**TRƯỜNG ĐẠI HỌC KINH TẾ ĐÀ NẴNG**

**- KHOA THỐNG KÊ – TIN HỌC -**

----\*\*\*----

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, biểu tượng

Mô tả được tạo tự động**

**BÁO CÁO NHÓM**

**MÔN HỌC: QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**Đề tài: Hệ thống quản lý nhà trọ sinh viên**

|  |  |  |
| --- | --- | --- |
| **GVHD** | **:** | ThS. Cao Thị Nhâm |
| **Nhóm** | **:** | Nhóm 1\_48K14.1 |
| **Sinh viên thực hiện** | **:** | Trương Văn Gia Bảo  Phạm Thị Thảo Huyền  Võ Thị Tuyết Ngân  Võ Thị Tuyết Sinh  Trần Văn Tuấn |

**ĐÀ NẴNG, 11/2024**

# MỤC LỤC

[MỤC LỤC i](#_Toc183782956)

[DANH MỤC HÌNH ẢNH iii](#_Toc183782957)

[DANH MỤC BẢNG BIỂU iv](#_Toc183782958)

[I. Thu thập dữ liệu 1](#_Toc183782959)

[II. Chuẩn hóa các bảng: 1](#_Toc183782960)

[2.1. Chuẩn hoá dữ liệu. 1](#_Toc183782961)

[2.2. Mô tả chi tiết các bảng: 2](#_Toc183782962)

[III. Xây dựng cơ sở dữ liệu. 6](#_Toc183782963)

[3.1. Tạo bảng phòng trọ 6](#_Toc183782964)

[3.2. Tạo bảng hợp đồng 6](#_Toc183782965)

[3.3. Tạo bảng khách thuê 7](#_Toc183782966)

[3.4. Tạo bảng hoá đơn 7](#_Toc183782967)

[3.5. Tạo bảng tài khoản 7](#_Toc183782968)

[3.6. Tạo bảng dịch vụ 8](#_Toc183782969)

[3.7. Tạo bảng hoá đơn chi tiết 8](#_Toc183782970)

[IV. Xây dựng module tạo dữ liệu dump cho các bảng 8](#_Toc183782971)

[4.1. Tạo dữ liệu cho bảng phòng trọ 8](#_Toc183782972)

[4.2. Tạo dữ liệu cho bảng hợp đồng 9](#_Toc183782973)

[4.3. Tạo dữ liệu cho bảng khách thuê 10](#_Toc183782974)

[4.4. Tạo dữ liệu cho bảng hoá đơn 11](#_Toc183782975)

[4.5. Tạo dữ liệu cho bảng dịch vụ 12](#_Toc183782976)

[4.6. Tạo dữ liệu cho bảng hoá đơn chi tiết 13](#_Toc183782977)

[4.7. Tạo dữ liệu cho bảng tài khoản 14](#_Toc183782978)

[V. Mô tả các module trong cơ sở dữ liệu 15](#_Toc183782979)

[5.1 Module thêm mới phòng trọ: 15](#_Toc183782980)

[5.2 Module Cập nhật thông tin phòng trọ 17](#_Toc183782981)

[5.3 Module Xóa thông tin phòng trọ 19](#_Toc183782982)

[5.4 Module Tìm thông tim phòng trọ 22](#_Toc183782983)

[5.5 Module Lọc phòng theo trạng thái 23](#_Toc183782984)

[5.6 Module Thêm mới hợp đồng 26](#_Toc183782985)

[5.7 Module Cập nhật thông tin hợp đồng 29](#_Toc183782986)

[5.8 Module Xóa hợp đồng 32](#_Toc183782987)

[5.9 Module Tìm kiếm hợp đồng theo mã hợp đồng 33](#_Toc183782988)

[5.10 Trigger Thêm khách thuê 35](#_Toc183782989)

[5.11 Trigger Cập nhật tình trạng phòng khi xóa khách thuê 37](#_Toc183782990)

[5.12 Module Thêm mới khách thuê 40](#_Toc183782991)

[5.13 Module Cập nhật thông tin khách thuê 43](#_Toc183782992)

[5.14 Module Xóa khách thuê 45](#_Toc183782993)

[5.15 Trigger Tính tổng tiền hóa đơn 47](#_Toc183782994)

[5.16 Module Xem hóa đơn theo phòng 49](#_Toc183782995)

[5.17 Module Xóa hóa đơn 50](#_Toc183782996)

[5.18 Module Tạo hóa đơn 51](#_Toc183782997)

[5.19 Module Sửa hóa đơn 53](#_Toc183782998)

[5.20 Module Thêm dịch vụ mới 55](#_Toc183782999)

[5.21 Module Cập nhật thông tin dịch vụ 56](#_Toc183783000)

[5.22 Module Xóa dịch vụ 57](#_Toc183783001)

[5.23 Trigger Cập nhật tổng tiền trong bảng HoaDon 59](#_Toc183783002)

[5.24 Module Tạo hóa đơn chi tiết 62](#_Toc183783003)

[5.25 Module Sửa hóa đơn chi tiết 64](#_Toc183783004)

[5.26 Module Xóa hóa đơn chi tiết 66](#_Toc183783005)

[5.27 Module Thêm tài khoản 67](#_Toc183783006)

[5.28 Module Sửa tài khoản 70](#_Toc183783007)

[5.29 Module Xóa tài khoản 72](#_Toc183783008)

[VI. Xây dựng cơ chế bảo mật phù hợp cho cơ sở dữ liệu 73](#_Toc183783009)

[6.1. Giới thiệu 73](#_Toc183783010)

[6.2. Lý do lựa chọn SQL Server Authentication 74](#_Toc183783011)

[VII. Xây dựng cơ chế backup dữ liệu tự động cho cơ sở dữ liệu 80](#_Toc183783012)

[VIII. Phương án giải quyết vấn đề dữ liệu lớn 88](#_Toc183783013)

[IX. Đề phòng phương án hệ thống bị tấn công bằng SQL Injection: 95](#_Toc183783014)

[9.1. Lý thuyết Injection: 95](#_Toc183783015)

[9.2. Phương án phòng chống 95](#_Toc183783016)

# DANH MỤC HÌNH ẢNH

[Hình 1 Sơ đồ quan hệ 2](#_Toc183782881)

[Hình 2 Kết quả tạo dữ liệu cho bảng phòng trọ 9](#_Toc183782882)

[Hình 3 Kết quả tạo dữ liệu cho bảng hợp đồng 10](#_Toc183782883)

[Hình 4 Kết quả tạo dữ liệu cho bảng khách thuê 11](#_Toc183782884)

[Hình 5 Kết quả tạo dữ liệu cho bảng hoá đơn 12](#_Toc183782885)

[Hình 6 Kết quả tạo dữ liệu cho bảng dịch vụ 13](#_Toc183782886)

[Hình 7 Kết quả tạo dữ liệu cho bảng hoá đơn chi tiết 13](#_Toc183782887)

[Hình 8 Kết quả tạo dữ liệu cho bảng tài khoản 14](#_Toc183782888)

[Hình 9 Thao tác cấp quyền cho Chủ trọ 74](file:///C:\Users\DELL\Downloads\Nhóm1_48K14.1_QTCSDL%20(1)%20(1).docx#_Toc183782889)

[Hình 10 Gán quyền xem 2 table Hóa đơn và Hợp đồng cho khách thuê 75](#_Toc183782890)

[Hình 11 Gán quyền cho chủ trọ và khách thuê thành công 75](#_Toc183782891)

[Hình 12 Trước khi thêm cột MatKhauMaHoa 77](#_Toc183782892)

[Hình 13 Sau khi thêm cột MatKhauMaHoa 77](#_Toc183782893)

[Hình 14 Kết quả sau khi mã hóa 78](#_Toc183782894)

[Hình 15 Kết quả sau khi xóa cột dữ liệu ban đầu 79](#_Toc183782895)

[Hình 16 Kết quả sau khi giải mã 79](#_Toc183782896)

[Hình 17 Vào mục backup tại server cloud muốn thực hiện backup 80](file:///C:\Users\DELL\Downloads\Nhóm1_48K14.1_QTCSDL%20(1)%20(1).docx#_Toc183782897)

[Hình 18 Chọn database và thiết lập thời gian backup 80](#_Toc183782898)

[Hình 19 Kết quả sau khi thiết lập 81](#_Toc183782899)

[Hình 20 Đăng nhập vào Cloud trên SQL Server 82](#_Toc183782900)

[Hình 21 Chọn task và chọn Export Data-tier Application 83](#_Toc183782901)

[Hình 22 Xuất hiện cửa sổ Export Data-tier Application -> Chọn next 84](#_Toc183782902)

[Hình 23 Chọn thư mục muốn tạo file backup và đặt tên file 85](#_Toc183782903)

[Hình 24 Chọn Next -> Chọn Finish 86](#_Toc183782904)

[Hình 25 Hoàn thành backup 87](#_Toc183782905)

[Hình 26 Kiểm tra lại đã có file backup 88](#_Toc183782906)

[Hình 27 Thông tin của server cloud vừa tạo 90](#_Toc183782907)

[Hình 28 Màn hình đăng nhập vào server cloud 91](#_Toc183782908)

[Hình 29 Màn hình kết quả sau khi kết nối database 91](#_Toc183782909)

[Hình 30 Chọn Tasks -> Deploy Database to Microsoft Azure SQL Database 92](#_Toc183782910)

[Hình 31 Xuất hiện cửa sổ Deploy Database -> chọn Next 93](#_Toc183782911)

[Hình 32 Nhập tên Database mới và chọn ổ đĩa lưu file tạm thời -> nhấn Next 94](#_Toc183782912)

[Hình 33 Sau khi finish và deploy thành công database từ máy lên Microsoft Azure 95](#_Toc183782913)

[Hình 34 Tạo view ảo 96](#_Toc183782914)

[Hình 35 Chọn thuộc tính bảng HopDong 97](#_Toc183782915)

[Hình 36 Chọn thuộc tính bảng HoaDon 98](#_Toc183782916)

[Hình 37 Lưu và đặt tên cho bảng view vừa tạo 99](#_Toc183782917)

[Hình 38 Tạo 2 bảng view thành công 99](#_Toc183782918)

[Hình 39 Kết quả khách thuê thực hiện thủ tục 100](#_Toc183782919)

# DANH MỤC BẢNG BIỂU

[Bảng 1: Bảng HoaDon 3](#_Toc183782920)

[Bảng 2: Bảng TaiKhoan 3](#_Toc183782921)

[Bảng 3: Bảng KhachThue 4](#_Toc183782922)

[Bảng 4: Bảng HopDong 4](#_Toc183782923)

[Bảng 5: Bảng PhongTro 5](#_Toc183782924)

[Bảng 6: Bảng DichVu 5](#_Toc183782925)

[Bảng 7: Bảng HoaDonChiTiet 6](#_Toc183782926)

[Bảng 9: Module thêm mới phòng trọ 16](#_Toc183782927)

[Bảng 10: Module cập nhật thông tin phòng trọ 19](#_Toc183782928)

[Bảng 11: Module xóa thông tin phòng trọ 21](#_Toc183782929)

[Bảng 12: Module tìm thông tin phòng trọ 23](#_Toc183782930)

[Bảng 13: Module lọc phòng theo trạng thái 25](#_Toc183782931)

[Bảng 15: Module thêm mới hợp đồng 28](#_Toc183782932)

[Bảng 16: Module cập nhật thông tin hợp đồng 31](#_Toc183782933)

[Bảng 17: Module xóa thông tin hợp đồng 33](#_Toc183782934)

[Bảng 18: Module tìm kiếm hợp đồng theo mã 34](#_Toc183782935)

[Bảng 19: Trigger Thêm khách thuê 36](#_Toc183782936)

[Bảng 20: Trigger Xóa khách thuê 39](#_Toc183782937)

[Bảng 22: Module thêm mới khách thuê 42](#_Toc183782938)

[Bảng 23: Module cập nhật khách thuê 45](#_Toc183782939)

[Bảng 24: Module xóa khách thuê 46](#_Toc183782940)

[Bảng 25: Module Trigger\_Tính tổng tiền hóa đơn 49](#_Toc183782941)

[Bảng 26; Module xem hóa đơn theo phòng 50](#_Toc183782942)

[Bảng 27: Module xóa hóa đơn 51](#_Toc183782943)

[Bảng 28: Module tạo hóa đơn 53](#_Toc183782944)

[Bảng 29: Module sửa hóa đơn 54](#_Toc183782945)

[Bảng 31: Module thêm dịch vụ mới 56](#_Toc183782946)

[Bảng 32: Module cập nhật lại thông tin của dịch vụ 57](#_Toc183782947)

[Bảng 33: Module xóa dịch vụ 58](#_Toc183782948)

[Bảng 34: Trigger Cập Nhật Thành Tiền 61](#_Toc183782949)

[Bảng 36: Module Tạo hóa đơn chi tiết 63](#_Toc183782950)

[Bảng 37: Module Sửa hóa đơn chi tiết 65](#_Toc183782951)

[Bảng 38: Module Xóa hóa đơn chi tiết 67](#_Toc183782952)

[Bảng 40: Mudule thêm tài khoản 69](#_Toc183782953)

[Bảng 41: Module Sửa tài khoản 71](#_Toc183782954)

[Bảng 42: Module Xóa tài khoản 73](#_Toc183782955)

# Thu thập dữ liệu

Nhóm 1 thừa kế từ bài tập môn phân tích và thiết kế hệ thống thông tin

# Chuẩn hóa các bảng:

### Chuẩn hoá dữ liệu.

- HoaDon (**MaHoaDon**, SoChuDien, TienDien, TienNuoc, TongTienThanhToan, NgayLapHoaDon, TrangThaiThanhToan, *MaPhong*)

- KhachThue ( **CCCD**, TenKhachThue, NgaySinh, SoDienThoai, GioiTinh, QueQuan, *MaPhong*)

- TaiKhoan (**TenDangNhap**, MatKhau, ChucVu*)*

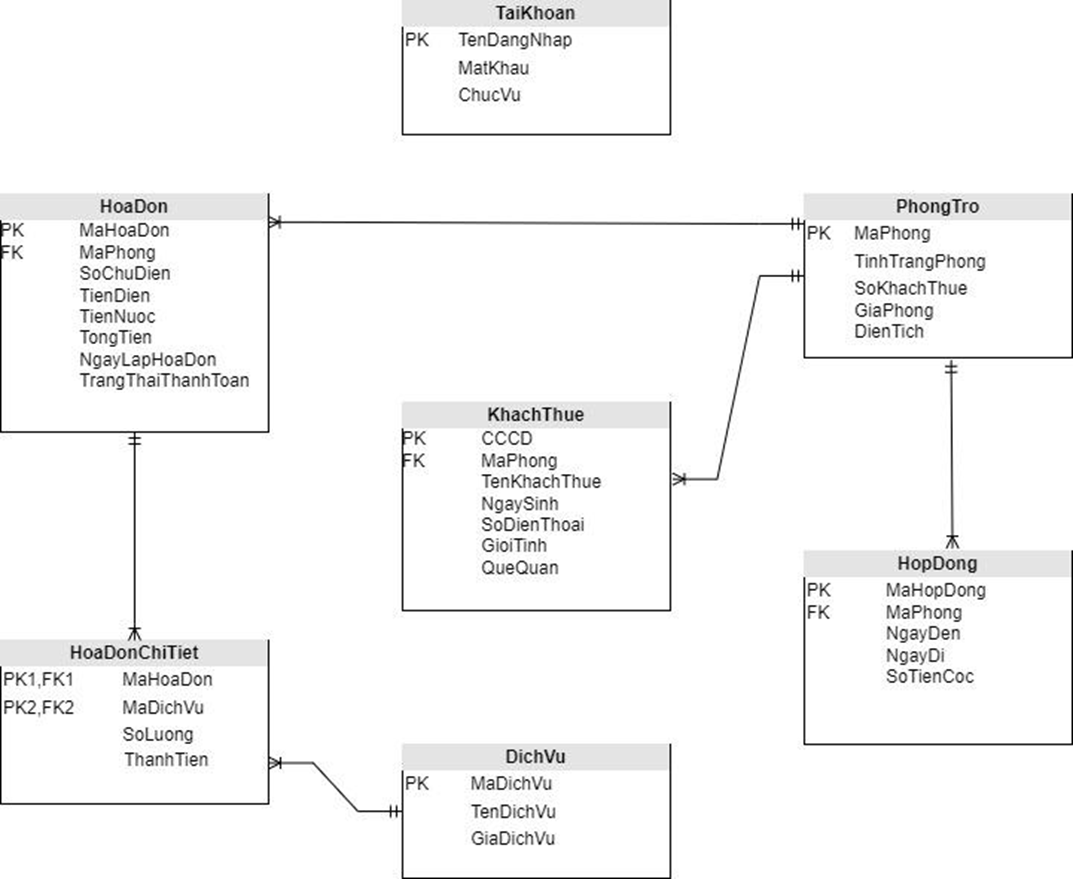
- PhongTro (**MaPhong,** TinhTrangPhong, SoKhachThue, GiaPhong, DienTich)

- HopDong (**MaHopDong**, NgayDen, NgayDi, SoTienCoc, *MaPhong*)

- HoaDonChiTiet (**MaHoaDon, MaDichVu**, SoLuong, ThanhTien)

- DichVu (**MaDichVu**, TenDichVu, GiaDichVu)

**Sơ đồ quan hệ:**



Hình 1 Sơ đồ quan hệ

### Mô tả chi tiết các bảng:

**Bảng HoaDon**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| 1 | MaHoaDon | Varchar(10) | Primary key | Mã hóa đơn |
| 2 | SoChuDien | Int | Not null | Số chữ điện |
| 3 | TienDien | Float | Not null | Tiền điện |
| 4 | TienNuoc | Float | Not null | Tiền nước |
| 5 | TongTien | Float | Not null | Tổng tiền thanh toán |
| 6 | NgayLapHoaDon | Date | Not null | Ngày lập hóa đơn |
| 7 | TrangThaiThanhToan | Nvarchar(30) | Not null | Trạng thái thanh toán |
| 8 | MaPhong | Varchar(20) | Foreign Key | Mã phòng |

Bảng 1: Bảng HoaDon

**Bảng TaiKhoan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| 1 | TenDangNhap | Varchar(10) | Primary Key | Tên đăng nhập |
| 2 | MatKhau | Varchar(16) | Not null | Mật khẩu |
| 3 | ChucVu | Nvarchar(20) | Not null | Chức vụ |

*Bảng 2: Bảng TaiKhoan*

**Bảng KhachThue:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| 1 | CCCD | Varchar(20) | Primary key | Căn cước công dân |
| 2 | TenKhachThue | Nvarchar(50) | Not null | Tên khách thuê |
| 3 | NgaySinh | Date | Not null | Ngày sinh |
| 4 | SoDienThoai | Varchar(20) | Not null | Số điện thoại |
| 5 | GioiTinh | Varchar(10) | Not null | Giới tính |
| 6 | QueQuan | Nvarchar(50) | Not null | Quê quán |
| 7 | MaPhong | Varchar(20) | Foreign Key | Mã phòng |

*Bảng 3: Bảng KhachThue*

**Bảng HopDong:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| 1 | MaHopDong | Varchar(20) | Primary key | Mã hợp đồng |
| 2 | NgayDen | Date | Not null | Ngày đến |
| 3 | NgayDi | Date | Not null | Ngày đi |
| 4 | SoTienCoc | Money | Not null | Số tiền cọc |
| 5 | MaPhong | Varchar(20) | Foreign key | Mã phòng |

Bảng 4: Bảng HopDong

**Bảng PhongTro:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| 1 | MaPhong | Varchar(20) | Primary key | Mã phòng |
| 2 | TinhTrangPhong | Varchar(10) | Not null | Tình trạng phòng |
| 3 | SoKhachThue | Int | Not null | Số lượng khách thuê |
| 4 | GiaPhong | Float | Not null | Giá phòng |
| 5 | DienTich | Float | Not null | Diện tích phòng |

Bảng 5: Bảng PhongTro

**Bảng DichVu:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| 1 | MaDichVu | Varchar(20) | Primary Key | Mã dịch vụ |
| 2 | TenDichVu | Nvarchar(50) | Not null | Tên dịch vụ |
| 3 | GiaDichVu | Money | Not null | Giá dịch vụ |

Bảng 6: Bảng DichVu

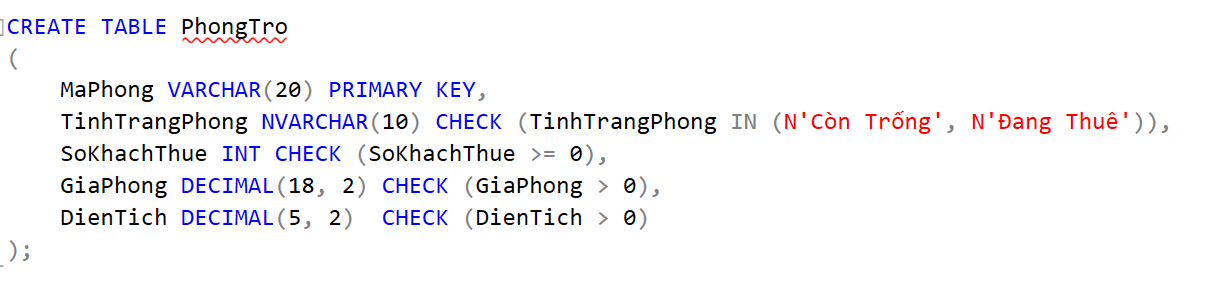
**Bảng HoaDonChiTiet:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên** | **Kiểu dữ liệu** | **Ràng buộc** | **Mô tả** |
| 1 | MaHoaDon | Varchar(10) | Primary Key, Foreign Key | Mã hóa đơn |
| 2 | MaDichVu | Varchar(20) | Primary Key, Foreign Key | Mã dịch vụ |
| 3 | SoLuong | Int | Not null | Số lượng |
| 4 | ThanhTien | Money | Not null | Thành tiền |

