BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC NAM CẦN THƠ

GIÁO TRÌNH

CƠ SỞ DỮ LIỆU

Ths. Võ Văn Phúc

CẦN THƠ 2017

LỜI MỞ ĐẦU

Cơ sở dữ liệu là một công cụ thu thập và sắp xếp thông tin. Cơ sở dữ liệu có thể lưu trữ thông tin về con người, sản phẩm, đơn hàng, hoặc bất kỳ điều gì khác. Nhiều cơ sở dữ liệu bắt đầu dưới dạng một danh sách trong chương trình xử lý văn bản hoặc bảng tính. Khi danh sách trở nên lớn hơn, sự dư thừa và không nhất quán bắt đầu xuất hiện trong dữ liệu. Dữ liệu trở nên khó hiểu ở dạng danh sách và có ít cách thức tìm kiếm hoặc kéo tập con của dữ liệu ra để xem lại. Sau khi các sự cố này bắt đầu xuất hiện, một ý hay là truyền dữ liệu vào cơ sở dữ liệu được tạo bởi một hệ thống quản lý cơ sở dữ liệu (DBMS).

Ngày nay, cơ sở dữ liệu đã có nhiều ứng dụng trong mọi hoạt động của xã hội. Muốn thiết kế và sử dụng cơ sở dữ liệu chúng ta phải nắm được các kỹ thuật cơ bản của cơ sở dữ liệu.

Giáo trình này nhằm trình bày các kỹ thuật cơ sở của cơ sở dữ liệu truyền thống, đó là mô hình liên kết thực thể, mô hình cơ sở dữ liệu quan hệ.

Giáo trình cũng trình bày cách thiết kế một cơ sở dữ liệu quan hệ, cách sử dụng các phép toán đại số quan hệ để tạo, cập nhật và truy vấn cơ sở dữ liệu và khái niệm phụ thuộc hàm ứng dụng trong lí thuyết thiết kế và chuẩn hóa cơ sở dữ liệu quan hệ.

Giáo trình cần thiết cho tất cả các đối tượng là sinh viên đại học, người muốn tìm hiểu và thiết kế các cơ sở dữ liệu quan hệ ứng dụng trong công tác quản lý.

- Tên môn học: **Cơ sở dữ liệu.**

- Mã số môn học:

- Thời gian: Lý thuyết 30 tiết.

- Mục tiêu: Trang bị các kiến thức cơ bản về Cơ sở dữ liệu.

- Những kiến thức cần phải được trang bị trước khi học: không.

- Nội dung:

Chương 1: CÁC KHÁI NIỆM CƠ BẢN VỀ CƠ SỞ DỮ LIỆU

Chương 2: MÔ HÌNH THỰC THỂ LIÊN KẾT

Chương 3: MÔ HÌNH QUAN HỆ

Chương 4: ĐẠI SỐ QUAN HỆ

Chương 5: NGÔN NGỮ TRUY VẤN SQL

Chương 6: RÀNG BUỘC TOÀN VẸN

Chương 7: PHỤ THUỘC HÀM VÀ CHUẨN HÓA CSDL QUAN HỆ, CÁC THUẬT TOÁN THIẾT KẾ CSDL QUAN HỆ

MỤC LỤC

CHƯƠNG 1. CÁC KHÁI NIỆM CƠ BẢN VỀ HỆ CƠ SỞ DỮ LIỆU

Các cơ sở dữ liệu và các hệ cơ sở dữ liệu đã trở thành một thành phần chủ yếu trong cuộc sống hàng ngày của xã hội hiện đại. Trong vòng một ngày con người có thể có nhiều hoạt động cần có sự giao tiếp với cơ sở dữ liệu như: đến ngân hàng để rút tiền và gửi tiền, đăng ký chỗ trên máy bay hoặc khách sạn, truy cập vào thư viện đã tin học hoá để tìm sách báo, đặt mua tạp chí ở một nhà xuất bản… Tại các ngân hàng, các cửa hàng, người ta cũng cập nhật tự động việc quản lý tiền bạc, hàng hoá.

Tất cả các giao tiếp như trên được gọi là các ứng dụng của cơ sở dữ liệu truyền thống. Trong các cơ sở dữ liệu truyền thống, hầu hết các thông tin được lưu giữ và truy cập là văn bản hoặc số. Những năm gần đây, những tiến bộ về kỹ thuật đã đưa đến những ứng dụng mới của cơ sở dữ liệu. Các cơ sở dữ liệu đa phương tiện bây giờ có thể lưu trữ hình ảnh, phim và tiếng nói. Các hệ thống thông tin địa lý có thể lưu trữ và phân tích các bản đồ, các dữ liệu về thời tiết và các ảnh vệ tinh. Kho dữ liệu và các hệ thống phân tích trực tuyến được sử dụng trong nhiều công ty để lấy ra và phân tích những thông tin có lợi từ các cơ sở dữ liệu rất lớn nhằm đưa ra các quyết định. Các kỹ thuật cơ sở dữ liệu động và thời gian thực được sử dụng trong việc kiểm tra các tiến trình công nghiệp và sản xuất. Các kỹ thuật tìm kiếm cơ sở dữ liệu đang được áp dụng cho World Wide Web để cung cấp việc tìm kiếm các thông tin cần thiết cho người sử dụng bằng cách duyệt qua Internet.

Để hiểu được các cơ sở kỹ thuật của cơ sở dữ liệu chúng ta phải bắt đầu từ các cơ sở kỹ thuật của cơ sở dữ liệu truyền thống. Mục đích của giáo trình này là nghiên cứu các cơ sở kỹ thuật đó. Trong chương này chúng ta sẽ định nghĩa cơ sở dữ liệu, hệ quản trị cơ sở dữ liệu, mô hình cơ sở dữ liệu và các thuật ngữ cơ bản khác.

2. Cơ sở dữ liệu

2.1- Định nghĩa cơ sở dữ liệu

Cơ sở dữ liệu và kỹ thuật cơ sở dữ liệu đã có ảnh hưởng rất lớn đến việc sử dụng máy tính. Có thể nói rằng cơ sở dữ liệu đóng vai trò quan trọng trong mọi

lĩnh vực có sử dụng máy tính như giáo dục, thương mại, kỹ nghệ, khoa học, thư viện,…. Thuật ngữ cơ sở dữ liệu trở thành một thuật ngữ phổ dụng.

Một ***cơ sở dữ liệu*** là một tập hợp các dữ liệu có liên quan với nhau, được lưu trữ trên máy tính, có nhiều người sử dụng và được tổ chức theo một mô hình. ***Dữ liệu*** là những sự kiện có thể ghi lại được và có ý nghĩa.

Ví dụ, để quản lý việc học tập trong một môi trường đại học, các dữ liệu là các thông tin về sinh viên, về các môn học, điểm thi….Chúng ta tổ chức các dữ liệu đó thành các bảng và lưu giữ chúng vào sổ sách hoặc sử dụng một phần mềm máy tính để lưu giữ chúng trên máy tính. Ta có một tập các dữ liệu có liên quan đến nhau và mang nhiều ý nghĩa, đó là một cơ sở dữ liệu.

2.2- Các tính chất của một cơ sở dữ liệu

Một cơ sở dữ liệu có các tính chất sau:

1. Một cơ sở dữ liệu biểu thị một khía cạnh nào đó của thế giới thực như hoạt động của một công ty, một nhà trường, một ngân hàng… Những thay đổi của thế giới thực phải được phản ánh một cách trung thực vào trong cơ sở dữ liệu. Những thông tin được đưa vào trong cơ sở dữ liệu tạo thành một không gian cơ sở dữ liệu hoặc là một "thế giới nhỏ" (miniworld) .

2. Một cơ sở dữ liệu là một tập hợp dữ liệu liên kết với nhau một cách logic và mang một ý nghĩa cố hữu nào đó. Một cơ sở dữ liệu không phải là một tập hợp tuỳ tiện.

3. Một cơ sở dữ liệu được thiết kế và được phổ biến cho một mục đích riêng. Nó có một nhóm người sử dụng có chủ định và có một số ứng dụng được xác định phù hợp với mối quan tâm của người sử dụng. Nói cách khác, một cơ sở dữ liệu có một nguồn cung cấp dữ liệu, một mức độ tương tác với các sự kiện trong thế giới thực và một nhóm người quan tâm tích cực đến các nội dung của nó.

Một cơ sở dữ liệu có thể có cỡ tuỳ ý và có độ phức tạp thay đổi. Có những cơ sở dữ liệu chỉ gồm vài trăm bản ghi (như cơ sở dữ liệu phục vụ việc quản lý lương ở một cơ quan nhỏ), và có những cơ sở dữ liệu có dung lượng rất lớn (như các cơ sở dữ liệu phục vụ cho việc tính cước điện thoại, quản lý nhân sự trên một phạm vi lớn). Các cơ sở dữ liệu phải được tổ chức quản lý sao cho những người sử dụng có thể tìm kiếm dữ liệu, cập nhật dữ liệu và lấy dữ liệu ra khi cần thiết. Một cơ sở dữ liệu có thể được tạo ra và duy trì một cách thủ công và cũng có thể được tin học

hoá. Một cơ sở dữ liệu tin học hoá được tạo ra và duy trì bằng bằng một nhóm chương trình ứng dụng hoặc bằng một *hệ quản trị cơ sở dữ liệu.*

3. Hệ quản trị cơ sở dữ liệu

3.1- Định nghĩa hệ quản trị cơ sở dữ liệu

Một ***hệ quản trị cơ sở dữ liệu*** là một tập hợp chương trình giúp cho người sử dụng tạo ra, duy trì và khai thác một cơ sở dữ liệu. Nó là một hệ thống phần mềm phổ dụng, làm dễ quá trình định nghĩa, xây dựng và thao tác cơ sở dữ liệu cho các ứng dụng khác nhau.

Định nghĩa một cơ sở dữ liệu bao gồm việc đặc tả các kiểu dữ liệu, các cấu trúc và các ràng buộc cho các dữ liệu sẽ được lưu trữ trong cơ sở.

Xây dựng một cơ sở dữ liệu là quá trình lưu trữ các dữ liệu trên các phương tiện lưu trữ được hệ quản trị cơ sở dữ liệu kiểm soát.

Thao tác một cơ sở dữ liệu bao gồm các chức năng như truy vấn cơ sở dữ liệu để lấy ra các dữ liệu cụ thể, cập nhật cơ sở dữ liệu để phản ánh các thay đổi trong thế giới nhỏ và tạo ra các báo cáo từ các dữ liệu.

Các hệ quản trị cơ sở dữ liệu dùng để thể hiện một cơ sở dữ liệu tin học hoá có thể là phổ dụng (là một phần mềm đóng gói) hoặc có thể là chuyên dụng (là một tập các phần mềm được tạo ra với một mục đích riêng).

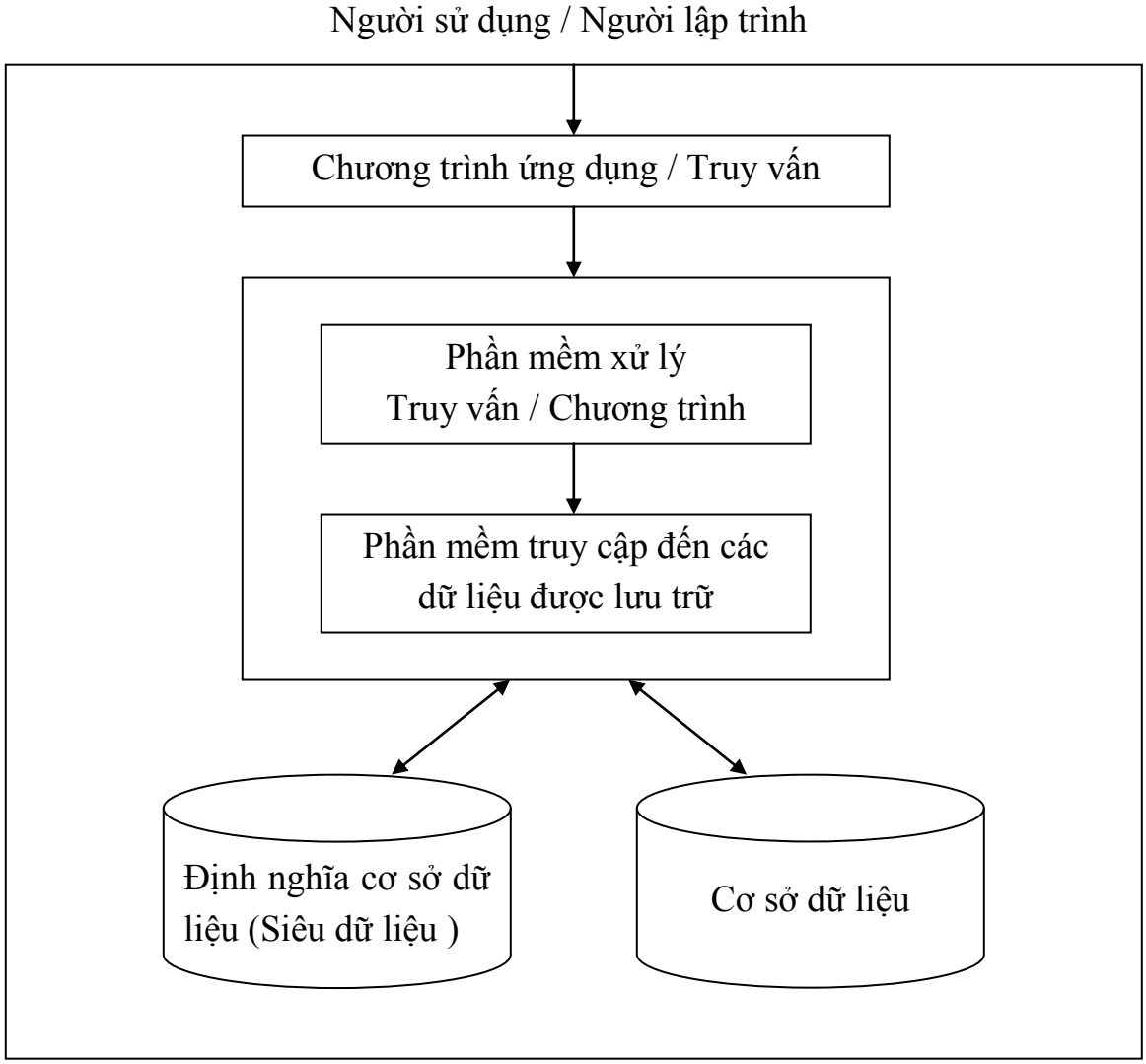
Người ta gọi cơ sở dữ liệu và hệ quản trị cơ sở dữ liệu bằng một thuật ngữ chung là ***hệ cơ sở dữ liệu.*** Môi trường của một hệ cơ sở dữ liệu được mô tả bằng hình vẽ dưới đây (hình I-1).

3.2- Các chức năng của một hệ quản trị cơ sở dữ liệu

Một hệ quản trị cơ sở dữ liệu hiện nay có các chức năng sau :

1. Lưu trữ các định nghĩa, các mối liên kết dữ liệu (gọi là siêu dữ liệu) vào một từ điển dữ liệu. Các chương trình truy cập đến cơ sở dữ liệu làm việc thông qua hệ quản trị cơ sở dữ liệu. Hệ quản trị cơ sở dữ liệu sử dụng dữ liệu trong từ điển dữ liệu để tìm kiếm các cấu trúc thành phần dữ liệu và các mối liên kết được yêu cầu. Mọi sự thay đổi trong các tệp cơ sở dữ liệu sẽ được tự động ghi lại vào từ điển dữ liệu. Như vậy, hệ quản trị cơ sở dữ liệu giải phóng người sử dụng khỏi việc lập trình cho các mối liên kết phức tạp trong mỗi chương trình, việc sửa đổi các

chương trình truy cập đến tệp cơ sở dữ liệu đã bị sửa đổi. Nói cách khác, hệ quản trị cơ sở dữ liệu loại bỏ sự phụ thuộc giữa dữ liệu và cấu trúc ra khỏi hệ thống.



ình 0-1. Môi trường hệ cơ sở dữ liệu

2. Tạo ra các cấu trúc phức tạp theo yêu cầu để lưu trữ dữ liệu. Nó giúp người sử dụng làm nhiệm vụ khó khăn là định nghĩa và lập trình cho các đặc trưng vật lý của dữ liệu.

3. Biến đổi các dữ liệu được nhập vào để phù hợp với các cấu trúc dữ liệu ở điểm 2. Như vậy, hệ quản trị cơ sở dữ liệu giúp người sử dụng phân biệt dạng logic và dạng vật lý của dữ liệu. Bằng việc duy trì sự độc lập dữ liệu, hệ quản trị cơ sở dữ liệu chuyển các yêu cầu logic thành các lệnh định vị một cách vật lý và lấy ra các dữ liệu yêu cầu. Điều đó cũng có nghĩa là hệ quản trị cơ sở dữ liệu tạo khuôn dạng

cho các dữ liệu được lấy ra để làm cho nó phù hợp với mong muốn logic của người sử dụng.

4. Tạo ra một hệ thống bảo mật và áp đặt tính bảo mật và riêng tư trong cơ sở dữ liệu.

5. Tạo ra các cấu trúc phức tạp cho phép nhiều người sử dụng truy cập đến dữ liệu

6. Cung cấp các thủ tục sao lưu và phục hồi dữ liệu để đảm bảo sự an toàn và toàn vẹn dữ liệu.

7. Xúc tiến và áp đặt các quy tắc an toàn để loại bỏ vấn đề toàn vẹn dữ liệu. Điều đó cho phép ta làm tối thiểu sự dư thừa dữ liệu và làm tối đa tính nhất quán dữ liệu.

8. Cung cấp việc truy cập dữ liệu thông qua một ngôn ngữ truy vấn. Một ngôn ngữ truy vấn là một ngôn ngữ phi thủ tục cho phép người sử dụng chỉ ra cái gì cần phải làm mà không cần phải chỉ ra nó được làm như thế nào. Các hệ quản trị cơ sở dữ liệu cũng cung cấp việc truy cập dữ liệu cho những người lập trình thông qua các ngôn ngữ thủ tục.

3.3- Các đặc trưng của giải pháp cơ sở dữ liệu

Trước khi khái niệm cơ sở dữ liệu ra đời, hệ thống tệp (file) là một phương pháp được áp dụng trong việc quản lý. Một tệp có thể được xem là một cặp hồ sơ lưu trữ các thông tin liên quan đến từng công việc riêng biệt. Ví dụ, trong một cơ quan, bộ phận tài vụ sẽ có một cặp hồ sơ liên quan đến lương của các nhân viên, bộ phận tổ chức có cặp hồ sơ liên quan đến vấn đề nhân sự… Việc xử lý để lấy ra các thông tin như là các thống kê về lương, về quá trình công tác… lúc đầu được thực hiện một cách thủ công. Dần dần, khối lượng thông tin ngày càng lớn, việc xử lý thông tin ngày càng phức tạp, người ta sử dụng máy tính vào việc quản lý. Các cặp hồ sơ được chuyển thành các tệp trên máy tính và việc xử lý thông tin được thực hiện bằng cách lập trình (trong một ngôn ngữ lập trình thế hệ 3).

Việc quản lý theo giải pháp hệ thống tệp có rất nhiều nhược điểm. Thứ nhất, đó là sự dư thừa thông tin: cùng một thông tin được lưu trữ nhiều lần (chẳng hạn, danh sách nhân viên có mặt trong tệp lương và cũng có mặt cả trong tệp nhân sự). Điều đó gây ra việc lãng phí bộ nhớ và dễ gây sai sót trong khi cập nhật dữ liệu, dễ sinh ra các dữ liệu không đúng đắn. Thứ hai, đó là việc phụ thuộc giữa chương

trình ứng dụng và dữ liệu. Mỗi khi có sự thay đổi cấu trúc tệp và các dữ liệu trong tệp, chương trình ứng dụng khai thác thông tin trên tệp đó cũng thay đổi theo. Điều đó gây ra khó khăn lớn cho việc bảo trì.

Giải pháp cơ sở dữ liệu ra đời đã giải quyết được những nhược điểm đó. Cụ thể, giải pháp cơ sở dữ liệu có những đặc trưng sau:

1. Bản chất tự mô tả của hệ cơ sở dữ liệu.

Một đặc trưng cơ bản của giải pháp cơ sở dữ liệu là hệ thống cơ sở dữ liệu không chỉ gồm có bản thân cơ sở dữ liệu mà còn có cả định nghĩa hoặc mô tả đầy đủ về cấu trúc cơ sở dữ liệu và các ràng buộc. Định nghĩa này được lưu trữ trong từ điển hệ thống, nó chứa các thông tin như là cấu trúc của mỗi tệp, kiểu và dạng lưu trữ của từng mục dữ liệu. Các thông tin được lưu giữ trong từ điển gọi là siêu dữ liệu (meta-data) và chúng mô tả cấu trúc của dữ liệu nguyên thuỷ (hình I-1). Phần mềm hệ quản trị cơ sở dữ liệu và những người sử dụng cơ sở dữ liệu sử dụng từ điển để lấy thông tin về cấu trúc của cơ sở dữ liệu.

2. Sự độc lập giữa chương trình và dữ liệu.

Trong hệ thống tệp, cấu trúc của các tệp cơ sở dữ liệu được nhúng vào trong các chương trình truy cập, vì vậy bất kỳ một thay đổi nào về cấu trúc của một tệp cũng đòi hỏi phải thay đổi tất cả các chương trình truy cập đến tệp đó. Ngược lại, các chương trình truy cập của hệ quản trị cơ sở dữ liệu không đòi hỏi việc thay đổi như thế. Cấu trúc của các tệp dữ liệu được lưu trữ trong từ điển tách rời với các chương trình truy cập. Tính chất này gọi là sự độc lập dữ liệu – chương trình.

3. Hỗ trợ các khung nhìn dữ liệu nhiều thành phần.

Một cơ sở dữ liệu có nhiều người sử dụng, mỗi một người có thể đòi hỏi một phối cảnh hoặc một khung nhìn (view) khác nhau. Một khung nhìn có thể là một tập con của cơ sở dữ liệu hoặc nó có thể chứa các dữ liệu ảo, đó là các dữ liệu được trích ra từ các tệp cơ sở dữ liệu khác nhau nhưng không được lưu trữ một cách rõ ràng. Một hệ quản trị cơ sở dữ liệu nhiều người sử dụng phải cung cấp nhiều công cụ để định nghĩa các khung nhìn nhiều thành phần.

4. Chia sẻ dữ liệu và nhiều người sử dụng.

Một hệ quản trị cơ sở dữ liệu nhiều người sử dụng phải cho phép nhiều người sử dụng truy cập đồng thời đến cơ sở dữ liệu. Hệ quản trị cơ sở dữ liệu phải có phần mềm kiểm tra cạnh tranh để đảm bảo rằng các người sử dụng cập nhật đến

cùng một cơ sở dữ liệu phải được thực hiện theo cách được kiểm tra để cho kết quả của các cập nhật là đúng đắn.

3.4- Ví dụ về một cơ sở dữ liệu

Chúng ta hãy xem xét một cơ sở dữ liệu mà nhiều người đã quen biết: cơ sở dữ liệu TRƯỜNG. Cơ sở dữ liệu này lưu giữ các thông tin liên quan đến sinh viên, các môn học, điểm… trong một môi trường đại học. Cơ sở dữ liệu được tổ chức thành 5 bảng, mỗi bảng lưu trữ các bản ghi dữ liệu cùng một kiểu. Bảng SINHVIÊN lưu giữ dữ liệu về các sinh viên, bảng MÔNHỌC lưu giữ các dữ liệu về các môn học, bảng HỌCPHẦN lưu giữ các dữ liệu về các học phần của các môn học, bảng ĐIỂM lưu giữ điểm của từng học phần của các sinh viên và bảng BIẾTTRƯỚC lưu giữ thông tin về các môn học cần biết trước của các môn học. Cấu trúc của cơ sở dữ liệu và một vài mẫu dữ liệu ví dụ được trình bày ở hình I-2.

SINHVIÊN	MãsốSV	HọtênSV	Lớp	Chuyênngành
	17	Nguyễn Nam	K45T	Tinhọc
	8	Lê Bắc	K45C	CôngnghệTT

MÔNHỌC	MãsốMH	TênMH	Sốđvht	Khoa
	101	Tinhọc cơ sở	8	Công nghệ
	102	Cấu trúc DL và GT	5	Công nghệ
	103	Toán rời rạc	5	Công nghệ
	104	Cơ sở dữ liệu	3	Công nghệ

HỌCPHẦN	MãsốHP	MãsốMH	Họckỳ	Năm	Têngiáoviên
	1011	101	1	2001	Vân
	1012	101	2	2002	Vân
	1031	103	1	2001	Hoàng
	1032	103	2	2002	Hoàng
	1020	102	3	2002	Lân
	1040	104	4	2002	Huy

ĐIỂM	MãsốSV	MãsốHP	Điểm
	17	1031	8
	17	1020	6
	8	1031	9
	8	1011	10
	8	1020	7
	8	1040	9

BIẾT TRƯỚC	Mã sốMH	MãsốMHbiếttrước
	104	102
	104	103
	102	101

Hình 0-2. Cơ sở dữ liệu TRƯỜNG

Để *định nghĩa* cơ sở dữ liệu này, chúng ta phải chỉ ra cấu trúc của các bản ghi của mỗi tệp (bảng) bằng cách đặc tả các kiểu khác nhau của các phần tử dữ liệu sẽ được lưu trữ trong mỗi bản ghi. Theo hình I-2 , mỗi bản ghi SINHVIÊN bao gồm các dữ liệu để biểu diễn Mã số sinh viên, Họ tên sinh viên, Lớp, Chuyên ngành. Mỗi bản ghi MÔNHỌC bao gồm các dữ liệu để biểu diễn Tên môn học, Mã số môn học, Số đơn vị học trình, Khoa,… Chúng ta phải chỉ ra một kiểu dữ liệu cho mỗi phần tử dữ liệu bên trong các bản ghi. Ví dụ, ta có thể đặc tả Họ tên sinh viên là một dãy ký tự có độ dài nhỏ hơn hoặc bằng 30, Mã số sinh viên là một số nguyên,….

Để *xây dựng* cơ sở dữ liệu TRƯỜNG, chúng ta lưu giữ các dữ liệu để biểu diễn mỗi sinh viên, mỗi môn học,… vào các tệp thích hợp. Để ý rằng các bản ghi trong các tệp khác nhau có thể có mối quan hệ với nhau. Ví dụ, bản ghi đối với Nguyễn Nam trong tệp SINHVIÊN có liên quan đến hai bản ghi trong tệp ĐIỂM. Các bản ghi này chỉ ra điểm của Nguyễn Nam trong hai học phần. Tương tự như vậy, các bản ghi trong tệp có mối quan hệ với các bản ghi trong tệp MÔNHỌC… Thông thường một cơ sở dữ liệu chứa nhiều kiểu bản ghi và chứa nhiều mối quan hệ giữa các tệp.

Thao tác cơ sở dữ liệu bao gồm việc truy vấn và cập nhật cơ sở dữ liệu. Truy vấn cơ sở dữ liệu là đưa ra các yêu cầu đối với cơ sở dữ liệu để lấy ra các thông tin cần thiết. Ví dụ, chúng ta có thể có các truy vấn như: "Liệt kê các môn học và điểm thi của sinh viên Nguyễn Nam", " Đưa ra danh sách các sinh viên thi trượt môn cơ sở dữ liệu". Cập nhật cơ sở dữ liệu bao gồm việc thêm vào cơ sở dữ liệu bản ghi, xoá bỏ các bản ghi hoặc sửa đổi các giá trị trong các bản ghi. Các truy vấn và các cập nhật phải được đặc tả trong ngôn ngữ hệ cơ sở dữ liệu một cách chính xác trước khi chúng được xử lý.

4. Mô hình cơ sở dữ liệu

Các loại cấu trúc cơ sở dữ liệu và mối liên hệ giữa chúng đóng vai trò rất lớn trong việc xác định tính hiệu quả của hệ quản trị cơ sở dữ liệu. Vì vậy, thiết kế cơ sở dữ liệu trở thành hoạt động chính trong môi trường cơ sở dữ liệu.

Việc thiết kế cơ sở dữ liệu được thực hiện đơn giản hơn nhiều khi ta sử dụng các mô hình. Các mô hình là sự trừu tượng đơn giản của các sự kiện trong thế giới thực. Các trừu tượng như vậy cho phép ta khảo sát các đặc điểm của các thực thể và các mối liên hệ được tạo ra giữa các thực thể đó. Việc thiết kế các mô hình tốt sẽ đưa ra các cơ sở dữ liệu tốt và trên cơ sở đó sẽ có các ứng dụng tốt. Ngược lại, mô hình không tốt sẽ đưa đến thiết kế cơ sở dữ liệu tồi và dẫn đến các ứng dụng không đúng.

Một ***mô hình cơ sở dữ liệu*** là một tập hợp các khái niệm dùng để biểu diễn các cấu trúc của cơ sở dữ liệu. Cấu trúc của một cơ sở dữ liệu là các kiểu dữ liệu, các mối liên kết và các ràng buộc phải tuân theo trên các dữ liệu. Nhiều mô hình còn có thêm một tập hợp các phép toán cơ bản để đặc tả các thao tác trên cơ sở dữ liệu.

4.1- Các loại mô hình cơ sở dữ liệu

Có rất nhiều mô hình dữ liệu đã được đề nghị. Chúng ta có thể phân loại các mô hình dữ liệu dựa trên các khái niệm mà chúng sử dụng để mô tả các cấu trúc cơ sở dữ liệu.

Các *mô hình dữ liệu bậc cao* hoặc *mô hình dữ liệu mức quan niệm* cung cấp các khái niệm gắn liền với cách cảm nhận dữ liệu của nhiều người sử dụng. Các mô hình này tập trung vào bản chất logic của biểu diễn dữ liệu, nó quan tâm đến *cái* được biểu diễn trong cơ sở dữ liệu chứ không phải *cách* biểu diễn dữ liệu.

Các *mô hình dữ liệu bậc thấp* hoặc các *mô hình dữ liệu vật lý* cung cấp các khái niệm mô tả chi tiết về việc các dữ liệu được lưu trữ trong máy tính như thế nào. Các khái niệm do mô hình dữ liệu vật lý cung cấp nói chung có ý nghĩa đối với các chuyên gia máy tính chứ không có ý nghĩa mấy đối với người sử dụng thông thường. Ở giữa hai loại mô hình này là một lớp các *mô hình dữ liệu thể hiện*, chúng cung cấp những khái niệm mà người sử dụng có thể hiểu được và không xa với cách tổ chức dữ liệu bên trong máy tính. Người ta còn gọi loại mô hình dữ liệu này là loại *mô hình dữ liệu mức logic*. Các mô hình dữ liệu thể hiện che giấu một số chi tiết về việc lưu trữ dữ liệu nhưng có thể được cài đặt trực tiếp trên hệ thống máy tính.

Trong chương II, chúng ta sẽ nghiên cứu một mô hình dữ liệu mức quan niệm, mô hình thực thể - liên kết, gọi tắt là mô hình ER (Entity – Relationship Model). Mô hình này sử dụng các khái niệm thực thể, thuộc tính, mối liên kết, để diễn đạt các đối tượng của thế giới thực. Một *thực thể* diễn đạt một đối tượng hoặc một khái niệm của thế giới thực. Ví dụ, một thực thể là một nhân viên hoặc một dự án được mô tả trong cơ sở dữ liệu. Một thuộc tính diễn đạt một đặc trưng nào đó của thực thể. Chẳng hạn, họ tên, lương… là các thuộc tính của thực thể nhân viên. Một mối liên kết giữa hai hay nhiều thực thể diễn đạt một mối quan hệ qua lại giữa các thực thể. Ví dụ, giữa thực thể nhân viên và thực thể dự án có mối liên kết một nhân viên *làm việc trên* một dự án. Mô hình dữ liệu hướng đối tượng cũng là một mô hình dữ liệu bậc cao. Nó sử dụng các khái niệm như lớp, phương thức, thông điệp… Bạn đọc có thể tìm hiểu về mô hình này trong các tài liệu [1], [2].

Các mô hình dữ liệu thể hiện là các mô hình được sử dụng thường xuyên nhất trong các hệ cơ sở dữ liệu thương mại. Ba mô hình nổi tiếng thuộc loại này là mô hình quan hệ, mô hình mạng và mô hình phân cấp. Các mô hình mạng và phân cấp ra đời trước và được sử dụng rộng rãi trong quá khứ (trước 1970). Vào đầu những năm 70, mô hình quan hệ ra đời. Do tính ưu việt của nó, mô hình quan hệ dần dần thay thế các mô hình mạng và phân cấp. Chúng ta sẽ nghiên cứu về mô hình quan hệ trong chương III.

Các mô hình dữ liệu vật lý mô tả cách lưu trữ dữ liệu trong máy tính giới thiệu các thông tin như khuôn dạng bản ghi, sắp xếp bản ghi, đường truy cập…

4.2- Lược đồ và trạng thái cơ sở dữ liệu

Trong một mô hình dữ liệu cần phải phân biệt rõ giữa *mô tả của cơ sở dữ liệu* và *bản thân cơ sở dữ liệu.* Mô tả của một cơ sở dữ liệu được gọi là *lược đồ cơ sở*

dữ liệu, nó được xác định rõ trong quá trình thiết kế cơ sở dữ liệu và không bị thay đổi thường xuyên. Đa số các mô hình dữ liệu có các quy ước hiển thị các lược đồ. Hiển thị của một lược đồ được gọi là biểu đồ của lược đồ đó. Một biểu đồ lược đồ chỉ thể hiện một vài khía cạnh của lược đồ như là các kiểu bản ghi, các mục dữ liệu và một số kiểu ràng buộc. Các khía cạnh khác không được thể hiện trong biểu đồ lược đồ.

Các dữ liệu trong một cơ sở dữ liệu có thể thay đổi một cách thường xuyên. Các dữ liệu trong một cơ sở dữ liệu tại một thời điểm cụ thể được gọi là một *trạng thái cơ sở dữ liệu* hoặc là ảnh (snapshot) của cơ sở dữ liệu. Nhiều trạng thái quan hệ có thể được xây dựng để làm tương ứng với một lược đồ cơ sở dữ liệu cụ thể. Mỗi khi chúng ta chèn vào hoặc loại bỏ một bản ghi, sửa đổi giá trị của một mục dữ liệu trong một bản ghi, chúng ta đã làm thay đổi trạng thái của cơ sở dữ liệu sang một trạng thái khác.

Việc phân biệt giữa lược đồ cơ sở dữ liệu và trạng thái cơ sở dữ liệu là rất quan trọng. Khi chúng ta định nghĩa một cơ sở dữ liệu mới, ta chỉ đặc tả lược đồ cơ sở dữ liệu cho hệ quản trị cơ sở dữ liệu. Tại thời điểm này, trạng thái của cơ sở dữ liệu là một trạng thái rỗng, không có dữ liệu. Chúng ta nhận được trạng thái ban đầu của cơ sở dữ liệu khi ta nhập dữ liệu lần đầu tiên. Từ đó trở đi, mỗi khi một phép toán cập nhật được thực hiện đối với cơ sở dữ liệu, chúng ta nhận được một trạng thái cơ sở dữ liệu khác. Tại mọi thời điểm, cơ sở dữ liệu có một trạng thái hiện tại. Hệ quản trị cơ sở dữ liệu có trách nhiệm đảm bảo rằng mỗi trạng thái cơ sở dữ liệu là một trạng thái vững chắc, nghĩa là một trạng thái thoả mãn cấu trúc và các ràng buộc được đặc tả trong lược đồ. Vì vậy, việc đặc tả một lược đồ đúng đắn cho hệ quản trị cơ sở dữ liệu là một việc làm cực kỳ quan trọng và lược đồ phải được thiết kế một cách cẩn thận. Hệ quản trị cơ sở dữ liệu lưu trữ các mô tả của các cấu trúc lược đồ và các ràng buộc – còn gọi là siêu dữ liệu – vào trong từ điển (catalog) của hệ quản trị sao cho phần mềm hệ quản trị cơ sở dữ liệu có thể tham khảo đến lược đồ khi nó cần. Đôi khi người ta còn gọi lược đồ là mục tiêu (intension) và trạng thái cơ sở dữ liệu là mở rộng (extension) của lược đồ.

5. Con người trong hệ cơ sở dữ liệu

Với một cơ sở dữ liệu lớn, rất nhiều người tham gia vào việc thiết kế, sử dụng và duy trì cơ sở dữ liệu. Những người liên quan đến hệ cơ sở dữ liệu được chia thành hai nhóm chính. Nhóm thứ nhất gồm những người mà công việc của họ liên

quan hàng ngày đến cơ sở dữ liệu, đó là những người quản trị cơ sở dữ liệu, thiết kế cơ sở dữ liệu, sử dụng cơ sở dữ liệu, phân tích hệ thống và lập trình ứng dụng. Nhóm thứ hai gồm những người làm việc để duy trì môi trường hệ cơ sở dữ liệu nhưng không quan tâm đến bản thân cơ sở dữ liệu, đó là những người thiết kế và cài đặt hệ quản trị cơ sở dữ liệu, phát triển công cụ, thao tác viên và bảo trì.

5.1- Người quản trị hệ cơ sở dữ liệu (Database Administrator – DBA)

Trong một tổ chức có nhiều người cùng sử dụng các tài nguyên, cần phải có một người giám sát và quản lý. Trong môi trường hệ cơ sở dữ liệu, các tài nguyên là cơ sở dữ liệu, hệ quản trị cơ sở dữ liệu và các phần mềm liên quan. Người quản trị hệ cơ sở dữ liệu là người chịu trách nhiệm quản lý các tài nguyên đó. Người này chịu trách nhiệm về việc cho phép truy cập cơ sở dữ liệu, tổ chức và hướng dẫn việc sử dụng cơ sở dữ liệu, cấp các phần mềm và phần cứng theo yêu cầu.

5.2- Người thiết kế cơ sở dữ liệu (Database Designer)

Người này chịu trách nhiệm xác định các dữ liệu sẽ được lưu giữ trong cơ sở, chọn các cấu trúc thích hợp để biểu diễn và lưu giữ các dữ liệu đó. Những nhiệm vụ này được thực hiện trước khi cơ sở dữ liệu được cài đặt và phổ biến. Người thiết kế có trách nhiệm giao thiệp với những người sử dụng tương lai để hiểu được các đòi hỏi của họ và đưa ra một thiết kế thoả mãn các yêu cầu đó. Anh ta cũng có nhiệm vụ giao thiệp với các nhóm người sử dụng và có khả năng hỗ trợ các yêu cầu của các nhóm.

5.3- Những người sử dụng (End User)

Những người sử dụng là những người mà công việc của họ đòi hỏi truy cập đến cơ sở dữ liệu để truy vấn, cập nhật và sinh ra các thông tin. Có thể chia những người sử dụng thành hai nhóm chính: những *người sử dụng thụ động* (tức là những người sử dụng không có nhiều kiến thức về hệ cơ sở dữ liệu) và những *người sử dụng chủ động* (là những người có hiểu biết tốt về hệ cơ sở dữ liệu).

Chức năng công việc của những người sử dụng thụ động (chiếm phần lớn những người sử dụng) gắn liền với việc truy vấn và cập nhật thường xuyên cơ sở dữ liệu bằng cách sử dụng các câu hỏi và các cập nhật chuẩn (gọi là các giao tác định sẵn) đã được lập trình và kiểm tra cẩn thận. Những người này chỉ cần học một

ít về các phương tiện do hệ quản trị cơ sở dữ liệu cung cấp và hiểu các kiểu giao tác chuẩn đã được thiết kế và cài đặt là đủ.

Những người sử dụng chủ động có hiểu biết tốt về hệ cơ sở dữ liệu, họ có thể tự cài đặt các ứng dụng riêng của mình để làm thoả mãn các yêu cầu phức tạp của họ.

5.4- Người phân tích hệ thống và lập trình ứng dụng

Người phân tích hệ thống xác định các yêu cầu của những người sử dụng (chủ yếu là những người sử dụng thụ động) để đặc tả các chương trình phù hợp với yêu cầu của họ.

Người viết chương trình ứng dụng thể hiện các đặc tả của những người phân tích thành chương trình, sau đó kiểm thử, sửa lỗi làm tài liệu và bảo trì các giao tác định sẵn.

5.5- Người thiết kế và cài đặt hệ quản trị dữ liệu

Đó là những người thiết kế, cài đặt các mô đun, giao diện của hệ quản trị cơ sở dữ liệu thành các phần mềm đóng gói. Một hệ quản trị cơ sở dữ liệu là một hệ thống phần mềm phức tạp bao gồm nhiều thành phần (mô đun). Đó là các mô đun cài đặt từ điển dữ liệu, ngôn ngữ truy vấn, bộ xử lý giao diện, truy cập dữ liệu, kiểm tra cạnh tranh, phục hồi và an toàn. Hệ quản trị cơ sở dữ liệu phải giao tiếp với các hệ thống phần mềm khác như hệ điều hành và các chương trình dịch cho nhiều ngôn ngữ khác nhau.

5.6- Những người phát triển công cụ

Là những người thiết kế và cài đặt các công cụ (tool), đó là các phần mềm đóng gói làm dễ việc thiết kế và sử dụng cơ sở dữ liệu.

5.7- Các thao tác viên và những người bảo trì

Là những người chịu trách nhiệm về việc chạy và bảo trì phần cứng và phần mềm của hệ thống.

6. Ngôn ngữ cơ sở dữ liệu và giao diện

6.1- Các ngôn ngữ hệ quản trị cơ sở dữ liệu

Một khi việc thiết kế cơ sở dữ liệu đã hoàn thành, cần phải chọn một hệ quản trị cơ sở dữ liệu để cài đặt cơ sở dữ liệu. Trong các hệ quản trị cơ sở dữ liệu hiện

nay thường có các ngôn ngữ: ngôn ngữ định nghĩa dữ liệu (data definition language – DDL) và ngôn ngữ thao tác dữ liệu (data manipulation language – DML).

Ngôn ngữ định nghĩa dữ liệu được sử dụng để định nghĩa các lược đồ. Hệ quản trị cơ sở dữ liệu có một chương trình dịch ngôn ngữ DDL, nhiệm vụ của nó là xử lý các câu lệnh DDL để xác định mô tả của cấu trúc lược đồ và lưu trữ mô tả lược đồ vào từ điển của hệ quản trị cơ sở dữ liệu.

Ngôn ngữ thao tác cơ sở dữ liệu được sử dụng để thao tác cơ sở dữ liệu. Các thao tác chính gồm có lấy ra, chèn vào, loại bỏ và sửa đổi các dữ liệu. Có hai kiểu ngôn ngữ thao tác dữ liệu chính: ngôn ngữ thao tác dữ liệu mức cao hoặc ngôn ngữ phi thủ tục hoặc ngôn ngữ thao tác dữ liệu mức thấp.

Ngôn ngữ thao tác dữ liệu mức cao có thể được sử dụng để diễn đạt các phép toán cơ sở dữ liệu một cách ngắn gọn. Phần lớn các hệ quản trị cơ sở dữ liệu cho phép nhập các lệnh của ngôn ngữ thao tác dữ liệu mức cao theo cách lặp (nghĩa là sau khi nhập một lệnh, hệ thống sẽ thực hiện lệnh đó rồi mới nhập lệnh tiếp theo) hoặc được nhúng vào một ngôn ngữ lập trình vạn năng. Trong trường hợp nhúng vào ngôn ngữ khác, các lệnh của ngôn ngữ thao tác dữ liệu phải được xác định bên trong chương trình sao cho một chương trình tiền dịch có thể nhận ra chúng và được hệ quản trị cơ sở dữ liệu xử lý.

Ngôn ngữ thao tác cơ sở dữ liệu mức thấp hoặc ngôn ngữ thủ tục phải được nhúng vào trong một ngôn ngữ lập trình vạn năng. Ngôn ngữ thao tác cơ sở dữ liệu kiểu này thường rút ra các bản ghi hoặc các đối tượng riêng rẽ và xử lý chúng một cách riêng rẽ. Vì vậy, chúng cần phải sử dụng các cấu trúc ngôn ngữ lập trình như vòng lặp, điều kiện,… để rút ra từng bản ghi một từ một tập các bản ghi. Ngôn ngữ thao tác dữ liệu mức thấp được gọi là ngôn ngữ "một lần một bản ghi". Các ngôn ngữ thao tác dữ liệu mức cao có thể dùng một lệnh để rút ra một lúc nhiều bản ghi nên chúng được gọi là ngôn ngữ "một lần một tập hợp".

6.2- Các loại giao diện hệ quản trị cơ sở dữ liệu

Các hệ quản trị cơ sở dữ liệu cung cấp rất nhiều loại giao diện người dùng thân thiện. Các loại giao diện chính gồm có:

Giao diện dựa trên bảng chọn: Các giao diện này cung cấp cho người sử dụng danh sách các lựa chọn, gọi là bảng chọn (menu) và hướng dẫn người sử dụng diễn đạt một yêu cầu từ đầu đến cuối. Các bảng chọn làm cho người sử dụng không cần

nhớ các lệnh và cú pháp của ngôn ngữ truy vấn. Các bảng chọn thả xuống đã trở thành kỹ thuật phổ biến trong các giao diện dựa trên cửa sổ. Chúng thường được sử dụng trong các giao diện quét, cho phép người sử dụng nhìn thấy nội dung của một cơ sở dữ liệu theo cách không có cấu trúc.

Giao diện dựa trên mẫu biểu: Các giao diện này hiển thị một mẫu biểu cho người sử dụng. Những người sử dụng có thể điền vào tất cả các ô của mẫu biểu để nhập các dữ liệu mới hoặc họ chỉ điền vào một số ô còn hệ quản trị cơ sở dữ liệu sẽ đưa ra các dữ liệu phù hợp cho các ô khác. Các mẫu biểu thường được thiết kế và được lập trình cho các người dùng đơn giản. Một số hệ thống có các tiện ích giúp người sử dụng từng bước xây dựng một mẫu biểu trên màn hình.

Giao diện đồ hoạ: Một giao diện đồ hoạ (GUI) thường hiển thị một lược đồ cho người sử dụng dưới dạng biểu đồ. Người dùng có thể thực hiện một truy vấn bằng cách thao tác trên biểu đồ. Trong nhiều trường hợp, GUI sử dụng cả các bảng chọn và các mẫu biểu. Đa số các GUI sử dụng các công cụ trỏ như chuột, phím để kích các phần của sơ đồ.

Giao diện cho người quản trị hệ thống: Đa số các hệ quản trị cơ sở dữ liệu có các lệnh ưu tiên, chỉ có những người quản trị hệ thống mới sử dụng các lệnh đó. Đó là các lệnh tạo ra các tài khoản (account), đặt các tham số cho hệ thống, cấp các tài khoản, thay đổi lược đồ hoặc tổ chức lại các cấu trúc lưu trữ của cơ sở dữ liệu.

7. Câu hỏi ôn tập

1. Định nghĩa các thuật ngữ : cơ sở dữ liệu, hệ quản trị cơ sở dữ liệu, hệ cơ sở dữ liệu, từ điển cơ sở dữ liệu, mô hình cơ sở dữ liệu.
2. Nêu các tính chất của một cơ sở dữ liệu
3. Nêu các chức năng của một hệ quản trị cơ sở dữ liệu
4. Giải thích các đặc trưng của giải pháp cơ sở dữ liệu
5. Định nghĩa mô hình cơ sở dữ liệu và phân loại
6. Liệt kê các người có liên quan đến hệ cơ sở dữ liệu.

CHƯƠNG 2 - MÔ HÌNH THỰC THỂ - LIÊN KẾT

Trong chương này chúng ta sẽ làm quen với mô hình thực thể - liên kết, gọi tắt là mô hình ER (Entity-Relationship Model). Đó là một mô hình dữ liệu mức quan niệm phổ biến, tập trung vào các cấu trúc dữ liệu và các ràng buộc. Mô hình này thường được sử dụng để thiết kế các ứng dụng cơ sở dữ liệu và nhiều công cụ thiết kế cơ sở dữ liệu sử dụng các khái niệm của nó.

1. Sử dụng mô hình quan niệm bậc cao cho việc thiết kế cơ sở dữ liệu

Quá trình thiết kế một cơ sở dữ liệu sử dụng mô hình quan niệm bậc cao được minh họa bằng hình II.1.

Bước đầu tiên là *tập hợp các yêu cầu và phân tích*. Trong bước này, người thiết kế cơ sở dữ liệu phỏng vấn những người sử dụng cơ sở dữ liệu để hiểu và làm tài liệu về các yêu cầu về dữ liệu của họ. Kết quả của bước này là một tập hợp ghi chép súc tích về các yêu cầu của những người sử dụng. Những yêu cầu sẽ được đặc tả càng đầy đủ và chi tiết càng tốt. Song song với việc đặc tả các yêu cầu dữ liệu, cần phải đặc tả các yêu cầu về chức năng của ứng dụng: đó là các thao tác do người sử dụng định nghĩa sẽ được áp dụng đối với cơ sở dữ liệu.

