Chương 4
Ngôn ngữ truy vấn SQL

- Giới thiệu
- Các lệnh về kiến trúc CSDL
- Các lệnh cập nhật dữ liệu
- Các lệnh truy vấn dữ liệu
- Khung nhìn

Giới thiệu

- Ngôn ngữ khai báo
  - Cài đặt dựa trên ĐSQH
- Chuẩn hóa cho các HQTCSDL quan hệ
  - Được phát triển bởi IBM (1970s)
  - Các version standard ANSI/ISO
    - SQL-86
    - SQL-92
    - SQL-99

Giới thiệu (tt)

- SQL hỗ trợ
  - Ngôn ngữ Định nghĩa dữ liệu
    - Mức quan niệm: CREATE SCHEMA, TABLE,...
    - Mức ngoài: CREATE VIEW, GRANT,…
    - Mức trong: CREATE INDEX, CLUSTER,...
  - Ngôn ngữ Thao tác dữ liệu
    - Truy vấn: SELECT
    - Cập nhật: INSERT, DELETE, UPDATE
  - Ngôn ngữ khai báo
    - Ràng buộc toàn vẹn
    - Phân quyền và bảo mật
    - Điều khiển giao tác

Giới thiệu (tt)

- SQL sử dụng thuật ngữ
  - Bảng ~ quan hệ
  - Cột ~ thuộc tính
  - Dòng ~ bộ

Các lệnh về kiến trúc CSDL

- Tạo cấu trúc cho một bảng mới
- Thêm/xóa các cột của bảng
- Xóa một bảng

Tạo cấu trúc cho một bảng mới

❖ ***Cú pháp:***

CREATE TABLE <tên bảng>**(**<tên cột><kiểu dữ liệu>
[<kích thước>][not null],…

[**Constraint** <bí danh> **PRIMARY KEY** (<khóa chính>)]

[**Constraint** <bí danh> **UNIQUE** (<khóa>,…)]

[**Constraint** <bí danh> **FOREIGN KEY** (<khóa ngoại>)
REFERENCES <tên bảng>, …]

[**Constraint** <bí danh> **CHECK** <điều kiện ràng buộc>,
…]**)**

Tạo cấu trúc cho một bảng mới (tt)

Ví dụ: Tạo bảng **KHOA**(MSKHOA,TENKHOA,CNKHOA)

CREATE TABLE **KHOA**(MSKHOA char(10) not null,
TENKHOA char(20) not null, CNKHOA char(50),

Constraint p_KH PRIMARY KEY (MSKHOA),

Constraint u_KH UNIQUE (TENKHOA))

Tạo cấu trúc cho một bảng mới (tt)

Ví dụ: Tạo bảng **SINHVIEN**(MSSV, HOTENSV, NGSINH, GIOITINH, HOCBONG, MSKHOA)

CREATE TABLE **SINHVIEN**(MSSV char(10) not null, HOTENSV char(30), NGSINH datetime, GIOITINH char(3), HOCBONG decimal(6,1), MSKHOA char(10) not null,

Constraint p_SV PRIMARY KEY (MSSV),

Constraint f_SV_KH FOREIGN KEY (MSKHOA) REFERENCES KHOA,

Constraint ck_HB CHECK HOCBONG between 0 and 200000)

Thêm/xóa các cột của một bảng

❖Thêm cột

❖ ***Cú pháp:***

ALTER TABLE <tên bảng> **ADD** <tên cột> <kiểu dữ liệu>,…

Ví dụ: Thêm cột SOCMND vào bảng SINHVIEN

ALTER TABLE **SINHVIEN** ADD SOCMND char(9)

Ví dụ: Thêm cột SOCBGD vào bảng KHOA

ALTER TABLE **KHOA** ADD SOCBGD smallint

Constraint ck_SOCBGD CHECK (SOCBGD>0)

Thêm/xóa các cột của một bảng

❖Xóa cột

❖ ***Cú pháp:***

ALTER TABLE <tên bảng> **DROP** <tên cột>

Ví dụ: Xóa cột SOCMND khỏi bảng SINHVIEN

ALTER TABLE **SINHVIEN** DROP SOCMND

Thêm/xóa các cột của một bảng

❖Chú ý

- Thao tác xóa là không hợp lệ nếu cột cần xóa là thuộc tính khóa.
- Có thể sử dụng câu lệnh ALTER TABLE để thêm, xóa ràng buộc toàn vẹn.

Thêm ràng buộc toàn vẹn

ALTER TABLE <tên bảng> **ADD**
CONSTRAINT <tên_RBTV> <RBTV>,…

Ví dụ: Thêm RBTV khóa chính cho bảng SINHVIEN

ALTER TABLE **SINHVIEN** ADD
CONSTRAINT p_SV PRIMARY KEY (MSSV)

Ví dụ: Thêm RBTV khóa ngoại cho bảng SINHVIEN

ALTER TABLE **SINHVIEN** ADD
CONSTRAINT f_SV_KH FOREIGN KEY (MSKHOA)
REFERENCES KHOA

Xóa ràng buộc toàn vẹn

ALTER TABLE <tên bảng> **DROP**
CONSTRAINT <tên_RBTV>

Ví dụ: Xóa RBTV khóa ngoại khỏi bảng SINHVIEN

ALTER TABLE **SINHVIEN** DROP f_SV_KH

Xóa một bảng

❖ ***Cú pháp:***

DROP TABLE <tên bảng>

Ví dụ: Xóa bảng NHANVIEN

DROP TABLE **NHANVIEN**

❖ **Chú ý:** Không thể xóa một bảng được một khóa ngoại tham chiếu đến.

