



HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG



BÀI GIẢNG MÔN
CƠ SỞ DỮ LIỆU

Giảng viên:

TS. Nguyễn Thị Thu Hiền

Điện thoại:

Email:

csdl.nth@gmail.com

Classcode:

1. Phân loại các ngôn ngữ truy vấn

❖ Đặt vấn đề

Cho 1 quan hệ:

Student

StudentID	Name	Address
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

- Câu hỏi: Tìm tên của các sinh viên nào sống ở Bundoora
 - Tìm các bộ của bảng Student có Address = Bundoora
 - Đưa ra các giá trị của thuộc tính Name của các bộ này

1. Phân loại các ngôn ngữ truy vấn

❖ Đặt vấn đề

Cho các quan hệ:

Course

CourseID	Name	Faculty
113	BCS	CSCE
101	MCS	CSCE

Student

StudentID	Name	Address
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Enrol

StudentID	CourseID
3936	101
1108	113
8507	101

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

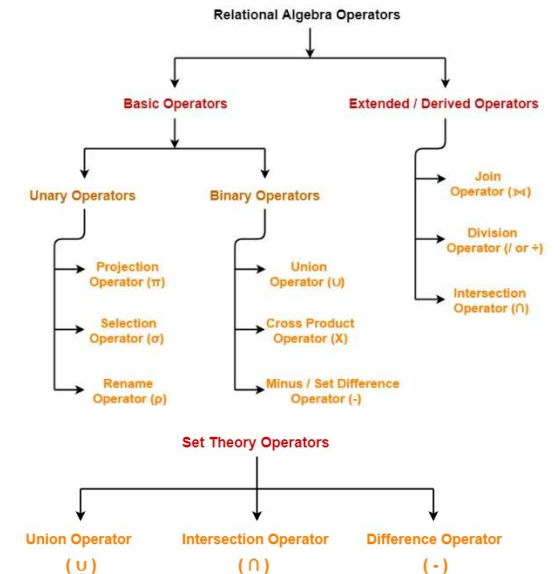
1. Phân loại các ngôn ngữ truy vấn

- 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

2. 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)

2. Ngôn ngữ đại số quan hệ

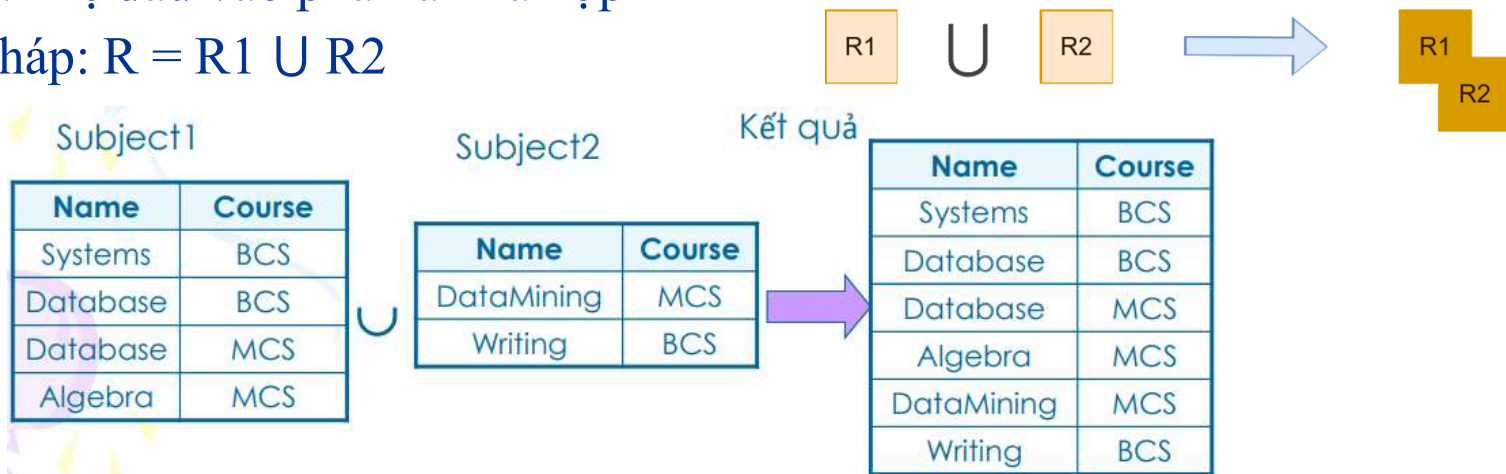
Phép toán tập hợp

● Định nghĩa: Quan hệ khả hợp

- 2 quan hệ r và s được gọi là khả hợp nếu chúng được xác định trên cùng 1 miền giá trị
- r xác định trên $D_1 \times D_2 \times \dots \times D_n$
- s xác định trên $D'_1 \times D'_2 \times \dots \times D'_m \Rightarrow D_i = D'_i$ và $n=m$

Phép hợp

- Định nghĩa: gồm các bộ thuộc ít nhất 1 trong 2 quan hệ đầu vào
- 2 quan hệ đầu vào phải là khả hợp
- Cú pháp: $R = R1 \cup R2$



2. Ngôn ngữ đại số quan hệ

Phép toán tập hợp

Phép giao

- Định nghĩa: gồm các bộ thuộc cả hai quan hệ đầu vào
- Cú pháp: $R = R1 \cap R2$



Subject1		Subject2		Kết quả	
Name	Course	Name	Course	Name	Course
Systems	BCS	DataMining	MCS	Systems	BCS
Database	BCS	Database	MCS	Database	MCS
Database	MCS	Systems	BCS		
Algebra	MCS	Writing	BCS		

2. Ngôn ngữ đại số quan hệ

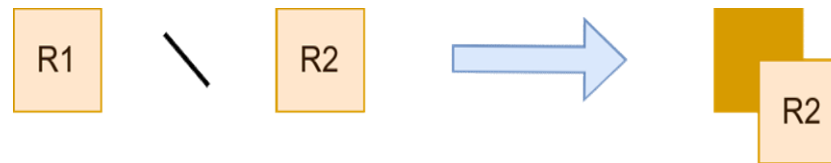
Phép toán tập hợp

Phép trừ

● Định nghĩa: gồm các bộ thuộc quan hệ thứ nhất nhưng không thuộc quan hệ thứ hai

● 2 quan hệ phải là khả hợp

● Cú pháp: $R = R1 \setminus R2$ hoặc $R = R1 - R2$



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

2. Ngôn ngữ đại số quan hệ

Phép toán tập hợp

Phép tích Đề-các

- Định nghĩa: là kết nối giữa từng bộ của quan hệ thứ nhất với mỗi bộ của quan hệ thứ hai
- Cú pháp: $Q = R \times S$

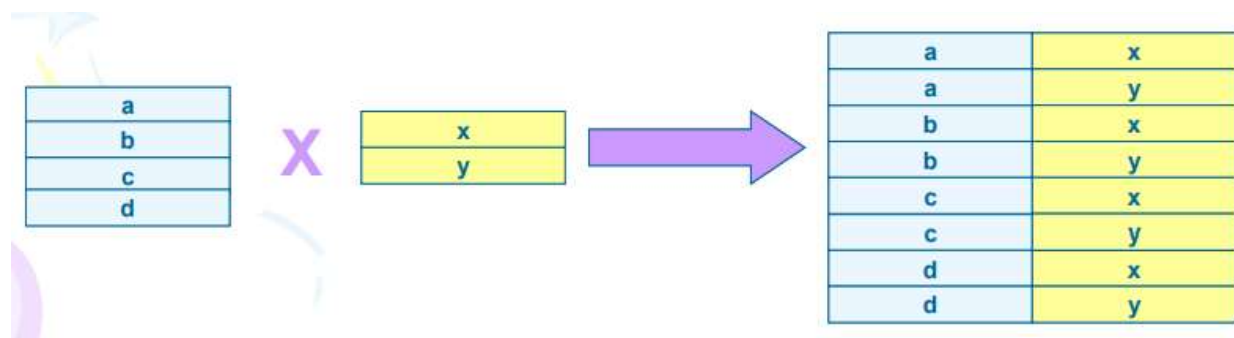
Tích Descartes của n tập hợp:

$$A_1 \times A_2 \times \dots \times A_n =$$

$$\{(a_1, a_2, \dots, a_n) \mid a_1 \in A_1, a_2 \in A_2, \dots, a_n \in A_n\}$$

$R \times S = \{ t \mid t \text{ có dạng } (a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_m) \text{ trong đó } (a_1, a_2, \dots, a_n) \in R \text{ và } (b_1, b_2, \dots, b_m) \in S \}$

- Dùng để kết hợp các bộ của các quan hệ lại với nhau
- Kết quả trả về là 1 quan hệ Q trong đó:
 - Mỗi bộ của Q là tổ hợp giữa 1 bộ trong R và 1 bộ trong S
 - Nếu R có u bộ và S có v bộ thì Q sẽ có “ uxv ” bộ
 - Nếu R có n thuộc tính và S có m thuộc tính thì Q sẽ có “ $n+m$ ” thuộc tính
($R^+ \cap Q^+ \neq \emptyset$)

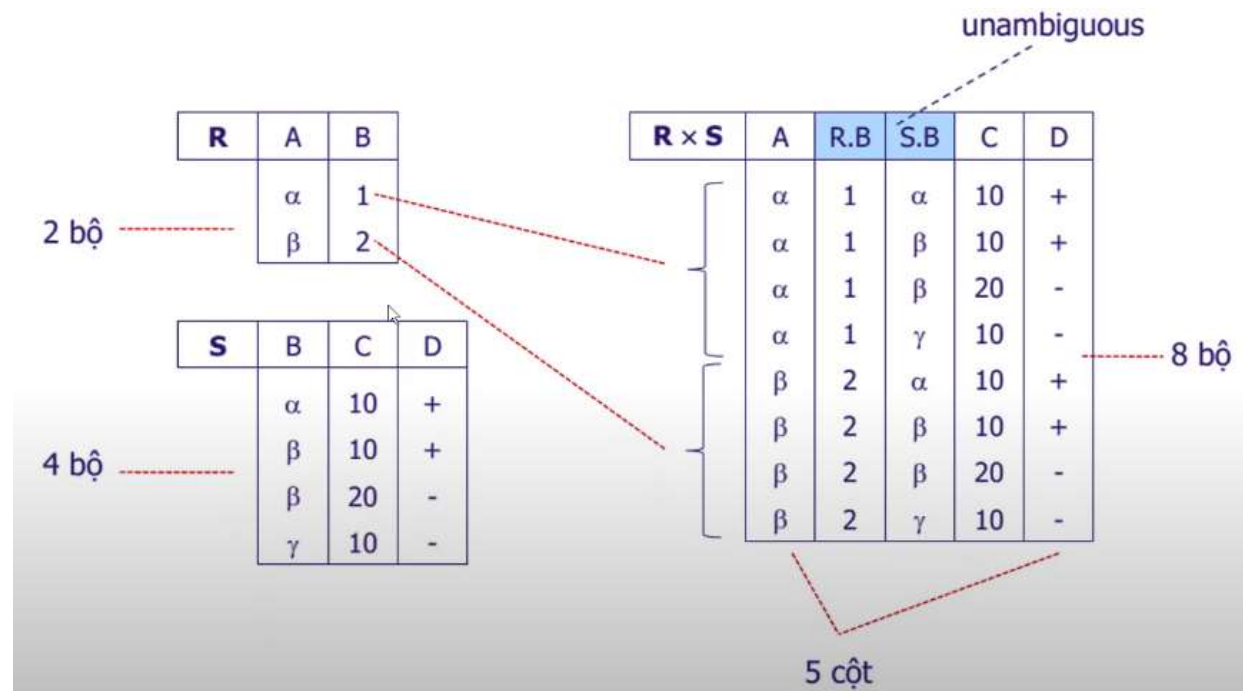


2. Ngôn ngữ đại số quan hệ

Phép toán tập hợp

Phép tích Đề-các

- Kết quả trả về là 1 quan hệ Q trong đó:
 - Mỗi bộ của Q là tổ hợp giữa 1 bộ trong R và 1 bộ trong S
 - Nếu R có u bộ và S có v bộ thì Q sẽ có “uxv” bộ
 - Nếu R có n thuộc tính và S có m thuộc tính thì Q sẽ có “n+m” thuộc tính
 $(R^+ \cap Q^+ \neq \emptyset)$