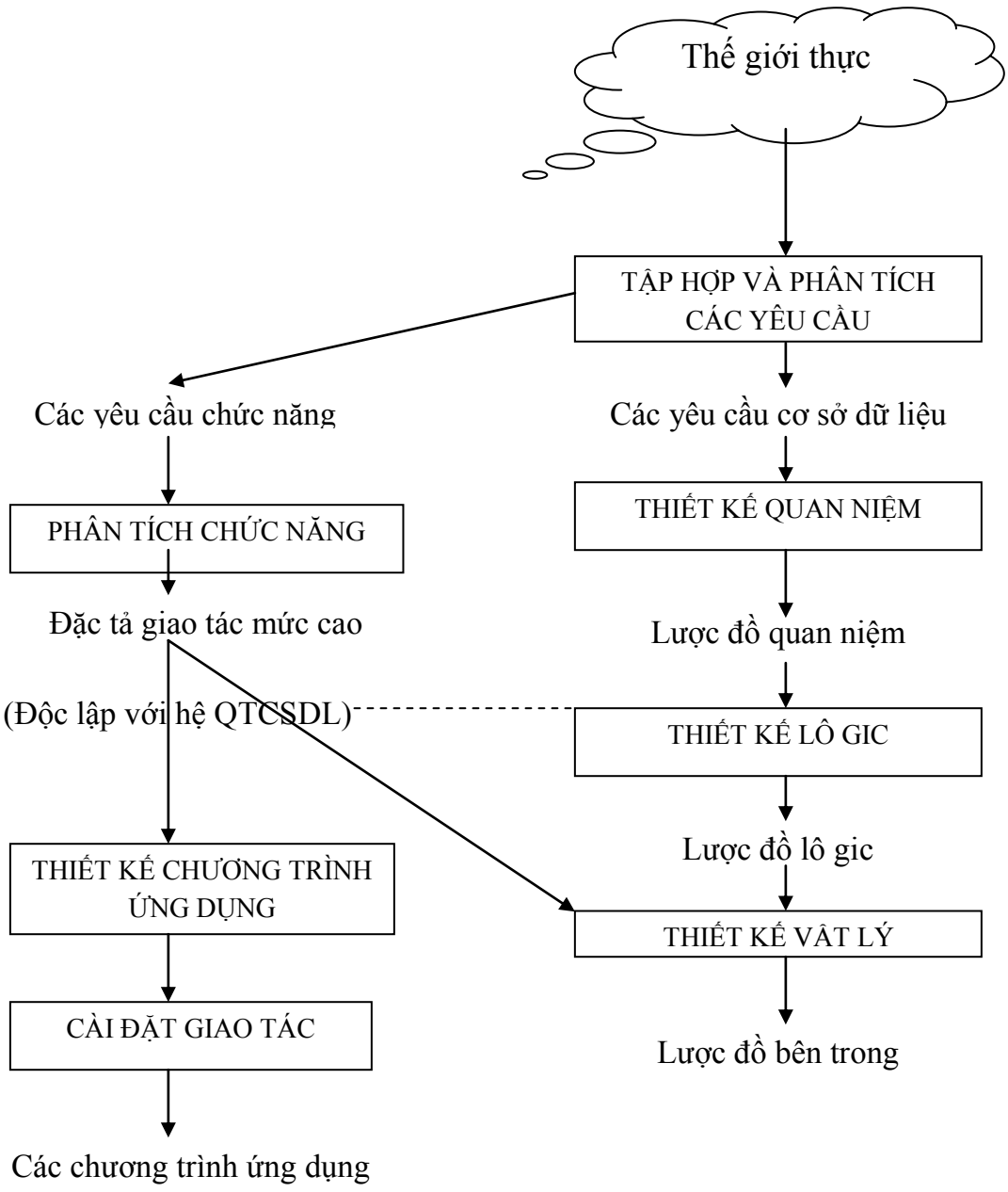
Mỗi khi tất cả các yêu cầu đã được thu thập và phân tích, bước tiếp theo là tạo ra lược đồ quan niệm cho cơ sở dữ liệu bằng cách sử dụng mô hình dữ liệu quan niệm mức cao. Bước này gọi là *thiết kế quan niệm*. Lược đồ quan niệm là một mô tả súc tích về các yêu cầu dữ liệu của những người sử dụng. Nó bao gồm các mô tả chi tiết của các kiểu thực thể, kiểu liên kết và các ràng buộc, chúng được biểu diễn bằng các khái niệm do các mô hình dữ liệu bậc cao cung cấp. Vì những khái niệm này không chứa các chi tiết cài đặt, chúng thường dễ hiểu và có thể sử dụng chúng để giao lưu với những người sử dụng. Lược đồ quan niệm mức cao cũng có thể được sử dụng như một dẫn chứng để đảm bảo rằng tất cả các đòi hỏi của người sử dụng đều thỏa mãn và các đòi hỏi này không chứa các mâu thuẫn. Giải pháp này cho phép những người thiết kế cơ sở dữ liệu tập trung vào việc đặc tả các tính chất của dữ liệu mà không cần quan tâm đến các chi tiết lưu trữ. Một thiết kế cơ sở dữ liệu quan niệm tốt sẽ làm dễ cho công việc của những người thiết kế cơ sở dữ liệu.

Trong quá trình (hoặc sau khi) thiết kế lược đồ quan niệm, chúng ta có thể sử dụng các phép toán cơ bản của mô hình dữ liệu để đặc tả các thao tác của người sử

dụng được xác định trong khi phân tích chức năng. Điều đó cũng giúp khẳng định rằng lược đồ quan niệm thỏa mãn mọi yêu cầu chức năng được xác định. Nếu có một số yêu cầu chức năng không thể nêu ra được trong lược đồ ban đầu thì ở bước này có thể có sự sửa đổi lược đồ quan niệm cho phù hợp.

Bước tiếp theo trong việc thiết kế cơ sở dữ liệu là việc cài đặt một cơ sở dữ liệu bằng cách sử dụng một hệ quản trị cơ sở dữ liệu có sẵn. Hầu hết các hệ quản trị cơ sở dữ liệu sử dụng một mô hình dữ liệu cài đặt (thể hiện), chẳng hạn như mô hình quan hệ hoặc đối tượng, vì vậy lược đồ quan niệm được chuyển từ mô hình dữ liệu bậc cao thành mô hình dữ liệu cài đặt. Bước này gọi là *thiết kế logic* hoặc là *ánh xạ mô hình dữ liệu.* Kết quả của bước này là một lược đồ cơ sở dữ liệu dưới dạng một mô hình dữ liệu cài đặt của hệ quản trị cơ sở dữ liệu.

Bước cuối cùng trong thiết kế cơ sở dữ liệu là *thiết kế vật lý*. Trong bước này ta phải chỉ ra các cấu trúc bên trong, các đường dẫn truy cập, tổ chức tệp cho các tệp cơ sở dữ liệu. Song song với các hoạt động đó, các chương trình ứng dụng cũng được thiết kế và cài đặt như là các giao tác (transaction) cơ sở dữ liệu tương ứng với các đặc tả giao tác mức cao.



Hình 0-1. Sơ đồ mô tả các bước chính của việc thiết kế

2. Các thành phần cơ bản của mô hình ER

2.1- Thực thể và thuộc tính

Đối tượng được trình bày trong mô hình ER là thực thể. *Thực thể* là một "vật" trong thế giới thực, có sự tồn tại độc lập. Một thực thể có thể là cụ thể, tức là chúng ta có thể cảm nhận được bằng các giác quan, hoặc có thể là trừu tượng, tức là cái mà chúng ta không cảm nhận được bằng các giác quan nhưng có thể nhận biết được

bằng nhận thức. Một cái ô tô, một nhân viên,… là những thực thể cụ thể. Một đơn vị công tác, một trường học… là những thực thể trừu tượng.

Mỗi một thực thể có các *thuộc tính*, đó là các đặc trưng cụ thể mô tả thực thể đó. Ví dụ, một thực thể *Nhânviên* được mô tả bằng *Họtên, Tuổi, Địachỉ, Lương*… của nhân viên đó. Một thực thể cụ thể sẽ có một giá trị cho mỗi thuộc tính của nó. Ví dụ, nhân viên nv1 có các giá trị cho các thuộc tính Họtên, Tuổi, Địachỉ, Lương của nó là " Lê Vân", 32, "Hà nội", 500000. Các giá trị thuộc tính mô tả mỗi thực thể sẽ trở thành một phần chính của các dữ liệu sẽ được lưu giữ trong cơ sở dữ liệu. Trong mô hình ER có mặt nhiều kiểu thuộc tính: thuộc tính đơn, thuộc tính phức hợp, thuộc tính đơn trị, thuộc tính đa trị, thuộc tính được lưu trữ, thuộc tính suy diễn được, thuộc tính có giá trị không xác định, thuộc tính phức tạp.

Thuộc tính đơn là thuộc tính không thể phân chia ra được thành các thành phần nhỏ hơn. Ví dụ, thuộc tính Tuổi của một nhân viên là một thuộc tính đơn. *Thuộc tính phức hợp* là thuộc tính có thể phân chia được thành các thành phần nhỏ hơn, biểu diễn các thuộc tính cơ bản hơn với các ý nghĩa độc lập. Ví dụ, thuộc tính Họtên của thực thể nhân viên có thể phân chia thành các tính Họđệm và Tên. Giá trị của một thuộc tính là sự kết hợp kết hợp các giá trị của các thuộc tính thành phần tạo nên nó. Việc phân chia một thuộc tính phức hợp thành các thuộc tính đơn tùy thuộc vào hoàn cảnh cụ thể.

Những thuộc tính có giá trị duy nhất cho một thực thể cụ thể gọi là các *thuộc tính đơn trị.* Ví dụ, Họtên là một thuộc tính đơn trị của thực thể nhân viên, mỗi nhân viên có một họ tên duy nhất. Trong một số trường hợp, một thuộc tính có thể có một tập giá trị cho cùng một thực thể. Những thuộc tính như vậy gọi là *thuộc tính đa trị.* Ví dụ, thuộc tính Bằngcấp của một người. Một người có thể không có bằng cấp nào, người khác có thể có một bằng, người khác nữa có thể có nhiều bằng. Như vậy, các người khác nhau có thể có một số giá trị khác nhau cho thuộc tính Bằngcấp. Thuộc tính Bằngcấp là một thuộc tính đa trị.

Thuộc tính được lưu trữ là các thuộc tính mà giá trị của nó được nhập vào khi cài đặt cơ sở dữ liệu. Trong một số trường hợp, hai hay nhiều thuộc tính có giá trị liên quan đến nhau. Ví dụ, thuộc tính Tuổi và thuộc tính Ngàysinh của một người. Với một người cụ thể, ta có thể tính tuổi của anh ta bằng cách lấy năm hiện tại trừ đi năm của Ngàysinh. Thuộc tính mà giá trị của nó có thể tính được thông qua giá trị của các thuộc tính khác gọi là *thuộc tính suy diễn được*.

Các giá trị không xác định (null values): Trong một số trường hợp, một thực thể cụ thể có thể không có các giá trị áp dụng được cho một thuộc tính. Ví dụ, thuộc tính Sốđiệnthoại của thực thể nhân viên sẽ không có giá trị đối với các nhân viên không có số điện thoại. Trong trường hợp như vậy, ta phải tạo ra một giá trị đặc biệt gọi là giá trị không xác định (null). Giá trị không xác định được tạo ra khi một thuộc tính có giá trị không áp dụng được hoặc khi không biết.

Các thuộc tính phức tạp: Là sự kết hợp của các thuộc tính phức hợp và đa trị.

2.2- Kiểu thực thể, tập thực thể, khóa và tập giá trị

Các kiểu thực thể và các tập thực thể: Một cơ sở dữ liệu thường chứa những nhóm thực thể như nhau. Ví dụ, một công ty thuê hàng trăm nhân viên và lưu giữ những thông tin tương tự liên quan đến mỗi nhân viên. Các thực thể nhân viên này chia sẻ các thuộc tính giống nhau nhưng mỗi thực thể có các giá trị riêng cho các thuộc tính đó. Một *kiểu thực thể* là một tập hợp các thực thể có các thuộc tính như nhau. Một kiểu thực thể trong cơ sở dữ liệu được mô tả bằng tên và các thuộc tính. Vídụ: NHÂNVIÊN (Họtên, Tuổi, Lương), CÔNGTY (Tên, Địađiểm, Giámđốc). Một tập hợp các thực thể của một kiểu thực thể cụ thể trong cơ sở dữ liệu tại một thời điểm được gọi là một tập thực thể, nó thường được tham chiếu đến bằng cách sử dụng tên của kiểu thực thể. Ví dụ, NHÂNVIÊN vừa dùng để chỉ một kiểu thực thể, vừa để chỉ tập hợp hiện tại của tất cả các thực thể nhân viên trong cơ sở dữ liệu. Hình II-2 minh họa các kiểu thực thể NHÂNVIÊN, CÔNGTY và các tập thực thể tương ứng.

Một kiểu thực thể được biểu diễn trong lược đồ ER như là một hộp hình chữ nhật có chứa tên kiểu thực thể. Các thuộc tính được đặt trong các hình ô van và được nối với các kiểu thực thể bằng các đường thẳng. Các thuộc tính phức hợp cũng được nối với các thuộc tính thành phần của nó bằng đường thẳng. Các thuộc tính đa trị được hiển thị trong các hình ô van đúp (hình II-3).

Một kiểu thực thể mô tả một *lược đồ* (hoặc một mục đích) cho một tập các thực thể chia sẻ cùng một cấu trúc. Tập hợp các thực thể của một kiểu thực thể cụ thể được nhóm vào một tập thực thể và được gọi là một *thể hiện* của một kiểu thực thể.

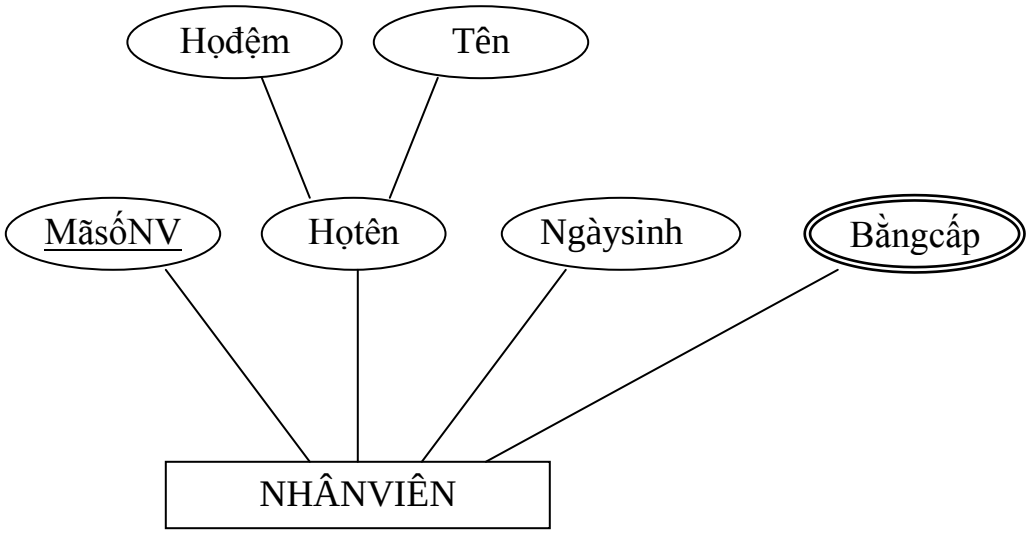
NHÂNVIÊN (Họtên, Tuổi, Lương)	CÔNGTY (Tên, Địadiểm, Giámđốc)
Nv1 (Lê Lan, 30, 800000) Nv2 (Trần Bá, 45, 1000000) Nv3 (Hoàng Vân, 25, 600000) . . .	Ct1 (CT Phần mềm, Hà nội, Vũ An) Ct2 (CT Hoa quả, Hải phòng, Lê Hà) Ct3 (CT Máy tính, Hà nội, Phan Anh) . . .

Hình 0-2. Kiểu thực thể và tập thực thể

Thuộc tính khóa của một kiểu thực thể: Một ràng buộc quan trọng trên các thực thể của một kiểu thực thể là *khóa*. Một kiểu thực thể thường có một thuộc tính mà các giá trị của nó là khác nhau đối với mỗi thực thể tiêng biệt trong một tập thực thể. Thuộc tính như vậy gọi là *thuộc tính khóa* và các giá trị của nó có thể dung để xác định từng thực thể một cách duy nhất. Ví dụ, thuộc tính Tên của kiểu thực thể CÔNGTY là khóa của kiểu thực thể đó vì mỗi thực thể công ty có một tên duy nhất. Đôi khi, nhiều thuộc tính kết hợp với nhau tạo thành một khóa, nghĩa là tổ hợp các giá trị của các thuộc tính này phải khác nhau đối với mỗi thực thể. Trong trường hợp như vậy ta có một thuộc tính *khóa phức hợp*. Chú ý rằng khóa phức hợp phải tối thiểu, nghĩa là tất cả các thuộc tính thành phần phải có mặt trong thuộc tính phức hợp để thỏa mãn tính chất duy nhất. Trong biểu đồ đồ họa của mô hình ER, thuộc tính khóa được biểu diễn bằng cách gạch ngang dưới tên của nó (hình II-3).

Khi chỉ ra rằng một thuộc tính là khóa của một kiểu thực thể nghĩa là tính chất duy nhất nêu trên phải được thỏa mãn đối với đối với mỗi mở rộng của kiểu thực thể. Như vậy, ràng buộc khóa cấm hai thực thể bất kỳ có giá trị cho thuộc tính khóa như nhau tại cùng một thời điểm. Đó là một ràng buộc trên tất cả các thể hiện của thực thể. Ràng buộc khóa cũng như các ràng buộc sẽ được giới thiệu về sau được lấy ra từ các ràng buộc của "thế giới nhỏ" của cơ sở dữ liệu.

Một kiểu thực thể có thể có nhiều hơn một thuộc tính khóa. Ví dụ, nếu một công ty có một mã số duy nhất và một tên duy nhất thì các thuộc tính Mãsốcông ty và Têncôngty đều là các thuộc tính khóa. Một kiểu thực thể cũng có thể không có khóa. Một thực thể không có khóa được gọi là *kiểu thực thể yếu*.



Hình 0-3. Biểu diễn kiểu thực thể và các thuộc tính

Miền giá trị của các thuộc tính: Mỗi thuộc tính đơn của một kiểu thực thể được kết hợp với một miền giá trị. Đó là một tập các giá trị có thể gán cho thuộc tính này đối với mỗi thực thể riêng biệt. Các miền giá trị không hiển thị trong các sơ đồ ER.

Một cách toán học, một thuộc tính A của kiểu thực thể E có tập giá trị V có thể được định nghĩa như là một hàm từ E vào tập hợp lực lượng P(V) của V:

$$A: E \rightarrow P(V)$$

Ta ký hiệu giá trị của thuộc tính A đối với thực thể e là A(e). Định nghĩa ở trên đúng cho các thuộc tính đơn trị, đa trị và thuộc tính không xác định. Một giá trị không xác định được biểu diễn bằng một tập rỗng. Với các thuộc tính đơn trị, A(e) là một giá trị đơn cho thực thể e. Các thuộc tính đa trị không có các hạn chế trên A(e). Với một thuộc tính phức hợp A, tập giá trị V là tích Đề các của $P(V_1)$x $P(V_2)$x….x$P(V_n)$, trong đó V_1, V_2, …, V_n là tập các giá trị cho các thành phần đơn của A.

2.3- Kiểu liên kết, tập liên kết và các thể hiện

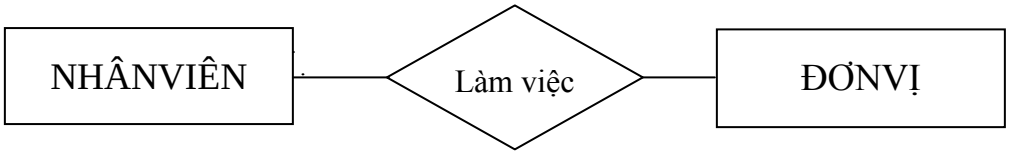
Một *kiểu liên kết* R giữa n kiểu thực thể E_1, E_2, …,E_n xác định một *tập liên kết* giữa các thực thể của các kiểu đó. Cũng như các kiểu thực thể và tập thực thể, một kiểu liên kết và tập liên kết tương ứng với nó cũng có tên chung là R. Một cách toán học, tập liên kết R là một tập hợp các thể hiện liên kết r_i , i= 1,2,… trong đó mỗi r_i liên kết n thực thể riêng biệt e_1,e_2,…,e_n và mỗi một thực thể e_j trong r_i là một thành phần của kiểu thực thể E_j , $1 \leq j \leq n$. Như vậy, một kiểu liên kết R là một quan

hệ toán học trên E_1, E_2, …, E_n hoặc có thể định nghĩa như là một tập con của tích Đề các E_1 x E_2 x …x E_n. Mỗi kiểu thực thể E_1,E_2, …, E_n được gọi là tham gia vào kiểu liên kết R, và tương tự, mỗi thực thể riêng biệt e_1, e_2, …, e_n được gọi là tham gia vào thể hiện liên kết r_i = (e_1,e_2,…,e_n).

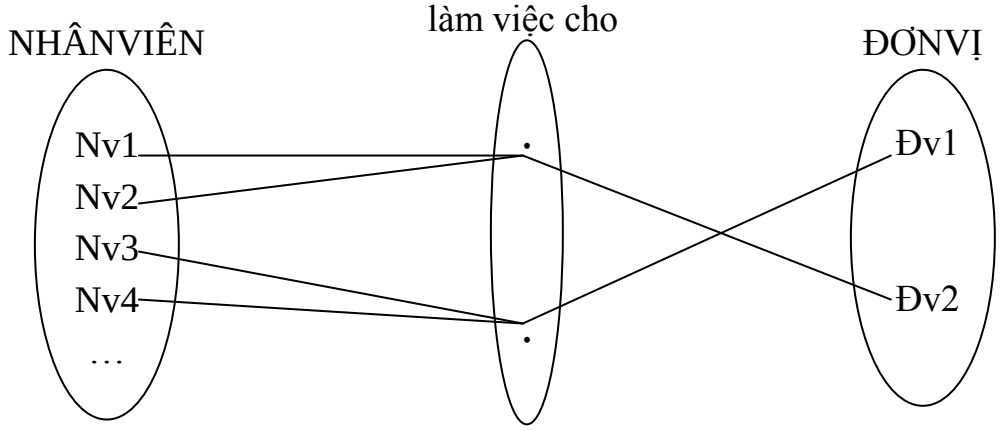
Một cách không hình thức, mỗi thể hiện liên kết r_i trong R là một sự kết hợp của các thực thể, mỗi thực thể thuộc về một kiểu thực thể tham gia vào liên kết. Mỗi liên kết r_i như vậy diễn đạt một sự kiện rằng các thực thể tham gia trong r_i có mối quan hệ với nhau theo một cách nào đó ở trong thế giới thực. Ví dụ, trong thực tế, các nhân viên làm việc cho các đơn vị, như vậy, có một kiểu liên kết liên kết *làm việc cho*, liên kết giữa kiểu thực thể NHÂNVIÊN và kiểu thực thể ĐƠNVỊ.

Trong sơ đồ ER, kiểu liên kết được biểu diễn bằng một hình thoi nối trực tiếp với các hình chữ nhật biểu diễn các kiểu thực thể tham gia vào liên kết. Hình II-4 minh họa kiểu liên kết và thể hiện liên kết

a) Kiểu liên kết



b) Thể hiện liên kết



Hình 0-4. Kiểu liên kết và thể hiện liên kết

2.4- Cấp liên kết, tên vai trò và kiểu liên kết đệ quy

Cấp của một kiểu liên kết là số các kiểu thực thể tham gia vào kiểu liên kết đó. Một kiểu liên kết có thể có cấp 1, cấp 2, cấp 3,…. Ví dụ, kiểu liên kết <*làm việc cho*> giữa kiểu thực thể NHÂNVIÊN và kiểu thực thể ĐƠNVỊ là một kiểu liên kết

cấp 2. Kiểu liên kết <*biết trước*> giữa kiểu thực thể MÔNHỌC với chính nó là một kiểu liên kết cấp 1…

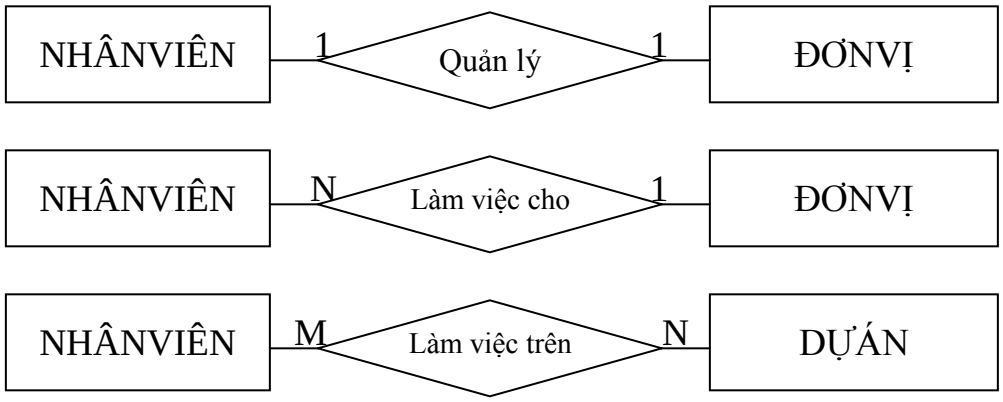
Đôi khi chúng ta có thể coi một kiểu liên kết như một thuộc tính của một kiểu thực thể. Ví dụ, nếu kiểu thực thể NHÂNVIÊN có thuộc tính Đơnvị để chỉ ra tên đơn vị mà nhân viên làm việc cho, thì thuộc tính Đơnvị biểu thị một kiểu liên kết. Nói cách khác, một thuộc tính của một kiểu thực thể hoặc có chức năng biểu thị một đặc trưng của kiểu thực thể, hoặc có chức năng biểu thị một kiểu liên kết giữa kiểu thực thể đó với các kiểu thực thể khác. Các thuộc tính biểu thị một kiểu liên kết có thể đơn trị hoặc đa trị tuỳ theo bản chất của mối liên kết.

Các tên vai trò và các kiểu liên kết đệ quy: Mỗi một kiểu thực thể tham gia vào một kiểu liên kết có một vai trò cụ thể trong liên kết. *Tên vai trò* dùng để chỉ rõ vai trò của các thực thể của kiểu thực thể tham gia liên kết, nó giúp đỡ việc giải thích ý nghĩa của liên kết. Ví dụ, trong kiểu liên kết NHÂNVIÊN <làm việc cho > ĐƠNVỊ, vai trò của các thực thể của kiểu thực thể NHÂNVIÊN là nhân viên hoặc công nhân còn vai trò của các thực thể của kiểu thực thể ĐƠNVỊ là đơn vị hoặc nơi thuê công nhân. Nếu các kiểu thực thể tham gia vào kiểu liên kết là khác nhau thì tên vai trò là hoàn toàn không cần thiết bởi vì có thể sử dụng tên các kiểu thực thể làm tên vai trò. Tuy nhiên, trong một số trường hợp, một kiểu thực thể có thể tham gia vào một kiểu liên kết với các vai trò khác nhau. Trong những trường hợp như vậy, tên vai trò trở nên cần thiết để phân biệt ý nghĩa của mỗi sự tham gia. Các kiểu liên kết như vậy gọi là *kiểu liên kết đệ quy*. Ví dụ, trong số các nhân viên làm việc cho một đơn vị, có các nhân viên được phân công giám sát các nhân viên khác. Như vậy sẽ có một kiểu liên kết giữa các thực thể của kiểu thực thể NHÂNVIÊN: NHÂNVIÊN <giám sát> NHÂNVIÊN. Kiểu thực thể NHÂNVIÊN tham gia hai lần vào kiểu liên kết <giám sát>, một lần với vai trò người giám sát, một lần với vai trò người bị giám sát.

2.5- Các ràng buộc trên các kiểu liên kết

Các kiểu liên kết thường có một số ràng buộc để hạn chế số các tổ hợp có thể của các thực thể có thể tham gia trong tập hợp liên kết tương ứng. Các ràng buộc này được xác định từ tình trạng của thế giới thực mà kiểu liên kết biểu diễn. Ví dụ, nếu công ty có quy chế là một nhân viên chỉ làm việc cho một đơn vị thì chúng ta phải mô tả ràng buộc này trong lược đồ. Có hai loại ràng buộc chính: tỷ số lực lượng và sự tham gia.

Tỷ số lực lượng: Tỷ số lực lượng cho một kiểu liên kết chỉ ra số các thể hiện liên kết mà một thực thể có thể tham gia. Với các kiểu liên kết cấp 2, có thể có các tỷ số lực lượng 1:1, 1:N, và M:N. Một kiểu liên kết có tỷ số lực lượng 1:1 giữa hai kiểu thực thể A và B có nghĩa là trong kiểu liên kết đó, một thực thể của kiểu A chỉ liên kết với một thực thể của kiểu B và ngược lại, một thực thể của kiểu B chỉ liên kết với một thực thể của kiểu A. Tỷ số lực lượng 1:N có nghĩa là một thực thể của kiểu A có thể liên kết với nhiều thực thể của kiểu B nhưng một thực thể của kiểu B chỉ liên kết với một thực thể của kiểu A. Trong kiểu liên kết có tỷ số lực lượng M:N, mỗi thực thể của kiểu A có thể liên kết với nhiều thực thể của kiểu B và ngược lại, mỗi thực thể của kiểu B có thể liên kết với nhiều thực thể của kiểu A. Trong biểu diễn của lược đồ ER, các tỷ số lực lượng được biểu diễn bằng cách ghi 1, N, M trên các hình thoi biểu diễn kiểu liên kết (hình II-5) .



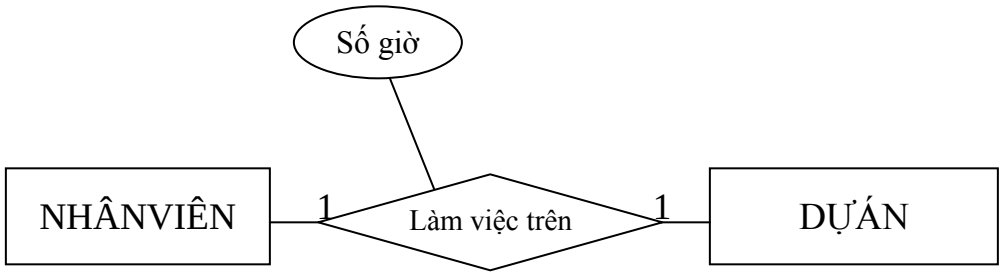
Hình 0-5. Tỷ số lực lượng của các kiểu liên kết

Các ràng buộc tham gia và sự phụ thuộc tồn tại: Ràng buộc tham gia chỉ ra rằng có phải sự tồn tại của một kiểu thực thể phụ thuộc vào một kiểu thực thể khác thông qua một kiểu liên kết hay không. Có hai kiểu ràng buộc tham gia: ràng buộc tham gia toàn bộ và ràng buộc tham gia bộ phận. Tham gia toàn bộ nghĩa là tất cả các thực thể của kiểu thực thể phải tham gia vào kiểu liên kết còn tham gia bộ phận nghĩa là chỉ một bộ phận các thực thể của kiểu thực thể tham gia vào kiểu liên kết. Ví dụ, xét kiểu liên kết NHÂNVIÊN <*quản lý*> ĐƠNVỊ. Trong thực tế, mỗi đơn vị phải có một người quản lý (là một nhân viên) nhưng không phải nhân viên nào cũng quản lý một đơn vị. Như vậy, sự tham gia của các thực thể đơn vị vào kiểu liên kết là toàn bộ còn sự tham gia của các thực thể nhân viên vào kiểu liên kết là bộ phận. Sự tham gia toàn bộ còn được gọi là *sự phụ thuộc tồn tại*.

Trong lược đồ ER, sự tham gia toàn bộ được biểu thị bằng đường nối đôi từ kiểu thực thể đến kiểu liên kết. Ví dụ :

2.6- Thuộc tính của các kiểu liên kết

Các kiểu liên kết cũng có thể có các thuộc tính, giống như các thuộc tính của các kiểu thực thể. Ví dụ, kiểu liên kết <*làm việc trên*> giữa các kiểu thực thể NHÂNVIÊN và DỰÁN có thể có thuộc tính Sốgiờ để ghi lại số giờ làm việc của một nhân viên trên một dự án. Các thuộc tính của kiểu liên kết cũng được biểu diễn bằng một hình ô van và được nối với kiểu liên kết. Ví dụ:



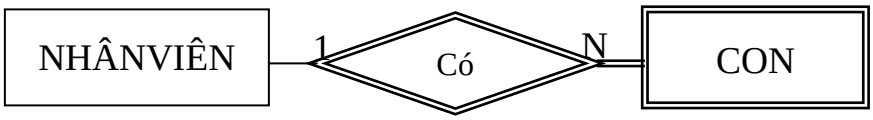
2.7- Các kiểu thực thể yếu

Các kiểu thực thể không có các thuộc tính khoá cho chính mình được gọi là các kiểu thực thể yếu. Ngược lại, các kiểu thực thể thông thường (nghĩa là có thuộc tính khoá) được gọi là kiểu thực thể mạnh. Các thực thể của một kiểu thực thể yếu được xác định bằng cách liên kết với các thực thể cụ thể của một kiểu thực thể khác phối hợp với một số giá trị thuộc tính của nó. Ta gọi kiểu thực thể khác đó là kiểu thực thể xác định hoặc *kiểu thực thể chủ.* Ta gọi kiểu liên kết giữa kiểu thực thể yếu và kiểu thực thể chủ của nó là liên kết xác định của thực thể yếu. Một kiểu thực thể yếu luôn luôn có một ràng buộc tham gia toàn bộ (tồn tại phụ thuộc) vào liên kết xác định của nó bởi vì một kiểu thực thể yếu không thể được xác định mà không có kiểu thực thể chủ. Ví dụ, trong một công ty, con của nhân viên và nhân viên có thể hưởng chế độ bảo hiểm theo nhân viên. Như vậy, sẽ có một kiểu liên kết NHÂNVIÊN <*có*> < CON>. Đây là một kiểu liên kết có tỷ số lực lượng 1:N. Các thuộc tính của kiểu thực thể CON là Họtên, Ngaysinh, Giớitính. Hai người con của hai nhân viên khác nhau có thể có cùng giá trị cho các thuộc tính nhưng nó là hai thực thể khác nhau. Chúng chỉ được xác định như hai thực thể khác nhau sau khi xác định một thực thể nhân viên cụ thể có liên quan đến từng người phụ thuộc. Mỗi

thực thể của kiểu thực thể NHÂNVIÊN được gọi là chủ của các thực thể của kiểu thực thể CON liên kết với nó.

Thông thường, các kiểu thực thể yếu có một *khoá bộ phận,* đó là một tập hợp các thuộc tính có thể xác định một cách duy nhất các thực thể yếu liên kết với cùng một thực thể chủ. Ví dụ, nếu hai người con của một nhân viên không bao giờ có tên giống nhau thì thuộc tính Họtên của kiểu thực thể CON là một khoá bộ phận. Trong trường hợp xấu nhất, thuộc tính phức hợp gồm tất cả các thuộc tính của thực thể yếu sẽ là một khoá bộ phận.

Trong sơ đồ ER, kiểu thực thể yếu và kiểu liên kết xác định của nó được biểu diễn bằng một hình chữ nhật và một hình thoi nét đôi. Ví dụ:



3. Ví dụ về thiết kế mô hình ER

Trong phần này, chúng ta xét ví dụ về việc xây dựng mô hình ER cho cơ sở dữ liệu công ty. Như ở trong phần I đã nói, bước đầu tiên trong việc thiết kế một cơ sở dữ liệu là tập hợp và phân tích các yêu cầu. Kết quả của bước này là một tập hợp các ghi chép súc tích về các yêu cầu người sử dụng cũng như tình trạng của nơi ta cần xây dựng cơ sở dữ liệu.

Giả sử rằng sau khi tập hợp các yêu cầu và phân tích, hoạt động của công ty được ghi chép lại như sau:

1. Công ty được tổ chức thành các đơn vị. Mỗi đơn vị có một tên duy nhất, một mã số duy nhất, một nhân viên cụ thể quản lý đơn vị. Việc nhân viên quản lý đơn vị được ghi lại bằng ngày nhân viên đó bắt đầu quản lý. Một đơn vị có thể có nhiều địa điểm.

2. Mỗi đơn vị kiểm soát một số dự án. Một dự án có một tên duy nhất, một mã số duy nhất và một địa điểm.

3. Với mỗi nhân viên chúng ta lưu giữ lại Họ tên, Mã số, địa chỉ, lương, giới tính, ngày sinh. Một nhân viên chỉ làm việc cho một đơn vị nhưng có thể làm việc trên nhiều dự án do nhiều đơn vị kiểm soát. Chúng ta lưu giữ lại số giờ làm việc của mỗi nhân viên trên một dự án. Mỗi nhân viên có thể có một người giám sát trực tiếp, người đó cũng là một nhân viên.

4. Mỗi nhân viên có những người con. Những người này được hưởng bảo hiểm theo nhân viên. Với mỗi người con của nhân viên, chúng ta lưu giữ Họ tên, giới tính, ngày sinh .

3.1- Xác định các kiểu thực thể, các thuộc tính và các kiểu liên kết

Theo các ghi chép ở trên, chúng ta có thể xác định các kiểu thực thể và các kiểu liên kết như sau:

1. CÔNGTY không phải là một kiểu thực thể vì ở đây ta có một công ty duy nhất.

2. ĐƠNVỊ là một kiểu thực thể với các thuộc tính Tên, Mãsố, Ngườiquảnlý, Ngàybắtđầu và Địađiểm. Các thuộc tính Tên, Mãsố, Địađiểm là các thuộc tính mô tả đơn vị, các thuộc tính Ngườiquảnlý, Ngàybắtđầu là các thuộc tính biểu thị một kiểu liên kết (với kiểu thực thể NHÂNVIÊN). Các thuộc tính đều là đơn và đơn trị, trừ thuộc tính Địađiểm, nó là một thuộc tính đa trị (một đơn vị có nhiều địa điểm). Các thuộc tính Tên, Mãsố là các thuộc tính khóa (vì mỗi đơn vị có một tên và một mã số duy nhất).

3. Kiểu thực thể DỰÁN có các thuộc tính Tên, Mãsố, Địađiểm, Đơnvịkiểmsoát. Các thuộc tính Tên, Mãsố, Địa điểm là các thuộc tính mô tả DỰÁN, thuộc tính Đơnvịkiểmsoát biểu thị kiểu liên kết với kiểu thực thể ĐƠNVỊ (một đơn vị kiểm soát một số dự án). Các thuộc tính Tên, Mãsố là các thuộc tính khóa.

4. Kiểu thực NHÂNVIÊN với các thuộc tính Họtên, Mãsố, Giới tính, Ngàysinh, Lương, Đơnvị, Ngườigiámsát. Thuộc tính Họtên là một thuộc tính phức hợp (gồm Họđệm, Tên). Các thuộc tính Đơnvị, Ngườigiámsát mô tả các kiểu liên kết giữa kiểu thực thể NHÂNVIÊN và các kiểu thực thể ĐƠNVỊ và NHÂNVIÊN tương ứng. Thuộc tính Mãsố là thuộc tính khóa.

5. Kiểu thực thể CON với các thuộc tính Nhânviên, Họtên, Giới tính, Ngàysinh. Thuộc tính Nhânviên mô tả kiểu liên kết với kiểu thực thể NHÂNVIÊN.

6. Kiểu liên kết ĐƠNVỊ <kiểm soát> DỰÁN là kiểu liên kết có tỷ số lực lượng 1:N (một đơn vị kiểm soát một số dự án, một dự án do một đơn vị quản lý). Sự tham gia của DỰÁN vào kiểu liên kết là toàn bộ (bởi vì dự án nào cũng được một đơn vị kiểm soát). Nếu đơn vị nào cũng có dự án thì việc tham gia của ĐƠNVỊ vào kiểu liên kết là toàn bộ, ngược lại sự tham gia là bộ phận.

7. Kiểu liên kết NHÂNVIÊN <làm việc cho> ĐƠNVỊ có tỷ số lực lượng N:1 (mỗi nhân viên làm việc cho một đơn vị nhưng mỗi đơn vị có nhiều nhân viên là việc). Sự tham gia của hai kiểu thực thể vào liên kết là toàn bộ.

8. Kiểu liên kết NHÂNVIÊN <quảnlý> ĐƠNVỊ có tỷ số lực lượng 1:1 (một nhân viên quản lý một đơn vị và một đơn vị có một nhân viên quản lý). Sự tham gia của kiểu thể NHÂNVIÊN vào kiểu liên kết là bộ phận (bởi vì không phải nhân viên nào cũng quản lý đơn vị), ngược lại, sự tham gia của kiểu thực thể ĐƠNVỊ vào kiểu liên kết là toàn bộ (bởi vì đơn vị nào cũng phải có người quản lý).

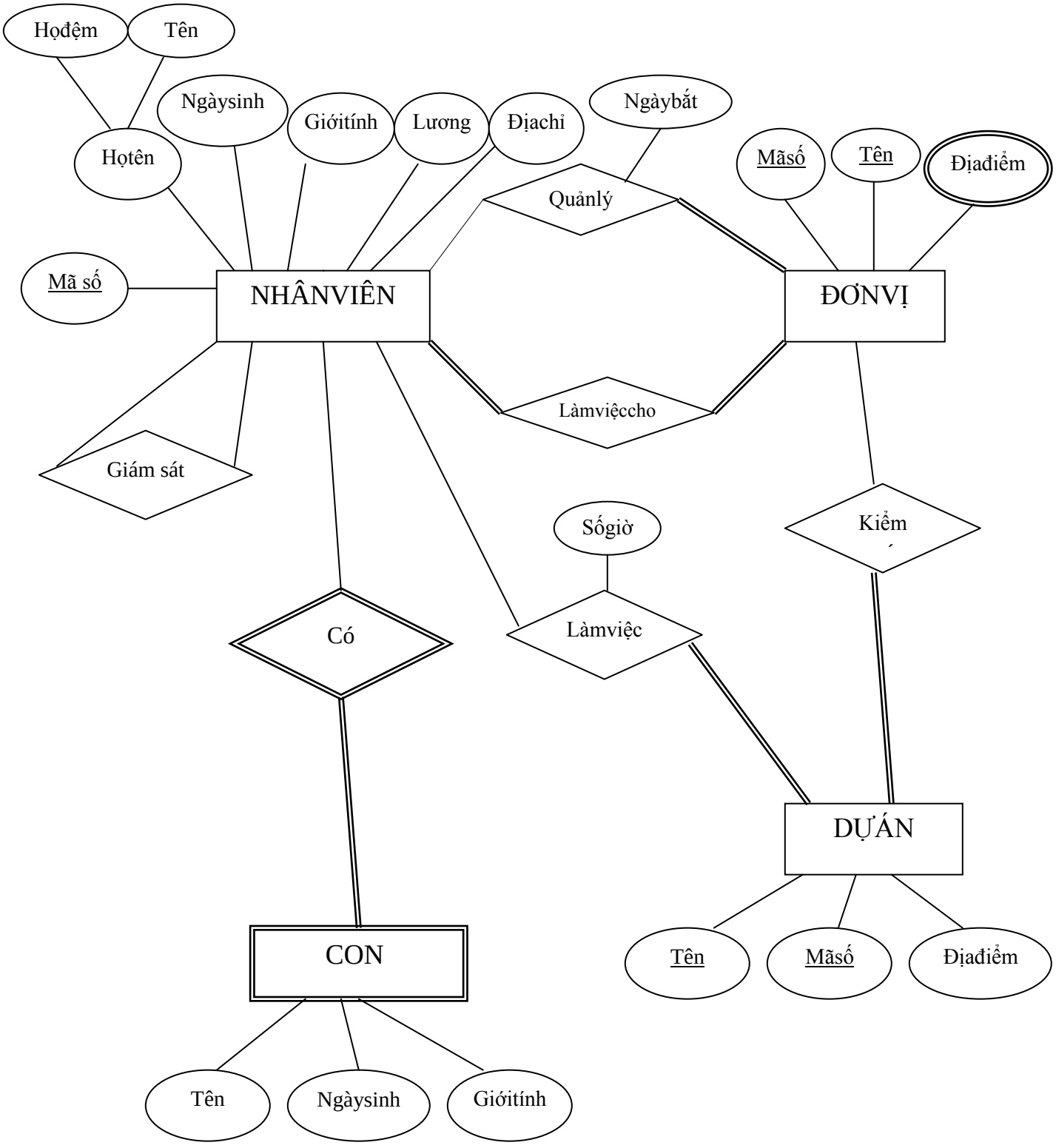
9. Kiểu liên kết NHÂNVIÊN <giám sát > NHÂNVIÊN có tỷ số lực lượng 1:N (một nhân viên có thể giám sát nhiều nhân viên khác). Sự tham gia của của kiểu thực thể NHÂNVIÊN (ở cả hai phía) là bộ phận (bởi vì không phải nhân viên nào cũng giám sát nhân viên khác, và không phải nhân viên nào cũng bị giám sát). Kiểu thực thể NHÂNVIÊN ở đây đóng hai vai trò khác nhau: vai trò người giám sát và vai trò người bị giám sát.

10. Kiểu liên kết NHÂNVIÊN <làm việc trên> DỰÁN là có tỷ số lực lượng là M:N (một nhân viên có thể làm việc trên nhiều dự án khác nhau và mỗi dự án có nhiều nhân viên làm việc). Sự tham gia của kiểu thực thể NHÂNVIÊN là bộ phận (bởi vì không phải tất cả nhân viên đều làm việc trên dự án) ngược lại, sự tham gia của kiểu thực thể DỰÁN là toàn bộ (bởi vì dự án nào cũng phải có nhân viên làm việc). Kiểu liên kết này có một thuộc tính là Sốgiờ, ghi lại số giờ làm việc của một nhân viên trên một dự án.

11. Kiểu liên kết NHÂNVIÊN <có> CON biểu thị mối liên hệ giữa kiểu thực thể NHÂNVIÊN và kiểu thực thể CON (một nhân viên có thể có những người con). Kiểu liên kết này có tỷ số lực lượng 1:N (một nhân viên có thể có nhiều người con nhưng mỗi con là con của chỉ một nhân viên). Sự tham gia của kiểu thực thể NHÂNVIÊN là bộ phận (không phải nhân viên nào cũng có con), ngược lại, sự tham gia của kiểu thực thể CON là toàn bộ (người con nào cũng phải là con của một nhân viên). Ngoài ra, kiểu thực thể CON là một kiểu thực thể yếu.

Sau khi phân tích như trên, để vẽ lược đồ ER ta loại bỏ các thuộc tính được xem như các kiểu liên kết ra khỏi các kiểu thực thể. Đó là các thuộc tính Ngườiquảnlý và Ngàybắtđầu của kiểu thực thể ĐƠNVỊ, thuộc tính Đonvịkiểmsoát của kiểu thực thể DỰÁN, thuộc tính Đonvị và thuộc tính Ngườigiámsát của kiểu thực thể NHÂNVIÊN, thuộc tính Nhânviên của kiểu thực thể PHỤTHUỘC .

Kết quả, chúng ta có lược đồ ER như sau:



Hình 0-6. Lược đồ ER "CÔNGTY"

4. Mô hình thực thể liên kết mở rộng (mô hình EER)

Một cách truyền thống, khi xây dựng một cơ sở dữ liệu chúng ta thường bắt đầu bằng việc xây dựng mô hình liên kết – thực thể (mô hình ER) rồi sau đó chuyển đổi nó thành mô hình quan hệ. Các khái niệm về mô hình ER có thể được coi là khá đầy đủ để trình bày các lược đồ cơ sở dữ liệu trong các ứng dụng cơ sở dữ liệu truyền thống, chủ yếu là các ứng dụng xử lý dữ liệu trong kinh doanh và trong công nghiệp. Ngày nay, các ứng dụng mới hơn cho công nghệ cơ sở dữ liệu đã trở nên phổ biến. Các cơ sở dữ liệu loại này đòi hỏi những yêu cầu phức tạp hơn so với các ứng dụng truyền thống. Để trình bày được các yêu cầu này một cách chính xác và rõ ràng, người thiết kế cơ sở dữ liệu phải sử dụng thêm các khái niệm mới. Việc thêm vào mô hình ER những khái niệm mới làm mở rộng mô hình này và tạo nên mô hình ER mở rộng (gọi tắt là mô hình EER – Enhanced Entity Relationship Model).

Mô hình EER bao gồm tất cả các khái niệm của mô hình ER, ngoài ra còn có các khái niệm như lớp, kiểu liên kết lớp cha/ lớp con, tính thừa kế, chuyên biệt, tổng quát, phạm trù.

4.1- Lớp cha, lớp con và sự thừa kế

Khái niệm đầu tiên trong mô hình EER là *lớp con* của một kiểu thực thể. Như ta đã biết, kiểu thực thể được sử dụng để biểu diễn cả kiểu của thực thể và tập hợp các thực thể cùng một kiểu trong cơ sở dữ liệu. Trong nhiều trường hợp, một kiểu thực thể có thể có các nhóm con các thực thể của nó và những nhóm con này cần được trình bày rõ ràng do ý nghĩa của nó đối với cơ sở dữ liệu.

Ví dụ, các thực thể thành viên của kiểu thực thể NHÂNVIÊN có thể được chia thành các nhóm nhỏ: KỸSƯ, NGƯỜIQUẢNLÝ, KỸTHUẬTVIÊN… Tập các thực thể trong các nhóm đó là một tập con của các thực thể trong tập thực thể nhân viên, nghĩa là mỗi thực thể là thành viên của một trong những nhóm này cũng là một nhân viên. Chúng ta gọi mỗi nhóm này là một *lớp con* của kiểu thực thể NHÂNVIÊN. Kiểu thực thể NHÂNVIÊN được gọi là *lớp cha* của các lớp con đó. Ta gọi quan hệ giữa lớp cha và một trong những lớp con của nó là *kiểu liên kết lớp cha/ lớp con*. Kiểu liên kết lớp cha/ lớp con thường được gọi là kiểu liên kết *là một (IS_A).* Chúng ta thường nói rằng một kỹ sư là một nhân viên, một kỹ thuật viên là một nhân viên.

Chú ý rằng một thực thể thành viên trong lớp con cùng biểu diễn một thực thể thực tại như một thành viên trong lớp cha, vì vậy, các thực thể thành viên trong lớp con và các thực thể thành viên trong lớp cha là giống nhau, nhưng vai trò của chúng hoàn toàn khác nhau. Khi chúng ta tạo một kiểu liên kết lớp cha/ lớp con trong hệ thống cơ sở dữ liệu, chúng ta có thể trình bày một thành viên của lớp con như như là một đối tượng riêng biệt, một bản ghi riêng biệt kết hợp với các thực thể của lớp cha của nó qua thuộc tính khoá. Kiểu liên kết lớp cha/ lớpcon là một kiểu liên kết có tỷ số lực lượng 1:1.

