GIÁO TRÌNH
CƠ SỞ DỮ LIỆU

Chương 1
KHÁI QUÁT VỀ CƠ SỞ DỮ LIỆU

1.1. CÁC KHÁI NIỆM CƠ BẢN

1. Cơ sở dữ liệu (database)

Một cơ sở dữ liệu (CSDL) là một tập hợp các dữ liệu có liên quan với nhau được lưu trữ trong máy tính theo một quy định nhất định nhằm phục vụ cho một mục đích quản lý nào đó.

Ví dụ: CSDL phục vụ cho việc quản lý các chuyến bay của một hãng hàng không cung cấp các thông tin vế số hiệu chuyến bay, nơi xuất phát, nơi đến, số chỗ ngồi, ngày bay của tất cả các chuyến bay trong năm; CSDL phục vụ cho công tác quản lý đào tạo trong trường chứa các dữ liệu phản ánh thông tin về học sinh, giáo viên, môn học, phòng học,...

2. Hệ quản trị cơ sở dữ liệu (*DataBase Management System)*

Hệ quản trị CSDL (HQTCSDL) là phần mềm cho phép người dùng giao tiếp với cơ sở dữ liệu, và thông qua đó cung cấp một môi trường thuận lợi và hiệu quả để tìm kiếm và lưu trữ thông tin của cơ sở dữ liệu.

Ví dụ: Hệ quản trị cơ sở dữ liệu FOXPRO, hệ quản trị cơ sở dữ liệu SQL SERVER,...

Việc phân biệt một HQTCSDL với các phần mềm khác được dựa trên hai đặc trưng cơ bản sau:

- *Quản lý dữ liệu bền*. Đây là khả năng lưu trữ các cơ sở dữ liệu phục vụ cho mục tiêu khai thác lâu dài.

- *Truy cập các khối lượng lớn dữ liệu một cách hiệu quả*. HQTCSDL cung cấp các thao tác cũng như phương pháp tổ chức dữ liệu nhằm giúp cho người sử dụng truy cập vào các dữ liệu trong CSDL một cách nhanh chóng và thuận lợi.

3. Hệ cơ sở dữ liệu

Hệ cơ sở dữ liệu là một hệ thống phần mềm nhằm quản lý cơ sở dữ liệu của một hệ thống thông tin cụ thể nào đó.

Như vậy các thành phần bên trong một hệ cơ sở dữ liệu gồm có: chương trình, cơ sở dữ liệu, hệ quản trị cơ sở dữ liệu, người sử dụng.

1.2. KHẢ NĂNG CỦA MỘT HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

- Quản lý cơ sở dữ liệu lâu dài (dữ liệu không bị mất khi kết thúc).
- Quản lý một số lượng lớn dữ liệu.

- ❖ Cho phép truy cập vào mỗi khối lượng dữ liệu lớn với yêu cầu xử lý nhanh.
- ❖ Cung cấp ít nhất một mô hình dữ liệu, qua đó người dùng có thể quan sát được dữ liệu.
- ❖ Cung cấp một ngôn ngữ bậc cao, qua đó người dùng có thể định nghĩa dữ liệu và xử lý dữ liệu.

1.3. Các mức trừu tượng hóa mô hình cơ sở dữ liệu

Mô hình dữ liệu là sự hình thức hóa toán học của dữ liệu bao gồm hai phần:

- Ký hiệu mô tả dữ liệu

- Tập hợp các phép toán

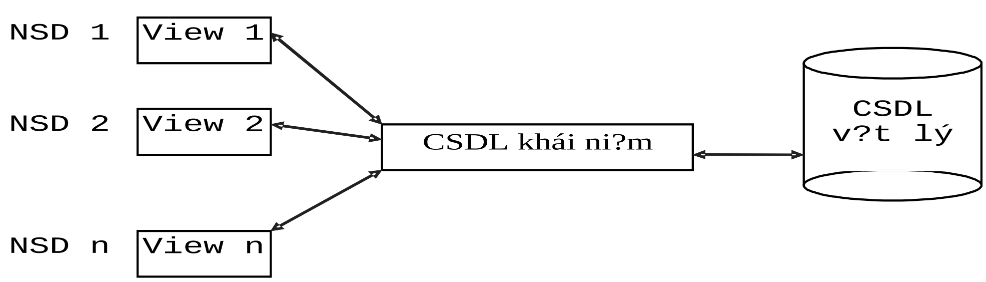
Mỗi mô hình CSDL có ba mức trừu tượng:

- Mức vật lý: Mức thấp nhất của sự trừu tượng mô tả dữ liệu được lưu trữ một cách thực sự như thế nào. Tại mức này CSDL được xem là một bộ các tập tin, các tập tin chỉ mục hoặc cấu trúc lưu trữ khác gọi chung là CSDL vật lý. CSDL vật lý tồn tại bền trong các thiết bị lưu trữ phụ và nhiều CSDL vật lý có thể được quản lý bởi một HQTCSDL.

- Mức khái niệm: Mức cao hơn tiếp theo của sự trừu tượng mô tả những dữ liệu nào được lưu trữ trong CSDL và các mối quan hệ nào tồn tại giữa các dữ liệu này. CSDL mức khái niệm là sự trừu tượng hóa thế giới thực khi nó gắn liền với người sử dụng CSDL. Một HQTCSDL cung cấp một ngôn ngữ định nghĩa mức khái niệm (*DDL-Data Definition Language*) thường gọi là mô hình cơ sở dữ liệu (database model).

- Mức khung nhìn (View): Mức cao nhất của sự trừu tượng mô tả chỉ một phần của toàn bộ CSDL. Mặc dầu sử dụng cấu trúc đơn giản hơn mức logic, một số phức tạp vẫn còn tồn tại do kích thước lớn của CSDL.Hệ thống có thể cung cấp nhiều khung nhìn đối với cùng một CSDL.

Ta có thể hình dung các mức trừu tượng của CSDL như trong hình vẽ sau đây:



Ví dụ. Xét một mảng dữ liệu hai chiều n x m phần tử .

- Tại mức quan niệm ta có thể hình dung mảng này như là một bảng có n dòng và m cột. Mảng này được được mô tả trong một vài ngôn ngữ lập trình C như sau:

int A[n][m] ;

Ở mức này ta không biết được dữ liệu được lưu trữ như thế nào trong thiết bị nhớ mà chỉ có thể biết rằng phần tử ở dòng i cột j là A[i][j].

- Mức vật lý của mảng chính là mảng được lưu trữ trong thiết bị nhớ.

- Mức khung nhìn: Từ mức trừu tượng ở trên ta có thể mô tả các khung nhìn của mảng. Chẳng hạn, cho biết tổng tất cả các giá trị trên dòng i.

$$f(i) = \sum_{j=1}^{m} A[i][j]$$

1.4. Lược đồ và thể hiện

Để xây dựng các CSDL chúng ta phải hoạch định cho các dữ liệu trong CSDL một qui cách lưu trữ. Qui cách này này được hình thức hoá như là khung cho mọi dữ liệu trong CSDL, gọi là *lược đồ CSDL*.

Ví dụ: Để tổ chức CSDL về các nhân viên trong một cơ quan chúng ta tạo trước lược đồ NHANVIEN(HOTEN,NAMSINH ,DIACHI, SODT) để lưu trữ các dữ liệu bao gồm: họ tên, năm sinh, địa chỉ, số điện thoại của các nhân viên.

Nội dung hiện thời của các dữ liệu trên lược đồ gọi là một thể hiện của lược đồ. Chẳng hạn, với lược đồ trên thì bảng sau đây là một thể hiện:

Nguyễn Van Anh	1957	Phú vang	3837756
Hoàng Thị Lan	1958	14. Lê Lợi, Huế	3825424
Huỳnh Dung	1958	315 Chi Lăng, Huế	3545245
Trần Quang	1956	12 Đống Đa, Huế	3822789
Phan Thanh Bình	1960	32 Chi Lăng, Huế	3345053

Như vậy một lược đồ có thể có nhiều thể hiện khác nhau.

1.5. Các ngôn ngữ CSDL

Trong các ngôn ngữ chương trình thì các chỉ thị mô tả và các chỉ thị thực hiện là hai bộ phận của một ngôn ngữ. Trong các HQTCSDL hai chức năng mô tả và tính toán nói chung được chia thành hai ngôn ngữ khác nhau.

Các ngôn ngữ định nghĩa dữ liệu (DDL)

Ngôn ngữ định nghĩa dữ liệu DDL (Data Definition Language) không phải là ngôn ngữ thủ tục mà chỉ là mô tả các loại đối tượng và quan hệ giữa các đối tượng.

Chẳng hạn, các chuyến bay của một hãng hàng không có thể được lưu trữ trong một CSDL được định nghĩa như sau:

CREATE TABLE CHUYEN-BAY

(SO:INT, NG: CHAR(6),CH:INT , TU : CHAR (3),DEN : CHAR (3));

trong đó SO là số hiệu chuyến bay, được mô tả là một số nguyên; NG là ngày tháng thực hiện chuyến bay, được mô tả là một chuỗi 6 ký tự; CH là số chỗ ngồi chưa được đặt trên chuyến bay, được mô tả là một số nguyên; TU, DEN là ký hiệu nơi xuất phát và nơi đến, mỗi địa điểm được viết tắt bởi 3 ký tự.

Các ngôn ngữ xử lý dữ liệu

Các hệ QTCSDL đòi hỏi phải có ngôn ngữ để xử lý các phép toán trên CSDL gọi là DML (Data Manipulation Language) hoặc ngôn ngữ truy vấn (Query Language).

Chẳng hạn, xét các yêu cầu xử lý dưới đây trên ngôn ngữ SQL đối với cơ sở dữ liệu CHUYEN-BAY.

- Giảm 4 chỗ ngồi trên chuyến bay 123 ngày 31 tháng 8 có thể viết:

UPDATE CHUYEN-BAY

SET CH = CH - 4

WHERE SO = 123 AND NG = ‘ AUG 31 ‘ ;

- Thêm vào chuyến bay 147 có 72 chỗ ngồi từ Hà nội (HAN) đến Huế (HUI).

INSERT INTO CHUYEN-BAY

VALUES (147, ‘AUG 21‘ , 72 , ‘HAN‘ , ‘HUI‘);

- Tìm lại từ CSDL số chỗ ngồi còn trống trên chuyến bay 148 ngày 24 tháng 7.

SELECT CH

FROM CHUYEN-BAY

WHERE SO= 148 AND NG = ‘JULY 24‘;

- Tìm lại tất cả các chuyến bay từ Hà nội đến Huế ngày 20 tháng 8.

SELECT SO

FROM CHUYEN-BAY

WHERE NG = ‘AUG 20‘

TU = ‘HAN‘ AND DEN = ‘HUI‘;

Các ngôn ngữ chủ (Host Language)

Ngoài những phép xử lý thông thường sẵn có trong các HQTCSDL các chương trình ứng dụng cần phải có thêm một số công việc phức tạp hơn. Chẳng hạn, một chương trình được sử dụng bằng một hãng hàng không để đăng ký chỗ không chỉ cần truy tìm số chỗ trống từ CSDL mà còn phải làm các công việc phức tạp hơn như in vé, làm giấy hẹn, đối thoại với người sử dụng,... Như vậy, các chương trình để xử lý dữ liệu cần được viết chung trong một ngôn ngữ chủ - đó là ngôn ngữ thuận tiện cho việc lập trình, chẳng hạn C, C++. Ngôn ngữ chủ được sử dụng cho các quyết định, hiển thị câu hỏi, đọc câu trả lời,...

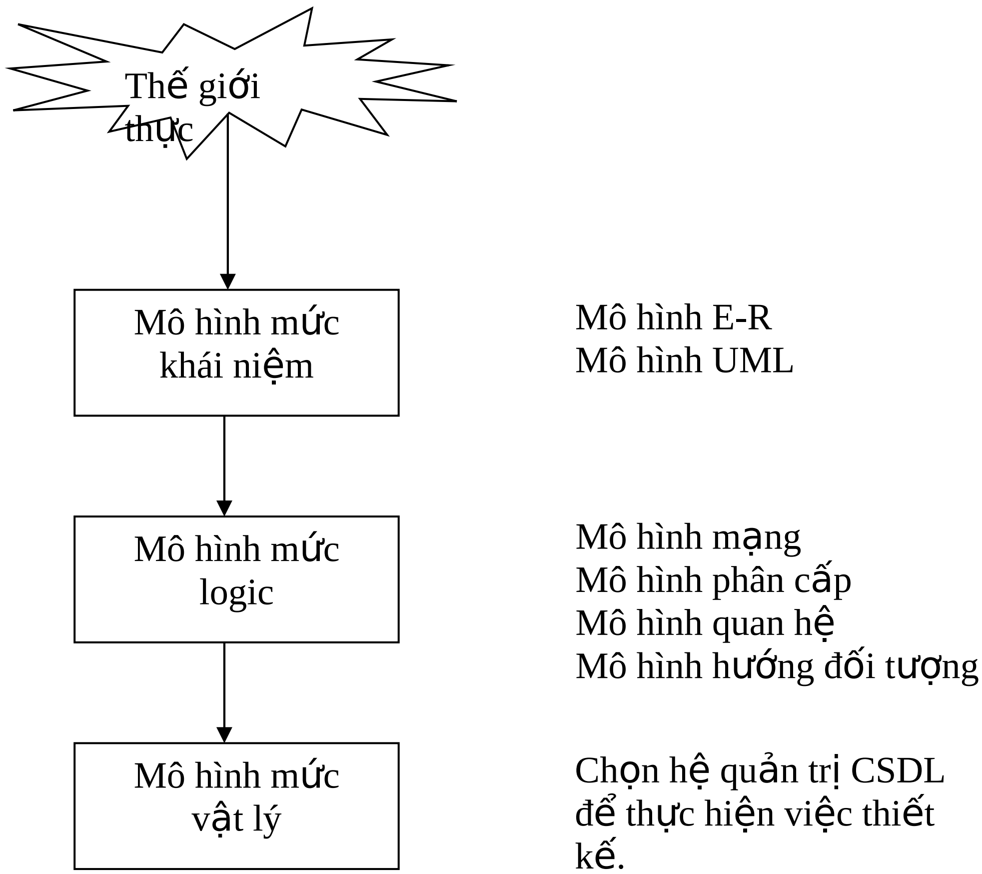
1.6. CÁC MÔ HÌNH CƠ SỞ DỮ LIỆU

Mỗi hệ quản trị cơ sở dữ liệu cung cấp một mô hình cho cơ sở dữ liệu và thông qua mô hình đó người dùng có thể thấy được bản chất của dữ liệu và mối quan hệ giữa các dữ liệu. Có các loại mô hình cơ sở dữ liệu như sau:

- Mô hình E-R (Entity – Relationship)
- Mô hình mạng
- Mô hình phân cấp
- Mô hình quan hệ
- Mô hình hướng đối tượng

1.7. QUY TRÌNH THIẾT KẾ CƠ SỞ DỮ LIỆU

Quá trình thiết kế một cơ sở dữ liệu cho một hệ cơ sở dữ liệu có thể được thực hiện bởi sơ đồ sau:


Thế giới thực
Mô hình mức khái niệm
Mô hình E-R
Mô hình UML
Mô hình mức logic
Mô hình mạng
Mô hình phân cấp
Mô hình quan hệ
Mô hình hướng đối tượng
Mô hình mức vật lý
Chọn hệ quản trị CSDL để thực hiện việc thiết kế.

Chương 2
MÔ HÌNH THỰC THỂ- MỐI QUAN HỆ
(Mô hình E-R)

2.1. GIỚI THIỆU

Mô hình E-R được đề xuất bởi P. Chen (1976). Đây là một mô hình khái niệm dựa vào việc nhận thức thế giới thực thông qua tập các đối tượng được gọi là các thực thể và các mối quan hệ giữa các đối tượng này.

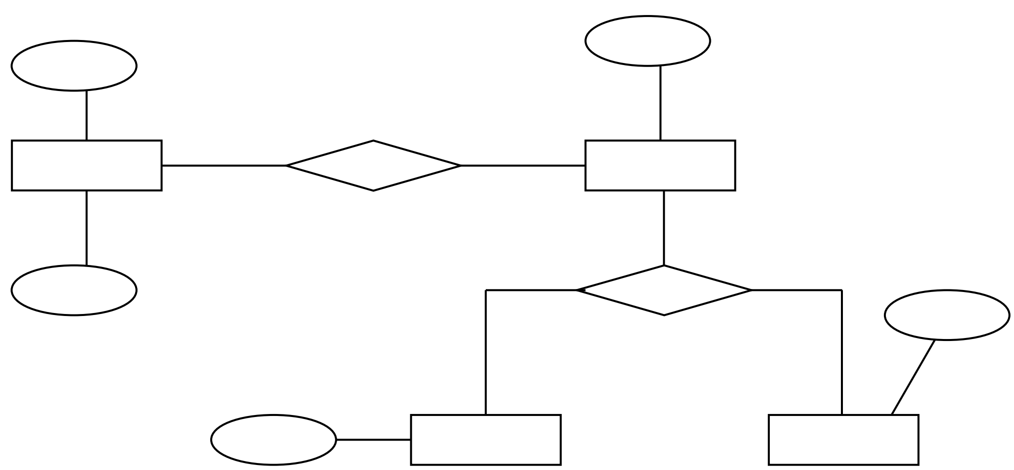
Thực thể (entity) là một vật thể tồn tại và phân biệt được với các vật thể khác. Chẳng hạn, mỗi cán bộ giảng dạy trong trường đại học là một thực thể, mỗi sinh viên là một thực thể, mỗi môn học là một thực thể,...

Một nhóm bao gồm các thực thể "tương tự" nhau tạo thành một *tập thực thể*.

Chẳng hạn, tập hợp các sinh viên trong khoa Công nghệ Thông tin là một tập thực thể, trong đó mỗi sinh viên là một thực thể; tập hợp các môn học cho sinh viên ngành Tin học là tập thực thể, trong đó mỗi môn học là một thực thể.

Để xác định một tập thực thể cần phải thiết lập một số hữu hạn các tính chất đặc trưng của tất cả các thực thể trong tập thực thể đó, gọi là các thuộc tính. Lựa chọn các tập thực thể là một bước quan trọng trong việc xây dựng sơ đồ về mối quan hệ thực thể phản ánh thông tin quản lý cho một thế giới thực nào đó.

Mô hình E-R thường được biểu diễn dưới dạng sơ đồ (sơ đồ E – R).



Trong đó:

- Các hình chữ nhật biểu diễn các tập thực thể

- Hình thoi biểu diễn mối quan hệ, chúng được liên kết với các tập thực thể bằng các cạnh vô hướng hoặc có hướng.

- Hình Oval biểu diễn thuộc tính, chúng được liên kết với các tập thực thể bằng các cạnh vô hướng.

Trong thực tế, có nhiều hệ thống thông tin được thiết kế xuất phát từ mô hình E – R.

Dựa vào mô hình E-R, các mô hình E-R mở rộng cũng đã được đề xuất nhằm biểu diễn sự phong phú và phức tạp của thế giới thực (mô hình UML là một ví dụ).

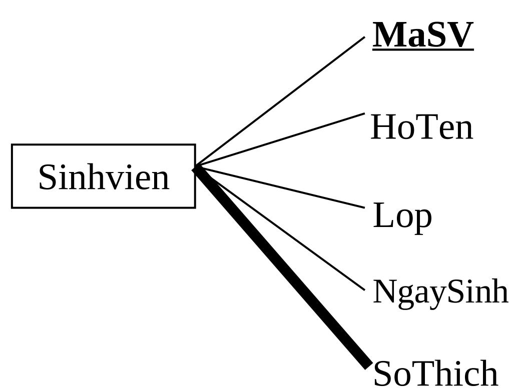
2.2. CÁC THÀNH PHẦN CƠ BẢN

2.2.1. Tập thực thể

Mỗi tập thực thể có một tập các tính chất đặc trưng, mỗi tính chất đặc trưng này được đặt bởi một tên gọi là **thuộc tính** (attribution) của tập thực thể. Thông tin về mỗi thực thể trong tập thực thể được xác định bởi một bộ giá trị các thuộc tính. Ứng với mỗi thuộc tính có một tập các giá trị cho thuộc tính đó gọi là miền. Lựa chọn tập các thuộc tính cho các tập thực thể là một bước quan trọng trong thiết kế một sơ đồ CSDL quan niệm.

Một thuộc tính hay một tập tối thiểu các thuộc tính mà các giá trị của nó xác định *duy nhất* một thực thể trong một tập thực thể gọi là **khóa** (key) cho tập thực thể đó.

Ví dụ: Để quản lý tập các sinh viên trong một trường đại học, người ta có thể sử dụng tập thực thể sinh viên bao gồm một số các thuộc tính sau:



Ta có một thể hiện của tập thực thể Sinhvien, chẳng hạn:
(K25-15, Lê Văn Nam, Tin_K30B, 15/12/83, {âm nhạc, bóng đá})

Lưu ý (trong việc thiết kế các tập thực thể):
- Thứ nhất, phát hiện một tập thực thể bằng cách phát hiện tập các đối tượng mà ta cần quản lý (có 2 phần tử trở lên). Từ đó xác định các

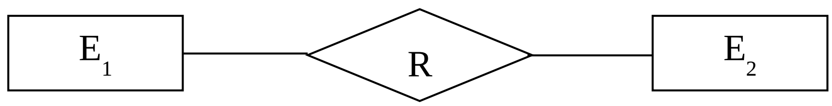
thông tin cần quản lý cho tập thực thể đó (các thông tin đó chính là các thuộc tính).

- Thứ hai, cần có thuộc tính khoá cho mỗi tập thực thể.
- Thứ ba, không sử dụng thuộc tính mà dữ liệu của nó được lấy từ thuộc tính của tập thực thể khác.

2.2.2. Mối quan hệ giữa các tập thực thể

Một mối quan hệ trong mô hình E - R biểu thị quan hệ giữa các thực thể của các tập thực thể.

Mối quan hệ R giữa hai tập thực thể E_1 và E_2 được biểu diễn trong sơ đồ E - R như sau:

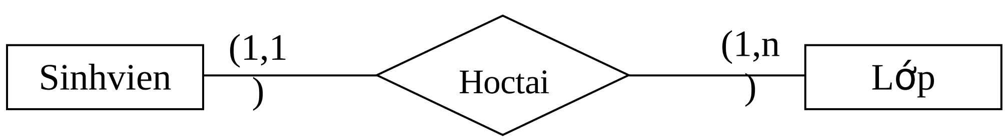


Ta có thể diễn tả khái niệm mối quan hệ giữa các tập thực thể một cách hình thức như sau**:** Mối quan hệ R trên các tập thực thể $E_1, E_2, ..., E_n$ là một tập con của tích Descartes $E_1 \times E_2 \times ... \times E_n$. Vì vậy, một thể hiện của R là một bộ n thành phần $(e_1, e_2, ..., e_n)$, gọi tắt là n-bộ, trong đó $e_i \in E_i$ $(i = 1..n)$. Nếu n-bộ $(e_1, e_2, ..., e_n)$ là một thể hiện của R thì ta nói rằng $e_1, e_2, ..., e_n$ có mối quan hệ R với nhau.