Các lệnh cập nhật dữ liệu

- Thêm dòng vào một bảng
- Xóa các dòng của một bảng
- Sửa đổi nội dung của các dòng trong một bảng

Thêm dòng vào một bảng

❖ Thêm một dòng

Cú pháp:

INSERT INTO <tên bảng> [(<danh sách các cột>)]
VALUES (<danh sách các giá trị>)

❖ Câu lệnh INSERT sẽ gặp lỗi nếu vi phạm RBTV:

- Khóa chính
- Tham chiếu
- NOT NULL - các cột có ràng buộc NOT NULL bắt buộc phải có giá trị

Thêm dòng vào một bảng

Ví dụ: Thêm một dòng mới vào bảng SINHVIEN

```
INSERT INTO SINHVIEN(MSSV, HOTENSV, NGSINH)
    VALUES ('sv01', 'Le Thu An', '20/3/1990')
INSERT INTO SINHVIEN(MSSV, HOTENSV, NGSINH)
    VALUES ('sv01', 'Le Thu An', Null)
INSERT INTO SINHVIEN
VALUES ('sv01', 'Le Thu An', '20/3/1990', 'Nữ', 180000, 'cntt')
```

Thêm dòng vào một bảng

❖ Thêm nhiều dòng

Cú pháp:

INSERT INTO <tên bảng> [(<danh sách các cột>)] <câu truy vấn con>

Thêm nhiều dòng

Ví dụ: Tạo danh sách sinh viên phải thi lại môn có mã số 'mh5'

```
CREATE TABLE SV_THILAI(MSSV char(10) not null,
    MSMH char(10) not null, DIEM float)
Constraint p_STL Primary key (MSSV,MSMH)

INSERT INTO SV_THILAI(MSSV, MSMH, DIEM)
    SELECT MSSV, MSMH, DIEM
    FROM KETQUA
    WHERE MSMH = 'mh5' and DIEM<5
```

Xóa các dòng của một bảng

Cú pháp:

DELETE FROM <tên bảng> [**WHERE** <điềukiện>]

Ví dụ: Xóa sinh viên có mã số 'sv10'

DELETE FROM **SINHVIEN** WHERE MSSV = 'sv10'

Ví dụ: Xóa những sinh viên học ở khoa có mã số 'kh3'

DELETE FROM **SINHVIEN** WHERE MSKHOA = 'kh3'

Ví dụ: Xóa tất cả các sinh viên

DELETE FROM **SINHVIEN**

Xóa các dòng của một bảng

- Lệnh DELETE có thể gây ra vi phạm ràng buộc tham chiếu:
  - Không cho xóa.
  - Xóa luôn những dòng có giá trịđang tham chiếu đến, dùng CASCADE.
  - Đặt NULL cho những giá trị tham chiếu.

Sửa đổi nội dung các dòng trong bảng

Cú pháp:

UPDATE <tên bảng> **SET** <tên cột>=<giá trị mới>, …
[**WHERE** <điềukiện>]

Ví dụ: Tăng học bổng lên 10% cho các sinh viên cư trú ở ngoài Tp.HCM

UPDATE **SINHVIEN** SET HOCBONG =HOCBONG*1.1

WHERE DIACHI <> 'Tp.HCM'

Sửa đổi nội dung các dòng trong bảng

- Lệnh UPDATE có thể gây ra vi phạm ràng buộc tham chiếu:
  - Không cho sửa
  - Sửa luôn những dòng có giá trị đang tham chiếu đến, dùng CASCADE
  - Đặt Null cho những giá trị tham chiếu.

Các lệnh truy vấn dữ liệu

- Câu lệnh truy vấn cơ bản
- Tên bí danh, mệnh đề WHERE, sử dụng *, distinct
- Phép toán tập hợp, câu truy vấn lồng, so sánh tập hợp
- Phép chia trong SQL
- Hàm kết hợp và gom nhóm
- Một số dạng truy vấn khác

Câu lệnh truy vấn cơ bản

❖ ***Cú pháp:***

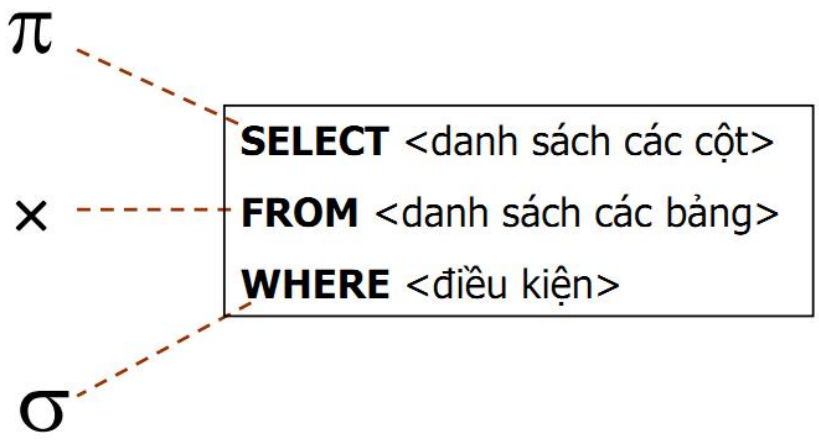
SELECT <danh sách các cột>

FROM <danh sách các bảng>

WHERE <điều kiện>

Câu lệnh truy vấn cơ bản (tt)

