CTT102 – Cơ sở dữ liêu

Tháng 1/2016

Truy vấn đơn giản

Tóm tắt nội dung bài thực hành:

Sử dụng ngôn ngữ SQL trả lời các câu truy vấn đơn giản

Bộ môn **Hệ thống thông tin**

Khoa Công nghệ thông tin

ĐH Khoa học tự nhiên TP HCM

MỤC LỤC

TRUY VẤN ĐƠN GIẢN

1 Mục tiêu và tóm tắt nội dung

Sử dụng ngôn ngữ SQL để trả lời một câu hỏi về truy vấn dữ liệu đơn giản.

Sau khi hoàn thành bài tập này sinh viên có thể:

- Hiểu được cú pháp của một câu truy vấn viết bằng ngôn ngữ SQL
- Viết được các câu truy vấn đơn giản

2 Hướng dẫn chi tiết

2.1.1 Cú pháp câu truy vấn tổng quát

Một cách tổng quát, câu truy vấn 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 dòng trùng lắp 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 phẩy (,)

   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 hoặc bắt buộc trong trường hợp sử dụng bảng hơn 1 lần khi truy vấn.
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.
7. **Điều kiện_2**: điều kiện lọc lại dữ liệu sau khi đã thực hiện gom nhóm 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.

2.1.2 Câu truy vấn đơn giản

```
SELECT  <danh sách thuộc tính>
FROM tên_bảng
```

Trong mệnh đề SELECT * được dùng với ý nghĩa lấy toàn bộ các cột của bảng.

Sử dụng từ khóa **AS** để đặt lại tên cho cột.

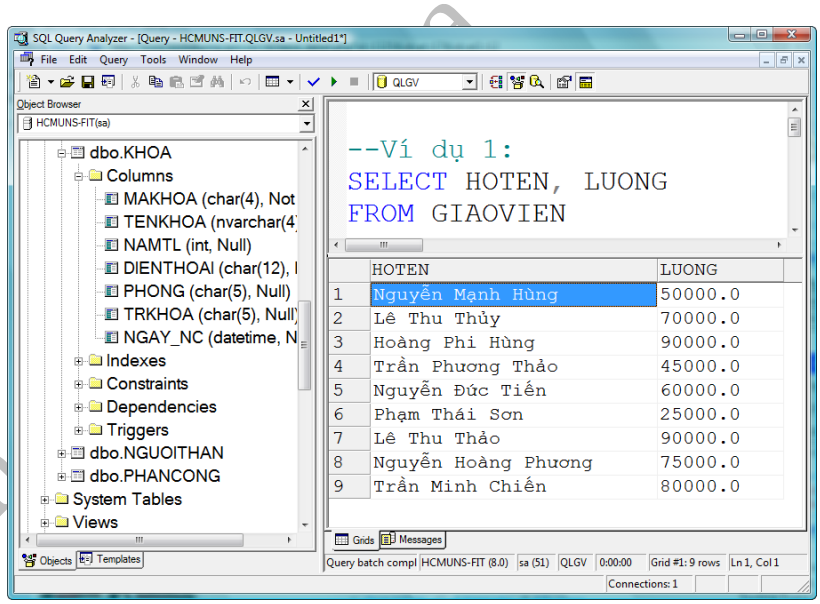
Dùng từ khoá **distinct** để loại bỏ các dòng trùng nhau và **all** để lấy tất cả các dòng dữ liệu. Mặc định không để gì cả chính là có dùng từ khóa **all**.

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

Ví dụ 1: Cho biết họ tên và lương của tất cả các giáo viên.

```
SELECT HOTEN, LUONG
FROM GIAOVIEN
```

Minh họa:



Ví dụ 2: Cho biết danh sách các giáo viên trong trường

```
SELECT *
FROM GIAOVIEN
```

Ví dụ 3: Cho biết họ tên, lương của tất cả các giáo viên.

```
--Đặt lại tên cột sử dụng từ khóa AS
SELECT HOTEN AS HOTEN_GV, LUONG AS  LUONGGV
FROM GIAOVIEN
```

2.1.3 Tìm kiếm có sắp xếp

Thuật ngữ **tìm kiếm** cũng có nghĩa tương tự với **truy vấn**

```
SELECT…
FROM…
ORDER BY thuộc_ tính_1 [ASC | DESC],  thuộc_tính_2 [ASC | DESC], ...
```

Ưu tiên sắp xếp: từ trái sang phải. Nghĩa là sắp xếp theo thuộc tính bên trái, nếu giá trị sắp xếp bằng nhau thì sẽ sắp xếp theo thuộc tính bên phải.

