CHƯƠNG 4: THỰC NGHIỆM

**A. Lựa chọn bài toán**

1. **Tổng quan:**

Ngày nay, với số lượng sinh viên ngày càng tăng cùng nhu cầu quản lý học tập chặt chẽ, việc xây dựng cơ sở dữ liệu để đăng ký học phần trở nên vô cùng cần thiết. Thông qua việc lưu trữ và cập nhật thông tin sinh viên, học phần, lớp học phần và kết quả đăng ký, hệ thống sẽ hỗ trợ sinh viên dễ dàng tra cứu, đăng ký và quản lý lịch học của mình. Đồng thời, nhà trường có thể kiểm soát số lượng đăng ký, điều kiện mở lớp và lập thời khóa biểu, từ đó đảm bảo quá trình học tập diễn ra thuận lợi, hiệu quả và đúng quy định.

1. **Mục tiêu:**

* Quản lý thông tin đăng ký học phần của sinh viên, theo dõi sinh viên nào đăng ký môn học nào, thuộc lớp nào, tình trạng đăng ký (đã đăng ký, hủy).
* Hỗ trợ công tác xếp lớp và tổ chức giảng dạy, dựa trên số lượng đăng ký, trường có thể mở đủ số lớp học phần, phân bổ giảng viên, phòng học.
* Cung cấp dữ liệu, báo cáo số lượng sinh viên đăng ký từng học phần, số lượng sinh đăng ký thành công hoặc hủy đăng ký.

1. **Yêu cầu:**
2. **Thiết kế cơ sở dữ liệu:**

* Phần này trình bày quá trình thiết kế cơ sở dữ liệu phục vụ cho hệ thống đăng ký học phần. Cụ thể:
* Mô tả các bảng dữ liệu chính: Sinh viên, Học phần, Giảng viên, Lớp học phần, Đăng ký.
* Xác định rõ các khóa chính, khóa ngoại và mối quan hệ giữa các bảng.
* Áp dụng các ràng buộc dữ liệu nhằm đảm bảo tính toàn vẹn và nhất quán thông tin.

### **Tạo dữ liệu mẫu**

* Tiến hành khởi tạo dữ liệu mẫu cho hệ thống để phục vụ quá trình kiểm thử và thực hiện truy vấn. Phần này bao gồm:
* Thêm thông tin sinh viên, học phần, giảng viên.
* Khởi tạo các lớp học phần cho từng học phần cụ thể trong học kỳ.
* Tạo dữ liệu đăng ký học phần của sinh viên.
* Đảm bảo dữ liệu có tính đa dạng để thử nghiệm các chức năng khác nhau của hệ thống.

### **Thực hiện các truy vấn SQL**

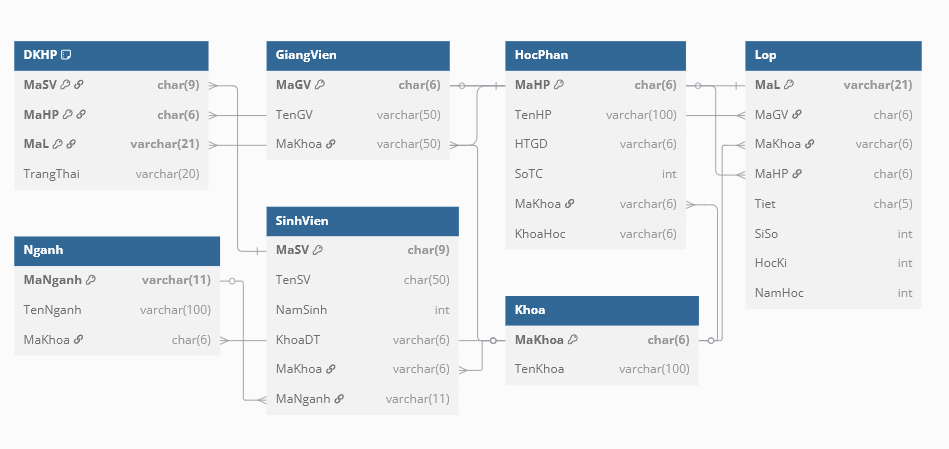
* Thực hiện các truy vấn kiểm chứng chức năng của hệ thống, bao gồm:
* Truy vấn thông tin sinh viên đã đăng ký học phần cụ thể.
* Thống kê số lượng sinh viên trong từng lớp học phần.
* Liệt kê danh sách học phần mà một sinh viên đã đăng ký
* Truy vấn thông tin lớp học phần theo học kỳ hoặc năm học.
* Các truy vấn liên quan đến giảng viên giảng dạy.

### **Đánh giá kết quả**

* Phần này đánh giá hiệu quả của hệ thống dựa trên dữ liệu thực nghiệm và truy vấn đã thực hiện, bao gồm:
* Tính chính xác: Các truy vấn cho kết quả đúng với dữ liệu đã nhập.
* Tính toàn vẹn dữ liệu: Không phát sinh lỗi do vi phạm ràng buộc khóa hoặc dữ liệu.
* Hiệu quả truy vấn: Truy vấn thực hiện nhanh với dữ liệu nhỏ; đề xuất tối ưu hóa khi dữ liệu lớn.
* Khả năng mở rộng: Thiết kế cơ sở dữ liệu có thể mở rộng để đáp ứng các yêu cầu phát sinh trong tương lai như quản lý lịch học, điều kiện mở môn, thêm ngành, môn mới hoặc xóa ngành, môn cũ,…

**B. Thiết kế mô hình dữ liệu**

1. **Sơ đồ ERD:**



1. **Lược đồ quan hệ:**

**KHOA**(MaKhoa, TenKhoa)

**NGANH**(MaNganh, TenNganh, MaKhoa)

**SINHVIEN**(MaSV, MaKhoa, MaNganh ,TenSV, NamSinh, KhoaDT)

**GIANGVIEN**(MaGV, MaKhoa ,TenGV)

**HOCPHAN**(MaHP, MaKhoa TenHP, HTGD, SoTC, KhoaHoc)

**LOP**(MaL, MaGV, MaKhoa, MaHP, Tiet, SiSo, HocKi, NamHoc)

**DKHP**(MaSV, MaHP, MaL, TrangThai)

1. **Mô tả kiểu dữ liệu:**

| Quan hệ | Thuộc tính | Kiểu dữ liệu | Diễn giải |
| --- | --- | --- | --- |
| KHOA | MaKhoa | varchar(5) | Khóa chính, Not null |
| TenKhoa | varchar(100) |  |
| NGANH | MaNganh | varchar(5) | Khóa chính, Not null |
| TenNganh | varchar(100) |  |
| MaKhoa | varchar(5) |  |
| SINHVIEN | MaSV | varchar(10) | Khóa chính, Not null |
| TenSV | varchar(100) |  |
| NamSinh | int |  |
| KhoaDT | varchar(5) |  |
| MaNganh | varchar(5) | tc NGANH |
| MaKhoa | varchar(5) | tc KHOA |
| GIANGVIEN | MaGV | varchar(10) | Khóa chính, Not null |
| TenGV | varchar(100) | Not null |
| MaKhoa | varchar(5) | tc KHOA |
| HOCPHAN | MaHP | varchar(10) | Khóa chính, Not null |
| TenHP | varchar(100) |  |
| SoTC | int |  |
| MaKhoa | varchar(5) | tc KHOA |
| HTGD | varchar(10) |  |
| LOP | MaL | varchar(5) | Khóa chính, Not null |
| MaGV | varchar(10) | tc GIANGVIEN |
| MaKhoa | varchar(5) | tc KHOA |
| MaHP | varchar(10) | tc HOCPHAN |
| Tiet | varchar(10) |  |
| SiSo | int |  |
| HocKi | int |  |
| NamHoc | int |  |
| DKHP | MaSV | varchar(10) | tc SINHVIEN |
| MaHP | varchar(10) | tc HOCPHAN |
| MaL | varchar(10) | tc LOP |
| TrangThai | varchar(50) |  |

1. **Tạo cơ sở dữ liệu trên IBM DB2**

- Tạo cơ sở dữ liệu, bảng và định nghĩa ràng buộc (PRIMARY KEY, FOREIGN KEY):

| */\**  *drop table DKHP;*  *drop table GiangVien;*  *drop table HocPhan;*  *drop table Lop;*  *drop table Nganh;*  *drop table SinhVien;*  *drop table Khoa;*  *\*/*  *CREATE TABLE DKHP(*  *MaSV char(9) NOT NULL,*  *MaHP char(6) NOT NULL,*  *MaL varchar(21) NOT NULL,*  *TrangThai varchar(20) NULL,*  *constraint pk\_dkhp primary key (MaSV, MaHP, MaL)*  *);*  *CREATE TABLE GiangVien(*  *MaGV char(6) NOT NULL,*  *TenGV varchar(50) NULL,*  *MaKhoa varchar(50) NULL,*  *constraint pk\_giangvien primary key (MaGV)*  *);*  *CREATE TABLE HocPhan(*  *MaHP char(6) NOT NULL,*  *TenHP varchar(100) NULL,*  *HTGD varchar(6) NULL,*  *SoTC int NULL,*  *MaKhoa varchar(6) NULL,*  *KhoaHoc varchar(6) NULL,*  *constraint pk\_hocphan primary key (MaHP)*  *);*  *CREATE TABLE Lop(*  *MaL varchar(21) NOT NULL,*  *MaGV char(6) NULL,*  *MaKhoa varchar(6) NULL,*  *MaHP char(6) NULL,*  *Tiet char(5) NULL,*  *SiSo int NULL,*  *HocKi int NULL,*  *NamHoc int NULL,*  *constraint pk\_lop primary key (MaL)*  *);*  *CREATE TABLE Nganh(*  *MaNganh varchar(11) NOT NULL,*  *TenNganh varchar(100) NULL,*  *MaKhoa char(6),*  *constraint pk\_nganh primary key (MaNganh)*  *);*  *CREATE TABLE SinhVien(*  *MaSV char(9) NOT NULL,*  *TenSV char(50) NULL,*  *NamSinh int NULL,*  *KhoaDT varchar(6) NULL,*  *MaKhoa varchar(6) NULL,*  *MaNganh varchar(11) NULL,*  *constraint pk\_sinhvien primary key (MaSV)*  *);*  *CREATE TABLE Khoa(*  *MaKhoa char(6) NOT NULL,*  *TenKhoa varchar(100) NULL,*  *constraint pk\_khoa primary key (MaKhoa)*  *);*  *alter table DKHP*  *add constraint fk\_dkhp\_masv*  *foreign key (MaSV) references SinhVien (MaSV);*  *alter table DKHP*  *add constraint fk\_dkhp\_mahp*  *foreign key (MaHP) references HocPhan (MaHP);*  *alter table DKHP*  *add constraint fk\_dkhp\_mal*  *foreign key (MaL) references Lop (MaL);*  *alter table GiangVien*  *add constraint fk\_giangvien\_makhoa*  *foreign key (MaKhoa) references Khoa (MaKhoa);*  *alter table HocPhan*  *add constraint fk\_hocphan\_makhoa*  *foreign key (MaKhoa) references Khoa (MaKhoa);*  *alter table Lop*  *add constraint fk\_lop\_magv*  *foreign key (MaGV) references GiangVien (MaGV);*  *alter table Lop*  *add constraint fk\_lop\_makhoa*  *foreign key (MaKhoa) references Khoa (MaKhoa);*  *alter table Lop*  *add constraint fk\_lop\_mahp*  *foreign key (MaHP) references HocPhan (MaHP);*  *alter table Nganh*  *add constraint fk\_nganh\_makhoa*  *foreign key (MaKhoa) references Khoa (MaKhoa);*  *alter table SinhVien*  *add constraint fk\_sinhvien\_makhoa*  *foreign key (MaKhoa) references Khoa (MaKhoa);*  *alter table SinhVien*  *add constraint fk\_sinhvien\_manganh*  *foreign key (MaNganh) references Nganh (MaNganh);*  *--#SET TERMINATOR @*  *create trigger trg\_ins\_sinhvien*  *after insert on SinhVien*  *referencing new as N*  *for each row*  *begin*  *if N.MaNganh not in (select MaNganh from Nganh where MaKhoa = N.MaKhoa)*  *then*  *signal sqlstate '75000' set message\_text='Loi: Ma nganh khong thuoc Khoa';*  *rollback;*  *end if;*  *end@*  *--#SET TERMINATOR @*  *create trigger trg\_upd\_sinhvien*  *after update of MaNganh, MaKhoa on SinhVien*  *referencing new as N*  *for each row*  *begin*  *if N.MaNganh not in (select MaNganh from Nganh where MaKhoa = N.MaKhoa)*  *then*  *signal sqlstate '75000' set message\_text='Loi: Ma nganh khong thuoc Khoa';*  *rollback;*  *end if;*  *end@* |
| --- |

1. Thêm dữ liệu:

| *insert into SinhVien*  *values*  *('23521300','Ho Anh Quoc','2005','K18','ISE','CNVN'),*  *('23520198','Nguyen Cao Cuong','2005','K18','ISE','CNTT'),*  *('23520255','Ngo Tran Phat Dat','2005','K18','ISE','CNTT'),*  *('23520916','Nguyen Tan Manh','2005','K18','ISE','CNTT'),*  *('23521243','Nguyen Ky Phuong','2005','K18','ISE','CNVN'),*  *('23520401','Bui Tung Lam','2005','K18','ISE','KHDL'),*  *('23520402','Tran Nhat Quang','2005','K18','CS','KHMT'),*  *('22520403','Le Dai Phat','2004','K17','NC','MMTTT'),*  *('23520404','Nguyen Thanh Huong','2005','K18','SE','KTPM'),*  *('23520405','Pham Minh Chi','2005','K18','SE','KTPM'),*  *('23520406','Dang Hai Dang','2005','K18','CE','KTMT'),*  *('22520407','Hoang Vu Bao','2004','K17','NC','ATTT'),*  *('21520408','Vo Ngoc Lam','2003','K16','IS','HTTT'),*  *('22520409','Le Phong Luu','2004','K17','CS','KHMT'),*  *('23520410','Duong Do Minh','2005','K18','ISE','KHDL'),*  *('23520411','Ly Thanh Truong','2005','K18','ISE','CNTT'),*  *('22520412','Cao Truong Son','2004','K17','CS','KHMT'),*  *('23520413','Ho Thao Nguyen','2005','K18','NC','MMTTT'),*  *('23520414','Do Nhu Y','2005','K18','IS','HTTT'),*  *('22520415','Vu Thien Huy','2004','K17','CS','TTNT'),*  *('22520416','Ngo Xuan Loc','2004','K17','CE','TKVM'),*  *('22520417','Tran Hoai Giang','2004','K17','NC','ATTT'),*  *('21520418','Trinh Dang Khoi','2003','K16','SE','KTPM'),*  *('23520419','Do Bao Chau','2005','K18','ISE','CNTT'),*  *('23520420','Tran Minh Nghi','2005','K18','IS','TMDT');*  *insert into GiangVien*  *values*  *('80275','Ta Thu Thuy','ISE'),*  *('80276','Do Thi Ngoc Tu','NC'),*  *('80277','Nguyen Thanh Thien','CS'),*  *('80278','Ho Nhat Anh','CE'),*  *('80279','Nguyen Thi Thu Huong','SE'),*  *('80280','Nguyen Doan Nhat','IS'),*  *('80281','Le Thanh Dung','MA'),*  *('80282','Luu Thuy Ngan','ISE'),*  *('80284','Dang Anh Tu','SE'),*  *('80285','Pham Thanh Duc','CE'),*  *('80286','Do Yen Linh','MA'),*  *('80287','Nguyen Van Khang','ISE'),*  *('80288','Huynh Trong Nghia','NC');*  *insert into HocPhan*  *values*  *('IE103','Quan li thong tin','LT','4','ISE','K18'),*  *('IS402','Dien toan dam may','LT','3','IS','K17'),*  *('SE310','Cong nghe .Net','LT','4','SE','K17'),*  *('DS102','Hoc may thong ke','LT','4','ISE','K17'),*  *('NT213','Bao mat va ung dung web','LT','4','NC','K16'),*  *('MA006','Giai tich','LT','4','MA','K19'),*  *('CS106','Tri tue nhan tao','LT','4','CS','K18'),*  *('IE104','Internet va cong nghe Web','LT','4','ISE','K18'),*  *('CE103','Xu li vi dieu khien','LT','4','CE','K17');*  *insert into Lop*  *values*  *('IE103.P21','80275','ISE','IE103','6789','70','2','2024'),*  *('IE103.P22','80287','ISE','IE103','1234','70','2','2024'),*  *('IS402.P25','80276','IS','IS402','123','40','2','2024'),*  *('SE310.P23','80279','SE','SE310','1234','70','2','2024'),*  *('SE310.P24','80284','SE','SE310','6789','70','2','2024'),*  *('DS102.P22','80282','ISE','DS102','1234','50','2','2024'),*  *('DS102.P23','80287','ISE','DS102','6789','50','2','2024'),*  *('NT213.P27','80288','NC','NT213','2345','40','2','2024'),*  *('MA006.P29','80281','MA','MA006','1234','100','2','2024'),*  *('MA006.P28','80286','MA','MA006','6789','100','2','2024'),*  *('CS106.P21','80277','CS','CS106','6789','50','2','2024'),*  *('IE104.P23','80275','ISE','IE104','1234','70','2','2024'),*  *('IE104.P24','80287','ISE','IE104','6789','70','2','2024'),*  *('CE103.P26','80285','CE','CE103','1234','50','2','2024'),*  *('CE103.P27','80278','CE','CE103','6789','50','2','2024');*  *insert into DKHP*  *values*  *('23521300','IE103','IE103.P21','Da dang ky'),*  *('23520198','IE103','IE103.P21','Da dang ky'),*  *('23520255','IE103','IE103.P21','Da dang ky'),*  *('23520916','IE103','IE103.P21','Da dang ky'),*  *('23521243','IE103','IE103.P21','Da dang ky'),*  *('23520401','DS102','DS102.P22','Huy'),*  *('23520402','CS106','CS106.P21','Da dang ky'),*  *('22520403','NT213','NT213.P27','Da dang ky'),*  *('23520404','SE310','SE310.P23','Da dang ky'),*  *('23520405','SE310','SE310.P24','Da dang ky'),*  *('23520406','CE103','CE103.P26','Huy'),*  *('22520407','NT213','NT213.P27','Da dang ky'),*  *('21520408','IS402','IS402.P25','Da dang ky'),*  *('22520409','CS106','CS106.P21','Da dang ky'),*  *('23520410','DS102','DS102.P22','Da dang ky'),*  *('23520411','IE104','IE104.P23','Da dang ky'),*  *('23520413','NT213','NT213.P27','Huy'),*  *('23520414','IS402','IS402.P25','Da dang ky'),*  *('22520415','CS106','CS106.P21','Da dang ky'),*  *('22520416','CE103','CE103.P26','Da dang ky'),*  *('22520417','NT213','NT213.P27','Da dang ky'),*  *('21520418','MA006','MA006.P28','Da dang ky'),*  *('23520419','IE103','IE104.P24','Da dang ky'),*  *('23520420','IE103','IE104.P24','Da dang ky');*  *insert into Khoa*  *values*  *('ISE','Khoa Khoa hoc va Ky thuat Thong tin'),*  *('SE','Khoa Ky thuat Phan mem'),*  *('CS','Khoa Khoa hoc May tinh'),*  *('CE','Khoa Ky thuat May tinh'),*  *('NC','Khoa Mang may tinh va Truyen thong'),*  *('IS','Khoa He thong Thong tin'),*  *('MA','Bo mon Toan - Ly');*  *insert into Nganh*  *values*  *('CNTT','Cong nghe Thong tin','ISE'),*  *('KHDL','Khoa hoc Du lieu','ISE'),*  *('CNVN','Cong nghe Thong tin Viet - Nhat','ISE'),*  *('KTPM','Ky thuat Phan mem','SE'),*  *('KHMT','Khoa hoc May tinh','CS'),*  *('TTNT','Tri tue Nhan tao','CS'),*  *('KTMT','Ky thuat May tinh','CE'),*  *('IOT','He thong Nhung va IOT','CE'),*  *('TKVM','Thiet ke Vi mach','CE'),*  *('TMDT','Thuong mai Dien tu','IS'),*  *('HTTT','He thong Thong tin','IS'),*  *('MMTTT','Mang may tinh va Truyen thong','NC'),*  *('ATTT','An toan thong tin','NC');* |
| --- |

