# MỤC LỤC

# *GIỚI THỆU CHUNG*

**SQL,** viết tắt của Structure Query Language, là một công cụ quản lý dữ liệu, đơn giản nhưng rất hiệu quả, được sử dụng phổ biến ở nhiều lĩnh vực. Mặc khác, hầu hết tất cả các ngôn ngữ lập trình bậc cao đều có hỗ trợ **SQL**. Các công cụ lập trình đều cho phép người sử dụng kết nối và truy cập tới CSDL bằng cách nhúng các câu lệnh SQL vào trong các ngôn ngữ lập trình hoặc viết lời gọi đến các chương trình con trên hệ quản trị CSDL.

**SQL** ngày càng đóng vai trò quan trọng khi mà hiện nay Internet ngày càng phát triển. SQL được sử dụng như là công cụ để giao tiếp giữa các trình ứng dụng phía máy khách với máy chủ cơ sở dữ liệu, **SQL** sẽ thực hiện việc truy cập thông tin và kết quả hiển thị trên ứng dụng khi người dùng yêu cầu.

Trong các hệ quản trị cơ sở dữ liệu, SQL xuất hiện với vai trò ngôn ngữ, là công cụ giao tiếp giữa người sử dụng và hệ quản trị cơ sở dữ liệu với nhiều vai trò khác nhau như: truy vấn dữ liệu, lập trình cơ sở dữ liệu, quản trị cơ sở dữ liệu, truy cập dữ liệu trên Internet, …

Để phục vụ nhu cầu học tập và nghiên cứu của sinh viên nói chung và sinh viên ngành Cao đẳng Bình Định nói riêng, một tài liệu tham khảo mang tính thực hành là cần thiết. Phân loại và giải chi tiết các dạng bài tập SQL sẽ giúp cho sinh viên nhận biết chính xác các dạng câu hỏi, sử dụng câu lệnh SQL hiệu quả nhất. Trong lập trình, tác giả sử dụng các thuật toán đơn giản, dễ hiểu để giải quyết các bài toán quản lý, đó là mục tiêu trong tài liệu này.

Trong tài liệu này, tác giả sử dụng CSDL Quản lý sinh viên làm bài mẫu từ đó sinh viên tự làm các bài tập trên CSDL bán hàng và các CSDL khác. Tài liệu cung cấp những kiến thức căn bản nhất về 2 nội dung chính là ngôn ngữ thao tác dữ liệu và lập trình với cơ sở dữ liệu, từ đó sinh viên có thể xây dựng một ứng dụng quản lý trên windows từ đơn giản đến phức tạp.

Trong mỗi chương tài liệu chia làm 2 phần chính là: tóm tắt lý thuyết và phân loại bài tập. Cụ thể chia thành 4 chương như sau:

**Chương 1:** Xây dựng Cơ sở dữ liệu. Trong chương này tác giả giới thiệu 2 CSDL mẫu, CSDL quản lý sinh viên và CSDL quản lý bán hàng, là 2 cơ sở dữ liệu mang tính cơ bản nhất, nó tập hợp tất cả các yêu cầu tổng quan để từ đó

sinh viên có thể làm một cách tương tự đối với các CSDL khác.

**Chương 2:** Câu lệnh truy vấn SQL. Trong chương này tác giả chia câu lệnh thao tác dữ liệu thành 6 dạng cơ bản, mỗi dạng có từ 3-4 bài tập minh hoạ, giúp sinh viên nhanh chóng nhận dạng đúng các yêu cầu của mỗi câu lệnh SQL.

**Chương 3:** Lập trình với SQL. Trong chương này tác giả chia cấu trúc lập trình thành 3 dạng: Hàm, Thủ tục và Con trỏ. Mỗi dạng bao gồm nhiều dạng bài khác nhau, mỗi dạng bài là cơ bản được tác giả chọn lọc và rất cần thiết trong lập trình ứng dụng sau này.

**Chương 4:** Một số đối tượng tiện ích khác, nhằm nâng cao kỹ năng lập trình, người lập trình phải hạn chế tối đa nhất các lỗi thường xẩy ra, lường trước lỗi và bẫy lỗi là kỹ năng cần thiết của người lập trình chuyên nghiệp.

**Phần đọc thêm**: Ứng dụng SQL trong lập trình C# căn bản. Trong chương này tác giả minh hoạ một số ứng dụng cơ bản, trong đó thể hiện một kết nối từ ứng dụng tới thủ tục trong hệ quản trị CSDL SQL Server. Giúp sinh viên thấy được mối liên hệ giữa lập trình CSDL với lập trình trên công cụ C#, được xem là kỹ thuật mang tính bảo mật cao.

Tài liệu tham khảo "Phân loại và giải chi tiết các dạng bài tập SQL" mang tính thực hành cao, là tài liệu gối đầu cho tất cả sinh viên đang ngồi ghế nhà trường, tài liệu giúp sinh viên học tốt các học phần liên quan như: Hệ quản trị CSDL Access, Hệ quản trị CSDL SQL, Lập trình Windows, Lập trình Website, …Tài liệu sẽ hoàn thiện hơn khi nhận nhiều ý kiến đóng góp quý báu của các bạn đọc. Tác giả rất mong nhận nhiều góp ý để tài liệu hữu ích hơn.

# CHƯƠNG 1. XÂY DỰNG CƠ SỞ DỮ LIỆU

## 1. CƠ SỞ DỮ LIỆU QUẢN LÝ SINH VIÊN

### 1.1 BÀI TOÁN

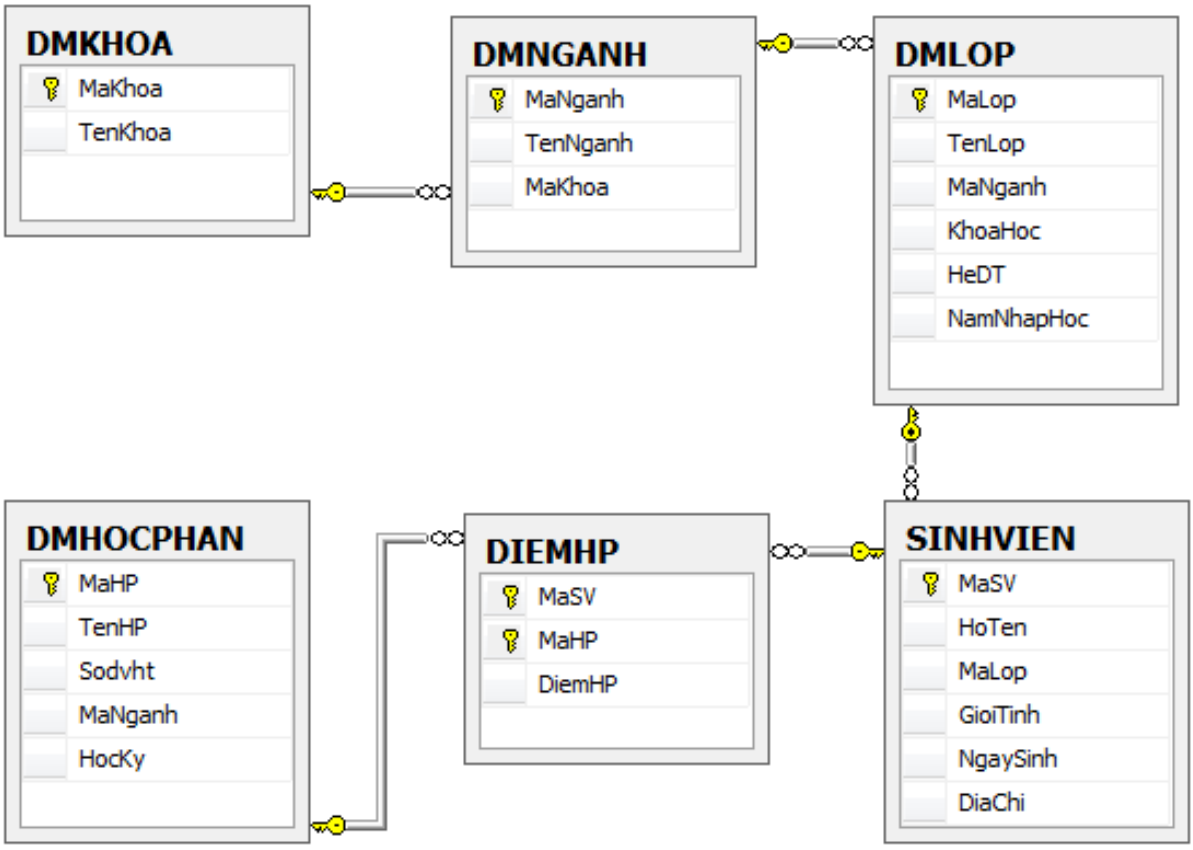
**Dữ liệu vào:**

- Danh mục các Ngành học của mỗi Khoa.
- Danh sách hồ sơ sinh viên gồm những thông tin đầu vào như Mã sinh viên, Họ tên, Giới tính, Ngày sinh, Địa chỉ, Khoá học, Hệ đào tạo, Khoa, Ngành học, Lớp học, ...
- Danh sách các học phần ở mỗi học kỳ theo từng ngành.
- Danh sách điểm học phần của mỗi sinh viên.

**Dữ liệu ra**:

- Thực hiện một số thống kê: Tính số lượng sinh viên mỗi lớp, mỗi ngành, mỗi khoa, …
- Phân lớp và đánh mã sinh viên theo các yêu cầu khác nhau từ dễ đến khó.
- Theo dõi chương trình giảng dạy các học phần theo từng ngành.
- Theo dõi điểm học phần của mỗi sinh viên của từng học kỳ, cả năm và cả khoá. Đưa ra danh sách sinh viên tích luỹ, ngừng tiến độ học tập.
- Đưa ra bảng điểm tổng hợp có xếp loại học tập theo từng lớp ở mỗi học kỳ, cả năm và cả khoá học.
- Xử lý dữ liệu: tạo các thủ tục (Procedure) hiển thị dữ liệu, tính toán, bổ sung, cập nhật, xoá, …

## 1.2. CƠ SỞ DỮ LIỆU QUAN HỆ




## 1.3. BẢNG DỮ LIỆU CHI TIẾT

### Bảng DMKHOA

| MaKhoa | TenKhoa |
|---|---|
| CNTT | Công nghệ thông tin |
| KT | Kế Toán |
| SP | Sư phạm |

### Bảng DMNGANH

| MaNganh | TenNganh | MaKhoa |
|---|---|---|
| 140902 | Sư pham toán tin | SP |
| 480202 | Tin học ứng dụng | CNTT |

### Bảng DMLOP

| MaLop | TenLop | MaNganh | KhoaHoc | HeDT | NamNhapHoc |
|---|---|---|---|---|---|
| CT11 | Cao đẳng tin học | 480202 | 11 | TC | 2013 |
| CT12 | Cao đẳng tin học | 480202 | 12 | CĐ | 2013 |
| CT13 | Cao đẳng tin học | 480202 | 13 | CĐ | 214 |

### Bảng SINHVIEN

| MaSV | HoTen | MaLop | GioiTinh | NgaySinh | DiaChi |
|---|---|---|---|---|---|
| 001 | Phan Thanh | CT12 | False | 09/12/1990 ... | Tuy Phuwowcs |
| 002 | Nguyễn Thị Cẩm... | CT12 | True | 01/12/1994 ... | Quy Nhơn |
| 003 | Võ Thị Hà | CT12 | True | 07/02/1995 ... | An Nhơn |
| 004 | Trần Hoài Nam | CT12 | False | 04/05/1994 ... | Tây sơn |
| 005 | Trần Văn Hoàng | CT13 | False | 08/04/1995 ... | Vĩnh Thạnh |
| 006 | Đặng Thị Thảo | CT13 | True | 06/12/1995 ... | Quy Nhơn |
| 007 | Lê Thị Sen | CT13 | True | 08/12/1994 ... | Phù Cát |
| 008 | Nguyễn Văn Huy | CT11 | False | 06/04/1995 ... | Phù Mỹ |
| 009 | Trần Thị Hoa | CT11 | True | 08/09/1994 ... | Hoài Nhơn |

## Bảng DMHOCPHAN

| MaHP | TenHP | Sodvht | MaNganh | HocKy |
|---|---|---|---|---|
| 001 | Toán cao cấp A1 | 4 | 480202 | 1 |
| 002 | Tiếng Anh 1 | 3 | 480202 | 1 |
| 003 | Vật lý đại cương | 4 | 480202 | 1 |
| 004 | Tiến anh 2 | 7 | 480202 | 1 |
| 005 | Tiếng anh 1 | 3 | 140909 | 2 |
| 006 | Xác suất thông kê | 3 | 140902 | 2 |

## Bảng DIEMHP

| MaSV | MaHP | DiemHP |
|---|---|---|
| 002 | 002 | 5.9 |
| 002 | 003 | 4.5 |
| 003 | 001 | 4.3 |
| 003 | 002 | 6.7 |
| 003 | 003 | 7.3 |
| 004 | 001 | 4.0 |
| 004 | 002 | 5.2 |
| 004 | 003 | 3.5 |
| 005 | 001 | 9.8 |
| 005 | 002 | 7.9 |
| 005 | 003 | 7.5 |
| 006 | 001 | 6.1 |
| 006 | 002 | 5.6 |
| 006 | 003 | 4.0 |
| 007 | 001 | 6.2 |

# 2. CƠ SỞ DỮ LIỆU QUẢN LÝ BÁN HÀNG

## 2.1. BÀI TOÁN

### Dữ liệu vào

- Danh sách các mặt hàng bán lẻ tại các cửa hàng (chẳng hạn như Siêu thị) gồm các thông tin Mã hàng, Tên hàng và đơn giá bán hiện tại.

- Danh sách các khách hàng Thành viên và VIP gồm các thông tin Họ tên, Địa chỉ, Số điện thoại.

- Danh sách các mặt hàng của từng hoá đơn.

### Dữ liệu ra

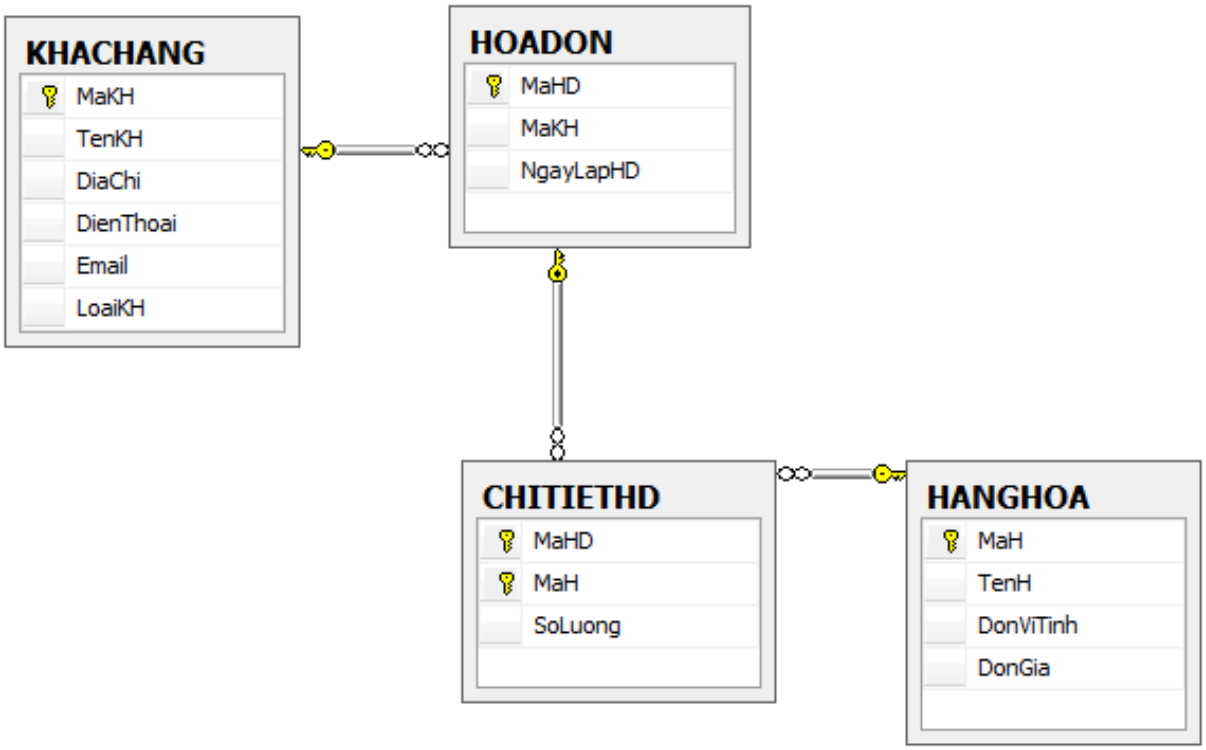
- Hoá đơn bán hàng cho mỗi khách hàng và theo dõi quá trình mua hàng của mỗi khách hàng để có những ưu đãi thích hợp.

- Theo dõi từng mặt hàng bán theo tháng, quý và năm. Những mặt hàng mức tiêu thụ cao, tiêu thu thấp để điều chỉnh giá phù hợp.

