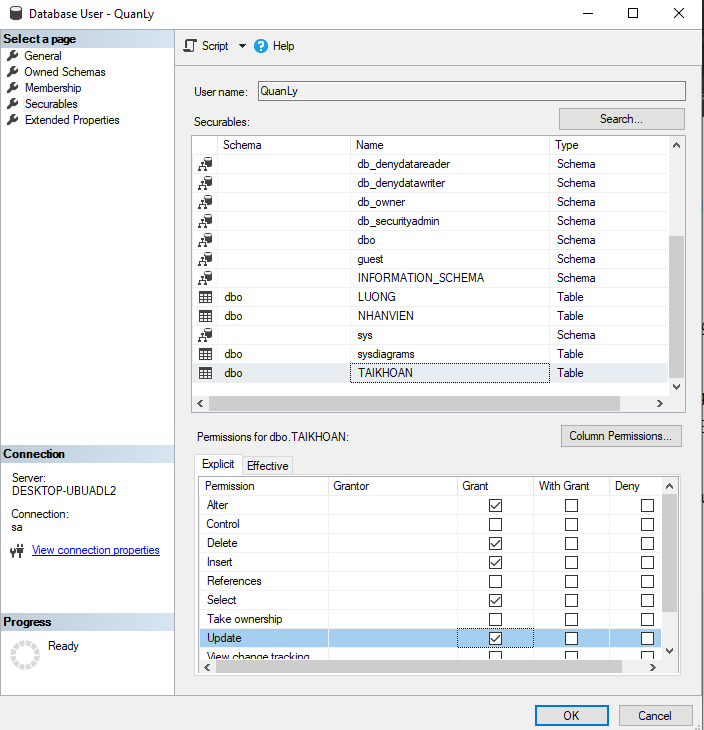
* Xem và truy xuất thông tin ở tất cả các bảng.
* **Thêm, sửa, xóa** và cập nhật dữ liệu thông tin trong các bảng:
  + Chức Vụ (CHUCVU)
  + Tài Khoản (TAIKHOAN)
  + Lương (LUONG)
  + Nhân Viên (NHANVIEN).







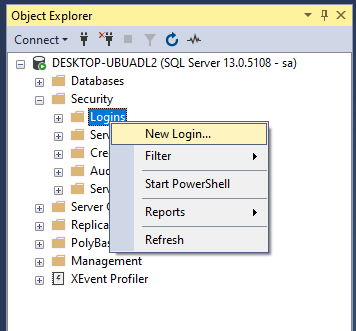


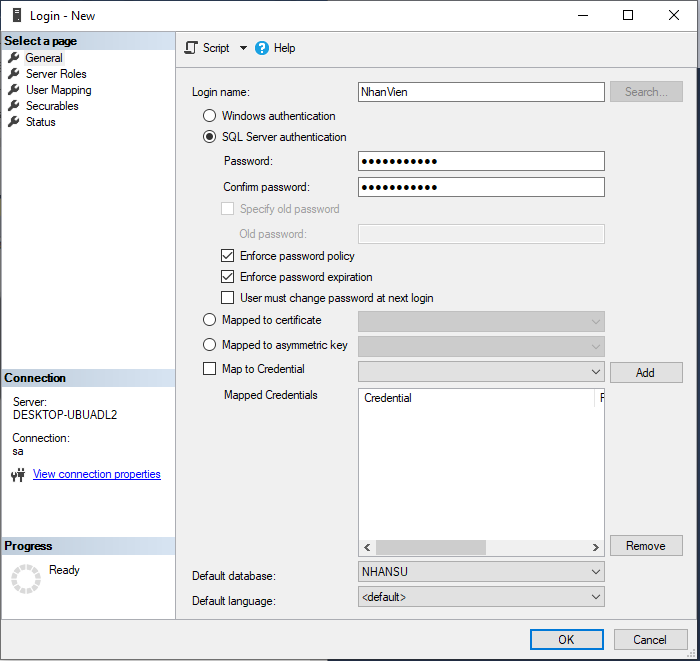


**5.** Nhấn **OK** - Hoàn thành thiết lập phân quyền cho người dùng **QuanLy**

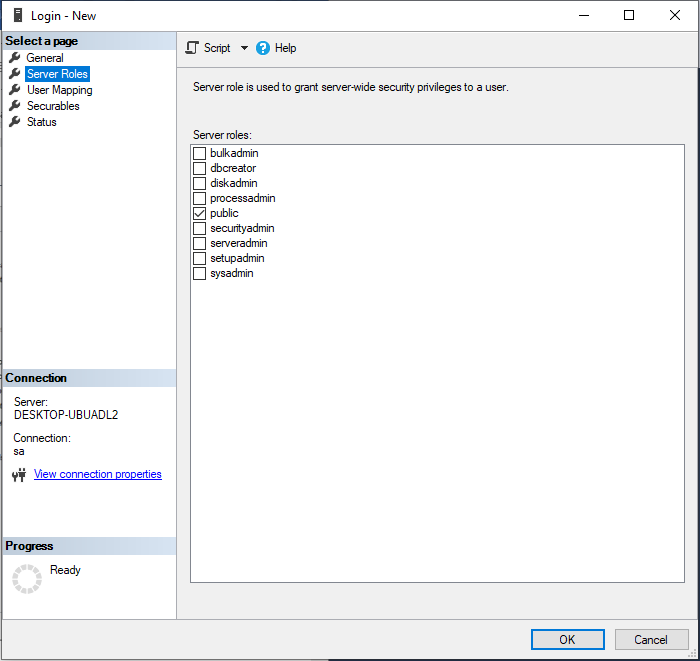
**c. Tạo người dùng cho Nhân viên (NhanVien)**

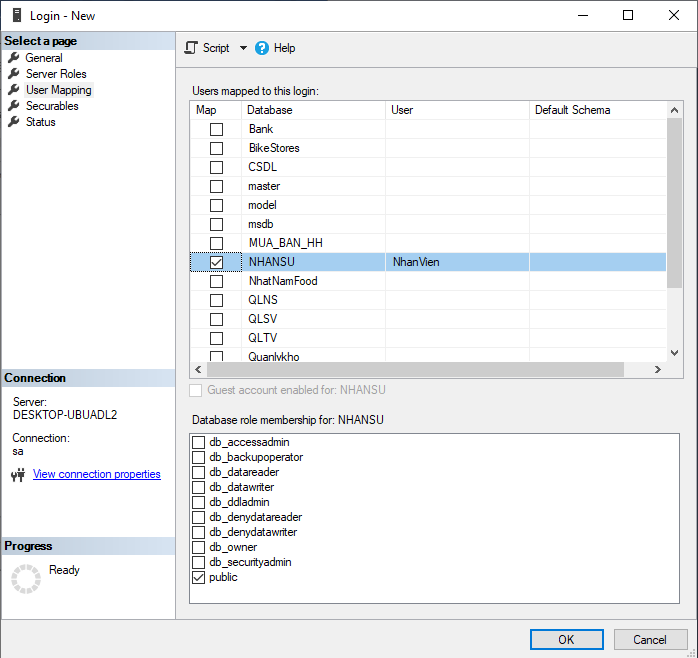
Lặp lại tương tự các bước tạo tài khoản như ở tài khoản **QuanLy**

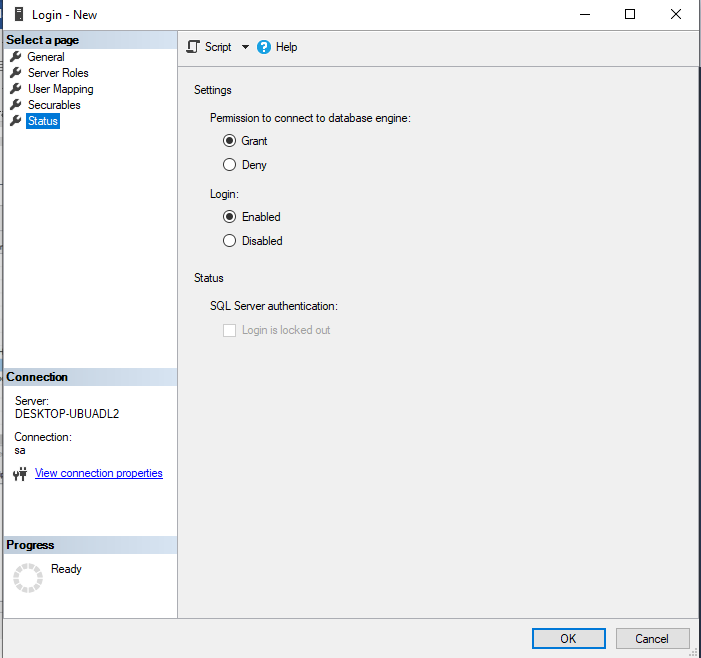


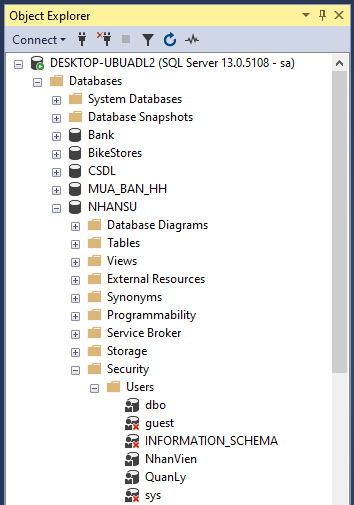


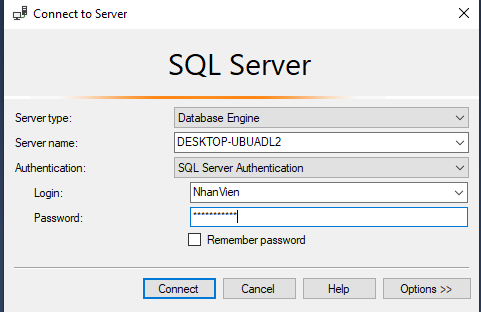
Với: user: NhanVien - password: nhanvien123