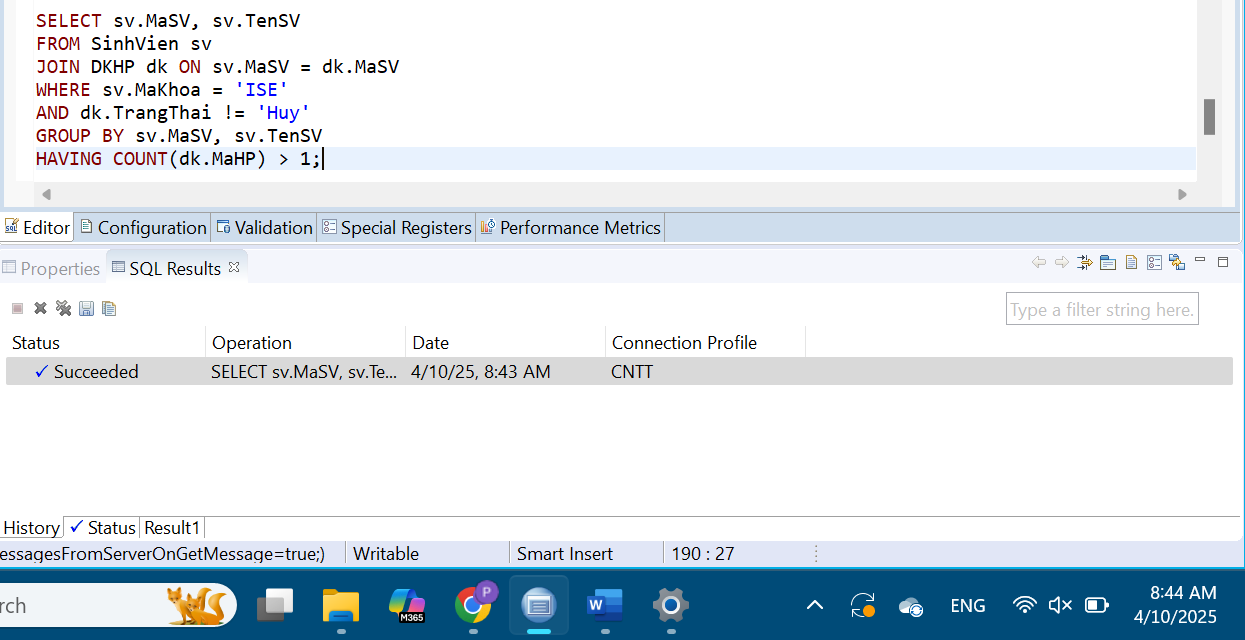
**C. Kết quả thực nghiệm**

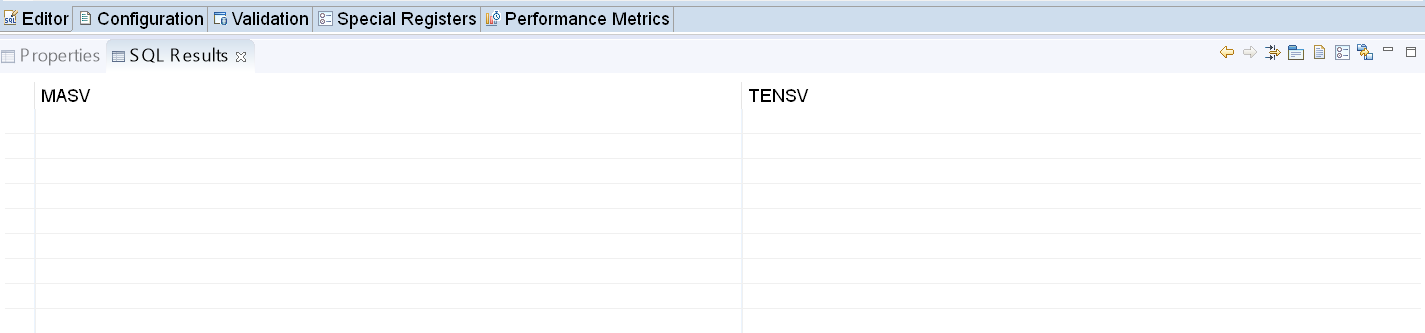
1. **Truy vấn cơ bản:**

a. Tìm tên các sinh viên thuộc khoa "ISE" đã đăng ký nhiều hơn 1 học phần và không có học phần nào bị huỷ.

| *select sv.MaSV, sv.TenSV*  *from SinhVien sv*  *join DKHP dk ON sv.MaSV = dk.MaSV*  *where sv.MaKhoa = 'ISE'*  *and dk.TrangThai != 'Huy'*  *group by sv.MaSV, sv.TenSV*  *having count(dk.MaHP) > 1;* |
| --- |

Kết quả:

****

****

* Kết Quả Truy Vấn
* Truy vấn trả về hai cột: MaSV (mã sinh viên) và TenSV (tên sinh viên).
* Kết quả chỉ bao gồm các sinh viên thuộc khoa "ISE", có trạng thái đăng ký học phần hợp lệ, và đã đăng ký ít nhất 2 học phần.

Giải thích:

* Các bước thực hiện:
* Lọc các sinh viên thuộc khoa "ISE" trong bảng SinhVien.
* Kết hợp với bảng DKHP để xác định các học phần mà sinh viên đã đăng ký.
* Đảm bảo các sinh viên không có học phần nào bị huỷ trong bảng DKHP (tức là TrangThai!='Huy').
* Tính số lượng học phần đã đăng ký của mỗi sinh viên, và chỉ lấy các sinh viên có số học phần đăng ký lớn hơn 1.
* Giải thích dưới dạng mô hình dữ liệu quan hệ:
* **Bảng SinhVien (sv)**
* Cột: MaSV (Mã Sinh Viên là khóa chính), TenSV (Tên Sinh Viên), MaKhoa (Mã Khoa là khóa ngoại tham chiếu đến thuộc tính MaKhoa trong quan hệ Khoa)
* Chức năng: Bảng này chứa thông tin của các sinh viên. Mỗi sinh viên sẽ có một mã số duy nhất (MaSV) và thuộc một khoa cụ thể (MaKhoa).
* **Bảng DKHP (dk)**
* Cột:
  + MaSV (Mã Sinh Viên), MaHP (Mã Học Phần), TrangThai (Trạng Thái Đăng Ký)
  + Trong đó cột MaSV và MaHP đồng thời là khóa chính và khóa ngoại:
* Chức năng: Bảng này lưu trữ thông tin về việc đăng ký học phần của các sinh viên. Cột TrangThai cho biết tình trạng đăng ký học phần (ví dụ: "Huy" có thể là trạng thái hủy đăng ký).
* **Mối Quan Hệ Giữa Các Bảng**
* Khóa Ngoại (Foreign Key):
  + Cột MaSV trong bảng DKHP là khóa ngoại tham chiếu đến cột MaSV trong bảng SinhVien.
  + Cột MaHP trong bảng DKHP là khóa ngoại tham chiếu đến cột MaHP trong bảng HocPhan
  + Điều này cho thấy mỗi bản ghi trong bảng DKHP sẽ liên kết với một sinh viên cụ thể trong bảng SinhVien.
* Mối quan hệ: *1 sinh viên có thể đăng ký* ***nhiều*** *học phần và 1 học phần có thể được đăng ký bởi* ***nhiều*** *sinh viên.*
* **Phân Tích Truy Vấn**
* Truy vấn SQL yêu cầu thực hiện các bước sau:
  + Lọc Sinh Viên Theo Mã Khoa (MaKhoa = 'ISE'):
  + Kết Hợp Bảng SinhVien và DKHP (JOIN):
  + Câu lệnh **JOIN DKHP dk ON sv.MaSV = dk.MaSV** kết hợp thông tin từ bảng SinhVien và bảng DKHP dựa trên mối quan hệ giữa cột MaSV. Điều này giúp lấy thông tin đăng ký học phần của các sinh viên.
  + Câu lệnh **WHERE sv.MaKhoa = 'ISE'** chỉ lấy những sinh viên thuộc khoa "ISE" từ bảng SinhVien.
  + Lọc Trạng Thái Đăng Ký:
    - Điều kiện **AND dk.TrangThai != 'Huy'** chỉ giữ lại các bản ghi mà trạng thái đăng ký học phần không phải là "Huy".
  + Nhóm Dữ Liệu (GROUP BY):
    - Câu lệnh **GROUP BY sv.MaSV, sv.TenSV** nhóm kết quả theo MaSV, TenSV. Điều này có nghĩa là sẽ cho kết quả DKHP của các học phần đã đăng ký cho mỗi sinh viên
  + Điều Kiện Đếm Học Phần:
    - **Điều kiện HAVING COUNT(dk.MaHP) > 1** yêu cầu chỉ những sinh viên có số lượng học phần đăng ký lớn hơn 1 mới đủ điều kiện hiển thị

b. Liệt kê tên học phần, tên giảng viên và tổng số sinh viên đã đăng ký (không bị huỷ) cho mỗi lớp, chỉ lấy các lớp có trên 3 sinh viên.

| *select TenHP, TenGV, count(MaSV) as TongSoSV*  *from HOCPHAN HP*  *join DKHP DK on HP.MaHP = DK.MaHP*  *join LOP L on L.MaHP = DK.MaHP*  *join GIANGVIEN GV on GV.MaGV = L.MaGV*  *where TrangThai = 'Da dang ky'*  *group by DK.MaHP, TenHP, TenGV*  *having count(MaSV) > 3;* |
| --- |

Kết quả:

**A screenshot of a computer

AI-generated content may be incorrect.**

Giải thích:

* Truy vấn thực hiện trên 4 quan hệ:
* **HOCPHAN(MaHP, TenHP, HTGD, SoTC, MaKhoa, KhoaHoc)**: chứa thông tin học phần.
* **DKHP(MaSV, MaHP, MaL, TrangThai)**: chứa thông tin sinh viên đăng ký học phần.
* **LOP(MaL, MaGV, MaKhoa, MaHP, Tiet, SiSo, HocKi, NamHoc)**: chứa thông tin của lớp học.
* **GIANGVIEN(MaGV, TenGV, MaKhoa)**: chứa thông tin của giảng viên.
* Các bước thực hiện:

1. Chọn các bản ghi trong quan hệ **DKHP** có **TrangThai = ‘Da dang ky’**.
2. Kết với quan hệ **HOCPHAN** để có **TenHP.**
3. Kết với quan hệ **LOP** để có **MaGV**.
4. Thông qua **MaGV** của **LOP**, kết với quan hệ **GIANGVIEN** để có **TenGV**.
5. Nhóm các hàng theo **MaHP, TenHP, TenGV** và lọc lấy những hàng có **COUNT(MaSV) > 3.**

c. Tìm tên ngành có số sinh viên đăng ký học phần “IE103" lớn nhất.

| *select ng.TenNganh*  *from DKHP dk, SinhVien sv, Nganh ng*  *where dk.MaSV = sv.MaSV and sv.MaNganh = ng.MaNganh*  *and dk.MaHP = 'IE103'*  *group by ng.TenNganh*  *order by count(\*) desc*  *fetch first 1 rows only;* |
| --- |

Kết quả:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Giải thích:

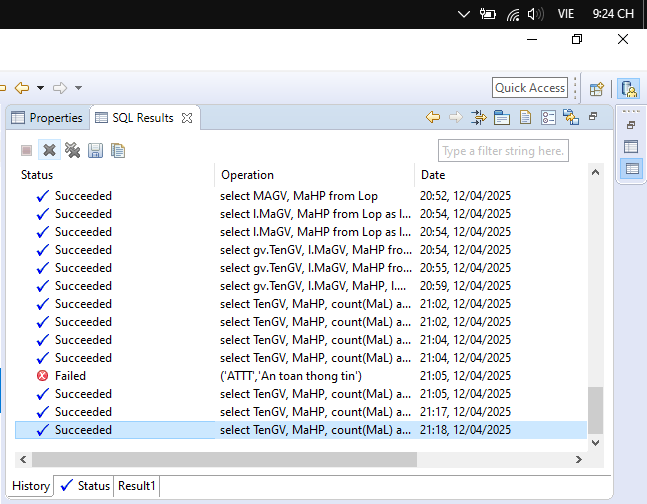
* Truy vấn thực hiện trên ba quan hệ:
* DKHP(MaSV, MaHP, MaL, ...): lưu thông tin sinh viên đăng ký học phần.
* SinhVien(MaSV, TenSV, MaNganh, ...): chứa thông tin sinh viên, liên kết với ngành qua MaNganh.
* Nganh(MaNganh, TenNganh): cung cấp tên ngành tương ứng.
* Các bước thực hiện:

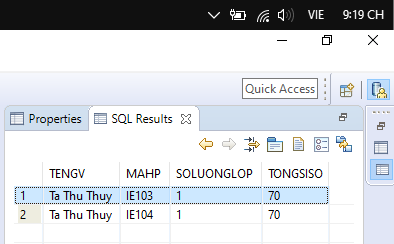
1. **Chọn** các bản ghi trong quan hệ DKHP có MaHP = 'IE103'.
2. **Kết nối** với SinhVien qua MaSV để lấy thông tin sinh viên.
3. **Kết nối** tiếp với Nganh qua MaNganh để truy xuất TenNganh.
4. **Nhóm** kết quả theo TenNganh và **đếm** số lượt đăng ký.
5. **Sắp xếp** giảm dần theo số lượng và **chọn ngành đầu tiên**.

* Kết quả trả về là **tên ngành có số sinh viên đăng ký học phần "IE103" nhiều nhất**.

d. Liệt kê các học phần do giảng viên tên chứa 'Thu Thuy' giảng dạy, kèm số lượng lớp và tổng sĩ số của mỗi học phần.

| *select TenGV, MaHP, count(MaL) as SoLuongLop, sum(SiSo) as TongSiSo*  *from Lop as l, GiangVien as gv*  *where l.MaGV=gv.MaGV and TenGV like '%Thu Thuy%'*  *group by MaHP, TenGV;* |
| --- |

Kết quả:



Kết quả truy vấn:

+ Kết quả trả về 4 cột là **TENGV, MAHP, SOLUONGLOP, TONGSISO.**

+ Kết quả có 2 dòng, các dòng này là kết quả của truy vấn tìm các học phần do giáo viên có tên chứa ‘Thu Thuy’ giảng dạy, **SOLUONGLOP** là số lượng lớp thuộc học phần đó, **TONGSISO** là tổng sĩ số sinh viên tham gia học phần đó.