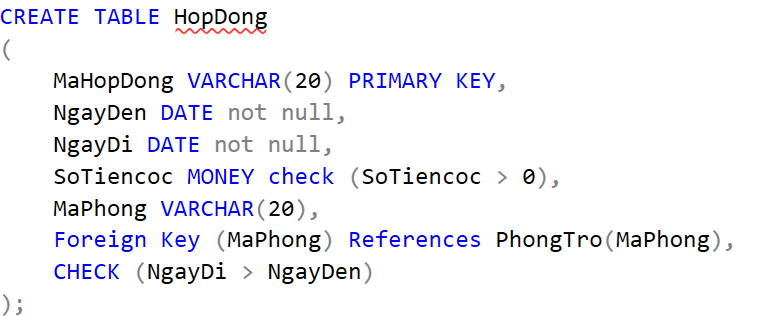
Bảng 7: Bảng HoaDonChiTiet

# Xây dựng cơ sở dữ liệu.

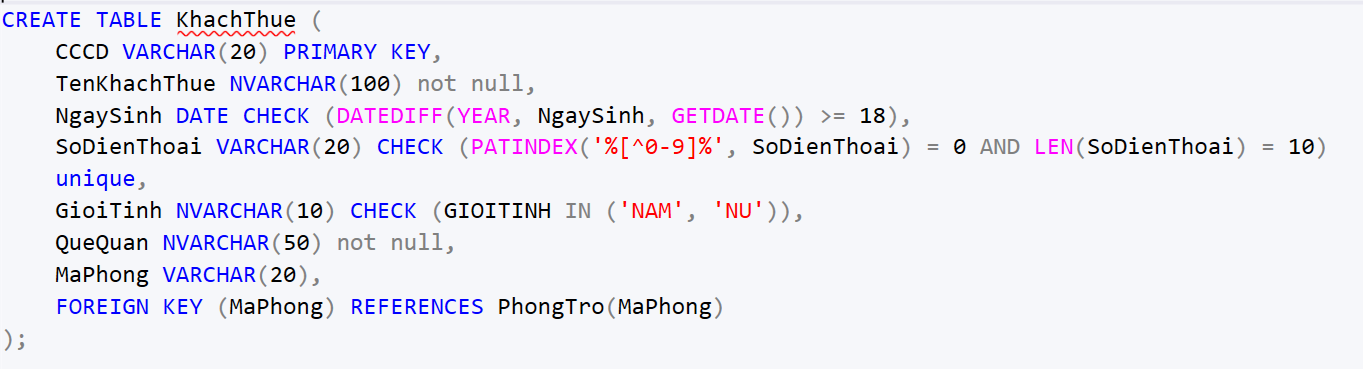
### Tạo bảng phòng trọ

****

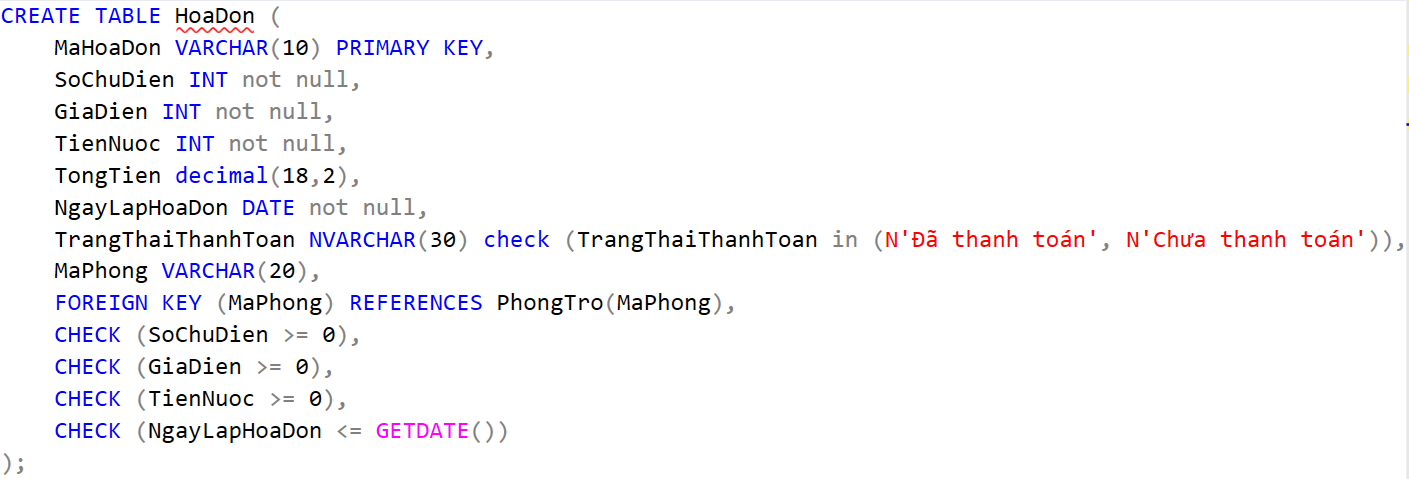
### Tạo bảng hợp đồng

****

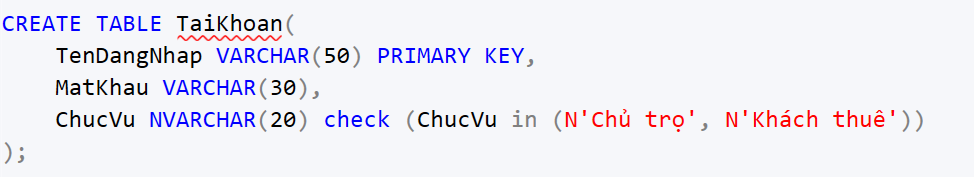
### Tạo bảng khách thuê

****

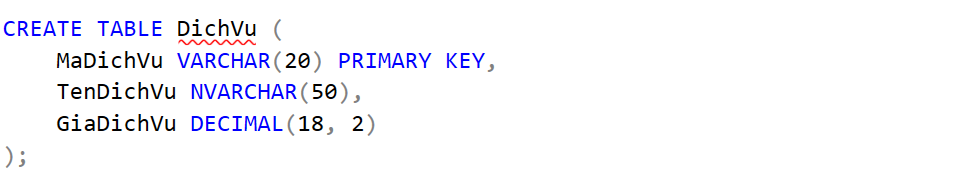
### Tạo bảng hoá đơn

****

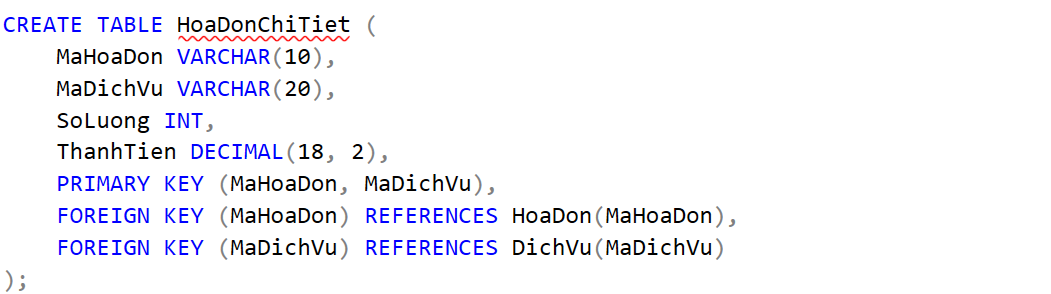
### Tạo bảng tài khoản

****

### Tạo bảng dịch vụ

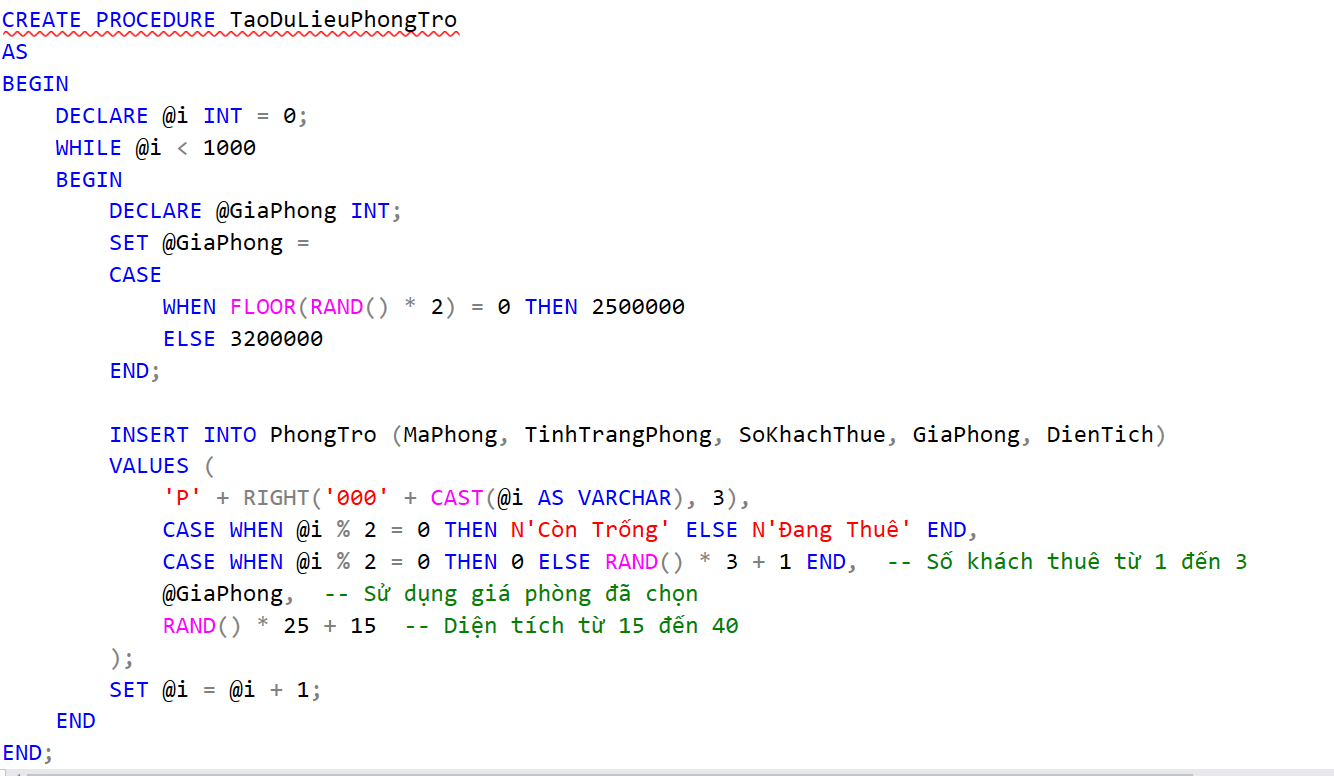
****

### Tạo bảng hoá đơn chi tiết

****

# Xây dựng module tạo dữ liệu dump cho các bảng

### Tạo dữ liệu cho bảng phòng trọ

****

* **Kết quả:**

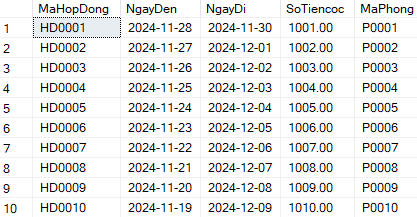


Hình 2 Kết quả tạo dữ liệu cho bảng phòng trọ

### Tạo dữ liệu cho bảng hợp đồng

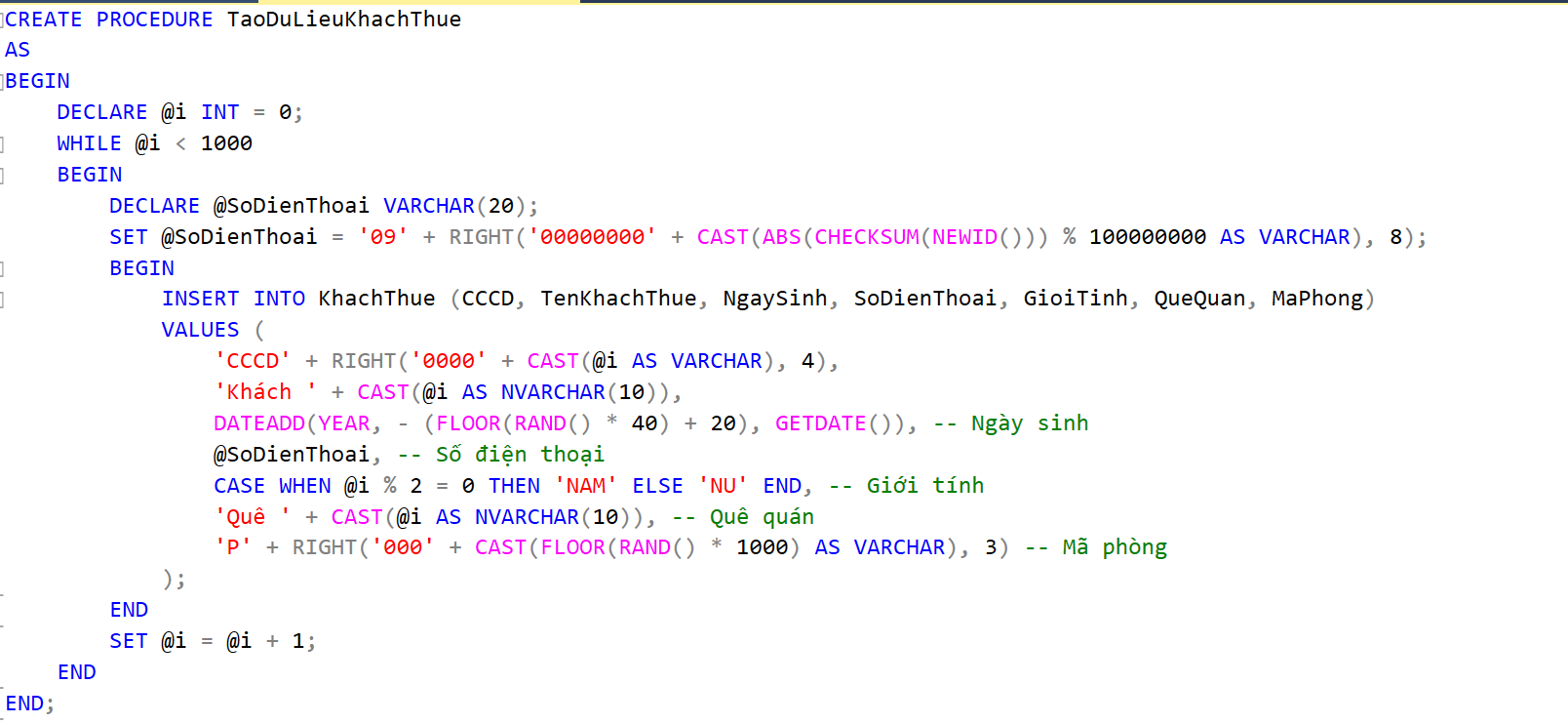
****

* **Kết quả:**



Hình 3 Kết quả tạo dữ liệu cho bảng hợp đồng

### Tạo dữ liệu cho bảng khách thuê

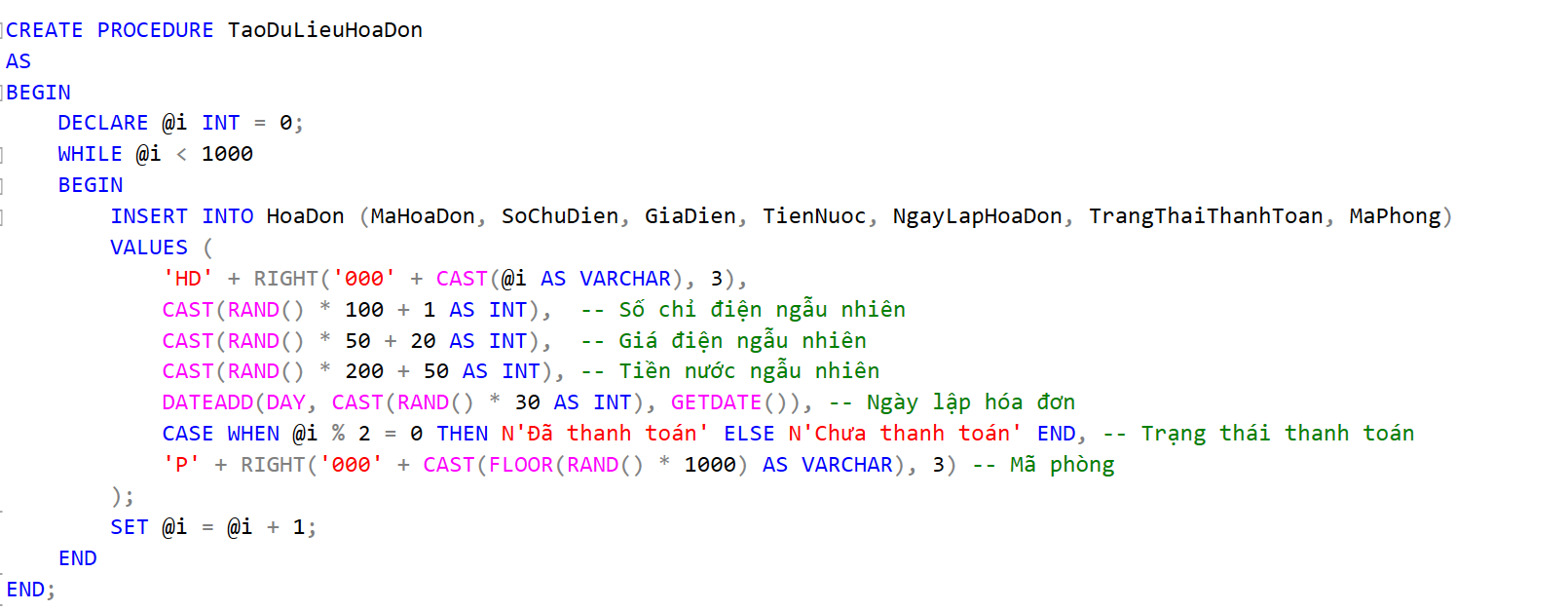
****

* **Kết quả:**



Hình 4 Kết quả tạo dữ liệu cho bảng khách thuê

### Tạo dữ liệu cho bảng hoá đơn

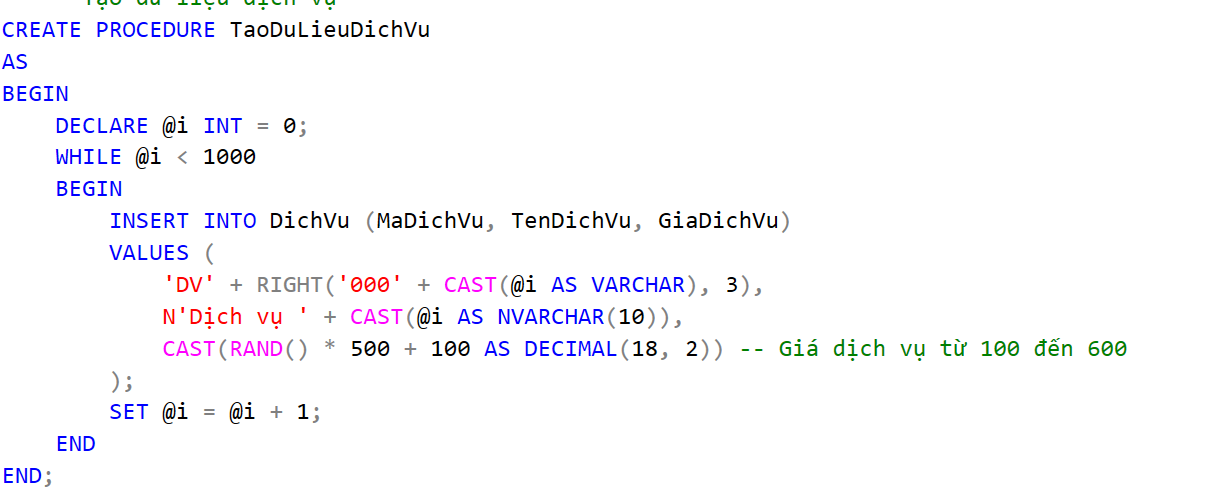
****

**-Kết quả:**

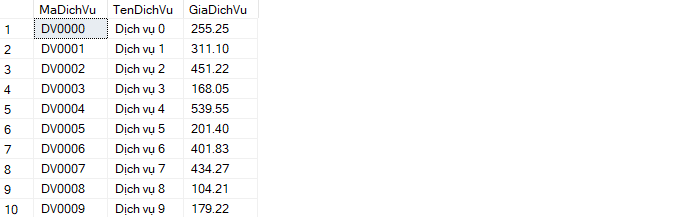


Hình 5 Kết quả tạo dữ liệu cho bảng hoá đơn

### Tạo dữ liệu cho bảng dịch vụ

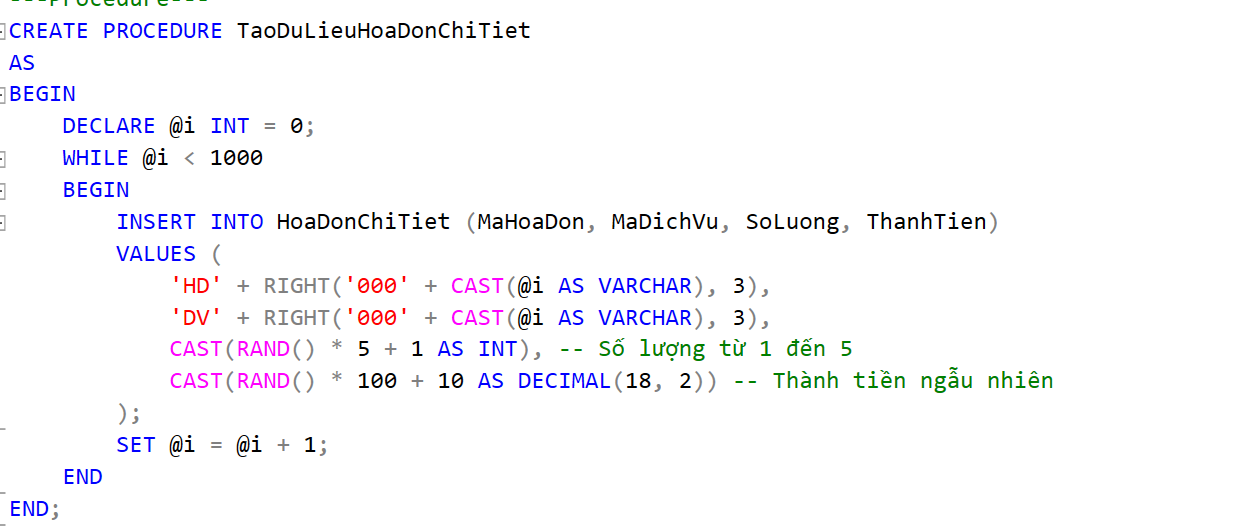
****

* **Kết quả:**

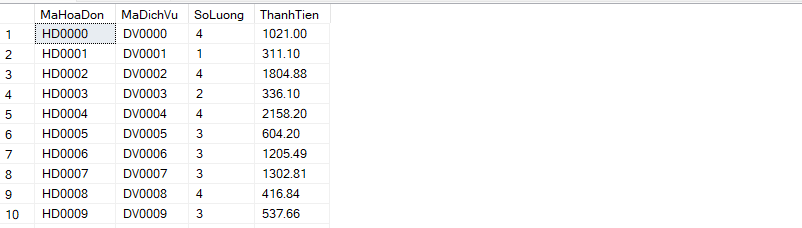


Hình 6 Kết quả tạo dữ liệu cho bảng dịch vụ

### Tạo dữ liệu cho bảng hoá đơn chi tiết

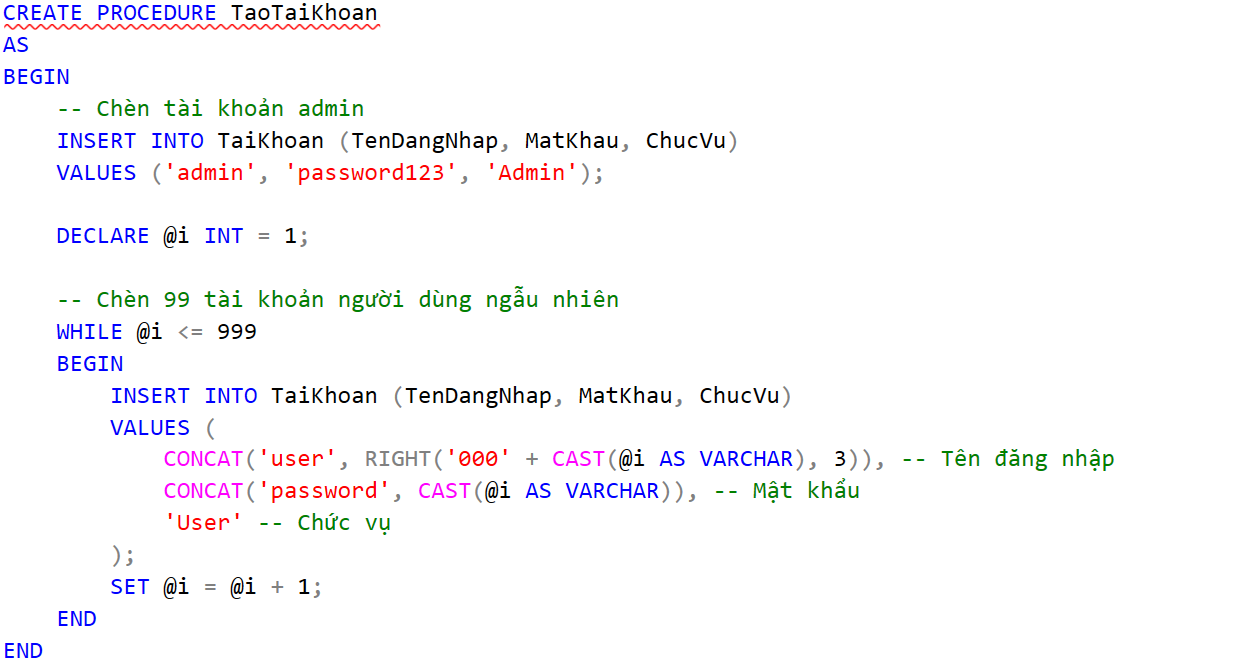
****

* **Kết quả:**

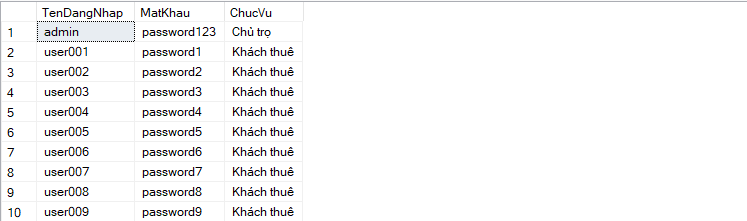


Hình 7 Kết quả tạo dữ liệu cho bảng hoá đơn chi tiết

### Tạo dữ liệu cho bảng tài khoản

****

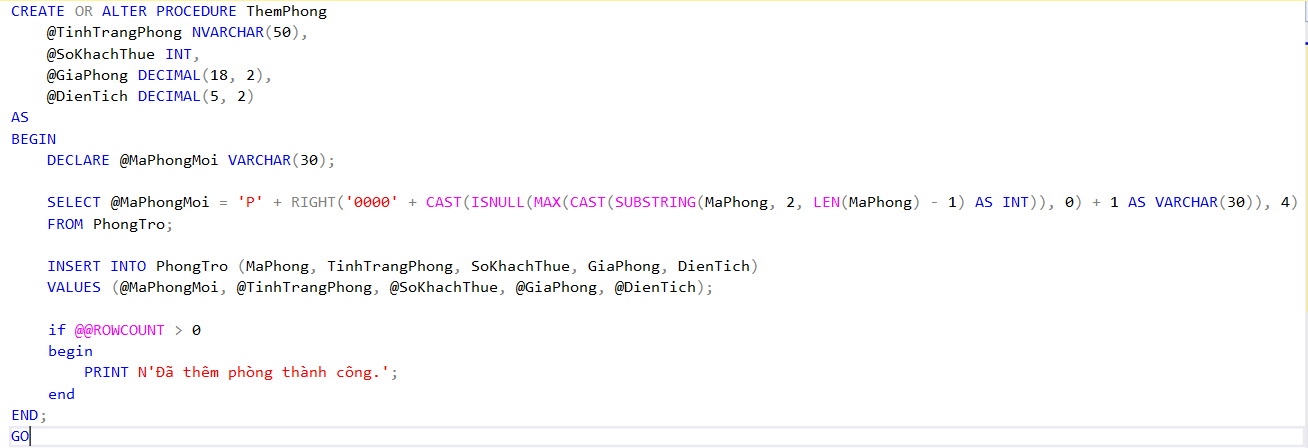
**- Kết quả:**



Hình 8 Kết quả tạo dữ liệu cho bảng tài khoản

# Mô tả các module trong cơ sở dữ liệu

## Module thêm mới phòng trọ:

****

|  |  |
| --- | --- |
| **STT** | 1. |
| **Tên** | Thêm mới phòng trọ |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục ThemPhong được sử dụng để thêm một phòng mới vào bảng PhongTro. Nó tự động tạo mã phòng dựa trên giá trị hiện có trong bảng và thêm các thông tin về tình trạng phòng, số khách thuê, giá phòng, và diện tích. |
| **Input** | **@TinhTrangPhong (NVARCHAR(50)):** Tình trạng hiện tại của phòng (ví dụ: "Trống", "Đã thuê").  **@SoKhachThue (INT):** Số lượng khách thuê tối đa mà phòng có thể chứa.  **@GiaPhong (DECIMAL(18, 2)):** Giá của phòng**.**  **@DienTich (DECIMAL(5, 2)):** Diện tích của phòng. |
| **Output** | * Kết quả thông báo thêm phòng thành công nếu quá trình thêm phòng hoàn tất. |
| **Process** | 1. **Xác định mã phòng mới (@MaPhongMoi):**   * Lấy mã phòng cao nhất hiện có trong bảng PhongTro, sau đó thêm 1 để tạo ra mã phòng mới. * Mã phòng mới có dạng 'P' kèm theo 4 chữ số (ví dụ: 'P0001', 'P0002').   2. **Chèn dữ liệu vào bảng PhongTro:**   * Thực hiện chèn mã phòng vừa tạo cùng với các thông tin về tình trạng phòng, số khách thuê, giá phòng và diện tích vào bảng.   3. **Kiểm tra kết quả:**   * Nếu câu lệnh INSERT thành công (với số hàng được thêm lớn hơn 0), in ra thông báo "Đã thêm phòng thành công." |

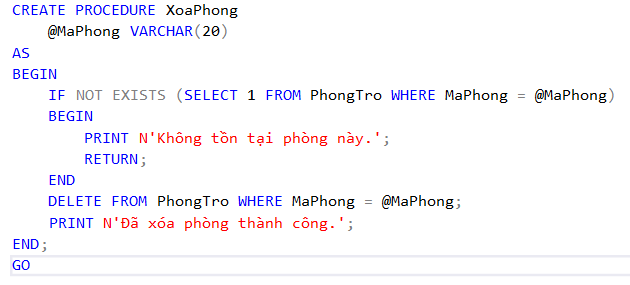
Bảng 9: Module thêm mới phòng trọ

## Module Cập nhật thông tin phòng trọ

|  |  |
| --- | --- |
| **STT** | 2. |
| **Tên** | Cập nhật thông tin phòng trọ |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục CapNhapThongTinPhong dùng để cập nhật thông tin của một phòng trong bảng PhongTro dựa trên mã phòng. Nó kiểm tra xem mã phòng có tồn tại hay không trước khi thực hiện cập nhật. |
| **Input** | **@MaPhong (VARCHAR(20)):** Mã của phòng cần cập nhật.  **@TinhTrangPhong (NVARCHAR(50)):** Tình trạng mới của phòng (ví dụ: "Trống", "Đã thuê").  **@SoKhachThue (INT):** Số lượng khách thuê tối đa cập nhật cho phòng.  **@GiaPhong (DECIMAL(18, 2)):** Giá phòng mới.  **@DienTich (DECIMAL(5, 2)):** Diện tích phòng cập nhật. |
| **Output** | Thông báo về kết quả:   * "Không tồn tại phòng này" nếu mã phòng không có trong hệ thống. * "Đã cập nhật thông tin thành công" nếu thông tin phòng được cập nhật thành công. |
| **Process** | 1. **Kiểm tra sự tồn tại của phòng:**   * Sử dụng SELECT 1 để kiểm tra xem có bất kỳ dòng nào tồn tại trong bảng PhongTro với mã phòng @MaPhong. * Nếu không tồn tại, in ra thông báo "Không tồn tại phòng này" và kết thúc thủ tục.   2. **Cập nhật thông tin phòng:**   * Nếu phòng tồn tại, thực hiện câu lệnh UPDATE để thay đổi các thông tin như tình trạng phòng, số khách thuê, giá phòng, và diện tích dựa trên các giá trị đầu vào.   3. **Thông báo kết quả:**   * In ra thông báo "Đã cập nhật thông tin thành công" sau khi quá trình cập nhật hoàn tất. |

Bảng 10: Module cập nhật thông tin phòng trọ

## 5.3 Module Xóa thông tin phòng trọ

****

|  |  |
| --- | --- |
| **STT** | 3. |
| **Tên** | Xóa thông tin phòng trọ |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục XoaPhong được sử dụng để xóa một phòng trong bảng PhongTro dựa trên mã phòng. Thủ tục này kiểm tra xem mã phòng có tồn tại trước khi thực hiện việc xóa. |
| **Input** | **@MaPhong (VARCHAR(20)):** Mã của phòng cần xóa. |
| **Output** | Thông báo về kết quả:   * "Không tồn tại phòng này" nếu mã phòng không có trong bảng. * "Đã xóa phòng thành công" nếu phòng đã được xóa thành công. |
| **Process** | 1. **Kiểm tra sự tồn tại của phòng:**   * Thực hiện câu lệnh SELECT 1 để kiểm tra xem có bất kỳ dòng nào trong bảng PhongTro khớp với mã phòng @MaPhong. * Nếu không tìm thấy phòng với mã tương ứng, in ra thông báo "Không tồn tại phòng này" và dừng thủ tục.   2. **Xóa phòng:**   * Nếu phòng tồn tại, thực hiện câu lệnh DELETE để xóa dòng chứa mã phòng @MaPhong khỏi bảng PhongTro.   3. **Thông báo kết quả:**   * Sau khi thực hiện xóa, in ra thông báo "Đã xóa phòng thành công". |

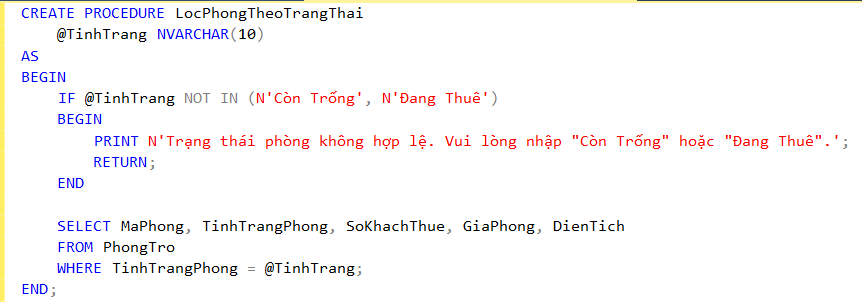
Bảng 11: Module xóa thông tin phòng trọ

## Module Tìm thông tim phòng trọ

|  |  |
| --- | --- |
| **STT** | 4. |
| **Tên** | Tìm thông tin phòng trọ |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục TimKiemThongTinPhong được sử dụng để tìm kiếm và trả về thông tin của một phòng trong bảng PhongTro dựa trên mã phòng. |
| **Input** | **@MaPhong (VARCHAR(20)):** Mã của phòng cần tìm kiếm. |
| **Output** | 1. Thông tin chi tiết về phòng (nếu tồn tại), bao gồm tất cả các cột trong bảng PhongTro.  2. Thông báo "Không tồn tại phòng này" nếu phòng không được tìm thấy. |
| **Process** | 1. **Kiểm tra sự tồn tại của phòng:**   * Sử dụng câu lệnh SELECT 1 để kiểm tra xem có phòng nào trong bảng PhongTro có mã khớp với @MaPhong không. * Nếu không tìm thấy phòng, in ra thông báo "Không tồn tại phòng này" và dừng thủ tục.   2. **Truy xuất thông tin phòng:**   * Nếu phòng tồn tại, thực hiện câu lệnh SELECT \* để lấy tất cả các thông tin của phòng từ bảng PhongTro, bao gồm mã phòng, tình trạng phòng, số khách thuê, giá phòng, và diện tích. |

Bảng 12: Module tìm thông tin phòng trọ

## 5.5 Module Lọc phòng theo trạng thái

****

|  |  |
| --- | --- |
| **STT** | 5. |