Một khái niệm quan trọng gắn với các lớp con là *sự thừa kế kiểu*. Kiểu của một thực thể được xác định bằng các thuộc tính và các kiểu liên kết mà nó tham gia. Vì mỗi thực thể thành viên trong lớp con cùng biểu diễn một thực thể thực tại như thực thể trong lớp cha nên các giá trị của thuộc tính của nó trong lớp con cũng phải giống như là giá trị của các thuộc tính của nó khi nó đóng vai trò là một thành viên trong lớp cha. Thực thể này cũng được thừa kế các liên kết trong lớp cha. Một lớp con với các thuộc tính riêng của nó cùng với tất cả các thuộc tính và kiểu liên kết kế thừa được từ lớp cha có quyền được coi như là một kiểu thực thể.

4.2- Chuyên biệt hoá, tổng quát hoá

4.2.1- <u>Chuyên biệt hoá</u>

Là quá trình xác định tập hợp các lớp con của một kiểu thực thể. Kiểu thực thể này được gọi là lớp cha trong chuyên biệt hoá. Tập các lớp con tạo nên một chuyên biệt hoá được xác định dựa trên cơ sở một đặc trưng phân biệt nào đó của các thực thể trong lớp cha. Ví dụ, tập các lớp con {THƯKÝ, KỸSƯ, KỸTHUẬTVIÊN} là một chuyên biệt hoá của lớp cha NHÂNVIÊN được xác định dựa trên *kiểu công việc* của các thực thể. Một kiểu thực thể có thể có một số chuyên biệt hoá dựa trên các đặc trưng khác nhau. Ví dụ, một chuyên biệt hoá khác của kiểu thực thể NHÂNVIÊN sinh ra tập các lớp con {NHÂNVIÊN_BIÊNCHẾ, NHÂNVIÊN_HỢPĐỒNG}. Trong chuyên biệt hoá này, các thực thể được phân biệt dựa trên cơ sở *hình thức trả tiền*.

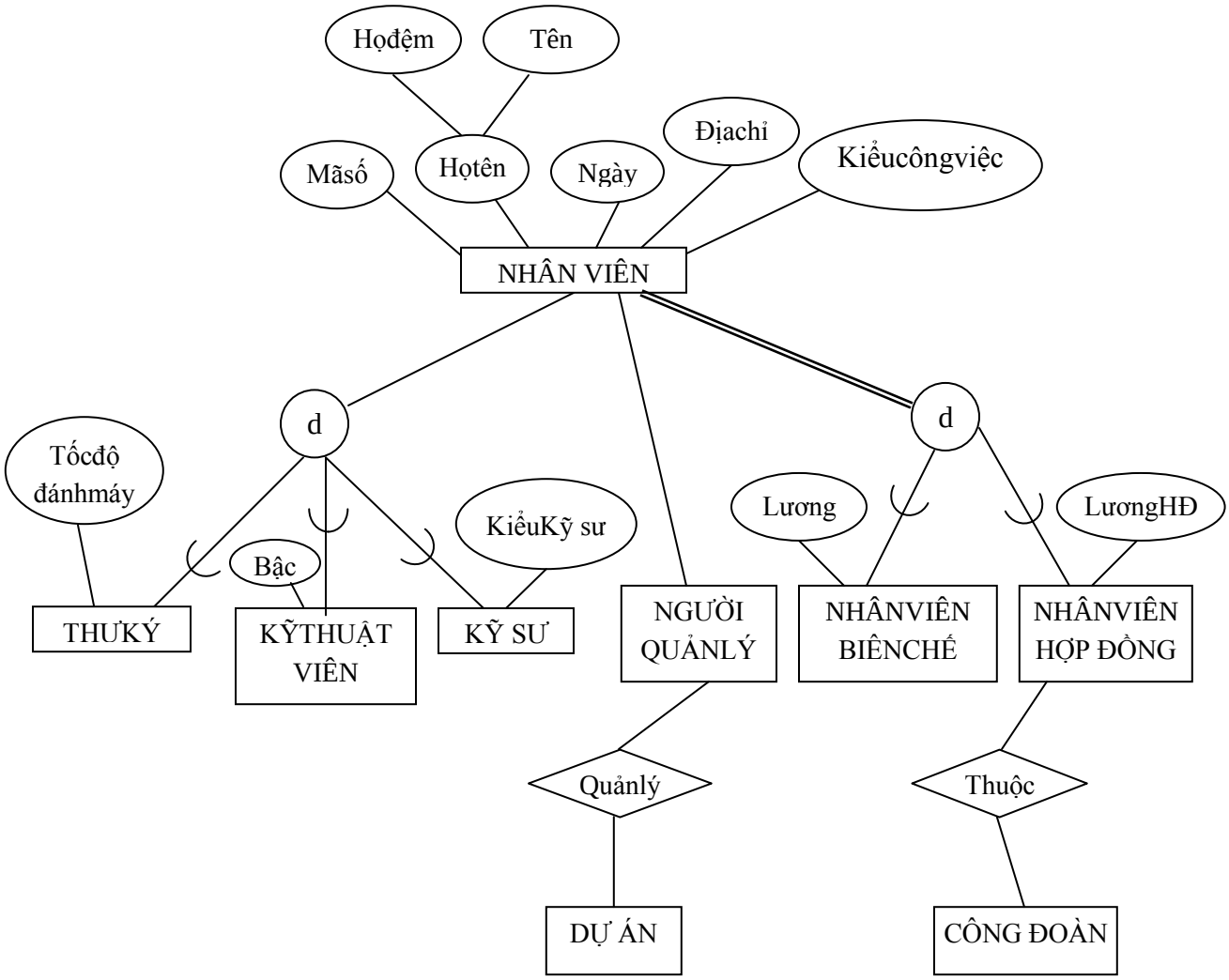
Một chuyên biệt hoá được biểu diễn trong sơ đồ EER như sau: Các lớp con xác định một chuyên biệt hoá được nối bằng các đường đến một vòng tròn, vòng tròn đó được nối với lớp cha. Ký hiệu *tập con* (trên mỗi đường nối một tập con với vòng tròn chỉ hướng của kiểu liên kết lớp cha / lớp con). Các thuộc tính chỉ áp dụng

cho các thực thể của một lớp con cụ thể - ví dụ như Tốcđộđánhmáy của lớp con THƯKÝ - được nối với hình chữ nhật biểu diễn lớp con đó. Các thuộc tính như vậy gọi là các thuộc tính riêng hoặc là các thuộc tính địa phương của lớp con. Tương tự, một lớp con có thể tham gia vào các kiểu liên kết riêng, ví dụ, lớp con NHÂNVIÊN_HỢPĐỒNG tham gia vào kiểu liên kết <thuộc về> (hình II-7).

Có hai lý do chính để đặt các kiểu liên kết lớp cha/lớp con và chuyên biệt hoá vào mô hình dữ liệu. Thứ nhất là có một số thuộc tính có thể áp dụng cho một số các thực thể chứ không phải cho toàn bộ các thực thể của lớp cha. Khi đó, một lớp con sẽ được xác định để nhóm các thực thể mà các thuộc tính đó có thể áp dụng được. Các thành viên của lớp con này có thể vẫn chia sẻ phần lớn các thuộc tính của chúng với các thành viên khác của lớp cha.Ví dụ, lớp con THƯKÝ có thuộc tính riêng là Tốcđộđánhmáy, lớp con KỸSƯ có thuộc tính riêng là Kiểukỹsư nhưng các thuộc tính khác của chúng là chung với kiểu thực thể NHÂNVIÊN. Lý do thứ hai là chỉ có các thành viên của lớp con có thể tham gia vào một số kiểu liên kết nào đó. Ví dụ, nếu chỉ có các nhân viên hợp đồng mới tham gia và công đoàn thì chúng ta có thể diễn đạt sự kiện đó bằng cách tạo ra một lớp con NHÂNVIÊN_HỢPĐỒNG của NHÂNVIÊN và liên kết lớp con này với kiểu thực thể CÔNGĐOÀN thông qua kiểu thực thể <thuộc về>.

Tóm lại, quá trình chuyên biệt hoá cho phép chúng ta làm các việc sau:

- Xác định một tập hợp các lớp con của một kiểu thực thể.
- Thiết lập các thuộc tính riêng cho mỗi lớp con.
- Thiết lập các kiểu liên kết riêng giữa mỗi lớp con và các kiểu thực thể khác hoặc các lớp con khác.

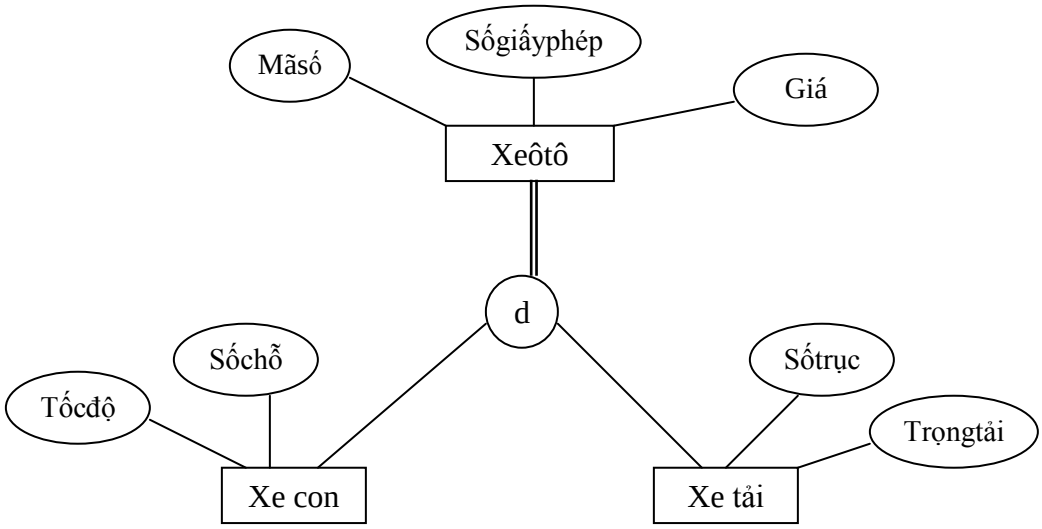


Hình 0-7. Biểu diễn lược đồ EER của chuyên biệt hoá

*4.2.2- **Tổng quát hoá***

Là quá trình đảo ngược của chuyên biệt hoá, trong đó ta bỏ qua sự khác nhau giữa một số kiểu thực thể, xác định các đặc tính chung của chúng và tổng quát hoá chúng thành một lớp cha của các kiểu thực thể đó. Ví dụ, ta có kiểu thực thể XECON với các thuộc tính (Mãsố, Sốgiấyphép, Giá, Tốcđộtốiđa, Sốchỗngồi) và kiểu thực thể XETẢI với các thuộc tính (Mãsố, Sốgiấyphép, Giá, Trọngtải, Sốcáctrục), các kiểu thực thể này có một số thuộc tính chung, chúng có thể được tổng quát hoá thành kiểu thực thể XEÔTÔ với các thuộc tính (Mãsố, Sốgiấyphép, Giá). Các kiểu thực thể XECON và XETẢI trở thành các lớp con của lớp cha XEÔTÔ. Như vậy, tổng quát hoá là quá trình tổng quát một kiểu thực thể từ các kiểu thực thể cho trước.

Một tổng quát hoá được biểu diễn trong sơ đồ EER giống như là một chuyên biệt hoá. Tổng quát là lớp cha còn chuyên biệt là các lớp con được sử dụng để tạo nên lớp cha (hình II-8).



Hình 0-8. Xeôtô được tổng quát hoá từ Xecon và Xetải

4.2.3- ***Phân cấp chuyên biệt và lưới chuyên biệt***

Bản thân các lớp con cũng có các lớp con của nó. Như vậy sẽ tạo ra một phân cấp chuyên biệt hoá hoặc một lưới chuyên biệt hoá. Một *phân cấp chuyên biệt hoá* có ràng buộc là một lớp con chỉ tham gia vào một kiểu liên kết lớp cha/lớp con như là một lớp con. Một *lưới chuyên biệt* có điều kiện là một lớp con có thể tham gia vào nhiều kiểu liên kết lớpcha/lớpcon như là một lớp con. Nói cách khác, một lớp con trong phân cấp chuyên biệt chỉ thừa kế một lớp cha, ngược lại, một lớp con trong lưới chuyên biệt có thể thừa kế nhiều lớp cha. Một lớp con thừa kế nhiều lớp cha thuộc các kiểu khác nhau được gọi là một *kiểu hợp (union type)* hoặc một *phạm trù* (category).

4.2.4- ***Các ràng buộc và các đặc trưng của chuyên biệt hoá, tổng quát hoá***

Trong một số chuyên biệt hoá, chúng ta có thể xác định một cách chính xác các thực thể sẽ là thành viên của một lớp con bằng cách đặt một điều kiện trên một thuộc tính nào đấy của lớp cha. Các lớp con như vậy được gọi là *các lớp con được xác định bằng điều kiện*. Nếu các lớp con của một chuyên biệt hoá có điều kiện thành viên trên cùng một thuộc tính của lớp cha thì chuyên biệt hoá đó cũng được gọi là chuyên biệt hoá *được xác định bằng thuộc tính*. Nếu việc xác định một lớp

con không theo một điều kiện nào thì lớp con đó được gọi là *được người sử dụng xác định.*

Có hai ràng buộc áp dụng cho một chuyên biệt hoá. *Ràng buộc rời rạc* chỉ ra rằng các lớp con của một chuyên biệt phải rời rạc. Điều này có nghĩa là một thực thể có thể là một thành viên của nhiều nhất là một trong số các lớp con của chuyên biệt hoá. Một chuyên biệt hoá được xác định bằng thuộc tính thoả mãn ràng buộc rời rạc nếu thuộc tính được sử dụng để xác định thành viên là đơn trị. Nếu các lớp con không thoả mãn ràng buộc rời rạc, các tập thực thể của chúng có thể chồng chéo nhau, nghĩa là một thực thể có thể là một thành viên của nhiều lớp con trong chuyên biệt hoá. Ràng buộc thứ hai trong chuyên biệt hoá gọi là *ràng buộc đầy đủ,* nó có thể là toàn bộ hoặc từng phần. Một ràng buộc chuyên biệt toàn bộ chỉ ra rằng mỗi thực thể trong lớp cha phải là một thành viên của một lớp con nào đó trong chuyên biệt. Một ràng buộc chuyên biệt từng phần cho phép một thực thể của lớp cha không thuộc về bất kỳ lớp con nào. Ví dụ, nếu một nhân viên phải hoặc là một nhân viên biên chế hoặc là một nhân viên hợp đồng thì (NHÂNVIÊN_BIÊNCHẾ, NHÂNVIÊN_HỢPĐỒNG) là một chuyên biệt toàn bộ của NHÂNVIÊN. Nếu một nhân viên có thể không phải là một thư ký, một kỹ sư hoặc một kỹ thuật viên thì chuyên biệt (THƯKÝ, KỸSƯ, KỸTHUẬTVIÊN) là một chuyên biệt từng phần của NHÂNVIÊN.

Trong sơ đồ của mô hình EER, nếu một chuyên biệt hoá là rời rạc thì ở giữa hình tròn nối với các lớp con có ghi chữ d (disjoin), còn một chuyên biệt là chồng chéo thì ở giữa hình tròn nối các lớp con có ghi chữ o (overlap).

4.3- Sơ đồ mô hình EER

Mô hình EER có biểu diễn đồ hoạ giống như mô hình ER, nghĩa là các kiểu thực thể (các lớp) được biểu diễn bằng các hình chữ nhật có ghi tên ở giữa, các thuộc tính của chúng được biểu diễn bằng các hình ô van nối với hình chữ nhật. Các kiểu liên kết được biểu diễn bằng các hình thoi và được nối với các kiểu thực thể tham gia liên kết. Tại các hình thoi có ghi rõ các tỷ số lực lượng tham gia của các kiểu thực thể tham gia vào kiểu liên kết. Ngoài ra, kiểu liên kết lớpcha/lớpcon được biểu diễn bằng một đường nối có thêm một ký hiệu tập con "⊂" ở giữa đường nối. Các lớp con trong một chuyên biệt được nối với một vòng tròn và vòng tròn được nối với lớp cha. Nếu chuyên biệt là rời rạc, giữa vòng tròn sẽ ghi chữ d, nếu chuyên biệt là chồng chéo, giữa vòng tròn có ghi chữ o.

5. Tổng kết chương và câu hỏi ôn tập

5.1- Tổng kết chương

Trong chương này chúng ta đã thảo luận về vai trò của mô hình dữ liệu bậc cao trong quá trình thiết kế cơ sở dữ liệu. Ta đã làm quen với các khái niệm cơ bản của mô hình liên kết - thực thể: kiểu thực thể, kiểu liên kết, và các thuộc tính của chúng. Các kiểu thuộc tính khác nhau cũng đã được xem xét: thuộc tính đơn, thuộc tính phức hợp, thuộc tính đơn trị, thuộc tính đa trị, thuộc tính lưu trữ, thuộc tính suy diễn được và các thuộc tính có giá trị null. Thông qua một ví dụ cụ thể, ta đã tiến hành xây dựng mô hình ER "CÔNGTY". Ngoài ra, chúng ta cũng đã nói đến mô hình EER, mở rộng của mô hình ER. Các khái niệm "mở rộng" như lớp, lớp con, kiểu liên kết lớp cha/lớp con, chuyên biệt hoá, tổng quát hoá cũng đã được giới thiệu và phân tích. Chúng ta cũng đã nói đến cách biểu diễn đồ hoạ của các mô hình ER và EER.

5.2- Câu hỏi ôn tập

1- Hãy nói về vai trò của mô hình dữ liệu bậc cao trong quá trình thiết kế cơ sở dữ liệu.

2- Liệt kê các trường hợp cần phải sử dụng giá trị null.

3- Định nghĩa các thuật ngữ sau: thực thể, thuộc tính, giá trị thuộc tính, thể hiện liên kết, thuộc tính phức hợp, thuộc tính đa trị, thuộc tính suy diễn được, thuộc tính phức tạp, thuộc tính khoá, miền giá trị.

4- Kiểu thực thể là gì? Tập thực thể là gì? Giải thích sự khác nhau giữa một thực thể, một kiểu thực thể và một tập thực thể.

5- Giải thích sự khác nhau giữa một thuộc tính và một tập giá trị.

6 - Kiểu liên kết là gì? Giải thích sự khác nhau giữa một thể hiện liên kết, một tập liên kết và một kiểu liên kết.

7- Vai trò tham gia là gì? Khi nào cần phải sử dụng các tên vai trò trong mô tả các kiểu liên kết.

8- Mô tả cách chỉ ra các ràng buộc cấu trúc trên các kiểu liên kết.

9- Với điều kiện nào một thuộc tính của một kiểu liên kết cấp 2 có thể chuyển thành một thuộc tính của một trong các kiểu thực thể tham gia vào kiểu liên kết.

10- Khi chúng ta nghĩ đến các liên kết như là các thuộc tính, các tập giá trị của các thuộc tính đó là gì?

11- Kiểu liên kết đệ quy là gì? Cho một số ví dụ về các kiểu liên kết đệ quy.

12- Khi nào khái niệm kiểu thực thể yếu được dùng trong mô hìn hoá cơ sở dữ liệu? Định nghĩa các thuật ngữ: kiểu thực thể chủ, kiểu thực thể yếu, khoá bộ phận, kiểu liên kết xác định.

13- Trình bày các khái niệm lớp, lớp con, chuyên biệt hoá, tổng quát hoá. Trong hoàn cảnh nào ta cần tách một lớp thành các lớp con.

14- Trình bày cách biểu diễn đồ hoạ của các mô hình ER và EER.

5.3- Bài tập

Bài 1: Xây dựng mô hình ER cho cơ sở dữ liệu TRƯỜNG

Hãy xây dựng lược đồ ER cho CSDL "TRƯỜNG", dựa trên các ghi chép sau:

1) Trường được chia thành các trường con: Trường KHTN, Trường KHXH, Trường Công nghệ,…. Mỗi trường có một hiệu trưởng quản lý. Mỗi hiệu trưởng quản lý một trường.
2) Mỗi trường có nhiều khoa. Chẳng hạn, trường KHTN có các khoa Toán, Lý, Hoá,… Mỗi một khoa chỉ thuộc về một trường. Thông tin về Khoa gồm Mã khoa, tên khoa, địa chỉ, số điện thoại, tên trường.
3) Mỗi Khoa cung cấp nhiều môn học. Mỗi môn học gồm có Tên môn học, mã số, số đơn vị học trình, trình độ, tên Khoa.
4) Mỗi môn học có thể có nhiều học phần.Mỗi học phần được lưu giữ bằng các thông tin: Mã học phần, Tên môn học, Tên giáo viên dạy, học kỳ.
5) Mỗi khoa có nhiều giáo viên làm việc, nhưng mỗi giáo viên chỉ làm việc cho một khoa. Mỗi một khoa có một chủ nhiệm khoa, đó là một giáo viên.
6) Mỗi giáo viên có thể dạy nhiều nhất là 4 học phần và cũng có thể không dạy học phần nào.
7) Mỗi sinh viên phải học nhiều học phần.
8) Mỗi một khoa có nhiều sinh viên, mỗi sinh viên chỉ thuộc về một khoa. Thông tin về mỗi sinh viên gồm: Mã sinh viên, Họ tên, địa chỉ, ngày sinh, giới tính, Lớp, Tên Khoa và chế độ đào tạo.

9) Mỗi sinh viên có một người giám sát (giáo viên chủ nhiệm), người đó là một giáo viên.

10) Sau mỗi học kỳ sẽ có một danh sách điểm để phân loại. Nó gồm các thông tin: Mã sinh viên, mã học phần, điểm bằng chữ, điểm bằng số.

Bài 2: Xây dựng mô hình ER cho cơ sở dữ liệu THƯ VIỆN.

Hãy xây dựng lược đồ ER cho CSDL "THƯ VIỆN", dựa trên các ghi chép sau:

1) Thư viện được chia ra thành các nhánh. Thông tin về mỗi nhánh gồm có Mã nhánh, Tên nhánh và Địa chỉ.

2) Mỗi cuốn sách trong thư viện có các thông tin về Mã sách, Tên sách Nhà xuất bản và Tác giả…

3) Một tác giả có thể viết nhiều cuốn sách. Một cuốn sách có thể có nhiều tác giả viết.

4) Một nhà xuất bản xuất bản nhiều cuốn sách. Một cuốn sách do một nhà xuất bản xuất bản. Thông tin về Nhà xuất bản gồm có Tên, Địachỉ và Sốđiệnthoại.

5) Một cuốn sách có thể có nhiều bản sao được lưu trữ tại các nhánh. Thông tin về bản sao sách gồm Mã sách, số các bản sao.

6) Thư viện có những người mượn sách. Thông tin về những người mượn sách gồm có Số thẻ, Họ tên, Địa chỉ và Số điện thoại.

7) Sách được cho các người mượn mượn tại các nhánh. Thông tin về một lần mượn gồm có Ngày mượn và ngày trả.

CHƯƠNG 3 - MÔ HÌNH QUAN HỆ, CÁC RÀNG BUỘC QUAN HỆ

Mô hình quan hệ được Ted Codd đưa ra đầu tiên vào năm 1970 và gây được chú ý ngay tức khắc vì tính đơn giản và các cơ sở toán học của nó. Mô hình quan hệ sử dụng khái niệm quan hệ toán học như là khối xây dựng cơ sở và có cơ sở lý thuyết của nó trong lý thuyết tập hợp và logic vị từ bậc nhất. Trong chương này chúng ta sẽ nói về các đặc trưng cơ bản của mô hình, các ràng buộc của chúng và tập hợp các phép toán của mô hình quan hệ.

1. Các khái niệm của mô hình quan hệ

Mô hình quan hệ biểu thị cơ sở dữ liệu như một tập các quan hệ. Mỗi quan hệ có thể được biểu diễn như một bảng giá trị, mỗi một dòng trong bảng biểu thị một tấp hợp các giá trị dữ liệu liên quan với nhau. Trong chương trước, chúng ta đã đưa ra các khái niệm về kiểu thực thể và kiểu liên kết như là các khái niệm để mô hình hoá dữ liệu của thế giới thực. Trong mô hình quan hệ, mỗi một dòng trong bảng biểu thị một sự kiện tương ứng với một thực thể hoặc một liên kết của thế giới thực. Tên bảng và tên các cột dùng để giúp giải thích ý nghĩa của các giá trị trong mỗi hàng. Mọi giá trị trong một cột đều cùng một kiểu dữ liệu

Theo thuật ngữ mô hình quan hệ hình thức, mỗi hàng được gọi là một bộ, mỗi đầu cột được gọi là một thuộc tính, và bảng được gọi là một quan hệ. Kiểu dữ liệu mô tả các kiểu của dữ liệu xuất hiện trong mỗi cột gọi là một miền

1.1- Miền, thuộc tính, bộ và quan hệ

Một ***miền D*** là một tập hợp các giá trị nguyên tử, điều đó có nghĩa là mỗi giá trị trong miền là không thể phân chia được trong phạm vi mô hình quan hệ. Để đặc tả một miền, người ta chỉ ra một tên, một kiểu dữ liệu và khuôn dạng dữ liệu. Một số ví dụ về định nghĩa miền:

. Họ tên: Tập hợp các dãy chữ cái có độ dài <= 30.

. Tuổi: Tập các số nguyên nằm trong khoảng từ 1 đến 80.

. Giới tính: Tập hợp gồm hai giá trị "Nam", "Nữ".

Ngoài ra, trong cơ sở dữ liệu người ta còn chỉ ra các thông tin phụ để thể hiện các giá trị của miền, chẳng hạn các đơn vị tính như tiền, trọng lượng,…

Một ***lược đồ quan hệ*** *R*, ký hiệu là $R(A_1,A_2,..., A_n)$, được tạo nên từ một tên quan hệ R một danh sách các thuộc tính $A_1,A_2,…, A_n$. Mỗi một thuộc tính A_i là tên vai trò của một miền D nào đó trong lược đồ quan hệ R. D được gọi là miền giá trị của A_i và được ký hiệu là $Dom(A_i)$. Một lược đồ quan hệ được sử dụng để mô tả một quan hệ, R được gọi là tên của quan hệ đó. Cấp của một quan hệ là số các thuộc tính của lược đồ quan hệ của nó. Ví dụ, ta có lược đồ cho quan hệ cấp 5: SINHVIÊN (Mãsố, Họtên, Ngàysinh, Giớitính, Địachỉ). Với lược đồ quan hệ này, SINHVIÊN là tên của quan hệ.

Một ***quan hệ*** (hoặc trạng thái quan hệ) r của lược đồ quan hệ $R(A_1,A_2,…, A_n)$ được ký hiệu là r(R), là tập hợp các n-bộ $r = \{t_1, t_2, ..., t_n\}$. Mỗi n-bộ t là một danh sách có thứ tự của n giá trị, $t = <v_1, v_2, …, v_n>$, trong đó mỗi v_i ,$1<= i <= n$, là một phần tử của $Dom(A_i)$ hoặc là một giá trị không xác định (null value). Giá trị thứ i của bộ t, tương ứng với thuộc tính A_i được ký hiệu là $t[A_i]$. Hình III-1 chỉ ra một ví dụ của quan hệ SINHVIÊN tương ứng với lược đồ quan hệ SINHVIÊN ở trên. Mỗi bộ trong quan hệ biểu diễn một thực thể sinh viên cụ thể. Quan hệ được biểu diễn như một bảng, trong đó mỗi bộ được hiển thị như một hàng và mỗi thuộc tính tương ứng với một đầu cột chỉ ra vai trò của các giá trị trong cột đó. Các giá trị không xác định biểu thị các thuộc tính mà giá trị của nó không biết được hoặc không tồn tại đối với từng bộ SINHVIÊN cụ thể.

SINHVIÊN	Họtên	Mã số	Ngàysinh	Giớitính	Địachỉ
	Lê Vân	4515202	12/09/84	Nữ	Hà nội
	Hoàng Tùng	4516802	21/03/84	Nam	Bắc ninh
	Trương Định	4620503	15/05/85	Nam	Hà nam
	Phạm An	4612203	16/04/85	Nam	Nam định
	Đỗ Cung	4521402	20/01/84	Nam	Nghệ an

Hình 0-1. Quan hệ SINHVIÊN

Định nghĩa quan hệ ở trên có thể phát biểu lại như sau: Một quan hệ r(R) là một quan hệ toán học cấp n trên các miền giá trị dom(A_1), dom(A_2), …, dom(A_n), đó là tập con của tích Đề các của các miền giá trị xác định R:

$$r(R) \subseteq (dom(A_1) \times dom(A_2) \times \ldots dom(A_n))$$

Tích Đềcác chỉ ra mọi tổ hợp có thể có của các giá trị từ các miền đã cho. Như vậy, nếu ta ký hiệu lực lượng của một miền D là $|D|$ và giả thiết rằng mọi miền đều hữu hạn thì tổng số các bộ trong tích Đề cac là:

$$|dom(A1)| * |dom(A2)| * \ldots * |dom(An)|$$

Ngoài tất cả các tổ hợp có thể có này, một trạng thái quan hệ ở một thời điểm cho trước - *gọi là trạng thái quan hệ hiện tại* - chỉ phản ánh các bộ giá trị biểu diễn một trạng thái cụ thể của thế giới thực. Nói chung, do trạng thái của thế giới thực thay đổi, quan hệ cũng bị thay đổi thành trạng thái quan hệ khác. Tuy nhiên, lược đồ R là ổn định, không thay đổi, trừ phi phải thêm vào một số thuộc tính để biểu diễn một thông tin mới chưa được lưu trữ trong quan hệ.

Có thể xảy ra trường hợp nhiều thuộc tính có cùng một miền giá trị. Các thuộc tính chỉ ra các vai trò khác nhau đối với miền. Ví dụ, hai thuộc tính ĐịachỉNV và ĐịachỉĐV có cùng miền giá trị nhưng thuộc tính thứ nhất tham chiếu đến địa chỉ của nhân viên còn địa chỉ thứ hai tham chiếu đến địa chỉ của đơn vị.

1.2- Các đặc trưng của các quan hệ

1.2.1- <u>Thứ tự của các bộ trong một quan hệ</u>

Một quan hệ được định nghĩa như một tập hợp các bộ. Các phần tử trong một tập hợp không có thứ tự, vì vậy các bộ trong một quan hệ không có một thứ tự cụ thể. Tuy nhiên, trong một tệp, các bản ghi được lưu trữ một cách vật lý trên đĩa vì vậy luôn có một thứ tự giữa các bản ghi. Thứ tự này chỉ rõ bản ghi thứ nhất, bản ghi thứ hai, …, bản ghi thứ n. Một cách tương tự, khi ta biểu diễn một quan hệ như là một bảng, các hàng được hiển thị theo một thứ tự nhất định.

Thứ tự các bộ không phải là một phần của định nghĩa quan hệ bởi vì một quan hệ cố gắng biểu diễn các sự vật ở mức trừu tượng hoặc lôgic. Có thể có nhiều thứ tự lôgic trên một quan hệ. Ví dụ, các bộ giá trị trong quan hệ SINHVIÊN ở hình III-1 có thể sắp xếp theo nhiều cách khác nhau: theo thứ tự logic của Họtên, theo thứ tự logic của Mãsố,… Định nghĩa quan hệ không chỉ ra thứ tự lôgic nào cả, vì

vậy không có thứ tự lôgic nào hơn thứ tự lôgic khác. Các quan hệ chứa cùng một số hàng như nhau nhưng các hàng được sắp xếp khác nhau được xem như đồng nhất với nhau. Khi một quan hệ được cài đặt như một tệp, một thứ tự vật lý có thể được chỉ ra trên các bản ghi của tệp.

1.2.2- Thứ tự của các giá trị bên trong một bộ

Theo định nghĩa quan hệ ở trên, một n-bộ là một danh sách có thứ tự của n giá trị. Như vậy thứ tự của các giá trị trong một bộ là quan trọng, từ đó suy ra thứ tự của các thuộc tính trong một lược đồ quan hệ cũng quan trọng. Tuy nhiên, ở mức lôgic, thứ tự của các thuộc tính và các giá trị của nó là không thực sự quan trọng khi giữ được sự tương ứng giữa các thuộc tính và các giá trị.

Có thể đưa ra một định nghĩa khác về quan hệ, định nghĩa này sẽ làm cho thứ tự của các giá trị trong một bộ là không cần thiết. Theo định nghĩa này, một lược đồ quan hệ R = {A1, A2,…,An} là một tập hợp các thuộc tính và một quan hệ r(R) là một tập hợp hữu hạn các ánh xạ r = {t_1, t_2, …., t_m}, trong đó mỗi t_i là một ánh xạ từ R vào D, trong đó D = dom(A_1)∪dom(A_2) ∪…∪dom(A_n). Trong định nghĩa này, t[A_i] phải ở trong dom(A_i) với 1<= i <= n với mỗi ánh xạ t_i trong r. Mỗi ánh xạ t_i được gọi là một bộ.

Theo định nghĩa này, một bộ có thể xem như một tập hợp các cặp (<thuộc tính>, <giá trị>), trong đó mỗi cặp cho một giá trị của ánh xạ từ một thuộc tính A_i đến một giá trị v_i của dom(A_i) . Vì tên thuộc tính xuất hiện cùng với giá trị của nó nên thứ tự của các thuộc tính là không quan trọng. Điều này làm nên ý nghĩa ở mức trừu tượng hoặc lôgic vì chẳng có lý do gì để thích có một giá trị thuộc tính xuất hiện trước một giá trị thuộc tính khác trong một bộ.

Khi một quan hệ được cài đặt như một tệp, các thuộc tính được sắp xếp một cách vật lý như là các trường trong một bản ghi. Trong trường hợp đó chúng ta sẽ sử dụng định nghĩa thứ nhất của quan hệ, trong đó các giá trị của các thuộc tính trong một bộ là có thứ tự vì nó làm đơn giản rất nhiều khái niệm. Tuy nhiên, định nghĩa thứ hai là tổng quát hơn.

1.2.3- Các giá trị trong một bộ

Mỗi giá trị trong một bộ là một giá trị nguyên tử, điều đó có nghĩa là nó không phân chia được thành các thành phần trong phạm vi của mô hình quan hệ. Như vậy, trong mô hình quan hệ không cho phép có các thuộc tính phức hợp và các thuộc

tính đa trị. Các thuộc tính đa trị phải được biểu diễn bằng các quan hệ còn các thuộc tính phức hợp chỉ được biểu diễn bằng các thuộc tính thành phần đơn của nó.

Các giá trị của một vài thuộc tính trong một bộ cụ thể có thể không biết được hoặc không thích ứng cho nó. Trường hợp đó, người ta sử dụng một giá trị đặc biệt gọi là giá trị null.Ví dụ, giả sử quan hệ SINHVIÊN có thuộc tính Sốđiệnthoạiởnhà. Trong một tập thể sinh viên, có người có điện thoại ở nhà, có người không có và cũng có người có nhưng không biết chắc. Với những trường hợp không có hoặc không biết chắc, thuộc tính Sốđiệnthoạiởnhà có giá trị null.

1.2.4- Thể hiện của một quan hệ

Một lược đồ quan hệ có thể được thể hiện như là một tuyên bố hoặc một khẳng định. Ví dụ lược đồ quan hệ SINHVIÊN ở trên khẳng định rằng, nói chung, một thực thể sinh viên có một mã số, họ tên, ngày sinh, giới tính, địa chỉ. Mỗi bộ trong quan hệ được thể hiện như là một sự kiện hoặc như một thể hiện cụ thể của một khẳng định. Ngoài các quan hệ biểu diễn các sự kiện về các thực thể, một số quan hệ có thể biểu diễn các sự kiện về mối liên kết. Ví dụ, lược đồ quan hệ NHÂNVIÊN_DỰÁN(MãsốNV, MãsốDA, Sốgiờ) khẳng định các nhân viên làm việc với các dự án. Mỗi bộ trong quan hệ này liên kết một nhân viên với một dự án mà anh ta làm việc cho nó.

Như vậy, mô hình quan hệ biểu diễn các sự kiện về thực thể và các sự kiện về liên kết dưới dạng duy nhất là các quan hệ.

2. Các ràng buộc quan hệ, lược đồ cơ sở dữ liệu quan hệ

Trong phần này chúng ta thảo luận về các hạn chế trên các dữ liệu trong một lược đồ cơ sở dữ liệu quan hệ. Các hạn chế đó được gọi là các ràng buộc. Có các loại ràng buộc: ràng buộc miền, ràng buộc khoá, ràng buộc toàn vẹn thực thể và ràng buộc toàn vẹn tham chiếu.

2.1- Các ràng buộc miền

Các ràng buộc miền chỉ ra rằng giá trị của mỗi thuộc tính A phải là một giá trị nguyên tử thuộc miền giá trị dom(A). Các kiểu dữ liệu liên kết với các miền bao gồm: các kiểu dữ liệu số chuẩn cho các số nguyên (short integer, integer, long integer), các số thực (float, double precision float). Ngoài ra còn các kiểu dữ liệu ký tự (dãy ký tự với độ dài cố định, dãy ký tự với độ dài thay đổi), ngày, thời gian và

tiền tệ. Các loại miền khác có thể là các miền con của một kiểu dữ liệu hoặc một kiểu dữ liệu đếm được trong đó mọi giá trị có thể được liệt kê rõ ràng

2.2- Ràng buộc khoá và ràng buộc trên các giá trị không xác định (null)

Một quan hệ được định nghĩa như một tập hợp các bộ. Theo định nghĩa, các phần tử của một tập hợp là khác nhau, vì vậy, mọi bộ trong quan hệ phải khác nhau. Điều đó có nghĩa là không có hai bộ có cùng một tổ hợp giá trị cho tất cả các thuộc tính của chúng. Thông thường, có tồn tại các tập con của các thuộc tính của một lược đồ quan hệ có tính chất là không có hai bộ nào ở trong mọi trạng thái quan hệ r của R có cùng một tổ hợp giá trị cho các thuộc tính của nó. Giả sử chúng ta ký hiệu một tập con như vậy là SK, khi đó với hai bộ khác nhau bất kỳ t_1 và t_2 trong một trạng thái quan hệ r của R chúng ta có ràng buộc là $t_1[SK] \neq t_2[SK]$.

Tập hợp thuộc tính SK như vậy được gọi là một *siêu khoá* của lược đồ quan hệ R. Một siêu khoá SK xác định rõ một ràng buộc về tính duy nhất, phát biểu rằng không có hai bộ khác nhau trong một trạng thái r của R có cùng một giá trị cho SK. Mỗi quan hệ có ít nhất là một siêu khoá mặc định, đó là tập hợp tất cả các thuộc tính của nó. Một *khoá K* của một lược đồ quan hệ R là một siêu khoá của R với tính chất là nếu bỏ đi bất kỳ thuộc tính A nào ra khỏi K thì sẽ còn lại một tập K không phải là siêu khoá của R. Như vậy, một khoá là một siêu khoá tối thiểu, nghĩa là đó là một siêu khoá mà ta không thể vứt bỏ thuộc tính nào ra khỏi nó mà vẫn giữ được ràng buộc về tính duy nhất.

Ví dụ, xét quan hệ SINHVIÊN với các thuộc tính Mãsố, Họtên, Ngàysinh, Giớitính, Địachỉ. Thuộc tính {Mãsố} là một khoá của SINHVIÊN bởi vì không có hai bộ sinh viên có cùng một giá trị cho Mãsố. Mọi tập hợp thuộc tính có chứa Mãsố, vídụ {Mãsố, Họtên, Ngàysinh}, đều là một siêu khoá. Tuy nhiên, siêu khoá {Mãsố, Họtên, Ngàysinh} không phải là khoá bởi vì nếu bỏ đi thuộc tính Họtên hoặc Ngàysinh hoặc cả hai thì nó vẫn còn là một siêu khoá.

Giá trị của một thuộc tính khoá có thể được sử dụng để xác định một cách duy nhất mỗi bộ trong một quan hệ. Ví dụ, giá trị 4515202 của Mãsố xác định một cách duy nhất bộ giá trị tương ứng với sinh viên Lê Vân trong quan hệ SINHVIÊN. Chú ý rằng một tập hợp thuộc tính tạo nên một khoá là một tính chất của lược đồ quan hệ. Điều ràng buộc là tính chất đó phải thỏa mãn trên mọi trạng thái của lược đồ. Một khoá được xác định từ ý nghĩa của các thuộc tính và tính chất là bất biến, tính chất đó phải thỏa mãn khi chúng ta chèn thêm các bộ mới vào quan hệ. Ví dụ, ta

không thể và không được chỉ định thuôc tính Họtên của quan hệ SINHVIÊN là khoá bởi vì không có gì đảm bảo rằng không tồn tại hai sinh viên có cùng họ tên.

Nói chung, một lược đồ quan hệ có thể có nhiều hơn một khoá. Trong trường hợp đó, mỗi một khoá được gọi là một *khoá dự tuyển*. Thông thường ta phải chỉ định một trong các khoá dự tuyển làm *khoá chính* của quan hệ. Khoá chính là một khoá dự tuyển mà các giá trị của chúng được dùng để xác định các bộ trong quan hệ. Ta quy ước rằng, các thuộc tính tạo nên khoá chính của một lược đồ quan hệ được gạch dưới. Ví dụ:

SINHVIÊN(Mãsố, Họtên, Ngàysinh, Giớitính, Địachỉ).

Chú ý rằng khi một lược đồ quan hệ có nhiều khoá dự tuyển, việc lựa chọn một khoá dự tuyển để làm khoá chính là tuỳ ý, tuy nhiên tốt nhất là chọn khoá chính gồm một thuộc tính hoặc có số các thuộc tính ít nhất.

Một ràng buộc khác trên các thuộc tính chỉ rõ khi nào thì cho phép các giá trị null. Những thuộc tính luôn luôn phải có một giá trị xác định và hợp lệ thì bị ràng buộc là NOT NULL.

2.3- Cơ sở dữ liệu quan hệ và lược đồ cơ sở dữ liệu quan hệ

Ở trên, chúng ta đã nói đến các lược đồ quan hệ đơn lẻ và các quan hệ đơn lẻ. Một cơ sở dữ liệu quan hệ thường gồm nhiều quan hệ với các bộ giá trị trong các quan hệ được liên kết với nhau theo nhiều cách. Trong phần này chúng ta sẽ định nghĩa một cơ sở dữ liệu quan hệ và một lược đồ cơ sở dữ liệu quan hệ. Một lược đồ cơ sở dữ liệu quan hệ S là một tập hợp các lược đồ quan hệ

$S = \{R_1, R_2, \ldots, R_n\}$ và một tập các ràng buộc toàn vẹn.

Một trạng thái cơ sở dữ liệu quan hệ (hoặc một cơ sở dữ liệu quan hệ) DB của S là một tập hợp các trạng thái quan hệ:

$$DB = \{r_1, r_2, \ldots, r_n\}$$

sao cho mỗi r_i là một trạng thái của R_i và sao cho các trạng thái quan hệ r_i thoả mãn các ràng buộc toàn vẹn chỉ ra trong tập các ràng buộc toàn vẹn.

Ví dụ, Hình III-2 trình bày một lược đồ cơ sở dữ liệu CÔNGTY và hình III-3 trình bày một cơ sở dữ liệu công ty.

NHÂNVIÊN(Họđệm, Tên, MãsốNV, Ngàysinh, Địachỉ, Giớitính, Lương,
MãsốNGS, MãsốĐV)

ĐƠNVỊ(TênĐV, MãsốĐV, MãsốNQL, Ngàybắtđầu)

ĐƠNVỊ_ĐỊAĐIỂM(MãsốĐV, ĐịađiểmĐV)

DỰÁN(TênDA, MãsốDA, ĐịađiểmDA, Mã sốĐV)

NHÂNVIÊN_DỰÁN(MãsốNV, MãsốDA, Sốgiờ)

CON(MãsốNV, TênPT, Giớitính, Ngàysinh)

Hình 0-2. Lược đồ cơ sở dữ liệu "CÔNG TY"

NHÂNVIÊN	MãsốNV	Họđệm	Tên	Ngày sinh	Địachỉ	Giớitính	Lương	MãsốNGS	MãsốĐV
	NV001	Lê	Vân	12/02/79	Hà nội	Nam	3000	NV002	5
	NV002	Trần Đức	Nam	14/02/66	Hà nội	Nam	4000	NV061	5
	NV010	Hoàng	Thanh	05/08/79	Nghệ an	Nữ	2500	NV014	4
	NV014	Phạm	Bằng	26/06/52	Bắc ninh	Nam	4300	NV061	4
	NV016	Nguyễn	Sơn	14/08/73	Hànam	Nam	3800	NV002	5
	NV018	Vũ Hương	Giang	26/03/83	Nam định	Nữ	2500	NV002	5
	NV025	Trần Lê	Hoa	15/03/80	Phú thọ	Nữ	2500	NV014	4
	NV061	Hoàng	Giáp	02/05/47	Hà tĩnh	Nam	5500	Null	1

ĐƠNVỊ	MãsốĐV	TênĐV	Mã sốNQL	Ngàybắtdầu
	5	Nghiên cứu	NV002	15/09/2000
	4	Hànhchính	NV014	24/06/1997
	1	Lãnhđạo	NV061	25/01/1992

ĐƠNVỊ_ĐỊAĐIỂM	Mã sốĐV	ĐịađiểmĐV
	1	Hà nội
	4	Hà nội
	5	Nam đinh
	5	Hà nội
	5	Bắc ninh

DỰÁN	TênDA	Mã sốDA	ĐịađiểmDA	Mã sốĐV
	DA01	1	Hà nội	5
	DA02	2	Nam định	5
	DA03	3	Bắc Ninh	5
	DA04	10	Hà nội	4
	DA05	20	Hà nội	1
	DA06	30	Hà nội	4

NHÂNVIÊN_DỰÁN	Mã sốNV	Mã sốDA	Sốgiờ
	NV001	1	32
	NV001	2	7
	NV016	3	40
	NV018	1	20
	NV018	2	20
	NV002	2	10
	NV002	3	10
	NV002	10	10
	NV002	20	10

NV010	30	30
NV010	10	10
NV025	10	35
NV025	30	5
NV014	30	20
NV014	20	15
NVO61	20	null

CON	MãsốNV	Têncon	Giớitính	Ngày sinh
	NV002	Giang	Nữ	04/05/1997
	NV002	Bình	Nam	25/10/1994
	NV002	Hoa	Nữ	03/05/1969
	NV014	Lan	Nữ	29/02/1953
	NV001	Bình	Nam	04/01/1999
	NV001	Hòa	Nũ	04/01/1999
	NV001	Hương	Nữ	05/05/1981

Hình 0-3. Cơ sở dữ liệu "CÔNGTY"

Trong một lược đồ cơ sở dữ liệu quan hệ, các thuộc tính biểu diễn cùng một khái niệm thế giới thực có thể (hoặc không) có cùng tên như nhau trong các quan hệ khác nhau. Ngược lại, các thuộc tính biểu diễn các khái niệm khác nhau có thể có tên như nhau trong các quan hệ khác nhau. Ví dụ, trong cơ sở dữ liệu CÔNGTY ở trên, các thuộc tính MãsốNV, MãsôNGS, MãsốNQL có tên khác nhau nhưng đều biểu diễn một khái niệm đó là mã số nhân viên (bởi vì người giám sát hoặc người quản lý cũng là nhân viên). Trong lúc đó, thuộc tính Giớitính có mặt trong hai quan hệ NHÂNVIÊN và CON, tuy nhiên Giớitính trong quan hệ NHÂNVIÊN là biểu thị giới tính của nhân viên còn Giớitính trong CON là biểu thị giới tính của người con.

Trong một số phiên bản trước của mô hình quan hệ, người ta yêu cầu rằng các thuộc tính biểu diễn cùng một khái niệm của thế giới thực thì phải có tên như nhau

trong mọi quan hệ. Điều đó sẽ gây ra khó khăn khi cùng một khái niệm thế giới thực được sử dụng trong các vai trò khác nhau.

Một hệ quản trị cơ sở dữ liệu phải có ngôn ngữ định nghĩa dữ liệu (Data definition language DDL) để định nghĩa lược đồ cơ sở dữ liệu quan hệ. Các hệ quản trị cơ sở dữ liệu hiện nay hầu như sử dụng SQL cho mục đích này.

Các ràng buộc toàn vẹn được chỉ ra trên một lược đồ cơ sở dữ liệu và được tôn trọng làm thoả mãn trên mỗi trạng thái cơ sở dữ liệu của lược đồ này. Ngoài các ràng buộc miền và ràng buộc khoá còn có thêm các ràng buộc được xem như một phần của mô hình quan hệ, đó là ràng buộc toàn vẹn thực thể và ràng buộc toàn vẹn tham chiếu.

2.4- Toàn vẹn thực thể, toàn vẹn tham chiếu và khoá ngoài

Ràng buộc toàn vẹn thực thể được phát biểu là: khoá chính phải luôn luôn có giá trị xác định, nghĩa là không được phép có giá trị null. Sở dĩ có điều đó là do giá trị của khoá chính được sử dụng để xác định các bộ giá trị riêng biệt trong một quan hệ. Việc có giá trị null cho khoá chính kéo theo việc chúng ta không thể xác định được một số bộ giá trị. Ví dụ, nếu có hai hay nhiều hơn các bộ giá trị có giá trị null cho khoá chính thì chúng ta không có khả năng phân biệt chúng.

Các ràng buộc khoá và ràng buộc toàn vẹn thực thể được chỉ ra trên các quan hệ riêng rẽ. *Ràng buộc toàn vẹn tham chiếu* được chỉ ra giữa hai quan hệ để duy trì sự tương ứng giữa các bộ của hai quan hệ. Một cách không hình thức, ràng buộc toàn vẹn tham chiếu được phát biểu là: một bộ giá trị trong một quan hệ có liên kết đến một quan hệ khác phải liên kết đến một bộ giá trị tồn tại trong quan hệ đó.

Để định nghĩa toàn vẹn tham chiếu một cách hình thức hơn, trước tiên chúng ta đưa ra khái niệm khoá ngoài: Một tập hợp các thuộc tính FK trong một lược đồ quan hệ R_1 là một khoá ngoài của R_1 tham chiếu đến quan hệ R_2 nếu nó thoả mãn hai quy tắc sau:

qt1. Các thuộc tính trong FK có cùng miền giá trị như các thuộc tính của khoá chính PK của R_2. Các thuộc tính FK được gọi là tham chiếu đến (hoặc là liên hệ đến) quan hệ R_2.