- Tổng hợp doanh thu của từng mặt hàng theo từng tháng, quý và năm.

- Tổng hợp tiền mua của từng khách hàng trong mỗi năm, tích điểm và in chiết khấu.

## 2.2. CƠ SỞ DỮ LIỆU QUAN HỆ



## 2.3. BẢNG DỮ LIỆU CHI TIẾT

### Bảng KHACHHANG

| MaKH | TenKH | DiaChi | DienThoai | Email | LoaiKH |
|---|---|---|---|---|---|
| KH001 | Nguyễn Thị MaiChi | Quy Nhơn | 09762334445 | MaiChi@gmail.com ... | VIP |
| KH00... | Phan Thị Thanh ... | Quy Nhơn | 09876655555 | *NULL* | TV |
| KH00... | Trần Văn Toàn | Tuy Phước | 98766555567 | ToanVan@gmail.com... | TV |
| KH00... | Trần Văn Ấn | Quy Nhơn | 98765545878 | *NULL* | VIP |

### Bảng HANGHOA

| MaH | TenH | DonViTinh | DonGia |
|---|---|---|---|
| H001 ... | Sữa đặc ông thọ | lon | 23000 |
| H002 ... | Keo dẻo Hồng Hà | gói | 80000 |
| H003 ... | Bánh xốp Quy Kinh đô | hộp | 120000 |
| H004 ... | Bánh quy LuXy | hộp | 150000 |
| H005 ... | Đường trắng Quy Hoà | gói | 20000 |
| H006 ... | Bánh LuXy Sài Gòn | hộp | 100000 |
| H007 ... | Sữa tươi TH TrueMilk | lốc | 30000 |

### Bảng HOADON

| MaHD | MaKH | NgayLapHD |
|---|---|---|
| 001 | KH001 | 01/02/2018 ... |
| 002 | KH001 | 02/03/2018 ... |
| 003 | KH002 | 01/02/2018 ... |
| 004 | KH002 | 01/03/2018 ... |
| 005 | KH003 | 02/03/2018 ... |
| 006 | KH004 | 02/05/2018 ... |
| 007 | KH003 | 03/05/2018 ... |
| 008 | KH003 | 04/05/2018 ... |

### Bảng CHITIETHD

| MaHD | MaH | SoLuong |
|---|---|---|
| 001 | H001 | 1 |
| 001 | H002 | 3 |
| 002 | H003 | 12 |
| 002 | H004 | 2 |
| 003 | H001 | 7 |
| 003 | H004 | 5 |
| 004 | H001 | 12 |
| 005 | H003 | 20 |
| 005 | H005 | 19 |
| 006 | H007 | 20 |
| 006 | H003 | 45 |
| 007 | H002 | 60 |
| 007 | H008 | 35 |

# CHƯƠNG 2. CÂU LỆNH TRUY VẤN SQL

## A. KIẾN THỨC CẦN NHỚ

### 1. Câu lệnh truy vấn với cấu trúc đơn giản

**Ý nghĩa**: Câu lệnh SELECT dùng để truy xuất dữ liệu từ một hay nhiều bảng.

**Cú pháp:**

SELECT [ALL|DISTINCT]|[TOP n] <danh_sách cột>

[INTO tên_bảng_mới]

FROM <bảng 1> INNER JOIN <bảng 2> ON <điều kiện kết nối>

... INNER JOIN <bảng n> ON <điều kiện kết nối>

[WHERE điều_kiện lọc]

[GROUP BY ds cột phân nhóm] [HAVING điều_kiện nhóm]

[ORDER BY cột_sắp_xếp][DESC | ASC]

***Giải thích*:**

- Danh sách cột: là dãy các cột/ biểu thức cột cách nhau bởi dấu phẩy. Dấu * có nghĩa là hiển thị tất cả các cột trong bảng.

- Tham chiếu đến cột Khoá của bảng: <Tên bảng>.<Tên cột>

- Điều kiện trong câu lệnh SELECT

  WHERE <Điều kiện>: Điều kiện nằm sau từ khóa WHERE, là một biểu thức Logic gồm các phép toán sau:

  Các toán tử kết hợp điều kiện: AND, OR

  Các toán tử so sánh: >,<,>=,<=,<>,!<,!>, =

  Kiểm tra giới hạn của dữ liệu: BETWEEN/NOT BETWEEN

  Toán tử thuộc tập hợp, không thuộc tập hợp:

IN (dãy giá trị | truy vấn SELECT|…), NOT IN ()

Kiểm tra khuôn dạng dữ liệu:

LIKE /NOT LIKE <nhóm ký tự đại diện>

Với ký tự đại diện:

%: đại diện cho một nhóm ký tự

_: đại diện cho một ký tự

[dãy ký tự]: ký tự đơn nằm trong dãy ký tự chỉ định như [0-9], [ABC]

[^dãy ký tự]: ký tự đơn KHÔNG nằm trong dãy ký tự chỉ định

- Một số hàm gộp dùng trong từ khoá GRPOUP

SUM([ALL|DISTINCT] biểu_thức): Tính tổng các giá trị của biểu thức.

AVG([ALL|DISTINCT] biểu_thức): Tính trung bình của các giá trị của biểu thức.

COUNT([ALL|DISTINCT]biểu_thức): Đếm số các giá trị trong biểu thức.

COUNT(*): Đếm số các dòng được chọn.

MAX(biểu_thức): Tính giá trị lớn nhất.

MIN(biểu_thức): Tính giá trị nhỏ nhất.

## 2. Câu lệnh truy vấn với cấu trúc phức tạp

### 2.1. Cấu trúc lồng nhau

**Ý nghĩa:**

Khi cần thực hiện phép kiểm tra giá trị của một biểu thức có thuộc hay không thuộc trong tập hợp các giá trị của truy vấn Con hay không, ta có thể sử dụng toán tử IN (NOT IN).

Nghĩa là có một truy vấn con được lồng vào trong điều kiện của một truy vấn chính, được dùng để lọc kết quả từ truy vấn chính bằng điều kiện IN hoặc NOT IN.

**Cấu trúc:**

SELECT . . .

WHERE <biểu_thức> [NOT] IN (Câu lệnh SELECT_con)

## 2.2. Cấu trúc lượng từ

**Ý nghĩa:**

Các lượng từ EXISTS, ALL, ANY : sử dụng trong trường hợp tập hợp các giá trị trong truy vấn con nhiều hơn một thì ta phải thêm lượng từ ở phía trước truy vấn con đó.

**Cấu trúc:**

Lượng từ ALL: thoả mãn tất cả các giá trị trong tập hợp

<Biểu thức cột> <Phép toán> All (Câu lệnh SELECT)

Lượng từ ANY : thoả mãn bất kỳ giá trị nào trong tập hợp

<Biểu thức cột> <Phép toán> ANY (Câu lệnh SELECT)

Lượng Từ EXISTS*:* Lượng từ EXISTS trả về giá trị True nếu kết quả của truy vấn Con khác rỗng, ngược lại trở về giá trị False. Tương tự NOT EXISTS

[NOT] EXISTS (truy_vấn_con)

## 2.3. Cấu trúc tập hợp

UNION: phép hợp

EXCEPT : phép hiệu

INTERSECT : phép giao

**Ý nghĩa:**

Mỗi truy vấn SELECT là một tập hợp các bộ giá trị. Các phép toán giữa các truy vấn cũng là phép toán trên tập hợp.

- Hợp của 2 hay nhiều truy vấn SELECT là một tập tất cả các bộ giá trị của các truy vấn đó.

- Hiệu của 2 truy vấn A và B là một tập tất cả các bộ giá trị thuộc truy vấn A nhưng không thuộc truy vấn B.

- Giao của 2 hay nhiều truy vấn SELECT là một tập gồm các bộ giá trị cùng thuộc các truy vấn đó.

**Yêu cầu**:

- Các dòng giống nhau trong tập kết quả sẽ bị loại bỏ.

- Các tập hợp tham gia trong phép toán phải khả hợp nghĩa là phải cùng tập các thuộc tính.

**Cú pháp**

*Câu_lệnh_1*

{UNION | EXCEPT | INTERSET} [ALL]

*Câu_lệnh_2*

[{UNION | EXCEPT | INTERSET} [ALL]

*Câu_lệnh_3*]

...

[{UNION | EXCEPT | INTERSET} [ALL]

Câu lệnh_n]

## 3. Bổ sung, cập nhật, xoá dữ liệu

### 3.1. Lệnh INSERT

**Ý nghĩa:**

Bổ sung các dòng dữ liệu vào cuối một bảng.

**Cú pháp lệnh**

**INSERT INTO *tên_bảng*[(*danh_sách_cột*)]**

**VALUES (*danh_sách_trị*)**

Bổ sung nhiều dòng dữ liệu bằng cách truy xuất dữ liệu từ các bảng dữ liệu khác.

**INSERT INTO *tên_bảng*[(*danh_sách_cột*)]**

**Câu lệnh SELECT**

## 3.2. Lệnh UPDATE

**Ý nghĩa:**

Cập nhật dữ liệu cho một hay nhiều cột trong bảng.

**Cú pháp**

UPDATE tên_bảng

SET Tên_cột 1 = biểu_thức 1,

Tên_cột 2 = biểu thức 2

[, ..., Tên_cột_k = biểu_thức_k]

[FROM Danh dách bảng]

[WHERE điều_kiện]

**WHERE:** Chỉ cập nhật cột có dòng thỏa mãn điều kiện.

**FROM:** Dữ liệu cập nhật liên qua tới nhiều bảng.

## 3.3. Lệnh DELETE

**Ý nghĩa:**

Xoá các dòng dữ liệu trong một bảng.

**Cú pháp**

DELETE FROM tên_bảng

[FROM danh_sách_bảng]

[WHERE điều_kiện]

Nếu không có WHERE thì xóa tất cả các dòng

## B. PHÂN LOẠI BÀI TẬP

### DẠNG 1: CÂU LỆNH TRUY VẤN CÓ ĐIỀU KIỆN

#### Bài số 1: Câu lệnh SQL không kết nối

1. Hiển thị danh sách gồm: MaSV, HoTen, MaLop, NgaySinh (dd/mm/yyyy), GioiTinh (Nam, Nữ) , Namsinh của những sinh viên có họ không bắt đầu bằng chữ N,L,T.

2. Hiển thị danh sách gồm: MaSV, HoTen, MaLop, NgaySinh (dd/mm/yyyy), GioiTinh (Nam, Nữ) , Namsinh của những sinh viên nam học lớp CT11.

3. Hiển thị danh sách gồm: MaSV, HoTen, MaLop, NgaySinh (dd/mm/yyyy), GioiTinh (Nam, Nữ) của những sinh viên học lớp CT11,CT12,CT13.

4. Hiển thị danh sách gồm: MaSV, HoTen, MaLop, NgaySinh (dd/mm/yyyy), GioiTinh (Nam, Nữ), Tuổi của những sinh viên có tuổi từ 19-21.

Lời giải:

```
1. SELECT MaSV, HoTen, MaLop,
   CONVERT(varchar(10),NgaySinh,103) AS NgaySinh,
   CASE  GioiTinh WHEN 1 THEN N'Nam' ELSE N'Nữ' END
   AS GioiTinh, YEAR(Ngaysinh) AS Namsinh
   FROM  SINHVIEN
   WHERE  HoTen  NOT  LIKE  N'[NLT]%'

2. SELECT MaSV, HoTen, MaLop,
   CONVERT(varchar(10),NgaySinh,103) AS NgaySinh,
   CASE GioiTinh WHEN 1THEN N'Nam' ELSE N'Nữ' END AS
   GioiTinh, YEAR(NgaySinh) AS NamSinh
   FROM SINHVIEN WHERE GioiTinh=1 AND MaLop='CT11'
3. SELECT MaSV, HoTen, MaLop, CONVERT(varchar(10),
   NgaySinh,103) AS NgaySinh,
```

```
   CASE GioiTinh WHEN 1THEN N'Nam' ELSE N'Nữ' END AS
   GioiTinh
   FROM SINHVIEN
   WHERE MaLop IN ('CT11','CT12', 'CT13')
4. SELECT MaSV, HoTen, MaLop,
   CONVERT(varchar(10),NgaySinh,103) AS NgaySinh,
   CASE GioiTinh WHEN 1THEN N'Nam' ELSE N'Nữ ' END AS
   Gioi tinh, YEAR(GETDATE())-YEAR(NgaySinh) AS Tuoi
   FROM SINHVIEN
   WHERE YEAR(GETDATE())-YEAR(NgaySinh)
   BETWEEN 19 AND 21
```

**Bài số 2: Câu lệnh SQL có kết nối**

1. Hiển thị danh sách gồm MaSV, HoTên, MaLop, DiemHP, MaHP của những sinh viên có điểm HP >= 5.

2. Hiển thị danh sách MaSV, HoTen , MaLop, MaHP, DiemHP được sắp xếp theo ưu tiên Mã lớp, Họ tên tăng dần.

3. Hiển thị danh sách gồm MaSV, HoTen, MaLop, DiemHP, MaHP của những sinh viên có điểm HP từ 5 đến 7 ở học kỳ I.

4. Hiển thị danh sách sinh viên gồm MaSV, HoTen, MaLop, TenLop, MaKhoa của Khoa có mã CNTT.

Lời giải:

```
1. SELECT SINHVIEN.MaSV, HoTen, MaLop, DiemHP, MaHP
   FROM SINHVIEN
   INNER JOIN  DIEMHP ON DIEMHP.MaSV=SINHVIEN.MaSV
   WHERE DiemHP>5
2. SELECT SINHVIEN.MaSV, HoTen, MaLop, TenLopDiemHP,
   MaHP
   FROM SINHVIEN
   INNER JOIN  DIEMHP ON DIEMHP.MaSV=SINHVIEN.MaSV
   INNER JOIN  DMLOP ON  SINHVIEN.MaLop=DMLOP.MaLop
   ORDER BY MaLop, HoTen ASC
```

```
3. SELECT SINHVIEN.MaSV, HoTen, MaLop, DiemHP, MaHP,
   Hocky
   FROM SINHVIEN
   INNER JOIN  DIEMHP ON DIEMHP.MaSV=SINHVIEN.MaSV
   WHERE  (DiemHP>=5 AND DiemHP<=7) AND  HocKy='1'
4. SELECT MaSV, HoTen, SINHVIEN.MaLop, TenLop, MaKhoa
   FROM SINHVIEN
   INNER JOIN  DMLOP ON SINHVIEN.MaLop=DMLOP.MaLop
   INNER JOIN  DMNGANH  ON DMNGANH.MaNganh=DMLOP.MaNganh
   WHERE  MaKhoa='CNTT'
```

## BÀI TẬP TỰ GIẢI

### Bài số 1:

1. Cho biết danh sách gồm MaKH, TenKH, NgaySinh, GioiTinh của khách hàng thành viên.

2. Cho biết danh sách gồm MaKH, TenKH, NgaySinh, GioiTinh của khách hàng nữ ở Quy Nhơn.

3. Cho biết danh sách gồm MaKH, TenKH, NgaySinh, GioiTinh của khách hàng VIP ở Quy Nhơn hoặc Tuy Phước.

4. Cho biết số lượng hoá đơn xuất vào tháng 8.

5. Cho biết danh sách các mặt hàng có giá bán từ 20 nghìn đến 50 nghìn.

6. Cho biết MaHD, MaH, SoLuong có số lượng bán >10.

- **Kết nối 2 hay nhiều bảng**

7. Cho biết MaHD, MaH, TenH, DonGia, SoLuong, ThanhTien của hoá đơn 001.

8. Cho biết MaHD, MaH, TenH, DonGia, SoLuong, ThanhTien có Thành tiền từ 1 triệu đến 2 triệu.

9. Cho biết thông tin khách hàng không mua hàng vào tháng 6.

10. Cho biết MaHD, NgayLapHD, MaHK, TenH, DonGia, SoLuong, ThanhTien bán vào tháng 6

11. Cho biết danh sách các mặt hàng đã bán được.

## DẠNG 2: CÂU LỆNH TRUY VẤN CÓ PHÂN NHÓM

### Bài số 1: Câu lệnh SQL có từ khoá GROUP BY không điều kiện.

1. Cho biết MaLop, TenLop, tổng số sinh viên của mỗi lớp.
2. Cho biết điểm trung bình chung của mỗi sinh viên, xuất ra bảng mới có tên DIEMTBC, biết rằng công thức tính DiemTBC như sau:

   DiemTBC = ∑ (DiemHP * SoDvht) / ∑ (SoDvht)

3. Cho biết điểm trung bình chung của mỗi sinh viên ở mỗi học kỳ.
4. Cho biết MaLop, TenLop, số lượng nam nữ theo từng lớp.

