

Ngôn ngữ SQL – Phần 1

Trọng tâm

- 1) Tạo bảng
- 2) Thay đổi cấu trúc bảng
- 3) Nhập, cập nhật dữ liệu

Cú pháp câu lệnh tạo bảng và khóa chính

Tạo bảng không có khóa chính

Cú pháp	Ví dụ
<pre>Create table [Tên Bảng] ([Thuộc tính 1] [Kiểu dữ liệu 1], [Thuộc tính 2] [Kiểu dữ liệu 2], [Thuộc tính 2] [Kiểu dữ liệu 3])</pre>	<pre>Create table NHAN_VIEN (manv char(10), hoten varchar(30), ngaysinh datetime, luong float)</pre>

Tạo bảng có khóa chính

Cú pháp	Ví dụ
<pre>Create table [Tên Bảng] ([Thuộc tính 1] [Kiểu dữ liệu 1], [Thuộc tính 2] [Kiểu dữ liệu 2], ... [Thuộc tính n] [Kiểu dữ liệu n], PRIMARY KEY ([Danh sách các thuộc tính làm khóa chính, cách nhau</pre>	<pre>Create table NHAN_VIEN (manv char(10), hoten varchar(30), ngaysinh datetime, luong float, PRIMAMRY KEY (manv)</pre>

bằng dấu phẩy)])
)	

Tạo bảng kèm theo khóa chính và khóa ngoại

<pre>Create table [Tên Bảng] ([Thuộc tính 1] [Kiểu dữ liệu 1], [Thuộc tính 2] [Kiểu dữ liệu 2], ... [Thuộc tính n] [Kiểu dữ liệu n], CONSTRAINT [Tên Khóa Chính] PRIMARY KEY ([Danh sách các thuộc tính làm khóa chính, cách nhau bằng dấu phẩy]) CONSTRAINT [Tên Khóa Ngoại] FOREIGN KEY ([Danh sách các thuộc tính khóa ngoại]) REFERENCES [Tên bảng tham chiếu]([Danh sách các thuộc tính khóa chính của bảng tham chiếu đến]))</pre>	<pre>Create table PHONG (Mapng char(10), Tenphg varchar(30), Diadiem varchar(40), Trphg char(10), CONSTRAINT PK_NHANVIEN PRIMARY KEY (mapng) CONSTRAINT FK_PHONG_NHANVIEN FOREIGN KEY (Trphg) REFERENCES NHAN_VIEN (Manv))</pre>
--	--

Các lệnh về thay đổi cấu trúc bảng

- Thêm, xóa, sửa một thuộc tính
- Thêm, xóa khóa chính
- Thêm, xóa khóa ngoại
- Thêm, xóa ràng buộc miền giá trị
- Thêm, xóa ràng buộc UNIQUE

Thêm, xóa, sửa một thuộc tính

Cú pháp	Ví dụ
---------	-------

<u>Thêm thuộc tính :</u> <code>Alter table</code> [Tên Bảng] <code>add</code> [Tên thuộc tính] [Kiểu dữ liệu]	<u>Thêm thuộc tính ĐịaChí</u> <code>Alter table</code> NhanVien <code>add</code> DiaChi <code>varchar</code> (20)
<u>Xóa thuộc tính :</u> <code>Alter table</code> [Tên Bảng] <code>drop column</code> [Tên thuộc tính]	<u>Xoá thuộc tính ĐịaChí</u> <code>Alter table</code> NhanVien <code>Drop</code> DiaChi
<u>Sửa thuộc tính :</u> <code>Alter table</code> [Tên Bảng] <code>alter column</code> [Tên thuộc tính] [Kiểu dữ liệu mới]	<u>Sửa thuộc tính ĐịaChí</u> <code>Alter table</code> NhanVien <code>Alter column</code> DiaChi <code>varchar</code> (50)

Thêm ràng buộc khóa chính, khóa ngoại, miền giá trị

Cú pháp	Ví dụ
<u>Thêm khóa chính :</u> <code>Alter table</code> [Tên Bảng] <code>add constraint</code> [Tên khóa chính] <code>PRIMARY KEY</code> ([Danh sách các thuộc tính của khóa chính]) <i>Lưu ý : Khi tạo khóa chính cho bảng ở bên ngoài lệnh tạo bảng thì các thuộc tính của khóa chính phải được khai báo là NOT NULL trong câu lệnh tạo bảng</i>	<u>Thêm khoá chính cho bảng nhân viên:</u> <code>Alter table</code> NhanVien <code>add constraint</code> PK_NHANVIEN <code>PRIMARY KEY</code> (MaNV)
<u>Thêm khóa ngoại :</u> <code>ALTER TABLE</code> [Tên Bảng] <code>ADD CONSTRAINT</code> [Tên khóa ngoại] <code>FOREIGN KEY</code> ([Danh sách các thuộc tính khoá ngoại]) <code>REFERENCES</code> [Tên bảng tham chiếu] (<u>Thêm khóa ngoại cho bảng PHÒNG:</u> <code>ALTER TABLE</code> PHONG <code>ADD CONSTRAINT</code> FK_PHONG_NHANVIEN <code>FOREIGN KEY</code> (trphg) <code>REFERENCES</code> NHANVIEN (manv)

[Danh sách các thuộc tính khoá chính của bảng tham chiếu tới])	
<u>Thêm ràng buộc miền giá trị</u> <code>ALTER TABLE</code> [Tên Bảng] <code>ADD CONSTRAINT</code> [Tên ràng buộc miền gt] <code>CHECK</code> ([Biểu thức điều kiện])	<u>Thêm ràng buộc phải thuộc Nam hoặc Nữ</u> <code>ALTER TABLE</code> NHANVIEN <code>ADD CONSTRAINT</code> C_PHAI <code>CHECK</code> (PHAI <code>IN</code> ('Nam', 'Nữ'))

Xóa ràng buộc khóa chính, khóa ngoại, miền giá trị

<code>Alter table</code> [Tên Bảng] <code>drop constraint</code> [Tên ràng buộc]	<u>Xóa khóa chính</u> <code>Alter table</code> NHANVIEN <code>drop constraint</code> PK_NHANVIEN <u>Xóa khóa ngoại</u> <code>Alter table</code> PHONG <code>drop constraint</code> FK_PHONG_NHANVIEN
--	---

Một số lưu ý :

- Tên khoá chính, khóa ngoại chỉ mang tính gợi nhớ.
- Danh sách các thuộc tính khoá ngoại cách nhau bằng dấu phẩy
- Danh sách các thuộc tính khoá chính cách nhau bằng dấu phẩy

Các lệnh xem thông tin của một bảng

Cú pháp	Ví dụ
<u>Xem cấu trúc bảng</u> <code>sp_help</code> [Tên Bảng]	<code>sp_help</code> NHANVIEN

<u>Xem thông tin khóa chính của bảng</u> <code>sp_pkeys</code> [Tên Bảng]	<code>sp_pkeys</code> NHANVIEN
<u>Xem thông tin khóa ngoại của bảng</u> <code>sp_fkeys</code> [Tên Bảng]	<code>sp_fkeys</code> NHANVIEN

