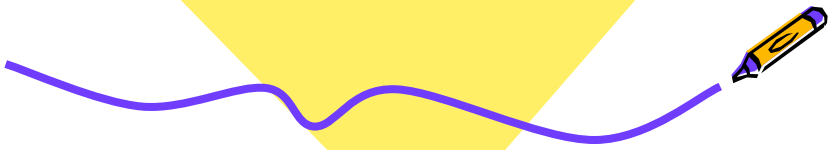
CƠ SỞ DỮ LIỆU

Chương 5 Ngôn ngữ SQL

GV: Phạm Thị Bạch Huệ
Email: *ptbhue@fit.hcmus.edu.vn*



Nội dung môn học

- Chương 1 Tổng quan về CSDL
- Chương 2 Mô hình ER
- Chương 3 Mô hình quan hệ
- Chương 4 Phép toán quan hệ
- Chương 5 Ngôn ngữ SQL
- Chương 6 Phép tính quan hệ
- Chương 7 Ràng buộc toàn vẹn
- Chương 8 Phụ thuộc hàm và dạng chuẩn

Mục tiêu chương

- Biết cách định nghĩa CSDL.
- Biết thao tác (tìm kiếm, thêm, xóa, sửa) trên cơ sở dữ liệu.

Lược đồ CSDL

1. NHANVIEN(MANV,HONV,TENLOT,TENNV,PHAI,LUONG, DIACHI, NGAYSINH, MA_NQL, PHG)
2. PHONGBAN (MAPB,TENPB,TRPHG,NGAYBĐ)
3. DIADIEM_PHG(MAPB, DIADIEM)
4. DEAN(MADA, TENDA, NGAYBD, PHONG, DIADIEM_DA)
5. PHANCONG (MANV, MADA, THOIGIAN)
6. THANNHAN(MANV,TENTN,PHAI,NGAYSINH,QUANHE)

Giới thiệu SQL

- SQL: Structured Query Language.
- SQL là ngôn ngữ chuẩn của nhiều HQT CSDL, gồm các câu lệnh định nghĩa dữ liệu, truy vấn và cập nhật dữ liệu.
- SQL sơ khai được gọi là SEQUEL (Structured English Query Language), do IBM phát triển trong hệ thống System R, 1974-1976.
- Gồm các phiên bản:
  - Chuẩn SQL-86 (SQL1) do ANSI (American National Standards Institute) và ISO (International Standards Organization).
  - Chuẩn SQL-92 (SQL2).
  - Chuẩn SQL-99 (SQL3).

Phân loại

- SQL gồm 2 nhóm câu lệnh:
  - DDL: Data Definition Language: tạo cấu trúc CSDL.
  - DML Data Manipulation Language: thao tác trên dữ liệu.
    - CREATE
    - SELECT
    - INSERT
    - UPDATE
    - DELETE

DDL

- SQL dùng:
  - Bảng ≡ Quan hệ.
  - Dòng ≡ Bộ.
  - Cột ≡ Thuộc tính.
- DDL dùng lệnh CREATE để:
  - Tạo lược đồ (scheme).
  - Tạo bảng (table).
  - Tạo khung nhìn (view).
  - Tạo ràng buộc toàn vẹn (assertion, trigger).

DDL - Kiểu dữ liệu

- Kiểu số:
  - Số nguyên: int, smallint.
  - Số thực: float, real, decimal, numeric.
- Chuỗi ký tự:
  - Char(n), varchar(n), text.
- Chuỗi bit:
  - Binary, varbinary, image.
- Boolean:
  - Bit.
- Ngày giờ:
  - Datetime.

CREATE TABLE

CREATE TABLE <Tên_bảng> (
<Tên_cột> <Kiểu_dữ_liệu> [<RBTV>],
<Tên_cột> <Kiểu_dữ_liệu> [<RBTV>],
…
[<RBTV>]
)

– Ví dụ:

```
CREATE TABLE PHONGBAN(
   MAPB              CHAR(5),
   TENPB             VARCHAR(30),
   TRPHG             CHAR(5),
   NGAYBĐ            DATETIME)
```

Các ràng buộc cơ bản

– Một số ràng buộc:
  - NOT NULL: chỉ định 1 cột không thể bằng NULL.
  - NULL.
  - Khóa chính.
  - Khóa ngoại.
  - UNIQUE: chỉ định 1 cột không nhận giá trị trùng.
  - DEFAULT: gán giá trị mặc định.
  - CHECK: kiểm tra một điều kiện nào đó.