Lưu ý: Một mối quan hệ phải tương ứng với một ngữ nghĩa xác định. Ví dụ xét hai tập thực thể: Sinhvien (tập các thực thể sinh viên) và Lớp (tập các thực thể lớp học), xét mối quan hệ Hoctai có ngữ nghĩa như sau:

(s,l) Hoctai với s thuộc Sinhvien, l thuộc Lớp ⇔ sinh viên s học tại lớp l.

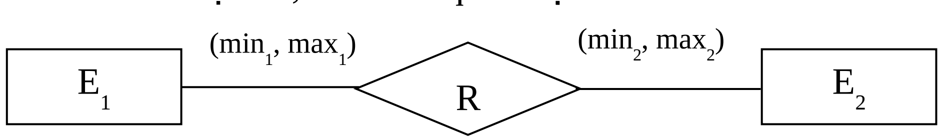
Sơ đồ biểu diễn mối quan hệ bằng E - R.



Lưu ý:

- **Ràng buộc về các bản số của một mối quan hệ:** trên mỗi cung nối giữa hình chữ nhật và hình thoi phải có cặp (min, max) được gọi là bản số của mối quan hệ. Nếu min/max lớn hơn 1, ta có thể viết tắt là n.

Để xác định một mối quan hệ là thuộc loại nào (1-1, 1-n, hay n-n), ta cần dựa vào bản số. Cụ thể, cho mối quan hệ R như sau:



Ràng buộc này chỉ ra rằng :

- Mỗi phần tử (thực thể) của E_1 có mối quan hệ R với ít nhất là min_1 phần tử của E_2, và nhiều nhất là max_1 phần tử của E_2.
- Tương tự, mỗi phần tử của E_2 có mối quan hệ R với ít nhất là min_2 phần tử của E_1, và nhiều nhất là max_2 phần tử của E_1.

Khi đó, mối quan hệ R giữa E_1 và E_2 là mối quan hệ: **max_2 - max_1**

- **Các thuộc tính của một mối quan hệ:** một mối quan hệ cũng có thể có các thuộc tính của riêng nó (đặc biệt là các mối quan hệ n - n). Các thuộc tính của một mối quan hệ quy ước rằng chỉ là các thuộc tính đơn trị.

Trong trường hợp mối quan hệ R có thuộc tính, nếu R là mối quan hệ 1-1 thì ta có thể chuyển thuộc tính này thành thuộc tính của một trong hai tập thực thể tham gia, và nếu R là mối quan hệ 1-n thì chuyển thuộc tính này thành thuộc tính của tập thực thể tương ứng với phía nhiều.

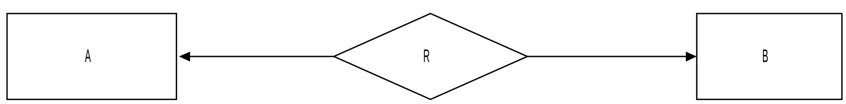
2.3. PHÂN LOẠI MỐI QUAN HỆ

2.3.1. Mối quan hệ nhị nguyên.

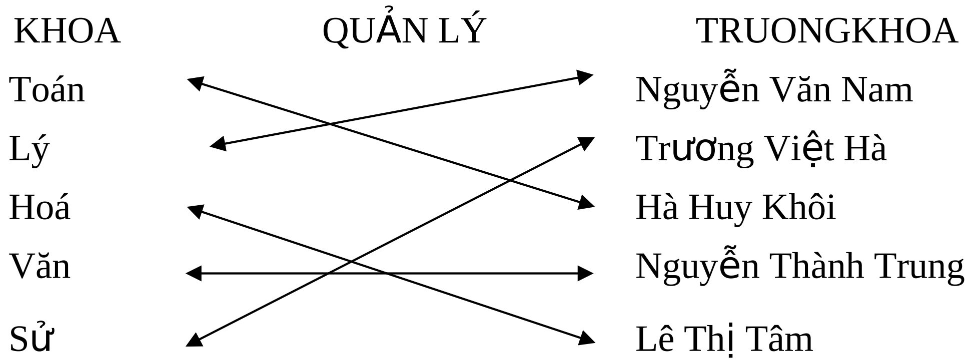
Đây là mối quan hệ giữa hai tập thực thể, bao gồm:

Quan hệ một - một: Mối quan hệ R giữa tập thực thể A và tập thực thể B được gọi là mối quan hệ một-một (hay 1-1) nếu mỗi thực thể của A có quan hệ R với duy nhất một thực thể của B và ngược lại mỗi thực thể của B có quan hệ R duy nhất với một thực thể của A.

Nếu R là mối quan hệ một - một giữa A và B thì có các cạnh định hướng từ hình thoi nhãn R đến các hình chữ nhật nhãn A và B.

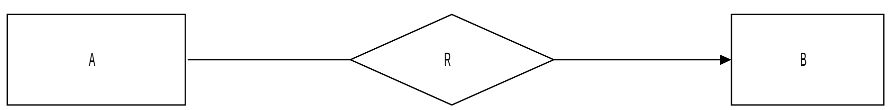


Ví dụ: Giả sử chúng ta đang xét hai tập thực thể sau: tập thực thể KHOA, bao gồm tất cả các khoa trong một trường đại học nào đó và tập thực thể TRUONGKHOA, bao gồm tất cả các trưởng khoa trong trường này. Mối quan hệ QUANLY giữa các tập thực thể KHOA và TRUONGKHOA theo nghĩa trưởng khoa X có quan hệ QUANLY với khoa Y nếu X là trưởng khoa của khoa Y. Rõ ràng rằng mối quan hệ này là một-một, vì rằng mỗi khoa có một trưởng khoa và mỗi trưởng khoa quản lý một khoa duy nhất. Ta có thể thấy mối quan hệ này một cách trực quan bởi hình sau:

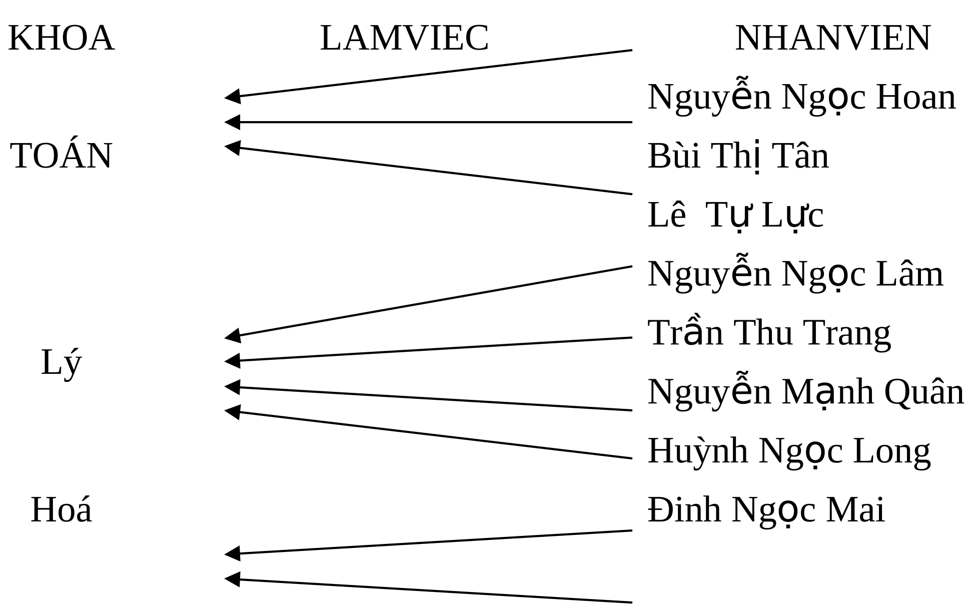


Quan hệ nhiều - một: Giả sử R là mối quan hệ giữa hai tập thực thể E_1 và E_2. Nếu một thực thể E_2 liên kết với 0 hoặc nhiều thực thể của E_1, và mỗi thực thể trong E_1 liên kết với nhiều nhất một thực thể của tập thực thể E_2 thì nói rằng R là mối quan hệ nhiều - một từ E_1 vào E_2.

Nếu R là mối quan hệ nhiều - một từ A vào B thì ta vẽ một cạnh định hướng từ hình thoi nhãn R vào hình chữ nhật nhãn B và một cạnh không định hướng từ hình thoi nhãn R vào hình chữ nhật nhãn A.

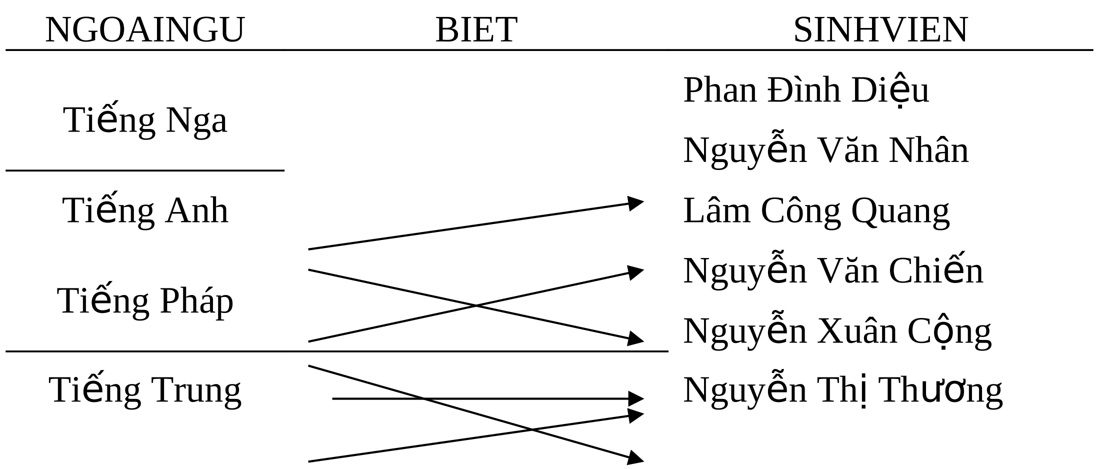


Ví dụ: Giả sử KHOA là tập thực thể bao gồm tất cả các khoa còn NHANVIEN là tập thể tất cả các cán bộ giảng dạy trong một trường đại học. Mối quan hệ LAMVIEC giữa KHOA và NHANVIEN theo nghĩa " nhân viên x làm việc trong khoa Y" là quan hệ nhiều - một từ tập thực thể NHANVIEN vào tập thực thể KHOA . Mối quan hệ này có thể được diễn tả bởi hình sau:



Quan hệ nhiều - nhiều: Cho hai tập thực thể E_1, E_2 và mối quan hệ R giữa chúng. Nếu một thực thể của E_1 có quan hệ R với 0 hoặc nhiều thực thể của E_2 và ngược lại, mỗi thực thể của E_2 có quan hệ R với 0 hoặc nhiều thực thể của E_1 thì ta nói rằng R là mối quan hệ nhiều-nhiều giữa E_1 và E_2 .

Ví dụ: Giả sử SINHVIÊN là tập thực thể các sinh viên cần khảo sát trong một trường đại học, còn NGOAINGU là tập thực thể các ngoại ngữ mà các sinh viên đã được học, thì mối quan hệ **BIET** với nghĩa "sinh viên x biết ngoại ngữ y" là một quan hệ nhiều - nhiều, vì rằng một sinh viên có thể biết nhiều ngoại ngữ và mỗi một ngoại ngữ có thể được biết bởi nhiều sinh viên.

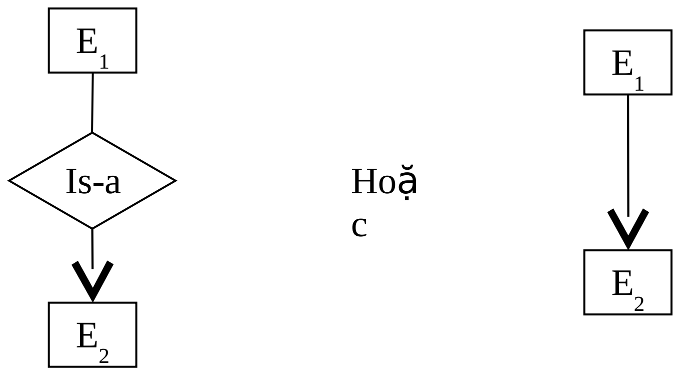


2.3.2. Mối quan hệ Is-a *(mối quan hệ kế thừa).*

Cho hai tập thực thể A và B chúng ta nói rằng A có mối quan hệ I-sa với B, ký hiệu là A Isa B, nếu mỗi thực thể của A là một thực thể của B.

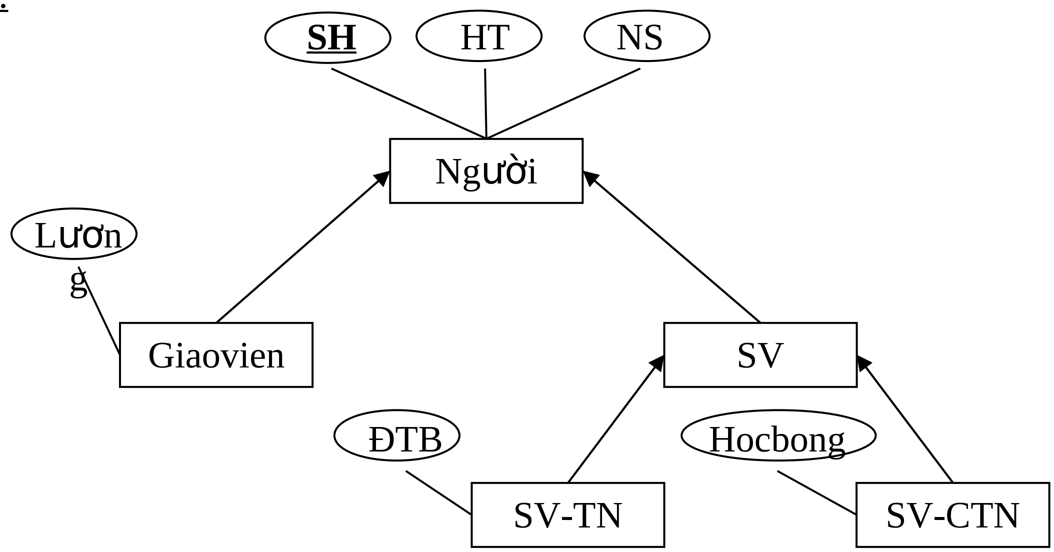
Như vậy A bao gồm các thuộc tính của B đồng thời thêm các thuộc tính khác. Chẳng hạn, B là tập thực thể các nhân viên của khoa Công nghệ Thông tin, A là tập thực thể các cầu thủ bóng đá trong khoa thì A Isa B, vì rằng một cầu thủ trong khoa cũng là một nhân viên của khoa. Ngoài những thuộc tính chung với tập thực thể A như họ tên, tuổi, học hàm, học vị, địa chỉ B còn thêm một số thuộc tính khác chẳng hạn như vị trí cầu thủ sẽ tham gia trong sân.

Mối quan hệ “Is-a” là trường hợp đặc biệt của mối quan hệ nhị nguyên 1-1. Ta có thể biểu diễn nó trong mô hình E-R như sau:



Nhận xét: Nếu E_1 Is-a E_2 thì mọi thực thể thuộc E_1 thì cũng thuộc E_2 và mọi thuộc tính nào có trong E_2 thì cũng có trong E_1.

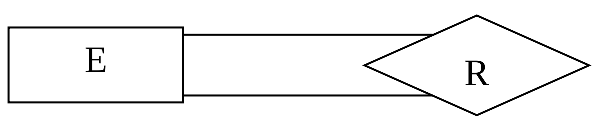
Ví dụ:



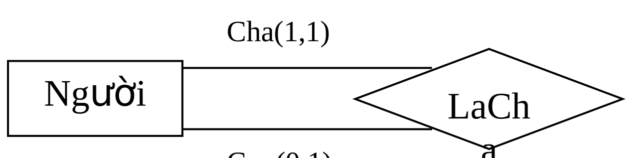
2.3.3 Mối quan hệ phản xạ *(mối quan hệ đệ quy).*

Là mối quan hệ giữa các thực thể của cùng một tập thực thể.

Biểu diễn



Ví dụ:



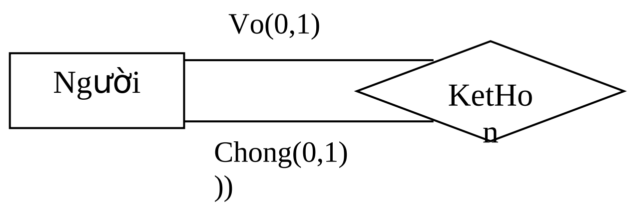
Ngữ nghĩa: $(n_1, n_2) \in$ LaCha $\Leftrightarrow$ n_1 là bố của n_2.

Lưu ý:

Đối với mối quan hệ phản xạ, chúng ta cần xác định rõ **tên vai trò** cho mỗi bản số có trong mối quan hệ này.

Các mối quan hệ phản xạ 1-1, 1-n, hoặc n-n cũng tương tự như mối quan hệ nhị nguyên 1-1, 1-n, n-n.

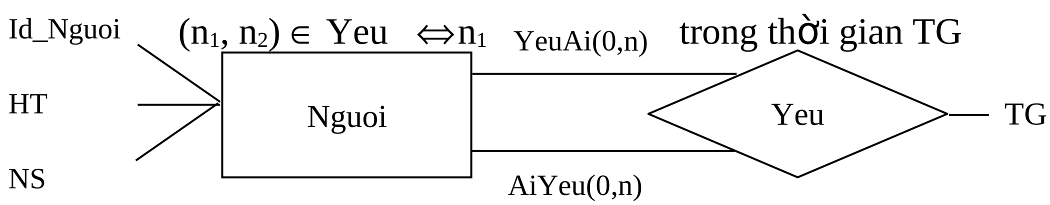
Ví dụ: (Mối quan hệ phản xạ 1-1):



Ngữ nghĩa: $(n_1, n_2) \in$ KetHon $\Leftrightarrow n_1$ hiện là chồng của n_2

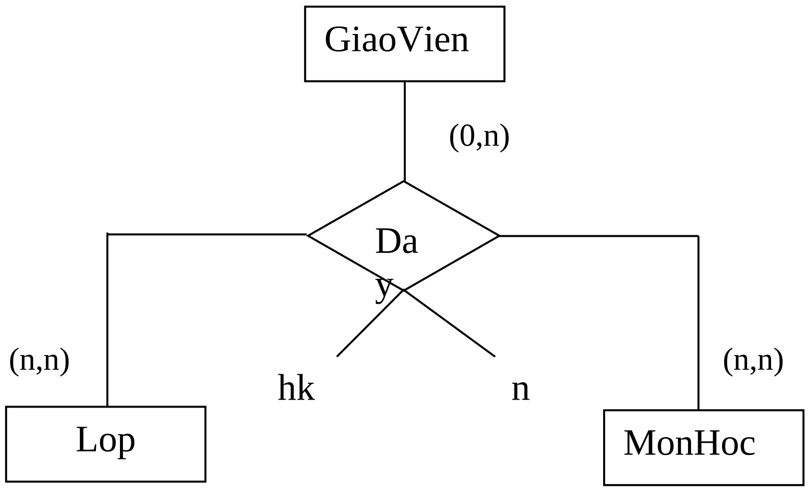
Ví dụ: (Mối quan hệ phản xạ n-n):

$(n_1, n_2) \in$ Yeu $\Leftrightarrow n_1$ trong thời gian TG



2.3.4. Mối quan hệ đa nguyên

Là mối quan hệ giữa 3 tập thực thể trở lên.



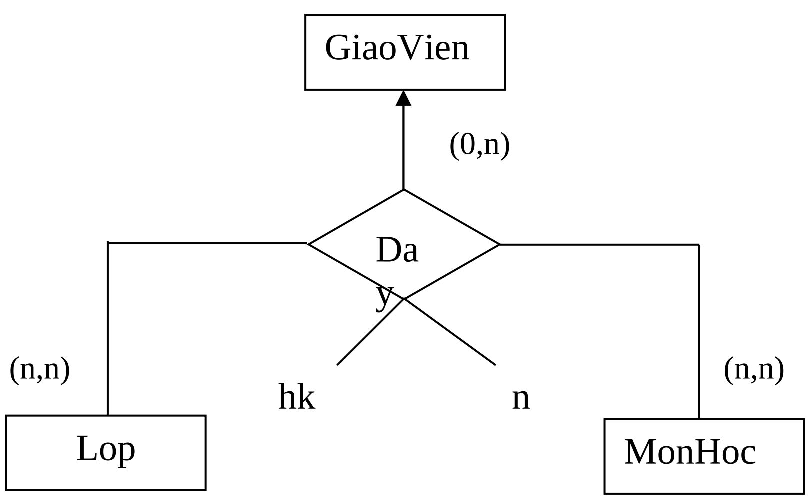
Ngữ nghĩa: (g, l, m) ∈Day ⇔giáo viên g dạy môn m cho lớp l vào học kỳ là hk của năm học n.

Lưu ý :

Ràng buộc hàm của mối quan hệ đa nguyên: Trong mối quan hệ đa nguyên, ngoài ràng buộc về bản số còn có "ràng buộc hàm". Ví dụ mối quan hệ Day nêu trên có ràng buộc hàm như sau:

$$\{Lop, MonHoc\} \rightarrow \{GiaoVien\}$$

(đọc là: Lop và MonHoc xác định GiaoVien). Điều này có nghĩa rằng: nếu biết trước lớp học l và môn học m thì xác định tối đa một giáo viên g của mối quan hệ. Khi đó, trên mô hình E-R sử dụng cung mũi tên như sau:



Chương 3
MÔ HÌNH QUAN HỆ

3.1. QUAN HỆ - LƯỢC ĐỒ QUAN HỆ

3.1.1. Quan hệ (Relation)

Quan hệ là một bảng mà không có dòng nào giống nhau, mỗi dòng của bảng được gọi là bộ (tuple) và mỗi cột của bảng được ký hiệu bằng một tên gọi là thuộc tính của quan hệ.

Ví dụ:

r =

A	B	C
a	b	c
a	d	b
b	c	a
b	a	a

Như vậy, ta có thể xem quan hệ như là một tập các bộ.

<u>Ví dụ:</u> t = (a, d, b) r

Vì vậy, người ta còn định nghĩa: quan hệ r trên tập thuộc tính $A_1, A_2, \ldots, A_n$:

$$r \subseteq Dom(A_1) \times Dom(A_2) \times \ldots \times Dom(A_n)$$

Trong đó, $Dom(A_i)$ là tập các giá trị có thể có của A_i (miền trị của A_i), với i = 1, 2, ..., n.

3.1.2. Lược đồ quan hệ (Relational Schema)

Lược đồ quan hệ là một cặp có thứ tự: R = <U , SC>

Trong đó:

U là tập hữu hạn các thuộc tính của lược đồ quan hệ R.

SC là tập các ràng buộc của lược đồ quan hệ R.

<u>Ví dụ:</u> Sinhvien = <U, SC>
U = {MaSV, Hoten, Ngaysinh}
SC: MaSV xác định duy nhất (khoá của Sinhvien).