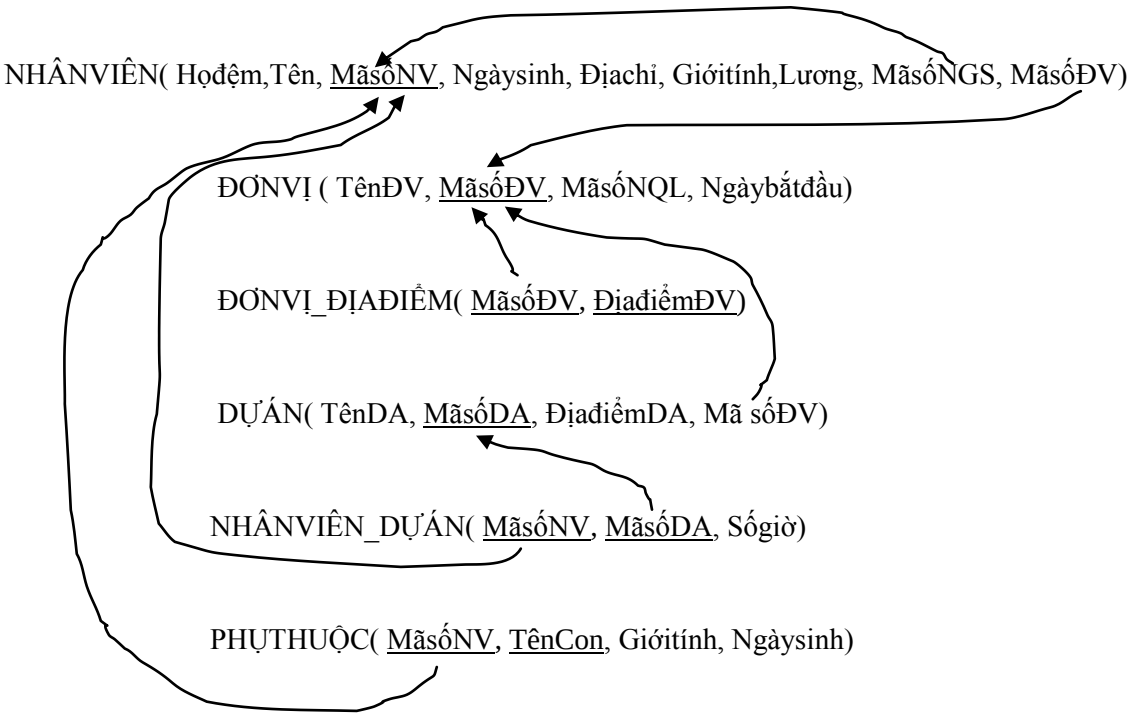
qt2. Một giá trị của FK trong một bộ t_1 của trạng thái hiện tại $r_1(R_1)$ hoặc có mặt như một giá trị của khoá chính của một bộ t_2 nào đấy trong trạng thái hiện tại $r_2(R_2)$, hoặc là null. Trong trường hợp này ta có $t_1[FK] = t_2[PK]$ và ta nói rằng bộ t_1

liên hệ (tham chiếu) đến bộ t_2. R_1 được gọi là quan hệ tham chiếu và R_2 được gọi là quan hệ bị tham chiếu.

Trong một cơ sở dữ liệu có nhiều quan hệ thường có nhiều ràng buộc toàn vẹn tham chiếu. Để chỉ ra các ràng buộc này, trước tiên ta phải có một hiểu biết rõ ràng về ý nghĩa hoặc vai trò của mỗi tập thuộc tính ở trong các lược đồ quan hệ khác nhau của cơ sở dữ liệu. Các ràng buộc toàn vẹn quy chiếu thường nảy sinh từ các mối liên kết giữa các thực thể được biểu diễn bằng các lược đồ quan hệ.

Chú ý rằng một khoá ngoài có thể tham chiếu đến quan hệ của chính nó. Trong trường hợp đó, khoá ngoài biểu thị một liên kết đệ quy.

Chúng ta có thể biểu diễn các ràng buộc tham chiếu bằng sơ đồ. Để làm điều đó ta vẽ một cạnh có hướng từ mỗi khoá ngoài đến quan hệ mà nó tham chiếu đến. Hình III-4 biểu diễn lược đồ ở hình 3.2 với các ràng buộc quy chiếu được biểu diễn theo cách này.



Hình 0-4. Lược đồ và sơ đồ tham chiếu

Ngoài các ràng buộc toàn vẹn ở trên, cơ sở dữ liệu còn phải thoả mãn một số ràng buộc khác, như ràng buộc trạng thái, ràng buộc chuyển tiếp… Các ràng buộc trạng thái xác định các ràng buộc mà một trạng thái vững chắc của cơ sở dữ liệu phải thoả mãn. Ví dụ về các ràng buộc đó là: "lương của một nhân viên không được

vượt quá lương của người giám sát nhân viên đó" hoặc "số giờ nhiều nhất mà một nhân viên có thể làm việc trong một tuần trên tất cả các dự án là 56 giờ". Các ràng buộc như vậy có thể được đặc tả và bắt tuân theo bằng cách sử dụng một ngôn ngữ đặc tả ràng buộc. Người ta có thể sử dụng các cơ cấu như là trigger hoặc assertion. Các ràng buộc chuyển tiếp có thể được định nghĩa để làm việc với những thay đổi trạng thái trong cơ sở dữ liệu. Ví dụ về ràng buộc này là: " lương của một nhân viên chỉ có thể tăng". Các ràng buộc như vậy thường được định nghĩa bằng cách sử dụng các quy tắc hoặc bằng các trigger.

CHƯƠNG 4 - ĐẠI SỐ QUAN HỆ

1. Các phép toán trên mô hình quan hệ

Trong phần này chúng ta thảo luận về các phép toán của mô hình quan hệ. Các phép toán của mô hình quan hệ có thể phân thành hai loại: các phép toán cập nhật và các phép toán đại số quan hệ. Các phép toán cập nhật được sử dụng để tạo ra một quan hệ đúng đắn. Các phép toán đại số quan hệ được sử dụng để đặc tả các phép lấy thông tin ra.

1.1- Các phép toán cập nhật

Các phép toán cập nhật gồm ba phép toán cơ bản là chèn, xoá và sửa đổi. Phép chèn được dùng để chèn một bộ giá trị hoặc nhiều bộ giá trị vào một quan hệ. Phép xoá dùng để loại bỏ các bộ giá trị và phép sửa đổi dùng để sửa đổi các giá trị của một số thuộc tính trong các bộ giá trị đã có. Mỗi khi các phép toán cập nhật được áp dụng, các ràng buộc trên lược đồ cơ sở dữ liệu có thể bị vi phạm. Trong phần này chúng ta sẽ nói đến khả năng vi phạm các ràng buộc của từng phép toán và các kiểu hành động có thể thực hiện khi một ràng buộc bị vi phạm.

1.1.1- Phép chèn (Insert)

Phép chèn cung cấp một danh sách các giá trị cho một bộ mới t được chèn vào trong một quan hệ R. Phép chèn có thể vi phạm các kiểu ràng buộc được mô tả ở trên. Các ràng buộc miền có thể bị vi phạm nếu một giá trị thuộc tính được cho không thuộc vào miền tương ứng. Các ràng buộc khoá có thể bị vi phạm nếu một giá trị khoá trong bộ mới t đã tồn tại trong một bộ khác ở trong quan hệ r(R). Sự toàn vẹn thực thể có thể bị vi phạm nếu khoá chính của bộ mới t là null. Sự toàn vẹn tham chiếu có thể bị vi phạm nếu một giá trị của một khoá ngoài trong t tham chiếu đến một bộ không tồn tại trong một quan hệ được tham chiếu. Ví dụ (với các bảng trong cơ sở dữ liệu CÔNGTY) :

- Chèn bộ giá trị <null, ‘Vũ’, ‘Hải’, ‘15/07/81’, ‘Hànội’, ‘Nam’, 3200, null, 4> vào quan hệ NHÂNVIÊN. Phép chèn này vi phạm ràng buộc toàn vẹn thực thể (giá trị null cho khoá chính). Phép chèn bị loại bỏ.
- Chèn bộ giá trị <‘NV002’,‘Trương’,‘Phi’,‘15/07/81’, ‘Hànội’, ‘Nam’, 3200, ‘NV067’, 4> vào quan hệ NHÂNVIÊN. Phép chèn này vi phạm ràng buộc khoá, bởi vì giá trị ‘NV002’ đã có ở trong bảng. Phép chèn bị loại bỏ.

- Chèn bộ giá trị <‘NV072’, ‘Vũ’, ‘Hải’, ‘15/07/81’, ‘Hànội’, ‘Nam’, 3200, ‘NV002’ , 7> vào quan hệ NHÂNVIÊN. Phép chèn này vi phạm ràng buộc toàn vẹn tham chiếu, trong quan hệ ĐƠNVỊ không có đơn vị có mã số = 7. Phép chèn bị loại bỏ.
- Chèn bộ giá trị <‘NV045’l, ‘Vũ’, ‘Hải’, ‘15/07/81’, ‘Hànội’, ‘Nam’, 3200, ‘NV002’, 4> vào quan hệ NHÂNVIÊN. Phép chèn thoả mãn tất cả các ràng buộc, vì thế nó được chấp nhận.

Nếu một phép chèn vi phạm một hoặc nhiều ràng buộc, tuỳ chọn mặc định là loại bỏ phép chèn. Trong trường hợp này, thường là các hệ quản trị cơ sở dữ liệu có thể thông báo cho người sử dụng nguyên nhân của việc loại bỏ phép chèn.

1.1.2- Phép xoá (Delete)

Phép xoá được sử dụng để xoá một hoặc nhiều bộ giá trị của một quan hệ. Phép xoá chỉ có thể vi phạm ràng buộc tham chiếu trong trường hợp bộ bị xoá được tham chiếu bởi một khoá ngoài từ các bộ khác trong cơ sở dữ liệu. Để chỉ rõ một phép xoá, cần phải đưa ra một điều kiện trên các thuộc tính của quan hệ để chọn các bộ sẽ bị xoá. Ví dụ:

- Xoá một bộ giá trị của quan hệ NHÂNVIÊN_DỰÁN có MãsốNV = ‘NV010’ và MãsốDA = 10. Phép xoá này được chấp nhận.
- Xoá bộ giá trị của NHÂNVIÊN có MãsốNV = ‘ NV010’. Phép xoá này không chấp nhận được bởi vì có các bộ trong NHÂNVIÊN_DỰÁN tham chiếu đến bộ này, như vậy là vi phạm ràng buộc toàn vẹn tham chiếu.
- Xoá bộ giá trị của NHÂNVIÊN có MãsốNV = ‘ NV002’. Phép xoá này cũng vi phạm ràng buộc toàn vẹn tham chiếu .

Ba tuỳ chọn được sẵn sàng được sử dụng nếu một phép xoá gây ra sự vi phạm. Tuỳ chọn thứ nhất là loại bỏ phép xoá. Tuỳ chọn thứ hai là cố gắng lan truyền phép xoá (cascade the deletion) bằng cách xoá đồng thời các bộ tham chiếu đến bộ bị xoá. Tuỳ chọn thứ ba là sửa đổi các giá trị của các thuộc tính tham chiếu gây ra sự vi phạm. Mỗi giá trị như vậy hoặc là làm cho bằng null hoặc được thay đổi thành bộ có hiệu lực tham chiếu khác. Chú ý rằng, nếu một thuộc tính tham chiếu gây ra sự vi phạm là một phần của khoá chính thì không thể làm cho thành null, bởi vì nếu

làm vậy thì sẽ vi phạm ràng buộc toàn vẹn thực thể. Có thể kết hợp cả ba tuỳ chọn ở trên.

1.1.3- Phép sửa đổi (Update)

Phép toán sửa đổi được dùng để thay đổi các giá trị của một hoặc nhiều thuộc tính trong một (hoặc nhiều) bộ của một quan hệ R nào đấy. Để lựa chọn các bộ cần được thay đổi, người sử dụng phải chỉ ra một điều kiện trên các thuộc tính. Ví dụ:

- Sửa đổi Lương của bộ NHÂNVIÊN có MãsốNV = 'NV018' thành 2800. Phép sửa đổi này được chấp nhận.
- Sửa đổi MãsốĐV của bộ NHÂNVIÊN có MãsốNV = 'NV018' thành 7. Phép sửa đổi này vi phạm ràng buộc toàn vẹn tham chiếu.
- Sửa đổi MãsốNV của bộ NHÂNVIÊN có MãsốNV = 'NV018' thành 'NV014'. Phép sửa đổi này vi phạm ràng buộc toàn vẹn thực thể và toàn vẹn tham chiếu.

Việc sửa đổi một thuộc tính không phải là một khoá chính hoặcc một khoá ngoài thường không gây ra các vi phạm ràng buộc, hệ quản trị cơ sở dữ liệu chỉ cần kiểm tra để khẳng định rằng giá trị mới là thuộc miền và kiểu giá trị đúng đắn. Việc sửa đổi giá trị một khoá chính tương tự như việc xoá một bộ và chèn bộ khác vào chỗ của nó. Như vậy chúng ta trở về trường hợp đã thảo luận với phép chèn và phép xoá. Nếu một thuộc tính khoá ngoài bị sửa đổi thì hệ quản trị cơ sở dữ liệu phải đảm bảo rằng giá trị mới tham chiếu đến một bộ có tồn tại trong quan hệ được tham chiếu (hoặc là null).

1.2- Các phép toán đại số quan hệ

Ngoài việc định nghĩa cấu trúc cơ sở dữ liệu và các ràng buộc, một mô hình dữ liệu phải chứa một tập hợp phép toán để thao tác dữ liệu. Tập hợp cơ sở các phép toán mô hình quan hệ tạo nên đại số quan hệ. Các phép toán này giúp cho người sử dụng xác định rõ các yêu cầu lấy tin cơ bản. Kết quả của một phép lấy tin là một quan hệ mới, có thể được tạo ra từ một hoặc nhiều quan hệ. Các quan hệ đó có thể được thao tác tiếp theo bằng cách sử dụng các phép toán của cùng đại số. Một dãy các phép toán quan hệ tạo nên một biểu thức đại số quan hệ mà kết quả của nó cũng là một quan hệ.

Các phép toán đại số quan hệ được chia thành hai nhóm. Một nhóm bao gồm các phép toán tập hợp lấy từ lý thuyết tập hợp toán học. Các phép toán đó là phép hợp, phép giao, phép trừ tập hợp và phép tích Đề các. Nhóm kia bao gồm các phép toán được xây dựng đặc biệt cho các cơ sở dữ liệu quan hệ. Các phép toán đó là phép chọn, phép chiếu, phép nối và một số các phép toán khác.

1.2.1- Phép chọn (SELECT)

Phép chọn đuợc sử dụng để chọn một tập hợp các bộ thoả mãn điều kiện chọn từ một quan hệ. Ta có thể xem phép chọn như một bộ lọc, nó chỉ giữ lại các bộ thoả mãn điều kiện đặt ra.

Phép chọn được ký hiệu là

$$\sigma_{<\text{điều kiện chọn}>}(\text{R})$$

trong đó ký hiệu σ được dùng để ký hiêu phép chọn, còn điều kiện chọn là một biểu thức lôgic được chỉ ra trên các thuộc tính của R. Chú ý rằng R nói chung là một biểu thức đại số quan hệ. Kết quả của một biểu thức đại số quan hệ là một quan hệ. Biểu thức đơn giản nhất chính là tên của một quan hệ của một cơ sở dữ liệu. Quan hệ kết quả của phép chọn có cùng thuộc tính như R. Ví dụ, để chọn các bộ NHÂNVIÊN thuộc về đơn vị có mã số là 4 hoặc các bộ NHÂNVIÊN có lương lớn hơn 3000 ta có thể viết một cách riêng rẽ như sau:

$$\sigma_{<\text{Mãsố} = 4>}(\text{NHÂNVIÊN})$$

$$\sigma_{<\text{Lương} > 3000>}(\text{NHÂNVIÊN})$$

Biểu thức logic chỉ ra trong <điều kiện chọn> được tạo nên từ một số hạng mục có dạng :

<tên thuộc tính> <phép so sánh> <giá trị hằng>

hoặc <tên thuộc tính> <phép so sánh> <tên thuộc tính>

trong đó <tên thuộc tính> là tên của một thuộc tính trong R, <phép so sánh> là một trong các phép toán so sánh {<, <=, =, >=, >, ≠} còn <giá trị hằng> là một giá trị hằng từ miền giá trị của thuộc tính. Các hạng mục có thể được nối với nhau bằng các phép toán lô gic AND, OR, NOT để tạo ra một điều kiện chọn chung. Ví dụ, để chọn ra các nhân viên làm việc ở đơn vị có mã số là 4 và có lương lớn hơn 3000 hoặc các nhân viên làm việc ở đơn vị có mã số là 5 và có lương lớn hơn 4000 ta có thể viết phép chọn như sau:

$$\sigma_{<\text{MãsốĐV} = 4>\text{AND}<\text{lương}>3000>\text{OR}<\text{MãsốĐV} = 5>\text{AND}<\text{lương}>3500>}(\text{NHÂNVIÊN})$$

Kết quả chỉ ra ở hình III-5.

MãsốNV	Họđệm	Tên	Ngàysinh	Địachỉ	Giớitính	Lương	MãsốNGS	MãsốĐV
NV002	Trần Đức	Nam	14/02/66	Hà nội	Nam	4000	NV061	5
NV014	Phạm	Bằng	26/06/52	Bắc ninh	Nam	4300	NV061	4
NV016	Nguyễn	Sơn	14/08/73	Hà nam	Nam	3800	NV002	5

Hình 0-1. Kết quả phép chọn

Chú ý rằng các phép toán so sánh trong tập hợp {<, <=, =, >=, >, ≠} áp dụng cho các thuộc tính có miền giá trị là các giá trị có thứ tự như là miền giá trị số. Miền giá trị các dãy ký tự được xem như có thứ tự dựa trên việc so sánh các dãy ký tự. Nếu miền giá trị của một thuộc tính là một tập hợp các giá trị không có thứ tự thì chỉ có các phép so sánh trong tập hợp { =, ≠ } là có thể áp dụng được. Ngoài ra, có thể còn các phép so sánh bổ sung, chẳng hạn như " là một dãy con của…" hoặc "trong khoảng từ… đến…".

Kết quả một phép chọn được xác định như sau: <Điều kiện chọn> được áp dụng cho mỗi bộ t trong R một cách độc lập. Điều đó được thực hiện bằng cách thay thế mỗi thuộc tính A_i trong điều kiện chon bằng giá trị $t[A_i]$ của nó trong bộ. Nếu điều kiện chọn cho giá trị đúng thì bộ t sẽ được chọn. Tất cả các bộ được chọn xuất hiện trong kết quả của phép chọn. Các phép toán logic AND, OR, NOT được thực hiện theo quy tắc bình thường của chúng.

Phép chọn là phép toán một ngôi, nghĩa là nó được áp dụng cho một quan hệ. Hơn nữa, phép chọn được áp dụng cho từng bộ một cách độc lập, vì vậy, các điều kiện chọn không thể liên quan đến nhiều bộ. Quan hệ kết quả của phép chọn có cấp giống như cấp của R. Số các bộ trong quan hệ kết quả luôn luôn nhỏ hơn hoặc bằng số các bộ trong R.

Phép chọn là một phép toán có tính chất giao hoán, nghĩa là

$$\sigma_{<\text{Điều kiện 1}>}(\sigma_{<\text{Điều kiện 2}>}(R)) = \sigma_{<\text{Điều kiện 2}>}(\sigma_{<\text{Điều kiện 1}>}(R))$$

Hơn nữa ta có thể kết hợp một loạt các phép chọn thành một phép chọn đơn giản bằng cách sử dụng phép toán AND. Ví dụ:

$$\sigma_{<\text{Điều kiện 1}>}(\sigma_{<\text{Điều kiện 2}>}(R)) = \sigma_{<\text{Điều kiện 2}>\text{AND}<\text{Điều kiện 1}>}(R)$$

1.2.2- Phép chiếu (PROJECT)

Nếu ta coi một quan hệ như một bảng thì phép chọn chọn một số hàng của bảng thoả mãn điều kiện chọn và bỏ qua các hàng không thoả mãn điều kiện chọn. Phép chiếu là phép toán chọn một số cột của bảng. Nếu chúng ta chỉ quan tâm đến một số thuộc tính của quan hệ, chúng ta dùng phép chiếu để chiếu lên các thuộc tính đó. Phép chiếu được ký hiệu là:

$$\pi_{<\text{danh sách các thuộc tính}>}(R)$$

trong đó π là ký hiệu dùng để biểu diễn phép chiếu và <danh sách các thuộc tính> là một danh sách con các thuộc tính của quan hệ R. Nói chung R là một biểu thức đại số quan hệ. Trường hợp đơn giản nhất nó là tên của một quan hệ của cơ sở dữ liệu. Kết quả của phép chiếu là một quan hệ chỉ có các thuộc tính nằm trong <danh sách các thuộc tính> và có cùng thứ tự như thứ tự của chúng có trong danh sách. Như vậy, cấp của quan hệ kết quả là số các thuộc tính có trong <danh sách các thuộc tính>.

Nếu <danh sách các thuộc tính> chỉ bao gồm các thuộc tính không phải thuộc tính khoá của R thì quan hệ kết quả có thể có những bộ trùng nhau. Phép chiếu loại bỏ mọi bộ trùng lặp, và như vậy, kết quả của phép chiếu là một tập hợp các bộ và là một quan hệ đúng đắn.

Ví dụ, phép chiếu:

$$\pi_{<\text{MãsốNV, Họđệm,Tên, Lương}>}(\text{NHÂNVIÊN})$$

cho kết quả là một quan hệ có các thuộc tính MãsốNV, Họđệm, Tên, Lương (hình III-6).

MãsốNV	Họđệm	Tên	Địachỉ	Lương
NV001	Lê	Vân	Hà nội	3000
NV002	Trần Đức	Nam	Hà nội	4000
NV010	Hoàng	Thanh	Nghệ an	2500
NV014	Phạm	Bằng	Bắc ninh	4300
NV016	Nguyễn	Sơn	Hànam	3800

NV018	Vũ Hương	Giang	Nam định	2500
NV025	Trần Lê	Hoa	Phúthọ	2500
NV061	Hoàng	Giáp	Hà tĩnh	5500

Hình 0-2. Kết quả phép chiếu

Số các bộ trong quan hệ kết quả từ một phép chiếu luôn luôn nhỏ hơn hoặc bằng số các bộ trong R. Nếu danh sách chiếu là một siêu khoá của R (nghĩa là nó chứa một khoá nào đó của R) thì quan hệ kết quả có cùng một số bộ như R. Ngoài ra, nếu <danh sách 2> chứa tất cả các thuộc tính có trong <danh sách 1> thì

$$\pi_{<\text{danh sách1}>}(\pi_{<\text{danh sách2}>}(R)) = \pi_{<\text{danh sách 1}>}(R)$$

Phép chiếu không có tính giao hoán.

1.2.3- Phép đặt lại tên (RENAME)

Chúng ta có thể áp dụng nhiều phép toán quan hệ liên tiếp nhau. Trong trường hợp đó hoặc chúng ta có thể viết các phép toán như là một biểu thức đại số quan hệ đơn bằng cách xếp lồng các phép toán lại với nhau, hoặc chúng ta có thể áp dụng mỗi phép toán tại một thời điểm và tạo ra các quan hệ kết quả trung gian. Trong trường hợp tạo các quan hệ trung gian, ta phải đặt tên cho quan hệ đó. Ví dụ: Để đưa ra Họtên và Lương của các Nhânviên làm việc ở đơn vị có Mãsố là 4 chúng ta phải áp dụng một phép chọn và một phép chiếu. Chúng ta có thể viết một biểu thức đại số quan hệ đơn như sau :

$$\pi_{<\text{Họtên, Lương}>}(\sigma_{<\text{Mãsố} = 4>}(\text{NHÂNVIÊN}))$$

Một cách khác, chúng ta có thể tạo ra kết quả trung gian và viết biểu thức trên thành dãy các phép toán như sau:

$$\text{KQTG} \leftarrow \sigma_{<\text{Mãsố} = 4>}(\text{NHÂNVIÊN})$$

$$\text{Ketqua} \leftarrow \pi_{<\text{Họtên, Lương}>}(\text{KQTG})$$

Thông thường việc phân tích một dãy phức tạp các phép toán bằng cách chỉ ra các quan hệ kết quả trung gian là dễ hơn việc viết một biểu thức đại số quan hệ đơn. Chúng ta có thể dùng kỹ thuật này để đặt lại tên (rename) cho các thuộc tính trong các quan hệ trung gian và kết quả. Để đặt lại tên cho các thuộc tính của một quan hệ, chúng ta liệt kê các tên mới của các thuộc tính trong cặp dấu ngoặc. Ví dụ:

$$R(\text{Họvà tên, Lương}) \leftarrow \pi_{< \text{Họtên , Lương} >}(\text{KQTG})$$

Cho kết quả là quan hệ R, trong đó thuộc tính Họtên được đặt lại tên thành Họvàtên.

Nếu không có việc đặt lại tên thì tên của các thuộc tính trong quan hệ kết quả của một phép chọn là giống như các tên trong quan hệ ban đầu và có cùng một thứ tự như thứ tự của các thuộc tính đó. Đối với phép chiếu, nếu không có việc đặt lại tên thì quan hệ kết quả có các tên thuộc tính giống như các tên trong danh sách chiếu và có cùng thứ tự như chúng xuất hiện trong danh sách.

Chúng ta có thể định nghĩa một phép toán đặt lại tên , nó có thể đặt lại tên cho một tên quan hệ hoặc các tên thuộc tính hoặc cả hai. Phép đặt lại tên được ký hiệu là:

$$\rho_{S(B1,B2,\ldots Bn)}(R) \quad \text{hoặc} \quad \rho_S(R) \quad \text{hoặc} \quad \rho_{(B1,B2,\ldots Bn)}(R)$$

trong đó ký hiệu ρ được dùng để ký hiệu phép toán đặt lại tên, S là tên quan hệ mới, $B_1,B_2,\ldots B_n$ là các tên thuộc tính mới. Biểu thức thứ nhất đặt lại tên quan hệ và các thuộc tính của nó. Nếu các thuộc tính của R là $A_1,A_2, \ldots A_n$ thì sau khi đặt lại tên, quan hệ có tên mới là S còn các thuộc tính có tên mới là $B_1, B_2, \ldots, B_n$. Biểu thức thứ hai chỉ đặt lại tên quan hệ, nghĩa là sau phép đặt lại tên, quan hệ có tên mới là S, còn các thuộc tính vẫn mang tên cũ. Biểu thức thứ ba chỉ đặt lại tên các thuộc tính, nếu các thuộc tính của R là $A_1,A_2, \ldots A_n$ thì sau khi đặt lại tên chúng có tên là $B_1, B_2, \ldots B_n$.

1.2.4- <u>Các phép toán lý thuyết tập hợp</u>

Nhóm tiếp theo của các phép toán đại số quan hệ là các phép toán toán học thông thường trên các tập hợp. Đó là các phép toán hợp, giao và trừ tập hợp. Các phép toán này là các phép toán hai ngôi, nghĩa là mỗi phép toán được áp dụng cho hai tập hợp. Khi áp dụng các phép toán này cho cơ sở dữ liệu quan hệ, hai quan hệ tham gia vào một trong các phép toán trên phải có kiểu của các bộ như nhau, hay nói cách khác, chúng phải có cùng một cấu trúc. Điều kiện này được gọi là *tương thích đồng nhất*. Hai quan hệ $R(A_1,A_2,\ldots, A_n)$ và $S(B_1, B_2, \ldots,B_n)$ được gọi là tương thích đồng nhất nếu chúng có cùng cấp n và dom(Ai) = dom(Bi) với 1<= i <= n. Điều đó có nghĩa là hai quan hệ có cùng số các thuộc tính và mỗi cặp thuộc tính tương ứng có cùng miền giá trị.

Các phép toán được định nghĩa như sau:

. Phép hợp: Hợp của hai quan hệ R và S, được ký hiệu là R ∪ S, cho kết quả là một quan hệ chứa tất cả các bộ có trong R hoặc ở trong S hoặc ở trong cả hai. Các bộ trùng lặp bị loại bỏ.

. Phép giao: Giao của hai quan hệ R và S , được ký hiệu là R ∩ S , cho kết quả là một quan hệ chứa tất các các bộ có trong cả hai quan hệ R và S.

. Phép trừ quan hệ: Phép trừ quan hệ R và S , được ký hiệu là R - S, cho kết quả là một quan hệ chứa tất cả các bộ có trong R nhưng không có trong S.

Ví dụ, xét hai quan hệ:

R	Họtên	Tuổi	Giớitính
	AA	20	Nam
	BB	18	Nữ
	CC	21	Nam
	DD	25	Nữ

S	Họtên	Tuổi	Giớitính
	BB	18	Nữ
	EE	20	Nam
	DD	25	Nữ
	FF	21	Nam

R∪S	Họtên	Tuổi	Giớitính
	AA	20	Nam
	BB	18	Nữ
	CC	21	Nam
	DD	25	Nữ
	EE	20	Nam
	FF	21	Nam

R∩S	Họtên	Tuổi	Giớitính
	BB	18	Nữ
	DD	25	Nữ

R - S	Họtên	Tuổi	Giớitính
	AA	20	Nam
	CC	21	Nam

Hình 0-3. Kết quả của các phép toán tập hợp

Chú ý rằng các phép toán hợp và giao là các phép toán giao hoán, nghĩa là:

$$R \cup S = S \cup R \quad \text{và} \quad R \cap S = S \cap R$$

Các phép toán trên cũng có tính chất kết hợp, nghĩa là

$$R \cup (S \cup T) = (R \cup S) \cup T \quad \text{và} \quad R \cap (S \cap T) = (R \cap S) \cap T$$

Phép toán trừ tập hợp không có tính chất giao hoán.

$$R - S \neq S - R$$

Ngoài các phép toán trên, còn có một phép toán gọi là tích Đề các. Tích Đề các còn gọi là tích hỗn hợp (cross product) hoặc là nối hỗn hợp (cross join), được ký hiệu là ×. Đó cùng là một phép toán hai ngôi nhưng những quan hệ mà nó áp dụng trên đó không phải là tương thích đồng nhất. Phép toán này được sử dụng để nối các bộ của hai quan hệ vào một kiểu kết hợp. Kết quả của

$$R(A_1, A_2, .. , A_n) \times S(B_1, B_2, \ldots, B_m)$$

là một quan hệ Q với n+m thuộc tính $Q(A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m)$. Quan hệ kết quả Q có các bộ được tạo thành do sự kết hợp một bộ của R và một bộ của S. Ví dụ, xét hai quan hệ R và S như sau:

R	A1	A2	A3
	aa	bb	cc
	ab	ba	ac

S	B1	B2	B3
	dd	da	db
	cd	cb	ac

R × S	A1	A2	A3	B1	B2	B3
	aa	bb	cc	dd	da	db
	aa	bb	cc	cd	cb	ac
	ab	ba	ac	dd	da	db
	ab	ba	ac	cd	cb	ac

Hình 0-4. Tích Đề các của hai quan hệ R và S.

Như vậy, nếu R có n_R bộ và S có n_S bộ thì R× S có n_R*n_S bộ. Phép toán này nếu áp dụng một mình thì không có ý nghĩa mấy. Nó chỉ có lợi khi tiếp theo bằng một phép chọn các giá trị tương thích của các thuộc tính xuất phát từ các quan hệ thành phần. Tích Đềcác kết hợp với một phép chọn cho ta một phép nối.

1.2.5- Phép nối (JOIN)

Phép nối được ký hiệu là $\bowtie$ và được dùng để kết hợp các bộ có liên hệ với nhau từ hai quan hệ thành một bộ. Phép toán này rất quan trọng đối với cơ sở dữ liệu quan hệ có nhiều bảng bởi vì nó cho phép ta xử lý các mối liên kết giữa các quan hệ. Dạng tổng quát của phép nối trên hai quan hệ $R(A_1, A_2,\ldots,A_n)$ và $S(B_1,B_2,\ldots, B_m)$ là

$$R \bowtie_{\text{< Điều kiện nối>}} S$$

Kết quả của phép nối là một quan hệ $Q(A_1,A_2,\ldots,A_n,B_1,B_2,\ldots,B_m)$ có n+m thuộc tính. Mỗi bộ của Q là một sự kết nối giữa một bộ của R và một bộ của S khi chúng thoả mãn điều kiện nối. Sự khác nhau giữa tích Đề các và phép nối là ở chỗ trong phép nối, chỉ có các bộ thoả mãn điều kiện nối mới xuất hiện trong kết quả, trong khi đó trong tích Đề các mọi tổ hợp của các bộ đều có trong kết quả. Điều kiện nối được chỉ ra trên các thuộc tính của hai quan hệ R và S và được tính toán cho mỗi tổ hợp các bộ. Mọi tổ hợp bộ mà điều kiện nối là đúng được chứa trong quan hệ kết quả Q như là một bộ đơn. Một điều kiện nối tổng quát có dạng

<điều kiện> AND <điều kiện> AND … AND <điều kiện>

trong đó mỗi điều kiện có dạng $A_i\ \theta\ B_j$, A_i là một thuộc tính của R, B_j là một thuộc tính của S, A_i và B_j có cùng miền và θ là một trong các dấu phép toán so sánh $\{<, <=, =, >=, >, \neq\}$. Một phép toán nối với điều kiện tổng quát như vậy gọi là một *phép nối tê-ta*. Các bộ có các thuộc tính nối là null không xuất hiện trong kết quả. Theo nghĩa đó, phép toán không nhất thiết phải xử lý mọi thông tin trong các quan hệ tham gia. Ví dụ :

Giả sử ta có hai quan hệ R và S như sau:

R	A1	A2	A3
	Aa	Ca	Ba
	Ab	Cb	Bb
	Ac	Ca	Ba
	Ad	Cc	Null
	Ae	Cd	Bb

S	B1	B2	B3
	Ba	Aaa	Bbb
	Bb	Ccc	Ddd

Khi đó kết quả của phép nối tê-ta R và S với điều kiện A3 = B1 sẽ cho kết quả là:

R ⋈ S	A1	A2	A3	B1	B2	B3
<A3 = B1>	Aa	Ca	Ba	Ba	Aaa	Bbb
	Ab	Cb	Bb	Bb	Ccc	Ddd
	Ac	Ca	Ba	Ba	Aaa	Bbb
	Ae	Cd	Bb	Bb	Ccc	Ddd

Hình 0-5. Phép nối tê-ta hai quan hệ

Phần lớn các phép nối chỉ cho phép các điều kiện nối với các so sánh bằng. Những phép nối chỉ sử dụng phép so sánh bằng được gọi là nối bằng (equi join). Ví dụ trong hình III-8 là một phép nối bằng. Chú ý rằng trong kết quả của phép nối bằng chúng ta thấy luôn luôn có một hoặc nhiều cặp thuộc tính có các giá trị như nhau trong mỗi bộ. Việc có các cặp thuộc tính có giá trị như nhau là thừa, vì vậy người ta đề nghị một phép nối mới gọi là nối tự nhiên, ký hiệu là *. Phép nối tự nhiên nhằm loại bỏ thuộc tính thứ hai (thuộc tính thừa) trong điều kiện nối bằng. Định nghĩa chuẩn của nối tự nhiên đòi hỏi hai thuộc tính nối (hoặc mỗi cặp thuộc tính nối) phải có tên như nhau trong cả hai quan hệ. Nếu các thuộc tính đó không cùng tên thì trước khi nối phải áp dụng phép toán đặt lại tên. Ví dụ, ta cần nối tự nhiên hai quan hệ R(A1,A2,A3) và S(B1,B2,B3) như trong ví dụ trên. Để có thể thực hiện được phép nối tự nhiên với điều kiện so sánh bằng, ta phải đổi tên thuộc tính B1 thành A3, nghĩa là ta phải viết:

$$R * \rho_{(A3,\ B2,B3)}(S)$$

Phép nối sẽ có kết quả như sau:

R * S	A1	A2	A3	B2	B3
	Aa	Ca	Ba	Aaa	Bbb
	Ab	Cb	Bb	Ccc	Ddd
	Ac	Ca	Ba	Aaa	Bbb
	Ae	Cd	Bb	Ccc	Ddd

Hình 0-6. Phép nối tự nhiên hai quan hệ

Nếu các thuộc tính mà trên đó nối tự nhiên được chỉ ra có tên như nhau thì việc đặt lại tên là không cần thiết.

Chú ý rằng nếu không có một tổ hợp các bộ nào thoả mãn điều kiện nối thì kết quả của một phép nối là một quan hệ rỗng không chứa bộ nào. Nói chung, nếu R có n_R bộ và S có n_S bộ thì kết quả của phép nối R với S sẽ có số các bộ lớn hơn 0 và nhỏ hơn $n_R.n_S$. Cỡ của một kết quả nối chia cho cỡ cực đại $n_R.n_S$ tạo nên một tỷ lệ gọi là *chọn lựa nối*, đó là một tính chất của mỗi điều kiện nối. Nếu không có điều kiện nối, mọi tổ hợp các bộ sẽ được chọn và phép nối trở thành một tích Đề các.

Phép nối được sử dụng để kết hợp các dữ liệu từ nhiều quan hệ sao cho các thông tin có liên hệ với nhau có thể được biểu diễn trong một bảng. Đôi khi phép nối được áp dụng nối một bảng với chính nó. Chúng ta có thể áp dụng phép nối tự nhiên và nối bằng để nối nhiều bảng với nhau. Nếu ta nối n bảng với nhau thì phải chỉ ra n-1 điều kiện nối.

1.2.6- <u>Tập hợp đầy đủ các phép toán quan hệ</u>

Người ta đã chỉ rằng tập hợp các phép toán đại số quan hệ $\{\sigma, \pi, \cup, -, \times\}$ là một tập đầy đủ, nghĩa là mọi phép toán đại số quan hệ khác có thể được biểu diễn thông qua các phép toán của tập hợp này. Ví dụ, phép giao có thể được biểu diễn bằng cách sử dụng các phép hợp và trừ tập hợp như sau:

$$R \cap S = (R \cup S) - ((R- S) \cup (S - R))$$

Như vậy, nói một cách chính xác là không cần phải có phép giao. Mỗi khi cần thực hiện một phép giao, ta chỉ cần đưa ra biểu thức phức tạp này là đủ.

Một ví dụ khác, một phép nối có thể được chỉ ra như một tích Đề các và sau đó là một phép chọn:

$$R \underset{\text{< Điều kiện nối>}}{\bowtie} S = \sigma_{\text{< Điều kiện chọn>}}(R \times S)$$

Một cách tương tự, ta có thể thay thế phép nối tự nhiên bằng một tích Đề các đi sau một phép đặt lại tên và sau đó là các phép toán chọn và chiếu. Như vậy các phép toán nối cũng không cần thiết. Tuy nhiên các phép toán đó rất quan trọng bởi vì chúng tiện dùng và rất thường xuyên được áp dụng trong các cơ sở dữ liệu. Các phép toán đó được đưa vào trong đại số quan hệ là do tiện dụng hơn là do cần thiết. Một phép toán khác cũng được đưa vào, đó là phép chia.

*1.2.7- **Phép chia***

Phép chia có lợi cho một loại truy vấn đặc biệt đôi khi có các ứng dụng trong cơ sở dữ liệu. Phép chia được áp dụng cho hai quan hệ R(Z) và S(X) và được ký hiệu là R(Z) ÷ S(X), trong đó $X \subset Z$. Giả sử Y = Z - X (như vậy $Z = X \cup Y$). Kết quả của phép chia là quan hệ T(Y) chứa một bộ t nếu các bộ t_R xuất hiện trong R với $t_R[Y] = t$ và với $t_R[X] = t_S$ với mọi bộ t_S trong S. Điều đó có nghĩa là để một bộ t xuất hiện trong kết quả T của phép chia, các giá trị trong t phải xuất hiện trong R trong sự kết nối với mọi bộ của S.

Ví dụ: Xét phép chia quan hệ R(A,B) cho quan hệ S(A) như hình vẽ dưới đây. Ta thấy chỉ có các thuộc tính B1 và B4 là kết nối với tất cả các bộ của S ở trong R. Vì vậy kết quả nhận được là một quan hệ T(B) với hai giá trị của B là B1 và B4.

Phép chia có thể được biểu diễn thông qua các phép toán π, ×, − như sau:

$$T_1 \leftarrow \pi_Y(R) \; ; \quad T_2 \leftarrow \pi_Y((S \times T_1) - R) \; ; \; T \leftarrow T_1 - T_2$$

R	A	B
	A1	B1
	A2	B1
	A3	B1
	A4	B1
	A1	B2
	A3	B2
	A2	B3
	A3	B3
	A4	B3
	A1	B4
	A2	B4
	A3	B4

S	B
	A1
	A2
	A3

T	A
	B1
	B4

Hình 0-7. Phép chia T(B) = R(A,B) ÷ S(B).

1.3- Các phép toán quan hệ bổ sung

Có nhiều truy vấn cơ sở dữ liêu không thể thực hiện được bằng các phép toán đại số cơ bản trình bày ở trên. Trong phần này chúng ta sẽ trình bày các phép toán bổ sung để biểu diễn các truy vấn đó. Các phép toán này làm tăng cường sức mạnh của đại số quan hệ.

1.3.1- Các hàm nhóm và các phép nhóm

Kiểu câu hỏi đầu tiên không thể biểu diễn được trong đại số quan hệ cơ sở là chỉ ra các hàm nhóm toán học trên các tập hợp giá trị của các cơ sở dữ liệu. Các ví dụ về các hàm như vậy có thể là đưa ra lương trung bình hoặc tổng lương của tất cả nhân viên, hoặc cho biết số các bộ của bảng nhân viên. Các hàm hay áp dụng để thu thập các giá trị số là hàm Tổng (SUM), Trungbình (AVERAGE), Tính giá trị lớn nhất (MAX), Giá trị bé nhất (MIN). Hàm Đếm (COUNT) được sử dụng để đếm các bộ giá trị.

Một kiểu câu hỏi hay dùng khác là đòi hỏi nhóm các bộ trong một quan hệ theo một giá trị của một số các thuộc tính của chúng và sau đó áp dụng các hàm nhóm một cách độc lập cho từng nhóm. Ví dụ, nhóm các bộ của quan hệ NHÂNVIÊN theo MãsốĐV. Như vậy, mỗi nhóm bao gồm các nhân viên cùng làm việc trong một đơn vị. Sau đó chúng ta có thể đưa ra mỗi giá trị của MãsốĐV cùng với lương trung bình của các nhân viên ở trong đơn vị.

Chúng ta có thể định nghĩa một phép toán nhóm như sau:

$$<\text{các thuộc tính nhóm}>\ \mathfrak{I}_{<\text{danh sách các hàm}>}(R)$$

trong đó $\mathfrak{I}$ là ký hiệu phép toán hàm nhóm, <các thuộc tính nhóm> là một danh sách các thuộc tính của quan hệ được chỉ ra trong R, <danh sách hàm> là danh sách các cặp (<hàm><thuộc tính)>). Trong các cặp như vậy, <hàm> là một trong các hàm cho phép như SUM, AVERAGE, MAX, MIN, COUNT, và <thuộc tính> là một thuộc tính của quan hệ được chỉ ra trong R. Quan hệ kết quả có các thuộc tính nhóm cộng với một thuộc tính cho mỗi phần tử trong danh sách hàm. Ví dụ, để lấy ra theo MãsốĐV các nhân viên và lương trung bình của các nhân viên theo từng đơn vị, ta có thể viết:

$$\text{MãsốĐV}\mathfrak{I}_{\ \text{COUNT ( ), AVERAGE(Lương)}}(\text{NHÂNVIÊN})$$

Kết quả được minh hoạ ở hình III-10.

MãsốĐV	COUNT()	AVERAGE(Lương)
1	1	5500
4	3	3100
5	4	3325

Hình 3.10 Minh hoạ phép toán nhóm

Nếu không chỉ ra thuộc tính nhóm thì các hàm được áp dụng cho các giá trị thuộc tính của tất cả các bộ trong quan hệ, vì vậy quan hệ kết quả chỉ có một bộ. Cần chú ý rằng, nói chung, các trùng lặp không được loại bỏ khi hàm nhóm được áp dụng. Kết quả của việc áp dụng một hàm nhóm là một quan hệ chứ không phải là một đại lượng vô hướng, thậm chí nếu nó chỉ có một giá trị.

1.3.2- Các phép toán khép kín đệ quy

Một kiểu phép toán khác, nói chung, không chỉ ra được trong các phép toán đại số quan hệ cơ sở là phép toán khép kín đệ quy. Phép toán này được áp dụng cho mối liên kết đệ quy giữa các bộ cùng kiểu.Với các phép toán này chúng ta phải sử dụng kỹ thuật lặp.

1.3.3- Các phép toán nối ngoài (outer join), hợp ngoài (outer union)

Trong phần này chúng ta thảo luận một vài mở rộng của phép toán nối và hợp. Các phép toán nối mô tả ở trên liên kết các bộ thoả mãn điều kiện nối. Như vậy, các bộ không có bộ liên kết sẽ bị loại khỏi kết quả nối. Các bộ với giá trị null trong các thuộc tính nối cũng bị loại. Một tập hợp các phép toán gọi là nối ngoài có thể được sử dụng khi chúng ta muốn giữ các bộ trong R hoặc S hoặc trong cả hai quan hệ trong kết quả của phép nối dù chúng có những bộ liên kết trong quan hệ kia hay không. Có ba phép nối ngoài gọi là nối ngoài trái (left outer join), nối ngoài phải (right outer join) và nối ngoài đầy đủ (full outer join), được ký hiệu tương ứng là:

Nối ngoài trái Nối ngoài phải Nối ngoài đầy đủ

Phép nối ngoài trái giữ lại mọi bộ trong quan hệ bên trái R trong phép nối. Nếu không có bộ liên kết nào được tìm thấy trong S thì các thuộc tính của S trong kết quả phép nối được "làm đầy" bằng các giá trị null.

Tương tự như vậy đối với các phép nối ngoài phải và các phép nối ngoài đầy đủ.

Phép toán hợp ngoài được mở rộng để lấy hợp của các bộ từ các quan hệ nếu các bộ không tương thích đồng nhất. Phép toán này chỉ lấy hợp của các quan hệ mà chúng chỉ tương thích bộ phận, nghĩa là chỉ một vài thuộc tính của chúng là tương thích phép hợp. Điều phải tôn trọng là danh sách các thuộc tính tương thích phải chứa một khoá cho cả hai quan hệ. Các bộ từ các quan hệ thành phần với cùng một khoá chỉ được biểu diễn một lần trong kết quả và có giá trị cho tất cả các thuộc tính trong kết quả. Các thuộc tính không tương thích phép hợp từ bất kỳ quan hệ nào cũng được giữ trong kết quả và các bộ không có giá trị cho các thuộc tính này cũng được lấp đầy bằng những giá trị null.

1.4- Một số ví dụ về truy vấn trong đại số quan hệ

Trong phần này, chúng ta xét một số ví dụ minh họa việc sử dụng các phép toán đại số quan hệ. Các ví dụ ở đây thực hiện trên cơ sở dữ liệu "CÔNG TY" ở mục II.3 chương III. Nói chung, một truy vấn có thể được thực hiện bằng nhiều cách, sử dụng các phép toán khác nhau. Trong các ví dụ sau, chúng ta xét một cách thực hiện, các bạn đọc có thể tự đưa ra các cách thực hiện khác.

Truy vấn 1: Đưa ra Họ đệm, Tên và địa chỉ của tất cả các nhân viên làm việc cho đơn vị có tên là "Nghiên cứu": (Các quan hệ TG1, TG2 là các kết quả trung gian)

$TG1 \leftarrow \sigma_{\text{TênĐV = "Nghiêncứu"}}(\text{ĐƠNVỊ})$

$TG2 \leftarrow (TG1 * \text{NHÂNVIÊN})$

$\text{KETQUA} \leftarrow \pi_{\text{Họđệm, Tên, Địachỉ}}(TG2)$

Theo cách thực hiện này, quan hệ TG1 chứa thông tin về đơn vị có tên "Nghiên cứu", quan hệ TG2 chứa thông tin về các nhân viên làm việc cho đơn vị "Nghiên cứu" và quan hệ KẾTQUẢ chứa các thông tin theo yêu cầu của truy vấn. Trong các bảng của chúng ta, các thuộc tính nối có tên như nhau nên có thể dùng phép nối tự nhiên.

Truy vấn 2: Với mỗi dự án đặt tại Hà nội, hãy liệt kê MãsốDA, TênĐV, Tên, Địachỉ, Ngàysinh của người quản lý đơn vị.

$TG1 \leftarrow \sigma_{\text{DiadiemDA="Hanoi"}}(\text{DỰÁN})$

$TG2 \leftarrow (TG1 * \text{ĐƠNVỊ})$

TG3 ← (TG2 * NHÂNVIÊN)

KẾTQUẢ ← $\pi_{\text{MãsốDA, MãsốĐV, Họđệm,Địachỉ, Ngàysinh}}$ (TG3)

Truyvấn 3: Hãy tìm tên của các nhân viên làm việc trên tất cả các dự án do đơn vị có mã số = 5 kiểm soát.

TG1 ← $\pi_{\text{MãsốDA}}$ ($\sigma_{\text{Mã sốDV = 5}}$ (DỰÁN))

TG2 ← $\pi_{\text{MãsốNV, MãsốDA}}$ (NHÂNVIÊN_DỰÁN)

TG3 ← TG2 ÷ TG1

KETQUA ← $\pi_{\text{Họđệm, Tên}}$ (TG3 * NHÂNVIÊN)

2. Chuyển đổi mô hình ER thành mô hình quan hệ

Như đã thảo luận ở chương II, bước tiếp theo sau việc xây dựng mô hình dữ liệu mức khái niệm, ta phải chuyển đổi mô hình đó thành một mô hình dữ liệu lôgic. Trong phần này chúng ta sẽ thảo luận về thuật toán chuyển đổi một mô hình ER thành ra mô hình quan hệ.

