CƠ SỞ DỮ LIỆU

Chương 3

Chương 3 - Ngôn ngữ định nghĩa và thao tác dữ liệu đối với mô hình quan hệ

NỘI DUNG:

- Các cách tiếp cận đối với thiết kế ngôn ngữ của CSDL quan hệ
  - Giới thiệu một số ngôn ngữ và phân loại
  - ➢ So sánh và đánh giá
- Một số ngôn ngữ dữ liệu mức cao
  - QBE (***Q****uery* ***B****y* ***E****xample*)
  - SQL (***S****tructured* ***Q****uery* ***L****anguage*)
- Kết luận

Ví dụ

- Tìm các sinh viên đăng ký khoá học có mã số 113
  - Tìm các giá trị SID trong bảng Enrol có Course tương ứng là 113
  - Đưa các bộ của bảng Student có SID trong các giá trị tìm thấy ở trên

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Enrol

SID	Course
3936	101
1108	113
8507	101

Course

No	Name	Dept
113	BCS	CSCE
101	MCS	CSCE

Phân loại các ngôn ngữ truy vấn

- Ngôn ngữ đại số
  - 1 câu hỏi = 1 tập các phép toán trên các quan hệ
  - Được biểu diễn bởi một biểu thức đại số (quan hệ)
- Ngôn ngữ tính toán vị từ
  - 1 câu hỏi = 1 mô tả của các bộ mong muốn
  - Được đặc tả bởi một vị từ mà các bộ phải thoả mãn
  - Phân biệt 2 lớp:
    - ngôn ngữ tính toán vị từ biến bộ
    - ngôn ngữ tính toán vị từ biến miền

Ngôn ngữ đại số quan hệ

- Gồm các phép toán tương ứng với các thao tác trên các quan hệ
- Mỗi phép toán
  - Đầu vào: một hay nhiều quan hệ
  - Đầu ra: một quan hệ
- Biểu thức đại số quan hệ = chuỗi các phép toán
- Kết quả thực hiện một biểu thức đại số là một quan hệ
- Được cài đặt trong phần lớn các hệ CSDL hiện nay

Các phép toán đại số quan hệ

- Phép toán quan hệ
  - Phép chiếu (*projection*)
  - Phép chọn (*selection*)
  - Phép kết nối (*join*)
  - Phép chia (*division*)
- Phép toán tập hợp
  - Phép hợp (*union*)
  - Phép giao (*intersection*)
  - Phép trừ (*difference*)
  - Phép tích đề-các (*cartesian product*)

Phép toán tập hợp

- Định nghĩa: Quan hệ khả hợp

  Hai quan hệ r và s được gọi là khả hợp nếu chúng được xác định trên cùng một miền giá trị

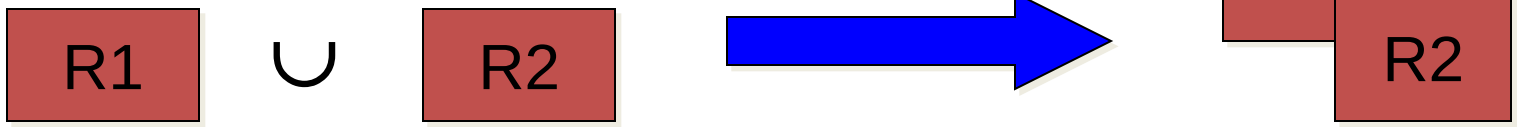
  r xác định trên $D_1 x\ D_2\ x \ldots x\ D_n$

  s xác định trên $D'_1 x\ D'_2\ x \ldots x\ D'_m$

  → $D_i = D'_i$ và $n=m$

Phép hợp

- Đ/n: gồm các bộ thuộc ít nhất một trong hai quan hệ đầu vào
- Hai quan hệ đầu vào phải là khả hợp
- Cú pháp: $R = R_1 \cup R_2$



Subject1

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS

∪

Subject2

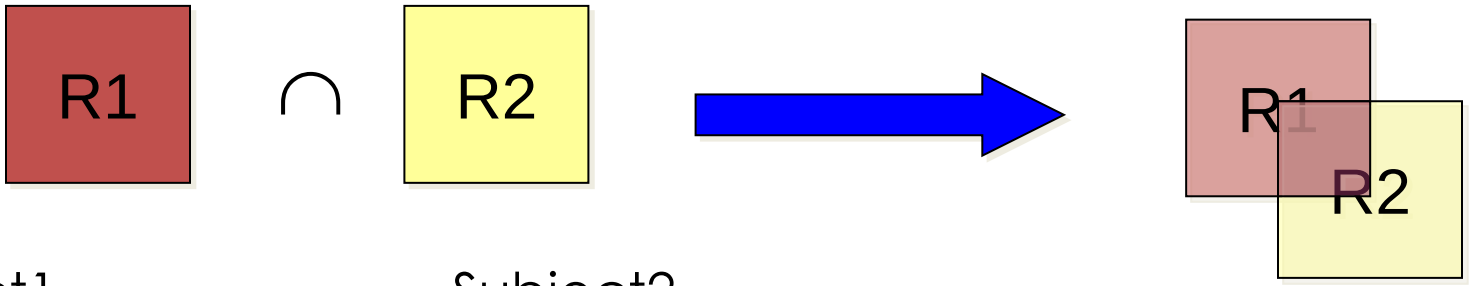
Name	Course
DataMining	MCS
Writing	BCS

Kết quả

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS
DataMining	MCS
Writing	BCS

Phép giao

- Đ/n: gồm các bộ thuộc cả hai quan hệ đầu vào
- Hai quan hệ đầu vào phải là khả hợp
- Cú pháp: $R_1 \cap R_2$



