

## **Chương 6:**

# **Thao tác dữ liệu với ngôn ngữ SQL**

---

**Tham khảo tài liệu [B]:**

**Chapter 5: SQL: Data Manipulation**

## Nội dung chương 6:

---

- ◆ Mục đích và tầm quan trọng của SQL
- ◆ Truy xuất DL từ CSDL với lệnh SELECT và:
  - Các điều kiện tại mệnh đề WHERE
  - Sắp xếp kết quả nhờ ORDER BY
  - Các hàm thống kê
  - Gom nhóm DL nhờ GROUP BY và HAVING
  - Các truy vấn con (Subqueries)

## Nội dung chương 6: (tt)

---

- Kết các bảng với nhau
- Thực hiện các phép toán trên tập hợp (**UNION, INTERSECT, EXCEPT**).
- ◆ Cập nhật DL dùng **INSERT, UPDATE, và DELETE**.

# Mục tiêu của ngôn ngữ SQL

---

- ◆ Ngôn ngữ SQL có 2 phần chính:
  - Ngôn ngữ DDL cho việc định nghĩa cấu trúc CSDL
  - Ngôn ngữ DML cho việc truy xuất và cập nhật dữ liệu
- ◆ Ngôn ngữ SQL có định dạng mềm dẻo
- ◆ Ngôn ngữ SQL thuộc loại ngôn ngữ phi thủ tục (chỉ cần miêu tả CÁI GÌ chúng ta muốn)

# Mục tiêu của ngôn ngữ SQL

---

- ◆ Ngôn ngữ SQL có thể dùng cho các loại người sử dụng khác nhau như người quản trị CSDL, người phát triển ứng dụng, và các loại người dùng khác
- ◆ Hiện nay, ngôn ngữ SQL đã đăng ký chuẩn ISO nên có tính chuẩn mực cao và trở thành một ngôn ngữ chính thống cho CSDL quan hệ.

# Viết lệnh SQL

---

- ◆ SQL gồm các từ khóa và các từ do user định nghĩa
  - Từ khóa (Reserved words) là các từ qui định sẵn của ngôn ngữ SQL
  - Từ do user định nghĩa (User-defined words) như tên quan hệ, tên cột, tên view, ...

# Viết lệnh SQL

---

- ◆ Hầu hết các từ trong câu lệnh SQL không phân biệt chữ thường/chữ HOA, trừ một số hằng ký tự
- ◆ Để dễ đọc, nên viết câu lệnh SQL theo cách có canh lề và xuống dòng hợp lý:
  - Mỗi mệnh đề nên viết một dòng
  - Dóng thẳng hàng lề trái các mệnh đề khi xuống dòng
  - Nếu một mệnh đề có nhiều phần thì mỗi phần nên viết trên một dòng

# Các giá trị hằng

---

- ◆ Các giá trị hằng không phải là số phải nằm trong dấu nháy đơn (vd. 'London').
- ◆ Các giá trị hằng là số không nằm trong dấu nháy nào cả (vd. 650.00).



# Câu lệnh SELECT

---

**SELECT [DISTINCT | ALL]**

**{ \* | [BiểuThứcCột [AS TênMới]] [,...] }**

**FROM           TênBảng [BíDanh] [, ...]**

**[WHERE       ĐiềuKiện]**

**[GROUP BY   DanhSáchCột]**

**[HAVING   ĐiềuKiệnChoNhóm]**

**[ORDER BY   DanhSáchCột]**

# Câu lệnh SELECT

---

**FROM**

Các bảng được dùng

**WHERE**

Điều kiện lọc các hàng dữ liệu

**GROUP BY**

Tạo nhóm các hàng có giá trị giống nhau ở 1 hay nhiều cột

**HAVING**

Điều kiện lọc để lấy 1 số nhóm

**SELECT**

Các cột muốn có trong kết quả

**ORDER BY**

Sắp thứ tự các hàng trong kết quả

## Ví dụ 6.1

---

Lập danh sách tất cả các nhân viên:

```
SELECT staffNo, fName, lName, address,  
       position, sex, DOB, salary, branchNo  
FROM Staff;
```

◆ Có thể dùng dấu \* để miêu tả “lấy tất cả các cột:

```
SELECT *  
FROM Staff;
```

## Ví dụ 6.1: lấy tất cả các cột, tất cả các hàng

**Table 5.1** Result table for Example 5.1.

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000.00	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000.00	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000.00	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000.00	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000.00	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000.00	B005