❖ SQL và ĐSQH:



SELECT L
FROM R ----------> $\pi_L(\sigma_C(R))$
WHERE C

Câu lệnh truy vấn cơ bản (tt)

Ví dụ: Cơ sở dữ liệu quản lý nhân viên (bài tập 2)

- Cho biết ngày sinh và địa chỉ của nhân viên có mã số '123456789'
  - ***SQL:***

```
SELECT  NGSINH, DCHI
FROM  NHANVIEN
WHERE  MANV = '123456789'
```

  - ***ĐSQH:***

$$\pi_{NGSINH,\ DCHI}\ (\sigma_{MANV='123456789'}\ (NHANVIEN))$$

Câu lệnh truy vấn cơ bản

- Cho biết mã số và họ tên của các nhân viên nam ở phòng số 5
  - **SQL:**

```
SELECT  MANV, HONV, TENLOT, TENNV
FROM  NHANVIEN
WHERE PHG=5 and  PHAI='Nam'
```

  - **ĐSQH:**

$$\pi_{MANV,HONV,TENLOT,TENNV}(\sigma_{PHG=5 \wedge PHAI='Nam'}(NHANVIEN))$$

Tên bí danh, mệnh đề WHERE

❖ Tên bí danh

- Câu truy vấn có sử dụng 2 cột có tên giống nhau thuộc 2 bảng khác nhau thì phải đặt tên bảng trước tên cột.
- Nếu 1 bảng sử dụng nhiều lần trong mệnh đề FROM thì phải sử dụng bí danh.

Tên bí danh, mệnh đề WHERE (tt)

Ví dụ: Tìm họ tên của từng nhân viên và người phụ trách trực tiếp của nhân viên đó.

```
SELECT  NV.HONV, NV.TENLOT, NV.TENNV,
   NVQL.HONV, NVQL.TENLOT, NVQL.TENNV
FROM  NHANVIEN   NV, NHANVIEN  NVQL
WHERE  NV.MA_NQL = NVQL.MANV
```

Tên bí danh, mệnh đề WHERE (tt)

SQL cho phép đổi tên bảng và tên cột, được thực hiện bằng mệnh đề AS (có thể dùng hay không tùy ý):

tên_cũ [AS] tên_mới

Ví dụ: Tìm họ tên và địa chỉ của nhân viên ở phòng 'Nghiên cứu'

```
SELECT  HONV+TENLOT+TENNV AS 'HO VA TEN',
        DCHI AS 'DIA CHI'
FROM  NHANVIEN, PHONGBAN
WHERE PHG=MAPHG and TENPHG = 'Nghien cuu'
```

Lưu ý trong mệnh đề WHERE

- Không có WHERE ⇔ WHERE TRUE

Ví dụ: Tìm mã số của tất cả các nhân viên

```
SELECT  MANV
FROM  NHANVIEN
```

- Trong mệnh đề FROM:
  - Hơn hai bảng mà không có điều kiện kết ⇔ phép tích Đềcác.

Ví dụ: Tìm tất cả các tổ hợp giữa mã số nhân viên (MANV) và tên phòng ban (TENPHG).

```
SELECT   MANV, TENPHG
FROM  NHANVIEN, PHONGBAN
```

Lưu ý trong mệnh đề WHERE (tt)

- Sử dụng BETWEEN…AND
  - Diễn đạt một giá trị biểu thức nằm trong một miền.
  - <biểu thức> **BETWEEN** <biểu thức> **AND** <biểu thức>

Ví dụ: Cho biết mã số và họ tên các nhân viên có mức lương từ 20000 đến 30000.

Lưu ý trong mệnh đề WHERE (tt)

Cách 1:

```
SELECT  MANV, HONV, TENLOT, TENNV
FROM  NHANVIEN
WHERE  LUONG >= 20000 AND LUONG <= 30000
```

Cách 2:

```
SELECT  MANV, HONV, TENLOT, TENNV
FROM  NHANVIEN
WHERE LUONG BETWEEN 20000 AND 30000
```

Lưu ý trong mệnh đề WHERE (tt)

Ví dụ: Cho biết mã số và họ tên các nhân viên có mức lương < 20000 hoặc >30000.

```
SELECT  MANV, TENNV
FROM  NHANVIEN
WHERE  LUONG NOT BETWEEN 20000 AND 30000
```

Lưu ý trong mệnh đề WHERE (tt)

❖ Giá trị NULL

Ví dụ: Cho biết các nhân viên không có người phụ trách trực tiếp.

```
SELECT  MANV, TENNV
FROM  NHANVIEN
WHERE  MA_NQL  IS  NULL
```

Ví dụ: Cho biết các nhân viên có người phụ trách trực tiếp.

```
SELECT  MANV, TENNV
FROM  NHANVIEN
WHERE  MA_NQL  IS  NOT  NULL
```

Sử dụng *

❖ Để lấy tất cả các cột, sử dụng ký tự dấu *

Ví dụ: Cho biết tất cả các thông tin về nhân viên của phòng số 5

```
SELECT  *
FROM  NHANVIEN
WHERE  PHONG = 5
```

Distinct

- SQL cho phép các bộ trùng nhau trong kết quả của câu truy vấn.
  - Thao tác loại bỏ các dòng trùng nhau có chi phí thời gian đáng kể.
  - Người sử dụng đôi khi muốn thấy những dòng trùng nhau trong bảng kết quả.
  - Khi áp dụng hàm kết hợp, tự động loại bỏ các dòng trùng nhau có thể cho kết quả SAI.