| **Tên** | Lọc phòng theo trạng thái |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục LocPhongTheoTrangThai được sử dụng để lọc và truy xuất thông tin các phòng trong bảng PhongTro dựa trên tình trạng phòng (còn trống hoặc đang thuê). |
| **Input** | **@TinhTrang (NVARCHAR(10**)): Tình trạng phòng cần lọc (có thể là "Còn Trống" hoặc "Đang Thuê"). |
| **Output** | 1. Thông tin về các phòng có trạng thái tương ứng, bao gồm: mã phòng, tình trạng phòng, số khách thuê, giá phòng, và diện tích.  2. Thông báo lỗi nếu giá trị nhập vào cho trạng thái không hợp lệ. |
| **Process** | 1. **Kiểm tra tính hợp lệ của tình trạng phòng:**   * Thủ tục kiểm tra giá trị @TinhTrang có phải là "Còn Trống" hoặc "Đang Thuê" không. Nếu không đúng, in ra thông báo lỗi: "Trạng thái phòng không hợp lệ. Vui lòng nhập 'Còn Trống' hoặc 'Đang Thuê'" và dừng thủ tục.   2. **Truy xuất danh sách phòng:**   * Nếu giá trị trạng thái hợp lệ, thực hiện câu lệnh SELECT để lấy thông tin các phòng có trạng thái phù hợp trong bảng PhongTro, bao gồm mã phòng, tình trạng phòng, số khách thuê, giá phòng và diện tích.   3. **Không có thông báo bổ sung:**   * Nếu kết quả tìm kiếm không có, truy vấn vẫn sẽ trả về tập rỗng. |

Bảng 13: Module lọc phòng theo trạng thái

## Module Thêm mới hợp đồng

|  |  |
| --- | --- |
| **STT** | 6. |
| **Tên** | Thêm mới hợp đồng |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục này dùng để thêm mới một hợp đồng vào bảng HopDong, với điều kiện rằng phòng cho thuê phải tồn tại và ngày đi phải lớn hơn ngày đến |
| **Input** | **@MaHopDong**: Mã hợp đồng (kiểu VARCHAR(20)), định danh cho hợp đồng cần thêm mới.  **@NgayDen**: Ngày đến (kiểu DATE), ngày bắt đầu thuê phòng.  **@NgayDi**: Ngày đi (kiểu DATE), ngày kết thúc thuê phòng.  **@SoTiencoc**: Số tiền cọc (kiểu MONEY), tiền cọc thuê phòng.  **@MaPhong**: Mã phòng (kiểu VARCHAR(20)), mã phòng mà hợp đồng thuê phòng liên quan đến. |
| **Output** | In ra thông báo tương ứng:   * "Thêm mới hợp đồng thành công." nếu hợp đồng được thêm mới thành công. * "Mã phòng không tồn tại." nếu mã phòng không hợp lệ (không tồn tại trong bảng PhongTro). * "Ngày đi phải lớn hơn ngày đến." nếu giá trị ngày đi không lớn hơn ngày đến |
| **Process** | 1.Kiểm tra mã phòng có tổn tại trong bảng phongtro  1.1.nếu có, kiểm tra @ngaydi < @ngayden  1.1.1.Đúng, in ra thông báo "Ngày đi phải lớn hơn ngày đến."  1.1.1.Sai, chuyển đến bước 2  1.2.nếu không, in ra thông báo 'Mã phòng không tồn tại.'  2. Thực hiện việc thêm mới hợp đồng |

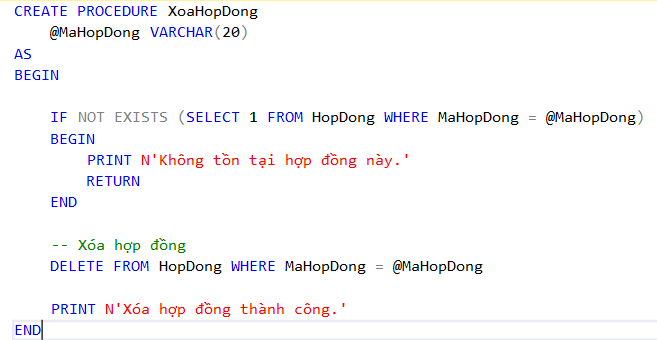
Bảng 15: Module thêm mới hợp đồng

## 5.7 Module Cập nhật thông tin hợp đồng

|  |  |
| --- | --- |
| **STT** | 7. |
| **Tên** | Cập nhật thông tin hợp đồng |
| **Loại** | Procedure |
| **Mục đích** | Dùng khi muốn thay đổi các thông tin liên quan trong bảng hợp đồng |
| **Input** | **@MaHopDong (VARCHAR(20)):** Mã của hợp đồng cần cập nhật.  **@NgayDen (DATE):** Ngày bắt đầu thuê phòng.  **@NgayDi (DATE):** Ngày kết thúc thuê phòng.  **@SoTiencoc (MONEY):** Số tiền đặt cọc.  **@MaPhong (VARCHAR(20)):** Mã phòng liên quan đến hợp đồng. |
| **Output** | - Thông báo về kết quả:   * "Không tồn tại hợp đồng này" nếu mã hợp đồng không có trong bảng. * "Ngày đi phải lớn hơn ngày đến" nếu ngày đi không hợp lệ. * "Đã cập nhật thông tin hợp đồng thành công" nếu cập nhật thành công. |
| **Process** | 1. **Kiểm tra sự tồn tại của hợp đồng:**   * Thực hiện câu lệnh SELECT 1 để kiểm tra xem hợp đồng với mã @MaHopDong có tồn tại hay không. * Nếu không tồn tại, in ra thông báo "Không tồn tại hợp đồng này" và dừng thủ tục.   2. **Kiểm tra tính hợp lệ của ngày đi và ngày đến:**   * So sánh giá trị @NgayDi và @NgayDen. Nếu @NgayDi nhỏ hơn hoặc bằng @NgayDen, in ra thông báo "Ngày đi phải lớn hơn ngày đến" và dừng thủ tục.   3. **Cập nhật thông tin hợp đồng:**   * Nếu các điều kiện trên đều hợp lệ, thực hiện câu lệnh UPDATE để cập nhật các thông tin như ngày đến, ngày đi, số tiền đặt cọc, và mã phòng trong bảng HopDong dựa trên mã hợp đồng @MaHopDong.   4. **Thông báo kết quả:**   * Sau khi cập nhật thành công, in ra thông báo "Đã cập nhật thông tin hợp đồng thành công". |

Bảng 16: Module cập nhật thông tin hợp đồng

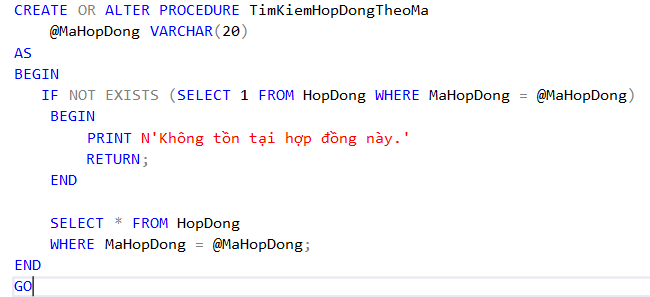
## 5.8 Module Xóa hợp đồng



|  |  |
| --- | --- |
| **STT** | 8. |
| **Tên** | Xóa thông tin hợp đồng |
| **Loại** | Procedure |
| **Mục đích** | Dùng khi muốn xóa thông tin một hợp đồng nào đó trong bảng hợp đồng |
| **Input** | @mahopdong varchar(20) |
| **Output** | In ra thông báo tương ứng:   * "Xóa hợp đồng thành công." nếu hợp đồng được xóa thành công. * "Không tồn tại hợp đồng này." nếu mã hợp đồng không có trong bảng HopDong. |
| **Process** | 1. Kiểm tra mã @MaHopDong có tồn tại trong bảng HopDong hay không  1.1.nếu không, in ra thông báo 'Không tồn tại hợp đồng này.  1.2. Nếu có, chuyển đến bước 2  2. Thực hiện lệnh DELETE để xóa hợp đồng từ bảng HopDong dựa trên giá trị của @MaHopDong. |

Bảng 17: Module xóa thông tin hợp đồng

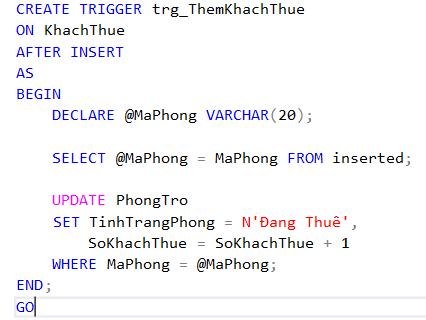
## 5.9 Module Tìm kiếm hợp đồng theo mã hợp đồng



|  |  |
| --- | --- |
| **STT** | 9. |
| **Tên** | Tìm kiếm hợp đồng theo mã |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục này được sử dụng để tìm kiếm thông tin chi tiết của một hợp đồng dựa trên mã hợp đồng cung cấp. Nếu không tìm thấy hợp đồng nào khớp với mã này, thủ tục sẽ thông báo rằng không có hợp đồng nào được tìm thấy |
| **Input** | @mahopdong varchar(20) |
| **Output** | Thông tin chi tiết về hợp đồng (nếu có hợp đồng khớp với mã hợp đồng).  Thông báo ‘’Không tìm thấy hợp đồng’’ này nếu không có hợp đồng nào khớp với mã được cung cấp. |
| **Process** | 1.Truy vấn bảng HopDong để tìm kiếm bản ghi với điều kiện mã hợp đồng trùng với @MaHopDong.  2.Nếu không có bản ghi nào trả về, sử dụng câu lệnh IF @@ROWCOUNT = 0 để kiểm tra số lượng bản ghi. Nếu bằng 0, thủ tục sẽ in ra thông báo "Không tìm thấy hợp đồng này" |

Bảng 18: Module tìm kiếm hợp đồng theo mã

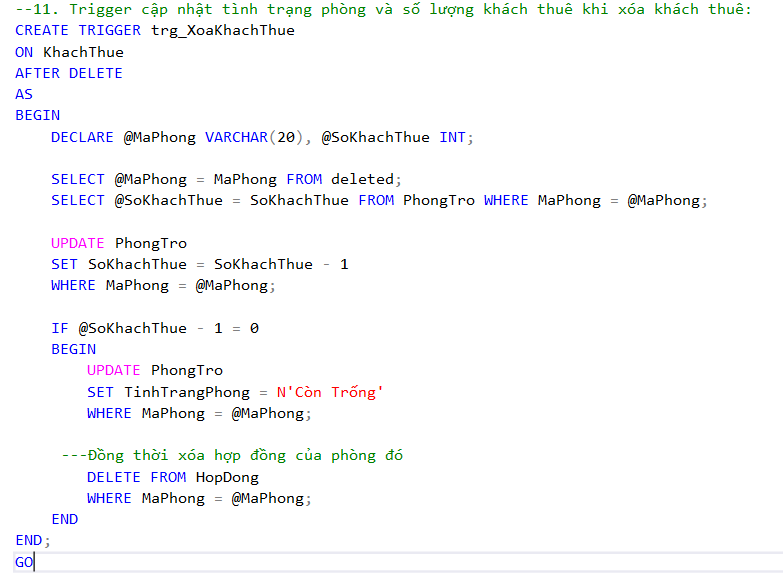
## 5.10 Trigger Thêm khách thuê



|  |  |
| --- | --- |
| **STT** | 10. |
| **Tên** | Trigger\_Thêm khách thuê |
| **Loại** | Trigger |
| **Mục đích** | Trigger trg\_ThemKhachThue được sử dụng để tự động cập nhật tình trạng phòng và số lượng khách thuê trong bảng PhongTro khi có một khách thuê mới được thêm vào bảng KhachThue. |
| **Input** | Trigger được kích hoạt sau khi thực hiện câu lệnh INSERT vào bảng KhachThue. |
| **Output** | Cập nhật tình trạng của phòng thành "Đang Thuê" và tăng số khách thuê trong bảng PhongTro. |
| **Process** | 1. **Lấy mã phòng từ bản ghi vừa được thêm:**   * Sử dụng câu lệnh SELECT @MaPhong = MaPhong FROM inserted để lấy mã phòng từ bảng tạm inserted chứa bản ghi mới được thêm vào bảng KhachThue.   2. **Cập nhật bảng PhongTro:**   * Câu lệnh UPDATE được thực hiện trên bảng PhongTro, nơi tình trạng phòng (TinhTrangPhong) được đặt thành "Đang Thuê" và số lượng khách thuê (SoKhachThue) của phòng có mã @MaPhong được tăng lên 1. |