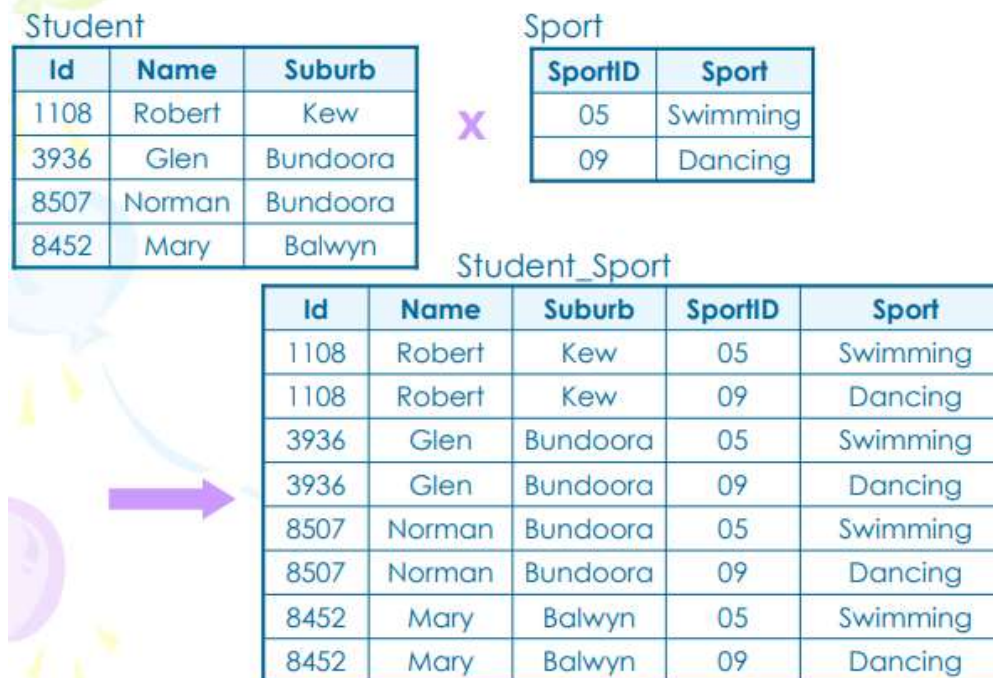


2. Ngôn ngữ đại số quan hệ

Phép toán tập hợp

Phép tích Đề-các

- Kết quả trả về là 1 quan hệ Q trong đó:
 - Mỗi bộ của Q là tổ hợp giữa 1 bộ trong R và 1 bộ trong S
 - Nếu R có u bộ và S có v bộ thì Q sẽ có “uxv” bộ
 - Nếu R có n thuộc tính và S có m thuộc tính thì Q sẽ có “n+m” thuộc tính
 $(R^+ \cap Q^+ \neq \emptyset)$



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

SportID	Sport
05	Swimming
09	Dancing

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

2. Ngôn ngữ đại số quan hệ

Phép toán tập hợp

Phép tích Đề-các

- Thông thường theo sau phép tích Đề-các là phép chọn nhằm lọc thông tin

$R \times S$

A	R.B	S.B	C	D
α	1	α	10	+
α	1	β	10	+
α	1	β	20	-
α	1	γ	10	-
β	2	α	10	+
β	2	β	10	+
β	2	β	20	-
β	2	γ	10	-

$\sigma_{A=S.B}(R \times S)$

A	R.B	S.B	C	D
α	1	α	10	+
β	2	β	10	+
β	2	β	20	-

2. Ngôn ngữ đại số quan hệ

Phép toán tập hợp Phép tích Đề-các

- **Ví dụ:** Với mỗi phòng ban, cho biết thông tin của người trưởng phòng
 - Quan hệ: PHONG_BAN, NHAN_VIEN
 - Thuộc tính: TRPHG, MAPHG, TENNV, HONV, ...

TENPHG	MAPHG	TRPHG	NG_NHANCHUC				
Nghien cuu	5	333445555	05/22/1988				
Dieu hanh	4	987987987	01/01/1995				
Quan ly	1	888665555	06/19/1981				

MANV	TENNV	HONV	NGSINH	DCHI	PHAI	LUONG	PHG
333445555	Tung	Nguyen	12/08/1955	638 NVC Q5	Nam	40000	5
999887777	Hang	Bui	07/19/1968	332 NTH Q1	Nu	25000	4
987654321	Nhu	Le	06/20/1951	291 HVH QPN	Nu	43000	4
987987987	Hung	Nguyen	09/15/1962	Ba Ria VT	Nam	38000	5

2. Ngôn ngữ đại số quan hệ

Phép toán tập hợp Phép tích Đề-các

- **Ví dụ:** Với mỗi phòng ban, cho biết thông tin của người trưởng phòng
 - Quan hệ: PHONG_BAN, NHAN_VIEN
 - Thuộc tính: TRPHG, MAPHG, TENNV, HONV, ...

TENPHG	MAPHG	TRPHG	NG_NHANCHUC
Nghien cuu	5	333445555	05/22/1988
Dieu hanh	4	987987987	01/01/1995
Quan ly	1	888665555	06/19/1981

MANV	TENNV	HONV	NGSINH	DCHI	PHAI	LUONG	PHG
333445555	Tung	Nguyen	12/08/1955	638 NVC Q5	Nam	40000	5
999887777	Hang	Bui	07/19/1968	332 NTH Q1	Nu	25000	4
987654321	Nhu	Le	06/20/1951	291 HVH QPN	Nu	43000	4
987987987	Hung	Nguyen	09/15/1962	Ba Ria VT	Nam	38000	5

TENPHG	MAPHG	TRPHG	NG_NHANCHUC	MANV	TENNV	HONV	...
Nghien cuu	5	333445555	05/22/1988	333445555	Tung	Nguyen	...
Dieu hanh	4	987987987	01/01/1995	987987987	Hung	Nguyen	...
Quan ly	1	888665555	06/19/1981	888665555	Vinh	Pham	...

2. Ngôn ngữ đại số quan hệ

Phép toán tập hợp Phép tích Đề-các

- **Ví dụ:** Với mỗi phòng ban, cho biết thông tin của người trưởng phòng
 - Quan hệ: PHONG_BAN, NHAN_VIEN
 - Thuộc tính: TRPHG, MAPHG, TENNV, HONV, ...

TENPHG	MAPHG	TRPHG	NG_NHANCHUC
Nghien cuu	5	333445555	05/22/1988
Dieu hanh	4	987987987	01/01/1995
Quan ly	1	888665555	06/19/1981

MANV	TENNV	HONV	NGSINH	DCHI	PHAI	LUONG	PHG
333445555	Tung	Nguyen	12/08/1955	638 NVC Q5	Nam	40000	5
999887777	Hang	Bui	07/19/1968	332 NTH Q1	Nu	25000	4
987654321	Nhu	Le	06/20/1951	291 HVH QPN	Nu	43000	4
987987987	Hung	Nguyen	09/15/1962	Ba Ria VT	Nam	38000	5

- B1: Tích Cartesian PHONG_BAN và NHAN_VIEN

$$PB_NV \leftarrow (NHAN_VIEN \times PHONG_BAN)$$

- B2: Chọn ra những bộ thỏa TRPHG=MANV

$$KQ \leftarrow \sigma_{TRPHG=MANV}(PB_NV)$$

2. Ngôn ngữ đại số quan hệ

Phép toán tập hợp
Phép tích Đề-các

- Ví dụ:** Xét 2 quan hệ của 2 lược đồ quan hệ NV1(Q_1) và KHOA(Q_2)

Q_1	MA_NV	TEN_NV	MA_PHG
	001	A	1
	002	B	1
	003	C	2

Q_2	MA_KH	TEN_KH
	KH01	X
	KH02	Y

- $Q = Q_1 \times Q_2$?

2. Ngôn ngữ đại số quan hệ

Phép toán tập hợp
Phép tích Đề-các

- Ví dụ:** Xét 2 quan hệ của 2 lược đồ quan hệ NV1(Q_1) và KHOA(Q_2)

Q_1	MA_NV	TEN_NV	MA_PHG
	001	A	1
	002	B	1
	003	C	2

Q_2	MA_KH	TEN_KH
	KH01	X
	KH02	Y

- $Q = Q_1 \times Q_2$?

Q	MA_NV	TEN_NV	MA_PHG	MA_KH	TEN_KH
	001	A	1	KH01	X
	002	B	1	KH01	X
	003	C	2	KH01	X
	001	A	1	KH02	Y
	002	B	1	KH02	Y
	003	C	2	KH02	Y

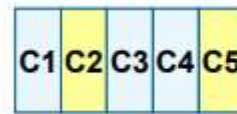
2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

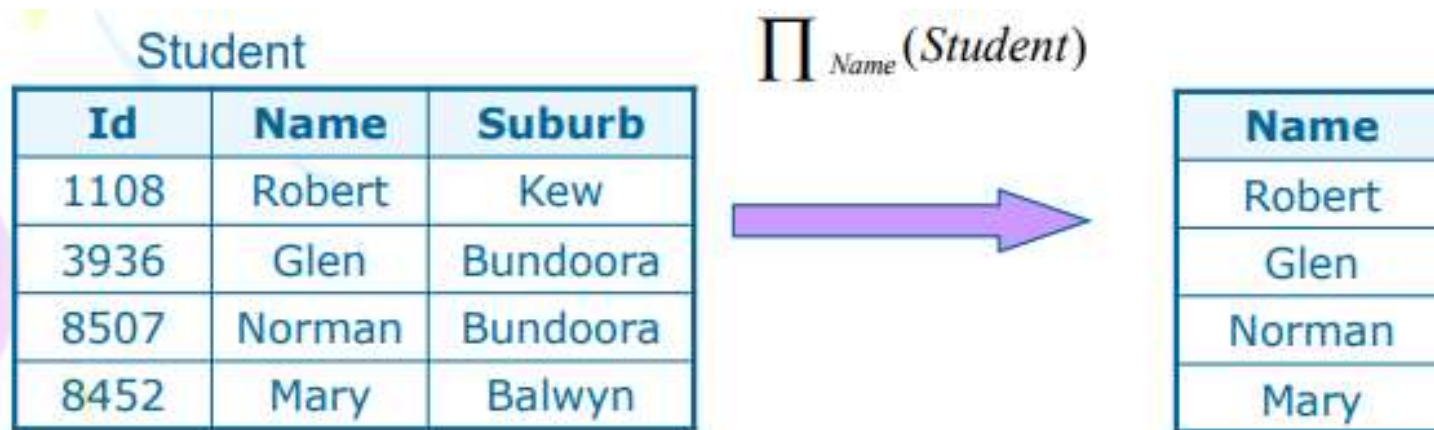
Phép chiếu

● Định nghĩa: Lựa chọn một số thuộc tính từ một quan hệ (chọn cột)

● Cú pháp: $\Pi_{A_1, A_2}(R)$



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



2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

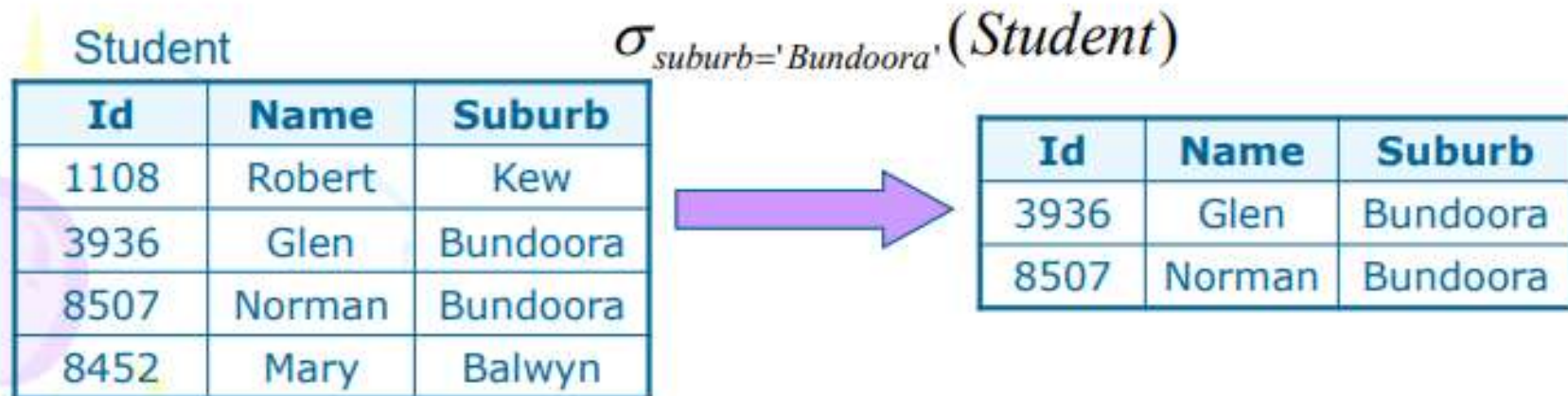
Phép chọn

- Định nghĩa: **Lựa chọn các bộ** trong một quan hệ thỏa mãn điều kiện cho trước (chọn hàng)

- Cú pháp: $\sigma_{\langle \text{Điều kiện} \rangle}(R)$



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



2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Phép chọn – Điều kiện?

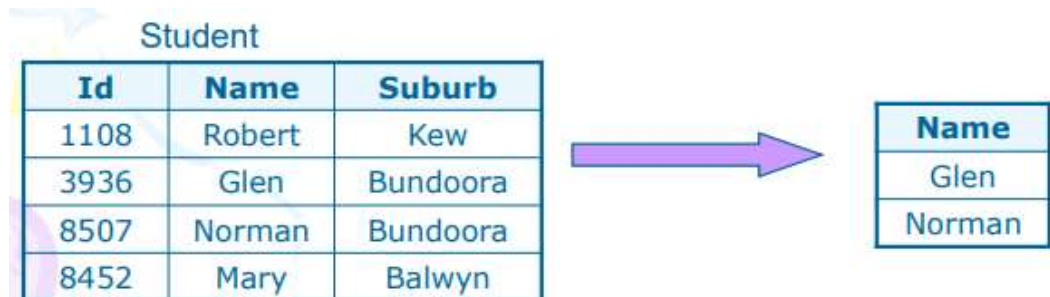
- Điều kiện chọn còn gọi là biểu thức chọn.
- Biểu thức chọn F: một tổ hợp logic của các toán hạng. Mỗi toán hạng là một phép so sánh đơn giản giữa 2 biến là hai thuộc tính hoặc giữa 1 biến là 1 thuộc tính và 1 giá trị hằng.