Lời giải:

```
1. SELECT SINHVIEN.MaLop, TenLop, COUNT(Masv) AS Siso
   FROM DMLOP INNER JOIN SINHVIEN ON
   DMLOP.MaLop=SINHVIEN.MaLop
   GROUP BY SINHVIEN.MaLop,TenLop
2. SELECT MaSV, SUM(DiemHP*Sodvht)/SUM(Sodvht) AS DiemTBC
   INTO DIEMTBC
   FROM DMHOCPHAN
   INNER JOIN  DIEMHP  ON DMHOCPHAN.MaHP=DIEMHP.MaHP
   GROUP BY MaSV
3. SELECT  HocKy,MaSV,SUM(DiemHP*Sodvht)/SUM(Sodvht) AS
   DiemTBC
   FROM DMHOCPHAN
   INNER JOIN DIEMHP ON DMHOCPHAN.MaHP=DIEMHP.MaHP
   GROUP BY HocKy,MaSV
   ORDER BY HocKy
```

```
4. SELECT SINHVIEN.MaLop,Tenlop,CASE  GioiTinh WHEN 0
   THEN N'Nữ' ELSE N'Nam' END AS GioiTinh, COUNT(MaSV)
   AS Soluong
   FROM DMLOP
   INNER JOIN SINHVIEN ON DMLOP.MaLop=SINHVIEN.MaLop
   GROUP BY SINHVIEN.MaLop,Tenlop,GioiTinh
   ORDER BY  SINHVIEN.MaLop
```

**Bài số 2: Câu lệnh SQL có từ khoá GROUP BY với điều kiện lọc.**

1. Cho biết điểm trung bình chung của mỗi sinh viên ở học kỳ 1.

   DiemTBC = ∑ (DiemHP * SoDvht) / ∑ (SoDvht)

2. Cho biết MaSV, HoTen, Số các học phần thiếu điểm (DiemHP<5) của mỗi sinh viên.

3. Đếm số sinh viên có điểm HP <5 của mỗi học phần.

4. Tính tổng số đơn vị học trình có điểm HP <5 của mỗi sinh viên.

Lời giải:

```
1. SELECT MaSV, SUM(DiemHP*Sodvht)/SUM(Sodvht) AS
   DiemTBC
   FROM DMHOCPHAN
   INNER JOIN DIEMHP ON DMHOCPHAN.MaHP=DIEMHP.MaHP
   WHERE HocKy='1'
   GROUP BY MaSV
2. SELECT SINHVIEN.MaSV, HoTen,  COUNT(MaHP) AS SLuong
   FROM DIEMHP
   INNER JOIN  SINHVIEN ON DIEMHP.MaSV=SINHVIEN.MaSV
   INNER JOIN  DMHOCPHAN ON DIEM.MaHP=DMHOCPHAN.MaHP
   WHERE DiemHP<5
   GROUP BY SINHVIEN.MaSV, HoTen
3. SELECT MaHP, COUNT(MaSV) AS SL_SV_Thieu
   FROM  DIEMHP
```

```
WHERE DiemHP<5
GROUP BY MaHP
```

4.
```
SELECT SINHVIEN.MaSV, Hoten, SUM(SoDVHT)AS
Tongdvht
FROM DIEMHP
INNER JOIN  SINHVIEN ON DIEMHP.MaSV=SINHVIEN.MaSV
INNER JOIN DMHOCPHAN ON DMHOCPHAN.MaHP=DIEMHP.MaHP
WHERE DiemHP<5
GROUP BY SINHVIEN.MaSV, HoTen
```

**Bài số 3: Câu lệnh SQL có từ khoá GROUP BY với điều kiện nhóm.**

1. Cho biết MaLop, TenLop có tổng số sinh viên >10.

2. Cho biết HoTen sinh viên có điểm Trung bình chung các học phần <3.

3. Cho biết HoTen sinh viên có ít nhất 2 học phần có điểm <5.

4. Cho biết HoTen sinh viên học TẤT CẢ các học phần ở ngành 140902.

5. Cho biết HoTen sinh viên học ít nhất 3 học phần mã '001', '002', '003'.

Lời giải:

1.
```
SELECT SINHVIEN.MaLop, Tenlop, COUNT(MaSV) AS Siso
FROM DMLOP
INNER JOIN SINHVIEN ON DMLOP.MaLop=SINHVIEN.MaLop
GROUP BY SINHVIEN.MaLop, Tenlop
HAVING   COUNT(MaSV)>10
```

2.
```
SELECT MaSV,HoTen ,SUM(DiemHP*Sodvht)/SUM(Sodvht) AS
DiemTBC
FROM DMHOCPHAN
INNER JOIN DIEMHP ON DMHOCPHAN.MaHP=DIEMHP.MaHP
INNER JOIN SINHVIEN ON SINHVIEN.MaSV=DIEMHP.MaSV
GROUP BY MaSV,HoTen
```

```
   HAVING  SUM(DiemHP*Sodvht)/SUM(Sodvht)<3
3. SELECT SINHVIEN.MaSV,Hoten,COUNT(MaHP) AS  Soluong
   FROM DIEMHP
   INNER JOIN  SINHVIEN  ON  DIEMHP.MaSV=SINHVIEN.MaSV
   WHERE  DiemHP<5
   GROUP BY SINHVIEN.MaSV, HoTen
   HAVING  COUNT(MaHP)>=2
4. SELECT HoTen,COUNT(MaHP) AS Soluong
   FROM DIEMHP
   INNER JOIN  SINHVIEN ON DIEMHP.MaSV=SINHVIEN.MaSV
   INNER JOIN DMLOP ON DMLOP.MaLop=SINHVIEN.MaLop
   WHERE MaNganh='140902'
   GROUP BY HoTen
   HAVING  COUNT(MaHP)=(SELECT  COUNT(MaHP) FROM
   DMHOCPHAN WHERE MaNganh='140902')
5. SELECT HoTen, COUNT(MaHP) AS  Soluong
   FROM DIEHP
   INNER JOIN SINHVIEN ON DIEMHP.MaSV=SINHVIEN.MaSV
   WHERE MaHP IN ('001','002','003')
   GROUP BY HoTen
   HAVING   COUNT(MaHP)>=3
```

**Bài số 4: Câu lệnh SQL có từ khoá TOP.**

1. Cho biết MaSV, HoTen  sinh viên có điểm TBC cao nhất ở học kỳ 1.

2. Cho biết MaSV, HoTen sinh viên có số học phần điểm HP  <5 nhiều nhất.

3. Cho biết MaHP, TenHP có số sinh viên điểm HP <5 nhiều nhất.

Lời giải:

```
1. SELECT TOP 1 SINHVIEN.MaSV, HoTen,
   SUM(DiemHP*Sodvht)/SUM(Sodvht) AS DiemTBC
```

```
   FROM DMHOCPHAN
   INNER JOIN DIEMHP ON DMHOCPHAN.MaHP=DIEMHP.MaHP
   INNER JOIN SINHVIEN ON SINHVIEN.MaSV=DIEMHP.MaSV
   WHERE  HocKy='1'
   GROUP BY SINHVIEN.MaSV, HoTen
   ORDER BYSUM(DiemHP*Sodvht)/SUM(Sodvht)  DESC
2. SELECT TOP 1 SINHVIEN.MaSV, HoTen,
   COUNT(MaHP) AS 'So Hoc phan'
   FROM DIEMHP
   INNER JOIN SINHVIEN ON SINHVIEN.MaSV=DIEMHP.MaSV
   WHERE DiemHP<5
   GROUP BY SINHVIEN.MaSV, HoTen
   ORDER BY COUNT(MaHP) DESC
3. SELECT TOP 1 DMHOCPHAN.MaHP, TenHP,
   COUNT(MaSV) AS 'So sinh vien'
   FROM  DMHOCPHAN
   INNER JOIN  DIEMHP  ON DMHOCPHAN.MaHP=DIEMHP.MaHP
   WHERE DiemHP<5
   GROUP BY DMHOCPHAN.MaHP, TenHP
   ORDER BY COUNT(MaSV)  DESC
```

**BÀI TẬP TỰ GIẢI:**

1. Cho biết MaKH, TenKH, Tổng Thành tiền của từng khách hàng.

2. Cho biết MaKH, TenKH, Tổng Thành tiền của khách hàng VIP.

3. Cho biết MaKH, TenKH, Tổng Thành tiền của từng khách hàng có Tổng thành tiền mua được >=20 triệu.

4. Cho biết MaH, TenH, Tổng số lượng của từng mặt hàng.

5. Cho biết MaHD, Tổng thành tiền của những hoá đơn có tổng thành tiền lớn hơn 5 triệu.

6. Cho biết hoá đơn bán ít nhất hai mặt hàng H001 và H002

7. Cho biết MaKH mua tất các các mặt hàng bánh.

8. Đếm số hoá đơn của mỗi khách hàng.

9. Cho biết Cho biết MaHD, Tổng thành tiền, Khuyến mãi 5% cho những hoá đơn có tổng thành tiền lớn hơn 500 nghìn.

10. Cho biết thông tin khách hàng VIP có tổng thành tiến trong năm 2018 nhỏ hơn 20 triệu.

11. Cho biết hoá đơn có tổng trị giá lớn nhất gồm các thông tin: Số hoá đơn, ngày bán, tên khách hàng, địa chỉ khách hàng, tổng trị giá của hoá đơn.

12. Cho biết hoá đơn có tổng trị giá lớn nhất trong tháng 5/2000 gồm các thông tin: Số hoá đơn, ngày, tên khách hàng, địa chỉ khách hàng, tổng trị giá của hoá đơn.

13. Cho biết hoá đơn có tổng trị giá nhỏ nhất gồm các thông tin: Số hoá đơn, ngày, tên khách hàng, địa chỉ khách hàng, tổng trị giá của hoá đơn.

14. Cho biết các thông tin của khách hàng có số lượng hoá đơn mua hàng nhiều nhất.

15. Cho biết các thông tin của khách hàng có số lượng hàng mua nhiều nhất.

16. Cho biết các thông tin về các mặt hàng mà được bán trong nhiều hoá đơn nhất.

17. Cho biết các thông tin về các mặt hàng mà được bán nhiều nhất.

## DẠNG 3: CÂU LỆNH TRUY VẤN VỚI CẤU TRÚC LỒNG NHAU

### Bài số 1: Cấu trúc lồng nhau phủ định (KHÔNG, CHƯA).

1. Cho biết Họ tên sinh viên KHÔNG học học phần nào.

2. Cho biết Họ tên sinh viên CHƯA học học phần có mã ‘001’.

3. Cho biết Tên học phần KHÔNG có sinh viên điểm HP <5.

4. Cho biết Họ tên sinh viên KHÔNG có học phần điểm HP<5

Lời giải:

```
1. SELECT MaSV, Hoten  FROM SINHVIEN
   WHERE MaSV NOT IN (SELECT MaSV FROM DIEMHP)
2. SELECT MaSV,HoTen  FROM SINHVIEN
   WHERE MaSV NOT IN (SELECT MaSV FROM DIEMHP
                      WHERE MaHP='001')
3. SELECT MaHP,TenHP FROM DMHOCPHAN
   WHERE MaHP NOT IN (SELECT MaHP FROM DIEMHP
                                 WHERE DiemHP<5)
4. SELECT DISTINCT SINHVIEN.MaSV, HoTen
   FROM SINHVIEN
   WHERE SINHVIEN.MaSV  NOT IN (SELECT DISTINCT MaSV
                           FROM DIEMHP WHERE DiemHP<5)
```

**Bài số 2: Cấu trúc lồng nhau không kết nối.**

1. Cho biết Tên lớp có sinh viên tên Hoa.
2. Cho biết HoTen sinh viên có điểm học phần ‘001’là <5.
3. Cho biết danh sách các học phần có số đơn vị học trình lớn hơn hoặc bằng số đơn vị học trình của học phần mã 001.

Lời giải:

```
1. SELECT TenLop FROM DMLOP
   WHERE MaLop IN (SELECT MaLop FROM SINHVIEN
                    WHERE HoTen LIKE N'% Hoa')
2. SELECT HoTen FROM SINHVIEN
   WHERE MaSV IN (SELECT MaSV FROM DIEMHP
                  WHERE DiemHP<5 AND MaHP='001')
3. SELECT  *  FROM  DMHOCPHAN
   WHERE SoDvht>=(SELECT SoDvht  FROM DMHOCPHAN
                  WHERE MaHP='001')
```

## BÀI TẬP TỰ GIẢI

1. Cho biết MaH, TenH chưa được bán.
2. Cho biết thông tin khách hàng chưa mua hàng vào tháng 5
3. Cho biết thông tin mặt hàng chưa được bán vào tháng 2.
4. Cho biết TenKH có mua mặt hàng BÁNH.

## DẠNG 4: CÂU LỆNH TRUY VẤN VỚI LƯỢNG TỪ ALL, ANY, EXISTS

### Bài số 1: Lượng từ ALL

1. Cho biết HoTen sinh viên có DiemHP cao nhất.
2. Cho biết HoTen sinh viên có tuổi cao nhất.
3. Cho biết MaSV, HoTen sinh viên có điểm học phần mã ‘001’ cao nhất.

**Lời giải:**

```
1. SELECT SINHVIEN.MaSV, HoTen, MaHP, DiemHP
   FROM DIEMHP
   INNER JOIN SINHVIEN ON SINHVIEN.MaSV=DIEMHP.MaSV
   WHERE DiemHP >=ALL(SELECT DiemHP FROM DIEMHP )
2. SELECT  HoTen, YEAR(GETDATE())-YEAR(NgaySinh)
   FROM SINHVIEN
   WHERE YEAR(GETDATE())-YEAR(NgaySinh) >=
   ALL (SELECT YEAR(GETDATE())-YEAR(NgaySinh)
   FROM SINHVIEN)
3. SELECT SINHVIEN.MaSV, HoTen
   FROM DIEMHP
   INNER JOIN SINHVIEN ON DIEMHP.MaSV=SINHVIEN.MaSV
   WHERE MaHP='001' AND DiemHP >=ALL(SELECT  DiemHP
                              FROM DIEMHP WHERE
   MaHP='001')
```

**Bài số 2: Lượng từ ANY**

1. Cho biết MaSV, MaHP có điểm HP lớn hơn bất kỳ các điểm HP của sinh viên mã ‘001’.
2. Cho biết sinh viên có điểm học phần nào đó lớn hơn gấp rưỡi điểm trung bình chung của sinh viên đó.

Lời giải:

1. 
```
SELECT MaSV, MaHP FROM DIEMHP WHERE DiemHP
>ANY(SELECT DiemHP FROM DIEMHP WHERE MaSV='001')
```

2. 
```
SELECT MaSV FROM DIEMTBC

WHERE DiemTBC*1.5 < ANY(SELECT DiemHP FROM DIEMHP
WHERE DIEMHP.MaSV=DIEMTBC.MaSV)
```

   *Chú ý*: bảng DIEMTBC được tạo ra khi thực hiện lệnh GROUP BY sau:

```
SELECT MaSV SUM(DiemHP*Sodvht)/SUM(Sodvht) AS
DiemTBC

IN TO DIEMTBC

FROM DMHOCPHAN INNER JOIN DIEMHP ON
DMHOCPHAN.MaHP=DIEMHP.MaHP

GROUP BY MaSV
```

**Bài số 3: Lượng từ EXISTS**

1. Cho biết MaSV, HoTen sinh viên đã ít nhất một lần học học phần nào đó.
2. Cho biết MaSV, HoTen sinh viên đã không học học phần nào.
3. Cho biết MaLop,TenLop đã không có sinh viên nào học.

Lời giải:

1. 
```
SELECT MaSV, HoTen FROM SINHVIEN

WHERE EXISTS(SELECT  *  FROM DIEMHP

                WHERE SINHVIEN.MaSV=DIEMHP.MaSV)
```

```
2. SELECT MaSV, HoTen  FROM SINHVIEN
   WHERE  NOT EXISTS(SELECT * FROM DIEMHP
                     WHERE SINHVIEN.MaSV=DIEMHP.MaSV)
3. SELECT MaLop,TenLop FROM DMLOP
   WHERE NOT EXISTS (SELECT *FROM SINHVIEN
                     WHERE SINHVIEN.MaLop=DMLOP.MaLop)
```

## DẠNG 5: CÂU LỆNH TRUY VẤN VỚI CẤU TRÚC TẬP HỢP

1. Cho biết MaSV đã học ít nhất một trong 2 học phần có mã là ‘001’, ‘002’.
2. Cho biết MaSV chưa học học phần nào.
3. Cho biết Mã sinh viên học ít nhất hai học phần có mã ‘001’ và ‘002’.

Lời giải:

```
1. SELECT MaSV FROM DIEMHP WHERE MaHP='001'
   UNION (SELECT MaSV FROM DIEMHP WHERE MaHP='002')
2. SELECT MaSV FROM SINHVIEN
   EXCEPT (SELECT MaSV FROM DIEMHP)
3. SELECT MaSV FROM DIEMHP WHERE MAHP='001'
   INTERSECT  (SELECT MaSV FROM DIEMHP WHERE
   MAHP='002')
```

## DẠNG 6: CÂU LỆNH BỔ SUNG, CẬP NHẬT, XOÁ DỮ DIỆU

### Bài số 1: Lệnh INSERT bổ sung dữ liệu

1. Bổ sung một dòng dữ liệu cho bảng DMKHOA bộ giá trị sau:

   (‘KT’, ‘Kế toán’).