– Đặt tên ràng buộc:

CONSTRAINT <Tên_RBTV> <RBTV>

Ví dụ

- Ví dụ:
```
CREATE TABLE NHANVIEN
    (
MANV              CHAR(5) PRIMARY KEY,
HONV              VARCHAR(30) NOT NULL,
TENLOT            VARCHAR (30) NOT NULL,
TENNV             VARCHAR(30) NOT NULL,
PHAI              CHAR(10) CHECK PHAI IN ('Nam', 'Nu'),
LUONG             INT DEFAULT (2000000),
DIACHI            VARCHAR (100),
NGAYSINH          DATETIME,
MA_NQL            CHAR(5),
PHG               CHAR(5)
FOREIGN KEY (MA_NQL) REFERENCES NHANVIEN (MANV),
FOREIGN KEY (PHG) REFERENCES PHONGBAN(MAPB))
)
```
-
```
CREATE TABLE PHONGBAN(
MAPB              CHAR(5) CONSTRAINT PK_PB PRIMARY KEY,
TENPB             VARCHAR(30),
TRPHG             CHAR(5),
NGAYBĐ            DATETIME
CONSTRAINT FK_PB FOREIGN KEY (TRPHG) REFERENCES NHANVIEN (MANV)
 )
```

ALTER TABLE

- Thay đổi cấu trúc hoặc ràng buộc của bảng.
- Gồm có: thêm/xóa/đổi kiểu dữ liệu cột, thêm/xóa ràng buộc.
- Lệnh thêm cột:

ALTER TABLE <Tên_bảng> ADD <Tên_cột> <Kiểu_dữ_liệu> [<RBTV>]

Ví dụ:

```
ALTER TABLE NHANVIEN ADD PHUCAP INT
```

– **Xóa cột:**

ALTER TABLE <Tên_bảng> DROP COLUMN <Tên_cột>

Ví dụ:

```
ALTER TABLE NHANVIEN DROP COLUMN PHUCAP
```

– **Thay đổi kiểu dữ liệu**

ALTER TABLE <Tên_bảng> **ALTER COLUMN** <Tên_cột> <Kiểu_dữ_liệu_mới>

Ví dụ:

```
ALTER TABLE NHANVIEN ALTER COLUMN PHUCAP
  FLOAT
```

– Thêm ràng buộc

ALTER TABLE <Tên_bảng> ADD
CONSTRAINT <Ten_RBTV> <RBTV>,
CONSTRAINT <Ten_RBTV> <RBTV>,
…

– Giả sử bảng NHANVIEN chưa khai báo khóa ngoại trên PHG:

```
ALTER TABLE NHANVIEN ADD CONSTRAINT
  FK_NV_PB FOREIGN KEY (PHG)
  REFERENCES PHONGBAN(MAPB)
```

– Xóa ràng buộc:

ALTER TABLE <Tên_bảng> DROP <Tên_RBTV>

– Xóa ràng buộc FK_NV

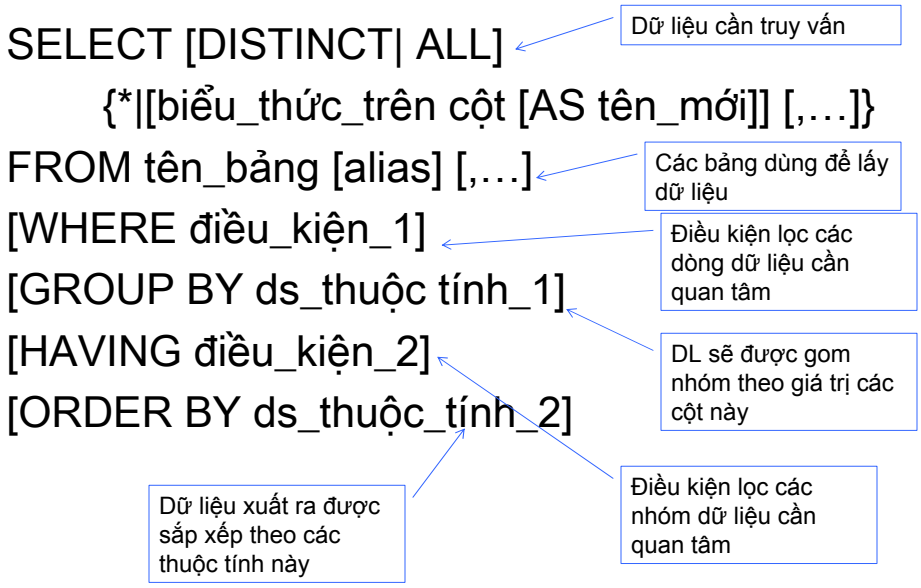
```
ALTER TABLE NHANVIEN DROP
  CONSTRAINT FK_NV_PB
```