## Ví dụ 6.2: Lấy 1 số cột

---

**SELECT staffNo, fName, lName, salary  
FROM Staff;**

**Table 5.2** Result table for Example 5.2.

staffNo	fName	lName	salary
SL21	John	White	30000.00
SG37	Ann	Beech	12000.00
SG14	David	Ford	18000.00
SA9	Mary	Howe	9000.00
SG5	Susan	Brand	24000.00
SL41	Julie	Lee	9000.00

## Ví dụ 6.3 Không sử dụng DISTINCT

---

**SELECT propertyNo  
FROM Viewing;**

propertyNo
PA14
PG4
PG4
PA14
PG36

## Ví dụ 6.3 Có sử dụng DISTINCT

---

- ◆ Dùng DISTINCT để loại bỏ dl trùng lặp:

```
SELECT DISTINCT propertyNo  
FROM Viewing;
```

propertyNo
PA14
PG4
PG36

## Ví dụ 6.4 Dùng các field tính toán

---

**SELECT** staffNo, fName, lName, salary/12  
**FROM** Staff;

**Table 5.4** Result table for Example 5.4.

staffNo	fName	lName	col4
SL21	John	White	2500.00
SG37	Ann	Beech	1000.00
SG14	David	Ford	1500.00
SA9	Mary	Howe	750.00
SG5	Susan	Brand	2000.00
SL41	Julie	Lee	750.00



## Ví dụ 6.4 Dùng các field tính toán

---

### ◆ Dùng AS TênFieldMới:

```
SELECT staffNo, fName, lName, salary/12  
      AS monthlySalary  
FROM Staff;
```

## Ví dụ 6.5

---

**SELECT** staffNo, fName, lName, position, salary  
**FROM** Staff  
**WHERE** salary > 10000;

**Table 5.5** Result table for Example 5.5.

staffNo	fName	lName	position	salary
SL21	John	White	Manager	30000.00
SG37	Ann	Beech	Assistant	12000.00
SG14	David	Ford	Supervisor	18000.00
SG5	Susan	Brand	Manager	24000.00

## Ví dụ 6.6

---

```
SELECT *  
FROM Branch  
WHERE city = 'London' OR city = 'Glasgow';
```

**Table 5.6** Result table for Example 5.6.

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B003	163 Main St	Glasgow	G11 9QX
B002	56 Clover Dr	London	NW10 6EU

## Ví dụ 6.7

---

**SELECT** staffNo, fName, lName, position, salary  
**FROM** Staff  
**WHERE** salary **BETWEEN** 20000 **AND** 30000;

**Table 5.7** Result table for Example 5.7.

staffNo	fName	lName	position	salary
SL21	John	White	Manager	30000.00
SG5	Susan	Brand	Manager	24000.00

## Ví dụ 6.7

---

### ◆ Thay thế BETWEEN

```
SELECT staffNo, fName, lName, position, salary  
FROM Staff  
WHERE salary >= 20000 AND salary <= 30000;
```

## Ví dụ 6.8 Toán tử IN

---

```
SELECT staffNo, fName, lName, position  
FROM Staff  
WHERE position IN ('Manager', 'Supervisor');
```

**Table 5.8** Result table for Example 5.8.

staffNo	fName	lName	position
SL21	John	White	Manager
SG14	David	Ford	Supervisor
SG5	Susan	Brand	Manager

## Ví dụ 6.8 Toán tử IN

---

### ◆ Thay thế IN

```
SELECT staffNo, fName, lName, position  
FROM Staff  
WHERE position='Manager' OR  
       position='Supervisor';
```

## Ví dụ 6.9 Toán tử LIKE

```
SELECT clientNo, fName, lName, address, telNo  
FROM PrivateOwner  
WHERE address LIKE '%Glasgow%';
```

**Table 5.9** Result table for Example 5.9.

ownerNo	fName	lName	address	telNo
CO87	Carol	Farrel	6 Achray St, Glasgow G32 9DX	0141-357-7419
CO40	Tina	Murphy	63 Well St, Glasgow G42	0141-943-1728
CO93	Tony	Shaw	12 Park Pl, Glasgow G4 0QR	0141-225-7025



## Ví dụ 6.9 Toán tử LIKE

---

- ◆ SQL có 2 biểu tượng thay thế trong mẫu:
  - %: chuỗi từ 0 đến nhiều ký tự
  - \_ (gạch dưới): đại diện 1 ký tự