2. Bổ sung một sinh viên cho bảng SINHVIEN (dữ liệu nào bất kỳ).

3. Bổ sung điểm học phần cho bảng DIEMHP (dữ liệu bất kỳ).

Lời giải:

```
1. INSERT INTO  KHOA(MaKhoa,TenKhoa) VALUES ('KT',N'Kế
   toán')
```

Hoặc

```
   INSERT INTO   KHOA VALUES ('KT', N'Kế toán', NULL)

2. INSERT INTO  SINHVIEN

      VALUES ('012', N'Nguyễn Văn Hoà', 'CT12', 'True',
   '12/02/1994', N'Quy Nhơn')

3. INSERT INTO  DIEMHP VALUES ('012', '001', 7)
```

**Bài số 2: Lệnh DELETE xoá dữ liệu**

1. Xóa những sinh viên  có DTBC <3 (sinh viên buộc thôi học).
2. Xóa những sinh viên không học học phần nào.
3. Xóa khỏi bảng DMLOP những lớp không có sinh viên nào

Lời giải:

Trước hết hãy tính điểm TBC (trung bình chung) của mỗi sinh viên và xuất ra bảng DIEMTBC.

```
   SELECT MaSV, SUM(DiemHP*Sodvht)/SUM(Sodvht) AS
   DiemTBC
   IN TO DIEMTBC
   FROM DMHOCPHAN
   INNER JOIN DIEMHP ON DMHOCPHAN.MaHP=DIEMHP.MaHP
   GROUP BY MaSV
1. DELETE FROM SINHVIEN
   WHERE MaSV IN (SELECT MaSV FROM DIEMTBC
                  WHERE DiemTBC<3)
2. DELETE FROM SINHVIEN
   WHERE MaSV NOT IN (SELECT DISTINCT MaSV
                      FROM DIEMHP)
```

```
3. DELETE FROM DMLOP
   WHERE MaLop NOT IN (SELECT DISTINCT MaLop
                       FROM SINHVIEN)
```

**Bài số 3: Lệnh UPDATE cập nhật dữ liệu**

1. Thêm cột XepLoai, Cập nhật dữ liệu cột XepLoai theo yêu cầu sau:

   Nếu DiemTBC >=8 thì xếp loại Giỏi, ngược lại

   Nếu DiemTBC >=7 thì xếp loại Khá, ngược lại

   Nếu DiemTBC >=5 thì xếp loại Trung bình, Ngược lại là yếu

2. Thêm cột XetLenLop, Cập nhập dữ liệu cho cột với yêu cầu sau:

   Nếu DiemTBC >=5 thì được lên lớp, ngược lại

   Nếu DiemTBC>=3 thì tạm ngừng tiến độ học tập

   Ngược lại Buộc thôi học.

Lời giải:

Bảng DIEMTBC được tạo ra từ câu lệnh GROUP BY ở phần trên.

Thêm cột XepLoai, XepLenLop cho bảng DIEMTBC.

```
ALTER TABLE DIEMTBC ADD XepLoai nvarchar(10)
ALTER TABLE DIEMTBC ADD XetLenLop nvarchar(50)
UPDATE DIEMTBC SET XepLoai = CASE
          WHEN DiemTBC>=8 THEN N'Giỏi'
          WHEN DiemTBC>=7 THEN N'Khá'
          WHEN DiemTBC>=5 THEN N'Trung bình'
          ELSE N'Yếu'
          END
1. UPDATE DIEMTBC SET XetLenLop= CASE
          WHEN DiemTBC >=5 THEN N'Được lên lớp'
          WHEN DiemTBC>=3 THEN  N'Tạm ngừng tiến độ '
                          ELSE N'Buộc thôi học'
          END
```

# CHƯƠNG 3: LẬP TRÌNH VỚI SQL

## A. KIẾN THỨC CẦN NHỚ

### 1. Khai báo và sử dụng biến

**Có 2 loại biến**: Cục bộ và toàn cục

**Biến cục bộ**: là biến chỉ sử dụng trong đoạn chương trình khai báo nó như Query Batch, Stored Procedure, Function, chứa giá trị thuộc một kiểu nhất định.

Biến cục bộ được bắt đầu bằng 1 ký hiệu @

**Khai báo**:

**DECLARE** <@*tên_biến*> <*Kiểu_dữ_liệu* >, …

**Gán giá trị cho biến**

**SET** @t*ên_biến* = {*giá_trị*|*biến*|b*iểu_thức*|*SELECT …*}

**Biến toàn cục**: là biến được sử dụng bất kỳ đâu trong hệ thống. Trong SQL biến toàn cục là các biến hệ thống do SQL Server cung cấp.

SQL tự cập nhật giá trị cho các biến này, người sử dụng không thể gán giá trị trực tiếp cho biến này

Bản chất là 1 hàm (Function) và bắt đầu bằng ký tự @@

**Một số biến toàn cục trong SQL**

| Tên biến | Ý nghĩa |
|---|---|
| @@ERROR | Mã số lỗi của câu lệnh T-SQL |
| @@FETCH_STATUS | Trạng thái truy cập Con trỏ:<br>0 nếu trạng thái truy cập thành công,<br>-1 nếu không thành công |

| | |
|---|---|
| @@IDENTITY | Giá trị xác định (identity) được thêm vào |
| @@ROWCOUNT | Số lượng dòng của kết quả câu lệnh SQL |
| @@SERVERNAME | Tên của Server địa phương |
| @@TRANSCOUNT | Số lượng những giao dịch đang được mở |
| @@VERSION | Thông tin về phiên bản SQL Server đang dùng |
| @@CURSOR_ROWS | Số lượng các dòng dữ liệu của Con trỏ |

## 2. Một số cấu trúc lệnh cơ bản

### 2.1. Cấu trúc IF…

**Cú pháp:**

***IF*** *<điều kiện>*

*Lệnh| Khối_lệnh 1*

[***ELSE*** *Lệnh| Khối_lệnh*]

*Khối lệnh là một hoặc nhiều lệnh nằm trong cặp từ khóa BEGIN…END*

***Giải thích cấu trúc***

Kiểm tra điều kiện, nếu điều kiện đúng thì thực hiện khối lệnh 1, ngược lại thực hiện khối lệnh 2 và kết thúc.

### 2.2. Cấu trúc CASE

**Cú pháp: Có hai dạng**

| **Dạng 1:** | **Dạng 2:** |
|---|---|
| CASE *Biểu_thức* | CASE |
| WHEN *Giá_trị 1 T*hen *kết_quả 1* | WHEN <*điều kiện 1*> THEN *kết_quả 1* |
| [WHEN Giá_trị 2 Then Kết_quả 2 | WHEN <*điều kiện 2*> THEN *kết_quả 2* |
| [...n] | [...n] |
| [ ELSE *kết_quả_khác*] | [ ELSE *kết_quả_khác*] |
| END | END |

*Giải thích dạng 1*:

Nếu biểu thức là giá trị 1 thì nhận kết quả 1 và kết thúc CASE, ngược lại nếu biểu thức là giá trị 2 thì nhận kết quả 2 và kết thúc CASE, … , ngoài ra thì nhận kết quả khác và kết thúc CASE.

*Giải thích dạng 2*:

Kiểm tra điều kiện, nếu điều kiện 1 đúng thì nhận kết quả 1 và kết thúc CASE, ngược lại nếu điều kiện 2 đúng thì nhận kết quả 2 và kết thúc CASE, …, ngoài ra nhận kết quả khác và kết thúc CASE.

## 2.3. Cấu trúc WHILE

**Cú pháp**

**WHILE** <*điều_kiện*>

BEGIN

*Lệnh*| *Khối lệnh*

[BREAK]

[CONTINUE]

*END*

Có thể thêm *Break* và *Continue* trong khối lệnh của while

BREAK: thoát khỏi vòng WHILE hiện hành.

CONTINUE : trở lại đầu vòng WHILE , bỏ qua các lệnh sau đó

***Giải thích cấu trúc***

Kiểm tra điều kiện, nếu điều kiện đúng thì thực hiện khối lệnh, tiếp tục kiểm tra điều kiện, cho đến khi nào điều kiện sai thì thoát khỏi WHILE.

Để vòng lặp không bị vô hạn thì trong nhóm lệnh phải có lệnh thay đổi điều kiện và sau một số lần lặp thì điều kiện sẽ sai và kết thúc WHILE.

## 3. THỦ TỤC (Stored Procedure)

Thủ tục là một đối tượng trong hệ quản trị CSDL bao gồm các câu lệnh SQL, chúng được kết hợp lại với nhau thành một khối lệnh, dùng để thực hiện một số công việc nào đó như cập nhật, thêm mới, xóa, hiển thị, tính toán và có thể trả về các giá trị.

**Thủ tục hệ thống**: là những thủ tục do SQL cung cấp (tự nghiên cứu System Stored Procedures) tên có tiếp đầu ngữ *sp_*

**Thủ tục người dùng**: do người dùng tạo ra, để dễ dàng phân biệt chúng ta quy định tên thủ tục có tiếp đầu ngữ *usp_*

**Tạo thủ tục:**

**CREATE PROCEDURE** *<Tên thủ tục>*

*Danh sách tham số vào*

*[Danh sách tham số ra <Output>]*

**AS**

**<Đoạn chương trình xử lý>**

[**RETURN** [*giá trị trả về*] ]

**GO**

***Lưu ý:***

Lệnh RETURN được sử dụng để kết thúc thủ tục và trả về giá trị là một số. Giá trị mặc định là RETURN 0 nghĩa là công việc thành công, quy ước RETURN -1 công việc không thành công.

**Lời gọi thủ tục**

**EXECUTE tên_thủ_tục [danh_sách_các_đối_số]**

Số lượng các đối số cũng như thứ tự của chúng phải phù hợp với số lượng và thứ tự của các tham số khi định nghĩa thủ tục.

**Chỉnh sửa thủ tục**

Thay từ khóa CREATE trong lệnh tạo thủ tục bằng từ khóa ALTER.

**Xóa thủ tục**

DROP PROCEDURE <Tên thủ tục>

**Mã hóa thủ tục**

Thủ tục sẽ được mã hoá nếu tuỳ chọn WITH ENCRYPTION được chỉ định. Nếu thủ tục đã được mã hoá, ta không thể xem được nội dung của thủ tục.

Thêm từ khóa WITH ENCRYPTION trong lệnh ALTER thủ tục.

**Biên dịch lại thủ tục**

Khi người sử dụng làm thay đổi tới những index của bảng. Stored Procedures phải được biên dịch lại (recompiled) để chấp nhận những thay đổi đó.

Thêm từ khóa WITH RECOMPILE **trong lệnh ALTER thủ tục**

## 4. HÀM (Function)

Hàm là một đối tượng trong hệ quản trị CSDL, tương tự như thủ tục. Điểm khác biệt giữa hàm và thủ tục là hàm trả về một giá trị**.** Giá trị trả về có thể là một bảng có được từ một câu truy vấn**.**

**Hàm hệ thống**: System Function.

**Hàm người dùng**: Do người dùng tạo ra gồm 3 dạng:

- Scalar_valued Function

Giá trị trả về là kiểu dữ liệu cơ sở (int, varchar, float, datetime…)

- Table_valued Function:

Giá trị trả về là một Table có được từ một câu truy vấn

- Aggregate Function:

Giá trị trả về là một bảng mà dữ liệu có được nhờ tích lũy dần sau một chuỗi thao tác xử lý và insert

**Tạo hàm:**

CREATE FUNCTION tên_hàm ([danh_sách_tham_số vào])

RETURNS (kiểu_trả_về_của_hàm| Table)

AS

BEGIN

Các_câu_lệnh_của_hàm

RETURN {Giá trị | Biến | Biểu thức | *Câu lệnh truy vấn*}

END

## 5. CON TRỎ (Cursor)

Là một cấu trúc dữ liệu, ánh xạ đến một danh sách gồm các dòng dữ liệu từ một kết quả truy vấn (SELECT), cho phép duyệt tuần tự các dòng dữ liệu và đọc giá trị từng dòng trong danh sách kết quả.

**Định nghĩa Con trỏ**

**DECLARE <Tên Con trỏ> CURSOR**

**FOR <Câu lệnh Truy vấn SELECT>**

*CON trỏ là cấu trúc toàn cục, duyệt theo một chiều từ đầu đế cuối, nội dung của Con trỏ có thể thay đổi.*

**Kiểm tra tình trạng Con trỏ:**

Biến hệ thống @@FETCH_STATUS

Cho biết lệnh fetch vừa thực hiện có thành công hay không. Là cơ sở để biết đã duyệt đến cuối danh sách hay chưa

Nếu @@FETCH_STATUS =0 thì thành công, Con trỏ đang ở vị trí dòng thỏa mãn điều kiện trong kết quả truy vấn.

Nếu @@FETCH_STATUS <>0 thì KHÔNG thành công, Con trỏ đang ở vị trí vượt qua dòng cuối cùng kết quả truy vấn.

**Các bước sử dụng Con trỏ trong lập trình**

B1. Định nghĩa CURSOR từ một kết quả SELECT

DECLARE <tên Con trỏ> CURSOR FOR <Câu lệnh SELECT>

B2. Mở Cursor:

OPEN <Ten Con trỏ> , Con trỏ tham chiếu đến dòng 0

B3. Truy cập đến các bản ghi

FETCH NEXT FROM <Cursor_name> INTO <ds biến>

B4. Kểm tra có thành công không:

Nếu @FETCH_STATUS = 0 thì xử lý lệnh, quay lại B3

Nếu @FETCH_STATUS <> 0 thì sang B5

B5. Đóng Cursor:

CLOSE <Cursor_name>

B6. Xoá tham chiếu của Cursor:

DEALLOCATE <Cursor_name>

Sử dụng Con trỏ có thể đến vị trí một dòng nhất định trong tập kết quả.

## 6. Một số hàm cơ bản:

### 6.1. Các hàm toán học:

1. ABS(*x*) : Trị tuyệt đối của x.
2. SQRT(*x*) : Căn bậc hai của x.
3. SQUARE( x) : x bình phương.
4. POWER( y, x ) : y lũy thừa x.
5. LOG(*x*) : Logarit  của x.
6. EXP(*x*) : Hàm mũ cơ số e của x.
7. SIGN(*x*) : Lây dấu của số x (-1: x<0, 0: x=0, +1: x>0).
8. ROUND(*x,n*) : Làm tròn tới n số thập phân.
9. CEILING( x) : Số nguyên nhỏ nhất nhưng lớn hơn x.
10. FLOOR(X) : Số nguyên lớn nhất nhưng nhỏ hơn x.
11. ... và các hàm lượng giác: SIN, COS, TAN, ASIN, ACOS, ATAN

...

### 6.2. Các hàm xử lý chuỗi

1. ACSII( ch ) : Mã ASCII của ký tự ch.
2. CHAR( n ) : Ký tự có mã ASCII là n.
3. LOWER( str ) : Trả vê chuỗi chữ thường.
4. UPPER(str) : Trả về chuỗi in hoa.
5. LTRIM(str) : Trả về chuỗi không có dấu cách bên trái.
6. RTRIM(str) : Trả về chuỗi không có dấu cách bên phải.
7. LEFT*(str,n*): Lấy *n* ký tự phía trái của dãy *str*.
8. RIGHT(*str,n*): Lấy *n* ký tự phía phải của dãy *str*.

9. SUBSTRING(*str, start, n*): Lây *n* ký tự của dãy str kể từ vị trí start trong dãy.

10. REPLACE(str1, str2, str3): thay thế tất cả str2 trong str1 bằng str3.

11. STUFF(*str1, start, n, str2* ): Thay thế n ký tự trong str1 từ vị trí start bằng chuỗi str2.

12. STR( x, len [, Dec]): Chuyển số x thành chuỗi.

## 6.3. Hàm xử lý ngày tháng

1. GETDATE(): Cho ngày tháng năm hiện tại.

2. DAY(*dd*): Cho số thứ tự ngày trong tháng của dd.

3. MONTH(*dd*): Cho số thứ tự tháng trong năm của dd.

4. YEAR(*dd*): Cho năm của biểu thức ngày *dd*.

## 6.4. Hàm chuyển đổi kiểu dữ liệu

1. CAST (biểu_thức AS kiểu_dữ_liệu)

Chuyển đổi giá trị của biểu thức sang kiểu được chỉ định.