Subject1

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS

∩

Subject2

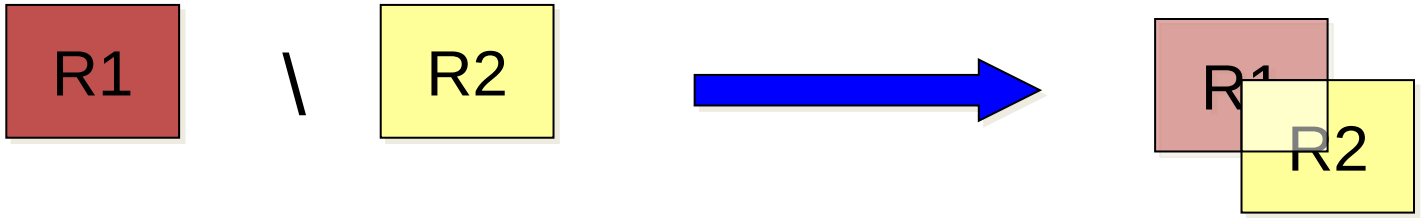
Name	Course
DataMining	MCS
Database	MCS
Systems	BCS
Writing	BCS

Kết quả

Name	Course
Systems	BCS
Database	MCS

Phép trừ

- Đ/n: gồm các bộ thuộc quan hệ thứ nhất nhưng không thuộc quan hệ thứ hai
  - Hai quan hệ phải là khả hợp
- Cú pháp: $R_1 \setminus R_2$ hoặc $R_1 - R_2$



Subject1

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS

\

Subject2

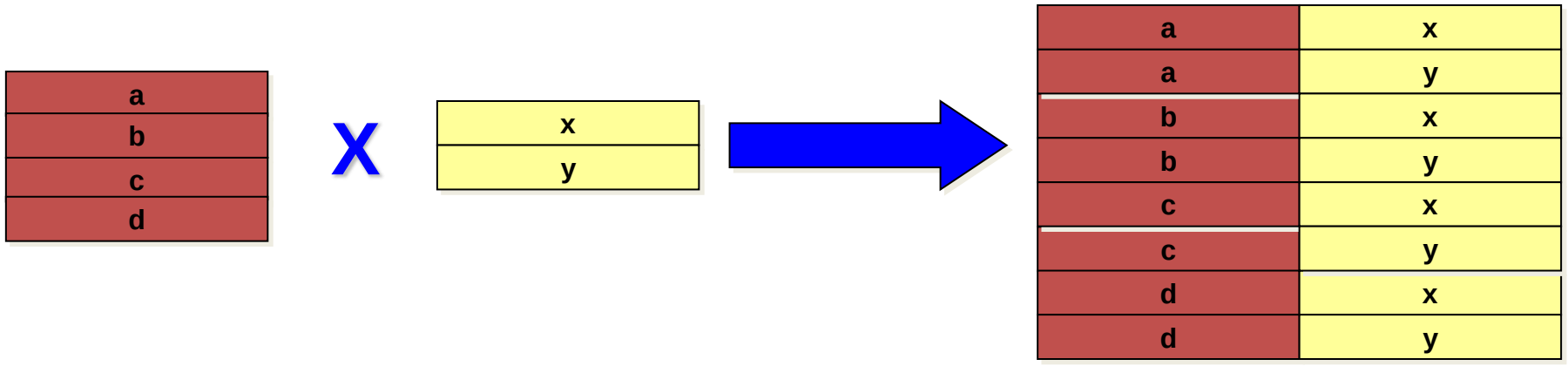
Name	Course
DataMining	MCS
Database	MCS
Systems	BCS
Writing	BCS

Kết quả

Name	Course
Database	BCS
Algebra	MCS

Phép tích Đề-các

- Đ/n: là kết nối giữa từng bộ của quan hệ thứ nhất với mỗi bộ của quan hệ thứ hai
- Cú pháp: $R = R_1 \; x \; R_2$



Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

X

Sport

SportID	Sport
05	Swimming
09	Dancing

Student_Sport

Id	Name	Suburb	SportID	Sport
1108	Robert	Kew	05	Swimming
1108	Robert	Kew	09	Dancing
3936	Glen	Bundoora	05	Swimming
3936	Glen	Bundoora	09	Dancing
8507	Norman	Bundoora	05	Swimming
8507	Norman	Bundoora	09	Dancing
8452	Mary	Balwyn	05	Swimming
8452	Mary	Balwyn	09	Dancing

Phép chiếu

- Đ/n: Lựa chọn một số thuộc tính từ một quan hệ.
- Cú pháp: $\prod_{A1,A2,\ldots}(R)$

❖Ví dụ: đưa ra danh sách tên của tất cả các sinh viên

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

$\prod_{Name}(Student)$

Name
Robert
Glen
Norman
Mary

Phép chọn

- Đ/n: Lựa chọn các bộ trong một quan hệ thoả mãn điều kiện cho trước.
- Cú pháp: $\sigma_{<condition>}(R)$

- Ví dụ: đưa ra danh sách những sinh viên sống ở Bundoora

Student

$\sigma_{suburb='Bundoora'}(Student)$

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Id	Name	Suburb
3936	Glen	Bundoora
8507	Norman	Bundoora

Phép chọn - Điều kiện ?

- Điều kiện chọn còn gọi là biểu thức chọn.
- Biểu thức chọn F: một tổ hợp logic của các toán hạng. Mỗi toán hạng là một phép so sánh đơn giản giữa hai biến là hai thuộc tính hoặc giữa một biến là một thuộc tính và một giá trị hằng.
  - Các phép so sánh trong F: $<, =, >, \leq, \geq, \neq$
  - Các phép toán logic trong F: $\wedge, \vee, \neg$

Ví dụ: chọn và chiếu

- Đưa ra tên của các sinh viên sống ở Bundoora

$$\prod_{Name}(\sigma_{suburb='Bundoora'}Student)$$

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Name
Glen
Norman

Phép kết nối (join) hai quan hệ r và s

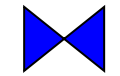
- Khái niệm ghép bộ: $u = (a_1,..,a_n)$; $v=(b_1,..,b_m)$
  $(u,v) = (a_1,..,a_n,b_1,..,b_m)$
- Phép kết nối hai quan hệ thực chất là phép ghép các cặp bộ của hai quan hệ thỏa mãn một điều kiện nào đó trên chúng.
- Biểu thức kết nối là phép hội của các toán hạng, mỗi toán hạng là một phép so sánh đơn giản giữa một thuộc tính của quan hệ r và một thuộc tính của quan hệ s.
- Cú pháp: $R1 \bowtie_{<<\text{điều kiện}>>} R2$