Lưu ý: (Quan hệ r trên lược đồ quan hệ R)

Cho lược đồ quan hệ R = <U, SC>, khi đó quan hệ r được gọi là quan hệ trên R nếu r có tập thuộc tính U và thỏa mãn các ràng buộc trong SC

Ta xem: R = { r | r có tập thuộc tính U và thỏa các ràng buộc trong SC}

Vài thuật ngữ thông dụng:

Một tập các lược đồ quan hệ trong một hệ thống thông tin thì được gọi là một ***mô hình cơ sở dữ liệu quan hệ*** (có thể được gọi tắt là ***mô hình quan hệ***, hay: ***lược đồ cơ sở dữ liệu quan hệ***).

Tập hợp các quan hệ (hiện hành) của các lược đồ quan hệ trong một mô hình quan hệ thì được gọi là *cơ sở dữ liệu quan hệ*.

3.2. KHOÁ CỦA QUAN HỆ

Định nghĩa: Cho quan hệ r của lược đồ quan hệ R với tập thuộc tính U = $\{A_1, A_2, \ldots, A_n\}$, gọi tắt là lược đồ $R(A_1, A_2, \ldots, A_n)$.

Tập $X \subseteq U$ (X : tập thuộc tính) được gọi là ***khoá*** của quan hệ r nếu nó thỏa mãn cả 2 điều kiện:

i) Với mọi bộ $t \in r$ đều có giá trị khác nhau trên X, khi đó X được gọi là ***siêu khoá*** của r.

ii) $\nexists X' \subset X$ (tập con thực sự của X): X' là siêu khoá của r.

Ví dụ:

r =

A	B	C
a	b	c
b	b	a
c	a	a

Suy ra: r có hai khoá: {A}, {B, C}, viết tắt là A và BC.

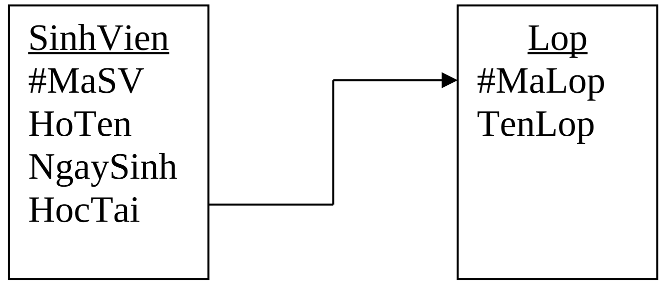
Lưu ý:

- X được gọi là *khoá của lược đồ quan hệ* R nếu X là khoá của mọi quan hệ r trên lược đồ quan hệ R.
- **Khoá chính (Primary key) của một lược đồ quan hệ**: Một lược đồ quan hệ có thể có nhiều khóa. Trong số đó, phải có đúng một khóa chính do người thiết kế cơ sở dữ liệu quy ước. Khóa chính của một lược đồ quan hệ thường được sử dụng cho việc tham chiếu dữ liệu. Trong sơ đồ biểu diễn một mô hình quan hệ, để biểu diễn ràng buộc khóa chính, ta sử dụng ký hiệu # ở ngay trước tên các thuộc tính của khóa chính
- **Khóa ngoại (foreign key) của một lược đồ quan hệ:** Cho 2 lược đồ quan hệ $R_1 = \langle U_1, SC_1 \rangle$ và $R_2 = \langle U_2, SC_2 \rangle$. Gọi $PK \subseteq U_1$ là khoá chính của R_1. Xét $FK \subseteq U_2$. Khi đó, ta gọi FK *là khoá ngoại của lược đồ quan hệ* R_2 *tham chiếu đến* R_1 nếu cơ sở dữ liệu luôn thỏa mãn 2 điều kiện sau:

  i. Miền trị của FK là trùng với miền trị của PK.

  ii. Giá trị của FK hoặc là NULL hoặc phải bằng một giá trị hiện có nào đó của PK.

Ví dụ: Hình vẽ sau biểu diễn ràng buộc khóa ngoài: {HocTai} là khóa ngoài của lược đồ quan hệ SinhVien tham chiếu đến lược đồ quan hệ Lop.



3.3. CHUYỂN ĐỔI MÔ HÌNH E-R SANG MÔ HÌNH QUAN HỆ

Vào: Sơ đồ E-R

Ra: Tập các lược đồ quan hệ (DB).

Quy ước về một số ký hiệu:

- ER: mô hình ER (giả thiết).
- DB: lược đồ cơ sở dữ liệu quan hệ cần tạo lập.
- U_R: tập tất cả các thuộc tính của lược đồ quan hệ R.
- Ω_E: tập tất cả các thuộc tính đơn vị của tập thực thể E.
- Ω_R: tập tất cả các thuộc tính của mối quan hệ R.
- PK_R: khoá chính của lược đồ quan hệ R (Primary Key).
- K_E: tập các thuộc tính khoá của tập thực thể E.
- min(E;R), max(E;R): các chỉ số tối thiểu & cực đại của bảng số trên cung nối tập thực thể F với mối quan hệ R.
- FK_R: tập tất cả các khoá ngoại của lược đồ quan hệ R.

Để chỉ FK là một khoá ngoại của lược đồ quan hệ R (tức là $FK \in FK_R$) tham chiếu đến khoá chính của quan hệ R' ta sử dụng ký hiệu: $FK \cong PK_{R'}$. Tên các thuộc tính có trong FK vẫn có thể khác với tên các thuộc tính có trong $PK_{R'}$ nhưng FK cần thỏa mãn đồng thời 2 điều kiện sau:

- Các thuộc tính trong FK phải có cùng miền trị với các thuộc tính tương ứng trong $PK_{R'}$.
- Giá trị của FK tại một bộ t chỉ có thể là NULL hoặc bằng giá trị $PK_{R'}$ tại một bộ t' nào đó $\in$ R'.

Các điều kiện trên của khoá ngoại FK đặc tả một ràng buộc toàn vẹn tham chiếu giữa hai quan hệ R & R'.

Lưu ý rằng, để chỉ ràng buộc toàn vẹn tham chiếu này đồng thời tên các thuộc tính có trong FK phải trùng tên với các thuộc tính tương ứng có trong $PK_{R'}$, ta sẽ ký hiệu: $FK \cong PK_{R'}$.

Khi đó, thuật toán chuyển đổi từ mô hình ER sang mô hình quan hệ lần lượt trải qua các bước sau:

Bước 1: Chuyển đổi các tập thực thể thành các lược đồ quan hệ.

Mỗi tập thực thể E được chuyển thành lược đồ quan hệ R(E) có cùng tên và cùng tập thuộc tính. Thuộc tính khoá của E chuyển thành khoá chính của R(E).

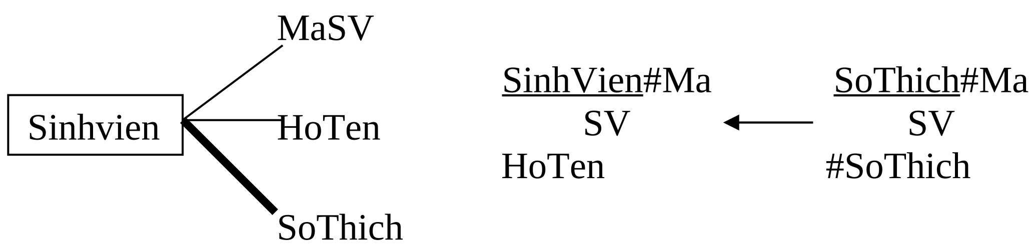
Lưu ý: Chuyển đổi thuộc tính đa trị

Nếu E có thuộc tính đa trị A thì trong lược đồ quan hệ, chúng ta phải tạo thêm lược đồ quan hệ mới để biểu diễn thuộc tính đa trị này.

$U_{R(A)} = \{PK_{R(E)}, A\}$

R(A) có khoá ngoại la $PK_{R(E)}$ tham chiếu đến $PK_{R(E)}$ cua R(E).

Lấy ví dụ thuộc tính sở thích của lược đồ quan hệ Sinhvien.



Chẳng hạn, ta có các quan hệ tương ứng với dữ liệu như sau:

SinhVien

MaSV	HoTen
1	Lê Văn A
2	Lê Văn B
3	Lê Văn C

SoThich

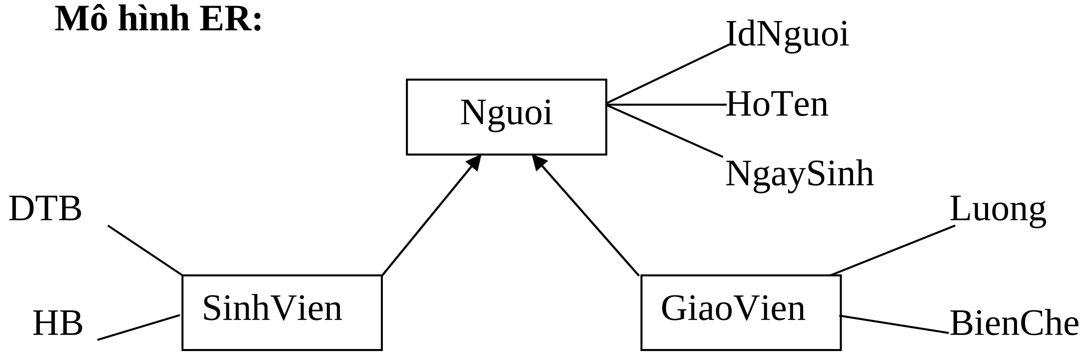
MaSV	SoThich
1	Phim
1	Bóng Đá
1	Nội Trợ
2	Phim
2	Bóng Đá

Chuyển đổi mối quan hệ Is – a (mối quan hệ kế thừa)

Để chuyển đổi mối quan hệ “Is - s” có hai cách thực hiện.

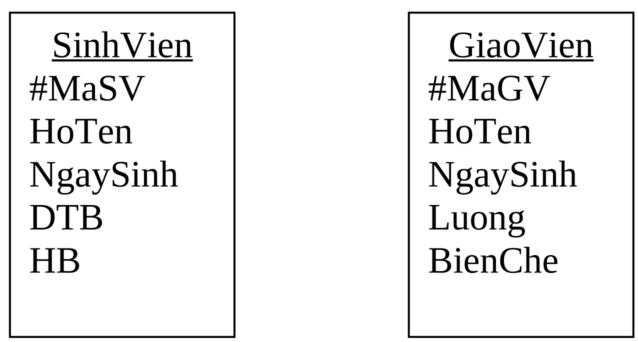
Ví dụ:

Mô hình ER:



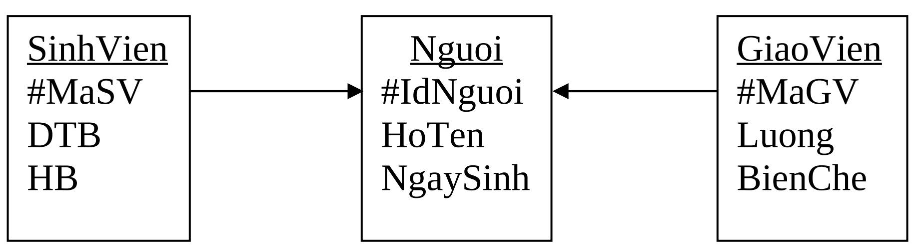
Kết quả chuyển đổi:

Cách 1: (Không sử dụng lược đồ quan hệ biểu diễn lớp cha)



Trong đó: MaSV, MaGV chính là IdNguoi

Cách 2: (Bổ sung khoá ngoại cho các lược đồ quan hệ biểu diễn lớp con)

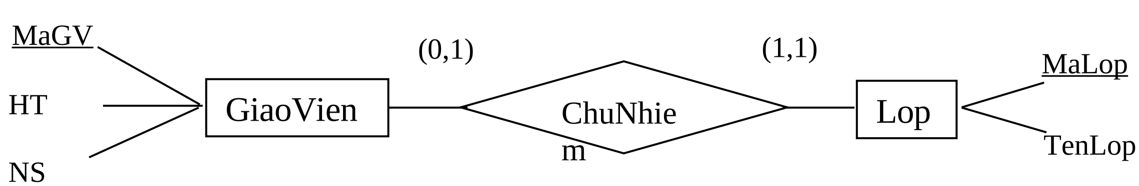


Bước 2: Chuyển đổi các mối quan hệ nhị nguyên 1 – 1.

Phương pháp: Bổ sung khoá ngoại cho một trong hai lược đồ quan hệ

Ví dụ:

Mô hình ER



Ngữ nghĩa: (g, l)∈ChuNhiem ⇔ Giáo viên g "hiện" là giáo viên chủ nhiệm của lớp l.

Kết quả chuyển đổi

<u>GiaoVien</u>#MaGV
HT
NS

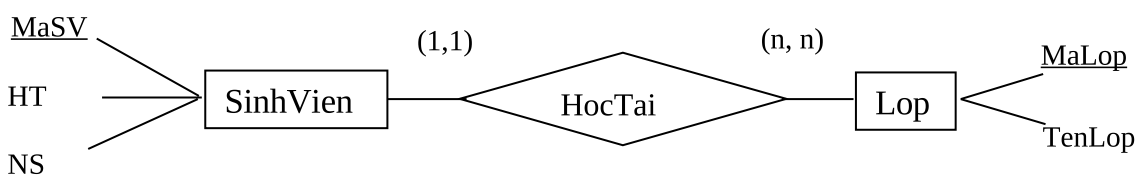
<u>Lop</u>#MaLop
TenLop
MaGVCN

Bước 3: Chuyển đổi các mối quan hệ nhị nguyên 1 – n (một - nhiều)

Phương pháp: Bổ sung khoá ngoại cho lược đồ quan hệ tương ứng với "phía nhiều".

Ví dụ:

Mô hình E-R



Kết quả chuyển đổi

SinhVien#MaSV
HT
NS
MaLop

Lop#MaLop
TenLop

Bước 4: Chuyển đổi các mối quan hệ nhị nguyên n-n (nhiều – nhiều)

Phương pháp: Xét mối quan hệ R giữa E_1 và E_2 là mối quan hệ n-n. Khi đó, ta cần tạo thêm một lược đồ quan hệ mới S để biểu diễn mối quan hệ R (đặt cùng tên với mối quan hệ). Trong đó, tập thuộc tính được xác định như sau: $U_S = PK_{R(E1)} \cup PK_{R(E2)} \cup \Omega$ với Ω là tập thuộc tính của mối quan hệ.

Các ràng buộc:

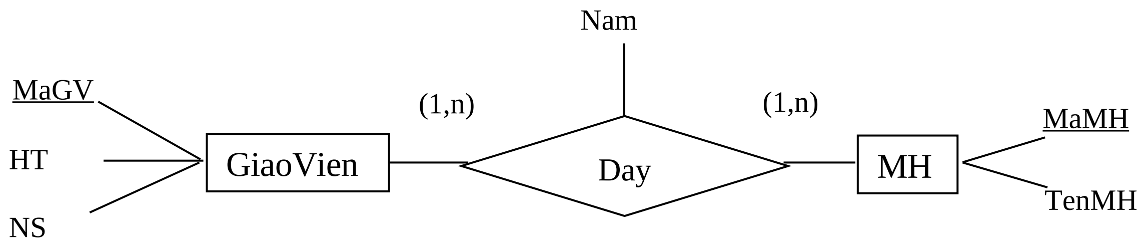
Khoá chính: $PK_S = PK_{R(E1)} \cup PK_{R(E2)}$

Khoá ngoại: S có hai khoá ngoại:

- $PK_{R(E1)}$ của S tham chiếu đến $R(E_1)$,
- $PK_{R(E2)}$ của S tham chiếu đến $R(E_2)$.

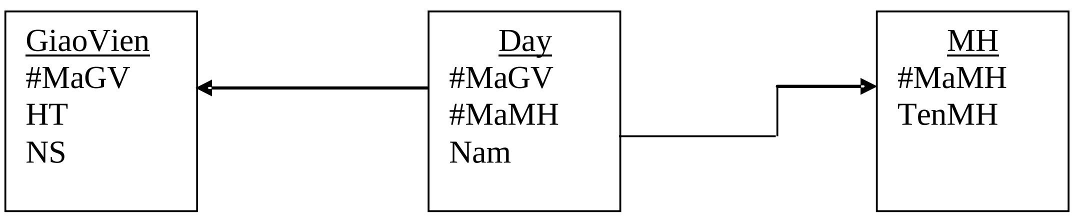
Ví dụ:

Mô hình E-R



Ngữ nghĩa: (g, m)∈Day ⇔Giáo viên g dạy môn học m bắt đầu từ năm Nam.

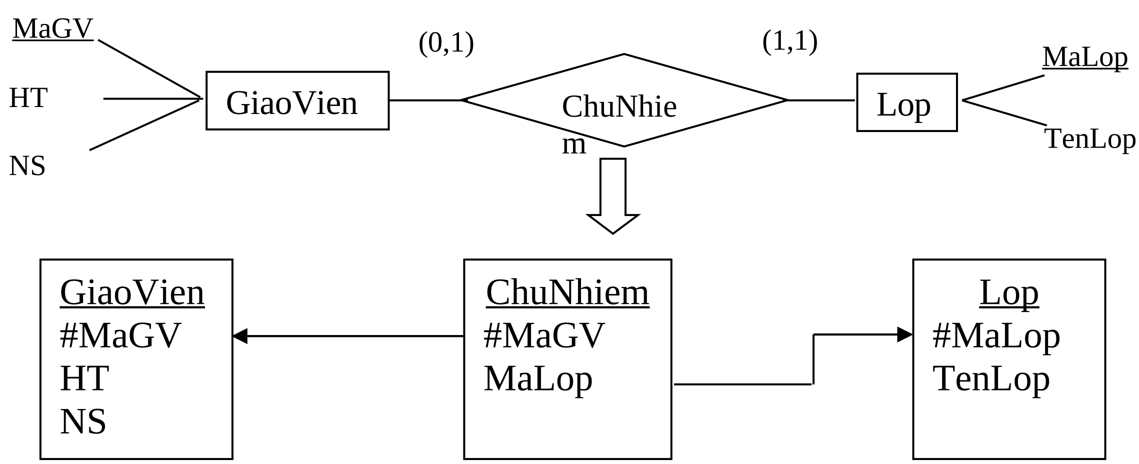
Kết quả chuyển đổi:



Lưu ý:

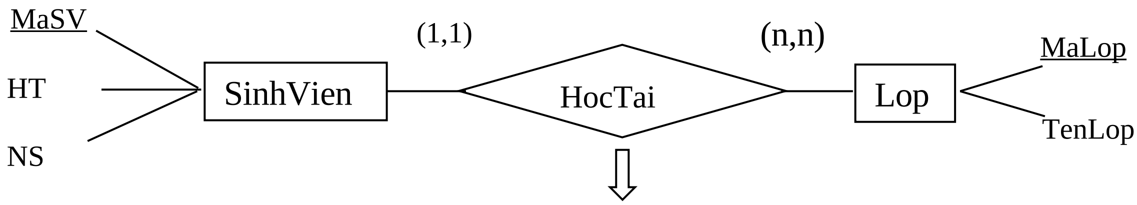
Đối với mối quan hệ 1-1, 1-n ta cũng có thể chuyển đổi tương tự như việc chuyển đổi đối với mối quan hệ n-n. Việc chuyển đổi chỉ khác về ràng buộc khoá chính mà thôi.

Mối quan hệ 1-1

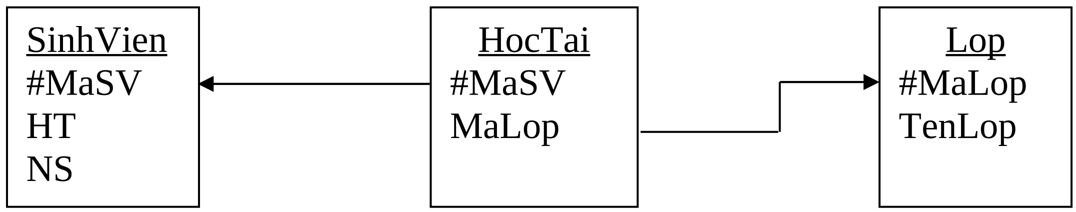


Trong trường hợp này, ta có thể chọn MaLop là khoá chính cũng được.

Mối quan hệ 1-n.



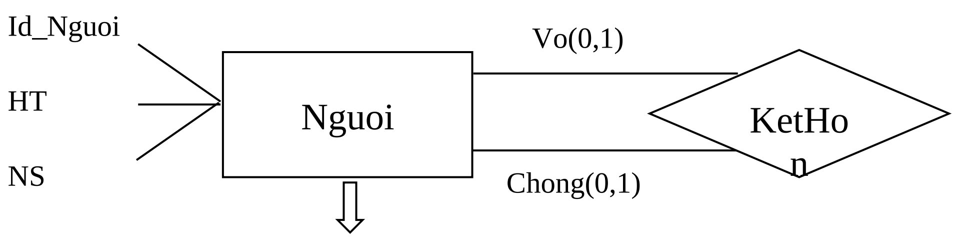
Kết quả chuyển đổi:

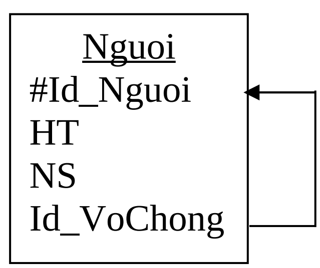


Bước 5: Chuyển đổi mối quan hệ phản xạ.

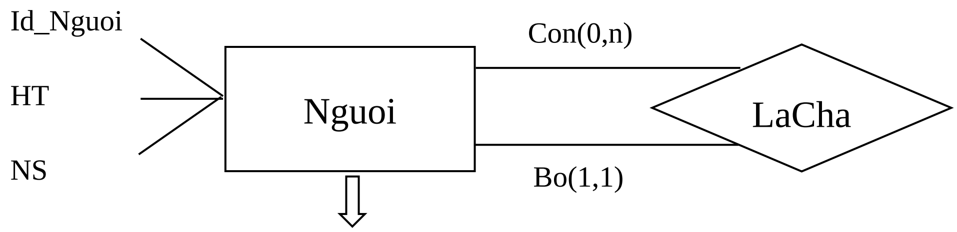
Được thực hiện tương tự như việc chuyển đổi mối quan hệ nhị nguyên 1-1, 1-n, n-n.