Nhập, cập nhật dữ liệu

Một số cú pháp nhập dữ liệu

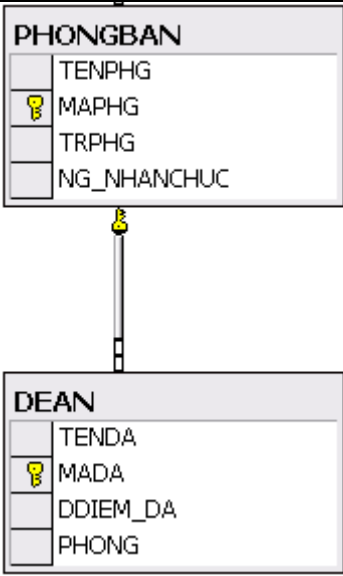
Cú pháp	Ví dụ
Ví dụ : NHANVIEN (<u>manv</u>, hoten, ngaysinh, phai, luong,phg)	
<u>Insert không tường minh</u> <code>Insert into</code> [Tên Bảng] <code>values</code> ([gt ₁], [gt ₂], ..., [gt _n])	<code>Insert into</code> NHANVIEN <code>values</code> ('NV001', 'Nguyen Van A', '12/30/1955', 'Nam', 5000, null)
<u>Insert tường minh</u> <code>Insert into</code> [Tên Bảng] ([tt ₁], [tt ₂], ..., [tt _n]) <code>values</code> ([gt ₁], [gt ₂], ..., [gt _n])	<code>Insert into</code> NHANVIEN(manv, hoten, phai, ngaysinh, luong) <code>values</code> ('NV001', 'Nguyen Van A', 'Nam', '12/30/1955', 5000)
<u>Insert từ một nguồn dữ liệu có sẵn :</u> <u>Chưa quan tâm :</u> Xem từ khóa <code>INSERT...SELECT</code>	

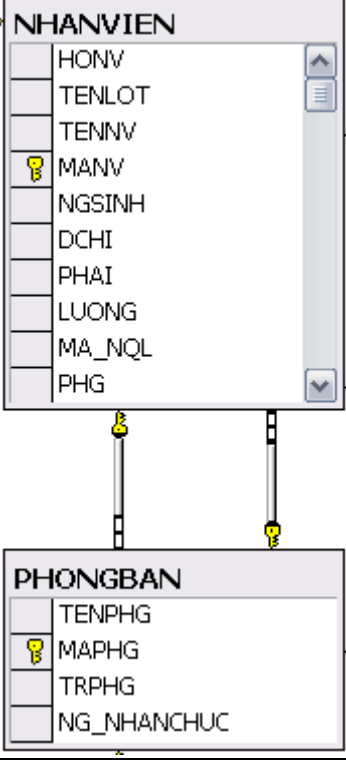
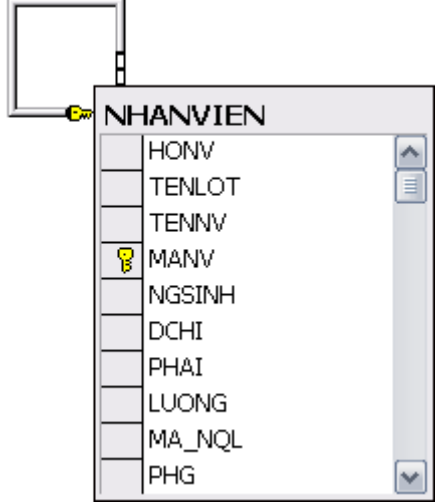
Một số lưu ý :

<u>Kiểu dữ liệu Unicode</u> Thêm kí tự N trước chuỗi Unicode	<code>Insert into</code> NHANVIEN <code>values</code> ('NV001', N'Nguyễn Văn Trường' , '12/30/1955', 'Nam', 5000, null)
<u>Kiểu dữ liệu ngày tháng</u> Định dạng nhập ngày tháng mặc định : 'mm/dd/yyyy'	<code>Insert into</code> NHANVIEN <code>values</code> ('NV001', N'Nguyễn Văn Trường' , '12/30/1955' , 'Nam', 5000, null)

<u>Insert một bộ có 1 giá trị là NULL</u> Dùng từ khóa null	Insert into NHANVIEN values ('NV001', 'Nguyễn Văn A', '12/30/1955', 'Nam', 5000, null)
<u>Thuộc tính NOT NULL</u> Nếu thuộc tính được khai báo trong cú pháp tạo bảng là NOT NULL thì bắt buộc phải có giá trị khi nhập 1 bộ vào bảng.	

Nhập dữ liệu khi đã có ràng buộc khóa ngoại:

	<u>Cách 1:</u> <ol style="list-style-type: none"> Bước 1 : Nhập phòng ban Bước 2 : Nhập đề án <u>Cách 2:</u> <ol style="list-style-type: none"> Bước 1 : Nhập DEAN, nhập PHONG = null Bước 2 : Nhập phòng ban Bước 3 : Cập nhật DEAN
--	---

	<p><u>Cách 1:</u></p> <p>Bước 1 : Nhập NHANVIEN, đặt PHG là null</p> <p>Bước 2 : Nhập PHONGBAN</p> <p>Bước 3 : Cập nhật thuộc tính PHG của NHANVIEN</p> <p><u>Cách 2 :</u></p> <p>Bước 1 : Nhập PHONGBAN, đặt TRPHG là null</p> <p>Bước 2 : Nhập NHANVIEN</p> <p>Bước 3 : Cập nhật TRPHG của phòng ban</p>
	<p><u>Cách 1 :</u></p> <ul style="list-style-type: none"> - Những nhân viên mà có MA_NQL là null thì nhập trước - Sau đó nhập những Nhân viên mà đã nhập thông tin về NQL của nhân viên đó. <p><u>Cách 2 :</u></p> <p>Bước 1. Nhập NHANVIEN, đặt thuộc tính MA_NQL là null</p> <p>Bước 2. Cập nhật MA_NQL của NHANVIEN</p>

Xóa bảng

Cú pháp câu lệnh xóa bảng:

Drop table [Tên bảng]	Drop table NHANVIEN
-----------------------	----------------------------

Lưu ý khi xóa bảng có liên quan đến khóa ngoại :

1. Nếu không có tham chiếu vòng thì tiến hành xóa bảng chứa khóa ngoại trước sau đó rồi xóa bảng còn lại, hoặc xóa khóa ngoại rồi sau đó tiến hành xóa các bảng
2. Nếu có khóa vòng thì xóa một khóa để mất khóa vòng rồi tiến hành làm như trường hợp 1

Xem dữ liệu của một bảng

<u>Xem nội dung của một bảng</u>	<u>Xem dữ liệu của bảng NHANVIEN</u>
<code>SELECT * FROM [Tên bảng]</code>	<code>SELECT * FROM NHANVIEN</code>
<u>Xóa nội dung của một bảng</u>	<u>Xóa nhân viên NV001 của bảng NHANVIEN</u>
<code>DELETE FROM [Tên bảng]</code>	<code>DELETE FROM NHANVIEN</code>
<code>WHERE [Biểu thức điều kiện]</code>	<code>WHERE manv = 'NV001'</code>
	<u>Xóa tất cả dữ liệu bảng NHANVIEN</u>
	<code>DELETE FROM NHANVIEN</code>

Ngôn ngữ SQL – Phần 2

I. Nội dung cần quan tâm

- 1) Tổng quát câu truy vấn.
- 2) Các loại truy vấn đơn giản.
- 3) Câu truy vấn group by.
- 4) Truy vấn lồng và Phép chia
- 5) Các dạng truy vấn khác.

II. Tổng quát

Một cách tổng quát, khối select gồm có 3 mệnh đề chính:

Select: Xác định các cột cần đưa ra kết quả.

From: Xác định các bảng cần lấy thông tin ra.

Where: Xác định các mẫu tin thỏa yêu cầu chọn lọc để đưa ra kết quả.

Ngoài ra, để mở rộng khả năng của ngôn ngữ, khối select-from-where còn được bổ sung thêm các mệnh đề **group by**, **having**, **order by**, các hàm hỗ trợ tính toán: **max**, **min**, **count**, **sum**, **avg**.