- Đưa ra danh sách các sinh viên và mã khoá học mà sinh viên đó tham gia:

$$\text{Student} \bowtie_{\text{Id=SID}} \text{Enrol}$$

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn



Id=SID

Enrol

SID	Course
3936	101
1108	113
8507	101

Kết quả

SID	Id	Name	Suburb	Course
1108	1108	Robert	Kew	113
3936	3936	Glen	Bundoora	101
8507	8507	Norman	Bundoora	101

Phép kết nối bằng - kết nối tự nhiên

- Định nghĩa: Nếu phép so sánh trong điều kiện kết nối là phép so sánh bằng thì kết nối gọi là **kết nối bằng**
- Định nghĩa: Phép kết nối bằng trên các thuộc tính cùng tên của hai quan hệ và sau khi kết nối một thuộc tính trong một cặp thuộc tính trùng tên đó sẽ bị loại khỏi quan hệ kết quả thì phép kết nối gọi là **kết nối tự nhiên**
- Cú pháp phép kết nối tự nhiên: $R_1 * R_2$

Takes

SID	SNO
1108	21
1108	23
8507	23
8507	29

*

Enrol

SID	Course
3936	101
1108	113
8507	101

SID	SNO	Course
1108	21	113
1108	23	113
8507	23	101
8507	29	101

- Đưa ra tên của các sinh viên sống ở Bundoora và mã khoá học mà sinh viên đó đăng ký:

$$\prod_{Name,Course}(\sigma_{Suburb='Bundoora'}(Student \triangleright\!\triangleleft_{Id=SID} Enrol))$$

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Enrol

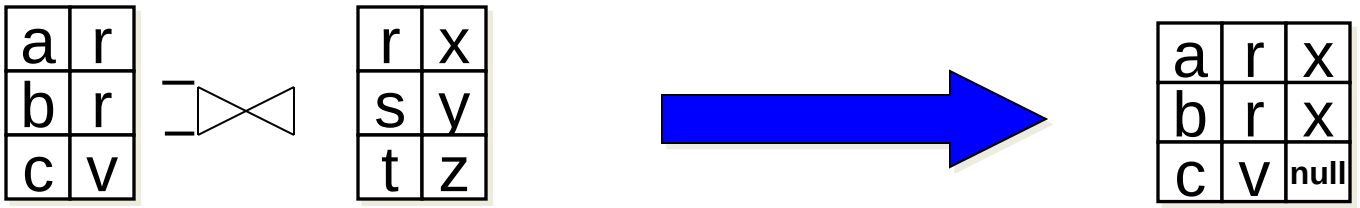
SID	Course
3936	101
1108	113
8507	101

Kết quả

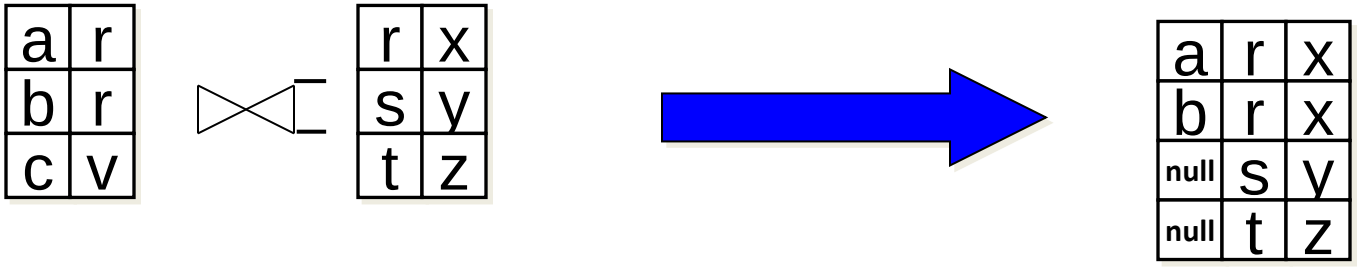
Name	Course
Glen	101
Norman	101

Phép kết nối ngoài

- Phép kết nối ngoài trái



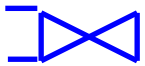
- Phép kết nối ngoài phải



- Đưa ra danh sách các sinh viên và mã khoá học mà sinh viên đó đăng ký nếu có

Student

ID	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn



Enrol

SID	Course
3936	101
1108	113
8507	101

Kết quả

ID	Name	Suburb	Course
1108	Robert	Kew	113
3936	Glen	Bundoora	101
8507	Norman	Bundoora	101
8452	Mary	Balwyn	**null**

Phép chia

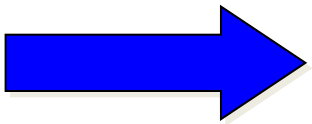
- Định nghĩa: Phép chia giữa một quan hệ r bậc n và quan hệ s bậc m (m<n) với sơ đồ quan hệ của s là tập con của sơ đồ quan hệ của r là một tập các (n-m) – bộ t sao cho khi ghép mọi bộ thuộc s với t thì ta đều có một bộ thuộc r
- Cú pháp: $R = R_1 : R_2$

$$\mathbf{r \div s = \{ t \mid \forall\ v \in s \Rightarrow (t,v) \in r\}}$$

a	x
a	y
a	z
b	x
c	y

:

x
z



a

- Ví dụ: Đưa ra môn học được dạy ở tất cả các khoá học

Subject

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS

:

Course

Course
BCS
MCS

Kết quả

Name
Database

Bài tập

- Cho CSDL gồm 3 quan hệ sau: S (Các hãng cung ứng), P (các mặt hàng), SP (các sự cung ứng).

S (S# SNAME STATUS CITY)

S#	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	10	Paris
S3	Black	30	Paris

SP (S# P# QTY)

S#	P#	QTY
S1	P1	300
S1	P2	200
S1	P3	400
S2	P1	300
S2	P2	400
S3	P2	200

P (P# PNAME COLOR WEIGHT CITY)

P#	PNAME	COLOR	WEIGHT	CITY
P1	Nut	red	12	London
P2	Bolt	green	17	Paris
P3	Screw	blue	17	Rom
P4	Screw	red	14	London

- Biểu diễn các truy vấn sau bằng đại số quan hệ:
  - Đưa ra danh sách các mặt hàng màu đỏ
  - Cho biết S# của các hãng cung ứng mặt hàng 'P1' hoặc 'P2'
  - Liệt kê S# của các hãng cung ứng cả hai mặt hàng 'P1' và 'P2'
  - Đưa ra S# của các hãng cung ứng ít nhất một mặt hàng màu đỏ
  - Đưa ra S# của các hãng cung ứng tất cả các mặt hàng.