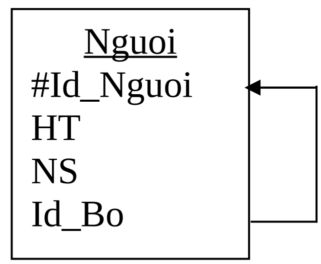
Ví dụ 1 (Mối quan hệ phản xạ 1-1)



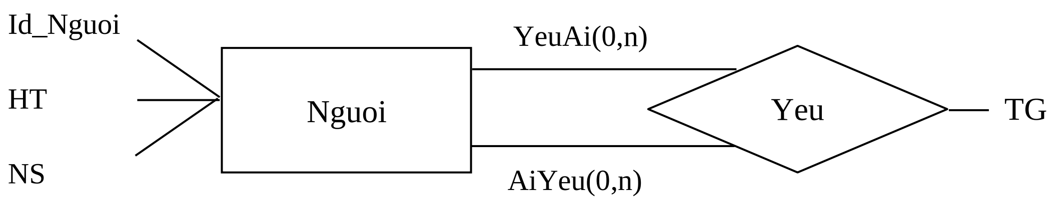


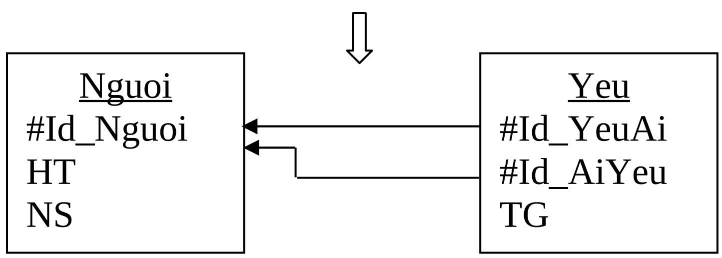
Ví dụ 2 (Mối quan hệ phản xạ 1-n)





Ví dụ 3 (Mối quan hệ phản xạ n-n)



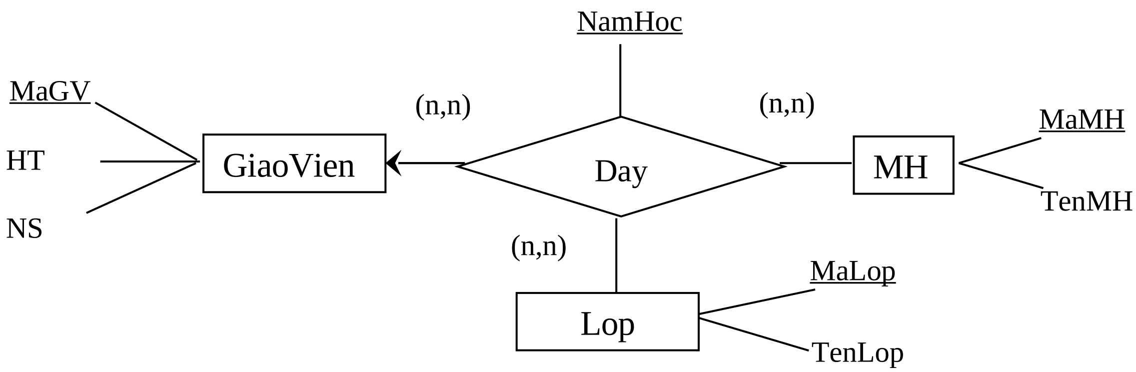


Bước 6: Chuyển đổi mối quan hệ đa nguyên.

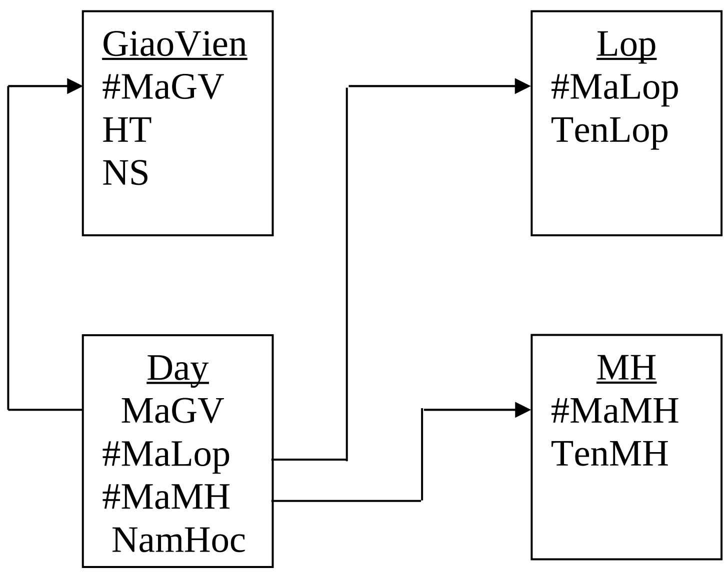
Phương pháp: Tương tự như phương pháp chuyển đổi mối quan hệ nhị nguyên n-n.

Ví dụ:

Mô hình E-R



Kết quả chuyển đổi:



Lưu ý:

Khoá chính của lược đồ quan hệ S là hợp các khoá chính của các lược đồ liên quan. Tuy nhiên, nếu có ràng buộc hàm thì phải loại bỏ khoá chính tương ứng với tập thực thể có mũi tên.

3.4. ĐẠI SỐ QUAN HỆ

3.4.1. Phép hợp (Union):

Hợp của hai quan hệ r và s có cùng một lược đồ, ký hiệu $r \cup s$, được xác định như sau:

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

3.4.2. Phép giao (Intersection):

Giao của hai quan hệ r và s có cùng một lược đồ, ký hiệu $r \cap s$, được xác định như sau:

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

3.4.3. Phép hiệu (Difference):

Hiệu của hai quan hệ r và s có cùng một lược đồ, ký hiệu r - s, được xác định như sau:

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

Lưu ý: Phép giao có thể biểu diễn thông qua phép hiệu: $r \cap s = r - (r - s)$

3.4.4. Tích Descartes (Cartersian Product):

Cho r và s là các quan hệ xác định lần lượt trên tập các thuộc tính $\{A_1, A_2, \ldots, A_n\}$ và $\{B_1, B_2, \ldots, B_m\}$. Tích Descartes của r và s, ký hiệu: $r \times s$, là tập các bộ có (m+n) thành phần sao cho n thành phần đầu là một bộ thuộc r và m thành phần sau là một bộ thuộc s. Hay:

$r \times s = \{(t_1,t_2) \mid t_1 \in r \wedge t_2 \in s\}$

<u>Ví dụ:</u> Xét

r =

A	B
a	b
a	c
b	c

s =

A	B
a	b
a	a

⇨ $r \cup s$ =

A	B
a	b
a	c
b	c
a	a

$r \cap s$ =

A	B
a	b

r - s =

A	B
a	c
b	c

Xét q =

C	D	E
a	b	c
a	c	d

r =

A	B
a	b
a	c
b	c

$r \times q$ =

A	B	C	D	E
a	b	a	b	c
a	b	a	c	d
a	c	a	b	c
a	c	a	c	d
b	c	a	b	c
b	c	a	c	d

Nhận xét: $|r \times s| = |r| \times |s|$

3.4.5. Phép chiếu (Projection):

Cho quan hệ r trên lược đồ quan hệ R = <U_R,SC>, và X $\subseteq U_R$, phép chiếu trên tập X của quan hệ r, ký hiệu: $\prod_X(r)$, được định nghĩa:

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

Ở đây, t[X] để chỉ một bộ t chỉ chứa các thành phần có trong X.

<u>Ví dụ:</u> Xét ví dụ ở trên.

$\prod_A(r) =$

A
a
b

$\prod_{CD}(q) =$

C	D
a	b
a	c

Lưu ý:

$|\prod_X(r)| \leq |r|$

Để đơn giản ta có thể ký hiệu CD thay cho {C, D}

3.4.6. Phép chọn (Selection):

Cho quan hệ r và f là một biểu thức logic mà mỗi bộ trên r có thể thỏa mãn hoặc không. Khi đó, phép chọn trên r các bộ thỏa mãn f, ký hiệu $\delta_f(r)$, được xác định như sau:

$$\delta_f(r) = \{t \mid f(t) \text{ là đúng}\}$$

Ví dụ:

r =

A	B
a	b
a	c
b	c

$\delta_{A='a'}(r) =$

A	B
a	b
a	c

3.4.7. Phép nối (Kết nối – join):

Xét hai quan hệ r và s. Gọi f là một biểu thức logic mà mỗi bộ thuộc tích Descartes r × s có thể thỏa mãn hoặc không. Khi đó, phép nối r và s theo điều kiện f, được ký hiệu: $r \bowtie_f s$, được xác định như sau:

$$r \bowtie_f s = \delta_f(r \times s)$$

<u>Ví dụ:</u>

r =

A	B	C
a	1	1
b	2	1
c	0	2

s =

C	D
1	2
2	1

Với f = r.B ≥ s.D, ta có:

$r \underset{f}{\bowtie} s =$

A	B	r.C	s.C	D
a	1	1	2	1
b	2	1	1	2
b	2	1	2	1

Lưu ý: Nếu r và s có các thuộc tính trùng tên nhau thì phải đổi tên các thuộc tính này trong quan hệ thu được từ $r \underset{f}{\bowtie} s$.

Phép nối tự nhiên: Nếu r và s có các thuộc tính chung và f là một biểu thức logic để chỉ mỗi cặp thuộc tính chung này đều có giá trị bằng nhau, thì ta gọi phép nối này là phép nối tự nhiên và ký hiệu là: $r \bowtie s$.

<u>Ví dụ:</u> Xét ví dụ trên.

$r \bowtie s =$

A	B	C	D
a	1	1	2
b	2	1	2
c	0	2	1

Lưu ý: Quan hệ thu được từ phép nối tự nhiên có thể loại bỏ các thuộc tính lặp.

3.4.8. Phép chia (Division):

Gọi r là quan hệ trên tập thuộc tính $\{A_1, A_2, \ldots, A_n\}$

s là quan hệ trên tập thuộc tính $\{A_{p+1}, A_{p+2}, \ldots, A_n\}$, với $p > 0$.

Khi đó:

$$r \div s = \{(a_1, a_2, \ldots, a_p) \mid \forall (a_{p+1}, a_{p+2}, \ldots, a_n) \in s; (a_1, a_2, \ldots, a_n) \in r\}$$

<u>Ví dụ:</u>

r =

A	B	C	D
a	b	c	d
a	b	e	f
b	c	e	f
e	d	c	d
e	d	e	f
a	b	d	e

s =

C	D
c	d
e	f

$\Rightarrow$ $r \div s =$

A	B
a	b
e	d

Nhận xét:

Phép chia có thể được định nghĩa thông qua phép toán khác. Với r, s lần lượt là quan hệ trên tập thuộc tính Z, X ($X \subseteq Z$).

$r \div s = T_1 - T_2$ (có tập thuộc tính: $Y = Z - X$)

$= T_1 - \prod_Y((s \times T_1) - r)$, với $T_1 = \prod_Y(r)$

Ví dụ: Xét ví dụ trên.

$T_1 = \prod_Y(r) =$

A	B
a	b
b	c
e	d

$T_1 \times s =$

A	B	C	D
a	b	c	d
b	c	c	d
e	d	c	d
a	b	e	f
b	c	e	f
e	d	e	f

$\theta = (T_1 \times s) - r =$

A	B	C	D
b	c	c	d
b	c	e	f

$\Rightarrow \prod_Y(\theta) =$

A	B
b	c

$\Rightarrow r \div s =$

A	B
a	b
e	d

3.4.9. Ứng dụng của đại số quan hệ:

Dựa vào Đại số quan hệ được xét ở trên, người ta có thể xây dựng các biểu thức đại số quan hệ để trả lời các câu truy vấn.

Ví dụ:

Xét lược đồ quan hệ đã trình bày trong mục trước. Xét 1 CSDL hiện hành bao gồm các quan hệ sau: NGUOI, GIAOVIEN, SINHVIEN, LOP, MONHOC, DAY. Sử dụng các phép toán Đại số quan hệ để trả lời các câu hỏi sau:

- ❖ Liệt kê họ tên những sinh viên sinh vào ngày 1/1/1985.

  $\prod_{Hoten} (\delta_{Ngaysinh = \{1/1/1985\}}(Nguoi \bowtie SinhVien))$

  $= \prod_{Hoten} (Sinhvien \bowtie (\delta_{Ngaysinh = \{1/1/1985\}}(Nguoi)))$

- ❖ Cho biết họ tên của những giáo viên có dạy môn CSDL.

  $\prod_{Hoten} (\delta_{IdMon = 'CSDL'}(Nguoi \bowtie Giaovien \bowtie Day))$

- ❖ Tìm họ tên & lương những giáo viên dạy môn học trên 60 tiết cho lớp nào đó có sĩ số > 60.

$$\prod_{Hoten,Luong}(\delta_{(Sotiet>60) \wedge (Siso>60)}(Nguoi \bowtie Giaovien \bowtie Day \bowtie Monhoc \bowtie Lop))$$

$$= \prod_{Hoten,Luong}(Nguoi \bowtie Giaovien \bowtie Day \bowtie \delta_{(Sotiet>60)}(Monhoc) \bowtie \delta_{(Siso>60)}(Lop))$$

Chương 4
MÔ HÌNH HƯỚNG ĐỐI TƯỢNG

4.1. GIỚI THIỆU CHUNG

Mô hình hướng đối tượng (Object-Oriented Model) hình thành vào cuối những năm 1980, phát triển ở Mỹ với nhiều hệ quản trị cơ sở dữ liệu được xây dựng dựa trên nền tảng lý thuyết của mô hình này như: O_2, Object Store, Objective, Orion, Jasmin, ...

Cho đến nay, mô hình này vẫn chưa có chuẩn thống nhất.

4.2. CÁC THÀNH PHẦN CƠ BẢN

4.2.1. Lớp, đối tượng và định danh đối tượng

Mô Hình Hướng Đối Tượng	**Mô Hình E-R**
Lớp	Tập thực thể
Đối tượng	Thực thể
Định danh đối tượng *(mặc định)*	Thuộc tính khoá *(do người sử dụng thiết kế)*

Lớp được hiểu như là một tập các thực thể, hay các đối tượng có cùng các đặc tính và hành vi giống nhau. Các đặc tính này được mô tả như các thuộc tính bên trong một lớp đối tượng. Các hành vi chính là các phương thức (methods) được thực hiện trên mỗi đối tượng của lớp đó.

Mỗi đối tượng trong một lớp được xác định thông qua tên của đối tượng. Người ta sử dụng thuộc tính định danh OID (Object Identifier) để xác định tên duy nhất cho các đối tượng trong mỗi lớp.

4.2.2. Thuộc tính và phương thức

Việc khai báo các thuộc tính thể hiện cấu trúc của lớp được khai báo. Mỗi thuộc tính có thể là thuộc tính đơn trị hoặc thuộc tính đa trị (sử dụng từ khoá “set” để khai bào). Ngoài ra, một thuộc tính có thể là một thuộc tính phức hợp: là thuộc tính được xác định từ tập các thuộc tính khác (sử dụng từ khoá “tuple” để khai báo).

Thuộc tính mối quan hệ là thuộc tính biểu diễn mối quan hệ giữa lớp này với lớp kia và giá trị của nó là OID của lớp kia.

Việc khai báo các phương thức của mỗi đối tượng trong một lớp nhằm phản ánh các hành vi được thực hiện trên mỗi đối tượng thuộc lớp đó.

Mẫu đặc tả cho một lớp có thể được xác định như sau:

Class <tên lớp>

    properties

        {khai báo các thuộc tính}

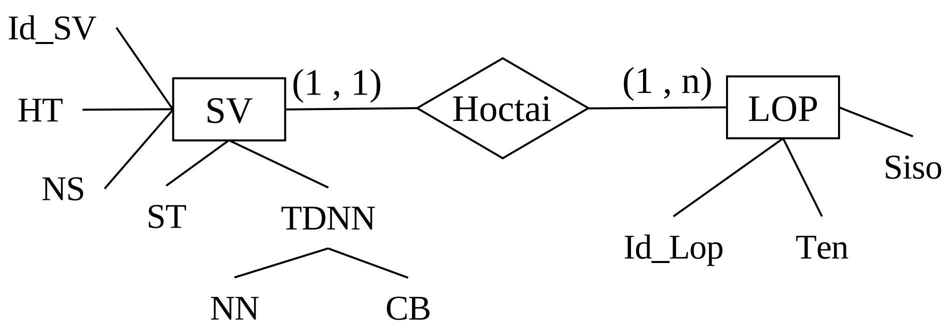
    Operations

        {khai báo các phương thức}

end <tên lớp>.

Ví dụ:

- Mô hình ER mở rộng



⇒Mô hình hướng đối tượng:

```
class  SV
   properties
      Id_SV :     string;
      HT :        string;
      NS :        Date;
      ST :        set(string);
      TDNN :      set( tuple( NN : string; CB : char(1)));
      Hoctai :    LOP;  {→mang giá trị OID của class LOP}
   operations
      ……….
end   SV;
{=====================================}
class  LOP
   properties
      Id_Lop :    string;
      Ten :       string;
      Siso :      integer;
      [Gom :      set(SV); {Inverse SV.Hoctai}]
   operations
      …………
end   LOP;
```

- Minh hoạ:

LOP:

OID	Id_Lop	Ten	Siso	Hoctai
Lop001	A	TinK25A	70	{SV1, SV2}
Lop002	B	TinK25B	80	{SV2, SV4}

SV:

OID	HT	NS	ST	TDNN (NN, CB)	Hoctai
SV1	X	1/1/85	{BB, BD, CN}	{(Anh, C), (Nga, B)}	Lop001
SV2	Y	11/3/84	{BĐ}		Lop001
SV3	Z				Lop002
SV4	T				Lop002

Lưu ý:

Để khai báo thuộc tính ngược của một thuộc tính quan hệ, ta dùng từ khoá Inverse. Ví dụ: [Gom : set(SV); {Inverse SV.Hoctai}]

Điều này có nghĩa Gom là thuộc tính mối quan hệ ngược của thuộc tính Hoctai thuộc lớp SV.

Việc dùng thuộc tính ngược cũng có ưu điểm là thuận tiện trong việc biểu diễn các câu truy vấn. Tuy nhiên nó cũng có nhược điểm đó là gây dư thừa dữ liệu.

4.2.3. Phân cấp lớp và sự kế thừa

Các lớp trong mô hình hướng đối tượng có thể được tổ chức theo một phân cấp lớp, tương tự như mối quan hệ "Is-a" của mô hình E-R. Ta nói rằng lớp C_2 là lớp con của lớp C_1, có nghĩa là tập các đặc tính (các thuộc tính và phương thức) của lớp C_1 là tập con của tập các đặc tính của lớp C_2, đồng thời tập các đối tượng của lớp C_2 lại là tập con của tập các đối tượng của lớp C_1.

4.3. CHUYỂN ĐỔI MÔ HÌNH ER MỞ RỘNG SANG MÔ HÌNH HƯỚNG ĐỐI TƯỢNG

Quá trình chuyển đổi được thực hiện qua các bước như sau:

4.3.1. Bước 1: Chuyển đổi các tập thực thể E.

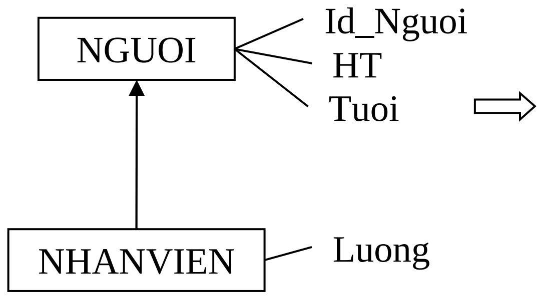
Mỗi tập thực thể của mô hình ER sẽ được chuyển đổi thành một lớp đối tượng có cùng tên và cùng tập thuộc tính. Ngoài ra, ta giả thiết rằng bất kỳ một mối quan hệ được xác định trong một mô hình ER chỉ có thể là giả thiết của một trong các bước được trình bày ở đây:

Lưu ý: (chuyển đổi mối quan hệ Is-a)

Nếu tập thực thể A có mối quan hệ Is-a với tập thực thể B thì lớp A sẽ kế thừa tất cả các thuộc tính trong lớp B bằng cách sử dụng từ khoá Inherits, đồng thời có thể bổ sung thêm các thuộc tính của lớp A.

Ví dụ:

Mô hình ER



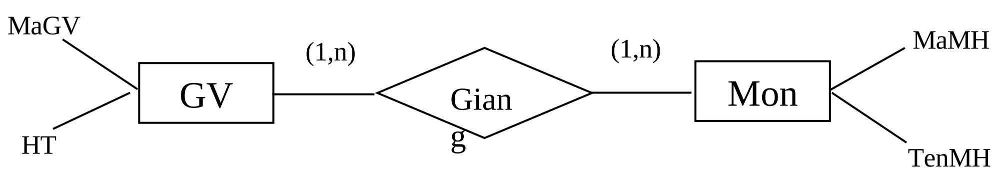
Mô hình hướng đối tượng

```
class NGUOI
    Id_Nguoi : string;
    HT : string;
    Tuoi : integer;
end NGUOI;

class NHANVIEN
    inherits : NGUOI;
    Luong : integer;
end NHANVIEN;
```

4.3.2. Bước 2: Chuyển đổi mối quan hệ nhị nguyên 1-1, 1-n, n-n (không có thuộc tính)

Nếu 2 tập thực thể A và B có mối quan hệ R (R không có các thuộc tính đính kèm), thì mỗi lớp tương ứng A và B, ngoài các thuộc tính có trong tập thực thể A và B, sẽ được bổ sung thêm thuộc tính R (thuộc tính mối quan hệ).




```
Class GV
        MaGV:       string;
        HT:         string;
        Giang:      Set(Mon);
End     GV;
Class Mon
        MaMH:       string;
        TenMH:      string
        [DuocGiangBoi: Set(GV); {Inverse GV.Giang}]
End Mon;
```

4.3.3. Bước 3: Chuyển đổi mối quan hệ nhị nguyên n-n (có thuộc tính).

Nếu mối quan hệ R của hai tập thực thể A_1 và A_2 có kèm các thuộc tính, khi đó, ngoài 2 lớp A_1 và A_2 tương ứng, ta cần bổ sung thêm một lớp mới C đóng vai trò trung gian. Cụ thể:

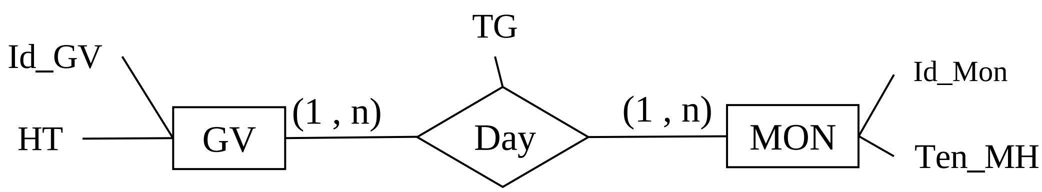
Lớp C bao gồm các thuộc tính sau:

- ❖ Các thuộc tính của mối quan hệ R.
- ❖ Hai thuộc tính có khai báo:

<Tên thuộc tính > : <Lớp A_1>

<Tên thuộc tính > : <Lớp A_2>

Ví dụ:



⇒Mô hình hướng đối tượng:

```
class GV
   Id_GV : string;
   HT : string;
   day1 : set(DAY);
end GV;
```

```
class MON
   Id_Mon : string;
   Ten_MH : string;
   day2 : set(DAY);
end MON;
```

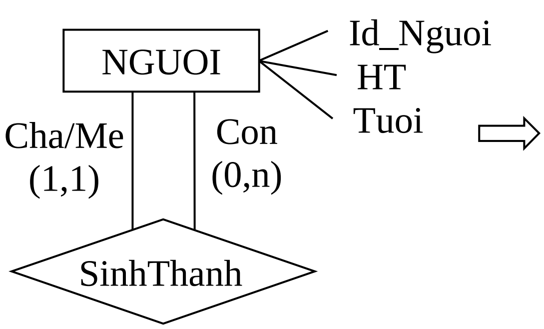
```
class DAY
   day1 : GV;
   day2 : MON;
   TG : string;
end DAY;
```

4.3.4. Bước 4: Chuyển đổi mối quan hệ phản xạ

Đối với mối quan hệ phản xạ, việc chuyển đổi được thực hiện tương tự như mối quan hệ nhị nguyên (bước 2 và bước 3).