Bảng 19: Trigger Thêm khách thuê

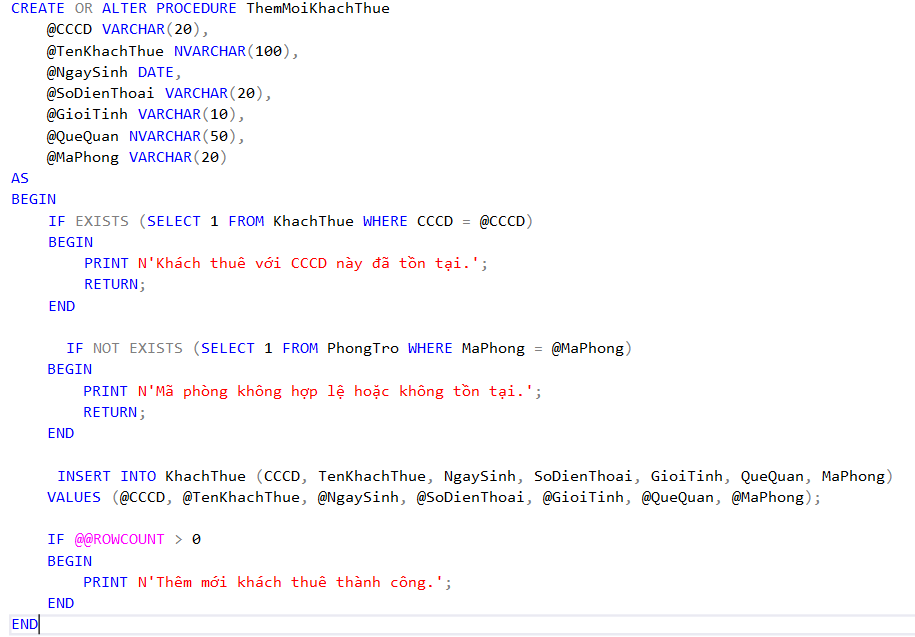
## 5.11 Trigger Cập nhật tình trạng phòng khi xóa khách thuê



|  |  |
| --- | --- |
| **STT** | 11. |
| **Tên** | Trigger\_Xóa khách thuê. |
| **Loại** | Trigger |
| **Mục đích** | Trigger trg\_XoaKhachThue được sử dụng để tự động cập nhật số lượng khách thuê và tình trạng phòng trong bảng PhongTro khi một khách thuê bị xóa khỏi bảng KhachThue. Đồng thời, nếu phòng không còn khách thuê nào, hợp đồng liên quan đến phòng đó cũng sẽ bị xóa. |
| **Input** | Trigger được kích hoạt sau khi thực hiện câu lệnh DELETE trên bảng KhachThue. |
| **Output** | 1. Giảm số lượng khách thuê trong phòng tương ứng.  2. Nếu phòng không còn khách thuê nào, tình trạng phòng được chuyển thành "Còn Trống" và hợp đồng liên quan đến phòng đó bị xóa. |
| **Process** | 1. **Lấy mã phòng từ bản ghi bị xóa:**   * Sử dụng câu lệnh SELECT @MaPhong = MaPhong FROM deleted để lấy mã phòng từ bản ghi bị xóa trong bảng tạm deleted.   2. **Lấy số lượng khách thuê hiện tại:**   * Câu lệnh SELECT @SoKhachThue = SoKhachThue FROM PhongTro WHERE MaPhong = @MaPhong để lấy số khách thuê hiện tại của phòng.   3. **Cập nhật số khách thuê:**   * Thực hiện câu lệnh UPDATE PhongTro SET SoKhachThue = SoKhachThue - 1 WHERE MaPhong = @MaPhong để giảm số lượng khách thuê của phòng đi 1.   4. **Kiểm tra số lượng khách thuê sau khi giảm:**   * Nếu số khách thuê còn lại sau khi giảm là 0 (@SoKhachThue - 1 = 0), trigger sẽ thực hiện:   + Cập nhật tình trạng phòng thành "Còn Trống" với câu lệnh UPDATE PhongTro SET TinhTrangPhong = N'Còn Trống' WHERE MaPhong = @MaPhong.   + Xóa hợp đồng liên quan đến phòng đó bằng câu lệnh DELETE FROM HopDong WHERE MaPhong = @MaPhong. |

Bảng 20: Trigger Xóa khách thuê

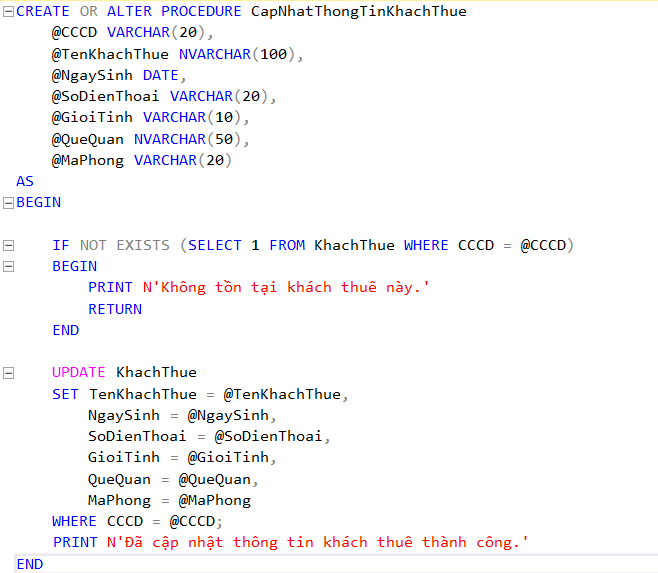
## 5.12 Module Thêm mới khách thuê

****

|  |  |
| --- | --- |
| **STT** | 12. |
| **Tên** | Thêm mới khách thuê |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục này dùng để thêm mới một khách thuê vào hệ thống. Trước khi thêm, thủ tục sẽ kiểm tra xem mã phòng có tồn tại trong bảng PhongTro không. Nếu có, thông tin khách thuê sẽ được thêm vào bảng KhachThue. Nếu không, thủ tục sẽ thông báo rằng mã phòng không tồn tại. |
| **Input** | @CCCD: Căn cước công dân của khách thuê (kiểu VARCHAR(20)).  @TenKhachThue: Tên đầy đủ của khách thuê (kiểu NVARCHAR(100)).  @NgaySinh: Ngày sinh của khách thuê (kiểu DATE).  @SoDienThoai: Số điện thoại của khách thuê (kiểu VARCHAR(20)).  @GioiTinh: Giới tính của khách thuê (kiểu VARCHAR(10)).  @QueQuan: Quê quán của khách thuê (kiểu NVARCHAR(50))  @MaPhong: Mã phòng khách thuê ở (kiểu VARCHAR(20)) |
| **Output** | -Thông báo 'Thêm mới khách thuê thành công' nếu thông tin được thêm vào thành công.  -Thông báo 'Mã phòng không tồn tại' nếu mã phòng không có trong bảng PhongTro. |
| **Process** | 1.Kiểm tra xem mã phòng (@MaPhong) có tồn tại trong bảng PhongTro hay không bằng câu lệnh IF EXISTS.  2.  2.1Nếu mã phòng tồn tại:  2.1.1.Thêm mới một bản ghi vào bảng KhachThue với các thông tin về căn cước công dân, tên khách thuê, ngày sinh, số điện thoại, giới tính, quê quán, và mã phòng.  2.1.2.In ra thông báo "Thêm mới khách thuê thành công".  2.2.Nếu mã phòng không tồn tại:  2.2.1.In ra thông báo "Mã phòng không tồn tại". |

Bảng 22: Module thêm mới khách thuê

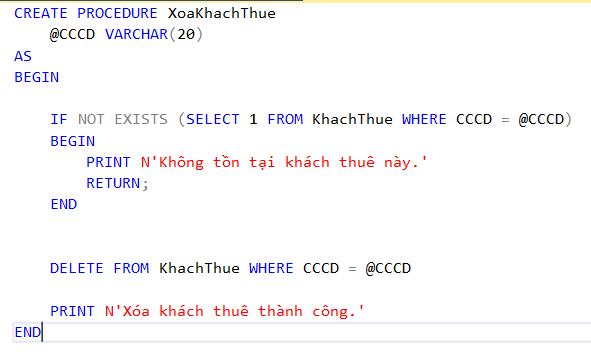
## 5.13 Module Cập nhật thông tin khách thuê



|  |  |
| --- | --- |
| **STT** | 13. |
| **Tên** | Cập nhật khách thuê |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục này được sử dụng để cập nhật thông tin của một khách thuê đã có trong hệ thống, dựa trên căn cước công dân (CCCD). Nếu không tồn tại khách thuê có CCCD như vậy, thủ tục sẽ dừng lại và thông báo rằng không tìm thấy khách thuê này |
| **Input** | @CCCD: Căn cước công dân của khách thuê (kiểu VARCHAR(20)).  @TenKhachThue: Tên đầy đủ của khách thuê (kiểu NVARCHAR(100)).  @NgaySinh: Ngày sinh của khách thuê (kiểu DATE).  @SoDienThoai: Số điện thoại của khách thuê (kiểu VARCHAR(20)).  @GioiTinh: Giới tính của khách thuê (kiểu VARCHAR(10)).  @QueQuan: Quê quán của khách thuê (kiểu NVARCHAR(50)).  @MaPhong: Mã phòng mà khách thuê đang ở (kiểu VARCHAR(20)) |
| **Output** | Thông báo Đã cập nhật thông tin khách thuê thành công nếu cập nhật thành công.  Thông báo Không tồn tại khách thuê này nếu khách thuê không tồn tại trong hệ thống |
| **Process** | 1.Kiểm tra xem khách thuê có tồn tại trong bảng KhachThue dựa trên giá trị @CCCD hay không bằng cách sử dụng câu lệnh IF NOT EXISTS.  1.1Nếu khách thuê không tồn tại, in ra thông báo "Không tồn tại khách thuê này" và dừng thủ tục.  1.2.Nếu tồn tại chuyển đến bước 2  2.Cập nhật thông tin trong bảng KhachThue bao gồm tên, ngày sinh, số điện thoại, giới tính, quê quán, và mã phòng dựa trên CCCD của khách thuê.  3.Sau khi cập nhật, in ra thông báo "Đã cập nhật thông tin khách thuê thành công". |

Bảng 23: Module cập nhật khách thuê

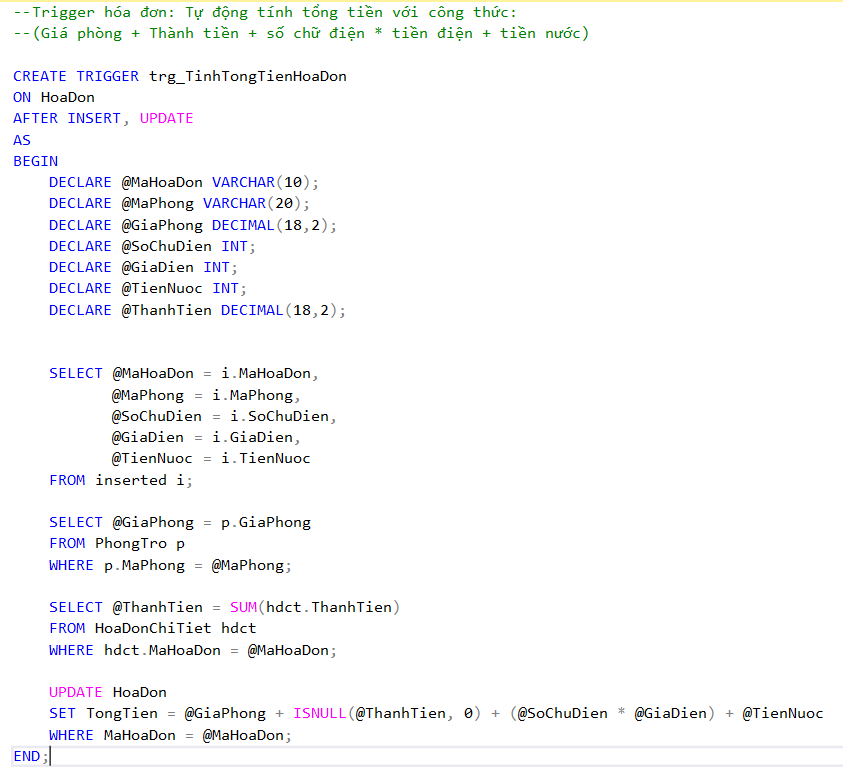
## 5.14 Module Xóa khách thuê



|  |  |
| --- | --- |
| **STT** | 14. |
| **Tên** | Xóa khách thuê |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục này được sử dụng để xóa thông tin của một khách thuê khỏi hệ thống, dựa trên căn cước công dân (CCCD). Nếu khách thuê không tồn tại, thủ tục sẽ dừng lại và thông báo rằng khách thuê này không tồn tại. |
| **Input** | @CCCD: Căn cước công dân của khách thuê (kiểu VARCHAR(20)). |
| **Output** | Thông báo ‘Xóa khách thuê thành công’ nếu khách thuê được xóa thành công.  Thông báo ‘Không tồn tại khách thuê’ này nếu khách thuê không tồn tại trong hệ thống. |
| **Process** | 1. Kiểm tra xem khách thuê có tồn tại trong bảng KhachThue dựa trên giá trị @CCCD hay không bằng cách sử dụng câu lệnh IF NOT EXISTS.  2 .Nếu khách thuê không tồn tại, in ra thông báo "Không tồn tại khách thuê này" và dừng thủ tục.  3.Nếu khách thuê tồn tại, thực hiện xóa khách thuê khỏi bảng KhachThue dựa trên CCCD.  4.Sau khi xóa, in ra thông báo "Xóa khách thuê thành công" |

Bảng 24: Module xóa khách thuê

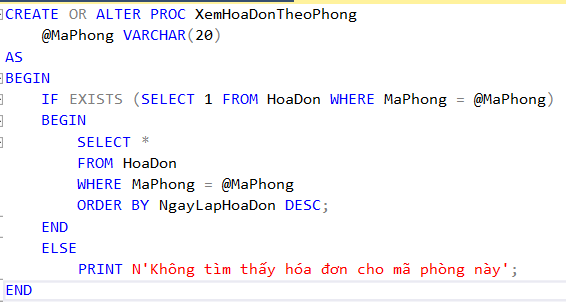
## 5.15 Trigger Tính tổng tiền hóa đơn



|  |  |
| --- | --- |
| **STT** | 15. |
| **Tên** | Trigger\_Tính tổng tiền hóa đơn |
| **Loại** | Trigger |
| **Mục đích** | Trigger trg\_TinhTongTienHoaDon tự động tính tổng số tiền (TongTien) của hóa đơn trong bảng HoaDon sau khi có một hóa đơn mới được thêm vào hoặc được cập nhật. Tổng số tiền bao gồm tiền phòng, tiền điện, tiền nước và các chi phí chi tiết khác từ bảng HoaDonChiTiet. |
| **Input** | Trigger được kích hoạt sau khi thực hiện câu lệnh INSERT hoặc UPDATE trên bảng HoaDon. |
| **Output** | Cập nhật trường TongTien trong bảng HoaDon với tổng số tiền phải trả cho hóa đơn đó. |
| **Process** | 1. **Lấy các thông tin cần thiết từ bản ghi mới thêm hoặc cập nhật:**   * Các giá trị @MaHoaDon, @MaPhong, @SoChuDien, @GiaDien, và @TienNuoc được lấy từ bảng tạm inserted (chứa các bản ghi vừa được thêm hoặc cập nhật trong bảng HoaDon).   2. **Lấy giá phòng từ bảng PhongTro:**   * Dựa vào @MaPhong, trigger sẽ lấy giá phòng (@GiaPhong) từ bảng PhongTro.   3. **Tính tổng thành tiền từ bảng chi tiết hóa đơn:**   * Trigger tính tổng các khoản chi phí từ bảng HoaDonChiTiet (liên kết với hóa đơn) và gán cho biến @ThanhTien.   4. **Tính tổng tiền:**   * Tổng tiền được tính bằng: @GiaPhong + @ThanhTien + (số chữ điện @SoChuDien nhân với giá điện @GiaDien) + @TienNuoc.   5. **Cập nhật tổng tiền vào bảng HoaDon:**   * Trigger cập nhật cột TongTien trong bảng HoaDon với giá trị tổng tiền vừa tính được. |

Bảng 25: Module Trigger\_Tính tổng tiền hóa đơn

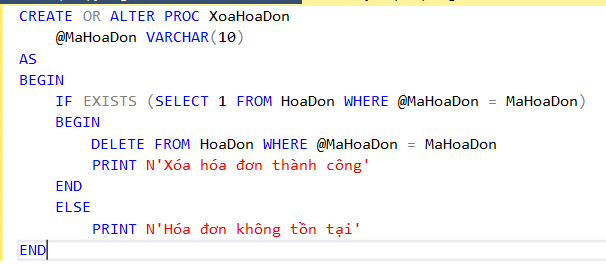
## 5.16 Module Xem hóa đơn theo phòng

****

|  |  |
| --- | --- |
| **STT** | 16. |
| **Tên** | Xem hóa đơn theo phòng |
| **Loại** | Procedure |
| **Mục đích** | Dùng để truy xuất thông tin chi tiết về các hóa đơn dựa trên MaPhong. Cho phép chủ trọ có thể dễ dàng kiểm tra tất cả các hóa đơn liên quan đến một phòng cụ thể, phục vụ các mục đích như quản lý thanh toán, đối chiếu dữ liệu hoặc xem chi tiết lịch sử sử dụng dịch vụ của phòng trọ. |
| **Input** | @MaPhong (kiểu dữ liệu: VARCHAR(20)) |
| **Output** | Xuất ra thông tin hóa đơn của phòng bao gồm mã hóa đơn, số điện tiêu thụ, giá điện, tiền nước, tổng tiền, ngày lập hóa đơn và trạng thái thanh toán |
| **Process** | Truy vấn các hóa đơn tương ứng với phòng từ bảng HoaDon, điều kiện @MaPhong=MaPhong |

Bảng 26; Module xem hóa đơn theo phòng

## 5.17 Module Xóa hóa đơn



|  |  |
| --- | --- |
| **STT** | 17. |
| **Tên** | Xóa hóa đơn |
| **Loại** | Procedure |
| **Mục đích** | Dùng để xóa một hóa đơn khỏi bảng HoaDon dựa trên mã hóa đơn (@MaHoaDon). Thủ tục sẽ kiểm tra xem hóa đơn có tồn tại không trước khi xóa. Nếu tồn tại, thủ tục sẽ thực hiện việc xóa hóa đơn, ngược lại sẽ thông báo rằng hóa đơn không tồn tại. |
| **Input** | @MaHoaDon (kiểu dữ liệu: VARCHAR(10)) |
| **Output** | In ra thông báo kết quả của việc xóa hóa đơn |
| **Process** | Kiểm tra sự tồn tại của MaPhong:  + Nếu có, thêm mới thông tin hóa đơn vào bảng HoaDon và in ra thông báo 'Tạo hóa đơn thành công'  + Ngược lại, in ra thông báo 'Mã phòng không tồn tại' |

Bảng 27: Module xóa hóa đơn

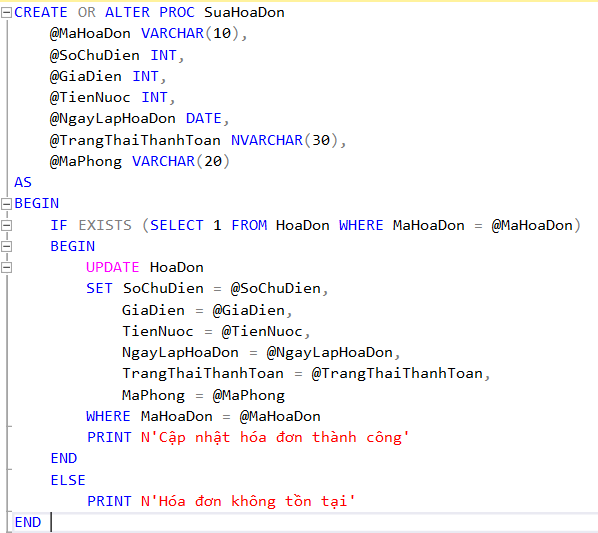
## 5.18 Module Tạo hóa đơn



|  |  |
| --- | --- |
| **STT** | 18. |
| **Tên** | Tạo hóa đơn |
| **Loại** | Procedure |
| **Mục đích** | Dùng để thêm một hóa đơn mới vào bảng HoaDon. Thủ tục kiểm tra xem mã phòng (@MaPhong) có tồn tại trong bảng PhongTro hay không trước khi tạo hóa đơn. Nếu mã phòng tồn tại, thủ tục sẽ chèn thông tin hóa đơn vào bảng HoaDon, nếu không sẽ thông báo lỗi. |
| **Input** | @MaHoaDon (kiểu: VARCHAR(10))  @SoChuDien (kiểu: INT)  @GiaDien (kiểu: INT)  @TienNuoc (kiểu: INT)  @NgayLapHoaDon (kiểu: DATE)  @TrangThaiThanhToan (kiểu: NVARCHAR(30))  @MaPhong (kiểu: VARCHAR(20)) |
| **Output** | In ra thông báo kết quả của việc tạo hóa đơn |
| **Process** | Kiểm tra sự tồn tại của MaPhong  + Nếu có, thêm mới thông tin hóa đơn vào bảng HoaDon và in ra thông báo 'Tạo hóa đơn thành công'  + Ngược lại, in ra thông báo 'Mã phòng không tồn tại' |

Bảng 28: Module tạo hóa đơn

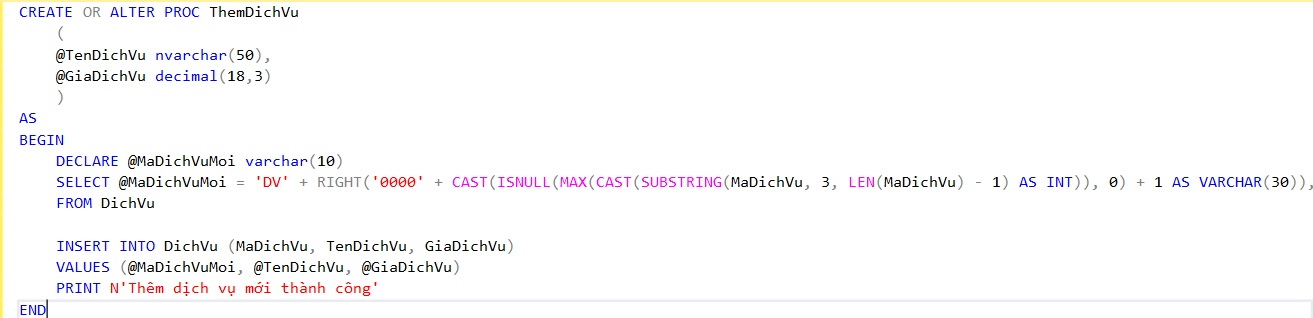
## 5.19 Module Sửa hóa đơn

****

|  |  |
| --- | --- |
| **STT** | 19. |
| **Tên** | Sửa hóa đơn |
| **Loại** | Procedure |
| **Mục đích** | Dùng để cập nhật thông tin của một hóa đơn đã có trong bảng HoaDon. Thủ tục kiểm tra xem mã hóa đơn (@MaHoaDon) có tồn tại trong bảng không trước khi thực hiện cập nhật. Nếu tồn tại, thông tin hóa đơn sẽ được cập nhật với các giá trị mới, nếu không sẽ thông báo rằng hóa đơn không tồn tại. |
| **Input** | @MaHoaDon (kiểu: VARCHAR(10))  @SoChuDien (kiểu: INT)  @GiaDien (kiểu: INT)  @TienNuoc (kiểu: INT)  @NgayLapHoaDon (kiểu: DATE)  @TrangThaiThanhToan (kiểu: NVARCHAR(30))  @MaPhong (kiểu: VARCHAR(20))  @MaPhong (kiểu: VARCHAR(20)) |
| **Output** | In ra thông báo kết quả của việc tạo hóa đơn |
| **Process** | Kiểm tra sự tồn tại của MaPhong  + Nếu có, thêm mới thông tin hóa đơn vào bảng HoaDon và in ra thông báo 'Tạo hóa đơn thành công'  + Ngược lại, in ra thông báo 'Mã phòng không tồn tại' |