Sau đây là cú pháp tổng quát của câu truy vấn dữ liệu:

```

SELECT [tính chất] <danh sách các thuộc tính_1>

FROM <danh sách các table hoặc query/view [as alias] >

[WHERE <điều kiện_1>]

[GROUP BY <danh sách các thuộc tính_2>]

[HAVING <điều kiện_2>]

[ORDER BY <danh sách các thuộc tính_3 [ASC | DESC]>]

```

Diễn giải :

1. Tính chất : Một trong các từ khóa: ALL (chọn ra tất cả các dòng trong bảng), DISTINCT (loại bỏ các cột trùng lặp thông tin), DISTINCTROW (loại bỏ các dòng trùng lặp thông tin), TOP <n> (chọn n dòng đầu tiên thỏa mãn điều kiện).
2. Danh sách các thuộc tính_1: tên các thuộc tính cho biết thông tin cần lấy.

Chú ý: Các thuộc tính cách nhau bởi dấu ‘,’

Nếu lấy tất cả các thuộc tính của 1 bảng tbl thì dùng: tbl.*

Nếu sau FROM chỉ có 1 table và lấy tất cả các field của table đó thì dùng select *

Nếu tồn tại 1 thuộc tính sau select xuất hiện ở 2 table sau FROM thì phải chỉ định rõ thuộc tính đó thuộc table nào.

3. Danh sách các table: các table chứa thông tin cần lấy. Khi tìm kiếm thông tin trên nhiều hơn 2 table thì phải kết các table lại với nhau (điều kiện kết đặt sau where)
4. Alias: bí danh (tên tắt) của bảng dùng cho các bảng có tên quá dài.
5. Điều kiện_1: là điều kiện để lọc dữ liệu.
6. Danh sách các thuộc tính_2: dữ liệu sẽ được gom nhóm theo các cột này, ưu tiên từ trái sang.
7. Điều kiện_2: điều kiện lọc lại dữ liệu sau khi đã thực hiện tính toán trên dữ liệu. Điều kiện này được áp dụng trên dữ liệu thỏa mãn điều kiện_1.
8. Danh sách các thuộc tính_3: sắp xếp dữ liệu theo cột nào, thứ tự là tăng (ASC) hoặc giảm (DESC). Mặc định là dữ liệu được sắp theo thứ tự tăng dần. Việc sắp xếp được thực hiện theo thứ tự ưu tiên từ trái qua phải.

III. Truy vấn đơn giản

SELECT <danh sách thuộc tính>

FROM tên_bảng

Sau select, * được dùng với ý nghĩa lấy toàn bộ các cột của bảng.

Dùng từ khóa **distinct** để loại bỏ các bộ trùng nhau và **all** để lấy tất cả các bộ dữ liệu. Mặc định không để gì cả chính là có dùng từ khóa all.

Sau select có thể dùng các biểu thức số học như: +, -, *, /, và có thể thực hiện các toán tử trên thuộc tính.

VD:

- Cho biết danh sách tất cả các nhân viên với tất cả các thông tin

```
SELECT *
FROM NHANVIEN
```

A. Tìm kiếm có sắp xếp

Để sắp xếp thứ tự dữ liệu, ta sử dụng mệnh đề ORDER BY:

SELECT...

FROM...

ORDER BY thuộc_tính_1[ASC|DESC], thuộc_tính_2[ASC|DESC], ...

Tập_thuộc_tính gồm 1 thuộc tính hoặc nhiều thuộc tính và độ ưu tiên tính từ trái sang phải.

VD:

Với câu lệnh: *select * from Table1 order by B desc, A asc* trên bảng dưới đây:

A	B
An	8
Binh	8
Chi	9
Hung	10

Ta sẽ được kết quả sau:

A	B
Hung	10
Chi	9
An	8
Binh	8

Đầu tiên là xếp thứ tự theo B trước, sau đó, với những giá trị B ngang nhau thì sẽ xếp theo A.

VD:

- Cho biết danh sách các nhân viên sắp tên theo thứ tự Alphabet

```
SELECT MANV, TENNV, PHAI, LUONG
FROM NHANVIEN
ORDER BY TENNV
|
```

- Cho biết danh sách các nhân viên theo từng phòng ban, trong từng phòng ban tên nhân viên sắp theo thứ tự

```
SELECT PHG, MANV, TENNV, PHAI, LUONG
FROM NHANVIEN
ORDER BY PHG, TENNV
```

B. Tìm kiếm với điều kiện đơn giản

Để hỗ trợ tìm kiếm có điều kiện, sử dụng mệnh đề WHERE trong câu lệnh SELECT với vị trí như sau:

1. AND và OR

SELECT...

FROM...

WHERE (điều_kiện_1) AND/OR(điều_kiện_n)

VD:

SINHVIEN (MASV, HOTEN, NGSINH, LOP)

- Cho danh sách các sinh viên của lớp TH01.

```
SELECT *
FROM SINHVIEN
WHERE LOP = 'TH01'
```

Lưu ý: Khi thuộc tính có thể nhận giá trị null, cần cẩn thận khi sử dụng để so sánh với nhiều điều kiện liên tiếp.

2. BETWEEN...AND , NOT BETWEEN ... AND

- Cho biết các nhân viên sinh trong khoảng năm 1955 đến 1960

```
SELECT *
FROM NHANVIEN
WHERE NGSINH between '1/1/1955' and '12/31/1960'
```

Hoặc

```
SELECT *
FROM NHANVIEN
WHERE year(NGSINH) between 1955 and 1960
```

Hoặc

```
SELECT *
FROM NHANVIEN
WHERE year(NGSINH) >= 1955 and year(NGSINH) <= 1960
```

3. IS NULL và IS NOT NULL

IS NULL và IS NOT NULL : Để kiểm tra một giá trị có phải là NULL | NOT NULL hay không

- Cho biết các nhân viên không có người quản lý trực tiếp

```
SELECT HONV, TENLOT, TENNV
FROM NHANVIEN
WHERE MA_NQL IS NULL
```

- Cho biết các nhân viên có người quản lý trực tiếp

```
SELECT HONV, TENLOT, TENNV
FROM NHANVIEN
WHERE MA_NQL IS NOT NULL
```

4. IN và NOT IN

IN và NOT IN dùng để kiểm tra một giá trị nằm trong hay không nằm trong một tập hợp nào đó hay không.

- Cho biết các đơn đặt hàng có đặt mặt hàng H1, H2, H3.

```
SELECT MADH
FROM DONHANG
WHERE MAMH IN ('H1', 'H2', 'H3')
```

C. Tìm kiếm có xử lý xâu ký tự

Để xử lý với các dữ liệu thuộc dạng xâu ký tự, ngôn ngữ SQL có hỗ trợ phép LIKE. Thông thường khi so sánh thuộc tính có kiểu dữ liệu thuộc dạng xâu ký tự thì người ta thường dùng LIKE chứ không dùng phép bằng =

VD:

- Hiện ra các sinh viên tên Trang

```
SELECT *
FROM SINHVIEN
WHERE HOTEN LIKE '%Trang'
```

% : dùng để đại diện cho nhiều ký tự đứng trước từ ‘Trang’

Ngoài ra còn có các ký tự sau để mô tả mẫu cần tìm:

_ thay thế cho ký tự bất kỳ.

Chú ý:

Like “ab\%cd%” cho ra những chuỗi bắt đầu với “ab%cd”

Like “ab\\cd%” cho ra những chuỗi bắt đầu với “ab\cd”

D. Tìm kiếm có điều kiện liên quan đến ngày tháng

VD:

DDH(MADH, NGAYDH, MAKH)

CTDH(MADH, MAHH, SOLUONG, DONGIA)

- Cho biết những đơn đặt hàng đặt trước ngày 01/01/2001

```
SELECT MADH, NGAYDH
FROM DDH
WHERE DATEDIFF(D, NGAYDH, '01/01/2001') > 0
```

- Cho biết những đơn đặt hàng đặt trước ngày 01/01/2001 là 1 tuần

```
SELECT MADH, NGAYDH
FROM DDH
WHERE DATEDIFF(D, NGAYDH, '01/01/2001') > 7
```

Lưu ý :

- Cho biết các nhân viên sinh ngày 30/4/1975

Cách 1 :

```
select *
from NHANVIEN
where NGSINH = '1/1/1965'
```

Cách 2 :

```
select *
from NHANVIEN
where DATEDIFF(d, NGSINH, '1/1/1965') = 0
```

→ Cách 2 : Chính xác hơn

E. Sử dụng các hàm trong khi tìm kiếm

- Sử dụng hàm trong mệnh đề where
- Sử dụng hàm trong mệnh đề select : Trong mệnh đề select ngoài việc được sử dụng các toán tử như +, -, *, / ta còn có thể sử dụng hàm đối với các thuộc tính.
 - Các hàm về ngày tháng :
 - DateDiff
 - DatePart
 - GetDate
 - Year
 - Month
 - Day
 - DateAdd
 - Các hàm về chuỗi
 - Các hàm chuyển đổi kiểu dữ liệu
 - Các hàm toán học

○ ...