Ví dụ:

Mô hình ER



Mô hình hướng đối tượng

```
class NGUOI
    Id_Nguoi : string;
    HT : string;
    Tuoi : integer;
    Cha, Me : NGUOI;
    Con : set(NGUOI);
end NGUOI;
```

4.3.5. Bước 5: Chuyển đổi mối quan hệ đa nguyên

Đối với mối quan hệ đa nguyên, việc chuyển đổi được thực hiện tương tự như mối quan hệ nhị nguyên n-n có thuộc tính (bước 3).

Nếu k tập thực thể A_1, A_2, …, A_k có quan hệ với nhau thông qua mối quan hệ đa nguyên R bậc k, thì ngoài k lớp A_1, A_2, …, A_k ta sẽ bổ sung thêm một lớp mới C đóng vai trò trung gian. Cụ thể:

Lớp C bao gồm các thuộc tính sau:

- Các thuộc tính của mối quan hệ R.
- Các thuộc tính R_i có khai báo:

<Tên thuộc tính > : <Lớp A_i>

Chương 5
LÝ THUYẾT THIẾT KẾ CƠ SỞ DỮ LIỆU QUAN HỆ

Như đã trình bày trong chương 3 mô hình CSDL quan hệ là một mô hình có tính độc lập dữ liệu cao, thuận lợi cho người sử dụng nên trong chương này chúng ta chúng ta đặc biệt quan tâm nghiên cứu một số phương pháp thiết kế một *mô hình CSDL tốt* từ một mô hình CSDL đã cho. Nội dung của chương chủ yếu trình bày các ràng buộc dữ liệu trong các quan hệ, các dạng chuẩn của các lược đồ CSDL quan hệ, và việc tách một lược đồ thành một số lược đồ con theo một dạng chuẩn nào đó.

5.1. Giới thiệu

Trong quá trình thiết kế một cơ sở dữ liệu quan hệ, một số vấn đề thường có thể xảy ra:

- Dư thừa dữ liệu
- Dị thường trong cập nhật dữ liệu

Ví dụ: Xét lược đồ quan hệ NC(TCC,DCC,TMH,GIA), trong đó TCC là tên người cung cấp, DCC là địa chỉ người cung cấp hàng, TMH là tên mặt hàng được cung cấp, GIA là giá tiền tương ứng của mặt hàng được cung cấp. Có một vài vấn đề nẩy sinh đối với lược đồ này:

1. Sự dư thừa: Địa chỉ của người cung cấp được lặp lại mỗi khi một mặt hàng được cung cấp. Chẳng hạn, trong quan hệ sau địa chỉ của ông Trần Ngọc An lặp lại 3 lần.

TCC	DCC	TMH	GIÁ
Trần Ngọc An	23. Lê Lợi	Khoai tây	4000
Lê Thị Nga	17. Hà nội	Cà chua	5000
Trần Ngọc An	23. Lê Lợi	Bắp cải	2000
Trần Ngọc An	23. Lê Lợi	Hành Tây	4000
Lê Văn Hoàn	18. Thuận An	Tỏi	100000
Đỗ Anh Tuấn	30. An Lăng	Cà rốt	8000

Bảng 3.1

2. Sự mâu thuẫn khi bổ sung: Nếu một người cung cấp hàng quen thuộc nào đó cung cấp mặt hàng mới nhưng địa chỉ anh ta đã thay đổi thì trong quan hệ sẽ xuất hiện sự mâu thuẩn (một người có hai điạ chỉ). Chẳng hạn, ông Lê Văn Hoàn sau khi chuyển về cư trú tại 15. Võ Thị Sáu bán thêm cho siêu thị mặt hàng su hào với giá 7000 đồng/kg quan hệ sẽ trở thành như sau:

TCC	DCC	TMH	GIÁ
Trần Ngọc An	23. Lê Lợi	Khoai tây	4000
Lê Thị Nga	17. Hà nội	Cà chua	5000
Trần Ngọc An	23. Lê Lợi	Bắp cải	2000
Trần Ngọc An	23. Lê Lợi	Hành Tây	4000
Lê Văn Hoàn	18. Thuận An	Tỏi	1000
Đỗ Anh Tuấn	30. An Lăng	Cà rốt	8000
Lê Văn Hoàn	15. Võ Thị Sáu	Su hào	7000

Bảng 3.2

3. Sự bất thường khi loại bỏ: Khi cần xóa các mặt hàng được cung cấp bởi một người chúng ta lại xóa hết các thông tin về người đó. Như vậy có thể xãy ra trường hợp thông tin về một người cung cấp mặt hàng nào đó không thể tìm thấy trong CSDL. Chẳng hạn, nếu siêu thị không cần mặt hàng Cà chua thì cơ sở dữ liệu ở bảng trên cần phải loại dòng thứ 2 ra khỏi bảng khi đó còn lại là:

TCC	DCC	TMH	GIÁ
Trần Ngọc An	23. Lê Lợi	Khoai tây	4000
Trần Ngọc An	23. Lê Lợi	Bắp cải	2000
Trần Ngọc An	23. Lê Lợi	Hành Tây	4000
Lê Văn Hoàn	18. Thuận An	Tỏi	100000
Đỗ Anh Tuấn	30. An Lăng	Cà rốt	8000
Lê Văn Hoàn	15. Võ Thị Sáu	Su hào	7000

Bảng 3.3

Do đó các thông tin về nhà cung cấp cà chua cũng không còn trong CSDL và chúng ta không thể tìm khi cần thiết.

4. Sự bất thường khi bổ sung: Một nhà cung cấp chưa cung cấp hàng thì không thể đưa địa chỉ, tên nhà cung cấp vào quan hệ. Chúng ta có thể đặt giá trị null cho các thành phần TMH và GIA của một bộ cho người cung cấp mới nhưng khi chúng ta đưa vào một mặt hàng với người cung cấp này, liệu chúng ta có nhớ để xóa bộ chứa giá trị null hay không?

Trong ví dụ đã nêu, những nhược điểm trên được khắc phục nếu chúng ta thay quan hệ NC bằng hai quan hê:

NCC(**TCC**,DCC) và CC(**TCC**,**TMH**,GIA)

<u>Khi đó quan hệ ở bảng 3.1 có thể được tách thành 2 bảng:</u>

TCC	DCC
Trần Ngọc An	23. Lê Lợi
Lê Thị Nga	17. Hà nội
Lê Văn Hoàn	18. Thuận An
Đỗ Anh Tuấn	30. An Lăng

Bảng 3.4

TCC	TMH	GIA
Trần Ngọc An	Khoai tây	4000
Lê Thị Nga	Cà chua	5000
Trần Ngọc An	Bắp cải	2000
Trần Ngọc An	Hành Tây	4000
Lê Văn Hoàn	Tỏi	1000
Đỗ Anh Tuấn	Cà rốt	8000

Bảng 3.5

Rõ ràng rằng dữ liệu ở bảng 3.4 và 3.5 được tổ chức trên hai lược đồ mới cùng cho một lượng thông tin như quan hệ trên lược đồ ban đầu, đồng thời tránh được các nhược điểm đã nêu ở trên.

Có hai phương pháp để thiết kế một cơ sở dữ liệu quan hệ

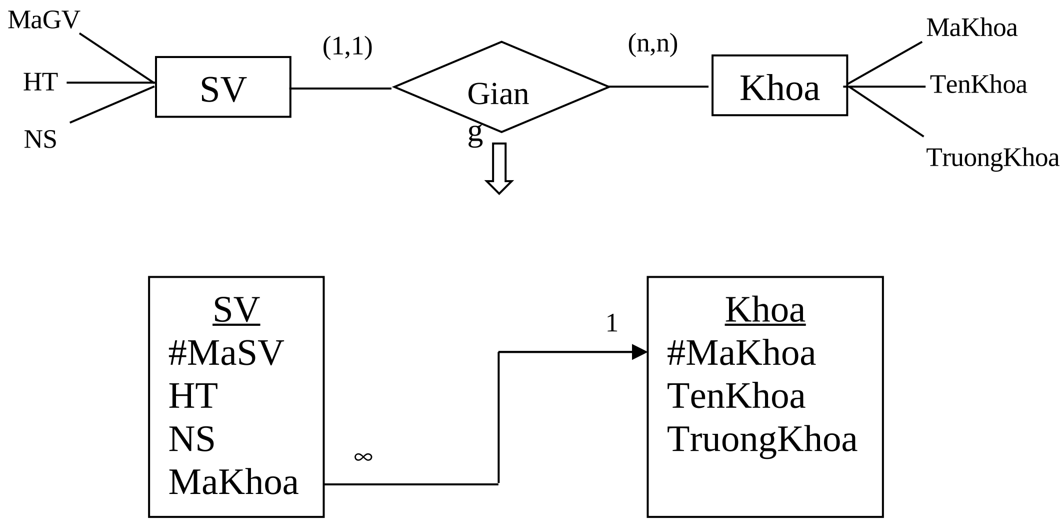
- ☝ Phương pháp 1: thiết kế xuất phát từ mô hình E-R.
- ☝ Phương pháp 2: thiết kế bằng phương pháp chuẩn hóa.

Ví dụ:

Giả sử ta cần xây dựng một hệ thống để quản lý thông tin của các sinh viên trong một trường đại học (MaSV, HT, NS) và thông tin về các khoa trong trường (MaKhoa, TenKhoa, TruongKhoa). Ngoài ra ta còn phải quản lý sinh viên thuộc khoa nào.

Thiết kế theo phương pháp 1.

Xây dựng mô hình E-R.



Thiết kế theo phương pháp 2.

Giả sử người ta chỉ sử dụng một lược đồ quan hệ để quản lý hệ thống này.

SinhVien = <U, SC>

Trong đó: U = {MaSV, HT, NS, MaKhoa, TenKHoa, TruongKhoa}

Cho quan hệ r như sau:

MaSV	HT	NS	MaKhoa	TenKhoa	TruongKhoa
1	Le Van A	1/1/08	A1	CNTT	Mr.Han
2	Le Van B	1/2/08	A1	CNTT	Mr.Han
3	Le Van C	1/3/08	A2	Toan	Mr.Phung
4	Le Van D	1/4/08	A2	Toan	Mr.Phung

SC = {MaSV $\rightarrow$ U ; MaKhoa $\rightarrow$ TenKhoa.TruongKhoa}

Việc thiết kế cơ sở dữ liệu chỉ sử dụng một lược đồ quan hệ như trên có thể nảy sinh các vấn đề như sau: Dư thừa dữ liệu, dị thường trong cập nhật dữ liệu : sửa, bổ sung, xoá.

Dựa theo phương pháp thứ hai, ta sẽ phát hiện ra những điều này do lược đồ quan hệ SinhViên không đạt chuẩn 3NF. Từ đó ta có thể thực hiện việc phân tách lược đồ quan hệ SinhVien thành hai lược đồ con SV và Khoa tương tự như kết quả cuối cùng của phương pháp 1.

5.2. CƠ SỞ LÝ THUYẾT CỦA PHỤ THUỘC HÀM

5.2.1. Qui ước về các ký hiệu:

☝ Các thuộc tính: A, B, C, …, A_1, A_2, …

- Tập các thuộc tính: X, Y, Z, …; ABC $\leftrightarrow$ {A, B, C}
- Hợp của các tập thuộc tính: XY $\leftrightarrow X\cup Y$; XYZ $\leftrightarrow X\cup Y\cup Z$; …
- Lược đồ quan hệ: R, S, …, RS = <U, SC> ;(SC : Set of Constraint)
- Quan hệ: r, s, …
- Bộ: t, t_1, t_2, …
- Với t là một bộ, X là tập thuộc tính thì ký hiệu t[X] để chỉ giá trị của bộ t trên tập thuộc tính X.

5.2.2. Phụ thuộc hàm (Functional Dependency)

Định nghĩa: (Quan hệ thỏa mãn phụ thuộc hàm)

Cho lược đồ quan hệ R = <U, SC>, cho X, Y $\subseteq$ U. Xét quan hệ r trên R. Quan hệ r được gọi là thỏa phụ thuộc hàm: X $\rightarrow$Y (đọc là X xác định Y, hoặc Y phụ thuộc hàm vào X) nếu và chỉ nếu: $\forall$ t_1, $t_2 \in$ r sao cho:

$$t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]$$

Nhận xét :

Do đó, r không thỏa X$\rightarrow$Y $\Leftrightarrow \exists t_1,t_2 \in$ r : $t_1[X] = t_2[X] \Rightarrow t_1[Y] \neq t_2[Y]$

Ví dụ:

r =

A	B	C	D
a	b	c	a
a	c	a	a
b	c	b	a

- r không thỏa : A $\rightarrow$B
- r thỏa : AB $\rightarrow$C
- r không thỏa : B $\rightarrow$C
- r thỏa : C $\rightarrow$A
- r thỏa : CD $\rightarrow$A
- r không thỏa : AD $\rightarrow$C

Thuật toán: Kiểm tra quan hệ r có thỏa mãn phụ thuộc hàm X $\rightarrow$Y không?

```
Function    Ktra(r, X, Y);
Begin
        Temp := true ;
        for i := 1 to n-1 do
          for j := i+1 to n do
                if  (ti[X]= tj[X] and ti[Y]<> tj[Y]) then
                        begin
                                temp:= false;
                                break;
```

```
            end;
    Ktra:= temp;
End;
```

Định nghĩa: (Lược đồ quan hệ thỏa mãn phụ thuộc hàm)

Cho lược đồ quan hệ R = <U, SC>, cho X, Y $\subseteq$ U. R được gọi là thỏa phụ thuộc hàm $X \rightarrow Y \Leftrightarrow \forall\, r \in R$: r thỏa $X \rightarrow Y$.

$\Leftrightarrow \forall\, r \in R, \forall\, t_1, t_2 \in r : t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]$

Lưu ý:

Thông thường ta quy ước rằng tập các ràng buộc SC chính là tập các phụ thuộc hàm và được ký hiệu là F. Vì vậy, một lược đồ quan hệ được ký hiệu là **R = <U, F>** thì R phải sẽ thỏa tất cả các phụ thuộc hàm trong F.

Định nghĩa: (Suy diễn logic)

Cho lược đồ quan hệ R = <U, F>, X, Y $\subseteq$ U.

Ta nói phụ thuộc hàm $X \rightarrow Y$ *được suy diễn logic từ* F, ký hiệu: $F \models X \rightarrow Y$, nếu với mỗi quan hệ r của R thỏa mãn các phụ thuộc hàm của F thì cũng thỏa $X \rightarrow Y$. Khi đó ta cũng nói rằng $X \rightarrow Y$ *là phụ thuộc hàm hệ quả* của F.

Ví dụ: F = {A $\rightarrow$B, B $\rightarrow$C} thì A $\rightarrow$C được suy diễn logic từ F (dễ thấy !)

Định nghĩa: (Bao đóng của tập phụ thuộc hàm)

Cho R = <U, F>. Bao đóng của tập phụ thuộc hàm F, ký hiệu là F^+, là tập tất cả các phụ thuộc hàm được suy diễn logic từ F, nghĩa là:

$$F^+ = \{X \rightarrow Y \mid X, Y \subseteq U \text{ và } F \models X \rightarrow Y\}$$

Ví dụ:

Cho R = <ABC, {A $\rightarrow$B, B $\rightarrow$C}>

Ta có F^+ = {A $\rightarrow$B, B $\rightarrow$C, A $\rightarrow$C, AB $\rightarrow$C, A $\rightarrow$A, ...}

Lưu ý: $F \subseteq F^+$

Định nghĩa: (Khoá của lược đồ quan hệ)

Cho R = <U, F>, cho X $\subseteq$ U. Khi đó X được gọi là khoá của R nếu và chỉ nếu thỏa mãn đồng thời 2 điều kiện:

1. $X \rightarrow U \in F^+$ (hay: X là siêu khoá của lược đồ quan hệ R)
2. Không $\exists$ X' $\subset$ X : X' là siêu khoá của R

Ví dụ 1: Cho R = <U, F>

U = ABC;   F = {A$\rightarrow$B, B$\rightarrow$C}

$\Rightarrow$X = A là khoá của lược đồ quan hệ R

Ta chứng minh: $A \rightarrow ABC \in F^+ \Leftrightarrow$ R thỏa $A \rightarrow ABC \Leftrightarrow \forall r \in R$: R thỏa A $\rightarrow$ABC (dễ dàng!).

Ví dụ 2: Cho R = <U, F>

U = ABC;   F = {AB$\rightarrow$C, C$\rightarrow$A}

R có hai khoá là AB và AC (vì sao ?)

Lưu ý:

- ☝ Khoá của một lược đồ quan hệ là không duy nhất.
- ☝ Một lược đồ quan hệ luôn $\exists$ khoá.

5.2.3. Hệ tiên đề Amstrong

Cho R = <U, F>, Armstrong đã đưa ra ba qui tắc (tiên đề, luật) sau:

- ☝ *Luật phản xạ:*

Nếu $X, Y \subseteq U$ và $X \subseteq Y$ thì $Y \rightarrow X \in F^+$

- ☝ *Luật gia tăng (tăng trưởng):*

Nếu $X, Y, Z \subseteq U$ và $X \rightarrow Y \in F^+$ thì $XZ \rightarrow YZ \in F^+$

- ☝ *Luật bắc cầu:*

Nếu $X, Y, Z \subseteq U$ và $X \rightarrow Y \in F^+$, $Y \rightarrow Z \in F^+$ thì $X \rightarrow Z \in F^+$

Ta có các quy tắc suy diễn mở rộng:

Mệnh đề: Cho R = <U, F>, $X, Y, Z, W \subseteq U$. Lúc đó:

1) Luật hợp: Nếu $X \rightarrow Y$ và $X \rightarrow Z$ thì $X \rightarrow YZ$

2) Luật tách: Nếu $X \rightarrow Y$ và $Z \subseteq Y$ thì $X \rightarrow Z$

3) Luật tựa bắc cầu: Nếu $X \rightarrow Y$ và $WY \rightarrow Z$ thì $WX \rightarrow Z$

Định lý: Hệ tiên đề Armstrong là đúng đắn và đầy đủ.

- ☝ Tính đầy đủ của hệ tiên đề trên được hiểu là với bất kỳ phụ thuộc hàm hệ quả nào ta luôn có thể sử dụng một số hữu hạn quy tắc trong hệ tiên đề này để chứng minh.

Ví dụ:

Cho R = <U, F>, trong đó: U = ABC, F = $\{A \rightarrow B, A \rightarrow C\}$

Chứng minh: $A \rightarrow BC \in F^+$

Ta có: $A \rightarrow B$ (1)

$A \rightarrow C$ (2)

Từ (1) $\Rightarrow A \rightarrow AB$ (3) (Luật gia tăng)

Từ (2) $\Rightarrow AB \rightarrow BC$ (4) (Luật gia tăng)

Từ (3) & (4) $\Rightarrow A \rightarrow BC$ (Luật bắc cầu)

$\Rightarrow$ đpcm

5.2.4. Bao đóng của tập thuộc tính (X^+)

Định nghĩa:

Cho R = <U, F>. Cho $X \subseteq U$. Khi đó, bao đóng của X, ký hiệu là X^+, được định nghĩa như sau:

$$X^+ = \{A \mid X \rightarrow A \in F^+\}$$

Lưu ý:

- ☝ Để chỉ rõ bao đóng của X được xác định trên tập phụ thuộc hàm F, ta ký hiệu: X^+_F
- ☝ X là siêu khoá $\Leftrightarrow X \rightarrow U \in F^+ \Leftrightarrow X^+ = U$
- ☝ $X \subseteq X^+$

Định lý: (Bài toán thành viên: Điều kiện cần và đủ để $X \rightarrow Y \in F^+$)

Cho R = <U, F> và X, Y $\subseteq$ U. Khi đó:

$$\boxed{X \rightarrow Y \in F^+ \Leftrightarrow Y \subseteq X^+}$$

Chứng minh: Giả sử $Y = A_1 \ldots A_n$, với $A_i \in U$ là các thuộc tính.

Đủ: Giả sử $Y \subseteq X^+$, suy ra $A_i \in X^+$, với mọi i =1,…,n.

Từ định nghĩa của X^+ ta có $X \rightarrow A_i$, suy ra $X \rightarrow Y$ (luật hợp).

Cần: Giả sử có $X \rightarrow Y \in F^+$, suy ra $X \rightarrow A_i \in F^+$(luật tách).

Từ đó suy ra $Y \subseteq X^+$.

Ví dụ:

Cho R = <U, F>, trong đó: U = ABC, F = $\{A \rightarrow B, A \rightarrow C\}$

$AB \rightarrow C \in F^+$ do $(AB)^+ = ABC \supseteq C$

$B \rightarrow A \notin F^+$ (do $B^+ = B$ không chứa A)

5.2.5. Thuật toán tính bao đóng của tập thuộc tính

Vào: R = <U, F> và $X \subseteq U$

Ra: X^+_F

Phương pháp:

```
Function BaoDong(X);
Begin
    new:= X;
    Repeat
        old:= new;
        for mỗi X →Y ∈ F do
            if X ⊇ new thenm
                new:= new ∪ Y;
    Until old = new;
    Return new;
End;
```

Ví dụ:

Cho R = <U, F>, trong đó:

U = ABCDEG

$F = \{AB \to C, C \to A, BC \to D, D \to EG, BE \to C, CG \to BD, CE \to AG\}$

Tính $(BD)^+$:

$X^{(0)} = BD$

$X^{(1)} = BD \cup \{EG\} = BDEG$

$X^{(2)} = BDEG \cup \{C\} = BDEGC$

$X^{(3)} = BDDEGC \cup \{A, AG\} = BDEGCA$

$X^{(4)} = BDEGCA = X^{(3)}$

$\Rightarrow (BD)^+ = ABCDEG$

Chứng minh BD là khoá của R:

Theo câu trên ta có BD là siêu khoá. Mặt khác ta có:

$B^+ = B \neq U \Rightarrow B \to U \notin F^+$

$D^+ = DEG \neq U \Rightarrow D \to U \notin F^+$

$\Rightarrow$ đpcm

Bài tập:

Viết thuật toán tựa Pascal nhằm kiểm tra tập X nào đó có phải là khoá của R hay không?

- Vào: R = <U, F> và $X \subseteq U$
- Ra: Yes/No

5.2.6. Phủ tối thiểu (phủ cực tiểu)

Định nghĩa: (Hai tập phụ thuộc hàm tương đương)

Cho 2 tập phụ thuộc hàm F và G. Khi đó: F được gọi là tương đương với G, ký hiệu: $F \Leftrightarrow G$, nếu và chỉ nếu $F^+ = G^+$.

Ví dụ:

Cho $F = \{A \to B, B \to C\}$, $G = \{A \to B, B \to C, A \to C\}$

$\Rightarrow F \Leftrightarrow G$

Ý nghĩa:

Cho R = <U, F> và G tương đương với F thì ta có thể dùng G thay cho F trong lược đồ quan hệ R. Tức là R = <U, F> = <U, G>

- Bổ đề 1: Nếu $F \subseteq G \Rightarrow F^+ \subseteq G^+$.
- Bổ đề 2: $(F^+)^+ = F^+$

Tính chất:

$$F \Leftrightarrow G \Leftrightarrow \begin{cases} F \subseteq G^+ (1) \\ G \subseteq F^+ (2) \end{cases}$$

Chứng minh, dùng 02 bổ đề ở trên:

(⇒): $F \subseteq F^+ = G^+ \Rightarrow (1)$ đúng

$G \subseteq G^+ = F^+ \Rightarrow (2)$ đúng

(⇐): $F \subseteq G^+ \Rightarrow (F)^+ \subseteq (G^+)^+ = G^+$ (bổ đề 1)

$G \subseteq F^+ \Rightarrow (G)^+ \subseteq (F^+)^+ = F^+$

Vậy $F^+ = G^+$, do đó F tương đương G.