Bảng 29: Module sửa hóa đơn

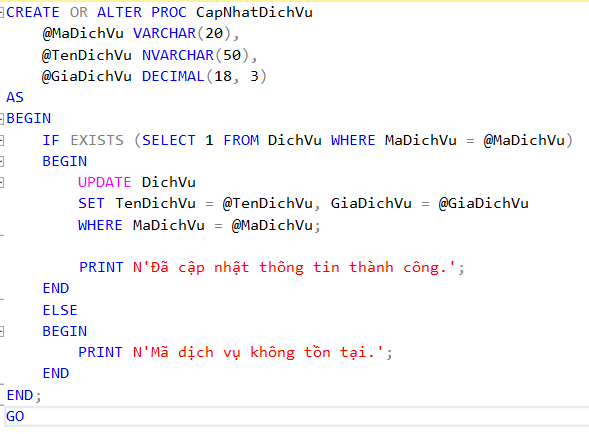
## 5.20 Module Thêm dịch vụ mới



|  |  |
| --- | --- |
| **STT** | 20. |
| **Tên** | Thêm dịch vụ mới |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục này dùng khi chủ trọ muốn thêm mới một dịch vụ vào bảng DichVu |
| **Input** | @MaDichVu varchar(20)  @TenDichVu nvarchar(50)  @GiaDichVu decimal(18,3) |
| **Output** | In ra thông báo tương ứng:  -"Dịch vụ đã tồn tại" nếu dịch vụ đã tồn tại trong bảng DichVu  -"Thêm dịch vụ mới thành công" nếu dịch vụ được thêm thành công vào bảng DichVu |
| **Process** | 1.Kiểm tra dịch vụ đã tồn tại trong bảng DichVu  -1.1.Nếu tồn tại -> In ra thông báo "Dịch vụ đã tồn tại"  -1.2.Ngược lại -> In ra thông báo "Thêm dịch vụ mới thành công" |

Bảng 31: Module thêm dịch vụ mới

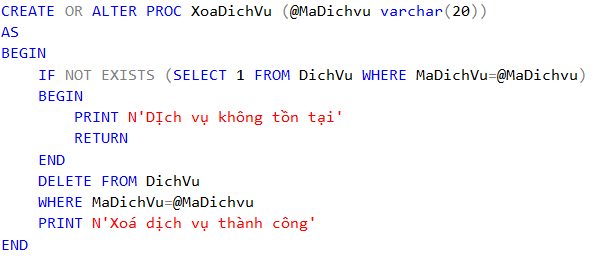
## 5.21 Module Cập nhật thông tin dịch vụ



|  |  |
| --- | --- |
| **STT** | 21. |
| **Tên** | Cập nhật lại thông tin của dịch vụ |
| **Loại** | Procedure |
| **Mục đích** | Dùng để cập nhật lại thông tin của dịch vụ đã có trong bảng DichVu |
| **Input** | @MaDichVu VARCHAR(20)  @TenDichVu NVARCHAR(50)  @GiaDichVu DECIMAL(18, 3) |
| **Output** | In ra thông báo kết quả của việc sửa thông tin của dịch vụ |
| **Process** | 1.Kiểm tra sự tồn tại của MaDichVu  1.1.Nếu tồn tại: cập nhật các thông tin tương ứng với MaDichVu và in ra thông báo 'Đã cập nhật thông tin thành công.'  1.2.Ngược lại: in ra thông báo 'Mã dịch vụ không tồn tại.' |

Bảng 32: Module cập nhật lại thông tin của dịch vụ

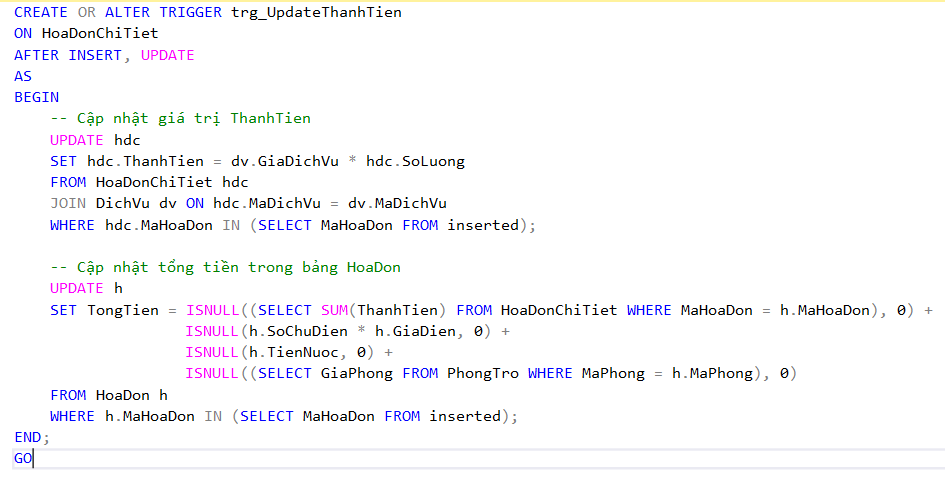
## 5.22 Module Xóa dịch vụ



|  |  |
| --- | --- |
| **STT** | 22. |
| **Tên** | Xóa dịch vụ |
| **Loại** | Procedure |
| **Mục đích** | Dùng để xoá dịch vụ khỏi bảng DichVu dựa trên mã dịch vụ |
| **Input** | @MaDichVu VARCHAR(20) |
| **Output** | In ra kết quả của việc xoá dịch vụ |
| **Process** | 1.Kiểm tra sự tồn tại của mã dịch vụ  1.1.Nếu không tồn tại:-> in ra thông báo: 'Dịch vụ không tồn tại'  1.2.Nếu tồn tại: thực hiện xoá dịch vụ ra khỏi bảng DichVu dựa trên @MaDichVu -> in ra thông báo: 'Xoá dịch vụ thành công' |

Bảng 33: Module xóa dịch vụ

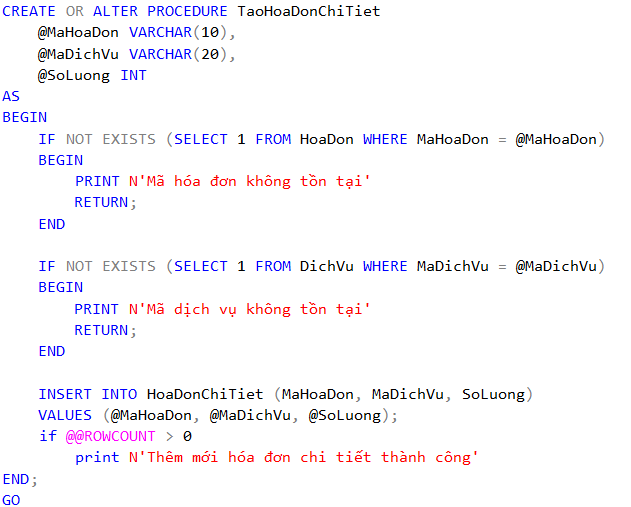
## 5.23 Trigger Cập nhật tổng tiền trong bảng HoaDon



|  |  |
| --- | --- |
| **STT** | 23. |
| **Tên** | Trigger\_Cập Nhật Thành Tiền |
| **Loại** | Trigger |
| **Mục đích** | Trigger trg\_UpdateThanhTien tự động cập nhật giá trị ThanhTien trong bảng HoaDonChiTiet và tổng số tiền (TongTien) trong bảng HoaDon mỗi khi có bản ghi mới được thêm vào hoặc bản ghi hiện có được cập nhật trong bảng HoaDonChiTiet. |
| **Input** | Trigger được kích hoạt sau khi thực hiện câu lệnh INSERT hoặc UPDATE trên bảng HoaDonChiTiet. |
| **Output** | 1. Cập nhật trường ThanhTien trong bảng HoaDonChiTiet dựa trên đơn giá dịch vụ và số lượng.  2. Cập nhật trường TongTien trong bảng HoaDon với tổng số tiền mới tính toán. |
| **Process** | 1. **Cập nhật giá trị ThanhTien:**   * Trigger sử dụng câu lệnh UPDATE để tính toán và cập nhật giá trị ThanhTien trong bảng HoaDonChiTiet. * Giá trị ThanhTien được tính bằng cách nhân GiaDichVu từ bảng DichVu với SoLuong trong bảng HoaDonChiTiet. * Điều kiện cập nhật dựa trên MaHoaDon, chỉ cập nhật cho những hóa đơn mà có bản ghi mới trong bảng inserted.   2. **Cập nhật tổng tiền trong bảng HoaDon:**   * Một câu lệnh UPDATE khác được thực hiện để tính và cập nhật tổng tiền (TongTien) trong bảng HoaDon. * Tổng tiền được tính bằng cách:   + Tổng hợp giá trị ThanhTien từ bảng HoaDonChiTiet liên quan đến MaHoaDon.   + Cộng thêm chi phí điện (SoChuDien \* GiaDien).   + Cộng thêm TienNuoc.   + Cộng thêm giá phòng từ bảng PhongTro dựa trên MaPhong. * Sử dụng ISNULL để đảm bảo rằng nếu không có giá trị nào thì sẽ được thay thế bằng 0, tránh trường hợp tính toán bị lỗi. |

Bảng 34: Trigger Cập Nhật Thành Tiền

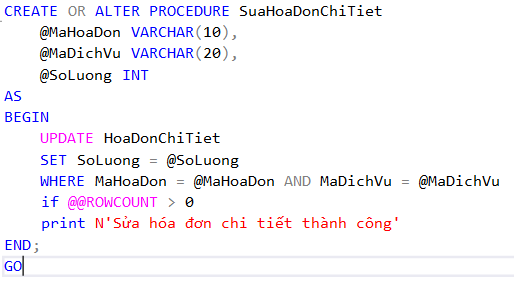
## 5.24 Module Tạo hóa đơn chi tiết

****

|  |  |
| --- | --- |
| **STT** | 24 |
| **Tên** | Tạo hóa đơn chi tiết |
| **Loại** | Procedure |
| **Mục đích** | Dùng khi muốn thêm mới 1 bản ghi vào bảng HoaDonChiTiet |
| **Input** | @MaHoaDon (VARCHAR(10)  @MaDichVu (VARCHAR(20)  @SoLuong (INT)  @ThanhTien (DECIMAL(18, 2) |
| **Output** | Thủ tục không trả về kết quả dưới dạng một tập dữ liệu, nhưng có thể thông báo lỗi trong trường hợp MaHoaDon hoặc MaDichVu không tồn tại. Nếu không có lỗi, bản ghi mới sẽ được thêm vào bảng HoaDonChiTiet |
| **Process** | 1. Kiểm tra xem @MaHoaDon có tồn tại trong bảng HoaDon hay không. Nếu không tồn tại, thủ tục sẽ phát sinh lỗi và dừng lại.  2. Kiểm tra xem @MaDichVu có tồn tại trong bảng DichVu hay không. Nếu không tồn tại, thủ tục sẽ phát sinh lỗi và dừng lại.  3. Nếu cả hai giá trị @MaHoaDon và @MaDichVu đều hợp lệ, thủ tục sẽ chèn một bản ghi mới vào bảng HoaDonChiTiet với các thông tin được cung cấp.  4. Bản ghi mới được thêm vào với các giá trị của MaHoaDon, MaDichVu, SoLuong, và ThanhTien. |

Bảng 36: Module Tạo hóa đơn chi tiết

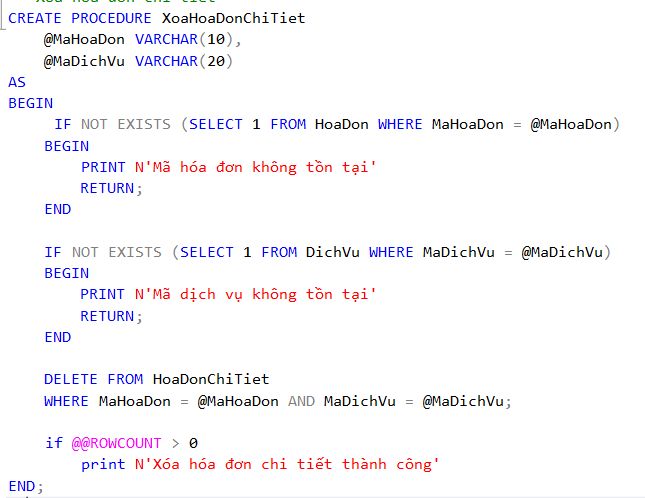
## 5.25 Module Sửa hóa đơn chi tiết



|  |  |
| --- | --- |
| **STT** | 25 |
| **Tên** | Sửa hóa đơn chi tiết |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục SuaHoaDonChiTiet được sử dụng để cập nhật số lượng của dịch vụ trong bảng HoaDonChiTiet dựa trên mã hóa đơn và mã dịch vụ cụ thể. |
| **Input** | 1. @MaHoaDon (VARCHAR(10)): Mã của hóa đơn mà người dùng muốn sửa đổi.  2. @MaDichVu (VARCHAR(20)): Mã của dịch vụ mà người dùng muốn cập nhật số lượng.  3. @SoLuong (INT): Số lượng mới của dịch vụ được cập nhật. |
| **Output** | Thông báo kết quả của quá trình cập nhật: nếu thành công, nó sẽ hiển thị "Sửa hóa đơn chi tiết thành công". |
| **Process** | 1. **Cập nhật số lượng dịch vụ:**   * Thủ tục thực hiện một câu lệnh UPDATE trên bảng HoaDonChiTiet để cập nhật trường SoLuong. * Điều kiện cập nhật được áp dụng cho những bản ghi mà có MaHoaDon và MaDichVu tương ứng với các tham số đầu vào.   2. **Kiểm tra kết quả:**   * Sau khi thực hiện câu lệnh UPDATE, thủ tục kiểm tra biến hệ thống @@ROWCOUNT để xác định xem có bao nhiêu bản ghi đã bị ảnh hưởng bởi câu lệnh cập nhật. * Nếu ít nhất một bản ghi được cập nhật (@@ROWCOUNT > 0), thủ tục sẽ in ra thông báo "Sửa hóa đơn chi tiết thành công". |

Bảng 37: Module Sửa hóa đơn chi tiết

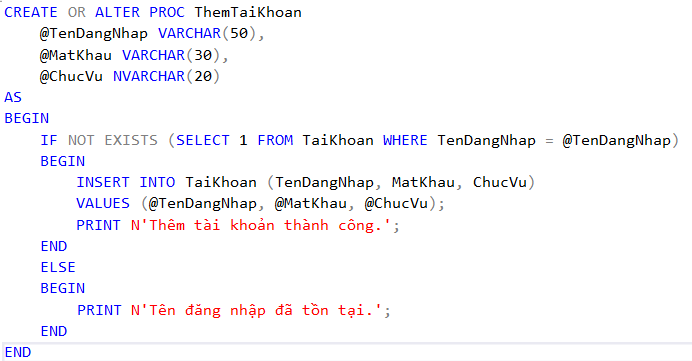
## 5.26 Module Xóa hóa đơn chi tiết



|  |  |
| --- | --- |
| **STT** | 26 |
| **Tên** | Xóa hóa đơn chi tiết |
| **Loại** | Procedure |
| **Mục đích** | Xóa một bản ghi cụ thể trong bảng HoaDonChiTiet, dựa trên MaHoaDon và MaDichVu. cho phép người dùng loại bỏ các chi tiết hóa đơn không còn cần thiết |
| **Input** | @MaHoaDon (VARCHAR(10)): Mã hóa đơn của bản ghi cần xóa.  @MaDichVu (VARCHAR(20)): Mã dịch vụ liên quan đến hóa đơn cần xóa. |
| **Output** | Thủ tục không trả về dữ liệu cụ thể. Nó sẽ xóa bản ghi trong bảng HoaDonChiTiet nếu có bản ghi phù hợp với các điều kiện MaHoaDon và MaDichVu |
| **Process** | 1. Thực hiện lệnh DELETE để xóa các bản ghi trong bảng HoaDonChiTiet với điều kiện MaHoaDon và MaDichVu tương ứng với các tham số đầu vào.  2. Nếu có bản ghi khớp với điều kiện, nó sẽ bị xóa. Nếu không có bản ghi nào phù hợp, không có thay đổi nào được thực hiện trong bảng. |

Bảng 38: Module Xóa hóa đơn chi tiết

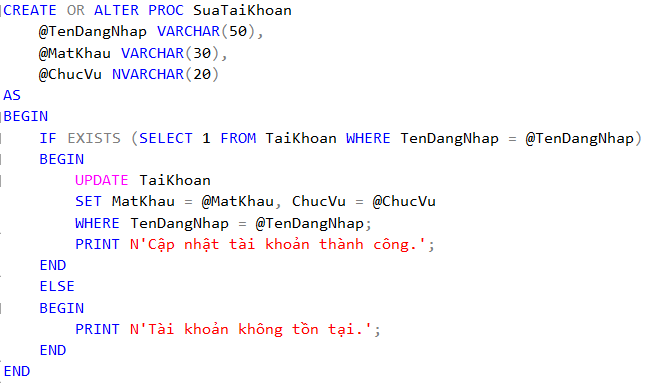
## 5.27 Module Thêm tài khoản



|  |  |
| --- | --- |
| **STT** | 27 |
| **Tên** | Thêm tài khoản |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục ThemTaiKhoan được sử dụng để thêm một tài khoản mới vào bảng TaiKhoan, đảm bảo rằng tên đăng nhập không bị trùng lặp. |
| **Input** | @TenDangNhap (VARCHAR(50)): Tên đăng nhập của tài khoản mới.  @MatKhau (VARCHAR(30)): Mật khẩu của tài khoản mới.  @ChucVu (NVARCHAR(20)): Chức vụ của tài khoản mới.. |
| **Output** | Thông báo kết quả: nếu thêm tài khoản thành công, nó sẽ hiển thị "Thêm tài khoản thành công." Nếu tên đăng nhập đã tồn tại, nó sẽ hiển thị "Tên đăng nhập đã tồn tại." |
| **Process** | 1. **Kiểm tra sự tồn tại của tên đăng nhập:**   * Thủ tục kiểm tra xem tên đăng nhập (@TenDangNhap) đã tồn tại trong bảng TaiKhoan hay chưa bằng cách sử dụng câu lệnh SELECT.   2. **Thêm tài khoản mới:**   * Nếu tên đăng nhập chưa tồn tại, thủ tục sẽ thực hiện một câu lệnh INSERT để thêm bản ghi mới vào bảng TaiKhoan với thông tin được cung cấp.   3. **Thông báo kết quả:**   * Nếu việc thêm tài khoản thành công, thủ tục sẽ in ra thông báo "Thêm tài khoản thành công." * Nếu tên đăng nhập đã tồn tại, thủ tục sẽ in ra thông báo "Tên đăng nhập đã tồn tại." |

Bảng 40: Mudule thêm tài khoản

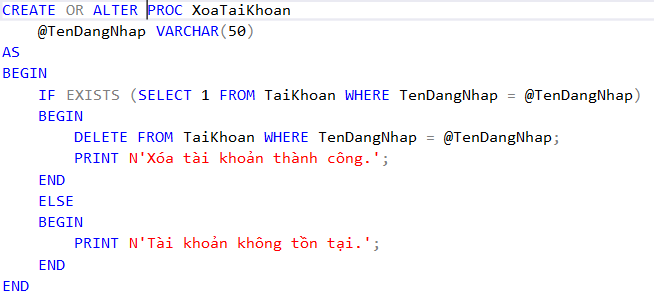
## 5.28 Module Sửa tài khoản



|  |  |
| --- | --- |
| **STT** | 29 |
| **Tên** | Sửa tài khoản |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục SuaTaiKhoan được sử dụng để cập nhật thông tin của một tài khoản đã tồn tại trong bảng TaiKhoan, bao gồm mật khẩu và chức vụ. |
| **Input** | @TenDangNhap (VARCHAR(50)): Tên đăng nhập của tài khoản cần cập nhật.  @MatKhau (VARCHAR(30)): Mật khẩu mới cho tài khoản.  @ChucVu (NVARCHAR(20)): Chức vụ mới cho tài khoản. |
| **Output** | Thông báo kết quả:   * Nếu tài khoản đã được cập nhật thành công, thông báo "Cập nhật tài khoản thành công." sẽ được hiển thị. * Nếu tài khoản không tồn tại, thông báo "Tài khoản không tồn tại." sẽ được hiển thị. |
| **Process** | 1. **Kiểm tra sự tồn tại của tài khoản:**   * Thủ tục kiểm tra xem tài khoản với tên đăng nhập (@TenDangNhap) có tồn tại trong bảng TaiKhoan hay không bằng cách sử dụng câu lệnh SELECT.   2. **Cập nhật thông tin tài khoản:**   * Nếu tài khoản tồn tại, thủ tục sẽ thực hiện một câu lệnh UPDATE để cập nhật mật khẩu và chức vụ với thông tin mới.   3. **Thông báo kết quả:**   * Nếu việc cập nhật tài khoản thành công, thủ tục sẽ in ra thông báo "Cập nhật tài khoản thành công." * Nếu tài khoản không tồn tại, thủ tục sẽ in ra thông báo "Tài khoản không tồn tại." |

Bảng 41: Module Sửa tài khoản

## 5.29 Module Xóa tài khoản



|  |  |
| --- | --- |
| **STT** | 29 |
| **Tên** | Xóa tài khoản |
| **Loại** | Procedure |
| **Mục đích** | Thủ tục XoaTaiKhoan được sử dụng để xóa một tài khoản trong bảng TaiKhoan dựa trên tên đăng nhập của tài khoản đó. |
| **Input** | @TenDangNhap (VARCHAR(50)): Tên đăng nhập của tài khoản cần xóa. |
| **Output** | Thông báo kết quả:   * Nếu tài khoản đã được xóa thành công, thông báo "Xóa tài khoản thành công." sẽ được hiển thị. * Nếu tài khoản không tồn tại, thông báo "Tài khoản không tồn tại." sẽ được hiển thị. |
| **Process** | 1. **Kiểm tra sự tồn tại của tài khoản:**   * Thủ tục kiểm tra xem tài khoản với tên đăng nhập (@TenDangNhap) có tồn tại trong bảng TaiKhoan hay không bằng cách sử dụng câu lệnh SELECT.   2. **Xóa tài khoản:**   * Nếu tài khoản tồn tại, thủ tục sẽ thực hiện một câu lệnh DELETE để xóa tài khoản khỏi bảng TaiKhoan.   3. **Thông báo kết quả:**   * Nếu việc xóa tài khoản thành công, thủ tục sẽ in ra thông báo "Xóa tài khoản thành công." * Nếu tài khoản không tồn tại, thủ tục sẽ in ra thông báo "Tài khoản không tồn tại." |

Bảng 42: Module Xóa tài khoản

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

### Giới thiệu

Xác thực người dùng là yếu tố cốt lõi đảm bảo tính bảo mật và quyền truy cập hợp lệ vào hệ thống cơ sở dữ liệu. SQL Server hỗ trợ hai cơ chế xác thực chính:

* Windows Authentication: Xác thực dựa trên thông tin tài khoản Windows.
* SQL Server Authentication: Xác thực thông qua tên đăng nhập và mật khẩu độc lập.

Trong hệ thống quản lý phòng trọ, nhóm quyết định sử dụng cơ chế SQL Server Authentication.