Bài tập

- Cho các quan hệ sau:

Supplier

sid	sname	size	city
S1	Dustin	100	London
S2	Rusty	70	Paris
S3	Lubber	120	London

Product

pid	pname	colour
P1	Screw	red
P2	Screw	green
P3	Nut	red
P4	Bolt	blue

SupplyProduct

sid	pid	quantity
S1	P1	500
S1	P2	400
S1	P3	100
S2	P2	200
S3	P4	100
S2	P3	155

- Biểu diễn các truy vấn sau bằng biểu thức đại số quan hệ:
  1) Đưa ra {sid,sname,size,city} của các Supplier có trụ sở tại London
  2) Đưa ra {pname} của tất cả các mặt hàng
  3) Đưa ra {sid} của các Supplier cung cấp mặt hàng P1 hoặc P2
  4) Đưa ra {sname} của các Supplier cung cấp mặt hàng P3
  5) Đưa ra {sname} của các hãng cung ứng ít nhất một mặt hàng màu đỏ
  6) Đưa ra {sid} của các hãng cung ứng tất cả các mặt hàng màu đỏ
  7) Đưa ra {sname} của các hãng có cung ứng mặt hàng màu đỏ hoặc màu xanh
  8) Đưa ra {sname} của các hãng cung ứng ít nhất một mặt hàng màu đỏ và ít nhất một mặt hàng màu xanh
  9) Đưa ra {sid} của các hãng không cung ứng mặt hàng nào

Ngôn ngữ QBE

QBE (*Query-By-Example*)

- Là một ngôn ngữ truy vấn dữ liệu
- Các câu truy vấn được thiết lập bởi một giao diện đồ hoạ
- Phù hợp với các câu truy vấn đơn giản, tham chiếu đến ít bảng
- Một số sản phẩm: IBM™ (IBM Query Management Facility), Paradox, MS. Access, ...

Truy vấn trên một quan hệ

- P.~ Print

Student	ID	Name	Suburb
		P._x	Bundoora

- Biểu thức đại số quan hệ tương đương

$$\sigma_{suburb='Bundoora'}(Student)$$

- Lựa chọn tất cả các cột

Student	ID	Name	Suburb
P.			Bundoora

- Sắp xếp

Student	ID	Name	Suburb
		P.AO(1)	P.AO(2)

- AO: sắp xếp tăng dần
- DO: sắp xếp giảm dần

Truy vấn trên nhiều quan hệ

- Đưa ra tên của các sinh viên có đăng ký ít nhất một khoá học

Student	ID	Name	Suburb
	_id	P._name	

Enrol	SID	Course
	_id	

- Đưa ra tên các sinh viên không đăng ký một khoá học nào

Student	ID	Name	Suburb
	_id	P._name	

Enrol	SID	Course
¬	_id	

Các tính toán tập hợp

- Các phép toán: AVG, COUNT, MAX, MIN, SUM
- Ví dụ: đưa ra tên các thành phố và số lượng sinh viên đến từ thành phố đó

Student	ID	Name	Suburb	
	_id		G.P.	P.COUNT._id

- G. ~ Grouping

Hộp điều kiện

- Được sử dụng để biểu diễn
  - Điều kiện trên nhiều hơn một thuộc tính
  - Điều kiện trên các trường tính toán tập hợp
- Ví dụ: đưa ra danh sách các thành phố có nhiều hơn 5 sinh viên

Student	ID	Name	Suburb	Condition
	_id		G.P.	COUNT._id > 5

Các thao tác thay đổi dữ liệu

- Xóa

Student	ID	Name	Suburb
D.	1108		

- Thêm

Student	ID	Name	Suburb
I.	1179	David	Evry

- Sửa

Student	ID	Name	Suburb
	1179		U.Paris

Tính đầy đủ của QBE

- Có thể biểu diễn cả 5 phép toán đại số cơ sở (σ,Π,$\cup$,\,x)

Định nghĩa dữ liệu trong QBE

- Sử dụng cùng qui cách và giao diện đồ họa như đối với truy vấn.

I.Student	I.	ID	Name	Suburb
KEY	I.	Y	N	N
TYPE	I.	CHAR(5)	CHAR(30)	CHAR(30)
DOMAIN	I.	Sid	SName	Surb
INVERSION	I.	Y	N	N

- Các khung nhìn

I.View V	I.	ID	Name	Course
	I.	_id	_name	_course

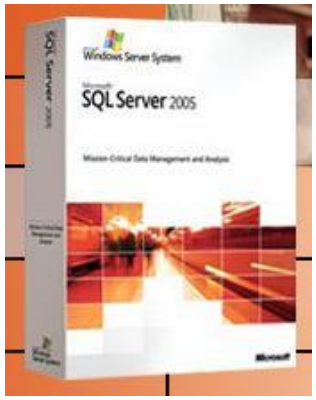
Student	ID	Name	Suburb
	_id	_name	

Enrol	SID	Course
	_id	_course

Ngôn ngữ SQL

SQL (Structured Query Language)

- 1975: SEQUEL
  - System-R
- 1976: SEQUEL2
- 1978/79: SQL
  - System-R
- 1986: chuẩn SQL-86
- 1989: chuẩn SQL-89
- 1992: chuẩn SQL-92
- 1996: chuẩn SQL-96



Các thành phần của SQL

- Ngôn ngữ định nghĩa dữ liệu (***D**ata **D**efinition **L**anguage*)
  - Cấu trúc các bảng CSDL
  - Các mối liên hệ của dữ liệu
  - Quy tắc, ràng buộc áp đặt lên dữ liệu
- Ngôn ngữ thao tác dữ liệu (***D**ata **M**anipulation **L**anguage*)
  - Thêm, xoá, sửa dữ liệu trong CSDL
  - Truy vấn dữ liệu
- Ngôn ngữ điều khiển dữ liệu (***D**ata **C**ontrol **L**anguage*)
  - Khai báo bảo mật thông tin
  - Quyền hạn của người dùng trong khai thác CSDL