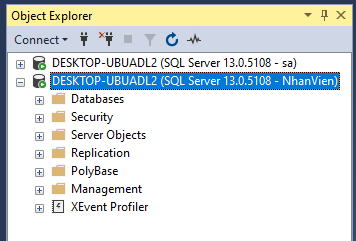


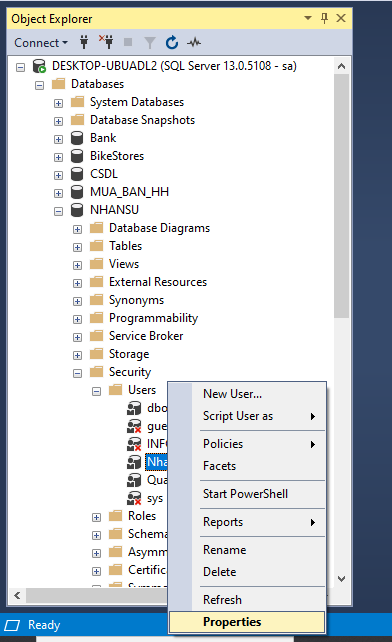


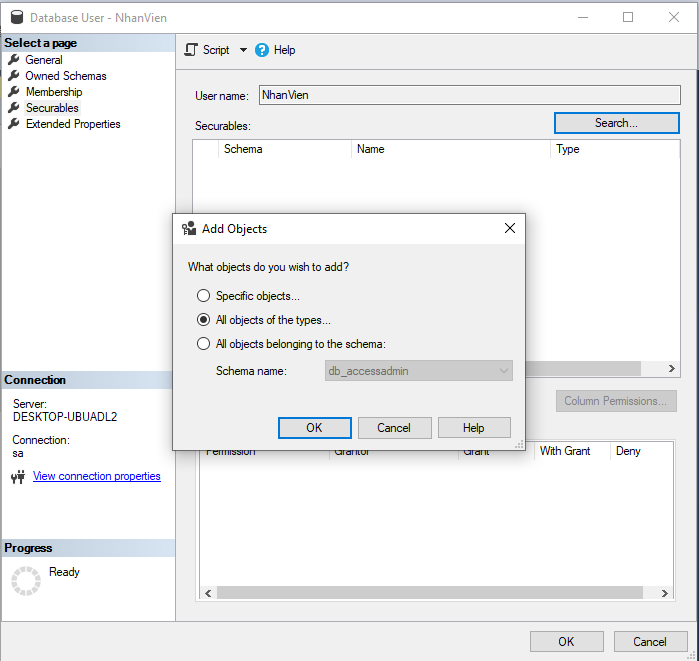


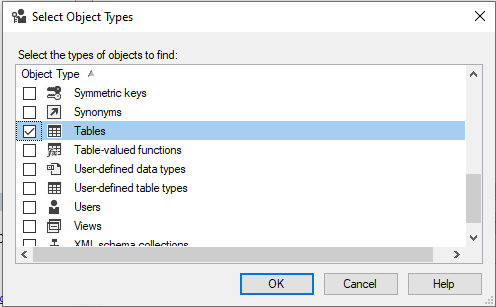






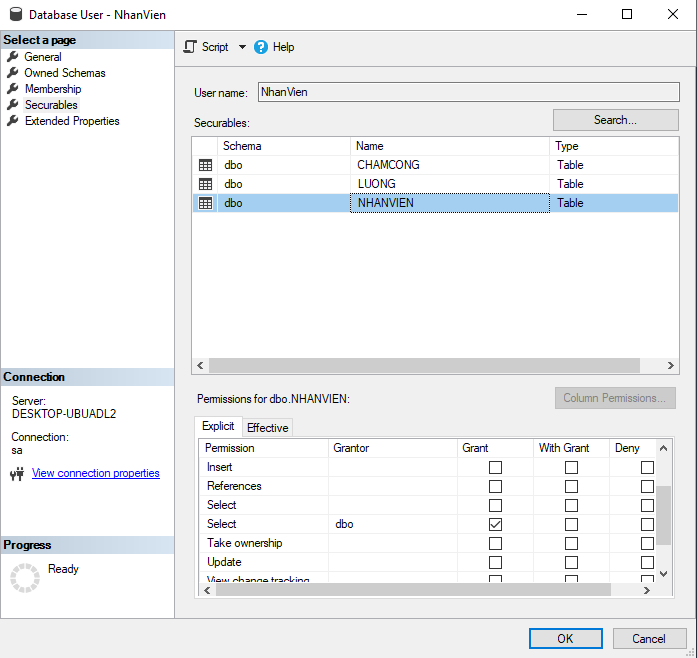


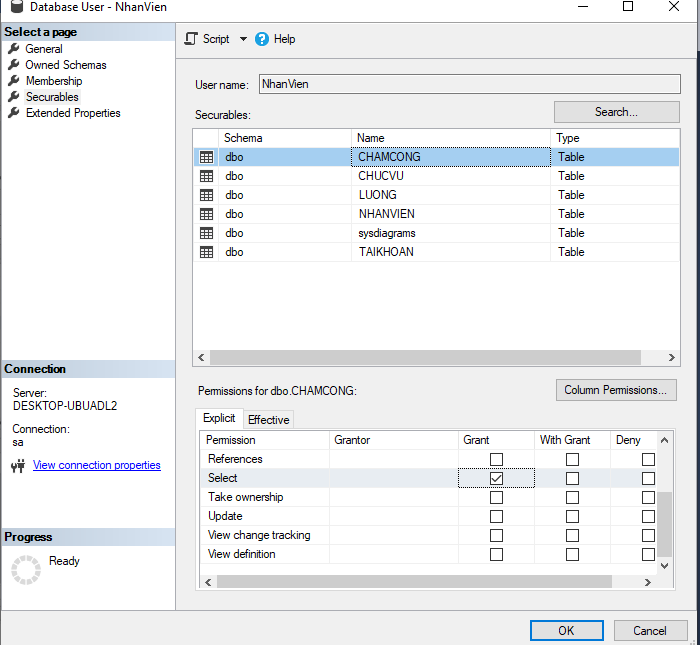


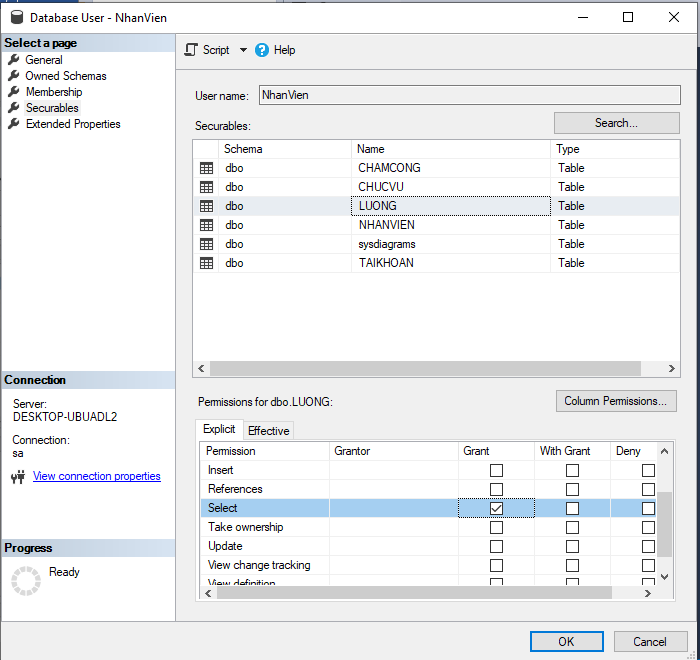


Thiết lập quyền cho **Nhân Viên:**

* **Truy xuất** thông tin trong các bảng:
  + Chấm Công (CHAMCONG)
  + Lương (LUONG)
  + Nhân Viên (NHANVIEN)

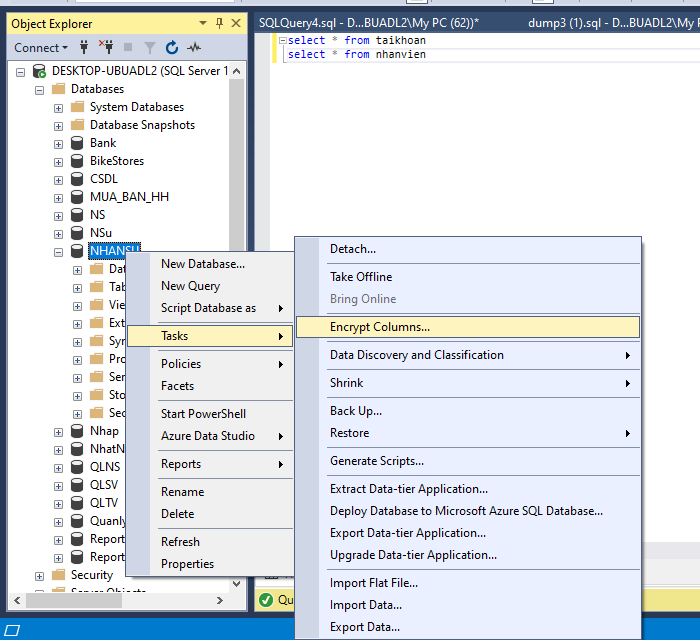


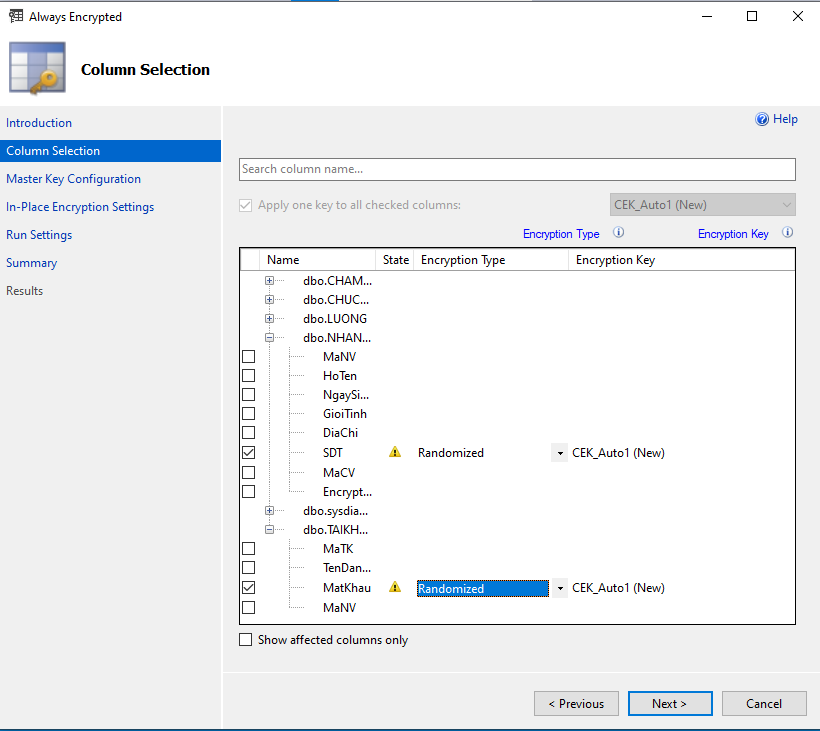


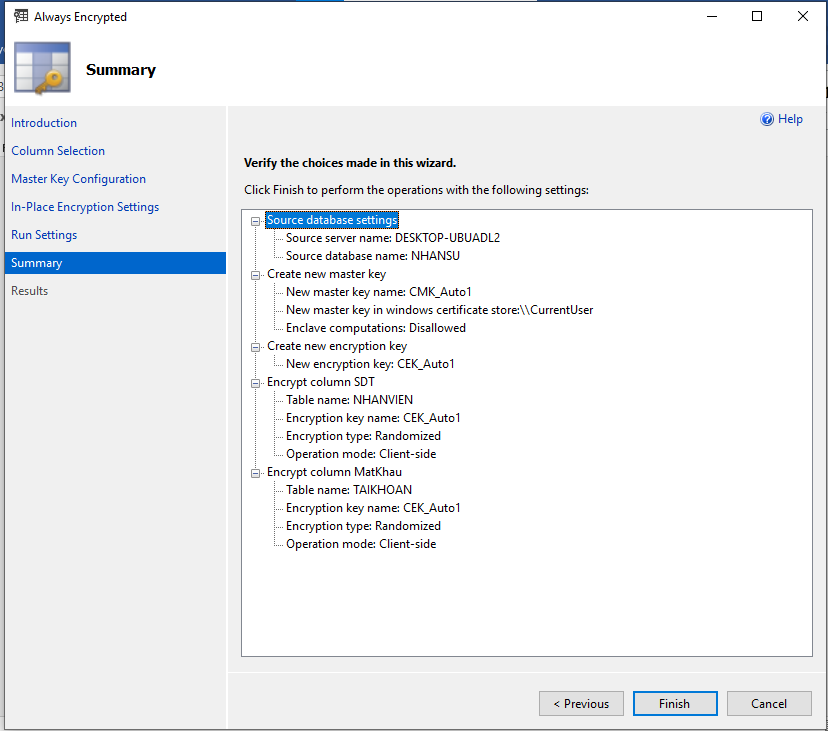


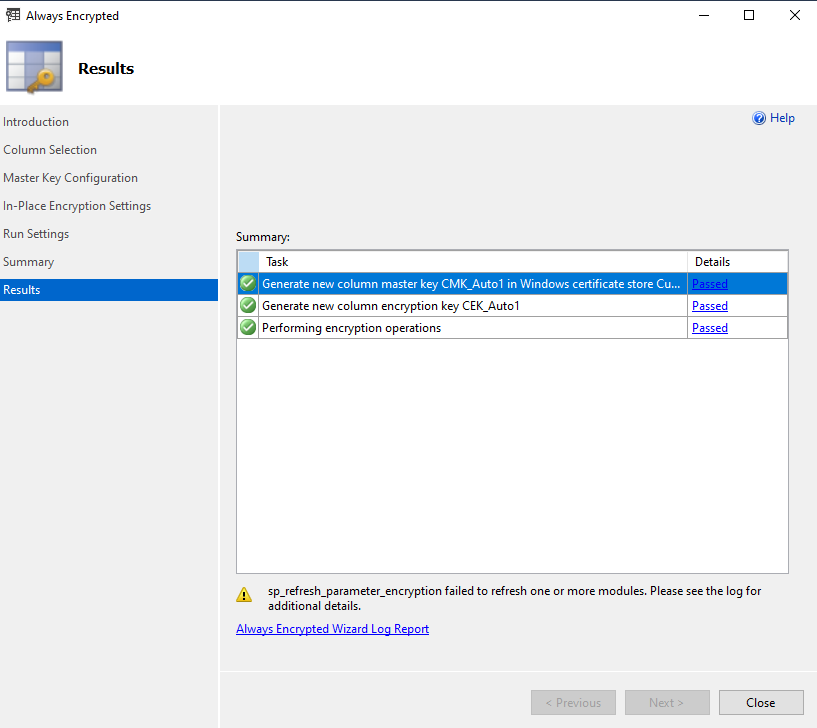
Nhấn **OK -** Hoàn thành thiết lập phân quyền cho ngườidùng **NhanVien**