– DROP TABLE <Tên_bảng>

– Ví dụ:

```
DROP TABLE NHANVIEN
```

Câu lệnh SQL tổng quát



Lưu ý

- Tối thiểu có SELECT-FROM, các mệnh đề còn lại cần dùng hay không phụ thuộc vào nhu cầu truy vấn dữ liệu.
- Thứ tự các mệnh đề trong câu truy vấn tổng quát không thể thay đổi.
- Không phụ thuộc chữ in hay thường.
- SQL là ngôn ngữ phi thủ tục, ta chỉ cần thể hiện:
  - cần dữ liệu gì,
  - ở đâu và
  - thỏa điều kiện gì.

Tìm tất cả dòng, tất cả cột

– Ví dụ: Cho danh sách tất cả các phòng ban.

```
SELECT MAPB, TENPB, TRPHG, NGAYBĐ
FROM PHONGBAN
```

– Dấu '*' đại diện cho tất cả các cột của 1 bảng.

Ví dụ trên có thể viết:

```
SELECT *
FROM PHONGBAN
```

Tìm tất cả dòng, vài cột

– Tương ứng với phép chiếu (Π) của ĐSQH.

– Ví dụ: Cho danh sách gồm mã phòng ban, tên nhân viên và lương.

```
SELECT PHG, HONV, TENLOT, TENNV,
  LUONG
FROM NHANVIEN
```

DISTINCT

– Ví dụ: Cho danh sách các đề án đã được phân công.

```
SELECT MADA
FROM PHANCONG
```

– Câu trên cho kết quả trùng. Để loại bỏ sự trùng lắp dữ liệu, ta viết:

```
SELECT DISTINCT MADA
FROM PHANCONG
```

Tính toán trên thuộc tính

– Ví dụ: Cho danh sách gồm có 3 cột: mã nhân viên, họ tên, lương nếu tăng 10% giá trị lương hiện tại.

```
SELECT MANV, HONV + ' ' + TENLOT +
  ' ' + TENNV, LUONG*1.1
FROM NHANVIEN
```

Bí danh

- Tương ứng với phép đổi tên thuộc tính trong ĐSQH.
- Kết quả ví dụ 4 cho ra các cột có tên khó hiểu, do cách đặt tên tự động của HQT CSDL đối với các thuộc tính có tính toán trên đó.
- Ta viết như sau:

```
SELECT HONV, HONV + ' ' + TENLOT +
  ' ' + TENNV AS HOTEN, LUONG*1.1
  AS LUONGMOI
FROM NHANVIEN
```

Tìm dữ liệu thỏa điều kiện

- Điều kiện được thành lập trên 1 thuộc tính. Có những kiểu điều kiện như sau:
  1. So sánh: =, <>, <, >, <=, >=.
  2. Miền.
  3. Tập hợp.
  4. Tìm chuỗi thỏa mẫu cho trước.
  5. Null.
- Điều kiện phức được thành lập dựa trên điều kiện đơn, bằng cách dùng các toán tử logic: AND, OR, NOT.

So sánh

– Ví dụ: Cho danh sách các nhân viên có lương nhiều hơn 2500000.

```
SELECT MANV, HONV, TENLOT, TENNV, LUONG
FROM NHANVIEN
WHERE LUONG > 2500000
```

– Ví dụ: Cho danh sách các đề án diễn ra ở HCM hoặc Đà Nẵng.

```
SELECT MADA, TENDA, DIADIEM_DA
FROM DEAN
WHERE DIADIEM_DA = 'HCM' OR DIADIEM =
  'Đà Nẵng'
```

Điều kiện liên quan đến miền

– Ví dụ: Cho danh sách các nhân viên có lương từ 3000000 đến 4000000.

```
SELECT MANV, HONV, TENLOT, TENNV, LUONG
FROM NHANVIEN
WHERE LUONG BETWEEN 300000 AND 4000000
```