- Các phép so sánh trong F: $<$, $>$, $=$, \leq , \geq , \neq
- Các phép toán logic trong F: \vee , \wedge , \neg

Ví dụ: chọn và chiếu

- Đưa ra tên của các sinh viên sống ở Bundoora
- Cú pháp:

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



2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Phép kết nối 2 quan hệ

- Khái niệm ghép bộ: $u = (a_1, \dots, a_n)$; $v = (b_1, \dots, b_m)$
 $\rightarrow (u, v) = (a_1, \dots, a_n, b_1, \dots, b_m)$
- Phép kết nối 2 quan hệ thực chất là phép ghép các cặp bộ của 2 quan hệ thỏa mãn 1 điều kiện nào đó trên chúng.
- Biểu thức kết nối là phép hội của các toán hạng, mỗi toán hạng là 1 phép so sánh đơn giản giữa 1 thuộc tính của quan hệ r và 1 thuộc tính của quan hệ s .
- Cú pháp:

$R1 \bowtie_{\langle\langle \text{điều kiện} \rangle\rangle} R2$
- Có thể xem: Phép kết nối = Phép tích Đề-các + Chọn
- Gọi là phép kết nối theta

2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Phép kết nối 2 quan hệ (tiếp)

- Được dùng để tổ hợp 2 bộ có liên quan từ 2 quan hệ thành 1 bộ (thỏa điều kiện)
- Ký hiệu $R \bowtie S$
 - $R(A_1, A_2, \dots, A_n)$ và (B_1, B_2, \dots, B_m)
- Kết quả của phép kết là một quan hệ Q
 - Có $n + m$ thuộc tính $Q(A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_m)$
 - Mỗi bộ của Q là tổ hợp của 2 bộ trong R và S , thỏa mãn một số điều kiện kết nào đó (điều kiện: θ)
 - ✦ Có dạng $A_i \theta B_j$
 - ✦ A_i là thuộc tính của R , B_j là thuộc tính của S
 - ✦ A_i và B_j có cùng miền giá trị
 - ✦ θ là phép so sánh $\neq, =, <, >, \leq, \geq$

2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Phép kết nối 2 quan hệ (tiếp)

Ví dụ 1: Đưa ra *danh sách các sinh viên* và *mã khoá học* mà sinh viên đó tham gia:

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

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

Student

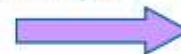
Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

\bowtie
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

2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Phép kết nối theta – Ví dụ 2

SINHVIEN (MASV, hoten, namsinh, diachi, #ML)

GIAOVIEN (MAGV, hotenGv, namsinhGv, diachiGv)

Kí hiệu: $\triangleright \triangleleft$ hoặc $\triangleright \triangleleft \theta$ hoặc $\triangleright \triangleleft$
 <điều kiện> <điều kiện> θ
 <điều kiện>

Tìm mã sinh viên và họ tên các sinh viên lớn tuổi hơn một giáo viên nào đó ?

SINHVIEN

MASV	hoten	namsinh	diachi	ML
b123	Nguyễn Tấn Tài	2000	An Giang	001
b789	Nguyễn A	1994	Hồng Ngự	002

GIAOVIEN

MAGV	hotenGv	namsinhGv	diachiGv
GV1	Nguyễn Văn B	1990	Cần Thơ
GV2	Nguyễn Thị C	1988	Vĩnh Long
GV3	Lê Thị D	1995	Cần Thơ

2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Phép kết nối theta – Ví dụ 2

SINHVIEN (MASV, hoten, namsinh, diachi, #ML)

GIAOVIEN (MAGV, hotenGv, namsinhGv, diachiGv)

Kí hiệu: $\triangleright \triangleleft$ hoặc $\triangleright \triangleleft \theta$ hoặc $\triangleright \triangleleft$
 <điều kiện> <điều kiện> θ
 <điều kiện>

Tìm mã sinh viên và họ tên các sinh viên lớn tuổi hơn một giáo viên nào đó ?

SINHVIEN

MASV	hoten	namsinh	diachi	ML
b123	Nguyễn Tấn Tài	2000	An Giang	001
b789	Nguyễn A	1994	Hồng Ngự	002

GIAOVIEN

MAGV	hotenGv	namsinhGv	diachiGv
GV1	Nguyễn Văn B	1990	Cần Thơ
GV2	Nguyễn Thị C	1988	Vĩnh Long
GV3	Lê Thị D	1995	Cần Thơ

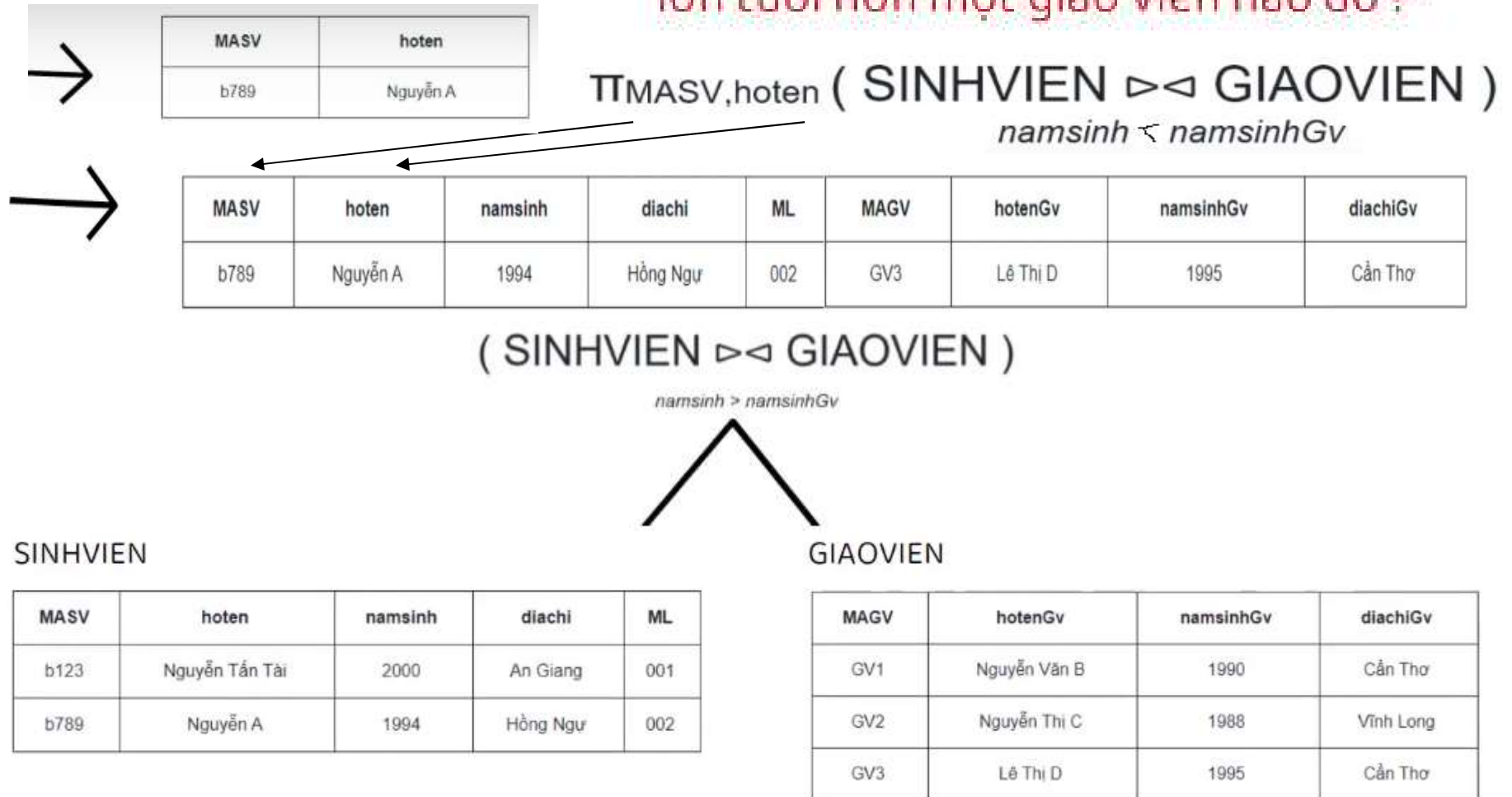
$\pi_{\text{MASV, hoten}} (\text{SINHVIEN} \triangleright \triangleleft \text{GIAOVIEN})$
namsinh > namsinhGv

2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Phép kết nối theta – Ví dụ 2

Tìm mã sinh viên và họ tên các sinh viên lớn tuổi hơn một giáo viên nào đó ?



2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Phép kết nối bằng/ Phép 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**

❖ Phép nối bằng (**Equi-join**)

○ Ký hiệu $R \bowtie_c S$

○ C là điều kiện so sánh bằng

● Định nghĩa: **Phép kết nối bằng trên các thuộc tính cùng tên của 2 quan hệ** và sau khi kết nối 1 thuộc tính trong 1 cặp thuộc tính trùng tên đó sẽ bị loại khỏi quan hệ kết quả thì phép kết nối gọi là **kết nối tự nhiên**

✱ Ký hiệu $R \bowtie S$ hay $R * S$

✱ $R^+ \cap Q^+ \neq \emptyset$ (phải có cột giống nhau)

✱ Kết quả của phép kết tự nhiên bỏ bớt đi 1 cột giống nhau

● Lưu ý: Phép nối tự nhiên là **phép nối bằng** mà các cặp thuộc tính dùng trong điều kiện **có cùng tên và cùng miền giá trị**

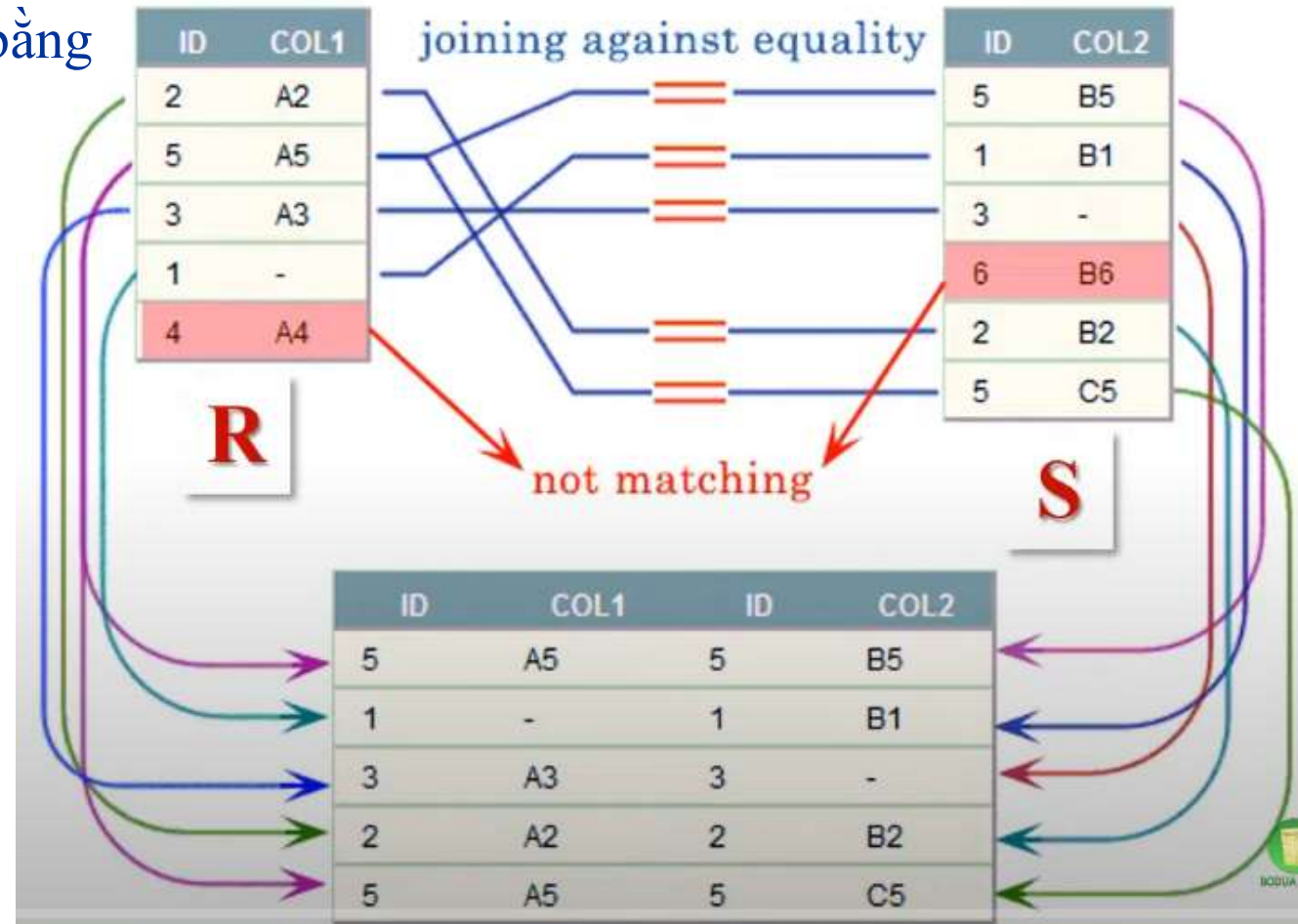
2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Phép kết nối bằng/ Phép kết nối tự nhiên

VD: Phép kết nối bằng

$R \bowtie_{r.ID=s.ID} S$



2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

VD: Phép kết nối TN

R

R ⋈ **S**

S

ITEM_ID	ITEM_NAME	ITEM_UNIT	COMPANY_ID
1	Chex Mix	Pcs	16
6	Cheez-It	Pcs	15
2	BN Biscuit	Pcs	15
3	Mighty Munch	Pcs	17
4	Pot Rice	Pcs	15
5	Jaffa Cakes	Pcs	18
7	Salt n Shake	Pcs	-

COMPANY_ID	COMPANY_NAME	COMPANY_CITY
18	Order All	Boston
15	Jack Hill Ltd	London
16	Akas Foods	Delhi
17	Foodies.	London
19	sip-n-Bite.	New York

** Same column came once

COMPANY_ID	ITEM_ID	ITEM_NAME	ITEM_UNIT	COMPANY_NAME	COMPANY_CITY
16	1	Chex Mix	Pcs	Akas Foods	Delhi
15	6	Cheez-It	Pcs	Jack Hill Ltd	London
15	2	BN Biscuit	Pcs	Jack Hill Ltd	London
17	3	Mighty Munch	Pcs	Foodies.	London
15	4	Pot Rice	Pcs	Jack Hill Ltd	London
18	5	Jaffa Cakes	Pcs	Order All	Boston

2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ Phân biệt 3 phép kết nối

Student				Enrol	
Id	Name	Suburb		SID	Course
1108	Robert	Kew	Id=SID	3936	101
3936	Glen	Bundoora		1108	113
8507	Norman	Bundoora		8507	101
8452	Mary	Balwyn			

Kết quả
Kết nối theta/Kết nối bằng

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

❖ Theta join:

Cho phép thực hiện các phép so sánh tùy ý (<,>,...)