Ý nghĩa:

Cho phép xây dựng thuật toán để kiểm tra xem hai tập phụ thuộc hàm có tương đương với nhau hay không.

Thuật toán kiểm tra F tương đương G

```
Procedure    KTTD(F, G) ;
Begin
        1. {Kiểm tra F ⊆ G+}
        for mỗi X→Y ∈ F do
                if  X+G ⊉ Y then
                        begin
                                write('F và G là không tương đương');
                                exit;
                        end;
        2. {Kiểm tra G ⊆ F+}
        for mỗi X→Y ∈ G do
                if  X+F ⊉ Y then
                        begin
                                write('F và G là không tương đương');
                                exit;
                        end;
      3. writeln(' F tương đương với G');
End;
```

Bổ đề:

Cho R = <U, F>, $X \subseteq U$, $A_i \in U$, i = 1..n

Ta có: $X \to A_1A_2..A_n \in F^+ \Leftrightarrow \begin{cases} X \to A_1 \in F^+ \\ X \to A_2 \in F^+ \\ \vdots \\ X \to A_n \in F^+ \end{cases}$

Tính chất:

Cho R=<U, F>. Khi đó bao giờ cũng ∃ một phụ thuộc hàm G

- $G \Leftrightarrow F$
- Vế phải của mọi phụ thuộc hàm trong G chỉ có một thuộc tính.

Định nghĩa: (Phủ tối thiểu của một lược đồ quan hệ)

Cho R = <U, F>, F được gọi là phủ tối thiểu của R khi và chỉ khi:

a. Vế phải của mọi phụ thuộc hàm trong F là chỉ có 1 thuộc tính.

b. Mọi phụ thuộc hàm trong F không có thuộc tính dư thừa ở vế trái, nghĩa là: $\forall\ X \to A \in F, \forall\ B \in X,$

$$((F \setminus \{X \to A\}) \cup (X \setminus \{B\} \to A))^+ \neq F^+$$

c. Trong F không có phụ thuộc hàm nào là dư thừa, nghĩa là:

$$\forall\ X \to A \in F,\ X \to A \notin (F \setminus \{X \to A\})^+$$

Nhận xét:

1) Kiểm tra điều kiện a) khá dễ dàng.
2) Điều kiện b) $\Leftrightarrow \forall\ X \to A \in F, \forall\ B \in X,$

$$((F \setminus \{X \to A\}) \cup (X \setminus \{B\} \to A))^+ \neq F^+$$

Ta có thể xem điều kiện trên $\Leftrightarrow$ điều kiện sau:

$$\forall\ X \to A \in F, \forall\ B \in X \text{ thì } X \setminus \{B\} \to A \notin F^+$$

$$\boldsymbol{\Leftrightarrow \forall\ X \to A \in F, \forall\ B \in X \text{ thì } (X \setminus \{B\})^+ \diagup A}$$

3) Điều kiện c) $\Leftrightarrow \forall\ X \to A \in F,\ X \to A \notin (F \setminus \{X \to A\})^+$

$$\boldsymbol{\Leftrightarrow \forall\ X \to A \in F, X^+_{F \setminus \{X \to A\}} \diagup A}$$

Ví dụ:

$$R = \langle U, F \rangle,\ U = ABC, \quad F = \{A \to B, AB \to C\}$$

Hỏi F có phải là phủ tối thiểu của R không?

☝ Kiểm tra điều kiện a: đúng

☝ Kiểm tra điều kiện b:

Xét $AB \to C \in F$, ta có:

X = AB, xem thử thuộc tính B có dư thừa không?

Ta có: $(X \setminus B)^+ = A^+ = ABC \supseteq C$ Suy ra thuộc tính B ở vế trái của phụ thuộc hàm $AB \to C$ là dư thừa. Vậy F không phải là phủ tối thiểu của R.

Ví dụ:

Cho R = <U, F>, U = ABC, $F = \{A \to B, B \to C, A \to C\}$. Hỏi F có phải là phủ tối thiểu của R không?

☝ Kiểm tra điều kiện a: đúng

☝ Kiểm tra điều kiện b: đúng

☝ Kiểm tra điều kiện c:

Xét $A \to C \in F$, ta có:

$F \setminus \{A \to C\} = \{A \to B, B \to C\}$

$A^+_{\{A \to B,\ B \to C\}} = ABC \supseteq C$ Vậy trong F có phụ thuộc hàm $A \to C$ là dư thừa, do đó F không phải là phủ tối thiểu của R.

Bài tập:

Cho R = <U, F>. Viết giải thuật tựa Pascal để kiểm tra F có phải là một phủ tối thiểu của R hay không?

Định nghĩa: (Phủ tối thiểu của tập phụ thuộc hàm)

Cho R = <U, F>. Tập phụ thuộc hàm G được gọi là một phủ tối thiểu của F nếu thỏa hai điều kiện:

- ☝ $G \Leftrightarrow F$
- ☝ G là phủ tối thiểu của R' = <U, G>

Lưu ý:

Phủ tối thiểu của một phụ thuộc hàm là không duy nhất

Ví dụ: Cho R = <U, F>, U = ABC

$F = \{A\to B, B\to A, A\to C, C\to A, B\to C, C\to B\}$

Khi đó, F có các phủ tối thiểu khác nhau như sau: (?)

$G_1 = \{A\to B, B\to A, A\to C, C\to A\}$

$G_2 = \{A\to B, B\to A, B\to C, C\to B\}$

$G_3 = \{A\to B, B\to C, C\to A\}$

Thuật toán (Tìm một phủ tối thiểu của F)

- ☝ Vào: R = <U, F>
- ☝ Ra: G (G là một phủ tối thiểu của T)
- ☝ Phương pháp:

Procedure TimPhuCucTieu(F);

1. Phân rã các phụ thuộc hàm trong F mà vế phải có 2 thuộc tính trở lên (làm cho F thỏa điều kiện a). Cụ thể:

$$X \to A_1A_2...A_n \xrightarrow{\text{Phân rã}} \begin{array}{l} X \to A_1 \\ X \to A_2 \\ \vdots \\ X \to A_n \end{array}$$

2. (Làm cho F thỏa điều kiện b)

for (mỗi $X \to A \in F$) do
  for (mỗi $B \in X$) do
    if ($(X \setminus \{B\})^+_F \supseteq A$) then
      $X := X \setminus \{B\}$;

3. (Làm cho F thỏa điều kiện c)

for (mỗi $X \to A \in F$) do
  if ($X^+_{F\setminus\{X\to A\}} \supseteq A$) then
    $F := F \setminus \{X\to A\}$;

4. Kết luận:

G := F;

End;

Ví dụ

Cho R = <U, F>, U = ABC, F = $\{A \to BC, AB \to C\}$

Tìm một phủ tối thiểu của F.

Bước 1:

F = $\{A \to B, A \to C, AB \to C\}$

Bước 2:

Xét $AB \to C \in F$, ta có:

$(AB\backslash B)^+ = A^+ = ABC \supseteq C$. Vậy thuộc tính B là dư thừa.

$(AB\backslash A)^+ = B^+ = B \not\supseteq C$. Vậy thuộc tính A không dư thừa.

$\Rightarrow F = \{A \to B, A \to C\}$

Bước 3:

Xét $A \to B$: $A^+_{F\backslash\{A \to B\}} = AC \not\supseteq B$

Xét $A \to C$: $A^+_{F\backslash\{A \to C\}} = AB \not\supseteq C$

Kết luận:

Vậy G = $\{A \to B, A \to C\}$ là một phủ tối thiểu của F cần tìm.

Lưu ý:

Trong thuật toán tìm một phủ tối thiểu của F, không thể thực hiện làm cho F thỏa điều kiện c) trước khi làm cho F thỏa điều kiện b) (nói cách khác, việc là cho F thỏa điều kiện c) luôn phải được thực hiện sau khi làm cho F thỏa điều kiện b)).

Phương pháp sai: (Bước 3 thực hiện trước bước 2)

Cho R = <U, F>, U = ABD, F = $\{B \to D, B \to A, D \to A, AB \to D\}$

- Nếu ta loại bỏ các phụ thuộc hàm dư thừa trước, thì lần lượt xét:

$B^+_{F\backslash\{B \to D\}} = BAD \supseteq D \Rightarrow$ loại bỏ $B \to D$.

$\Rightarrow F = \{B \to A, D \to A, AB \to D\}$

$B^+_{F\backslash\{B \to A\}} = B \not\supseteq A$

$D^+_{F\backslash\{D \to A\}} = D \not\supseteq A$

$(AB)^+_{F\backslash\{AB \to D\}} = AB \not\supseteq D$

- Tiếp đến, loại bỏ thuộc tính dư thừa ở vế trái, chỉ xét: $AB \to D$

Ta có: $B^+_F = BAD \supseteq D \Rightarrow$ loại bỏ A

$\Rightarrow F = \{B \to A, D \to A, B \to D\}$

- Thế nhưng, ta thấy $B \to A \in F$ là phụ thuộc hàm dư thừa.

Phương pháp đúng:

- Loại bỏ thuộc tính dư thừa ở vế trái, chỉ xét: $AB \to D$

Ta có: $B^+_F = BAD \supseteq D \Rightarrow$ loại bỏ A

$\Rightarrow F = \{B \to A, D \to A, B \to D\}$

- Loại bỏ các phụ thuộc hàm dư thừa:

$B \to D$:

Ta có: $B^+_{F\backslash\{B \to D\}} = BA \not\supseteq D$

$B \rightarrow A$:

Ta có: $B^{+}_{F\setminus\{B\rightarrow A\}} = BDA \supseteq A \Rightarrow$ loại bỏ $B \rightarrow A$

$\Rightarrow F = \{D \rightarrow A, B \rightarrow D\}$

$D \rightarrow A$:

Ta có: $D^{+}_{F\setminus\{D\rightarrow A\}} = D \;/\; A$

- Kết luận: $F = \{D \rightarrow A, B \rightarrow D\}$

5.2.7. Các thuật toán xác định một khoá của một lược đồ quan hệ

X là khoá của lược đồ quan hệ R = <U, F> nếu:

1) $X \rightarrow U \in F^{+}$ hay $X^{+}_{F} = U$ (X là siêu khoá)

2) Không tồn tại $X' \subset X$, $X'^{+} = U$

Thuật toán tìm một khoá của một lược đồ quan hệ.

- Vào: R =<U,F>
- Ra : K (một khoá của R)
- Phương pháp:

```
Function Key(R)
Begin
    K:= U;
    For mỗi A ∈ U do
        If (K\A)+F = U then
            K := K \ {A}; {loại bỏ thuộc tính A}
    Return K;
End;
```

Ví dụ

R = <U,F>, U = ABCDEG; F = {B   C,C   B,A   GD}

K = ABCDEG

Ta có:

- $(K\setminus A)_F^{+} = (BCDEG)^{+} = BCDEG \neq U$
- $(K\setminus B)_F^{+} = (ACDEG)^{+} = ACDEGB = U$

$\Rightarrow K = ACDEG$ (loại B)

- $(K\setminus C)_F^{+} = (ADEG)^{+} = ADEG \neq U$
- $(K\setminus D)_F^{+} = (ACEG)^{+} = ACBEGD = U$

$\Rightarrow K = ACEG$ (loại D)

- $(K\setminus E)_F^{+} = (ACG)^{+} = ACGBD \neq U$
- $(K\setminus G)_F^{+} = (ACE)^{+} = ACEGDB = U$

Vậy **K = ACE** là khoá của lược đồ quan hệ.

Chú ý: Khi thay đổi thứ tự việc loại bỏ các phần tử của K, ta có thể thu được khóa khác của lược đồ quan hệ.

Chúng ta có thể cải tiến thuật toán trên thông qua định lý sau:

Định lý Hồ Thuần - Nguyễn Văn Bào: (Điều kiện cần để X là khoá).

Cho R = <U, F>, cho $X \subseteq U$. Khi đó, nếu X là khoá của R thì:

$$(U \setminus P) \subseteq X \subseteq (U \setminus P) \cup (T \cap P)$$

Trong đó: $T = \bigcup_{X\ \ Y\ \ F} X$

$P = \bigcup_{X\ \ Y\ \ F} Y$

Ý nghĩa:

Định lý này cho phép ta thu hẹp phạm vi tìm kiếm khoá

Ví dụ: Cho R = <U,F>, trong đó:

U = ABCDEG; F = {B C,C B,A GD}

� T = ABC, P = BCDG �� T �P=BC

Gọi X là khoá của R: $U \setminus P = AE \subseteq X \subseteq (U \setminus P) \cup (T \cap P) = AEBC$

Nếu gọi K là tập tất cả các khoá của R $\Rightarrow$ K chỉ có thể chứa AE, AEB, AEC, AEBC.

Ta có: $(AE)^+ = AEGD \neq U$ $AE \notin K$

$(AEB)^+ = AEBGDC = U \Rightarrow AEB \in K$

$(AEC)^+ = AEBGDC = U \Rightarrow AEC \in K$

Có thể chỉ ra K = {AEB, AEC}

Ngoài ra, AEBC chỉ là siêu khoá của R

Hệ quả 1:

> Nếu $(U \setminus P)^+ = U$ thì $U \setminus P$ là khoá duy nhất của R.

Ví dụ:

Xét R = <U, F>, với $U = A_1A_2A_3A_4A_5A_6$

$F = \{A_1 \quad A_2, A_3 \quad A_4, A_5 \quad A_6\}$

$\Rightarrow P = A_2A_4A_6$ mà $(U \setminus P)^+ = (A_1A_3A_5)^+ = A_1A_3A_5A_2A_4A_6 = U$

$\Rightarrow$ R có 1 khoá duy nhất là $A_1A_3A_5$

Hệ quả 2:

> Nếu $T \cap P = \varnothing$ thì $U \setminus P$ là khoá duy nhất của R.

Hệ quả 3:

> $(U \setminus P) \cup (T \cap P)$ là siêu khoá của R.

Nhận xét:

Từ 3 hệ quả trên cho phép ta xác định giải thuật để tìm một khoá của lược đồ quan hệ R như sau :

Vào: R = <U, F>

Ra: Tìm 1 khoá của R

Phương pháp:

```
Function Key(R)
Begin
        K := (U \ P) ∪ (T ∩ P)
        For  <mỗi A ∈  (T ∩ P)> do
                If <(K \ A)+ = U> then
                        K := K \ A;   {loại bỏ thuộc tính A}
        Return K.
End;
```

Ví dụ:

R = <U,F>, U = ABCDEG; F = {B C,C B,A GD}

T =ABC

P = BCDG �� T �P = BC

X = (U\P) ∪ (T∩P) = ABCE

$(X \setminus B)^+ = U \Rightarrow X = ACE$ (loại B)

$(X \setminus C)^+ = (AE)^+ \neq U$

$\Rightarrow ACE \in K$

5.2.8. Giải thuật xác định tất cả các khoá của một lược đồ quan hệ.

Phương pháp 1: Sử dụng cây tìm kiếm

Ý tưởng:

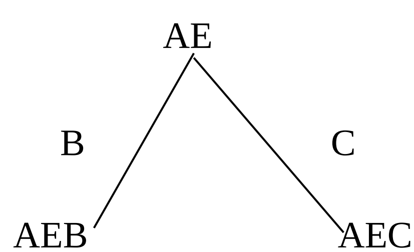
Dựng một cây có nút gốc là U\P, rồi thêm dần các thuộc tính còn thiếu trong T∩P cho đến khi được các siêu khoá

Ví dụ:

Cho R = <U, F> với U = ABCDEG

F = {B →C, C →B, A →GD} $\begin{cases} P = BCGD \\ T = ABC \end{cases} \Rightarrow \begin{cases} U \setminus P = AE \\ T \cap P = BC \end{cases}$

Dựng cây:



Tính $(AE)^+ = AEGD \neq U$

$(AEB)^+ = AEBCGD = U$

$(AEC)^+ = AECBGD = U$

Kết luận:

K = {AEB, AEC}

Mô tả phương pháp:

Bước 1: tính $(U\backslash P)^+$. Nếu $(U\backslash P)^+ = U$ thì kết luận K = {U\P}. Ngược lại, dựng cây có nút gốc là U\P và các nút con là: (U\P)A, với $A \in T \cap P$

Bước 2: lần lượt duyệt cây theo chiều rộng đối với các nút chưa được đánh dấu đến chừng nào mọi nút lá của cây đều được đánh dấu.

Lưu ý:

Tại mỗi nút X chưa được đánh dấu ta thực hiện thủ tục sau:

```
Procedure    DuyetNut (X);
Begin
        If X chứa nút đã đánh dấu khoá then
                Đánh dấu X là nút đã xét
        Else if  X+ = U then
                Đánh dấu X là nút khoá
        Else
                Tạo các nút con của X bằng cách bổ sung các thuộc tính còn
                thiếu trong T∩P.
End;
```

Phương pháp 2: Sử dụng định lý Lucchessi-Osborn.

Định lý Lucchessi và Osborn: (điều kiện cần và đủ để bổ sung khoá)

Cho R = <U, F>. Gọi $\mathcal{K}$ là một tập khác rỗng các khoá của lược đồ quan hệ R. Khi đó, điều kiện cần và đủ để có thể bổ sung khoá mới vào $\mathcal{K}$ là:

$\exists\ K \in \mathcal{K}, \exists\ X \rightarrow Y \in F$: $T = X \cup (K \setminus Y)$ không chứa phần tử nào của K.

Lưu ý:

Phần chứng minh của định lý này chứng tỏ rằng nếu điều kiện trên được thỏa mãn thì khoá mới được bổ sung vào $\mathcal{K}$ là khoá k' nhận T làm siêu khoá. Từ đây, ta có thể xác định được giải thuật tìm tất cả các khoá của R như sau:

5.2.8.3. Thuật toán tìm tất cả các khoá của một lược đồ quan hệ

Vào: R = <U, F>

Ra: K

Phương pháp:

```
Procedure TimTatCaKhoa(R);
Begin
        1. Tìm một khoá K ∈ 𝒦  {theo thuật toán tìm một khóa}
                𝒦 := {K};
        2. For  <mỗi K ∈ 𝒦> do
                For  <mỗi X     Y ∈ F> do
                begin
                        T := X ∪ (K \ Y) ;
                        If  <T không chứa phần tử nào của 𝒦> then
                                begin
                                        Tìm một khoá K' nhận T làm siêu khoá ;
                                        𝒦 := 𝒦 ∪ {K'}
                                        goto 2; {không cần xét lại (K, X→Y) đã xét}
                                endIf;
                end;
        Return;
End;
```

Ví dụ:

Cho R = <U, F> với U = ABCDEG

$F = \{B \to C, C \to B, A \to GD\}$

Tìm một khóa của R theo thuật toán tìm 1 khóa: ABE

Đặt $\mathcal{K} = \{ABE\}$

☝ Xét $K = ABE \in \mathcal{K}$:

Xét $B \to C \in F \Rightarrow T = B \cup (K \backslash C) = ABE$ chứa phần tử của $\mathcal{K}$

Xét $C \to B \in F \Rightarrow T = C \cup (K \backslash B) = ACE$ không chứa phần tử nào của $\mathcal{K}$

Tìm khoá K' nhận T = ACE làm siêu khoá (theo thuật toán đã biết), ta tìm được khóa là K' = ACE $\Rightarrow \mathcal{K} = \{ABE, ACE\}$

Xét $A \to GD \in F \Rightarrow T = ABE$ chứa phần tử của $\mathcal{K}$

☝ Xét $K = ACE \in \mathcal{K}$:

Xét $B \to C \in F \Rightarrow T = ABE$ chứa phần tử của $\mathcal{K}$

Xét $C \to B \in F \Rightarrow T = ACE$ chứa phần tử của $\mathcal{K}$

Xét $A \to GD \in F \Rightarrow T = ACE$ chứa phần tử của $\mathcal{K}$

Vậy $\mathcal{K} = \{ABE, ACE\}$

5.3. LÝ THUYẾT PHÂN TÁCH

Từ một CSDL lớn, có thể làm nảy sinh sự dư thừa dữ liệu và những dị thường trong cập nhập dữ liệu. Chính vì vậy cần phân tách lược đồ này thành các lược đồ con. Trong lí thuyết phân tách, yêu cầu việc phân tách này phải đảm bảo được tính chất “bảo toàn thông tin”.

Ngoài ra, phân tích này sẽ tạo điều kiện thuận lợi cho người lập trình nếu nó còn bảo đảm được tính chất “bảo toàn phụ thuộc hàm”.

Định nghĩa: (Phân tách / Phép tách)

Cho lược đồ quan hệ R = <U, F>, các lược đồ con $R_1 = <U_1, F_1>$, $R_2 = <U_2, F_2>$, …, $R_n = <U_n, F_n>$. ρ được gọi là một phân tách của R thành các lược đồ con $R_1, R_2, \ldots, R_n$, (ký hiệu $\left.\begin{array}{l}\rho = (R_1, R_2, \ldots, R_n)\\ \rho = (U_1, U_2, \ldots, U_n)\end{array}\right)$ nếu $\bigcup_{i=1}^{n} U_i = U$.

Ví dụ:

Cho R = <U, F> với U = ABCD

$\Rightarrow$ ρ = (AB, BC, ACD) là 1 phân tách của R.

Nhận xét:

Cho r là một quan hệ trên R và $\rho = (U_1, U_2, \ldots, U_n)$ là một phân tách của R. Khi đó, phân tách ρ cho ta các quan hệ con tương ứng $r_1, r_2, \ldots, r_n$ được xác định như sau:

$$r_i = \pi_{U_i}(r) \qquad (i = \overline{1,n})$$

5.3.1. Phân tách bảo toàn thông tin

(Phép tách có kết nối không mất thông tin)

Định nghĩa: (Phân tách bảo toàn thông tin trên một quan hệ)

Cho R = <U, F> và phép tách $\rho = (U_1, U_2, \ldots, U_n)$ trên R. Khi đó phép tách ρ được gọi là bảo toàn thông tin trên r nếu và chỉ nếu:

$$\pi_{U_1}(r) \bowtie \pi_{U_2}(r) \bowtie \ldots \bowtie \pi_{U_n}(r) = r, \quad \forall r \in R$$

Định nghĩa: (Phân tách bảo toàn thông tin trên một lược đồ quan hệ)

Cho lược đồ quan hệ R = <U, F>.

Khi đó, phép tách $\rho = (U_1, U_2, \ldots, U_n)$ được gọi là bảo toàn thông tin trên R nếu $\forall r \in R$: ρ là bảo toàn thông tin trên r.

Định lý 1: (Điều kiện đủ để một phân tách là bảo toàn thông tin)

Cho R = <U, F> và X, Y $\subseteq$ U.