Điều kiện liên quan đến tập hợp

– Ví dụ: Cho danh sách các đề án diễn ra ở HCM hoặc Đà Nẵng.

```
SELECT MADA, TENDA, DIADIEM_DA
FROM DEAN
WHERE DIADIEM_DA IN ('HCM', 'Đà
  Nẵng')
```

Tìm chuỗi

1. % : chuỗi bất kỳ, có thể rỗng.
2. _ : ký tự đơn bất kỳ.
3. DIACHI LIKE 'H%': địa chỉ bắt đầu bởi chữ H.
4. DIACHI LIKE 'H_ _ ': địa chỉ có đúng 3 ký tự, bắt đầu bởi H.
5. DIACHI LIKE '%e': địa chỉ là chuỗi bất kỳ kết thúc bởi ký tự e.
6. DIACHI NOT LIKE 'H%': địa chỉ không bắt đầu bởi H.

– Ví dụ: Cho danh sách các nhân viên ở Tp. HCM.

```
SELECT MANV, HONV, TENLOT, TENNV,
  DIACHI
FROM NHANVIEN
WHERE DIACHI LIKE '%Tp. HCM%'
```

Điều kiện liên quan giá trị Null

– Ví dụ: Cho danh sách các nhân viên chưa được bố trí phòng.

```
SELECT *
FROM NHANVIEN
WHERE PHG IS NULL
```

Sắp xếp dựa trên 1 cột

– Từ khóa theo sau thuộc tính dùng để sắp xếp: ASC (sắp tăng, mặc định), DESC (sắp giảm).

Ví dụ: Cho danh sách nhân viên sắp theo mã phòng.

```
SELECT *
FROM NHANVIEN
ORDER BY PHG
```

Sắp xếp dựa trên nhiều cột

Ví dụ: Cho danh sách nhân viên sắp tăng dần theo mã phòng, đối với từng phòng sắp theo thứ tự lương giảm dần.

```
SELECT *
FROM NHANVIEN
ORDER BY PHG, LUONG DESC
```

Hàm tính toán

- Count:đếm số giá trị khác null của trường đối số.
- Sum: tính tổng các giá trị của trường đối số.
- Avg: tính giá trị trung bình của trường đối số.
- Min: trả về giá trị nhỏ nhất trên trường đối số.
- Max: trả về giá trị lớn nhất trên trường đối số.
- Đặc điểm:
  - Nhận đối số là 1 trường và trả về 1 giá trị.
  - Count, min, max áp dụng cho trường kiểu số lẫn kiểu không phải là số.
  - Sum, avg chỉ áp dụng trên trường kiểu số.

Hàm tính toán

- Các hàm tính toán chỉ thao tác trên các giá trị khác null, trừ count (*).
- Count(*) đếm số dòng của 1 bảng, dù dòng đó có giá trị null hay giá trị trùng.
- DISTINCT dùng để loại bỏ sự trùng lặp trước khi vận dụng hàm, nhưng DISTINCT không có tác dụng đối với min, max.

Hàm tính toán

- Nếu câu SELECT có dùng hàm tính toán và không có mệnh đề GROUP BY thì không được liệt kê ở mệnh đề SELECT các thuộc tính không phải là đối số của hàm tính toán đã dùng.
- Ví dụ: Câu sau đây SAI:

```
SELECT PHG, COUNT(LUONG)
FROM NHANVIEN
```

Count()

- Ví dụ: Cho biết có tất cả bao nhiêu nhân viên.

```
SELECT COUNT (*)
FROM NHANVIEN
```

- Ví dụ: Cho biết có bao nhiêu nhân viên có lương lớn hơn 3000000.

```
SELECT COUNT(*)
FROM NHANVIEN
WHERE LUONG > 3000000
```

Count DISTINCT

– Có bao nhiêu đề án đã được phân công.

