INT1313 – Cơ Sở Dữ Liệu

Mô hình Dữ Liệu Quan hệ và SQL

SQL

Giảng viên: Lê Hà Thanh

lehathanh@ptithcm.edu.vn

Hành trình

- Mô hình Dữ liệu Quan hệ
- Các Ràng buộc CSDL Quan hệ
- SQL
- Đại số quan hệ và Phép tính quan hệ
- Thiết kế Lược đồ CSDL quan hệ từ lược đồ khái niệm

Nội dung

- Giới thiệu SQL
- Định nghĩa dữ liệu SQL và các kiểu dữ liệu
- Dùng SQL để mô tả các ràng buộc
- Truy vấn dữ liệu
- Chèn, xóa và cập nhật dữ liệu

Tổng quan về SQL

- SQL = Structured Query Language: Ngôn ngữ truy vấn có cấu trúc.
  - Tiền thân là SEQUEL (Structured English QUEry Language) được phát triển tại phòng nghiên cứu của IBM cho hệ CSDL SYSTEM R.

- SQL là ngôn ngữ tiêu chuẩn cho các hệ quản trị CSDL quan hệ.

- Là ngôn ngữ CSDL toàn diện, gồm :
  - Các lệnh định nghĩa dữ liệu
  - Các lệnh truy vấn và cập nhật dữ liệu
  - Cho phép thiết lập các chế độ xem (view) dữ liệu
  - Xác lập các chỉ định bảo mật và ủy quyền
  - Xác định ràng buộc toàn vẹn
  - Chỉ định kiểm soát giao dịch
  - Các quy tắc để nhúng câu lệnh SQL vào ngôn ngữ lập trình

Định nghĩa dữ liệu SQL, Các kiểu dữ liệu

SQL và mô hình quan hệ

Dùng khái niệm riêng để biểu diễn các khái niệm của mô hình quan hệ hình thức:

Bảng (**table**) : quan hệ (*relation*)

Dòng (**row**) : bộ dữ liệu (*tuple*)

Cột (**column**) : thuộc tích (*attribute*)

- Lệnh CREATE: tạo lược đồ, bảng, kiểu, miền; các cấu trúc định nghĩa quan sát (view) dữ liệu, đánh giá (assertion) và cơ chế kích hoạt (trigger)

Khái niệm lược đồ (schema) và danh mục (catalog)

Lược đồ SQL (CSDL) được xác định bằng tên lược đồ, và gồm:

- Một mã định danh xác định người dùng hoặc tài khoản sở hữu lược đồ
- Các mô tả từng thành phần trong lược đồ:
  - Bảng (*tables*)
  - Kiểu dữ liệu (*types*)
  - Ràng buộc (*constraints*)
  - Miền (*domains*)
  - Các cấu trúc khác như quy định cấp quyền (*authorization grants*) truy cập và xử lý dữ liệu

- Catalogs: tập hợp các lược đồ CSDL

CREATE SCHEMA : tạo lược đồ CSDL

CREATE SCHEMA COMPANY **AUTHORIZATION** “Jsmith”;

Chú ý:

- Đặc quyền tạo lược đồ CSDL và các thành phần của CSDL được cấp cho một số người dùng thích hợp
- Việc cấp quyền được thực hiện bởi quản trị hệ thống hoặc quản trị hệ CSDL (DBA – DataBase Administrator)

CREATE TABLE : tạo bảng

- Tạo mới và đặt tên cho một quan hệ
- Xác lập các thuộc tính: tên thuộc tính, kiểu dữ liệu, ràng buộc NOT NULL
- Mô tả các ràng buộc ban đầu trên các thuộc tính: ràng buộc khóa, toàn vẹn thực thể, toàn vẹn tham chiếu.

Chú ý: Dùng **ALTER TABLE** để bổ sung các ràng buộc sau khi tạo bảng.