Distinct (tt)

- Để loại bỏ các dòng trùng nhau, thêm từ khóa **DISTINCT** sau SELECT.

Ví dụ: Tìm lương của các nhân viên, không loại bỏ dòng trùng nhau.

```
SELECT  LUONG
FROM  NHANVIEN
```

Ví dụ: Tìm lương của các nhân viên, loại bỏ dòng trùng nhau.

```
SELECT  distinct LUONG
FROM  NHANVIEN
```

Phép toán trên chuỗi

- SQL có một phép toán so sánh chuỗi để so sánh chuỗi ký tự.
  - Ký tự %: thay thế một chuỗi ký tự.
  - Ký tự _ : thay thế một ký tự.

Ví dụ: Cho biết họ tên các nhân viên có địa chỉ ở 'TP.HCM'

```
SELECT  HONV, TENLOT, TENNV
FROM  NHANVIEN
WHERE  DCHI  like  '%TP.HCM%'
```

Biểu thức số học

- Mệnh đề SELECT có thể chứa các biểu thức số học +, - , *, / cũng như các hàm trên các cột bao gồm các cột và các hằng số.

Ví dụ: Tìm mã nhân viên, tên nhân viên và lương tăng thêm 10% của các nhân viên được phân công đề án 'Sản phẩm X'

```
SELECT   MANV, TENNV, LUONG*1.1
FROM  NHANVIEN, PHANCONG, DEAN
WHERE  MANV = MA_NVIEN and SODA = MADA
      and TENDA = 'San pham X'
```

Sắp xếp khi hiển thị các dòng

- Mệnh đề **ORDER BY** dùng để sắp xếp các dòng trong kết quả câu truy vấn dựa trên giá trị của các thuộc tính.
- **Cú pháp:**

SELECT <danh sách các cột>

FROM <danh sách các bảng>

WHERE <điều kiện>

ORDER BY <danh sách các cột>

Sắp xếp khi hiển thị các dòng (tt)

Ví dụ: Cho biết mã nhân viên và số đề án mà họ tham gia, sắp xếp theo thứ tự giảm dần của mã nhân viên và tăng dần của số đề án.

```
SELECT  MA_NVIEN, SODA
FROM   PHANCONG
ORDER  BY MA_NVIEN DESC, SODA
```

Sắp xếp khi hiển thị các dòng (tt)

MA_NVIEN	SODA
999887777	10
999887777	30
987987987	10
987987987	30
987654321	10
987654321	20
987654321	30

Phép toán tập hợp

- Phép toán tập hợp gồm có hội (UNION), giao (INTERSECT), trừ (EXCEPT).
  - Điều kiện: khả hợp
- Thực hiện các phép toán này sẽ tự động loại bỏ **các dòng trùng**; để lấy các dòng trùng thì dùng **UNION ALL**, **INTERSECT ALL, EXCEPT ALL**

  Một dòng xuất hiện ***m*** lần trong r và ***n*** lần trong s thì xuất hiện:
  - **m+n** lần trong r UNION ALL s
  - **min(m,n)** lần trong EXCEPT ALL s

Phép toán tập hợp (tt)

- ***Phép hội:***

SELECT <ds cột> FROM <ds bảng> WHERE <điều kiện>

UNION [ALL]

SELECT <ds cột> FROM <ds bảng> WHERE <điều kiện>

- ***Phép giao:***

SELECT <ds cột> FROM <ds bảng> WHERE <điều kiện>

INTERSECT [ALL]

SELECT <ds cột> FROM <ds bảng> WHERE <điều kiện>

Phép toán tập hợp (tt)

➢ ***Phép trừ:***

SELECT <ds cột> FROM <ds bảng> WHERE <điều kiện>

EXCEPT [ALL]

SELECT <ds cột> FROM <ds bảng> WHERE <điều kiện>

❖ Có thể viết lại bằng câu truy vấn lồng.