Để xem thông tin chi tiết về các hàm có thể sử dụng Book Onlines

- Cho biết họ tên nhân viên và tuổi của nhân viên

```
select HONV, TENLOT, TENNV, datediff(yyyy, NGSINH, getdate()) as TUOI
from NHANVIEN
```

- Cho biết năm sinh của nhân viên

```
select HONV, TENLOT, TENNV, year(NGSINH) as NAMSINH
from NHANVIEN
```

- Cho biết họ và tên đầy đủ của nhân viên

```
select HONV + ' ' + TENLOT + ' ' + TENNV as HoVaTen
from NHANVIEN
```

F. Tìm kiếm từ nhiều bảng

Để tìm kiếm thông tin mà thông tin đó nằm ở nhiều bảng khác nhau thì khai báo sử dụng các bảng đó tại mệnh đề FROM. Tùy theo thông tin cần hiển thị mà chúng ta sẽ sử dụng điều kiện tại mệnh đề WHERE sao cho thích hợp.

VD:

- Cho biết mã nhân viên, tên nhân viên, tên phòng ban mà nhân viên trực thuộc.

```
SELECT MANV, TENNV, TENPB
FROM NHANVIEN, PHONGBAN
WHERE NHANVIEN.MAPB = PHONGBAN.MAPB
```

G. Dùng toán tử θ some, θ all, exists, not exists

Lưu ý: \triangleleft some và not in, \triangleleft all = not in.

IV. Câu truy vấn sử dụng Group By

A. Các hàm tính toán

SQL sử dụng các hàm sau: Count, Max, Min, Sum, Avg. Hàm Count dùng đối số * có nghĩa là đếm tất cả các mẫu tin thỏa điều kiện đếm mà không cần quan tâm đến bất kỳ cột nào.

- Có tất cả bao nhiêu sinh viên trong lớp th01

```
SELECT COUNT(*)
FROM SINHVIEN
WHERE LOP = 'TH01'
```

B. Mệnh đề group by

Dùng để gom nhóm dữ liệu, thường dùng kết hợp với một hàm tính toán kể trên.

- Tính điểm trung bình của từng sinh viên, biết rằng điểm số lưu trong bảng KETQUA(MASV, MAMH, DIEM)

```
SELECT MASV, AVG(DIEM)
FROM KETQUA
GROUP BY MASV
```

- Cho biết lương lớn nhất trong từng phòng ban

NHANVIEN(MANV, TENNV, PHAI, LUONG, PHG)

PHONGBAN(MAPB, TENPB, TRPHG)

THANNHAN(MA_NVN, TENTN, PHAI, QUANHE)

```
SELECT PHG, MAX(LUONG)
FROM NHANVIEN
GROUP BY PHG
```

C. Mệnh đề having

Mệnh đề HAVING thường được sử dụng cùng với mệnh đề GROUP BY. Sau HAVING là biểu thức điều kiện. Biểu thức điều kiện này không tác động vào toàn bảng được chỉ ra ở mệnh đề from mà chỉ tác động lần lượt từng nhóm các mẫu tin đã chỉ ra trong mệnh đề group by.

- Cho biết các sinh viên có điểm trung bình lớn hơn hoặc bằng 8.0

```
SELECT MASV, AVG(DIEM)
FROM KETQUA
GROUP BY MASV
HAVING AVG(DIEM) >= 8.0
```

V. Truy vấn lồng

A. Tìm kiếm có lượng từ EXISTS, ANY và ALL

- Cho danh sách các nhân viên có ít nhất 1 thân nhân.

```
SELECT MANV, TENNV
FROM NHANVIEN
WHERE EXISTS (SELECT TENTN
FROM THANNHAN
WHERE THANNHAN.MA_NVN = NHANVIEN.MANV)
```

Câu này có thể viết lại như sau:

```
SELECT MANV, TENNV
FROM NHANVIEN
WHERE (SELECT COUNT(*)
FROM THANNHAN
WHERE THANNHAN.MA_NVN = NHANVIEN.MANV) > 0
```


Chú ý: = ANY tương đương với toán tử IN

- Cho biết nhân viên có lương lớn nhất.

```
SELECT MANV, LUONG
FROM NHANVIEN
WHERE LUONG >= ALL (SELECT LUONG FROM NHANVIEN)
```

Hoặc có thể viết như sau:

```
SELECT MANV, LUONG
FROM NHANVIEN
WHERE LUONG = (SELECT MAX(LUONG) FROM NHANVIEN)
```

- Cho biết sinh viên có điểm trung bình lớn nhất.

```
SELECT MASV, AVG(DIEM)
FROM KETQUA
GROUP BY MASV
WHERE AVG(DIEM) >= ALL (SELECT AVG(DIEM)
FROM KETQUA
GROUP BY MASV)
```

Có 2 loại truy vấn lồng

B. Loại 1: Lồng phân cấp

Mệnh đề WHERE của truy vấn con không tham chiếu đến thuộc tính của các quan hệ trong mệnh đề FROM ở truy vấn cha

Khi thực hiện, câu truy vấn con sẽ được thực hiện trước

Ví dụ:

- Cho biết các nhân viên cùng phòng với nhân viên “Nguyễn Văn A”

```
SELECT MANV, TENNV
FROM NHANVIEN
WHERE PHG IN (
    SELECT PHG
    FROM NHANVIEN
    WHERE TENNV = 'Nguyễn Văn A'
)
```

Quan hệ **NHANVIEN** ở truy vấn con không liên quan đến quan hệ **NHANVIEN** ở truy vấn cha

- Tìm những nhân viên có lương lớn hơn lương của tất cả nhân viên ở phòng 4.

```
SELECT MANV, TENNV
FROM NHANVIEN
WHERE LUONG > (
    SELECT MAX(LUONG)
    FROM NHANVIEN
    WHERE PHG = 4
)
```

- Tìm phòng ban có đông nhân viên nhất (gom nhóm + truy vấn lồng phân cấp)

```
select pb.TENPHG, count(*) as SOLUONG
from NHANVIEN nv, PHONGBAN pb
where nv.PHG = pb.MAPHG
group by nv.PHG, pb.TENPHG
having count(*) >= ALL(
    select count(*)
    from NHANVIEN
    group by PHG)
```

- Cho biết họ tên nhân viên (HONV, TENLOT, TENNV) có mức lương lớn hơn mức lương của một nhân viên nào đó của phòng "Nghiên cứu"

```
select HONV, TENLOT, TENNV, LUONG, PHG
from NHANVIEN
where LUONG > any(select nv.LUONG
    from NHANVIEN nv, PHONGBAN pb
    where nv.PHG=pb.MAPHG
    and pb.TENPHG=N'Nghiên cứu' )
```

-

C. Loại 2: Lồng tương quan

Mệnh đề WHERE của truy vấn con tham chiếu ít nhất một thuộc tính của các quan hệ trong mệnh đề FROM ở truy vấn cha.

Khi thực hiện, câu truy vấn con sẽ được thực hiện nhiều lần, mỗi lần tương ứng với một bộ của truy vấn cha.