## Ví dụ 6.10 dùng giá trị NULL

---

Liệt kê thông tin việc xem tài sản PG4 nếu khi xem nó khách hàng không cho ý kiến

```
SELECT clientNo, viewDate  
FROM Viewing  
WHERE propertyNo = 'PG4' AND  
comment IS NULL;
```

## Ví dụ 6.10 dùng giá trị NULL

---

**Table 5.10** Result table for Example 5.10.

clientNo	viewDate
CR56	26-May-01

- ◆ Có thể dùng (IS NOT NULL) cho điều kiện ngược lại

## Ví dụ 6.11 Sắp xếp dữ liệu của kết quả

---

**SELECT** staffNo, fName, lName, salary  
**FROM** Staff  
**ORDER BY** salary **DESC**;

**Table 5.11** Result table for Example 5.11.

staffNo	fName	lName	salary
SL21	John	White	30000.00
SG5	Susan	Brand	24000.00
SG14	David	Ford	18000.00
SG37	Ann	Beech	12000.00
SA9	Mary	Howe	9000.00
SL41	Julie	Lee	9000.00

## Ví dụ 6.12 Sắp xếp dl kết quả trên nhiều cột

---

**SELECT** propertyNo, type, rooms, rent  
**FROM** PropertyForRent  
**ORDER BY** type, rent **DESC**;

**Table 5.12(b)** Result table for Example 5.12 with two sort keys.

propertyNo	type	rooms	rent
PG16	Flat	4	450
PL94	Flat	4	400
PG36	Flat	3	375
PG4	Flat	3	350
PA14	House	6	650
PG21	House	5	600

# Các hàm thống kê trong lệnh SELECT

---

- ◆ SQL chuẩn ISO có 5 hàm thống kê:  
(dùng sau SELECT và HAVING)

**COUNT**      đếm

**SUM**          tính tổng

**AVG**          tính trung bình cộng

**MIN**          lấy giá trị nhỏ nhất

**MAX**          lấy giá trị lớn nhất

## Các hàm thống kê trong lệnh SELECT

---

- ◆ COUNT, MIN và MAX dùng cho cả các giá trị số và không phải số
- ◆ SUM và AVG chỉ dùng cho cả các giá trị số
- ◆ Trừ hàm COUNT(\*), các hàm khác loại bỏ giá trị NULL trước rồi mới tính kết quả
- ◆ COUNT(\*) đếm tất cả các hàng kể cả giá trị NULL và trùng lặp
- ◆ Muốn loại bỏ trùng lặp ta dùng DISTINCT

## Các hàm thống kê trong lệnh SELECT

---

- ◆ Các hàm thống kê chỉ dùng sau SELECT, HAVING và ORDER BY
- ◆ Nếu sau SELECT có dùng hàm thống kê thì thường phải có GROUP BY. Lệnh sau là sai:

```
SELECT staffNo, COUNT(salary)  
FROM Staff;
```



## Ví dụ 6.13 Dùng hàm COUNT(\*)

---

```
SELECT COUNT(*) AS count  
FROM PropertyForRent  
WHERE rent > 350;
```

**Table 5.13** Result table for Example 5.13.

count
5

## Ví dụ 6.14 Dùng COUNT(DISTINCT)

---

Có bao nhiêu nhà khác nhau được xem trong tháng 5 năm 2001?

```
SELECT COUNT(DISTINCT propertyNo) AS count  
FROM Viewing  
WHERE date BETWEEN '1-May-01'  
AND '31-May-01';
```

Table 5.14 Result table for Example 5.14.

count
2

## Ví dụ 6.15 Dùng COUNT và SUM

---

Tìm số lượng các giám đốc và tổng số lương của họ?

```
SELECT COUNT(staffNo) AS count,  
        SUM(salary) AS sum  
FROM Staff  
WHERE position = 'Manager';
```

**Table 5.15** Result table for Example 5.15.

count	sum
2	54000.00

## Ví dụ 6.16 Dùng MIN, MAX, AVG

---

```
SELECT MIN(salary) AS min,  
        MAX(salary) AS max,  
        AVG(salary) AS avg  
FROM Staff;
```

**Table 5.16** Result table for Example 5.16.

min	max	avg
9000.00	30000.00	17000.00

## Lệnh SELECT có GROUP BY

---

- ◆ Dùng mệnh đề GROUP BY để lấy các giá trị thống kê theo từng nhóm dl (1 nhóm là 1 số hàng)
- ◆ Nội dung sau SELECT và GROUP BY có liên quan mật thiết với nhau: mỗi thành phần sau SELECT phải là 1 giá trị đơn trên từng nhóm dl, sau SELECT có thể là:
  - Tên các cột
  - Các hàm thống kê
  - Các hằng số
  - Biểu thức cấu thành từ các phần vừa kể trên

## Lệnh SELECT có GROUP BY

---

- ◆ Tất cả các tên cột trong SELECT phải xuất hiện trong GROUP BY trừ khi tên cột chỉ dùng trong các hàm thống kê.
- ◆ Nếu WHERE được dùng với GROUP BY, WHERE được áp dụng trước, sau đó mới hình thành các group từ các hàng dữ liệu đã thỏa mãn điều kiện WHERE.