Phép hội

Ví dụ: Danh sách những đề án (TENDA) mà nhân viên tham gia đề án có họ là 'Đinh' hoặc những đề án (TENDA) mà người trưởng phòng chủ trì đề án có họ là 'Đinh'.

```
SELECT  TENDA
FROM  DEAN, PHANCONG, NHANVIEN
WHERE  MADA=SODA and  MANV=MA_NVIEN and HONV=
    'Dinh'
UNION
SELECT  TENDA
FROM  DEAN, PHONGBAN, NHANVIEN
WHERE  PHONG=MAPHG and TRPHG=MANV and HONV=
    'Dinh'
```

Phép giao

Ví dụ: Tìm những nhân viên có người thân cùng tên và cùng giới tính

```
SELECT  TENNV, PHAI, MANV
FROM  NHANVIEN
INTERSECT
SELECT  TENTN, PHAI, MA_NVIEN
FROM  THANNHAN
```

Phép trừ

Ví dụ: Tìm những nhân viên không có thân nhân nào

```
SELECT  MANV
FROM  NHANVIEN
EXCEPT
SELECT  MA_NVIEN
FROM  THANNHAN
```

Câu truy vấn lồng

Ví dụ: Cho biết mã số và tên của các nhân viên ở phòng 'Nghiên cứu'.

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

Câu truy vấn lồng (tt)

❖ **Cú pháp:**

Câu truy vấn cha
(Outer query)

```
SELECT <danh sách các cột>
FROM <danh sách các bảng>
WHERE <so sánh tập hợp> (
        SELECT <danh sách các cột>
        FROM <danh sách các bảng>
        WHERE <điều kiện>)
```

Câu truy vấn con
(Subquery)

Câu truy vấn lồng (tt)

- Các câu lệnh SELECT có thể lồng nhau ở nhiều mức.
- Các câu truy vấn con trong cùng một mệnh đề WHERE được kết hợp bằng phép nối logic.
- Câu truy vấn con thường trả về một tập các giá trị.
- Mệnh đề WHERE của câu truy vấn cha:
  - <biểu thức> <so sánh tập hợp> <truy vấn con>
  - So sánh tập hợp thường đi cùng với một số toán tử:
    - [NOT] IN
    - ALL
    - ANY hoặc SOME
  - Kiểm tra sự tồn tại:
    - [NOT] EXISTS

Câu truy vấn lồng (tt)

❖ Truy vấn 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**.

❖ Truy vấn 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.

Câu truy vấn lồng (tt)

Ví dụ: Cho biết mã số và tên của các nhân viên ở phòng 'Nghiên cứu'.

Cách 1: Lồng tương quan (sử dụng EXISTS)

Cách 2: Lồng phân cấp (sử dụng IN)

Câu truy vấn lồng (tt)

Cách 1: SELECT MANV, TENNV
FROM NHANVIEN
WHERE **EXISTS** (SELECT *
FROM PHONGBAN
WHERE PHG=MAPHG
and TENPHG='Nghien cuu')

Cách 2: SELECT MANV, TENNV
FROM NHANVIEN
WHERE PHG **IN** (SELECT MAPHG
FROM PHONGBAN
WHERE TENPHG='Nghien cuu')

Câu truy vấn lồng (tt)

Ví dụ: Tìm những nhân viên không có thân nhân nào

Cách 1:

```
SELECT  *
FROM  NHANVIEN
WHERE  NOT EXISTS (SELECT *
                   FROM  THANNHAN
                   WHERE MANV=MA_NVIEN)
```

Cách 2:

```
SELECT *
FROM  NHANVIEN
WHERE  MANV  NOT IN (SELECT  MA_NVIEN
                     FROM  THANNHAN)
```

Câu truy vấn lồng (tt)

Ví dụ: Cho biết họ tên của nhân viên có mức lương lớn nhất trong từng phòng ban.

```
SELECT  PHG, HONV, TENNV
FROM  NHANVIEN  NV1
WHERE  LUONG >= ALL (SELECT LUONG
                     FROM  NHANVIEN  NV2
                     WHERE  NV1.PHG=NV2.PHG )
```

Nhận xét

- **IN**:
  - <tên cột> IN <câu truy vấn con>
  - Thuộc tính ở mệnh đề SELECT của truy vấn con phải có cùng kiểu dữ liệu với thuộc tính ở mệnh đề WHERE của truy vấn cha.
- **EXISTS**:
  - Không cần có thuộc tính, hằng số hay biểu thức nào khác đứng trước.
  - Không nhất thiết liệt kê tên thuộc tính ở mệnh đề SELECT của truy vấn con.
  - Những câu truy vấn có = ANY hay IN đều có thể chuyển thành câu truy vấn có EXISTS.

Nhận xét (tt)

❖ **Tập giá trị:**

- Vị trí của truy vấn con trong câu truy vấn lồng có thể được thay thế bằng một tập giá trị.

Ví dụ: Cho biết mã số của các nhân viên làm việc cho đề án 1, 2 hoặc 3.

Nhận xét (tt)

❖ ***Cách 1:***

```
SELECT  DISTINCT  MA_NVIEN
FROM  PHANCONG
WHERE  SODA  IN (1,2,3)
```

❖ ***Cách 2:***

```
SELECT  MA_NVIEN
FROM  PHANCONG
WHERE  SODA  IN (SELECT  MADA
                 FROM  DEAN
                 WHERE  MADA=1 or MADA=2 or MADA=3)
```

Phép chia trong SQL

R	A	B	C	D	E
	α	a	α	a	1
	α	a	γ	a	1
	α	a	γ	b	1
	β	a	γ	a	1
	β	a	γ	b	3
	γ	a	γ	a	1
	γ	a	γ	b	1
	γ	a	β	b	1

S	D	E
b_i	a	1
	b	1

R+S	A	B	C
a_i	α	a	γ
	γ	a	γ

Phép chia trong SQL (tt)

- R÷S là tập các giá trị a_i trong R sao cho không có giá trị b_i nào trong S làm cho bộ (a_i, b_i) không tồn tại trong R.
- Sử dụng NOT EXISTS để biểu diễn

Ví dụ: Tìm tên các nhân viên được phân công làm tất cả các đồ án

⇔ Tìm tên các nhân viên mà không có đề án nào là không được phân công làm.