CREATE TABLE : tạo lược đồ CSDL COMPANY

```
CREATE TABLE EMPLOYEE
        ( Fname                 VARCHAR(15)             NOT NULL,
          Minit                 CHAR,
          Lname                 VARCHAR(15)             NOT NULL,
          Ssn                   CHAR(9)                 NOT NULL,
          Bdate                 DATE,
          Address               VARCHAR(30),
          Sex                   CHAR,
          Salary                DECIMAL(10,2),
          Super_ssn             CHAR(9),
          Dno                   INT                     NOT NULL,
        PRIMARY KEY (Ssn),
CREATE TABLE DEPARTMENT
        ( Dname                 VARCHAR(15)             NOT NULL,
          Dnumber               INT                     NOT NULL,
          Mgr_ssn               CHAR(9)                 NOT NULL,
          Mgr_start_date        DATE,
        PRIMARY KEY (Dnumber),
        UNIQUE (Dname),
        FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn) );
CREATE TABLE DEPT_LOCATIONS
        ( Dnumber               INT                     NOT NULL,
          Dlocation             VARCHAR(15)             NOT NULL,
        PRIMARY KEY (Dnumber, Dlocation),
        FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber) );
```

CREATE TABLE : tạo lược đồ CSDL COMPANY

```
CREATE TABLE PROJECT
        ( Pname                 VARCHAR(15)         NOT NULL,
          Pnumber               INT                 NOT NULL,
          Plocation             VARCHAR(15),
          Dnum                  INT                 NOT NULL,
        PRIMARY KEY (Pnumber),
        UNIQUE (Pname),
        FOREIGN KEY (Dnum) REFERENCES DEPARTMENT(Dnumber) );
CREATE TABLE WORKS_ON
        ( Essn                  CHAR(9)             NOT NULL,
          Pno                   INT                 NOT NULL,
          Hours                 DECIMAL(3,1)        NOT NULL,
        PRIMARY KEY (Essn, Pno),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn),
        FOREIGN KEY (Pno) REFERENCES PROJECT(Pnumber) );
CREATE TABLE DEPENDENT
        ( Essn                  CHAR(9)             NOT NULL,
          Dependent_name        VARCHAR(15)         NOT NULL,
          Sex                   CHAR,
          Bdate                 DATE,
          Relationship          VARCHAR(8),
        PRIMARY KEY (Essn, Dependent_name),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn) );
```

Các kiểu dữ liệu và miền giá trị trong SQL

- Numeric: (INTEGER/INT/SMALLINT), (FLOAT/REAL/DOUBLE PRECISION); DECIMAL(i,j)
- Character-string: CHAR(n), VARCHAR(n); CHARACTER LARGE OBJECT (CLOB)
- Bit-string: BIT(n), BINARY LARGE OBJECT (BLOB)
- Boolean: {TRUE, FALSE, UNKNOWN}
- DATE/TIME/TIME WITH TIME ZONE
- TIMESTAMP/INTERVAL

CREATE DOMAIN : Định nghĩa miền dữ liệu mới, ví dụ:

CREATE DOMAIN SSN_TYPE **AS** CHAR(9);

CREATE TYPE : Định nghĩa kiểu dữ liệu mới

Mô tả ràng buộc trong SQL

Ràng buộc thuộc tính và thiết lập mặc định thuộc tính

- **NOT NULL**: thuộc tính không được phép nhận giá trị NULL.

- **DEFAULT** <value>: nếu không cung cấp giá trị cụ thể cho thuộc tính, thuộc tính được mặc định nhận giá trị <value> cho trước.
  - Nếu không xác định giá trị mặc định, NULL sẽ được gán mặc định cho các thuộc tính không bị quy định NOT NULL.

- **CHECK** <attribute/domain definition>: kiểm tra thuộc tính và miền giá trị.

```
Dnumber INT NOT NULL CHECK (Dnumber > 0 AND Dnumber < 21);
```

```
CREATE DOMAIN D_NUM AS INTEGER
CHECK (D_NUM > 0 AND D_NUM < 21);
```

Ví dụ:

```
CREATE TABLE EMPLOYEE
    ( … ,
      Dno          INT          NOT NULL        DEFAULT 1,
    CONSTRAINT EMPPK
      PRIMARY KEY (Ssn),
    CONSTRAINT EMPSUPERFK
      FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn)
                        ON DELETE SET NULL          ON UPDATE CASCADE,
    CONSTRAINT EMPDEPTFK
      FOREIGN KEY(Dno) REFERENCES DEPARTMENT(Dnumber)
                        ON DELETE SET DEFAULT      ON UPDATE CASCADE);
CREATE TABLE DEPARTMENT
    ( … ,
      Mgr_ssn CHAR(9)          NOT NULL        DEFAULT '888665555',
      … ,
    CONSTRAINT DEPTPK
      PRIMARY KEY(Dnumber),
    CONSTRAINT DEPTSK
      UNIQUE (Dname),
    CONSTRAINT DEPTMGRFK
      FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn)
                        ON DELETE SET DEFAULT      ON UPDATE CASCADE);
CREATE TABLE DEPT_LOCATIONS
    ( … ,
    PRIMARY KEY (Dnumber, Dlocation),
    FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber)
                        ON DELETE CASCADE          ON UPDATE CASCADE);
```