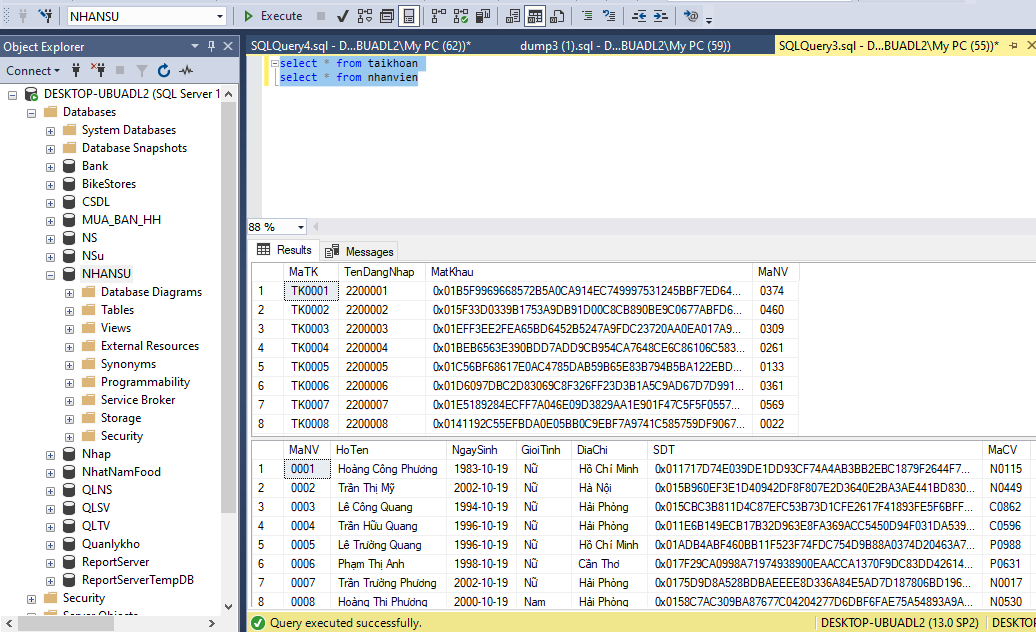
### Mã hóa các dữ liệu nhạy cảm





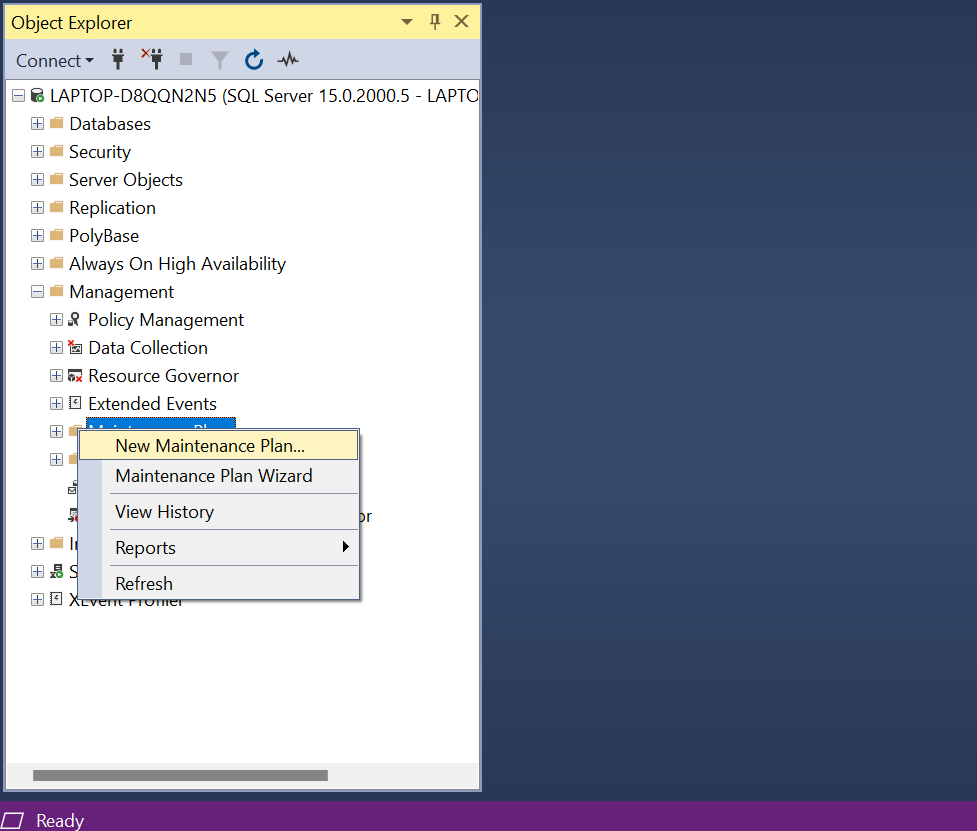


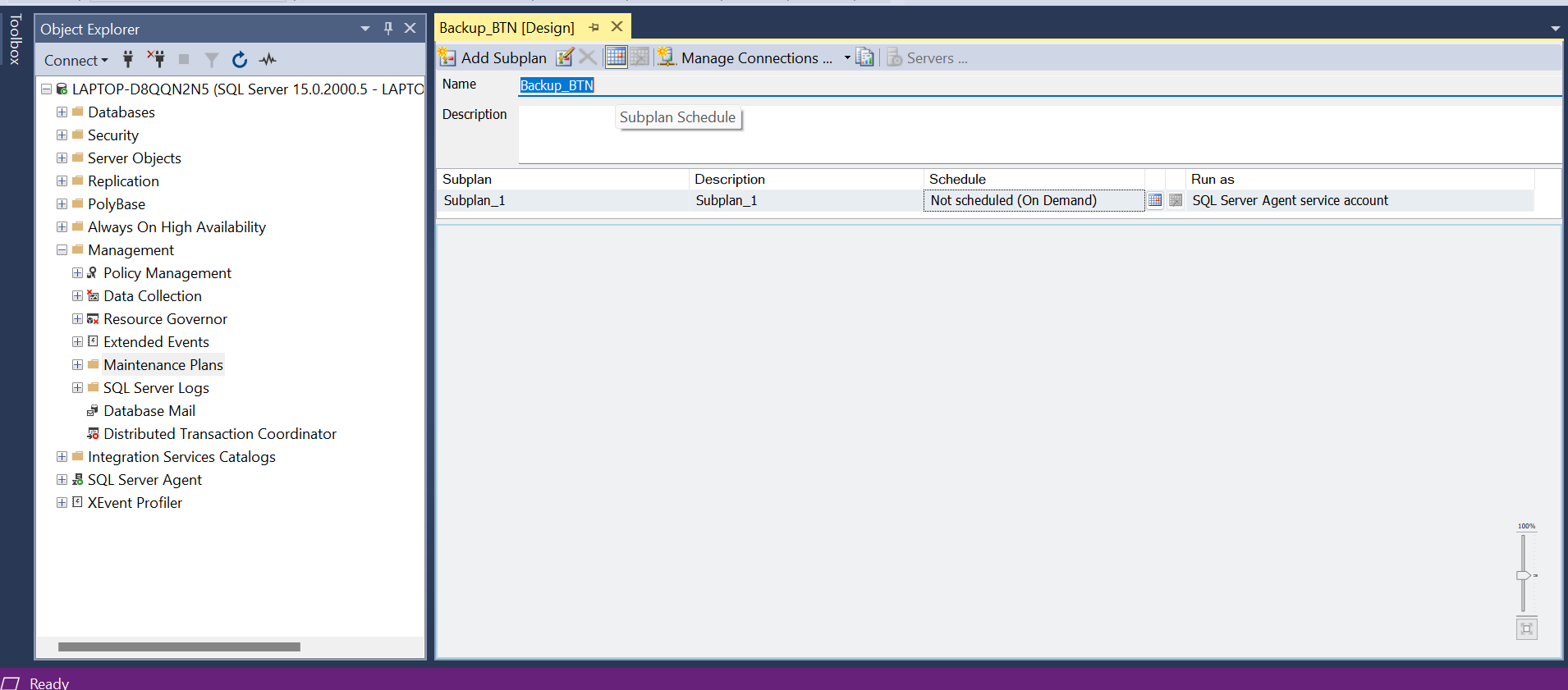




## **Xây dựng cơ chế backup dữ liệu tự động cho cơ sở dữ liệu**

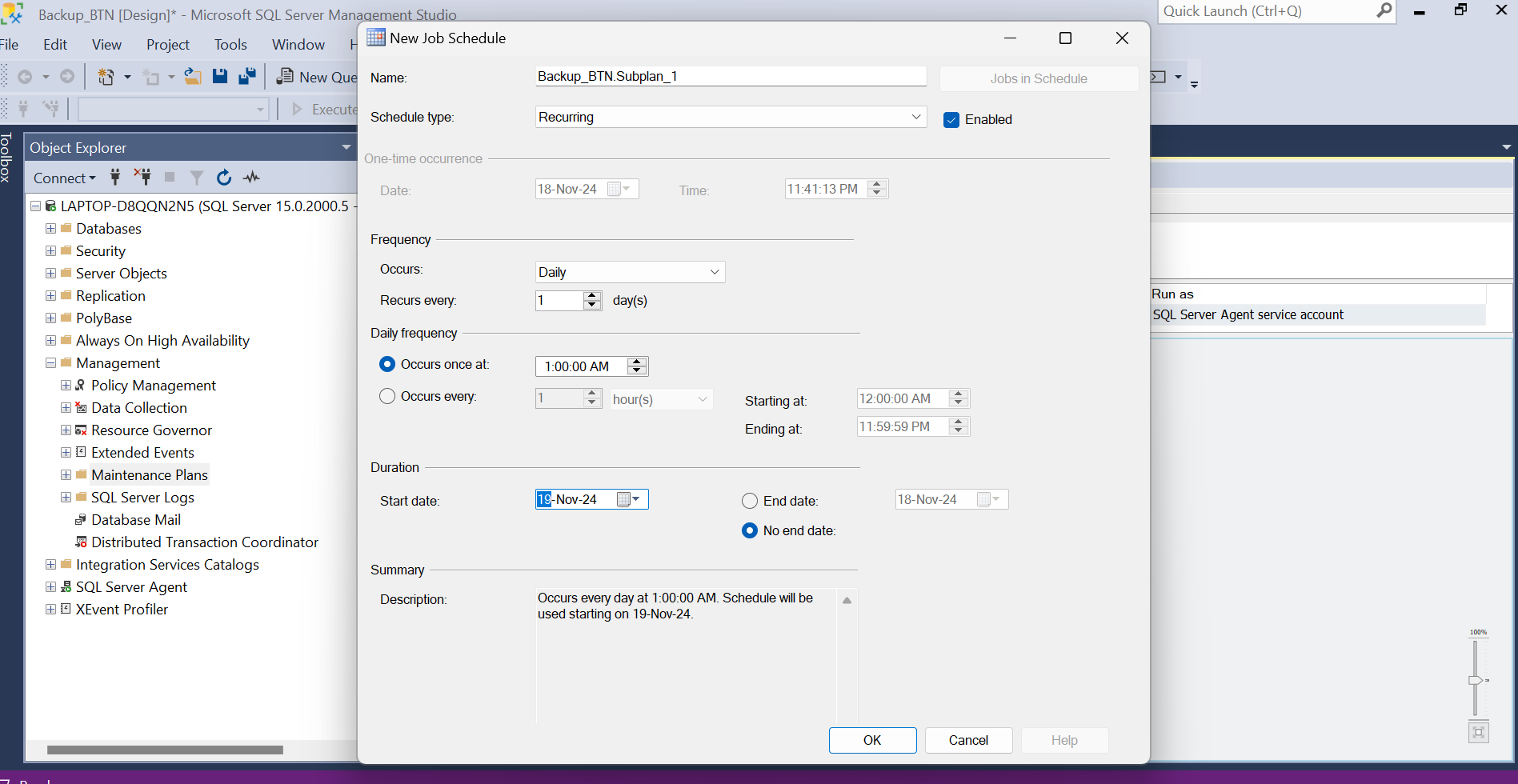
Bước 1: Tại mục Management, tạo bản Maintenance Plan mới.





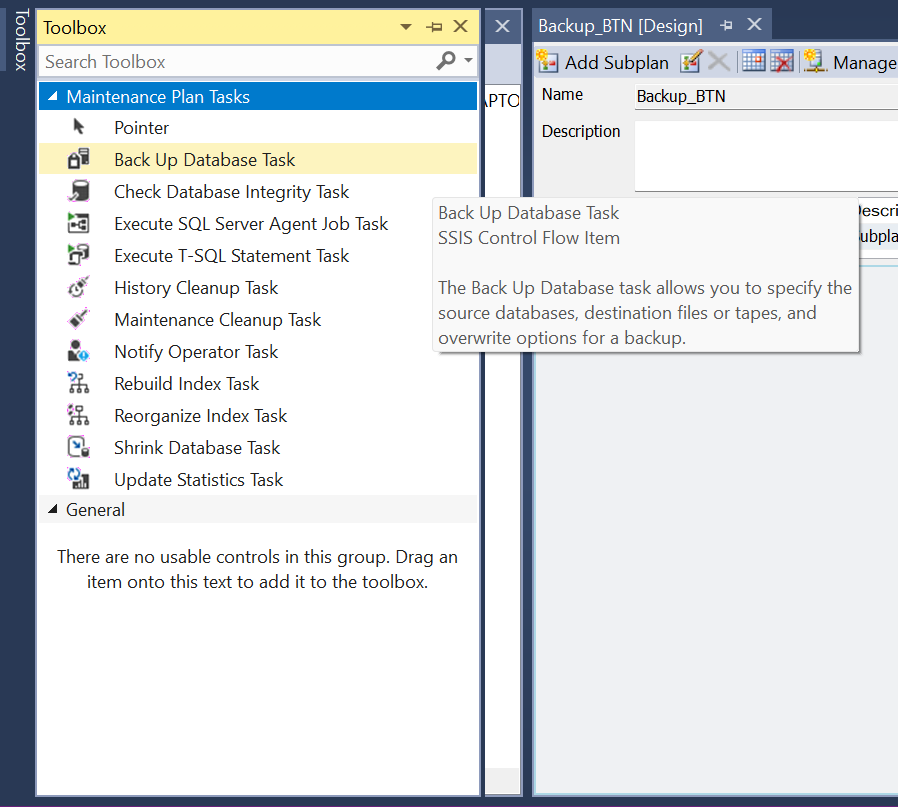
*Giao diện Maintenance mới khởi tạo*

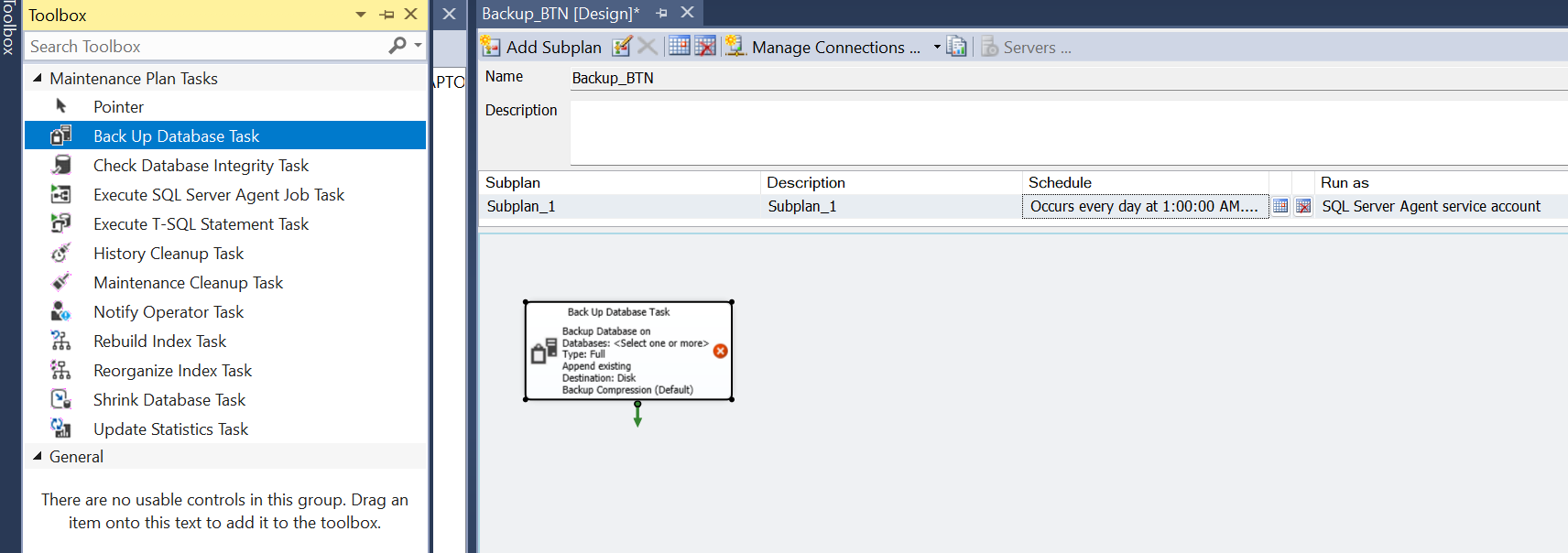
Bước 2: Chọn Subplan Schedule -> Mở bảng Job Schedule Properties để tạo lịch trình Full Backup