Giải thích:

*\*Các bước thực hiện:*

- Lọc các học phần do giảng viên có tên chứa “Thu Thuy” trong bảng GiangVien kết với bảng **Lop** thông qua thuộc tính **MaGV**.

- Dùng group by theo **MaHP** và **TenGV** để thực hiện tính toán:

+ **count(MaL)** để tính số lượng lớp thuộc mỗi học phần, đặt alias là **SOLUONGLOP**.

+ **sum(SiSo)** để tính tổng sĩ số của mỗi học phần, đặt alias là **TONGSISO**.

- Phân Tích Truy Vấn

Truy vấn SQL yêu cầu thực hiện các bước sau:

+ Kết hợp Bảng **GiangVien** và **Lop** theo câu lệnh **where l.MaGV=gv.MaGV.**

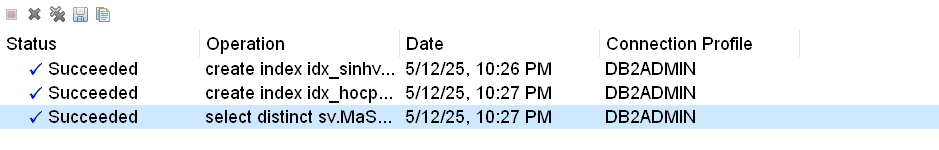
+ Điều kiện **TenGV** like ‘%Thu Thuy%’ giúp lọc các bản ghi mà cột **TenGV** có giá trị chứa ‘Thu Thuy’

+ Sau khi tìm được các bản ghi thỏa điều kiện trên, ta sẽ group by theo **MaHP** để tính số lượng lớp và sĩ số theo hai hàm góm nhóm là **count(MaL)** và **sum(SiSo)**

e. Tạo chỉ mục trên cột khóa và thực hiện yêu cầu tìm tên sinh viên đăng kí môn khác với khoa quản lí.

| */\**  *drop index idx\_sinhvien\_MaKhoa*  *drop index idx\_hocphan\_MaKhoa*  *\*/*  *create index idx\_sinhvien\_MaKhoa on SinhVien(MaKhoa);*  *create index idx\_hocphan\_MaKhoa on HocPhan(MaKhoa);*  *select distinct sv.MaSV, sv.TenSV*  *from SinhVien sv*  *join DKHP dkhp on sv.MaSV = dkhp.MaSV*  *join HocPhan hp on dkhp.MaHP = hp.MaHP*  *where sv.MaKhoa <> hp.MaKhoa;* |
| --- |

Kết quả:





Giải thích:

* Truy vấn thực hiện trên ba quan hệ:
* **SinhVien(MaSV, TenSV, MaKhoa, ...):** chứa thông tin sinh viên, bao gồm mã khoa của sinh viên.
* **DKHP(MaSV, MaHP, MaL, ...):** lưu thông tin đăng ký học phần của sinh viên, kết nối giữa sinh viên và học phần.
* **HocPhan(MaHP, TenHP, MaKhoa, ...):** chứa thông tin học phần, bao gồm mã khoa quản lý học phần đó.
* Các bước thực hiện:

1. Tạo chỉ mục (index) để tối ưu truy vấn theo **MaKhoa** trên bảng **SinhVien** và **HocPhan**.
2. Kết nối **SinhVien** với **DKHP** qua **MaSV**, rồi tiếp tục nối với **HocPhan** qua **MaHP**.
3. Lọc các bản ghi có **MaKhoa** của sinh viên khác với **MaKhoa** của học phần.
4. Dùng **DISTINCT** để loại bỏ trùng lặp sinh viên trong kết quả.

### Kết quả trả về: Danh sách sinh viên (gồm MSSV và Họ tên) đã đăng ký ít nhất một học phần không thuộc khoa quản lý của sinh viên đó.

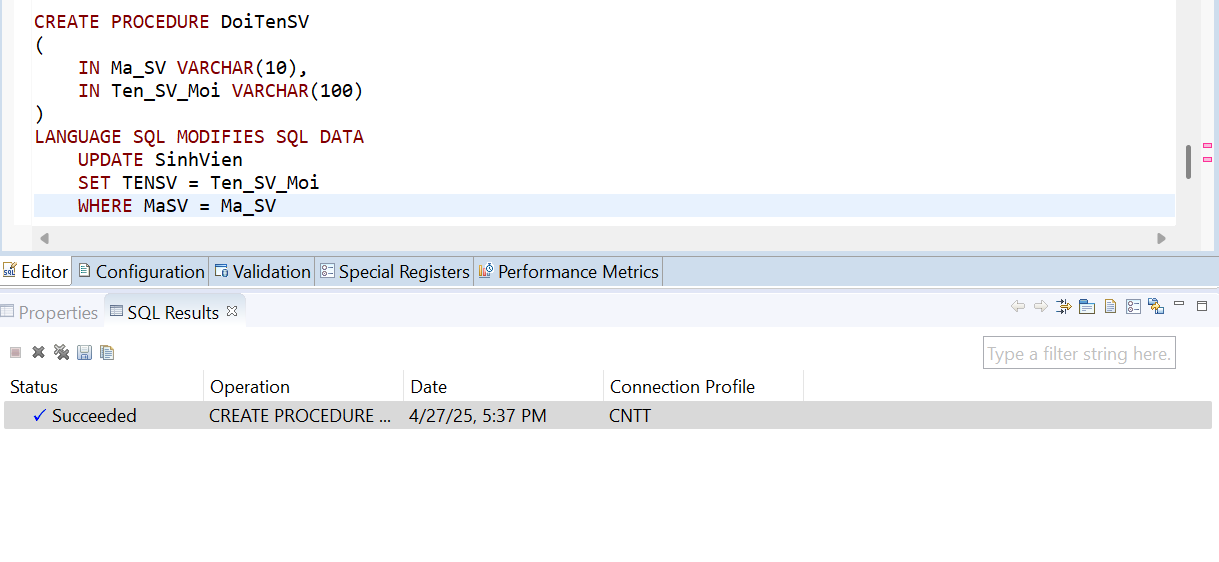
**2. Truy vấn nâng cao:**

* So với SQL Server, Db2 hỗ trợ việc thay đổi ký tự kết thúc câu lệnh (set terminator) cho phép người dùng tùy chỉnh ký tự kết thúc khi viết các block SQL phức tạp như thủ tục hoặc trigger, giúp phân biệt rõ ràng giữa các câu lệnh con. Trong Db2, mặc định terminator là dấu chấm phẩy (;), nhưng có thể thay đổi bằng câu lệnh SET TERMINATOR. Trong khi đó, SQL Server không sử dụng cơ chế này mà thay vào đó dùng từ khóa GO để phân tách các batch lệnh trong môi trường như SQLCMD hoặc SSMS.

1. **Stored procedured với tham số vào:**

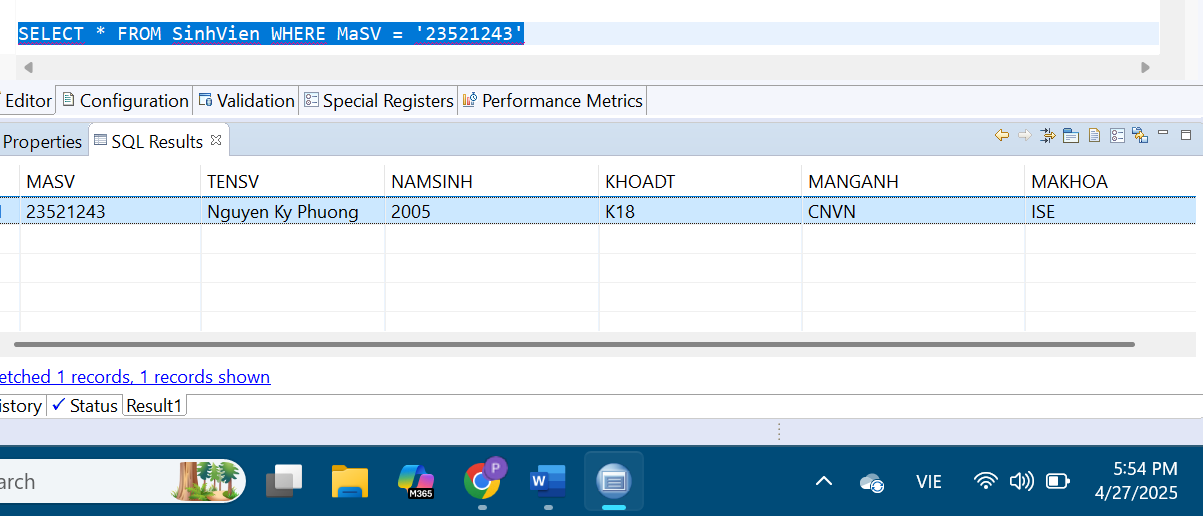
Yêu cầu: Đổi tên sinh viên với đầu vào là mã sinh viên.

| *--#SET TERMINATOR @*  *create procedure DoiTenSV (*  *in Ma\_SV varchar(10),*  *in Ten\_SV\_Moi varchar(100)*  *)*  *language sql modifies sql data*  *begin*  *update SinhVien*  *set TENSV = Ten\_SV\_Moi*  *where MaSV = Ma\_SV;*  *end*  *@* |
| --- |

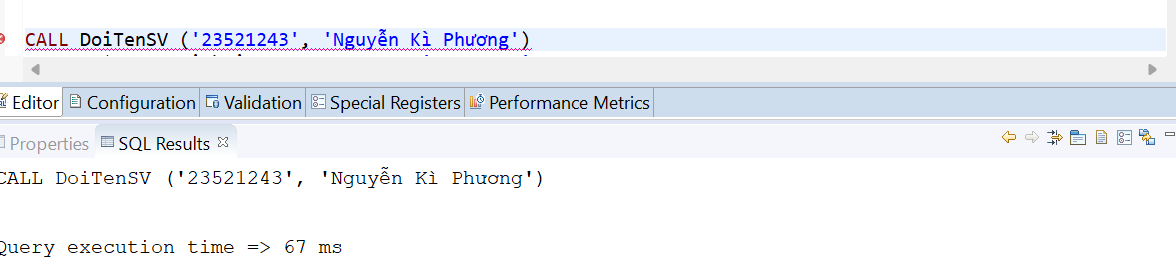
****

Kết quả:

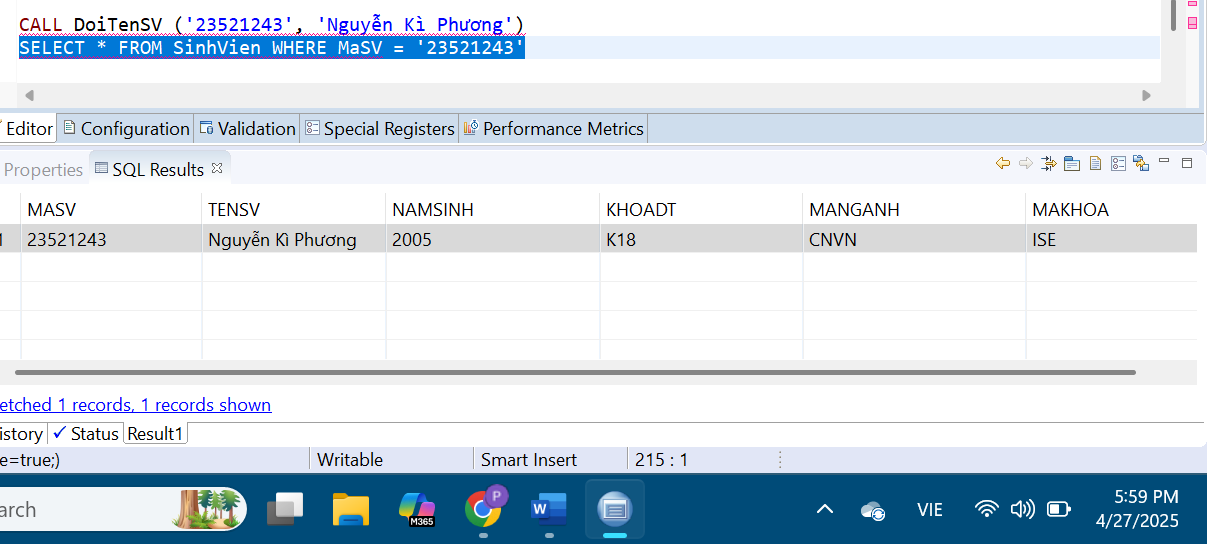
* Thực hiện truy vấn với MaSV = 235212423
* Ban đầu:



* Sau khi gọi procedure:



* Kiểm tra kết quả:



* Sau khi gọi procedure thì TenSV ban đầu là Nguyễn Kỳ Phương đã thay đổi thành TenSV lúc đưa tham số vào khi gọi procedure là Nguyễn Kì Phương

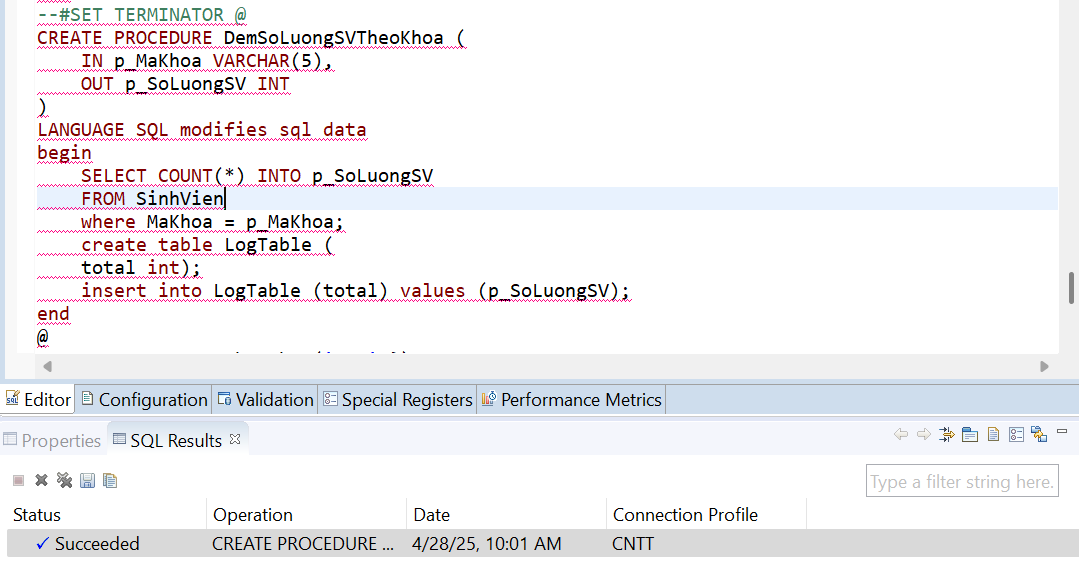
Giải thích:

* **CREATE PROCEDURE DoiTenSV:** Tạo một thủ tục (procedure) mới tên là **DoiTenSV** trong hệ quản trị cơ sở dữ liệu (ở đây là IBM DB2 hoặc một hệ SQL tương tự).
* (...): Khai báo các tham số đầu vào (parameters) cho thủ tục.
* **IN Ma\_SV VARCHAR(10):** Tham số đầu vào tên **Ma\_SV** (kiểu VARCHAR(10)) dùng để nhận mã sinh viên cần đổi tên. **IN** nghĩa là chỉ nhận vào (input), không trả dữ liệu ngược ra ngoài.
* **IN Ten\_SV\_Moi VARCHAR(100):** Tham số đầu vào tên **Ten\_SV\_Moi** (kiểu VARCHAR(100)) dùng để nhận tên mới sẽ cập nhật cho sinh viên.
* **LANGUAGE SQL:** Thông báo cho hệ thống rằng phần thân thủ tục (**body**) sẽ được viết bằng ngôn ngữ SQL chuẩn.
* **MODIFIES SQL DATA:** Báo rằng thủ tục nàycó thao tác thay đổi dữ liệu trong database (ví dụ: INSERT, UPDATE, DELETE...). Nếu không có MODIFIES SQL DATA, thủ tục chỉ được phép đọc dữ liệu (READS SQL DATA) hoặc không làm gì với dữ liệu (NO SQL).
* **UPDATE SinhVien**: Câu lệnh cập nhật bảng SinhVien.
* **SET TENSV = Ten\_SV\_Moi:** Gán tên mới (Ten\_SV\_Moi) vào cột TENSV (tức là cột tên sinh viên).
* **WHERE MaSV = Ma\_SV:** Chỉ cập nhật cho dòng dữ liệu có MaSV (mã sinh viên) đúng bằng giá trị Ma\_SV mà thủ tục nhận vào.