Ràng buộc Khóa

- **PRIMARY KEY** : xác định một hoặc nhiều thuộc tính tạo thành khóa chính của quan hệ

```
Dnumber INT PRIMARY KEY,
```

- **UNIQUE** : xác định các khóa thay thế, nhận các giá trị duy nhất

```
Dname VARCHAR(15) UNIQUE,
```

Ràng buộc Toàn vẹn tham chiếu

FOREIGN KEY : khai báo ràng buộc khóa ngoài để quy định tính toàn vẹn tham chiếu

```
FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn) );
```

Xử lý vi phạm Ràng buộc Toàn vẹn tham chiếu

- Từ chối thực hiện lệnh vi phạm bằng tùy chọn **RESTRICT**, hoặc
- Định nghĩa hành động thay thế:
  - **SET NULL**:
  - **SET DEFAULT**:
  - **CASCADE**: thực hiện "lan truyền" hành động tương ứng trên các đối tượng tham chiếu.
    Áp dụng:
    - Các quan hệ thể hiện mối liên hệ. Ví dụ: WORKS_ON
    - Các quan hệ biểu diễn các thuộc tính đa trị. Ví dụ: DEPT_LOCATIONS
    - Các quan hệ biểu diễn các kiểu thực thể yếu. Ví dụ: DEPENDENT
  - Dùng kết hợp với khai báo xác định sự kiện ON DELETE / ON UPDATE

Ví dụ:

```
CREATE TABLE EMPLOYEE
    ( … ,
      Dno          INT          NOT NULL        DEFAULT 1,
    CONSTRAINT EMPPK
      PRIMARY KEY (Ssn),
    CONSTRAINT EMPSUPERFK
      FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn)
                        ON DELETE SET NULL          ON UPDATE CASCADE,
    CONSTRAINT EMPDEPTFK
      FOREIGN KEY(Dno) REFERENCES DEPARTMENT(Dnumber)
                        ON DELETE SET DEFAULT      ON UPDATE CASCADE);
CREATE TABLE DEPARTMENT
    ( … ,
      Mgr_ssn CHAR(9)          NOT NULL        DEFAULT '888665555',
      … ,
    CONSTRAINT DEPTPK
      PRIMARY KEY(Dnumber),
    CONSTRAINT DEPTSK
      UNIQUE (Dname),
    CONSTRAINT DEPTMGRFK
      FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn)
                        ON DELETE SET DEFAULT      ON UPDATE CASCADE);
CREATE TABLE DEPT_LOCATIONS
    ( … ,
    PRIMARY KEY (Dnumber, Dlocation),
    FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber)
                        ON DELETE CASCADE          ON UPDATE CASCADE);
```

Đặt tên cho các ràng buộc

CONSTRAINT <constraint name>

- Tên của ràng buộc là duy nhất
- Dễ dàng theo dõi một ràng buộc nếu muốn bỏ và thay thế bởi ràng buộc khác
- Đặt tên cho ràng buộc là tùy chọn
- Có thể tạm thời hoãn kiểm tra một ràng buộc cho đến khi hoàn tất giao dịch (vấn đề này sẽ được đề cập lại sau)

Xác định ràng buộc trên bộ dữ liệu bằng CHECK

Còn gọi là ràng buộc theo dòng (**row-based** constraints):

- Áp dụng cho từng dòng
- Được kiểm tra bất cứ khi nào một dòng được chèn vào bảng hoặc được cập nhật.