### Lý do lựa chọn SQL Server Authentication

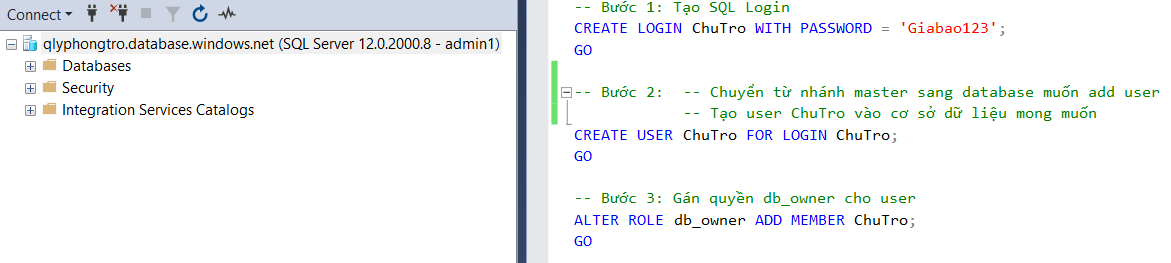
1. Độc lập với Windows: SQL Server Authentication không yêu cầu tài khoản Windows, mang lại tính linh hoạt cao trong việc quản lý người dùng.
2. Hỗ trợ truy cập từ xa: Phù hợp trong các tình huống người dùng cần truy cập cơ sở dữ liệu từ ngoài mạng nội bộ hoặc qua các ứng dụng bên ngoài.
3. Dễ dàng triển khai: Quản trị viên có thể nhanh chóng thiết lập, cấp quyền và quản lý tài khoản trực tiếp trên SQL Server mà không phụ thuộc vào hệ thống Windows.

* **Phân quyền:**
* **ChuTro:**

+ Bước 1: Tạo SQL Login

+ Bước 2: Chuyển từ nhánh master sang database muốn add user

Tạo user ChuTro vào cơ sở dữ liệu mong muốn

+ Bước 3: Gán quyền db\_owner cho user

Hình 9 Thao tác cấp quyền cho Chủ trọ

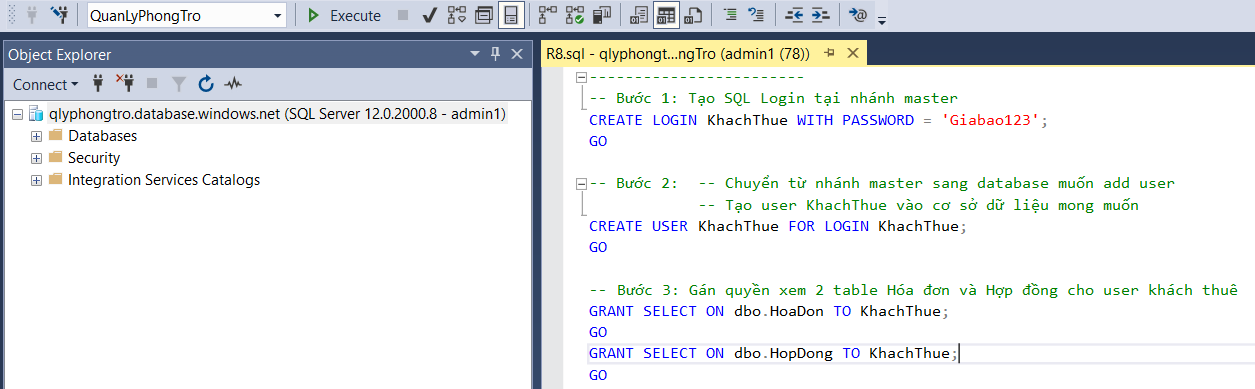
* **dbo** là một user đặc biệt có tất cả các quyền trên database. Một login account là thành viên của **db\_owner** database role thì cũng có quyền như dbo. Điều đó đồng nghĩa với việc Chủ trọ sẽ có toàn bộ quyền đối với database.
* **KhachThue:**

+ Bước 1: Tạo SQL Login tại nhánh master

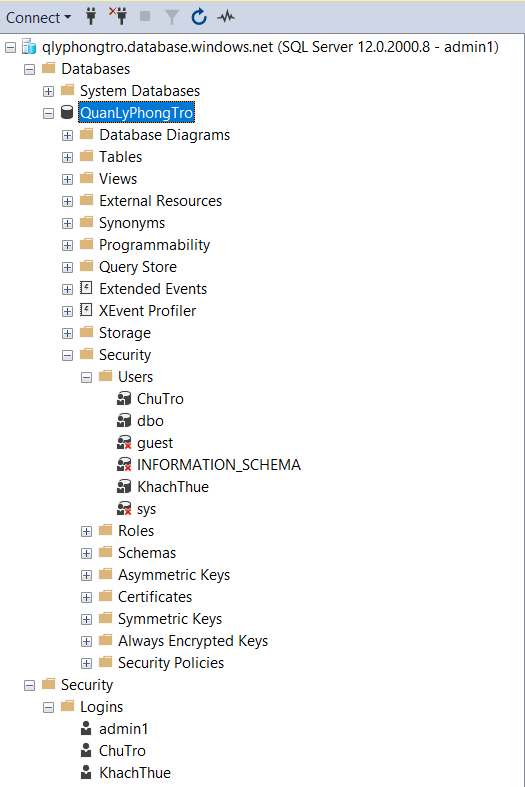
+ Bước 2: Chuyển từ nhánh master sang database muốn add user

Tạo user KhachThue vào cơ sở dữ liệu mong muốn

+ Bước 3: Gán quyền xem 2 table Hóa đơn và Hợp đồng cho user khách thuê

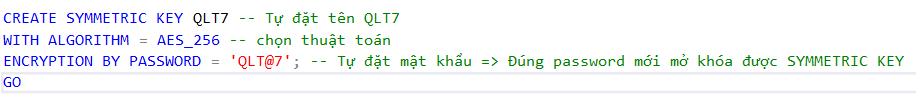


Hình 10 Gán quyền xem 2 table Hóa đơn và Hợp đồng cho khách thuê

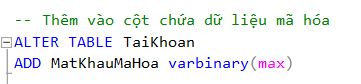


Hình 11 Gán quyền cho chủ trọ và khách thuê thành công

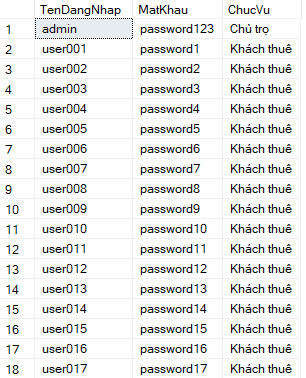
* **Mã hoá**: Mã hóa dữ liệu là quá trình chuyển đổi thông tin từ dạng có thể đọc được sang dạng không thể đọc được nếu không có khóa giải mã. Đây là một kỹ thuật quan trọng trong bảo mật thông tin, đặc biệt là trong các hệ thống quản lý cơ sở dữ liệu. Việc mã hóa giúp bảo vệ dữ liệu nhạy cảm như mật khẩu, thông tin cá nhân khỏi bị truy cập trái phép.
* Trong bài báo cáo này, chúng tôi trình bày cách sử dụng mã hóa dữ liệu trong SQL Server với khóa đối xứng (*Symmetric Key*) để bảo vệ dữ liệu người dùng.

**Bước 1**: Tạo một khóa đối xứng (*Symmetric Key*) sử dụng thuật toán AES\_256. Khóa này sẽ được bảo vệ bằng một mật khẩu do người dùng định nghĩa.

**Bước 2:** Trong bảng TaiKhoan, cần thêm một cột mới có kiểu dữ liệu VARBINARY(MAX) để lưu trữ dữ liệu sau khi đã mã hóa. Cột này sẽ thay thế cho cột chứa dữ liệu gốc không mã hóa.

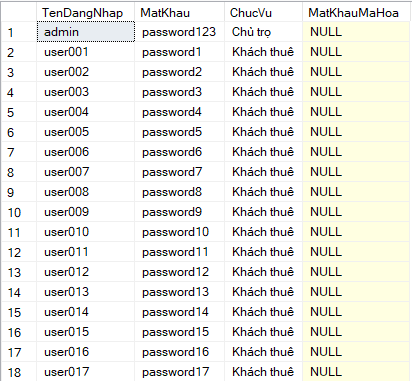


* **Trước khi thêm:**



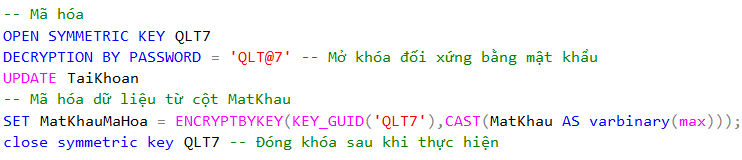
Hình 12 Trước khi thêm cột MatKhauMaHoa

* **Sau khi thêm:**

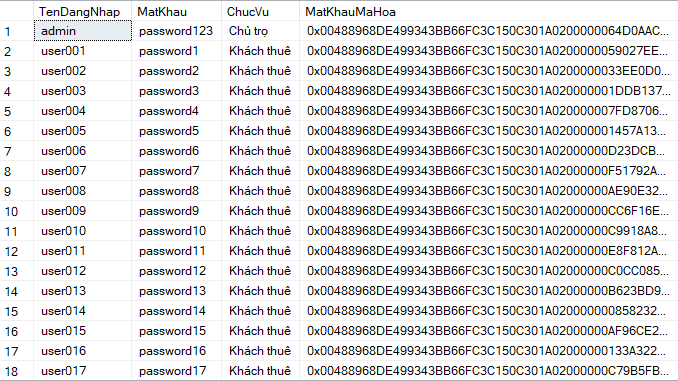


Hình 13 Sau khi thêm cột MatKhauMaHoa

**Bước 3:** Dữ liệu nhạy cảm, cụ thể là cột **MatKhau**, sẽ được mã hóa bằng khóa đối xứng vừa tạo. Sau đó, dữ liệu mã hóa sẽ được lưu vào cột **MatKhauMaHoa**.

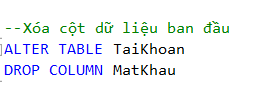


* **Sau khi chạy:**

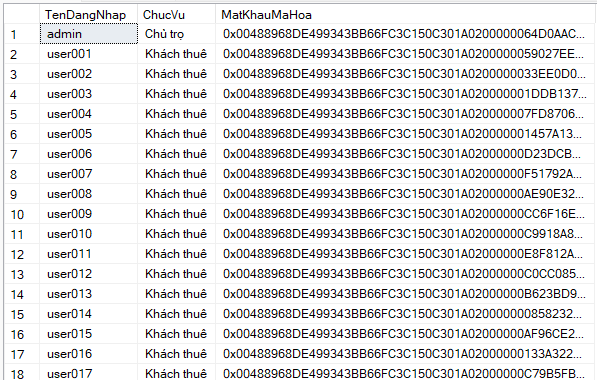
****

Hình 14 Kết quả sau khi mã hóa

**Bước 4:** Sau khi mã hóa thành công, cột chứa dữ liệu gốc không mã hóa (**MatKhau**) cần được xóa để đảm bảo bảo mật, tránh rò rỉ thông tin nhạy cảm.

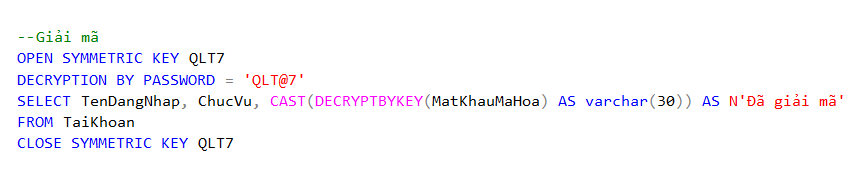


* **Sau khi chạy:**

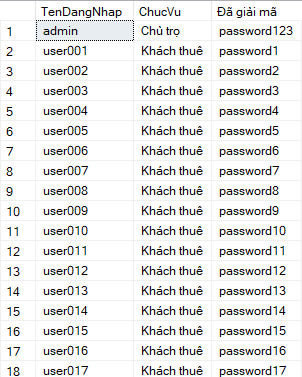


Hình 15 Kết quả sau khi xóa cột dữ liệu ban đầu

\*\*\* Trong một số trường hợp, dữ liệu mã hóa cần được giải mã để hiển thị dưới dạng dễ đọc. Sử dụng khóa đối xứng đã tạo, dữ liệu trong cột **MatKhauMaHoa** sẽ được giải mã và chuyển về định dạng ban đầu.

****

* **Sau khi chạy:**

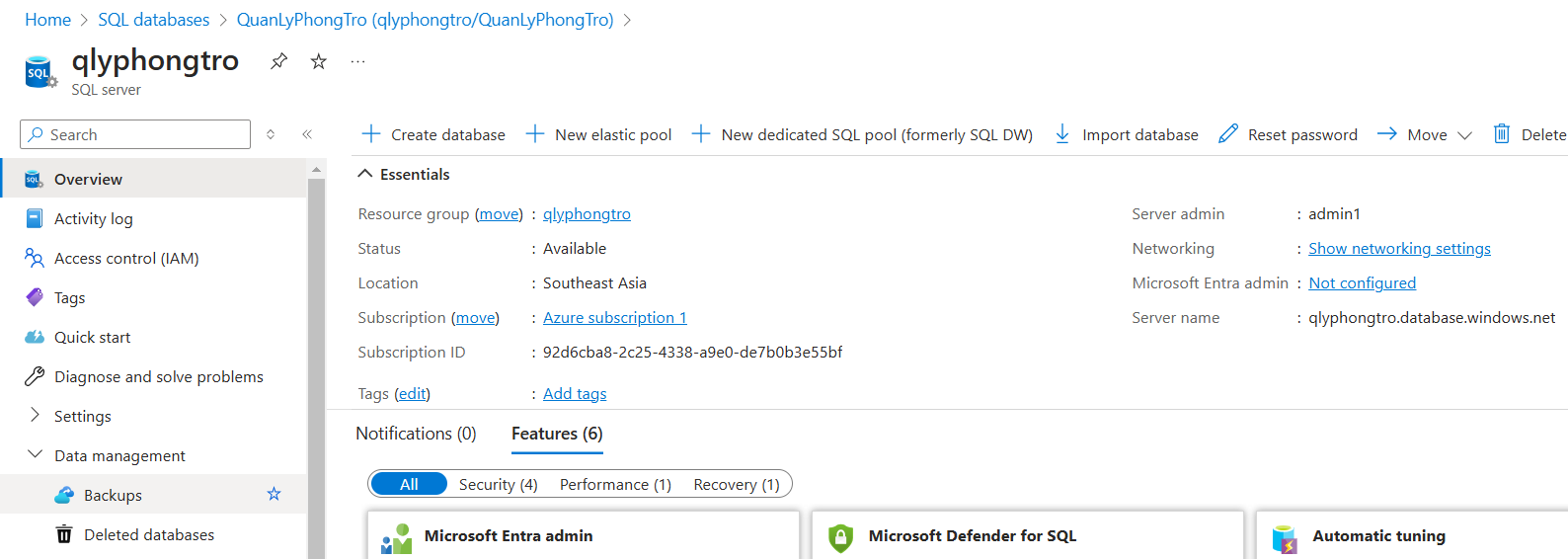


Hình 16 Kết quả sau khi giải mã

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

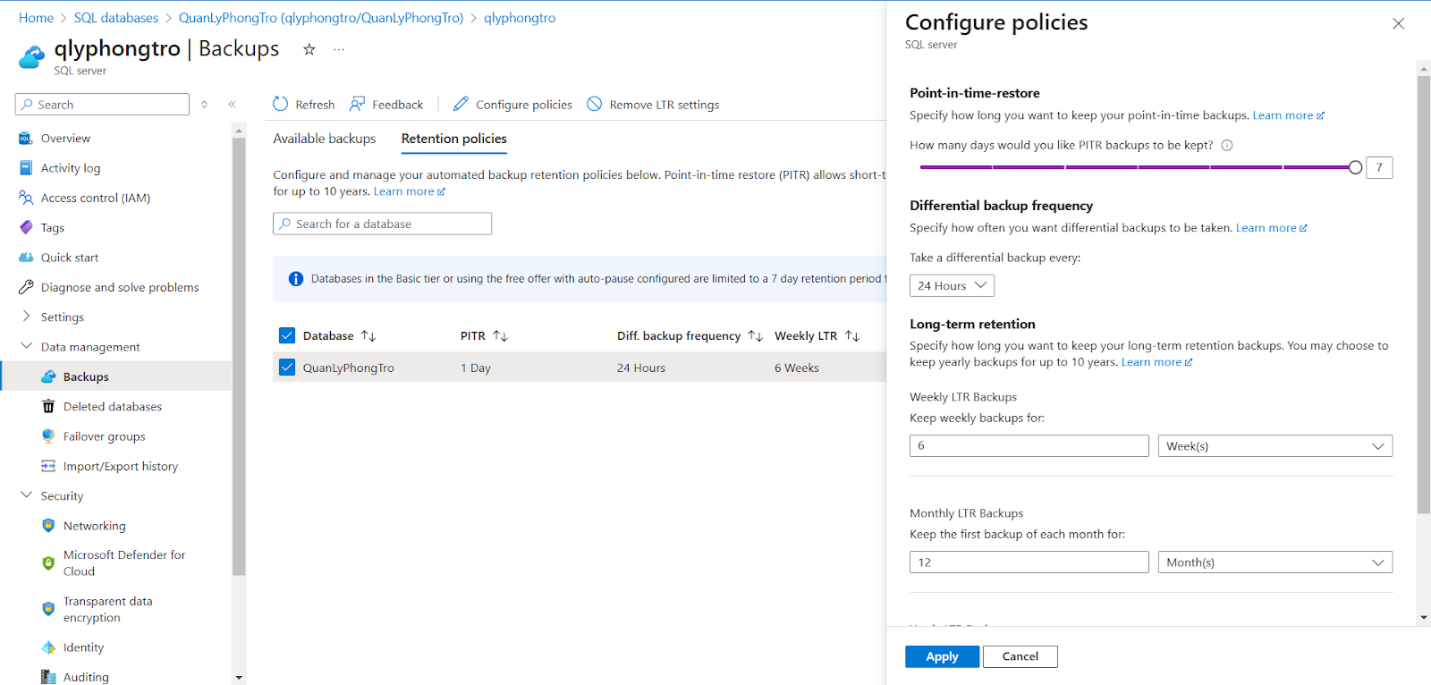
Việc khách thuê vào và ra, sử dụng các dịch vụ diễn ra hàng ngày, chính vì vậy để tránh tình trạng mất, thất lạc dữ liệu thì chúng ta cần phải xây dựng một cơ chế backup phù hợp cho cơ sở dữ liệu.

Vì đặc trưng của cơ sở dữ liệu quản lý phòng trọ là dữ liệu được thay đổi, cập nhập liên tục trong một ngày nên lựa chọn đầu tiên chính là cơ chế: Automatic Daily Database Backup.

* **Vào mục backup tại server cloud muốn thực hiện backup:**

Hình 17 Vào mục backup tại server cloud muốn thực hiện backup

* **Chọn database và thiết lập thời gian backup:**

****

Hình 18 Chọn database và thiết lập thời gian backup

* **PITR (Point-in-Time Restore):** Đây là số ngày sao lưu định kỳ (backup) để bạn có thể khôi phục dữ liệu tại một thời điểm cụ thể (Point-in-Time). Giá trị ở đây là **7 ngày**, nghĩa là các sao lưu sẽ được lưu trữ trong vòng 1 ngày để khôi phục nếu cần.
* **Tần suất tạo bản sao lưu khác biệt (Differential backup):**

 Bản sao lưu khác biệt chỉ lưu lại những thay đổi từ bản sao lưu toàn bộ (Full backup) gần nhất.

Trong cấu hình này, mỗi bản sao lưu khác biệt sẽ được thực hiện sau **24 giờ**.

* **Weekly LTR Backups:**

Giữ lại các bản sao lưu hàng tuần trong **6 tuần**.

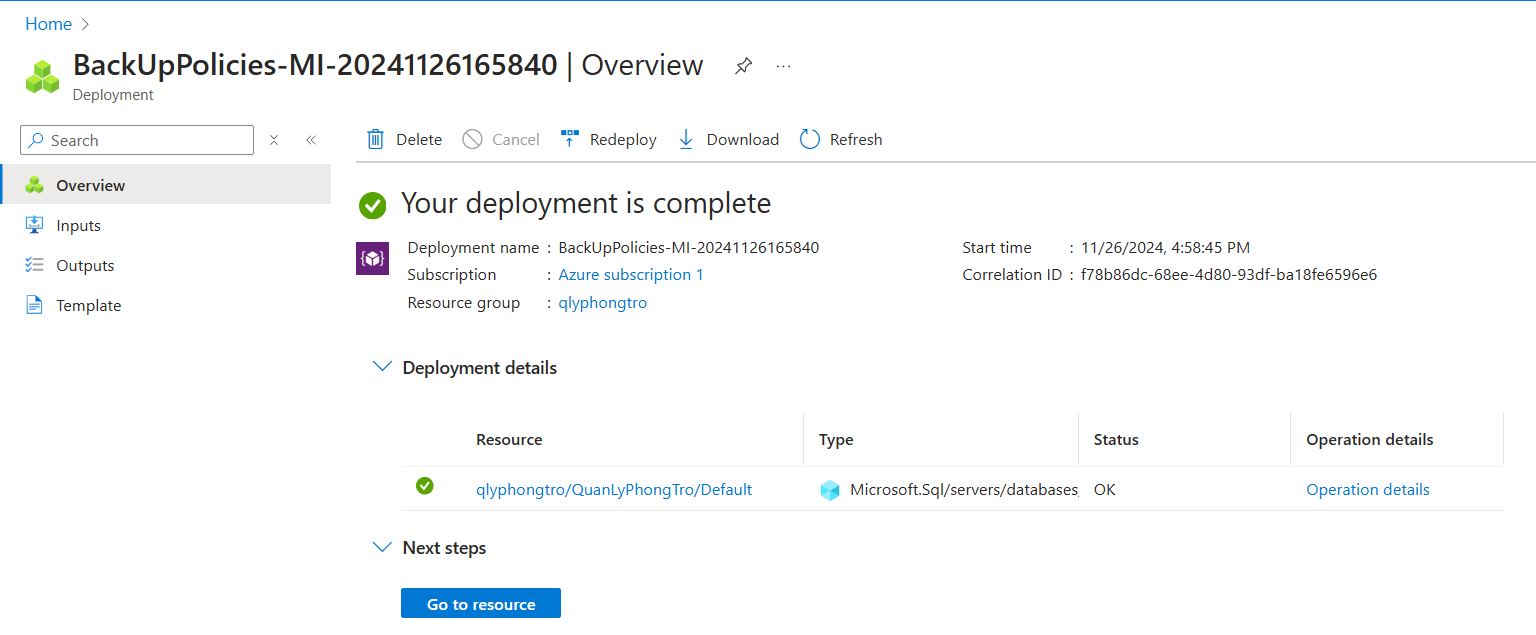
Điều này có nghĩa là một bản sao lưu mỗi tuần sẽ được giữ trong tối đa 6 tuần.

* **Monthly LTR Backups:**

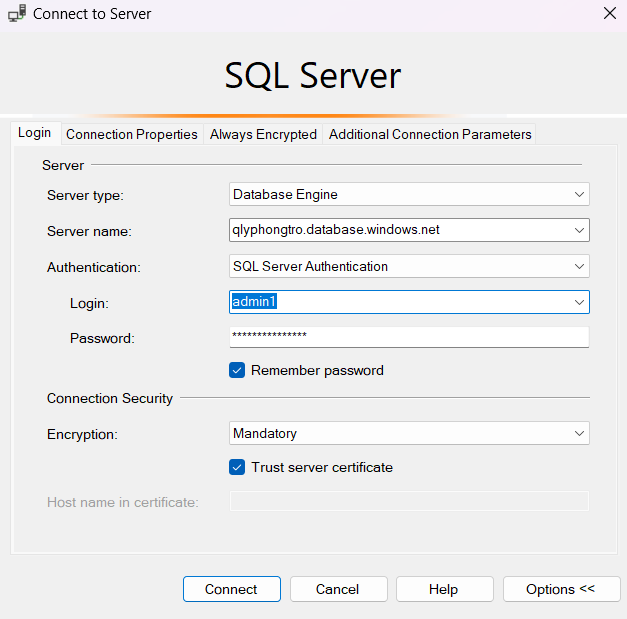
Giữ lại bản sao lưu đầu tiên của mỗi tháng trong **12 tháng** (1 năm).