Câu SAI:

```
SELECT COUNT (MADA)
FROM PHANCONG
```

Câu đúng:

```
SELECT COUNT(DISTINCT MADA)
FROM PHANCONG
```

– Có bao nhiêu nhân viên thuộc phòng số 5 và tổng lương của họ.

```
SELECT COUNT (*), SUM(LUONG)
FROM NHANVIEN
WHERE PHG = 5
```

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

```
SELECT MIN (LUONG) AS THAPNHAT, MAX
  (LUONG) AS CAONHAT, AVG(LUONG) AS
  TRUNGBINH
FROM NHANVIEN
```

Group by

- GROUP BY được dùng để tạo ra các nhóm dữ liệu trước khi vận dụng hàm.
- Các thuộc tính sau mệnh đề GROUP BY gọi là thuộc tính gom nhóm.
  - Hàm sẽ được thực hiện trên từng nhóm khi câu truy vấn có mệnh đề GROUP BY.
  - Mỗi thuộc tính liệt kê sau SELECT sẽ có 1 giá trị đối với từng nhóm.
  - Tất cả các thuộc tính sau SELECT phải xuất hiện ở mệnh đề GROUP BY (trừ thuộc tính mang giá trị là hàm).
  - Có thể có các thuộc tính xuất hiện ở mệnh đề GROUP BY nhưng không xuất hiện sau SELECT.
  - Hai dòng mang giá trị null trên thuộc tính gom nhóm sẽ được gom thành cùng một nhóm.
  - Thứ tự thực hiện: (1) điều kiện sau WHERE (2) GROUP BY (3) hàm tính toán trên nhóm (4) điều kiền sau HAVING.

Group by

- Ví dụ: Cho biết mỗi phòng ban có bao nhiêu nhân viên và tổng lương của các nhân viên trong từng phòng.

```
SELECT PHG, COUNT(*), SUM (LUONG)AS TONG
FROM NHANVIEN
GROUP BY PHG
```

- Ví dụ: Cho biết lương trung bình của nhân viên nam và nhân viên nữ trong phòng số 5.

```
SELECT PHAI, AVG(LUONG)AS TRUNGBINH
FROM NHANVIEN
WHERE PHG = 5
GROUP BY PHAI
```

Having

– Ví dụ: Cho danh sách các phòng ban có lương trung bình của các nhân viên nam lớn hơn 4000000

```
SELECT PHG, AVG (LUONG)
FROM NHANVIEN
WHERE PHAI = 'Nam'
GROUP BY PHG
HAVING AVG (LUONG) > 4000000
```

Câu truy vấn con

– Là câu truy vấn xuất hiện trong một câu truy vấn khác. Kết quả của câu truy vấn con sẽ được dùng cho mệnh đề SELECT khác.
– Một câu truy vấn con có thể được dùng trong các mệnh đề: WHERE, HAVING, INSERT, UPDATE, DELETE.
– Câu truy vấn con có thể trả về:
  - Một giá trị, tức một dòng một cột.
  - Nhiều dòng một cột.
  - Nhiều dòng nhiều cột.

Câu truy vấn con

– Ví dụ: Cho danh sách các nhân viên thuộc phòng ban tên là ‘Nghiên cứu’

```
SELECT MANV, HONV, TENLOT, TENNV
FROM NHANVIEN
WHERE PHG =(SELECT MAPB
            FROM PHONGBAN
            WHERE TENPB = ‘Nghien
 cuu’)
```

Câu truy vấn con

– Có thể dùng câu truy vấn con sau một toán tử so sánh ở mệnh đề WHERE hoặc HAVING.

– Ví dụ: Cho danh sách các nhân viên có lương lớn hơn lương trung bình của toàn bộ nhân viên.

```
SELECT MANV,HONV,TENLOT,TENNV
FROM NHANVIEN
WHERE LUONG > (SELECT AVG(LUONG)
                FROM NHANVIEN)
```

Câu truy vấn con - IN

– Ví dụ: Cho biết danh sách các nhân viên có tham gia đề án.

```
SELECT MANV, HONV, TENLOT, TENNV
FROM NHANVIEN
WHERE MANV IN (SELECT MANV
FROM PHANCONG)
```

ANY & ALL

– Nếu câu truy vấn con cho kết quả rỗng thì mệnh đề ALL có giá trị TRUE còn mệnh đề ANY có giá trị FALSE.

– Chuẩn ISO dùng SOME tương đương với ANY.