## Ví dụ 6.17 Dùng GROUP BY

---

**Tìm số nhân viên ở mỗi chi nhánh và tổng số lương của họ?**

```
SELECT    branchNo,  
          COUNT(staffNo) AS count,  
          SUM(salary) AS sum  
FROM Staff  
GROUP BY branchNo  
ORDER BY branchNo;
```

## Ví dụ 6.17 Dùng GROUP BY

---

**Table 5.17** Result table for Example 5.17.

branchNo	count	sum
B003	3	54000.00
B005	2	39000.00
B007	1	9000.00



## Ví dụ 6.18 Dùng HAVING

---

Với mỗi chi nhánh có từ 2 nhân viên trở lên, Tìm số nhân viên ở mỗi chi nhánh đó và tổng số lương của họ?

```
SELECT branchNo,  
        COUNT(staffNo) AS count,  
        SUM(salary) AS sum  
FROM Staff  
GROUP BY branchNo  
HAVING COUNT(staffNo) > 1  
ORDER BY branchNo;
```

## Ví dụ 6.18 Dùng HAVING

---

**Table 5.18** Result table for Example 5.18.

branchNo	count	sum
B003	3	54000.00
B005	2	39000.00

## **Ví dụ 6.19 Dùng truy vấn con**

---

**Lập DS nhân viên tại chi nhánh ở '163 Main St'.**

```
SELECT staffNo, fName, lName, position  
FROM Staff  
WHERE branchNo =  
    (SELECT branchNo  
        FROM Branch  
        WHERE street = '163 Main St');
```

## Ví dụ 6.19 Dừng truy vấn con

---

**Table 5.19** Result table for Example 5.19.

staffNo	fName	lName	position
SG37	Ann	Beech	Assistant
SG14	David	Ford	Supervisor
SG5	Susan	Brand	Manager

## **Ví dụ 6.20 Dùng truy vấn con và hàm thống kê**

---

**Lập DS NV có lương lớn hơn lương trung bình, và cho biết số lương vượt hơn của họ?**

```
SELECT staffNo, fName, lName, position,  
       salary – (SELECT AVG(salary) FROM Staff) As  
       SalDiff  
FROM Staff  
WHERE salary >  
       (SELECT AVG(salary)  
       FROM Staff);
```

## Ví dụ 6.20 Dùng truy vấn con và hàm thống kê

**Table 5.20** Result table for Example 5.20.

staffNo	fName	lName	position	salDiff
SL21	John	White	Manager	13000.00
SG14	David	Ford	Supervisor	1000.00
SG5	Susan	Brand	Manager	7000.00

## Ví dụ 6.21 TRUY VẤN CON LÒNG NHAU, sử dụng IN

---

Lập DS tài sản được quản lý bởi nhân viên chi nhánh tại '163 Main St'.

```
SELECT propertyNo, street, city, postcode, type, rooms, rent
FROM PropertyForRent
WHERE staffNo IN
    (SELECT staffNo
     FROM Staff
     WHERE branchNo =
         (SELECT branchNo
          FROM Branch
          WHERE street = '163 Main St'));
```

## Ví dụ 6.21 TRUY VẤN CON LÒNG NHAU, sử dụng IN

---

**Table 5.21** Result table for Example 5.21.

propertyNo	street	city	postcode	type	rooms	rent
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375
PG21	18 Dale Rd	Glasgow	G12	House	5	600



## ANY và ALL

---

- ◆ ANY và ALL có thể dùng với truy vấn con để tạo ra một cột số
- ◆ Với ALL, điều kiện chỉ đúng khi nó thỏa với tất cả các giá trị của truy vấn con.
- ◆ Với ANY, điều kiện đúng khi nó thỏa với bất kỳ giá trị nào của truy vấn con.
- ◆ Nếu truy vấn con rỗng thì ALL trả về giá trị TRUE, ANY trả về giá trị FALSE.
- ◆ SOME có thể dùng thay cho ANY.