2.1- Các quy tắc chuyển đổi

Thuật toán chuyển đổi được thực hiện theo các bước sau (dựa trên CSDL "CÔNG TY"):

Bước 1 : Với mỗi kiểu thực thể thông thường E trong lược đồ ER, hãy tạo một quan hệ R chứa mọi thuộc tính đơn của E. Với các thuộc tính phức hợp, chỉ lấy các thuộc tính thành phần đơn của nó. Chọn một trong các thuộc tính khoá của E làm khoá chính cho R. Nếu khoá được chọn của E là phức hợp (gồm nhiều thuộc tính) thì tập các thuộc tính đơn đó sẽ cùng nhau tạo nên khoá chính của R.

Ví dụ: Giả sử ta có kiểu thực thể ĐƠNVỊ với các thuộc tính là MãsốĐV, TênĐV, ĐịađiểmĐV trong đó các thuộc tính khoá là MãsốĐV, TênĐV (do mỗi đơn vị có một tên duy nhất), và ĐịađiểmĐV là một thuộc tính đa trị (do mỗi đơn vị có nhiều địa điểm). Khi đó kiểu thực thể ĐƠNVỊ được chuyển thành quan hệ ĐƠNVI với các thuộc tính MãsốĐV, TênĐV. Khoá chính của quan hệ là MãsốĐV (chọn một trong hai thuộc tính khoá của kiểu thực thể).

Bước 2: Với mỗi kiểu thứ thể yếu W trong lược đồ ER cùng với kiểu thực thể chủ E, hãy tạo một quan hệ R chứa tất các các thành phần đơn (hoặc các thành phần

đơn của các thuộc tính phức hợp) của W như là các thuộc tính của R. Đưa các thuộc tính khoá chính của các quan hệ tương ứng với kiểu thực thể chủ làm khoá ngoài của R. Các thuộc tính này sẽ xác định kiểu liên kết của W. Khoá chính của R là một tổ hợp của khoá chính của các quan hệ tương ứng với kiểu thực thể chủ và khoá bộ phận của kiểu thực thể yếu W nếu có.

Ví dụ: Giả sử ta có kiểu liên kết NHÂNVIÊN <có> CON trong đó NHÂNVIÊN là kiểu thực thể chủ với các thuộc tính MãsốNV, Họđệm, Tên, Ngàysinh, Giớitính. Thuộc tính khoá của NHÂNVIÊN là MãsốNV. CON là kiểu thực thể phụ thuộc (vào thực thể NHÂNVIÊN) với các thuộc tính là Họtêncon, Ngàysinh, Giớitính. Kiểu thực thể này không có thuộc tính khoá. Khi đó kiểu thực thể NHÂNVIÊN được chuyển thành quan hệ NHÂNVIÊN với các thuộc tính như trên. Kiểu thực thể CON được chuyển thành quan hệ CON với các thuộc tính MãsốNV, Họtêncon, Ngàysinh, Giớitính. Quan hệ này có khoá ngoài là MãsốNV, khoá chính là Mã sốNV, Họtêncon.

Bước 3: Với mỗi kiểu liên kết 1:1 R trong lược đồ ER, hãy xác định các quan hệ S và T tương ứng với các kiểu thực thể tham gia trong R. Hãy chọn một trong các quan hệ, chẳng hạn S, và đưa khoá chính của T vào làm khoá ngoài trong S. Tốt nhất là chọn S là một kiểu thực thể tham gia toàn bộ vào R. Đưa tất các các thuộc tính đơn (hoặc các thành phần đơn của các thuộc tính phức hợp) của kiểu liên kết 1:1 R vào làm các thuộc tính của S.

Chú ý: Có một cách chuyển đổi mối liên kết 1:1 nữa là nhập hai kiểu thực thể và mối liên kết thành một quan hệ. Cách này thường được áp dụng khi cả hai kiểu thực thể đều tham gia toàn bộ vào liên kết.

Ví dụ: Giả sử ta có kiểu liên kết NHÂNVIÊN <*quản lý*> ĐƠNVỊ, với các thuộc tính của các kiểu thực thể giống như ở trên. Kiểu liên kết <*quản lý*> là một kiểu liên kết 1:1, đồng thời sự tham gia của NHÂNVIÊN vào kiểu liên kết là bộ phận (không phải nhân viên nào cũng quản lý đơn vị), sự tham gia của ĐƠNVỊ là đầy đủ (một đơn vị luôn luôn phải có một người quản lý). Khi đó, kiểu thực thể NHÂNVIÊN sẽ được chuyển thành quan hệ NHÂNVIÊN với các thuộc tính của nó, còn kiểu thực thể ĐƠNVỊ sẽ được chuyển thành quan hệ ĐƠNVỊ với các thuộc tính của kiểu thực thể ĐƠNVỊ cộng thêm với thuộc tính MãsốNV và thuộc tính của kiểu liên kết <*quản lý*>, nếu có. Thuộc tính MãsốNV sẽ là khoá ngoài cho quan hệ ĐƠNVỊ. Để làm rõ vai trò người quản lý, khi chuyển sang quan hệ ĐƠNVỊ, người

ta đổi tên thuộc tính MãsốNV thành MãsốNQL (Mã số người quản lý). Ngoài ra, kiểu liên kết <*quản lý*> có một thuộc tính là Ngàybắtđầu, thuộc tính này cũng được đưa vào quan hệ ĐƠNVỊ.

Bước 4: Với mỗi kiểu liên kết hai ngôi R kiểu 1:N, hãy xác định quan hệ S biểu diễn kiểu thực thể tham gia ở phía N của kiểu liên kết. Đưa khoá chính của quan hệ T biểu diễn kiểu thực thể tham gia vào R ở phía 1 vào làm khoá ngoài trong S. Làm như vậy là vì mỗi thực thể cụ thể của phía N được liên kết với nhiều nhất là một thực thể cụ thể của phía 1 của kiểu liên kết. Đưa các thuộc tính đơn (hoặc các thành phần đơn của các thuộc tính phức hợp) của kiểu liên kết 1:N vào làm các thuộc tính của S.

Ví dụ: Giả sử ta có kiểu liên kết NHÂNVIÊN <làm việc cho> ĐƠNVỊ, trong đó các kiểu thực thể NHÂNVIÊN, ĐƠNVỊ là các kiểu thực thể ở trên. Kiểu liên kết <làm việc cho> là kiểu liên kết N:1 (một nhân viên chỉ làm việc cho một đơn vị và mỗi đơn vị có nhiều nhân viên làm việc cho). Khi đó, Kiểu thực thể ĐƠNVỊ sẽ được chuyển thành quan hệ ĐƠNVỊ với các thuộc tính của kiểu thực thể ĐƠNVỊ còn kiểu thực thể NHÂNVIÊN sẽ được chuyển thành quan hệ NHÂNVIÊN với các thuộc tính của kiểu thực thể NHÂNVIÊN cộng thêm với thuộc tính MãsốĐV (là khoá chính của quan hệ ĐƠNVỊ). Thuộc tính MãsốĐV sẽ là thuộc tính khoá ngoài của quan hệ NHÂNVIÊN.

Bước 5: Với mỗi kiểu liên kết N:M hai ngôi R, hãy tạo ra một quan hệ mới S để biểu diễn R. Đưa các khoá chính của các quan hệ biểu diễn các kiểu thực thể tham gia vào làm khoá ngoài của S. Tổ hợp các khoá chính đó sẽ tạo nên khoá chính của S. Đưa tất cả các thuộc tính đơn (hoặc các thành phần đơn của các thuộc tính phức hợp) của kiểu liên kết N:M vào làm các thuộc tính của S. Chú ý rằng ta không thể biểu diễn một kiểu liên kết N:M bằng một thuộc tính khoá ngoài đơn giản trong một trong các quan hệ tham gia (như đã làm với các kiểu liên kết 1:1 và 1:N) vì tỷ số lực lượng N:M.

Ví dụ: Giả sử ta có kiểu liên kết NHÂNVIÊN <làm việc với> DỰÁN. Kiểu thực thể NHÂNVIÊN có các thuộc tính như trên với thuộc tính khoá là MãsốNV. Kiểu thực thể DỰÁN có các thuộc tính là MãsốDA, TênDA, ĐịađiểmDA trong đó thuộc tính khoá là MãsốDA. Kiểu liên kết < làm việc với> là một kiểu liên kết N:M (một nhân viên có thể làm việc với nhiều dự án và mỗi dự án có nhiều nhân viên làm việc với). Kiểu liên kết này có một thuộc tính là Sốgiờ để lưu số giờ mà mỗi

nhân viên làm việc cho một dự án. Khi đó kiểu liên kết <làm việc với> sẽ được chuyển thành một quan hệ có tên là NHÂNVIÊN_DỰ ÁN với các thuộc tính MãsốNV, MãsốDA, Sốgiờ trong đó hai thuộc tính MãsốNV, MãsốDA tạo thành khoá chính (phức hợp) cho quan hệ.

Bước 6: Với mỗi thuộc tính đa trị A, hãy tạo ra một quan hệ mới R. Quan hệ R này sẽ chứa một thuộc tính tương ứng với A cộng với thuộc tính khoá K của quan hệ biểu diễn kiểu thực thể hoặc kiểu liên kết có thuộc tính là A làm khoá ngoài của R. Khoá chính của R là một tổ hợp của A và K. Nếu thuộc tính đa trị là phức hợp thì chúng ta chỉ đưa vào R các thành phần đơn của nó.

Ví dụ: Xét kiểu thực thể ĐƠNVỊ ở trên. Thuộc tính ĐịađiểmĐV là một thuộc tính đa trị. Khi chuyển thành mô hình quan hệ nó sẽ được chuyển thành một quan hệ có khoá chính là MãsốĐV, Địa điểm và có thể có thêm một số thuộc tính khác lưu thông tin về địa điểm.

Bước 7: Với mỗi kiểu liên kết n ngôi R, trong đó n > 2, hãy tạo ra một quan hệ S để biểu diễn R. Đưa các khoá chính của các quan hệ biểu diễn các kiểu thực thể tham gia vào làm khoá ngoài của S. Đưa tất cả các thuộc tính đơn (hoặc các thành phần đơn của các thuộc tính phức hợp) của kiểu liên kết n-ngôi vào làm thuộc tính của S. Khoá chính của S thường là một tổ hợp các khoá chính của các quan hệ biểu diễn các kiểu thực thể tham gia. Tuy nhiên, nếu ràng buộc lực lượng trên một kiểu thực thể E nào đó tham gia vào R là 1 thì khoá chính của S không được chứa thuộc tính khoá ngoài tham chiếu đến quan hệ E tương ứng với kiểu thực thể E.

Ví dụ: Giả sử chúng ta có kiểu liên kết ĐẠILÝ <cung cấp> VẬTTƯ <cho> DỰÁN. Đây là một kiểu liên kết cấp ba. Giả sử rằng kiểu thực thể ĐẠILÝ có thuộc tính khoá là MãsốĐL, kiểu thực thể VẬTTƯ có thuộc tính khoá là MãsốVT, kiểu thực thể DỰÁN có thuộc tính khoá là MãsốDA còn kiểu liên kết <cung cấp> có thuộc tính là Sốlượng để lưu số lượng vật tư mà một đai lý cung cấp cho môt dự án. Khi đó kiểu liên kết <cung cấp> sẽ được chuyển thành một quan hệ có tên là CUNGCẤP với các thuộc tính MãsốĐL, MãsốVT , MãsốDA, Sốlượng và khoá chính gồm ba thuộc tính MãsốĐL, MãsốVT , MãsốDA.

2.2- Chuyển đổi mô hình cụ thể

Trong chương 2 chúng ta đã phân tích và thiết kế mô hình ER cho bài toán CÔNGTY. Áp dụng các bước của thuật toán ở trên, chúng ta có mô hình quan hệ cho bài toán CÔNGTY như sau:

NHÂNVIÊN(Họđệm,Tên, MãsốNV, Ngàysinh, Địachỉ, Giớitính, Lương, MãsôNGS, MãsốĐV)

ĐƠNVỊ(TênĐV, MãsốĐV, MãsốNQL, Ngàybắtđầu)

ĐƠNVỊ_ĐỊAĐIỂM(MãsốĐV, ĐịađiểmĐV)

DỰÁN(TênDA, MãsốDA, ĐịađiểmDA, MãsốĐV)

NHÂNVIÊN_DỰÁN(MãsốNV, MãsốDA, Sốgiờ)

PHỤTHUỘC(MãsốNV, Têncon, Giớitính, Ngàysinh)

Hình 0-8. Lược đồ cơ sở dữ liệu "CÔNGTY"

3. Tổng kết chương và câu hỏi ôn tập

3.1- Tổng kết chương

Trong chương này chúng ta đã trình bày các khái niệm cơ bản của mô hình dữ liệu quan hệ. Chúng ta cũng đã thảo luận về đại số quan hệ và các phép toán bổ sung được sử dụng để thao tác các quan hệ. Chương này bắt dầu bằng việc giới thiệu các khái niệm miền, thuộc tính và bộ giá trị. Lược đồ quan hệ được định nghĩa như một danh sách các thuộc tính mô tả cấu trúc của một quan hệ. Một quan hệ (hoặc trạng thái quan hệ) là một tập hợp các bộ giá trị phù hợp với lược đồ.

Có nhiều đặc trưng làm phân biệt các quan hệ với các bảng hoặc các tệp thông thường. Trước tiên, các bộ trong một quan hệ là không có thứ tự. Đặc trưng thứ hai liên quan đến thứ tự của của các thuộc tính trong một lược đồ quan hệ và thứ tự tương ứng của các giá trị bên trong một bộ. Mặc dù ta đã đưa ra một định nghĩa quan hệ khác để chứng minh rằng hai thứ tự này là không cần thiết, tuy nhiên, để thuận tiện ta vẫn đòi hỏi các thuộc tính và các giá trị trong bộ là có thứ tự. Chúng ta cũng đã thảo luận về các giá trị trong các bộ và giới thiệu các giá trị null để biểu diễn thông tin bị thiếu hoặc không biết.

Tiếp theo, chúng ta đã thảo luận về các ràng buộc mô hình quan hệ. Đó là các ràng buộc miền, ràng buộc khóa, các khái niệm về siêu khóa, khóa dự tuyển, khóa

chính và ràng buộc NOT NULL trên các thuộc tính. Sau đó, chúng ta đã định nghĩa cơ sở dữ liệu và lược đồ cơ sở dữ liệu quan hệ. Các ràng buộc toàn vện thực thể và toàn vẹn tham chiếu cũng đã được định nghĩa và phân tích. Toàn vẹn thực thể ngăn cấm việc khóa chính có giá trị null. Toàn vẹn tham chiếu được sử dụng để duy trì sự nhất quán của việc tham chiếu trong các bộ từ các quan hệ khác nhau.

Các phép cập nhật trên mô hình quan hệ gồm Insert, Delete, Update. Mỗi một phép toán có thể vi phạm các kiểu ràng buộc nhất định. Mỗi khi một phép toán được áp dụng, trạng thái cơ sở dữ liệu sau khi phép toán được thực hiện phải được kiểm tra để đảm bảo rằng không có một ràng buộc nào bị vi phạm.

Tiếp theo chúng ta đã mô tả đại số quan hệ cơ sở, đó là một tập hợp các phép toán thao tác quan hệ và có thể được sử dụng để đưa ra các truy vấn. Chúng ta đã định nghĩa và phân tích cách sử dụng các phép toán như chiếu, chọn, tích Đềcác, nối, phép đặt lại tên. Các phép toán tập hợp như giao, hợp, trừ cũng được định nghĩa và phân tích.

Tiếp theo, chúng ta thảo luận về các kiểu truy vấn quan trọng không thể sử dụng được các phép toán đại số quan hệ cơ sở. Chúng ta đã giới thiệu phép toán hàm nhóm để làm việc với các kiểu yêu cầu nhóm. Các kiểu truy vấn đệ quy cũng được thảo luận và giới thiệu cách chỉ ra một số kiểu truy vấn đệ quy. Các phép nối ngoài, hợp ngoài, mở rộng của phép nối và phép hợp cũng được thảo luận ở cuối chương.

Cuối cùng, chúng ta làm quen với thuật toán chuyển đổi từ mô hình ER sang mô hình quan hệ. Mô hình ER cho "CÔNGTY" được xây dựng ở chương II đã được chuyển đổi thành lược đồ cơ sở dữ liệu quan hệ.

3.2- Câu hỏi ôn tập

1) Định nghĩa các thuật ngữ sau: miền, thuộc tính, n-bộ, lược đồ quan hệ, trạng thái quan hệ, cấp của quan hệ, lược đồ cơ sở dữ liệu, trạng thái cơ sở dữ liệu.
2) Vì sao các bộ trong một quan hệ là không có thứ tự.
3) Vì sao không cho phép các bộ trùng lặp trong một quan hệ.
4) Siêu khóa và khóa khác nhau ở chỗ nào.
5) Vì sao phải chỉ định một trong các khóa dự tuyển làm khóa chính.

6) Nêu những đặc trưng làm cho các quan hệ khác với các bảng hoặc các tệp thông thường.

7) Nêu các lý do về việc tồn tại các giá trị không xác định trong các quan hệ.

8) Hãy giải thích về ràng buộc toàn vẹn thực thể và ràng buộc toàn vẹn tham chiếu. Vì sao các ràng buộc này là quan trọng?

9) Định nghĩa khóa ngoài. Khái niệm này dùng để làm gì? Các khóa ngoài đóng vai trò như thế nào trong phép nối?

10) Hãy giải thích các phép toán cập nhật trên các quan hệ và các kiểu ràng buộc toàn vẹn phải được kiểm tra đối với mỗi phép toán cập nhật.

11) Liệt kê các phép toán đại số quan hệ và mục đích của từng phép toán.

12) Tương thích hợp là gì? Vì sao các phép toán hợp, giao, trừ đòi hỏi các quan hệ tham gia vào phép toán phải tương thích hợp?

13) Hãy giải thích các kiểu truy vấn cần có việc đặt lại tên các thuộc tính để chỉ ra truy vấn một cách rõ ràng.

14) Hãy nêu các kiểu phép toán nối khác nhau.

15) Phép toán hàm là gì? Nó được dùng vì mục đích nào?

16) Các phép nối ngoài khác với các phép nối trong như thế nào? Phép hợp ngoài khác với phép hợp như thế nào?

3.3- Bài tập

1) Chuyển đổi các lược đồ ER của các bài tập 1, 2 ở chương II thành lược đồ cơ sở dữ liệu quan hệ.

2) Cho lược đồ cơ sở dữ liệu Thưviện:

SACH(Mãsách, Tênsách, TênNXB)

SACH_TACGIA(Masach, TênTG)

NHAXUATBAN(TênNXB, Địachỉ, Điệnthoại)

SACH_BANSAO(Mãsách, Mãnhánh, Sốlượngbảnsao)

NHANH_THUVIEN(Mãnhanh, Tênnhánh, Địachỉ)

SACH_MUON(Mãsách, Mãnhánh, Sốthẻ, Ngàymượn, Ngàytrả)

NGUOIMUON(Sốthẻ, Tên, Địachỉ, Điệnthoại)

Hãy viết các biểu thức quan hệ cho các truy vấn sau đây trên cơ sở dữ liệu Thưviện:

1. Có bao nhiêu bản sao của cuốn sách "The Lost Tribe" có trong nhánh thư viện có tên là "Shapstown".
2. Có bao nhiêu bản sao của cuốn sách "The Lost Tribe" có trong mỗi nhánh thư viện.
3. Đưa ra tên của tất cả người muợn chưa mượn cuốn sách nào.
4. Với mỗi cuốn sách được mượn ra từ nhánh thư viện "Shapstown" có ngày trả là ngày hôm nay, hãy đưa ra Tên sách, Tên người mượn và địa chỉ người mượn.
5. Với mỗi thư viện nhánh, hãy đưa ra tên nhánh thư viện và tổng số sách được mượn ra từ nhánh này.
6. Đưa ra tên, địa chỉ và số các sách do người này mượn với những người mượn nhiều hơn 5 cuốn sách.
7. Với mỗi cuốn sách có tác giả (hoặc đồng tác giả) là "Stephen King", hãy đưa ra tên sách và số lượng các bản sao có tại nhánh thư viện có tên là "Central".

3) Cho cơ sở dữ liệu CÔNGTY gồm các lược đồ:

NHÂNVIÊN(Mã sốNV, Họđệm, Tên, Ngàysinh, Giớitính, Địachỉ, Lương, MãsốNGS, Mã sốĐV)

ĐƠNVỊ(MãsốĐV, TênĐV, MãsốNQL, Ngàybắtđầu)

DỰÁN(MãsốDA, TênDA, ĐịađiểmDA, MãsốĐV)

PHỤTHUỘC(MãsốNV, TênPT, Ngày sinh, Giớitính, Quanhệ)

NHÂNVIÊN_DỰÁN(MãsốNV, MãsốDA, Sốgiờ)

ĐƠNVỊ_ĐỊAĐIỂM(MãsốĐV, Địađiểm)

Hãy viết các biểu thức quan hệ thực hiện các truy vấn sau:

a) Đưa ra tên và địa chỉ của tất cả các nhân viên làm việc cho đơn vị nghiên cứu.

b) Với mỗi dự án có địa điểm tại Hà nội, hãy liệt kê mã số dự án, mã số của đơn vị kiểm soát, Tên, địa chỉ và ngày sinh của người quản lý đơn vị

c) Tìm tên của các nhân viên làm việc trên tất cả các dự án do đơn vị có mã số 5 kiểm soát.

d) Tạo ra một danh sách các mã số dự án đối với các dự án có một nhân viên hoặc một người quản lý đơn vị kiểm soát dự án có tên là ‘Nam’.

e) Đưa ra tên của tất cả các nhân viên có nhiều hơn hoặc bằng 2 người phụ thuộc.

f) Đưa ra các nhân viên không có người phụ thuộc.

g) Đưa ra tên của những người quản lý có ít nhất là một người phụ thuộc.

CHƯƠNG 5 - NGÔN NGỮ TRUY VẤN SQL

Truy vấn dữ liệu trong sql là thao tác trích xuất thông tin được lưu trữ trong các table. Thông tin được truy xuất thông qua các cột và thông tin cần trích xuất có thể thuộc một hoặc nhiều bảng.

Thao tác này được sử dụng rất nhiều trong các hệ thống phần mềm hoặc website, chẳng hạn khi các bạn đăng nhập vào facebook thì hệ thống sẽ thực hiện truy vấn dữ liệu để kiểm tra tích hợp lệ của tài khoản đăng nhập, ….

1. Ngôn ngữ SQL

SQL viết tắt của **S**tructured **Q**uery **L**anguage, là ngôn ngữ truy vấn có cấu trúc. SQL cho phép chúng ta tạo cơ sở dữ liệu, thao tác trên dữ liệu như lưu trữ dữ liệu, sửa dữ liệu hoặc xóa dữ liệu hoặc truy vấn dữ liệu.

Đa số các hệ quản trị cơ sở dữ liệu hiện nay sử dụng SQL (MS SQL Server – T-SQL, Microsoft Access, Oracle – PL/SQL, DB2, MySQL…)

SQL có thể chia thành 4 nhóm

Nhóm truy vấn dữ liệu (DQL): gồm các lệnh truy vấn lựa chọn (Select) để lấy thông tin nhưng không làm thay đổi dữ liệu trong các bảng

Nhóm định nghĩa dữ liệu (DDL): Gồm các lệnh tạo, thay đổi các bảng dữ liệu (Create, Drop, Alter, …)

Nhóm thao tác dữ liệu (DML): Gồm các lệnh làm thay đổi dữ liệu lưu trong các bảng (Insert, Delete, Update,…)

Nhóm điều khiển dữ liệu (DCL): Gồm các lệnh quản lý quyền truy nhập vào dữ liệu và các bảng (Grant, Revoke, …)

1.1- NGÔN NGỮ ĐỊNH NGHĨA DỮ LIỆU (DDL-Data Default Languege)

1.1.1- Tạo một cơ sở dữ liệu

Cú pháp:

```
Create Database <Tên CSDL>
```

Tạo một cơ sở dữ liệu có tên là QLTV _ Quản lý thư viện

```
Create Database QLTV;
```

1.1.2- Tạo một bảng

Cú pháp

```
CREATE TABLE <Ten bang>
(
    Tên_thuộc_tính1 Kiểu_tt1 [NOT NULL],
    Tên_thuộc_tính2 Kiểu_tt2 [NOT NULL],
    Tên_thuộc_tínhn Kiểu_ttn [NOT NULL]
    [, CONSTRAINT mệnh đề ]
)
```

Trong đó, mệnh đề

```
CONSTRAINT
```

cho phép ta khai báo các ràng buộc dữ liệu (chi tiết sẽ được trình bày ở phần sau).
Tạo bảng **DOCGIA**, có các thuộc tính:

```
CREATE TABLE DOCGIA(
   MaDG Text(10) NOT NULL PRIMARY KEY,
   TenDG Text(30) NOT NULL,
   DiaChi Text(50) NOT NULL,
   Tuoi NUMBER)
```

Bảng này sẽ được nhận một tên gọi và một cấu trúc (danh sách tên các thuộc tính và một vài đặc trưng). Khi mới được tạo, bảng chưa có dữ liệu, chỉ là một cấu trúc lôgic có thể tiếp nhận các dữ liệu.

1.1.3- Tên của bảng

Tên của bảng được xác định ngay sau lệnh

```
CREATE TABLE
```

.

Mỗi HQTCSDL có một quy tắc đặt tên riêng.

- Tên bảng phải bắt đầu bằng một chữ cái, có dưới 30 kí tự (chữ cái, chữ số, và dấu '_').
- Tên bảng phải khác tên gọi khác của bảng hay của khung nhìn và với tên gọi đã dành riêng của SQL.
- Không phân biệt hoa, thường.

1.1.4- Xác định các thuộc tính

Trong lệnh tạo bảng ta phải xác định cấu trúc của bảng. Cần phải xác định mỗi thuộc tính của một định nghĩa kết thúc bằng dấu '**,**' và gồm:

- Tên thuộc tính
- Loại dữ liệu và độ dài
- Các ràng buộc có liên quan.

1.1.5- Các loại dữ liệu

Các loại dữ liệu được sử dụng còn tùy theo HQTCSDL.

1.1.6- Các loại dữ liệu được sử dụng trong MS Access

Kiểu dữ liệu	Miêu tả	Kích cỡ
Text	Sử dụng ký tự hoặc kết hợp giữa ký tự và số, như địa chỉ, hoặc những số không yêu cầu tính toán, như số điện thoại, mã nước, mã vùng…	Khả năng lưu trữ tối đa (FieldSize)là 255 ký tự.
Memo	Sử dụng khi bạn cần lưu trữ một lượng thông tin lớn, ví dụ như trường thông tin ghi chú về một cán bộ.	Khả năng lưu trữ tối đa là 65.536 ký tự.
Number	Number: Sử dụng cho những dữ liệu cần tính toán (loại trừ tính tiền, sử dụng Currency Type).	Khả năng lưu trữ có thể là 1, 2, 4, 8 tuỳ thuộc bào kiểu dữ liệu ta chọn (byte, integer, long integer, single, double, decimal), riêng đối với kiểu dữ liệu ReplicationID (GUI) thi khả năng lưu trữ là 16 byte.
Date/Time	Lưu trữ thông tin về thời gian.	Sử dụng 8 byte để lưu trữ.
Currency	Sử dụng Currency cho các dữ liệu cần tính toán. Phần thập	Khả năng lưu trữ là 8 byte.

	phân có thể có từ 1 đến 4 số.	
AutoNumber	Đây là kiểu số tự động tăng với bước tăng là 1. Ta không thể cập nhật lại được trường này.	Sử dụng 4 byte để lưu trữ. Nếu chọn kiểu dữ liệu là ReplicationID thì khả năng lưu trữ có thể lên tới 16 byte.
Yes/No	Kiểu dữ liệu YES/NO chỉ chứa một trong 2 giá trị (Yes/No, True/False, On/ Off)Y	Sử dụng 1 bite để lưu trữ.
OLE Object	Đối tượng (như là một văn bản trong Microsoft Word, dữ liệu đồ hoạ, âm thanh, hoặc một kiểu dữ liệu nhị phân…)	Sử dụng 1 GB để lưu trữ (tuỳ thuộc vào dung lượng của đĩa).

Ngoài ra còn 2 loại dữ liệu khác như Hyperlink, Lookup Wizard.

Đối với kiểu dữ liệu Number, ta còn có thể lựa chọ chi tiết:

Kiểu dữ liệu	Miêu tả	**Độ chính xác thập phân**	Kích cỡ
Byte	Lưu trữ số từ 0 đến 255 (không có phân số)	Không	1 byte
Decimal	Lưu trữ tối đa 10^38-1	28	12bytes
Integer	Lưu trữ số từ -32,768 to 32,767 (không có phân số).	Không	2 bytes
Long Integer	Lưu trữ số từ -2,147,483,648 tới 2,147,483,647 (không có phân số).	None	4 bytes
Single	Lưu trữ số từ -3.402823E38 to -1.401298E-45 cho giá trị âm và từ 1.401298E-45 to 3.402823E38 giá trị dương.	7	4 bytes
Double	Lưu trữ số từ -1.79769313486231E308 tới -4.94065645841247E-324 cho giá trị âm và từ 4.94065645841247E–324 to 1.79769313486231E308 giá trị dương.	15	8 bytes

1.1.7- Các loại dữ liệu được sử dụng trong Oracle

1. CHAR

Kiểu CHAR dùng để khai báo một chuỗi có chiều dài cố định, khi khai báo biến hoặc cột kiểu CHAR với chiều dài chỉ định thì tất cả các mục tin của biến hay cột này đều có cùng chiều dài được chỉ định. Các mục tin ngắn hơn ORACLE sẽ tự động thêm vào các khoảng trống cho đủ chiều dài. ORACLE không cho phép gán mục tin dài hơn chiều dài chỉ định đối với kiểu CHAR. Chiều dài tối đa cho phép của kiểu CHAR là 255 byte

2. VARCHAR2

Kiểu VARCHAR2 dùng để khai báo chuỗi ký tự với chiều dài thay đổi. Khi khai báo một biến hoặc cột kiểu VARCHAR2 phải chỉ ra chiều dài tối đa, các mục tin chứa trong biến hay cột kiểu VARCHAR2 có chiều dài thực sự là chiều dài của mục tin. ORACLE không cho phép gán mục tin dài hơn chiều dài tối đa chỉ định đối với kiểu VARCHAR2. Chiều dài tối đa kiểu VARCHAR2 là 2000 byte

3. VARCHAR

Hiện tại ORACLE xem kiểu VARCHAR2 và VARCHAR là như nhau, tuy nhiên ORACLE khuyên nên dùng VARCHAR2. ORACLE dự định trong tương lai dùng kiểu VARCHAR để chứa các chuỗi với chiều dài biến đổi, nhưng trong phép so sánh sẽ được chỉ định theo nhiều ngữ nghĩa khác nhau.

4. NUMBER

Kiểu số của ORACLE dùng để chứa các mục tin dạng số dương, số âm, số với dấu chấm động.

NUMBER(p, s)

Trong đó:

p: số chữ số trước dấu chấm thập phân (precision), p từ 1 đến 38 chữ số

s: số các chữ số tính từ dấu chấm thập phân về bên phải (scale), s từ -84 đến 127

NUMBER(p) số có dấu chấm thập phân cố định với precision bằng p và scale bằng 0

NUMBER số với dấu chấm động với precision bằng 38. Nhớ rằng scale không được áp dụng cho số với dấu chấm động.

Ví dụ sau cho thấy cách thức ORACLE lưu trữ dữ liệu kiểu số tùy theo cách định precision và scale khác nhau:

Dữ liệu thực	Kiểu	Lưu trữ
7456123.89	NUMBER	7456123.89
7456123.89	NUMBER(9)	7456123
7456123.89	NUMBER(9,2)	7456123.89
7456123.89	NUMBER(9,1)	7456123.8
7456123.89	NUMBER(6)	Không hợp lệ

7456123.8	NUMBER(15,1)	7456123.8
7456123.89	NUMBER(7,-2)	7456100
7456123.89	NUMBER(-7,2)	Không hợp lệ

5. FLOAT

Dùng để khai báo kiểu số dấu chấm động, với độ chính xác thập phân 38 hay độ chính xác nhị phân là 126.

FLOAT(b) Khai báo kiểu dấu chấm động với độ chính xác nhị phân là b, b từ 1 đến 126. Có thể chuyển từ độ chính xác nhị phận sang độ chính xác thập phân bằng cách nhân độ chính xác nhị phân với 0.30103.

6. LONG

Dùng để khai báo kiểu chuỗi ký tự với độ dài biến đổi, chiều dài tối đa của kiểu LONG là 2 gigabyte. Kiểu LONG thường được dùng để chứa các văn bản.

Có một số hạn chế khi dùng kiểu LONG:

- Một table không thể chứa nhiều hơn một cột kiểu LONG.
- Dữ liệu kiểu LONG không thể tham gia vào các ràng buộc toàn vẹn, ngoại trừ kiểm tra NULL và khác NULL.
- Không thể index một cột kiểu LONG.
- Không thể truyền tham số kiểu LONG cho hàm hoặc thủ tục.
- Các hàm không thể trả về dữ liệu kiểu LONG.
- Trong câu lệnh SQL có truy cập các cột kiểu LONG, thì việc cập nhật hoặc khóa các bảng chỉ cho phép trong cùng một CSDL

Ngoài ra, các cột kiểu LONG không được tham gia trong các thành phần sau của câu lệnh SQL:

- Các mệnh đề

```
WHERE
```

,

```
GROUP BY
```

,

```
ORDER BY
```

,

```
CONNECT BY
```

hoặc với tác tử

DISTINCT

trong câu lệnh

SELECT

- Các hàm sử dụng trong câu lệnh SQL như SUBSTR, INSTR.
- Trong danh sách lựa chọn của câu lệnh

SELECT

có sử dụng mệnh đề

GROUP BY

- Trong danh sách lựa chọn của câu hỏi con, câu hỏi có sử dụng các toán tử tập hợp.
- Trong danh sách lựa chọn của câu lệnh

CREATE TABLE AS SELECT

7. DATE

Dùng để chứa dữ liệu ngày và thời gian. Mặc dù kiểu ngày và thời gian có thể được chứa trong kiểu CHAR và NUMBER.

Với giá trị kiểu DATE, những thông tin được lưu trữ gồm thế kỷ, năm, tháng, ngày, giờ, phút, giây. ORACLE không cho phép gán giá trị kiểu ngày trực tiếp, để gán giá trị kiểu ngày, bạn phải dùng TO_DATE để chuyển giá trị kiểu chuỗi ký tự hoặc kiểu số.

Nếu gán một giá trị kiểu ngày mà không chỉ thời gian thì thời gian mặc định là 12 giờ đêm, Nếu gán giá trị kiểu ngày mà không chỉ ra ngày, thì ngày mặc định là ngày đầu của tháng. Hàm SYSDATE cho biết ngày và thời gian hệ thống.

Tính toán đối với kiểu ngày:

Đối với dữ liệu kiểu ngày, bạn có thể thực hiện các phép toán cộng và trừ.

- SYSDATE+1 ngày hôm sau
- SYSDATE-7 cách đây một tuần
- SYSDATE+(10/1440) mười phút sau
- Ngày Julian: Là giá trị số cho biết số ngày kể từ ngày 1 tháng giêng năm 4712 trước công nguyên.

```
SELECT TO_CHAR (TO_DATE('01-01-1992', 'MM-DD-YYYY'), 'J') JULIAN FROM DUAL
```

Cho kết quả:

```
JULIAN
----------------------------------------------
```

2448623

8. RAW và LONG RAW

Kiểu RAW và LONG RAW dùng để chứa các chuỗi byte, các dữ liệu nhị phân như hình ảnh, âm thanh. Các dữ liệu kiểu RAW chỉ có thể gán hoặc truy cập chứ không được thực hiện các thao tác như đối với chuỗi ký tự.

Kiểu RAW giống như kiểu VARCHAR2 và kiểu LONG RAW giống kiểu LONG, chỉ khác nhau ở chổ ORACLE tự động chuyển đổi các giá trị kiểu CHAR, VARCHAR2 và LONG giữa tập hợp ký tự của CSDL và tập ký tự của các ứng dụng.

9. ROWID

Mỗi mẫu tin trong CSDL có một địa chỉ có kiểu ROWID. ROWID gồm block.row.file, trong đó:

block : chuỗi hệ hexa cho biết block chứa row

row : chuỗi hệ hexa cho biết row trong block

file : chuỗi hệ hexa cho biết database file chứa block

0000000F.0000.0002

Row đầu tiên trong block 15 của data file thứ hai.

10. MLSLABEL

Kiểu MLSLABEL dùng để chứa label dạng nhị phân mà ORACLE dùng để đảm bảo hoạt động của bản thân hệ thống.

1.1.8- Các loại dữ liệu sử dụng trong SQL SERVER

Phần này sẽ được trình bày trong phần thực hành.

1.1.9- Các loại ràng buộc trong bảng dữ liệu

Các dạng constraint gồm:

- NOT NULL
- UNIQUE
- PRIMARYKEY
- FOREIGN KEY (Referential)
- CHECK

1.1.10- NOT NULL- Không rỗng

Khi có mệnh đề

NOT NULL

có trong định nghĩa của một cột thì ta bắt buộc thuộc tính này phải có giá trị. Nếu ta không chỉ thị gì trong định nghĩa của thuộc tính thì nó có thể có hoặc không có giá trị.

```
CREATE TABLE NHANVIEN(
    MaNV NUMBER(10) NOT NULL,
    TenNV CHAR(30)
)
```

1.1.11- UNIQUE-Duy nhất

Chỉ ra ràng buộc duy nhất, các giá trị của cột chỉ trong mệnh đề UNIQUE trong các row của table phải có giá trị khác biệt. Giá trị null là cho phép nêu UNIQUE dựa trên một cột.

```
CREATE TABLE NHANVIEN (
        MaNV NUMBER(10) NOT NULL,
        TenNV CHAR(30),
        DiachiNV CHAR(50))
```

CONSTRAINT UNQ_Ten_Diachi UNIQUE(Ten,Diachi))

1.1.12- PRIMARY KEY- Khoá chính

Chỉ ra ràng buộc duy nhất (giống), tuy nhiên khoá là dạng khoá

```
UNIQUE
```

. cấp cao nhất. Một table chỉ có thể có một

```
PRIMARY KEY
```

. Các giá trị trong PRIMARY KEY phải

```
NOT NULL
```

Cú pháp:

```
[CONSTRAINT constraint_name ]
```

```
PRIMARY KEY [CLUSTERED|NONCLUSTERED]
[( colname [,colname2 [...,colname16]])]

CREATE TABLE NHANVIEN
(
    MaNV char(10) NOT NULL primary key,
    TenNV char(30),
    DiachiNV char(50)
)
```

Hoặc ta có thể viết câu lệnh sau:

```
CREATE TABLE NHANVIEN
(
    MaNV char(10) NOT NULL,
    TenNV char(30),
    DiachiNV char(50),
    CONSTRAINT NV_P_K PRIMARY KEY (MaNV)
)
```

1.1.13- FOREIGN KEY-Khoá ngoại

Chỉ ra mối liên hệ ràng buộc tham chiếu giữa bảng này với bảng khác.

Từ khoá

```
ON DELETE CASCADE
```

được chỉ định trong dạng khoá này để chỉ khi dữ liệu cha bị xoá thì dữ liệu con cũng tự động bị xoá theo.

Cú pháp:

```
[CONSTRAINT constraint_name ]
[FOREIGN KEY (colname [,colname2 [...,colname16]])]
REFERENCES reference_table [(ref_colname[,ref_colname2[...,ref_colname 16]])]
```

Hai bảng DONVI và bảng NHANVIEN có mối quan hệ cha – con (1_N). Thuộc tính MaDV trong bảng NHANVIEN(bảng con) là khoá ngoại, được tham chiếu từ thuộc tính MaDV của bảng DONVI(bảng cha)

Ta tạo 2 bảng như sau

```
CREATE TABLE DONVI
(
    MaDV char(2) primary key,
    TenDV char(20) not null
)
CREATE TABLE NHANVIEN
(
    MaNV char(10) primary key,
    TenNV char(30) not null,
    Diachi char(50),
    madv char(2)
    CONSTRAINT k_n_madv FOREIGN KEY(madv) REFERENCES DONVI(MaDV)
)
```

1.1.14- CHECK- Ràng buộc kiểm tra giá trị

Ràng buộc CHECK được sử dụng để yêu cầu các giá trị trong cột, hoặc khuôn dạng dữ liệu trong cột phải theo một quy tắc nào đó. Trên một cột có thể có nhiều ràng buộc này. Để khai báo một rang buộc CHECK cho một cột nào đó ta dùng cú pháp sau.

Cú pháp:

```
[CONSTRAINT constraint_name]
   CHECK (expression)
```

Trong đó, expression là một biểu thức logic. Sau khi có ràng buộc này, giá trị nhập vào cho cột phải thoả mãn điều kiện mới được chấp nhận.

```
CREATE TABLE NHANVIEN
(
    MaNV CHAR(10) NOT NULL PRIMARY KEY,
    TenNV CHAR(30),
    Luong NUMBER(10,2)
    CONSTRAINT CK_SAL CHECK(SAL>500)
)
```

1.1.15- DEFAULT-Mặc định

Ràng buộc DEFAULT được sử dụng để quy định giá trị mặc định cho một cột. Giá trị này sẽ tự động gán cho cột nếu người sử dụng không nhập vào khi bổ sung bản ghi.

Cú pháp:

```
[CONSTRAINT constraint_name]
   DEFAULT {const_expression/nonarguments_function/NULL}
   CREATE TABLE NHANVIEN
(
    MaNV char(10) primary key,
    TenNV char(30) not null,
    Gioitinh char(3) DEFAULT ‘Nam’
)
```

1.1.16- Sửa đổi cấu trúc

Có thể sửa đổi cấu trúc của bảng hiện đang tồn tại bằng lệnh ALTER. Chúng ta có thể thêm một thuộc tính (cột) mới, thay đổi cấu trúc của một thuộc tính (cột) đang có, bổ sung khoá, bổ sung ràng buộc.

Cú pháp tổng quát

```
ALTER TABLE table_name
   [ADD
   {col_name column_properties [column_constraints]
   [[,]table_constraint ] }
   [,{next_col_name|next_table_constraint}]...]
```

```
[DROP
[CONSTRAINT] constraint_name1
[, constraint_name2]...]
ALTER
{col_name column_properties [column_constraints]
[[,]table_constraint ] }
[,{next_col_name|next_table_constraint}]...]
```

Thêm một ràng buộc CHECK

```
ALTER TABLE DONVI
ADD CONSTRAINT check_madv
CHECK (MaDV LIKE '[0-9][0-9]')
```

Thêm một thuộc tính

Cú pháp:

```
ALTER TABLE <Tên_bảng>
ADD COLUMN Tên_cột , Kiểu_cột[(size)])

ALTER TABLE DONVI
    ADD(GhiChu, VARCHAR(255))
```

Trong một số HQTCSDL ta cần phải thêm từ khoá COLUMN như sau:

Cú pháp:

```
ALTER TABLE <Tên_bảng>
ADD COLUMN Tên_cột , Kiểu_cột[(size)] )

ALTER TABLE NHANVIEN
ADD COLUMN GhiChu Text(50));
```

Thay đổi kiểu của một thuộc tính

Cú pháp:

```
ALTER TABLE <Tên_bảng>
ALTER (Tên_cột, Kiểu_mới)

ALTER TABLE NHANVIEN
ALTER(HoTen, VARCHAR(40))
```

Trong một số HQTCSDL ta cần phải thêm từ khoá COLUMN như sau

Cú pháp:

```
ALTER TABLE <Tên_bảng>
ALTER COLUMN Tên_cột , Kiểu_cột_mới[(size)]

ALTER TABLE NHANVIEN
ALTER COLUMN GhiChu Memo
```

Xóa một thuộc tính

Cú pháp:

```
ALTER TABLE <Tên_bảng>
DROP <Tên_thuộc_tính>
ALTER TABLE NHANVIEN
DROP GhiChu
```

Trong một số HQTCSDL ta cần phải thêm từ khoá COLUMN như sau

Cú pháp:

```
ALTER TABLE <Tên_bảng>
   DROP COLUMN Tên_cột

ALTER TABLE NHANVIEN
DROP COLUMN GhiChu
```

1.1.17- Xoá đối tượng

Cú pháp:

```
DROP <Object_name>
DROP TABLE SINHVIEN
```

1.1.18- Khung Nhìn (VIEW)

Các quan hệ được định nghĩa với lệnh CREATE TABLE tồn tại thực sự trong cơ sở dữ liệu. Như vậy, hệ thống SQL lưu trữ các bảng trong một tổ chức vật lý nào đó. Chúng là thường trực và tồn tại lâu dài, chỉ bị thay đổi khi thực hiện lệnh INSERT hoặc các lệnh cập nhật.

Có một lớp các quan hệ khác của SQL, gọi là các khung nhìn, không tồn tại một cách vật lý. Đúng hơn là chúng được định nghĩa bằng một biểu thức giống như một truy vấn. Các khung nhìn có thể được truy vấn như là chúng tồn tại một cách vật lý, và trong một số trường hợp, chúng ta có thể sửa đổi các khung nhìn.

(1) Khai báo các khung nhìn

Dạng đơn giản nhất của một định nghĩa khung nhìn là

1. Các từ khoá CREATE VIEW,
2. Tên của khung nhìn,
3. Từ khoá AS và
4. Một truy vấn Q.

Truy vấn này là định nghĩa của khung nhìn. Mỗi khi chúng ta truy vấn khung nhìn, SQL ứng xử như là Q đã được thực hiện tại thời điểm đó và truy vấn được áp dụng đối với quan hệ do Q sinh ra. Như vậy, một khai báo khung nhìn đơn giản có dạng:
CREATE VIEW < tên khung nhìn> AS < định nghĩa khung nhìn>;
Vídụ 40: Giả sử chúng ta muốn có một khung nhìn là một phần của quan hệ NHÂNVIÊN, chứa MãsốNV, Họđệm,Tên, Lương và MãsốĐV của các nhân viên có địa chỉ là ‘Hànội’. Chúng ta có thể định nghĩa khung nhìn này bằng:
1) CREATE VIEW NVHÀNỘI AS

2) SELECT MãsốNV, Họđệm,Tên, Lương, MãsốĐV

3) FROM NHÂNVIÊN

4) WHERE Địachỉ = ‘Hà nội’ ;

Theo định nghĩa này, tên của khung nhìn là NVHÀNỘI, các thuộc tính của khung nhìn là MãsốNV, Họđệm,Tên, Lương, Địachỉ, MãsốĐV. Định nghĩa của khung nhìn là từ dòng 2 đến dòng 4).

(2) Truy vấn các khung nhìn

Quan hệ NVHÀNỘI không chứa các bộ theo nghĩa thông thường. Đúng hơn là nếu chúng ta truy vấn NVHÀNỘI, các bộ thích hợp sẽ nhận được từ bảng cơ sở NHÂNVIÊN, vì vậy truy vấn có thể được trả lời. Kết quả là chúng ta có thể hỏi NVHÀNỘI hai lần cùng một truy vấn và nhận được các trả lời khác nhau. Lý do là ở chỗ, mặc dù chúng ta không thay đổi định nghĩa của khung nhìn NVHÀNỘI nhưng bảng cơ sở NHÂNVIÊN có thể bị thay đổi trong thời gian giữa hai lần truy vấn.
Vídụ 41 Chúng ta có thể truy vấn khung nhìn NVHÀNỘI như thể nó là một bảng được lưu giữ, chẳng hạn:

```
SELECT Tên
FROM NVHÀNỘI
WHERE MãsốĐV = 4 ;
```

Định nghĩa của khung nhìn NVHÀNỘI được sử dụng để biến đổi truy vấn ở trên thành truy vấn mới chỉ nhắm đến bảng cơ sở NHÂNVIÊN. Truy vấn trên tương đương với truy vấn

```
SELECT Tên
FROM NHÂNVIÊN
WHERE Địachỉ = ‘Hà nội’AND MãsốĐV = 4 ;
```

Ví dụ 42 Có thể viết các truy vấn chứa cả bảng lẫn khung nhìn, chẳng hạn:

```
SELECT TênĐV, Tên
FROM NVHÀNỘI, ĐƠNVỊ
WHERE NVHÀNỘI.MãsốĐV = ĐƠNVỊ.MãsốĐV
```

Truy vấn này đòi hỏi tên của đơn vị và tên của các nhân viên có địa chỉ tại Hà nội.

Vídụ 43 Chúng ta hãy xét một truy vấn phức tạp hơn được sử dụng để định nghĩa một khung nhìn.

```
CREATE VIEW NVĐV AS
```

SELECT TênĐV, Tên

FROM NHÂNVIÊN, ĐƠNVỊ

WHERE NHÂNVIÊN.MãsốĐV = ĐƠNVỊ.MãsốĐV;

Chúng ta có thể truy vấn khung nhìn này như thể nó là một quan hệ được lưu trữ, ví dụ:

SELECT Tên

FROM NVĐV

WHERE Tên = ‘Thanh’;

Truy vấn ở trên tương đương với truy vấn:

SELECT Tên

FROM NHÂNVIÊN, ĐƠNVỊ

WHERE (NHÂNVIÊN.MãsốĐV = ĐƠNVỊ.MãsốĐV)

AND (Tên = ‘Thanh’);

1.2- Ngôn ngữ truy vấn dữ liệu (DQL-Data Query Language)

Bao gồm các lệnh cho phép truy vấn dữ liệu mà không làm thay đổi dữ liệu hoặc các đối tượng trong CSDL. Đó là các truy vấn bắt đầu bằng từ khóa *SELECT*. Trả về một bộ các thuộc tính hoặc một tập hợp các bộ thuộc tính.