Ví dụ:

- Tìm những nhân viên không có thân nhân nào:

```
SELECT MANV, TENNV
FROM NHANVIEN n
WHERE NOT EXISTS
```

```
(SELECT *
FROM THANNHAN t
WHERE t.MANV = n.MANV)
```

Trong truy vấn con này có tham chiếu đến thuộc tính **MANV** của quan hệ NHANVIEN n trên truy vấn cha

- Tìm tất cả các nhân viên làm việc ở phòng nghiên cứu

```
SELECT MANV, TENNV
FROM NHANVIEN n
WHERE EXISTS
```

```
(SELECT *
FROM PHONGBAN p
WHERE TENPHG = 'Nghiên cứu'
and p.MAPHG=n.PHG)
```

Trong truy vấn con này có tham chiếu đến thuộc tính **PHG** của quan hệ NHANVIEN n trên truy vấn cha

VI. Phép chia

Có 2 cách thực hiện:

Cách 1: Sử dụng NOT EXISTS + NOT IN hoặc NOT EXISTS + NOT EXISTS

Cách 2: Sử dụng mệnh đề GROUP BY + HAVING

VD:

- Tìm nhân viên được phân công làm việc trong tất cả các đề án do phòng **Nghiên cứu** quản lí

Cách 1:

Sử dụng NOT EXISTS + NOT IN

```
SELECT n.MANV, n.TENNV
FROM NHANVIEN n
WHERE NOT EXISTS
(
    SELECT *
    FROM PHONGBAN pb, DEAN d
    WHERE d.PHONG = pb.MAPHG AND pb.TENPHG = 'Nghiên cứu' AND
    d.MADA NOT IN
    (
        SELECT pc.MADA
        FROM PHANCONG pc
        WHERE pc.MA_NVIENT = n.MANV
    )
)
```

Sử dụng NOT EXISTS + NOT EXISTS

```
SELECT n.MANV, n.TENNV
FROM NHANVIEN n
WHERE NOT EXISTS
(
    SELECT *
    FROM PHONGBAN pb, DEAN d
    WHERE d.PHONG = pb.MAPHG AND pb.TENPHG = 'Nghiên cứu'
    AND NOT EXISTS
    (
        SELECT *
        FROM PHANCONG pc
        WHERE pc.MA_NVIENT = n.MANV AND pc.MADA = d.MADA
    )
)
```

Cách 2: Sử dụng GROUP BY + HAVING

```

SELECT n.MANV, n.TENNV
FROM NHANVIEN n, PHANCONG pc, PHONGBAN pb1, DEAN d1
WHERE n.MANV = pc.MA_NVIEN AND pc.MADA = d1.MADA
and d1.PHONG = pb1.MAPHG AND pb1.TENPHG = 'Nghiên cứu'
GROUP BY n.MANV, n.TENNV
HAVING COUNT (DISTINCT pb.MADA) =
        (SELECT COUNT(DISTINCT d2.MADA)
        FROM DEAN d2, PHONGBAN pb2
        WHERE d2.PHONG = pb2.MAPHG AND pb2.TENPHG = 'Nghiên cứu')

```

VII. Các loại truy vấn khác

A. Truy vấn con ở mệnh đề SELECT

- Với mỗi nhân viên, cho biết họ, tên nhân viên và số thân nhân của họ

```

SELECT HONV, TENLOT, TENNV, ( select count(*)
                             from THANNHAN
                             where MA_NVIEN = nv.MANV) as SoTN
FROM NHANVIEN nv

```

- Với mỗi phòng ban, cho biết tên phòng ban và lương trung bình của phòng ban

```

SELECT pb.TENPHG, ( select avg(LUONG)
                   from NHANVIEN
                   where PHG = pb.MAPHG) as LuongTB
FROM PHONGBAN pb

```

B. Truy vấn con ở mệnh đề FROM

Kết quả trả về của một câu truy vấn phụ là một bảng

- Bảng trung gian trong quá trình truy vấn
- Không có lưu trữ thật sự

VD:

- Cho biết những phòng ban (TENPHG) có lương trung bình của các nhân viên lớn hơn 20000

```

SELECT TENPHG, TEMP.LUONG_TB
FROM PHONGBAN, (SELECT PHG, AVG(LUONG) AS LUONG_TB
                FROM NHANVIEN
                GROUP BY PHG
                HAVING AVG(LUONG) > 20000 ) AS TEMP
WHERE MAPHG=TEMP.PHG

```

C. Điều kiện kết ở mệnh đề FROM

VD:

- Tìm mã và tên các nhân viên làm việc tại phòng ‘Nghien cuu’

```
SELECT MANV, TENNV
FROM NHANVIEN INNER JOIN PHONGBAN ON PHG=MAPHG
WHERE TENPHG='Nghien cuu'
```

- Cho biết họ tên nhân viên và tên phòng ban mà họ là trưởng phòng nếu có

```
SELECT TENNV, HONV, TENPHG
FROM PHONGBAN RIGHT JOIN NHANVIEN ON MANV=TRPHG
```

- Tìm họ tên các nhân viên và tên các đề án nhân viên tham gia nếu có

```
SELECT NV.TENNV, NV.TENDA
FROM (PHANCONG PC JOIN DEAN DA ON SODA=MADA)
RIGHT JOIN NHANVIEN NV ON PC.MA_NVIENT=NVS.MANV
```

D. Cấu trúc Case

- Cho biết họ tên các nhân viên đã đến tuổi về hưu (nam 60 tuổi, nữ 55 tuổi)

```
SELECT HONV, TENNV
FROM NHANVIEN
WHERE YEAR(GETDATE()) - YEAR(NGSINH) >= ( CASE PHAI
                                            WHEN 'Nam' THEN 60
                                            WHEN 'Nu' THEN 55
                                            END )
```

- Cho biết họ tên các nhân viên và năm về hưu

```
SELECT HONV, TENNV,
(CASE PHAI
 WHEN 'Nam' THEN YEAR(NGSINH) + 60
 WHEN 'Nu' THEN YEAR(NGSINH) + 55
END ) AS NAMVEHUU
FROM NHANVIEN
```

Ngôn ngữ SQL – Phần 3

I. Mục lục

I. Mục lục	1
II. Một số lưu ý về phép kết.....	1
A. Inner joins (Kết bằng)	1
B. Right (Outer) joins (Kết phải)	2
C. Left (Outer) joins (Kết trái).....	3
D. Full (Outer) joins.....	3
III. Phép chia, phép hội, phép giao và phép trừ	5
A. Phép chia	5
B. Phép hội (UNION)	6
C. Phép giao (Intersect)	7
D. Phép trừ	8

II. Một số lưu ý về phép kết

<u>SINHVIEN</u>		
	malop	tenlop
1	L1	10A
2	L2	10B
3	L3	10C

<u>LOP</u>			
	masv	hoten	malop
1	01	A	L1
2	02	B	L2
3	03	C	L2
4	04	D	L1
5	05	E	L1

Yêu cầu: Cho biết sĩ số của mỗi lớp

A. Inner joins (Kết bằng)

Phép kết Inner joins giữa 2 bảng A và B \rightarrow là một bảng C = {các bộ trong đó mỗi bộ là sự kết hợp của các bộ trong A với các bộ trong B sao cho điều kiện kết được thỏa mãn}

- Phép kết inner join giữa SINHVIEN và LOP

```
select *
from SINHVIEN sv join LOP l on sv.malop = l.malop
```

Kết quả

	masv	hoten	malop	malop	tenlop
1	01	A	L1	L1	10A
2	02	B	L2	L2	10B
3	03	C	L2	L2	10B
4	04	D	L1	L1	10A
5	05	E	L1	L1	10A