*Full Back Up lần đầu cho hệ thống*

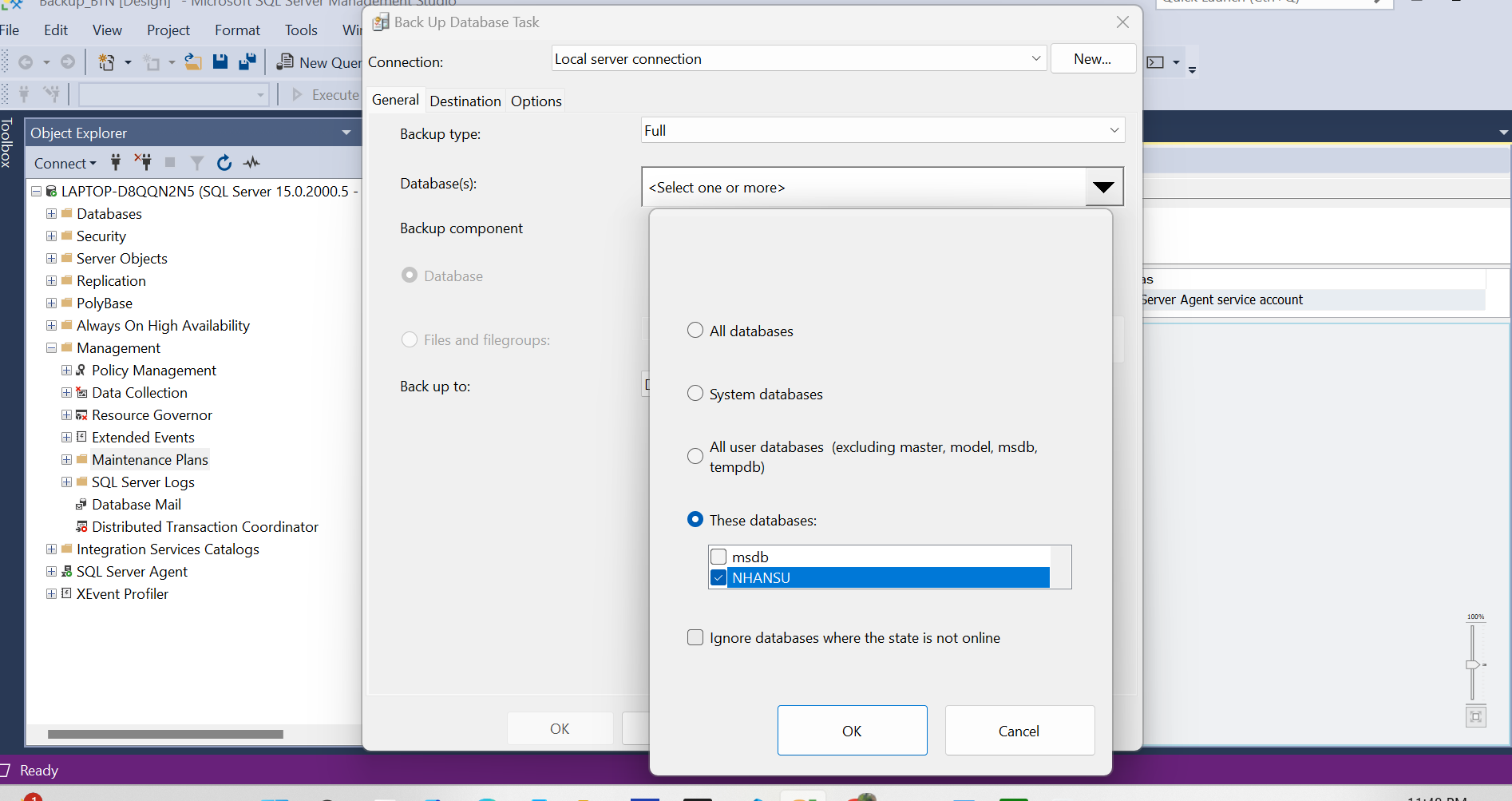
Bước 3: Mở Toolbox  -> Mở Maintenance Plan Tasks  -> Chọn Back Up Database Task để tạo Task



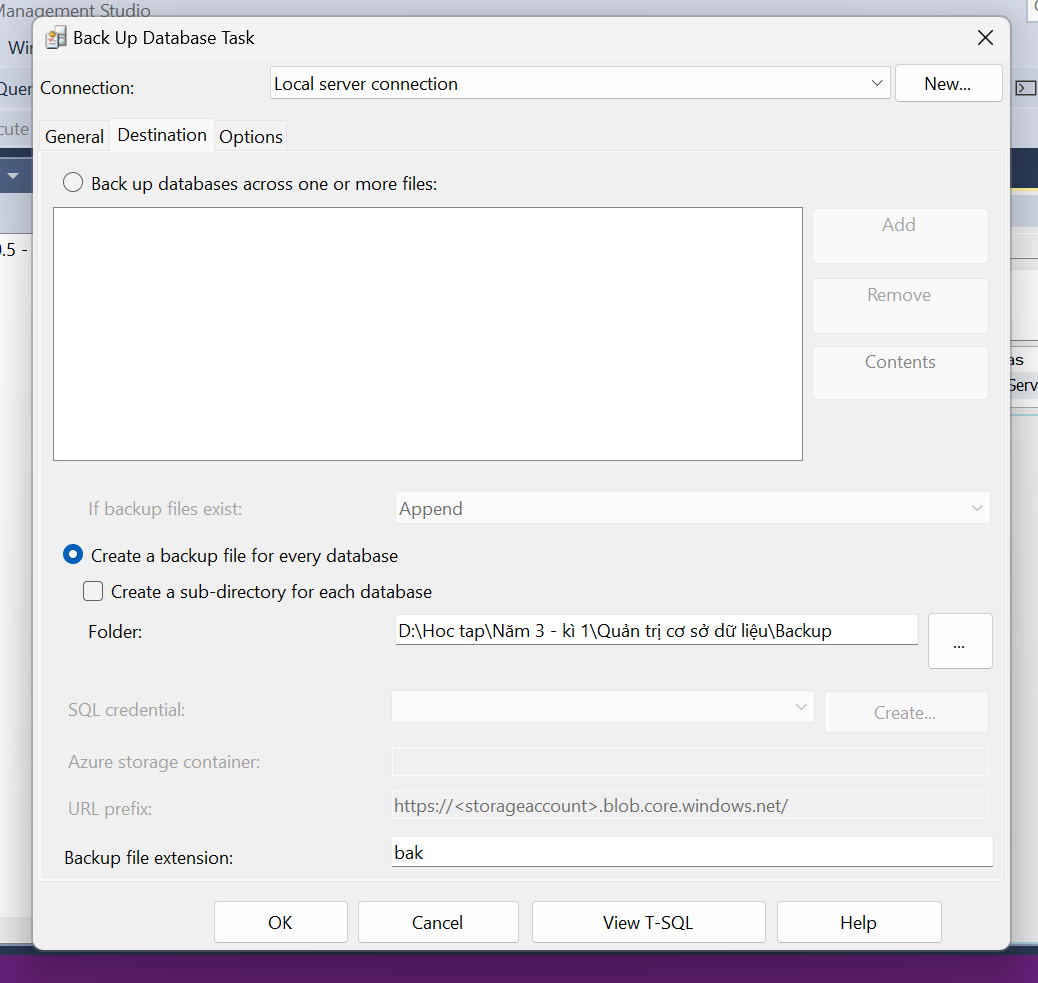


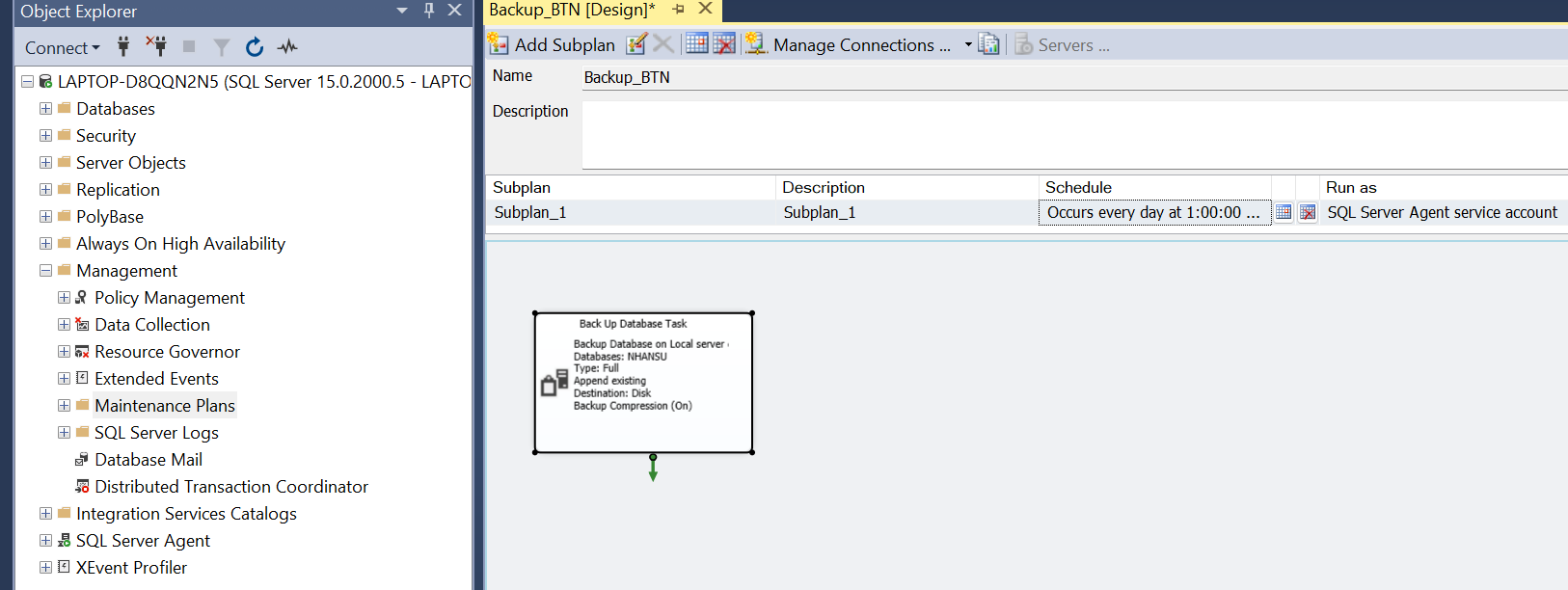
*Bản task mới*

Bước 4: Kích đúp chuột vào bản task -> Thiết lập cài đặt sao lưu cho Task Full Backup