2. CONVERT(kiểu_dữ_liệu, biểu_thức [,kiểu_chuyển_đổi])

Hàm có chức năng chuyển đổi giá trị của biểu thức sang kiểu dữ liệu được chỉ định. Tham số *kiểu_chuyển_đổi* là một giá trị số thường được sử dụng khi chuyển đổi giá trị kiểu ngày sang kiểu chuỗi nhằm qui định khuôn dạng dữ liệu được hiển thị và được qui định như sau:

**Kiểu chuyển đổi**

| | |
|---|---|
| 101 | mm/dd/yy |
| 102 | yy.mm.dd |
| 103 | dd/mm/yy |

## B. PHÂN LOẠI BÀI TẬP

### DẠNG 1: HÀM

#### Bài số 1: Viết hàm xếp loại dựa vào điểm

```
CREATE  FUNCTION  XEPLOAI
(@Diem numeric(4,1))  RETURNS  nvarchar(10)
AS
BEGIN
     DECLARE  @xl  nvarchar(10)
     SET @xl=CASE
     WHEN @Diem>=8 THEN N'Giỏi'
     WHEN @Diem>=7 THEN N'Khá'
     WHEN @Diem>=5 THEN N'Trung bình'
     ELSE  N'Yếu'
     END
     RETURN  @xl
END
```

Ứng dụng:

Hiển thị danh sách gồm: MaSV, DiemTBC, Xếp Loại của mỗi sinh viên và xuất ra bảng mới tên là DIEMTBC.

```
SELECT MaSV, SUM(DiemHP*Sodvht)/SUM(Sodvht) AS
DiemTBC,
dbo.XEPLOAI(SUM(DiemHP*Sodvht)/SUM(Sodvht) AS 'Xếp
loại'
IN TO DIEMTBC
FROM DMHOCPHAN INNER JOIN DIEMHP ON
DMHOCPHAN.MaHP=DIEMHP.MaHP
GROUP BY MaSV
```

#### Bài số 2: Viết hàm tách tên từ chuỗi Họ tên

```
CREATE  FUNCTION  TACHTEN(@ht nvarchar(30))
```

```
RETURNS  nvarchar(10)
AS
BEGIN
     DECLARE  @ten varchar(10), @L int, @i int,@j
     int,@kt varchar(10)
     SET @L=LEN(@ht)
     SET @i=1
     WHILE  @i<=@L
     BEGIN
          SET @kt=SUBSTRING(@ht,@i,1)
          IF @kt=''  SET  @j=@i
          SET @i=@i+1
     END
     SET @ten=SUBSTRING(@ht,@j+1,10)
RETURN  @ten
END
```

Ứng dụng

Hiển thị danh sách sinh viên gồm: MaSV, HoTen, MaLop, NgaySinh (dd/mm/yyyy), GioiTinh (Nam, Nữ) được sắp xếp theo thứ tự ưu tiên MaLop, Tên sinh viên.

```
SELECT MaSV, HoTen, MaLop,
CONVERT(varchar(10),NgaySinh,103) AS NgaySinh, CASE
GioiTinh WHEN 1 THEN N'Nam' ELSE N'Nữ' END AS
GioiTinh, YEAR(Ngaysinh) AS Namsinh
FROM  SINHVIEN
ORDER BY MaLop, dbo.TACHTEN(HoTen) ASC
```

**Bài số 3: Viết hàm đọc điểm nguyên ra thành chữ tương ứng**

Lời giải

```
CREATE FUNCTION DOCDIEMNGUYEN(@diem int)
```

```
RETURNS  nvarchar(10)
AS
BEGIN
     DECLARE  @diemChu nvarchar(10)
     SET @diemChu=CASE
     WHEN @diem=1 THEN N'Một'
     WHEN @diem=2 THEN N'Hai'
     WHEN @diem=3 THEN N'Ba'
     WHEN @diem=4 THEN N'Bốn'
     WHEN @diem=5 THEN N'Năm'
     WHEN @diem=6 THEN N'Sáu'
     WHEN @diem=7 THEN N'Bảy'
     WHEN @diem=8 THEN N'Tám'
     WHEN @diem=9 THEN N'Chín'
     WHEN @diem=10 THEN N'Mười'
     END
RETURN  @diemchu
```

- *Cách khác*

```
CREATE  FUNCTION  DOCDIEMNGUYEN(@Diem int)
RETURNS  nvarchar(20)
AS
BEGIN
     DECLARE  @ChuSo nvarchar(60), @KetQua
nvarchar(20)
     SET @ChuSo=N'Không,Một  ,Hai  ,Ba   ,Bốn  ,Năm
,Sáu  ,Bảy  ,Tám  ,Chín ,Mười '
     SET @KetQua=SUBSTRING(@ChuSo, @Diem*6+1,5)
     RETURN  @KetQua
END
```

Chú ý: Đảm bảo mỗi chữ số có độ dài là 5

**Bài số 4: Viết hàm đọc điểm 1 chữ số thập phân ra thành chữ tương ứng**

Lời giải:

```
CREATE FUNCTION DOC_DIEMTP(@Diemtp numeric(4,1))
RETURNS  nvarchar(20)
AS
BEGIN
    DECLARE  @pn tinyint, @ptp tinyint, @kq nvarchar(20)
    SET @pn=FLOOR(@Diemtp)
    SET @ptp=(@Diemtp*10)%10
    SET @kq=dbo.DOCDIEMNGUYEN(@pn)+N' phẩy
    '+dbo.DOCDIEMNGUYEN(@ptp)
RETURN  @kq
END
```

**Ứng dụng:**

Hiển thị danh sách gồm MaSV, HoTen, MaHP, DiemHP, Điểm chữ.

```
SELECT MaSV, HoTen, MaHP, DiemHP,
dbo.DOC_DIEMTP(DiemHP) AS 'Điểm chữ'
FROM SINHVIEN
INNER  JOIN DIEMHP  ON SINHVIEN.MaSV=DIEMHP.MaSV
```

**Bài số 4: Các dạng hàm liên quan đến tính toán trong CSDL**

1. Viết hàm tính điểm trung bình chung của sinh viên có mã chỉ định ở học kỳ bất kỳ.

2. Viết hàm tính tổng số đơn vị học trình của các học phần điểm <5 của sinh viên có mã chỉ định.

3. Viết hàm đếm số sinh viên có điểm HP <5 của học phần chỉ định.

4. Viết hàm đếm số học phần có điểm HP <5 của sinh viên chỉ định.

Lời giải:

1. 
```
CREATE FUNCTION  DIEM_SV(@MaSV varchar(11),@HocKy
char(2))  RETURNS  numeric(4,1)
BEGIN
DECLARE  @DiemTBC numeric(4,1)
IF NOT EXISTS (SELECT * FROM DIEMHP WHERE
MaSV=@MaSV)
    RETURN  0
   ELSE
    SET @DiemTBC=(
    SELECT SUM(DiemHP*Sodvht)/SUM(Sodvht)
    FROM DMHOCPHAN
    INNER JOIN DIEMHP ON DMHOCPHAN.MaHP=DIEMHP.MaHP
    WHERE HocKy=@HocKy AND MaSV=@MaSV)
 RETURN  @DiemTBC
END
```

Thực hiện hàm

```
PRINT dbo.DIEM_SV('001',1)
SELECT MaSV, HoTen, dbo.DIEM_SV(MaSV,1) FROM SINHVIEN
```

2. 
```
CREATE  FUNCTION  HocTrinh_SV(@MaSV varchar(11))
RETURNS  int
BEGIN
DECLARE  @HocTrinh  int
IF NOT EXISTS (SELECT * FROM DIEMHP WHERE MaSV=@MaSV)
RETURN  0
SET @HocTrinh=(SELECT SUM(SoDVHT)
FROM DIEMHP
INNER JOIN DMHOCPHAN ON DIEMHP.MaHP=DMHOCPHAN.MaHP
WHERE DiemHP<5 AND MaSV=@MaSV )
```

```
    RETURN  @HocTrinh
    END
    CREATE  FUNCTION  DEM_SV(@MaHP  varchar(11))  RETURNS
int
    BEGIN
    DECLARE  @SoSinhVien   int
    IF NOT EXISTS (SELECT * FROM DIEMHP WHERE MaHP=@MaHP)
    RETURN  0
       ELSE
    SET @ SoSinhVien =(SELECT COUNT(MaSV)
                       FROM DIEMHP
                        WHERE DiemHP<5 AND MaHP=@MaHP )
       RETURN  @SoSinhVien
       END
```

3.
```
CREATE FUNCTION DEM_HP(@MaSV varchar(11))
    RETURNS  int
    BEGIN
    DECLARE  @SoHocPhan  numeric(4,1)
    IF NOT EXISTS (SELECT * FROM DIEMHP WHERE MaSV=@MaSV)
    RETURN  0
    ELSE
    SET @SoHocPhan =( SELECT COUNT(MaHP)
    FROM DIEMHP
    WHERE DiemHP<5 AND MaSV=@MaSV)
    RETURN  @SoHocPhan
    END
```

**BÀI TẬP TỰ GIẢI:**

1. Viết hàm TACHHODEM dùng để tách họ đệm từ chuỗi Họ tên.

Chẳng hạn: Nguyễn Thị Thuỳ -> Nguyễn Thị

2. Viết hàm TACHHO dùng để tách họ từ chuỗi Họ tên,

Chẳng hạn: Nguyễn Thị Thuỳ -> Nguyễn

3. Viết hàm đọc số có 3 chữ số thành chữ tương ứng

4. Viết hàm đọc số có 12 chữ số thành chữ tương ứng.

5. Viết hàm tính doanh thu cuả năm chỉ định

6. Viết hàm tính doanh thu cuả tháng chỉ định

7. Viết hàm tính doanh thu của khách hàng chỉ định

8. Viết hàm tính tổng số lượng bán được cho từng mặt hàng chỉ định và tháng chỉ định, nếu tháng không nhập vào tức là tính tất cả các tháng.

## DẠNG 2: THỦTỤC

### DẠNG BÀI 1: Tạo thủ tuc cập nhật, bổ sung , xoá dữ liệu.

**Bài số 1:**

Viết chương trình tính điểm trung bình chung theo từng học kỳ với mã lớp chỉ định.

Lời giải

```
CREATE PROCEDURE TinhDTBC
@HocKy char(3),@MaLop varchar(10)
AS
IF NOT EXISTS (SELECT * FROM SINHVIEN WHERE MaLop=@MaLop)
BEGIN
     PRINT N'Lớp này không có sinh viên'
     RETURN  -1
END
SELECT HocKy, SINHVIEN.MaSV,  HoTen,
SUM(DiemHP*Sodvht)/SUM(Sodvht) AS diemTBC
FROM DIEMHP
```

```
INNER JOIN  DMHOCPHAN  ON DIEMHP.MaHP=DMHOCPHAN.MAHP
INNER JOIN  SINHVIEN ON SINHVIEN.MaSV=DIEMHP.MaSV
WHERE MaLop=@MaLop AND HocKy=@HocKy
GROUP BY HocKy, SINHVIEN.MaSV, HoTen
ORDER BY HocKy
GO
```

Thực hiện gọi thủ tục

```
EXEC  TinhDTBC '1', 'CT12'
```

**Bài số 2: Tạo thủ tục bổ sung dữ liệu cho bảng SINHVIEN.**

Lời giải:

```
CREATE PROCEDURE  INSERT_SINHVIEN
@MaSV varchar(11), @HoTen nvarchar(30),  @MaLop
varchar(5), @GioiTinh bit, @NgaySinh  datetime,
@DiaChi  nvarchar(4)
AS
BEGIN
     IF EXISTS(SELECT * FROM SINHVIEN WHERE
MaSV=@MaSV)
     BEGIN
          PRINT N'Sinh viên này đã có, nhập mã khác'
          RETURN  -1
     END
     IF NOT EXISTS (SELECT * FROM DMLOP
                    WHERE MaLop=@MaLop)
     BEGIN
          PRINT N'Lop này không có trong danh mục'
          RETURN  -1
     END
     INSERT INTO  SINHVIEN
```

```
        VALUES (@MaSV, @HoTen,  @MaLop, @GioiTinh,
        @NgaySinh, @DiaChi)
        END
```

Thực hiện gọi thủ tục

```
EXEC INSERT_SINHVIEN
'012',N'Nguyễn Văn Thanh','CT12','True','12/12/1989',N'Quy
Nhơn'
```

**Bài số 3: Tạo thủ tục bổ sung dữ liệu cho bảng DIEMHP.**

```
CREATE PROCEDURE  INSERT_DIEM
@MaSV varchar(11), @MaHP varchar(5), @DiemHP
numeric(4,1)
as
BEGIN
IF NOT EXISTS (SELECT * FROM SINHVIEN
                                WHERE MaSV=@MaSV)
BEGIN
PRINT N'Sinh viên này không tồn tại'
RETURN  -1
END
IF NOT EXISTS (SELECT * FROM DMHOCPHAN
               WHERE MaHP=@MaHP)
BEGIN
PRINT N'Học phần này không có trong danh mục'
RETURN  -1
END
IF  EXISTS (SELECT * FROM DIEMHP
                      WHERE MaHP=@MaHP AND MaSV=@MaSV)
BEGIN
PRINT  N'Sinh viên này đã học học phần này'
RETURN  -1
```

```
END
INSERT INTO  DIEMHP values (@MaSV,@MaHP,@DiemHP)
END
```

Thực hiện gọi thủ tục

```
EXECUTE INSERT_Diem '010','004',8.7
```

**Bài số 4: Viết thủ tục xoá sinh viên có mã chỉ định.**

```
CREATE PROCEDURE DELETE_SINHVIEN
@MaSV varchar(11)
as
BEGIN
     IF NOT EXISTS (SELECT * FROM SINHVIEN
                    WHERE MaSV=@MaSV)
     BEGIN
     PRINT N'Sinh viên này không có'
     RETURN  -1
     END
DELETE FROM SINHVIEN WHERE MaSV=@MaSV
PRINT N'Xoá thành công'
END
```

**Bài số 5: Cập nhật lại điểm có mã chỉ định.**

```
CREATE PROCEDURE UpDate_DIEM
@MaSV varchar(11), @MaHP varchar(5), @DiemHP
numeric(4,1)
as
BEGIN
IF NOT EXISTS (SELECT * FROM SINHVIEN
                 WHERE MaSV=@MaSV)
BEGIN
PRINT N'Sinh viên này không tồn tại'
```

```
RETURN  -1
END
IF NOT EXISTS (SELECT * FROM DMHOCPHAN
               WHERE MaHP=@MaHP)
BEGIN
PRINT N'Học phần này không có trong danh mục'
RETURN  -1
END
UPDATE DIEMHP SET DiemHP=@DiemHP
WHERE MaSV=@MaSV AND MaHP=@maHP
END
Thực hiện gọi thủ tục
EXECUTE UPDATE_Diem '010','004',4
```

Bài số 6: Cập nhật lại dữ liệu cho khoa có mã chỉ định.

```
CREATE PROCEDURE UpDate_KHOA
@MaKhoa varchar(5),@MaKhoaMoi varchar(5), @TenKhoa
nvarchar(30)
AS
BEGIN
    IF NOT EXISTS (SELECT * FROM DMKHOA
                       WHERE MaKhoa=@MaKhoa)
    BEGIN
    PRINT N'Khoa này không tồn tại'
    RETURN  -1
    END
UPDATE DMKHOA SET MaKhoa=@MaKHoaMoi, TenKhoa=@TenKHoa
WHERE MaKhoa=@MaKhoa
END
```

**Bài số 7**: Viết đoạn chương trình phân lớp thành 2 lớp A,B với điều kiện là: Nếu Mã sinh viên là số lẻ thì là lớp A, nếu Mã sinh viên là chẵn thì là lớp B

Lời giải

Thêm cột PhanLop vào bảng SINHVIEN

```
ALTER TABLE SINHVIEN
ADD PhanLop varchar(5)
     CREATE PROCEDURE Phan2lop
     @MaLop varchar(10)
     AS
     IF NOT EXISTS (SELECT * FROM SINHVIEN
                    WHERE MaLop=@MaLop)
     BEGIN
          PRINT N'Lớp này không có sinh viên'
          RETURN  -1
     END
     UPDATE  SINHVIEN  SET  PhanLop=RTRIM(MaLop)+'A'
     WHERE RTRIM(MaSV)%2=1 AND MaLop=@MaLop
     UPDATE SINHVIEN SET PhanLop=RTRIM(MaLop)+'B'
     WHERE RTRIM(MaSV)%2=0 AND MaLop=@MaLop
```

**Bài số 8:** Viết đoạn chương trình phân lớp thành A,B,C,D,.. bất kỳ với số lượng lớp chỉ định. Phân lớp được quy định như sau:

- Nếu Mã sinh viên chia cho số lớp dư 0 là lớp 'A'
- Nếu Mã sinh viên chia cho số lớp dư 1 là lớp 'B'
- ….

Lời giải:

```
CREATE PROCEDURE Phanlop
@solop int, @MaLop varchar(10)
AS
```

```
IF NOT EXISTS (SELECT * FROM SINHVIEN WHERE
MaLop=@MaLop)
BEGIN
     PRINT N'Lớp này không có sinh viên'
     RETURN  -1
END
DECLARE  @chuoi varchar(5),@i int
SET @chuoi='ABCDEFGH'
SET @i=1
WHILE @i<=@solop
BEGIN
     UPDATE SINHVIEN SET
     PhanLop=RTRIM(MaLop)+SUBSTRING(@chuoi,@i,1)
     WHERE RTRIM(MaSV)%@solop=@i-1 AND MaLop=@MaLop
     SET @i=@i+1
END
GO
```

**DẠNG BÀI 2: Tạo thủ tục hiển thị dữ liệu với các điều kiện chỉ định.**

**Bài toán 1**:

Tạo thủ tục: Hiển thị danh sách gồm MaSV, HoTen, , MaLop, DiemHP, MaHP của những sinh viên có DiemHP nhỏ hơn số chỉ định, nếu không có thì hiển thị thông báo không có sinh viên nào.