## Ví dụ 6.22 Dùng ANY/SOME

---

Tìm các nhân viên mà lương của họ lớn hơn ít nhất lương của 1 người làm ở chi nhánh B003.

```
SELECT staffNo, fName, lName, position,  
       salary  
FROM Staff  
WHERE salary > SOME  
      (SELECT salary  
       FROM Staff  
       WHERE branchNo = 'B003');
```

## Ví dụ 6.22 Dùng ANY/SOME

- ◆ Truy vấn con bên trong có kết quả là tập hợp {12000, 18000, 24000} và truy vấn bên ngoài chọn các nhân viên có lương lớn hơn bất kỳ giá trị nào trong 3 giá trị trên của tập hợp.

**Table 5.22** Result table for Example 5.22.

staffNo	fName	lName	position	salary
SL21	John	White	Manager	30000.00
SG14	David	Ford	Supervisor	18000.00
SG5	Susan	Brand	Manager	24000.00

## Ví dụ 6.23 Dùng ALL

---

**Tìm các nhân viên mà lương của họ lớn hơn lương của tất cả mọi nhân viên làm ở chi nhánh B003.**

```
SELECT staffNo, fName, lName, position,  
       salary  
FROM Staff  
WHERE salary > ALL  
      (SELECT salary  
       FROM Staff  
       WHERE branchNo = 'B003');
```

## Ví dụ 6.23 Dùng ALL

---

**Table 5.23** Result table for Example 5.23.

staffNo	fName	lName	position	salary
SL21	John	White	Manager	30000.00

## Truy vấn trên nhiều bảng

---

- ◆ Có thể dùng truy vấn con trên cùng bảng dữ liệu với truy vấn chính.
- ◆ Nếu các cột dữ liệu của kết quả được lấy từ nhiều bảng thì phải sử dụng phép kết bảng.
- ◆ Để thực hiện phép kết, phải liệt kê các bảng tham gia phép kết vào mệnh đề **FROM**.

## Truy vấn trên nhiều bảng

---

- ◆ Có thể dùng bí danh cho các bảng trong mệnh đề FROM.
- ◆ Bí danh đứng sau tên bảng, cách tên bảng 1 khoảng trống.
- ◆ Bí danh giúp tránh sự nhầm lẫn (tốt hơn cách chỉ sử dụng tên bảng).

## Ví dụ 6.24 Phép kết đơn giản

---

Lập DS tên tất cả khách hàng có xem nhà và có đưa ra lời nhận xét.

```
SELECT c.clientNo, fName, lName,  
       propertyNo, comment  
FROM Client C, Viewing v  
WHERE c.clientNo = v.clientNo;
```



## Ví dụ 6.24 Phép kết đơn giản

- ◆ Chỉ có những hàng ở 2 bảng cùng thỏa ( $c.clientNo = v.clientNo$ ) mới có mặt trong kết quả.
- ◆ Đây là một phép kết tương đương.

**Table 5.24** Result table for Example 5.24.

clientNo	fName	lName	propertyNo	comment
CR56	Aline	Stewart	PG36	too small
CR56	Aline	Stewart	PA14	
CR56	Aline	Stewart	PG4	
CR62	Mary	Tregear	PA14	no dining room
CR76	John	Kay	PG4	too remote

## Dùng phép kết với lệnh JOIN

---

- ◆ SQL cung cấp một số cách thực hiện phép kết:

**FROM Client c JOIN Viewing v ON c.clientNo = v.clientNo**

**FROM Client JOIN Viewing USING clientNo**

**FROM Client NATURAL JOIN Viewing**

- ◆ Trong 3 cách trên, FROM đã thay gồm luôn phần WHERE của phép kết.
- ◆ Cách thứ nhất tạo ra 2 cột giống hệt nhau.