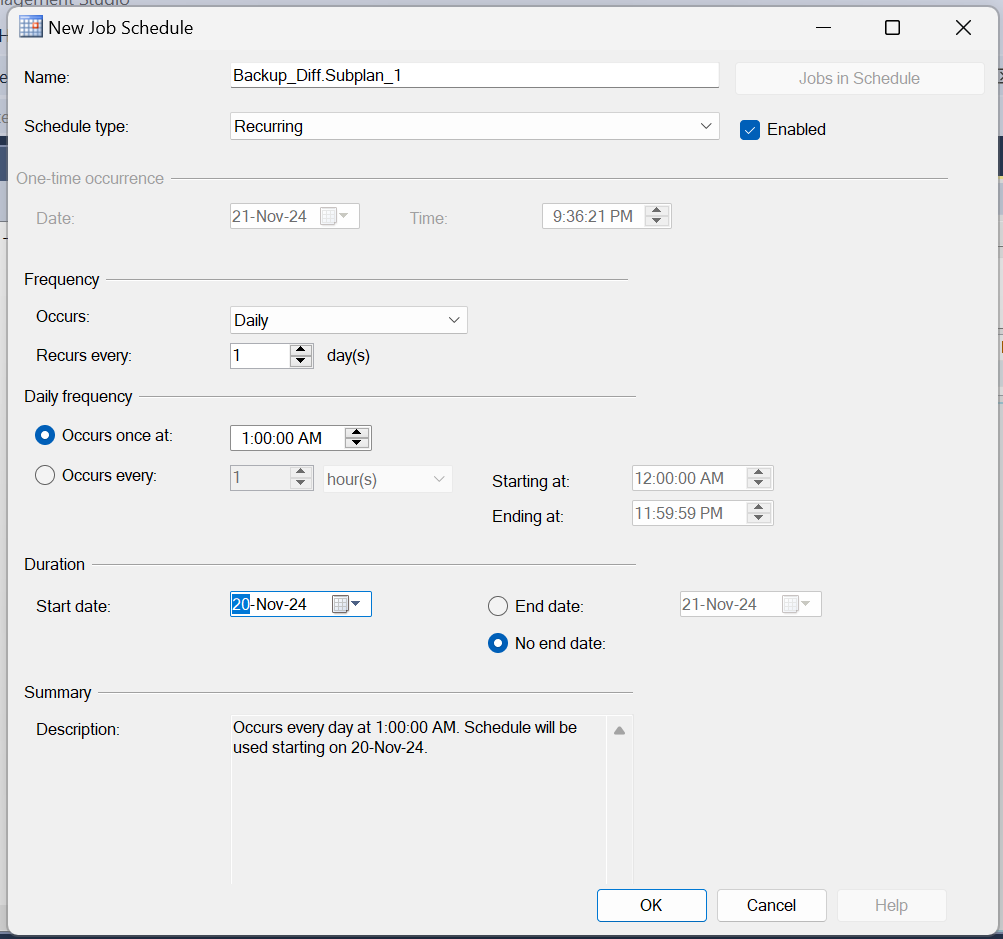


Bước 5: Chọn vị trí lưu dữ liệu backup

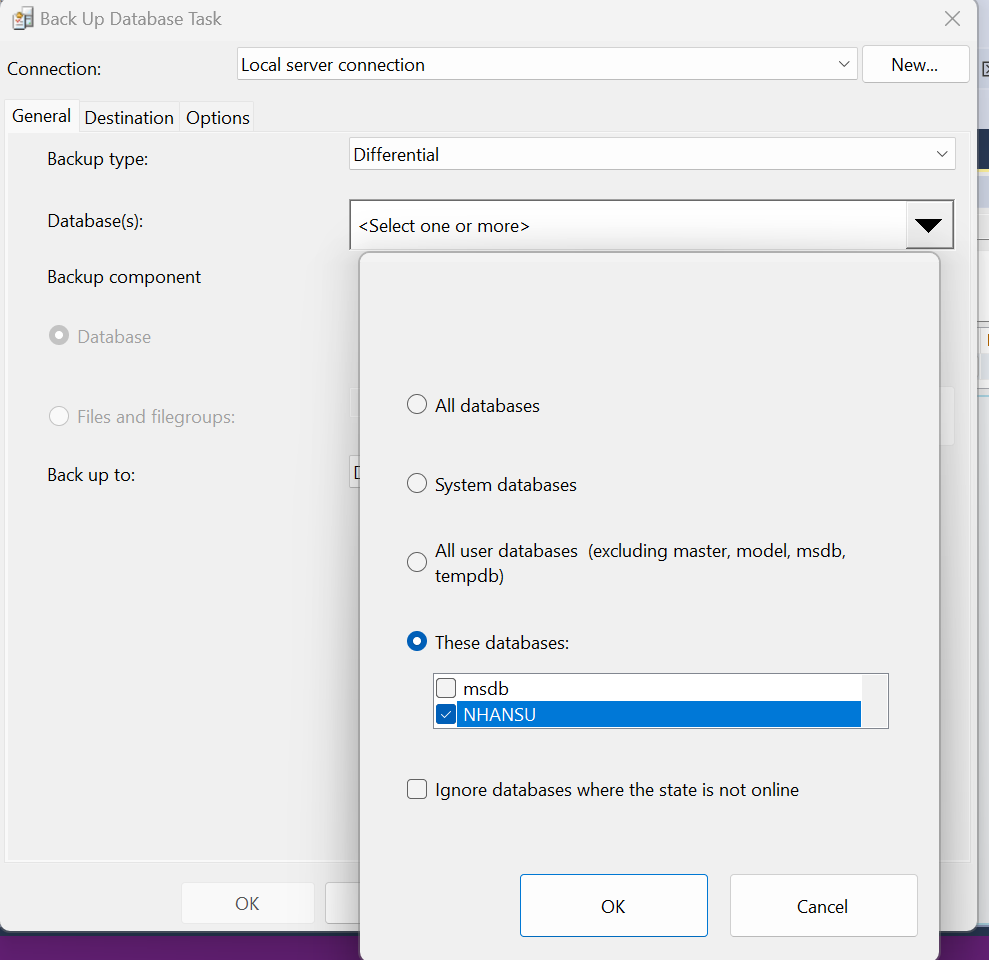




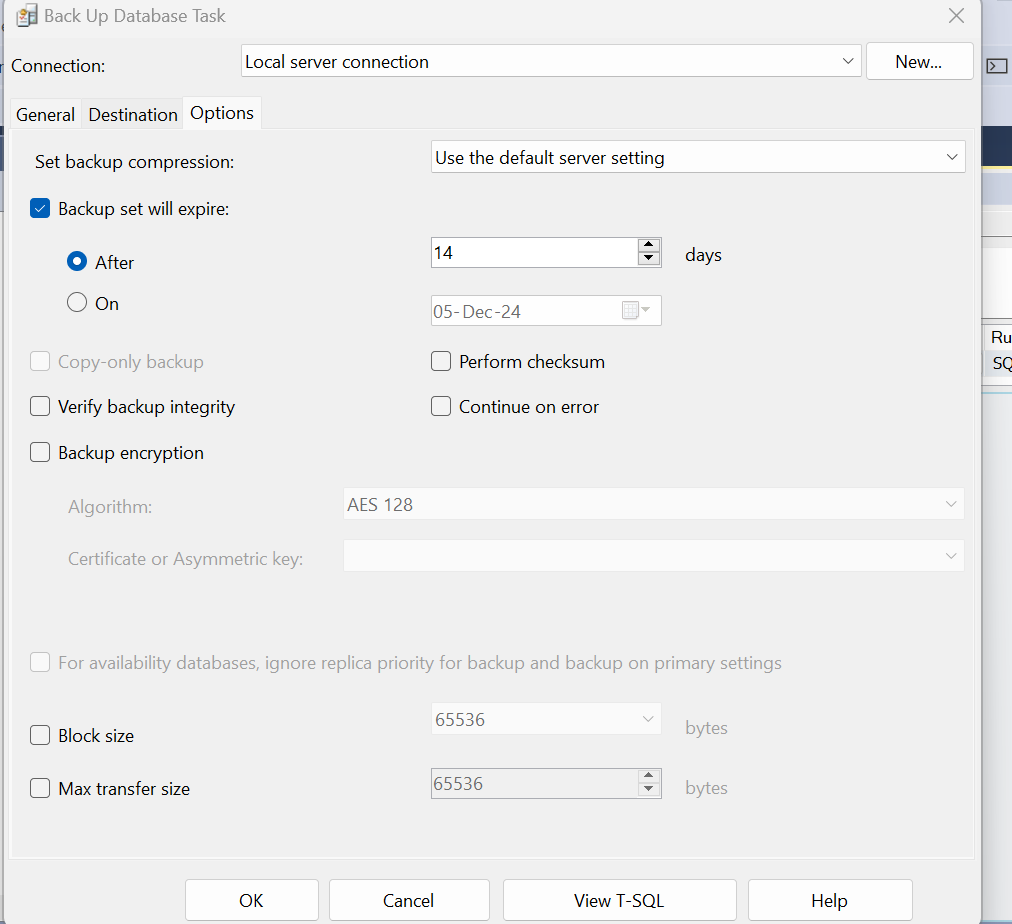
*Bản task hoàn chỉnh*

**Tương tự, thực hiện Diffrerential Backup** 

*Tạo lịch trình Differential Backup định kì.*

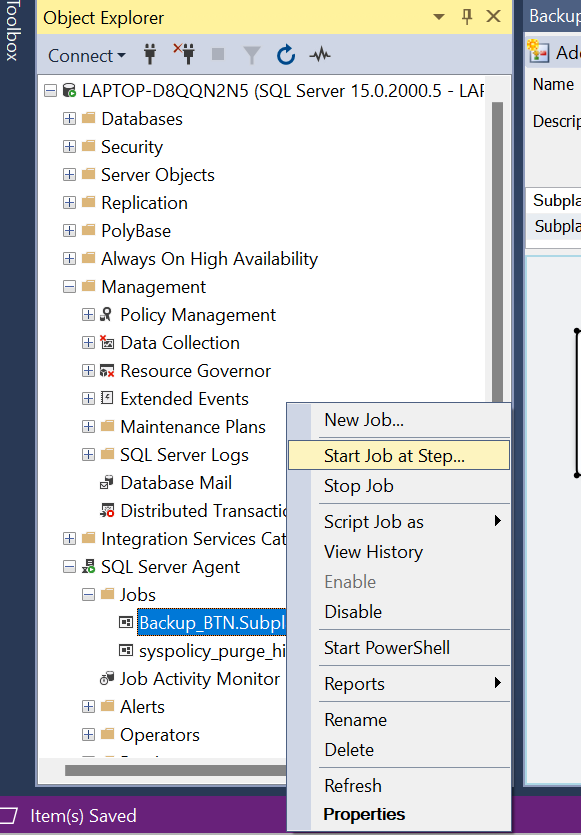


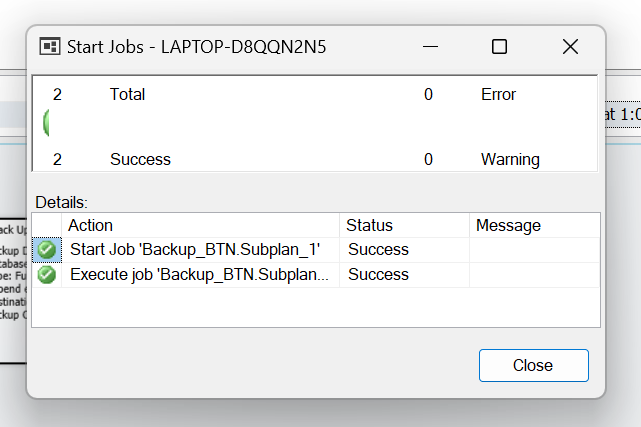
*Thiết lập cài đặt cho Task Differential Backup*



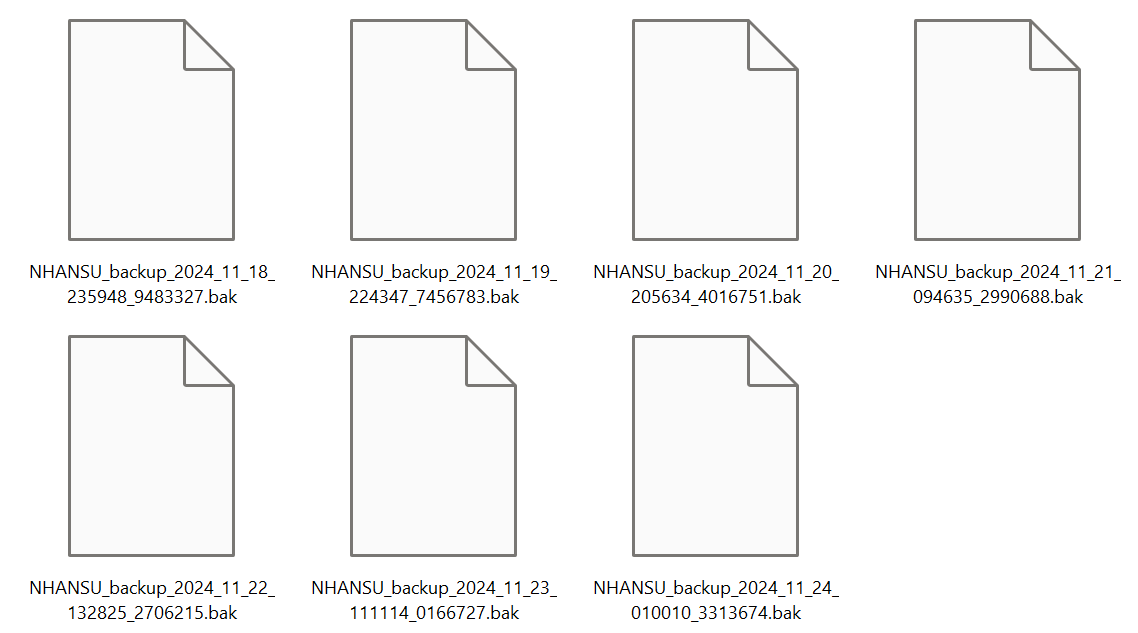
*Thiết lập xoá bản backup cũ định kì*

Bước 6: : Khởi động Task để thực hiện Backup





*Task khởi động thành công tạo backup*



*Bản backup mới được tạo thành công*

## **PHƯƠNG ÁN GIẢI QUYẾT VẤN ĐỀ DỮ LIỆU RẤT LỚN KHÔNG ĐỦ LƯU TRỮ TRÊN MỘT Ổ ĐĨA**

Trường hợp dung lượng trên một ổ đĩa quá lớn và không thể lưu trữ được nữa, thì nhóm đã đề xuất phương án để giải quyết này là phân tán dữ liệu qua nhiều máy chủ để quản lý hiệu quả hơn. Phương án đã được thực hiện như sau:

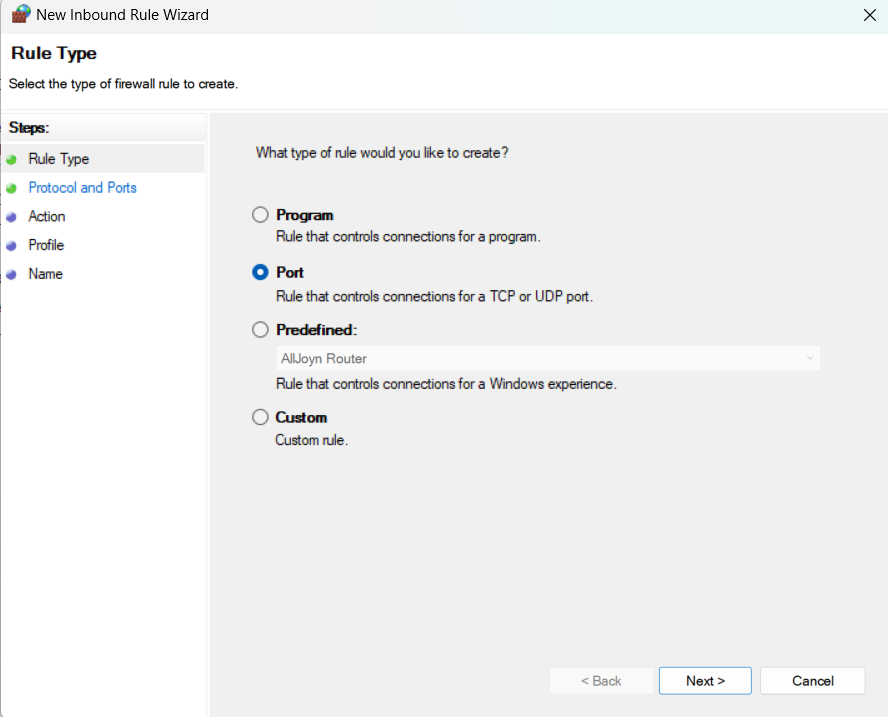
* **Server gốc**: Đây là **máy chủ chính** (quản lý), có toàn quyền thực hiện các thao tác **SELECT**, **INSERT**, **UPDATE**, và **DELETE** trên cả dữ liệu của chính nó và dữ liệu được lưu trữ trên Server B.
* **Server lưu trữ**: Đây là **máy chủ phụ** (dữ liệu được phân tán), chỉ cho phép truy vấn dữ liệu (**SELECT**) từ phía Server A, không có quyền thực hiện các thao tác ghi dữ liệu như **INSERT**, **UPDATE**, hoặc **DELETE** trên Server A hoặc chính nó.