Lời giải

```
CREATE PROCEDURE HIENTHI_Diem
@DiemHP varchar(5)
AS
IF NOT EXISTS (SELECT * FROM  DIEMHP
                              WHERE DiemHP < @DiemHP)
     PRINT N'không có sinh viên nào'
```

```
ELSE
SELECT SINHVIEN.MaSV, HoTen, MaLop, MaHP, DiemHP
FROM DIEMHP
INNER JOIN SINHVIEN ON DIEMHP.MaSV = SINHVIEN.MaSV
WHERE DIEMHP.DiemHP < @DiemHP
GO
```

- *Thực hiện gọi thử tục*

```
EXCE HIENTHI_Diem 5
```

**Bài toán 2:**

Tạo thủ tục: Hiển thị Hoten sinh viên CHƯA học học phần có mã chỉ định, Kiểm tra Mã học phần chỉ định có trong danh mục không, Nếu không có thì hiển thị thông báo không có học phần này.

Lời giải:

```
CREATE PROCEDURE HIENTHI_MaHP
@MaHP varchar(5)
AS
IF NOT EXISTS (SELECT * FROM DMHOCPHAN
                              WHERE MaHp=@MaHP)
     PRINT N'Không có học phần này'
ELSE
SELECT HoTen FROM SINHVIEN
WHERE MaSV NOT IN (SELECT MaSV FROM DIEMHP
                            WHERE MaHp=@MaHP)
GO
```

Thực hiện thủ tục

```
EXCE HIENTHI_MaHP '001'
```

**Bài toán 3**:

Tạo thủ tục: Hiển thị danh sách gồm: MaSV, HoTen, MaLop, NgaySinh (dd/mm/yyyy), GioiTinh (Nam, Nữ),Tuổi của những sinh viên có tuổi trong khoảng chỉ định. Nếu không có thì hiển thị thông báo không có sinh viên nào

Lời giải:

```
CREATE PROCEDURE HIENTHI_Tuoi
@tuoi1 tinyint, @tuoi2 tinyint
AS
IF NOT EXISTS (SELECT * FROM SINHVIEN WHERE
YEAR(GETDATE())-YEAR(Ngaysinh) BETWEEN @tuoi1 AND
@tuoi2)
     PRINT 'không có sinh viên nào'
ELSE
     SELECT
     MaSV,HoTen,MaLop,CONVERT(char(10),NgaySinh,103),
     CASE  gioitinh WHEN 0 THEN N'Nữ'ELSE N'Nam' END,
     YEAR(GETDATE())-YEAR(Ngaysinh) AS 'Tuổi'
     FROM SINHVIEN
     WHERE YEAR(GETDATE())-YEAR(Ngaysinh)

   BETWEEN @tuoi1 AND @tuoi2
GO
```

- *Thực hiện gọi thử tục*

EXECUTE HIENTHI_tuoi 30,40

**Bài toán 4:**

Tạo thủ tục:Cho biết MaKhoa, Tên Khoa, tổng số sinh viên của Khoa chỉ định. Kiểm tra điều kiện Mã khoa có trong bảng danh mục không.

Lời giải:

```
CREATE PROCEDURE HIENTHI_KHOA
```

```
@MaKhoa varchar(6)
AS
IF NOT EXISTS (SELECT * FROM DMLOP
               WHERE MaKhoa=@MaKhoa)
BEGIN
     PRINT N'Khoa này không có sinh viên nào'
     RETURN  -1
END
SELECT DMKHOA.MaKhoa,TenKhoa,
COUNT(SINHVIEN.MaSV) AS'số lượng'
FROM SINHVIEN
INNER JOIN DMLOP ON DMLOP.MaLop=SINHVIEN.MaLop
INNER JOIN DMNGANH ON DMNGANH.MaNganh=DMLOP.MaNganh
INNER JOIN DMKHOA ON DMKHOA.MaKhoa=DMNGANH.MaKhoa
WHERE DMKHOA.MaKhoa=@MaKhoa
GROUP BY DMKHOA.MaKhoa,TenKhoa
GO
```

**Bài toán 5:**

Tạo thủ tục: Hiển thị MaLop,TenLop, Tổng số SV mỗi lớp của khoa có mã chỉ định, Kiểm tra điều kiện MaKhoa có trong bảng Danh mục không, Nếu không có thì hiển thị thông báo Không có lớp này.

Lời giải

```
CREATE PROCEDURE HIENTHI_KHOA2
@MaKhoa varchar(6)
AS
IF NOT EXISTS (SELECT * FROM DMKHOA
               WHERE MaKhoa=@MaKhoa)
BEGIN
     PRINT N'Không có lớp này'
     RETURN  -1
```

```
END
SELECT DMLOP.MaLop,TenLop, COUNT(MaSV) AS SoLuong
FROM SINHVIEN
INNER JOIN DMLOP ON SINHVIEN.MaLop=DMLOP.MaLop
INNER JOIN DMNGANH ON DMNGANH.MaNganh=DMLOP.MaNganh
WHERE MaKhoa=@MaKhoa
GROUP BY DMLOP.MaLop,TenLop
GO
```

- Thực hiện gọi thủ tục

```
EXECUTE HIENTHI_KHOA2 'CNTT'
```

**Bài toán 6:**

Tạo thủ tục: Tính điểm trung bình chung từng học kỳ theo từng sinh viên của lớp có mã chỉ định.

Lời giải

```
CREATE PROCEDURE HIENTHI_DTBC
@MaLop varchar(5)
AS
SELECT  DIEMHP.MaSV, HocKy,
SUM(DiemHP*SoDvht)/SUM(SoDvht) AS  'Điểm TBC'
FROM DIEMHP
INNER JOIN DMHOCPHAN ON DIEMHP.MaHP=DMHOCPHAN.MaHP
INNER JOIN SINHVIEN ON DIEMHP.MaSV=SINHVIEN.MaSV
WHERE MaLop=@MaLop
GROUP BY DIEMHP.MaSV, HocKy
GO
```

Thực hiện gọi thủ tục

```
EXECUTE HIENTHI_DTBC 'TH11'
```

**Bài toán 7.**

Tạo thủ tục: Hiển thị danh sách gồm: MaSV, HoTen, MaLop, MaKhoa, NgaySinh (dd/mm/yyyy),GioiTinh (Nam, Nữ) của những sinh viên ở Khoa có mã chỉ định, Nếu không có thì hiển thị thông báo Không có sinh viên nào.

Lời giải

```
CREATE PROCEDURE HIENTHI_KHOA3
@MaKhoa varchar(10)
AS
IF NOT EXISTS(SELECT * FROM DMNGANH WHERE MaKhoa=@MaKhoa)
   PRINT N'Khoa này không có trong danh mục'
ELSE
   SELECT MaSV, HoTen, DMLOP.MaLop,
   CASE  GioiTinh WHEN 1 THEN N'Nam' ELSE N'Nữ' END AS
   GioiTinh,
   CONVERT (char(10),ngaysinh,103) AS NgaySinh
   FROM SINHVIEN
   INNER JOIN DMLOP ON SINHVIEN.MaLop=DMLOP.MaLop
   INNER JOIN DMNGANH ON DMNGANH.MaNganh=DMLOP.MaNganh
   WHERE MaKhoa=@MaKhoa
GO
```

- Thực hiện gọi thủ tục

EXECUTE HIENTHI_KHOA3 'CNTT'

**Bài toán 8:**

Tạo thủ tục: Cho biết Hoten sinh viên KHÔNG có điểm HP <5 ở lớp có mã chỉ định, Kiểm tra Mã lớp chỉ định có trong danh mục không, Nếu không thì hiển thị thông báo.

Lời giải

```
CREATE PROCEDURE KIEMTRA_LOP
```

```
@MaLop varchar(6)
AS
IF NOT EXISTS (SELECT * FROM DMLOP WHERE MaLop=@MaLop)
     PRINT N'Lớp này không có trong danh mục'
ELSE
     SELECT MaSV, HoTen FROM SINHVIEN
     WHERE MaLop=@MaLop
     AND MaSV NOT IN(SELECT MaSV   FROM DIEMHP
                                    WHERE DiemHP<5)
GO
```

- Thực hiện gọi thủ tục

```
EXECUTE KIEMTRA_LOP 'CT12'
```

**Bài toán 9:**

Tạo thủ tục: Hiển thị danh sách gồm: MaSV, HoTen, MaLop, NgaySinh (dd/mm/yyyy), GioiTinh(Nam, Nữ), của những sinh viên học lớp có mã chỉ định. Kiểm tra MaLop chỉ định có tồn tại trong bảng không, nếu không có thì hiển thị thông báo Không có lớp đó.

Lời giải

```
CREATE   PROCEDURE GETALL_SINHVIEN
@MaLop varchar(5)
AS
IF NOT EXISTS (SELECT * FROM SINHVIEN
                          WHERE MaLop=@ MaLop)
     PRINT N'Lớp đó không có trong danh mục'
ELSE
     SELECT MaSV, HoTen, MaLop,
     CASE  GioiTinh WHEN 1 THEN N'Nam' ELSE N'Nữ' END,
     CONVERT(char(10), NgaySinh,103)
     FROM SINHVIEN
```

```
        WHERE MaLop=@MaLop
    GO
```

- Thực hiện gọi thủ tục

```
        EXECUTE GETALL_SINHVIEN 'CT12'
```

## BÀI TẬP TỰ GIẢI

1. Hiển thị danh các khách hàng đã mua hàng trong ngày chỉ định (ngày là tham số truyền vào)
2. Hiển thị danh sách 5 khách hàng  có tổng trị giá các đơn hàng lớn nhất.
3. Hiển thị danh sách 10 mặt hàng có số lượng bán lớn nhất.
4. Cập nhật cột Khuyến mãi như sau: Khuyến mãi 5% thành tiền nếu SoLuong >100, 10% thành tiền nếu SoLuongL >500.
5. Tính trị giá cho mỗi hoá đơn.
6. Cập nhật cho cột Loại khách hàng: là VIP nếu tổng thành tiền trong năm lớn hơn hoặc bằng 20 triệu.

## DẠNG 3: CON TRỎ

### Bài số 1: Tạo thủ tục đánh Số báo danh theo từng lớp chỉ định.

Trước hết thêm cột SBD vào bảng SINHVIEN

```
ALTER TABLE SINHVIEN
ADD SBD varchar(4)
```

Lời giải

```
    CREATE PROCEDURE DanhSBD
    @MaLop varchar(5)
    AS
    DECLARE  cur_SBD CURSOR
    FOR SELECT MaSV
```

```
FROM SINHVIEN WHERE MaLop=@MaLop
ORDER BY MaLop, dbo.TACHTEN(HoTen)
OPEN cur_SBD
DECLARE  @MaSV varchar(5),@i int
FETCH  NEXT FROM cur_SBD INTO  @MaSV
SET @i=1
WHILE  @@fetch_status = 0
BEGIN
     UPDATE SINHVIEN SET
SBD=RIGHT('0000'+LTRIM(STR(@i)),4)
     WHERE MaSV=@MaSV
     FETCH NEXT FROM cur_SBD INTO  @MaSV
     SET @i=@i+1
END
CLOSE  cur_SBD
DEALLOCATE cur_SBD
Go
```

**Bài số 2: Tạo thủ tục đánh số báo danh tự động**

khi sang lớp mới thì SBD đánh lại từ đầu.

Chú ý: Lớp gồm những sinh viên cùng ngành học và khoá học (khác với cột PhanLop).

Lời giải

```
CREATE PROCEDURE DanhSBD_Lop
AS
DECLARE  cur_SBD CURSOR
FOR SELECT MaSV, MaLop
FROM SINHVIEN ORDER BY MaLop,dbo.TACHTEN(HoTen)
OPEN cur_SBD
DECLARE  @MaSV varchar(5),@MaLop varchar(5),@i int
```

```
DECLARE  @Lop varchar(5)
FETCH NEXT FROM cur_SBD INTO  @MaSV,@MaLop
WHILE  @@fetch_status = 0
BEGIN
 SET @Lop=@MaLop
 SET @i=1
 WHILE  @MaLop=@Lop AND @@fetch_status = 0
 BEGIN
      UPDATE SINHVIEN
      SET SBD=RIGHT('0000'+LTRIM(STR(@i)),4)
      WHERE MaSV=@MaSV
      FETCH NEXT FROM cur_SBD INTO  @MaSV,@MaLop
      SET @i=@i+1
 END
END
CLOSE  cur_SBD
DEALLOCATE cur_SBD
GO
```

**Bài số 3: Tạo thủ tục cập nhật mã thẻ sinh viên với công thức như sau:**

MaThe = Năm nhập học (2 chữ số) + Mã Ngành (6 chữ số) + Số thứ tự

Lời giải:

```
CREATE PROCEDURE DanhTheSV
@MaNganh varchar(10), @NamNhapHoc char(4)
AS
DECLARE  cur_MaSV CURSOR
FOR SELECT MaSV, MaNganh, NamNhaphoc
FROM SINHVIEN
INNER JOIN DMLOP ON SINHVIEN.MaLop=DMLOP.MaLop
```

```
    WHERE MaNganh=@MaNganh AND NamNhapHoc=@NamNhapHoc
    ORDER BY dbo.TACHTEN(HoTen)
    DECLARE  @MaSV varchar(5), 65  ıt
    FETCH NEXT FROM cur_MaSV INTO
    @MaSV,@MaNganh,@NamNhapHoc
    SET @i=1
    WHILE  @@fetch_status = 0
    BEGIN
    UPDATE SINHVIEN
    SET @MaThe=RIGHT(@NamNhapHoc,2)+RTRIM(@MaNganh)
    +RIGHT('000'+LTRIM(STR(@i)),3)
    WHERE MaSV=@MaSV
    FETCH NEXT FROM cur_MaSV
    INTO  @MaSV, @MaNganh, @NamNhapHoc
    SET @i=@i+1
    END
    CLOSE  cur_MaSV
    DEALLOCATE cur_MaSV
    GO
```

* Thực hiện gọi thủ tục

```
EXECUTE DanhTheSV ‘480202’,2012
```

**Bài số 4: Viết thủ tục phân lớp theo yêu cầu khác nhau**

1. Phân thành 2 lớp A,B với tỉ lệ nam, nữ như nhau, Mã lớp là tham số truyền vào chỉ định.
2. Phân lớp với số lượng lớp chỉ định và tỉ lệ nam, nữ như nhau. Mã lớp và số lượng là 2 tham số truyền vào chỉ định.

Ghi chú: tạo thêm cột PhanLop varchar(5)

```
ALTER TABLE SINHVIEN
ADD PhanLop varchar(5)
```

Lời giải 1:

```
CREATE PROCEDURE PhanLop
@MaLop varchar(6)
AS
DECLARE  cur_phanlop CURSOR
FOR SELECT MaSV FROM SINHVIEN WHERE MaLop=@MaLop
 ORDER BY MaLop,GioiTinh, dbo.TACHTEN(HoTen)
OPEN  cur_phanlop
DECLARE  @i int
FETCH NEXT FROM cur_phanlop
SET @i=1
WHILE  @@FETCH_STATUS=0
BEGIN
     IF  @i % 2=0
          UPDATE SINHVIEN SET
          PhanLop=LTRIM(@MaLop)+'A' WHERE CURRENT OF
          cur_phanlop
     ELSE
          UPDATE SINHVIEN SET
          PhanLop=LTRIM(@MaLop)+'B' WHERE CURRENT OF
          cur_phanlop
     SET @i=@i+1
     FETCH NEXT FROM cur_phanlop
END
CLOSE cur_phanlop
DEALLOCATE cur_phanlop
```

Lời giải 2:

```
CREATE PROCEDURE PhanLop3
@MaLop varchar(6),@SoLop int
AS
```

```
DECLARE  cur_phanlop CURSOR FOR
SELECT MaSV FROM SINHVIEN WHERE MaLop=@MaLop
ORDER BY MaLop, GioiTinh, dbo.TACHTEN(HoTen)
OPEN  cur_phanlop
DECLARE  @i int, @chuoi char(5), @j int
SET @chuoi='ABCDEF'
FETCH NEXT FROM cur_phanlop
SET @i=1
WHILE  @@FETCH_STATUS=0
BEGIN
     SET @j=@i % @SoLop
     BEGIN
         UPDATE SINHVIEN SET
         PhanLop=LTRIM(@MaLop)+SUBSTRING(@chuoi,@j+1,1)
         WHERE CURRENT OF cur_phanlop
         SET @i=@i+1
         FETCH NEXT FROM  cur_phanlop
    END
END
CLOSE cur_phanlop
DEALLOCATE cur_phanlop
```

- Thực hiện gọi thủ tục

```
EXECUTE PhanLop3 'CT12',3
```

# CHƯƠNG 4: MỘT SỐ ĐỐI TƯỢNG TIỆN ÍCH KHÁC

## A. KIẾN THỨC CẦN NHỚ

### 1. TRANSACTION

Transaction (Giao tác) là một tập hợp có thứ tự các thao tác và chúng chỉ có thể cùng nhau thành công hoặc cùng nhau thất bại.