## Ví dụ 6.26 Kết 3 bảng

---

Với mỗi chi nhánh, liệt kê các nhân viên có quản lý tài sản, bao gồm thông tin thành phố của chi nhánh và mã tài sản được quản lý.

```
SELECT b.branchNo, b.city, s.staffNo,  
       fName, lName, propertyNo  
FROM branch b, staff s, property_for_rent p  
WHERE b.branchNo = s.branchNo AND  
       s.staffNo = p.staffNo  
ORDER BY b.branchNo, s.staffNo,  
         propertyNo;
```

## Ví dụ 6.26 Kết 3 bảng

**Table 5.26** Result table for Example 5.26.

branchNo	city	staffNo	fName	lName	propertyNo
B003	Glasgow	SG14	David	Ford	PG16
B003	Glasgow	SG37	Ann	Beech	PG21
B003	Glasgow	SG37	Ann	Beech	PG36
B005	London	SL41	Julie	Lee	PL94
B007	Aberdeen	SA9	Mary	Howe	PA14

### ◆ Một cách khác viết FROM :

**FROM (branch b JOIN Staff s USING branchNo) AS  
bs JOIN PropertyForRent p USING staffNo**

## **Ví dụ 6.27 Nhóm nhiều cột**

---

**Tìm số lượng tài sản đang được quản lý bởi từng nhân viên**

```
SELECT s.branchNo, s.staffNo, COUNT(*) AS count  
FROM Staff s, PropertyForRent p  
WHERE s.staffNo = p.staffNo  
GROUP BY s.branchNo, s.staffNo  
ORDER BY s.branchNo, s.staffNo;
```

## Ví dụ 6.27 Nhóm nhiều cột

---

**Table 5.27(a)** Result table for Example 5.27.

branchNo	staffNo	count
B003	SG14	1
B003	SG37	2
B005	SL41	1
B007	SA9	1

# Câu lệnh SQL thực hiện phép tích Đề-các

---

```
SELECT      [DISTINCT | ALL]
            { * | columnList }
FROM Table1 CROSS JOIN Table2
```

# Phép kết ngoài

---

- ◆ Phép kết ngoài cho phép giữ lại các hàng không thỏa điều kiện kết
- ◆ Ví dụ xét 2 bảng sau:

Branch1

branchNo	bCity
B003	Glasgow
B004	Bristol
B002	London

PropertyForRent1

propertyNo	pCity
PA14	Aberdeen
PL94	London
PG4	Glasgow



# Phép kết ngoài

---

## ◆ Phép kết (trong) của 2 bảng:

**SELECT b.\*, p.\***

**FROM Branch1 b, PropertyForRent1 p**

**WHERE b.bCity = p.pCity;**

**Table 5.27(b)** Result table for inner join of Branch1 and PropertyForRent1 tables.

branchNo	bCity	propertyNo	pCity
B003	Glasgow	PG4	Glasgow
B002	London	PL94	London

## Ví dụ 6.28 Phép kết ngoài

Liệt kê thông tin tất cả chi nhánh, bên cạnh là thông tin các tài sản cùng thành phố với các chi nhánh.

**SELECT b.\*, p.\***

**FROM Branch1 b LEFT JOIN**

**PropertyForRent1 p ON b.bCity = p.pCity;**

**Table 5.28** Result table for Example 5.28.

branchNo	bCity	propertyNo	pCity
B003	Glasgow	PG4	Glasgow
B004	Bristol	NULL	NULL
B002	London	PL94	London

## Ví dụ 6.29 Phép kết ngoài bên phải

Liệt kê thông tin chi nhánh có cùng thành phố với thông tin các tài sản (liệt kê tất cả các tài sản)

**SELECT b.\*, p.\***

**FROM Branch1 b RIGHT JOIN**

**PropertyForRent1 p ON b.bCity = p.pCity;**

**Table 5.29** Result table for Example 5.29.

branchNo	bCity	propertyNo	pCity
NULL	NULL	PA14	Aberdeen
B003	Glasgow	PG4	Glasgow
B002	London	PL94	London

## Ví dụ 6.30 Phép kết ngoài đầy đủ

Liệt kê thông tin tất cả chi nhánh, bên cạnh là thông tin tất cả các tài sản cùng thành phố với các chi nhánh.

**SELECT b.\*, p.\***

**FROM Branch1 b FULL JOIN**

**PropertyForRent1 p ON b.bCity = p.pCity;**

**Table 5.30** Result table for Example 5.30.

branchNo	bCity	propertyNo	pCity
NULL	NULL	PA14	Aberdeen
B003	Glasgow	PG4	Glasgow
B004	Bristol	NULL	NULL
B002	London	PL94	London

# EXISTS và NOT EXISTS

---

- ◆ EXISTS và NOT EXISTS chỉ dùng với các truy vấn con, có kết quả là true hoặc false
- ◆ EXISTS chỉ có giá trị True khi và chỉ khi truy vấn con có tồn tại ít nhất 1 hàng.
- ◆ EXISTS có giá trị False khi truy vấn con có kết quả rỗng.
- ◆ NOT EXISTS có giá trị ngược với EXISTS.