Ngôn ngữ định nghĩa dữ liệu

- Các thông tin được định nghĩa bao gồm
  - Sơ đồ quan hệ
  - Kiểu dữ liệu hay miền giá trị của mỗi thuộc tính
  - Các ràng buộc toàn vẹn
  - Các chỉ số đối với mỗi bảng
  - Thông tin an toàn và ủy quyền đối với mỗi bảng
  - Cấu trúc lưu trữ vật lý của mỗi bảng trên đĩa

➢ Được biểu diễn bởi các lệnh định nghĩa dữ liệu

Quy ước đặt tên và kiểu dữ liệu

- Quy ước đặt tên
  - 32 ký tự: chữ cái, số, dấu _
- Kiểu dữ liệu (SQL-92)
  - char(n)
  - varchar(n)
  - int
  - smallint
  - numeric(p,d)
  - real, double
  - float(n)
  - date
  - time

Cú pháp

- Tạo bảng

CREATE TABLE tên-bảng(
    cột-1 kiểu-dữ-liệu-1 [**NOT NULL**], ...,
    cột-2 kiểu-dữ-liệu-2 [**NOT NULL**], ...,
        .....
    [**CONSTRAINT** tên-ràng-buộc kiểu-ràng-buộc]
    ...
);

- Xoá bảng

DROP TABLE tên-bảng

```
CREATE TABLE Supplier(
   sid char(4) NOT NULL,
   sname varchar(30) NOT NULL,
   size smallint,
   city varchar(20),
   CONSTRAINT KhoachinhS primary key(sid)
);

CREATE TABLE Product(
   pid char(4) NOT NULL,
   pname varchar(30) NOT NULL,
   colour char(8),
   weight int,
    city varchar(20),
   CONSTRAINT KhoachinhP primary key(pid)
);
```

```
CREATE TABLE SupplyProduct(
        sid char(4) NOT NULL,
        pid char(4) NOT NULL,
        quantity smallint,
        primary key(sid,pid),
        foreign key(sid) references Supplier(sid),
        foreign key(pid) references Product(pid),
        check(quantity >0)
  );
```

Kiểu ràng buộc

- Ràng buộc toàn vẹn về giá trị miền

  CONSTRAINT <tên ràng buộc>

  CHECK <điều kiện>

- Ràng buộc toàn vẹn về khoá ngoại hay phụ thuộc tồn tại

  CONSTRAINT <tên ràng buộc>

  FOREIGN KEY (fk_i) **REFERENCES** tên-bảng(k_i);

Thêm/xoá/sửa cột của các bảng

- Thêm

  ALTER TABLE <tên bảng>

  ADD COLUMN <tên cột> <kiểu dữ liệu> [NOT NULL];

- Xoá

  ALTER TABLE <tên bảng>

  DROP COLUMN <tên cột>;

- Sửa

  ALTER TABLE <tên bảng>

  CHANGE COLUMN <tên cột> TO <kiểu dữ liệu mới>;

- ALTER TABLE SupplyProduct ADD COLUMN price real NOT NULL;
- ALTER TABLE SupplyProduct DROP COLUMN price;
- ALTER TABLE Supplier CHANGE COLUMN sname TO varchar(20);

Thêm/xóa các ràng buộc

- Thêm

  ALTER TABLE <tên bảng>

  ADD CONSTRAINT <tên ràng buộc>
  <kiểu ràng buộc>

- Xóa

  ALTER TABLE <tên bảng>

  DROP CONSTRAINT <tên ràng buộc>

Ngôn ngữ thao tác dữ liệu

- Cú pháp câu lệnh SQL:

SELECT [**DISTINCT**] <DS cột>|*|<Biểu thức>|<Hàm TV>
FROM <DS bảng>
[**WHERE** <Điều kiện tìm kiếm>]
[**GROUP BY** <DS cột> [**HAVING** <Điều kiện>]]
[**ORDER BY** <Danh sách cột> [**ASC**|**DESC**]]
[**UNION** |**INTERSECT**| **MINUS** <Câu truy vấn khác>]

Truy vấn không điều kiện trên một bảng

- Tìm thông tin từ các cột của bảng
  - **SELECT** <DS cột>
    FROM <Tên bảng>;
  - **SELECT** *
    FROM <Tên bảng>;
- Ví dụ

```
SELECT    Name
FROM      Student;
```

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Robert	Bundoora
8452	Mary	Balwyn

$$\prod_{Name}(Student)$$

Name
Robert
Glen
Mary

- Đưa ra tên của các mặt hàng

  SELECT pname FROM Product;
- Đưa ra tên khác nhau của các mặt hàng

  SELECT DISTINCT pname

  FROM Product;
- Đưa ra toàn bộ thông tin về các hãng cung ứng

  SELECT * FROM Supplier;
- Đưa ra mã số hãng cung ứng, mã mặt hàng được cung ứng và 10 lần số lượng mặt hàng đã được cung ứng

  SELECT sid, pid, quantity*10

  FROM SupplyProduct;

Truy vấn có điều kiện trên một bảng

- Chọn các bản ghi (dòng)

```
SELECT    <DS cột>
FROM      <Tên bảng>
WHERE     <Điều kiện tìm kiếm>
```

- Ví dụ

```
SELECT    *
FROM      Student
WHERE     Suburb='Bundoora' ;
```

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Robert	Bundoora
8452	Mary	Balwyn

$\sigma_{Suburb='Bundoora'}(Student)$

Id	Name	Suburb
3936	Glen	Bundoora
8507	Robert	Bundoora

- Đưa ra tên của các hãng cung ứng có trụ sở tại London

```
SELECT sname FROM Supplier
WHERE city = 'London';
```

- Đưa ra mã số và tên của các hãng cung ứng nằm ở London và có số nhân viên lớn hơn 75

```
SELECT sid, sname FROM Supplier
WHERE city = 'London' AND size > 75;
```