1. **Stored procedured với tham số vào và ra:**

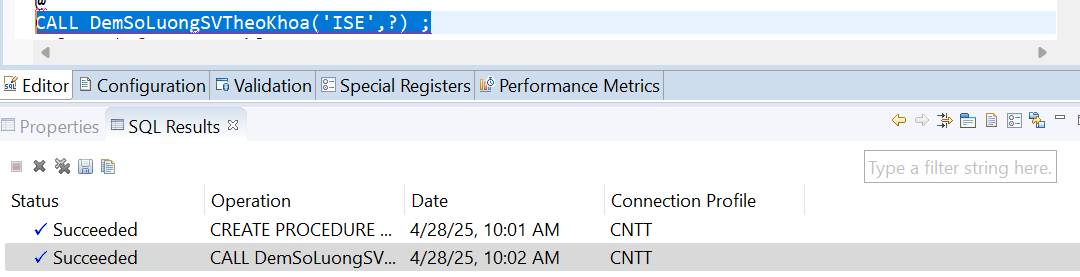
Yêu cầu: Đổi tên sinh viên với đầu vào là mã sinh viên.

| *--#SET TERMINATOR @*  *create procedure DemSoLuongSVTheoKhoa (*  *in p\_MaKhoa varchar(5),*  *out p\_SoLuongSV int*  *)*  *language sql modifies sql data*  *begin*  *select count(\*) into p\_SoLuongSV*  *from SinhVien*  *where MaKhoa = p\_MaKhoa;*  *create table LogTable (*  *total int);*  *insert into LogTable (total) values (p\_SoLuongSV);*  *end*  *@* |
| --- |

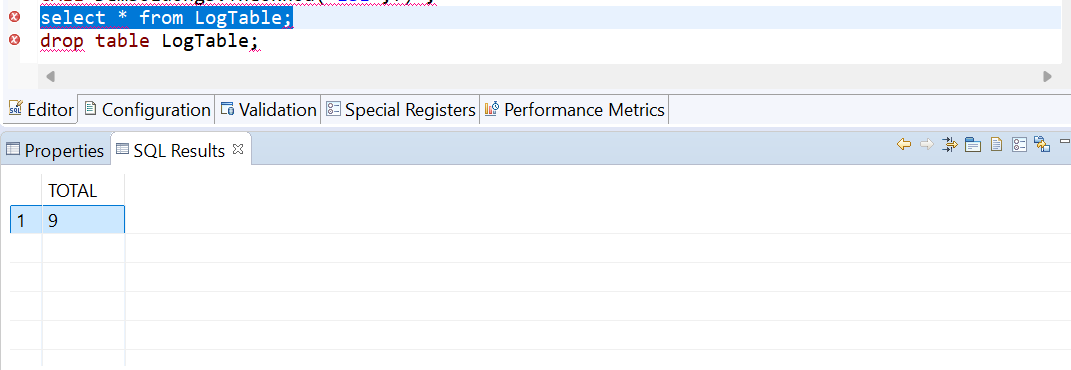
****

Kết quả:

* Thực hiện PROCEDURE



* Kiểm tra kết quả:



* Sau khi gọi procedure thì tham số đầu vào ‘ISE’ sẽ được đưa vào để tính tổng SinhVien thuộc khoa ‘ISE’ và đưa vào bảng LogTable và vì ***PROCEDURE không cho phép thực hiện SELECT*** nên phải truy vấn bảng LogTable bên ngoài để xem kết quả và sau khi xem ***phải xóa bảng để sử dụng lần sau*.**

Giải thích:

### **--#SET TERMINATOR @**

* Đặt dấu kết thúc câu lệnh (terminator) là @ thay vì ;.
* Vì trong thân CREATE PROCEDURE có rất nhiều dấu ;, nếu không thay đổi thì Db2 sẽ hiểu nhầm và báo lỗi.

### **CREATE PROCEDURE DemSoLuongSVTheoKhoa ( IN p\_MaKhoa VARCHAR(5), OUT p\_SoLuongSV INT )**

- Tạo thủ tục mới tên là: **DemSoLuongSVTheoKhoa.**

* Có 2 tham số:
* **p\_MaKhoa:** tham số đầu vào (IN), kiểu chuỗi VARCHAR(5), là mã khoa cần đếm.
* **p\_SoLuongSV**: tham số đầu ra (OUT), kiểu số nguyên INT, để trả về số lượng sinh viên.

### **LANGUAGE SQL**

### Chỉ định ngôn ngữ của thủ tục là **SQL** thuần (không phải PL/SQL hoặc ngôn ngữ khác).

### **modifies sql data**

* Thông báo cho hệ thống rằng thủ tục này sẽ thay đổi dữ liệu trong database (vì có INSERT vào bảng LogTable).

### **begin**

* Bắt đầu thân của thủ tục.

### **SELECT COUNT(\*) INTO p\_SoLuongSV FROM SinhVien WHERE MaKhoa = p\_MaKhoa;**

* Đếm số sinh viên trong bảng **SinhVien** mà có **MaKhoa** bằng **p\_MaKhoa**.
* Kết quả (số lượng) được lưu vào biến đầu ra **p\_SoLuongSV**.

### **create table LogTable (total int);**

* Tạo bảng mới tên là LogTable, có một cột total kiểu int.
* Mục đích: để lưu tạm số lượng sinh viên đếm được (giống kiểu log ghi nhận kết quả).

### **insert into LogTable (total) values (p\_SoLuongSV);**

* Chèn một dòng mới vào **LogTable**, trong đó giá trị total chính là số lượng sinh viên vừa đếm được.

### **end @**

* Kết thúc thủ tục.
* Nhớ phải kết thúc bằng @ vì trước đó đã set terminator @.

### **CALL DemSoLuongSVTheoKhoa('ISE', ?) ;**

* Gọi thực thi thủ tục **DemSoLuongSVTheoKhoa**.
* Truyền vào:
* 'ISE': mã khoa cần đếm sinh viên.
* ?: chỗ để nhận kết quả đầu ra **p\_SoLuongSV**.

### **select \* from LogTable;**

* Xem nội dung bảng **LogTable**.
* Sẽ hiện ra số lượng sinh viên có MaKhoa = 'ISE'.

### **drop table LogTable;**

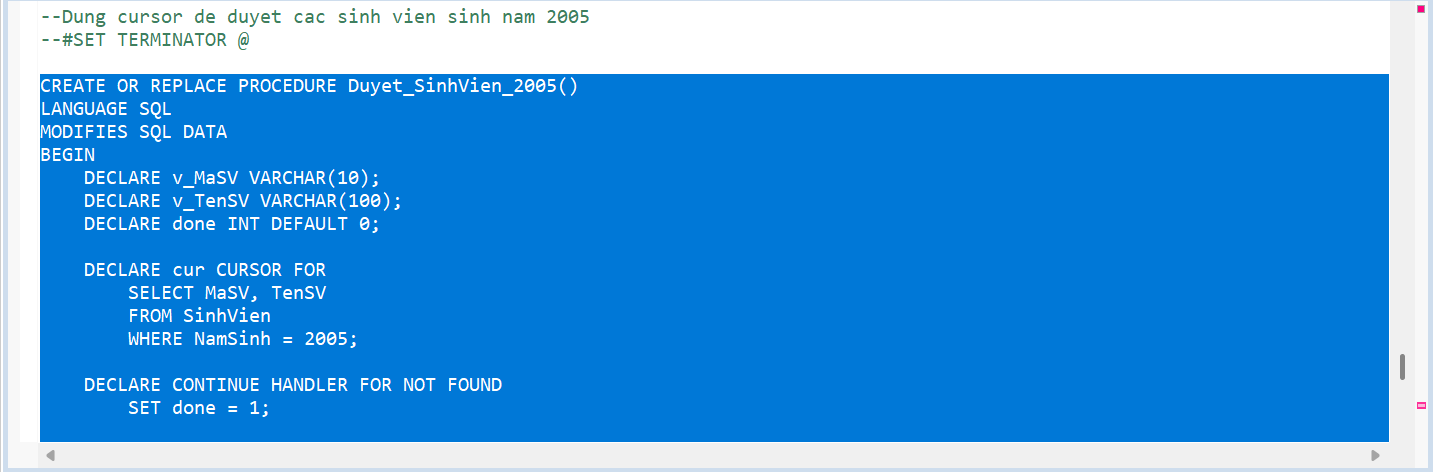
* Xóa bảng LogTable sau khi đã dùng xong (tránh để thừa dữ liệu log).

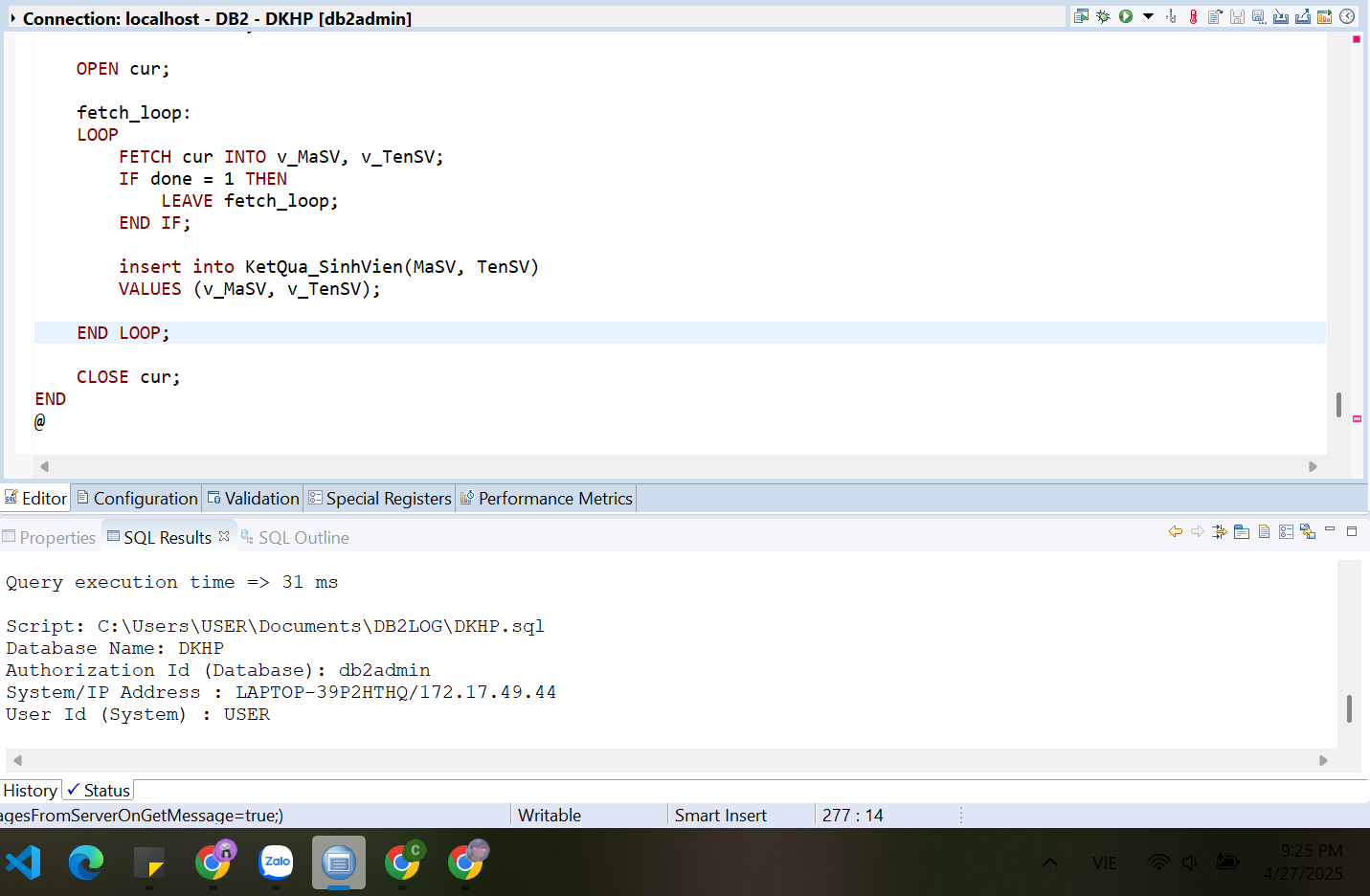
1. **Cursor:**

Yêu cầu: uyệt sinh viên sinh năm 2005

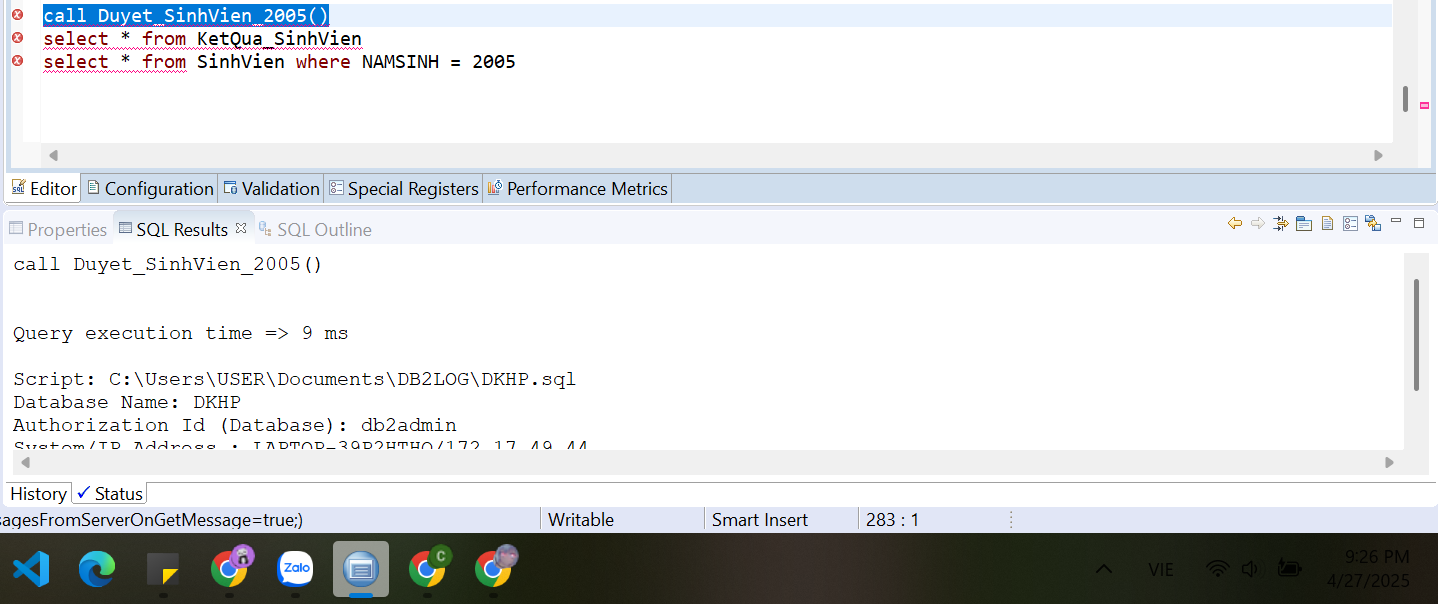
| *–Tao bang KetQua\_SinhVien*  *DROP TABLE KetQua\_SinhVien;*  *CREATE TABLE KetQua\_SinhVien (*  *MaSV VARCHAR(10),*  *TenSV VARCHAR(100)*  *);*  *--Dung cursor de duyet cac sinh vien sinh nam 2005*  *--#SET TERMINATOR @*  *CREATE OR REPLACE PROCEDURE Duyet\_SinhVien\_2005()*  *LANGUAGE SQL*  *MODIFIES SQL DATA*  *BEGIN*  *DECLARE v\_MaSV VARCHAR(10);*  *DECLARE v\_TenSV VARCHAR(100);*  *DECLARE done INT DEFAULT 0;*  *DECLARE cur CURSOR FOR*  *SELECT MaSV, TenSV*  *FROM SinhVien*  *WHERE NamSinh = 2005;*  *DECLARE CONTINUE HANDLER FOR NOT FOUND*  *SET done = 1;*  *OPEN cur;*  *fetch\_loop:*  *LOOP*  *FETCH cur INTO v\_MaSV, v\_TenSV;*  *IF done = 1 THEN*  *LEAVE fetch\_loop;*  *END IF;*    *insert into KetQua\_SinhVien (MaSV, TenSV)*  *VALUES (v\_MaSV, v\_TenSV);*    *END LOOP;*  *CLOSE cur;*  *END*  *@*  *call Duyet\_SinhVien\_2005()*  *select \* from KetQua\_SinhVien* |
| --- |

Kết quả:

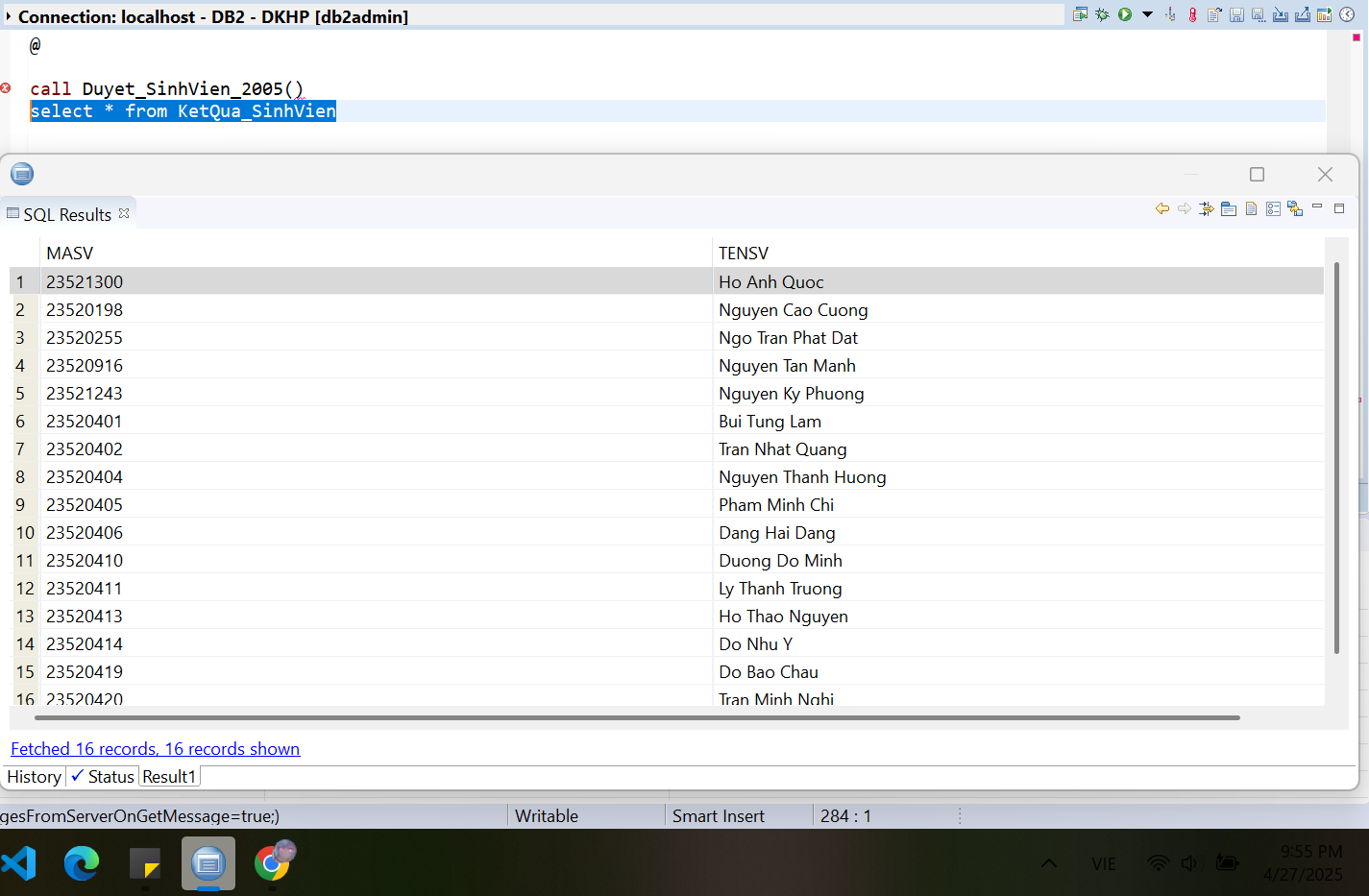




Gọi hàm duyệt sinh viên:



Thực hiện truy vấn và kết quả:



Giải thích:

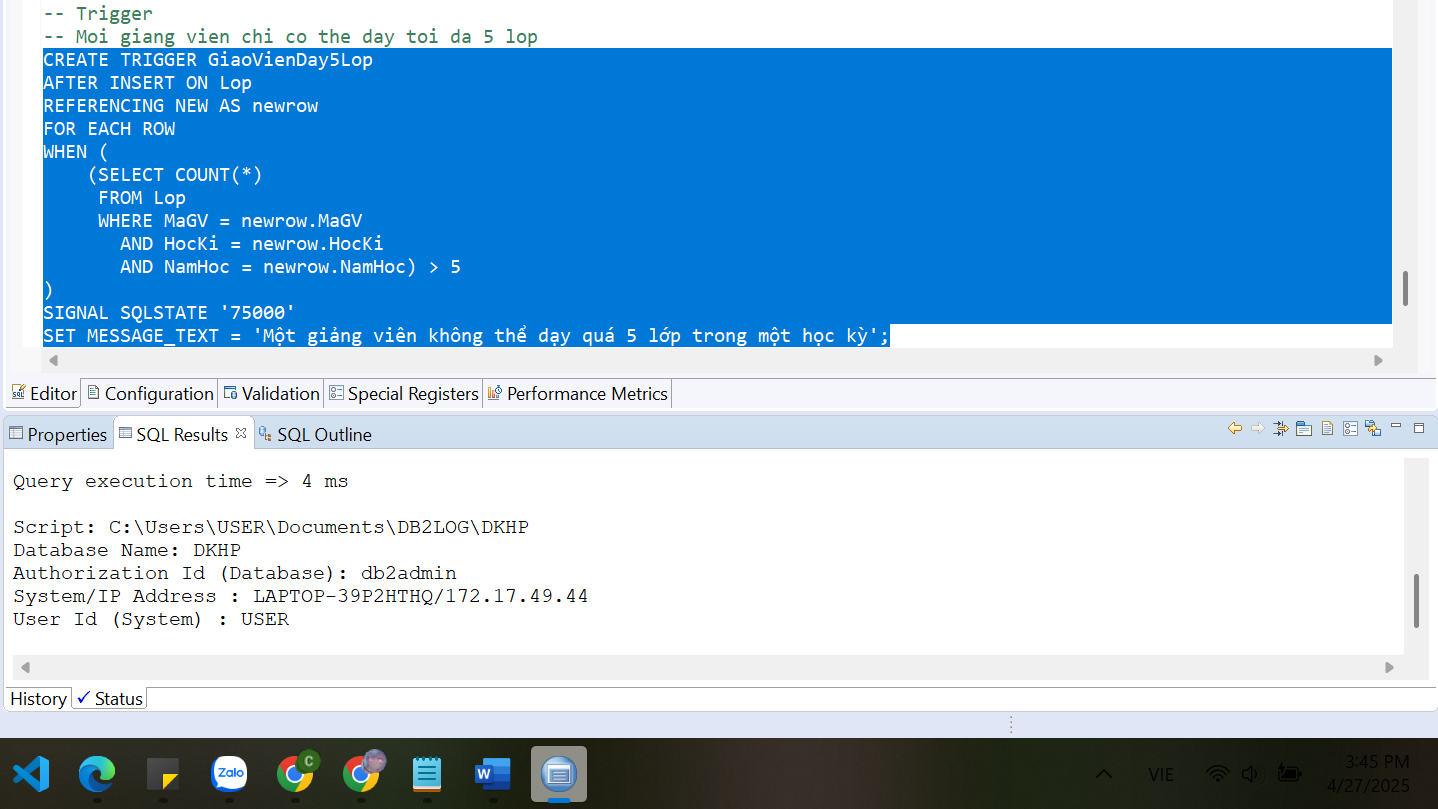
* Thủ tục **Duyet\_SinhVien\_2005** dùng để quét hết sinh viên sinh năm 2005 trong bảng SinhVien, rồi chép từng bạn đó vào bảng **KetQua\_SinhVien**.
* Đầu tiên, cần khai báo biến để lưu mã và tên sinh viên.
* Tạo cursor để chọn danh sách sinh viên sinh năm 2005.
* Mở cursor và duyệt từng dòng.
* Lấy 1 sinh viên
* Nếu còn dữ liệu → insert vào **KetQua\_SinhVien**
* Nếu hết → thoát vòng lặp
* Đóng cursor sau khi xong.

1. **Trigger:**

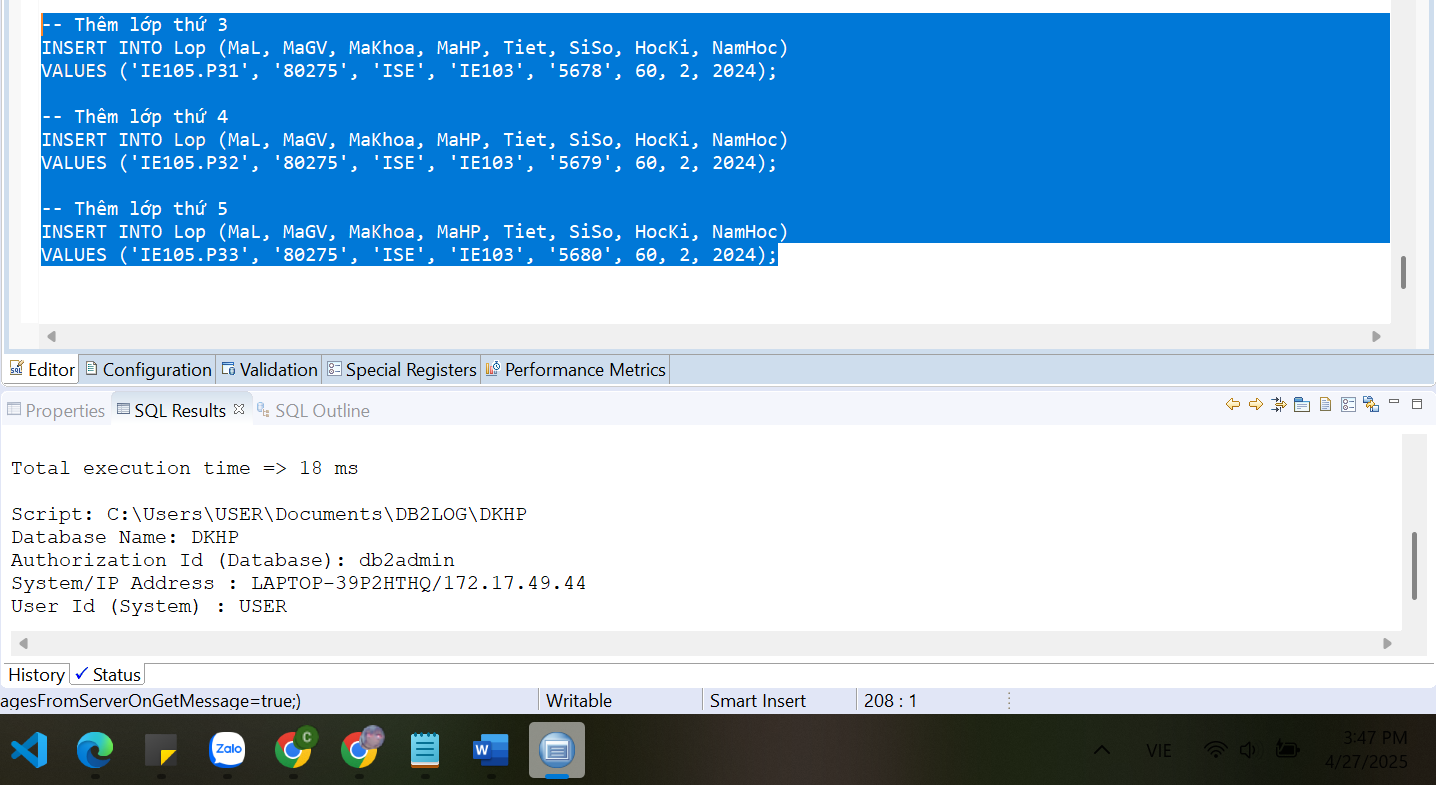
Yêu cầu: Mỗi giảng viên chỉ được dạy tối đa 5 lớp mỗi học kỳ

| *CREATE TRIGGER GiaoVienDay5Lop*  *AFTER INSERT ON Lop*  *REFERENCING NEW AS newrow*  *FOR EACH ROW*  *WHEN (*  *(SELECT COUNT(\*)*  *FROM Lop*  *WHERE MaGV = newrow.MaGV*  *AND HocKi = newrow.HocKi*  *AND NamHoc = newrow.NamHoc) > 5*  *)*  *SIGNAL SQLSTATE '75000'*  *SET MESSAGE\_TEXT = 'Mot giang vien khong the day qua 5 lop trong 1 hoc ky';*  *call Duyet\_SinhVien\_2005()*  *select \* from KetQua\_SinhVien* |
| --- |

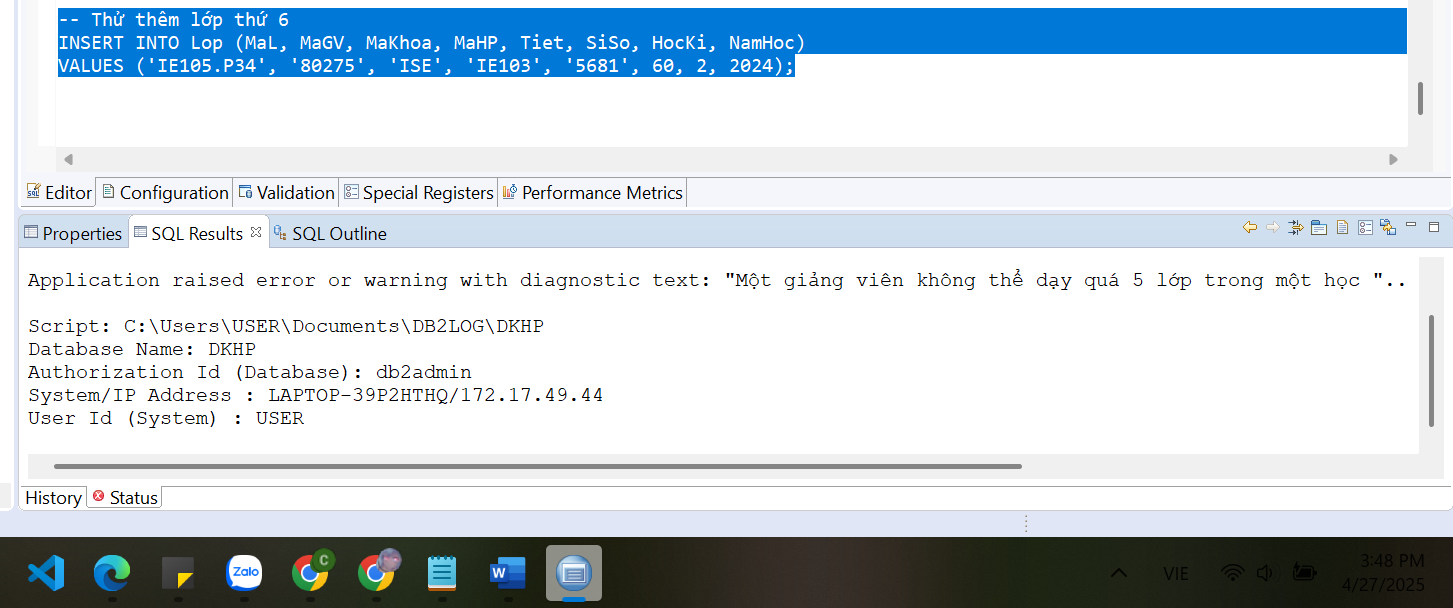
Kết quả:



Khi thêm 3 lớp cho giáo viên mã 80275, thành công.



Khi thêm lớp thứ 6, thông báo lỗi:



Giải thích:

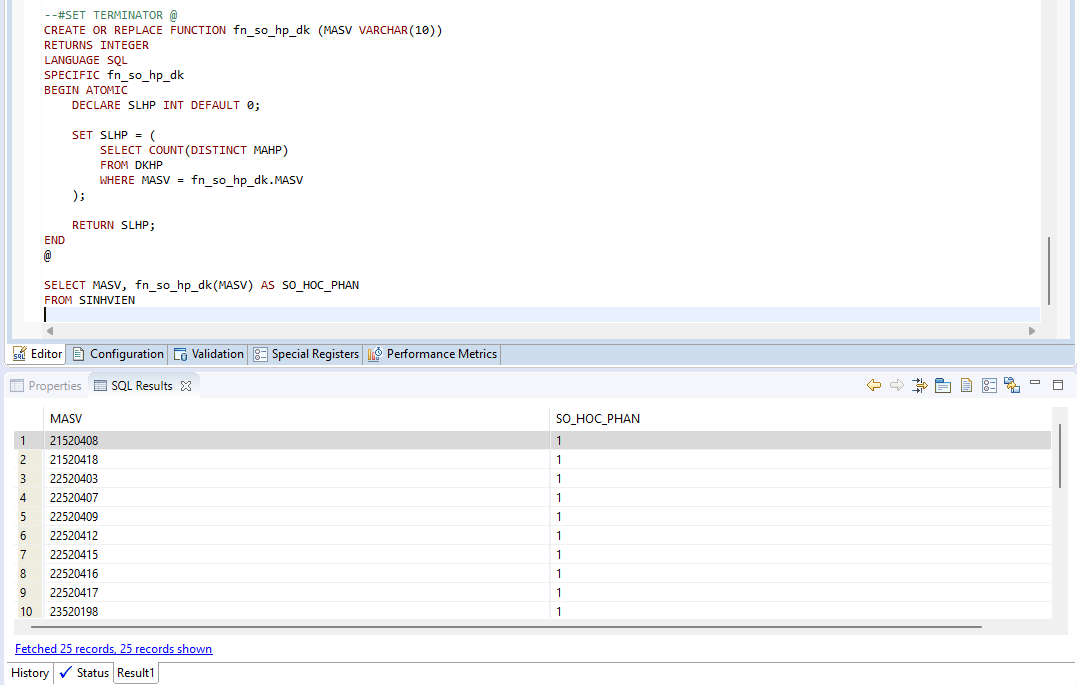
* Khi insert vào bảng **Lop**, trigger sẽ kiểm tra ngay lập tức: Đếm số lớp mà giảng viên (**MaGV**) đang dạy cùng **HocKi**, **NamHoc**.
* Nếu số lớp > 5:
* Ngay lập tức trigger bật → signal lỗi (SQLSTATE = ‘75000’)
* Rollback giao dịch, không cho thêm lớp.
* Nếu số lớp <=5: Cho phép insert thành công.

1. **Function:**

Yêu cầu:

| *--#SET TERMINATOR @*  *CREATE OR REPLACE FUNCTION fn\_so\_hp\_dk (MASV VARCHAR(10))*  *RETURNS INTEGER*  *LANGUAGE SQL*  *SPECIFIC fn\_so\_hp\_dk*  *BEGIN ATOMIC*  *DECLARE SLHP INT DEFAULT 0;*  *SET SLHP = (*  *SELECT COUNT(DISTINCT MAHP)*  *FROM DKHP*  *WHERE MASV = fn\_so\_hp\_dk.MASV*  *);*  *RETURN SLHP;*  *END*  *@* |
| --- |

Kết quả:



Giải thích:

### **--#SET TERMINATOR @**

- Thay dấu kết thúc lệnh SQL mặc định “ ; ” bằng dấu “ @ ”.