Nhận xét : Thông tin về lớp 10C bị mất

- Tính số của lớp

1

```
select l.malop, l.tenlop, count(*) as SiSo
from SINHVIEN sv join LOP l on sv.malop = l.malop
group by l.malop, l.tenlop
```

Kết quả :

	malop	tenlop	SiSo
1	L1	10A	3
2	L2	10B	2

Nhận xét : Số của lớp 10C (bằng 0) không được xuất ra, vì thông tin lớp 10C đã bị mất sau phép kết bằng

B. Right (Outer) joins (Kết phải)

Phép kết Right Outer joins giữa 2 bảng A và B → là một bảng C = {các bộ trong đó mỗi bộ là sự kết hợp của các bộ trong A với các bộ trong B sao cho điều kiện kết được thỏa mãn} + {các bộ còn lại trong B mà không thỏa điều kiện kết với bất kỳ một bộ trong A nào}

- Phép kết Right (Outer) Joins giữa SINHVIEN và LOP

```
select *
from SINHVIEN sv right join LOP l on sv.malop = l.malop
```

Kết quả :

	masv	hoten	malop	malop	tenlop
1	01	A	L1	L1	10A
2	04	D	L1	L1	10A
3	05	E	L1	L1	10A
4	02	B	L2	L2	10B
5	03	C	L2	L2	10B
6	NULL	NULL	NULL	L3	10C

Nhận xét : Thông tin về lớp 10C vẫn được giữ lại sau phép kết phải

- Tính sĩ số của lớp

2

```
select l.malop, l.tenlop, count(sv.MaLop) as SiSo
from SINHVIEN sv right join LOP l on sv.malop = l.malop
group by l.malop, l.tenlop
```

Kết quả :

	malop	tenlop	SiSo
1	L1	10A	3
2	L2	10B	2
3	L3	10C	0

Nhận xét : Sĩ số của các lớp không có học sinh (10 C) vẫn được xuất ra (vì phép kết không mất thông tin về lớp)

Câu hỏi :

1

2

1. Tại sao và lại khác nhau ?

2

2. Tại sao là `count(sv.MaLop)` chứ không phải là `count(l.MaLop)`

C. Left (Outer) joins (Kết trái)

Phép kết Left (Outer) joins giữa 2 bảng A và B \rightarrow là một bảng C = {các bộ trong đó mỗi bộ là sự kết hợp của các bộ trong A với các bộ trong B sao cho điều kiện kết được thỏa mãn} + {các bộ còn lại trong A mà không thỏa điều kiện kết với một bộ bất kỳ trong B nào}

D. Full (Outer) joins

Phép kết Full Outer joins giữa 2 bảng A và B \rightarrow là một bảng C = {các bộ trong đó mỗi bộ là sự kết hợp của các bộ trong A với các bộ trong B sao cho điều kiện kết được thỏa mãn} + {các bộ còn lại trong A mà không thỏa điều kiện kết với bất kỳ một bộ trong B nào} + {các bộ còn lại trong B mà không thỏa điều kiện kết với bất kỳ một bộ trong A nào}

PUBLISHER

	pid	pname	pcity
1	1	Algodata Infosystems 1	MIA
2	2	Algodata Infosystems 2	NYO
3	3	Algodata Infosystems 3	MAN

AUTHORS

	auid	firstname	lastname	city
1	1	Reginald	Blotchett-Halls	NYO
2	2	Michel	DeFrance	OAS
3	3	Innes	del Castillo	CAN
4	4	Ann	Dull	LND
5	5	Marjorie	Green	CAL
6	6	Morningstar	Greene	CAL
7	7	Burt	Gringlesby	LOS
8	8	Sheryl	Hunter	NYO

```
select *
from AUTHORS full join PUBLISHERS on city=pcity
```

Kết quả :

	auid	first...	lastname	city	pid	pname	pcity
1	1	Reginald	Blotch...	NYO...	2	Algoda...	NYO
2	2	Michel	DeFrance	OAS...	NULL	NULL	NULL
3	3	Innes	del Ca...	CAN...	NULL	NULL	NULL
4	4	Ann	Dull	LND...	NULL	NULL	NULL
5	5	Marjorie	Green	CAL...	NULL	NULL	NULL
6	6	Morni...	Greene	CAL...	NULL	NULL	NULL
7	7	Burt	Gringlesby	LOS...	NULL	NULL	NULL
8	8	Sheryl	Hunter	NYO...	2	Algoda...	NYO
9	NULL	NULL	NULL	NULL	3	Algoda...	MAN
10	NULL	NULL	NULL	NULL	1	Algoda...	MIA

- Cho biết những tác giả và nhà xuất bản ở cùng thành phố
- Cho biết số lượng tác giả và nhà xuất bản ở cùng thành phố, số lượng tác giả mà không có nhà xuất bản nào ở cùng thành phố và số lượng nhà xuất bản mà không có tác giả nào ở cùng thành phố

```

select  count( case when auid is null then 1 end) as
        [Số tác giả không cùng thành phố với bất kỳ NXB nào],
        count( case when pid is null then 1 end) as
        [Số NXB không cùng thành phố với bất kỳ tác giả nào],
        count(case when city = pcity then 1 end) as
        [Số NXB và Tác giả cùng thành phố]
from AUTHORS full join PUBLISHERS on city=pcity

```

III. Phép chia, phép hội, phép giao và phép trừ

A. Phép chia

- Tìm nhân viên làm việc tất cả các đề án của công ty
 - **Bước 1** : Tìm các đề án của công ty mà nhân viên '001' chưa làm

Cách 1 : Dùng NOT EXISTS

```

select *
from DEAN da
where not exists (
    select *
    from PHANCONG pc
    where pc.MA_NVIEN = '001' and
          pc.MADA = da.MADA)

```

Cách 2 : Dùng NOT IN

```

select *
from DEAN
where MADA not in (
    select MADA
    from PHANCONG
    where MA_NVIEN='001')

```

- **Bước 2** : Nhận xét rằng : Nếu kết quả trả ra là không có bộ nào → Nhân viên '001' làm tất cả các đề án. Ngược lại, nếu kết quả trả ra là có từ 1 bộ dữ liệu trở lên → có đề án của công ty mà nhân viên '001' chưa làm → nhân viên '001' không làm mọi đề án của công ty.

Do vậy, Tìm các nhân viên làm mọi đề án của công ty tương đương với việc, kiểm tra từng nhân viên, nếu danh sách các đề án của công ty nhân viên đó chưa làm là rỗng (không có bộ nào) → nhân viên làm mọi đề án của công ty, ngược lại thì nhân viên đó không làm mọi đề án của công ty.

Cách 1 : Nếu bước 1 sử dụng NOT EXISTS

```

select *
from NHANVIEN nv
where not exists (
    select *
    from DEAN da
    where not exists (
        select *
        from PHANCONG pc
        where pc.MA_NVIEN = nv.MANV and
              pc.MADA = da.MADA))

```

Cách 2 : Nếu bước 1 sử dụng NOT IN

```

select *
from NHANVIEN nv
where not exists (
    select *
    from DEAN
    where MADA not in (
        select MADA
        from PHANCONG
        where MA_NVIEN=nv.MANV))

```

B. Phép hội (UNION)

Phép hội (Union) sử dụng để tổng hợp dữ liệu từ các bảng → 1 bảng

- UNION : Các dòng trùng lặp sẽ được bỏ đi
- UNION ALL : Lấy tất cả các dòng của các bảng

Điều kiện để thực hiện được Union : Các bảng phải có cùng số lượng thuộc tính và tương ứng kiểu dữ liệu giữa các cột.

Các cột của bảng kết xuất chính là các cột trong bảng đầu tiên.