Mục tiêu:

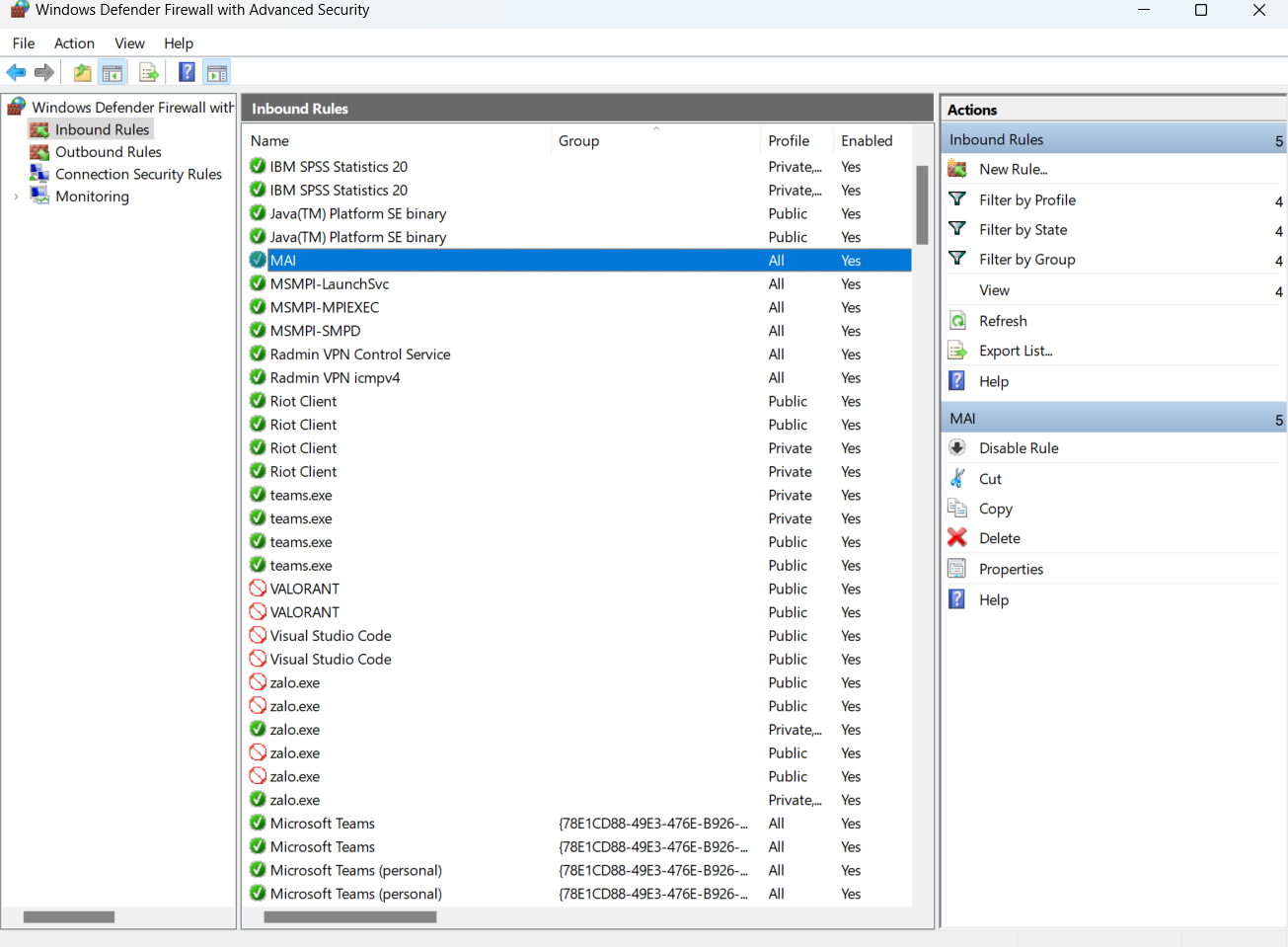
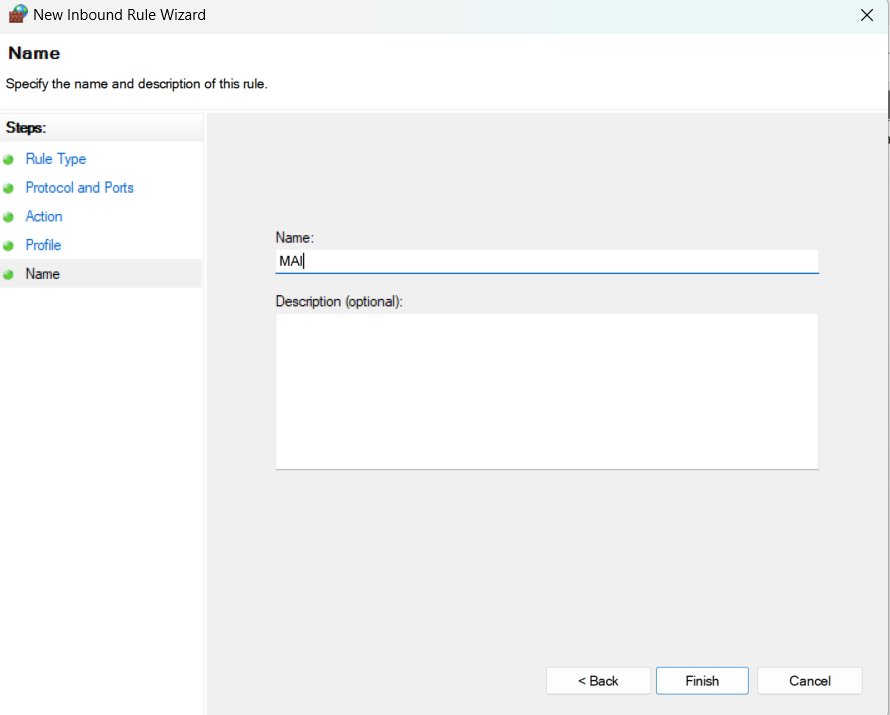
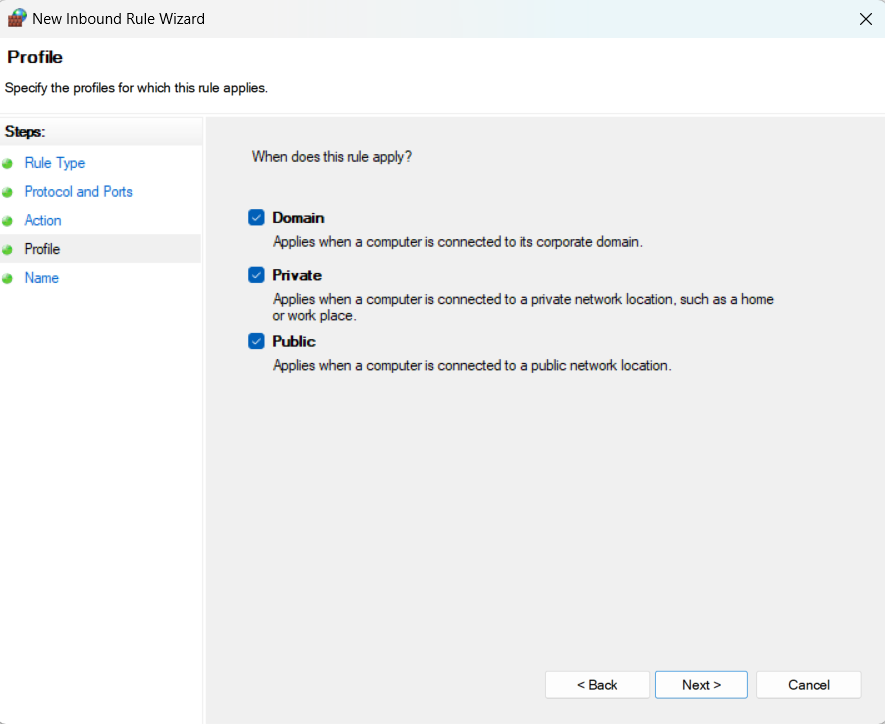
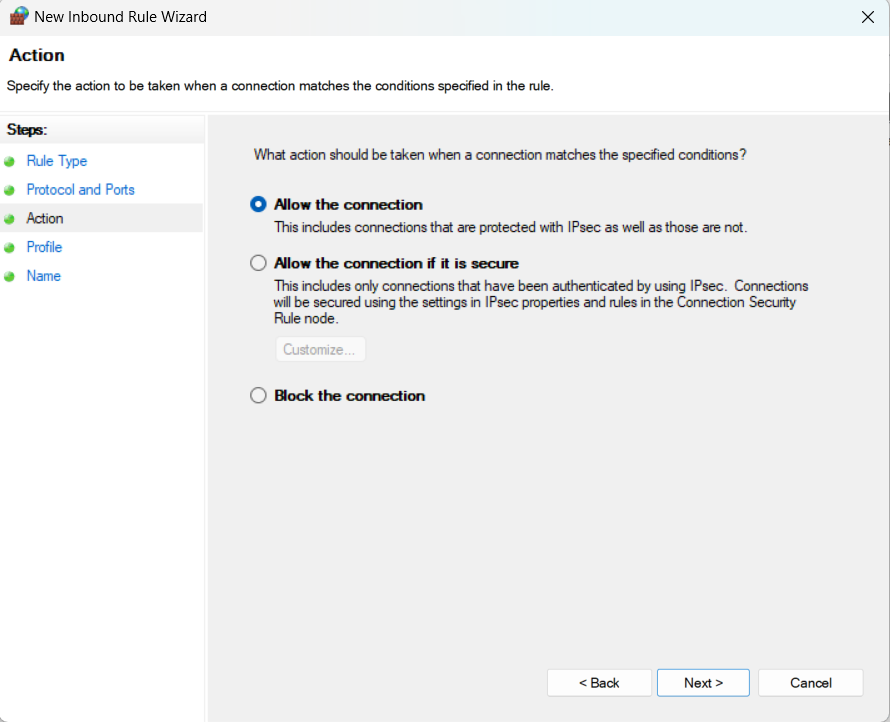
* **Server gốc** đóng vai trò trung tâm, giúp thực hiện và quản lý mọi thao tác trên cả hai server.
* **Server lưu trữ** chỉ phục vụ lưu trữ và cung cấp dữ liệu khi cần thiết, hạn chế quyền ghi nhằm đảm bảo tính nhất quán và an toàn cho dữ liệu phân tán.

Các bước thực hiện:

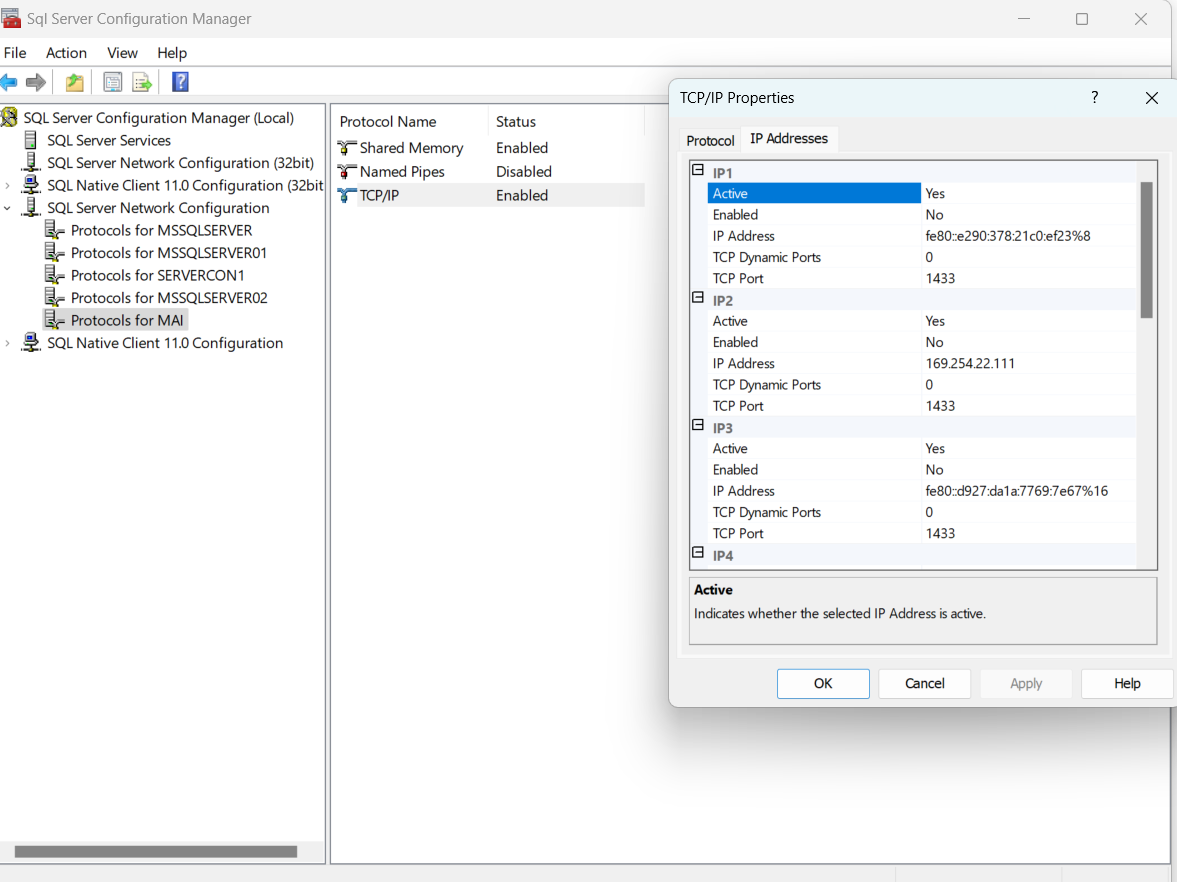
Bước 1: Mở cổng 1433 ở máy server gốc(server quản lý) 🡪 truy cập Tường Lửa 🡪Avanced setting 🡪 New rule ở Inbound 🡪 (Các bước mô tả ở 5 ảnh dưới đây)