- Trong một function/stored procedure có nhiều dấu “ ; “ nội bộ, nên cần chọn một ký hiệu khác để đánh dấu kết thúc toàn bộ function.

- Ví dụ:

* Trong function: **DECLARE, SET, RETURN** đều kết thúc bằng “ ; ”
* Nhưng kết thúc function thì cần “ @ ”.

### **RETURNS INTEGER**

* Function này sẽ trả về một giá trị kiểu **INTEGER** (số nguyên).

### **LANGUAGE SQL**

* Function này được viết bằng **SQL** thuần (không phải bằng Java, C, hay các ngôn ngữ khác).

### **DECLARE SLHP INT DEFAULT 0;**

* Khai báo biến nội bộ **SLHP**, kiểu **INT**.
* Giá trị khởi tạo mặc định ban đầu là **0**.

### **SET SLHP = ( SELECT COUNT(DISTINCT MAHP) FROM DKHP WHERE MASV = fn\_so\_hp\_dk.MASV );**

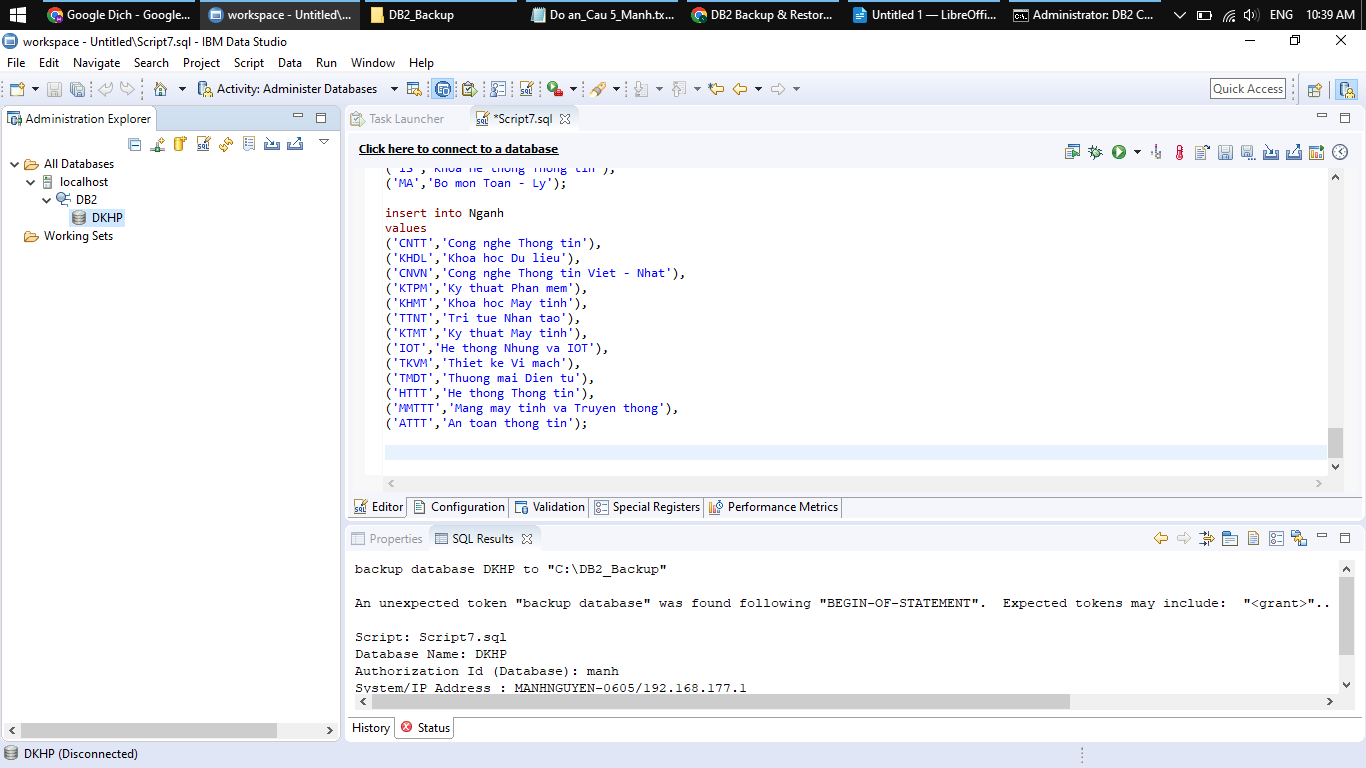
* Gán cho biến **SLHP** giá trị là:
* Số lượng (**COUNT**) các môn học khác nhau (**DISTINCT MAHP**).
* Mà sinh viên có mã số MASV đã đăng ký (DKHP là bảng đăng ký học phần).
* Điểm quan trọng: **fn\_so\_hp\_dk.MASV**: Tham số đầu vào của function.

1. **Backup:**

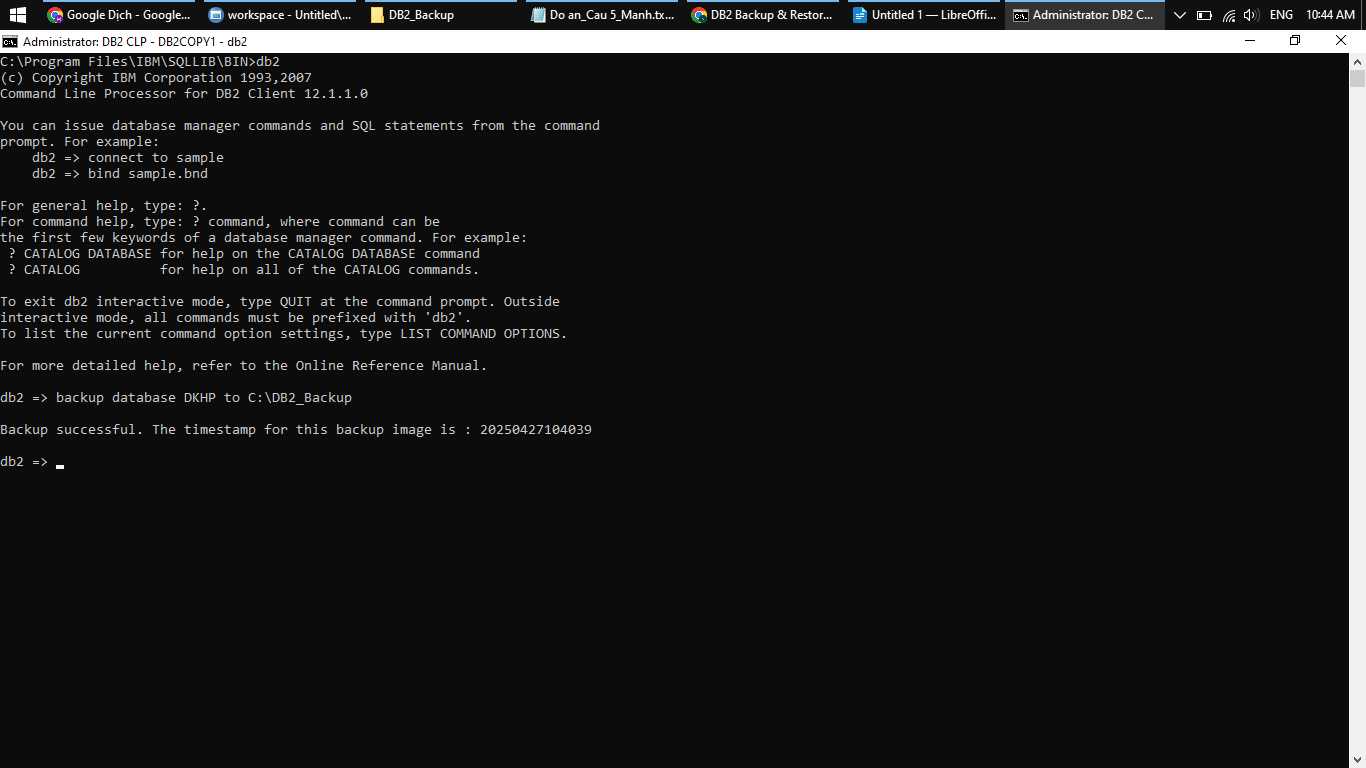
| *backup database DKHP to C:\DB2\_Backup;* |
| --- |

Kết quả:

Hình ảnh Database DKHP đã được tạo trước đó



Hình ảnh chạy câu lệnh backup database thành công



Giải thích:

- Sử dụng câu lệnh **backup database <tên database> to <địa chỉ thư mục lưu file backup>;** để backup cơ sở dữ liệu.

- DKHP là cơ sở dữ liệu Đăng kí học phần mà nhóm đã tạo trước đó.

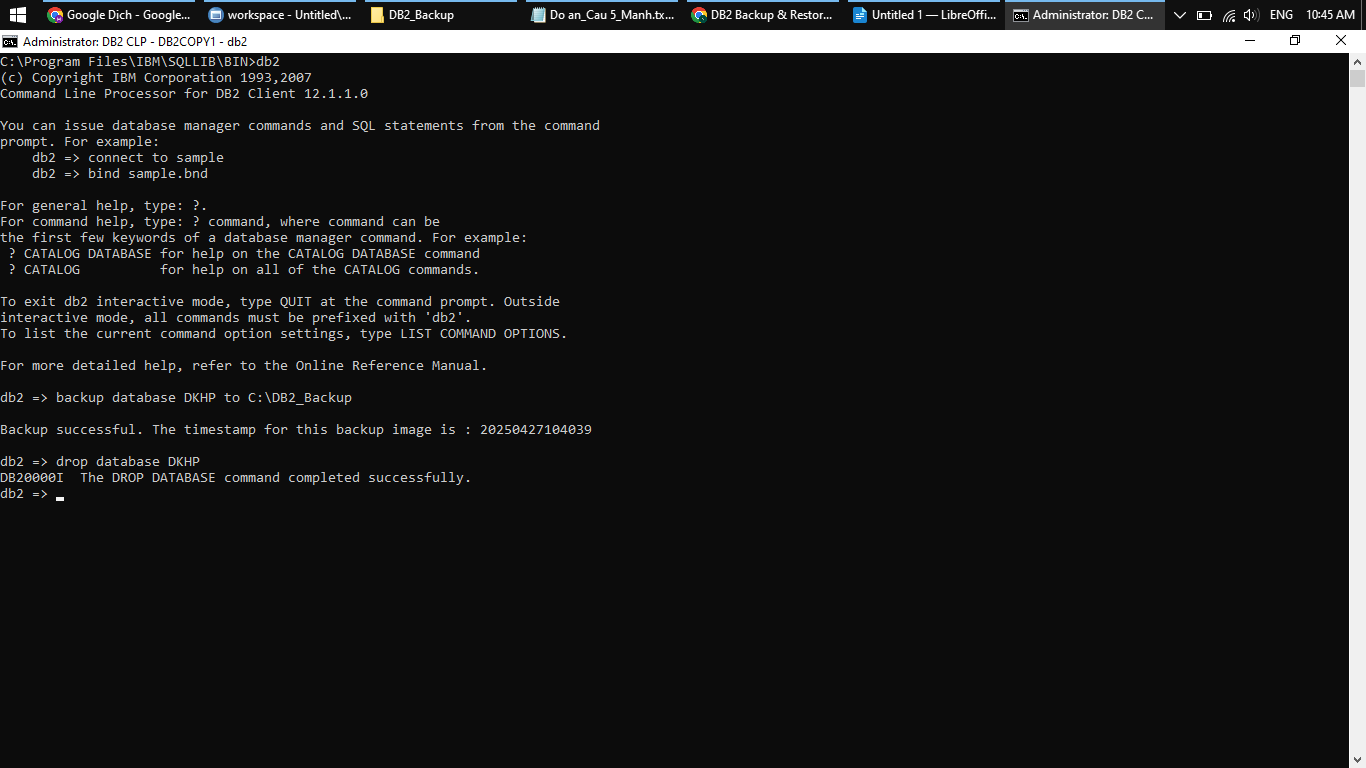
- **C:\DB2\_Backup** là địa chỉ của thư mục lưu file backup.

1. **Restore**

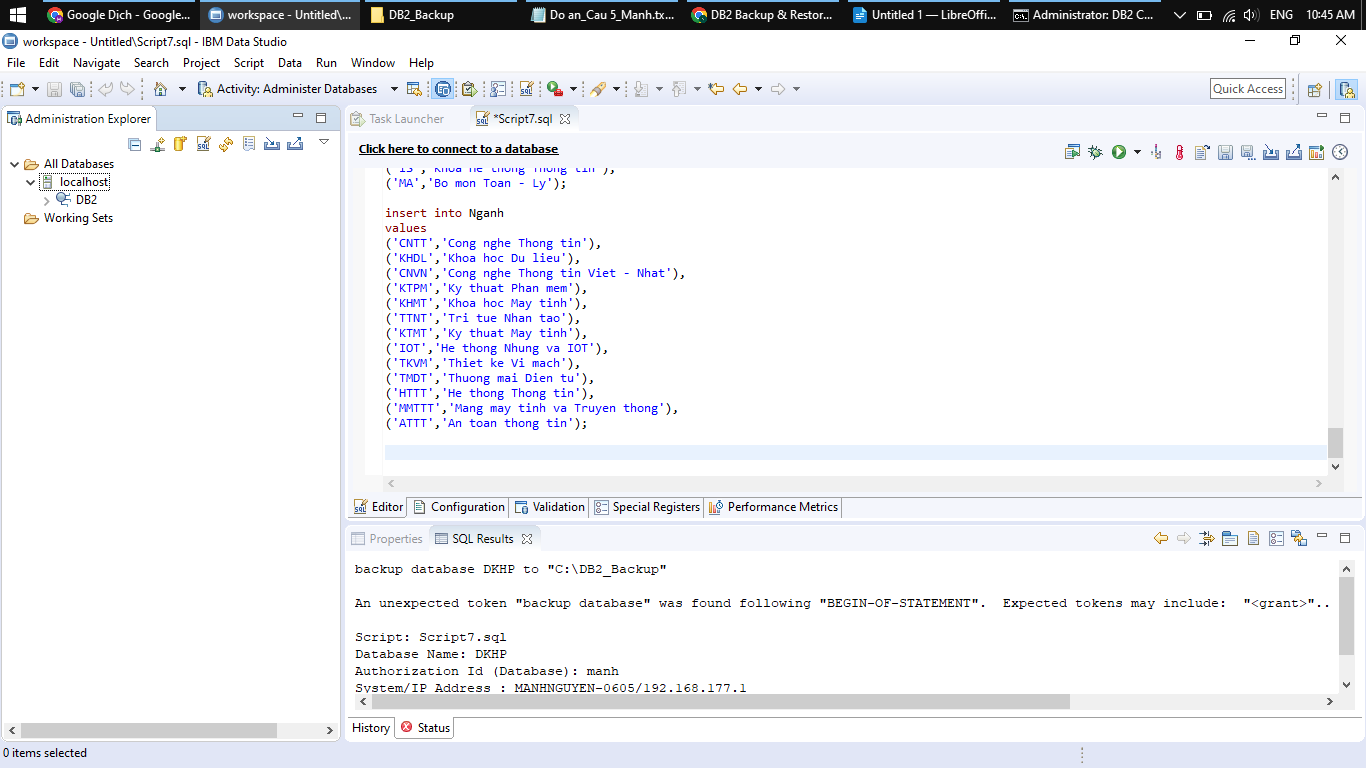
| *drop database DKHP;*  *restore database DKHP from C:\DB2\_Backup taken at 20250427104039 replace existing;*  *connect to DKHP;* |
| --- |

Kết quả:

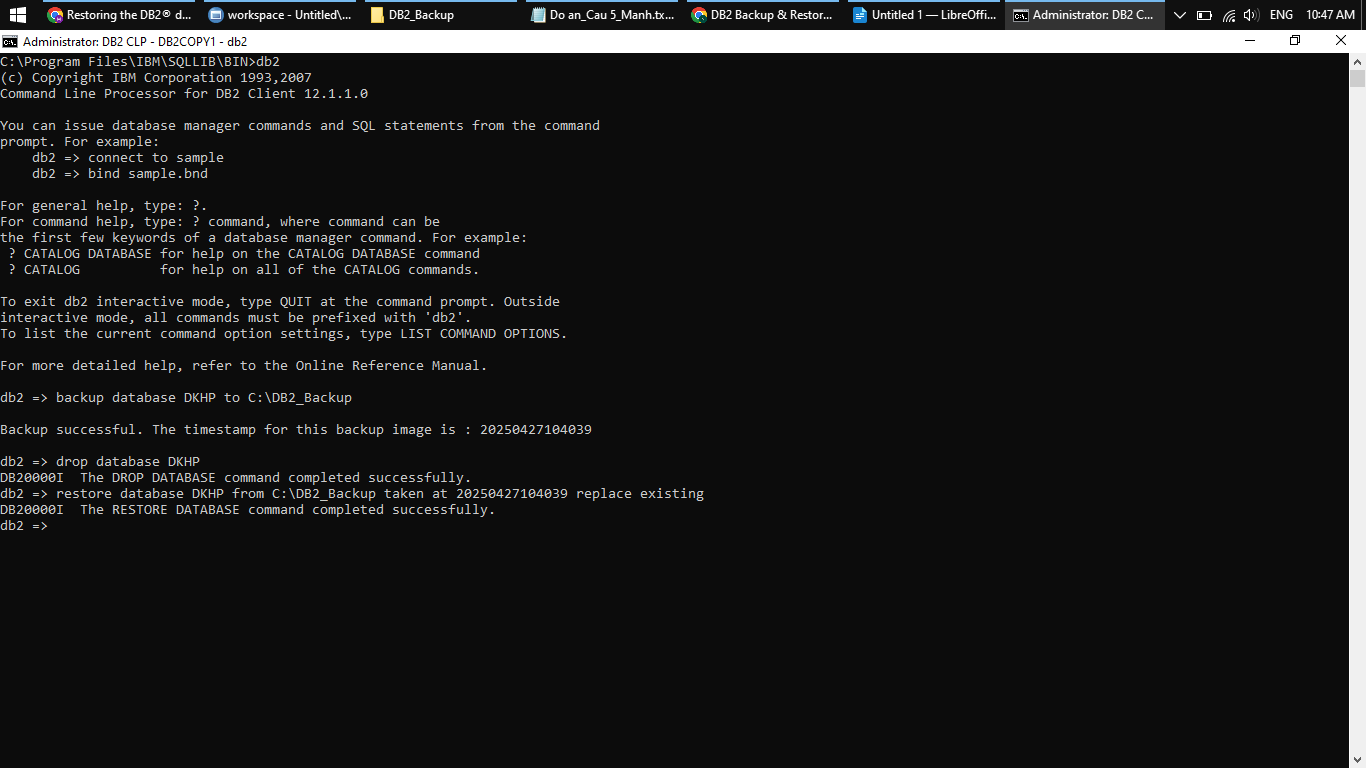
Câu lệnh drop database chạy thành công



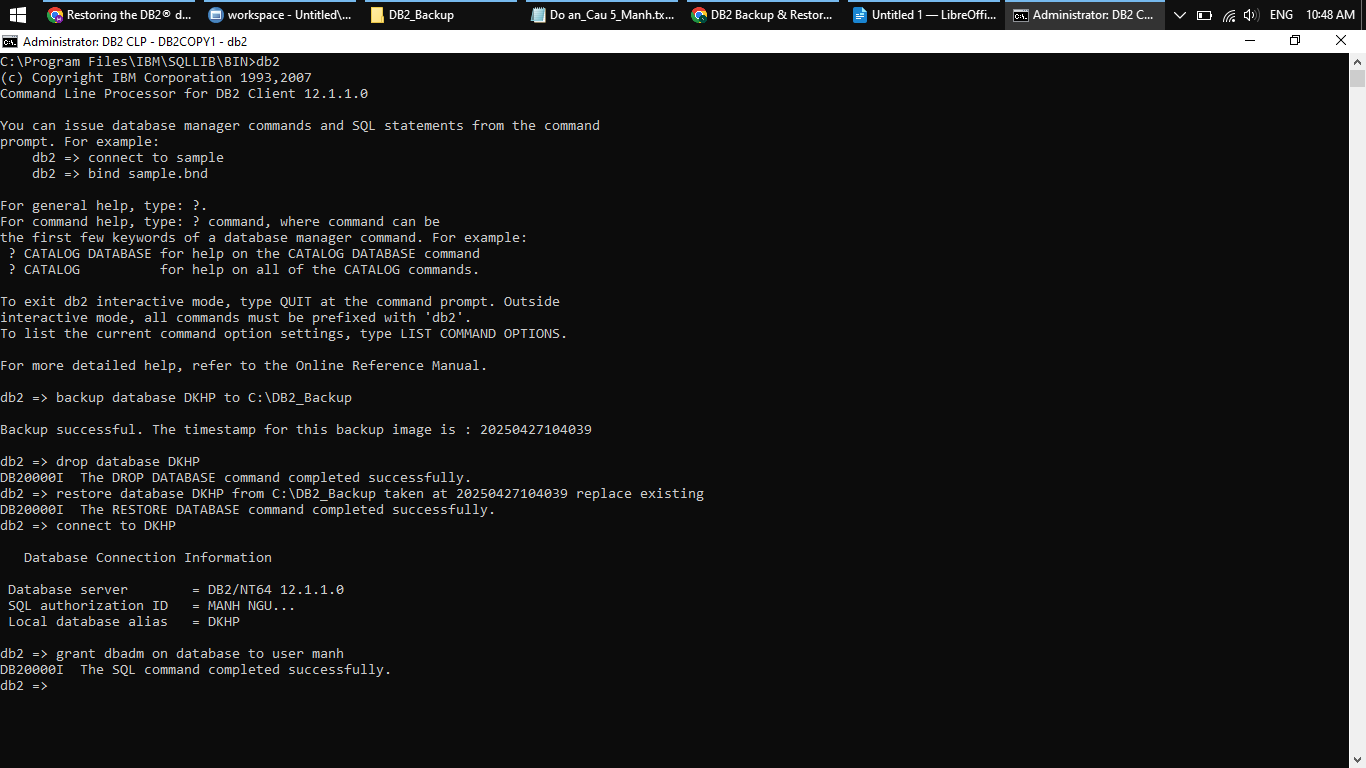
Kiểm tra trong DB2 Studio thì thấy database DKHP đã bị mất



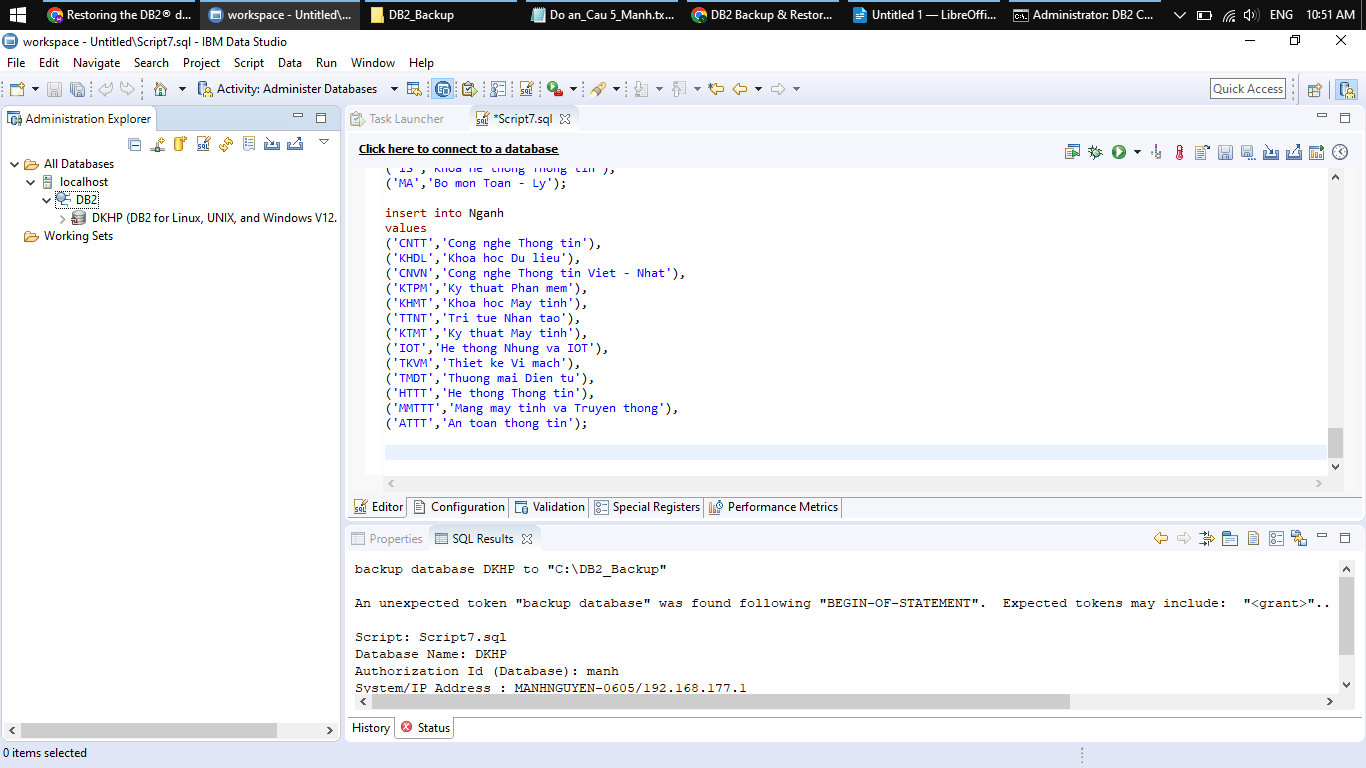
Chạy câu lệnh restore database thành công



Kiểm tra bằng lệnh connect to DKHP thì thấy database đã được khôi phục



Vào lại DB2 Data Studio thì thấy database được xuất hiện trở lại



Giải thích:

- Dùng lệnh **drop database DKHP** để xóa database DKHP, giả lập là cơ sở dữ liệu đã bị mất và cần được khôi phục

- Dùng lệnh **restore database <tên database> from <địa chỉ thư mục lưu trữ file backup> taken at <timestamp> replace existing;** để khôi phục database.

- DKHP là tên của database Đăng kí học phần

- C:\DB2\_Backup là tên thư mục lưu file backup ở phần Backup.

- 20250427104039 là timestamp của file backup, đó chính là thời gian đã backup cơ sở dữ liệu.

- **replace existing** sẽ thay thế database hiện tại bởi database mới cùng tên.

- **connect to DKHP** sẽ kiểm tra là database DKHP đã được phục hồi hay chưa.

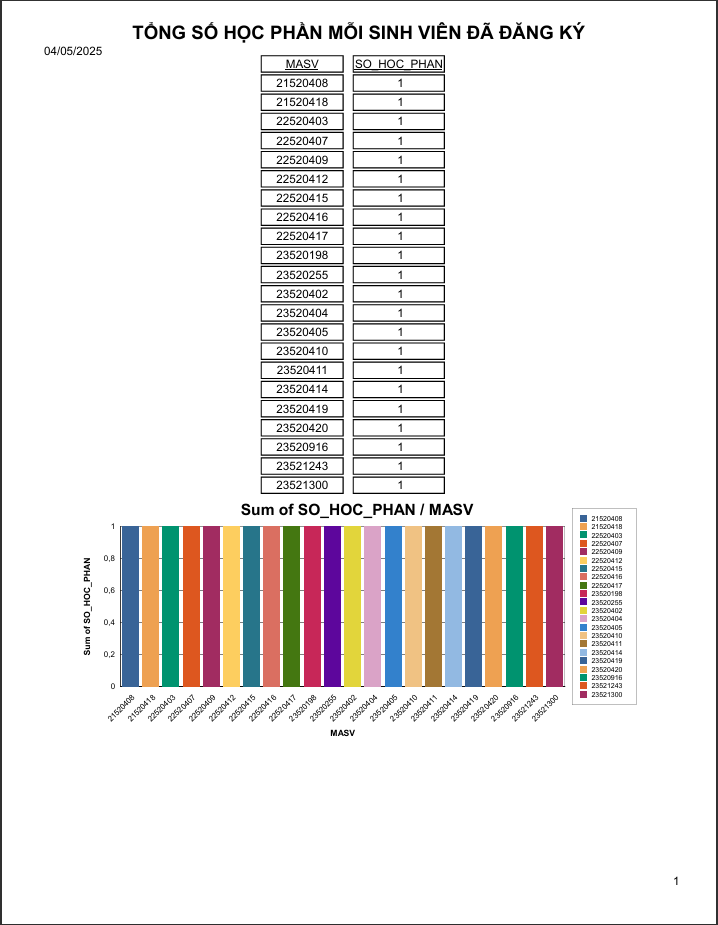
1. **Crystal Report:**

Yêu cầu: Trực quan tổng số học phần mà mỗi sinh viên đăng kí

Tạo view

| *CREATE OR REPLACE VIEW v\_so\_hp\_dk\_moi\_sv AS*  *SELECT*  *MASV,*  *COUNT(DISTINCT MAHP) AS SO\_HOC\_PHAN*  *FROM DKHP*  *WHERE TRANGTHAI = 'Da dang ky'*  *GROUP BY MASV* |
| --- |

Kết quả:



Giải thích:

* Liên kết Crystal Report với Db2 database
* Nhập View vào report
* Tạo report

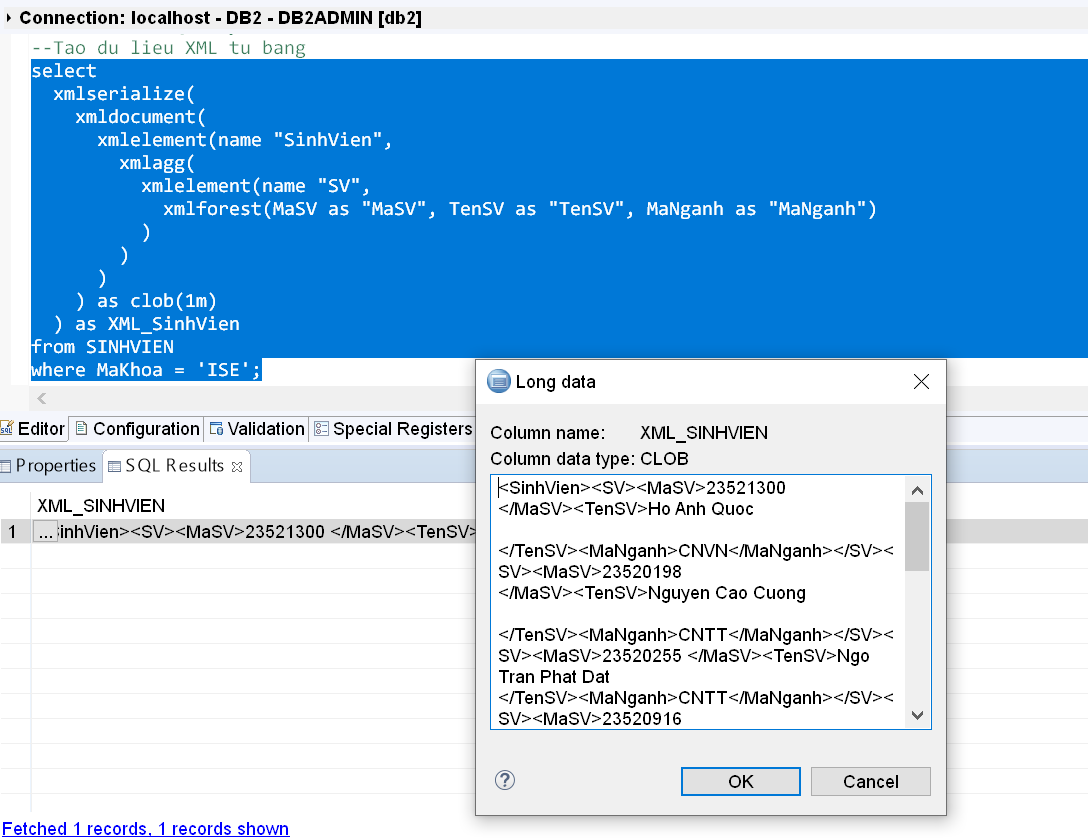
1. **XML/XPath/XQuerry:**

* So với SQL Server, Db2 hỗ trợ XML chuẩn với kiểu dữ liệu XML và tích hợp đầy đủ XPath cùng XQuery, cho phép truy vấn phức tạp và kết hợp SQL với XQuery mượt mà, tối ưu cho xử lý XML phức tạp. Trong khi đó, SQL Server cũng có kiểu dữ liệu XML và hỗ trợ XPath qua các phương thức như .query(), .value(), nhưng chỉ hỗ trợ một phần XQuery, chủ yếu sử dụng trong các hàm XML tích hợp trên cột XML, phù hợp hơn với xử lý XML đơn giản đến trung bình.

Yêu cầu: Tạo dữ liệu XML từ bảng SinhVien

| *--Tao du lieu XML tu bang*  *select*  *xmlserialize(*  *xmldocument(*  *xmlelement(name "SinhVien",*  *xmlagg(*  *xmlelement(name "SV",*  *xmlforest(MaSV as "MaSV", TenSV as "TenSV", MaNganh as "MaNganh")*  *)*  *)*  *)*  *) as clob(1m)*  *) as XML\_SinhVien*  *from SINHVIEN*  *where MaKhoa = 'ISE';* |
| --- |

Kết quả:



Giải thích:

* **xmlforest(...)**: Tạo ra nhiều phần tử con cùng lúc từ các cột.
* **xmlelement(name "SV", ...)**: Bao bọc mỗi sinh viên trong phần tử **<SV>...</SV>**.
* **xmlagg(...)**: Gom tất cả các sinh viên thành danh sách dưới phần tử **<SinhVien>**.
* **xmldocument(...)**: Tạo một XML document hoàn chỉnh.
* **xmlserialize(... as clob(1m))**: Chuyển kết quả XML thành văn bản CLOB để dễ thao tác/lưu trữ.

CLOB là viết tắt của Character Large Object – một kiểu dữ liệu trong SQL được thiết kế để lưu trữ chuỗi văn bản có kích thước rất lớn.