## Ví dụ 6.31 Truy vấn dùng EXISTS

---

**Tìm tất cả nhân viên làm việc ở chi nhánh London.**

```
SELECT staffNo, fName, lName, position  
FROM Staff s  
WHERE EXISTS  
  (SELECT *  
   FROM Branch b  
   WHERE s.branchNo = b.branchNo AND  
         city = 'London');
```

## Ví dụ 6.31 Truy vấn dùng EXISTS

---

**Table 5.31** Result table for Example 5.31.

staffNo	fName	lName	position
SL21	John	White	Manager
SL41	Julie	Lee	Assistant

## Ví dụ 6.31 Truy vấn dùng EXISTS

---

- ◆ Có thể viết lại dùng JOIN, không dùng EXISTS:

```
SELECT staffNo, fName, lName, position  
FROM Staff s, Branch b  
WHERE s.branchNo = b.branchNo AND  
       city = 'London';
```



## Ví dụ 6.32 Dùng phép hội UNION

---

Liệt kê tất cả các thành phố có chi nhánh hay có tài sản.

```
(SELECT city  
FROM Branch  
WHERE city IS NOT NULL) UNION  
(SELECT city  
FROM PropertyForRent  
WHERE city IS NOT NULL);
```

## Ví dụ 6.32 Dùng phép hội UNION

---

– Hay:

```
(SELECT *  
  FROM Branch  
  WHERE city IS NOT NULL)  
UNION CORRESPONDING BY city  
(SELECT *  
  FROM PropertyForRent  
  WHERE city IS NOT NULL);
```

**Table 5.32** Result table for Example 5.32.

city
London
Glasgow
Aberdeen
Bristol

## Ví dụ 6.33 Dùng phép giao INTERSECT

---

Liệt kê tất cả các thành phố vừa có chi nhánh vừa có tài sản.

```
(SELECT city FROM Branch)
```

```
INTERSECT
```

```
(SELECT city FROM PropertyForRent);
```

## Ví dụ 6.33 Dùng phép giao INTERSECT

---

◆ Hay:

**(SELECT \* FROM Branch)  
INTERSECT CORRESPONDING BY city  
(SELECT \* FROM PropertyForRent);**

**Table 5.33** Result table for Example 5.33.

city
Aberdeen
Glasgow
London

## Ví dụ 6.33 Dùng phép giao INTERSECT

---

- ◆ Có thể viết lại không dùng INTERSECT:

```
SELECT b.city  
FROM Branch b PropertyForRent p  
WHERE b.city = p.city;
```

- ◆ Hay:

```
SELECT DISTINCT city FROM Branch b  
WHERE EXISTS  
(SELECT * FROM PropertyForRent p  
WHERE p.city = b.city);
```

## Ví dụ 6.34 Dùng phép trừ EXCEPT

Liệt kê tất cả các thành phố có chi nhánh nhưng không có tài sản.

```
(SELECT city FROM Branch)
city FROM EXCEPT
(SELECT PropertyForRent);
```

◆ Or

```
(SELECT * FROM Branch)
EXCEPT CORRESPONDING BY city
(SELECT * FROM PropertyForRent);
```

Table 5.34 Result table for Example 5.34.

city
Bristol

## Ví dụ 6.34 Dùng phép trừ EXCEPT

---

- ◆ Có thể viết lại không dùng EXCEPT:

```
SELECT DISTINCT city FROM Branch  
WHERE city NOT IN  
  (SELECT city FROM PropertyForRent);
```

- ◆ Hay:

```
SELECT DISTINCT city FROM Branch b  
WHERE NOT EXISTS  
  (SELECT * FROM PropertyForRent p  
   WHERE p.city = b.city);
```

# Thêm dữ liệu với INSERT INTO

---

**INSERT INTO TênBảng [ (DanhSáchCột) ]  
VALUES (DanhSáchGiáTrịDữLiệu)**

- ◆ **DanhSáchCột** phần lựa chọn; Nếu bỏ qua, SQL sẽ lấy danh sách cột nguyên thủy như trong lệnh **CREATE TABLE**.
- ◆ **Danh sách dữ liệu** phải tương ứng với vị trí các cột, và tương thích với kiểu dữ liệu của từng cột