Khi đó, nếu $X \rightarrow Y \in F^+$ sao cho $X \cap Y = \emptyset$ và $Z = (U \backslash XY) \neq \emptyset$ thì: ρ = (XY, XZ) là bảo toàn thông tin (BTTT).

Ví dụ:

Cho $r \in$ NKBH = <U, F>, với F = {STT→U, MH→TH, MH→ĐG}

r =

STT	TK	NG	MH	TH	ĐG	SL
1	A	1/1/2000	A_1	Cà	10	3
2	A		A_2	Tiêu	20	7
3	B		A_2	Tiêu	20	1
4	C		A_1	Cà	10	2

ρ = (HG, NK) với HG (MH, TH, ĐG); NK (STT, TK, NG, MH, SL)

Từ F $\Rightarrow$MH $\rightarrow$ {TH, ĐG} $\in F^+$

$\Rightarrow \rho = (U_1, U_2)$ là BTTT.

Định lý 2: (Điều kiện cần và đủ để một phân tách thành hai lược đồ con là bảo toàn thông tin)

Cho R = <U, F>. Khi đó, phân tách $\rho = (U_1, U_2)$ trên R bảo toàn thông tin nếu và chỉ nếu:

$$U_1 \cap U_2 \rightarrow U_1 \backslash U2 \in F^+$$

hoặc $U_1 \cap U_2 \rightarrow U_2 \backslash U_1 \in F^+$

Ví dụ: xét ví dụ NKBH ở trên

$(U_1 \cap U_2)^+ = (MH)^+ = \{MH, TH, ĐG\}$

$(U_1 \setminus U_2) = \{TH, ĐG\}$

Vậy $(U_1 \setminus U_2) \subseteq (U_1 \cap U_2)^+$

$\Rightarrow U_1 \cap U_2 \rightarrow U_1 \backslash U_2 \in F^+$

$\Rightarrow \rho = (U_1, U_2)$ là bảo toàn thông tin.

Thuật toán kiểm tra tính chất bảo toàn thông tin của một phân tách:

Vào: R = <U, F> với $U = \{A_1, A_2, \ldots, A_n\}$ và $\rho = (U_1, U_2, \ldots, U_k)$

Ra: Yes/No

Phương pháp:

Bước 1 (Lập bảng):

Thành lập 1 bảng gồm k dòng và n cột. Các cột được ký hiệu bởi các thuộc tính A_i ($i = \overline{1,n}$), các dòng được kí hiệu bởi các tập thuộc tính U_j ($j= \overline{1,k}$), các phần tử của bảng sẽ được ghi là a_i hoặc b_{ij} theo nguyên tắc sau:

- phần tử dòng i cột j là a_j nếu $A_j \in U_i$
- phần tử cột dòng i cột j là b_{ij} nếu ngược lại

Bước 2 (Biến đổi bảng):

Repeat

For mỗi $X \to Y \in F$ do

Thực hiện việc bằng trên Y với những dòng có chung X (ưu tiên biến đổi b_{ij} thành a_j)

Until bảng không thay đổi đối với vòng for

Bước 3: (Kết luận)

- ☝ Nếu tồn tại một dòng (bộ) hoàn toàn các giá trị a_i ($i = \overline{1,n}$) $\Rightarrow \rho$ bảo toàn thông tin
- ☝ Ngược lại $\Rightarrow \rho$ không bảo toàn thông tin.

Ví dụ:

Cho R = <U, F>, với U = ABCDE

F = {A →C, B →C, C →D, DE →C, DE →A}

Và ρ = (AD, AB, BE, CDE, AE)

Bước 1: <Lập bảng>

	A	B	C	D	E
AD	a_1	b_{12}	b_{13}	a_4	b_{15}
AB	a_1	a_2	b_{23}	b_{24}	b_{25}
BE	b_{31}	a_2	b_{33}	b_{34}	a_5
CDE	b_{41}	b_{42}	a_3	a_4	a_5
AE	a_1	b_{52}	b_{53}	b_{54}	a_5

Bước 2: Biến đổi bảng

Xét A →C

	A	B	C	D	E
AD	$\mathbf{a_1}$	b_{12}	$\mathbf{b_{13}}$	a_4	b_{15}
AB	$\mathbf{a_1}$	a_2	$\mathbf{b_{13}}$	b_{24}	b_{25}
BE	b_{31}	a_2	b_{33}	b_{34}	a_5
CDE	b_{41}	b_{42}	a_3	a_4	a_5
AE	$\mathbf{a_1}$	b_{52}	$\mathbf{b_{13}}$	b_{54}	a_5

Xét B →C:

	A	B	C	D	E
AD	a_1	b_{12}	b_{13}	a_4	b_{15}
AB	a_1	$\mathbf{a_2}$	$\mathbf{b_{13}}$	b_{24}	b_{25}
BE	b_{31}	$\mathbf{a_2}$	$\mathbf{b_{13}}$	b_{34}	a_5
CDE	b_{41}	b_{42}	a_3	a_4	a_5
AE	a_1	b_{52}	b_{13}	b_{54}	a_5

Xét $C \rightarrow D$

	A	B	C	D	E
AD	a_1	b_{12}	$\mathbf{b_{13}}$	$\mathbf{a_4}$	b_{15}
AB	a_1	a_2	$\mathbf{b_{13}}$	$\mathbf{a_4}$	b_{25}
BE	b_{31}	a_2	$\mathbf{b_{13}}$	$\mathbf{a_4}$	a_5
CDE	b_{41}	b_{42}	a_3	a_4	a_5
AE	a_1	b_{52}	$\mathbf{b_{13}}$	$\mathbf{a_4}$	a_5

Xét $DE \rightarrow C$:

	A	B	C	D	E
AD	a_1	b_{12}	b_{13}	a_4	b_{15}
AB	a_1	a_2	b_{13}	a_4	b_{25}
BE	b_{31}	a_2	$\mathbf{a_3}$	$\mathbf{a_4}$	$\mathbf{a_5}$
CDE	b_{41}	b_{42}	$\mathbf{a_3}$	$\mathbf{a_4}$	$\mathbf{a_5}$
AE	a_1	b_{52}	$\mathbf{a_3}$	$\mathbf{a_4}$	$\mathbf{a_5}$

$DE \rightarrow A$

	A	B	C	D	E
AD	a_1	b_{12}	b_{13}	a_4	b_{15}
AB	a_1	a_2	b_{13}	a_4	b_{25}
BE	$\mathbf{a_1}$	$\mathbf{a_2}$	$\mathbf{a_3}$	$\mathbf{a_4}$	$\mathbf{a_5}$
CDE	$\mathbf{a_1}$	b_{42}	a_3	$\mathbf{a_4}$	$\mathbf{a_5}$
AE	$\mathbf{a_1}$	b_{52}	a_3	$\mathbf{a_4}$	$\mathbf{a_5}$

Từ bảng này ta thấy dòng 3 có bộ (a_1, a_2, a_3, a_4, a_5). Vậy phân tách ρ là bảo toàn thông tin.

Lưu ý:

Trong quá trình thực hiện bước 2 (chưa xong) nếu đã tồn tại một dòng toàn là các giá trị a_i thì ta cũng đã có thể kết luận ngay phân tách ρ là bảo toàn thông tin.

5.3.2. Phân tách bảo toàn phụ thuộc hàm

Định nghĩa: (Phân tách bảo toàn phụ thuộc hàm)

Cho R = <U, F>, $\rho = (R_1, R_2, \ldots, R_k)$ là phân tách trên R với $R_i = \langle U_i, F_i \rangle$ ($i = \overline{1,k}$). Khi đó, ρ được gọi là phân tách bảo toàn phụ thuộc hàm nếu $\bigcup_{i=1}^{k} F_i$ tương đương F (hay $(\bigcup_{i=1}^{k} F_i)^+ = F^+$).

Ví dụ:

Cho R = <U, F>, trong đó: U = ABCD, F = {A $\rightarrow$BCD, C $\rightarrow$D}, và ρ = (R_1, R_2), với R_1 = <CD, {C $\rightarrow$D}>, R_2 = <ABC, {A $\rightarrow$BC}>.

Do G = {C $\rightarrow$D, A $\rightarrow$BC} tương đương F nên ρ là phép phân tách bảo toàn phụ thuộc hàm.

Nếu cho trước R = <U, F>, ρ = (U_1, U_2, …, U_k) thì: việc xác định các F_i ($i = \overline{1,k}$) được thực hiện dựa vào định nghĩa sau:

Định nghĩa: (Chiếu của tập phụ thuộc hàm F trên tập thuộc tính U_i)

Cho R = <U, F> và $U_i \subseteq U$. Khi đó, chiếu của F trên tập thuộc tính U_i, ký hiệu là $\Pi_{U_i}(F)$, được xác định như sau:

$$\Pi_{U_i}(F) = \{X \rightarrow A \in F^+ \mid X, A \subseteq U_i \text{ và } A \not\in X\}$$

Ví dụ:

Cho R = <U,F>, với U = ABCD, F = {A $\rightarrow$D, D $\rightarrow$B, A $\rightarrow$C}

Xét ρ = (ABC, AD) (U_1 = ABC, U_2=AD)

$\Rightarrow$ F_1 = (A C, A B) (chiếu của F lên tập U_1)

và F_2 = {A D} (chiếu của F lên tập U_2)

Nhận xét:

$\Pi_{U_i}(F)$ là bao gồm các phụ thuộc hàm tầm thường và không tầm thường (phụ thuộc hàm tầm thường là phụ thuộc hàm mà vế phải là con của vế trái). Ở đây chúng ta dùng ($A \not\in X$) là để loại bỏ các phụ thuộc hàm tầm thường. Vì vậy việc xác định các F_i ($i = \overline{1,k}$) là dựa vào việc xác định phủ tối thiểu của $\Pi_{U_i}(F)$:

$$F_i = \text{Phủ tối thiểu } (\Pi_{U_i}(F)) \quad (i = \overline{1,k})$$

Thuật toán: Tìm $\Pi_X(F)$

Bước 1:

Tính các X'^+, $\forall X' \subset X$ và $X' \neq \varnothing$

Bước 2:

Xác định các phụ thuộc hàm của $\Pi_X(F)$, dựa vào:

```
Begin
    F' = ∅
    For mỗi X'+ = A1A2...Ak do
        For i = 1 to k do
            If (Ai ∈ X'+) and (Ai ∉ X') then
                F' = F' ∪ {X' → Ai}
    ΠX(F) = F';
End;
```

Ví dụ:

Cho R = <ABCD, {A→D, D→B, A→C}>, và X = ABC

A_F^+ = ABCD $\quad$ $(AB)_F^+$ = ABCD

B_F^+ = B $\quad$ $(AC)_F^+$ = ABCD

C_F^+ = C $\quad$ $(BC)_F^+$ = BC.

$F_1' = \{A \to B, A \to C, AB \to C, AC \to B\}$

Nhận xét:

Các phụ thuộc hàm tầm thường sẽ bị loại bỏ khi xét một phủ tối thiểu với F_i.

Ví dụ: Xét ví dụ trên:

Ta có 1 phủ tối thiểu của F_1' là:

$F_1 = \{A \quad C, A \quad B\}$

Để kiểm tra một phân tách có bảo toàn phụ thuộc hàm hay không, theo phương pháp nêu trên ta phải thực hiện việc xác định các F_i ($i = \overline{1,k}$). Từ đó suy ra G = $\bigcup_{i=1}^{k} F_i$ và việc kiểm tra F tương đương G dựa vào tính chất tương đương sau:

F tương đương G ⇔
G $\quad$ F^+ $\quad$ (1)
F $\quad$ G^+ $\quad$ (2)

Do (1) là hiển nhiên nên ta chỉ cần kiểm tra (2) dựa vào thuật toán sau:

Thuật toán: Kiểm tra F ⊆ G $^+$

Vào: F, G

Ra: Yes/No

Phương pháp:

```
Procedure KiemTra;
Begin
    For mỗi X →Y ∈ F do
        If X_G^+ / Y then

            Begin
                Write('ρ không bảo toàn phụ thuộc hàm');
                Exit;
            End;
    Write('ρ bảo toàn phụ thuộc hàm');
End;
```

Thuật toán: Tính X_G^+ mà không cần phải xác định G

Để có thể tính được X_G^+ mà không cần phải xác định G (không cần xác định các F_i) ta có thể dựa vào thuật toán sau:

Vào: R = <U, F>
$\rho = (R_1, R_2, \ldots, R_n)$
$X \subseteq U$

Ra: X_G^+

Phương pháp:

Procedure Tính X_G^+;
Begin
    Z := X;
    Repeat
        For i:= 1 to n do
            $Z := Z \quad ((Z \cap U_i)^+_F \cap U_i)$
    Until <Z không thay đổi>
    $X_G^+ = Z$;
End;

Ví dụ:

Cho R = <U, F>, U = ABCD, F = {A →B, B →C, C →D, D →A}, và ρ = (AB, BC, CD) có bảo toàn phụ thuộc hàm

- Xét D →A ∈ F, tính D_G^+ như sau:
  1. Z := D
  2. • i := 1; U_1:= AB
     $\Rightarrow Z := Z \cup ((Z \cap U_i)_F^+ \cap U_i) = D \cup ((D \cap AB)_F^+ \cap AB) = D$
     • i := 2; U_2 = BC
     $\Rightarrow Z := D \cup ((D \cap BC)_F^+ \cap BC) = D$
     • i := 3; U_3 = CD
     $\Rightarrow Z := D \cup ((D \cap CD)_F^+ \cap CD) = CD$: Z thay đổi ⇒quay lại
     • i := 1 ⇒Z = CD
     • i := 2 ⇒Z = BCD
     • i := 3 ⇒Z = BCD
     • i := 1 ⇒Z = ABCD
     • i := 2 ⇒Z = ABCD
     • i := 3 ⇒Z = ABCD
     $\Rightarrow \quad D_G^+ = ABCD \Rightarrow D \to A \in G^+$
- Xét A →B ∈ F, tính $A^+_G = ABCD \supseteq B$
- Xét B →C ∈ F, tính $B^+_G = ABCD \supseteq C$
- Xét C →D ∈ F, tính $C^+_G = ABCD \supseteq D$

⇒ Kết luận: ρ bảo toàn phụ thuộc hàm

Lưu ý:

Một phân tách ρ bảo toàn thông tin nhưng có thể không bảo toàn phụ thuộc hàm.

Ví dụ:

R = <U, F> , với U =CSZ, F = {CS →Z, Z →C}

Xét ρ = (ZC, SZ), ρ bảo toàn thông tin (bài tập) nhưng ρ không bảo toàn phụ thuộc hàm.

Một phân tách bảo toàn phụ thuộc hàm nhưng cũng có thể không bảo toàn thông tin.

Ví dụ:

Cho R = <U, F>, với U = ABCD, F = {A →B, C →D}

Xét ρ = (AB, CD), ρ là bảo toàn phụ thuộc hàm vì:

$F_1 = \{A \rightarrow B\}$

$F_2 = \{C \rightarrow D\}$

Nhưng rõ ràng ρ là không bảo toàn thông tin.

Nhận xét:

Một phân tách bảo toàn thông tin nhưng không bảo toàn phụ thuộc hàm thì vẫn có thể sử dụng được. Tuy nhiên điều này sẽ gây phiền phức đối với người lập trình, bởi lẽ việc cập nhật dữ liệu trên một quan hệ $r_i \in R_i$ ($i = \overline{1,k}$) sẽ có thể làm cho quan hệ gốc r không thỏa F. Do đó việc cập nhật thường được thực hiện theo quy trình sau:

- *Bước 1:* Cập nhật dữ liệu trên r_i
- *Bước 2: Tính r*
- *Bước 3:* Nếu r thỏa F thì việc cập nhật được chấp nhận, còn nếu không việc cập nhật sẽ bị huỷ bỏ.

5.4. LÝ THUYẾT CHUẨN HOÁ

Một lược đồ quan hệ quan hệ thiết kế không tốt sẽ gây ra những dị thường dữ liệu như dư thừa dữ liệu và do việc cập nhật dữ liệu. Để tránh dị thường dữ liệu, lược đồ quan hệ cần thiết phải biến đổi thành các dạng phù hợp. Quá trình đó được xem là quá trình chuẩn hóa lược đồ quan hệ.

Lý thuyết chuẩn hoá sẽ xác định các dạng chuẩn (Norm Form) của một lược đồ quan hệ. Đồng thời cho phép xây dựng các thuật toán để phân tách một lược đồ thành các lược đồ con sao cho các lược đồ con đều thuộc một dạng chuẩn nào đó.

5.4.1. Dạng chuẩn 1 (1NF)

Định nghĩa:

Một lược đồ R được gọi là thuộc **dạng chuẩn 1** (ký hiệu: R ∈ 1NF) nếu miền giá trị của các thuộc tính trong R chỉ chứa những giá trị nguyên tố (không thể phân chia được), hay các thuộc tính này đều là các giá trị đơn và đơn trị.

Ngược với thuộc tính đơn là thuộc tính phức hợp (đây là thuộc tính được xây dựng từ nhiều thuộc tính khác).

Ngược với thuộc tính đơn trị là thuộc tính đa trị (đây là thuộc tính mà giá trị của nó là một tập hợp).

Quy ước:

Tất cả các lược đồ quan hệ nếu không nói rõ nó không là 1NF thì xem như nó thuộc 1NF.

5.4.2. Dạng chuẩn 2 (2NF)

Định nghĩa: (Thuộc tính khoá)

Cho R= <U, F>. Khi đó, thuộc tính A được gọi là **thuộc tính khoá** nếu A thuộc một khoá nào đó của R. Ngược lại, A được gọi là **thuộc tính không khoá.**

Ví dụ:

1) R = <U, F>, với U = ABCD, F = {AB $\rightarrow$C, C $\rightarrow$D} $\Rightarrow$tập các khóa K của R là K = {AB} $\Rightarrow$R có hai thuộc tính khoá: A và B, và R có hai thuộc tính không khoá: C và D.

2) R = <U, F>, với U = ABCD, F = {AB $\rightarrow$C, B $\rightarrow$D, BC $\rightarrow$A}
$\Rightarrow$ tập các khóa K của R là K = {AB, BC} $\Rightarrow$R có ba thuộc tính khoá: A, B và C, còn D là thuộc tính không khoá.

Định nghĩa: (Phụ thuộc hàm đầy đủ)

Cho R = <U, F>. Khi đó, $X \rightarrow Y \in F$ được gọi là một **phụ thuộc hàm đầy đủ** (đọc là: Y phụ thuộc hàm đầy đủ vào X) nếu:

$$\nexists Z \subset X \text{ sao cho } Z \rightarrow Y \in F^+$$

Ví dụ:

NKBH = <U, F>

với U = {NG, SP, MH, ĐG, SL}, F = {{SP, MH}$\rightarrow$U, MH$\rightarrow$ĐG}

$\Rightarrow$\{SP, MH\} $\rightarrow$ĐG $\in F^+$ không là phụ thuộc hàm đầy đủ (vì $\exists$ MH $\subset$ {MH, SP}: MH $\rightarrow$ĐG $\in F^+$)

Nhận xét:

$X \rightarrow U \in F^+$ là một phụ thuộc hàm đầy đủ khi X là khoá của R.

Định nghĩa: (2NF)

Cho R = <U, F>, R được gọi là thuộc **dạng chuẩn 2** (ký hiệu R $\in$ 2NF) nếu với mọi thuộc tính không khoá của R là phụ thuộc hàm đầy đủ vào mọi khoá của R.

Như vậy:

- $R \in$ 2NF nếu và chỉ nếu $\forall A$ là thuộc tính không khoá, $\forall X \in$ thuộc tập các khoá của R thì $X \rightarrow A \in F^+$ là phụ thuộc hàm đầy đủ.
- $R \notin$ 2NF nếu và chỉ nếu $\exists$ A là thuộc tính không khoá và $\exists X \in$ thuộc tập các khóa của R sao cho $X \rightarrow A \in F^+$ không là phụ thuộc hàm đầy đủ.

Các ví dụ:

Ví dụ 1: Xét ví dụ trên, lược đồ NKBH có tập các khoá là K = {SP, MH}, do đó hai thuộc tính khoá là SP và MH. Suy ra các thuộc tính không khoá: NG, ĐG, SL.

Theo trên ta có : $\{SP, MH\} \rightarrow ĐG \in F^+$ không là phụ thuộc hàm đầy đủ vì $MH \rightarrow ĐG \Rightarrow NKBH \notin$ 2NF.

Ví dụ 2: Cho R = <SAIP, $\{SI \rightarrow P, S \rightarrow A\}$>. Lúc đó $R \notin$ 2NF.

Thật vậy, R chỉ có khóa SI nên A là thuộc tính không khóa, mặt khác A không phụ thuộc hàm đầy đủ vào khóa SI vì $S \rightarrow A$.

Ví dụ 3: Cho R = <SAIP, $\{SI \rightarrow P\}$>. $R \in$ 2NF.

Thật vậy, R chỉ có khóa SI nên A, P là các thuộc tính không khóa, mặt khác A và P phụ thuộc hàm đầy đủ vào khóa SI vì không có tập con nào của SI xác định hàm A và P.

Thuật toán: Kiểm tra R thuộc 2NF

Vào: R = <U, F>

Ra: Yes/No

Phương pháp:

```
Procedure    KiemTra2NF (R);
Begin
    For mỗi A không phải là thuộc tính khoá do
        For mỗi X ∈ K do                {K là tập các khóa của R}
            If  <X →A không là phụ thuộc hàm đầy đủ> Then
                Return No;
    Return Yes;
End;
```

Nhận xét:

1. Nếu mọi khoá của lược đồ quan hệ R chỉ có một thuộc tính thì $R \in$ 2NF. Hay $(\forall X \in K: |X| = 1) \Rightarrow R \in$ 2NF.

2. Lược đồ quan hệ R 2NF $\Leftrightarrow \forall X \rightarrow A \in F^+$, với $A \notin X$ và $X \subset K$ (K là khoá của R) thì A là thuộc tính khoá. (bài tập)

5.4.3. Dạng chuẩn 3 (3NF)

Ví dụ:

Cho NKBH = <U, F>, với U = {STT, NGAY, MH, TH, ĐG, SL},
F = {STT→U, MH→TH, MH→ĐG}

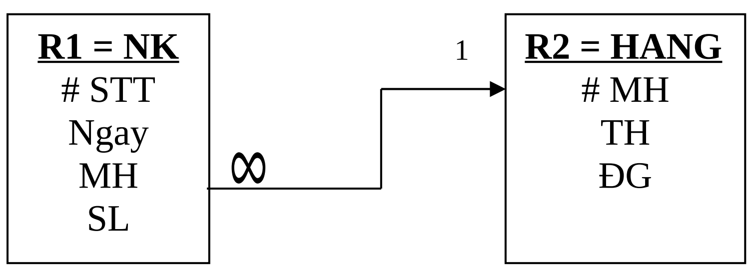
$\Rightarrow$ NKBH $\in$ 2NF (do lược đồ có một khoá duy nhất là STT chỉ có một thuộc tính) nhưng vẫn tồn tại dư thừa dữ liệu trong lược đồ này.