- Tập bị chia: PHANCONG(MA_NVIEN, SODA)
- Tập chia: DEAN(MADA)
- Tập kết quả: KQ(MA_NVIEN)
- Kết KQ với NHANVIEN để lấy ra TENNV

Phép chia trong SQL (tt)

```
SELECT  NV.TENNV
FROM  NHANVIEN  NV, PHANCONG  PC1
WHERE  NV.MANV=PC1.MA_NVIEN AND
    NOT EXISTS (SELECT *
                FROM DEAN DA
               WHERE NOT EXISTS (SELECT *
                                 FROM PHANCONG PC2
                            WHERE PC2.SODA=DA.MADA
                    AND   PC1.MA_NVIEN=PC2.MA_NVIEN ))
```

Hàm kết hợp và gom nhóm

❖ Hàm kết hợp

- COUNT
  - COUNT(*) đếm số dòng
  - COUNT(<tên thuộc tính>) đếm số giá trị khác NULL của thuộc tính
  - COUNT(DISTINCT <tên thuộc tính>) đếm số giá trị khác nhau và khác NULL của thuộc tính
- MIN
- MAX
- SUM
- AVG
- Các hàm kết hợp được đặt ở mệnh đề SELECT hoặc HAVING

Hàm kết hợp và gom nhóm (tt)

Ví dụ: Tìm tổng lương, lương cao nhất, lương thấp nhất và lương trung bình của các nhân viên.

```
SELECT  sum(LUONG), max(LUONG), min(LUONG),
        avg(LUONG)
FROM  NHANVIEN
```

Ví dụ: Cho biết số lượng nhân viên của phòng 'Nghiên cứu'

```
SELECT  count(*)  AS  SL_NV
FROM   NHANVIEN, PHONGBAN
WHERE   PHG=MAPHG AND TENPHG='Nghien cuu'
```

Hàm kết hợp và gom nhóm (tt)

Ví dụ: Cho biết số lượng nhân viên của từng phòng ban.

?

Hàm kết hợp và gom nhóm (tt)

❖ **Gom nhóm**

- **Cú pháp:**

SELECT <danh sách các cột>

FROM <danh sách các bảng>

WHERE <điều kiện>

GROUP BY <danh sách các cột gom nhóm>

Sau khi gom nhóm mỗi nhóm các bộ sẽ có cùng giá trị tại các thuộc tính gom nhóm.

Hàm kết hợp và gom nhóm (tt)

Ví dụ: Cho biết số lượng nhân viên của từng phòng ban.

```
SELECT  PHG, count(*) AS SL_NV
FROM  NHANVIEN
GROUP  BY  PHG
```

Hoặc:

```
SELECT  TENPHG, count(*) AS SL_NV
FROM  NHANVIEN, PHONGBAN
WHERE  PHG=MAPHG
GROUP  BY  TENPHG
```

Hàm kết hợp và gom nhóm (tt)

Ví dụ: Với mỗi nhân viên cho biết mã số, họ tên, số lượng đề án và tổng thời gian mà họ tham gia.

```
SELECT  MA_NVIEN, count(*) AS  SL_DA,
            sum(THOIGIAN) AS TONG_TG
FROM  PHANCONG
GROUP BY MA_NVIEN
```

Hoặc:

```
SELECT  MA_NVIEN, HONV, TENNV, count(*) AS  SL_DA,
            sum(THOIGIAN) AS TONG_TG
FROM  NHANVIEN, PHANCONG
WHERE  MANV=MA_NVIEN
GROUP BY MA_NVIEN,HONV,TENNV
```

Hàm kết hợp và gom nhóm (tt)

Ví dụ: Cho biết những nhân viên tham gia từ 2 đề án trở lên.

?

Hàm kết hợp và gom nhóm (tt)

❖ **Điều kiện trên nhóm**

- **Cú pháp:**

SELECT <danh sách các cột>

FROM <danh sách các bảng>

WHERE <điều kiện>

GROUP BY <danh sách các cột gom nhóm>

HAVING <điều kiện trên nhóm>

Hàm kết hợp và gom nhóm (tt)

Ví dụ: Cho biết những nhân viên tham gia từ 2 đề án trở lên.

```
SELECT  MA_NVIEN
FROM  PHANCONG
GROUP  BY  MA_NVIEN
HAVING  count(*) >= 2
```

Hàm kết hợp và gom nhóm (tt)

Ví dụ: Cho biết những nhân viên (họ tên) có trên 2 thân nhân.