## **Ví dụ 6.35 Dùng lệnh INSERT INTO ... VALUES...**

---

**Thêm 1 hàng dữ liệu vào bảng Staff:**

**INSERT INTO Staff**

**VALUES ('SG16', 'Alan', 'Brown', 'Assistant', 'M',  
Date'1957-05-25', 8300, 'B003');**

## **Ví dụ 6.35 Dùng lệnh INSERT INTO... VALUES...**

**Khi thêm dữ liệu vào một bảng phải thêm đủ dữ liệu cho các cột bắt buộc phải nhập liệu.**

```
INSERT INTO Staff (staffNo, fName, lName,  
                    position, salary, branchNo)  
VALUES ('SG44', 'Anne', 'Jones',  
        'Assistant', 8100, 'B003');
```

◆ **Hay:**

```
INSERT INTO Staff  
VALUES ('SG44', 'Anne', 'Jones', 'Assistant', NULL,  
        NULL, 8100, 'B003');
```

## Dùng lệnh INSERT INTO... SELECT...

---

- ◆ Thêm dữ liệu là kết quả của lệnh SELECT vào một bảng:

```
INSERT INTO TênBảng [ (DanhSáchCột) ]  
SELECT ...
```

## **Ví dụ 6.37 Dùng lệnh INSERT INTO... SELECT...**

---

Giả sử có một bảng dữ liệu chứa số lượng tài sản được quản lý bởi các nhân viên:

**StaffPropCount(staffNo, fName, lName, propCnt)**

Hãy tạo dữ liệu cho bảng StaffPropCount dùng bảng Staff và bảng PropertyForRent.

## **Ví dụ 6.37 Dùng lệnh INSERT INTO... SELECT...**

---

```
INSERT INTO StaffPropCount  
(SELECT s.staffNo, fName, lName, COUNT(*))  
FROM Staff s, PropertyForRent p  
WHERE s.staffNo = p.staffNo  
GROUP BY s.staffNo, fName, lName)  
UNION  
(SELECT staffNo, fName, lName, 0  
FROM Staff  
WHERE staffNo NOT IN  
(SELECT DISTINCT staffNo  
FROM PropertyForRent));
```

## Ví dụ 6.37 Dùng lệnh INSERT INTO... SELECT...

**Table 5.35** Result table for Example 5.37.

staffNo	fName	lName	propCount
SG14	David	Ford	1
SL21	John	White	0
SG37	Ann	Beech	2
SA9	Mary	Howe	1
SG5	Susan	Brand	0
SL41	Julie	Lee	1

- ◆ Nếu phần sau UNION bị bỏ đi, danh sách sẽ thiếu các nhân viên không quản lý tài sản nào.

# Lệnh sửa dữ liệu UPDATE ...SET...

---

**UPDATE TênBảng**

**SET TênCột1 = DữLiệu1**

**[, TênCột2 = DữLiệu2...]**

**[WHERE ĐiềuKiệnLọcTìm]**

- ◆ Tên bảng có thể thế bằng tên view loại cập nhật được.
- ◆ Các dữ liệu phải tương thích kiểu với từng cột.

## **Ví dụ 6.38/39 Lệnh sửa dữ liệu UPDATE... SET...**

---

**Tăng lương tất cả nhân viên lên 3%.**

```
UPDATE Staff  
SET salary = salary*1.03;
```

**Tăng lương tất cả giám đốc lên 5%.**

```
UPDATE Staff  
SET salary = salary*1.05  
WHERE position = 'Manager';
```



## **Ví dụ 6.40 Lệnh sửa dữ liệu UPDATE... SET...**

---

**Thăng chức David Ford (staffNo = 'SG14') thành giám đốc và tăng lương lên 18,000.**

**UPDATE Staff**

**SET position = 'Manager', salary = 18000**

**WHERE staffNo = 'SG14';**

# Lệnh xóa các hàng dữ liệu DELETE

---

**DELETE FROM TênBảng  
[WHERE ĐiềuKiệnLọcTìmDL]**

- ◆ Tên bảng có thể thế bằng tên view loại cập nhật được.
- ◆ Nếu không có điều kiện lọc tìm dữ liệu thì câu lệnh sẽ xóa tất cả các hàng dữ liệu.

## **Ví dụ 6.41/42 Lệnh xóa các hàng dữ liệu DELETE**

---

**Xóa dữ liệu xem nhà PG4.**

```
DELETE FROM Viewing  
WHERE propertyNo = 'PG4';
```

**Xóa toàn bộ dữ liệu xem nhà.**

```
DELETE FROM Viewing;
```

---

## **HẾT CHƯƠNG 6**

**Mời các anh chị sinh viên tham khảo tài liệu  
các nội dung tiếp theo.**

**Chúc các anh chị học tốt.**