TABLE1		TABLE2	
a	b	a	b
1	2	2	7
3	4	3	4
2	3	2	3
4	5	1	6

UNION

UNION ALL

<pre>select * from table1 union select * from table2</pre>	<pre>select * from table1 union all select * from table2</pre>																																
<p><u>Kết quả:</u></p> <table border="1"> <thead> <tr> <th>a</th><th>b</th></tr> </thead> <tbody> <tr><td>1</td><td>2</td></tr> <tr><td>1</td><td>6</td></tr> <tr><td>2</td><td>3</td></tr> <tr><td>2</td><td>7</td></tr> <tr><td>3</td><td>4</td></tr> <tr><td>4</td><td>5</td></tr> </tbody> </table>	a	b	1	2	1	6	2	3	2	7	3	4	4	5	<p><u>Kết quả :</u></p> <table border="1"> <thead> <tr> <th>a</th><th>b</th></tr> </thead> <tbody> <tr><td>1</td><td>2</td></tr> <tr><td>3</td><td>4</td></tr> <tr><td>2</td><td>3</td></tr> <tr><td>4</td><td>5</td></tr> <tr><td>2</td><td>7</td></tr> <tr><td>3</td><td>4</td></tr> <tr><td>2</td><td>3</td></tr> <tr><td>1</td><td>6</td></tr> </tbody> </table>	a	b	1	2	3	4	2	3	4	5	2	7	3	4	2	3	1	6
a	b																																
1	2																																
1	6																																
2	3																																
2	7																																
3	4																																
4	5																																
a	b																																
1	2																																
3	4																																
2	3																																
4	5																																
2	7																																
3	4																																
2	3																																
1	6																																

C. Phép giao (Intersect)

Sử dụng EXISTS hoặc IN để thực hiện phép giao

- Tìm các nhân viên có làm đề án của phòng nghiên cứu và vừa là trưởng phòng

Nhận xét : nhân viên có làm đề án của phòng nghiên cứu và vừa là trưởng phòng = Nhân viên làm đề án của phòng nghiên cứu \cap Nhân viên là trưởng phòng

- Sử dụng ...IN (...) AND ...IN (...) \rightarrow Lồng phân cấp

```
select *
from NHANVIEN
where MANV in ( select TRPHG
                from PHONGBAN pb)
and MANV in ( select pc.MA_NVIEN
                from PHANCONG pc, DEAN da, PHONGBAN pb
                where pc.MADA=da.MADA and
                      da.PHONG=pb.MAPHG and
                      pb.TENPHG = N'Nghiên Cứu')
```

- Sử dụng ... EXISTS \rightarrow Lồng tương quan

```

select *
from NHANVIEN nv, PHONGBAN pb
where pb.TRPHG = nv.MANV and exists (
    select *
    from PHANCONG pc, DEAN da, PHONGBAN pb
    where pc.MADA=da.MADA and
          da.PHONG=pb.MAPHG and
          pb.TENPHG = N'Nghiên Cứu' and
          pc.MA_NVIENT=nv.MANV)

```

- Sử dụng ... IN → Lồng phân cấp

```

select *
from NHANVIEN nv, PHONGBAN pb
where pb.TRPHG = nv.MANV and MANV in (
    select pc.MA_NVIENT
    from PHANCONG pc, DEAN da, PHONGBAN pb
    where pc.MADA=da.MADA and
          da.PHONG=pb.MAPHG and
          pb.TENPHG = N'Nghiên Cứu')

```

Sử dụng EXISTS và NOT EXISTS để thực hiện phép giao và phép trừ

D. Phép trừ

Sử dụng NOT EXISTS hoặc NOT IN để thực hiện phép trừ

- Tìm các nhân viên không tham gia đề án nào

Nhận xét : Nhân viên không tham gia đề án = Tất cả nhân viên – Nhân viên có tham gia đề án

Sử dụng NOT EXISTS

```

SELECT NV.HONV, NV.TENLOT, NV.TENNV
FROM NHANVIEN NV
WHERE NOT EXISTS (
    SELECT *
    FROM PHANCONG
    WHERE MA_NVIENT=NV.MANV)

```

SỬ DỤNG NOT IN

```

SELECT NV.HONV, NV.TENLOT, NV.TENNV
FROM NHANVIEN NV
WHERE NV.MANV NOT IN(
    SELECT MA_NVIENT
    FROM PHANCONG)

```

- Đề án không có nhân viên nào tham gia

Nhận xét : Đề án không có nhân viên tham gia = Tất cả các đề án – Đề án có nhân viên tham gia

Sử dụng NOT EXISTS

```
SELECT *  
FROM DEAN da  
WHERE NOT EXISTS (  
    SELECT *  
    FROM PHANCONG  
    WHERE MADA = da.MADA)
```

Sử dụng NOT IN

```
SELECT *  
FROM DEAN da  
WHERE MADA NOT IN (  
    SELECT MADA  
    FROM PHANCONG)
```

Ngôn ngữ SQL – Phần 4

I. Mục lục

I.	Mục lục	1
II.	Tổng hợp dữ liệu sử dụng COMPUTE, COMPUTE BY, CUBE.....	2
A.	COMPUTE	2
B.	COMPUTE BY	2
C.	GROUP BY ... WITH CUBE	2
D.	GROUP BY ... WITH ROLLUP.....	3
III.	Các câu lệnh INSERT, UPDATE mở rộng.....	4
A.	UPDATE dữ liệu từ dữ liệu có sẵn	4
B.	INSERT dữ liệu vào một bảng từ một bảng có sẵn	4
IV.	Cấu trúc CASE.....	4
A.	Cấu trúc 1	4
B.	Cấu trúc 2	5

II. Tổng hợp dữ liệu sử dụng COMPUTE, COMPUTE BY, CUBE

A. COMPUTE

Sử dụng để tổng hợp dữ liệu của các bảng

- Cho biết các nhân viên, tổng lương, lương trung bình của tất cả các nhân viên

```
SELECT *
FROM NHANVIEN
COMPUTE SUM (LUONG) , AVG (LUONG) , MIN (LUONG) , MAX (LUONG)
```

B. COMPUTE BY

- Cho biết các nhân viên của từng phòng, tổng lương, lương trung bình của từng phòng

```
SELECT *
FROM NHANVIEN
ORDER BY PHG
COMPUTE SUM (LUONG) , AVG (LUONG) , MIN (LUONG) , MAX (LUONG) BY PHG
```

- Lưu ý :

- Các thuộc tính sau COMPUTE ... BY phải có trong danh sách các thuộc tính sau ORDER BY
- Không đặt tên kết quả trả ra được

C. GROUP BY ... WITH CUBE

	itemid	itemname	color	quantity
1	1	Table	Blue	124
2	2	Table	Red	223
3	3	Chair	Blue	101
4	4	Chair	Red	210

- Tổng hợp số lượng của các item theo tên và màu, theo từng tên, theo từng màu, tổng số item

```
SELECT ItemName, Color, SUM(Quantity) AS QtySum
FROM Inventory
GROUP BY ItemName, Color WITH CUBE
```

Kết quả là :

	ItemName	Color	QtySum	
1	Chair	Blue	101	Số lượng Item tên Chair và màu Blue là 101
2	Chair	Red	210	
3	Chair	NULL	311	Số lượng Item tên Table (màu bất kỳ) là : 347
4	Table	Blue	124	
5	Table	Red	223	Số lượng Item (tên bất kỳ, màu bất kỳ) = tổng số item là 658
6	Table	NULL	347	
7	NULL	NULL	658	Số lượng Item màu Red (tên bất kỳ) : 433
8	NULL	Blue	225	
9	NULL	Red	433	

Nhận xét

- Group by n thuộc tính → sẽ thống kê theo 2ⁿ tiêu chí
- Những thống kê mà không có dữ liệu sẽ không được xuất ra

D. GROUP BY ... WITH ROLLUP

- Tổng hợp số lượng của các item theo **từng tên** và màu, theo **từng tên**, tổng số item

```
SELECT ItemName, Color, SUM(Quantity) AS QtySum
FROM Inventory
GROUP BY ItemName, Color WITH ROLLUP
```