❖ Equi-join:

Giống như theta join, nhưng chỉ so sánh bằng

❖ Natural join:

Giống như phép nối bằng, trên các thuộc tính cùng tên trong các quan hệ. Ngoài ra, nối tự nhiên còn loại bỏ các cột trùng lặp trong phép so sánh (*chỉ giữ lại 1 cột*)

Takes	
SID	SNO
1108	21
1108	23
8507	23
8507	29

*

Enrol	
SID	Course
3936	101
1108	113
8507	101

Kết nối tự nhiên

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

2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Ví dụ các phép kết nối:

- Hãy cho biết MASV và họ tên các sinh viên học môn TH104 ?



Ví dụ về chọn, chiếu, kết nối:

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

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

Tự tìm hiểu các phép kết nối mở rộng

Phép kết nối mở rộng:

- Bên trái -> ⋈
- Bên phải -> ⋈
- Cả 2 bên -> ⋈

2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Ví dụ các phép kết nối: Lời giải



- Hãy cho biết MASV và họ tên các sinh viên học môn TH104 ?

Cách 1:

$A \leftarrow \sigma_{MM='TH104'}(HOC) * SINHVIEN$

$B \leftarrow \pi_{MASV}(A)$

$C \leftarrow B * SINHVIEN$

$\pi_{MASV, hoten}(C)$

Cách 2:

$A \leftarrow \sigma_{MM='TH104'}(HOC * SINHVIEN)$

$\pi_{MASV, hoten}(A)$

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

$$\pi_{Name, Course} (\sigma_{Suburb='Bundoora'} (Student \bowtie_{Id=SID} Enrol))$$

2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Phép chia

● Định nghĩa: Phép chia giữa 1 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 sao cho khi ghép mọi bộ thuộc s với t thì ta đều có một bộ thuộc r

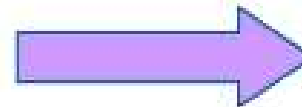
● Cú pháp: $R = r : s$ (hoặc r/s)

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

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

2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Phép chia

- Ví dụ: Hãy cho biết mã giáo viên dạy tất cả các môn học ?

DAY (#MAGV, #MAMH)

MONHOC (MAMH)

Quan hệ DAY

MAGV	MAMH
231	TH409
231	TH364
231	TH334
232	TH334
232	TH409
232	TH364
233	TH409
233	TH364

Quan hệ MONHOC

MAMH
TH409
TH334
TH364

MAGV không trùng nhau

A

MAGV
231
232
233

2. Ngôn ngữ đại số quan hệ

Phép toán quan hệ

Phép chia

- Ví dụ: Hãy cho biết mã giáo viên dạy tất cả các môn học ?

DAY (#MAGV, #MAMH)

MONHOC (MAMH)

DAY / MONHOC

Quan hệ DAY

MAGV	MAMH
231	TH409
231	TH364
231	TH334
232	TH334
232	TH409
232	TH364
233	TH409
233	TH364

Quan hệ MONHOC

MAMH
TH409
TH334
TH364

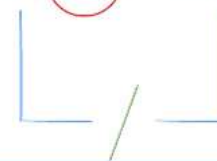
MAGV không trùng nhau

A

MAGV
231
232
233

Hãy cho biết mã giáo viên dạy tất cả các môn học ?

TIPS



2. Ngôn ngữ đại số quan hệ

■ Lời nhắc

- Hợp: $R \cup S : \in R / \in S / \in R \& S$
- Giao: $R \cap S : \in R \& S$
- Trừ: $R - S : \in R \& \text{không} \in S$
- Chọn: $\sigma_P(R) \rightarrow$ Chọn vài bộ thỏa đk P
- Chiếu: $\pi_{A_1, A_2, \dots, A_k}(R) \rightarrow$ Chọn vài cột
- Tích: $R \times S : \rightarrow u \times v$ bộ & $n+m$ thuộc tính
- Join: $R \bowtie_C S = X + \sigma \rightarrow n+m$ thuộc tính

Số thuộc tính
không đổi

• Phép hợp (Union)

$$r \cup s = \{t \mid t \in r \vee t \in s\}$$

Ví dụ:

r	(A	B	C)	s	(A	B	C)	$r \cup s = h$	(A	B	C)
a ₁	b ₁	c ₁	a ₁	b ₁	c ₁	a ₁	b ₁	c ₁	a ₁	b ₁	c ₁
a ₁	b ₁	c ₂	a ₁	b ₂	c ₁	a ₁	b ₁	c ₂	a ₁	b ₂	c ₁
a ₁	b ₂	c ₂	a ₁	b ₂	c ₂	a ₁	b ₂	c ₁	a ₁	b ₂	c ₂
a ₂	b ₂	c ₂				a ₁	b ₂	c ₂	a ₂	b ₂	c ₂
a ₃	b ₂	c ₂				a ₃	b ₂	c ₂	a ₃	b ₂	c ₂

• Phép giao (intersection)

$$r \cap s = \{t \mid t \in r \wedge t \in s\}$$

Ví dụ:

r	(A	B	C)	s	(A	B	C)
a ₁	b ₁	c ₁	a ₁	b ₁	c ₁		
a ₁	b ₁	c ₂	a ₁	b ₂	c ₁		
a ₁	b ₂	c ₂	a ₁	b ₂	c ₂		
a ₂	b ₂	c ₂					
a ₃	b ₂	c ₂					

$r \cap s = g$	(A	B	C)
a ₁	b ₁	c ₁	
a ₁	b ₂	c ₂	

2. Ngôn ngữ đại số quan hệ

■ Lời nhắc

- Hợp: $R \cup S : \in R / \in S / \in R \& S$
 - Giao: $R \cap S : \in R \& S$
 - Trừ: $R - S : \in R \& \text{không} \in S$
 - Chọn: $\sigma_P(R) \rightarrow$ Chọn vài bộ thỏa đk P
- Số thuộc tính không đổi
- Chiếu: $\pi_{A_1, A_2, \dots, A_k}(R) \rightarrow$ Chọn vài cột
 - Tích: $R \times S : \rightarrow u \times v \text{ bộ} \& n+m \text{ thuộc tính}$
 - Join: $R \bowtie_C S = X + \sigma \rightarrow n+m \text{ thuộc tính}$

• Phép trừ (minus)

$$r - s = \{ t \mid t \in r \wedge t \notin s \}$$

Ví dụ:

r	(A	B	C)
	a ₁	b ₁	c ₁
	a ₁	b ₁	c ₂
	a ₁	b ₂	c ₂
	a ₂	b ₂	c ₂
	a ₃	b ₂	c ₂

s	(A	B	C)
	a ₁	b ₁	c ₁
	a ₁	b ₂	c ₁
	a ₁	b ₂	c ₂

r - s = t	(A	B	C)
	a ₁	b ₁	c ₂
	a ₂	b ₂	c ₂
	a ₃	b ₂	c ₂

• Phép tích Đề - Các (Cartesian Product)

$$r \times s = \{ t \mid t = (a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_m) \wedge (a_1, a_2, \dots, a_n) \in r \wedge (b_1, b_2, \dots, b_m) \in s \}$$

Ví dụ:

r	(A	B	C)
	a ₁	b ₁	1
	a ₂	b ₂	2
	a ₃	b ₃	3

s	(D	E)
	1	e ₁
	2	e ₂
	3	e ₃

r x s = p	(A	B	C	D	E)
	a ₁	b ₁	1	1	e ₁
	a ₁	b ₁	1	2	e ₂
	a ₁	b ₁	1	3	e ₃
	a ₂	b ₂	2	1	e ₁
	a ₂	b ₂	2	2	e ₂
	a ₂	b ₂	2	3	e ₃
	a ₃	b ₃	3	1	e ₁
	a ₃	b ₃	3	2	e ₂
	a ₃	b ₃	3	3	e ₃

2. Ngôn ngữ đại số quan hệ

■ Lời nhắc

- Hợp: $R \cup S : \in R / \in S / \in R \& S$
 - Giao: $R \cap S : \in R \& S$
 - Trừ: $R - S : \in R \& \text{không} \in S$
 - Chọn: $\sigma_P(R) \rightarrow$ Chọn vài bộ thỏa đk P
- Số thuộc tính không đổi
- Chiếu: $\pi_{A_1, A_2, \dots, A_k}(R) \rightarrow$ Chọn vài cột
 - Tích: $R \times S : \rightarrow u \times v \text{ bộ} \& n+m \text{ thuộc tính}$
 - Join: $R \bowtie_C S = X + \sigma \rightarrow n+m \text{ thuộc tính}$

• Phép chiếu (Projection)

$$\Pi_X(r) = \{ t[X] \mid t \in r \}$$

Ví dụ:

$$X = \{ A, B \}; Y = \{ C \}$$

r	(A B C)	$\Pi_X(r) = s_1(A B)$	$\Pi_Y(r) = s_2(C)$
	a ₁ b ₁ c ₁	a ₁ b ₁	c ₁
	a ₁ b ₁ c ₂	a ₁ b ₂	c ₂
	a ₁ b ₂ c ₂	a ₂ b ₂	
	a ₂ b ₂ c ₂	a ₃ b ₂	
	a ₃ b ₂ c ₂		

• Phép chọn (Selection)

$$\sigma_F(r) = \{ t \mid t \in r \wedge F(t) = \text{đúng} \}$$

Ví dụ:

r	(A B C)	$\sigma_{A=a_1}(r) = r_1$	$\sigma_{A=a_1 \wedge C=c_2}(r) = r_2$
	a ₁ b ₁ c ₁		
	a ₁ b ₁ c ₂	a ₁ b ₁ c ₂	a ₁ b ₁ c ₂
	a ₁ b ₂ c ₂	a ₁ b ₂ c ₂	a ₁ b ₂ c ₂
	a ₂ b ₂ c ₂		
	a ₃ b ₂ c ₂		

2. Ngôn ngữ đại số quan hệ

■ Lời nhắc

- Hợp: $R \cup S : \in R / \in S / \in R \& S$
 - Giao: $R \cap S : \in R \& S$
 - Trừ: $R - S : \in R \& \text{không} \in S$
 - Chọn: $\sigma_P(R) \rightarrow$ Chọn vài bộ thỏa đk P
- Số thuộc tính không đổi
- Chiếu: $\pi_{A_1, A_2, \dots, A_k}(R) \rightarrow$ Chọn vài cột
 - Tích: $R \times S : \rightarrow u \times v$ bộ & $n+m$ thuộc tính
 - Join: $R \bowtie_C S = X + \sigma \rightarrow n+m$ thuộc tính

• Phép kết nối (join)

$$r \bowtie_F s = \{ t \mid t = (u, v) \wedge u \in r \wedge v \in s \wedge F(t) = \text{đúng} \}$$

Ví dụ:

$$F = (C \leq D); F' = (C = D)$$

r	(A	B	C)	s	(D	E)
	a ₁	b ₁	1		1	e ₁
	a ₂	b ₂	2		2	e ₂
	a ₃	b ₃	3		3	e ₃

$$r \bowtie_F s = k$$

(A	B	C	D	E)
a ₁	b ₁	1	1	e ₁
a ₁	b ₁	1	2	e ₂
a ₁	b ₁	1	3	e ₃
a ₂	b ₂	2	2	e ₂
a ₂	b ₂	2	3	e ₃
a ₃	b ₃	3	3	e ₃

$$r \bowtie_{F'} s = k'$$

(A	B	C	D	E)
a ₁	b ₁	1	1	e ₁
a ₂	b ₂	2	2	e ₂
a ₃	b ₃	3	3	e ₃

• Kết nối tự nhiên (natural join)

$$r(U) * s(V) = \{ t[U \cup V] \mid t[U] \in r \wedge t[V] \in s \}$$

p	(A	B	C)	q	(C	D)	p*q = z	(A	B	C	D)
a ₁	b ₁	1		1	d ₁		a ₁	b ₁	1	d ₁	
a ₂	b ₁	1		1	d ₂		a ₁	b ₁	1	d ₂	
a ₃	b ₂	2		2	d ₂		a ₂	b ₁	1	d ₁	
							a ₂	b ₁	1	d ₂	
							a ₃	b ₂	2	d ₂	