Biểu diễn điều kiện lựa chọn

- Các phép toán quan hệ: =, !=, <, >, <=, >=
- Các phép toán logic: NOT, AND, OR
- Phép toán phạm vi: BETWEEN, IN, LIKE
  - Kiểu dữ liệu số
    - attr **BETWEEN** val1 **AND** val2 (⇔ (attr>=val1) and (attr<=val2))
    - attr **IN** (val1, val2, ...) (⇔ (attr=val1) or (attr=val2) or ...)
  - Kiểu dữ liệu xâu
    - **LIKE**: sử dụng đối sánh mẫu xâu với các ký tự thay thế cho 1 ký tự bất kỳ (_, ?), thay thế cho 1 xâu ký tự bất kỳ (*, %)

    (PostGreSQL sử dụng dấu % và dấu _)

- Đưa ra thông tin của các hãng cung ứng có số nhân viên trong khoảng từ 100 đến 150

```
SELECT * FROM Supplier
WHERE size BETWEEN 100 AND 150;
```

- Đưa ra mã số của hãng cung ứng mặt hàng P1 hoặc P2
  - Cách 1:

```
SELECT sid FROM SupplyProduct
WHERE pid = 'P1' OR pid = 'P2';
```

  - Cách 2:

```
SELECT sid FROM SupplyProduct
WHERE pid IN ('P1', 'P2');
```

- Đưa ra thông tin của hãng sản xuất có trụ sở đặt tại thành phố bắt đầu bằng chữ New (New York, New Jersey, New Mexico, New Hampshire)

```
SELECT * FROM SUPPLIER
WHERE city LIKE 'New%';
```

Loại trừ các bản ghi trùng nhau

- Từ khoá **DISTINCT**

  SELECT DISTINCT <DS cột>

  FROM <DS bảng>

- Ví dụ: đưa ra danh sách tên các khoa (Dept) tương ứng với các khoá học (Course). Mỗi giá trị chỉ hiện thị một lần

  SELECT DISTINCT Dept

  FROM Course

Truy vấn có sử dụng phép toán đổi tên

- SQL cho phép đổi tên các bảng và các cột trong một câu truy vấn (sau mệnh đề SELECT và FROM) sử dụng cấu trúc:
- <tên cũ> AS <tên mới>
  - Đưa ra tên và số nhân viên của các hãng cung ứng ở Paris

```
SELECT sname AS HangOParis, size AS SoNhanVien
FROM Supplier
WHERE city = 'Paris';
SELECT    SID , Stud.Name as SName,
          Sub.Name as Subject
FROM      Student as Stud,Takes,
          Subject as Sub
WHERE     (Id=SID) and (SNO = No)
```

Truy vấn phức tạp trên nhiều bảng

- Điều kiện kết nối

```
SELECT  <DS cột>
FROM    <DS bảng>
WHERE   <Điều kiện tìm kiếm>
```

- Ví dụ: đưa ra danh sách mã sinh viên (Id), tên sinh viên (Name), thành phố (Suburb), mã khoá học (Course) mà các sinh viên đã đăng ký

```
SELECT  Id, Name, Suburb, Course
FROM    Student, Enrol
WHERE   Id=SID;
```

Kết nối tự nhiên

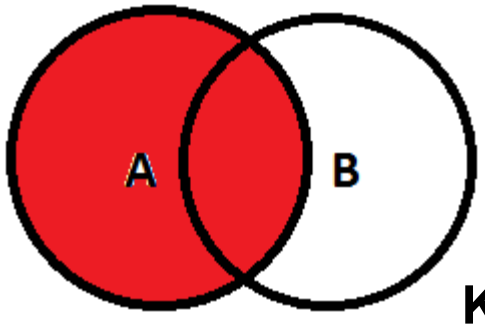
```
SELECT <DS cột>
FROM A, B, C
WHERE A.CộtX = B.CộtX AND B.CộtY = C.CộtY
```

```
SELECT <DS cột>
FROM A NATURAL JOIN B NATURAL JOIN C
```