Cách 1:
```
SELECT  HONV+TENLOT+TENNV  AS HOTENNV
FROM  NHANVIEN
WHERE  (SELECT  count(*)
        FROM  THANNHAN
        WHERE  MANV=MA_NVIEN)>2
```

Cách 2:
```
SELECT  HONV+TENLOT+TENNV  AS HOTENNV
FROM  NHANVIEN  NV, THANNHAN  TN
WHERE  NV.MANV=TN.MA_NVIEN
GROUP BY  NV.MANV, HONV, TENLOT, TENNV
HAVING  count(TN.TENTN) > 2
```

Hàm kết hợp và gom nhóm (tt)

Ví dụ: Cho biết những phòng ban (TENPHG) có lương trung bình của các nhân viên lớn hơn 20000.

```
SELECT  PHG, avg(LUONG) AS LUONG_TB
FROM  NHANVIEN
GROUP  BY PHG
HAVING avg(LUONG) > 20000
```

Hoặc:

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

Hàm kết hợp và gom nhóm (tt)

Nhận xét:

- Mệnh đề GROUP BY:
  - Các thuộc tính trong mệnh đề **SELECT** (trừ những thuộc tính trong các hàm kết hợp) phải xuất hiện trong mệnh đề **GROUP BY.**
- Mệnh đề HAVING:
  - Sử dụng các hàm kết hợp trong mệnh đề SELECT để kiểm tra một số điều kiện nào đó.
  - Chỉ kiểm tra điều kiện trên nhóm, không là điều kiện lọc trên từng bộ.
  - Sau khi gom nhóm điều kiện trên nhóm mới được thực hiện.

Nhận xét (tt)

- Thứ tự thực hiện câu truy vấn có mệnh đề GROUP BY và HAVING:
  - (1)Chọn ra những dòng thỏa điều kiện trong mệnh đề WHERE.
  - (2)Những dòng này sẽ được gom thành nhiều nhóm tương ứng với mệnh đề GROUP BY.
  - (3)Áp dụng các hàm kết hợp cho mỗi nhóm.
  - (4)Bỏ qua những nhóm không thỏa điều kiện trong mệnh đề HAVING.
  - (5)Rút trích các giá trị của các cột và hàm kết hợp trong mệnh đề SELECT.

Ví dụ

❖ Tìm những phòng ban có lương trung bình cao nhất

```
SELECT  PHG, avg(LUONG) AS  LUONG_TB
FROM  NHANVIEN
GROUP  BY PHG
HAVING  avg(LUONG) = (SELECT MAX(AVG(LUONG))
                      FROM  NHANVIEN
                      GROUP  BY  PHG)
```

Viết đúng:

```
SELECT  PHG, avg(LUONG) AS  LUONG_TB
FROM  NHANVIEN
GROUP  BY PHG
HAVING  avg(LUONG) >= all (SELECT  avg(LUONG)
                           FROM  NHANVIEN
                           GROUP  BY  PHG)
```

Ví dụ

❖ Tìm tên các nhân viên được phân công làm tất cả các đề án

Sử dụng COUNT để biểu diễn phép chia:

Cho R(A,B), S(B), thực hiện R÷S

```
SELECT R.A
FROM R
[WHERE R.B IN (SELECT S.B FROM S   [WHERE <ĐK>]]
GROUP BY R.A
HAVING COUNT(DISTINCT R.B) = ( SELECT COUNT(S.B)
                               FROM S
                               [WHERE <ĐK>])
```

Ví dụ

❖ Tìm tên các nhân viên được phân công làm tất cả các đề án

```
SELECT  MANV, TENNV
FROM  NHANVIEN, PHANCONG
WHERE  MANV=MA_NVIEN
GROUP  BY MANV, TENNV
HAVING count(*) = (SELECT count(*)
                   FROM DEAN )
```

Một số dạng truy vấn khác

- Truy vấn con ở mệnh đề FROM
- Điều kiện kết ở mệnh đề FROM
- Cấu trúc CASE

Câu 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
- **Cú pháp:**

SELECT <danh sách các cột>

FROM R1, R2, (<truy vấn con>) **AS** tên_bảng

WHERE <điều kiện>

Câu truy vấn con ở mệnh đề FROM (tt)

Ví dụ: Cho biết những phòng ban (TENPHG) có lương trung bình của các nhân viên lớn hơn 20000.

Cách 1:

```
SELECT  MAPHG, TENPHG, avg(LUONG) AS LUONG_TB
FROM  NHANVIEN  NV, PHONGBAN PB
WHERE NV.PHG=PB.MAPHG
GROUP  BY MAPHG, TENPHG
HAVING avg(LUONG) > 20000
```

Câu truy vấn con ở mệnh đề FROM (tt)

Ví dụ: Cho biết những phòng ban (TENPHG) có lương trung bình của các nhân viên lớn hơn 20000.