STT	Ngay	MH	TH	ĐG	SL
1	01/01/01	A1	Tiêu	50	10
2	02/01/01	A2	Cafe	20	20
3	03/01/01	A1	Tiêu	50	30
4	04/01/01	A2	Cafe	20	40

$\Rightarrow$ Phân tách NKBH thành 2 lược đồ con:

HANG = $<U_1, F_1>$, với U_1 = {MH, TH, ĐG} và $F_1 = \{MH \rightarrow U_1\}$

NK = $<U_2, F_2>$, với U_2 = {STT, NGAY, MH, SL} và $F_2 = \{STT \rightarrow U_2\}$



Ta sẽ chứng minh rằng:

- NKBH $\notin$ 3NF
- HANG, NK $\in$ 3NF

Định nghĩa: (R $\in$ 3NF)

Cho R = <U, F>. Khi đó R được gọi là thuộc **dạng chuẩn 3** (ký hiệu $R \in$ 3NF) nếu $\forall X \rightarrow A \in F^+$ với $A \notin X$ thì:

- hoặc X là siêu khoá
- hoặc A là thuộc tính khoá

Từ định nghĩa suy ra:

$R \notin$ 3NF $\Leftrightarrow \exists X \rightarrow A \in F^+$ với $A \notin X$ sao cho thỏa hai điều kiện sau:

1. X không là siêu khoá
2. A là thuộc tính không khoá

Ví dụ 1:

Xét ví dụ trên, NKBH = <U, F>, với

U = {STT, NGAY, MH, TH, ĐG, SL},

F = {STT$\rightarrow$U, MH$\rightarrow$TH, MH$\rightarrow$ĐG}

NKBH chỉ có một khóa STT và dễ dàng thấy NKBH $\notin$ 3NF bởi vì tồn tại phụ thuộc hàm MH $\rightarrow$ TH $\in F^+$ mà MH không là siêu khoá, TH cũng không phải là thuộc tính khoá.

Xem hai lược đồ con của NKBH là HANG và NK:

HANG = $<U_1, F_1>$, với U_1 = {MH, TH, ĐG} và $F_1 = \{MH \rightarrow U_1\}$

NK = $<U_2, F_2>$, với U_2 = {STT, NGAY, MH, SL} và $F_2 = \{STT \rightarrow U_2\}$

Rõ ràng HANG có duy nhất khóa là MH và NK có duy nhất khóa là STT, từ đó dễ thấy HANG và NK $\in$ 3NF.

Ví dụ 2:

Cho R = <CSZ, {CS $\rightarrow$ Z, Z $\rightarrow$ C}>. Lược đồ này có hai khóa là CS và SZ. Vậy R không có thuộc tính không khóa nên R $\in$ 3NF.

Ví dụ 3:

Cho R = <SIDM, {SI $\rightarrow$ D, SD $\rightarrow$ M}>. Lược đồ này chỉ có một khóa là SI, D, M là các thuộc tính không khóa. R $\notin$ 3NF vì $\exists$ SD $\rightarrow$ M.

Chú ý: Nếu R không chứa thuộc tính không khóa thì R $\in$ 3NF.

Thuật toán: Kiểm tra 3NF

Vào: R = <U, F>

Ra: Yes/No

Phương pháp:

```
Procedure    KiemTra3NF(R);
Begin
        For mỗi X không là siêu khoá của R do
                If  <X⁺ ⊇ A là thuộc tính không khoá và A ∉ X> then
                        Return No; {R không thuộc 3NF}
        Return Yes; {R thuộc 3NF}
End;
```

Nhận xét

Nếu R $\in$ 3NF thì R $\in$ 2NF

Nếu R $\in$ 2NF thì R $\in$ 1NF

Ta có thể chứng minh rằng: R $\notin$ 3NF $\Leftrightarrow \exists$ X $\rightarrow$ Y $\in F^+$ và Y $\rightarrow$ A $\in F^+$ với:

- X là khoá
- Y không là siêu khoá
- A là thuộc tính không khoá và $A \notin Y$

Hay:

$R \notin$ 3NF $\Leftrightarrow \exists$ X là khoá Y, $Y \rightarrow A$ và A không là thuộc tính khoá, $A \notin Y$

Ví dụ:

Xét ví dụ trên: NKBH = <U, F>, với

U = {STT, NGAY, MH, TH, ĐG, SL},

F = {STT→U, MH→TH, MH→ĐG}

NKBH $\notin$ 3NF, vì: $\exists${STT}_khoá MH →TH_không khoá.

Thuật toán 1: Phân tách thành các lược đồ con 3NF bảo toàn thông tin

Vào: R = <U, F>

Ra: $\rho = (R_1, R_2, \ldots, R_k)$ với $R_i \in$ 3NF ($i = \overline{1,k}$) và ρ là bảo toàn thông tin.

Phương pháp:

Bước 1: Kiểm tra $R \in$ 3NF

- Nếu $R \in$ 3NF: không phân tách và dừng
- Nếu $R \notin$ 3NF thì $\exists$ X_khoá Y, mà Y →A_không khoá, $A \notin Y$. Phân tách R thành 2 lược đồ con: ρ = (YA, U\A)

Bước 2: Kiểm tra các lược đồ con

Kiểm tra lần lượt các lượt các lược đồ con có thuộc 3NF không, nếu không thuộc thì lại phân tách tiếp (theo quy tắc chỉ ra trong bước 1) cho đến khi nào tất cả các lược đồ con đều thuộc dạng chuẩn 3NF. Bấy giờ chúng ta sẽ có một cây phân tách (cây nhị phân) mà các nút lá là các lược đồ con thuộc chuẩn 3NF. Đó là kết quả cần tìm.

```
Procedure    PhanTach(U, F);
Begin
    If (∃ X →A, X_không siêu khoá, A_không khoá, A ∉ X) then
        begin
            PhanTach(XA, F1);
            PhanTach(U\A, F2);
        end;
End;
```

Lưu ý:

Nếu phân tách $\rho = (U_1, \ldots, U_n)$ tồn tại U_i là tập con của U_j ($i \neq j$) thì ta loại bỏ lược đồ tương ứng với tập thuộc tính U_i.

Ví dụ:

Xét ví dụ trên: $\exists$ {STT} $\to$ {MH, TH} $\to$ ĐG

$\Rightarrow \rho$ = ({MH, TH, ĐG}, {STT, NGAY, MH, TH, SL})

$\in$ 3NF $\quad\quad$ $\notin$ 3NF, do $\exists$ STT / MH $\to$ TH

{MH,TH} $\quad\quad$ {STT,NGAY,MH,SL}

$\in$ 3NF $\quad\quad$ $\in$ 3NF

$\Rightarrow \rho$ = ({MH,TH, ĐG},{STT,NGAY,MH,SL})

Lưu ý:

Thuật toán trên không duy nhất

Ví dụ: Xét ví dụ trên

$\exists$ {STT} / {MH} $\to$ TH.

$\Rightarrow \rho$ = ({MH, TH} , {STT, NGAY, MH, ĐG, SL})

$\in$ 3NF $\quad\quad$ $\notin$ 3NF, do: $\exists$ STT / MH $\to$ ĐG

{MH, ĐG} $\quad\quad$ {STT,NGAY,MH,SL}

$\in$ 3NF $\quad\quad$ $\in$ 3NF

$\Rightarrow \rho$ = ({MH, TH}, {MH, ĐG}, {STT, NGAY, MH, SL})

Lưu ý

Ta có thể gộp các lược đồ có dạng như sau để tạo thành một lược đồ mà vẫn bảo toàn thông tin và phụ thuộc hàm.

$$\left.\begin{array}{l} R_1 = <AA_1, \{A \to A_1\}> \\ R_2 = <AA_2, \{A \to A_2\}> \\ \ldots \\ R_k = <AA_k, \{A \to A_k\}> \end{array}\right\} \Leftrightarrow R = <AA_1A_2 \ldots A_k, \{A \to A_1A_2 \ldots A_k\}> \in 3NF$$

Thuật toán 2: Phân tách thành các lược đồ con 3NF bảo toàn thông tin và bảo toàn phụ thuộc hàm

Vào: R = <U, F>

Ra: $\rho = (R_0, R_1, R_2, \ldots, R_k)$ với R_i 3NF ($i = \overline{0,k}$), ρ là bảo toàn thông tin và bảo toàn phụ thuộc hàm.

Phương pháp:

{Phân tách thành các lược đồ con 3NF bảo toàn thông tin và bảo toàn phụ thuộc hàm}

Bước 1: Xác định phủ tối thiểu của F:

$F' = \{X_i \rightarrow A_i \mid i = \overline{1,m}\}$

Bước 2: Tìm một khoá X bất kì của R.

Bước 3: Xác định các lược đồ con $R_0 = <U_0, F_0>$ với $U_0 = X$

$R_i = <U_i, F_i>$ với $U_i = X_iA_i$ $(i = \overline{1,m})$

Lưu ý

- Nếu $\exists i \neq j$ mà $U_i \subseteq U_j$ $(i, j = \overline{0,m})$ thì loại bỏ R_i. Quá trình này sẽ tiếp tục cho đến khi không thể loại bỏ được một R_i nào nữa.
- Chúng ta có thể gộp các lược đồ như trong phần 5.4.3.9. Lưu ý

Ví dụ

Cho $R = <U, F>$, với U = ABCD, $F = \{A\rightarrow B, B\rightarrow C, CD\rightarrow A, AC\rightarrow D\}$

Bước 1: Ta có một phủ tối thiểu của F là:

$F = \{A\rightarrow B, B\rightarrow C, CD\rightarrow A, A\rightarrow D\}$

Bước 2: Ta có A là một khoá của R.

Bước 3: $R_0 = <U_0, F_0>$ với $U_0 = A$, $F_0 = \varnothing$

$R_1 = <AB, \{A\rightarrow B\}>$

$R_2 = <BC, \{B\rightarrow C\}>$

$R_3 = <ACD, \{CD\rightarrow A, A\rightarrow D, A\rightarrow C\}>$

$R_4 = <AD, \{A\rightarrow D\}>$

Loại R_0 và R_4

Kết luận: ρ = (AB, BC, ACD)

Lưu ý:

Thuật toán không duy nhất, bởi vì:

- Do cách tìm một khoá ban đầu có thể khác nhau.
- Do cách xác định phủ tối thiểu là không duy nhất.

5.4.4. Dạng chuẩn BCNF (Boyce - Codd Normal Form)

Ví dụ

Cho $R = <U, F>$, với U = CSZ và $F = \{CS\rightarrow Z, Z\rightarrow C\}$. Như đã thấy ở trên, $R \in$ 3NNF, tuy nhiên lược đồ này vẫn còn dư thừa dữ liệu, chẳng hạn xét quan hệ $r \in R$ như sau:

r =

C	S	Z
VN	Hue	84
VN	HN	84
Mỹ	A	1
Mỹ	B	1
Mỹ	C	2
Mỹ	D	2
X	A	99

$\Rightarrow$ Dư thừa

Tách thành lược đồ con:

C	Z
VN	84
Mỹ	1
Mỹ	2
X	99

$\Downarrow$ Không dư thừa

$\bowtie$

S	Z
Hue	84
HN	84
A	1
B	1
C	2
D	2
A	99

= r

$\Rightarrow$ Không dư thừa

Ta thấy $R \in$ 3NF (vì R có 2 khoá: CS và SZ), nhưng R vẫn dư thừa. Ta sẽ chứng minh $R \notin$ BCNF.

Định nghĩa: (Dạng chuẩn BCNF)

Cho R = <U, F>. Khi đó R thuộc dạng chuẩn BCNF (ký hiệu $R \in$ BCNF) nếu với mọi $X \rightarrow A \in F^+$ với $A \notin X$ thì X là siêu khoá của R.

Nhận xét:

- $R \in$ BCNF $\Rightarrow R \in$ 3NF
- $R \notin$ BCNF $\Leftrightarrow \exists X \rightarrow A \in F^+$ với $A \notin X$ và X không là siêu khoá.

Ví dụ:

Cho R = <U, F>, với U = CSZ và F = {$CS \rightarrow Z$, $Z \rightarrow C$}, $R \notin$ BCNF vì $\exists\ Z \rightarrow C \in F^+$ mà vế trái (Z) không phải là siêu khoá.

Thuật toán: Kiểm tra một lược đồ quan hệ có thuộc BCNF

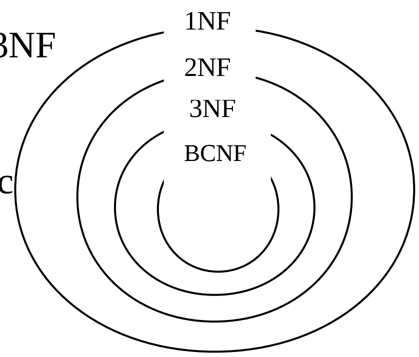
Vào: R = <U, F>

Ra: Yes/No

Phương pháp:

```
Procedure   KiemTraBCNF(U, F);
Begin
        For <mỗi X không là siêu khoá> do
                If X^+_F ≠ X then
                        Return No;
        Return Yes;
End;
```

Nhận xét:



- Tất cả các dạng chuẩn: 1NF, 2NF, 3NF đều có thể dư thừa dữ liệu.
- $R \notin BCNF \Leftrightarrow \exists$ dư thừa dữ liệu.
- Khi $R \in BCNF$, tất cả các phụ thuộc dữ liệu đều là các ràng buộc về khoá.

Thuật toán 1 (Phân tách R thành các lược đồ thỏa BCNF và bảo toàn thông tin)

Phải tìm chiếu của các phụ thuộc hàm trên các lược đồ con.

Vào: $R = <U, F>$

Ra: $\rho = (R_0, R_1, R_2, \ldots, R_k)$ bảo toàn thông tin và $R_i \in BCNF$

Phương pháp: xây dựng cây phân tách theo các thủ tục sau:

```
Procedure    BCNF_PTH (U, F);
Begin
        If <∃ X→A ∈ F+ với A ∉ X với X  không là siêu khoá> Then
                begin
                        BCNF_PTH(XA, ∏XA(F));
                        BCNF_PTH(U\A, ∏U\A(F));
                end;
End;
```

Ví dụ:

Cho $R = <U, F>$, với $U = CSZ$ và $F = \{CS \rightarrow Z, Z \rightarrow C\}$.

$R \notin BCNF$ vì $Z \rightarrow C \in F^+$, Z không là siêu khoá (tập khoá: {CS, ZS}). Do vậy ta phân tách R thành 2 lược đồ con như sau:

R

$R_1 = <ZC, \{Z \rightarrow C\}>$ $\Rightarrow R_1 \in BCNF$

$R_2 = <SZ, \varnothing>$ $\Rightarrow R_2 \in BCNF$

Nhận xét:

Ưu điểm: việc phân tách các lược đồ thành các lược đồ con BCNF là dừng.

Nhược điểm: có độ phức tạp hàm mũ (đó là số thuộc tính của mỗi lược đồ con).

Bổ đề 1:

Mọi lược đồ có hai thuộc tính đều thuộc dạng chuẩn BCNF. Điều này có nghĩa: $|U| = 2 \Rightarrow R = <U, F> \in$ BCNF.

Chứng minh:

Giả sử R = <AB, F>, xet các trường hợp sau:

$$\left.\begin{array}{l} F = \phi \\ F = \{A \rightarrow B\} \\ F = \{B \rightarrow A\} \\ F = \{A \rightarrow B, B \rightarrow A\} \end{array}\right\} \Rightarrow R = <U, F> \in BCNF$$

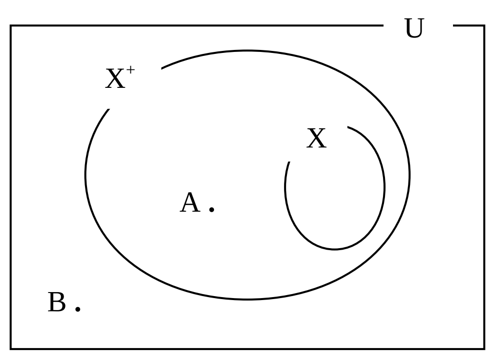
Bổ đề 2:

Nếu $R \notin$ BCNF thì $\exists$ A, B $\in$ U (A $\neq$ B): $U \backslash AB \rightarrow A \in F^+$. Tức là: nếu $\nexists$ A, B $\in$ U (A $\neq$ B): $U \backslash AB \rightarrow A \in F^+$ thì $R \in$ BCNF.

Chứng minh: bằng phương pháp phản chứng

Giả sử $R \notin$ BCNF $\Rightarrow \exists$ $X \rightarrow A \in F^+$: $A \notin X$ và X không là siêu khoá ($X^+_F \neq U$).

Do $X^+ \neq U \Rightarrow \exists$ $B \in U$: $B \notin X^+$ $(A \neq B) \Rightarrow B \notin X \Rightarrow X \subseteq U \backslash AB$. $\Rightarrow$ $U \backslash AB \rightarrow X \in F^+$. Ngoài ra: $X \rightarrow A \in F^+ \Rightarrow U \backslash AB \rightarrow A \in F^+$ (mâu thuẫn) $\Rightarrow$ điều phải chứng minh.



Dựa vào hai bổ đề ở trên ta có thuật toán sau đây:

Thuật toán 2: Phân tách R thành các lược đồ thỏa BCNF và bảo toàn thông tin (không cần tính chiếu của F trên các lược đồ con).

Vào: R = <U, F>

Ra: $\rho = (R_0, R_1, R_2, \dots, R_k)$ bảo toàn thông tin và $R_i \in$ BCNF

Phương pháp:

```
Procedure    BCNF_PTH(U, F);
Begin
        ρ:= ∅;
        Z:= U;
        Repeat
                PhanTach(Z, Y, A’);{nhằm xác định Y và A’ để phân tách Z
        thành hai lược đồ con Y ∈ BCNF và Z \ A’};
                ρ := ρ ∪{Y};
                Z := Z \ A’ ;
        Until ∄ A, B ∈ Z (A ≠ B): A∈(Z-AB)^+;
End;
```

Trong đó:

```
Procedure    PhanTach(Z, Y, A’);
Begin
        Y:= Z;
        While ∃ A, B ∈ Y: A ≠ B và A ∈ (Y-AB)^+ do
                begin
                        Y:= Y - B;
                        A’:= A;
                end;
End;
```

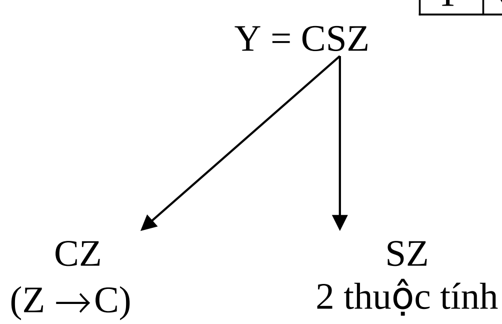
Ví dụ

Cho R = <U, F>, với U = CSZ và F = {$CS \to Z$, $Z \to C$}

Y= CSZ

$\exists\ Z \to C \in F^+$

A	C
B	S
Y	CZ



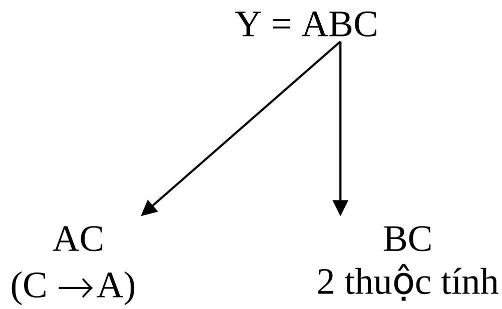
$\Rightarrow \rho$ = (CZ, SZ) là phân tách cần tìm.

Nhận xét

Ưu điểm: thuật toán này có độ phức tạp tính toán là đa thức $O(n^2m)$ với n = |U|, m = |F|.

Nhược điểm: Đôi khi lại phân tách một lược đồ quan hệ đã ở dạng chuẩn BCNF.

Ví dụ 1: Cho R = <ABC , F = {C →A, C →B}>



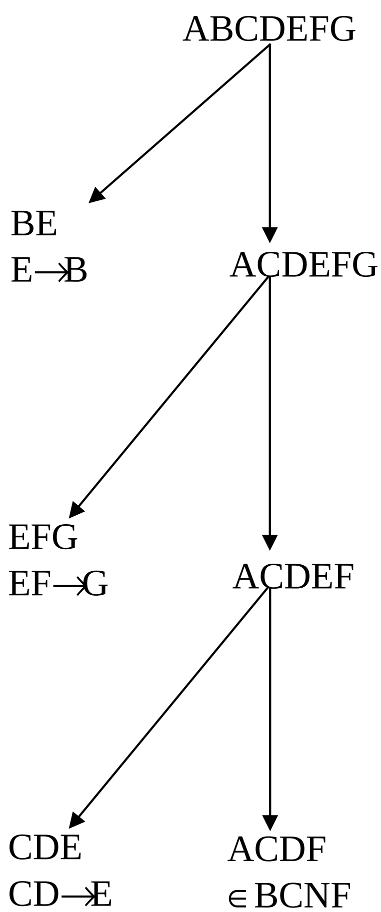
A	A
B	B
Y	AC

C^+ = CAB

ρ = (AC, BC) là phân tách cần tìm.

Ví dụ 2:

Cho R = <ABCDEFG, {AB →D, CD →E, E →B, EF →G}>



AB B B B BBA
C D F GYBCDEFG
BDEFG BEFG BEG BEBCDEFG
= $(CDEFG)^+$

AG G GBA
C DYCDEFG
DEFG EFG

AE EBA
FYCDEF CDE

⇒ρ = (BD, EFG, CDE, ACDF).

Nhận xét

Cả hai thuật toán trên đều có thể cho phân tách ρ không duy nhất.

Rõ ràng không tồn tại một thuật toán phân tách thành BCNF vừa bảo toàn thông tin, vừa bảo toàn phụ thuộc hàm.

TÀI LIỆU THAM KHẢO

[1]. Ullman, J.D., *Nguyên lý các hệ cơ sở dữ liệu và hệ tri thức - tập* 1, 2, 3, Trần Đức Quang biên dịch, Nhà xuất bản Thống kê, 1999.

[2]. Hồ Thuần, Hồ Cẩm Hà, *Các hệ cơ sở dữ liệu – Lí thuyết và thực hành - tập 1, 2*, NXB Giáo dục, 2004

[3]. Nguyễn Kim Anh, *Nguyên lý các hệ cơ sở dữ liệu*, NXB Đại học Quốc gia Hà Nội, 2004

[4]. Đỗ Trung Tuấn, *Cơ sở dữ liệu*, NXB Giáo dục, 1998.

[5]. Nguyễn Tuệ, *Giáo trình Cơ sở dữ liệu*, Đại học Quốc gia Hà Nội, 1999.

[6]. Nguyễn Bá Tường, *Cơ sở dữ liệu - Lý thuyết và thực hành*, NXB Khoa học kỹ thuật, 2001.

[7]. Lê Tiến Vương, *Nhập môn cơ sở dữ liệu quan hệ*, NXB Khoa học kỹ thuật, 1996.

[8]. Anstey, D., E. Bertino, E. Marcros, *Advanced Database Technology and Design*, Artech House, 2000.

[9]. Communication of the ACM, *Special Issue on Next-Generation Database Systems*, Vol. 34, No. 10, 1991.

[10]. Date, C. J., *An Introduction to Database Systems*, Wesley, 1994.

[11]. Elmasri, R., S. B. Navathe, *Fundamentals of Database Systems*, 5th Edition, Addison Wesley, 2006.

MỤC LỤC