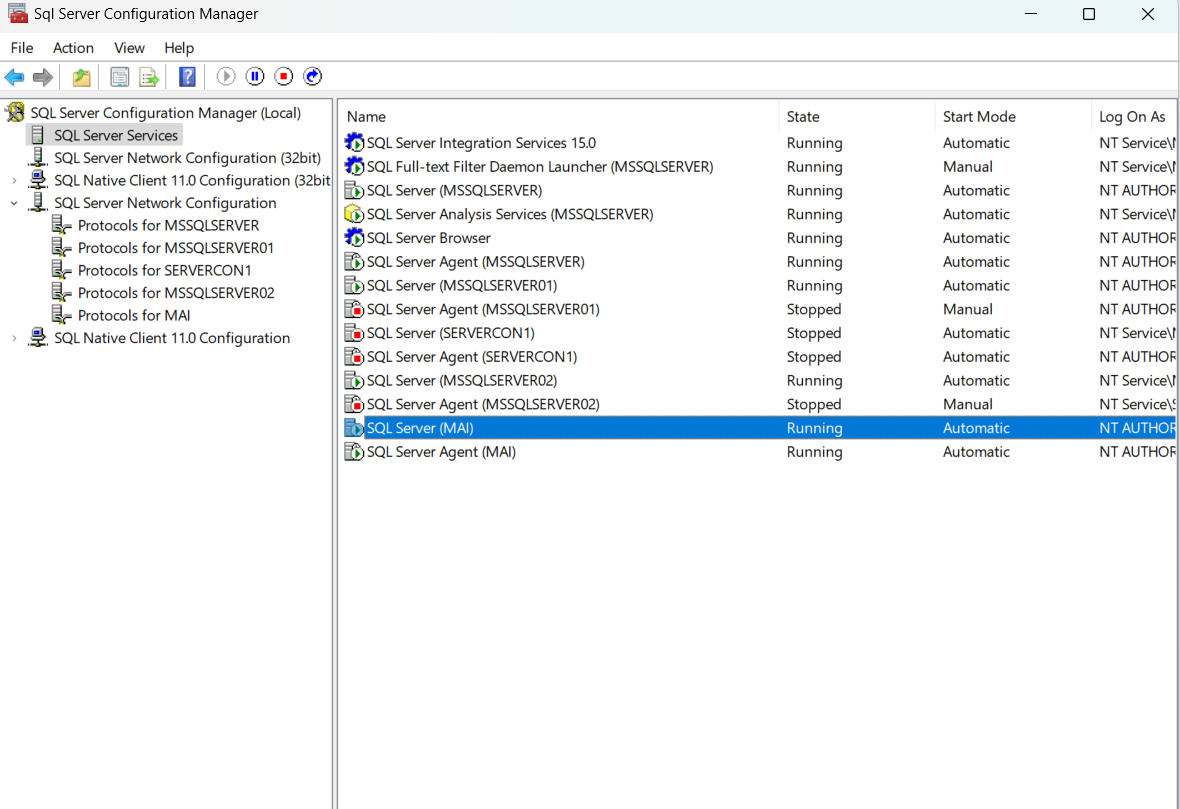






Bước 2: Đảm bảo server quản lý đã mở cổng 1433 🡪 SQL Server Configuration Manager





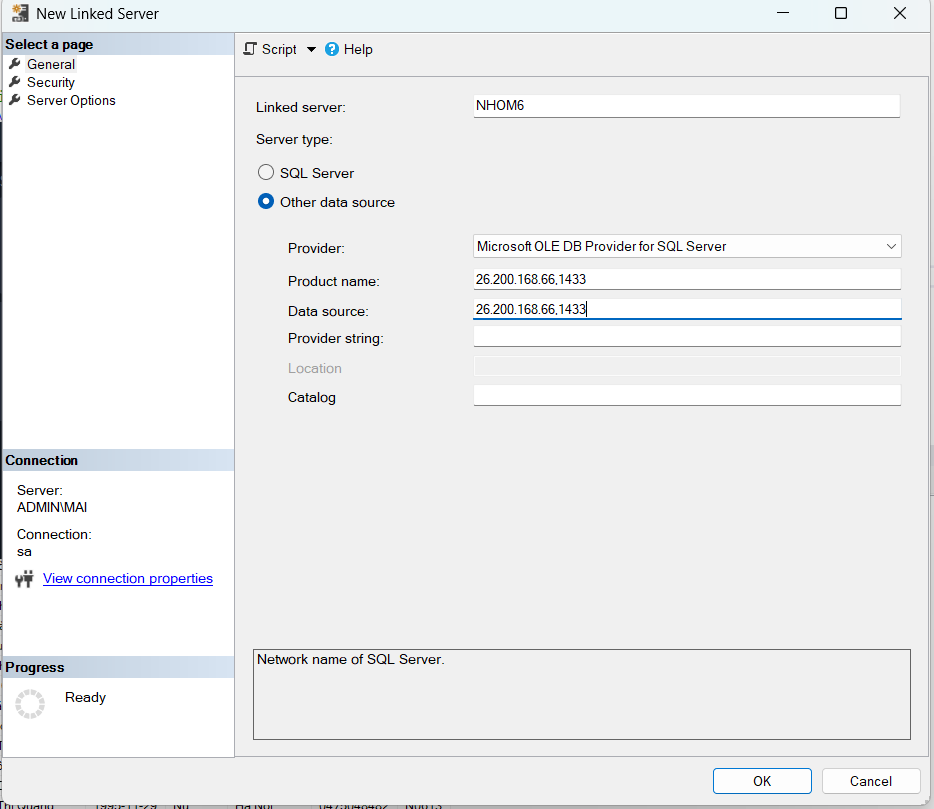
*Đảm bảo SQL Server và SQL Server Agent đã Running*

Bước 3: Tương tự, thực hiện mở cổng 1433 ở Laptop chưa Server phụ

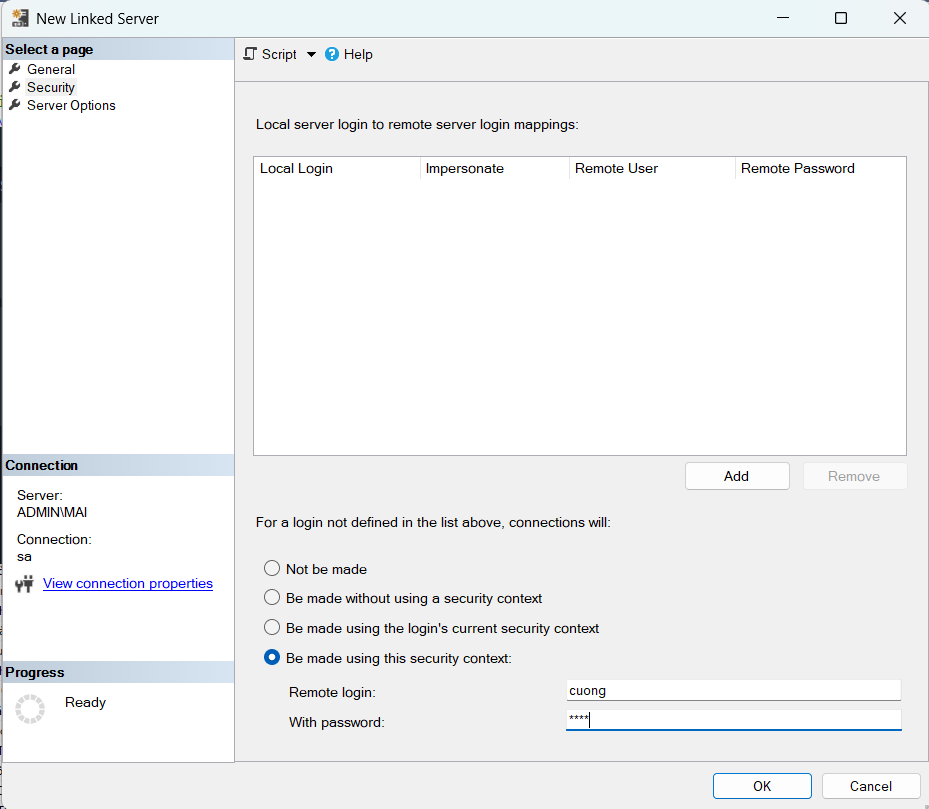
Bước 4: Ở Server lưu trữ, tạo một database có cấu trúc giống với Database NHANSU của Hệ Thống Quản Lý NHân Sự, bao gồm bảng NHANVIEN, CHAMCONG,LUONG, TAIKHOAN. (Không tạo bảng CHUCVU vì không quan trọng)

Bước 5: Tạo tài khoản cho người dùng ở Server lưu trữ (có mật khẩu đăng nhập) và gán quyền chỉ được đọc (SELECT) cho tài khoản đó.

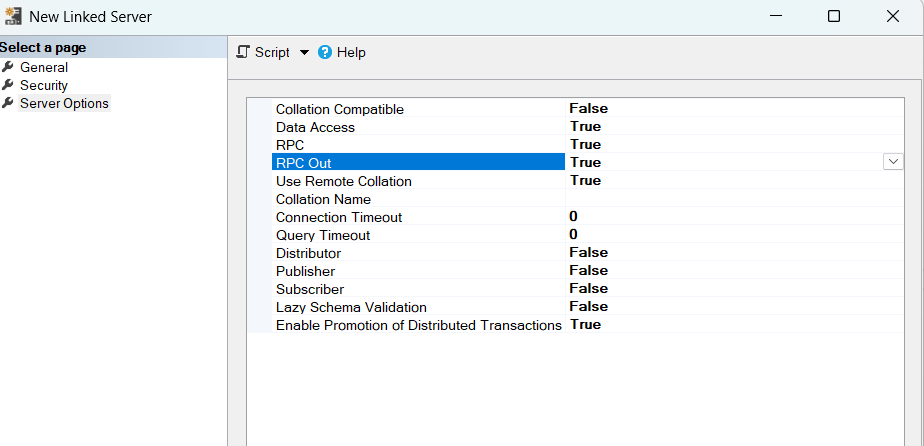
Bước 6: Ở Server quản lý tiến hành Linked Server đến Server lưu trữ(có tên NHOM6), gồm các bước ở những hình sau



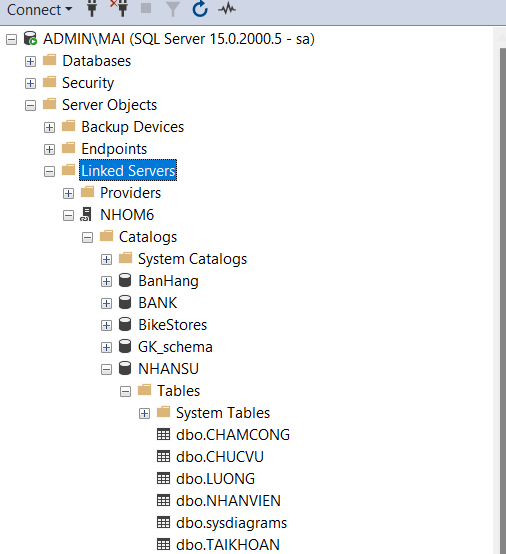
*Tên server lưu trữ(NHOM6) và nhập IP của server lưu trữ để linked server*