2. Ngôn ngữ đại số quan hệ

■ Lời nhắc

- Hợp: $R \cup S : \in R / \in S / \in R \& S$
 - Giao: $R \cap S : \in R \& S$
 - Trừ: $R - S : \in R \& \text{không} \in S$
 - Chọn: $\sigma_P(R) \rightarrow$ Chọn vài bộ thỏa đk P
 - Chiếu: $\pi_{A_1, A_2, \dots, A_k}(R) \rightarrow$ Chọn vài cột
 - Tích: $R \times S : \rightarrow u \times v$ bộ & $n+m$ thuộc tính
 - Join: $R \bowtie_C S = X + \sigma \rightarrow n+m$ thuộc tính
- Số thuộc tính không đổi

• Phép chia (Division)

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

Ví dụ:

p	(A	B	C	D	E)	s	(D	E)
	a ₁	b ₁	1	1	e ₁		1	e ₁
	a ₁	b ₁	1	2	e ₂		2	e ₂
	a ₁	b ₁	1	3	e ₃		3	e ₃
	a ₂	b ₂	2	1	e ₁			
	a ₂	b ₂	2	2	e ₂			
	a ₂	b ₂	2	3	e ₃			
	a ₃	b ₃	3	1	e ₁			
	a ₃	b ₃	3	2	e ₂			
	a ₃	b ₃	3	3	e ₃			

$p \div s = q$	(A	B	C)
	a ₁	b ₁	1
	a ₂	b ₂	2
	a ₃	b ₃	3

2. Ngôn ngữ đại số quan hệ

■ Lời nhắc

Symbol (Name)	Example of Use
σ (Selection)	$\sigma_{\text{salary} \geq 85000}(\text{instructor})$ Trả về các hàng của mỗi quan hệ đầu vào thỏa mãn điều kiện tiên quyết
Π (Projection)	$\Pi_{ID, salary}(\text{instructor})$ Xuất các thuộc tính được chỉ định từ tất cả các hàng của quan hệ đầu vào. Xóa các bộ trùng lặp khỏi đầu ra.
\bowtie (Natural join)	$\text{instructor} \bowtie \text{department}$ Xuất ra các cặp hàng từ hai quan hệ đầu vào có cùng giá trị trên tất cả các thuộc tính có cùng tên.
\times (Cartesian product)	$\text{instructor} \times \text{department}$ Xuất ra tất cả các cặp hàng từ hai quan hệ đầu vào (bất kể chúng có cùng giá trị trên các thuộc tính chung hay không)
\cup (Union)	$\Pi_{name}(\text{instructor}) \cup \Pi_{name}(\text{student})$ Xuất ra hợp của các bộ từ hai quan hệ đầu vào.

2. Ngôn ngữ đại số quan hệ

❖ Lưu ý:

SINHVIEN (MASV, hoten, namsinh, diachi, #ML)

GIAOVIEN (MAGV, hotenGv, namsinhGv, diachiGv)

HOC (#MASV, #MM)

1. Liệt kê danh sách tên sinh viên và tên giáo viên ?
2. Tìm mã số sinh viên vừa học môn TH409 vừa học môn TH104 ?
3. Tìm mã số sinh viên của sinh viên học môn TH409 và không học môn TH104 ?

2. Ngôn ngữ đại số quan hệ

❖ Lưu ý

SINHVIEN (MASV, hoten, namsinh, diachi, #ML)

GIAOVIEN (MAGV, hotenGv, namsinhGv, diachiGv)

HOC (#MASV, #MM)

1. Liệt kê danh sách tên sinh viên và tên giáo viên ?

A

B

$\pi_{\text{namsinh}}(\text{SINHVIEN}) \cup \pi_{\text{namsinhGv}}(\text{GIAOVIEN})$

2. Tìm mã số sinh viên vừa học môn TH409 vừa học môn TH104 ?

$\pi_{\text{MASV}}(\sigma_{\text{MM}=\text{'TH409'}}(\text{HOC})) \cap \pi_{\text{MASV}}(\sigma_{\text{MM}=\text{'TH104'}}(\text{HOC}))$

3. Tìm mã số sinh viên của sinh viên học môn TH409 và không học môn TH104 ?

$\pi_{\text{MASV}}(\sigma_{\text{MM}=\text{'TH409'}}(\text{HOC})) - \pi_{\text{MASV}}(\sigma_{\text{MM}=\text{'TH104'}}(\text{HOC}))$

2. Ngôn ngữ đại số quan hệ

❖ Lưu ý:

SINHVIEN (MASV, hoten, namsinh, diachi, #ML)

GIAOVIEN (MAGV, hotenGv, namsinhGv, diachiGv)

HOC (#MASV, #MM)

1. Liệt kê danh sách tên sinh viên và tên giáo viên ?
2. Tìm mã số sinh viên vừa học môn TH409 vừa học môn TH104 ?
3. Tìm mã số sinh viên của sinh viên học môn TH409 và không học môn TH104 ?

Tóm tắt

Chỗ nối tiếp giữa 2 vế:

- Là chữ "và", thì sẽ dùng phép hợp.
- Là chữ "vừa", thì sẽ dùng phép giao.
- Là kiểu "có cái này nhưng không có cái kia", thì sẽ dùng phép trừ.

2. Ngôn ngữ đại số quan hệ

Các phép toán tổng hợp và nhóm (thống kê số liệu)

- ❖ Ngoài việc chỉ cần truy xuất một số bộ và thuộc tính nhất định của một hoặc nhiều quan hệ, chúng ta thường muốn thực hiện một số dạng tổng hợp dữ liệu hoặc một số dạng nhóm dữ liệu.
- ❖ Các phép toán này không thể được thực hiện bằng các phép toán đại số quan hệ cơ bản được xem xét ở trên. Tuy nhiên, các phép toán bổ sung đã được đề xuất.

2. Ngôn ngữ đại số quan hệ

Ví dụ: Đối với mỗi biểu thức, hãy giải thích bằng lời biểu thức đó thực hiện chức năng gì.

- a. $\sigma_{year \geq 2009}(takes) \bowtie student$
- b. $\sigma_{year \geq 2009}(takes \bowtie student)$
- c. $\Pi_{ID, name, course_id}(student \bowtie takes)$

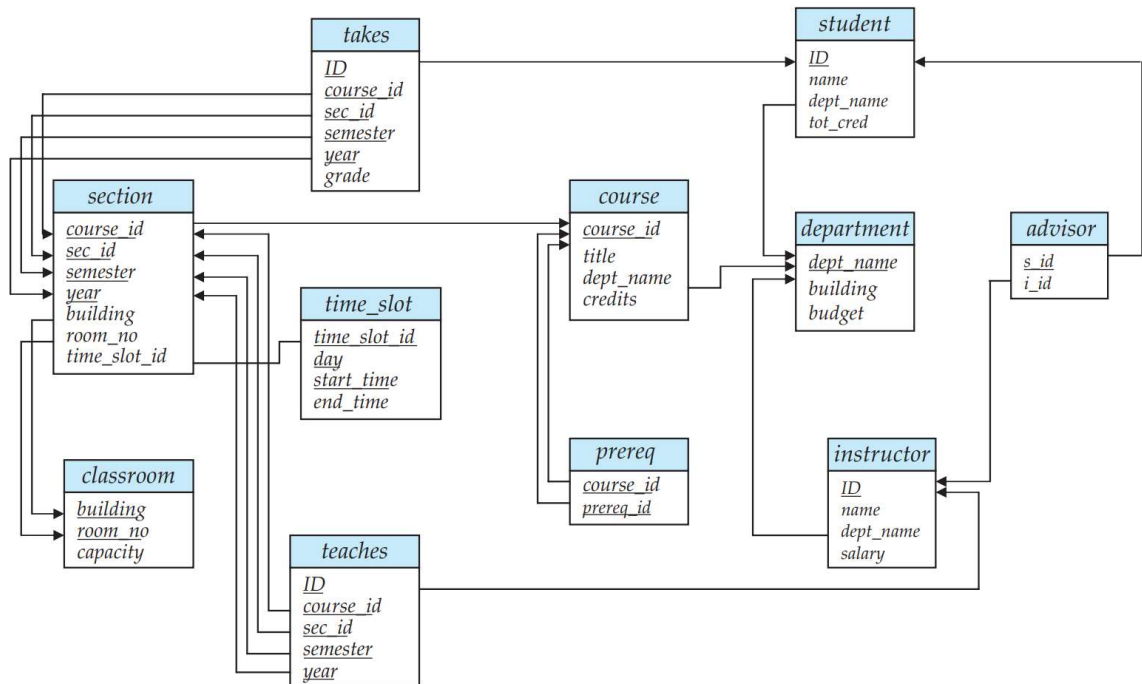
2. Ngôn ngữ đại số quan hệ

Ví dụ: Đối với mỗi biểu thức, hãy giải thích bằng lời biểu thức đó thực hiện chức năng gì.

- $\sigma_{year \geq 2009}(takes) \bowtie student$
- $\sigma_{year \geq 2009}(takes \bowtie student)$
- $\Pi_{ID, name, course_id}(student \bowtie takes)$

- Đối với mỗi sinh viên học ít nhất một khóa học trong năm 2009, hãy hiển thị thông tin của sinh viên cùng với thông tin về các khóa học mà sinh viên đã học. Các thuộc tính trong kết quả là:

$ID, name, dept\ name, tot_cred, course_id, section_id, semester, year, grade$



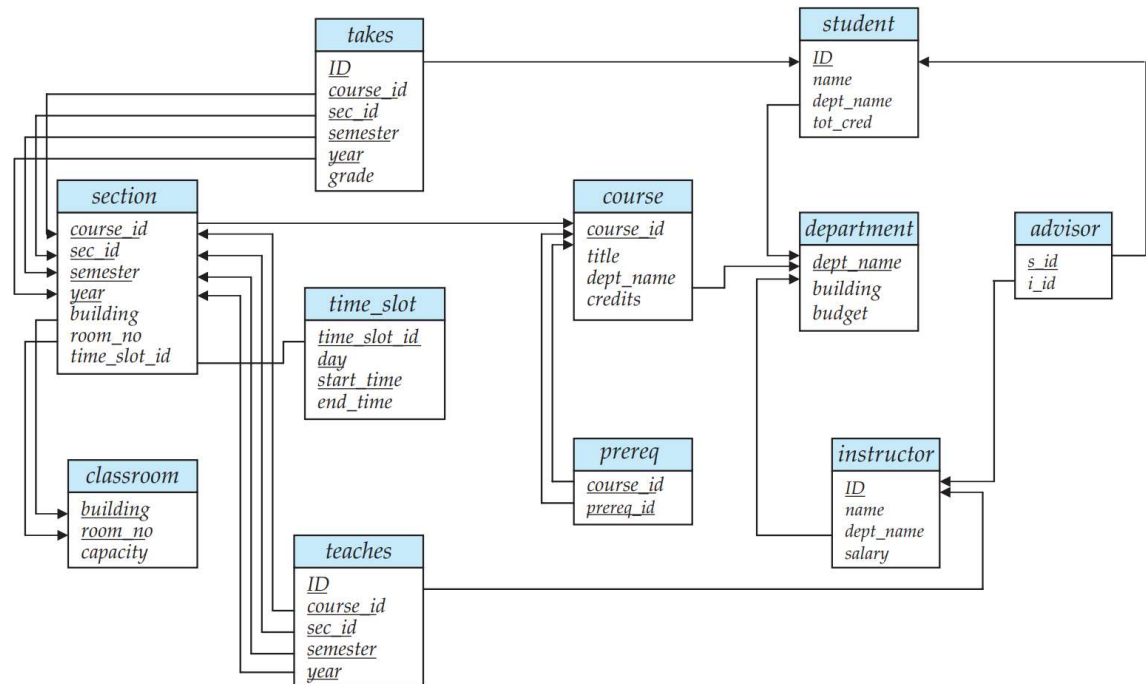
2. Ngôn ngữ đại số quan hệ

Ví dụ: Đối với mỗi biểu thức, hãy giải thích bằng lời biểu thức đó thực hiện chức năng gì.

a. $\sigma_{year > 2009}(takes) \bowtie student$

- $\sigma_{year \geq 2009}(takes) \bowtie student$
- $\sigma_{year \geq 2009}(takes \bowtie student)$
- $\Pi_{ID, name, course_id}(student \bowtie takes)$

b. Tương tự như (a); việc lựa chọn có thể được thực hiện trước thao tác nối.





3. Ngôn ngữ truy vấn SQL

3. Ngôn ngữ truy vấn SQL

- ❖ Tổng quan về Ngôn ngữ truy vấn SQL
- ❖ Định nghĩa dữ liệu SQL
- ❖ Cấu trúc truy vấn cơ bản của truy vấn SQL
- ❖ Các hoạt động cơ bản bổ sung
- ❖ Hoạt động tập hợp
- ❖ Giá trị Null
- ❖ Hàm tổng hợp
- ❖ Truy vấn con lồng nhau
- ❖ Sửa đổi cơ sở dữ liệu

- ❖ Ngôn ngữ IBM Sequel được phát triển như một phần của dự án System R tại Phòng nghiên cứu IBM San Jose
- ❖ Đổi tên thành Ngôn ngữ truy vấn có cấu trúc (SQL)
- ❖ Tiêu chuẩn ANSI và ISO SQL:
 - SQL-86
 - SQL-89
 - SQL-92
 - SQL:1999 (tên ngôn ngữ đã tuân thủ Y2K!)
 - SQL:2003
- ❖ Các hệ thống thương mại cung cấp hầu hết, nếu không muốn nói là tất cả, các tính năng của SQL-92, cùng với các bộ tính năng khác nhau từ các tiêu chuẩn sau này và các tính năng độc quyền đặc biệt.
 - Không phải tất cả các ví dụ ở đây đều có thể hoạt động trên hệ thống cụ thể của bạn

Các phần của SQL

- ❖ DML -- cung cấp khả năng truy vấn thông tin từ cơ sở dữ liệu và chèn các bộ dữ liệu vào, xóa các bộ dữ liệu khỏi và sửa đổi các bộ dữ liệu trong cơ sở dữ liệu.
- ❖ integrity – DDL bao gồm các lệnh để chỉ định các ràng buộc toàn vẹn.
- ❖ View definition -- DDL bao gồm các lệnh để xác định các chế độ xem.
- ❖ Transaction control –bao gồm các lệnh để chỉ định thời điểm bắt đầu và kết thúc của các giao dịch.
- ❖ Embedded SQL và dynamic SQL -- xác định cách các câu lệnh SQL có thể được nhúng trong các ngôn ngữ lập trình mục đích chung.
- ❖ Authorization –bao gồm các lệnh để chỉ định quyền truy cập vào các mối quan hệ và chế độ xem.