Kết nối hai bảng

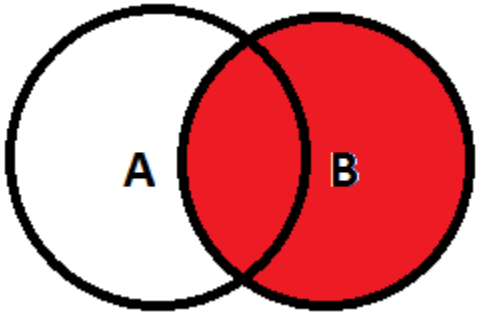
Kết nối ngoài trái

SELECT <DS cột>
FROM A LEFT JOIN B
ON A.Key = B.Key



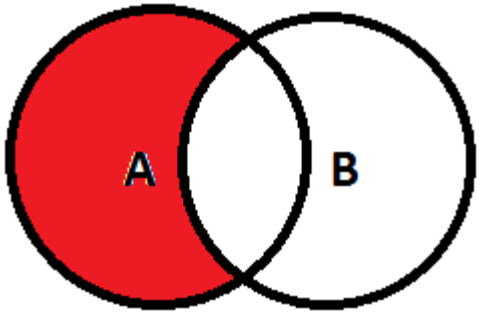
Kết nối ngoài phải

SELECT <DS cột>
FROM A RIGHT JOIN B
ON A.Key = B.Key



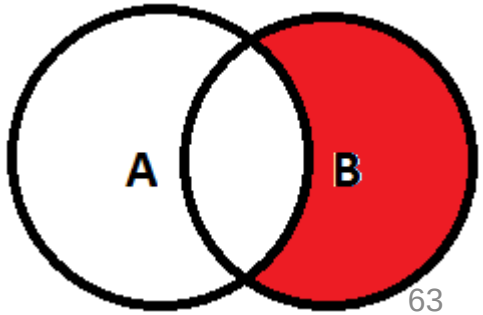
Kết nối ngoài trái

SELECT <DS cột>
FROM A LEFT JOIN B
ON A.Key = B.Key
WHERE B.Key IS NULL



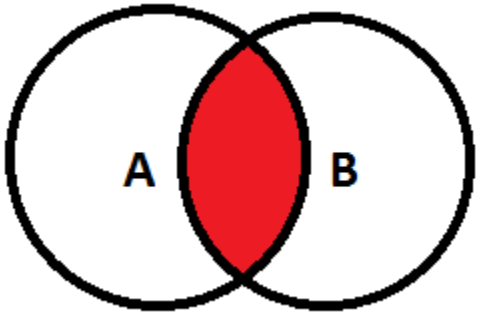
Kết nối ngoài phải

SELECT <DS cột>
FROM A RIGHT JOIN B
ON A.Key = B.Key
WHERE A.Key IS NULL



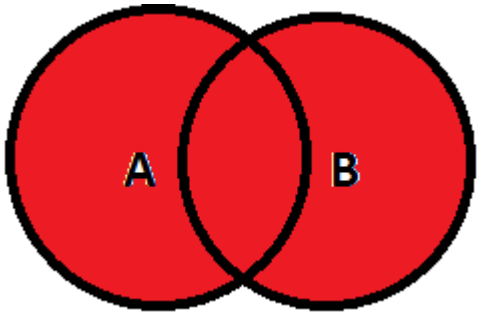
Kết nối trong

SELECT <DS cột>
FROM A INNER JOIN B
ON A.Key = B.Key



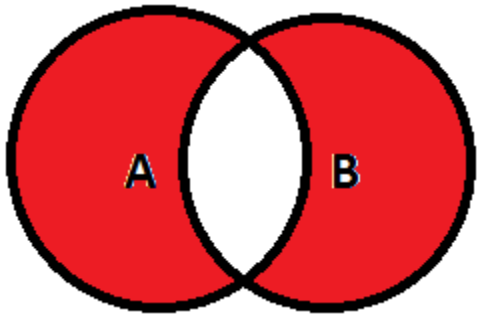
Kết nối OUTER JOIN

SELECT <DS cột>
FROM A FULL OUTER JOIN B
ON A.Key = B.Key



Kết nối OUTER JOIN

SELECT <DS cột>
FROM A FULL OUTER JOIN B
ON A.Key = B.Key
WHERE A.Key IS NULL OR
B.Key IS NULL



- Đưa ra tên của hãng có cung ứng mặt hàng P1

```
SELECT sname
FROM Supplier S, SupplyProduct SP
WHERE S.sid = SP.sid AND SP.pid = 'P1';
```

- Đưa ra tên và mã số của hãng cung ứng ít nhất một mặt hàng màu đỏ

```
SELECT sname, S.sid
FROM Supplier S, SupplyProduct SP, Product P
WHERE S.sid = SP.sid AND P.pid = SP.pid AND P.colour =
    'red';
```

Tìm kiếm có sắp xếp

- Sắp xếp các bản ghi kết quả theo một thứ tự cho trước

```
SELECT    <DS cột>
FROM      <DS bảng>
[WHERE    <Điều kiện tìm kiếm>]
ORDER BY  <DS cột> [ASC|DESC]
```

- Ví dụ: đưa ra danh sách tên các sinh viên theo thứ tự tăng dần

```
SELECT    Name
FROM      Student
ORDER BY Name ASC
```

Phân nhóm các bản ghi kết quả

- Phân nhóm các bản ghi kết quả theo giá trị của một hoặc nhiều thuộc tính

```
SELECT      <DS cột>
FROM        <DS bảng>
[WHERE      <Điều kiện tìm kiếm>]
[GROUP BY   <DS cột>]
```

- Cột được chỉ ra trong mệnh đề GroupBy được sử dụng làm cơ sở để chia nhóm. Cột này cũng bắt buộc phải được chỉ ra trong mệnh đề Select
- Ví dụ đưa ra tên các sinh viên nhóm theo thành phố của sinh viên đó

```
SELECT Suburb, Name
FROM Student
GROUP BY Suburb
```

```
SELECT Suburb, Count(Id)
FROM Student
GROUP BY Suburb
```

Điều kiện hiển thị các bản ghi kết quả

- Lựa chọn các bản ghi kết quả để hiển thị

  SELECT <DS cột>
  FROM <DS bảng>
  [**WHERE** <Điều kiện tìm kiếm>]
  GROUP BY <Ds cột> **HAVING** <Điều kiện>

- Ví dụ: đưa ra tên các thành phố có nhiều hơn 3 sinh viên

  SELECT Suburb, COUNT(ID)
  FROM Student
  GROUP BY Suburb
  HAVING COUNT(ID) > 3

Các phép toán tập hợp: UNION, MINUS, INTERSECT

- Ví dụ: đưa ra danh sách tên các môn học không có sinh viên nào tham dự

```
SELECT DISTINCT  Subject.Name
FROM  Subject
MINUS
SELECT DISTINCT  Subject.Name
FROM   Student, Takes, Subject
WHERE  Student.Id = Takes.SID and  Takes.SNO = Subject.No
```

- Tìm sid của hãng cung ứng đồng thời 2 mặt hàng P1 và P2

```
SELECT sid FROM SupplyProduct WHERE pid = 'P1'
INTERSECT
SELECT sid FROM SupplyProduct WHERE pid = 'P2'
```

- Tìm mã số của hãng không cung ứng mặt hàng nào

```
SELECT sid FROM Supplier
MINUS
SELECT sid FROM SupplyProduct
```

Các câu truy vấn lồng nhau

- Là trường hợp các câu truy vấn (con) được viết lồng nhau
- Thường được sử dụng để
  - Kiểm tra thành viên tập hợp (**IN, NOT IN**)
  - So sánh tập hợp (**>ALL, >=ALL, <ALL,<=ALL,=ALL, NOT IN, SOME,**)
    - vd: SELECT *
      FROM Supplier
      WHERE SIZE>=ALL(SELECT SIZE FROM Supplier);
  - Kiểm tra các bảng rỗng (**EXISTS** hoặc **NOT EXISTS**)
- Các truy vấn con lồng nhau thông qua mệnh đề **WHERE**