*Nhập tên và mật khẩu của tài khoản của Server lưu trữ đã tạo ở Bước 5*

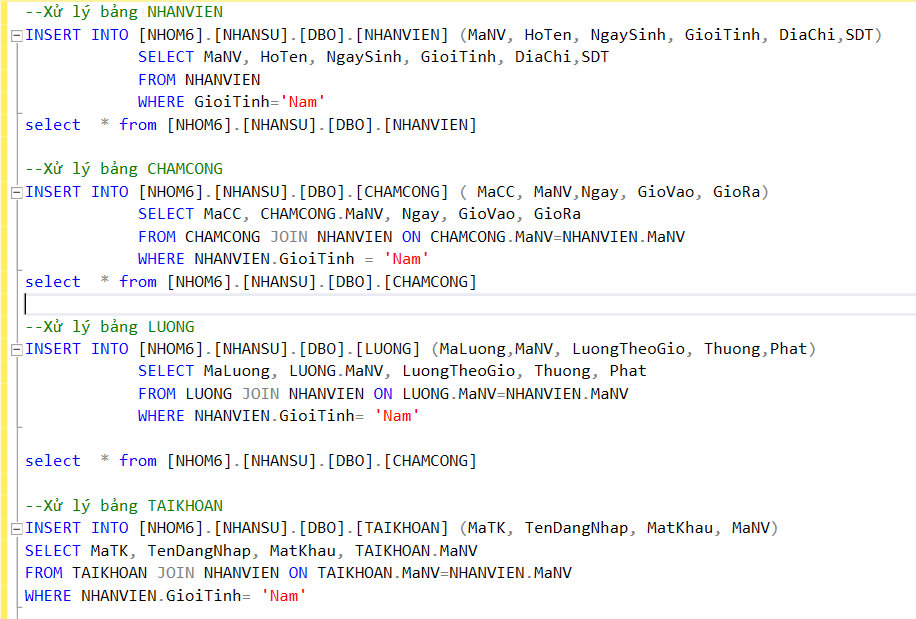


*Cài đặt Data Access, RPC, và RPC Out từ FALSE 🡪 TRUE*

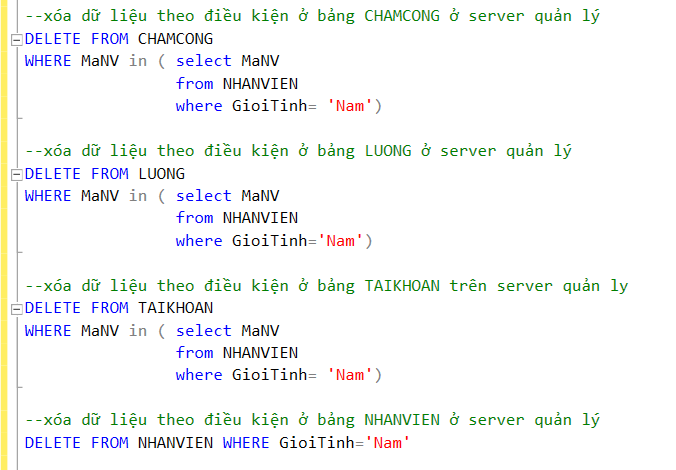


*Server lưu trữ đã linked server thành công với Server quản lý*

Bước 7: Tiến hành phân tán dữ liệu từ Server Quản lý xuống Server lưu trữ NHOM6 theo điều kiện phân những dữ liệu thuộc về giới tính ‘Nam’

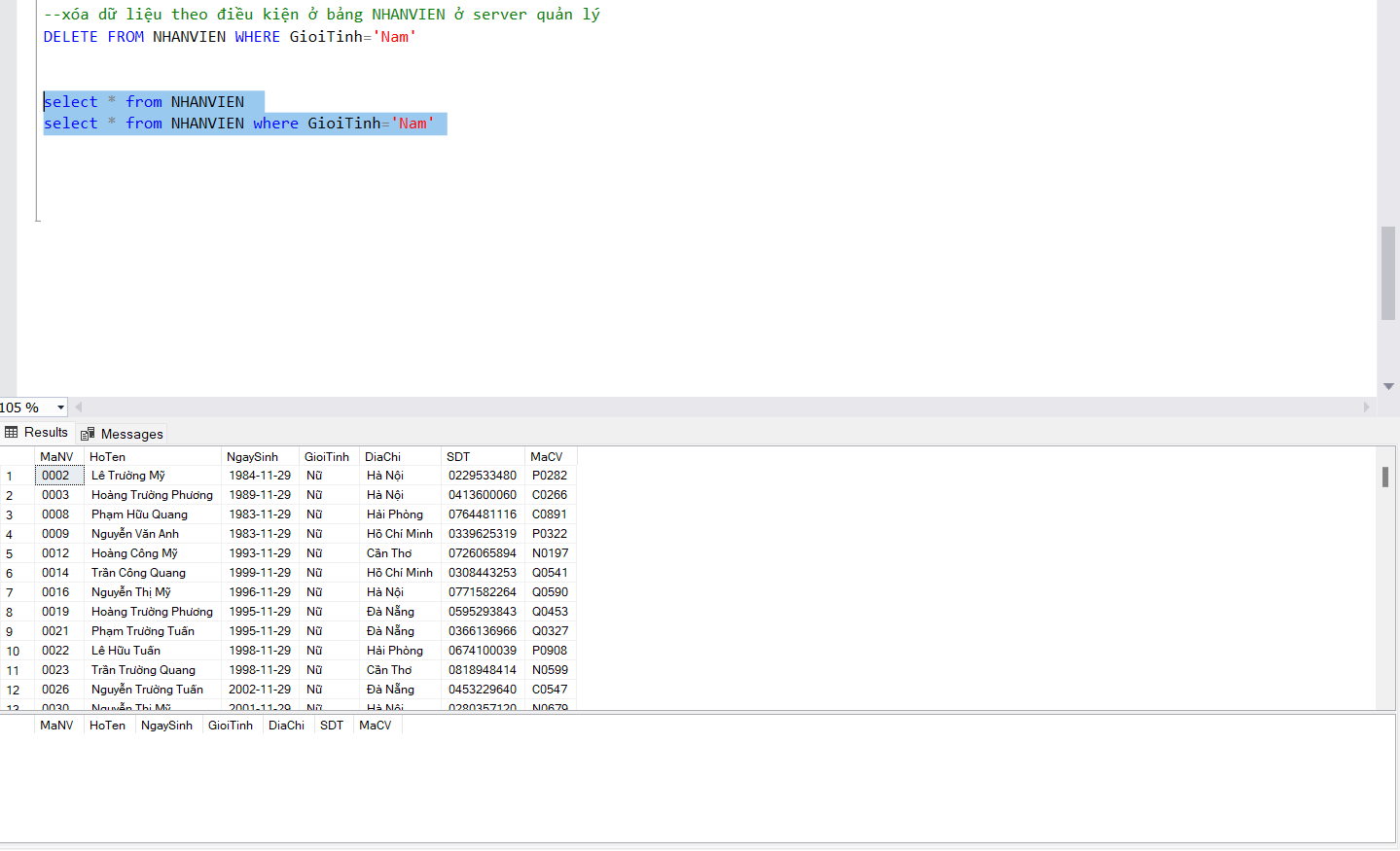


Bước 8: Tiến hành xóa những dữ liệu đã phân tán cho serv er lưu trữ ở server quản lý





*Dữ liệu đã được phân tán từ Server quản lý 🡪 Server lưu trữ*



*Minh chứng ở Server Quản lý không còn dữ liệu đã phân tán*

## **ĐỀ RA VÀ THỰC HIỆN PHƯƠNG ÁN ĐỂ HẠN CHẾ TẤN CÔNG SQL INJECTION**

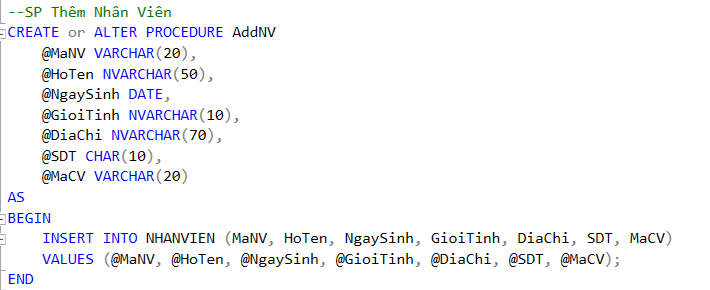
Phương pháp: Thiết lập truy cập dữ liệu qua thủ tục (Stored Procedures)

**Lợi ích:** sử dụng Stored Procedures để hạn chế quyền truy cập dữ liệu trực tiếp vào các bảng. Người dùng chỉ có quyền gọi thủ tục mà không có quyền thao tác trực tiếp với các bảng dữ liệu.

**Thực hiện:**

### Thủ tục thêm mới dữ liệu:

* Thủ tục thêm mới một nhân viên

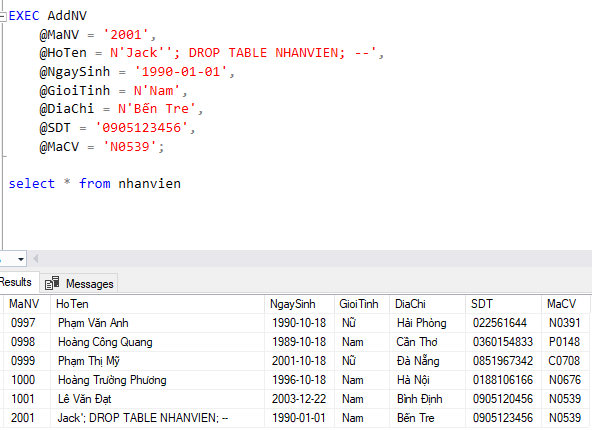


￼

Thủ tục này được thiết kế để thêm một nhân viên mới vào bảng **NHANVIEN** với các thông tin chi tiết như (MaNV, HoTen, NgaySinh, GioiTinh, DiaChi, SDT, MaCV).

Cách ngăn chặn: Thủ tục sử dụng **tham số** cho đầu vào giúp ngăn chặn khi có trường hợp người dùng nhập đầu vào độc hại và sẽ giúp loại bỏ chuỗi SQL động.

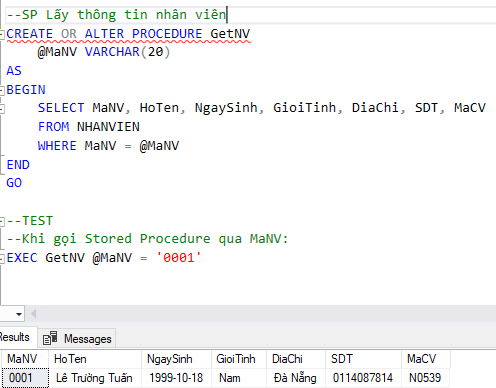
Ví dụ, nếu người dùng cố gắng đưa mã độc qua tham số đầu vào, như sau:



Tham số hóa đảm bảo rằng giá trị @HoTen (hoặc bất kỳ tham số nào khác) sẽ được xử lý như **một giá trị**. Nếu người dùng nhập @HoTen = N'Jack''; DROP TABLE NHANVIEN; --' thì SQL Server **không xem nó như một đoạn mã SQL**, mà chỉ coi đó là **giá trị chuỗi của biến** (dù có chứa ký tự đặc biệt như ', ;, --). Vì vậy sẽ giúp ngăn chặn được việc chèn mã độc thay đổi cấu trúc của câu lệnh SQL ban đầu. Mã độc ('; DROP TABLE NHANVIEN; -- ’) sẽ không được thực thi.

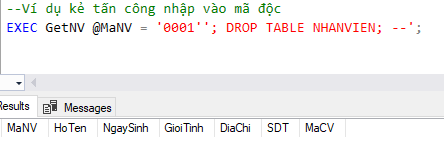
### Thủ tục truy xuất dữ liệu

* Ví dụ: Thủ tục lấy thông tin nhân viên thông qua mã nhân viên, thay vì sử dụng câu lệnh (SELECT, SELECT \*) tránh được rò rỉ dữ liệu.



Sử dụng tham số @MaNV còn giúp ngăn chặn việc chèn mã độc vào câu lệnh truy vấn:

Với tham số hóa, SQL Server chỉ coi toàn bộ chuỗi '0001'; DROP TABLE NHANVIEN; -- là một **giá trị** để tìm kiếm trong cột MaNV

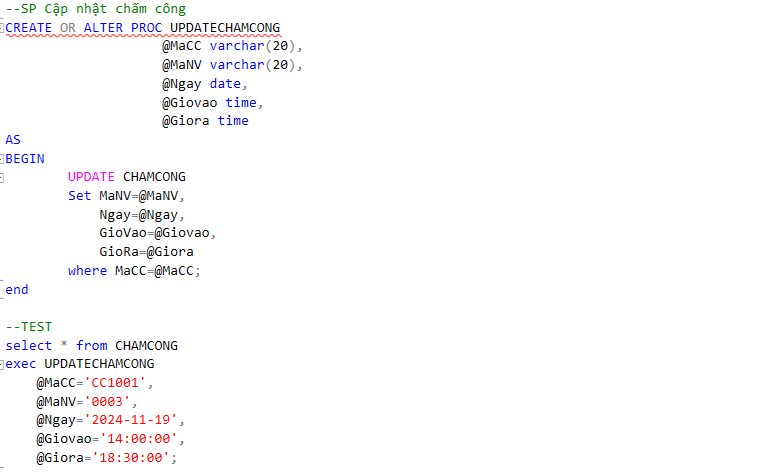


Kết quả: Không có bản ghi nào được tìm thấy, và mã độc không thực thi được.

### Thủ tục cập nhật dữ liệu

* Ví dụ: Cập nhật thông tin chấm công của nhân viên

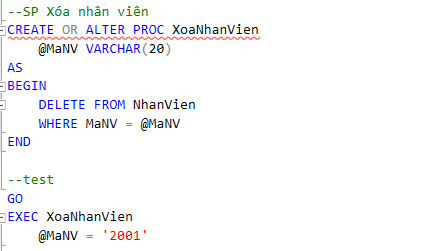
Thủ tục sẽ cập nhật thông tin trong bảng CHAMCONG như mã nhân viên (MaNV), ngày chấm công (Ngay), giờ vào (GioVao), giờ ra (GioRa) dựa trên mã chấm công (MaCC) khi người dùng cần sửa thông tin chấm công do nhập sai, hoặc cập nhật lại giờ làm việc của nhân viên.



Cách này hạn chế rủi ro từ việc cập nhật dữ liệu trực tiếp và giúp kiểm soát các thay đổi dễ dàng hơn.

### Xóa dữ liệu bằng thủ tục

* Ví dụ: Xóa nhân viên theo mã nhân viên (MaNV)



￼

Việc xóa dữ liệu cũng sẽ ngăn chặn SQL Injection thông qua tham số hóa. Và khi cấp quyền thực thi cho thủ tục thay vì cấp quyền Delete trực tiếp trên bảng, người dùng sẽ không thể xóa trực tiếp dữ liệu từ bảng. Thay vào đó, họ chỉ có thể gọi thủ tục để thực hiện hành động xóa.  Ngoài ra, thao tác Xóa dữ liệu thông qua thủ tục còn giúp dễ dàng kiểm soát (tạo thêm các điều kiện bảo vệ cần thiết trước khi thật sự xóa dữ liệu).