Ví dụ 4: Cho biết danh sách các giáo viên (họ tên, phái, lương) sắp xếp giảm dần theo lương

```
SELECT HOTEN, PHAI, LUONG
FROM GIAOVIEN
ORDER BY LUONG DESC
```

Ví dụ 5: Cho biết họ tên, mã bộ môn và lương của giáo viên. Sắp xếp tăng dần theo mã bộ môn và giảm dần theo lương.

```
SELECT HOTEN, MABM, LUONG
FROM GIAOVIEN
ORDER BY MABM, LUONG DESC
```

HOTEN	MABM	LUONG
Hoàng	HTTT	80000
Dũng	KHMT	60000
An	KHMT	70000
Thủy	HTTT	90000

⇨

HOTEN	MABM	LUONG
An	KHMT	70000
Dũng	KHMT	60000
Thủy	HTTT	90000
Hoàng	HTTT	80000

2.1.4 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 cú pháp:

```
SELECT …
FROM …
WHERE (điều kiện 1) AND/OR (điều kiện 2) .... (điều kiện n)
```

1. *Các toán tử so sánh*

SQL hỗ trợ các toán tử so sánh sau: **>** (lớn hơn), **<** (nhỏ hơn), **=** (bằng), **!=** (khác), **<>** (khác), **>=** (lớn hơn hoặc bằng), **<=** (nhỏ hơn hoặc bằng)

Ví dụ 6: Cho biết các giáo viên có lương lớn hơn 50000

```
SELECT *
FROM GIAOVIEN
WHERE luong > 50000
```

Ví dụ 7: Cho biết các giáo viên có phái là Nam

```
SELECT *
FROM GIAOVIEN
WHERE PHAI = 'Nam'
```

2. *AND, OR và NOT*

SQL hỗ trợ các toán tử logic sau: **AND** (và), **OR** (hoặc) và **NOT** (phủ định)

Ví dụ 8: Cho biết các giáo viên của bộ môn HTTT mà có lương lớn hơn 40000

```
SELECT *
FROM GIAOVIEN
WHERE MABM= 'HTTT' AND LUONG > 40000
```

Ví dụ 9: Cho biết các giáo viên nhân viên **không** thuộc bộ môn HTTT và **không** có lương lớn hơn 40000

```
SELECT *
FROM NHANVIEN
WHERE NOT (MABM = 'HTTT') AND NOT (LUONG > 40000)
```

hoặc

```
SELECT *
FROM NHANVIEN
WHERE (NOT (MABM = 'HTTT')) AND (NOT (LUONG > 40000))
```

hoặc

```
SELECT *
FROM NHANVIEN
WHERE (MABM != 'HTTT') AND (LUONG <= 40000)
```

3. *BETWEEN...AND , NOT BETWEEN ... AND*

Ví dụ 10: Cho biết các giáo viên sinh trong khoảng năm 1955 đến 1960

```
SELECT *
FROM GIAOVIEN
WHERE NGSINH BETWEEN '1/1/1955' AND '12/31/1960'
```

hoặc

```
SELECT *
FROM GIAOVIEN
WHERE year(NGSINH) BETWEEN 1955 AND 1960
```

hoặc

```
SELECT *
FROM GIAOVIEN
WHERE year(NGSINH) >= 1955 AND year(NGSINH) <=1960
```

4. 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.

Ví dụ 11: Cho biết các giáo viên không có người quản lý trực tiếp

```
SELECT *
FROM GIAOVIEN
WHERE MANQL IS NULL
```

Ví dụ 12: Cho biết các giáo viên có người quản lý trực tiếp

```
SELECT *
FROM GIAOVIEN
WHERE MANQL IS NOT NULL
```

5. IN và NOT IN

IN và **NOT IN**: Để kiểm tra một giá trị có (không) nằm trong một tập hợp các giá trị hay không.

Ví dụ 13: Cho biết các giáo viên **có lương** là 20000, 30000 hoặc 40000.

```
SELECT *
FROM GIAOVIEN
WHERE LUONG IN (20000, 30000, 40000)
hoặc
SELECT *
FROM GIAOVIEN
WHERE LUONG = 20000 OR LUONG = 30000 OR LUONG = 40000
```

2.1.5 Tìm kiếm với điều kiện liên quan đến chuỗi ký tự

Để xử lý với các dữ liệu thuộc dạng xâu ký tự, ngoài toán tử so sánh bằng = để sử dụng so sánh chuỗi tuyệt đối thì ngôn ngữ SQL có hỗ trợ toán tử so sánh chuỗi LIKE để so sánh chuỗi tương đối.