- Kiểm tra thành viên tập hợp với IN và NOT IN:
  - Đưa ra mã số của các hãng cung ứng đồng thời 2 mặt hàng P1 và P2:

```
SELECT DISTINCT sid FROM SupplyProduct
WHERE pid = 'P1' AND sid IN (SELECT sid FROM
  SupplyProduct SP2 WHERE SP2.pid = 'P2');
```

  - Đưa ra sid của các hãng không cung ứng mặt hàng P3:

```
SELECT sid FROM SupplyProduct
WHERE sid NOT IN (SELECT sid From SupplyProduct SP2
  WHERE SP2.pid = 'P3');
```

- So sánh tập hợp: Sử dụng các phép toán <,>, >=,<=,=,!= (<>) kèm với các mệnh đề ANY và ALL
  - Đưa ra tên của các hãng có số nhân viên đông nhất:

```
SELECT sname FROM Supplier
WHERE size >= ALL(SELECT size FROM Supplier)
```

  - Đưa ra sid của hãng cung ứng một mặt hàng với số lượng bằng ít nhất 1 trong số lượng các mặt hàng được cung ứng bởi S2

```
SELECT sid FROM SupplyProduct
WHERE sid != 'S2' AND quantity = ANY(SELECT quantity
    FROM SupplyProduct SP2 WHERE SP2.sid = 'S2');
```

- Kiểm tra tập hợp rỗng với EXISTS và NOT EXISTS
  - EXISTS(câu truy vấn con): nhận giá trị đúng khi câu truy vấn con cho ra kết quả là một quan hệ khác rỗng
  - NOT EXISTS(câu truy vấn con): nhận giá trị đúng khi câu truy vấn con cho ra kết quả là một quan hệ rỗng

- Đưa ra thông tin của các nhà cung cấp đã cung ứng ít nhất một mặt hàng

```
SELECT * FROM Supplier S
WHERE EXISTS (SELECT sid FROM SupplyProduct SP
    WHERE S.sid = SP.sid);
```

- Đưa ra thông tin của các nhà cung cấp không cung ứng mặt hàng nào

```
SELECT * FROM Supplier S
WHERE NOT EXISTS (SELECT * FROM SupplyProduct
    SP WHERE S.sid = SP.sid);
```

Các hàm thư viện

- Hàm tính toán trên nhóm các bản ghi
  - MAX/MIN
  - SUM
  - AVG
  - COUNT
- Hàm tính toán trên bản ghi
  - Hàm toán học: ABS, SQRT, LOG, EXP, SIGN, ROUND
  - Hàm xử lý xâu ký tự: LEN, LEFT, RIGHT, MID
  - Hàm xử lý thời gian: DATE, DAY, MONTH, YEAR, HOUR, MINUTE, SECOND
  - Hàm chuyển đổi kiểu giá trị: FORMAT

SQL Server

```
SELECT * FROM GiangVien
WHERE DATEPART(year, GETDATE()) - DATEPART(year, NgaySinh) > 40
```

PostGreSQL

```
SELECT * FROM "GiangVien"
WHERE date_part('year', current_date) - date_part('year', "NgaySinh") > 40
```

Một số ví dụ với các hàm thư viện

- Có bao nhiêu mặt hàng khác nhau được cung ứng

```
SELECT COUNT(DISTINCT pid)
FROM SupplyProduct;
```

- Có tổng cộng bao nhiêu nhân viên làm cho các hãng ở Paris

```
SELECT SUM(size) FROM Supplier
WHERE city = 'Paris';
```

- Đưa ra số lượng mặt hàng trung bình mà hãng S1 cung ứng

```
SELECT AVG(quantity)
FROM SupplyProduct
WHERE sid = 'S1';
```

Một số truy vấn phức tạp

- Đưa ra tên của hãng S1 và tổng số lượng các mặt hàng mà hãng đó cung ứng

```
SELECT sname, SUM(quantity)
FROM Supplier S, SupplyProduct SP
WHERE S.sid = SP.sid AND S.sid = 'S1'
GROUP BY sname;
```

- Đưa ra mã số các hãng cung ứng và số lượng trung bình các mặt hàng được cung ứng bởi từng hãng

```
SELECT sid, AVG(quantity) FROM SupplyProduct
GROUP BY sid;
```

- Đưa ra mã số các hãng cung ứng mà số lượng mặt hàng trung bình được cung cấp bởi hãng đó là trong khoảng từ 75 đến 100

```
SELECT sid, AVG(quantity) FROM SupplyProduct
GROUP BY sid HAVING AVG(quantity) BETWEEN 75 AND 100
```

Thêm bản ghi vào bảng

- **INSERT INTO** table[(col1,col2,…)]
  VALUES (exp1,exp2,…)
- **INSERT INTO** table[(col1,col2,…)]
  SELECT col1,col2, …
  FROM tab1, tab2, …
  WHERE <dieu_kien>

– Ví dụ

- **INSERT INTO** Student **VALUES** ('1179','Jane','California');
- **INSERT INTO** Student(Id, Name, Suburb)
  VALUES ('1180','David','NewYork');
- **INSERT INTO** Student(Name, Id, Suburb)
  VALUES ('Mary','1181','Texas');
- **INSERT INTO** Student(Id, Name, Suburb)
  VALUES ('1182','John','Ohio'), ('1183','Tom','Georgia'), ('1184','Declan','Arizona');

Xóa bản ghi trong bảng

DELETE FROM <Tên bảng>
WHERE <Điều kiện xóa>;

- Ví dụ:

DELETE FROM SupplyProduct
WHERE sid = 'S4';
DELETE FROM Student
WHERE Suburb = 'Indiana';

Sửa dữ liệu trong bảng

```
UPDATE <tên bảng>
SET (<Tên cột> = Giá trị mới , …)
[WHERE <Điều kiện sửa đổi>];
```

- Ví dụ:
  - Hãng S1 chuyển tới Milan

```
UPDATE Supplier
SET city = 'Milan'
WHERE sid = 'S1';
```

  - Tất cả các mặt hàng được cung cấp với số lượng nhỏ hơn 100 đều tăng số lượng lên 1.5 lần

```
UPDATE SupplyProduct
SET quantity = quantity * 1.5
WHERE quantity < 100;
```