Cách 2:

```
SELECT  MAPHG, TENPHG, avg(LUONG) AS LUONG_TB
FROM  PHONGBAN PB, (SELECT  PHG, avg(LUONG) AS
                                        LUONG_TB
                    FROM  NHANVIEN
                    GROUP  BY PHG
                HAVING avg(LUONG)>20000) AS  LUONG_NV
WHERE  PB.MAPHG=LUONG_NV.PHG
```

Điều kiện kết ở mệnh đề FROM

❖ **Kết bằng:**

SELECT <danh sách các cột>

FROM R1 [**INNER**] **JOIN** R2 **ON** <biểu thức>

WHERE <điều kiện>

❖ **Kết ngoài:**

SELECT <danh sách các cột>

FROM R1 **LEFT**|**RIGHT** [**OUTER**] **JOIN** R2 **ON** <biểu thức>

WHERE <điều kiện>

Điều kiện kết ở mệnh đề FROM (tt)

Ví dụ: Tìm mã và tên các nhân viên làm việc tại phòng 'Nghiên cứu'

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

Cách khác:

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

Điều kiện kết ở mệnh đề FROM (tt)

Ví dụ: Cho biết họ tên nhân viên và tên phòng ban mà họ là trưởng phòng nếu có.

```
SELECT  TENNV, HONV, TENPHG
FROM  NHANVIEN, PHONGBAN
WHERE  MANV=TRPHG
```

Viết đúng:

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

Điều kiện kết ở mệnh đề FROM (tt)

Ví dụ: Tìm họ tên các nhân viên và tên các đề án nhân viên tham gia nếu có.

?

Cấu trúc CASE

Cấu trúc CASE cho phép kiểm tra điều kiện và xuất thông tin theo từng trường hợp.

❖ **Cú pháp:**

CASE <tên cột>

WHEN <giá trị> **THEN** <biểu thức>

WHEN <giá trị> **THEN** <biểu thức>

…

[**ELSE** <biểu thức>]

END

Cấu trúc CASE

Ví dụ: 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 )
```

Cấu trúc CASE

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
```

Câu lệnh truy vấn tổng quát

SELECT <danh sách các cột>

FROM <danh sách các bảng>

[**WHERE** <điều kiện>]

[**GROUP BY** <các cột gom nhóm>]

[**HAVING** <điều kiện trên nhóm>]

[**ORDER BY** <các cột sắp thứ tự>]

Khung nhìn

❖Bảng là một quan hệ được **tổ chức lưu trữ** vật lý trong CSDL.

❖Khung nhìn cũng là một quan hệ:

- Không được lưu trữ vật lý (bảng ảo).
- Không chứa dữ liệu.
- Được định nghĩa từ những bảng khác.
- Có thể truy vấn hay cập nhật thông qua khung nhìn.

❖Tại sao phải sử dụng khung nhìn?

Khung nhìn (tt)

❖Tại sao phải sử dụng khung nhìn?

- Che dấu tính phức tạp của dữ liệu.
- Đơn giản hóa các câu truy vấn.
- Hiển thị dữ liệu dưới dạng tiện dụng nhất.
- An toàn dữ liệu.

Khung nhìn (tt)

- Định nghĩa khung nhìn
- Truy vấn trên khung nhìn
- Xóa khung nhìn
- Cập nhật dữ liệu trên khung nhìn

Định nghĩa khung nhìn

❖ Cú pháp:

CREATE VIEW <tên khung nhìn> [<danh sách cột>] **AS**

<câu truy vấn>

Bảng ảo này có:

- Danh sách thuộc tính trùng với các thuộc tính trong mệnh đề SELECT.
- Số dòng phụ thuộc vào điều kiện ở mệnh đề WHERE.
- Dữ liệu được lấy từ các bảng ở mệnh đề FROM.

Định nghĩa khung nhìn (tt)

Ví dụ: Cho biết họ tên nhân viên, tên đề án và thời gian nhân viên đó tham gia đề án

```
Create  View  PHANCONG1 as
    Select   HONV,TENLOT,TENNV,TENDA,THOIGIAN
    From NHANVIEN, DEAN, PHANCONG
    Where  MANV=MA_NVIEN and SODA=MADA
```

⇒**PHANCONG1**(HONV,TENLOT,TENNV,TENDA, THOIGIAN)

Định nghĩa khung nhìn (tt)

Ví dụ: Với mỗi phòng ban, cho biết tên phòng, số lượng nhân viên và tổng lương

```
Create View THONGKE_PB(TENP, SONV,LUONGTC)
    as
    Select TENPHG, count(NV.MANV), sum(LUONG)
    From NHANVIEN NV, PHONGBAN PB
    Where PHG=MAPHG
    Group By MAPHG,TENPHG
```

⇒ **THONGKE_PB**(TENP, SONV, LUONGTC)

Truy vấn trên khung nhìn

- Tuy không chứa dữ liệu nhưng có thể thực hiện các câu truy vấn trên khung nhìn.

Ví dụ: Tìm họ tên nhân viên làm việc trong đề án 'Sản phẩm X'.

```
SELECT  TENDA, HONV, TENLOT, TENNV
FROM  PHANCONG1
WHERE  TENDA= 'San pham X'
```

Xóa khung nhìn

DROP VIEW <tên khung nhìn>

Ví dụ: Xóa khung nhìn PHANCONG1

DROP VIEW PHANCONG1

Cập nhật dữ liệu trên khung nhìn

- Có thể dùng các câu lệnh INSERT, DELETE và UPDATE cho các **khung nhìn đơn giản**.
  - Khung nhìn đơn giản là khung nhìn được xây dựng trên 1 bảng và có khóa chính của bảng.
- Không thể cập nhật dữ liệu nếu:
  - Khung nhìn có dùng từ khóa DISTINCT.
  - Khung nhìn có sử dụng các hàm kết hợp.
  - Khung nhìn được xây dựng từ bảng có ràng buộc trên cột.
  - Khung nhìn được xây dựng từ nhiều bảng.

Thảo luận