1.2.1- Cú pháp câu lệnh SELECT

```
SELECT [DISTINCT] Column(s)
FROM TableName, Views
[WHERE Conditions]
[GROUP BY Row(s)]
[HAVING]
[ORDER BY Column(s) [asc|desc]]
```

Lưu ý:

- Các mệnh đề trong cặp dấu [] không bắt buộc
- **DISTINCT** có thể là: **Distinct**: trả về các bản ghi không trùng lặp nhau hoặc **Top**: trả về n (hay %) bản ghi tìm thấy từ trên xuống
- Mệnh đề **WHERE** cho phép truy vấn lựa chọn theo hàng
- Mệnh đề **GROUP BY** cho phép nhóm dữ liệu theo hàng

- Mệnh đề **HAVING** cho phép truy vấn lựa chọn theo nhóm
- Mệnh đề **ORDER BY** cho phép sắp xếp dữ liệu theo cột

Truy vấn lựa chọn tất cả các hàng và cột

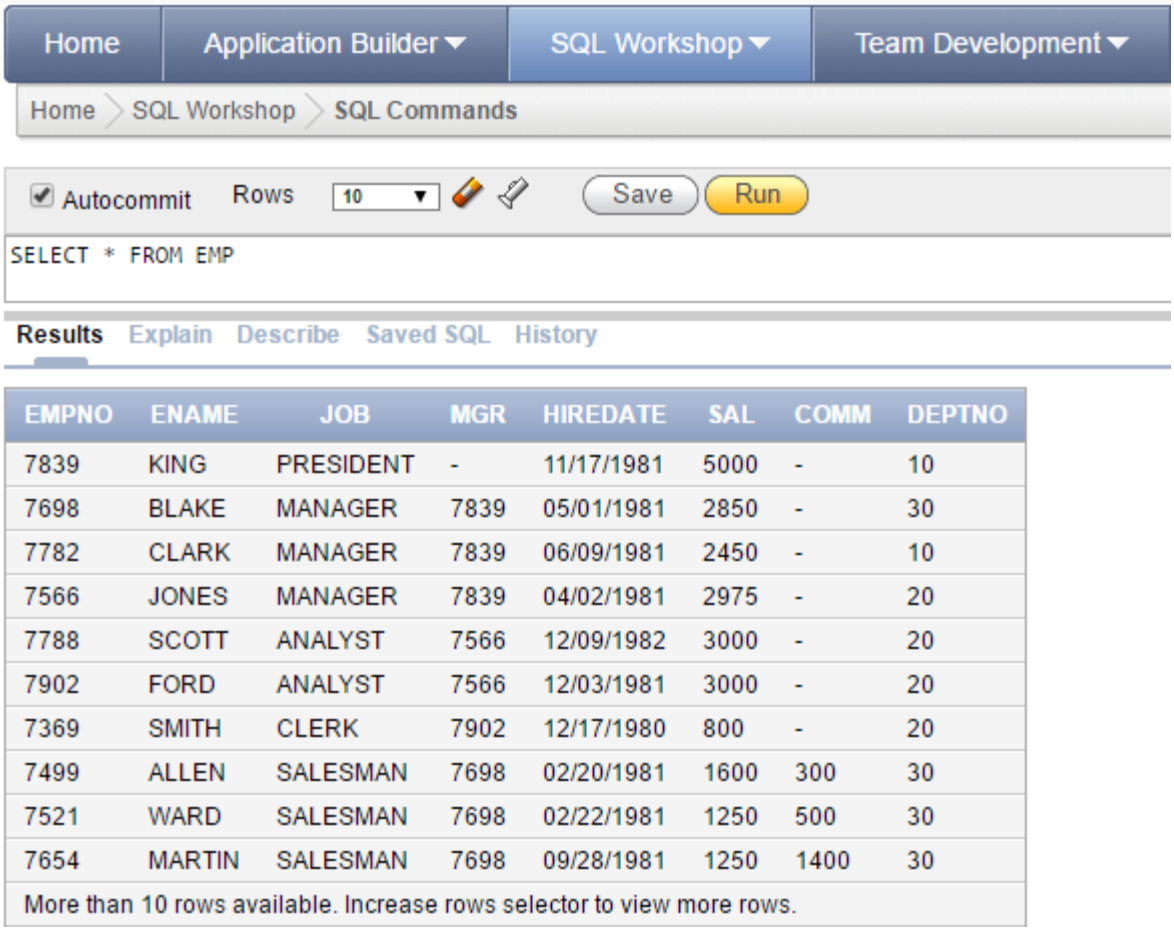
```
SELECT * FROM  TableName
```

Truy vấn lựa chọn một số cột

```
SELECT  Column1, Column2 … FROM  TableName
```

Ví dụ truy vấn lựa chọn tất cả các cột của bảng EMP

```
SELECT * FROM EMP
```



EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT	-	11/17/1981	5000	-	10
7698	BLAKE	MANAGER	7839	05/01/1981	2850	-	30
7782	CLARK	MANAGER	7839	06/09/1981	2450	-	10
7566	JONES	MANAGER	7839	04/02/1981	2975	-	20
7788	SCOTT	ANALYST	7566	12/09/1982	3000	-	20
7902	FORD	ANALYST	7566	12/03/1981	3000	-	20
7369	SMITH	CLERK	7902	12/17/1980	800	-	20
7499	ALLEN	SALESMAN	7698	02/20/1981	1600	300	30
7521	WARD	SALESMAN	7698	02/22/1981	1250	500	30
7654	MARTIN	SALESMAN	7698	09/28/1981	1250	1400	30

More than 10 rows available. Increase rows selector to view more rows.

1.2.2- Mệnh đề WHERE

```
SELECT [DISTINCT] Column(s)
FROM TableName
WHERE  Conditions
```

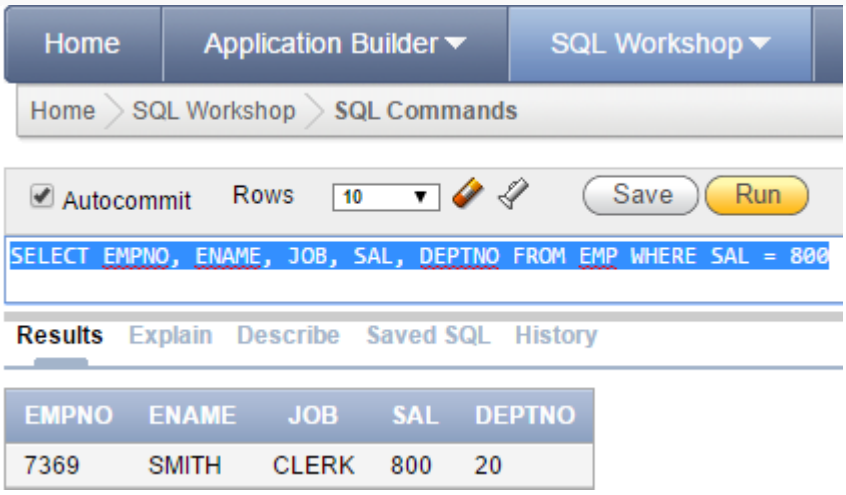
Một số toán tử (Operator) sử dụng trong biểu thức Conditions: Toán tử so sánh, toán tử lôgíc và so sánh xâu dùng toán tử LIKE

1.2.3- *Các toán tử so sánh*

=	So sánh bằng
<> hoặc !=	Khác
>	Lớn hơn
<	Nhỏ hơn
>=	Lớn hơn hoặc bằng
<=	Nhỏ hơn hoặc bằng
BETWEEN value1 AND value2	So sánh nằm trong khoảng value1 và value2
LIKE	So sánh chuỗi tương đối

Ví dụ hiển thị thông tin nhân viên có lương bằng 800

```
SELECT EMPNO, ENAME, JOB, SAL, DEPTNO FROM EMP WHERE SAL = 800
```



1.2.4- *Toán tử LIKE*

Cho phép so sánh một chuỗi với chuỗi khác (có chứa các ký tự đại diện) sử dụng toán tử LIKE. Các ký tự đại diện (Wildcard):

Ký tự	Mô tả
_	Thay cho một ký tự đơn
%	Thay cho một chuỗi
[]	Thay cho một ký tự đơn trong khoảng được bao bởi cặp dấu ngoặc

	vuông
[^]	Thay cho một ký tự đơn bất kỳ không nằm trong khoảng được bao bởi cặp dấu ngoặc vuông

Ví dụ sử dụng ký tự _

```
SELECT * FROM EMP WHERE ENAME LIKE 'A____'
```

Results Explain Describe Saved SQL History

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	7698	02/20/1981	1600	300	30
7876	ADAMS	CLERK	7788	01/12/1983	1100	-	20

Ví dụ sử dụng ký tự %

```
SELECT * FROM EMP WHERE JOB LIKE '%LE%'
```

Results Explain Describe Saved SQL History

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	12/17/1980	800	-	20
7499	ALLEN	SALESMAN	7698	02/20/1981	1600	300	30
7521	WARD	SALESMAN	7698	02/22/1981	1250	500	30
7654	MARTIN	SALESMAN	7698	09/28/1981	1250	1400	30
7844	TURNER	SALESMAN	7698	09/08/1981	1500	0	30
7876	ADAMS	CLERK	7788	01/12/1983	1100	-	20
7900	JAMES	CLERK	7698	12/03/1981	950	-	30
7934	MILLER	CLERK	7782	01/23/1982	1300	-	10

8 rows returned in 0.01 seconds Download

1.2.5- Mệnh đề ORDER BY

Cho phép sắp xếp kết quả truy vấn theo cột và có thể sắp xếp kết quả theo chiều: Tăng dần (asc) hoặc giảm dần (desc). Bên dưới là cú pháp

```
SELECT [DISTINCT] Column(s)
FROM TableName
[WHERE  Conditions ]
ORDER BY Column(s) [asc|desc]
```

Ví dụ dùng toán tử BETWEEN…AND và mệnh đề ORDER BY

```
SELECT * FROM EMP
WHERE SAL BETWEEN 3000 AND 8000
ORDER BY SAL DESC
```

Results Explain Describe Saved SQL History

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT	-	11/17/1981	5000	-	10
7902	FORD	ANALYST	7566	12/03/1981	3000	-	20
7788	SCOTT	ANALYST	7566	12/09/1982	3000	-	20

3 rows returned in 0.00 seconds Download

1.2.6- Toán tử lôgíc

Các toán tử lôgíc gồm AND, OR và NOT. AND và OR được sử dụng để kết nối các điều kiện tìm kiếm chỉ ra trong mệnh đề WHERE. NOT phủ định kết quả tìm kiếm.

Ví dụ sử dụng toán tử AND

```
SELECT * FROM EMP
WHERE SAL >= 3000 AND JOB = 'ANALYST'
```

Results Explain Describe Saved SQL History

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7788	SCOTT	ANALYST	7566	12/09/1982	3000	-	20
7902	FORD	ANALYST	7566	12/03/1981	3000	-	20

2 rows returned in 0.00 seconds Download

1.2.7- Mệnh đề GROUP BY

Mệnh đề GROUP BY cho phép nhóm các hàng dữ liệu có giá trị giống nhau thành một nhóm. Các tính toán (thường sử dụng các hàm truy vấn nhóm) sẽ được tính trên mỗi nhóm. Một số hàm nhóm như:

Min(column)	Tìm giá trị nhỏ nhất trong cột column
Max(column)	Tìm giá trị lớn nhất trong cột column
Avg(column)	Tìm giá trị trung bình của cột column
Count (*)	Đếm số dòng

Ví dụ 1: Tìm mức lương lớn nhất, nhỏ nhất, lương trung bình và tổng lương trong bảng EMP

```
SELECT MAX(SAL), MIN(SAL), AVG(SAL), SUM(SAL) FROM EMP
```

Results Explain Describe Saved SQL History

MAX(SAL)	MIN(SAL)	AVG(SAL)	SUM(SAL)
5000	800	2073.214285714285714285714285714285714429	29025

1 rows returned in 0.00 seconds Download

Ví dụ 2: Hiển thị lương lớn nhất và nhỏ nhất trong mỗi phòng ban

```
SELECT DEPTNO, MAX(SAL), MIN(SAL)
FROM EMP
GROUP BY DEPTNO
```

Results Explain Describe Saved SQL History

DEPTNO	MAX(SAL)	MIN(SAL)
30	2850	950
20	3000	800
10	5000	1300

3 rows returned in 0.00 seconds Download

1.2.8- Mệnh đề HAVING

Mệnh đề HAVING được sử dụng làm điều kiện nhóm. Vì vậy muốn sử dụng HAVING, chúng ta phải kết hợp với GROUP BY.

Ví dụ: Hiển thị lương nhỏ nhất trong mỗi phòng ban với điều kiện lương nhỏ nhất từ 900 trở lên

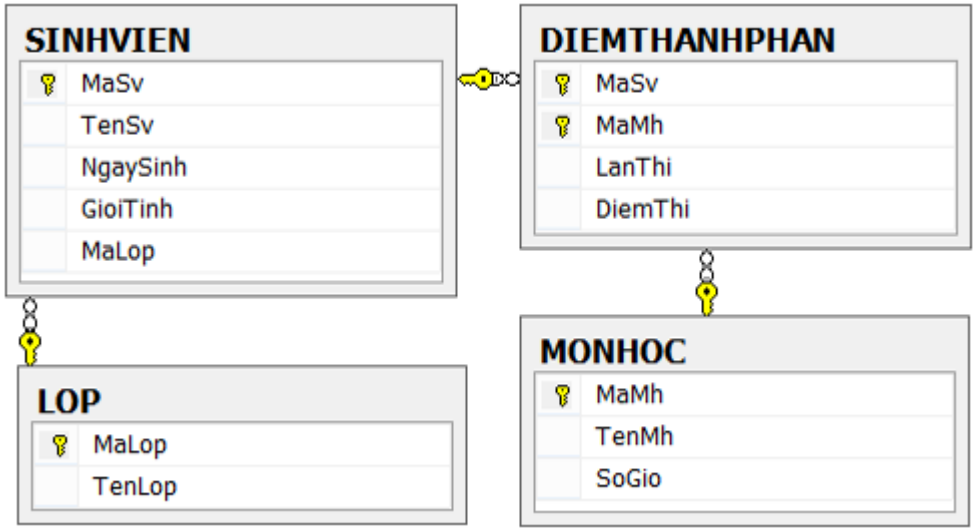
```
SELECT DEPTNO, MAX(SAL), MIN(SAL)
FROM EMP
GROUP BY DEPTNO
HAVING MIN(SAL) >= 900
```

Results Explain Describe Saved SQL History

DEPTNO	MAX(SAL)	MIN(SAL)
30	2850	950
10	5000	1300

2 rows returned in 0.01 seconds Download

Truy vấn dữ liệu từ nhiều bảng



Yêu cầu: Hãy hiển thị thông tin gồm **mã sinh viên, tên sinh viên, tên lợp, mã môn học** và **điểm thi** của những sinh viên có điểm thi >= 5

Mệnh đề SELECT trên nhiều bảng

Khi truy vấn trên nhiều bảng, phải kết nối các bảng. Có hai kiểu kết nối:

- Kết nối trong: mệnh đề WHERE chỉ ra các trường khóa của các bảng cần kết nối phải như nhau hoặc dùng từ khóa JOIN trong mệnh đề FROM
- Kết nối ngoài: sử dụng từ khóa LEFT/RIGHT OUTER JOIN trong mệnh đề FROM

Ví dụ về kết nối trong sử dụng mệnh đề WHERE

Autocommit Display 10

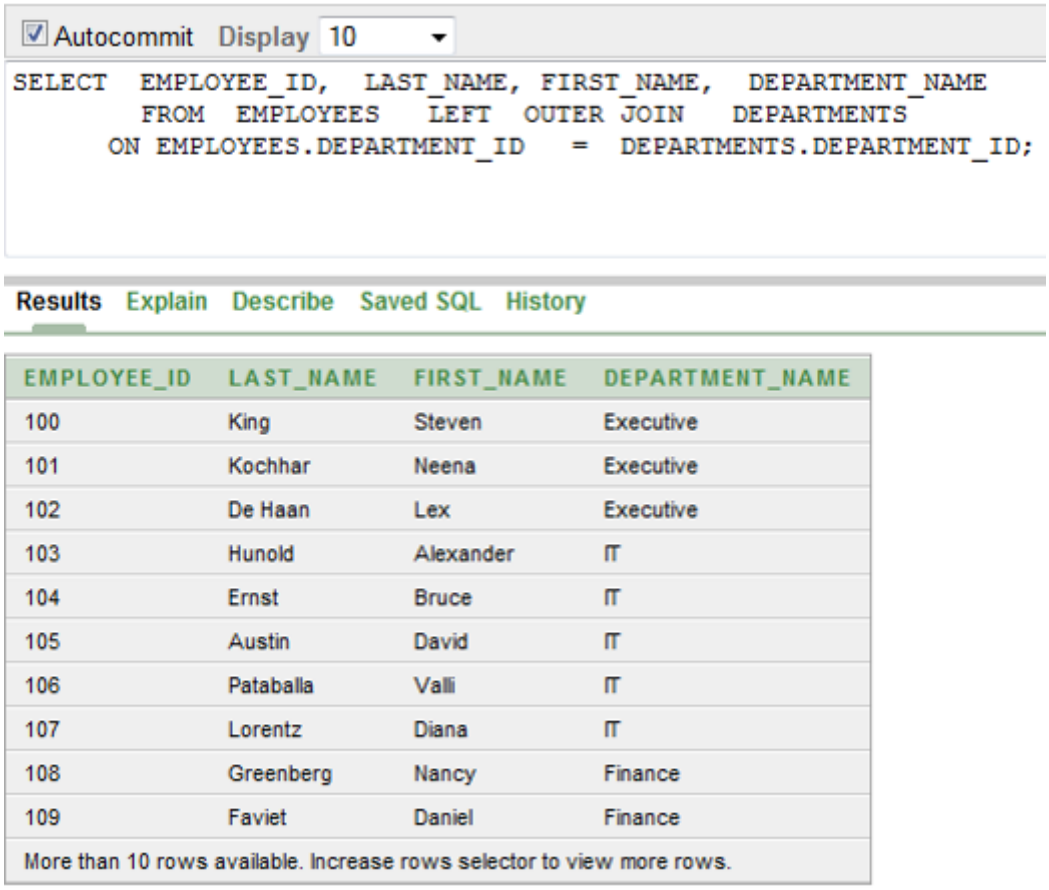
```
SELECT  EMPLOYEE_ID,  LAST_NAME, FIRST_NAME,  DEPARTMENT_NAME
      FROM  EMPLOYEES, DEPARTMENTS
      WHERE EMPLOYEES.DEPARTMENT_ID   =  DEPARTMENTS.DEPARTMENT_ID;
```

Results Explain Describe Saved SQL History

EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_NAME
100	King	Steven	Executive
101	Kochhar	Neena	Executive
102	De Haan	Lex	Executive
103	Hunold	Alexander	IT
104	Ernst	Bruce	IT
105	Austin	David	IT
106	Pataballa	Valli	IT
107	Lorentz	Diana	IT
108	Greenberg	Nancy	Finance
109	Faviet	Daniel	Finance

More than 10 rows available. Increase rows selector to view more rows.

Ví dụ về kết nối ngoài



So sánh kết quả hai kiểu kết nối

Kết nối trong chỉ trả về kết quả khi tìm được DEPARTMENT_ID tương ứng của nhân viên trong bảng DEPARTMENT

Kết nối ngoài trả về kết quả ngay cả khi không tìm được DEPARTMENT_ID của nhân viên trong bảng DEPARTMENT

2. Bài tập

Yêu cầu thực hiện

Cho cơ sở dữ liệu QuanLyNhanSu như sau

NHANVIEN	HONV	TENLOT	TENNV	PHAI	LUONG	MANV	NGSINH	DIACHI	PHG
	Đinh	Lê	Tiên	Nam	4000	123456789	01/09/1965	Nguyễn Trãi, Q5	1
	Nguyễn	Thị	Loan	Nữ	2500	333445555	12/08/1955	Nguyễn Huệ, Q1	5
	Nguyễn	Lan	Anh	Nữ	4300	666884444	15/09/1962	Lê Lợi, Q1	5
	Trần	Thanh	Tâm	Nam	3800	453453453	31/07/1972	Trần Não, Q2	2

DEAN	TENDA	MADA	DDIEM_DA	PHG
	Sản phẩm X	1	Quy Nhơn	5
	Sản phẩm Y	2	Nha Trang	5
	Sản phẩmZ	3	TP HCM	5
	Tin học hoá	10	Bình Dương	4

PHONGBAN	PHG	TENPHG
	1	Nhân sự
	2	Kế hoạch
	3	Kinh doanh
	4	Marketing
	5	Kế toán

PHANCONG	MANV	MADA	SOGIO
	123456789	1	32.0
	123456789	2	8.0
	666884444	3	40.0
	453453453	1	20.0

2.1- Truy vấn dữ liệu trong sql

Câu 1: Tạo cơ sở dữ liệu trên

Câu 2: Truy vấn dữ liệu trong sql

1. Hiển thị tất cả thông tin của bảng NHANVIEN
2. Hiển thị thông tin của những nhân viên ở phòng số 5
3. Hiển thị mã nhân viên, họ nhân viên, tên lót và tên nhân viên của những nhân viên ở phòng số 5 và có lương >= 3000
4. Hiển thị mã nhân viên, tên nhân viên của những nhân viên có lương từ 2000 đến 8000
5. Hiển thị thông tin của những nhân viên ở địa chỉ có tên đường là Nguyễn
6. Cho biết số lượng nhân viên
7. Cho biết số lượng nhân viên trong từng phòng ban
8. Hiển thị thông tin về mã nhân viên, tên nhân viên và tên phòng ban ở phòng kế toán
9. Tổng kết bài học

Ngôn ngữ truy vấn SQL bao gồm các lệnh có cấu trúc cho phép Tạo CSDL và Thêm, Sửa, hoặc Xóa dữ liệu.

Có 4 nhóm lệnh SQL chính, trong đó nhóm lệnh truy vấn dữ liệu (Select) cho phép trích ra phần dữ liệu cần lấy mà không làm thay đổi dữ liệu.

2.2- Bài tập thực hành lệnh cập nhật dữ liệu

Truy vấn cơ sở dữ liệu quản lý tài khoản ngân hàng đã thiết kế ở bài thực hành số 2

Câu 1: Tạo các truy vấn dữ liệu trong sql sử dụng trên một bảng

1.1. Tạo truy vấn hiển thị thông tin tất cả các tài khoản có kiểu là “Checking”

1.2. Tạo truy vấn hiển thị các thông tin gồm (MaKH, SoTK, KieuTK, NgayMoTK) của các tài khoản có kiểu “Tài khoản cá nhân trong nước” và sắp xếp kết quả hiển thị sao cho ngày mở gần nhất sẽ được hiển thị trước (tức là ngày mở được sắp xếp giảm dần)

1.3. Tạo truy vấn hiển thị Tên, địa chỉ, thành phố các khách hàng sống tại thành phố “Hà Nội”. Sắp xếp kết quả theo thứ tự Alphabet của tên khách hàng

1.4. Tạo truy vấn hiển thị thông tin các khách hàng đã mở tài khoản trước ngày01/07/2011.

Câu 2: Tạo các truy vấn dữ liệu trong sql sử dụng trên nhiều bảng

2.1. Thực hiện truy vấn hiển thị thông tin khách hàng, kiểu tài khoản, ngày mở.

2.2. Thực hiện truy vấn hiển thị tất cả các thông tin như truy vấn trên và thỏa mãn điều kiện ngày mở sau ngày 01/07/2011.

2.3. Thực hiện truy vấn hiển thị tất cả các thông tin như truy vấn trên và thỏa mãn điều kiện ngày mở sau ngày 01/07/2011 và kiểu tài khoản là “Tài khoản tổ chức trong nước”. Thông tin hiển thị được sắp xếp theo thứ tự Alphabet của tên khách hàng.

2.4. Thực hiện truy vấn hiển thị các thông tin: SoTK,KieuTK,SoTienGD,

MoTaGD,ThoiGianGD,SoDuTaiKhoan cuả tài khoản 000000003 và giao dịch

ngày 20/07/2007.

2.5. Liệt kê các thông tin về tất cả các lần giao dịch gồm: HoTenKH, DiaChi, SoTK, KieuTK, ThoiGianGD, SoTienGD, MoTaGD, SoDuTaiKhoan của khách hàng có số tài khoản là: 500000

CHƯƠNG 6- RÀNG BUỘC TOÀN VẸN QUAN HỆ

(Entegrity constraint)

1. Ràng buộc toàn vẹn - các yếu tố của ràng buộc toàn vẹn

1.1- Ràng Buộc Toàn Vẹn

Trong mỗi CSDL luôn tồn tại nhiều mối liên hệ giữa các thuộc tính, giữa các bộ. Sự liên hệ này có thể xảy ra trong một lược đồ quan hệ hoặc trong các lược đồ quan hệ của một cơ sở dữ liệu. Các mối liên hệ này là những điều kiện bất biến mà tất cả các bộ của những quan hệ có liên quan trong CSDL đều phải thỏa mãn ở mọi thời điểm. Những điều kiện bất biến đó được gọi là ràng buộc toàn vẹn. Trong thực tế ràng buộc toàn vẹn là các quy tắc quản lý được áp đặt trên các đối tượng của thế giới thực.

Nhiệm vụ của người phân tích thiết kế là phải phát hiện càng đầy đủ và chính xác các ràng buộc toàn vẹn càng tốt và mô tả chúng một cách chính xác trong hồ sơ phân tích thiết kế - đó là một việc làm rất quan trọng và rất cần thiết.

Trong một cơ sở dữ liệu, ràng buộc toàn vẹn được xem như là một công cụ để diễn đạt ngữ nghĩa của CSDL. Một CSDL được thiết kế cồng kềnh nhưng nó thể hiện được đầy đủ ngữ nghĩa của thực tế vẫn có giá trị cao hơn rất nhiều so với một cách thiết kế gọn nhẹ nhưng nghèo nàn về ngữ nghĩa vì thiếu các ràng buộc toàn vẹn của cơ sở dữ liệu.

Công việc kiểm tra ràng buộc toàn vẹn thường được tiến hành vào thời điểm cập nhật dữ liệu (thêm, sửa, xóa). Những ràng buộc toàn vẹn phát sinh cần phải được ghi nhận và xử lý một cách tường minh (thường là bởi một hàm chuẩn hoặc một đoạn chương trình).

1.2- Các Yếu Tố Của Ràng Buộc Toàn Vẹn

Mỗi ràng buộc toàn vẹn có 3 yếu tố: điều kiện, bối cảnh và tầm ảnh hưởng.

1.2.1- Điều kiện

Điều kiện của một ràng buộc toàn vẹn R có thể được biểu diễn bằng ngôn ngữ tự nhiên, thuật giải, ngôn ngữ đại số tập hợp, đại số quan hệ,… ngoài ra điều kiện

của ràng buộc toàn vẹn cũng có thể được biểu diễn bằng phụ thuộc hàm. Chẳng hạn, với lược đồ quan hệ SV thì có một ràng buộc toàn vẹn như sau:

Với r là một quan hệ của Sv ta có ràng buộc toàn vẹn sau

$\forall$ $t_1, t_2 \in r$

$t_1.MASV \neq t_2.MASV$

cuối $\forall$

1.2.2- Bối cảnh

Bối cảnh của một ràng buộc toàn vẹn là những quan hệ mà ràng buộc đó có hiệu lực hay nói một cách khác, đó là những quan hệ cần phải được kiểm tra ràng buộc toàn vẹn. Bối cảnh của một ràng buộc toàn vẹn có thể là một hoặc nhiều quan hệ. Chẳng hạn với ràng buộc toàn vẹn trên thì bối cảnh là một quan hệ Sv

1.2.3- Tầm ảnh hưởng

Trong quá trình phân tích thiết kế một CSDL, người phân tích cần lập bảng tầm ảnh hưởng cho một ràng buộc toàn vẹn nhằm xác định thời điểm cần phải tiến hành kiểm tra các ràng buộc toàn vẹn đó. Các thời điểm cần phải kiểm tra RBTV chính là những thời điểm cập nhật dữ liệu (thêm /sửa/ xóa)

Một bảng tâm ảnh hưởng của một RBTV có dạng sau:

(Tên RBTV)	Thêm(T)	Sửa(S)	Xóa(X)
r_1	+	-	-
r_2			
...	...	..	..
...	...	...	...
r_n			

Bảng này chứa toàn các ký hiệu + hoặc –

Chẳng hạn + tại ô tương ứng với dòng r_1, cột thêm thì có nghĩa là khi thêm một bộ vào quan hệ r_1 thì cần phải kiểm tra RBTV

Dấu - Tại ô tương ứng với dòng r_1, cột sửa thì có nghĩa là khi sửa một bộ trên quan hệ r_1 thì không cần phải kiểm tra RBTV này,…

2. PHÂN LOẠI RÀNG BUỘC TOÀN VẸN

Trong quá trình phân tích thiết kế cơ sở dữ liệu, người phân tích phải phát hiện tất cả các ràng buộc toàn vẹn tiềm ẩn trong CSDL đó. Việc phân loại các ràng buộc toàn vẹn là rất có ích, nó nhằm giúp cho người phân tích có được một định hướng, tránh bỏ sót những ràng buộc toàn vẹn. Các ràng buộc toàn vẹn có thể được chia làm hai loại chính như sau:

+ Ràng buộc toàn vẹn trên phạm vi là một quan hệ bao gồm :Ràng buộc toàn vẹn miền giá trị, ràng buộc toàn vẹn liên thuộc tính, ràng buộc toàn vẹn liên bộ.
+ Ràng buộc toàn vẹn trên phạm vi nhiều quan hệ bao gồm :Ràng buộc toàn vẹn phụ thuộc tồn tại, ràng buộc toàn vẹn liên bộ - liên quan hệ, ràng buộc toàn vẹn liên thuộc tính - liên quan hệ.

Để minh họa cho phần lý thuyết của chương này, ta nêu ví dụ sau đây

Ví dụ

Cho một CSDL C dùng để quản lý việc đặt hàng và giao hàng của một công ty. Lược đồ CSDL C gồm các lược đồ quan hệ như sau:

Q_1: Khach (MAKH,TENKH,DCKH,DT)

Tân từ: Mỗi khách hàng có một mã khách hàng (MAKH) duy nhất, mỗi MAKH xác định một tên khách hàng (TENKH), một địa chỉ (DCKH), một số điện thoại (DT).

Q_2: Hang(MAHANG,TENHANG,QUYCACH,DVTINH)

Tân từ: Mỗi mặt hàng có một mã hàng (MAHANG) duy nhất, mỗi MAHANG xác định một tên hàng (TENHANG), quy cách hàng (QUYCACH), đơn vị tính (DVTINH).

Q_3: Dathang(SODH,MAHANG,SLDAT,NGAYDH,MAKH)

Tân từ: Mỗi lần đặt hàng có số đặt hàng (SODH) xác định một ngày đặt hàng (NGAYDH) và mã khách hàng tương ứng (MAKH). Biết mã số đặt hàng

và mã mặt hàng thì biết được số lượng đặt hàng(SLDAT). Mõi khách hàng trong một ngày có thể có nhiều lần đặt hàng

Q_4: Hoadon(SOHD, NGAYLAP, SODH, TRIGIAHD, NGAYXUAT)

Tân từ: Mỗi hóa đơn có một mã số duy nhất là SOHD, mỗi hóa đơn bán hàng có thể gồm nhiều mặt hàng. Mỗi hóa đơn xác định ngày lập hóa đơn (NGAYLAP), ứng với số đặt hàng nào (SODH). Giả sử rằng hóa đơn bán hàng theo yêu cầu của chỉ một đơn đặt hàng có mã số là SODH và ngược lại, mỗi đơn đặt hàng chỉ được giải quyết chỉ trong một hóa đơn. ***Do điều kiện khách quan có thể công ty không giao đầy đủ các mặt hàng cũng như số lượng từng mặt hàng như yêu cầu trong đơn đặt hàng nhưng không bao giờ giao vượt ngoài yêu cầu***. Mỗi hóa đơn xác định một trị giá của các mặt hàng trong hóa đơn (TRIGIAHD) và một ngày xuất kho giao hàng cho khách (NGAYXUAT)

Q_5: Chitiethd (SOHD, MAHANG, GIABAN, SLBAN)

Tân từ: Mỗi SOHD, MAHANG xác định giá bán (GIABAN) và số lượng bán (SLBAN) của một mặt hàng trong một hóa đơn.

Q_6: Phieuthu(SOPT, NGAYTHU, MAKH, SOTIEN)

Tân tư: Mỗi phiếu thu có một số phiếu thu (SOPT) duy nhất, mỗi SOPT xác định một ngày thu (NGAYTHU) của một khách hàng có mã khách hàng là MAKH và số tiền thu là SOTIEN. Mỗi khách hàng trong một ngày có thể có nhiều số phiếu thu.

2.1- Ràng buộc toàn vẹn liên bộ

Ràng buộc toàn vẹn liên bộ là sự ràng buộc toàn vẹn giữa các bộ trong cùng một quan hệ .

Ràng buộc toàn vẹn liên bộ hay còn gọi là ràng buộc toàn vẹn về khóa. Đây là loại ràng buộc toàn vẹn rất phổ biến, nó có mặt trong mọi lược đồ quan hệ của CSDL và thường được các hệ quản trị CSDL tự động kiểm tra.

Ví dụ: Với r là một quan hệ của Khach ta có ràng buộc toàn vẹn sau

R_1: $\forall\ t_1, t_2 \in r$

t_1. MAKH ≠ t_2. MAKH

Cuối ∀

R_1	Thêm	Sửa	Xóa
r	+	+	-

2.2- Ràng buộc toàn vẹn về phụ thuộc tồn tại:

Ràng buộc toàn vẹn về phụ thuộc tồn tại còn được gọi là ràng buộc toàn vẹn về khóa ngoại. Cũng giống như ràng buộc toàn vẹn về khóa chính, ràng buộc toàn vẹn về phụ thuộc tồn tại rất phổ biến trong CSDL

Ví dụ: Với r, s lần lượt là một quan hệ của Dathang, Khach ta có ràng buộc toàn vẹn sau

R_2: r[MAKH] ⊆ s[MAKH]

R_2	Thêm	Sửa	Xóa
r	+	+	−
s	−	+	+

2.3- Ràng buộc toàn vẹn về miền giá trị

Ràng buộc toàn vẹn có liên quan đến miền giá trị của các thuộc tính trong một quan hệ. Ràng buộc này thường gặp. Một số hệ quản trị CSDL đã tự động kiểm tra một số ràng buộc loại này.

Ví dụ: Với r là một quan hệ của Hoadon ta có ràng buộc toàn vẹn sau

R_3: ∀ t ∈ r

t.TRIGIAHD > 0

Cuối ∀

R_3	Thêm	Sửa	Xóa
r	+	+	-

2.4- Ràng buộc toàn vẹn liên thuộc tính

Ràng buộc toàn vẹn liên thuộc tính là mối liên hệ giữa các thuộc tính trong một lược đồ quan hệ.

Ví dụ: Với r là một quan hệ của Hoadon ta có ràng buộc toàn vẹn sau

R_4: ∀ t ∈ r

t.NGAYLAP <= t.NGAYXUAT

Cuối ∀

R_4	Thêm	Sửa	Xóa
r	+	+	-

2.5- Ràng buộc toàn vẹn liên thuộc tính liên quan hệ

Ràng buộc loại này là mối liên hệ giữa các thuộc tính trong nhiều lược đồ quan hệ.

Ví dụ: Với r, s lần lượt là quan hệ của Dathang, Hoadon ta có ràng buộc toàn vẹn sau

R_5: ∀ t_1 ∈ r, t_2 ∈ s

Nếu t_1.SODH = t_2.SODH thì

t_1.NGAYDH <= t_2.NGAYXUAT

Cuối ∀

R_5	Thêm	Sửa	Xóa
r	+	+	+
s	+	+	+

2.6- Ràng buộc toàn vẹn về thuộc tính tổng hợp

Ràng buộc toàn vẹn về thuộc tính tổng hợp được xác định trong trường hợp mỗi thuộc tính A của một lược đồ quan hệ Q được tính toán giá trị từ các thuộc tính của các lược đồ quan hệ khác.

3. Bài tập

1/ Hãy tìm các ràng buộc toàn vẹn có trong CSDL cho các bài tập được liệt kê trong chương 3.

2/ QUẢN LÝ THI TỐT NGHIỆP PTCS

Một phòng giáo dục huyện muốn lập một hệ thống thông tin để quản lý việc làm thi tốt nghiệp phổ thông cơ sở. Công việc làm thi được tổ chức như sau:

Lãnh đạo phòng giáo dục thành lập nhiều hội đồng thi (mỗi hội đồng thi gồm một trường hoặc một số trường gần nhau). Mỗi hội đồng thi có một mã số duy nhất (MAHĐT), một mã số hội đồng thi xác định tên hội đồng thi(TENHĐT), họ tên chủ tịch hội đồng(TENCT), địa chỉ (ĐCHĐT),điện thoại(ĐTHĐT).

Mỗi hội đồng thi được bố trí cho một số phòng thi, mỗi phòng thi có một số hiệu phòng(SOPT) duy nhất, một phòng thi xác định địa chỉ phòng thi (ĐCPT). Số hiệu phòng thi được đánh số khác nhau ở tất cả các hội đồng thi.

Giáo viên của các trường trực thuộc phòng được điều động đến các hội đồng để coi thi, mỗi trường có thể có hoặc không có thí sinh dự thi, mỗi trường có một mã trường duy nhất (MATR), mỗi mã trường xác định một tên trường(TENTR),địa chỉ (ĐCTR), loại hình đào tạo (LHĐT) (Công lập, chuyên, bán công, dân lập, nội trú,…). Giáo viên của một trường có thể làm việc tại nhiều hội đồng thi. Một giáo viên có một mã giáo viên(MAGV), một mã giáo viên xác định tên giáo viên (TENGV), chuyên môn giảng dạy (CHUYENMON), chức danh trong hội đồng thi(CHUCDANH)

Các thí sinh dự thi có một số báo danh duy nhất(SOBD), mỗi số báo danh xác định tên thí sinh(TENTS), ngày sinh (NGSINH), giới tính (PHAI), mỗi thí sinh được xếp thi tại một phòng thi nhất định cho tất cả các môn, mỗi thí sinh có thể có chứng chỉ nghề (CCNGHE) hoặc không (thuộc tính CCNGHE kiểu chuỗi, CCNGHE=”x” nếu thí sinh có chứng chỉ nghề và CCNGHE bằng rỗng nếu thí sinh không có chứng chỉ nghề).Thí sinh của cùng một trường chỉ dự thi tại một hội đồng thi.

Mỗi môn thi có một mã môn thi duy nhất(MAMT), mỗi mã môn thi xác định tên môn thi(TENMT). Giả sử toàn bộ các thí sinh đều thi chung một số môn do sở giáo dục quy định. Mỗi môn thi được tổ chức trong một buổi của một ngày nào đó.

Ứng với mỗi môn thi một thí sinh có một điểm thi duy nhất(ĐIEMTHI)

Dựa vào phân tích ở trên, giả sử ta có lược đồ CSDL sau:

Q_1: HĐ(MAHĐT,TENHĐT, TENCT, ĐCHĐT,ĐTHĐT)

Q_2: PT(SOPT,ĐCPT,MAHĐT)

Q_3: TS(SOBD, TENTS,NGSINH,PHAI,CCNGHE, MATR,SOPT)

Q_4: MT(MAMT,TENMT,BUOI,NGAY)

Q_5: GV(MAGV,TENGV,CHUYENMON,CHUCDANH,MAHĐT,MATR)

Q_6: TR(MATR,TENTR,ĐCTR,LHĐT)

Q_7: KQ(SOBD,MAMT,ĐIEMTHI)

Yêu cầu:

a) Hãy xác định khóa cho từng lược đồ quan hệ.
b) Tìm tất cả các ràng buộc toàn vẹn có trong CSDL trên.
c) Dựa vào lược đồ CSDL đã thành lập, hãy thực hiện các câu hỏi sau đây bằng ngôn ngữ đại số quan hệ.
   1. Danh sách các thí sinh thi tại phòng thi có số hiệu phòng thi (SOPT) là "100". Yêu cầu các thông tin:SOBD,TENTS,NGSINH,TENTR
   2. Kết quả của môn thi có mã môn thi (MAMT) là "T" của tất cả các thí sinh có mã trường(MATR) là "NTMK", kết quả được sắp theo chiều giảm dần của điểm thi(ĐIEMTHI). Yêu cầu các thông tin:SOBD,TENTS, ĐIEMTHI
   3. Kết quả thi của một học sinh có SOBD là MK01. Yêu cầu : TENMT,ĐIEMTHI
   4. Tổng số thí sinh có chứng chỉ nghề(CCNGHE) của mỗi trường, thông tin cần được sắp theo chiều tăng dần của TENTR. Yêu cầu các thông tin: MATR, TENTR, SOLUONGCC

----oOo----

CHƯƠNG 7 - PHỤ THUỘC HÀM VÀ CHUẨN HÓA CƠ SỞ DỮ LIỆU QUAN HỆ, CÁC THUẬT TOÁN THIẾT KẾ CƠ SỞ DỮ LIỆU QUAN HỆ

Trong chương này chúng ta sẽ thảo luận về một số vấn đề lý thuyết đã được phát triển nhằm mục đích chọn được lược đồ quan hệ "tốt", nghĩa là đo đạc một cách hình thức vì sao tập hợp các thuộc tính này nhóm vào trong các lược đồ quan hệ thì tốt hơn nhóm kia. Chúng ta có thể nói đến "tính tốt" của các lược đồ quan hệ ở hai mức: mức thứ nhất là mức lôgic, mức thứ hai là mức cài đặt. Mức thứ nhất liên quan đến việc các người sử dụng thể hiện các lược đồ quan hệ và ý nghĩa của các thuộc tính của chúng như thế nào. Mức thứ hai liên quan đến việc các bộ trong một quan hệ cơ sở được lưu trữ và cập nhật như thế nào.

Việc thiết kế cơ sở dữ liệu có thể được thực hiện bằng cách sử dụng hai giải pháp: *dưới lên* (bottom-up) hoặc *trên xuống* (top-down). Phương pháp thiết kế dưới lên xem các mối liên kết cơ bản giữa các thuộc tính riêng rẽ như là điểm xuất phát và sử dụng chúng để xây dựng nên các quan hệ. Giải pháp này còn có tên gọi là *thiết kế bằng tổng hợp* (design by synthesis). Ngược lại, phương pháp thiết kế trên xuống, còn gọi là *thiết kế bằng phân tích* (design by analyse) bắt đầu từ một số các nhóm thuộc tính trong các quan hệ nhận được từ thiết kế quan niệm và các hoạt động chuyển đổi. Sau đó việc *thiết kế bằng phân tích* được áp dụng đối với các quan hệ một cách riêng rẽ và tập thể dẫn đến việc tách các quan hệ cho đến khi đạt được tính chất mong muốn.

1. Các nguyên tắc thiết kế lược đồ quan hệ

1.1- Ngữ nghĩa của các thuộc tính quan hệ

Khi chúng ta nhóm các thuộc tính để tạo nên một lược đồ quan hệ, ta giả thiết rằng có một ý nghĩa nào đó gắn với các thuộc tính. Ý nghĩa này, còn gọi là *ngữ nghĩa*, chỉ ra việc hiểu các giá trị thuộc tính lưu giữ trong các bộ của một quan hệ như thế nào. Nói cách khác, các giá trị thuộc tính trong một bộ liên hệ với nhau như thế nào. Nếu việc thiết kế khái niệm được làm một cách cẩn thận, sau đó là một chuyển đổi sang các quan hệ thì hầu hết ngữ nghĩa đã được giải thích và thiết kế kết quả có một ý nghĩa rõ ràng. Nói chung, việc giải thích ngữ nghĩa của quan hệ càng dễ dàng thì việc thiết kế lược đồ quan hệ càng tốt. Một ví dụ về thiết kế lược đồ

quan hệ tốt là lược đồ cơ sở dữ liệu "CÔNG TY". Trong lược đồ đó, các thuộc tính đều có ý nghĩa rõ ràng, không có tính mập mờ. Nguyên tắc sau sẽ hỗ trợ cho việc thiết kế lược đồ quan hệ.

Nguyên tắc 1: Thiết kế một lược đồ quan hệ sao cho dễ giải thích ý nghĩa của nó. Đừng tổ hợp các thuộc tính từ nhiều kiểu thực thể và kiểu liên kết vào một quan hệ đơn. Một cách trực quan, nếu một lược đồ quan hệ tương ứng với một kiểu thực thể hoặc một kiểu liên kết thì ý nghĩa trở nên rõ ràng. Ngược lại, một quan hệ tương ứng với một hỗn hợp các thực thể và liên kết thì ý nghĩa trở nên không rõ ràng.

1.2- Thông tin dư thừa trong các bộ và sự dị thường cập nhật

Một mục tiêu của thiết kế lược đồ là làm tối thiểu không gian lưu trữ các quan hệ cơ sở. Các thuộc tính được nhóm vào trong các lược đồ quan hệ có một ảnh hưởng đáng kể đến không gian lưu trữ. Nếu cùng một thông tin được lưu giữ nhiều lần trong cơ sở dữ liệu thì ta gọi đó là dư thừa thông tin và điều đó sẽ làm lãng phí không gian nhớ. Ví dụ, giả sử ta có bảng cơ sở sau đây:

NHÂNVIÊN_ĐƠNVỊ

MãsốNV	Họđệm	Tên	Ngày sinh	Địachỉ	MãsốĐV	TênĐV	MãsốNQL
NV001	Lê	Vân	12/02/79	Hà nội	5	Nghiêncứu	NV002
NV002	Trần Đức	Nam	14/02/66	Hà nội	5	Nghiêncứu	NV002
NV010	Hoàng	Thanh	05/08/79	Nghệ an	4	Hànhchính	NV014
NV014	Phạm	Bằng	26/06/52	Bắc ninh	4	Hànhchính	NV014
NV016	Nguyễn	Sơn	14/08/73	Hà nam	5	Nghiêncứu	NV002
NV018	Vũ Hương	Giang	26/03/83	Nam định	5	Nghiêncứu	NV002
NV025	Trần Lê	Hoa	15/03/80	Phú thọ	4	Hànhchính	NV014
NV061	Hoàng	Giáp	02/05/47	Hà tĩnh	1	Lãnhđạo	NV061

Ở đây có sự dư thừa thông tin. Nếu một đơn vị có nhiều nhân viên làm việc thì thông tin về đơn vị (Mã số, Tên đơn vị, Mã số người quản lý) được lưu giữ nhiều lần trong bảng. So với việc dùng hai bảng NHÂNVIÊN và ĐƠNVỊ riêng rẽ như trong lược đồ "CÔNG TY", việc sử dụng bảng này làm lãng phí không gian nhớ.

Ngoài việc lãng phí không gian nhớ nó còn dẫn đến một vấn đề nghiêm trọng là sự dị thường cập nhật. Dị thường cập nhật bao gồm: Dị thường Chèn, dị thường Xoá, dị thường Sửa đổi. Những dị thường cập nhật này sẽ đưa vào cơ sở dữ liệu những thông tin "lạ" và làm cho cơ sở dữ liệu mất tính đúng đắn.

Dị thường Chèn: Gây ra khó khăn khi chèn các bộ giá trị vào bảng hoặc dẫn đến vi phạm ràng buộc. Ví dụ:

Để chèn một bộ giá trị cho một nhân viên mới vào bảng NHÂNVIÊN_ĐƠNVỊ ngoài các thông tin về nhân viên, ta phải đưa vào các thông tin về đơn vị mà anh ta làm việc hoặc các giá trị null (nếu nhân viên không làm việc cho đơn vị nào cả). Các thông tin về đơn vị phải được đưa vào một cách đúng đắn, phù hợp với các thông tin của đơn vị đó trong các bộ khác. Trong lúc đó, với lược đồ cơ sở dữ liệu "CÔNGTY" chúng ta không phải lo lắng gì, vì các thông tin về một đơn vị chỉ được lưu trữ một lần.

Rất khó chèn một đơn vị mới vào quan hệ NHÂNVIÊN_ĐƠNVỊ nếu đơn vị đó không có nhân viên nào làm việc. Cách giải quyết duy nhất là điền các giá trị null vào các thuộc tính của nhân viên. Điều đó làm nảy sinh vấn đề về ràng buộc bởi vì MãsốNV là khóa chính của quan hệ.

Dị thường Xóa: Gây ra việc mất thông tin khi xóa. Ví dụ khi ta xóa một bộ giá trị trong bảng NHÂNVIÊN_ĐƠNVỊ. Nếu nhân viên tương ứng với bộ giá trị đó là người cuối cùng làm việc cho đơn vị thì phép xóa sẽ kéo theo việc làm mất thông tin về đơn vị.

Dị thường Sửa đổi: Gây ra việc sửa đổi hàng loạt khi ta muốn sửa đổi một giá trị trong một bộ nào đó. Ví dụ, ta muốn sửa giá trị của thuộc tính MãsốNQL của đơn vị 5. Điều đó kéo theo ta phải sửa giá trị của thuộc tính này trong tất cả các bộ ứng với đơn vị 5. Dựa trên các dị thường ở trên, chúng ta có thể phát biểu nguyên tắc sau:

Nguyên tắc 2: Thiết kế các lược đồ quan hệ cơ sở sao cho không sinh ra những dị thường cập nhật trong các quan hệ. Nếu có xuất hiện những dị thường cập nhật thì phải ghi chép lại một cách rõ ràng và phải đảm bảo rằng các chương trình cập nhật dữ liệu sẽ thực hiện một cách đúng đắn.

1.3- Các giá trị không xác định trong các bộ

Trong một số thiết kế lược đồ, chúng ta có thể nhóm nhiều thuộc tính với nhau vào một quan hệ "béo". Nếu nhiều thuộc tính không thích hợp cho mọi bộ trong một quan hệ, chúng ta sẽ kết thúc với nhiều giá trị null trong các bộ đó. Điều đó có thể làm tăng không gian ở mức lưu trữ và có thể dẫn đến vấn đề về hiểu ý nghĩa của các thuộc tính. Việc chỉ ra các phép nối ở mức lôgic cũng sẽ gặp khó khăn. Một vấn đề nữa với các giá trị null là các hàm nhóm như COUNT, SUM không đếm được đối với chúng. Hơn nữa, các giá trị null có thể nhiều cách giải thích, chẳng hạn như thuộc tính không áp dụng được cho bộ này, giá trị của thuộc tính cho bộ này là không có hoặc giá trị cho thuộc tính là có nhưng vắng mặt. Tóm lại, các giá trị null có nhiều ý nghĩa khác nhau .

Nguyên tắc 3: Tránh càng xa càng tốt việc đặt vào trong các quan hệ cơ sở những thuộc tính mà các giá trị của chúng thường xuyên là null. Nếu không thể tránh được các giá trị null thì phải đảm bảo rằng chúng chỉ áp dụng trong các trường hợp đặc biệt và không áp dụng cho một số lớn các bộ trong quan hệ.