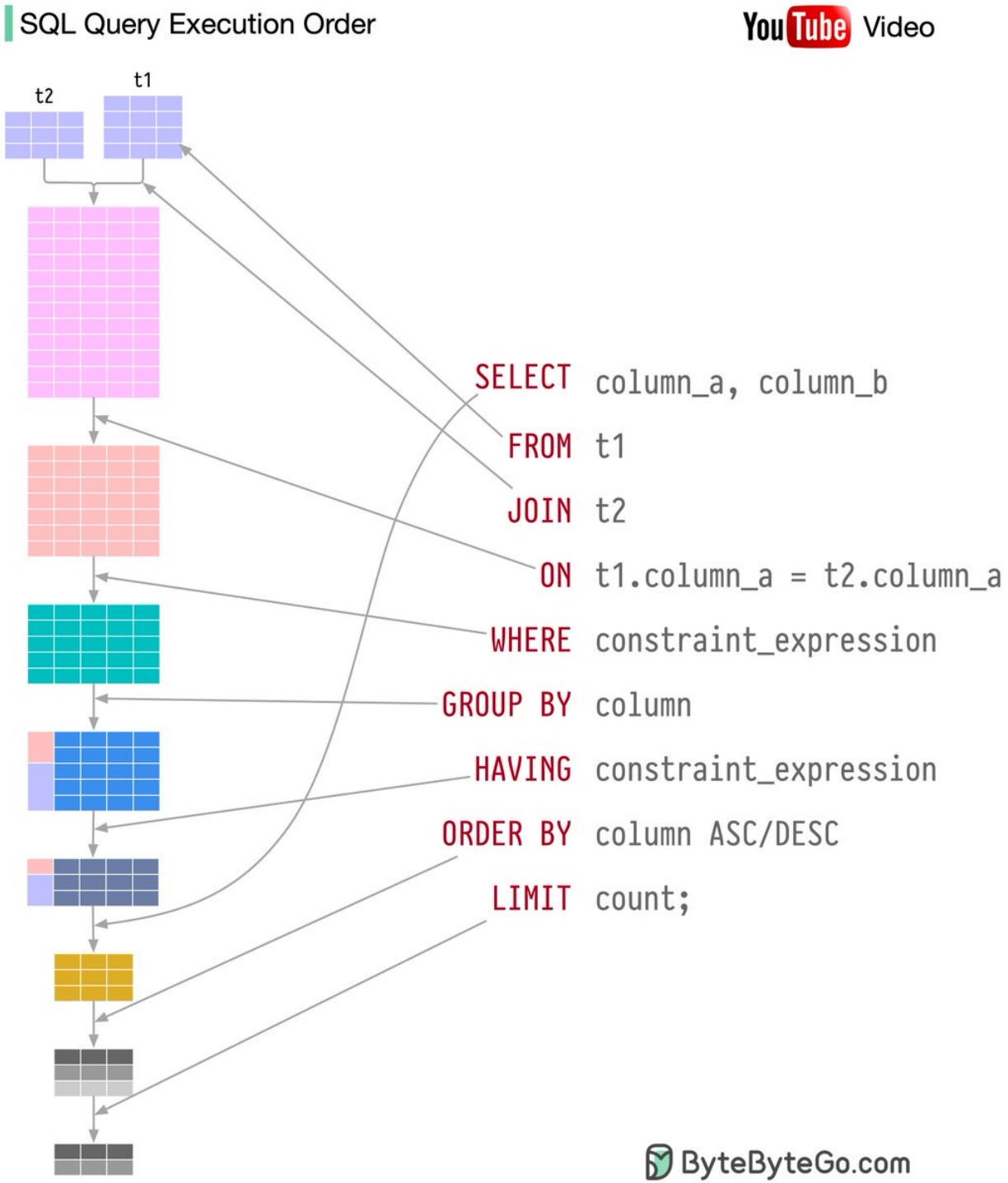
Ví dụ:

```
CREATE TABLE
    ...
CHECK (Dept_create_date <= Mgr_start_date);
```

- Có thể được dùng để quy định các ràng buộc tổng quát trong câu lệnh **CREATE ASSERTION**

Các truy vấn truy xuất dữ liệu cơ bản trong SQL

Lệnh SELECT


SQL Query Execution Order
YouTube Video
t2
t1
SELECT column_a, column_b
FROM t1
JOIN t2
ON t1.column_a = t2.column_a
WHERE constraint_expression
GROUP BY column
HAVING constraint_expression
ORDER BY column ASC/DESC
LIMIT count;
ByteByteGo.com

SELECT … FROM … WHERE …

Dạng căn bản của lệnh SELECT, còn gọi là một ánh xạ (mapping), là khối select-from-where, có dạng:

SELECT <attribute list>
FROM <table list>
WHERE <condition>;

Trong đó:

- <attribute list>: danh sách tên các thuộc tính để trích xuất các giá trị dữ liệu, còn gọi là các thuộc tính chiếu (*projection attributes*)
- <table list>: danh sách tên các quan hệ trong truy vấn
- <condition>: biểu thức điều kiện xác định các bộ sẽ được truy xuất bởi lệnh truy vấn, gồm các điều kiện chọn (*selection conditions*), điều kiện kết (*join conditions*)

SELECT: Ví dụ 0

Query 0: Lấy ngày sinh và địa chỉ của nhân viên có tên 'John B. Smith'.