Ngôn ngữ định nghĩa dữ liệu SQL

Ngôn ngữ định nghĩa dữ liệu SQL (DDL) cho phép chỉ định thông tin về các mối quan hệ, bao gồm:

- ❖ Sơ đồ cho mỗi mối quan hệ.
- ❖ Kiểu giá trị liên kết với mỗi thuộc tính.
- ❖ Các ràng buộc về tính toàn vẹn
- ❖ Bộ chỉ mục cần duy trì cho mỗi quan hệ.
- ❖ Thông tin bảo mật và ủy quyền cho mỗi quan hệ.
- ❖ Cấu trúc lưu trữ vật lý của mỗi quan hệ trên đĩa.

Các loại miền giá trị trong SQL

- ❖ **char(n)**. Chuỗi ký tự có độ dài cố định, với độ dài do người dùng chỉ định n.
- ❖ **varchar(n)**. Chuỗi ký tự có độ dài thay đổi, với độ dài tối đa do người dùng chỉ định n.int.
- ❖ **Integer** (một tập hợp hữu hạn các số nguyên phụ thuộc vào máy).
- ❖ **smallint**. Small integer (một tập hợp phụ thuộc vào máy của kiểu miền số nguyên).
- ❖ **numeric(p,d)**. Số dấu phẩy cố định, với độ chính xác do người dùng chỉ định là p chữ số, với d chữ số ở bên phải dấu thập phân. (ví dụ: numeric(3,1), cho phép lưu trữ chính xác 44,5, nhưng không phải 444,5 hoặc 0,32)
- ❖ **real, double precision**. Số dấu phẩy động và số dấu phẩy động có độ chính xác kép, với độ chính xác phụ thuộc vào máy.
- ❖ **float(n)**. Số dấu phẩy động, với độ chính xác do người dùng chỉ định là ít nhất n chữ số.

- ❖ Một quan hệ SQL được định nghĩa bằng lệnh create table:

create table r

$(A_1 D_1, A_2 D_2, \dots, A_n D_n,$
 $(\text{integrity-constraint}_1),$
 $\dots,$
 $(\text{integrity-constraint}_k))$

- r is the name of the relation
- each A_i is an attribute name in the schema of relation r
- D_i is the data type of values in the domain of attribute A_i

- ❖ Ví dụ:

create table *instructor* (
 ID **char(5),**
 $name$ **varchar(20),**
 $dept_name$ **varchar(20),**
 $salary$ **numeric(8,2))**

Ràng buộc toàn vẹn trong Create Table

- ❖ Các loại ràng buộc toàn vẹn
 - **primary key** (A_1, \dots, A_n)
 - **foreign key** (A_m, \dots, A_n) **references** r
 - **not null**
- ❖ SQL ngăn chặn bất kỳ bản cập nhật nào vào cơ sở dữ liệu vi phạm ràng buộc toàn vẹn.
- ❖ Ví dụ:

```
create table instructor (  
  ID          char(5),  
  name       varchar(20) not null,  
  dept_name varchar(20),  
  salary    numeric(8,2),  
  primary key (ID),  
  foreign key (dept_name) references department);
```

Một số định nghĩa quan hệ khác

- ❖ **create table** *student* (
 ID **varchar**(5),
 name **varchar**(20) not null,
 dept_name **varchar**(20),
 tot_cred **numeric**(3,0),
 primary key (*ID*),
 foreign key (*dept_name*) **references** *department*);

- ❖ **create table** *takes* (
 ID **varchar**(5),
 course_id **varchar**(8),
 sec_id **varchar**(8),
 semester **varchar**(6),
 year **numeric**(4,0),
 grade **varchar**(2),
 primary key (*ID*, *course_id*, *sec_id*, *semester*, *year*) ,
 foreign key (*ID*) **references** *student*,
 foreign key (*course_id*, *sec_id*, *semester*, *year*) **references** *section*);

- ❖ **create table** *course* (
 course_id **varchar**(8),
 title **varchar**(50),
 dept_name **varchar**(20),
 credits **numeric**(2,0),
 primary key (*course_id*),
 foreign key (*dept_name*) **references** *department*);

❖ Insert

- **insert into** *instructor* **values** ('10211', 'Smith', 'Biology', 66000);

❖ Delete

- Remove all tuples from the *student* relation
 - **delete from** *student*

❖ Drop Table

- **drop table** *r*

❖ Alter

- **alter table** *r* **add** *A D*
 - where *A* is the name of the attribute to be added to relation *r* and *D* is the domain of *A*.
 - All exiting tuples in the relation are assigned *null* as the value for the new attribute.
- **alter table** *r* **drop** *A*
 - where *A* is the name of an attribute of relation *r*
 - Dropping of attributes not supported by many databases.

Cấu trúc truy vấn cơ bản

❖ Một truy vấn SQL điển hình có dạng:

```
select  $A_1, A_2, \dots, A_n$   
from  $r_1, r_2, \dots, r_m$   
where  $P$ 
```

- A_i represents an attribute
- R_i represents a relation
- P is a predicate.

❖ Kết quả của truy vấn SQL là một quan hệ.

Mệnh đề select

- ❖ Mệnh đề **select** liệt kê các thuộc tính mong muốn trong kết quả của truy vấn
 - Tương ứng với phép chiếu của đại số quan hệ
- ❖ Ví dụ: tìm tên của tất cả các giảng viên:

```
select name  
from instructor
```
- ❖ LƯU Ý: Tên SQL không phân biệt chữ hoa chữ thường (tức là bạn có thể sử dụng chữ hoa hoặc chữ thường).
 - E.g., *Name* \equiv *NAME* \equiv *name*
 - Một số người sử dụng chữ in hoa ở bất cứ nơi nào chúng tôi sử dụng phong chữ đậm.

Mệnh đề select (tiếp)

- ❖ SQL cho phép trùng lặp trong các quan hệ cũng như trong kết quả truy vấn.
- ❖ Để buộc loại bỏ các trùng lặp, hãy chèn từ khóa **distinct** sau **select**.
- ❖ Tìm tên khoa của tất cả các giảng viên và xóa các bản trùng lặp

```
select distinct dept_name  
from instructor
```

- ❖ Từ khóa **all** chỉ rõ rằng không nên xóa các mục trùng lặp.

```
select all dept_name  
from instructor
```

dept_name
Comp. Sci.
Finance
Music
Physics
History
Physics
Comp. Sci.
History
Finance
Biology
Comp. Sci.
Elec. Eng.

The select Clause (Cont.)

- ❖ Dấu hoa thị trong mệnh đề select biểu thị “tất cả các thuộc tính”

```
select *  
from instructor
```

- ❖ Một thuộc tính có thể là một nghĩa đen không có mệnh đề **from**

```
select '437'
```

- Kết quả là một bảng có một cột và một hàng duy nhất có giá trị “437”
- Có thể đặt tên cho cột bằng cách sử dụng:

```
select '437' as FOO
```

- ❖ Một thuộc tính có thể là một nghĩa đen có mệnh đề **from**

```
select 'A'  
from instructor
```

- Kết quả là một bảng có một cột và N hàng (số lượng các bộ trong bảng *instructors*), mỗi hàng có giá trị “A”

The select Clause (Cont.)

- ❖ Mệnh đề **select** có thể chứa các biểu thức số học liên quan đến phép toán $+$, $-$, $*$ và $/$, và hoạt động trên các hằng số hoặc thuộc tính của các bộ.

- Truy vấn

```
select ID, name, salary/12  
from instructor
```

→ sẽ trả về một quan hệ giống với quan hệ giảng viên, ngoại trừ giá trị của thuộc tính lương được chia cho 12.

- Có thể đổi tên “*salary/12*” bằng cách sử dụng mệnh đề as:

```
select ID, name, salary/12 as monthly_salary
```


Mệnh đề where

- ❖ Mệnh đề **where** chỉ định các điều kiện mà kết quả phải thỏa mãn
 - Tương ứng với vị từ lựa chọn của đại số quan hệ.

- ❖ Để tìm tất cả các giảng viên trong khoa Comp. Sci.

```
select name
from instructor
where dept_name = 'Comp. Sci.'
```

- ❖ SQL cho phép sử dụng các phép nối logic **and**, **or**, và **not**
- ❖ Các toán hạng của các phép nối logic có thể là các biểu thức liên quan đến các toán tử so sánh **<**, **<=**, **>**, **>=**, **=** và **<>**.
- ❖ Có thể áp dụng phép so sánh cho kết quả của các biểu thức số học
- ❖ Để tìm tất cả các giảng viên trong khoa Comp. Sci. có mức lương **> 70000**

```
select name
from instructor
where dept_name = 'Comp. Sci.' and salary > 70000
```

<i>name</i>
Katz
Brandt

Mệnh đề **from**

- ❖ Mệnh đề **from** liệt kê các quan hệ liên quan đến truy vấn
 - Tương ứng với phép toán tích Descartes của đại số quan hệ.
- ❖ Tìm phép toán tích Descartes *instructor X teaches*

select *

from *instructor, teaches*

- Tạo ra mọi cặp *instructor – teaches* có thể, với tất cả các thuộc tính từ cả hai quan hệ.
 - Đối với các thuộc tính chung (ví dụ: *ID*), các thuộc tính trong bảng kết quả được đổi tên bằng cách sử dụng tên quan hệ (ví dụ: *instructor.ID*)
- ❖ Tích Descartes không hữu ích khi sử dụng trực tiếp nhưng hữu ích khi kết hợp với điều kiện mệnh đề **where** (phép toán lựa chọn trong đại số quan hệ).

- ❖ Tìm tên của tất cả các giảng viên đã dạy một số khóa học và `course_id`
 - ***select name, course_id***
from instructor , teaches
where instructor.ID = teaches.ID
- ❖ Tìm tên của tất cả các giảng viên trong khoa Art đã dạy một số khóa học và `course_id`
 - ***select name, course_id***
from instructor , teaches
where instructor.ID = teaches.ID
and instructor. dept_name = 'Art'

<i>name</i>	<i>course_id</i>
Srinivasan	CS-101
Srinivasan	CS-315
Srinivasan	CS-347
Wu	FIN-201
Mozart	MU-199
Einstein	PHY-101
El Said	HIS-351
Katz	CS-101
Katz	CS-319
Crick	BIO-101
Crick	BIO-301
Brandt	CS-190
Brandt	CS-190
Brandt	CS-319
Kim	EE-181

- ❖ SQL cho phép đổi tên các quan hệ và thuộc tính bằng cách sử dụng mệnh đề **as**:

old-name as new-name

- ❖ Tìm tên của tất cả các giảng viên có mức lương cao hơn một số giảng viên trong 'Comp. Sci'.

- **select distinct** *T.name*
from *instructor as T, instructor as S*
where *T.salary > S.salary and S.dept_name = 'Comp. Sci.'*

- ❖ Từ khóa là tùy chọn và có thể bị lược bỏ
instructor as T \equiv instructor T

Ví dụ tự tham gia

❖ Quan hệ *emp-super*

<i>person</i>	<i>supervisor</i>
Bob	Alice
Mary	Susan
Alice	David
David	Mary

- ❖ Tìm người giám sát của “Bob”
- ❖ Tìm người giám sát của người giám sát “Bob”
- ❖ Bạn có thể tìm thấy TẤT CẢ những người giám sát (trực tiếp và gián tiếp) của “Bob” không?

- ❖ SQL bao gồm một toán tử khớp chuỗi để so sánh trên các chuỗi ký tự. Toán tử **like** sử dụng các mẫu được mô tả bằng hai ký tự đặc biệt:
 - Phần trăm (%). Ký tự % khớp với bất kỳ chuỗi con nào.
 - Dấu gạch dưới (_). Ký tự _ khớp với bất kỳ ký tự nào.
- ❖ Tìm tên của tất cả các giảng viên có tên bao gồm chuỗi con “dar”.

```
select name  
from instructor  
where name like '%dar%'
```

- ❖ Phù hợp với chuỗi “100%”

```
like '100 \%' escape '\'
```

trong đó chúng ta sử dụng dấu gạch chéo ngược (\) làm ký tự thoát.

- ❖ Các mẫu phân biệt chữ hoa chữ thường.
- ❖ Ví dụ về khớp mẫu:
 - 'Intro%' khớp với bất kỳ chuỗi nào bắt đầu bằng "Intro".
 - '%Comp%' khớp với bất kỳ chuỗi nào chứa "Comp" làm chuỗi con.
 - '___' khớp với bất kỳ chuỗi nào có chính xác ba ký tự.
 - '___ %' khớp với bất kỳ chuỗi nào có ít nhất ba ký tự.
- ❖ SQL hỗ trợ nhiều thao tác chuỗi như
 - nối chuỗi (sử dụng "||")
 - chuyển đổi từ chữ hoa sang chữ thường (và ngược lại)
 - tìm độ dài chuỗi, trích xuất chuỗi con, v.v.