Ví dụ 14: Cho biết các giáo viên có địa chỉ ở TP HCM

```
SELECT *
FROM GIAOVIEN
WHERE DIACHI LIKE '%TP HCM'
```

Lưu ý về các **ký tự đại diện** khi sử dụng LIKE[1]:

%	: đại diện cho một **chuỗi ký tự** bất kỳ
_	: đại diện cho một **ký tự** bất kỳ
[ ]	: đại diện cho các ký tự nằm trong một khoảng nào đó.

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'**

[1] Sinh viên tham khảo các ký tự đại diện khác trong Book Onlines

2.1.6 Tìm kiếm với điều kiện liên quan đến ngày giờ

Một số hàm liên quan đến ngày giờ mà SQL hỗ trợ:

- **datediff**: Hàm tính khoảng cách (hiệu) 2 thời gian theo một đơn vị nào đó (đơn vị ngày, tháng, năm, giờ, phút, giây)
- **datepart**: Hàm lấy một phần trong giá trị thời gian (ngày, tháng, năm, giờ, phút, giây)
- **year**, **month**, **day**: Hàm lấy năm, tháng, ngày của một giá trị thời gian truyền vào.
- **getdate**: Hàm lấy ngày hiện hành của hệ thống

Ví dụ 15: Cho biết những đề tài bắt đầu sau ngày 30/4/2005

```
SELECT TENDT, CAPQL
FROM DETAI
WHERE datediff(d, TGBD, '4/30/2005') < 0
```

Ví dụ 16: Cho biết những đề tài kết thúc trước 1 tuần so với ngày 31/12/2007

```
SELECT *
FROM DETAI
WHERE datediff(d, TGKT, '12/31/2007') > 7
```

Ví dụ 17: Cho biết các đề tài có ngày bắt đầu là 30/4/2005

Cách 1:

```
SELECT *
FROM DETAI
WHERE TGBD = '4/30/2005'
```

Cách 2:

```
SELECT *
FROM DETAI
WHERE datediff(d, TGBD, '4/30/2005') = 0
```

Lưu ý: Khi so sánh với kiểu dữ liệu datetime, sử dụng các hàm sẽ chính xác hơn. Ví dụ: Trong **cách 1**, nếu một đề tài mà có TGBD = '4/30/2005 17:00:00' thì sẽ không xuất ra trong kết quả.

2.1.7 Sử dụng các hàm khi tìm kiếm

SQL cho phép sử dụng hàm trong câu truy vấn ở nhiều hình thức:

- Sử dụng hàm trong mệnh đề WHERE: biểu thức điều kiện
- 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.
- Sử dụng hàm trong mệnh đề ORDER BY
- Một số hàm mà SQL có hỗ trợ[2]:
    - Các hàm về ngày tháng :
        - datediff, datepart
        - year, month, day
        - getdate
        - dateadd
    - Các hàm về chuỗi
        - len
        - replace
        - charindex
        - reverse
    - Các hàm chuyển đổi kiểu dữ liệu
        - convert
        - cast
    - Các hàm toán học
        - floor, ceil
    - …

Ví dụ 18: Cho biết họ tên và tuổi của các giáo viên

```
SELECT HOTEN, datediff(yyyy, NGSINH, getdate()) as TUOI
FROM GIAOVIEN
hoặc
SELECT HOTEN, year(getdate()) - year(NGSINH) as TUOI
FROM GIAOVIEN
```

[2] Sinh viên tự tìm hiểu cách sử dụng các hàm này trong Book Onlines

Ví dụ 19: Cho biết các giáo viên sinh năm 1975

```
SELECT *
FROM GIAOVIEN
WHERE year(NGSINH) = 1975
```

Ví dụ 20: Cho biết lương và lương của giáo viên sau khi đã tăng 10%

```
SELECT LUONG AS LUONG_TRUOC, LUONG * 1.1 AS LUONG_SAU
FROM GIAOVIEN
```

Ví dụ 21: Cho biết danh sách tên đề tài và năm bắt đầu thực hiện, sắp xếp giảm dần theo năm bắt đầu.

```
SELECT TENDT, year(TGBD) AS NAMBD
FROM DETAI
ORDER BY year(TGBD) DESC
```

2.1.8 Các phép toán tập hợp

Cú pháp chung:

```
SELECT … FROM … WHERE …
UNION | INTERSECT | EXCEPT
SELECT … FROM … WHERE …
```

Điều kiện để thực hiện được phép toán tập hợp:

- Cùng số lượng thuộc tính
- Cùng kiểu dữ liệu của các thuộc tính

Ví dụ 22: Cho biết thông tin các trưởng bộ môn có tham gia đề tài

```
SELECT TRBOMON
FROM BOMON
INTERSECT
SELECT MAGV
FROM GIAOVIEN
```

2.1.9 Phép kết

Truy vấn dữ liệu trên nhiều bảng và sử dụng quan hệ giữa các bảng một cách phù hợp để có được điều kiện kết đúng.