Q0:
```
SELECT   Bdate, Address
FROM     EMPLOYEE
WHERE    Fname = 'John' AND Minit = 'B' AND Lname = 'Smith';
```

Bdate	Address
1965-01-09	731Fondren, Houston, TX

SELECT: Ví dụ 1

Query 1: Lấy tên và địa chỉ của tất cả nhân viên làm trong phòng nghiên cứu 'Research'

Q1: **SELECT** Fname, Lname, Address
FROM EMPLOYEE, DEPARTMENT
WHERE Dname = 'Research' **AND** Dnumber = Dno**;**

Điều kiện chọn: Dname = 'Research',

Điều kiện kết: Dnumber = Dno, để lấy ra bộ từ EMPLOYEE và DEPARTMENT có cùng mã số phòng ban

Fname	Lname	Address
John	Smith	731 Fondren, Houston, TX
Franklin	Wong	638 Voss, Houston, TX
Ramesh	Narayan	975 Fire Oak, Humble, TX
Joyce	English	5631 Rice, Houston, TX

SELECT: Ví dụ 2, truy vấn select-project-join

Query 2: "Với mọi dự án triển khai tại 'Stafford', hãy liệt kê mã số dự án, mã số phòng ban quản lý, và họ, địa chỉ, ngày sinh của người quản lý phòng ban này."

Q2: **SELECT** Pnumber, Dnum, Lname, Address, Bdate
FROM PROJECT, DEPARTMENT, EMPLOYEE
WHERE Dnum = Dnumber **AND** Mgr_ssn = Ssn **AND** Plocation = 'Stafford';

Pnumber	Dnum	Lname	Address	Bdate
10	4	Wallace	291Berry, Bellaire, TX	1941-06-20
30	4	Wallace	291Berry, Bellaire, TX	1941-06-20

Quy cách dùng tên thuộc tính và biến: .

Dùng tên quan hệ với toán tử tham chiếu (toán tử 'chấm' - .) để xác định thuộc tính.

Ví dụ: viết lại Q1

```
Q1A:  SELECT  Fname, EMPLOYEE.Name, Address
      FROM    EMPLOYEE, DEPARTMENT
      WHERE   DEPARTMENT.Name = 'Research' AND
              DEPARTMENT.Dnumber = EMPLOYEE.Dnumber;
```

Hoặc,

```
Q1′:  SELECT  EMPLOYEE.Fname, EMPLOYEE.LName,
              EMPLOYEE.Address
      FROM    EMPLOYEE, DEPARTMENT
      WHERE   DEPARTMENT.DName = 'Research' AND
              DEPARTMENT.Dnumber = EMPLOYEE.Dno;
```

Quy cách dùng tên thuộc tính và biến: *alias*

Có thể đặt biệt danh (*alias*) cho các bảng, thường để làm gọn các tên dài.

```
Q1B:    SELECT    E.Fname, E.LName, E.Address
        FROM      EMPLOYEE AS E, DEPARTMENT AS D
        WHERE     D.DName = 'Research' AND D.Dnumber = E.Dno;
```

Quy cách dùng tên thuộc tính và biến: *alias*

Đặt biệt danh (*alias*) cho các bảng để dùng khi tham chiếu đến cùng quan hệ có thể gây nhầm lẫn.

Query 8: Với mỗi nhân viên, lấy ra họ và tên của người này, và họ và tên của người quản lý trực tiếp của họ.

```
Q8:   SELECT   E.Fname, E.Lname, S.Fname, S.Lname
      FROM     EMPLOYEE AS E, EMPLOYEE AS S
      WHERE    E.Super_ssn = S.Ssn;
```

E.Fname	E.Lname	S.Fname	S.Lname
John	Smith	Franklin	Wong
Franklin	Wong	James	Borg
Alicia	Zelaya	Jennifer	Wallace
Jennifer	Wallace	James	Borg
Ramesh	Narayan	Franklin	Wong
Joyce	English	Franklin	Wong
Ahmad	Jabbar	Jennifer	Wallace

Chọn chiếu trên toàn bộ thuộc tính: *

Q1C: **SELECT** *
FROM EMPLOYEE
WHERE DNO = 5;

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-09-01	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5

Q1D: **SELECT** *
FROM EMPLOYEE, DEPARTMENT
WHERE Dname = 'Research' **AND** Dno = Dnumber;

Q10A: **SELECT** *
FROM EMPLOYEE, DEPARTMENT;

Chọn tất cả các bộ dữ liệu

Query 9: Chon tất cả các mã quản lý trong EMPLOYEE.