Nghĩa là: nếu có một thao tác không hoàn thành được thì toàn bộ giao tác cũng không hoàn thành.

**Cú pháp**

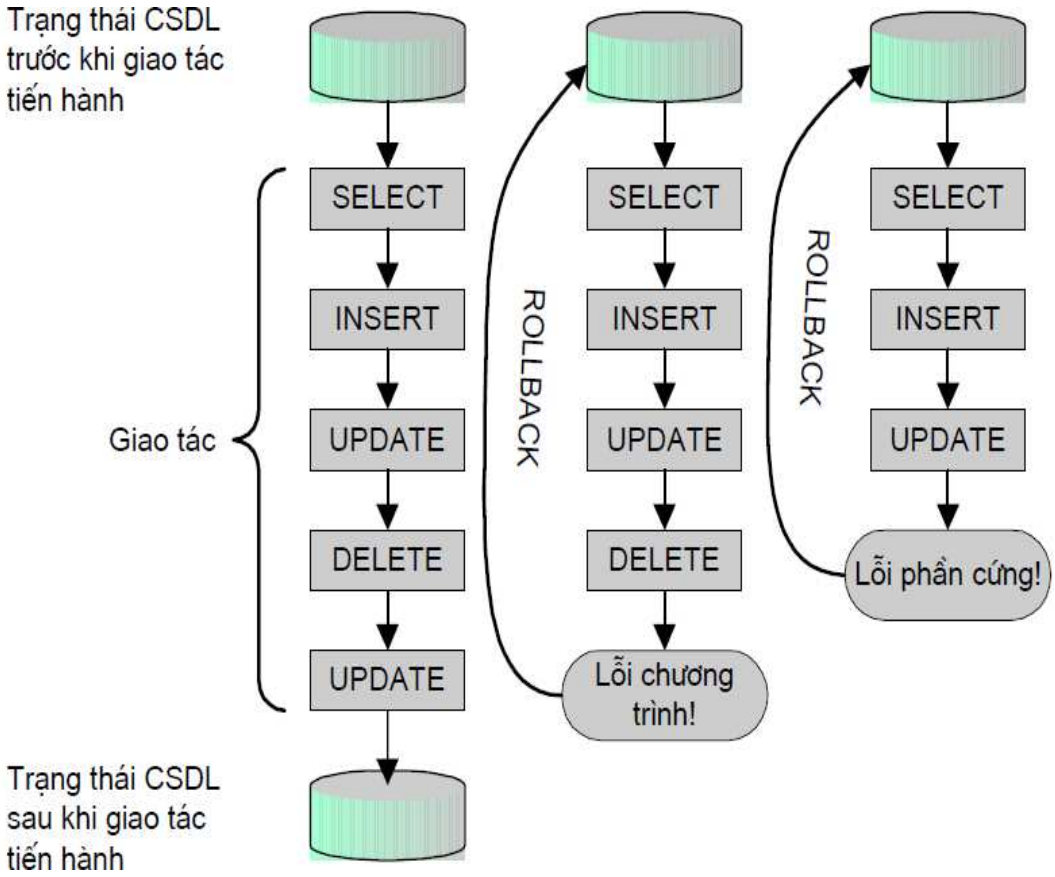
BEGIN [TRANSACTION]

<nhóm lệnh>

ROLLBACK [TRANSACTION/TRAN]

| COMMIT [TRANSACTION/TRAN]

- Sơ đồ hoạt động của giao tác:



## 2. TRIGGER

Trigger là một loại stored Procedure đặc biệt có đặc điểm sau:

- Tự động thực hiện khi có lệnh **INSERT, DELETE hoặc UPDATE** trên dữ liệu

- Thường dùng để kiểm tra các **ràng buộc toàn vẹn** của CSDL hoặc **các qui tắc** nghiệp vụ.

- Một TRIGGER được định nghĩa trên một bảng, nhưng các xử lý trong TRIGGER có thể sử dụng nhiều bảng khác.

**Xoá TRIGGER: DROP TRIGGER <Tên TRIGGER>**

# B. PHÂN LOẠI BÀI TẬP

## DẠNG 1: Tạo bẫy lỗi INSERT

**Bài số 1**: Tạo một Trigger để kiểm tra tính hợp lệ của dữ liệu được nhập vào một bảng SINHVIEN là dữ liệu MaSV là không rỗng.

Lời giải

```
CREATE TRIGGER  INSERTSINHVIEN
      ON SINHVIEN
      FOR INSERT
      AS
      IF ((SELECT MaSV FROM INSERTED) = '')
      BEGIN
      PRINT N'Mã sinh viên phải được nhập'
      ROLLBACK TRANSACTION
        END
```

**Bài số 2**: Thực hiện việc kiểm tra rằng buộc khoá ngoại trong bảng SINHVIEN là mã lớp phải tồn tại trong bảng DMLOP.

Lời giải

```
CREATE  TRIGGER  SV_INSERT
 ON SINHVIEN
 FOR  INSERT
 AS
 IF NOT EXISTS(SELECT * FROM DMLOP, INSERTED
              WHERE  DMLOP.MaLop=INSERTED.MaLop)
 BEGIN
      PRINT N'Mã lớp không có trong danh mục'
      ROLLBACK TRANSACTION
 END
```

**Bài số 3:**

Tạo một Trigger khi thêm một sinh viên trong bảng SINHVIEN ở một lớp nào đó thì cột Siso của lớp đó trong bảng DMLOP tự động tăng lên 1. đảm bảo tính toàn ven dữ liệu khi thêm một sinh viên mới trong bảng SINHVIEN thì sinh viên đó phải có mã lớp trong bảng DMLOP. Đảm bảo tính toàn vẹn dữ liệu khi thêm là mã lớp phải có trong bảng DMLOP.

Lời giải:

Trước hết tạo thêm một cột SiSo

```
ALTER DMLOP
ADD SiSo int
```

Tầm ảnh hưởng

| | **Thêm** | **Xóa** | **Sửa** |
|---|---|---|---|
| **SINHVIEN** | + | | |
| **DMLOP** | | | + (Siso) |

```
CREATE  TRIGGER Trg_SVINSERT
 ON SINHVIEN
```

```
FOR INSERT
AS
IF NOT EXISTS(SELECT * FROM DMLOP, INSERTED
    WHERE  DMLOP.MaLop=INSERTED.MaLop)
    ROLLBACK TRANSACTION
    ELSE
UPDATE DMLOP SET DMLOP.Siso=DMLOP.Siso+1
FROM INSERTED
WHERE DMLOP.MaLop=INSERTED.MaLop
```

Áp dụng

```
INSERT INTO  SINHVIEN(MaSV, Hoten, MaLop)
VALUES ('020, N'Nguyễn Văn Anh', 'CT11')
```

1. Khi lệnh INSERT thực thi thì một TRIGGER FOR INSERT sẽ tự động thực hiện.
2. Dữ liệu Sinh viên 020 bổ sung thêm sẽ được đưa vào bảng tạm thời INSERTED.
3. Thực hiện nhóm lệnh cập nhật Siso của lớp vừa thêm trong bảng DMLOP.

**DẠNG 2: Bẫy lỗi DELETE**

**Bài số 1**: Tạo một Trigger không cho phép xoá các sinh viên ở lớp CT12.

```
CREATE TRIGGER DELETESV2
ON SINHVIEN
FOR DELETE
AS
IF EXISTS(SELECT * FROM DELETED WHERE MaLop='CT12')
BEGIN
```

```
        PRINT  N'Bạn không thể xoá sv lớp CT12'
        ROLLBACK TRANSACTION
    END
```

Bài số 2: Tạo một Trigger không cho phép xoá nhiều hơn 2 lớp trong bảng DMLOP

Lời giải:

```
CREATE TRIGGER DELETE_Lop
    ON DMLOP
    FOR DELETE
    AS
    IF ((SELECT COUNT (*) FROM DELETED) > 2)
    BEGIN
          PRINT  N'Bạn không thể xoá hơn 2 lớp'
          ROLLBACK TRANSACTION
    END
```

**Bài số 3**: Tạo một Trigger sao cho khi xóa một sinh viên mới từ bảng SINHVIEN thì SiSo của lớp tương ứng trong bảng DMLOP tự động giảm xuống 1.

Lời giải

Tầm ảnh hưởng

| | Thêm | Xóa | Sửa |
|---|---|---|---|
| SINHVIEN | | + | |
| DMLOP | | | + (Siso) |

```
CREATE  TRIGGER Trg_SV_DELETE
    ON SINHVIEN
    FOR DELETE
    AS
```

```
UPDATE DMLOP
SET DMLOP.Siso=DMLOP.Siso-1
FROM DELETED
WHERE DMLOP.MaLop=DELETED.MaLop
```

Áp dụng

```
DELETE FROM SINHVIEN WHERE MaSV='001'
```

1. Khi lệnh DELETE thực thi thì một TRIGGER FOR DELETE sẽ tự động thực hiện:
2. Dữ liệu sinh viên 001 sẽ được đưa vào bảng tạm thời DELETED.
3. Thực hiện nhóm lệnh cập nhật Siso của lớp vừa xóa trong bảng DMLOP

**DẠNG 3: Bẫy lỗi UPDATE**

**Bài số 1:** Tạo một Trigger kiểm tra điều kiện cho cột Điểm là <=10

Lời giải

```
CREATE TRIGGER DiemUPDATE
    ON DIEMHP
    FOR UPDATE
    AS
    IF ((SELECT DiemHP FROM INSERTED) > 10)
    BEGIN
        PRINT  N'Nhập lại điểm <=10'
        ROLLBACK TRANSACTION
    END
```

**Bài số 2:** Tạo Trigger bẫy lỗi cho khoá ngoại của bảng SINHVIEN khi chỉnh sửa.

Lời giải:

```
CREATE TRIGGER LOP_UPDATE
ON SINHVIEN
```

```
FOR UPDATE
AS
IF UPDATE(MaLop)
BEGIN
     IF NOT EXISTS(SELECT * FROM DMLOP, INSERTED
                    WHERE  DMLOP.MaLop=INSERTED.MaLop)
     PRINT N'Mã lớp không có trong danh mục'
      ROLLBACK TRANSACTION
END
```

**Bài số 3:**

Tạo ra Trigger sao cho khi cập nhật MaLop một sinh viên trong bảng SINHVIEN thì SiSo của lớp tương ứng trong bảng DMLOP tự động thay đổi.

Tầm ảnh hưởng

| | **Thêm (INSERT)** | **Xóa (DELETE)** | **Sửa (UPDATE)** |
|---|---|---|---|
| **SINHVIEN** | | | +(MaLop) |
| **DMLOP** | | | + (Siso) |

Thêm cột SiSo cho bảng DMLOP

```
ALTER TABLE DMLOP
ADD SiSo int
```

Lời giải:

```
CREATE  TRIGGER Trg_SINHVIEN_UPDATE
     ON SINHVIEN
     FOR UPDATE
     AS
     IF UPDATE(MaLop)
     BEGIN
          IF NOT EXISTS(SELECT * FROM DMLOP, INSERTED
```

```
                    WHERE DMLOP.MaLop=INSERTED.MaLop)
        ROLLBACK TRANSACTION
    ELSE
    BEGIN
        UPDATE DMLOP
        SET DMLOP.Siso=DMLOP.Siso-1
        FROM DELETED
        WHERE DMLOP.MaLop=DELETED.MaLop
        UPDATE DMLOP
        SET DMLOP.Siso=DMLOP.Siso+1
        FROM INSERTED
        WHERE DMLOP.MaLop=INSERTED.MaLop
    END
```

Thực hiện thay đổi mã lớp của sinh viên mã 001 là CT12 (dữ liệu cũ là CT11)

```
UPDATE SINHVIEN SET MaLop='CT12'WHERE MaSV='001'
```

Khi lệnh UPDATE thực thi thì một TRIGGER FOR UPDATE sẽ tự động thực hiện

- Đưa dữ liệu sinh viên 001 lớp CT12 vào bảng INSERTED.
- Đưa dữ liệu sinh viên 001 lớp cũ CT11 vào bảng DELETED.
- Lệnh cập nhật Siso của lớp vừa cập nhật trong bảng DMLOP.

**Bài số 4**: Hãy tạo ra Trigger sao cho khi sửa MaLop **những** sinh viên trong bảng SINHVIEN thì SiSo của lớp tương ứng trong bảng DMLOP tự động thay đổi.

Lời giải:

```
CREATE TRIGGER trg_SINHVIEN_update_Siso
ON SINHVIEN
FOR UPDATE
AS
```

```
IF UPDATE(MaLop)
BEGIN
 IF NOT EXISTS(SELECT * FROM DMLOP,INSERTED
 WHERE  DMLOP.MaLop=INSERTED.MaLop)
 ROLLBACK TRANSACTION
ELSE
     BEGIN
     UPDATE DMLOP
     SET Siso = Siso - (SELECT COUNT(DELETED.Masv)
     FROM DELETED)
     WHERE MaLop IN (SELECT MaLop FROM DELETED)
     UPDATE DMLOP
     SET Siso = Siso + (SELECT COUNT(INSERTED.Masv)
     FROM INSERTED)
     WHERE MaLop IN (SELECT MaLop FROM INSERTED)
     END
END
```

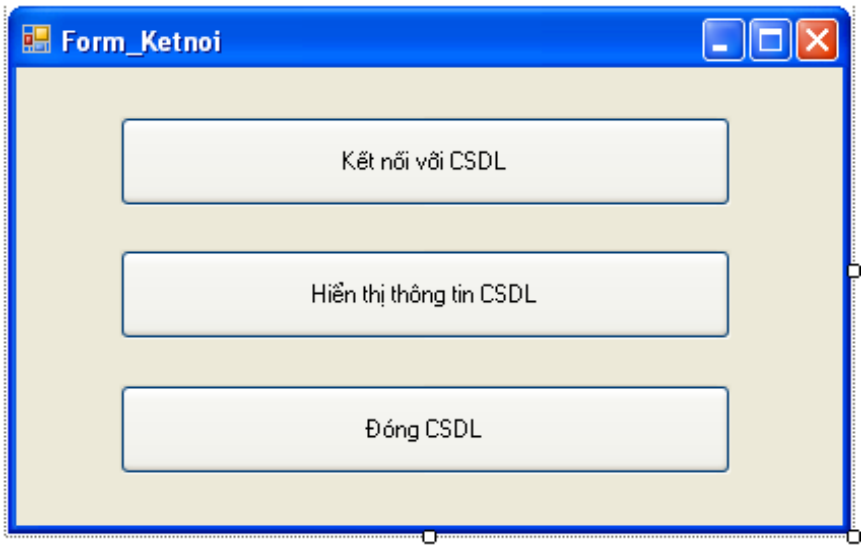
## BÀI TẬP TỰ GIẢI

1. Thực hiện việc kiểm tra rằng buộc khoá ngoại ở bảng HOADON và CHITIETHD.
2. Không cho phép cascade delete trong các rằng buộc khoá ngoại. Ví dụ không cho phép xoá các CTHOADON nào có SOHD còn trong bảng HOADON.
3. Không cho phép user nhập vào hai mặt hàng có cùng tên.
4. Khi user đặt hàng thì KHUYENMAI là 5% nếu SL >100, 10% nếu SL >500.
5. Không cho phép user xoá một lúc nhiều hơn một mặt hàng.
6. Chỉ bán mặt hàng BÁNH với số lượng là bội số của 10.