– Ví dụ: Cho biết nhân viên nào có lương lớn hơn ít nhất giá trị lương bất kỳ của một nhân viên thuộc phòng số 5.

```
SELECT MANV, HONV, TENLOT, TENNV
FROM NHANVIEN
WHERE LUONG > SOME (SELECT LUONG
FROM NHANVIEN
WHERE PHG = 5)
```

– Ví dụ: Cho biết nhân viên nào có lương lớn hơn tất cả giá trị lương của các nhân viên thuộc phòng số 5.

```
SELECT MANV, HONV, TENLOT, TENNV
FROM NHANVIEN
WHERE LUONG > ALL (SELECT LUONG
FROM NHANVIEN
WHERE PHG = 5)
```

Truy vấn từ nhiều bảng

– Ta có thể truy xuất dữ liệu từ nhiều bảng.
– Ví dụ: Cho danh sách các nhân viên thuộc phòng ban tên là ‘Nghiên cứu’.

```
SELECT MANV, HONV, TENLOT, TENNV
FROM NHANVIEN N, PHONGBAN P
WHERE N.PHG = P.MAPB AND TENPB =
  ‘Nghien cuu’
```

Kết trái (Left join)

–Ví dụ: Cho biết tên các nhân viên và mã đề án mà nhân viên đó có tham gia, những ai không có tham gia đề án thì thông tin đề án là NULL.

```
SELECT MANV, TENNV, MADA
FROM NHANVIEN NV LEFT JOIN PHANCONG
PC ON NV.MANV = PC.MANV
```

Tương tự có kết phải (Right join), và kết ngoài (full outer join). Kết quả của phép kết ngoài là hội của kết quả phép kết trái và kết phải.

Exists, not exists

– Được dùng trong câu truy vấn con, EXISTS trả về TRUE nếu kết quả câu truy vấn con có ít nhất 1 dòng.

– Ví dụ: Cho danh sách các nhân viên có tham gia đề án.

```
SELECT *
FROM NHANVIEN NV
WHERE EXISTS (SELECT * FROM
  PHANCONG WHERE MANV = NV.MANV)
```

– Ví dụ: Cho danh sách các nhân viên không có tham gia đề án.

```
SELECT *
FROM NHANVIEN NV
WHERE NOT EXISTS (SELECT * FROM
  PHANCONG WHERE MANV = NV.MANV)
```

Hội

– Ví dụ: Cho danh sách các nhân viên có tham gia đề án tên ‘X’ hoặc ‘Y’.

```
SELECT MANV, TENNV
FROM NHANVIEN NV, PHANCONG PC, DEAN
  DA
WHERE NV.MANV = PC.MANV AND
PC.MADA = DA.MADA AND (TENDA = ‘X’
  OR TENDA = ‘Y’)
```

Giao

– Ví dụ: Cho danh sách các nhân viên vừa tham gia đề án tên 'X' vừa tham gia đề án tên 'Y'.

```
SELECT MANV, TENNV
FROM NHANVIEN
WHERE MANV IN (SELECT MANV FROM PHANCONG
  PC1, DEAN DA1 WHERE PC1.MADA =
  DA1.MADA AND TENDA = 'X')
AND MANV IN (SELECT MANV FROM PHANCONG
  PC2, DEAN DA2 WHERE PC2.MADA =
  DA2.MADA AND TENDA = 'Y')
```

Hiệu

– Ví dụ: Cho danh sách các nhân viên có tham gia đề án tên 'X' nhưng không có tham gia đề án tên 'Y'.

```
SELECT MANV, TENNV
FROM NHANVIEN
WHERE MANV IN (SELECT MANV FROM PHANCONG
  PC1, DEAN DA1 WHERE PC1.MADA =
  DA1.MADA AND TENDA = 'X')
AND MANV NOT IN (SELECT MANV FROM
  PHANCONG PC2, DEAN DA2 WHERE PC2.MADA
  = DA2.MADA AND TENDA = 'Y')
```

Insert

– Ví dụ: Phân công nhân viên mã 001 làm đề án mã là DAX trong thời gian 10 giờ.

```
Insert into PHANCONG values ('001',
  'DAX', 10)
```

– Ví dụ: Phân công nhân viên mã 001 làm tất cả các đề án do phòng số 5 chủ trì.

```
Insert into PHANCONG (SELECT 001,
  MADA, NULL FROM DEAN WHERE PHONG
  = 5)
```

Update

– Ví dụ: Cập nhật lương của các trưởng phòng tăng 10%.

```
UPDATE NHANVIEN
SET LUONG = LUONG*1.1
WHERE MANV IN (SELECT TRPHG FROM
  PHONGBAN)
```

Delete

– Ví dụ: Xóa các phân công liên quan đến nhân viên mã là ‘001’

```
DELETE PHANCONG WHERE MANV = ‘001’
```