Q9: **SELECT** Ssn
FROM EMPLOYEE**;**

E.Fname
123456789
333445555
999887777
987654321
666884444
453453453
987987987
888665555

Chọn tất cả tổ hợp dữ liệu

Query 10: Lấy tất cả các tổ hợp của
EMPLOYEE Ssn và
DEPARTMENT Dname.

Q10: **SELECT** Ssn, Dname
FROM EMPLOYEE, DEPARTMENT**;**

→ SAI THÔNG TIN!

Ssn	Dname
123456789	Research
333445555	Research
999887777	Research
987654321	Research
666884444	Research
453453453	Research
987987987	Research
888665555	Research
123456789	Administration
333445555	Administration
999887777	Administration
987654321	Administration
666884444	Administration
453453453	Administration
987987987	Administration
888665555	Administration
123456789	Headquarters
333445555	Headquarters
999887777	Headquarters
987654321	Headquarters
666884444	Headquarters
453453453	Headquarters
987987987	Headquarters
888665555	Headquarters

Truy vấn để lấy tập hợp các bộ dữ liệu

- Truy vấn SQL thường cho ra kết quả gồm nhiều bộ dữ liệu có thể trùng lặp.

  SELECT …

tương đương với

  SELECT ALL …

- Khai báo **DISTINCT** để loại bỏ các kết quả trùng lặp, nghĩa là lấy tập hợp các bộ dữ liệu riêng biệt.

  SELECT DISTINCT …

Truy vấn để lấy tập hợp các bộ dữ liệu – Ví dụ

Query 11: Lấy bảng lương của tất cả các nhân viên.

Q11: **SELECT** **ALL** Salary
FROM EMPLOYEE**;**

Salary
30000
40000
25000
43000
38000
25000
25000
55000

Q11A: Lấy ra tất cả các giá trị lương riêng biệt.

Q11A: **SELECT** **DISTINCT** Salary
FROM EMPLOYEE**;**

Salary
30000
40000
25000
43000
38000
55000

Truy vấn với các toán tử tập hợp

- **UNION**: phép hợp tập hợp
- **EXCEPT**: phép trừ tập hợp
- **INTERSECT**: phép giao tập hợp

→ Kết quả cũng là tập các bộ dữ liệu: đã loại bỏ các bộ trùng lặp.

- *Chú ý:* hai quan hệ thành phần của truy vấn phải đảm bảo:
  - Có cùng thuộc tính và
  - Thứ tự xuất hiện của các thuộc tính phải giống nhau.

Truy vấn với các toán tử tập hợp – Ví dụ với UNION

Query 4: "Tạo một danh sách gồm tất cả các mã số dự án có sự tham gia của nhân viên có họ là 'Smith', dưới vai trò là công nhân (worker) hoặc là người quản lý của phòng ban (manager) đang kiểm soát dự án."

Q4A:

```
( SELECT   DISTINCT Pnumber
  FROM     PROJECT, DEPARTMENT, EMPLOYEE
  WHERE    Dnum = Dnumber AND Mgr_ssn = Ssn
           AND Lname = 'Smith' )

UNION

( SELECT   DISTINCT Pnumber
  FROM     PROJECT, WORKS_ON, EMPLOYEE
  WHERE    Pnumber = Pno AND Essn = Ssn AND Lname = 'Smith' );
```

Truy vấn với các phép toán đa tập hợp (multiset)

- UNION **ALL**: phép hợp tập hợp
- EXCEPT **ALL**: phép trừ tập hợp
- INTERSECT **ALL**: phép giao tập hợp

→ Kết quả gồm đầy đủ các bộ dữ liệu, kể cả các bộ trùng lặp khi thực hiện phép toán tập hợp

Truy vấn với các phép toán đa tập hợp (multiset) – Ví dụ

a) Hai bảng R(A) và S(A)

b) R(A) UNION **ALL** S(A)

c) R(A) EXCEPT **ALL** S(A)

d) R(A) INTERSECT **ALL** S(A)

(a)

R

A
a1
a2
a2
a3

S

A
a1
a2
a4
a5

(b)

T

A
a1
a1
a2
a2
a2
a3
a4
a5

(c)

T

A
a2
a3

(d)

T

A
a1
a2

Kiểm tra so mẫu chuỗi bằng LIKE

- % : đại diện thay thế cho một chuỗi con có từ 0 trở lên các ký tự
- _ : thay thế cho một ký tự