1.4- Sinh ra các bộ giả

Nhiều khi chúng ta đưa vào cơ sở dữ liệu những quan hệ không đúng, việc áp dụng các phép toán (nhất là các phép nối) sẽ sinh ra các bộ giá trị không đúng, gọi là các bộ "giả". Ví dụ, xét hai lược đồ quan hệ:

NV_ĐĐ (Tên, ĐịađiểmDA)

NV_DA1(Mã sốNV, Mã sốDA, Sốgiờ, TênDA, ĐịađiểmDA)

NV_ĐĐ	Tên	ĐịađiểmDA
	Vân	Hànội
	Vân	Namđịnh
	Sơn	Bắcninh
	Giang	Hànội

NV_DA1	MãsốNV	MãsốDA	Sốgiờ	TênDA	ĐịađiểmDA
	NV001	1	32	DA01	Hànội
	NV001	2	7	DA02	Namđịnh
	NV016	3	40	DA03	Bắcninh
	NV018	1	20	DA01	Hànội

Bây giờ ta nối tự nhiên hai quan hệ trên với nhau, ta có quan hệ

	MãsốNV	MãsốDA	Sốgiờ	TênDA	Địađiểm	Tên
	NV001	1	32	DA01	Hànội	Vân
*	NV001	1	32	DA01	Hànội	Giang
	NV001	2	7	DA02	Namđịnh	Vân
	NV016	3	40	DA03	Bắcninh	Sơn
*	NV018	1	20	DA01	Hànội	Vân
	NV018	1	20	DA01	Hànội	Giang

Ta thấy các dòng đánh dấu * là các bộ "giả". Đấy là các bộ giá trị không có trên thực tế.

Nguyên tắc 4: Thiết kế các lược đồ quan hệ sao cho chúng có thể được nối với điều kiện bằng trên các thuộc tính là khoá chính hoặc khoá ngoài theo cách đảm bảo không sinh ra các bộ "giả". Đừng có các quan hệ chứa các thuộc tính nối khác với các tổ hợp khoá chính-khoá ngoài. Nếu không tránh được những quan hệ như vậy thì đừng nối chúng trên các thuộc tính đó, bởi vì các phép nối có thể sinh ra các bộ "giả".

2. Các phụ thuộc hàm

Khái niệm cơ bản nhất trong thiết kế lược đồ quan hệ là khái niệm phụ thuộc hàm. Trong phần này chúng ta sẽ định nghĩa hình thức khái niệm này và cách sử dụng nó để định nghĩa các dạng chuẩn cho các lược đồ quan hệ.

2.1- Định nghĩa phụ thuộc hàm

Một phụ thuộc hàm là một ràng buộc giữa hai nhóm thuộc tính của một cơ sở dữ liệu. Giả sử rằng lược đồ cơ sở dữ liệu của ta có n thuộc tính $A_1,A_2,\ldots,A_n$ và ta hãy nghĩ rằng toàn bộ cơ sở dữ liệu được mô tả bằng một lược đồ quan hệ chung $R=\{A_1, A_2,\ldots., A_n\}$. Đừng nên nghĩ rằng cơ sở dữ liệu của chúng ta sẽ được lưu trữ trong một bảng chung, chúng ta chỉ sử dụng khái niệm này để phát triển lý thuyết hình thức về sự phụ thuộc dữ liệu. Giả sử X và Y là hai tập con của R.

Một phụ thuộc hàm, ký hiệu là $X\rightarrow Y$, giữa hai tập thuộc tính X và Y chỉ ra một ràng buộc trên các bộ có thể có tạo nên một trạng thái quan hệ r của R. Ràng buộc đó là: với hai bộ bất kỳ t_1 và t_2 trong r , nếu có $t_1[X] = t_2[X]$ thì cũng phải có $t_1[Y] = t_2[Y]$. Điều đó có nghĩa là các giá trị của thành phần Y của một bộ trong R phụ thuộc vào, hoặc được xác định bởi, các giá trị của thành phần X. Nói cách khác, các giá trị của thành phần X của một bộ xác định một cách duy nhất các giá trị của thành phần Y. Chúng ta cũng nói rằng có một phụ thuộc hàm từ X vào Y hoặc Y phụ thuộc hàm vào X. Phụ thuộc hàm được viết tắt là FD (functional dependency). Tập thuộc tính X được gọi là vế trái của FD, tập thuộc tính Y được gọi là vế phải của FD.

Như vậy, X xác định hàm Y trong lược đồ quan hệ R khi và chỉ khi nếu hai bộ của r(R) bằng nhau trên các giá trị của X thì chúng nhất thiết phải bằng nhau trên các giá trị của Y. Ta có các nhận xét sau:

- Nếu có một ràng buộc trên các trạng thái của R là chỉ có một bộ giá trị duy nhất của X trong mọi thể hiện quan hệ r(R) thì điều đó kéo theo $X \rightarrow Y$ với mọi tập con các thuộc tính Y của R.
- Nếu $X \rightarrow Y$ thì không thể nói gì về $Y \rightarrow X$.

Một phụ thuộc hàm là một tính chất ngữ nghĩa của các thuộc tính. Những người thiết kế cơ sở dữ liệu sẽ dùng hiểu biết của họ về ý nghĩa của các thuộc tính của R để chỉ ra các phụ thuộc hàm có thể có trên mọi trạng thái quan hệ của r(R) của R. Khi ngữ nghĩa của hai tập thuộc tính trong R chỉ ra rằng có thể có một phụ thuộc hàm, chúng ta sẽ đặc tả phụ thuộc hàm như một ràng buộc. Các trạng thái quan hệ r(R) thoả mãn các ràng buộc phụ thuộc hàm được gọi là các trạng thái hợp pháp của R, bởi vì chúng tuân theo các ràng buộc phụ thuộc hàm. Như vậy, việc sử

dụng chủ yếu của các phụ thuộc hàm là dùng để mô tả một lược đồ quan hệ R bằng việc chỉ ra các ràng buộc trên các thuộc tính phải thoả mãn ở mọi thời điểm.

Một phụ thuộc hàm là một tính chất của lược đồ quan hệ R chứ không phải là tính chất của một trạng thái hợp pháp r của R. Vì vậy, một phụ thuộc hàm không thể được phát hiện một cách tự động từ một trạng thái r mà phải do một người hiểu biết ngữ nghĩa của các thuộc tính xác định một cách rõ ràng. Ví dụ, ta có quan hệ sau:

DẠY	Giáoviên	Mônhọc	Tàiliệu
	AA	Mônhọc1	XX
	AA	Mônhọc2	YY
	BB	Mônhọc3	ZZ
	CC	Mônhọc4	TT

Mới nhìn qua, chúng ta có thể nói có một phụ thuộc hàm Tàiliệu $\rightarrow$ Mônhọc, tuy nhiên chúng ta không thể khẳng định được vì điều đó chỉ đúng với trạng thái quan hệ này, biết đâu trong trạng thái quan hệ khác có thể có hai môn học khác nhau sử dụng cùng một tài liệu tham khảo. Với một trạng thái cụ thể, chúng ta chỉ có thể khẳng định là không có một phụ thuộc hàm giữa nhóm thuộc tính này và nhóm thuộc tính khác. Để làm điều đó chúng ta chỉ cần đưa ra một phản ví dụ. Chẳng hạn, ở trong quan hệ trên chúng ta có thể khẳng định rằng không có phụ thuộc hàm giữa Giáoviên và Mônhọc bằng cách chỉ ra ví dụ là AA dạy hai môn học " Mônhọc1" và "Mônhọc2" vậy Giáo viên không thể xác định duy nhất Môn học.

Để biểu diễn các phụ thuộc hàm trong một lược đồ quan hệ, chúng ta sử dụng khái niệm sơ đồ phụ thuộc hàm. Mỗi FD được biểu diễn bằng một đường nằm ngang. Các thuộc tính ở vế trái của FD được nối với đường biểu diễn FD bằng các đường thẳng đứng, các thuộc tính ở vế phải của FD được nối với đường biểu diễn FD bằng mũi tên chỉ đến các thuộc tính.

Ví dụ: Ta có lược đồ quan hệ

MƯỢN(Sốthẻ, Mãsốsách, Tênngườimượn, Tênsách, Ngàymượn)

Với các phụ thuộc hàm:

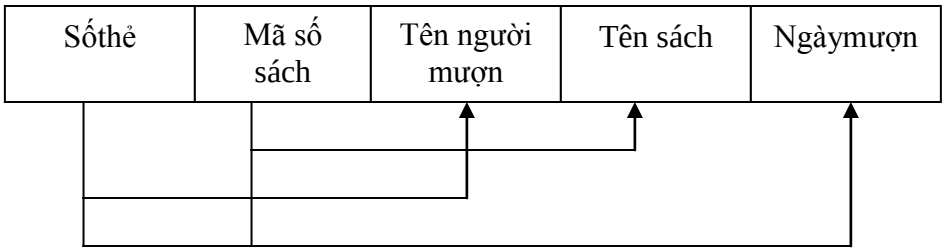
Sốthẻ $\rightarrow$ Tênngườimượn

Mãsốsách $\rightarrow$ Tênsách

Sốthẻ, Mãsốsách $\rightarrow$ Ngàymượn

Có sơ đồ phụ thuộc hàm như sau:

MƯỢN



2.2- Các quy tắc suy diễn đối với các phụ thuộc hàm

Chúng ta ký hiệu F là tập các phụ thuộc hàm được xác định trên một lược đồ quan hệ R. Một phụ thuộc hàm $X \rightarrow Y$ được gọi là suy diễn được từ một tập các phụ thuộc hàm F được xác định trên R nếu $X \rightarrow Y$ đúng trong mỗi trạng thái quan hệ r là mở rộng hợp pháp của R, nghĩa là mỗi khi r làm thoả mãn mọi phụ thuộc hàm trong F, $X \rightarrow Y$ cũng đúng trong r. Tập hợp tất cả các phụ thuộc hàm suy diễn được từ F được gọi là bao đóng của F và được ký hiệu là F^+. Để xác định một cách suy diễn các phụ thuộc hàm có hệ thống, chúng ta phải phát hiện một tập hợp các quy tắc suy diễn. Tập quy tắc này sẽ được sử dụng để suy diễn các phụ thuộc hàm mới từ một tập các phụ thuộc hàm cho trước. Ta sử dụng ký hiệu $F \models X \rightarrow Y$ để ký hiệu phụ thuộc hàm $X \rightarrow Y$ được suy diễn từ tập các phụ thuộc hàm F. Để cho tiện, ta viết tắt phụ thuộc hàm có dạng $\{X,Y\} \rightarrow Z$ thành $XY \rightarrow Z$ (nghĩa là ta nối các biến và bỏ dấu ngoặc nhọn đi). Có 6 quy tắc suy diễn đối với các phụ thuộc hàm:

QT1 (quy tắc phản xạ) : Nếu $X \supset Y$ thì $X \rightarrow Y$

QT2 (quy tắc tăng) : $\{ X \rightarrow Y \} \models XZ \rightarrow YZ$

QT3 (quy tắc bắc cầu) : $\{ X \rightarrow Y, Y \rightarrow Z \} \models X \rightarrow Y$

QT4 (quy tắc chiếu) : $\{ X \rightarrow YZ \} \models X \rightarrow Y$ và $X \rightarrow Z$

QT5 (quy tắc hợp) : $\{ X \rightarrow Y , X \rightarrow Z \} \models X \rightarrow YZ$

QT6 (quy tắc tựa bắc cầu) : $\{ X \rightarrow Y, WY \rightarrow Z \} \models WX \rightarrow Z$

Quy tắc phản xạ phát biểu rằng một tập hợp các thuộc tính luôn luôn xác định chính nó hoặc một tập con bất kỳ của nó. Vì QT1 tạo ra các phụ thuộc luôn luôn đúng, những phụ thuộc như vậy được gọi là *tầm thường*. Một cách hình thức, một phụ thuộc hàm $X \rightarrow Y$ là tầm thường nếu $X \supset Y$, ngược lại, nó được gọi là *không tầm thường*. Quy tắc tăng nói rằng việc thêm cùng một tập thuộc tính vào cả hai vế của một phụ thuộc hàm sẽ tạo ra một phụ thuộc hàm đúng đắn. Theo quy tắc 3, các phụ thuộc hàm là bắc cầu. Quy tắc chiếu (QT4) nói rằng chúng ta có thể bỏ bớt các thuộc tính ra khỏi vế phải của phụ thuộc hàm. Việc áp dụng nhiều lần quy tắc này có thể tách phụ thuộc hàm $X \rightarrow \{A_1, A_2,\ldots, A_n\}$ thành một tập hợp phụ thuộc hàm $\{ X\rightarrow A_1, X\rightarrow A_2, ..., X\rightarrow A_n\}$. Quy tắc hợp cho phép chúng ta làm ngược lại, ta có thể gộp các phụ thuộc hàm $\{ X\rightarrow A_1, X\rightarrow A_2, ..., X\rightarrow A_n\}$ thành một phụ thuộc hàm đơn $X\rightarrow \{A_1, A_2, ..., A_n\}$.

Có thể chứng minh các quy tắc suy diễn ở trên một cách trực tiếp hoặc bằng phản chứng dựa trên định nghĩa của phụ thuộc hàm. Để chúng minh phản chứng, ta giả thiết một quy tắc là không đúng và chỉ ra rằng điều đó là không thể. Sau đây là chứng minh các quy tắc.

Quy tắc 1:

Giả sử rằng $X \supset Y$ và hai bộ t_1 và t_2 trong một thể hiện quan hệ r của R sao cho $t_1[X] = t_2[X]$. Khi đó $t_1[Y] = t_2[Y]$ bởi vì $X \supset Y$, như vậy $X\rightarrow Y$ phải xẩy ra trong r.

Quy tắc 2: (chứng minh phản chứng)

Giả sử rằng $X\rightarrow Y$ đúng trong một thể hiện quan hệ r của R nhưng $XZ\rightarrow YZ$ không đúng. Khi đó phải có hai bộ t_1 và t_2 trong r sao cho:

(1) $t_1[X] = t_2[X]$,

(2) $t_1[Y] = t_2[Y]$,

(3) $t_1[XZ] = t_2[XZ]$ và

(4) $t_1[YZ] \neq t_2[YZ]$. Điều đó là không thể bởi vì từ (1) và (3) chúng ta suy ra

(5) $t_1[Z] = t_2[Z]$, và từ (2) và (5) ta suy ra $t_1[YZ] = t_2[YZ]$, mâu thuẫn với (4).

Quy tắc 3: Giả sử rằng

(1) X→ Y và (2) Y→ Z cả hai đúng trong một quan hệ r. Khi đó với mọi bộ t_1 và t_2 trong r sao cho $t_1[X] = t_2[X]$ ta phải có (3) $t_1[Y] = t_2[Y]$ theo giả thiết (1). Như vậy chúng ta cũng phải có (4) $t_1[Z] = t_2[Z]$ theo (3) và giả thiết (2). Vậy X→ Z phải đúng trong r.

Chúng ta có thể chứng minh các quy tắc từ quy tắc 4 đến quy tắc 6 theo phương pháp trên. Tuy nhiên ta có thể lợi dụng các quy tắc đã được chứng minh là đúng để chứng minh chúng. Sau đây ta chứng minh theo cách đó.

Quy tắc 4:

1. X→ YZ (cho trước)

2. YZ → Y (sử dụng QT1 và YZ ⊃ Y)

3. X→ Y (sử dụng QT3 trên 1. và 2.)

Quy tắc 5:

1. X→ Y (cho trước)

2. X→ Z (cho trước)

3. X→ YX (sử dụng QT2 trên 1. bằng cách thêm vào cả hai vế X, và XX=X)

4. YX→ YZ (sử dụng QT2 trên 2. bằng cách thêm vào cả hai vế Y)

5. X→YZ (sử dụng QT3 trên 3. và 4.)

Quy tắc 6:

1. X→Y (cho trước)

2. WY→Z (cho trước)

3. WX→WY (sử dụng QT2 trên 1. bằng cách thêm vào cả hai vế W)

4. WX→ Y (sử dụng QT3 trên 3. và 2.)

Amstrong đã chứng minh rằng các quy tắc suy diễn từ QT1 đến QT3 là đúng và đầy đủ. Đúng có nghĩa là cho trước một tập các phụ thuộc hàm F trên một lược đồ quan hệ R , bất kỳ một phụ thuộc hàm nào suy diễn được bằng cách áp dụng các quy tắc từ QT1 đến QT3 cũng đúng trong mỗi trạng thái quan hệ r của R thoả mãn các phụ thuộc hàm trong F. Đầy đủ có nghĩa là việc sử dụng các quy tắc từ QT1 đến QT3 lặp lại nhiều lần để suy diễn các phụ thuộc hàm cho đến khi không còn

suy diễn được nữa sẽ cho kết quả là một tập hợp đầy đủ các phụ thuộc hàm có thể được suy diễn từ F. Nói cách khác, tập hợp các phụ thuộc hàm F^+ (bao đóng của F) có thể xác định được từ F bằng cách áp dụng các quy tắc suy diễn từ QT1 đến QT3. Các quy tắc từ QT1 đến QT3 được gọi là các *quy tắc suy diễn Amstrong.*

Thông thường, những người thiết kế cơ sở dữ liệu đầu tiên chỉ ra một tập các phụ thuộc hàm dễ xác định được nhờ ngữ nghĩa của các thuộc tính của R. Sau đó, sử dụng các quy tắc Amstrong để suy diễn các phụ thuộc hàm bổ sung. Một cách có hệ thống, để xác định tất cả các phụ thuộc hàm bổ sung là đầu tiên hãy xác định mỗi tập thuộc tính X xuất hiện ở vế trái của một phụ thuộc hàm nào đấy trong F và sau đó xác định tập hợp tất cả các thuộc tính phụ thuộc vào X. Như vậy, với mỗi tập thuộc tính X, chúng ta xác định tập X^+ các thuộc tính phụ thuộc hàm vào X dựa trên F. X^+ được gọi là bao đóng của X dưới F. Thuật toán xác định X^+ như sau:

Thuật toán 4.1: (xác định X^+, bao đóng của X dưới F)

$X^+ = X$;

Repeat

Old $X^+ = X^+$;

với mỗi phụ thuộc hàm $Y \rightarrow Z$ trong F thực hiện

nếu $X^+ \supset Y$ thì $X^+ = X^+ \cup Z$;

until (X^+ = Old X^+);

Ví dụ: Xét lược đồ quan hệ

R = {MãsốNV, HọtênNV, MãsốDA, TênDA, ĐịađiểmDA, Sốgiờ}

và tập phụ thuộc hàm F = {MãsốNV $\rightarrow$ HọtênNV,

MãsốDA $\rightarrow$ TênDA, ĐịađiểmDA,

{MãsốNV, MãsốDA} $\rightarrow$ Số giờ}

Áp dụng thuật toán 4.1 ta có:

$\{MãsốNV\}^+$ = { MãsốNV, HọtênNV}

$\{MãsốDA\}^+$ = { MãsốDA, TênDA, ĐịađiểmDA}

$\{MãsốNV, MãsốDA\}^+$ = {MãsốNV, HọtênNV, MãsốDA, TênDA, Sốgiờ}

Bao đóng và khóa: Để ý rằng nếu X^+ là là tập tất cả các thuộc tính của quan hệ thì có nghĩa là X xác định hàm các thuộc tính còn lại, hay nói cách khác X là một siêu khóa. Chúng ta có thể kiểm tra xem một tập thuộc tính X có phải là khóa của một quan hệ bằng cách trước tiên xem X^+ có chứa tất cả các thuộc tính của quan hệ hay không sau đó kiểm tra không có một tập con S nào được lập từ X bằng cách loại bỏ một thuộc tính của X thỏa mãn S^+chứa tất cả các thuộc tính của quan hệ (nghĩa là X là siêu khóa tối thiểu). Ví dụ:

Xét lược đồ quan hệ R(A,B,C,D,E,F) và tập phụ thuộc hàm

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

Ta có $\{A,B\}^+ = \{A,B,C,D,E,F\}$, $A^+ = \{A,C,D\}$, $B^+ = \{B,E\}$, vậy AB là khóa của quan hệ.

2.3- Sự tương đương của các tập phụ thuộc hàm

Trong phần này chúng ta thảo luận về sự tương đương của hai tập phụ thuộc hàm. Một tập hợp các phụ thuộc hàm E được phủ bởi một tập các phụ thuộc hàm F - hoặc F phủ E - nếu mỗi một phụ thuộc hàm trong E đều ở trong F^+ ,điều đó có nghĩa là mỗi phụ thuộc hàm trong E có thể suy diễn được từ F. Hai tập phụ thuộc hàm E và F là tương đương nếu $E^+ = F^+$. Như vậy tương đương có nghĩa là mỗi phụ thuộc hàm trong E có thể suy diễn được từ F và mỗi phụ thuộc hàm trong F có thể suy diễn được từ E.

Chúng ta có thể xác định xem F có phủ E hay không bằng cách tính X^+ đối với F đối với mỗi thuộc hàm $X \rightarrow Y$ trong E và sau đó kiểm tra xem X^+ này có các thuộc tính trong Y hay không (nghĩa là $X^+ \supset Y$). Nếu điều đó xẩy ra với mỗi phụ thuộc hàm trong E, thì F phủ E. Chúng ta xác định xem E và F có tương đương hay không bằng cách kiểm tra E phủ F và F phủ E.

Ví dụ : Xét hai tập phụ thuộc hàm

$$F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow AD, E \rightarrow H \}$$

$$E = \{ A \rightarrow CD, \ E \rightarrow AH \}$$

Ta chứng minh F phủ E:

Tìm bao đóng của các vế trái của các phụ thuộc hàm trong E theo F. Áp dụng thuật toán 4.1 ở trên, ta có:

$\{A\}^+ = \{ A, C, D \}$;

$\{E\}^+ = \{E, A,D, H\}$

ta thấy các bao đóng này chứa các vế phải tương ứng. Từ đó suy ra F phủ E.

Ta chứng minh E phủ F:

Tìm bao đóng của các vế trái của các phụ thuộc hàm trong F theo E. Ta có

$\{A\}^+ =\{A,C,D\}$,

$\{AC\}+ = \{ A,C,D\}$,

$\{E\}+ = \{E,A,H\}$, ta thấy các bao đóng này chứa các vế phải tương ứng. Từ đó suy ra E phủ F.

Tóm lại E tương đương với F.

2.4- Các tập phụ thuộc hàm tối thiểu

Một tập phụ thuộc hàm là *tối thiểu* nếu nó thoả mãn các điều kiện sau đây:

1. Vế phải của các phụ thuộc hàm trong F chỉ có một thuộc tính.

2. Chúng ta không thể thay thế bất kỳ một phụ thuộc hàm $X \rightarrow A$ trong F bằng phụ thuộc hàm $Y \rightarrow A$, trong đó Y là tập con đúng của X mà vẫn còn là một tập phụ thuộc hàm tương đương với F.

3. Chúng ta không thể bỏ đi bất kỳ phụ thuộc hàm nào ra khỏi F mà vẫn có một tập phụ thuộc hàm tương đương với F.

Chúng ta có thể nghĩ về tập tối thiểu các phụ thuộc hàm như là một tập hợp ở dạng chuẩn không có sự dư thừa. Điều kiện 1 đảm bảo rằng mỗi phụ thuộc hàm là ở dạng chính tắc với một thuộc tính ở vế phải. Điều kiện 2 và 3 đảm bảo rằng không có sự dư thừa trong các phụ thuộc hoặc do có các thuộc tính dư thừa ở vế trái của phụ thuộc, hoặc do có một phụ thuộc có thể được suy diễn từ các phụ thuộc khác ở trong F.

Một *phủ tối thiểu* của một tập phụ thuộc hàm F là một tập tối thiểu các phụ thuộc hàm F_{min} tương đương với F. Thường có rất nhiều các phủ tối thiểu cho một tập các phụ thuộc hàm. Chúng ta luôn luôn có thể tìm được ít nhất là một phủ tối thiểu G cho một tập các phụ thuộc hàm F bất kỳ theo thuật toán 4.2 sau đây:

Thuật toán 4.2: (Tìm phủ tối thiểu G cho F)

1. Đặt G := F;

2. Thay thế mỗi phụ thuộc hàm $X \rightarrow \{A_1, A_2, ..., A_n\}$ trong G bằng n phụ thuộc hàm $X \rightarrow A_1$, $X \rightarrow A_2$, … , $X \rightarrow A_n$.

3. Với mỗi phụ thuộc hàm $X \rightarrow A$ trong G,

với mỗi thuộc tính B là một phần tử của X

nếu $((G - (X \rightarrow A) \cup ((X - \{B\}) \rightarrow A)$ là tương đương với G

thì thay thế $X \rightarrow A$ bằng $(X - \{B\}) \rightarrow A$ ở trong G

4. Với mỗi phụ thuộc hàm $X \rightarrow A$ còn lại trong G

nếu $(G - \{X \rightarrow A\})$ là tương đương với G

thì loại bỏ $X \rightarrow A$ ra khỏi G

Ví dụ áp dụng : Tìm phủ tối thiểu cho tập phụ thuộc hàm:

$\{A \rightarrow B, A \rightarrow C, B \rightarrow A, B \rightarrow C, C \rightarrow A, C \rightarrow B\}$

Áp dụng thuật toán trên, chúng ta có thể tìm được các phủ tối thiểu sau:

1) Do $A \rightarrow B$ và $B \rightarrow C$ nên $A \rightarrow C$ là thừa. Do $C \rightarrow B$ và $B \rightarrow A$ nên $C \rightarrow A$ là thừa. Bỏ những phụ thuộc hàm thừa đi, ta có $\{A \rightarrow B, B \rightarrow A, B \rightarrow C, C \rightarrow B\}$ là một phủ tối thiểu.

2) Do $A \rightarrow B$ và $B \rightarrow C$ nên $A \rightarrow C$ là thừa. Do có $B \rightarrow C$ và $C \rightarrow A$ nên $B \rightarrow A$ là thừa. Do có $C \rightarrow A$ và $A \rightarrow B$ nên $C \rightarrow B$ là thừa. Bỏ những phụ thuộc hàm thừa đi, ta nhận được một phủ tối thiểu khác là $\{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$

3. Các dạng chuẩn dựa trên khóa chính

Sau khi đã nghiên cứu các phụ thuộc hàm và một số tính chất của chúng, bây giờ chúng ta sẽ sử dụng chúng như thông tin về ngữ nghĩa của các lược đồ quan hệ. Ta giả sử rằng mỗi một quan hệ được cho trước một tập các phụ thuộc hàm và mỗi quan hệ có một khoá chính. Trong phần này chúng ta sẽ nghiên cứu các dạng chuẩn và quá trình chuẩn hoá các lược đồ quan hệ.

3.1- Nhập môn về chuẩn hoá

Quá trình chuẩn hoá (do Codd đề nghị 1972) lấy một lược đồ quan hệ và thực hiện một loạt các kiểm tra để xác nhận nó có thoả mãn một dạng chuẩn nào đó hay không. Quá trình này được thực hiện theo phương pháp trên xuống bằng việc đánh

giá mỗi quan hệ với tiêu chuẩn của các dạng chuẩn và tách các quan hệ nếu cần. Quá trình này có thể xem như là việc thiết kế quan hệ bằng phân tích. Lúc đầu, Codd đề nghị ba dạng chuẩn gọi là dạng chuẩn 1, dạng chuẩn 2 và dạng chuẩn 3. Một định nghĩa mạnh hơn cho dạng chuẩn 3 gọi là dạng chuẩn Boyce-Codd do Boyce và Codd đề nghị muộn hơn. Tất cả các dạng chuẩn này dựa trên các phụ thuộc hàm giữa các thuộc tính của một quan hệ. Sau đó, dạng chuẩn 4 (4NF) và dạng chuẩn 5 (5NF) được đề nghị dựa trên các phụ thuộc hàm đa trị và các phụ thuộc hàm nối.

Chuẩn hoá dữ liệu có thể được xem như một quá trình phân tích các lược đồ quan hệ cho trước dựa trên các phụ thuộc hàm và các khoá chính của chúng để đạt đến các tính chất mong muốn:

(1) Cực tiểu sự dư thừa và

(2) Cực tiểu các phép cập nhật bất thường.

Các lược đồ quan hệ không thoả mãn các kiểm tra dạng chuẩn sẽ được tách ra thành các lược đồ quan hệ nhỏ hơn thoả mãn các kiểm tra và có các tính chất mong muốn. Như vậy, thủ tục chuẩn hoá cung cấp cho những người thiết kế cơ sở dữ liệu:

- Một cơ cấu hình thức để phân tích các lược đồ quan hệ dựa trên các khoá của nó và các phụ thuộc hàm giữa các thuộc tính của nó.
- Một loạt các kiểm tra dạng chuẩn có thể thực hiện trên các lược đồ quan hệ riêng rẽ sao cho cơ sở dữ liệu quan hệ có thể được chuẩn hoá đến một mức cần thiết.

Dạng chuẩn của một quan hệ liên quan đến điều kiện dạng chuẩn cao nhất mà nó thoả mãn. Các dạng chuẩn khi được xem xét độc lập với các sự kiện khác không đảm bảo một thiết kế cơ sở dữ liệu tốt. Nói chung, việc xác minh riêng biệt từng lược đồ quan hệ ở dạng chuẩn này dạng chuẩn nọ là chưa đủ. Tốt hơn là quá trình chuẩn hoá thông qua phép tách phải khẳng định một vài tính chất hỗ trợ mà tất cả các lược đồ quan hệ phải có. Chúng gồm hai tính chất sau:

- Tính chất nối không mất mát (hoặc nối không phụ thêm), nó đảm bảo rằng vấn đề tạo ra các bộ giả không xuất hiện đối với các lược đồ quan hệ được tạo ra sau khi tách.

- Tính chất bảo toàn sự phụ thuộc, nó đảm bảo rằng từng phụ thuộc hàm sẽ được biểu hiện trong các quan hệ riêng rẽ nhận được sau khi tách.

Tính chất nối không mất mát là rất quan trọng, phải đạt được bằng mọi giá, còn tính chất bảo toàn phụ thuộc thì cũng rất mong muốn nhưng đôi khi có thể hy sinh.

Trước khi định nghĩa các dạng chuẩn chúng ta xem lại định nghĩa các khóa của một quan hệ. Một *siêu khóa* (superkey) của một lược đồ quan hệ R = {A_1, A_2, .., A_n} là một tập con S các thuộc tính của R, S $\subseteq$ R, có tính chất là không có hai bộ t_1, t_2 trong một trạng thái quan hệ hợp pháp r nào của R mà $t_1[S] = t_2[S]$. Một *khóa* K là một siêu khóa có tính chất là nếu bỏ đi bất kỳ thuộc tính nào ra khỏi K thì K không còn là siêu khóa nữa. Điều đó có nghĩa là khóa là một siêu khóa tối thiểu. Nếu một lược đồ quan hệ có nhiều hơn một khóa thì các khóa đó được gọi là các *khóa dự tuyển*. Một trong những khóa dự tuyển sẽ được chỉ định làm *khóa chính* và các khóa còn lại được gọi là các *khóa phụ* (secondary key). Một lược đồ quan hệ phải có một khóa chính. Một thuộc tính của một lược đồ quan hệ R được gọi là một *thuộc tính khóa* của R nếu nó là một thành phần của khóa chính của R. Một thuộc tính được gọi là thuộc tính *không khóa* nếu nó không phải là một thuộc tính khóa.

3.2- Dạng chuẩn 1

Một quan hệ được gọi là ở *dạng chuẩn 1* (1NF) nếu miền giá trị của một thuộc tính chỉ chứa các giá trị nguyên tử (đơn, không phân chia được) và giá trị của mỗi thuộc tính trong một bộ phải là một giá trị đơn lấy từ miền giá trị của thuộc tính đó. Như vậy, 1NF không cho phép quan hệ có các thuộc tính đa trị hoặc có các nhóm thuộc tính lặp (quan hệ trong quan hệ). Nó chỉ cho phép các giá trị của các thuộc tính là nguyên tử.

Ví dụ, xét các quan hệ ĐƠNVỊ và NHÂNVIÊN_DỰÁN như hình vẽ dưới đây. Các quan hệ này không thỏa mãn điều kiện 1NF. Quan hệ ĐƠNVỊ chứa thuộc tính đa trị Địađiểm, quan hệ NHÂNVIÊN_DỰÁN chứa nhóm các thuộc tính lặp (Tênnhânviên, Sốgiờ).

ĐƠNVỊ	MãsốĐV	TênĐV	Mã sốNQL	Địađiểm
	5	Nghiêncứu	NV002	Namđịnh, Hànội,Bắcninh
	4	Hànhchính	NV014	Hànôi
	1	Lãnhđạo	NV061	Hànội

NHÂNVIÊN_DỰÁN	MãsốDA	TênDA	Tênnhânviên	Sốgiờ
	1	DA01	Vân Nam	15 20
	2	DA02	Nam Thanh Bằng	10 12 28
	3	DA03	Thanh	20

Để đạt đến dạng chuẩn 1 đối với các quan hệ ở trên chúng ta dùng phương pháp sau:

Loại bỏ các thuộc tính vi phạm dạng chuẩn 1 và đặt chúng vào một bảng riêng cùng với khoá chính của quan hệ ban đầu. Khoá chính của bảng này là một tổ hợp của khoá chính của quan hệ ban đầu và thuộc tính đa trị hoặc khoá bộ phận của nhóm lặp.

Các thuộc tính còn lại lập thành một quan hệ với khóa chính là khóa chính ban đầu.

Áp dụng : Lược đồ quan hệ ĐƠNVỊ ở trên sẽ được tách thành hai:

ĐƠNVỊ (Mã sốĐV, TênĐV, Mã sốNQL)

ĐƠNVỊ_ĐỊAĐIỂM (MãsốĐV, Địađiểm)

Lược đồ quan hệ NHÂNVIÊN_DỰÁN cũng được tách thành hai:

DỰÁN (MãsốDA, TênDA)

NHÂNVIÊN_DỰÁN(MãsốDA, Tênnhânviên, Sốgiờ)

3.3- Dạng chuẩn 2

Dạng chuẩn 2 (2NF) dựa trên khái niệm phụ thuộc hàm đầy đủ. Một phụ thuộc hàm $X \rightarrow Y$ là một phụ thuộc hàm đầy đủ nếu loại bỏ bất kỳ thuộc tính A nào ra khỏi X thì phụ thuộc hàm không còn đúng nữa. Điều đó có nghĩa là, với thuộc tính A bất kỳ, $A \in X$, $(X - \{A\})$ không xác định Y. Một phụ thuộc hàm $X \rightarrow Y$ là phụ thuộc bộ phận nếu có thể bỏ một thuộc tính $A \in X$, ra khỏi X phụ thuộc hàm vẫn đúng, điều đó có nghĩa là với $A \in X$, $(X - \{A\}) \rightarrow Y$.

Ví dụ, xét lược đồ quan hệ

NHÂNVIÊN_DỰÁN(MãsốNV, MãsốDA, Sốgiờ, HọtênNV, TênDA, ĐịađiểmDA)

MãsốNV, MãsốDA → Sốgiờ là phụ thuộc hàm đầy đủ

MãsốNV, MãsốDA → HọtênNV là phụ thuộc hàm bộ phận, bởi vì có phụ thuộc hàm

MãsốNV →HọtênNV

Việc kiểm tra đối với 2NF bao gồm việc kiểm tra đối với các phụ thuộc hàm có các thuộc tính ở vế trái của nó là một bộ phận của khoá chính. Nếu khoá chính chứa một thuộc tính đơn thì không cần phải kiểm tra. Một lược đồ quan hệ R là ở *dạng chuẩn 2* nếu nó thỏa mãn dạng chuẩn 1 và mỗi thuộc tính không khoá A trong R là phụ thuộc hàm đầy đủ vào khoá chính của R.

Nếu một quan hệ không thoả mãn điều kiện 2NF ta có thể chuẩn hoá nó để có các quan hệ 2NF như sau: Loại bỏ các thuộc tính không khoá phụ thuộc vào một bộ phận khoá chính và tách thành ra một bảng riêng, khoá chính của bảng là bộ phận khoá mà chúng phụ thuộc vào. Các thuộc tính còn lại lập thành một quan hệ, khóa chính của nó là khóa chính ban đầu.

Ví dụ, xét lược đồ quan hệ:

NHÂNVIÊN_DỰÁN(MãsốNV, MãsốDA, Sốgiờ, HọtênNV, TênDA, ĐịađiểmDA)

với các phụ thuộc hàm:

MãsốNV, MãsốDA → Sốgiờ

MãsốNV →HọtênNV

MãsốDA→TênDA, ĐịađiểmDA

Ta thấy ở đây có những thuộc tính không khoá phụ thuộc vào một bộ phận của khoá chính, như vậy nó không thoả mãn điều kiên 2NF.

Áp dụng phương pháp chuẩn hoá trên, lược đồ được tách thành các lược đồ như sau:

N_D1(MãsốDA, TênDA, ĐịađiểmDA)

N_D2(MãsốNV , HọtênNV)

N_D3(MãsốNV, MãsốDA, Sốgiờ)

3.4- Dạng chuẩn 3

Dạng chuẩn 3 (3NF) dựa trên khái niệm phụ thuộc bắc cầu. Một phụ thuộc hàm X $\rightarrow$ Y trong một lược đồ quan hệ R là một phụ thuộc hàm bắc cầu nếu có một tập hợp thuộc tính Z không phải là một khoá dự tuyển cũng không phải là một tập con của một khoá nào và cả hai X $\rightarrow$ Z và Z $\rightarrow$Y đều đúng. Theo định nghĩa nguyên thuỷ của Codd, một lược đồ quan hệ R là ở 3NF nếu nó thoả mãn 2NF và không có thuộc tính không khoá nào của R là phụ thuộc bắc cầu vào khoá chính.

Nếu một lược đồ quan hệ không thoả mãn điều kiện 3NF, ta có thể chuẩn hoá nó để có được các lược đồ 3NF như sau: Loại bỏ các thuộc tính phụ thuộc bắc cầu ra khỏi quan hệ và tách chúng thành một quan hệ riêng có khoá chính là thuộc tính bắc cầu. Các thuộc tính còn lại lập thành một quan hệ có khóa chính là quan hệ ban đầu.

Ví dụ: Xét lược đồ quan hệ

NHÂNVIÊN_ĐƠNVỊ(HọtênNV, MãsốNV, Ngàysinh, Địachỉ, MãsốĐV, TênĐV, MãsốNQL)

Với các phụ thuộc hàm:

MãsốNV$\rightarrow$ HọtênNV, Ngày sinh, Địachỉ, MãsốĐV, TênĐV, MãsốNQL

MãsốDV$\rightarrow$ TênĐV, Mã sốNQL

Các thuộc tính TênĐV, MãsốNQL phụ thuộc bắc cầu vào khoá chính, lược đồ quan hệ không thoả mãn điều kiện 3NF.

Áp dụng phương pháp chuẩn hoá ở trên, lược đồ được tách ra như sau:

NV_DV1(HọtênNV, MãsốNV, Ngàysinh, Địachỉ, MãsốĐV)

NV_DV2(MãsốĐV, TênĐV, MãsốNQL)

3.5- Dạng chuẩn Boyce-Codd

Một lược đồ quan hệ R được gọi là ở dạng chuẩn Boyce-Codd (BCNF) nếu nó là ở dạng chuẩn 3NF và không có các thuộc tính khóa phụ thuộc hàm và thuộc tính không khóa.

Ví dụ: Lược đồ

R (A_1,A_2,A_3,A_4,A_5)

Với các phụ thuộc hàm:

$A1,A_2 \rightarrow A_3,A_4,A_5$

$A_4 \rightarrow A_2$

Quan hệ này vi phạm dạng chuẩn BCNF bởi vì có thuộc tính khóa (A_2) phụ thuộc hàm vào thuộc tính không khóa (A_4).

Nếu một lược đồ quan hệ không thoả mãn điều kiện BCNF, ta có thể chuẩn hoá nó để có được các lược đồ BCNF như: Loại bỏ các thuộc tính khóa phụ thuộc hàm vào thuộc tính không khóa ra khỏi quan hệ và tách chúng thành một quan hệ riêng có khoá chính là thuộc tính không khóa gây ra phụ thuộc.

Áp dụng phương pháp chuẩn hóa ở trên, lược đồ được tách ra như sau:

R1($\underline{A_4}$, A_2)

R2($\underline{A_1}$, $\underline{A_4}$, A_3, A_5)

Ví dụ áp dụng:

Cho lược đồ quan hệ R = {$\underline{A}$,$\underline{B}$,C,D,E,F,G,H,I,J} có khóa chính là A,B

Với tập các phụ thuộc hàm :

$A,B \rightarrow C,D,E,F,G,H,I,J$

$A \rightarrow E,F,G,H,I,J$

$F \rightarrow I, J$

$D \rightarrow B$

Do có có phụ thuộc hàm $A \rightarrow E,F,G,H,I,J$ mà A là một bộ phận của khóa chính nên quan hệ R là vi phạm 2NF. Ta tách R thành R_1($\underline{A}$,E,F,G,H,I,J) và R_2($\underline{A}$,$\underline{B}$,C,D). Trong R_1, do có phụ thuộc hàm $F \rightarrow$ I, J, nên ta có I,J phụ thuộc bắc cầu vào khóa chính, R_1 là quan hệ vi phạm 3NF. Trong R_2 ta có phụ thuộc hàm $D \rightarrow B$ trong đó B là một thuộc tính khóa, R_2 vi phạm BCNF. Tách R_1 và R_2 ta có:

R_{11}($\underline{F}$,I,J) , R_{12}($\underline{A}$,E,F,G,H), R_{21}($\underline{D}$,B), R_{22}($\underline{A}$,$\underline{D}$,C)

4. Các thuật toán thiết kế cơ sở dữ liệu quan hệ và các dạng chuẩn cao hơn

Như chúng ta đã thảo luận trong đầu chương IV, có hai cách chính để thiết kế cơ sở dữ liệu quan hệ. Cách thứ nhất là *thiết kế trên-xuống* (top-down design). Đây

là cách hay được sử dụng nhất trong thiết kế ứng dụng cơ sở dữ liệu thương mại. Nó bao gồm việc thiết kế một lược đồ quan niệm trong một mô hình dữ liệu bậc cao, chẳng hạn như mô hình EER, sau đó ánh xạ lược đồ quan niệm vào một tập quan hệ sử dụng các thủ tục ánh xạ như đã nói đến trong chương III. Sau đó, mỗi một quan hệ được phân tích dựa trên các phụ thuộc hàm và các khóa chính được chỉ định bằng cách áp dụng các thủ tục chuẩn hóa như đã nói đến trong phần III chương này để loại bỏ các phụ thuộc hàm bộ phận và các phụ thuộc hàm bắc cầu. Việc phân tích các phụ thuộc không mong muốn cũng có thể được thực hiện trong quá trình thiết kế quan niệm bằng cách phân tích các phụ thuộc hàm giữa các thuộc tính bên trong các kiểu thực thể và các kiểu liên kết để ngăn ngừa sự cần thiết có sự chuẩn hóa phụ thêm sau khi việc ánh xạ được thực hiện.

Cách thứ hai là *thiết kế dưới-lên* (bottom-up design), một kỹ thuật tiếp cận và nhìn nhận việc thiết kế lược đồ cơ sở dữ liệu quan hệ một cách chặt chẽ trên cơ sở các phụ thuộc hàm được chỉ ra trên các thuộc tính của cơ sở dữ liệu. Sau khi người thiết kế chỉ ra các phụ thuộc, người ta áp dụng một thuật toán chuẩn hóa để tổng hợp các lược đồ quan hệ. Mỗi một lược đồ quan hệ riêng rẽ ở dạng chuẩn 3NF hoặc BCNF hoặc ở dạng chuẩn cao hơn.

Trong phần này chúng ta chủ yếu trình bày cách tiếp cận thứ hai. Trước tiên chúng ta sẽ định nghĩa lại các dạng chuẩn một cách tổng quát, sau đó trình bày các thuật toán chuẩn hóa và các kiểu phụ thuộc khác. Chúng ta cũng sẽ trình bày chi tiết hơn về hai tính chất cần có là nối không phụ thêm (mất mát) và bảo toàn phụ thuộc. Các thuật toán chuẩn hóa thường bắt đầu bằng việc tổng hợp một lược đồ quan hệ rất lớn, gọi là *quan hệ phổ quát* (universal relation), chứa tất cả các thuộc tính của cơ sở dữ liệu. Sau đó chúng ta thực hiện lặp đi lặp lại việc tách (decomposition) dựa trên các phụ thuộc hàm và các phụ thuộc khác do người thiết kế cơ sở dữ liệu chỉ ra cho đến khi không còn tách được nữa hoặc không muốn tách nữa.

4.1- Định nghĩa tổng quát các dạng chuẩn

Nói chung, chúng ta muốn thiết kế các lược đồ của chúng ta sao cho chúng không còn các phụ thuộc bộ phận và các phụ thuộc bắc cầu bởi vì các kiểu phụ thuộc này gây ra các sửa đổi bất thường. Các bước chuẩn hóa thành 3NF, BCNF đã được trình bày trong phần trước loại bỏ các phụ thuộc bộ phận và bắc cầu dựa trên khóa chính. Các định nghĩa này không tính đến các khóa dự tuyển của quan hệ. Trong phần này chúng ta sẽ đưa ra các định nghĩa về các dạng chuẩn tổng quát hơn,

có tính đến tất cả các khóa dự tuyển. Cụ thể, *thuộc tính khóa* được định nghĩa lại *là một bộ phận của một khóa dự tuyển*. Các phụ thuộc hàm bộ phận, đầy đủ, bắc cầu bây giờ sẽ được định nghĩa đối với tất cả các khóa dự tuyển của quan hệ.

Định nghĩa dạng chuẩn 1: Một lược đồ quan hệ R là ở dạng chuẩn 1 (1NF) nếu miền giá trị của các thuộc tính của nó chỉ chứa các *giá trị nguyên tử* (đơn, không phân chia được) và giá trị của một thuộc tính bất kỳ trong một bộ giá trị phải là một giá trị đơn thuộc miền giá trị của thuộc tính đó.

Định nghĩa dạng chuẩn 2: Một lược đồ quan hệ R là ở dạng chuẩn 2 (2NF) nếu mỗi thuộc tính không khóa A trong R không phụ thuộc bộ phận vào một khóa bất kỳ của R.

Ví dụ: Xét lược đồ quan hệ

R={A,B,C,D,E,F}

Với các phụ thuộc hàm A $\rightarrow$ B,C,D,E,F; B,C $\rightarrow$ A,D,E,F; B $\rightarrow$ F; D $\rightarrow$E.

Lược đồ trên có hai khóa dự tuyển là A và {B,C}. Ta chọn A làm khóa chính. Do có phụ thuộc hàm B $\rightarrow$ F nên F phụ thuộc bộ phận vào khóa {B,C}, lược đồ vi phạm chuẩn 2NF (chú ý rằng, trong định nghĩa dạng chuẩn dựa trên khóa chính, lược đồ này không vi phạm 2NF).

Định nghĩa dạng chuẩn 3: Một lược đồ quan hệ R là ở dạng chuẩn 3 (3NF) nếu khi một phụ thuộc hàm X $\rightarrow$ A thỏa mãn trong R, thì:

1) Hoặc X là một siêu khóa của R.

2) Hoặc A là một thuộc tính khóa của R.

Ví dụ: Xét lược đồ quan hệ R ở ví dụ trên. Giả sử nó được tách thành hai lược đồ:

R1 = {A,B,C,D,E}

R2 = {B, F}.

Do có phụ thuộc hàm D $\rightarrow$ E trong đó D không phải thuộc tính khóa, E cũng không phải là thuộc tính khóa, nên R1 vi phạm chuẩn 3NF

Định nghĩa dạng chuẩn Boyce- Codd: Một lược đồ quan hệ là ở dạng chuẩn Boyce-Codd (BCNF) nếu khi một phụ thuộc hàm X $\rightarrow$ A thỏa mãn trong R thì X là một siêu khóa của R.

Ví dụ: Xét lược đồ R = {A, B, C, D} có A là khóa chính và {B,C} là khóa dự tuyển. Nếu có tồn tại một phụ thuộc hàm D $\rightarrow$ B thì lược đồ này vi phạm BCNF vì B là một thuộc tính khóa (chú ý rằng trong trường hợp định nghĩa dạng chuẩn dựa trên khóa chính, lược đồ này không vi phạm BCNF).

4.2- Các thuật toán thiết kế lược đồ cơ sở dữ liệu quan hệ

4.2.1- Tách quan hệ và tính không đầy đủ của các dạng chuẩn

Tách quan hệ: Các thuật toán thiết kế cơ sở dữ liệu quan hệ được trình bày trong phần này bắt đầu từ một lược đồ quan hệ vũ trụ đơn R = {A_1, A_2, …, A_n} chứa tất cả các thuộc tính của cơ sở dữ liệu. Với giả thiết quan hệ vũ trụ, tên của mỗi thuộc tính là duy nhất. Tập hợp F các phụ thuộc hàm thỏa mãn trên các thuộc tính của R do những người thiết kế cơ sở dữ liệu chỉ ra sẽ được các thuật toán sử dụng. Sử dụng các phụ thuộc hàm, các thuật toán sẽ tách lược đồ quan hệ vũ trụ R thành một tập hợp các lược đồ quan hệ D = {R_1, R_2, …, R_m}, tập hợp đó sẽ là lược đồ cơ sở dữ liệu quan hệ. D được gọi là một phép tách (decomposition) của R. Chúng ta phải đảm bảo rằng mỗi thuộc tính trong R sẽ xuất hiện trong ít nhất là một lược đồ quan hệ R_i trong phép tách để nó khỏi bị "mất ". Một cách hình thức, ta có *điều kiện bảo toàn thuộc tính sau đây*:

$$\cup R_i = R$$

Tính không đầy đủ của các dạng chuẩn: Mục đích của chúng ta là mỗi quan hệ riêng rẽ R_i trong phép tách D là ở dạng chuẩn BCNF hoặc 3NF. Tuy nhiên, điều đó không đủ để đảm bảo một thiết kế cơ sở dữ liệu tốt. Bên cạnh việc xem xét từng quan hệ riêng rẽ, chúng ta cần xem xét toàn bộ phép tách. Ví dụ, xét hai quan hệ:

NV_ĐĐ(Tên, ĐịađiểmDA)

NV_DA1(Mã sốNV, Mã sốDA, Sốgiờ, TênDA, ĐịađiểmDA)

Ở phần I.4 chương này, ta thấy rằng dù quan hệ NV_ĐĐ là một quan hệ ở dạng BCNF nhưng khi chúng ta đem nối tự nhiên với quan hệ NV_DA1 thì chúng ta nhận được một quan hệ có chứa các bộ giả. Điều đó xảy ra là do ngữ nghĩa không rõ ràng của quan hệ NV_ĐĐ. Đó là một lược đồ quan hệ được thiết kế tồi. Chúng ta cần phải có tiêu chuẩn khác để cùng với các điều kiện 3NF và BCNF ngăn ngừa các thiết kế tồi như vậy. Trong các phần tiếp theo chúng ta sẽ nối đến các điều kiện phụ thêm phải thỏa mãn trên phép tách D.