Thứ tự hiển thị Bộ (Tuple)

- ❖ Liệt kê theo thứ tự chữ cái tên của tất cả các giảng viên

```
select distinct name  
from   instructor  
order by name
```

- ❖ Chúng ta có thể chỉ định **desc** theo thứ tự giảm dần hoặc **asc** theo thứ tự tăng dần cho mỗi thuộc tính; thứ tự tăng dần là mặc định.

- Ví dụ: **order by** *name* **desc**

- ❖ Có thể sắp xếp theo nhiều thuộc tính

- Ví dụ: **order by** *dept_name*, *name*

Các vị ngữ của mệnh đề Where

- ❖ SQL bao gồm một toán tử so sánh **between**
- ❖ Ví dụ: Tìm tên của tất cả các giảng viên có mức lương từ 90.000 đến 100.000 đô la (tức là ≥ 90.000 đô la và ≤ 100.000 đô la)

```
select name  
from instructor  
where salary between 90000 and 100000
```

- ❖ So sánh bộ
 - **select** *name, course_id*
from *instructor, teaches*
where (*instructor.ID, dept_name*) = (*teaches.ID, 'Biology'*);

Hoạt động tập hợp

- ❖ Tìm các khóa học diễn ra vào mùa thu năm 2017 Fall 2017 **hoặc** Spring 2018
(**select** *course_id* **from** *section* **where** *sem* = 'Fall' **and** *year* = 2017)
union
(**select** *course_id* **from** *section* **where** *sem* = 'Spring' **and** *year* = 2018)
- ❖ Tìm các khóa học diễn ra vào mùa thu năm 2017 Fall 2017 **và** Spring 2018
(**select** *course_id* **from** *section* **where** *sem* = 'Fall' **and** *year* = 2017)
intersect
(**select** *course_id* **from** *section* **where** *sem* = 'Spring' **and** *year* = 2018)
- ❖ Tìm các khóa học diễn ra vào mùa thu năm 2017 Fall 2017 **mà không vào** Spring 2018
(**select** *course_id* **from** *section* **where** *sem* = 'Fall' **and** *year* = 2017)
except
(**select** *course_id* **from** *section* **where** *sem* = 'Spring' **and** *year* = 2018)
- ❖ Để giữ lại tất cả các trùng lặp sử dụng:
 - **union all**,
 - **intersect all**
 - **except all**.

- ❖ Các tuple có thể có giá trị null, được biểu thị bằng **null**, đối với một số thuộc tính của chúng
- ❖ null biểu thị một giá trị không xác định hoặc giá trị không tồn tại.
- ❖ Kết quả của bất kỳ biểu thức số học nào có null là null
 - Ví dụ: $5 + \text{null}$ trả về null
- ❖ Có thể sử dụng vị ngữ **is null** để kiểm tra các giá trị null.
 - Ví dụ: Tìm tất cả các giảng viên có mức lương là null.

```
select name  
from instructor  
where salary is null
```
- ❖ Vị ngữ **is not null** sẽ thành công nếu giá trị mà nó áp dụng không phải là null.

- ❖ SQL coi kết quả của bất kỳ phép so sánh nào liên quan đến giá trị null là không xác định (ngoại trừ các vị ngữ là null và không phải là null).
 - Ví dụ: $5 < \text{null}$ hoặc $\text{null} <> \text{null}$ hoặc $\text{null} = \text{null}$
- ❖ Vị ngữ trong mệnh đề **where** có thể liên quan đến các phép toán Boolean (**and**, **or**, **not**); do đó, định nghĩa của các phép toán Boolean cần được mở rộng để xử lý giá trị không xác định.
 - **and** : $(\text{true and unknown}) = \text{unknown}$,
 $(\text{false and unknown}) = \text{false}$,
 $(\text{unknown and unknown}) = \text{unknown}$
 - **or**: $(\text{unknown or true}) = \text{true}$,
 $(\text{unknown or false}) = \text{unknown}$
 $(\text{unknown or unknown}) = \text{unknown}$
- ❖ Kết quả của vị ngữ mệnh đề **where** được coi là sai nếu nó được đánh giá là không xác định

Các hàm kết tập

- ❖ Các hàm này hoạt động trên tập hợp nhiều giá trị của một cột trong quan hệ và trả về một giá trị

avg: average value

min: minimum value

max: maximum value

sum: sum of values

count: number of values

- ❖ Tìm mức lương **trung bình** của giảng viên khoa Khoa học máy tính
 - `select avg (salary)`
`from instructor`
`where dept_name= 'Comp. Sci.';`
- ❖ Tìm **tổng số** giảng viên giảng dạy một khóa học trong học kỳ mùa xuân năm 2018
 - `select count (distinct ID)`
`from teaches`
`where semester = 'Spring' and year = 2018;`
- ❖ Tìm **số lượng** các bộ trong quan hệ *course*
 - `select count (*)`
`from course;`

Các hàm kết tập– Group By

- ❖ Tìm mức lương trung bình của giảng viên trong mỗi khoa
 - **select dept_name, avg (salary) as avg_salary**
from instructor
group by dept_name;
- ❖ Các thuộc tính trong mệnh đề **select** bên ngoài các hàm kết hợp phải xuất hiện trong danh sách **group by**
 - */* truy vấn lỗi */*
select dept_name, ID, avg (salary)
from instructor
group by dept_name;

dept_name	avg_salary
Biology	72000
Comp. Sci.	77333
Elec. Eng.	80000
Finance	85000
History	61000
Music	40000
Physics	91000

ID	name	dept_name	salary
76766	Crick	Biology	72000
45565	Katz	Comp. Sci.	75000
10101	Srinivasan	Comp. Sci.	65000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000
12121	Wu	Finance	90000
76543	Singh	Finance	80000
32343	El Said	History	60000
58583	Califieri	History	62000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
22222	Einstein	Physics	95000

Các hàm kết tập – mệnh đề Having

- ❖ Tìm tên và mức lương trung bình của tất cả các phòng ban **có mức lương trung bình lớn hơn 42000**

```
select dept_name, avg (salary) as avg_salary  
from instructor  
group by dept_name  
having avg (salary) > 42000;
```

- ❖ Lưu ý: các vị ngữ trong mệnh đề **having** được áp dụng sau khi hình thành các nhóm trong khi các vị ngữ trong mệnh đề **where** được áp dụng trước khi hình thành các nhóm

Các truy vấn con lồng nhau

- ❖ SQL cung cấp một cơ chế để lồng các truy vấn con. **Một truy vấn con** là một biểu thức **select-from-where** được lồng trong một truy vấn khác.
- ❖ Việc lồng có thể được thực hiện trong truy vấn SQL sau:

select A_1, A_2, \dots, A_n
from r_1, r_2, \dots, r_m
where P

cụ thể:

- **Mệnh đề from:** r_i có thể được thay thế bằng bất kỳ truy vấn con hợp lệ nào
- **Mệnh đề Where:** P có thể được thay thế bằng một biểu thức có dạng:

B <operation> (subquery)

B là thuộc tính và <operation> to be defined later.

- **Mệnh đề Select:**

A_i có thể được thay thế bằng một truy vấn con tạo ra một giá trị đơn

Các truy vấn con lồng nhau

- ❖ Truy vấn lồng nhau là một cách để thực hiện các truy vấn phức tạp bằng cách nhúng một truy vấn vào một truy vấn khác. Truy vấn bên ngoài có thể áp dụng một số điều kiện cho kết quả của truy vấn bên trong.
- ❖ Sử dụng các bảng **STUDENT**, **COURSE**, **STUDENT_COURSE** để hiểu các truy vấn lồng nhau.

STUDENT				
S_ID	S_NAME	S_ADDRESS	S_PHONE	S_AGE
S1	RAM	DELHI	9455123451	18
S2	RAMESH	GURGAON	9652431543	18
S3	SUJIT	ROHTAK	9156253131	20
S4	SURESH	DELHI	9156768971	18

COURSE	
C_ID	C_NAME
C1	DSA
C2	Programming
C3	DBMS

STUDENT_COURSE	
S_ID	C_ID
S1	C1
S1	C3
S2	C1
S3	C2
S4	C2
S4	C3

Các truy vấn con lồng nhau

❖ **Truy vấn lồng nhau độc lập:** Trong các truy vấn lồng nhau độc lập, việc thực thi truy vấn bắt đầu từ truy vấn trong cùng đến truy vấn ngoài cùng. Việc thực hiện truy vấn bên trong độc lập với truy vấn bên ngoài, nhưng kết quả của truy vấn bên trong được sử dụng để thực hiện truy vấn bên ngoài.

- Các toán tử khác nhau như **IN**, **NOT IN**, **ANY**, **ALL**, v.v. được sử dụng để viết các truy vấn lồng nhau độc lập.

❖ **IN:** Nếu muốn tìm ra **S_ID** đã đăng ký trong **C_NAME** 'DSA' hoặc 'DBMS', có thể viết nó với sự trợ giúp của truy vấn lồng nhau độc lập và toán tử IN.

- Từ bảng **COURSE** , có thể tìm ra **C_ID** cho **C_NAME** 'DSA' hoặc 'DBMS'
- Sử dụng các **C_ID** này để tìm các **S_ID** từ BẢNG **STUDENT_COURSE**.

COURSE	
C_ID	C_NAME
C1	DSA
C2	Programming
C3	DBMS

STUDENT_COURSE	
S_ID	C_ID
S1	C1
S1	C3
S2	C1
S3	C2
S4	C2
S4	C3

Các truy vấn con lồng nhau

- ❖ **IN:** Nếu muốn tìm ra **S_ID** đã đăng ký trong **C_NAME** 'DSA' hoặc 'DBMS', có thể viết nó với sự trợ giúp của truy vấn lồng nhau độc lập và toán tử IN.

- ❖ **BƯỚC 1:** Tìm **C_ID** cho **C_NAME** ='DSA' hoặc 'DBMS'

```
Select C_ID
from COURSE
where C_NAME = 'DSA' or C_NAME = 'DBMS'
```

- ❖ **BƯỚC 2:** Sử dụng **C_ID** của bước 1 để tìm **S_ID**

```
Select S_ID
from STUDENT_COURSE
where C_ID IN (SELECT C_ID from COURSE where C_NAME = 'DSA'
or C_NAME='DBMS')
```

- **Truy vấn bên trong** sẽ trả về một tập hợp có các thành viên C1 và C3 và truy vấn bên ngoài sẽ trả về những **S_ID** mà **C_ID** bằng với bất kỳ thành viên nào của tập hợp (C1 và C3 trong trường hợp này). Vì vậy, nó sẽ trả về S1, S2 và S4.
- **Lưu ý:** Nếu chúng ta muốn tìm tên của những **STUDENT** đã đăng ký vào 'DSA' hoặc 'DBMS', thì có thể thực hiện như sau:

```
Select S_NAME from STUDENT where S_ID IN
(Select S_ID from STUDENT_COURSE where C_ID IN
(SELECT C_ID from COURSE where C_NAME='DSA' or C_NAME='DBMS'));
```

Các truy vấn con lồng nhau

❖ **NOT IN:** Nếu chúng ta muốn tìm ra **S_ID** của **STUDENT** chưa đăng ký 'DSA' cũng như 'DBMS', thì có thể thực hiện như sau:

Select **S_ID**

from **STUDENT**

where **S_ID** NOT IN

(Select **S_ID** from **STUDENT_COURSE** where **C_ID** IN

(SELECT **C_ID** from **COURSE** where **C_NAME**='DSA'

or **C_NAME**='DBMS'));

- Truy vấn trong cùng sẽ trả về một tập hợp có các thành viên C1 và C3.
- Truy vấn bên trong thứ hai sẽ trả về những **S_ID** mà **C_ID** bằng với bất kỳ thành viên nào của tập hợp (C1 và C3 trong trường hợp này) là S1, S2 và S4.
- Truy vấn ngoài cùng sẽ trả về những **S_ID** trong đó **S_ID** không phải là thành viên của tập hợp (S1, S2 và S4). Vì vậy nó sẽ trả về S3.

Các truy vấn con lồng nhau

- ❖ **Truy vấn lồng nhau có liên quan:** Trong các truy vấn lồng nhau có liên quan, đầu ra của truy vấn bên trong phụ thuộc vào hàng hiện đang được thực thi trong truy vấn bên ngoài.
- ❖ Ví dụ; Nếu muốn tìm ra **S_NAME** trong **STUDENT** những người đã đăng ký vào **C_ID = 'C1'**, thì có thể thực hiện với sự trợ giúp của truy vấn lồng nhau có liên quan như:

```
Select S_NAME  
from STUDENT S  
where EXISTS (select * from STUDENT_COURSE SC where  
S.S_ID=SC.S_ID and SC.C_ID='C1')
```

- Với mỗi hàng **STUDENT S**, nó sẽ tìm các hàng từ **STUDENT_COURSE** trong đó **S. S_ID = SC. S_ID** và **SC. C_ID = 'C1'**.
- Nếu đối với **S_ID** từ **STUDENT S**, ít nhất một hàng tồn tại trong **STUDENT_COURSE SC** với **C_ID = 'C1'** thì truy vấn bên trong sẽ trả về true và **S_ID** tương ứng sẽ được trả về dưới dạng đầu ra.