## PHẦN ĐỌC THÊM

# ỨNG DỤNG SQL TRONG LẬP TRÌNH C# CƠ BẢN

## Bài số 1. Tạo Form kết nối




### Code

```
namespace Quanly_diemsv
{
    public partial class Form_Ketnoi : Form
    {
       public Form_Ketnoi()
        {
            InitializeComponent();
        }
     private static SqlConnection conn; // Khai báo biến conn là lớp kết nối SqlConnection
     private static String ConnectString = "Data Source=MOBI-E2D6A25F65;Initial Catalog=QLSV;Integrated Security=True"; // Khai báo biến ConnectString là kiểu String nhận chỗi két nối
    private void button1_Click(object sender, EventArgs e)
      {
        try
```

```
            {
                conn = new SqlConnection(ConnectString);
// Khởi tạo biến conn với đối tượng kết nối CSDL
                conn.Open(); // Mở kết nối
                MessageBox.Show("Kết nối thành công");
                button2.Enabled = true; // Nút button2
sáng lên
                button3.Enabled = true; // Nút button3
sáng lên

            }
            catch (Exception ex)
            {
              MessageBox.Show("Kết nối thất bại");
            }
        }
    private void button2_Click(object sender, EventArgs e)
        {
            String strStatus = "Closed"; // Định nghĩa
biến strStatus kiểu String nhận chuỗi 'Closed'
            if          (conn.State==ConnectionState.Open)
strStatus="Openned";
            MessageBox.Show("Thông tin kết nối hiện tại
là" + strStatus + "     Tên Server là:       " +
conn.DataSource);
        }
    private void button3_Click(object sender, EventArgs e)
        {
            conn.Close();
            MessageBox.Show("CSDL SQL đóng");
        }
```

```
        }
    }
```

## Bài số 2: Tạo Form hiển thị danh sách sinh viên

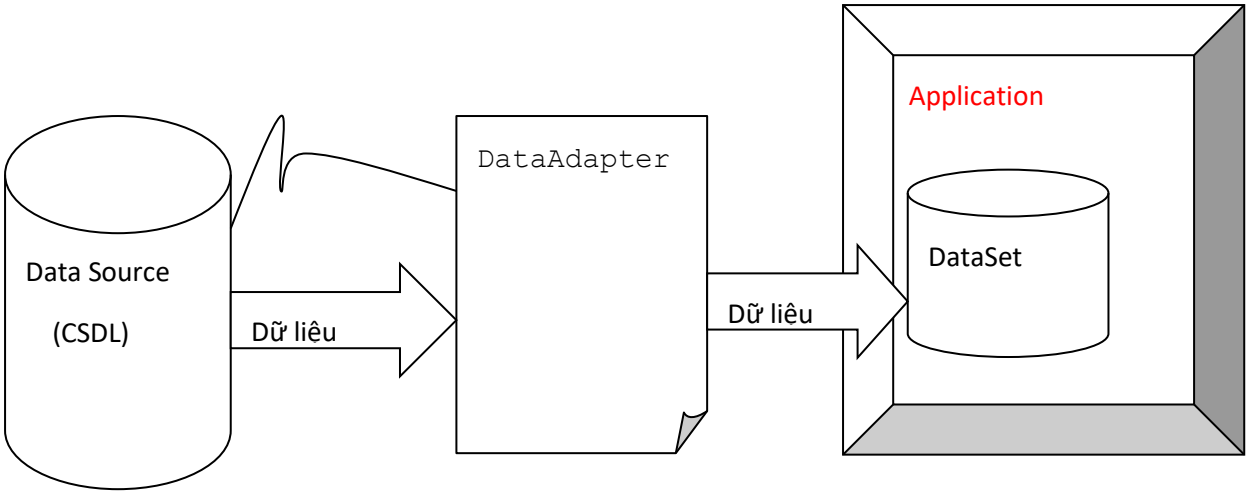
### Các lớp sử dụng trong đoạn chương trình

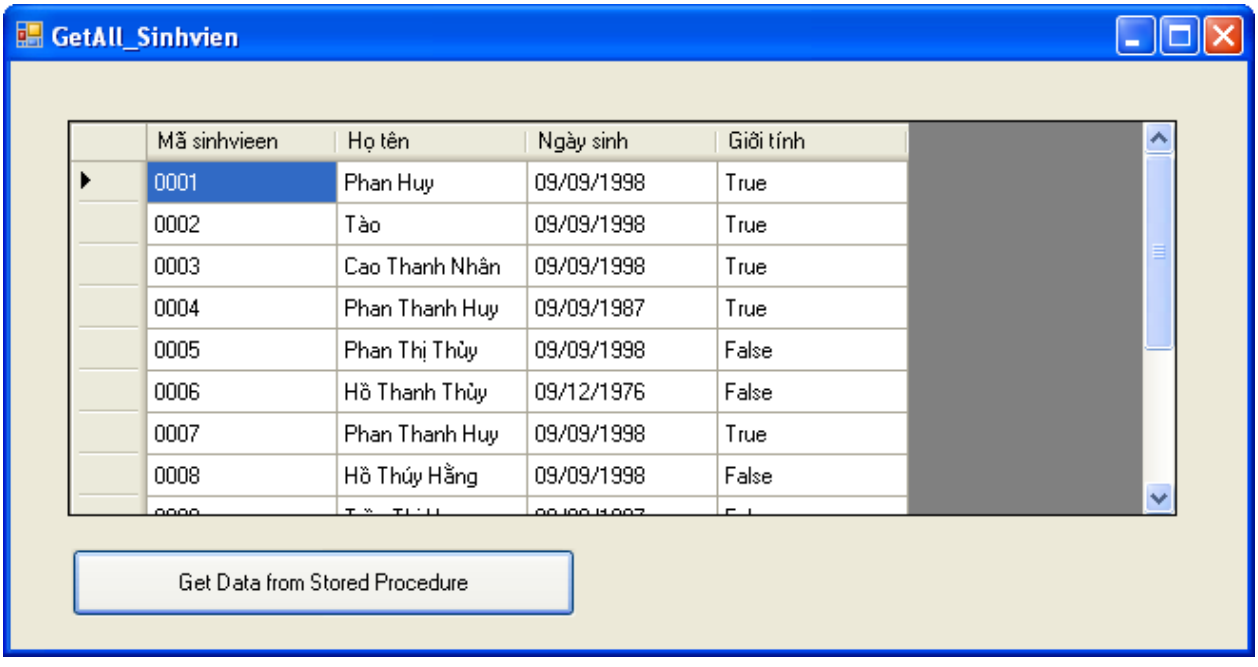
Connection: Lớp kết nối CSDL

SqlCommand: Lớp chứa lệnh SQL

DataAdapter: Lớp chứa dữ liệu có kết nối

DataSet: Lớp chứa dữ liệu không kết nối



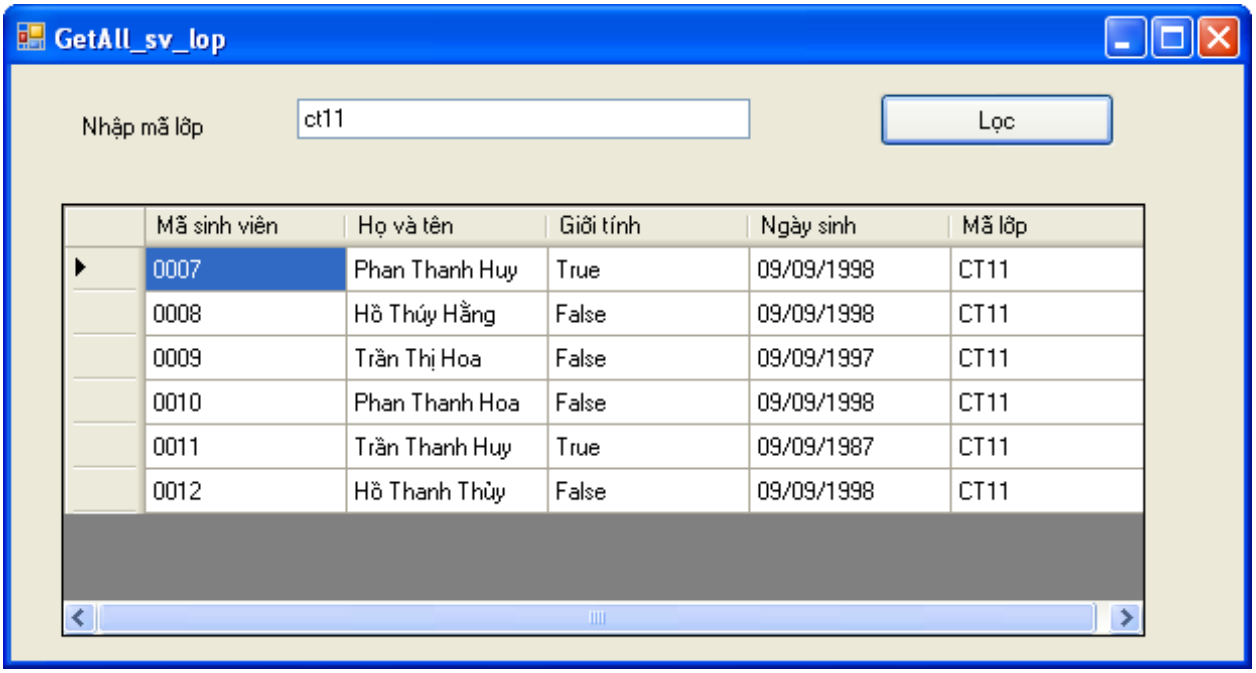



CODE

```
    SqlConnection conn = new SqlConnection("Data
Source=MOBI-E2D6A25F65;Initial  Catalog=QLSV;Integrated
Security=True"); // Khởi tạo biến conn với đối tượng kết
nối
        public GetAll_sv_lop()
        {
            InitializeComponent();
        }
      private void btnGetData_Click(object sender,
EventArgs e)
        {
            try
            {
              conn.Open(); // Mở kết nối
              SqlCommand cmd = new SqlCommand(
"GetALL_Sinhvien",conn);
//khởi tạo biến cmd với đối tượng SqlCommand để thực thi
thụ tục tên usp_GetALL_Sinhvien trong CSDL
              Cmd.CommandType =
CommandType.StoredProcedure;
// xác định thuộc tính CommandType của đối tượng cmm là
kiểu thủ tục
    SqlDataAdapter da = new SqlDataAdapter(cmd);
// Khởi tạo biến da thuộc lớp SqlDataAdapter nhận dữ liệu
từ đồi tượng cmd (sau đó chuyển vào Dataset)
    DataSet ds = new DataSet();
//  Khởi tạo biến ds thuộc lớp DataSet là CSDL không kết
nối với SQL
         da.Fill(ds, "SV");
// Thêm một bảng ảo 'SV' chứa dữ liệu của đối tượng da vào
Dataset ds
```

```
            dataGridView1.DataSource = ds.Tables["SV"];
// Đưa dữ liệu trong bảng SV lên đối tượng dataGridView1
trong Form
            conn.Close(); // Đóng kết nối
                }
                catch (Exception ex)
                {
                    throw new
Exception(ex.Message.ToString());
                }
            }
        }
```

**Bài số 3: Tạo Form Lọc danh sách sinh viên theo lớp**

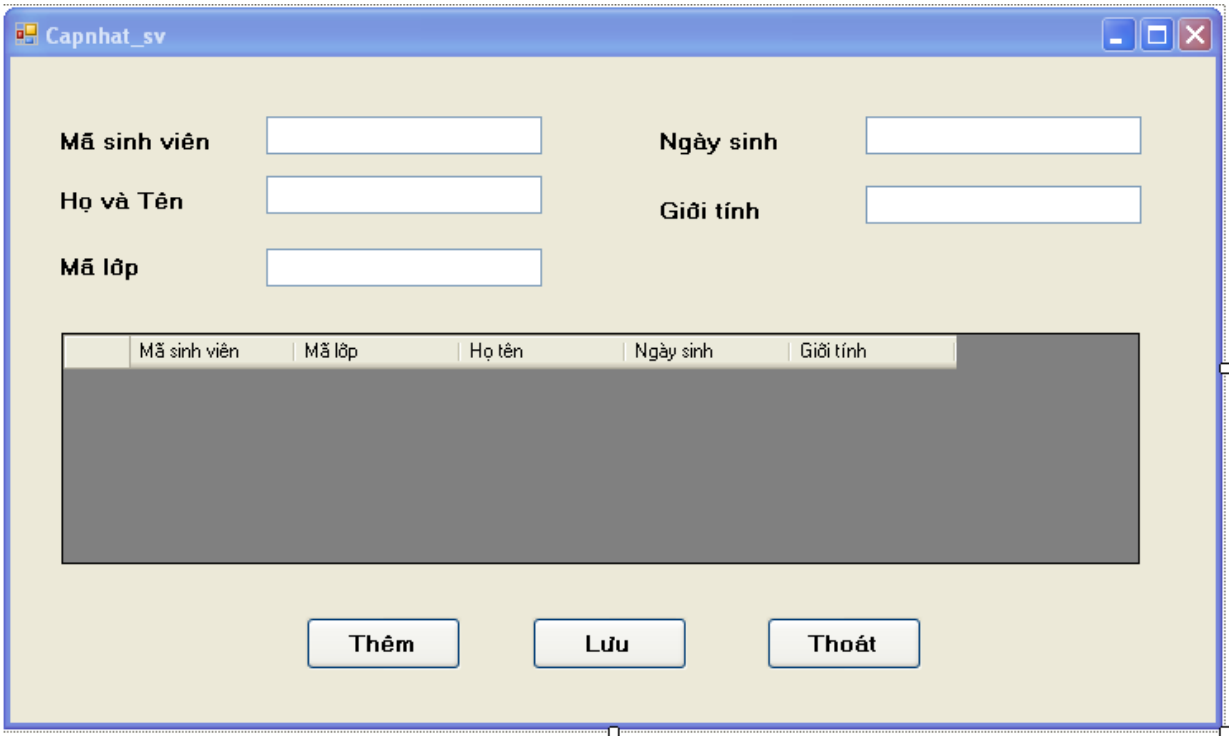



CODE:

```
SqlConnection conn = new SqlConnection("Data Source=MOBI-
E2D6A25F65;Initial Catalog=QLSV;Integrated
Security=True");
     public GetAll_sv_lop()
        {
```

```
            InitializeComponent();
        }
    private void button1_Click(object sender, EventArgs e)
        {
           conn.Open();
            try
            {
       SqlCommand cmd = new SqlCommand("GETALL_SinhVien",
conn);
                // Khởi tạo biến cmd với đối tượng
SqlCommand để thực thi thủ tục ten usp_GETALL_SinhVien
trong csdl
         cmd.CommandType = CommandType.StoredProcedure;
          cmd.Parameters.Add(new  SqlParameter("@maLop",
txtmalop.Text.Trim()));
// Khởi tạo biến @MaLop nhận giá trị trên Text và truyền
vào tham số của đối tượng cmd
                SqlDataAdapter da = new
SqlDataAdapter(cmd);
                DataSet ds = new DataSet();
                da.Fill(ds, "SV");
                dataGridView1.DataSource =
ds.Tables["SV"];
                conn.Close();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            conn.Close();
           }
        }
```

## Bài số 4: Tạo Form nhập dữ liệu cho bảng SINHVIEN




**CODE:**

```
public partial class Capnhat_sv : Form
    {
        SqlConnection conn = new SqlConnection("Data
Source=MOBI-E2D6A25F65;Initial Catalog=QLSV;Integrated
Security=True");
     public Capnhat_sv()
        {
            InitializeComponent();
        }
    private void txtThem_Click(object sender, EventArgs e)
        {
            txtMaSV.Text = " ";
            txtMaLop.Text = " ";
            txtHoTen.Text = " ";
            txtNgaysinh.Text = " ";
            txtGioitinh.Text = " ";
```

```
            txtMaSV.Focus();
        }
     private void txtLuu_Click(object sender, EventArgs e)
        {
            try
            {
                conn.Open();
                SqlCommand cmd = new SqlCommand
("usp_Insert_SinhVien", conn);// ten proc trong csdl
                cmd.CommandType =
CommandType.StoredProcedure;
                SqlParameter TBao = new SqlParameter();
                TBao.ParameterName = "@tbloi";
                TBao.SqlDbType = SqlDbType.NVarChar;
                TBao.Direction =
ParameterDirection.Output;
               cmd.Parameters.Add(TBao);
               cmd.Parameters.Add(new
SqlParameter("@MaSV", txtMaSV.Text.Trim()));
                cmd.Parameters.Add(new
SqlParameter("@MaLop", txtMaLop.Text.Trim()));
                cmd.Parameters.Add(new
SqlParameter("@HoTen", txtHoTen.Text.Trim()));
                cmd.Parameters.Add(new
SqlParameter("@Ngaysinh", txtNgaysinh.Text.Trim()));
                cmd.Parameters.Add(new
SqlParameter("@Gioitinh", txtGioitinh.Text.Trim()));
                cmd.ExecuteNonQuery();
                MessageBox.Show("Lỗi:  " + TBao.Value);
                cmd.Dispose();
                conn.Close();
                RefreshDataGrid();
            }
```

```
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        private void RefreshDataGrid()
        {
            try
            {
                conn.Open();
                SqlCommand cmd = new
SqlCommand("usp_GetALL_Sinhvien", conn);
                cmd.CommandType =
CommandType.StoredProcedure;
                SqlDataAdapter da = new
SqlDataAdapter(cmd);
                DataSet ds = new DataSet();
                da.Fill(ds, "SV");
                dataGridView1.DataSource =
ds.Tables["SV"];
                conn.Close();
            }
            catch (Exception ex)
            {
                throw new
Exception(ex.Message.ToString());
            }
        }
        private void Capnhat_sv_Load(object sender,
EventArgs e)
        {
            RefreshDataGrid();
        }}
```

## *Tài liệu tham khảo*

1. Giáo trình hệ quản trị cơ sở dữ liệu SQL Server - *Trần Nguyên Phong* - Khoa CNTT - Đại học Huế.

2.Tìm hiểu C# và một số ứng dụng – *Phạm Văn Việt, Trương Lập Vỹ* - Khoa CNTT - Đại học khoa học Tự nhiên.