Query 12: Lấy tất cả các nhân viên có địa chỉ tại vùng Houston, Texas.

```
Q12:   SELECT   Fname, Lname
       FROM     EMPLOYEE
       WHERE    Address LIKE '%Houston, TX%';
```

Query 12A: Tìm tất cả nhân viên được sinh ra trong thập niên 1950;

```
Q12A:  SELECT   Fname, Lname
       FROM     EMPLOYEE
       WHERE    Address LIKE '_ _ 5 _ _ _ _ _ _ _';
```

Bdate
1965-09-01
1955-12-08
1962-09-15
1972-07-31

Sử dụng các tính toán số học trong câu truy vấn

- Kiểu số học với các phép toán: +, -, *, /
- Ghép chuỗi: ||
- Kiểu date, timestamp, khoảng thời gian: +, -
- So sánh: <, <=, >, >=, **BETWEEN**

Sử dụng các tính toán số học trong câu truy vấn

Query 13: Hiển thị mức lương kết quả nếu mọi nhân viên làm việc trong dự án 'ProductX' được tăng thêm 10% lương.

```
Q13:  SELECT   E.Fname, E.Lname, 1.1 * E.Salary AS Increased_sal
      FROM     EMPLOYEE AS E, WORKS_ON AS W, PROJECT AS P
      WHERE    E.Ssn = W.Essn AND W.Pno = P.Pnumber AND
               P.Pname = 'ProductX' ;
```

Query 14: Lấy tất cả nhân viên trong phòng ban số 5 có mức lương trong khoảng từ $30,000 đến $40,000.

```
Q14:  SELECT   *
      FROM     EMPLOYEE
      WHERE    (Salary BETWEEN 30000 AND 40000) AND Dno = 5;
```

Sắp thứ tự kết quả truy vấn – ORDER BY

Query 15: Lấy danh sách các nhân viên và các dự án họ đang làm việc, sắp thứ tự theo phòng ban và, trong phạm vi mỗi phòng ban, tiếp tục sắp theo họ, tên theo thứ tự alphabet.

Q15:
```
SELECT   D.Dname, E.Lname, E.Fname, P.Pname
FROM     DEPARTMENT AS D, EMPLOYEE AS E, WORKS_ON AS W,
         PROJECT AS P
WHERE    D.Dnumber = E.Dno AND E.Ssn = W.Essn AND W.Pno = P.Pnumber
ORDER BY D.Name, E.Lname, E.Fname;
```

- Cách sắp mặc định là tăng dần: **ASC**.
- Sắp giảm dần: **DESC**

```
…
ORDER BY D.Name DESC, E.Lname ASC, E.Fname ASC;
```

Lệnh INSERT, DELETE, UPDATE

INSERT

Chèn thêm một bộ dữ liệu (một dòng) và quan hệ (bảng).

```
INSERT INTO <table_name>
VALUES      (<list of values for the tuple>);
```

- Các giá trị phải được sắp theo đúng thứ tự của các thuộc tính đã xác định trong lệnh tạo quan hệ CREATE TABLE.
- Có thể khai báo tường minh các tên thuộc tính tương ứng với các giá trị sẽ được chèn vào quan hệ.
- Phải cung cấp giá trị cho các thuộc tính được khai báo NOT NULL hoặc không chỉ định giá trị mặc định.
  Có thể không cung cấp giá trị cho các thuộc tính cho phép nhận giá trị NULL hoặc có chỉ định giá trị mặc định.
- Cho phép chèn một lúc nhiều bộ, cách nhau bởi dấu ‘,’

INSERT – Ví dụ

```
CREATE TABLE EMPLOYEE
        ( Fname          VARCHAR(15)        NOT NULL,
          Minit          CHAR,
          Lname          VARCHAR(15)        NOT NULL,
          Ssn            CHAR(9)            NOT NULL,
          Bdate          DATE,
          Address        VARCHAR(30),
          Sex            CHAR,
          Salary         DECIMAL(10,2),
          Super_ssn      CHAR(9),
          Dno            INT                NOT NULL,
        PRIMARY KEY (Ssn),
```

U1: **INSERT INTO** EMPLOYEE
VALUES (‘Richard’, ‘K’, ‘Marini’, ‘653298653’, ‘1962-12-30’,
’98 Oak Forest, Katy, TX’, ‘M’, 37000, ‘653298653’, 4);