STUDENT				
S_ID	S_NAME	S_ADDRESS	S_PHONE	S_AGE
S1	RAM	DELHI	9455123451	18
S2	RAMESH	GURGAON	9652431543	18
S3	SUJIT	ROHTAK	9156253131	20
S4	SURESH	DELHI	9156768971	18

STUDENT_COURSE	
S_ID	C_ID
S1	C1
S1	C3
S2	C1
S3	C2
S4	C2
S4	C3

Set Membership (Thành viên tập)

Set Membership

- ❖ Tìm các khóa học được mở vào mùa thu năm 2017 và mùa xuân năm 2018

```
select distinct course_id  
from section  
where semester = 'Fall' and year= 2017 and  
       course_id in (select course_id  
                     from section  
                     where semester = 'Spring' and year= 2018);
```

- ❖ Tìm các khóa học được mở vào mùa thu năm 2017 nhưng không mở vào mùa xuân năm 2018

```
select distinct course_id  
from section  
where semester = 'Fall' and year= 2017 and  
       course_id not in (select course_id  
                        from section  
                        where semester = 'Spring' and year= 2018);
```

Set Membership (Cont.)

- ❖ Hãy nêu tên tất cả những giảng viên có tên không phải là “Mozart” hay “Einstein”

```
select distinct name  
from instructor  
where name not in ('Mozart', 'Einstein')
```

- ❖ Tìm tổng số sinh viên (riêng biệt) đã học các phần khóa học do giảng viên có *ID* 10101 giảng dạy

```
select count (distinct ID)  
from takes  
where (course_id, sec_id, semester, year) in  
      (select course_id, sec_id, semester, year  
       from teaches  
       where teaches.ID= 10101);
```

- ❖ Lưu ý: Truy vấn trên có thể được viết theo cách đơn giản hơn nhiều. Công thức trên chỉ đơn giản là để minh họa các tính năng của SQL

Set Comparison (So sánh tập)

So sánh tập– mệnh đề “some”

- ❖ Tìm tên những giảng viên có mức lương cao hơn mức lương của một số giảng viên (ít nhất một giảng viên) trong khoa Sinh học.

```
select distinct T.name  
from instructor as T, instructor as S  
where T.salary > S.salary and S.dept name = 'Biology';
```

- ❖ Truy vấn tương tự sử dụng mệnh đề > **some**

```
select name  
from instructor  
where salary > some (select salary  
                     from instructor  
                     where dept name = 'Biology');
```

Định nghĩa của mệnh đề “some”

❖ $F <\text{comp}> \text{some } r \Leftrightarrow \exists t \in r \text{ such that } (F <\text{comp}> t)$

Where $<\text{comp}>$ can be: $<, \leq, >, =, \neq$

$(5 < \text{some } \begin{array}{|c|} \hline 0 \\ \hline 5 \\ \hline 6 \\ \hline \end{array}) = \text{true}$
(read: 5 < some tuple in the relation)

$(5 < \text{some } \begin{array}{|c|} \hline 0 \\ \hline 5 \\ \hline \end{array}) = \text{false}$

$(5 = \text{some } \begin{array}{|c|} \hline 0 \\ \hline 5 \\ \hline \end{array}) = \text{true}$

$(5 \neq \text{some } \begin{array}{|c|} \hline 0 \\ \hline 5 \\ \hline \end{array}) = \text{true (since } 0 \neq 5)$

$(= \text{some}) \equiv \text{in}$

However, $(\neq \text{some}) \not\equiv \text{not in}$

So sánh tập– mệnh đề “all”

- ❖ Tìm tên của tất cả các giảng viên có mức lương cao hơn mức lương của tất cả các giảng viên trong khoa Sinh học.

```
select name  
from instructor  
where salary > all (select salary  
                        from instructor  
                        where dept name = 'Biology');
```


Định nghĩa của mệnh đề “all”

$$\diamond F <\text{comp}> \mathbf{all} \ r \Leftrightarrow \forall t \in r \ (F <\text{comp}> t)$$

$$(5 < \mathbf{all} \begin{array}{|c|} \hline 0 \\ \hline 5 \\ \hline 6 \\ \hline \end{array}) = \text{false}$$

$$(5 < \mathbf{all} \begin{array}{|c|} \hline 6 \\ \hline 10 \\ \hline \end{array}) = \text{true}$$

$$(5 = \mathbf{all} \begin{array}{|c|} \hline 4 \\ \hline 5 \\ \hline \end{array}) = \text{false}$$

$$(5 \neq \mathbf{all} \begin{array}{|c|} \hline 4 \\ \hline 6 \\ \hline \end{array}) = \text{true (since } 5 \neq 4 \text{ and } 5 \neq 6)$$

$(\neq \mathbf{all}) \equiv \mathbf{not in}$

However, $(= \mathbf{all}) \not\equiv \mathbf{in}$

- ❖ Câu trúc **exists** trả về giá trị **true** nếu truy vấn con đôi số không rỗng.
- ❖ **exists** $r \Leftrightarrow r \neq \emptyset$
- ❖ **not exists** $r \Leftrightarrow r = \emptyset$
- ❖ Sử dụng mệnh đề **exists**:
- ❖ Một cách khác để chỉ định truy vấn “Tìm tất cả các khóa học được giảng dạy trong cả học kỳ Mùa thu năm 2017 và học kỳ Mùa xuân năm 2018”

```
select course_id
from section as S
where semester = 'Fall' and year = 2017 and
      exists (select *
              from section as T
              where semester = 'Spring' and year = 2018
                 and S.course_id = T.course_id);
```

- ❖ Tên tương quan – biến *S* trong truy vấn bên ngoài
- ❖ Truy vấn con tương quan – truy vấn bên trong

Sử dụng mệnh đề “not exists”

- ❖ Tìm tất cả sinh viên đã học tất cả các khóa học được cung cấp trong khoa Sinh học.

```
select distinct S.ID, S.name  
from student as S  
where not exists ( (select course_id  
                    from course  
                    where dept_name = 'Biology')  
except  
                  (select T.course_id  
                    from takes as T  
                    where S.ID = T.ID));
```

- Truy vấn lồng nhau đầu tiên liệt kê tất cả các khóa học được cung cấp trong khoa Sinh học
- Truy vấn lồng nhau thứ hai liệt kê tất cả các khóa học mà một sinh viên cụ thể đã học

- ❖ Chú ý: $X - Y = \emptyset \Leftrightarrow X \subseteq Y$
- ❖ Lưu ý: Không thể viết truy vấn này bằng cách sử dụng = all và các biến thể của nó

Test for Absence of Duplicate Tuples

- ❖ Cấu trúc **unique** kiểm tra xem một truy vấn con có bất kỳ bộ dữ liệu trùng lặp nào trong kết quả của nó hay không.
- ❖ Cấu trúc **unique** đánh giá là " true " nếu một truy vấn con nhất định không chứa bất kỳ bản sao nào.
- ❖ Tìm tất cả các khóa học được cung cấp nhiều nhất một lần trong năm 2017
- ❖

```
select T.course_id
from course as T
where unique ( select R.course_id
                from section as R
                where T.course_id= R.course_id
                and R.year = 2017);
```

Các truy vấn con trong mệnh đề From

Các truy vấn con trong mệnh đề From

- ❖ SQL cho phép sử dụng biểu thức truy vấn con trong mệnh đề **from**
- ❖ Tìm mức lương trung bình của các giảng viên trong các khoa có mức lương trung bình lớn hơn 42.000 đô la.”

```
select dept_name, avg_salary  
from ( select dept_name, avg (salary) as avg_salary  
      from instructor  
      group by dept_name)  
where avg_salary > 42000;
```

- ❖ Chú ý: không cần sử dụng mệnh đề **having**
- ❖ Cách viết khác cho truy vấn trên

```
select dept_name, avg_salary  
from ( select dept_name, avg (salary)  
      from instructor  
      group by dept_name)  
as dept_avg (dept_name, avg_salary)  
where avg_salary > 42000;
```

- ❖ Mệnh đề with cung cấp một cách để xác định mối quan hệ tạm thời mà định nghĩa của nó chỉ khả dụng cho truy vấn trong đó mệnh đề with xuất hiện.

- ❖ Tìm tất cả các phòng ban có ngân sách tối đa
with *max_budget (value)* **as**
 (**select** **max**(*budget*)
 from *department*)
select *department.name*
from *department, max_budget*
where *department.budget = max_budget.value;*

Các truy vấn phức tạp sử dụng mệnh đề With

- ❖ Tìm tất cả các phòng ban có tổng lương lớn hơn mức lương trung bình của tổng lương tại tất cả các phòng ban

```
with dept_total (dept_name, value) as
    (select dept_name, sum(salary)
     from instructor
     group by dept_name),
dept_total_avg(value) as
    (select avg(value)
     from dept_total)
select dept_name
from dept_total, dept_total_avg
where dept_total.value > dept_total_avg.value;
```


Truy vấn con vô hướng

- ❖ Truy vấn phụ vô hướng là truy vấn được sử dụng khi mong đợi một giá trị đơn
- ❖ Liệt kê tất cả các khoa cùng với số lượng giảng viên trong mỗi khoa

```
select dept_name,  
      ( select count(*)  
        from instructor  
        where department.dept_name = instructor.dept_name)  
      as num_instructors  
from department;
```

- ❖ Lỗi thời gian chạy nếu truy vấn con trả về nhiều hơn một bộ kết quả

- ❖ Xóa các tuple khỏi một quan hệ nhất định.
- ❖ Chèn các tuple mới vào một quan hệ nhất định
- ❖ Cập nhật các giá trị trong một số tuple trong một quan hệ nhất định

- ❖ Xóa tất cả giảng viên

```
delete from instructor
```

- ❖ Xóa tất cả giảng viên ở khoa Tài chính

```
delete from instructor
```

```
where dept_name = 'Finance';
```

- ❖ Xóa tất cả các bộ trong quan hệ *instructor* đối với những giảng viên làm việc tại khoa có trụ sở tại tòa nhà Watson.

```
delete from instructor
```

```
where dept name in (select dept name  
                        from department  
                        where building = 'Watson');
```

- ❖ Xóa tất cả các giảng viên có mức lương thấp hơn mức lương trung bình của các giảng viên

```
delete from instructor  
where salary < (select avg (salary)  
                  from instructor);
```

- Vấn đề: khi chúng ta xóa các tuple khỏi *instructor*, mức lương trung bình sẽ thay đổi
- Giải pháp được sử dụng trong SQL:
 1. Đầu tiên, tính **avg** (*salary*) và tìm tất cả các tuple để xóa
 2. Tiếp theo, xóa tất cả các tuple được tìm thấy ở trên (mà không tính lại **avg** hoặc kiểm tra lại các tuple)

- ❖ Thêm một tuple mới vào *course*

insert into *course*

values ('CS-437', 'Database Systems', 'Comp. Sci.', 4);

- ❖ hoặc tương đương

insert into *course* (*course_id*, *title*, *dept_name*, *credits*)

values ('CS-437', 'Database Systems', 'Comp. Sci.', 4);

- ❖ Thêm một tuple mới vào *student* với *tot_creds* được đặt thành null

insert into *student*

values ('3003', 'Green', 'Finance', *null*);

- ❖ Mỗi sinh viên trong khoa Music đạt được hơn 144 tín chỉ sẽ trở thành giảng viên khoa Music với mức lương là 18.000 đô la

insert into *instructor*

select *ID, name, dept_name, 18000*

from *student*

where *dept_name = 'Music' and total_cred > 144;*

- ❖ Câu lệnh **select from where** được đánh giá đầy đủ trước khi bắt kỳ kết quả nào của nó được chèn vào mỗi quan hệ.

Nếu không, các truy vấn như

insert into table1 select * from table1

sẽ gây ra vấn đề

- ❖ Tăng lương 5% cho tất cả giảng viên

update *instructor*

set *salary* = *salary* * 1.05

- ❖ Tăng lương 5% cho những giảng viên có thu nhập dưới 70000

update *instructor*

set *salary* = *salary* * 1.05

where *salary* < 70000;

- ❖ Tăng lương 5% cho những giảng viên có mức lương thấp hơn mức trung bình

update *instructor*

set *salary* = *salary* * 1.05

where *salary* < (**select** **avg** (*salary*)
from *instructor*);

- ❖ Tăng lương cho những giảng viên có mức lương trên 100.000 đô la lên 3% và tất cả những người khác lên 5%

- Viết hai câu lệnh cập nhật:

update *instructor*

set *salary* = *salary* * 1.03

where *salary* > 100000;

update *instructor*

set *salary* = *salary* * 1.05

where *salary* <= 100000;

- Thứ tự rất quan trọng
- Có thể thực hiện tốt hơn bằng cách sử dụng câu lệnh case (**Câu lệnh Case cho các cập nhật có điều kiện**)

update *instructor*

set *salary* = **case**

when *salary* <= 100000 **then** *salary* * 1.05

else *salary* * 1.03

end

Các Updates với truy vấn con vô hướng

- ❖ Tính toán lại và cập nhật giá trị `tot_creds` cho tất cả học sinh

update *student S*

```
set tot_cred = (select sum(credits)
                from takes, course
                where takes.course_id = course.course_id and
                      S.ID = takes.ID and
                      takes.grade <> 'F' and
                      takes.grade is not null);
```

- ❖ Đặt `tot_creds` thành null cho những sinh viên chưa học bất kỳ khóa học nào
- ❖ Thay vì `sum(credits)`, sử dụng:

```
case
  when sum(credits) is not null then sum(credits)
  else 0
end
```