Ví dụ: Nếu bạn sao lưu ngày 1 của mỗi tháng, bản sao lưu đó sẽ được giữ lại trong 12 tháng.

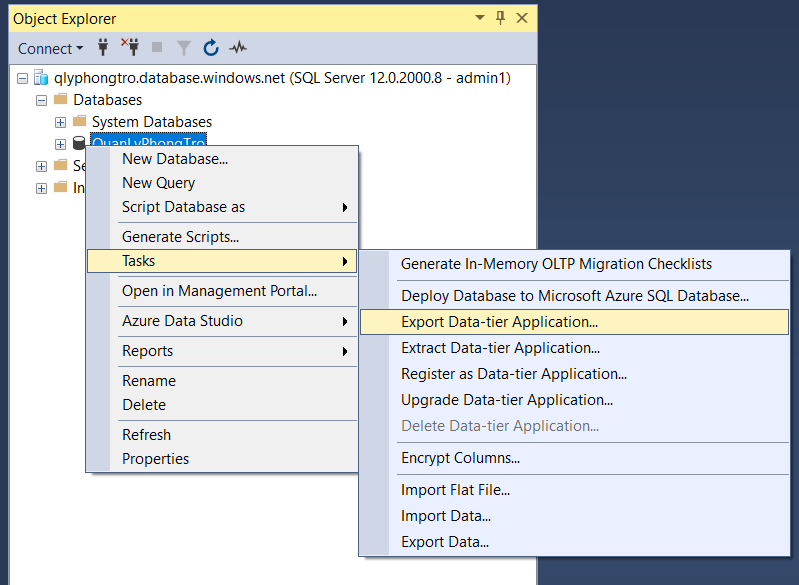
* **Yearly LTR Backups:** Giữ lại bản sao lưu hằng năm trong **10 năm**.
* **Ngoài ra**, bạn có thể chọn **tuần nào của năm** sẽ được sử dụng làm bản sao lưu hằng năm. Trong cấu hình này, bản sao lưu của **Tuần 16 (Week 16)** sẽ được giữ lại như bản sao lưu của năm.
* **Thiết lập thành công:**

****

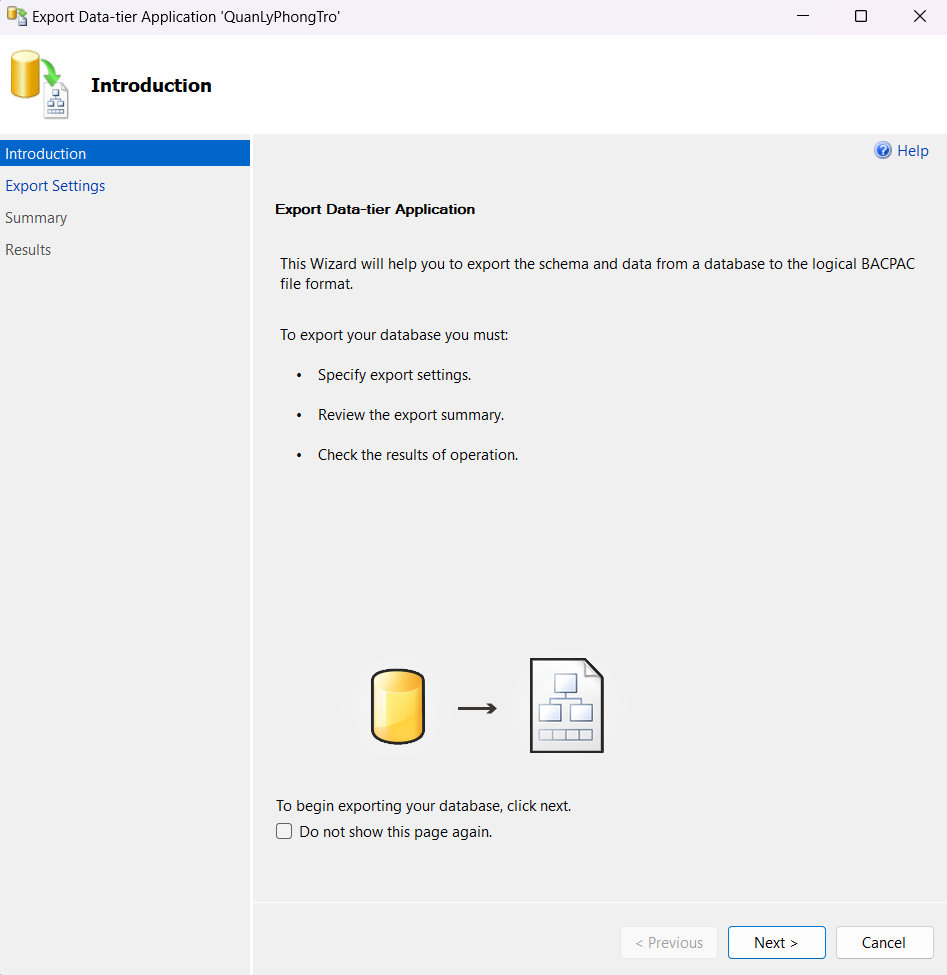
Hình 19 Kết quả sau khi thiết lập



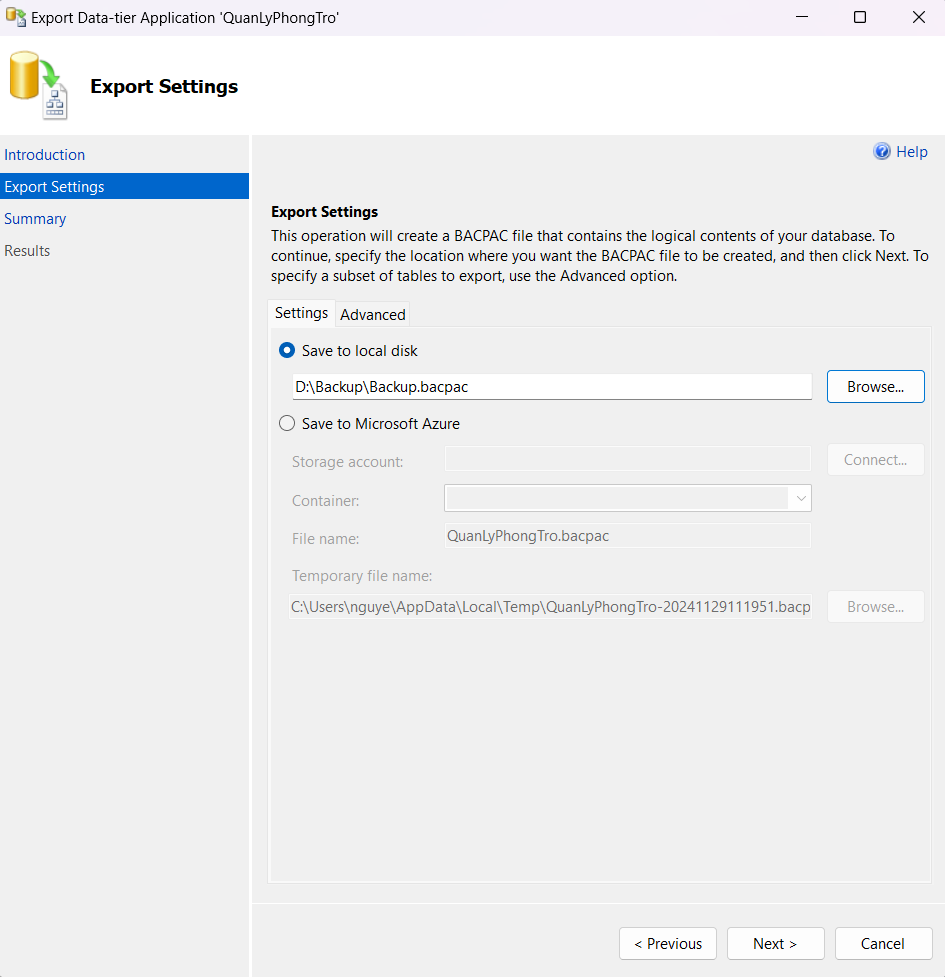
Hình 20 Đăng nhập vào Cloud trên SQL Server

****

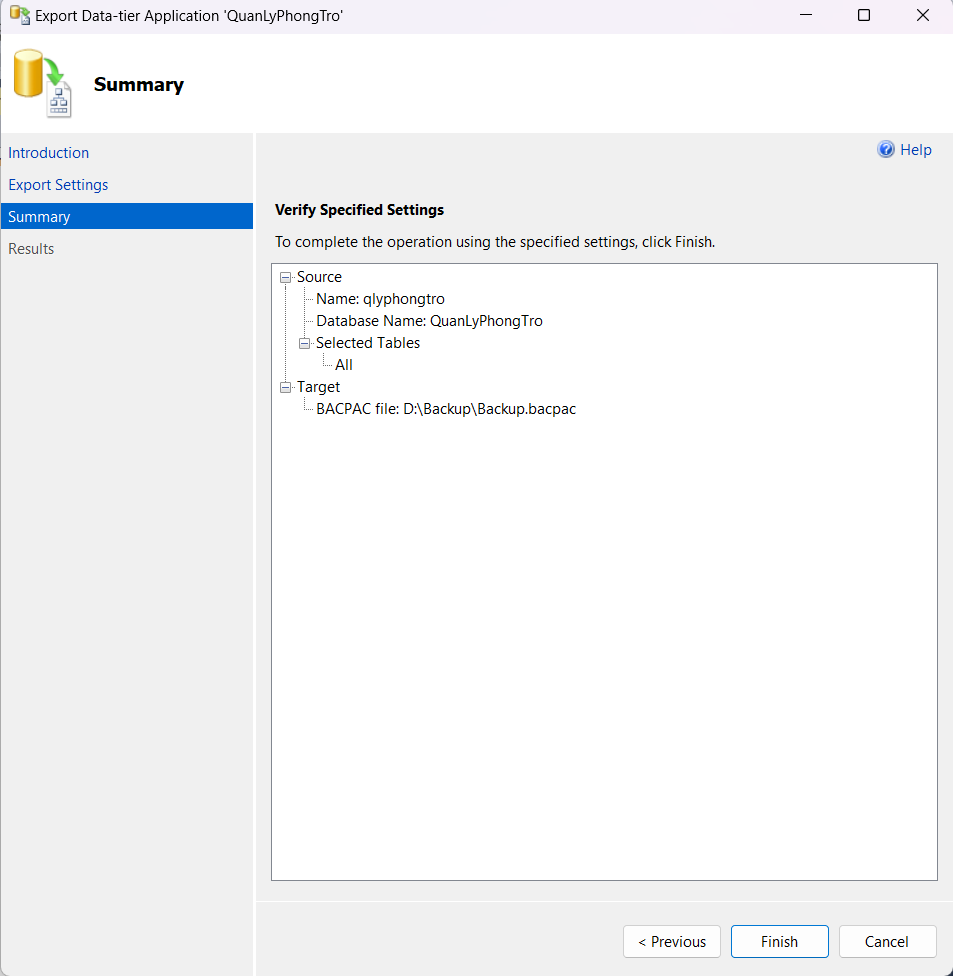
Hình 21 Chọn task và chọn Export Data-tier Application

****

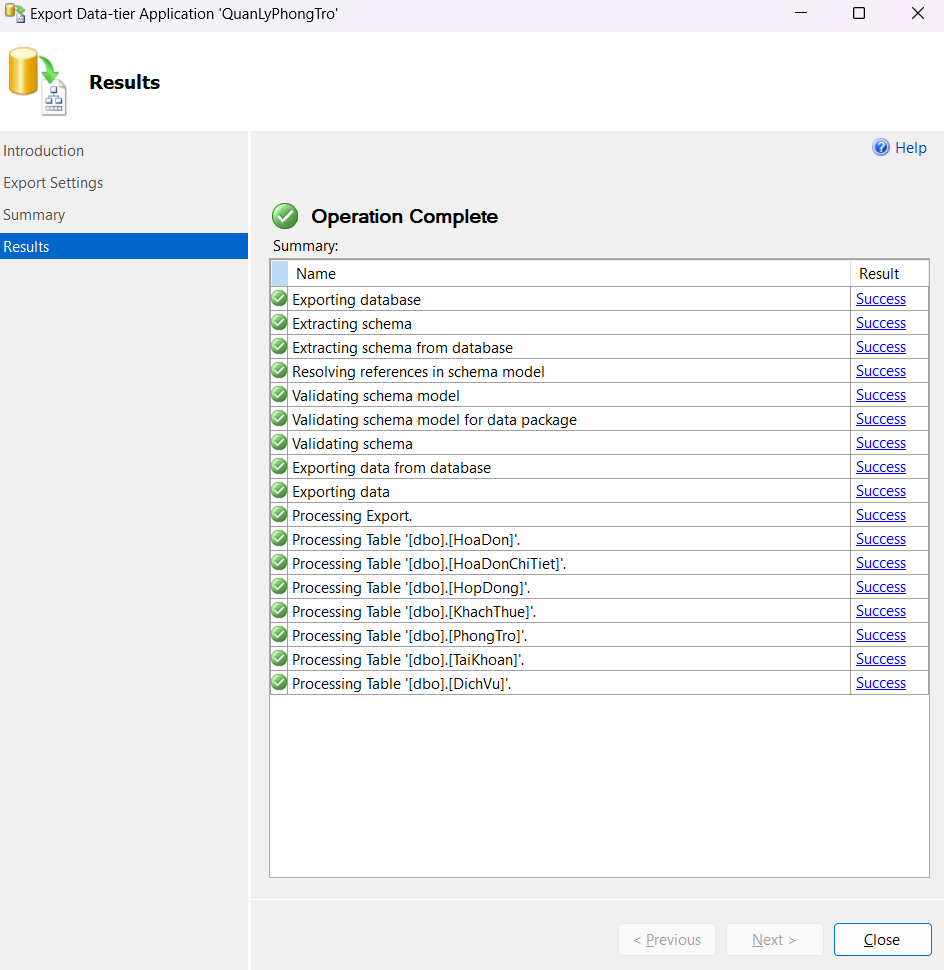
Hình 22 Xuất hiện cửa sổ Export Data-tier Application -> Chọn next

****

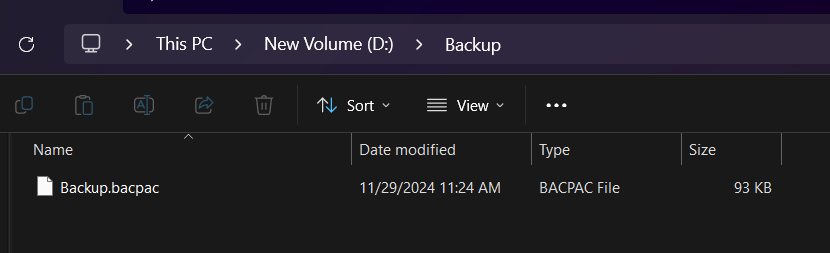
Hình 23 Chọn thư mục muốn tạo file backup và đặt tên file

****

Hình 24 Chọn Next -> Chọn Finish

****

Hình 25 Hoàn thành backup

****

Hình 26 Kiểm tra lại đã có file backup

# Phương án giải quyết vấn đề dữ liệu lớn

**1. Vấn đề hiện tại**

Hệ thống quản lý phòng trọ hiện tại đang sử dụng cơ sở dữ liệu SQL Server để lưu trữ thông tin về khách thuê, hợp đồng, thanh toán, và các thông tin liên quan khác. Khi lượng dữ liệu tăng lên do nhu cầu lưu trữ lâu dài và số lượng giao dịch ngày càng nhiều, dung lượng máy chủ nội bộ trở nên quá tải. Việc mở rộng hệ thống lưu trữ nội bộ đòi hỏi chi phí lớn và thiếu linh hoạt, đồng thời gây ra nguy cơ mất mát dữ liệu khi xảy ra lỗi phần cứng hoặc sự cố hệ thống.

Do đó, cần một giải pháp lưu trữ dữ liệu trên nền tảng đám mây nhằm đảm bảo an toàn, khả năng mở rộng linh hoạt, và hiệu quả về chi phí.

**2. Đề xuất giải pháp: Sử dụng Microsoft Azure Cloud**

Microsoft Azure Cloud là một trong những nền tảng đám mây phổ biến, cung cấp các dịch vụ lưu trữ và quản lý dữ liệu toàn diện, an toàn và hiệu quả. Đây là dịch vụ mà nhóm sử dụng:

Azure SQL Database:  
Hỗ trợ cơ sở dữ liệu quan hệ (SQL Server), đảm bảo hiệu suất cao, tính ổn định, và cung cấp khả năng sao lưu, phục hồi dữ liệu tự động.

**3. Lý do chọn Microsoft Azure**

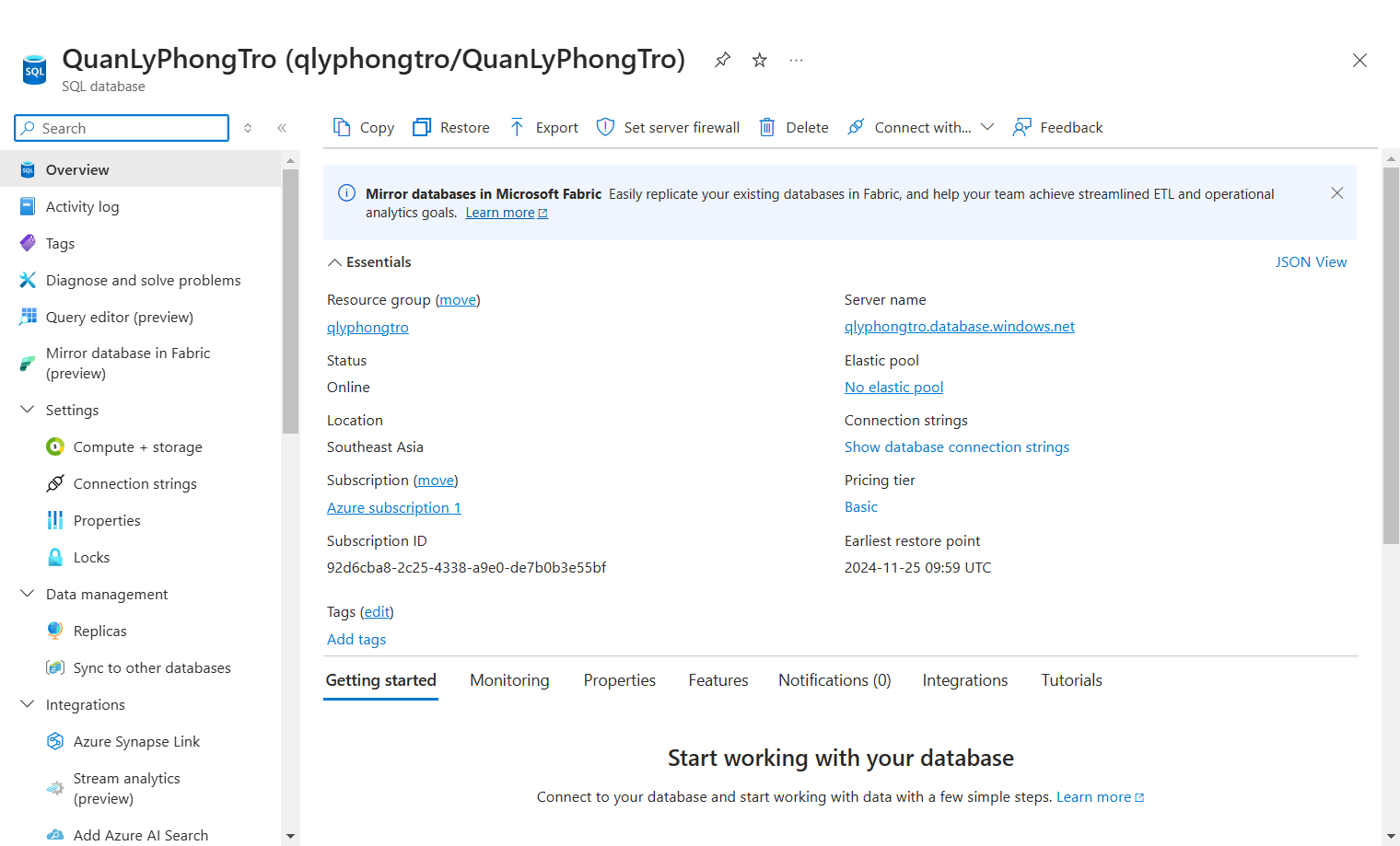
* Tính linh hoạt:  
  Azure cung cấp nhiều dịch vụ phù hợp với các nhu cầu lưu trữ khác nhau, từ truy cập thường xuyên đến lưu trữ lâu dài, và dễ dàng tích hợp với hệ thống hiện tại.
* Bảo mật cao:  
  Azure tuân thủ các tiêu chuẩn bảo mật nghiêm ngặt (như ISO 27001, GDPR), đảm bảo dữ liệu không bị mất mát hoặc truy cập trái phép.
* Khả năng mở rộng:  
  Azure hỗ trợ tăng dung lượng lưu trữ hoặc hiệu suất xử lý dữ liệu khi cần, mà không gây gián đoạn hệ thống.
* Chi phí tối ưu:  
  Chính sách "pay-as-you-go" giúp bạn chỉ trả chi phí dựa trên tài nguyên thực tế sử dụng, tránh lãng phí.
* Tích hợp dễ dàng:  
  Azure hỗ trợ các công cụ và dịch vụ sẵn có (như API, Azure Logic Apps) để tích hợp nhanh chóng với hệ thống quản lý phòng trọ.

**4. Hiệu quả mong đợi**

* Giảm áp lực lưu trữ nội bộ:  
  Tối ưu hóa lưu trữ bằng cách chuyển toàn bộ dữ liệu lên Azure, giảm gánh nặng cho máy chủ nội bộ.
* Cải thiện hiệu suất:  
  Tăng tốc độ truy cập dữ liệu, đảm bảo hoạt động của hệ thống quản lý phòng trọ được mượt mà.
* An toàn và bảo mật:  
  Dữ liệu được sao lưu định kỳ, bảo vệ trước các rủi ro như mất mát do lỗi kỹ thuật hoặc truy cập trái phép.
* Tiết kiệm chi phí lâu dài:  
  Thanh toán linh hoạt theo mức sử dụng, giảm thiểu chi phí vận hành không cần thiết.
* Khả năng mở rộng:  
  Khi số lượng phòng trọ hoặc dữ liệu tăng, hệ thống vẫn hoạt động ổn định mà không phải nâng cấp phần cứng.

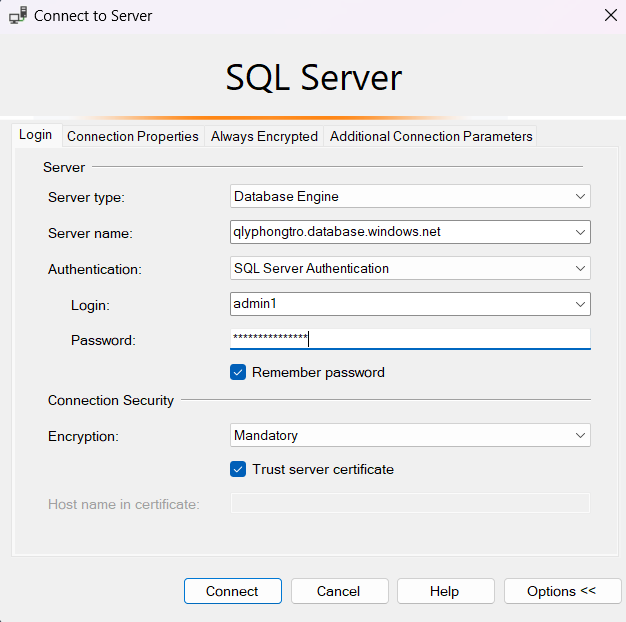
Microsoft Azure là giải pháp lý tưởng để đáp ứng nhu cầu lưu trữ dữ liệu lớn, bảo mật, và hỗ trợ nâng cao hiệu quả quản lý hệ thống phòng trọ.