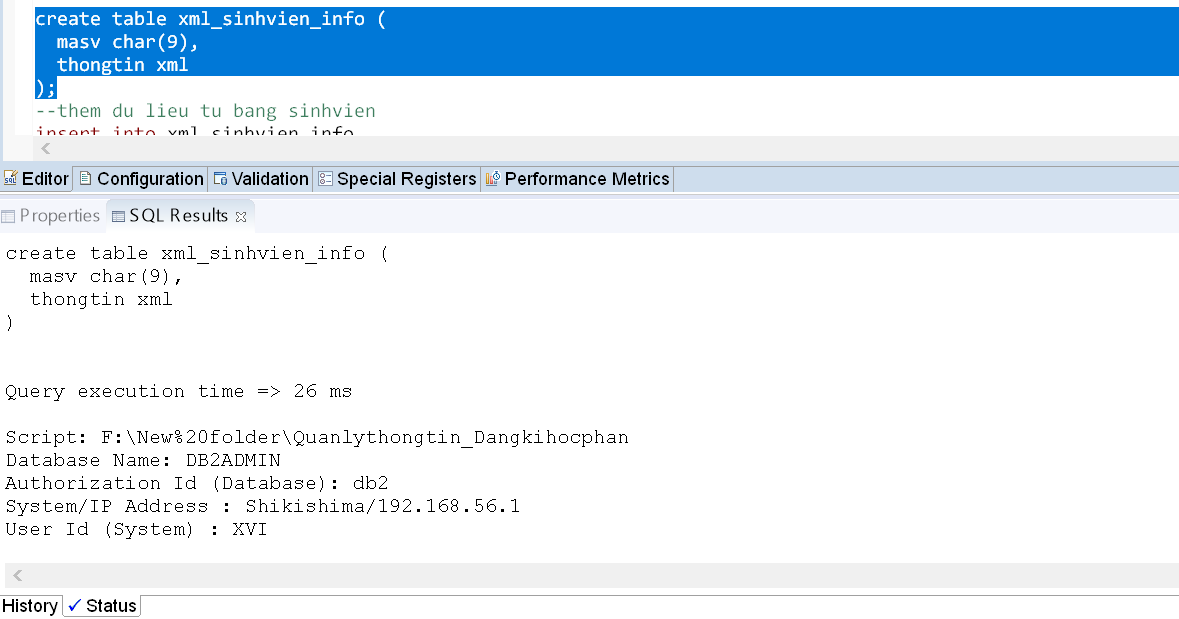
Yêu cầu:

Tạo bảng lưu XML sinh viên thuộc khoa ISE và chèn dữ liệu vào bảng, sau đó tìm sinh viên học ngành có mã ngành là CNTT, dùng XPath

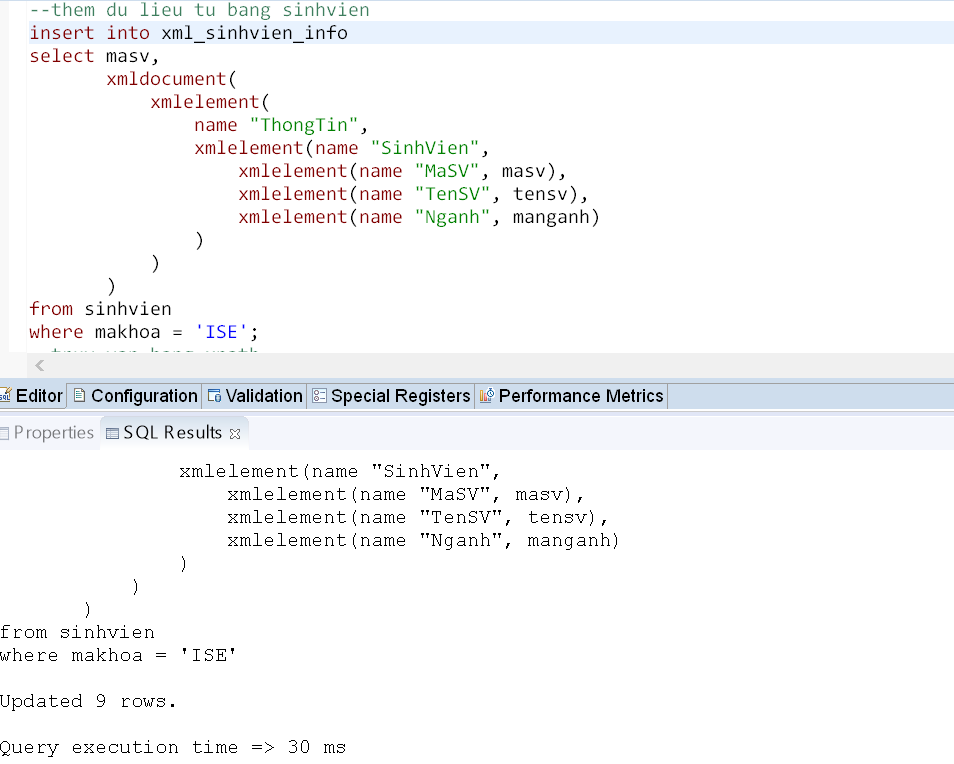
| *--XPath truy van du lieu XML*  *--Tao bang XML*  *drop table xml\_sinhvien\_info;*  *create table xml\_sinhvien\_info (*  *masv char(9),*  *thongtin xml*  *);*  *--them du lieu tu bang sinhvien*  *insert into xml\_sinhvien\_info*  *select masv*  *xmldocument(*  *xmlelement(*  *name "ThongTin",*  *xmlelement(name "SinhVien",*  *xmlelement(name "MaSV", masv),*  *xmlelement(name "TenSV", tensv),*  *xmlelement(name "Nganh", manganh)*  *)*  *)*  *)*  *from sinhvien*  *where makhoa = 'ISE';*  *--truy van bang xpath*  *select masv*  *from xml\_sinhvien\_info*  *where xmlcast(*  *xmlquery('/ThongTin/SinhVien/Nganh' passing thongtin returning sequence)*  *as varchar(20)*  *) = 'CNTT';*  *--xem ket qua*  *select*  *masv,*  *xmlserialize(thongtin as varchar(500)) as xml\_dulieu*  *from xml\_sinhvien\_info;* |
| --- |

Kết quả:

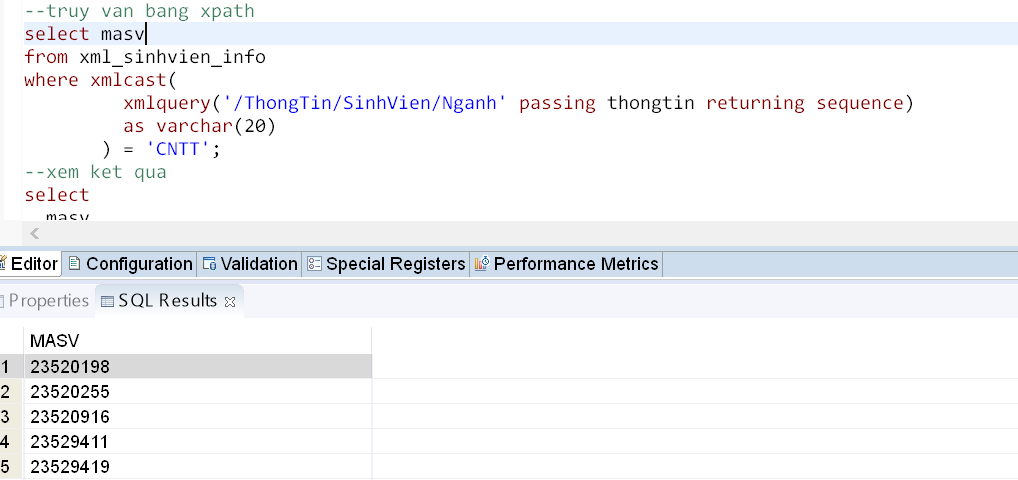
* Tạo bảng xml\_sinhvien\_info



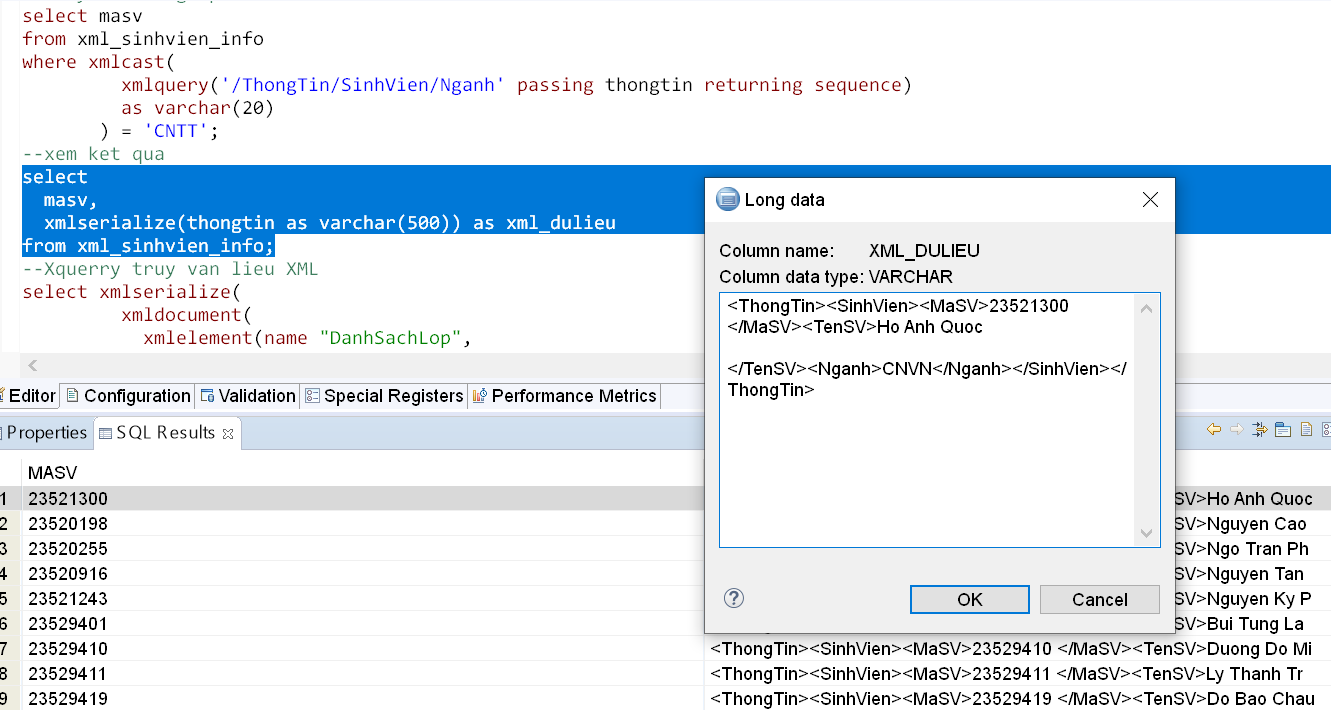
* Thêm dữ liệu vào bảng xml\_sinhvien\_info đã tạo ở trên từ bảng SinhVien



* Truy vấn bằng XPath để tìm MSSV sinh viên thuôc ngành CNTT



* Xem dữ liệu có trong bảng xml\_sinhvien\_info



Giải thích:

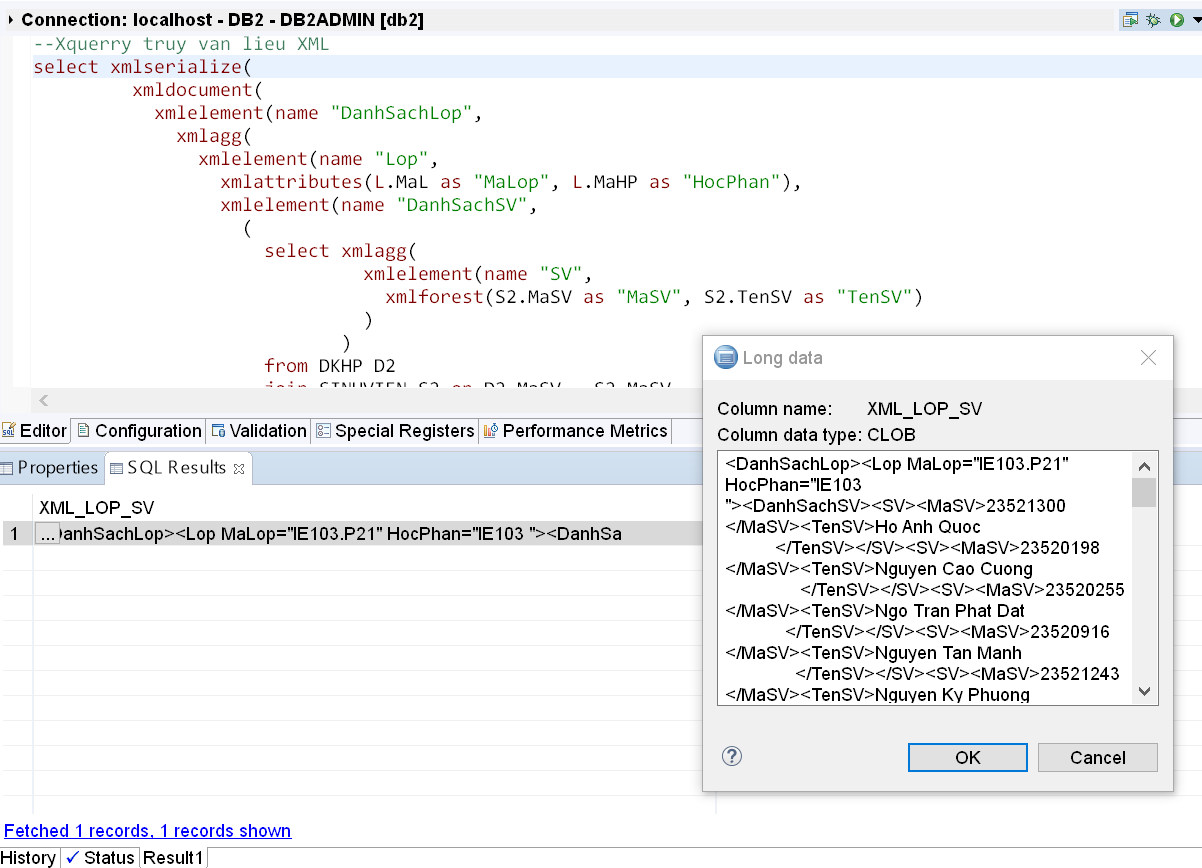
Lấy dữ liệu từ bảng **SinhVien** (chỉ lấy những sinh viên thuộc khoa 'ISE').

* Tạo tài liệu XML gồm:
* Gốc là <ThongTin>
* **xmlelement(...):** Tạo phần tử XML
* **xmldocument(...):** Gói nội dung thành tài liệu
* Bên trong là phần tử <SinhVien>, chứa 3 phần tử con:
* <MaSV>: mã sinh viên.
* <TenSV>: tên sinh viên.
* <Nganh>: mã ngành.
* **xmlquery(...)**: dùng XPath để trích thông tin từ XML.
* **xmlcast(...):** Ép kiểu dữ liệu kết quả XML
* **xmlserialize(...)** để chuyển kiểu dữ liệu XML thành chuỗi văn bản.

Yêu cầu: Danh sách các lớp học cùng với danh sách sinh viên đã đăng ký trong từng lớp thuộc khoa ISE, dùng XQuerry

| *--Xquerry truy van lieu XML*  *select xmlserialize(*  *xmldocument(*  *xmlelement(name "DanhSachLop",*  *xmlagg(*  *xmlelement(name "Lop",*  *xmlattributes(L.MaL as "MaLop", L.MaHP as "HocPhan"),*  *xmlelement(name "DanhSachSV",*  *(*  *select xmlagg(*  *xmlelement(name "SV",*  *xmlforest(S2.MaSV as "MaSV", S2.TenSV as "TenSV")*  *)*  *)*  *from DKHP D2*  *join SINHVIEN S2 on D2.MaSV = S2.MaSV*  *where D2.MaL = L.MaL and D2.TrangThai = 'Da dang ky'*  *)*  *)*  *)*  *)*  *)*  *) as clob(1m)*  *) as XML\_Lop\_SV*  *from LOP L*  *where exists (*  *select 1 from DKHP D where D.MaL = L.MaL and D.TrangThai = 'Da dang ky'*  *);* |
| --- |

Kết quả:



Giải thích:

* **Alias** **XML\_Lop\_SV**: tên của cột kết quả.
* **xmlagg(...):** Gộp nhiều phần tử XML thành một danh sách  
  + Tạo phần tử <Lop> cho mỗi lớp trong bảng LOP.  
  + Thêm 2 thuộc tính:
* MaLop: mã lớp.
* HocPhan: mã học phần (MaHP).
* Bên trong mỗi <Lop> là phần tử <DanhSachSV> chứa danh sách sinh viên.
* **xmlattributes(...):** Gán thuộc tính cho phần tử XML
* **xmlforest(...):** Tạo nhiều phần tử con XML
* **where exists:** Chỉ lấy những lớp có ít nhất 1 sinh đã đăng kí

**D. Tổng kết**

* Trong chương này, hệ thống quản lý đăng ký học phần đã được kiểm thử thông qua quá trình thiết kế cơ sở dữ liệu, khởi tạo dữ liệu mẫu và thực hiện các truy vấn SQL trên nền tảng IBM DB2. Các truy vấn được xây dựng phản ánh đúng các chức năng cốt lõi của hệ thống như: quản lý sinh viên, giảng viên, học phần và đăng ký học phần. Kết quả thực nghiệm cho thấy hệ thống hoạt động ổn định, dữ liệu được tổ chức logic, đảm bảo tính toàn vẹn và dễ dàng mở rộng. Những đánh giá đạt được về tính chính xác, hiệu quả và khả năng đáp ứng yêu cầu thực tế là nền tảng để triển khai hệ thống trong các môi trường ứng dụng thực tế sau này.