Thống kê theo **từng tên** và màu, theo **từng tên** → tương tự **WITH CUBE** nhưng loại bỏ bớt tiêu chí thống kê

Kết quả là :

	ItemName	Color	QtySum	
1	Chair	Blue	101	Số lượng Item tên Chair và màu Blue là 101
2	Chair	Red	210	
3	Chair	NULL	311	Số lượng Item tên Table (màu bất kỳ) là : 347
4	Table	Blue	124	
5	Table	Red	223	Số lượng Item (tên bất kỳ, màu bất kỳ) = tổng số item là 658
6	Table	NULL	347	
7	NULL	NULL	658	

III. Các câu lệnh INSERT, UPDATE mở rộng

Cho các quan hệ sau :

SINHVIEN (MASV, HOTEN, DIEMTB, HANG)

SINHVIENGIOI(MASV, HOTEN, DIEMTB)

A. UPDATE dữ liệu từ dữ liệu có sẵn

- Cập nhật hạng của sinh viên

```
update SINHVIEN
set HANG = (SELECT count (*)
            FROM SINHVIEN sv
            WHERE sv.DIEMTB >= SINHVIEN.DIEMTB)
```

B. INSERT dữ liệu vào một bảng từ một bảng có sẵn

- Thêm dữ liệu vào bảng SINHVIENGIOI các sinh viên có điểm trung bình từ 8.0 trở lên

```
insert into SINHVIENGIOI
select MASV, HOTEN, DIEMTB
from SINHVIEN
where DIEMTB > 8
```

IV. Cấu trúc CASE

A. Cấu trúc 1

```
CASE input_expression
WHEN when_expression THEN result_expression
[ ...n ]
[
    ELSE else_result_expression
]
END
```

Ví dụ :

- Cho biết họ tên các nhân viên và năm về hưu

```
SELECT HONV, TENNV,
(CASE PHAI
    WHEN 'Nam' THEN YEAR(NGSINH) + 60
    WHEN 'Nu' THEN YEAR(NGSINH) + 55
END ) AS NAMVEHUU
FROM NHANVIEN
```

- Cho biết họ tên các nhân viên đã đến tuổi về hưu (nam 60 tuổi, nữ 55 tuổi)

```

SELECT HONV, TENNV
FROM NHANVIEN
WHERE YEAR(GETDATE()) - YEAR(NGSINH) >= ( CASE PHAI
                                           WHEN 'Nam' THEN 60
                                           WHEN 'Nu' THEN 55
                                           END )

```

B. Cấu trúc 2

```

CASE
  WHEN Boolean_expression THEN result_expression
  [ ...n ]
  [
    ELSE else_result_expression
  ]
END

```

Ví dụ:

- Cho biết sinh viên và xếp loại học lực của sinh viên

```

select MASV, CASE
              WHEN DIEMTB >= 8 THEN N'Giỏi'
              WHEN DIEMTB >= 7 THEN N'Khá'
              WHEN DIEMTB >= 6 THEN N'Trung bình khá'
              WHEN DIEMTB >= 5 THEN N'Trung bình'
              END
from SINHVIEN

```

Ngôn ngữ SQL – Phần 5

I. Đối tượng SYSOBJECTS

A. Giới thiệu

- Sysobjects → là một bảng của hệ thống, lưu thông tin về tất cả các đối tượng có trong cơ sở dữ liệu. Ví dụ như là, các bảng của hệ thống, các bảng do người dùng tạo ra, các khóa chính, các khóa ngoại, các ràng buộc check constraint, ràng buộc UNIQUE, ... các hàm do người dùng định nghĩa, các store procedure, ...
- Từ khóa trong book onlines : **sysobjects**

B. Cấu trúc của bảng SYSOBJECTS

Tên cột	Kiểu dữ liệu	Ý nghĩa
name	sysname	Object name.
Id	int	Object identification number.
xtype	char(2)	Object type
uid	smallint	User ID of owner object.
info	smallint	Reserved. For internal use only.
status	int	Reserved. For internal use only.
base_schema_ver	int	Reserved. For internal use only.
replinfo	int	Reserved. For use by replication.
parent_obj	int	Object identification number of parent object (for example, the table ID if a trigger or constraint).
crdate	datetime	Date the object was created.
ftcatid	smallint	Identifier of the full-text catalog for all user tables registered for full-text indexing, and 0 for all user tables not registered.
schema_ver	int	Version number that is incremented every time the schema for a table changes.
stats_schema_ver	int	Reserved. For internal use only.
type	char(2)	Object type
userstat	smallint	Reserved.
sysstat	smallint	Internal status information.
indexdel	smallint	Reserved.
refdate	datetime	Reserved for future use.
version	int	Reserved for future use.
deltrig	int	Reserved.
instrig	int	Reserved.

updtrig	int	Reserved.
seltrig	int	Reserved.
category	int	Used for publication, constraints, and identity.
cache	smallint	Reserved.

C. Các giá trị của TYPE và XTYPE

XTYPE	Ý nghĩa	TYPE	Ý nghĩa
C	CHECK constraint	C	CHECK constraint
D	Default or DEFAULT constraint	D	Default or DEFAULT constraint
F	FOREIGN KEY constraint	F	FOREIGN KEY constraint
L	Log	FN	Scalar function
FN	Scalar function	IF	Inlined table-function
IF	Inlined table-function	K	PRIMARY KEY or UNIQUE constraint
P	Stored procedure	L	Log
PK	PRIMARY KEY constraint (type is K)	P	Stored procedure
RF	Replication filter stored procedure	R	Rule
S	System table	RF	Replication filter stored procedure
TF	Table function	S	System table
TR	Trigger	TF	Table function
U	User table	TR	Trigger
UQ	UNIQUE constraint (type is K)	U	User table
V	View	V	View
X	Extended stored procedure	X	Extended stored procedure

D. Một số ví dụ minh họa ứng dụng

- Liệt kê các bảng do người dùng tạo ra

```
select *
from sysobjects
where type='U'
```

- Kiểm tra một đối tượng có tồn tại hay không để thực hiện một thao tác nào đó

Kiểm tra nếu bảng NHAN_VIEN tồn tại thì xóa bảng

```
if exists (
    select *
    from sysobjects
    where name= 'NHAN_VIEN' and type='U')
begin
    drop table NHAN_VIEN
end
```

Kiểm tra nếu khóa ngoại FK_NHANVIEN_PHONGBAN tồn tại thì xóa

```
if exists (
    select *
    from sysobjects
    where name= 'FK_NHANVIEN_PHONGBAN' and type='F')
begin
    drop table FK_NHANVIEN_PHONGBAN
end
```

II. Đối tượng SYSDATABASES

A. Giới thiệu

- SYSDATABASES là một bảng lưu thông tin về các cơ sở dữ liệu của hệ thống. Mỗi dòng trong SYSDATABASES là thông tin về một CSDL nào đó.
- SYSDATABASES là một table chỉ có trong CSDL master.

- Từ khóa trong book onlines : **sysdatabases**

B. Cấu trúc bảng – Một số thuộc tính cần quan tâm

Column name	Data type	Description
name	sysname	Tên của databases
dbid	smallint	Database ID
sid	varbinary(85)	System ID of the database creator.
mode	smallint	Used internally for locking a database while it is being created.
crdate	datetime	Creation date.
filename	nvarchar(260)	Operating-system path and name for the database's primary file.
...		

C. Ví dụ

- Lấy danh sách các tên database của hệ thống

```
-- Vì sysdatabases chỉ có trong master --> use master
use master
go
select name
from sysdatabases
```

- Kiểm tra một database có tồn tại hay không

Kiểm tra nếu database QLDA tồn tại thì xóa

```
-- Vì sysdatabases chỉ có trong master --> use master
use master
go
if exists (select *
          from sysdatabases
          where name ='QLDA')
begin
    drop database QLDA
end
```