4.2.2- Phép tách và sự bảo toàn phụ thuộc

Việc mỗi phụ thuộc hàm $X \rightarrow Y$ trong F hoặc được xuất hiện trực tiếp trong một trong các lược đồ quan hệ R_i trong phép tách D hoặc có thể được suy diễn từ các phụ thuộc hàm có trong R_i là rất có lợi. Ta gọi đó là *điều kiện bảo toàn phụ thuộc*. Chúng ta muốn bảo toàn phụ thuộc bởi vì mỗi phụ thuộc trong F biểu thị một ràng buộc trong cơ sở dữ liệu. Nếu như một trong các phụ thuộc không được thể hiện trong một quan hệ riêng rẽ R_i nào đó của phép tách, chúng ta không thể ép buộc ràng buộc này đối với quan hệ riêng rẽ, thay vào đó, chúng ta nối hai hoặc nhiều quan hệ trong phép tách và sau đó kiểm tra rằng phụ thuộc hàm thỏa mãn trong kết quả của phép nối. Rõ ràng đó là một thủ tục không hiệu quả và không thực tiễn.

Việc các phụ thuộc chính xác được chỉ ra ở trong F xuất hiện trong các quan hệ riêng rẽ của phép tách D là không cần thiết. Chỉ cần hợp của các phụ thuộc thỏa mãn trên các quan hệ riêng rẽ trong D là tương đương với F là đủ. Bây giờ chúng ta định nghĩa các khái niệm này một cách hình thức.

Cho trước một tập hợp các phụ thuộc F trên R, *phép chiếu của F trên R_i*, ký hiệu là $\pi_{Ri}(F)$ trong đó R_i là một tập con của R, là một tập hợp các phụ thuộc hàm $X \rightarrow Y$ trong F^+ sao cho các thuộc tính trong $X \cup Y$ đều được chứa trong R_i. Như vậy, phép chiếu của F trên mỗi lược đồ quan hệ R_i trong phép tách D là tập hợp các phụ thuộc hàm trong F^+, bao đóng của F, sao cho các thuộc tính ở vế trái và vế phải của chúng đều ở trong R_i. Ta nói rằng phép tách D = {R1, R2, …, Rm} của R bảo toàn phụ thuộc đối với F nếu hợp của các phép chiếu của F trên mỗi R_i trong D là tương đương với F. Điều đó có nghĩa là:

$$((\pi_{R1}(F)) \cup (\pi_{R2}(F)) \cup \ldots \cup (\pi_{Rm}(F)))^+ = F^+$$

Nếu một phép tách là không bảo toàn phụ thuộc, một vài phụ thuộc sẽ bị mất trong phép tách. Để kiểm tra xem một phụ thuộc hàm $X \rightarrow B$, trong đó X là tập thuộc tính thuộc về R_i, B là một thuộc tính thuộc R_i có thỏa mãn trong R_i hay không ta làm như sau: Trước hết tính X^+ , sau đó với mỗi thuộc tính B sao cho

1. B là một thuộc tính của R_i
2. B là ở trong X^+
3. B không ở trong X

Khi đó phụ thuộc hàm $X \rightarrow B$ thỏa mãn trong R_i.

Một ví dụ về phép tách không bảo toàn phụ thuộc. Xét lược đồ quan hệ:

R = { A,B,C,D} với các phụ thuộc hàm:

$A \rightarrow BCD$; $BC \rightarrow DA$; $D \rightarrow B$

Lược đồ này có hai khóa dự tuyển là A và BC. Lược đồ này vi phạm BCNF. Nó được tách thành:

R1 = {D,B}, lược đồ này chứa phụ thuộc hàm $D \rightarrow B$

R2 = {A,C,D}, lược đồ này chứa phụ thuộc hàm $A \rightarrow CD$

Rõ ràng sau khi tách, phụ thuộc hàm $BC \rightarrow DA$ bị mất.

Định lý: Luôn luôn tìm được một phép tách bảo toàn phụ thuộc D đối với F sao cho mỗi quan hệ R_i trong D là ở 3NF. Phép tách D đực thực hiện theo thuật toán sau đây:

Thuật toán 5.1: Tạo một phép tách bảo toàn phụ thuộc $D = \{R_1, R_2, \ldots, R_m\}$ của một quan hệ vũ trụ R dựa trên một tập phụ thuộc hàm F sao cho mỗi R_i trong D là ở 3NF. Thuật toán này chỉ đảm bảo tính chất bảo toàn phụ thuộc, không đảm bảo tính chất nối không mất mát.

Input: Một quan hệ vũ trụ R và một tập phụ thuộc hàm F trên các thuộc tính của R.

1) Tìm phủ tối thiểu G của F.
2) Với mỗi vế trái X của một phụ thuộc hàm xuất hiện trong G, hãy tạo một lược đồ trong D với các thuộc tính $\{X \cup \{A_1\} \cup \{A_2\} \cup \ldots \cup \{A_k\}\}$ trong đó $X \rightarrow A_1$, $X \rightarrow A_2, \ldots, X \rightarrow A_k$ chỉ là các phụ thuộc hàm trong G với X là vế trái (X là khóa của quan hệ này).
3) Đặt các thuộc tính còn lại (những thuộc tính chưa được đặt vào quan hệ nào) vào một quan hệ đơn để đảm bảo tính chất bảo toàn thuộc tính.

Ví dụ áp dụng:

Xét lược đồ: R = { A,B,C,D} , với các phụ thuộc hàm:

$F = \{A \rightarrow BCD;\ BC \rightarrow DA;\ D \rightarrow B\}$

Lược đồ này có hai khóa dự tuyển là A và BC.

Ta thực hiện thuật toán như sau: Trước tiên ta tìm G là phủ tối thiểu của F. Theo thuật toán tìm phủ tối thiểu, đầu tiên ta làm cho các vế phải trong G chỉ chứa một thuộc tính, ta có:

$G = \{A \rightarrow B; A \rightarrow C; A \rightarrow D; BC \rightarrow D; BC \rightarrow A; D \rightarrow B\}$

Sau đó ta bỏ đi các phụ thuộc hàm thừa (là các phụ thuộc hàm có thể suy diễn được từ các phụ thuộc hàm khác). Ta thấy $A \rightarrow B$ là thừa vì có $A \rightarrow D$, $D \rightarrow B$. Vậy G còn lại là:

$G = \{A \rightarrow C; A \rightarrow D; BC \rightarrow D; BC \rightarrow A; D \rightarrow B\}$. Lược đồ R sẽ được tách thành:

R_1(A,C,D); R2(B,C,D,A); R3(D,B) với các khóa chính được gạch dưới.

Rõ ràng rằng tất cả các phụ thuộc hàm trong G đều được thuật toán bảo toàn bởi vì mỗi phụ thuộc xuất hiện trong một trong các quan hệ của phép tách D. Bởi vì G tương đương với F, tất cả các phụ thuộc của F cũng được bảo toàn hoặc trực tiếp bằng thuật toán hoặc được suy diễn từ những phụ thuộc hàm trong các quan hệ kết quả, như vậy tính chất bảo toàn phụ thuộc được đảm bảo.

4.2.3- Phép tách và kết nối không mất mát

Phép tách D phải có một tính chất nữa là nối không mất mát (hoặc tính chất nối không phụ thêm), nó đảm bảo rằng không có các bộ giả được tạo ra khi áp dụng một phép nối tự nhiên vào các quan hệ trong phép tách. Chúng ta đã đưa ra ví dụ về phép tách không có tính chất nối không mất thông tin ở phần I.4 chương này. Trong phép tách đó, khi ta thực hiện phép nối tự nhiên trên các quan hệ của phép tách, rất nhiều các bộ giả đã sinh ra.

Một cách hình thức, ta nói rằng một phép tách $D = \{ R_1, R_2,\ldots,R_m\}$ của R có tính chất nối không mất mát (không phụ thêm) đối với một tập hợp phụ thuộc hàm F trên R nếu với mỗi trạng thái quan hệ r của R thỏa mãn F thì

$$* (\pi_{R1}(r) , \pi_{R1}(r) \ \ldots, \pi_{R1}(r)) = r$$

trong đó * là phép nối tự nhiên của các quan hệ trong D.

Nếu một phép tách không có tính chất nối không mất mát thông tin thì chúng ta có thể nhận được các bộ phụ thêm (các bộ giả) sau khi áp dụng các phép chiếu và nối tự nhiên. Nghĩa của từ mất mát ở đây là mất mát thông tin chưa không phải mất

các bộ giá trị. Vì vậy, với tính chất này ta nên gọi chính xác hơn là tính chất nối không phụ thêm.

Chúng ta có thuật toán để kiểm tra một phép tách có tính chất nối không mất mát thông tin hay không như sau:

Thuật toán 5.2: Kiểm tra tính chất nối không mất mát

Input: Một quan hệ vũ trụ $R(A_1,A_2,\ldots A_n)$, một phép tách $D = \{R_1, R_2, \ldots, R_m\}$ của R và một tập F các phụ thuộc hàm.

1) Tạo một ma trận S có m hàng, n cột. Mỗi cột của ma trận ứng với một thuộc tính, mỗi hàng ứng với mỗi quan hệ R_i
2) Đặt $S(i,j) = 1$ nếu thuộc tính A_j thuộc về quan hệ R_i và bằng 0 trong trường hợp ngược lại.
3) Lặp lại vòng lặp sau đây cho đến khi nào việc thực hiện vòng lặp không làm thay đổi S: Với mỗi phụ thuộc hàm $X \rightarrow Y$ trong F, xác định các hàng trong S có các ký hiệu 1 như nhau trong các cột ứng với các thuộc tính trong X. Nếu có một hàng trong số đó chứa 1 trong các cột ứng với thuộc tính Y thì hãy làm cho các làm cho các cột tương ứng của các hàng khác cũng chứa 1.
4) Nếu có một hàng chứa toàn ký hiệu "1" thì phép tách có tính chất nối không mất mát, ngược lại, phép tách không có tính chất đó.

Cho trước một quan hệ R được tách thành một số quan hệ $R_1, R_2, ..,R_m$. Thuật toán 5.2 bắt đầu bằng việc tạo ra một trạng thái quan hệ r trong ma trận S. Hàng i trong S biểu diễn một bộ t_i (tương ứng với quan hệ R_i). Hàng này có các ký hiệu "1" trong các cột tương ứng với các thuộc tính của R_i và các ký hiệu "0" trong các cột còn lại. Sau đó thuật toán biến đổi các hàng của ma trận này (trong vòng lặp của bước 3) sao cho chúng biểu diễn các bộ thỏa mãn tất cả các phụ thuộc hàm trong F. Ở cuối vòng lặp áp dụng các phụ thuộc hàm, hai hàng bất kỳ trong S – chúng biểu diễn hai bộ trong r – có các giá trị giống nhau đối với các thuộc tính của X ở vế trái của phụ thuộc hàm $X\rightarrow Y$ trong F sẽ cũng có các giá trị giống nhau đối với các thuộc tính của vế phải Y. Có thể chỉ ra rằng sau khi áp dụng vòng lặp của bước 3, nếu một hàng bất kỳ trong S kết thúc với toàn ký hiệu "1" thì D có tính chất nối không mất mát đối với F. Mặt khác, nếu không có hàng nào kết thúc bằng tất cả ký hiệu "1" thì D không thỏa mãn tính chất nối không mất mát. Trong trường hợp sau,

trạng thái quan hệ r được biểu diễn bằng S ở cuối thuật toán sẽ là một ví dụ về một trạng thái quan hệ r của R thỏa mãn các phụ thuộc trong F nhưng không thỏa mãn điều kiện nối không mất mát . Như vậy, quan hệ này được dùng như một phản ví dụ chứng minh rằng D không có tính chất nối không mất mát đối với F. Chú ý rằng các ký hiệu "1" và "0" không có ý nghĩa đặc biệt gì ở cuối thuật toán.

Ví dụ áp dụng 1:

R = (MãsốNV, TênNV, MãsốDA, TênDA, ĐịađiểmDA, Sốgiờ)

R1= (TênNV, ĐịađiểmDA)

R2 = (MãsốNV, MãsốDA, Sốgiờ, TênDA, ĐịađiểmDA)

F= { Mã sốNV→ TênNV, MãsốDA → {TênDA, ĐịađiểmDA}, {MãsốNV, Mã sốDA}→ Sốgiờ}

	MãsốNV	TênNV	Mã sốDA	TênDA	ĐịađiểmDA	Sốgiờ
R1	0	1	0	0	1	0
R2	1	0	1	1	1	1

Xét lần lượt phụ thuộc hàm MãsốNV → TênNV, MãsốDA → {TênDA, ĐịađiểmDA}, {MãsốNV, Mã sốDA} → Sốgiờ. Ta thấy không có trường hợp nào các thuộc tính tương ứng với các vế trái đều có giá trị bằng 1, vì vậy ta không thể làm gì để biến đổi ma trận. Ma trận không chứa một hàng gồm toàn ký hiệu "1". Phép tách là mất mát.

Ví dụ áp dụng 2:

R = (MãsốNV, TênNV, MãsốDA, TênDA, ĐịađiểmDA, Sốgiờ)

R1= (MãsốNV, TênNV)

R2 = (MãsốDA, TênDA, ĐịađiểmDA)

R3 = (MãsốNV, MãsốDA, Sốgiờ)

F= {Mã sốNV→ TênNV, MãsốDA → {TênDA, ĐịađiểmDA}, {MãsốNV, MãsốDA} → Sốgiờ}

	MãsốNV	TênNV	Mã sốDA	TênDA	ĐịađiểmDA	Sốgiờ
R1	1	1	0	0	0	0
R2	0	0	1	1	1	0
R3	1	0	1	1	1	0

(Giá trị ban đầu của ma trận S)

	MãsốNV	TênNV	Mã sốDA	TênDA	ĐịađiểmDA	Sốgiờ
R1	1	1	0	0	0	0
R2	0	0	1	1	1	0
R3	1	~~0~~ 1	1	~~0~~ 1	~~0~~ 1	1

(Ma trận S sau khi áp dụng hai phụ thuộc hàm đầu tiên dòng cuối cùng ko chứa toàn ký hiệu "a"). Ma trận chứa một hàng gồm toàn ký hiệu 1. Phép tách này là không mất mát.

Hình 0-1. Thuật toán kiểm tra nối không mất mát

Thuật toán 5.2 cho phép chúng ta kiểm tra xem một phép tách D cụ thể có tuân theo tính chất nối không mất mát hay không. Câu hỏi tiếp theo là liệu có một thuật toán tách một lược đồ quan hệ vũ trụ R = {A_1, A_2, …, A_n} thành một phép tách D = {R_1, R_2, …,R_m} sao cho mỗi R_i là ở BCNF và phép tách D có tính chất nối không mất mát đối với F hay không? Câu trả lời là có. Trước khi trình bày thuật toán, ta xem một số tính chất của các phép tách nối không mất mát nói chung.

Tính chất 1: Một phép tách D = {R_1,R_2} của R có tính chất nối không mất mát đối với một tập phụ thuộc hàm F trên R khi và chỉ khi

- Hoặc phụ thuộc hàm $((R_1 \cap R_2) \rightarrow (R_1 - R_2))$ ở trong F^+.
- Hoặc phụ thuộc hàm $((R_1 \cap R_2) \rightarrow (R_2 - R_1))$ ở trong F^+.

Với tính chất này, chúng ta có thể kiểm tra lại các phép tách chuẩn hóa trong 4.3 và sẽ thấy rằng các phép tách đó là thỏa mãn tính chất nối không mất mát.

Tính chất 2: Nếu một phép tách D = {R_1, R_2, …, R_m} của R có tính chất nối không mất mát đối với một tập phụ thuộc hàm F trên R và nếu một phép tách D_1 =

$\{Q_1, Q_2, \ldots, Q_k\}$ của R_i có tính chất nối không mất mát đối với phép chiếu của F trên R_i thì phép tách D2 = $\{ R_1, R_2, \ldots, R_{i-1}, Q_1, Q_2, \ldots, Q_k, R_{i+1}, \ldots, R_m\}$ của R có tính chất nối không mất mát đối với F.

Tính chất này nói rằng nếu một phép tách D đã có tính chất nối không mất mát đối với một tập F và chúng ta tiếp tục tách một trong các quan hệ R_i trong D thành phép tách khác D_1 (l = 1,2,..k) có tính chất nối không mất mát đối với $\pi_{Ri}(F)$ thì việc thay R_i trong D bằng D_1 (l = 1,2,..k) cũng tạo ra một phép tách có tính chất nối không mất mát đối với F.

Thuật toán 5.3 sau đây sử dụng hai tính chất trên để tạo ra một phép tách D = $\{R_1, R_2, \ldots, R_m\}$ của một quan hệ vũ trụ R dựa trên một tập các phụ thuộc hàm F sao cho mỗi R_i là BCNF.

Thuật toán 5.3: Tách quan hệ thành các quan hệ BCNF với tính chất nối không mất mát.

Input: Một quan hệ vũ trụ R và một tập hợp các phụ thuộc hàm F trên các thuộc tính của R.

1. Đặt D := {R} ;
2. Khi có một lược đồ quan hệ Q trong D không phải ở BCNF, thực hiện vòng lặp: Với mỗi một lược đồ quan hệ Q trong D không ở BCNF hãy tìm một phụ thuộc hàm $X \rightarrow Y$ trong Q vi phạm BCNF và thay thế Q trong D bằng hai lược đồ quan hệ (Q-Y) và $(X \cup Y)$. Quá trình lặp dừng khi không còn quan hệ nào trong D vi phạm BCNF.

Mỗi lần đi vào vòng lặp trong thuật toán 5.3, chúng ta tách một quan hệ Q không phải BCNF thành hai lược đồ quan hệ. Theo các tính chất 1 và 2, phép tách D có tính chất nối không mất mát. Kết thúc thuật toán, tất cả các quan hệ trong D sẽ ở BCNF.

Trong bước 2 của thuật toán 5.3, cần xác định xem một lược đồ quan hệ Q có ở BCNF hay không. Một phương pháp để làm điều đó là kiểm tra. Với mỗi phụ thuộc hàm $X \rightarrow Y$ trong Q, ta tính X^+. Nếu X^+ không chứa tất cả các thuộc tính trong Q thì $X \rightarrow Y$ vi phạm BCNF bởi vì X không phải là một siêu khóa.

Một kỹ thuật nữa dựa trên quan sát rằng khi một lược đồ quan hệ Q vi phạm BCNF thì có tồn tại một cặp thuộc tính A,B trong Q sao cho $\{Q - \{A,B\}\} \rightarrow A$.

Bằng việc tính bao đóng $\{Q - \{A,B\}\}^+$ cho mỗi cặp thuộc tính {A,B} của Q và kiểm tra xem bao đóng có chứa A (hoặc B) hay không, chúng ta có thể xác định được Q có ở BCNF hay không.

Ví dụ áp dụng: Xét lược đồ quan hệ

R = { A, B, C, D, E, F)

Với các phụ thuộc hàm:

$A \rightarrow BCDEF$, $BC \rightarrow ADEF$, $B \rightarrow F$, $D \rightarrow E$, $D \rightarrow B$

Lược đồ quan hệ này có hai khóa dự tuyển là A và BC.

Ta có $B \rightarrow F$ vi phạm BCNF vì B không phải là siêu khóa, R được tách thành:

R1(B,F) với phụ thuộc hàm $B \rightarrow F$

R2(A,B,C,D,E) với các phụ thuộc hàm $A \rightarrow SCDE$, $BC \rightarrow ADF$, $D \rightarrow E$, $D \rightarrow B$

Do $D \rightarrow E$ vi phạm BCNF (D là một thuộc tính không khóa), R2 được tách thành:

R21(D,E) với phụ thuộc hàm $D \rightarrow E$

R22(ABCD) với các phụ thuộc hàm $A \rightarrow BCD$, $BC \rightarrow AD$, $D \rightarrow B$

Do D B vi phạm BCNF (Dkhông phải là thuộc tính khóa), R22 được tách thành:

R221(D,B)

R222(A,B,D) với phụ thuộc hàm $A \rightarrow BD$ (phụ thuộc hàm $BC \rightarrow AD$ bị mất)

Tóm lại, ta có phép tách D = {R1, R21, R221, R222}. Phép tách này có tính chất nối không mất thông tin nhưng không bảo toàn phụ thuộc.

Nếu chúng ta muốn có một phép tách có tính chất nối không mất mát và bảo toàn phụ thuộc thì ta phải hài lòng với các lược đồ quan hệ ở dạng 3NF. Thuật toán sau đây là cải tiến của thuật toán 5.1, tạo ra một phép tách thỏa mãn :

- Bảo toàn phụ thuộc.
- Có tính chất nối không mất mát.
- Mỗi lược đồ quan hệ kết quả là ở dạng 3NF.

Thuật toán 5.4: Thuật toán tổng hợp quan hệ với tính chất bảo toàn phụ thuộc và nối không mất mát.

Input: Một quan hệ vũ trụ R và một tập các phụ thuộc hàm F trên các thuộc tính của R.

1) Tìm phủ tối thiểu G cho F.
2) Với mỗi vế trái X của một phụ thuộc hàm xuất hiện trong G hãy tạo ra một lược đồ quan hệ trong D với các thuộc tính $\{X \cup \{A_1\} \cup \{A_2\} \cup \ldots \cup \{A_k\}\}$, trong đó $X \rightarrow A_1$, $X \rightarrow A_2, \ldots$, $X \rightarrow A_k$ chỉ là các phụ thuộc hàm ở trong G với X là vế trái (X là khóa của quan hệ này).
3) Nếu không có lược đồ quan hệ nào trong D chứa một khóa của R thì hãy tạo ra thêm một lược đồ quan hệ trong D chứa các thuộc tính tạo nên một khóa của R.

Bước 3 của thuật toán 5.4 đòi hỏi phải xác định một khóa K của R. Để xác định một khóa K của R, ta sử dụng thuật toán sau

Thuật toán xác định khóa: Tìm một khóa K của R dựa trên tập F các phụ thuộc hàm.

1) Đặt K := R;
2) Với mỗi thuộc tính A trong K

   {tính $(K-A)^+$ đối với F;

   Nếu $(K-A)^+$ chứa tất cả các thuộc tính trong R thì đặt K := K-{A}};

*Chú ý: Chúng ta có nhận xét sau: Nếu quan hệ có khóa thì các thuộc tính khóa của quan hệ phải là các tập con của tập hợp các thuộc tính ở vế phải các phụ thuộc hàm trong F. Vì vậy, để tìm được các khóa nhanh hơn, trước tiên chúng ta tính R_F là hợp của các thuộc tính ở các vế trái của các phụ thuộc hàm trong F, sau đó đi tính bao đóng của tất cả các tập con của R_F. Nếu bao đóng của tập con nào chứa tất cả các thuộc tính của R thì tập đó là một siêu khóa. Để kiểm tra nó là một khóa ta thực hiện như bước 2) của thuật toán trên.

Không phải lúc nào cũng có khả năng tìm được một phép tách thành các lược đồ quan hệ bảo toàn phụ thuộc và mỗi lược đồ trong phép tách là ở BCNF. Các lược đồ quan hệ trong phép tách theo thuật toán ở trên thường là 3NF. Để có các lược đồ BCNF, chúng ta có thể kiểm tra các lược đồ quan hệ 3NF trong phép tách

một cách riêng rẽ để xem nó có thỏa mãn BCNF không. Nếu có lược đồ quan hệ R_i không ở BCNF thì ta có thể tách tiếp hoặc để nguyên nó là 3NF.

4.3- Các phụ thuộc hàm đa trị và dạng chuẩn 4

Trong phần này chúng ta thảo luận khái niệm phụ thuộc hàm đa trị và định nghĩa dạng chuẩn 4. Các phụ thuộc đa trị hệ quả của dạng chuẩn 1 không cho phép một thuộc tính của một bộ có một tập giá trị (nghĩa là các thuộc tính đa trị). Nếu chúng ta có hai hoặc nhiều hơn các thuộc tính độc lập và đa trị trong cùng một lược đồ quan hệ thì chúng ta phải lặp lại mỗi một giá trị của một trong các thuộc tính với mỗi giá trị của thuộc tính khác để giữ cho trạng thái quan hệ nhất quán và duy trì tính độc lập giữa các thuộc tính. Ràng buộc đó được chỉ ra bằng một phụ thuộc đa trị.

4.3.1- <u>Định nghĩa phụ thuộc đa trị</u>

Giả thiết có một lược đồ quan hệ R, X và Y là hai tập con của R. Một phụ thuộc đa trị (MVD), ký hiệu là $X \rightarrow\rightarrow Y$, chỉ ra ràng buộc sau đây trên một trạng thái quan hệ bất kỳ của R: Nếu hai bộ t_1 và t_2 tồn tại trong R sao cho $t_1[X] = t_2[X]$ thì hai bộ t_3 và t_4 cũng tồn tại trong R với các tính chất sau:

. $t_3[X] = t_4[X] = t_1[X] = t_2[X]$

. $t_3[Y] = t_1[Y]$ và $t_4[Y] = t_2[Y]$

. $t_3[Z] = t_2[Z]$ và $t_4[Z] = t_1[Z]$ với $Z = (R- (X \cup Y))$

Khi $X\rightarrow\rightarrow Y$ thỏa mãn, ta nói rằng X đa xác định Y. Bởi vì tính đối xứng trong định nghĩa, khi $X \rightarrow\rightarrow Y$ thỏa mãn trong R, $X\rightarrow\rightarrow Z$ cũng thỏa mãn trong R. Như vậy $X\rightarrow\rightarrow Y$ kéo theo $X\rightarrow\rightarrow Z$ và vì thế đôi khi nó được viết là $X\rightarrow\rightarrow Y|Z$

Định nghĩa hình thức chỉ ra rằng, cho trước một giá trị cụ thể của X, tập hợp các giá trị của Y được xác định bởi giá trị này của X là được xác định hoàn toàn bởi một mình X và không phụ thuộc vào các giá trị của các thuộc tính còn lại Z của R. Như vậy, mỗi khi hai bộ tồn tại có các giá trị khác nhau của Y nhưng cùng một giá trị X thì các giá trị này của Y phải được lặp lại trong các bộ riêng rẽ với mỗi giá trị khác nhau của Z có mặt với cùng giá trị của X. Điều đó tương ứng một cách không hình thức với Y là một thuộc tính đa trị của các thực thể được biểu diễn bằng các bộ trong R.

Ví dụ về phụ thuộc đa trị:

NHÂNVIÊN	TênNV	TênDA	TênconNV
	Nam	DA01	Lan
	Nam	DA02	Hoa
	Nam	DA01	Hoa
	Nam	DA02	Lan

Trong bảng trên có hai phụ thuộc đa trị là:

TênNV$\rightarrow\rightarrow$TênDA, TênNV$\rightarrow\rightarrow$TênconNV

Một MVD X$\rightarrow\rightarrow$Y được gọi phụ thuộc đa trị tầm thường nếu:

a) Y là một tập con của X

b) hoặc X $\cup$ Y = R

Một MVD không thỏa mãn a) hoặc b) được gọi là một MVD không tầm thường. Nếu chúng ta có một phụ thuộc đa trị không tầm thường trong một quan hệ, chúng ta có thể phải lặp các giá trị một cách dư thừa trong các bộ. Trong quan hệ NHÂNVIÊN ở ví dụ trên, các giá trị 'DA01', 'DA02' của TênDA được lặp lại với mỗi giá trị của TênconNV (một cách đối xứng, các giá trị 'Lan', 'Hoa' được lặp lại với mỗi giá trị của TênDA). Rõ ràng ta không mong muốn có sự dư thừa đó. Tuy nhiên, lược đồ quan hệ trên là ở BCNF bởi vì không có phụ thuộc hàm nào thỏa mãn trong quan hệ đó. Vì vậy, chúng ta phải định nghĩa một dạng chuẩn thứ tư mạnh hơn BCNF và ngăn cấm các lược đồ quan hệ như quan hệ NHÂNVIÊN.

4.3.2- Các quy tắc suy diễn đối với các phụ thuộc hàm và phụ thuộc đa trị

Các quy tắc từ Qt1 đến Qt8 sau đây tạo nên một tập hợp đúng đắn và đầy đủ cho việc suy diễn các phụ thuộc hàm và phụ thuộc đa trị từ một tập các phụ thuộc cho trước. Giả thiết rằng tất cả các thuộc tính được chứa trong một lược đồ quan hệ "vũ trụ" R = $\{A_1, A_2, \ldots, A_n\}$ và X, Y, Z, W là các tập con của R.

Qt1) (quy tắc phản xạ cho FD): Nếu X $\supseteq$ Y thì X $\rightarrow$ Y

Qt2) (quy tắc tăng cho FD): {X $\rightarrow$Y} |= XZ $\rightarrow$ YZ

Qt3) (quy tắc bắc cầu cho FD): { $X \rightarrow Y, Y \rightarrow Z$ } |= $X \rightarrow Z$

Qt4) (quy tắc bù cho MVD): {$X \rightarrow\rightarrow Y$ } |= {$X \rightarrow\rightarrow (R-(X \cup Y))$}

Qt5) (quy tắc tăng cho MVD): Nếu $X \rightarrow\rightarrow Y$ và $W \supseteq Z$ thì $WX \rightarrow\rightarrow YZ$

Qt6) (quy tắc bắc cầu cho MVD): {$X \rightarrow\rightarrow Y, Y \rightarrow\rightarrow Z$ } |= $X \rightarrow\rightarrow (Z - Y)$

Qt7) (quy tắc tái tạo cho FD và MVD): {$X \rightarrow Y$} |= $X \rightarrow\rightarrow Y$

Qt8) (quy tắc liên hợp cho FD và MVD): Nếu $X \rightarrow\rightarrow Y$ và có tồn tại W với các tính chất a) $W \cap Y = \emptyset$, b) $W \rightarrow Z$ và c) $Y \supseteq Z$ thì $X \rightarrow Z$.

Qt1 đến Qt3 là các quy tắc suy diễn Amstrong đối với các phụ thuộc hàm. Qt4 đến Qt6 là các quy tắc suy diễn chỉ liên quan đến các phụ thuộc đa trị. Qt7 và Qt8 liên kết các phụ thuộc hàm và các phụ thuộc đa trị. Đặc biệt, Qt7 nói rằng một phụ thuộc hàm là một trường hợp đặc biệt của một phụ thuộc đa trị. Điều đó có nghĩa là mỗi phụ thuộc hàm cũng là một phụ thuộc đa trị bởi vì nó thỏa mãn định nghĩa hình thức của phụ thuộc đa trị. Về cơ bản, một phụ thuộc hàm $X \rightarrow Y$ là một phụ thuộc đa trị $X \rightarrow\rightarrow Y$ với một hạn chế phụ rằng có nhiều nhất là một giá trị của Y được kết hợp với mỗi giá trị của X. Cho trước một tập hợp các phụ thuộc hàm và phụ thuộc đa trị chỉ ra trên R = {A1, A2, …, An}, chúng ta có thể sử dụng các quy tắc từ Qt1 đến Qt8 để suy ra tập hợp đầy đủ các phụ thuộc (hàm và đa trị) F^+ đúng trong mọi trạng thái quan hệ r của R thỏa mãn F. Chúng ta lại gọi F^+ là bao đóng của F.

4.3.3- Dạng chuẩn 4

Định nghĩa: Một lược đồ quan hệ R là ở dạng chuẩn 4 (4NF) đối với một tập hợp các phụ thuộc F (gồm các phụ thuộc hàm và phụ thuộc đa trị) nếu với mỗi phụ thuộc đa trị không tầm thường $X \rightarrow\rightarrow Y$ trong F^+ , X là một siêu khóa của R.

Như vậy, một lược đồ quan hệ vi phạm 4NF nếu nó chứa các phụ thuộc hàm đa trị không mong muốn. Ví dụ, lược đồ quan hệ NHÂNVIÊN ở ví dụ trên là vi phạm 4NF bởi vì trong các phụ thuộc hàm đa trị TênNV$\rightarrow\rightarrow$TênDA và TênNV$\rightarrow\rightarrow$ Têncon, TênNV không phải là một siêu khóa .

Giả sử chúng ta tách bảng NHÂNVIÊN thành hai bảng như sau:

NV_DA	TênNV	TênDA
	Nam	DA01
	Nam	DA02

NV_CON	TênNV	TênconNV
	Nam	Lan
	Nam	Hoa

Hai bảng này là ở 4NF bởi vì các phụ thuộc đa trị TênNV$\rightarrow\rightarrow$TênDA và TênNV$\rightarrow\rightarrow$TênconNV là các phụ thuộc đa trị tầm thường. Trong hai bảng này không có các phụ thuộc đa trị không tầm thường cũng như không có các phụ thuộc hàm.

4.3.4- Tách có tính chất nối không mất mát thành các quan hệ 4NF

Khi chúng ta tách một lược đồ quan hệ R thành $R_1 = (X \cup Y)$ và $R_2 = (R-Y)$ dựa trên phụ thuộc hàm đa trị $X \rightarrow\rightarrow Y$ đúng trong R, phép tách có tính chất nối không mất mát. Đó cũng là điều kiện cần và đủ cho một phép tách một lược đồ thành hai lược đồ có tính chất nối không mất mát. Ta có tính chất sau:

Tính chất 1': Các lược đồ quan hệ R_1 và R_2 tạo thành một phép tách có tính chất nối không mất mát của R khi và chỉ khi $(R_1 \cap R_2) \rightarrow\rightarrow (R_1 - R_2)$ (hoặc $(R_1 \cap R_2) \rightarrow\rightarrow (R_1 - R_2)$).

Áp dụng tính chất trên chúng ta có thuật toán tạo một phép tách có tính chất nối không mất mát thành các lược đồ quan hệ ở dạng 4NF.

Thuật toán 5.5: Tách quan hệ thành các quan hệ 4NF với tính chất nối không mất mát.

Input: Một quan hệ vũ trụ R và một tập phụ thuộc hàm và phụ thuộc đa trị F.

1. Đặt D := {R};
2. Khi có một lược đồ quan hệ Q trong D không ở 4NF, thực hiện:

   {Chọn một lược đồ quan hệ Q trong D không ở 4NF;

   Tìm một phụ thuộc đa trị không tầm thường $X \rightarrow\rightarrow Y$ trong Q vi phạm 4NF;

   Thay thế Q trong D bằng hai lược đồ quan hệ $(Q - Y)$ và $(X \cup Y)$};

Ví dụ áp dụng:

Xét lược đồ NHÂNVIÊN(TênNV, TênDA, TênconNV). Ta có phụ thuộc hàm đa trị TênNV$\rightarrow\rightarrow$TênDA trong đó TênNV không phải là một siêu khóa, vậy nó vi

phạm 4NF. Ta tách thành NV_DA(TênNV, TênDA), NV_CON(TênNV, TênconNV).

4.4- Các phụ thuộc nối và dạng chuẩn 5

Như chúng ta đã thấy, các tính chất 1 và tính chất 1' cho điều kiện để một lược đồ quan hệ R được tách thành hai lược đồ quan hệ R1 và R2 và phép tách có tính chất nối không mất mát. Tuy nhiên, trong một số trường hợp, có thể không có phép tách có tính chất nối không mất mát của R thành hai lược đồ quan hệ nhưng có thể có phép tách có tính chất nối không mất mát thành nhiều hơn hai quan hệ. Hơn nữa, có thể không có phụ thuộc hàm nào trong R các chuẩn cho đến BCNF và có thể không có phụ thuộc đa trị nào có trong R vi phạm 4NF. Khi đó chúng ta phải sử dụng đến một phụ thuộc khác gọi là phụ thuộc nối và nếu có phụ thuộc nối thì thực hiện một phép tách đa chiều thành dạng chuẩn 5 (5NF).

Một *phụ thuộc nối* (JD), ký hiệu là JD(R_1, R_2, …, R_n) trên lược đồ quan hệ R chỉ ra một ràng buộc trên các trạng thái r của R. Ràng buộc đó tuyên bố rằng mỗi trạng thái hợp pháp r của R phải có phép tách có tính chất nối không mất mát thành R1, R2, …, Rn. Điều đó nghĩa là:

$$*(\pi_{R1}(r), \pi_{R2}(r), \ldots, \pi_{Rn}(r)) = r$$

Một phụ thuộc nối JD(R_1, R_2, …, R_n) là một phụ thuộc nối tầm thường nếu một trong các lược đồ quan hệ R_i ở trong JD(R_1, R_2, …, R_n) là bằng R.

Một lược đồ quan hệ R là ở dạng chuẩn 5 (5NF) (hoặc dạng chuẩn nối chiếu PJNF – Project-Join normal form) đối với một tấp F các phụ thuộc hàm, phụ thuộc đa trị và phụ thuộc nối nếu với mỗi phụ thuộc nối không tầm thường JD(R_1, R_2, …, R_n) trong F^+, mỗi R_i là một siêu khóa của R.

Ví dụ: Xét quan hệ CUNGCẤP gồm toàn các thuộc tính khóa

CUNGCẤP	Tênnhàcungcấp	Tênhàng	TênDựán
	Ncc1	Bulong	Dựán1
	Ncc1	Đaiốc	Dựán2
	Ncc2	Bulong	Dựán2
	Ncc3	Đaiốc	Dựán3

Ncc2	Đinh	Dựán1
Ncc2	Bulong	Dựán1
Ncc1	Bulong	Dựán2

Giả thiết rằng ràng buộc phụ thêm sau đây luôn đúng: Khi một nhà cung cấp S cung cấp hàng P VÀ một dự án J sử dụng hàng P VÀ nhà cung cấp S cung cấp ít nhất là một hàng cho dự án J THÌ nhà cung cấp S cũng sẽ cung cấp hàng P cho dự án J. Ràng buộc này chỉ ra một phụ thuộc nối JD(R_1,R_2,R_3) giữa ba phép chiếu R_1(Tênnhàcungcấp,Tênhàng), R_2(Tênnhàcungcấp,Têndựán),R_3(Tênhàng,TênDựán) của quan hệ CUNGCẤP. Quan hệ CUNGCẤP được tách thành ba quan hệ R_1, R_2, R_3 ở dạng chuẩn 5. Chú ý rằng nếu ta áp dụng phép nối tự nhiên cho từng đôi quan hệ một thì sẽ sinh ra các bộ giả, nhưng nếu áp dụng phép nối tự nhiên cho cả ba quan hệ thì không sinh ra các bộ giả.

R1

Tênnhàcungcấp	Tênhàng
Ncc1	Bulong
Ncc1	Đaiốc
Ncc2	Bulong
Ncc3	Đaiốc
Ncc2	Đinh

R2

Tênnhàcungcấp	Têndựán
Ncc1	Dựán1
Ncc1	Dựán2
Ncc2	Dựán2
Ncc3	Dựán3
Ncc2	Dựán1

R3

Tênhàng	Têndựán
Bulong	Dựán1
Đaiốc	Dựán2
Bulong	Dựán2
Đaiốc	Dựán3
Đinh	Dựán1

Việc phát hiện các phụ thuộc nối trong các cơ sở dữ liệu thực tế với hàng trăm thuộc tính là một điều rất khó khăn. Vì vậy, thực tiễn thiết kế cơ sở dữ liệu hiện nay thường không chú ý đến nó.

Nói chung, trong thực tế thiết kế cơ sở dữ liệu, người ta chỉ chuẩn hóa các bảng đến 3NF, BCNF là đủ

5. Tổng kết chương và câu hỏi ôn tập

5.1- Tổng kết chương

Trong chương này chúng ta đã nói đến các nguy hiểm có thể xảy ra trong việc thiết kế cơ sở dữ liệu, xác định một cách không hình thức một số chuẩn mực để chỉ ra một lược đồ quan hệ là "tốt" hay "tồi" và đưa ra một số nguyên tắc không hình

thức cho một thiết kế tốt. Sau đó chúng ta đã trình bày một vài khái niệm cho phép ta thiết kế quan hệ theo cách *trên-xuống* bằng cách phân tích các quan hệ một cách riêng rẽ. Chúng ta đã định nghĩa quá trình thiết kế này bằng phân tích và tách bằng cách giới thiệu quá trình chuẩn hóa.

Những vấn đề về các bất thường cập nhật xảy ra khi có sự dư thừa xảy ra trong các quan hệ cũng đã được đề cập đến. Các chuẩn mực không hình thức của các lược đồ quan hệ tốt bao gồm ngữ nghĩa của thuộc tính rõ ràng và đơn giản, ít giá trị null trong các mở rộng của quan hệ. Một phép tách tốt phải tránh được việc sinh ra các bộ giả khi thực hiện phép nối.

Chúng ta đã định nghĩa khái niệm phụ thuộc hàm và thảo luận một số tính chất của nó. Các phụ thuộc hàm là các nguồn thông tin ngữ nghĩa cơ bản về các thuộc tính của lược đồ quan hệ. Chúng ta đã chỉ ra cách suy diễn các phụ thuộc phụ thêm dựa trên một tập các phụ thuộc hàm cho trước và một tập các quy tắc suy diễn. Chúng ta đã định nghĩa các khái niệm bao đóng và phủ tối thiểu của một tập phụ thuộc hàm và cung cấp thuật toán tính phủ tối thiểu. Ta cũng đã chỉ ra làm thế nào để kiểm tra xem hai tập phụ thuộc hàm có tương đương nhau hay không.

Tiếp theo, chúng ta đã mô tả quá trình chuẩn hóa để đạt đến các thiết kế tốt bằng cách kiểm tra các quan hệ đối với các kiểu phụ thuộc hàm không mong muốn. Chúng ta đã cung cấp cách chuẩn hóa liên tiếp dựa trên khóa chính được định nghĩa trước trong mỗi quan hệ và sau đó giảm nhẹ đòi hỏi này và đưa ra các định nghĩa tổng quát của các dạng chuẩn có tính đến tất cả các khóa dự tuyển của một quan hệ.

Trong phần IV chúng ta đã trình bày nhiều thuật toán chuẩn hóa. Đó là thuật toán tổng hợp quan hệ tạo ra các quan hệ 3NF từ một lược đồ quan hệ vũ trụ dựa trên một tập các phụ thuộc hàm do người thiết kế cơ sở dữ liệu xác định. Các thuật toán tạo ra các quan hệ BCNF (hoặc 4NF) bằng cách tách không mất mát liên tiếp các quan hệ không chuẩn hóa thành hai quan hệ thành phần tại một thời điểm. Chúng ta đã thảo luận về hai tính chất quan trọng của phép tách: tính chất nối không mất mát (hoặc không phụ thêm) và tính chất bảo toàn phụ thuộc. Một thuật toán kiểm tra phép tách không mất mát và một thuật toán kiểm tra tính không mất mát của một phép tách thành hai quan hệ cúng đã được trình bày. Chúng ta cũng đã thấy rằng việc tổng hợp các quan hệ ở dạng 3NF đảm bảo cả hai tính chất trên là có khả năng còn việc tổng hợp các quan hệ BCNF chỉ có khả năng đảm bảo tính không mất mát, không thể đảm bảo tính bảo toàn phụ thuộc.

Cuối cùng, chúng ta đã nghiên cứu các loại phụ thuộc khác: phụ thuộc đa trị và phụ thuộc nối, đưa ra định nghĩa các dạng chuẩn 4, dạng chuẩn 5 và thuật toán tách các quan hệ vi phạm thành quan hệ 4NF, 5NF. Việc phát hiện các phụ thuộc nối rất khó khăn nên trong thiết kế thực tiễn người ta thường bỏ qua nó.

5.2- Câu hỏi ôn tập

1) Hãy giải thích ngữ nghĩa của thuộc tính như là một độ đo không hình thức về tính tốt đối với một lược đồ quan hệ.
2) Hãy thảo luận về các bất thường chèn, xóa và sửa đổi. Vì sao chúng được xem là không tốt? Hãy minh họa bằng ví dụ.
3) Hãy trình bày vấn đề các bộ giả và làm thế nào để ngăn ngừa chúng?
4) Trình bày các nguyên tắc đối với việc thiết kế lược đồ quan hệ. Hãy minh họa việc vi phạm các nguyên tắc đó sẽ có hại như thế nào?
5) Phụ thuộc hàm là gì? Ai là người chỉ ra các phụ thuộc hàm giữa các thuộc tính của một lược đồ quan hệ?
6) Vì sao chúng ta không thể suy ra một phụ thuộc hàm từ một trạng thái quan hệ cụ thể?
7) Vì sao các quy tắc suy diễn của Amstrong (Qt1 đến Qt3) là quan trọng?
8) Tính đầy đủ và tính đúng đắn của các quy tắc suy diễn Amstrong là gì?
9) Bao đóng của một tập phụ thuộc hàm là gì?
10) Khi nào thì hai tập phụ thuộc hàm là tương đương? Làm thế nào để kiểm tra tính tương đương của chúng?
11) Tập tối thiểu các phụ thuộc hàm là gì? Có phải mỗi tập tối thiểu phụ thuộc hàm có một tập tối thiểu tương đương hay không?
12) Thuật ngữ quan hệ không chuẩn hóa ám chỉ cái gì?
13) Định nghĩa các dạng chuẩn 1NF, 2NF, 3NF, BCNF dựa trên khóa chính và các dạng chuẩn dưới dạng tổng quát. Sự khác nhau của hai định nghĩa là gì?
14) Phụ thuộc hàm nào cần tránh khi một quan hệ là ở 3NF?
15) Định nghĩa dạng chuẩn Boyce-Codd. Nó khác gì với 3NF? Vì sao nó được xem là mạnh hơn 3NF?

16) Điều kiện bảo toàn thuộc tính trên một phép tách là gì?

17) Vì sao các dạng chuẩn tự nó là chưa đủ như là một điều kiện cho một thiết kế lược đồ tốt?

18) Tính chất bảo toàn phụ thuộc đối với một phép tách là gì? Vì sao nó là quan trọng?

19) Vì sao chúng ta không thể đảm bảo rằng một phép tách các lược đồ quan hệ không BCNF thành BCNF là bảo toàn phụ thuộc? Hãy cho một phản ví dụ.

20) Tính chất nối không mất mát (không phụ thêm) của một phép tách là gì? Vì sao nó là quan trọng?

21) Giữa các tính chất bảo toàn phụ thuộc và nối không mất mát cái nào là nhất thiết phải thỏa mãn? Vì sao? Phụ thuộc hàm đa trị là gì? Nó chỉ ra ràng buộc gì? Khi nào nó sinh ra?

22) Hãy minh họa quá trình tạo ra các quan hệ ở dạng chuẩn 1? Làm thế nào để có dạng chuẩn 1 một cách đúng đắn, tránh được phụ thuộc đa trị?

23) Định nghĩa dạng chuẩn 4. Nó có lợi gì?

24) Định nghĩa phụ thuộc nối và dạng chuẩn 5.

5.3- Bài tập

1) Hãy kiểm tra các quy tắc suy diễn đối với các phụ thuộc hàm sau đây là đúng hay sai:

   a) $\{W \rightarrow Y, X \rightarrow Z\} \models \{WX \rightarrow Y\}$

   b) $\{X \rightarrow Y\}$ và $Y \supseteq Z \models \{X \rightarrow Z\}$

   c) $\{X \rightarrow Y , X \rightarrow W, WY \rightarrow Z\} \models \{X \rightarrow Z\}$

   d) $\{XY \rightarrow Z, Y \rightarrow W\} \models \{XW \rightarrow Z\}$

   e) $\{X \rightarrow Z, Y \rightarrow Z\} \models \{X \rightarrow Y\}$

   f) $\{X \rightarrow Y, Z \rightarrow W\} \models \{XZ \rightarrow YW\}$

   g) $\{XY \rightarrow Z , Z \rightarrow X\} \models \{Z \rightarrow Y\}$

   h) $\{X \rightarrow Y, Y \rightarrow Z\} \models \{X \rightarrow YZ\}$

   i) $\{XY \rightarrow Z, Z \rightarrow W\} \models \{X \rightarrow W\}$

2) Cho lược đồ quan hệ R(A,B,C,D,E,F,G,H,I,J) và tập phụ thuộc hàm sau đây:

F1 = {AB $\rightarrow$ C, A $\rightarrow$ DE, B $\rightarrow$ F, F $\rightarrow$ GH, D$\rightarrow$ IJ}

a) Khóa của quan hệ là gì? Hãy tách quan hệ thành 2NF, sau đó thành 3NF.

b) Làm lại câu a) với tập phụ thuộc hàm sau:

G1= { AB $\rightarrow$ C, BD $\rightarrow$ EF, AD$\rightarrow$ GH, A $\rightarrow$ I , H $\rightarrow$ J }

3) Xét quan hệ R(A,B,C,D,E) và các phụ thuộc hàm sau:

AB $\rightarrow$C, CD $\rightarrow$E, DE $\rightarrow$ B.

AB có phải là khóa dự tuyển của quan hệ không? Vì sao? Hãy tìm một khóa của nó.

4) Cho quan hệ sau:

A	B	C	BộID
10	b1	c1	#1
10	b2	c2	#2
11	b4	c1	#3
12	b3	c4	#4
13	b1	c1	#5
14	b3	c4	#6

Những phụ thuộc hàm nào sau đây là đúng: A $\rightarrow$ B , B $\rightarrow$ C, C $\rightarrow$ B, B $\rightarrow$ A, C $\rightarrow$ A. Nếu có những phụ thuộc hàm sai, hãy giải thích vì sao.