U1A: **INSERT INTO** EMPLOYEE (Fname, Lname, Dno, Ssn)
VALUES (‘Richard’, ‘Marini’, 4, ‘653298653’);

INSERT – Kiểm tra ràng buộc toàn vẹn

U2: **INSERT INTO** EMPLOYEE (Fname, Lname, Ssn, Dno)
VALUES (‘Robert’, ‘Hatcher’, ‘980760540’, 2)**;**

- U2 bị từ chối thực hiện do vi phạm ràng buộc toàn vẹn tham chiếu khi chèn với phòng ban Dno = 2 không tồn tại trong DEPARTMENT

U2A: **INSERT INTO** EMPLOYEE (Fname, Lname, Dno)
VALUES (‘Robert’, ‘Hatcher’, 5)**;**

- U2A bị từ chối thực hiện do vi phạm ràng buộc giá trị của Ssn phải là NOT NULL

Chèn dữ liệu từ kết quả truy vấn

```
U3A:  CREATE  TABLE      WORKD_ON_INFO
      ( Emp_name          VARCHAR(15),
        Proj_name         VARCHAR(15),
        Hours_per_week    DECIMAL(3,1) );
```

```
U3B:  INSERT  INTO  WORKS_ON_INFO ( Emp_name, Proj_name,
                      Hours_per_week )
      SELECT        E.Lname, P.Pname, W.Hours
      FROM          PROJECT AS P, WORKS_ON AS W, EMPLOYEE AS E
      WHERE         P.Pnumber = W.Pno AND W.Essn = E.Ssn;
```

Chèn dữ liệu từ tập tin ngoài

- Tập tin phải định dạng dữ liệu như các dòng trong bảng dữ liệu
  Ví dụ: tập tin .csv

- Viết chương trình đọc nội dung các dòng dữ liệu từ tập tin, và thực hiện chèn từng dòng vào bảng.

Chèn từ bảng sẵn có cùng dữ liệu

```
CREATE  TABLE     D5EMPS  LIKE  EMPLOYEE
(SELECT           E.*
 FROM             EMPLOYEE AS E
 WHERE            E.Dno = 5) WITH DATA;
```

DELETE

Xóa các bộ dữ liệu khỏi một quan hệ.

```
DELETE  FROM   <table_name>
WHERE          <condition>;
```

Chú ý:

- Khi thực hiện xóa ở các quan hệ có liên hệ tham chiếu sẽ kích hoạt cơ chế kiểm tra ràng buộc toàn vẹn tham chiếu, nếu được định nghĩa.
  → thực hiện xóa lan truyền.
- Tùy thuộc điều kiện kiểm tra, số lượng bộ bị xóa có thể từ 0 đến toàn bộ bộ dữ liệu có trong bảng.
  Không cung cấp điều kiện kiểm tra **WHERE**, tất cả bộ dữ liệu sẽ bị xóa.
- Khi tất cả bộ dữ liệu bị xóa, bảng trở thành bảng rỗng, vẫn tồn tại trong CSDL.
  → để xóa bảng, dùng lệnh **DROP TABLE**.

DELETE – Một số ví dụ

U4A: **DELETE FROM** EMPLOYEE
WHERE Lname = ‘Brown’**;**

U4B: **DELETE FROM** EMPLOYEE
WHERE Ssn = ‘123456789’**;**

U4A: **DELETE FROM** EMPLOYEE
WHERE Dno = 5**;**

U4A: **DELETE FROM** EMPLOYEE**;**

UPDATE

Thay đổi giá trị thuộc tính của một hay nhiều bộ dữ liệu.

```
UPDATE    <table_name>
SET       {<attribute_name> = <value>,}
WHERE      <condition>;
```

- **SET** : xác định các thuộc tính cần thay đổi và giá trị mới.

Chú ý:

- Thực hiện thay đổi giá trị khóa chính có thể phải thay đổi tới các giá trị khóa ngoại ở các quan hệ có liên hệ tham chiếu.
- Việc thay đổi áp dụng với một hay nhiều bộ dữ liệu thỏa mãn điều kiện kiểm tra.
- **UPDATE** chỉ áp dụng cho từng bảng.

UPDATE – Ví dụ

U5: **UPDATE** PROJECT
SET Plocation = ‘Bellaire’, Dnum = 5
WHERE Pnumber = 10**;**

U6: **UPDATE** EMPLOYEE
SET Salary = Salary * 1.1
WHERE Dno = 5**;**

Kết thúc phần này

Bài tập chương 6.