* **Thông tin của server cloud vừa tạo:**



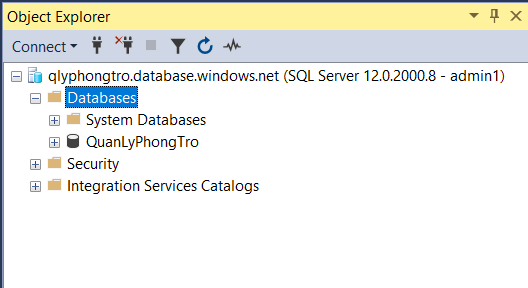
Hình 27 Thông tin của server cloud vừa tạo

* **Đăng nhập vào server cloud:**

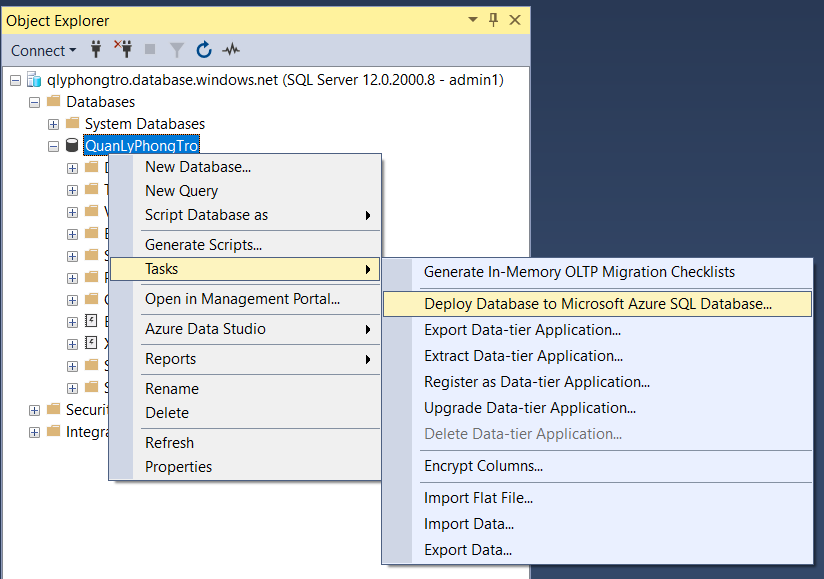


Hình 28 Màn hình đăng nhập vào server cloud

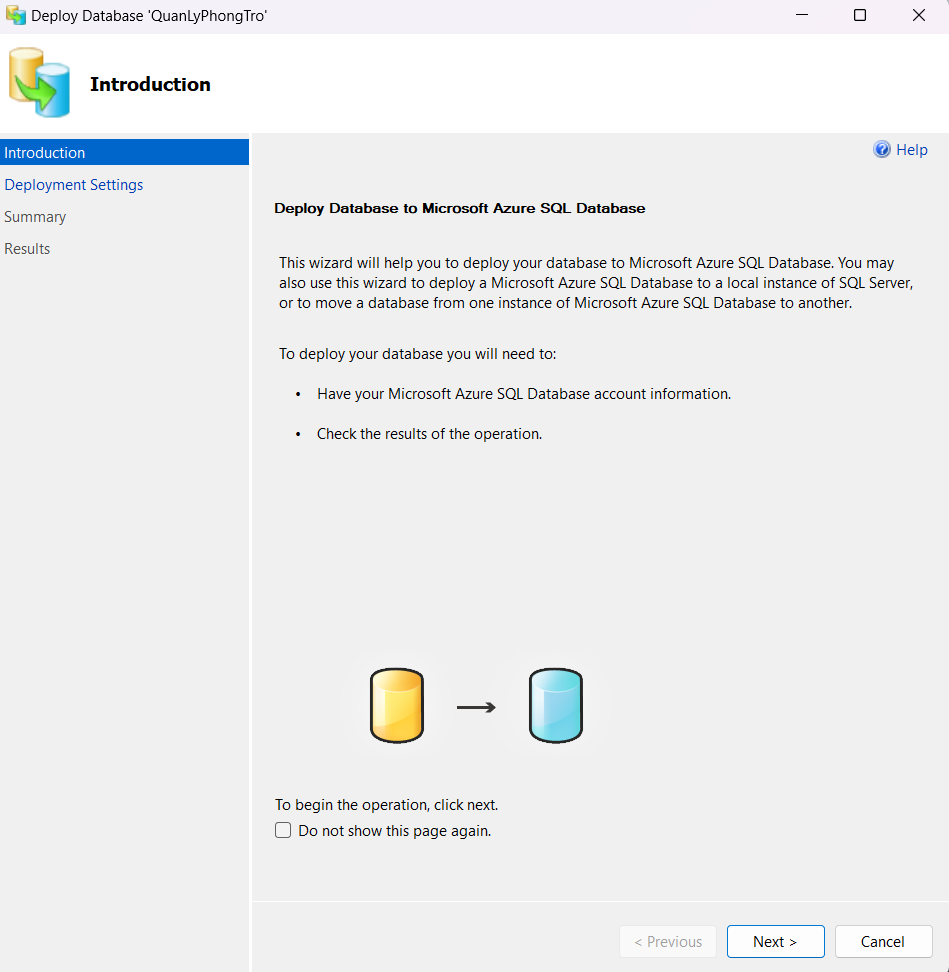
**→ Kết nối database thành công:**



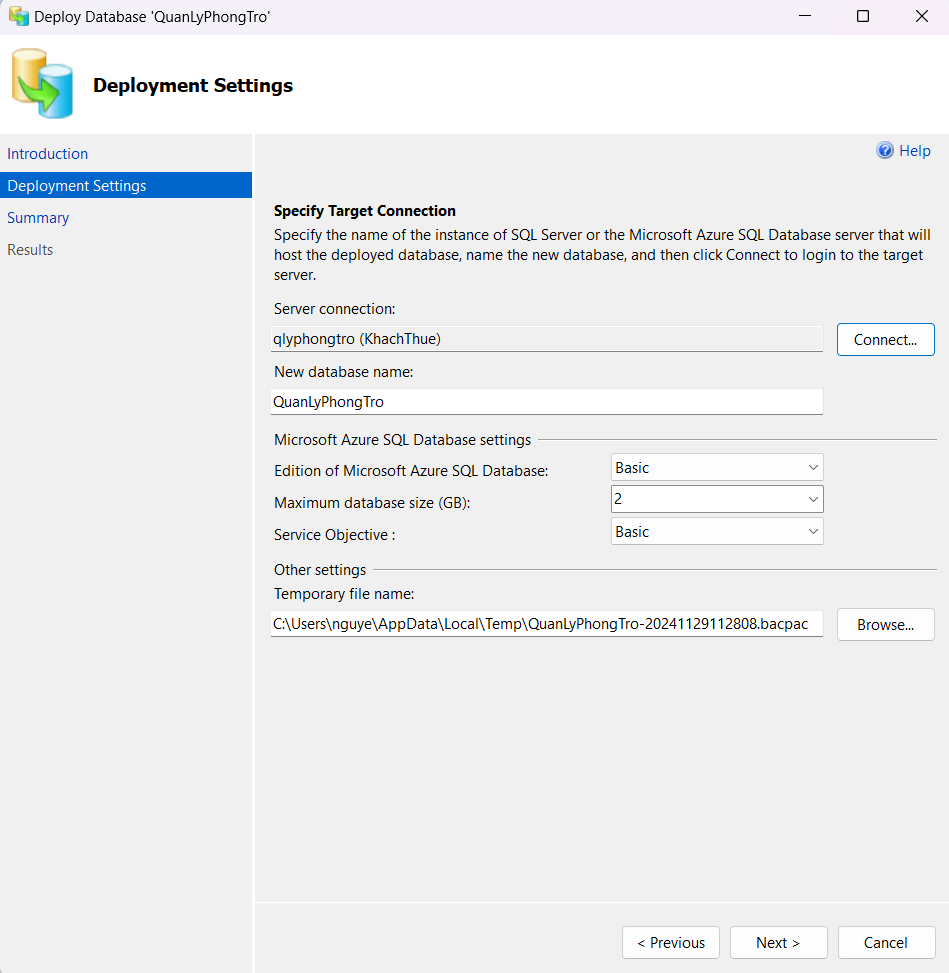
Hình 29 Màn hình kết quả sau khi kết nối database



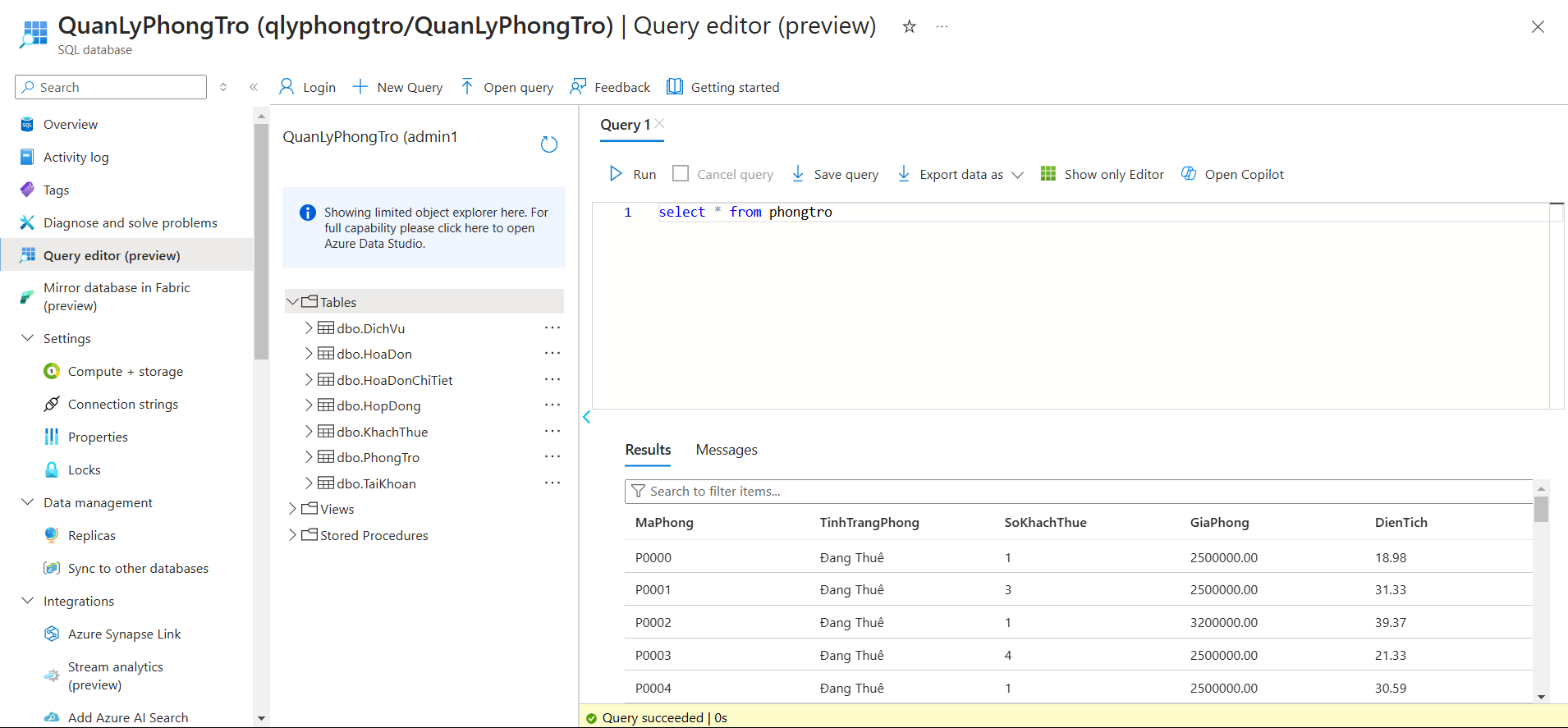
Hình 30 Chọn Tasks -> Deploy Database to Microsoft Azure SQL Database

****

Hình 31 Xuất hiện cửa sổ Deploy Database -> chọn Next



Hình 32 Nhập tên Database mới và chọn ổ đĩa lưu file tạm thời -> nhấn Next

****

Hình 33 Sau khi finish và deploy thành công database từ máy lên Microsoft Azure

# Đề phòng phương án hệ thống bị tấn công bằng SQL Injection:

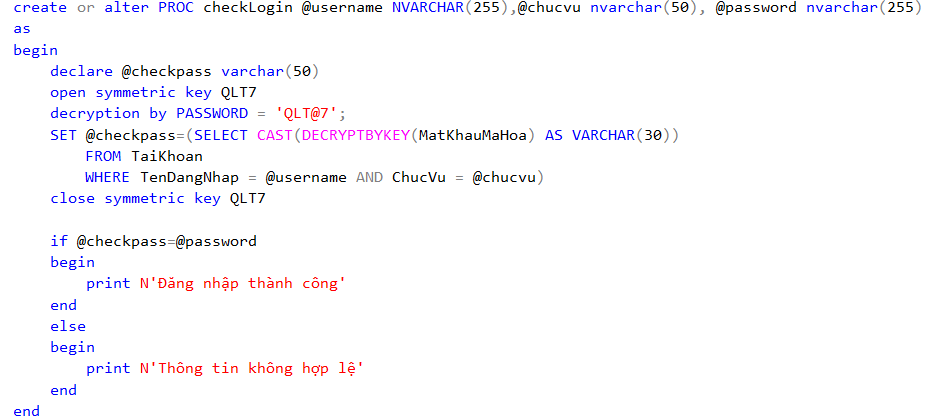
### Lý thuyết Injection:

SQL injection là một kỹ thuật cho phép những kẻ tấn công lợi dụng lỗ hổng của việc kiểm tra dữ liệu đầu vào trong các ứng dụng web và các thông báo lỗi của hệ quản trị cơ sở dữ liệu trả về để inject (tiêm vào) và thi hành các câu lệnh SQL bất hợp pháp. SQL injection có thể cho phép những kẻ tấn công thực hiện các thao tác, delete, insert, update, v.v. trên cơ sở dữ liệu của ứng dụng, thậm chí là server mà ứng dụng đó đang chạy.

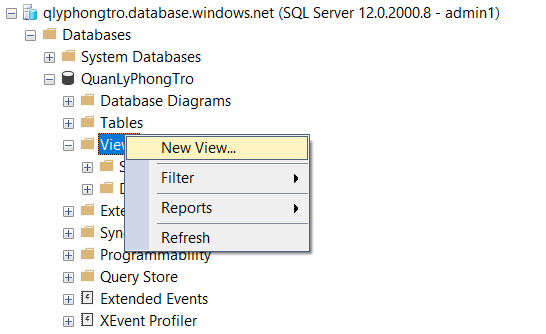
### Phương án phòng chống

Có thể đề phòng tấn công bằng SQL Injection bằng các biện pháp: Tham số hóa, tạo view ảo, phân quyền người dùng, Backup dữ liệu, …

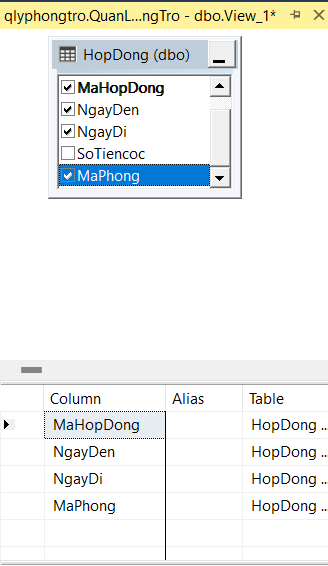
**Tham số hóa (Parameterized Queries)** là một phương pháp hiệu quả để chống **SQL Injection**, một lỗ hổng phổ biến trong các ứng dụng web. Kỹ thuật này đảm bảo rằng dữ liệu đầu vào từ người dùng không được phép can thiệp hoặc sửa đổi cú pháp của câu lệnh SQL

* Cách hoạt động: Thay vì nhúng trực tiếp dữ liệu người dùng vào chuỗi SQL, bạn sử dụng các tham số trong câu truy vấn và cung cấp giá trị của tham số đó riêng biệt. Điều này đảm bảo rằng dữ liệu từ người dùng được xử lý như **dữ liệu thô**, không phải là mã lệnh SQL.
* Ví dụ:

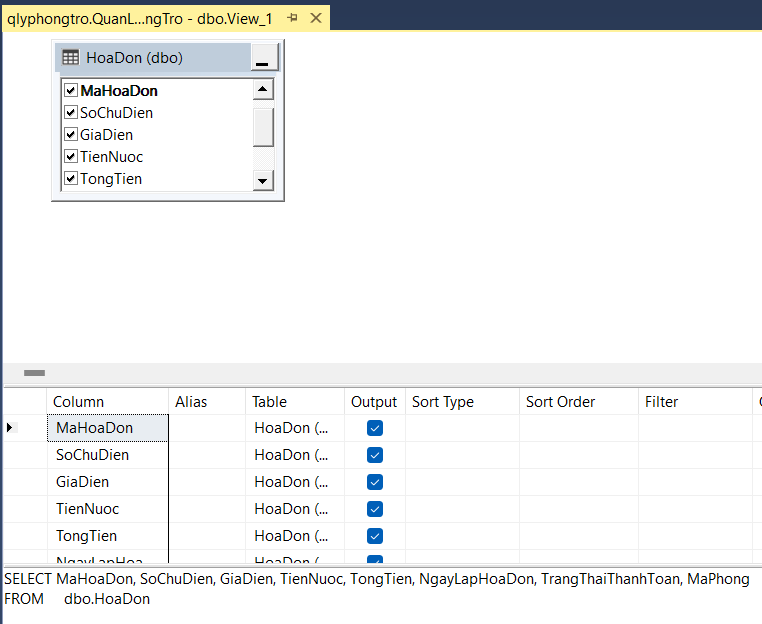
* **Tạo view ảo, chọn bảng cùng các thuộc tính muốn cho khách thuê có thể xem.**

****

Hình 34 Tạo view ảo

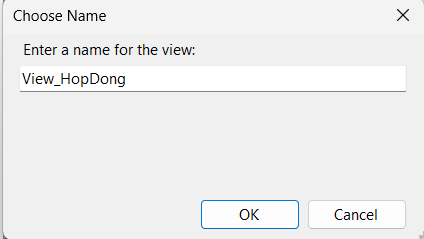
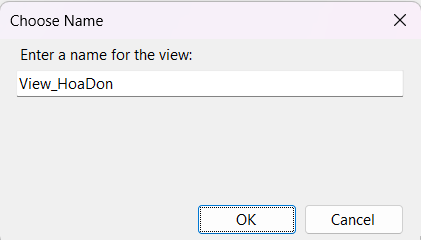
****

Hình 35 Chọn thuộc tính bảng HopDong

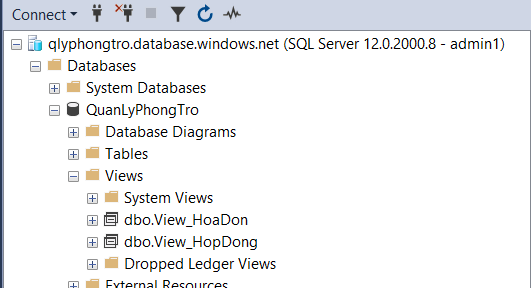


Hình 36 Chọn thuộc tính bảng HoaDon

* **Lưu và đặt tên cho bảng view vừa tạo:**

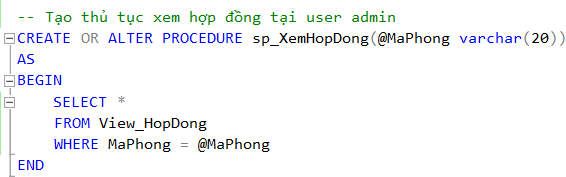
****

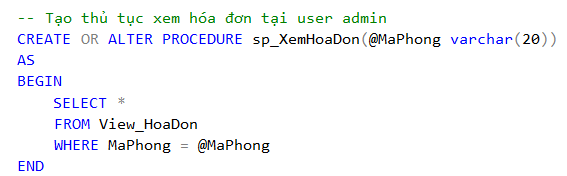
Hình 37 Lưu và đặt tên cho bảng view vừa tạo



Hình 38 Tạo 2 bảng view thành công

* **Admin thực hiện thủ tục sp\_XemHopDong và sp\_XemHoaDon:**

****

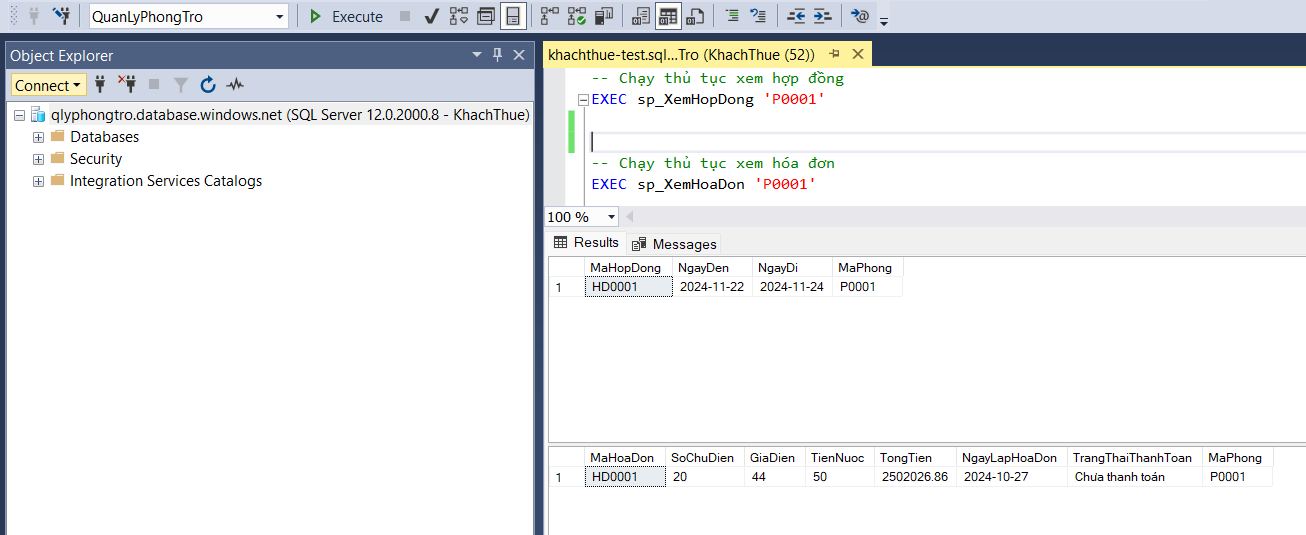


* **Admin thực hiện cấp quyền xem cho user khách thuê:**

**https://lh7-rt.googleusercontent.com/docsz/AD_4nXeiQhkfX9LKVmRw_V42N-BiohKlyWbpY6ZmPmCCPUF6Dbe1p61u8NihMz9MnVi7YgZDOOHgspeIwFT_TQzwQ3AnOyoypwMEKiRmMky6f9hFKwNwxtrBCLwMKw-w2KOLLeLr1e-VLw?key=hWM2xeL1TvJKw_Jykzo5UfSY**



* **Khách thuê thực hiện thủ tục và có kết quả như sau:**

****

Hình 39 Kết quả khách thuê thực hiện thủ tục

**→** Vừa bảo mật được các thuộc tính mình đã từ chối quyền xem và từ chối khách thuê truy vấn được dữ liệu của tất cả các khách thuê khác.

**-       Phân quyền người dùng:** Đã tạo login cho cơ sở dữ liệu ở R5, gồm:

·       Login ChuTro

·       Login KhachThue

Người dùng sẽ được phân quyền thành ChuTro hay KhachThue khi đăng nhập vào hệ thống. ChuTro có toàn quyền với cơ sở dữ liệu, còn KhachThue chỉ có quyền xem dữ liệu

**-        Backup dữ liệu:** Đã xây dựng cơ chế tự động backup dữ liệu tại R6

Có thể khôi phục dữ liệu từ file backup khi cần hoặc bị mất dữ liệu gốc

**------------ HẾT ------------**