Ví dụ giữa GIAOVIEN và BOMON có 2 mối quan hệ:

- Một giáo viên làm việc cho một bộ môn. Để biết giáo viên làm việc cho bộ môn nào ta dựa vào mối quan hệ giữa 2 thuộc tính MABM của GIAOVIEN và MABM môncủa BOMON.
- Một bộ môn có một giáo viên làm trưởng bộ môn. Để biết giáo viên nào làm trưởng của bọ môn nào ta dựa vào mối quan hệ giữa 2 thuộc tính TRBOMON của BOMON và MAGV của GIAOVIEN.

Hai ví dụ sau thể hiện 2 mối quan hệ của GIAOVIEN và BOMON trong 2 nhu cầu truy vấn khác nhau:

Ví dụ 23: Cho biết tên giáo viên và tên bộ môn mà giáo viên đó làm việc.

```
SELECT HOTEN, TENBM
FROM GIAOVIEN, BOMON
WHERE GIAOVIEN.MABM = BOMON.MABM
```

Ví dụ 24: Cho biết tên giáo viên và tên bộ môn mà giáo viên làm trưởng bộ môn của bộ môn đó.

```
SELECT HOTEN, TENBM
FROM BOMON, GIAOVIEN
WHERE BOMON.TRUONGBM = GIAOVIEN.MAGV
```

2.1.10 Sử dụng ALIAS[3]

Khi tìm kiếm trên nhiều bảng **để** làm câu truy vấn dễ hiểu và ngắn gọn hoặc giữa các bảng có thuộc tính trùng tên, hoặc sử dụng một bảng nhiều hơn 1 lần trong câu truy vấn đặt tên lại các bảng để có thể phân biệt các bảng, các thuộc tính.

Ví dụ 25: Cho biết tên khoa và tên trưởng khoa của khoa đó (Sử dụng **ALIAS**):

```
SELECT K.TENKHOA, GV.HOTEN
FROM KHOA AS K, GIAOVIEN AS GV
WHERE K.TRKHOA = GV.MAGV
hoặc
SELECT K.TENKHOA, GV.HOTEN
FROM KHOA K, GIAOVIEN GV
WHERE K.TRKHOA = GV.MAGV
```

Ví dụ 26: Cho biết tên giáo viên và tên những người thân của giáo viên:

```
SELECT GV.HOTEN AS TENGV, NT.TEN AS TENNT
FROM GIAOVIEN AS GV, NGUOITHAN AS NT
WHERE NT.MAGV = GV.MAGV
```

Ví dụ 27: Cho biết tên giáo viên và tên người quản lý của giáo viên đó

```
SELECT GV.HOTEN AS TENGV, NQL.HOTEN AS TENNQL
FROM GIAOVIEN AS GV, GIAOVIEN AS NQL
WHERE GV.MANQL = NQL.MAGV
```

Ví dụ 28: Cho biết tên giáo viên và tên khoa mà giáo viên đó trực thuộc.

```
SELECT GV.HOTEN, K.TENKHOA
FROM GIAOVIEN GV, BOMON BM, KHOA K
WHERE GV.MABM = BM.MABM AND BM.MAKHOA = K.MAKHOA
```

[3] Khi truy vấn trên các bảng có các thuộc tính cùng tên thì việc sử dụng ALIAS sẽ giúp tránh được việc nhập nhằng khi sử dụng các thuộc tính. Ngoài ra, việc sử dụng ALIAS trong những câu truy vấn trên nhiều bảng sẽ làm cho câu truy vấn dễ đọc hơn.

2.1.11 Sử dụng JOIN

Cú pháp:

```
SELECT …
FROM (TABLE1 JOIN TABLE 2 ON [Điều kiện kết ] ) JOIN TABLE3 ON [Điều kiện kết]
WHERE …
```

Ví dụ 29: Cho biết tên giáo viên và tên bộ môn mà giáo viên đó làm việc (Viết lại ví dụ 23 sử dụng JOIN).

Sử dụng điều kiện kết ở WHERE

```
SELECT G.HOTEN, B.TENBM
FROM GIAOVIEN G JOIN BOMON B
WHERE G.MABM = B.MABM
```

Sử dụng **JOI**N

```
SELECT G.HOTEN, B.TENBM
FROM GIAOVIEN G JOIN BOMON B ON G.MABM = B.MABM
```

HẾT